மனோன்மணியம் சுந்தரனார் பல்கலைக்கழகம்

MANONMANIAM SUNDARANAR UNIVERSITY
TIRUNELVELI-627 012

தொலைநிலை தொடர் கல்வி இயக்ககம்

DIRECTORATE OF DISTANCE AND CONTINUING EDUCATION

M.Sc. MATHEMATICS

I YEAR

OPERATIONS RESEARCH

Sub. Code: SMAE22

Prepared by

Dr. S. Kalaiselvi

Assistant Professor

Department of Mathematics

Sarah Tucker College(Autonomous), Tirunelvei-7.

M.Sc. MATHEMATICS –I YEAR

SMAE22: OPERATIONS RESEARCH

SYLLABUS

Unit I

Transportation Models and its Variants: Definition of the Transportation Model-Non- Traditional Transportation Model-Transportation Algorithm- The Assignment Model.

Chapter 1: Sections 1.1-1.4

Unit II

Network Analysis: Network Definitions-Minimal Spanning Tree Algorithm- Shortest Route Problem-Maximum Flow Model-CM-PERT.

Chapter 2: Sections 2.1-2.5

Unit-III

Integer Linear Programming: Introduction- Applications-Integer Programming Solutions-Algorithms.

Chapter 3: Sec 3.1-3.3

Unit IV

Inventory Theory: Basic Elements of an Inventory Model-Deterministic Models: Single item Stock Model with and without Price Breaks-Multiple Items Stock Model with Storage Limitations- Probabilistic Models: Continuous Review Model-Single Period Models.

Chapter 4 - Sections 4.1-4.9

Unit V

Queuing Theory: Basic elements of Queuing Model-Role of Poisson and Exponential Distributions- Pure Birth and Death Models-Specialised Poisson Queues-(M/G/1):GD/∞/∞)-Pollaczek-Khintechine Formula.

Chapter 5: Sections 5.1-5.5

Text Book: Operations Research (Sixth Edition), Hamdy A. Taha, Prentice Hall of India Private Limited, New Delhi

SMAE22: OPERATIONS RESEARCH

CONTENTS

Unit I

Transportation Models and its Variants: Definition of the Transportation Model-Non- Traditional Transportation Model-Transportation Algorithm- The Assignment Model.

Chapter 1: Sections 1.1-1.4

1.1.Definition of the Transportation Model:

The general problem is represented by the network in Figure 1.1. There are *m* sources and *n* destinations, each represented by a **node**. The **arcs** represent the routes linking the sources and the destinations. Arc *(i,* j) joining source *i* to destination *j* carries two pieces of information: the transportation cost per unit, C_{ij}, and the amount shipped, Xij' The amount of supply at source *i* is *ai* and the amount of demand at destination j is b_j. The objective of the model is to determine the unknowns Xij that will minimize the total transportation cost while satisfying all the supply and demand restrictions.

Example 1:

MG Auto has three plants in Los Angeles, Detroit, and New Orleans, and two major distribution centers in Denver and Miami. The capacities of the three plants during the next quarter are 1000, 1500, and 1200 cars. The quarterly demands at the two distribution centers are 2300 and 1400 cars. The mileage chart between the plants and the distribution centers is given in Table 1.l. The trucking company in charge of transporting the cars charges 8 cents per mile per car. The transportation costs per car on the different routes, rounded to the closest dollar, are given in Table 1.2.

The LP model of the problem is given as

Minimize z= $80x_{11} + 215x_{12} + 100x_{21} + 108x_{22} + 102x_{31} + 68x_{32}$

	Denver	Miami
Los Angeles	1000	2690
Detroit	1250	1350
New Orleans	1275	850

Table 1.1 Mileage Chart

	Denver(1)	Miami(2)
Los Angeles(1)	80	215
Detroit (2)	100	108
New Orleans (3)	102	68

Table 1.2 Transportation Cost per Car

Subject to

$x_{11} + x_{12} = 1000$ (Los Angeles)

$x_{21} + x_{22} = 1500$ (Detroit)

$x_{31} + x_{32} = 1200$ **(**New Orleans)

$x_{11} + x_{21} + x_{32} = 2300$ (Denver)

$x_{12} + x_{22} + x_{32} = 1400$ (Miami)

$x_{ij} \geq 0, i = 1,2,3, j = 1,2$

These constraints are all equations because the total supply from the three sources (= 1000 + 1500 + 1200 = 3700 cars) equals the total demand at the two destinations (= 2300 + 1400 = 3700 cars).

The LP model can be solved by the simplex method. However, with the special structure of the constraints we can solve the problem more conveniently using the **transportation tableau** shown in Table 1.3.

	Denver(1)	Miami(2)	Supply
Los Angeles(1)	80 x_{11}	215 x_{12}	1000
Detroit (2)	100 x_{21}	108 x_{22}	1500
New Orleans (3)	102 x_{31}	68 x_{32}	1200
Demand	2300	1400	

Table 1.3
MG Transportation Model

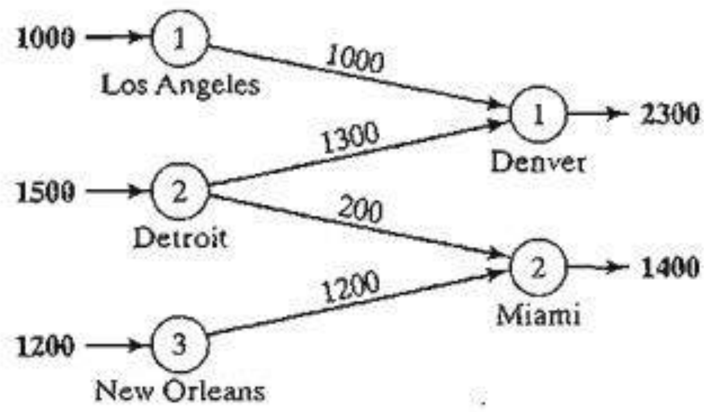


Figure 1.1

Optimal solution of MG Auto model

The optimal solution in Figure 1.1 (obtained by TORA) calls for shipping 1000 cars from Los Angeles to Denver, 1300 from Detroit to Denver, 200 from Detroit to Miami, and 1200 from New Orleans to Miami. The associated minimum transportation cost is computed as 1000 x $80 + 1300 x $100 + 200 x $108 + 1200 x $68 = $313,200.

Balancing the Transportation Model. The transportation algorithm is based on the assumption that the model is balanced, meaning that the total demand equals the total supply. If the model is unbalanced, we can always add a dummy source or a dummy destination to restore balance.

Example 2:

In the MG model, suppose that the Detroit plant capacity is 1300 cars (instead of 1500). The total supply (= 3500 cars) is less than the total demand (= 3700 cars), meaning that part of the demand at Denver and Miami will not be satisfied.

Because the demand exceeds the supply, a dummy source (plant) with a capacity of 200 cars (= 3700 - 3500) is added to balance the transportation model. The unit transportation costs from the dummy plant to the two destinations are zero because the plant does not exist.

Table 1.4 gives the balanced model together with its optimum solution. The solution shows that the dummy plant ships 200 cars to Miami, which means that Miami will be 200 cars short of satisfying its demand of 1400 cars. We can make sure that a specific destination does not experience shortage by assigning a very high unit transportation cost from the dummy source to that destination. For example, a penalty of $1000 in the Dummy-Miami cell will prevent shortage at Miami. Of course, we cannot use this "trick" with all the destinations, because shortage must occur somewhere in the system.

The case where the supply exceeds the demand can be demonstrated by assuming that the demand at Denver is 1900 cars only. In this case, we need to add a dummy distribution center to "receive" the surplus supply. Again, the unit transportation costs to the dummy distribution center are zero, unless we require a factory to "ship out" completely. In this case, we must assign a high unit transportation cost from the designated factory to the dummy destination.

	Denver	Miami	Supply
Los Angeles	80 1000	215	1000
Detroit	100 1300	108	1300
New Orleans	102	68 1200	1200
Dummy Plant	0	0 200	200
Demand	2300	1400	

Table 1.4 MG Model with dummy Plant

	Denver	Miami	Dummy	Supply
Los Angeles	80 1000	215	0	1000
Detroit	100 1300	108	0 400	1500
New Orleans	102	68 1200	0	1200
Demand	2300	1400	400	

Table 1.5 MG Model with dummy Destination

Table 1.5 gives the new model and its optimal solution. The solution shows that the Detroit plant will have a surplus of 400 cars.

1.2. Non-traditional Transportation Models:

The application of the transportation model is not limited to *transporting* commodities between geographical sources and destinations. This section presents two applications in the areas of production-inventory control and tool sharpening service.

Example 3:(Production-Inventory Control)

Boralis manufactures backpacks for serious hikers. The demand for its product occurs during March to June of each year. Boralis estimates the demand for the four months to be 100, 200, 180, and 300 units, respectively. The company uses part-time labor to manufacture the backpacks and, accordingly, its production capacity varies monthly. It is estimated that Boralis can produce 50, 180,280, and 270 units in March through June. Because the production capacity and demand for the different months do not match, a current month's demand may be satisfied in one of three ways.

a) Current month's production.

b) Surplus production in an earlier month.

c) Surplus production in a later month (backordering).

In the first case, the production cost per backpack is $40. The second case incurs an additional holding cost of $.50 per backpack per month. In the third case, an additional penalty cost of $2.00 per backpack is incurred for each month delay. Boralis wishes to determine the optimal production schedule for the four months.

The situation can be modeled as a transportation model by recognizing the following parallels between the elements of the production-inventory problem and the transportation model:

Transportation	Production-inventory
1.Source i	1.Production period i
2.Destination	2.Demand period j
3.Supply amount at source i	3.Production capacity of period i
4. Demand at destination j	4. Demand for period j
5.Unit transportation cost from source i to destination j	5.Unit cost (production+ inventory +penalty) in period i for period j

	1	2	3	4	Capacity
1	$40.00	$40.50	$41.00	$41.50	**50**
2	$42.00	$40.00	$40.50	$41.00	**180**
3	$44.00	$42.00	$40.00	$40.50	**280**
4	$46.00	$44.00	$42.00	$40.00	**270**
Demand	**100**	**200**	**180**	**300**	

Table 1.6 Transportation Model for Example 3

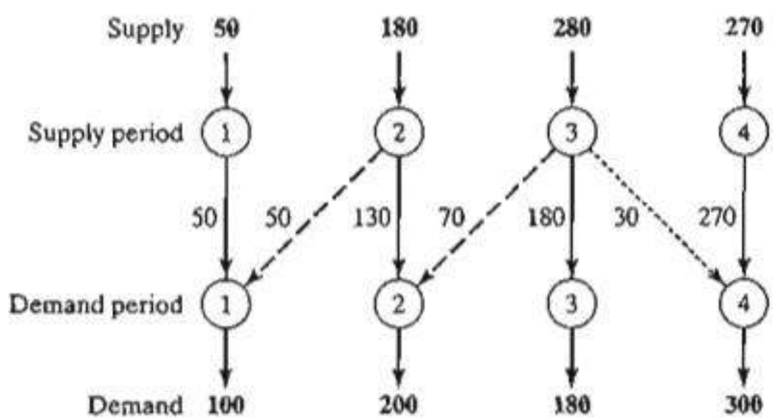


Figure 1.2

Optimal solution of the production- inventory model

The unit "transportation" cost from period i to period is computed as

$$c_{ij} = \begin{cases} \text{Production cost in } i, i = j \\ \text{Production cost in } i + \text{holding cost from } i \text{ to } j, i < j \\ \text{Production cost in } i + \text{penalty cost from } i \text{ to } j, i > j \end{cases}$$

For example,

$c_{11} = \$40.00$

$c_{24} = \$\ 40.00+(\$0.50+\$0.50) = \$\ 41.00$

$c_{41} = \$40.00 + (\$2.00 + \$2.00 + \$2.00) = \$46.00$

The optimal solution is summarized in Figure 1.2. The dashed lines indicate back-ordering, the dotted lines indicate production for a future period, and the solid lines show production in a period for itself. The total cost is $31,455.

Example 4:(Tool Sharpening)

Arkansas Pacific operates a medium-sized saw mill. The mill prepares different types of wood that range from soft pine to hard oak according to a weekly schedule. Depending on the type of wood being milled, the demand for sharp blades varies from day to day according to the following I-week (7-day) data:

Day	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Demand (blades)	24	12	14	20	18	14	22

The mill can satisfy the daily demand in the following manner:

a) Buy new blades at the cost of $12 a blade.

b) Use an overnight sharpening service at the cost of $6 a blade.

c) Use a slow 2-day sharpening service at the cost of $3 a blade.

The situation can be represented as a transportation model with eight sources and seven destinations. The destinations represent the 7 days of the week. The sources of the model are defined as follows: Source 1 corresponds to buying new blades, which, in the extreme case, can provide sufficient supply to cover the demand for all 7 days (= 24 + 12 + 14 + 20 + 18 + 14 + 22 = 124). Sources 2 to 8 correspond to the 7 days of the week. The amount of supply for each of these sources equals the number of used blades at the end of the associated day. For ex-ample, source 2 (i.e., Monday) will have a supply of used blades equal to the demand for Mon-day. The unit "transportation cost" for the model is $12, $6, or $3, depending on whether the blade is supplied from new blades, overnight sharpening, or 2-day sharpening. Notice that the overnight service means that used blades sent at the *end* of day i will be available for use at the *start* of day $i + 1$ or day $i + 2$, because the slow 2-day service will not be available until the *start* of day $i + 3$. The "disposal" column is a dummy destination needed to balance the model. The complete model and its solution are given in Table 1.7

	1 Mon.	2 Tue.	3 Wed.	4 Thu.	5 Fri.	6 Sat.	7 Sun.	8 Disposal	
1-New	$12 24	$12 2	$12	$12	$12	$12	$12	$0 98	124
2-Mon.	M	$6 10	$6 8	$3 6	$3	$3	$3	$0	24
3-Tue.	M	M	$6 6	$6	$3 6	$3	$3	$0	12
4-Wed.	M	M	M	$6 14	$6	$3	$3	$0	14
5-Thu.	M	M	M	M	$6 12	$6	$3 8	$0	20
6-Fri.	M	M	M	M	M	$6 14	$6	$0 4	18
7-Sat.	M	M	M	M	M	M	$6 14	$0	14
8-Sun.	M	M	M	M	M	M	M	$0 22	22
	24	12	14	20	18	14	22	124	

Table 1.7 Tool Sharpening Problem Expressed as a Transportation Model

The problem has alternative optima at a cost of $840.The following table summarizes one such solution.

Period	Number of sharp blades (Target day) New	Overnight	2-day	Disposal
Mon.	24 (Mon.)	10(Tue.) + 8(Wed.)	6 (Thu.)	0
Tues.	2 (Tue.)	6 (Wed.)	6 (Fri.)	0
Wed.	0	14 (Thu.)	0	0
Thu.	0	12 (Fri.)	8 (Sun.)	0
Fri.	0	14 (Sat.)	0	4
Sat.	0	14 (Sun.)	0	0
Sun.	0	0	0	22

Remarks. The model in Table 1.7 is suitable only for the first week of operation because it does not take into account the *rotational* nature of the days of the week, in the sense that this week's days can act as sources for next week's demand. One way to handle this situation is to assume that the very first week of operation starts with all new blades for each day. From then on, we use a model consisting of exactly 7 sources and 7 destinations corresponding to the days of the week. The new model will be similar to Table 1.7 less source "New" and destination "Disposal." Also, only diagonal cells will be blocked (unit cost = *M*). The remaining cells will have a unit cost of either $3.00 or $6.00. For example, the unit cost for cell (Sat., Mon.) is $6.00 and that for cells (Sat., Tue.), (Sat., Wed.), (Sat., Thu.), and (Sat., Fri.) is $3.00. The table below gives the solution costing $372. As expected, the optimum solution will always use the 2-day service only.

	Week *i* + 1							
Week *i*	*Mon.*	*Tue.*	*Wed.*	*Thu.*	*Fri.*	*Sat.*	*Sun.*	Total
Mon.				6			18	24
Tue.					8		4	12
Wed.	12					2		14
Thu.	8	12						20
Fri.	4		14					18
Sat.				14				14
Sun.					10	12		22
Total	24	12	14	20	18	14	22	

1.3.The Transportation Algorithm:

The transportation algorithm follows the exact steps of the simplex method. However, instead of using the regular simplex tableau, we take advantage of the special structure of the transportation model to organize the computations in a more convenient form.

The special transportation algorithm was developed early on when hand computations were the norm and the shortcuts were warranted. Today, we have powerful computer codes that can solve a transportation model of any size as a regular Lp' Nevertheless, the transportation algorithm, aside from its historical significance, does pro-vide insight into the use of the theoretical primal-dual relationships to achieve a practical end result, that of improving hand computations. The details of the algorithm are explained using the following numeric example.

		Mill 1	Mill 2	Mill 3	Mill 4	Supply
Silo	1	10 x_{11}	2 x_{12}	20 x_{13}	11 x_{14}	15
	2	12 x_{21}	7 x_{22}	9 x_{23}	20 x_{24}	25
	3	4 x_{31}	14 x_{32}	16 x_{33}	18 x_{34}	10
Demand		5	15	15	15	

Table 1.8. SunRay Transportation Model

Example 5: (SunRay Transport)

SunRay Transport Company ships truckloads of grain from three silos to four mills. The supply (in truckloads) and the demand (also in truckloads) together with the unit transportation costs per truckload on the different routes are summarized in the transportation model in Table 5.16. The unit transportation costs, *eij*, (shown in the northeast corner of each box) are in hundreds of dollars. The model seeks the minimum-cost shipping schedule *Xij* between silo *i* and mill *j* (i = *1,2,3; j* = 1,2,3,4).

Summary of the Transportation Algorithm. The steps of the transportation algorithm are exact parallels of the simplex algorithm.

Step 1. Determine a *starting* basic feasible solution, and go to step 2.

Step 2. Use the optimality condition of the simplex method to determine the *entering variable* from among all the non-basic variables. If the optimality condition is satisfied, stop. Otherwise, go to step 3.

Step 3. Use the feasibility condition of the simplex method to determine the *leaving variable* from among all the current basic variables, and find the new basic solution. Return to step 2.

1. Determination of the Starting Solution

A general transportation model with *m* sources and *n* destinations has *m* + *n* constraint equations, one for each source and each destination. However, because the transportation model is always balanced (sum of the supply = sum of the demand), one of these equations is redundant. Thus, the model has *m* + *n* - 1 independent constraint equations, which means that the starting basic solution consists of $m + n - 1$ basic variables. Thus, in Example 5.3-1, the starting solution has 3 + 4 - 1 = 6 basic variables.

The special structure of the transportation problem allows securing a non-artificial starting basic solution using one of three methods:

a) Northwest-corner method

b) Least-cost method

c) Vogel approximation method

The three methods differ in the "quality" of the starting basic solution they produce, in the sense that a better starting solution yields a smaller objective value. In general, though not always, the Vogel method yields the best starting basic solution, and the northwest-corner method yields the worst. The trade-off is that the northwest-corner method involves the least amount of computations.

Northwest-Corner Method. The method starts at the northwest-corner cell (route) of the tableau (variable x_{11}).

Step 1. Allocate as much as possible to the selected cell, and adjust the associated amounts of supply and demand by subtracting the allocated amount.

Step 2. Cross out the row or column with zero supply or demand to indicate that no further assignments can be made in that row or column. If both a row and a column net to zero simultaneously, *cross out one only,* and leave a zero sup-ply (demand) in the uncrossed-out TOW (column).

Step 3. If *exactly one* row or column is left uncrossed out, stop. Otherwise, move to the cell to the right if a column has just been crossed out or below if a row has been crossed out. Go to step 1.

Example 6:

The application of the procedure to the model of Example 5. gives the starting basic solution in Table 1.9 The arrows show the order in which the allocated amounts are generated.

The starting basic solution is

$$x_{11} = 5, x_{12} = 10,$$

$$x_{22} = 5, x_{23} = 15, x_{24} = 5$$

$$x_{34} = 10$$

The associated cost of the schedule is

$$z = 5 \times 10 + 10 \times 2 + 5 \times 7 + 15 \times 9 + 5 \times 20 + 10 \times 18 = 520$$

Example 7:

Using North west comer method find a basic feasible Solution to the following Transportation Problem:

	W_1	W_2	W_3	Supply(a_i)
F_1	8	10	12	900
F_2	12	13	12	1000
F_3	14	10	11	1200
Demand	1200	1000	900	

Solution

	W_1	W_2	W_3	Supply(a_i)
F_1	8	10	2	900
F_2	12	13	12	1000
F_3	14	10	11	1200
Demand (b_j)	1200	1000	900	

$\Sigma a_i = 900 + 1000 + 1200$

$= 3100$

$\Sigma b_j = 1200 + 1000 + 900$

$= 3100$

$\Sigma a_i = \Sigma b_j$

The Transportation problem is balanced basic feasible Solution.

	W_1	W_2	W_3	Supply
F_1	300 8	10	12	900/0
F_2	12	13	12	1000
F_3	14	10	11	1200
Demand	1200/300	1000	900	

	W_1	W_2	W_3	Supply
F_2	300 12	13	12	1000/700
F_3	14	10	11	1200
Demand	300/0	1000	900	

	W_2	W_3	Supply
F_2	700 13	12	700/0
F_3	10	11	1200/900
Demand	1000/300	900	

	W_2	W_3	Supply
F_3	300 10	900 11	1200
Demand	300	900	

Initial basic feasible Solution is given in the following table

	W_1	W_2	W_3	Supply
F_1	900 8	10	12	900
F_2	300 12	700 13	12	1000
F_3	14	10	900 11	1200

Demand1200 1000900x_1=900,

x_{21}=300, x_{22}=700 x_{32}=300,

x_{33}=9000

The Total transportation cost is

$z=\sum_{i=1}^{n} \sum_{j=1}^{m} C_{ij}x_{ij}$

=8x900+12x300+13x700+10x300+11x900

=7200+3600+9100+3000+9900

=32800

Least-Cost Method:

The least-cost method finds a better starting solution by concentrating on the cheapest routes. The method assigns as much as possible to the cell with the smallest unit cost (ties are broken arbitrarily). Next, the satisfied row or column is crossed out and the amounts of supply and demand are adjusted accordingly.

	1	2	3	4	Supply
1	10 (5) →	2 (10) ↓	20	11	15
2	12	7 (5) →	9 (15) →	20 (5) ↓	25
3	4	14	16	18 (10)	10
Demand	5	15	15	15	

Table 1.9. Northwest-Corner Starting Solution

If both a row and a column are satisfied simultaneously, *only one is crossed out*, the same as in the northwest-corner method. Next, look for the uncrossed-out cell with the smallest unit cost and repeat the process until exactly one row or column is left uncrossed out.

Example 8:

The least-cost method is applied to Example 5 in the following manner:

1. Cell (1, 2) has the least unit cost in the tableau (= $2). The most that can be shipped through (1,2) is X_{12} = 15 truckloads, which happens to satisfy both row 1 and column 2 simultaneously. We arbitrarily cross out column 2 and adjust the supply in row 1 to 0.

2. Cell (3,1) has the smallest uncrossed-out unit cost (= $4). Assign X_{31} = 5, and cross out column 1 because it is satisfied, and adjust the demand of row 3 to 10 - 5 = 5 truckloads.

3. Continuing in the same manner, we successively assign 15 truckloads to cell (2, 3), 0 truckloads to cell (1,4), 5 truckloads to cell (3, 4), and 10 truckloads to cell (2,4) (verify!).

The resulting starting solution is summarized in Table 1.10. The arrows show the order in which the allocations are made. The starting solution (consisting of 6 basic variables) is x_{12} = 15, x_{14} = 0, x_{23} = 15, x_{24} = 10, x_{31} = 5, x_{34} = 5. The associated objective value is

z = 15 x 2 + 0 x 11 + 15 x 9 + 10 x 20 + 5 x 4 + 5 x 18

=30+0+135+200+20+90

= $475

The quality of the least-cost starting solution is better than that of the northwest-corner method (Example 6) because it yields a smaller value of *z* ($475 versus $520 in the northwest-corner method).

Example 9:

Solve the following transportation problem by using least cost method

	W_1	W_2	W_3	Supply
F_1	8	10	12	900
F_2	12	13	12	1000
F_3	14	10	11	1200
Demand	1200	1000	900	

Solution

	W_1	W_2	W_3	Supply
F_1	900 8	10	12	900/0
F_2	12	13	12	1000
F_3	14	10	11	1200
Demand	1200/300	1000	900	

	W_1	W_2	W_3	Supply	Row penalty
F_2	12	13	12	1000	(0)
F_3	14	1000 10	11	1200/200	(1)
Demand	300	1000/0	900		
Column Penalty	(2)	(3) ↑	(1)		

	W_1	W_3	Supply	Row penalty
F_2	12	12	1000	(0)
F_3	14	200 11	200	(3)←
Demand	300	900/700		
Column Penalty	(2)	(1)		

	W_1	W_3	Supply	Row penalty
F_2	12	12	1000	(0)
Demand	300	700		
Column Penalty	(12)	(12)		

Initial Basic Feasible Solution

	W_1	W_2	W_3	Supply a_2
F_1	900 8	10	12	900
F_2	300 12	13	700 12	1000
F_3	14	1000 10	200 11	1200
Demand	1200	1000	900	

Total Transportation cost is

$$z=8 \times 900+300 \times 12+12 \times 700+10 \times 1000+11 \times 200$$

$$=7200+3600+8400+10000+2200=31400$$

Vogel Approximation Method (VAM):

VAM is an improved version of the least-cost method that generally, but not always, produces better starting solutions.

Step 1. For each row (column), determine a penalty measure by subtracting the *smallest* unit cost element in the row (column) from the *next smallest* unit cost element in the same row (column).

	1	2	3	4	Supply
1	10	(start) 2 15	20	11 0	15
2	12	7	9 15	(end) 20 10	25
3	4 5	14	16	18 5	10
Demand	5	15	15	15	

Table 1.10. Least-Cost Starting Solution

Step 2.

Identify the row or column with the largest penalty. Break ties arbitrarily. Allocate as much as possible to the variable with the least unit cost in the selected row or column. Adjust the supply and demand, and cross out the satisfied row *or* column. If a row and a column are satisfied simultaneously, only one of the two is crossed out, and the remaining row (column) is assigned zero supply (demand).

Step 3.

(a) If exactly one row or column with zero supply or demand remains un-crossed out, stop.

(b) If one row (column) with *positive* supply (demand) remains uncrossed out, determine the basic variables in the row (column) by the least-cost method. Stop.

(c) If all the uncrossed out rows and columns have (remaining) zero supply and demand, determine the *zero* basic variables by the least-cost method. Stop.

(d) Otherwise, go to step 1.

Example 10:

VAM is applied to Example 5. Table 1.11 computes the first set of penalties.

Because row 3 has the largest penalty (= 10) and cell (3, 1) has the smallest unit cost in that row, the amount 5 is assigned to x_{31}. Column 1 is now satisfied and must be crossed out. Next, new penalties are recomputed as in Table 1.12.

Table 1.12 shows that row 1 has the highest penalty (= 9). Hence, we assign the maximum amount possible to cell (1,2), which yields $x_{12} = 15$ and simultaneously satisfies both row 1 and 5. column 2. We arbitrarily cross out column 2 and adjust the supply in row 1 to zero.

Continuing in the same manner, row 2 will produce the highest penalty (= 11), and we as-sign x_{23} = 15, which crosses out column 3 and leaves 10 units in row 2. Only column 4 is left, and it has a positive supply of 15 units. Applying the least-cost method to that column, we successively assign $x_{14} = 0$, $x_{34} = 5$, and $x_{24} = 10$ (verify!). The associated objective value for this solution is

$z = 15 \times 2 + 0 \times 11 + 15 \times 9 + 10 \times 20 + 5 \times 4 + 5 \times 18$

$=30+0+135+200+20+90$

$= \$475$

This solution happens to have the same objective value as in the least-cost method.

	1	2	3	4		Row penalty
1	10	2	20	11	15	10 − 2 = 8
2	12	7	9	20	25	9 − 7 = 2
3	4 5	14	16	18	10	14 − 4 = 10
	5	15	15	15		
Column penalty	10 − 4 = 6	7 − 2 = 5	16 − 9 = 7	18 − 11 = 7		

Table 1.11. Row and Column Penalties in VAM

	1	2	3	4		Row penalty
1	10	2	20	11	15	9
2	12	7	9	20	25	2
3	4 5	14	16	18	10	2
	5	15	15	15		
Column penalty	—	5	7	7		

Table 1.12. First Assignment in VAM (x_{31} = 5)

Example 11:

Using Vogel approximation method find the basic solution to the following transportation method.

	W_1	W_2	W_3	Supply
F_1	8	10	12	900
F_2	12	13	12	1000
F_3	14	10	11	1200
Demand	1200	1000	900	

Solution:

	W_1	W_2	W_3	Supply	Row penalty
F_1	900 8	10	12	900/0	(2)
F_2	12	13	12	1000	(0)
F_3	14	10	11	1200	(1)
Demand	1200/300	1000	900		
Column Penalty	(4) ↑	(0)	(1)		

	W_1	W_2	W_3	Supply	Row penalty
F_2	12	13	12	1000	(0)
F_3	14	1000 10	11	1200/200	(1)
Demand	300	1000/0	900		
Column Penalty	(2)	(3) ↑	(1)		

	W_1	W_3	Supply	Row penalty
F_2	12	12	1000	(0)
F_3	14	200 11	200	(3)←
Demand	300	900/700		
Column Penalty	(2)	(1)		

	W_1	W_3	Supply	Row penalty
F_2	12	12	1000	(0)
Demand	300	700		
Column Penalty	(12)	(12)		

Initial Basic Feasible Solution

	W_1	W_2	W_3	Supplya_2
F_1	900 8	10	12	900
F_2	300 12	13	700 12	1000
F_3	14	1000 10	200 11	1200
Demand	1200	1000	900	

Total Transportation cost is

$$z = 8 \times 900 + 300 \times 12 + 12 \times 700 + 10 \times 1000 + 11 \times 200$$

$$= 7200 + 3600 + 8400 + 10000 + 2200 = 31400$$

Exercises 1:

Compare the starting solutions obtained by the northwest-corner, least-cost, and Vogel methods for each of the following models:

*(a)

0	2	1	6
2	1	5	7
2	4	3	7
5	5	10	

(b)

1	2	6	7
0	4	2	12
3	1	5	11
10	10	10	

(c)

5	1	8	12
2	4	0	14
3	6	7	4
9	10	11	

2. Iterative Computations of the Transportation Algorithm

After determining the starting solution (using any of the three methods in Section 5.3.1), we use the following algorithm to determine the optimum solution:

Step 1. Use the simplex *optimality condition* to determine the *entering variable* as the current non-basic variable that can improve the solution. If the optimality condition is satisfied, stop. Otherwise, go to step 2.

Step 2. Determine the *leaving variable* using the simplex *feasibility condition.* Change the basis, and return to step 1.

The optimality and feasibility conditions do not involve the familiar row operations used in the simplex method. Instead, the special structure of the transportation model allows simpler computations.

Example 9:

Solve the transportation model of Example 5, starting with the northwest-corner solution.

Table 1.13 gives the northwest-corner starting solution as determined in Table 1.9, Example 6

The determination of the entering variable from among the current nonbasic variables (those that are not part of the starting basic solution) is done by computing the nonbasic coefficients in the z-row, using the method of multipliers.

In the method of multipliers, we associate the multipliers *Ui* and *v;* with row *i* and column j of the transportation tableau. For each current *basic* variable *Xi;*, these multipliers are to satisfy the following equations:

$u_i + v_j = c_{ij}$, for each *basic* x_{ij}

As Table 1.13 shows, the starting solution has 6 basic variables, which leads to 6 equations in 7 unknowns. To solve these equations, the method of multipliers calls for arbitrarily setting any $u_i = 0$, and then solving for the remaining variables as shown below.

Basic Variable	(u, v) Equation	Solution
x_{11}	$u_1 + v_1 = 10$	Set $u_1 = 0 \rightarrow v_1 = 10$
x_{12}	$u_1 + v_2 = 2$	$u_1 = 0 \rightarrow v_1 = 2$
x_{22}	$u_2 + v_2 = 7$	$v_2 = 2 \rightarrow u_2 = 5$
x_{23}	$u_2 + v_3 = 9$	$u_2 = 5 \rightarrow v_3 = 4$
x_{24}	$u_2 + v_4 = 20$	$u_2 = 5 \rightarrow v_4 = 15$
x_{34}	$u_3 + v_4 = 18$	$v_4 = 15 \rightarrow u_3 = 3$

To summarize, we have

$u_1 = 0,\ u_2 = 5, u_3 = 3$

$v_1 = 10,\ v_2 = 2, v_3 = 4, v_4 = 15$

Next, we use u_j and v_j to evaluate the non-basic variables by computing $u_j + v_j - c_{ij}$, for each Non-basic x_{ij}

	1	2	3	4	Supply
1	10 5	2 10	20	11	15
2	12	7 5	9 15	20 5	25
3	4	14	16	18 10	10
Demand	5	15	15	15	

Table 1.13. Starting Iteration

The results of these evaluations are shown in the following table:

Nonbasic Variable	$u_j + v_j - c_{ij}$
x_{13}	$u_1 + v_3 - c_{13} = 0 + 4 - 20 = -16$
x_{14}	$u_1 + v_4 - c_{14} = 0 + 15 - 11 = 4$
x_{21}	$u_2 + v_1 - c_{21} = 5 + 10 - 12 = 3$
x_{31}	$u_3 + v_1 - c_{31} = 3 + 10 - 4 = 9$
x_{32}	$u_3 + v_2 - c_{32} = 3 + 2 - 14 = -9$
x_{33}	$u_3 + v_3 - c_{33} = 3 + 4 - 16 = -9$

The preceding information, together with the fact that $u_i + v_j - c_{ij} = 0$ for each basic x_{ij}, is actually equivalent to computing the *z-row* of the simplex tableau, as the following summary shows.

Basic	x_{11}	x_{12}	x_{13}	x_{14}	x_{23}	x_{22}	x_{24}	x_{31}	x_{32}	x_{33}	x_{34}
z	0	0	-16	4	3	0	0	9	-9	-9	0

Because the transportation model seeks to *minimize* cost, the entering variable is the one having the *most positive* coefficient in the z-row. Thus, x_{31} is the entering variable.

The preceding computations are usually done directly on the transportation tableau as shown in Table 1.14, meaning that it is not necessary really to write the *(u,* v)-equations explicitly. Instead, we start by setting $u_1 = 0$. Then we can compute the v-values of all the columns that have *basic* variables in row I-namely, v_I and v_2. Next, we compute u_2 based on the *(u,* v)-equation of basic x_{22}. Now, given u_2, we can compute v_3 and v_4. Finally, we determine u_3 using the basic equation of x_{33}. Once all the *u's* and v's have been determined, we can evaluate the non-basic variables by computing $u_i + v_j - c_{ij}$ for each non-basic X_{ij}. These evaluations are shown in Table 1.14 in the boxed southeast corner of each cell.

Having identified x_{31} as the entering variable, we need to determine the leaving variable. Remember that if x_{31} enters the solution to become basic, one of the current basic variables must leave as non-basic (at zero level).

	$v_1 = 10$	$v_2 = 2$	$v_3 = 4$	$v_4 = 15$	Supply
$u_1 = 0$	5 (cost 10)	10 (cost 2)	(cost 20) [−16]	(cost 11) [4]	15
$u_2 = 5$	(cost 12) [3]	5 (cost 7)	15 (cost 9)	5 (cost 20)	25
$u_3 = 3$	(cost 4) [9]	(cost 14) [−9]	(cost 16) [−9]	10 (cost 18)	10
Demand	5	15	15	15	

Table 1.14. Iteration 1 Calculations

The selection of x_{31} as the entering variable means that we want to ship through this route because it reduces the total shipping cost. What is the most that we can ship through the new route? Observe in Table 1.14 that if route (3, 1) ships () units (i.e., $x_{31} = \Phi$), then the maximum value of Φ is determined based on two conditions.

a) Supply limits and demand requirements remain satisfied.

b) Shipments through all routes remain nonnegative.

These two conditions determine the maximum value of Φ and the leaving variable in the following manner. First, construct a *closed loop* that starts and ends at the entering variable cell, (3, 1). The loop consists of *connected horizontal* and *vertical* segments only (no diagonals are al-lowed).7 Except for the entering variable cell, each corner of the closed loop must coincide with a basic variable. Table 1.15 shows the loop for x_{31}. Exactly one loop exists for a given entering variable.

Next, we assign the amount Φ to the entering variable cell (3, 1). For the supply and demand limits to remain satisfied, we must alternate between subtracting and adding the amount Φ at the successive *corners* of the loop as shown in Table 1.15 (it is immaterial whether the loop is traced in a clockwise or counter clockwise direction). For $\Phi \geq 0$, the new values of the variables then remain nonnegative if

$x_{11} = 5 - \theta \geq 0$

$x_{22} = 5 - \theta \geq 0$, $x_{34} = 10 - \theta \geq 0$

The corresponding maximum value of Φ is 5, which occurs when both x_{11} and x_{22} reach zero level. Because only one current basic variable must leave the basic solution, we can choose either x_{11} or x_{22} as the leaving variable. We arbitrarily choose X_{11} to leave the solution.

The selection of x_{31} (= 5) as the entering variable and x_{11} as the leaving variable requires adjusting the values of the basic variables at the corners of the closed loop as Table 1.16 shows. Because each unit shipped through route (3, 1) reduces the shipping cost by $9 (= $u_3 + v_1 - c_{31}$), the total cost associated with the new schedule is $9 X 5 = $45 less than in the previous schedule. Thus, the new cost is $520 - $45 = $475.

	$v_1 = 10$	$v_2 = 2$	$v_3 = 4$	$v_4 = 15$	Supply
$u_1 = 0$	10 $5 - \Theta$	2 $10 + \Theta$	20 −16	11 4	15
$u_2 = 5$	12 3	7 $5 - \Theta$	9 15	20 $5 + \Theta$	25
$u_3 = 3$	4 Θ 9	14 −9	16 −9	18 $10 - \Theta$	10
Demand	5	15	15	15	

Table 1.15. Determination of Closed Loop for x_{31}

	$v_1 = 1$	$v_2 = 2$	$v_3 = 4$	$v_4 = 15$	Supply
$u_1 = 0$	10 −9	2 $15 - \Theta$	20 −16	11 Θ 4	15
$u_2 = 5$	12 −6	7 $0 + \Theta$	9 15	20 $10 - \Theta$	25
$u_3 = 3$	4 5	14 −9	16 −9	18 5	10
Demand	5	15	15	15	

Table 1.16. Iteration 2 Calculations

	$v_1 = -3$	$v_2 = 2$	$v_3 = 4$	$v_4 = 11$	Supply
$u_1 = 0$	10; −13	2; **5**	20; −16	11; **10**	**15**
$u_2 = 5$	12; −10	7; **10**	9; **15**	20; −4	**25**
$u_3 = 7$	4; **5**	14; −5	16; −5	18; **5**	**10**
Demand	**5**	**15**	**15**	**15**	

Table 1.17. Iteration 3 Calculations (Optimal)

Given the new basic solution, we repeat the computation of the multipliers *u* and *v*, as Table 1.17 shows. The entering variable is x_{14}. The closed loop shows that $x_{14} = 10$ and that the leaving variable is x_{24}.

The new solution, shown in Table 1.17, costs \$4 X 10 = \$40 less than the preceding one, thus yielding the new cost \$475 - \$40 = \$435. The new $u_i + v_j - c_{ij}$ are now negative for all non-basic x_{ij}. Thus, the solution in Table 1.17 is optimal.

The following table summarizes the optimum solution.

From silo	To mill	Number of truckloads
1	2	5
1	4	10
2	2	10
2	3	15
3	1	5
3	4	5
Optimal cost = \$435		

3. Simplex Method Explanation of the Method of Multipliers

The relationship between the method of multipliers and the simplex method can be explained based on the primal-dual relationships. From the special structure of the LP representing the transportation model (see Example 1 for an illustration), the associated dual problem can be written as

$$\text{Maximize } z = \sum_{i=1}^{m} a_i u_i + \sum_{j=1}^{n} b_j v_j$$

subject to

$$u_i + v_j \leq c_{ij}, \text{ all } i \text{ and } j$$

$$u_i \text{ and } v_j \text{ unrestricted}$$

where

a_i = Supply amount at source i

b_j = Demand amount at destination j

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

u_i = Dual variable of the constraint associated with source i

v_j = Dual variable of the constraint associated with destination j

the objective-function coefficients (reduced costs) of the variable x_{ij} equal the difference between the left- and right-hand sides of the corresponding dual constraint-that is, $u_i + v_j - c_{ij}$. However, we know that this quantity must equal zero for each *basic variable*, which then produces the following result: $u_i + v_j = c_{ij}$, for each basic variable x_{ij}

There is $m + n - 1$ such equations whose solution (after assuming an arbitrary value $u_1 = 0$) yields the multipliers u_i and v_j. Once these multipliers are computed, the entering variable is determined from all the *non-basic* variables as the one having the largest positive $u_i + v_j - c_{ij}$.

1.4.The Assignment Model:

"The best person for the job" is an apt description of the assignment model. The situation can be illustrated by the assignment of workers with varying degrees of skill to jobs. A job that happens to match a worker's skill costs less than one in which the operator is not as skilful. The objective of the model is to determine the minimum-cost assignment of workers to jobs.

The general assignment model with n workers and n jobs is represented in Table 1.18.

The element c_{ij} represents the cost of assigning worker i to job j $(i, j = 1, 2, ..., n)$. There is no loss of generality in assuming that the number of workers always

		Jobs				
		1	2	...	n	
	1	c_{11}	c_{12}	...	c_{1n}	1
	2	c_{21}	c_{22}	...	c_{2n}	1
Worker	⋮	⋮	⋮	⋮	⋮	⋮
	n	c_{n1}	c_{n2}	...	c_{nn}	1
		1	1	...	1	

Table 1.18. Assignment model

equals the number of jobs, because we can always add fictitious workers or fictitious jobs to satisfy this assumption.

The assignment model is actually a special case of the transportation model in which the workers represent the sources, and the jobs represent the destinations. The supply (demand) amount at each source (destination) exactly equals 1. The cost of "transporting" worker i to job j is c_{ij}. In effect, the assignment model can be solved directly as a regular transportation model. Nevertheless, the fact that all the supply and demand amounts equal 1 has led to the development of a simple solution algorithm called the Hungarian method. Although the new solution method appears totally un-related to the transportation model, the algorithm is actually rooted in the simplex method, just as the transportation model is.

1. The Hungarian Method

We will use two examples to present the mechanics of the new algorithm. The next section provides a simplex-based explanation of the procedure.

Example 10:

Joe Klyne's three children, John, Karen, and Terri, want to earn some money to take care of personal expenses during a school trip to the local zoo. Mr. Klyne has chosen three chores for his children: mowing the lawn, painting the garage door, and washing the family cars. To avoid anticipated sibling competition, he asks them to submit (secret) bids for what they feel is fair pay for each of the three chores. The understanding is that a U three children will abide by their father's decision as to who gets which chore. Table 5.32 summarizes the bids received. Based on this information, how should Mr. Klyne assign the chores?

The assignment problem will be solved by the Hungarian method.

Step 1. For the original cost matrix, identify each row's minimum, and subtract it from all the entries of the row.

	Mow	Paint	Wash
John	$15	$10	$9
Karen	$9	$15	$10
Terri	$10	$12	$8

Table 1.19. Klyne's Assignment Problems

Step 2. For the matrix resulting from step 1, identify each column's minimum, and subtract it from all the entries of the column.

Step 3. Identify the optimal solution as the feasible assignment associated with the zero elements of the matrix obtained in step 2.

Let p_i and q_j be the minimum costs associated with row i and column j as defined in steps 1 and 2, respectively. The row minimums of step 1 are computed from the original cost matrix as shown in Table 1.20

Next, subtract the row minimum from each respective row to obtain the reduced matrix in Table 1.21

The application of step 2 yields the column minimums in Table 1.21. Subtracting these values from the respective columns, we get the reduced matrix in Table 1.22

	Mow	Paint	Wash	Row minimum
John	15	10	9	$p_1 = 9$
Karen	9	15	10	$p_2 = 9$
Terri	10	12	8	$p_3 = 8$

Table 1.20. Step 1 of the Hungarian Method

	Mow	Paint	Wash
John	6	1	0
Karen	0	6	1
Terri	2	4	0
Column minimum	$q_1 = 0$	$q_2 = 1$	$q_3 = 0$

Table 1.21. Step 2 of the Hungarian Method

	Mow	Paint	Wash
John	6	0	0
Karen	0	5	1
Terri	2	3	0

Table 1.22. Step 3 of the Hungarian Method

The cells with underscored zero entries provide the optimum solution. This means that John gets to paint the garage door, Karen gets to mow the lawn, and Terri gets to wash the family cars. The total cost to Mr. Klyne is 9 + 10 + 8 = $27. This amount also will always equal $(p_1 + p_2 + p_3) + (q_1 + q_2 + q_3)$ = (9 + 9 + 8) + (0 + 1 + 0) = $27. (A justification of this result is given in the next section.)

The given steps of the Hungarian method work well in the preceding example because the zero entries in the final matrix happen to produce a *feasible* assignment (in the sense that each child is assigned a distinct chore). In some cases, the zeros created by steps 1 and 2 may not yield a feasible solution directly, and further steps are needed to find the optimal (feasible) assignment. The following example demonstrates this situation.

Example 11:

Suppose that the situation discussed in Example 10 is extended to four children and four chores. Table 1.23 summarizes the cost elements of the problem.

The application of steps 1 and 2 to the matrix in Table 1.23 (using $p_1 = 1$, $p_2 = 7$, $p_3 = 4$, $p_4 = 5$, $q_1 = 0$, $q_2 = 0$, $q_3 = 3$, and $q_4 = 0$) yields the reduced matrix in Table 1.24 (verify!).

The locations of the zero entries do not allow assigning unique chores to all the children. For example, if we assign child 1 to chore 1, then column 1 will be eliminated, and child 3 will not have a zero entry in the remaining three columns. This obstacle can be accounted for by adding the following step to the procedure outlined in Example 10

Step 2a. If no feasible assignment (with all zero entries) can be secured from steps 1 and 2,

(i) Draw the *minimum* number of horizontal and vertical lines in the last reduced matrix that will cover *all* the zero entries.

		Chore 1	Chore 2	Chore 3	Chore 4
Child	1	$1	$4	$6	$3
	2	$9	$7	$10	$9
	3	$4	$5	$11	$7
	4	$8	$7	$8	$5

Table 1.23. Assignment Model

		Chore 1	Chore 2	Chore 3	Chore 4
Child	1	0	3	2	2
	2	2	0	0	2
	3	0	1	4	3
	4	3	2	0	0

Table 1.24. Reduced Assignment Matrix

		Chore 1	Chore 2	Chore 3	Chore 4
Child	1	0	3	2	2
	2	2	0	0	2
	3	0	1	4	3
	4	3	2	0	0

Table 1.25. Application of step 2a

		Chore 1	Chore 2	Chore 3	Chore 4
Child	1	**0**	2	1	1
	2	3	0	**0**	2
	3	0	**0**	3	2
	4	4	2	0	**0**

Table 1.26. Optimal Assignment

(ii) Select the *smallest* uncovered entry, subtract it from every uncovered entry, then add it to every entry at the intersection of two lines.

(iii) If no feasible assignment can be found among the resulting zero entries, repeat step 2a. Otherwise, go to step 3 to determine the optimal assignment.

The application of step 2a to the last matrix produces the shaded cells in Table 1.25. The smallest unshaded entry (shown in italics) equals 1. 111is entry is added to the bold intersection cells and subtracted from the remaining shaded cells to produce the matrix in Table 1.26

The optimum solution (shown by the underscored zeros) calls for assigning child 1 to chore 1, child 2 to chore 3, child 3 to chore 2, and child 4 to chore 4. The associated optimal cost is 1 + 10 + 5 + 5 = $21. The same cost is also determined by summing the p_is, the q_is, and the entry that was subtracted after the shaded cells were determined-that is, (1 + 7 + 4 + 5) + (0 + 0 + 3 + 0) + (1) = $21.

2. Simplex Explanation of the Hungarian Method

The assignment problem in which *n* workers are assigned to *n* jobs can be represented as an LP model in the following manner: Let c_{ij} be the cost of assigning worker *i* to job *j*, and define

$$x_{ij} = \begin{cases} 1, & \textit{if worker i is assigned to job j} \\ 0, & \textit{otherwise} \end{cases}$$

Then the LP model is given as

Minimize $z=\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}x_{ij}$

Subject to

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1,2, \ldots .. n$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad j = 1,2, \ldots .. n$$

$x_{ij} = 0 \text{ or } 1$

The optimal solution of the preceding LP model remains unchanged if a constant is added to or subtracted from any row or column of the cost matrix (c_{ij}). To prove this point, let p_i and q_j be constants subtracted from row *i* and column j. Thus, the cost element c_{ij} is changed to

$$c'_{ij} = c_{ij} - p_i - q_j$$

Now

$$\sum_i \sum_j c'_{ij}x_{ij} = \sum_i \sum_j (c_{ij} - p_i - q_j)x_{ij} = \sum_i \sum_j c_{ij}x_{ij} - \sum_i p_i\left(\sum_j x_{ij}\right) - \sum_j q_j\left(\sum_i x_{ij}\right)$$

$$= \sum_i \sum_j c_{ij}x_{ij} - \sum_i p_i(1) - \sum_j q_j(1)$$

$$= \sum_i \sum_j c_{ij}x_{ij} - \text{constant}$$

Because the new objective function differs from the original one by a constant, the optimum values of x_{ij} must be the same in both cases. The development thus shows that steps 1 and 2 of the Hungarian method, which call for subtracting p_i from row *i* and then subtracting q_j from column j, produce an equivalent assignment model. In this regard, if a feasible solution can be found among the zero entries of the cost matrix created by steps 1 and 2, then it must be optimum because the cost in the modified matrix cannot be less than zero.

If the created zero entries cannot yield a feasible solution, then step 2a (dealing with the covering of the zero entries) must be applied.

The reason $(p_1 + p_2 + \ldots + p_n) + (q_1 + q_2 + \ldots + q_n)$ gives the optimal objective value is that it represents the dual objective function of the assignment model.

UNIT II

Network Analysis: Network Definitions-Minimal Spanning Tree Algorithm- Shortest Route Problem-Maximum Flow Model-CM-PERT.

Chapter 2: Sections 2.1-2.5

2.1. Network Definition:

A network consists of a set of nodes linked by arcs (or branches). The notation for describing a network is *(N, A)*, where *N* is the set of nodes and *A* is the set of arcs. As an illustration, the network in Figure 6.1 is described as

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

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

Associated with each network is a flow (e.g., oil products flow in a pipeline and automobile traffic flows in highways). In general, the flow in a network is limited by the capacity of its arcs, which may be finite or infinite.

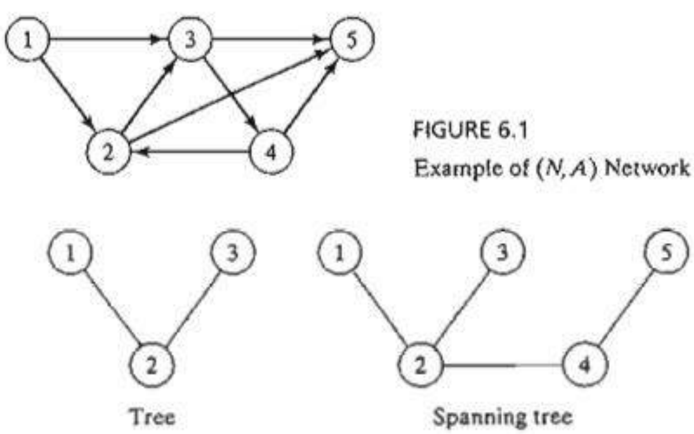


FIGURE 6.1
Example of (*N*, *A*) Network

Figure 2.1

An arc is said to be directed or oriented if it allows positive flow in one direction and zero flow in the opposite direction. A directed network has all directed arcs.

A path is a sequence of distinct arcs that join two nodes through other nodes regardless of the direction of flow in each arc. A path forms a cycle or a loop if it connects a node to itself through other nodes. For example, in Figure 2.1, the arcs (2,3), (3, 4), and (4,2) form a cycle.

A connected network is such that every two distinct nodes are linked by at least one path. The network in Figure 2.1 demonstrates this type of network. A tree is a *cycle-free* connected network comprised of a *subset* of all the nodes, and a spanning tree is a tree that links *all* the nodes of the network. Figure 2.2 provides examples of a tree and a spanning tree from the network in Figure 2.1.

Example 1: (Bridges of Konigsberg)

The Prussian city of Konigsberg (now Kalingrad in Russia) was founded in 1254 on the banks of river Pergel with seven bridges connecting its four sections (labeled *A, B,* C, and *D)* as shown in Figure 2.3. A problem circulating among the inhabitants of the city was to find out if a *round trip*

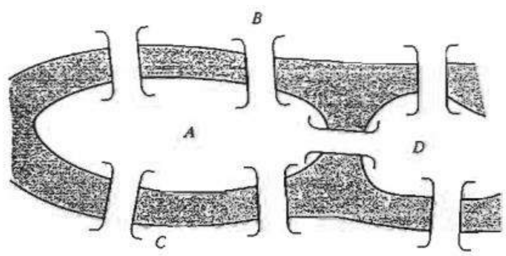


Figure 2.2

Bridges of Konigsberg

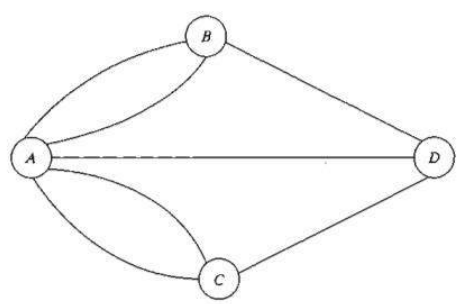


Figure 2.3

Network representation of Konigsberg problem

of the four sections could be made with each bridge being crossed exactly once. No limits were set on the number of times any of the four sections could be visited.

In the mid-eighteenth century, the famed mathematician Leonhard Euler developed a special "path construction" argument to prove that it was impossible to make such a trip. Later, in the early nineteenth century the same problem was solved by representing the situation as a net-work in which each of the four sections *(A, B,* C, and *D)* is a node and each bridge is an arc joining applicable nodes, as shown in Figure 2.3.

The network-based solution is that the desired round trip (starting and ending in one section of the city) is impossible, because there are four nodes and each is associated with an *odd* number of arcs, which does not allow distinct entrance and exit (and hence distinct use of the bridges) to each section of the city. The example demonstrates how the solution of the problem is facilitated by using network representation.

2.2. Minimal Spanning Tree Algorithm:

The minimal spanning tree algorithm deals with linking the nodes of a network, directly or indirectly, using the shortest total length of connecting branches. A typical application occurs in the construction of paved roads that link several rural towns. The road between two towns may pass through one or more other towns. The most economical design of the road system calls for minimizing the total miles of paved roads, a result that is achieved by implementing the minimal spanning tree algorithm.

The steps of the procedure are given as follows. Let $N = \{I, 2, \ldots, n\}$ be the set of nodes of the network and define

C_k =Set of nodes that have been permanently connected at iteration k

$\overline{C_k}$ = Set of nodes as yet to be connected permanently after iteration k

Ste 0. Set $C_0 = \emptyset$ and $\overline{C_0} = N$.

Step 1. Start with any node I in the unconnected set $\overline{C_0}$ and set $C_1 = \{i\}$, which renders

$\overline{C_1} = N - \{i\}$. Set k=2.

General Step k: Select a node, j^*, in the unconnected set $\overline{C_{k-1}}$ that yields the shortest arc to a node in the connected set C_{k-1}. Link j^* permanently to C_{k-1} and remove it from $\overline{C_{k-1}}$; that is,

$C_k = C_{k-1} + \{j^*\}$, $\overline{C_k} = \overline{C_{k-1}} - \{j^*\}$

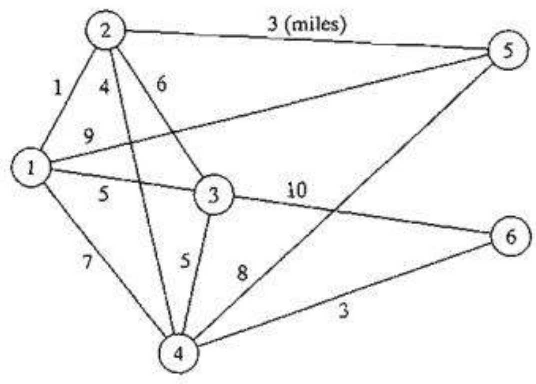


Figure 2.4

Cable connections for Midwest TV Company

Example 2:

Midwest TV Cable Company is in the process of providing cable service to five new housing development areas. Figure 6.6 depicts possible TV linkages among the five areas. The cable miles are shown on each arc. Determine the most economical cable network.

The algorithm starts at node 1 (any other node will do as well), which gives

$$C_1 = \{1\}, \overline{C}_1 = \{2, 3, 4, 5, 6\}$$

The iterations of the algorithm are summarized in Figure 2.5. The thin arcs provide all the candidate links between C and Bar(C). The thick branches represent the permanent links between the nodes of the connected set c: and the dashed branch represents the new (permanent) link added at each iteration. For example, in iteration 1, branch (1,2) is the shortest link (= 1 mile) among all the candidate branches from node 1 to nodes 2,3,4,5, and 6 of the unconnected set Bar(C_1). Hence, link (1,2) is made permanent and $j^* = 2$, which yields $C_2 = \{1,2\}, \overline{C_2} = \{3,4,5,6\}$

The solution is given by the minimal spanning tree shown in iteration 6 of Figure 6.7. The resulting minimum cable miles needed to provide the desired cable service are 1 + 3 + 4 + 3 + 5 = 16 miles.

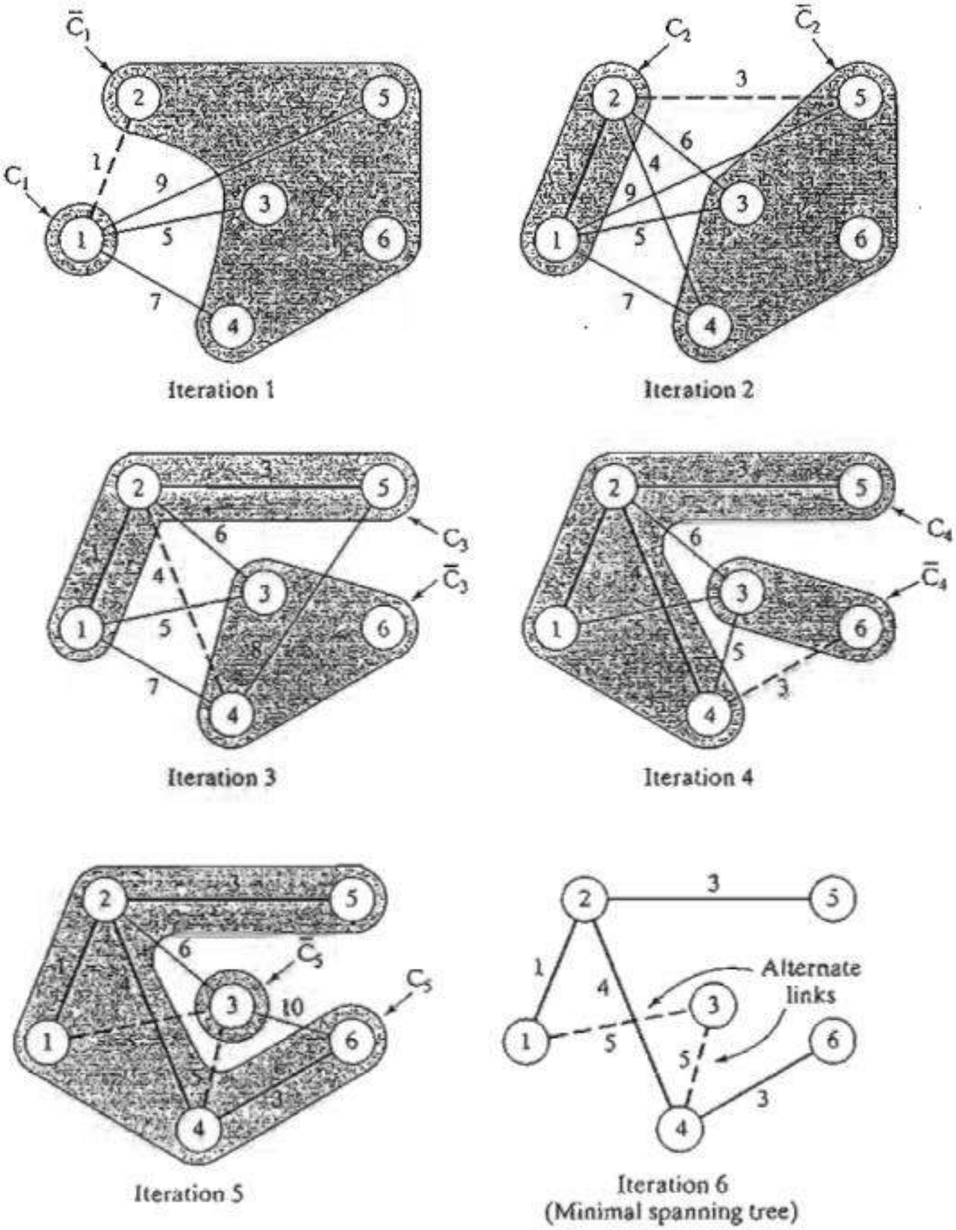


Figure 2.5

Solution iterations for Midwest TV Company

2.3. Shortest Route Problem:

The shortest-route problem determines the shortest route between a source and destination in a transportation network. Other situations can be represented by the same model, as illustrated by the following examples.

Examples of the Shortest-Route Applications

Example 4 :(Equipment Replacement)

Rent Car is developing a replacement policy for its car fleet for a 4-year planning horizon. At the start of each year, a decision is made as to whether a car should be kept in operation or replaced.

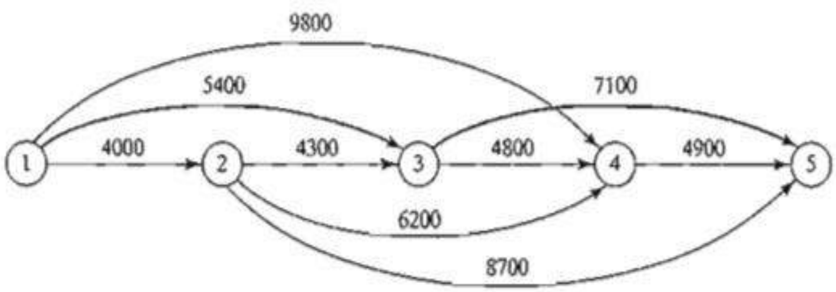


Figure 2.6

Equipment replacement problem as a shortest route model

A car must be in service a minimum of 1 year and a maximum of 3 years. The following table provides the replacement cost as a function of the year a car is acquired and the number of years in operation.

Equipment acquired at start of year	Replacement cost ($) for given years in operation		
	1	*2*	*3*
1	4000	5400	9800
2	4300	6200	8700
3	4800	7100	—
4	4900	—	—

The problem can be formulated as a network in which nodes 1 to 5 represent the start of years 1 to 5. Arcs from node l (year 1) can reach only nodes 2,3, and 4 because a car must be in operation between 1 and 3 years. The arcs from the other nodes can be interpreted similarly. The length of each arc equals the replacement cost. The solution of the problem is equivalent to finding the shortest route between nodes 1 and 5.

Figure 2.6. shows the resulting network. Using TORA, the shortest route (shown by the thick path) is 1 -> 3 -> 5. The solution means that a car acquired at the start of year 1 (node 1) must be replaced after 2 years at the start of year 3 (node 3). The replacement car will then be kept in service until the end of year 4. The total cost of this replacement policy is $12,500 (= $5400 + $7100).

Example 5: (Most Reliable Route)

Smart drives daily to work. Having just completed a course in network analysis, Smart is able to determine the shortest route to work. Unfortunately, the selected route is heavily patrolled by police, and with all the fines paid for speeding, the shortest route may not be the best choice. Smart has thus decided to choose a route that maximizes the probability of *not* being stopped by police.

The network in Figure 2.7 shows the possible routes between home and work, and the associated probabilities of not being stopped on each segment. The probability of not being

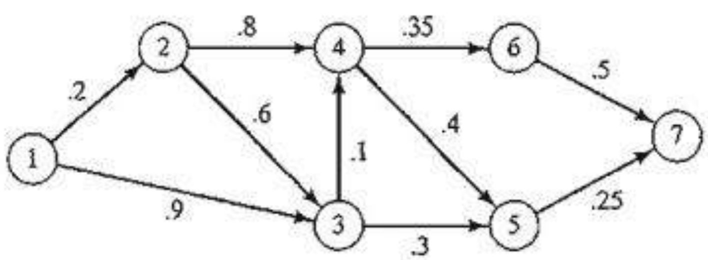


Figure 2.7

Most reliable route network model

stopped on a route is the product of the probabilities associated with its segments. For example, the probability of not receiving a fine on the route 1 -> 3 -> 5 -> 7 is .9 X .3 X .25 = .0675. Smart's objective is to select the route that *maximizes* the probability of not being fined.

The problem can be formulated as a shortest-route model by using a logarithmic transformation that converts the product probability into the sum of the logarithms of probabilities that is, if $p_{lk} = p_1$ X p_2 X ... X p_k is the probability of not being stopped, then log p_{lk} = log p_1 + log p_2 + ... + log p_k.

Mathematically, the maximization of log p_{lk} is equivalent to the maximization of log p_{1k}. Because log $p_{1k} \leq 0$, the maximization of log p_{1k} is equivalent to the minimization of -log p_{1k}. Using this transformation, the individual probabilities pj in Figure 2.7 are replaced with -log p_i for all j in the network, thus yielding the shortest-route network in Figure 2.8.

Using TORA, the shortest route in Figure 2.8 is defined by the nodes 1,3,5, and 7 with a corresponding "length" of 1.1707 (= -log P_{17}). Thus, the maximum probability of not being stopped is p_{17} = .0675 only, not very encouraging news for Smart!

Example 6: (Three-Jug Puzzle)

An 8-gallon jug is filled with fluid. Given two empty 5- and 3-gallon jugs, we want to divide the 8 gallons of fluid into two equal parts using the three jugs. No other measuring devices are allowed. What is the smallest number of transfers (decantation's) needed to achieve this result?

You probably can guess the solution to this puzzle. Nevertheless, the solution process can be systematized by representing the problem as a shortest-route problem.

A node is defined to represent the amount of fluid in the 8-,5-, and 3-gallon jugs, respectively. This means that the network starts with node (8, 0, 0) and terminates with the desired

Most-reliable-route representation as a shortest-route model

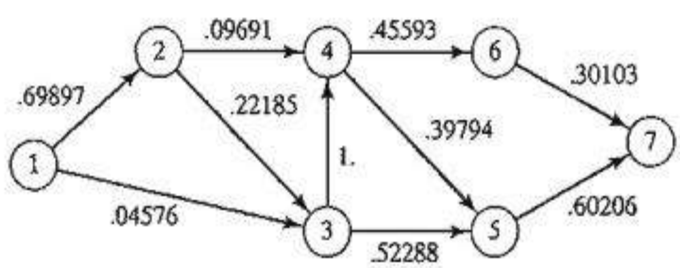


Figure 2.8

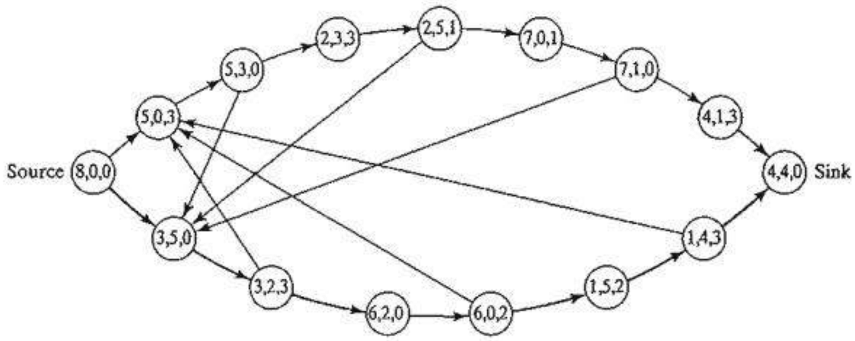


Figure 2.9

Three-Jug Puzzle representation as a shortest route model

solution node (4,4,0). A new node is generated from the current node by decanting fluid from one jug into another.

Figure 2.9 shows different routes that lead from the start node (8,0,0) to the end node (4, 4, 0). The arc between two successive nodes represents a single transfer, and hence can be assumed to have a length of 1 unit. The problem thus reduces to determining the shortest route between node (8,0,0) and node (4,4,0).

The optimal solution, given by the bottom path in Figure 2.9 requires 7 transfers.

2.3.1. Shortest -Route Algorithms

This section presents two algorithms for solving both cyclic (i.e., containing loops) and acyclic networks:

1. Dijkstra's algorithm
2. **Floyd's algorithm**

Dijkstra's algorithm is designed to determine the shortest routes between the source node and every other node in the network. Floyd's algorithm is more general because it allows the determination of the shortest route between *any* two nodes in the network.

Dijkstra's Algorithm. Let u_i be the shortest distance from source node 1 to node *i*, and define d_{ij} (≥ 0) as the length of arc (i, j). Then the algorithm defines the label for an immediately succeeding node *j* as

$$[u_j, i] = [u_i + d_{ij}, i], d_{ij} \geq 0$$

The label for the starting node is [0, --], indicating that the node has no predecessor. Node labels in Dijkstra's algorithm are of two types: *temporary* and *permanent.* A temporary label is modified if a shorter route to a node can be found. If no better route can be found, the status of the temporary label is changed to permanent.

Step 0. Label the source node (node 1) with the *permanent* label [0, —]. Set $i = 1$.

Step i. **(a)** Compute the *temporary* labels $[u_i + d_{ij}, i]$ for each node j that can be reached from node i, *provided j is not permanently labeled.* If node j is already labeled with $[u_j, k]$ through another node k and if $u_i + d_{ij} < u_j$, replace $[u_j, k]$ with $[u_i + d_{ij}, i]$.

(b) If *all* the nodes have *permanent* labels, stop. Otherwise, select the label $[u_r, s]$ having the shortest distance $(= u_r)$ among all the *temporary* labels (break ties arbitrarily). Set $i = r$ and repeat step *i.*

Example 7:

The network in Figure 2.10 gives the permissible routes and their lengths in miles between city 1 (node 1) and four other cities (nodes 2 to 5). Determine the shortest routes between city 1 and each of the remaining four cities.

Iteration 0. Assign the *permanent* label [0, --] to node 1.

Iteration 1. Nodes 2 and 3 can be reached from (the last permanently labeled) node 1. Thus, the list of labeled nodes (temporary and permanent) becomes

Node	Label	Status
1	[0,1]	Permanent
2	[0+100,1]=[100,1]	Temporary
3	[0+30,1]=[30,1]	Temporary

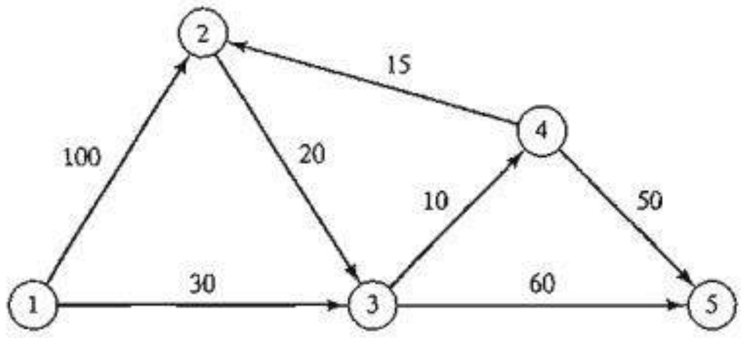


Figure 2.10

Network example for Dijkstra's shortest route algorithm

For the two temporary labels [100, 1] and [30, 1], node 3 yields the smaller distance (u_3 = 30). Thus, the status of node 3 is changed to permanent.

Iteration 2. Nodes 4 and 5 can be reached from node 3, and the list of labeled nodes becomes

Node	Label	Status
1	[0, -]	Permanent
2	[100,1]	Temporary
3	[30,1]	Permanent
4	[30+10,3]=[40,3]	Temporary
5	[30+60,3]=[90,3]	Temporary

The status of the temporary label [40,3] at node 4 is changed to permanent (u_4 = 40).

Iteration 3. Nodes 2 and 5 can be reached from node 4. Thus, the list of labeled nodes is updated as Node 2's temporary label [100, 1] obtained in iteration 1 is changed to [55,4) in iteration 3 to indicate that a shorter route has been found through node 4. Also, in iteration 3, node 5 has two alternative labels with the same distance u_5 = 90.

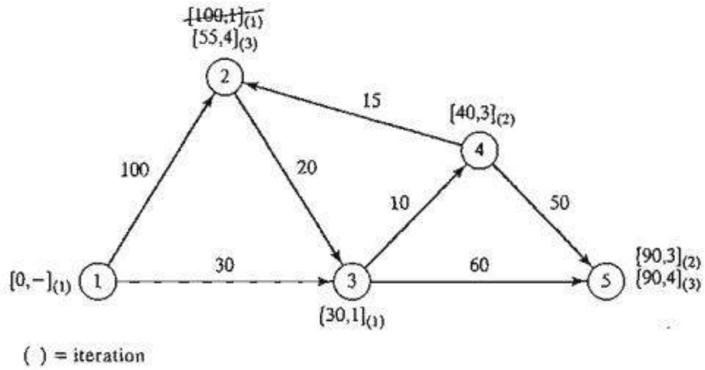


Figure 2.11. Dijkstra's labelling procedure

The list for iteration 3 shows that the label for node 2 is now permanent.

Iteration 4. Only node 3 can be reached from node 2. However, node 3 has a permanent label and cannot be relabeled. The new list of labels remains the same as in iteration 3 except that the label at node 2 is now permanent. This leaves node 5 as the only temporary label Because node 5 does not lead to other nodes, its status is converted to permanent, and the process ends.

The computations of the algorithm can be carried out more easily on the network, as Figure 2.11 demonstrates.

The shortest route between nodes 1 and any other node in the network is determined by starting at the desired destination node and backtracking through the nodes using the information given by the permanent labels. For example, the following sequence determines the shortest route from node 1 to node 2:

(2) -> [55, 4] -> (4) -> [40, 3] -> (3) -> [30,1] -> (1)

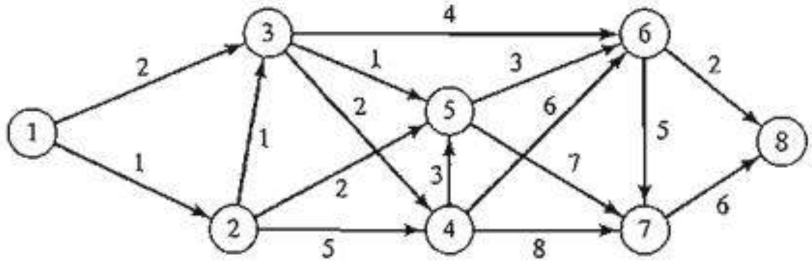
Thus, the desired route is 1 -> 3 -> 4 -> 2 with a total length of 55 miles.

Exercises 1:

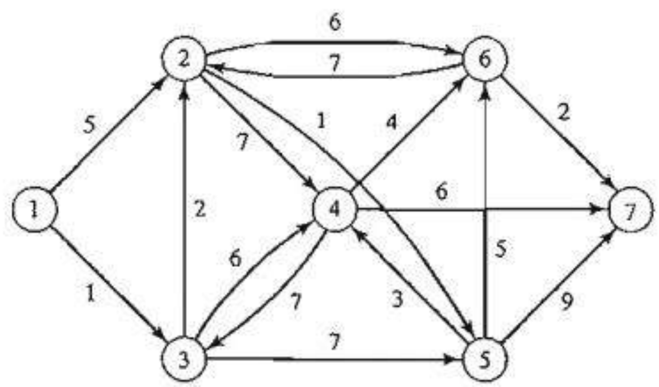
1. The network in Figure gives the distances in miles between pairs of cities 1,2, ., . , and 8. Use Dijkstra's algorithm to find the shortest route between the following cities:

a. Cities 1 and 8

b. Cities 1 and 6



2. Use Dijkstra's algorithm to find the shortest route between node 1 and every other node in the network this figure.



c. Cities 4 and 8

d. Cities 2 and 6

Floyd's Algorithm. Floyd's algorithm is more general than Dijkstra's because it determines the shortest route between *any* two nodes in the network. The algorithm represents an *n-node* network as a square matrix with *n* rows and *n* columns. Entry (~j) of the matrix gives the distance d_{ij} from node *i* to node j, which is finite if *i* is linked directly to j, and infinite otherwise.

The idea of Floyd's algorithm is straightforward. Given three nodes i j, and *k* in Figure 6.19 with the connecting distances shown on the three arcs, it is shorter to reach *j* from *i* passing through *k* if

$$d_{ik} + d_{kj} < d_{ij}$$

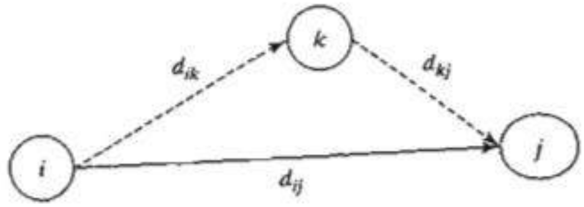


Figure 2.12

Floyd's triple operation

In this case, it is optimal to replace the direct route from $i \to j$ with the indirect route $i \to k \to j$. This triple operation exchange is applied systematically to the network using the following steps:

Step 0. Define the starting distance matrix D_o and node sequence matrix S_o as given below.

$$D_0 = \begin{array}{c|cccccc|} & 1 & 2 & \cdots & j & \cdots & n \\ \hline 1 & - & d_{12} & \cdots & d_{1j} & \cdots & d_{1n} \\ 2 & d_{21} & - & \cdots & d_{2j} & \cdots & d_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ i & d_{i1} & d_{i2} & \cdots & d_{ij} & \cdots & d_{in} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ n & D_{n1} & d_{n2} & \cdots & d_{nj} & \cdots & - \\ \hline \end{array}$$

The diagonal elements are marked with (--) to indicate that they are blocked. Set k = 1.

$$S_0 = \begin{array}{c|cccccc|} & 1 & 2 & \cdots & j & \cdots & n \\ \hline 1 & - & 2 & \cdots & j & \cdots & n \\ 2 & 1 & - & \cdots & j & \cdots & n \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ i & 1 & 2 & \cdots & j & \cdots & n \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ n & 1 & 2 & \cdots & j & \cdots & - \\ \hline \end{array}$$

General step *k*. Define row *k* and column *k* as *pivot row* and *pivot column*. Apply the *triple operation* to each element d_{jj} in D_{k-l}, for all *i* and *j*. If the condition

$d_{ik} + d_{kj} < d_{ij}, (i \neq k, j \neq k and\ i \neq j)$

is satisfied, make the following changes:

a) Create D_k by replacing d_{ij} in D_{k-1} with $d_{jk} + d_{kj}$

b) Create S_k by replacing S_{ij} in S_{k-l} with k. Set $k = k + 1$. If $k = n + 1$, stop; else repeat step k.

Step k of the algorithm can be explained by representing D_{k-1} as shown in Figure 6.20. Here, row k and column k define the current pivot row and column. Row i

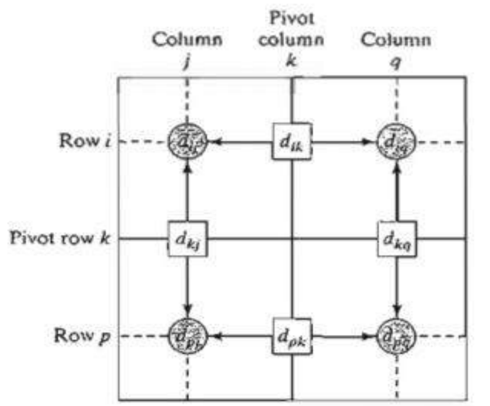


Figure 2.13

Implementation of triple operation in matrix form

represents any of the rows 1, 2, ... , and $k - 1$, and row p represents any of the rows $k + 1$, $k + 2$, ... , and n. Similarly, column j represents any of the columns 1,2,... , and $k - 1$, and column q represents any of the columns $k + 1$, $k + 2$, ... , and n. The *triple operation* can be applied as follows. If the sum of the elements on the pivot row and the pivot column (shown by squares) is smaller than the associated intersection element (shown by a circle), then it is optimal to replace the intersection distance by the sum of the pivot distances.

After n steps, we can determine the shortest route between nodes i and j from the matrices D_n and Sn using the following rules:

a) From D_m d_{ij} gives the shortest distance between nodes i and j.

b) From *Sm* determine the intermediate node $k = s_{ij}$ that yields the route $i \rightarrow k \rightarrow j$. If $s_{ik} = k$ and $s_{kj} = j$, stop; all the intermediate nodes of the route have been found. Otherwise, repeat the procedure between nodes *i* and *k*, and between nodes *k* and *j*.

Example 8:

For the network in Figure 2.14, find the shortest routes between every two nodes. The distances (in miles) are given on the arcs. Arc (3,5) is directional, so that no traffic is allowed from node 5 to node 3. All the other arcs allow two-way traffic.

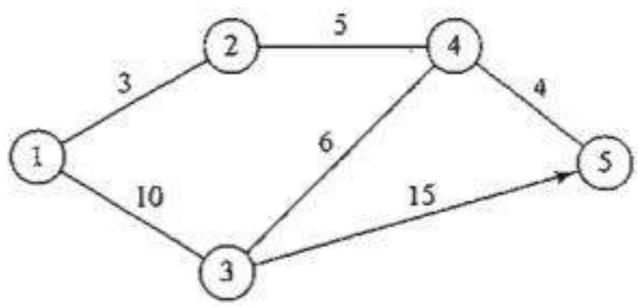


Figure 2.14

Iteration 0: The matrices Do and So give the initial representation of the network. Do is symmetrical, except that *dS3* = 00 because no traffic is allowed from node 5 to node 3.

D_0

	1	2	3	4	5
1	-	3	10	∞	∞
2	3	-	∞	5	∞
3	10	∞	-	6	15
4	∞	5	6	-	4
5	∞	∞	∞	4	-

S_0

	1	2	3	4	5
1	-	2	3	4	5
2	1	-	3	4	5
3	1	2	-	4	5
4	1	2	3	-	5
5	1	2	3	4	-

Iteration 1. Set *k*= 1. The pivot row and column are shown by the lightly shaded first row and first column in the Do-matrix. The darker cells, *d23* and *d32,* are the only ones that can be improved by the *triple operation.* Thus, Dl and SI are obtained from *Do* and So in the following manner:

1. Replace *d23* with *d2k* + *dk3* = 3 + 10 = 13 and set S23 = 1(S23=k)
2. Replace *d32* with *d3k* + *dk2*= 10 + 3 = 13 and set *S32* = 1(S32=k)

D_1

	1	2	3	4	5
1	-	3	10	∞	∞
2	3	-	**13**	5	∞
3	10	**13**	-	6	15
4	∞	5	6	-	4
5	∞	∞	∞	4	-

S_1

	1	2	3	4	5
1	-	2	3	4	5
2	1	-	**1**	4	5
3	1	**1**	-	4	5
4	1	2	3	-	5
5	1	2	3	4	-

Iteration 2: Set ***k*= 2,** as shown by the lightly shaded row and column in D1 . The *triple operation* is applied to the darker cells in D1, and S1 ' The resulting changes are shown in bold in *D2* and *S2*.

1. Replace *d14* with *d1k* + *dk4* =3 + 5 = 8 and set S14 = 2 (S23=k=2)
2. Replace *d41* with *d4k* + *dk1*= 5 + 3 = 8 and set *S32* = 2(S32=k=2)

D_2

	1	2	3	4	5
1	-	3	10	8	∞
2	3	-	13	5	∞
3	10	13	-	6	15
4	8	5	6	-	4
5	∞	∞	∞	4	-

S_2

	1	2	3	4	5
1	-	2	3	2	5
2	1	-	**1**	4	5
3	1	**1**	-	4	5
4	2	2	3	-	5
5	1	2	3	4	-

Iteration3: Set *k* =3, as shown by the shaded row and column in *D2*. The new matrices are given by D3 and *S3*

D_3

	1	2	3	4	5
1	-	3	10	8	25
2	3	-	13	5	28
3	10	13	-	6	15
4	8	5	6	-	4
5	∞	∞	∞	4	-

S_3

	1	2	3	4	5
1	-	2	3	2	3
2	1	-	**1**	4	3
3	1	**1**	-	4	5
4	2	2	3	-	5
5	1	2	3	4	-

Iteration 4: Set *k* = 4, as shown by the shaded row and column in D3. The new matrices are given by *D4* and *S4*

D_4

	1	2	3	4	5
1	-	3	10	8	12
2	3	-	11	5	9
3	10	11	-	6	10
4	8	5	6	-	4
5	12	9	10	4	-

S_4

	1	2	3	4	5
1	-	2	3	2	4
2	1	-	**4**	4	4
3	1	**4**	-	4	4
4	2	2	3	-	5
5	4	4	4	4	-

Iteration 5: Set *k* = 5, as shown by the shaded row and column in *D4*• No further improvements are possible in this iteration.

Iteration 5. Set k = 5, as shown by the shaded row and column in D_4. No further improvements are possible in this iteration.

The final matrices D_4 and S_4 contain all the information needed to determine the shortest route between any two nodes in the network. For example, from D_4, the shortest distance from node 1 to node 5 is d_{15} = 12 miles. To determine the associated route, recall that a segment (i, j) represents a direct link only if s_{ij} = j. Otherwise, i and j are linked through at least one other intermediate node. Because s_{ij} = 4! = 5, the route is initially given as 1 →4 → 5. Now, because s_{14} = 2 != 4, the segment'(I,4) is not a *direct* link, and 1 →4 is replaced with 1 → 2 → 4, and the route 1 → 4 →5 now becomes 1 → 2 → 4 → 5. Next, because s_{12} = 2, s_{24} = 4, and s_{45} = 5, no further "dissecting" is needed, and 1 → 2 → 4 → 5 defines the shortest route.

2.3.2. Linear Programming Formulation of the Shortest-Route Problem

This section provides an LP model for the shortest-route problem. The model is general in the sense that it can be used to find the shortest route between any two nodes in the network. In this regard, it is equivalent to Floyd's algorithm.

Suppose that the shortest-route network includes n nodes and that we desire to determine the shortest route between any two nodes sand t in the network. The LP assumes that one unit of flow enters the network at node s and leaves at node t.

Define

$$x_{ij} = \text{amount of flow in arc } (i, j) = \begin{cases} 1, \text{ if arc } (i, j) \text{ is on the shortest route} \\ 0, \text{ otherwise} \end{cases}$$

$$c_{ij} = \text{length of arc } (i, j)$$

Thus, the objective function of the linear program becomes

$$\text{Minimize } z = \sum_{\substack{\text{all defined} \\ \text{arcs } (i, j)}} c_{ij} x_{ij}$$

The constraints represent the *conservation-of-flow equation* at each node:

$$\text{Total input flow} = \text{Total output flow}$$

Mathematically, this translates for node j to

$$\begin{pmatrix}\text{External input} \\ \text{into node } j\end{pmatrix} + \sum_{\substack{i \\ \text{all defined} \\ \text{arcs } (i, j)}} x_{ij} = \begin{pmatrix}\text{External output} \\ \text{from node } j\end{pmatrix} + \sum_{\substack{k \\ \text{all defined} \\ \text{arcs } (j, k)}} x_{jk}$$

Example 9:

Consider the shortest-route network of Example 6 Suppose that we want to determine the shortest route from node 1 to node 2-that is, s = 1 and t = 2. Figure 2.15 shows how the unit of flow enters at node 1 and leaves at node 2.

We can see from the network that the flow-conservation equation yields

$$
\begin{aligned}
&\text{Node 1:} \quad 1 = x_{12} + x_{13} \\
&\text{Node 2:} \; x_{12} + x_{42} = x_{23} + 1 \\
&\text{Node 3:} \; x_{13} + x_{23} = x_{34} + x_{35} \\
&\text{Node 4:} \quad x_{34} = x_{42} + x_{45} \\
&\text{Node 5:} \; x_{35} + x_{45} = 0
\end{aligned}
$$

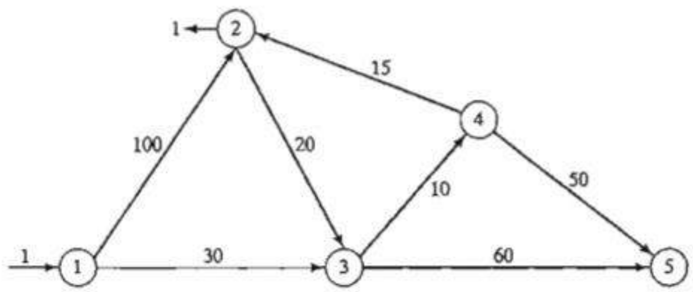


Figure 2.15

Insertion of unit flow to determine shortest route between node s=1 and node t=2

The complete LP can be expressed as

	x_{12}	x_{13}	x_{23}	x_{34}	x_{35}	x_{42}	x_{45}		
Minimize z =	100	30	20	10	60	15	50		
Node 1	1	1						=	1
Node 2	−1		1			−1		=	−1
Node 3		−1	−1	1	1			=	0
Node 4				−1		1	1	=	0
Node 5					−1		−1	=	0

Notice that column x_{ij} has exactly one "1" entry in row i and one "-I" entry in row j, a typical property of a network LP.

The optimal solution $z = 55$, $x_{13} = 1, x_{34} = 1, x_{42} = 1,$

This solution gives the shortest route from node 1 to node 2 as 1 → 3→ 4 → 2, and the associated distance is z = 55 (miles).

2.4. Maximal Flow Model:

Consider a network of pipelines that transports crude oil from oil wells to refineries. Intermediate booster and pumping stations are installed at appropriate design distances to move the crude in the network. Each pipe segment has a finite maximum discharge rate of crude flow (or capacity). A pipe segment may be uni- or bidirectional, depending on its design. Figure 2.16 demonstrates a typical pipeline network. How can we deter-mine the maximum capacity of the network between the wells and the refineries?

The solution of the proposed problem requires equipping the network with a single source and a single sink by using unidirectional infinite capacity arcs as shown by dashed arcs in Figure 2.16.

2.4.1. Enumeration of Cuts

A cut defines a set of arcs which when deleted from the network will cause a total disruption of flow between the source and sink nodes. The cut capacity equals the sum of the capacities of its arcs. Among *all* possible cuts in the network, the cut with the *smallest capacity* gives the maximum flow in the network.

Example 10:

Consider the network in Figure 2.17. The bidirectional capacities are shown on the respective arcs using the convention in Figure 2.17. For example, for arc (3,4), the flow limit is 10 units from 3 to 4 and 5 units from 4 to 3.

Figure 2.20 illustrates three cuts whose capacities are computed in the following table.

Cut	Associated arcs	Capacity
1	(1,2), (1,3), (1,4)	20 + 30 + 10 = 60
2	(1,3), (1,4), (2,3), (2,5)	30 + 10 + 40 + 30 = 110
3	(2,5), (3,5), (4,5)	30 + 20 + 20 = 70

Capacitated network connecting wells and refineries through booster stations

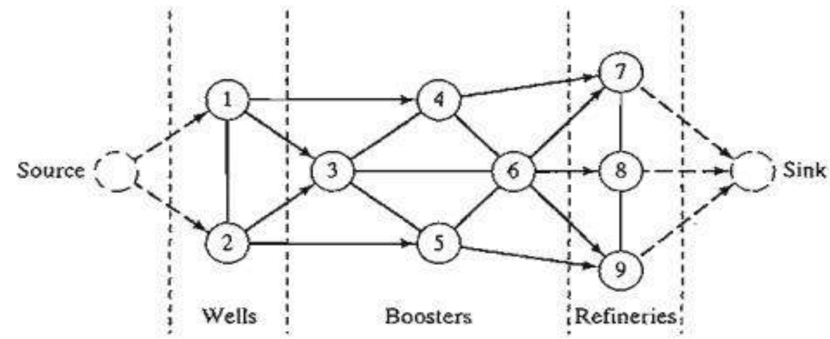


Figure 2.16

Figure 2.17

Arc flows $\overline{c_{ij}}$ from $i \to j$ and $\overline{c_{ij}}$ $from$ $j \to i$

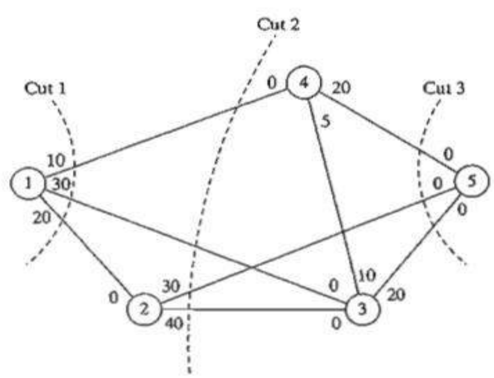


Figure 2.28. Examples of cuts in flow networks

The only information we can glean from the three cuts is that the maximum flow in the net-work cannot exceed 60 units. To determine the maximum flow, it is necessary to enumerate *all* the cuts, a difficult task for the general network. Thus, the need for an efficient algorithm is imperative.

2.4.2. Maximal Flow Algorithm

The maximal flow algorithm is based on finding **breakthrough paths** with net *positive* flow between the source and sink nodes. Each path commits part or all of the capacities of its arcs to the total flow in the network.

Consider arc (i, j) with (initial) capacities $(\overline{C}_{ij}, \overline{C}_{ji})$. As portions of these capacities are committed to the flow in the arc, the **residuals** (or remaining capacities) of the arc are updated. We use the notation (c_{ij}, c_{ji}) to represent these residuals.

For a node *j* that receives flow from node *i*, we attach a label $[a_i, i]$, where *aj* is the flow from node *i* to node j. The steps of the algorithm are thus summarized as follows.

Step 1. For all arcs (i, j), set the residual capacity equal to the initial capacity— that is $(c_{ij}, c_{ji}) = (\overline{C}_{ij}, \overline{C}_{ji})$. Let $a_1 = \infty$ and label source node 1 with $[\infty, -]$. Set $i = 1$, and go to step 2.

Step 2. Determine S_i, the set of unlabeled nodes *j* that can be reached directly from node *i* by arcs with *positive* residuals (that is, $c_{ij} > 0$ for all $j \in S_i$). If $S_i \neq \emptyset$, go to step 3. Otherwise, go to step 4.

Step 3. Determine $k \in S_i$ such that

$$c_{ik} = \max_{j \in S_i}\{c_{ij}\}$$

Set $a_k = c_{ik}$ and label node *k* with $[a_k\ i]$. If $k = n$, the sink node has been labeled, and a *breakthrough path.* is found, go to step 5. Otherwise, set $i = k$, and go to step 2.

Step 4. (Backtracking). If $i = 1$, no breakthrough is possible; go to step 6. Otherwise, let *r* be the node that has been labeled *immediately* before current node *i* and remove *i* from the set of nodes adjacent to *r*. Set $i = r$, and go to step 2.

Step 5. (Determination of residuals). Let $N_p = (1,\ k_1\ k_2,\ \ldots\ , n)$ define the nodes of the *path* breakthrough path from source node 1 to sink node *n*. Then the maximum flow along the path is computed as

$$f_p = \min\{a_1, a_{k_1}, a_{k_2}, \ldots, a_n\}$$

The residual capacity of each arc along the breakthrough path is *decreased* by f_p in the direction of the flow and *increased* by f_p in the reverse direction-that is, for nodes *i* and *j* on the path, the residual flow is changed from the current $(c_{ij}, c_{ij})$to

(a) $(c_{ij} - f_p, c_{ji} + f_p)$ if the flow is from i to j

(b) $(c_{ij} + f_p, c_{ji} - f_p)$ if the flow is from j to i

Reinstate any nodes that were removed in step 4. Set *i* = 1, and return to step 2 to attempt a new breakthrough path.

Step 6. (Solution).

(a) Given that *m* breakthrough paths have been determined, the maximal flow in the network is

$$F = f_1 + f_2 + \cdots + f_m$$

(b) Using the *initial* and *final* residuals of arc (i, j), $(\overline{C}_{ij}, \overline{C}_{ji})$ and (c_{ij}, c_{ji}), respectively, the optimal flow in arc (i, j) is computed as follows: Let $(\alpha, \beta) = (\overline{C}_{ij} - c_{ij}, \overline{C}_{ji} - c_{ji})$. If $\alpha > 0$, the optimal flow from i to j is α. Otherwise, if $\beta > 0$, the optimal flow from j to i is β. (It is impossible to have both α and β positive.)

The backtracking process of step 4 is invoked when the algorithm becomes "dead-ended" at an intermediate node. The flow adjustment in step 5 can be explained via the simple flow network in Figure 2.19 Network (a) gives the first breakthrough path N l = {I, 2, 3, 4} with its maximum flow It = 5. Thus, the residuals of each of arcs (1, 2), (2,3), and (3,4) are changed from (5,0) to (0,5), per step 5. Network (b) now gives the second breakthrough path N2 = {I, 3, 2, 4} with f2 = 5. After making the necessary flow adjustments, we get network (c), where no further breakthroughs are possible. What happened in the transition from (b) to (c) is nothing but a cancellation of a previously committed flow in the direction 2 →3. The algorithm is able to "remember" that

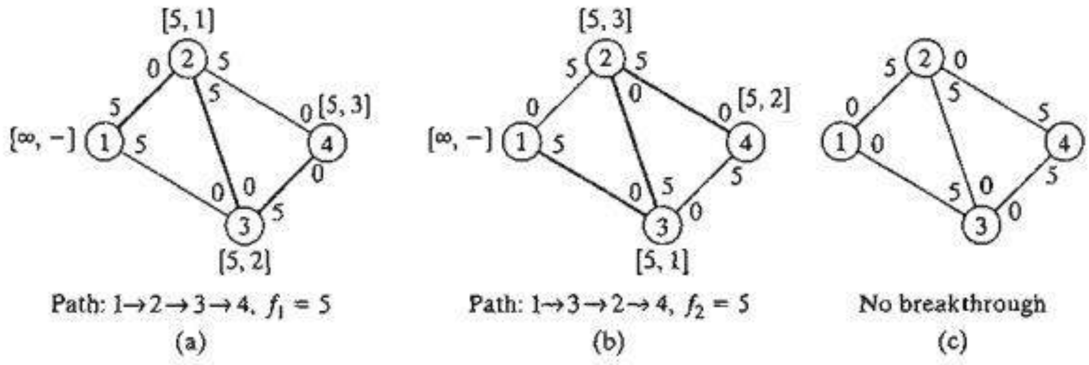


Figure 2.19

a flow from 2 to 3 has been committed previously only because we have increased the capacity in the reverse direction from 0 to 5 (per step 5).

Example 11:

Determine the maximal flow in the network of Example 10 (Figure 2.18). Figure 2.20provides a graphical summary of the iterations of the algorithm. You will find it helpful to compare the description of the iterations with the graphical summary.

Iteration 1. Set the initial residuals (c_{ij}, c_{ji}) equal to the initial capacities $(\overline{C}_{ij}, \overline{C}_{ji})$.

Step 1. Set $a_1 = \infty$ and label node 1 with $[\infty, —]$. Set $i = 1$.

Step 2. $S_1 = \{2, 3, 4\}$ $(\neq \emptyset)$.

Step 3. $k = 3$, because $c_{13} = \max\{c_{12}, c_{13}, c_{14}\} = \max\{20, 30, 10\} = 30$. Set $a_3 = c_{13} = 30$, and label node 3 with [30,1]. Set $i = 3$, and repeat step 2.

Step 2. $S_3 = (4, 5)$.

Step 3. $k = 5$ and $a_5 = c_{35} = \max\{10, 20\} = 20$. Label node 5 with [20, 3]. Breakthrough is achieved. Go to step 5.

Step 5. The breakthrough path is determined from the labels starting at node 5 and moving backward to node 1—that is, $(5) \rightarrow [20, 3] \rightarrow (3) \rightarrow [30, 1] \rightarrow (1)$. Thus, $N_1 = \{1, 3, 5\}$ and $f_1 = \min\{a_1, a_3, a_5\} = \{\infty, 30, 20\} = 20$. The residual capacities along path N_1 are

$$(c_{13}, c_{31}) = (30 - 20, 0 + 20) = (10, 20)$$

$$(c_{35}, c_{53}) = (20 - 20, 0 + 20) = (0, 20)$$

Iteration 2

Step 1. Set $a_1 = \infty$, and label node 1 with $[\infty, —]$. Set $i = 1$.

Step 2. $S_1 = \{2, 3, 4\}$.

Step 3. $k = 2$ and $a_2 = c_{12} = \max\{20, 10, 10\} = 20$. Set $i = 2$, and repeat step 2.

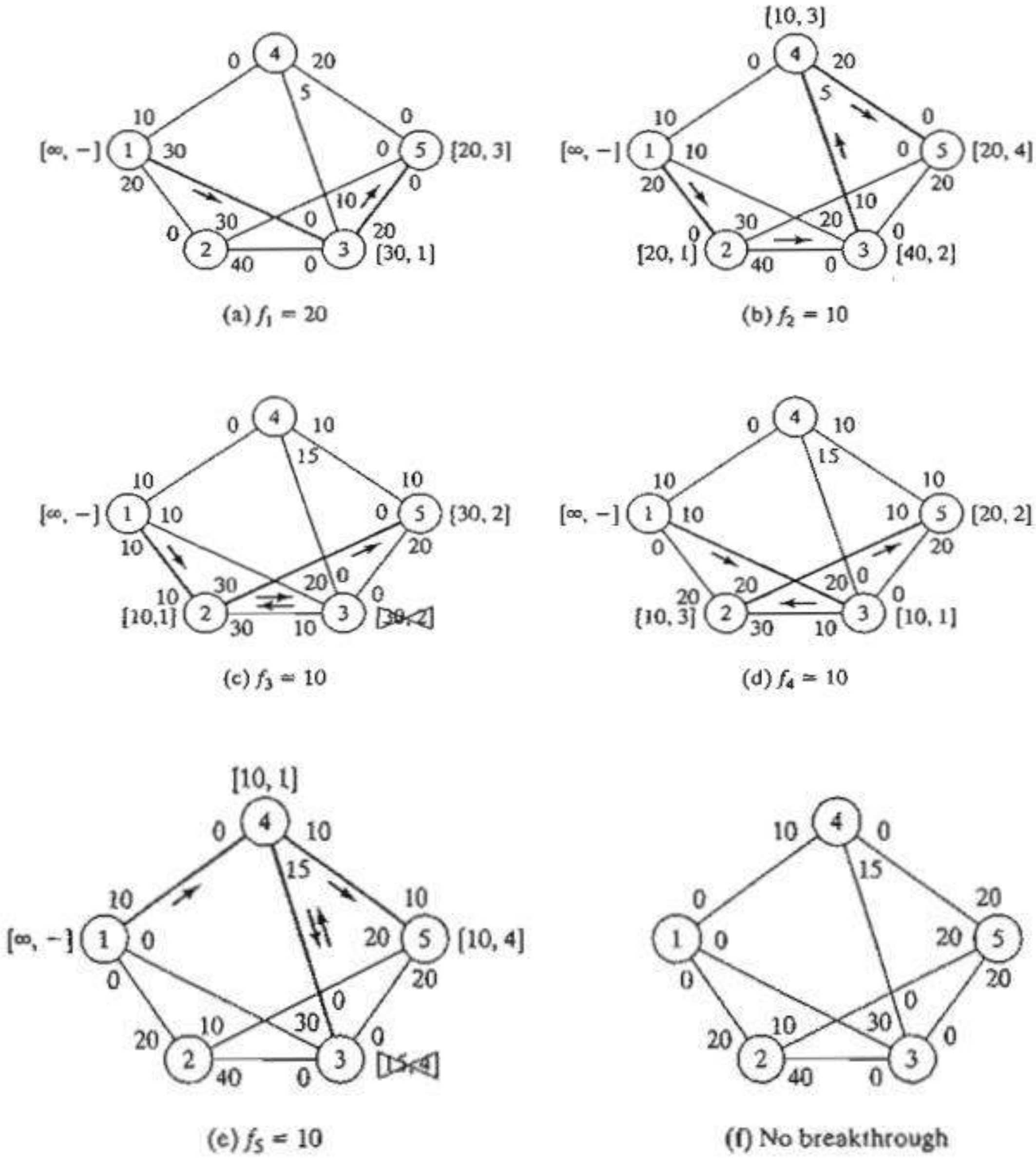


Figure 2.20

Iterations of the maximum flow algorithm of Example 11

Step 2. $S_2 = \{3, 5\}$.

Step 3. $k = 3$ and $a_3 = c_{23} = 40$. Label node 3 with [40, 2]. Set $i = 3$, and repeat step 2.

Step 2. $S_3 = \{4\}$ (note that $c_{35} = 0$—hence, node 5 cannot be included in S_3).

Step 3. $k = 4$ and $a_4 = c_{34} = 10$. Label node 4 with [10, 3]. Set $i = 4$, and repeat step 2.

Step 2. $S_4 = \{5\}$ (note that nodes 1 and 3 are already labeled—hence, they cannot be included in S_4).

Step 3. $k = 5$ and $a_5 = c_{45} = 20$. Label node 5 with [20, 4]. Breakthrough has been achieved. Go to step 5.

Step 5. $N_2 = \{1, 2, 3, 4, 5\}$ and $f_2 = \min\{\infty, 20, 40, 10, 20\} = 10$. The residuals along the path of N_2 are

$$(c_{12}, c_{21}) = (20 - 10, 0 + 10) = (10, 10)$$

$$(c_{23}, c_{32}) = (40 - 10, 0 + 10) = (30, 10)$$

$$(c_{34}, c_{43}) = (10 - 10, 5 + 10) = (0, 15)$$

$$(c_{45}, c_{54}) = (20 - 10, 0 + 10) = (10, 10)$$

Iteration 3

Step 1. Set $a_1 = \infty$ and label node 1 with [∞, —]. Set $i = 1$.

Step 2. $S_1 = \{2, 3, 4\}$.

Step 3. $k = 2$ and $a_2 = c_{12} = \max\{10, 10, 10\} = 10$. (Though ties are broken arbitrarily, TORA always selects the tied node with the smallest index. We will use this convention throughout the example.) Label node 2 with [10,1]. Set $i = 2$, and repeat step 2.

Step 2. $S_2 = \{3, 5\}$.

Step 3. $k = 3$ and $a_3 = c_{23} = 30$. Label node 3 with [30, 2]. Set $i = 3$, and repeat step 2.

Step 2. $S_3 = \emptyset$ (because $c_{34} = c_{35} = 0$). Go to step 4 to backtrack.

Step 3. *Backtracking.* The label [30, 2] at node 3 gives the immediately preceding node $r = 2$. Remove node 3 from further consideration *in this iteration* by crossing it out. Set $i = r = 2$, and repeat step 2.

Step 2. $S_2 = \{5\}$ (note that node 3 has been removed in the backtracking step).

Step 3. $k = 5$ and $a_5 = c_{25} = 30$. Label node 5 with $[30, 2]$. Breakthrough has been achieved; go to step 5.

Step 5. $N_3 = \{1, 2, 5\}$ and $c_5 = \min\{\infty, 10, 30\} = 10$. The residuals along the path of N_3 are

$$(c_{12}, c_{21}) = (10 - 10, 10 + 10) = (0, 20)$$

$$(c_{25}, c_{52}) = (30 - 10, 0 + 10) = (20, 10)$$

Iteration 4. This iteration yields $N_4 = \{1, 3, 2, 5\}$ with $f_4 = 10$ (verify!).

Iteration 5. This iteration yields $N_5 = \{1, 4, 5\}$ with $f_5 = 10$ (verify!).

Iteration 6. All the arcs out of node 1 have zero residuals. Hence, no further breakthroughs are possible. We turn to step 6 to determine the solution.

Step 6. Maximal flow in the network is $F = f_1 + f_2 + \cdots + f_5 = 20 + 10 + 10 + 10 + 10 = 60$ units. The flow in the different arcs is computed by subtracting the last residuals (c_{ij}, c_{ji}) in iterations 6 from the initial capacities $(\overline{C}_{ij}, \overline{C}_{ji})$, as the following table shows.

Arc	$(\overline{C}_{ij}, \overline{C}_{ji}) - (c_{ij}, c_{ji})_6$	Flow amount	Direction
(1,2)	(20, 0) − (0, 20) = (20, −20)	20	1 → 2
(1,3)	(30, 0) − (0, 30) = (30, −30)	30	1 → 3
(1,4)	(10, 0) − (0, 10) = (10, −10)	10	1 → 4
(2,3)	(40, 0) − (40, 0) = (0, 0)	0	—
(2,5)	(30, 0) − (10, 20) = (20, −20)	20	2 → 5
(3,4)	(10, 5) − (0, 15) = (10, −10)	10	3 → 4
(3,5)	(20, 0) − (0, 20) = (20, −20)	20	3 → 5
(4,5)	(20, 0) − (0, 20) = (20, −20)	20	4 → 5

2.4.3. Linear Programming Formulation of Maximal Flow Mode

Define x_{ij} as the amount of flow in arc (i, j) with capacity c_{ij} . The objective is to determine x_{ij} for all i and j that will maximize the flow between start node s and terminal node t subject to flow restrictions (input flow = output flow) at all but nodes s and t.

Example 12:

In the maximal flow model of Figure 2.18 (Example 11), s = 1 and t = 5. The following table summarizes the associated LP with two different, but equivalent, objective functions depending on whether we maximize the output from start node 1 (= z.) or the input to terminal node

	x_{12}	x_{13}	x_{14}	x_{23}	x_{25}	x_{34}	x_{35}	x_{43}	x_{45}	
Maximize z_1 =	1	1	1							
Maximize z_2 =					1		1		1	
Node 2	1			−1	−1					= 0
Node 3		1		1		−1	−1	1		= 0
Node 4			1			1		−1	−1	= 0
Capacity	20	30	10	40	30	10	20	5	20	

The optimal solution using either objective function is

$$x_{12} = 20, x_{13} = 30, x_{14} = 10, x_{25} = 20, x_{34} = 10, x_{35} = 20, x_{45} = 20$$

The associated maximum flow is $z_1 = z_2 = 60$.

2.5.CPM and PERT

CPM (Critical Path Method) and PERT (Program Evaluation and Review Technique) are network-based methods designed to assist in the planning, scheduling, and control of projects. A project is defined as a collection of interrelated activities with each activity

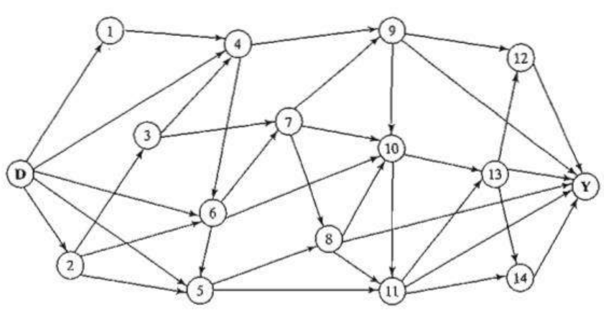


Figure 2.21

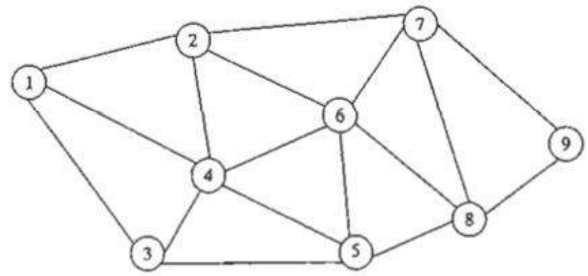


Figure 2.22

consuming time and resources. The objective of CPM and PERT is to provide analytic means for scheduling the activities. Figure 2.23 summarizes the steps of the techniques. First, we define the activities of the project, their precedence relationships, and their time requirements. Next, the precedence relationships among the activities are represented by a network. The third step involves specific computations to develop the time schedule for the project. During the actual execution of the project things may not proceed as planned, as some of the activities may be expedited or delayed. When this hap-pens, the schedule must be revised to reflect the realities on the ground. This is the reason for including a feedback loop between the time schedule phase and the net-work phase, as shown in Figure 2.23.

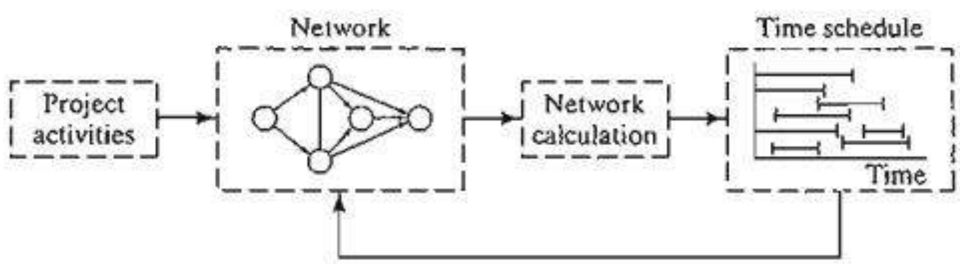


Figure 2.23

The two techniques, CPM and PERT, which were developed independently, differ in that CPM assumes deterministic activity durations and PERT assumes probabilistic durations. This presentation will start with CPM and then proceed with the details of PERT.

2.5.1. Network Representation:

Each activity of the project is represented by an arc pointing in the direction of progress in the project. The nodes of the network establish the precedence relation-ships among the different activities.

Three rules are available for constructing the network.

Rule 1. *Each activity is represented by one, and only one, arc.*

Rule 2. *Each activity must be identified by two distinct end nodes.*

Figure 2.24 shows how a dummy activity can be used to represent two concurrent activities, *A* and *B.* By definition, a dummy activity, which normally is depicted by a

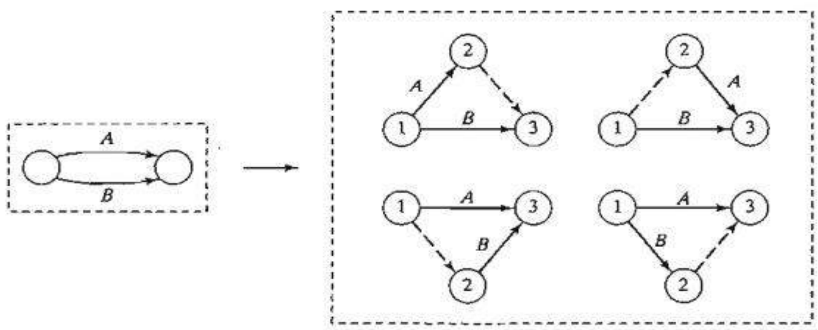


Figure 2.24

Use of dummy activity to produce unique representation of concurrent activities

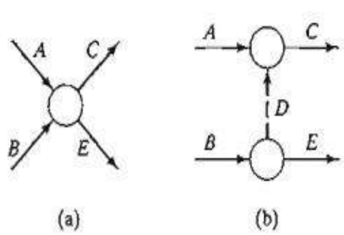


Figure 2.25

dashed are, consumes no time or resources. Inserting a dummy activity in one of the four ways shown in Figure 2.24, we maintain the concurrence of *A* and *B*, and provide unique end nodes for the two activities (to satisfy rule 2).

Rule 3. *To maintain the correct precedence relationships, the following questions must be answered as each activity is added to the network:*

i. *What activities must immediately precede the current activity?*

ii. *What activities must follow the current activity?*

iii. *What activities must occur concurrently with the current activity?*

The answers to these questions may require the use of dummy activities to ensure correct precedences among the activities. For example, consider the following segment of a project:

a) Activity C starts immediately after *A* and *B* have been completed.

b) Activity *E* starts only after *B* has been completed.

Part (a) of Figure 2.25 shows the incorrect representation of the precedence relation-ship because it requires both *A* and *B* to be completed before *E* can start. In part (b), the use of a dummy activity rectifies the situation.

Example 13:

A publisher has a contract with an author to publish a textbook. The (simplified) activities associated with the production of the textbook are given below. The author is required to submit to the publisher a hard copy and a computer file of the manuscript. Develop the associated net-work for the project.

Activity	Predecessor(s)	Duration (weeks)
A: Manuscript proofreading by editor	—	3
B: Sample pages preparation	—	2
C: Book cover design	—	4
D: Artwork preparation	—	3
E: Author's approval of edited manuscript and sample pages	*A, B*	2
F: Book formatting	*E*	4
G: Author's review of formatted pages	*F*	2
H: Author's review of artwork	*D*	1
I: Production of printing plates	*G, H*	2
J: Book production and binding	*C, I*	4

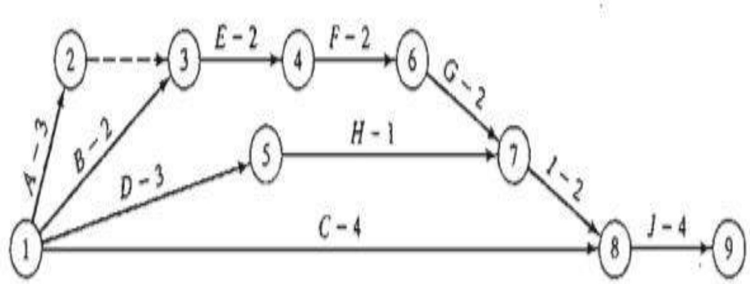


Figure 2.26

Figure 2.26 provides the network describing the precedence relationships among the different activities. Dummy activity (2,3) produces unique end nodes for concurrent activities *A* and *B.* It is convenient to number the nodes in ascending order in the direction of progress in the project.

2.5.2. Critical Path (CPM) Computations:

The end result in CPM is the construction of the time schedule for the project (see Figure 2.27). To achieve this objective conveniently, we carry out special computations that produce the following information:

a. Total duration needed to complete the project.

b. Classification of the activities of the project as *critical* and *noncritical.*

An activity is said to be critical if there is no "leeway" in determining its start and finish times. A noncritical activity allows some scheduling slack, so that the start time of the activity can be advanced or delayed within limits without affecting the completion date of the entire project.

To carry out the necessary computations, we define an event as a point in time at which activities are terminated and others are started. In terms of the network, an event corresponds to a node. Define

$\square_j$ = Earliest occurrence time of event j

Δ_j = Latest occurrence time of event j

D_{ij} = Duration of activity (i, j)

The definitions of the *earliest* and *latest* occurrences of event j are specified relative to the start and completion dates of the entire project.

The critical path calculations involve two passes: The forward pass determines the *earliest* occurrence times of the events, and the backward pass calculates their *latest* occurrence times.

Forward Pass (Earliest Occurrence limes, $\square$). The computations start at node 1 and advance recursively to end node *n*.

Initial Step. Set $\square_1 = 0$ to indicate that the project starts at time O.

General Step j. Given that nodes *p, q,* .... , and *v* are linked *directly* to node j by incoming activities *(p, j), (q, j),* , and *(v,* j) and that the earliest occurrence times of events (nodes) *p, q,* ... , and *v* have already been computed, then the earliest occurrence time of event j is computed as

$$\Delta_j = \min\{\Delta_p - D_{jp}, \Delta_q - D_{jq}, \ldots, \Delta_v - D_{jv}\}$$

The backward pass is complete when Δ_1 at node 1 is computed. At this point, $\Delta_1 = \square_1 (= 0)$.

Based on the preceding calculations, an activity (i, j) will be *critical* if it satisfies three conditions.

1. $\Delta_i = \square_i$
2. $\Delta_j = \square_j$
3. $\Delta_j - \Delta_i = \square_j - \square_i = D_{ij}$

The three conditions state that the earliest and latest occurrence times of end nodes *i* and *j* are equal and the duration D_{ij} fits "tightly" in the specified time span. An activity that does not satisfy all three conditions is thus *noncritical.*

By definition, the critical activities of a network must constitute an uninterrupted path that spans the entire network from start to finish.

Example 14:

Determine the critical path for the project network in Figure 2.27 All the durations are in days.

Forward Pass

Node 1. Set $\square_1 = 0$

Node 2. $\square_2 = \square_1 + D_{12} = 0 + 5 = 5$

Node 3. $\square_3 = \max\{\square_1 + D_{13}, \square_2 + D_{23}\} = \max\{0 + 6, 5 + 3\} = 8$

Node 4. $\square_4 = \square_2 + D_{24} = 5 + 8 = 13$

Node 5. $\square_5 = \max\{\square_3 + D_{35}, \square_4 + D_{45}\} = \max\{8 + 2, 13 + 0\} = 13$

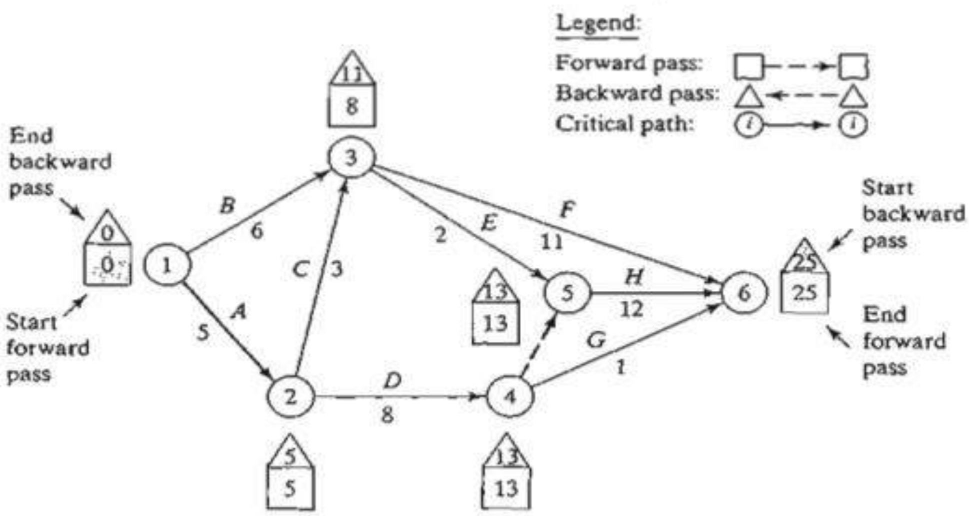


Figure 2.27

Forward and backward pass calculations for the project of Example 14

$$\textbf{Node 6.} \quad \square_6 = \max\{\square_3 + D_{36}, \square_4 + D_{46}, \square_5 + D_{56}\}$$
$$= \max\{8 + 11, 13 + 1, 13 + 12\} = 25$$

The computations show that the project can be completed in 25 days.

Backward Pass

Node 6. Set $\Delta_6 = \square_6 = 25$

Node 5. $\Delta_5 = \Delta_6 - D_{56} = 25 - 12 = 13$

Node 4. $\Delta_4 = \min\{\Delta_6 - D_{46}, \Delta_5 - D_{45}\} = \min\{25 - 1, 13 - 0\} = 13$

Node 3. $\Delta_3 = \min\{\Delta_6 - D_{36}, \Delta_5 - D_{35}\} = \min\{25 - 11, 13 - 2\} = 11$

Node 2. $\Delta_2 = \min\{\Delta_4 - D_{24}, \Delta_3 - D_{23}\} = \min\{13 - 8, 11 - 3\} = 5$

Node 1. $\Delta_1 = \min\{\Delta_3 - D_{13}, \Delta_2 - D_2\} = \min\{11 - 6, 5 - 5\} = 0$

Correct computations will always end with $\Delta_1 = 0$.

The forward and backward pass computations can be made directly on the network as shown in Figure 2.27 Applying the rules for determining the critical activities, the critical path is 1 -> 2 -> 4 -> 5 -> 6, which, as should be expected, spans the network from start (node 1) to finish (node 6). The sum of the durations of the critical activities [(1,2), (2, 4), (4, 5), and (5, 6)J equals the duration of the project (= 25 days). Observe that activity (4,6) satisfies the first two conditions for a critical activity

$(\Delta_4 = \square_4 = 13$ and $\Delta_5 = \square_5 = 25)$ but not the third $(\square_6 - \square_4 \neq D_{46})$.

Hence, the activity is noncritical.

2.5.3. Construction of the Time Schedule:

This section shows how the information obtained from the calculations in Previous Section can be used to develop the time schedule. We recognize that for an activity (i, j), $\square_i$ represents the earliest start time, and Δ_j represents the latest completion (fine. This means that the interval ($\square_i$, Δ_j) delineates the (maximum) span during which activity (i, j) may be scheduled without delaying the entire project.

Construction of Preliminary Schedule. The method for constructing a preliminary schedule is illustrated by an example.

Example 15:

Determine the time schedule for the project of Example 14 (Figure 2.27).

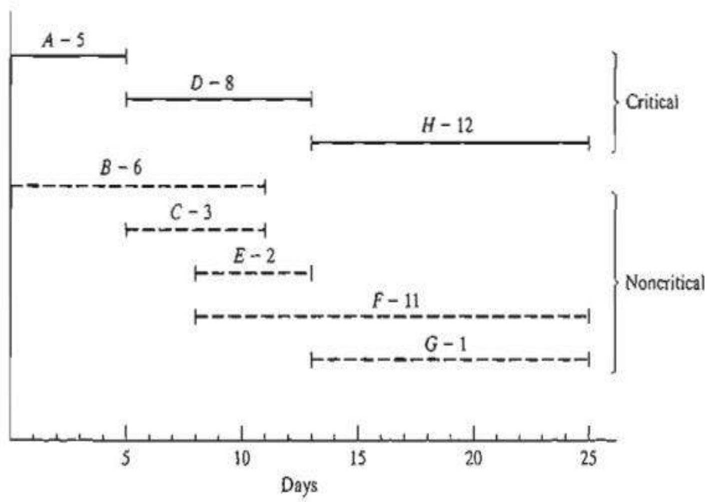


Figure 2.28

We can get a preliminary time schedule for the different activities of the project by delineating their respective time spans as shown in Figure 2.28. Two observations are in order.

a. The critical activities (shown by solid lines) must be stacked one right after the other to ensure that the project is completed within its specified 25-day duration.

b. The noncritical activities (shown by dashed lines) have time spans that are larger than their respective durations, thus allowing slack (or "leeway") in scheduling them within their allotted time intervals.

How should we schedule the noncritical activities within their respective spans? Normally, it is preferable to start each noncritical activity as early as possible. In this manner, slack periods will remain opportunely available at the end of the allotted span where they can be used to absorb unexpected delays in the execution of the activity. It may be necessary, however, to delay the start of a noncritical activity past its earliest start time. For example, in Figure 2.30, suppose that each of the noncritical activities E and *F* requires the use of a bulldozer, and that only one is available. Scheduling both *E* and *F* as early as possible requires two bulldozers between times 8 and 10. We can remove the overlap by starting *E* at time 8 and pushing the start time of *F* to somewhere between times 10 and 14.

If all the noncritical activities can be scheduled as early as possible, the resulting schedule automatically is feasible. Otherwise, some precedence relationships may be violated if noncritical activities are delayed past their earliest time. Take for example activities C and E in Figure 2.28. In the project network (Figure 2.27), though C must be completed before *E*, the spans of C and *E* in Figure 2.28 allow us to schedule C between times 6 and 9, and *E* between times 8 and 10, which violates the requirement that C precede *E*. The need for a "red flag" that automatically reveals schedule conflict is thus evident. Such information is provided by computing the *floats* for the noncritical activities.

Determination of the Floats. Floats are the slack times available within the allotted span of the noncritical activity. The most common are the total float and the free float.

Figure 2.29 gives a convenient summary for computing the total float (TF_{ij}) and the free float (FF_{ij}) for an activity (i, j). The total float is the excess of the time span defined from the *earliest* occurrence of event *i* to the *latest* occurrence of event *j* over the duration of (i, j)-that is, $TF_{ij} = \Delta_j - \square_i - D_{ij}$

The free float is the excess of the time span defined from the *earliest* occurrence of event *i* to the *earliest* occurrence of event *j* over the duration of (i, j)-that is, $FF_{ij} = \square_j - \square_i - D_{ij}$

By definition, $FF_{ij} \leq TF_{ij}$

Red-Flagging Rule. For a noncritical activity (i, j)

(a) If $FF_{ij} = TF_{ij}$, then the activity can be scheduled anywhere within its $(\square_j, \Delta_j)$ span without causing schedule conflict.

(b) If $FF_{ij} < TF_{ij}$, then the start of the activity can be delayed by at most FF_{ij} relative to its earliest start time $(\square_i)$ without causing schedule conflict. Any delay larger than FF_{ij} (but not more than TF_{ij}) must be coupled with an equal delay relative to $\square_j$ in the start time of all the activities leaving node j.

The implication of the rule is that a noncritical activity (i, j) will be red-flagged if its $FF_{ij} < TF_{ij}$. This red flag is important only if we decide to delay the start of the activity past its *earliest start time, $\square_i$, in which case we must pay attention to the start times of the activities leaving node j to avoid schedule conflicts.*

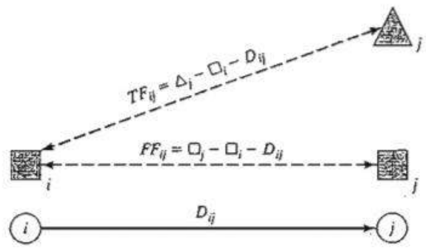


Figure 2.29

Example 16:

Compute the floats for the noncritical activities of the network in Example 14, and discuss their use in finalizing a schedule for the project.

The following table summarizes the computations of the total and free floats. It is more convenient to do the calculations directly on the network using the procedure in Figure 2.29.

Noncritical activity	Duration	Total float (*TF*)	Free float (*FF*)
B(1,3)	6	11 − 0 − 6 = 5	8 − 0 − 6 = 2
C(2,3)	3	11 − 5 − 3 = 3	8 − 5 − 3 = 0
E(3,5)	2	13 − 8 − 2 = 3	13 − 8 − 2 = 3
F(3,6)	11	25 − 8 − 11 = 6	25 − 8 − 11 = 6
G(4,6)	1	25 − 13 − 1 = 11	25 − 13 − 1 = 11

The computations red-flag activities Band C because their $FF < TF$. The remaining activities *(E*, *F*, and G) have $FF = T\ F$, and hence may be scheduled anywhere between their earliest start and latest completion times.

To investigate the significance of the red-flagged activities, consider activity *B*. Because its *T F* = 5 days, this activity can start as early as time 0 or as late as time 5 (see Figure 6.45). However, because its *F F* = 2 days, starting *B* anywhere between time 0 and time 2 will have no effect on the succeeding activities *E* and *F*. If, however, activity *B* must start at time 2 + *d* (≤ 5), then the start times of the immediately succeeding activities *E* and *F* must be pushed forward past their earliest start time (= 8) by at least *d*. In this manner, the precedence relationship between *B* and its successors *E* and *F* is preserved.

Turning to red-flagged activity C, we note that its *F F* = O. This means that *any* delay in starting C past its earliest start time (= 5) must be coupled with at least an equal delay in the start of its successor activities *E* and *F*.

2.5.4. Linear Programming Formulation of CPM:

A CPM problem can thought of as the opposite of the shortest-route problem, in the sense that we are interested in finding the *Longest* route of a unit flow entering at the start node and terminating at the finish node. We can thus apply the shortest route LP formulation in Section 2.3.3 to CPM in the following manner. Define

x_{ij} = Amount of flow in activity *(i,* j), for all defined *i* and *j*

D_{ij} = Duration of activity *(i, j)*, for all defined *i* and *j*

Thus, the objective function of the linear program becomes

$$\text{Maximize } z = \sum_{\substack{\text{all defined} \\ \text{activities } (i,\, j)}} D_{ij} x_{ij}$$

(Compare with the shortest route LP formulation in Section 6.3.3 where the objective function is minimized.) For each node, there is one constraint that represents the conservation of flow:

Total input flow = Total output flow

All the variables, *Xij*, are nonnegative.

Example 17:

The· LP formulation of the project of Example 14 (Figure 2.29) is given below. Note that nodes 1 and 6 are the start and finish nodes, respectively.

	A	B	C	D	E	F	Dummy	G	H	
	x_{12}	x_{13}	x_{23}	x_{24}	x_{35}	x_{36}	x_{45}	x_{46}	x_{56}	
Maximize z =	6	6	3	8	2	11	0	1	12	
Node 1	−1	−1								= −1
Node 2	1		−1	−1						= 0
Node 3		1	1		−1	−1				= 0
Node 4				1			−1	−1		= 0
Node 5					1		1		−1	= 0
Node 6						1		1	1	= 1

The optimum solution is

$$z = 25, x_{12}(A) = 1, x_{24}(D) = 1, x_{45}(\text{Dummy}) = 1, x_{56}(H) = 1, \text{all others} = 0$$

The solution defines the critical path as *A* --> *D* --> Dummy ---> *H*, and the duration of the project is 25 days. The LP solution is not complete, because it determines the critical path, but does not provide the data needed to construct the CPM chart. We have seen in Figure 6.48, however, that AMPL can be used to provide all the needed information without the LP optimization.

PERT Networks

PERT differs from CPM in that it bases the duration of an activity on three estimates:

a. Optimistic time, *a*, which occurs when execution goes extremely well

b. Most likely time, *m*, which occurs when execution is done under normal conditions.

c. Pessimistic time, *b*, which occurs when execution goes extremely poorly.

The range *(a, b)* encloses all possible estimates of the duration of an activity. The estimate *m* lies somewhere in the range *(a, b)*. Based on the estimates, the average duration time, Bar(*D)*, and variance, *v*, are approximated as:

$$\bar{D} = \frac{a + 4m + b}{6}$$

$$v = \left(\frac{b - a}{6}\right)^2$$

CPM calculations given in Sections 2.5.2 and 2.5.3 may be applied directly, with Bar(D) replacing the single estimate D.

It is possible now to estimate the probability that a node *j* in the network will occur by a prespecified scheduled time, s_j. Let ej be the earliest occurrence time of node j. Because the durations of the activities leading from the start node to node *j* are random variables, *ej* also must be a random variable. Assuming that all the activities in the network are statistically independent, we can determine the mean, $E\{e_j\}$, and variance, var$\{e_j\}$, in the following manner. If there is only one path from the start node to node *j*, then the mean is the sum of expected durations, Bar(*D)*, for all the activities along this path and the variance is the sum of the variances, *v*, of the same activities. On the other hand, if more than one path leads to node *j*, then it is necessary first to determine the statistical distribution of the duration of the longest path. This problem is rather difficult because it is equivalent to determining the distribution of the maximum of two or more random variables. A simplifying assumption thus calls for computing the mean and variance, $E\{e_j\}$ and var$\{e_j\}$, as those of the path to node *j* that has the largest sum of *expected* activity durations. If two or more paths have the same mean, the one with the largest variance is selected because it reflects the most uncertainty and, hence, leads to a more conservative estimate of probabilities.

Once the mean and variance of the path to node *j*, *E{ej}* and var{*ej*}, have been computed, the probability that node *j* will be realized by a present time s_j is calculated using the following formula:

$$P\{e_j \le S_j\} = P\left\{\frac{e_j - E\{e_j\}}{\sqrt{\text{var}\{e_j\}}} \le \frac{S_j - E\{e_j\}}{\sqrt{\text{var}\{e_j\}}}\right\} = P\{z \le K_j\}$$

where

$$z = \text{Standard normal random variable}$$

$$K_j = \frac{S_j - E\{e_j\}}{\sqrt{\text{var}\{e_j\}}}$$

Example 18:

Consider the project of Example 14 To avoid repeating critical path calculations, the values of *a*, *m*, and *b* in the table below are selected such that *Bar* $(D_{ij}) = D_{ij}$ for all *i* and j in Example 14

Activity	i–j	(a, m, b)	Activity	i–j	(a, m, b)
A	1–2	(3, 5, 7)	E	3–5	(1, 2, 3)
B	1–3	(4, 6, 8)	F	3–6	(9, 11, 13)
C	2–3	(1, 3, 5)	G	4–6	(1, 1, 1)
D	2–4	(5, 8, 11)	H	5–6	(10, 12, 14)

The mean Bar(D_{ij}) and variance v_{ij} for the different activities are given in the following table. Note that for a dummy activity *(a, m, b)* = (0,0,0), hence its mean and variance also equal zero.

Activity	i–j	$\bar{D}_{ij}$	V_{ij}	Activity	i–j	$\bar{D}_{ij}$	V_{ij}
A	1–2	5	.444	E	3–5	2	.111
B	1–3	6	.444	F	3–6	11	.444
C	2–3	3	.444	G	4–6	1	.000
D	2–4	8	1.000	H	5–6	12	.444

The next table gives the longest path from node 1 to the different nodes, together with their associated mean and standard deviation.

Node	Longest path based on mean durations	Path mean	Path standard deviation
2	1-2	5.00	0.67
3	1-2-3	8.00	0.94
4	1-2-4	13.00	1.20
5	1-2-4-5	13.00	1.20
6	1-2-4-5-6	25.00	1.37

Finally, the following table computes the probability that each node is realized by time s_j specified by the analyst.

Node j	Longest path	Path mean	Path standard deviation	S_j	K_j	$P\{z \leq K_j\}$
2	1-2	5.00	0.67	5.00	0	.5000
3	1-2-3	8.00	0.94	11.00	3.19	.9993
4	1-2-4	13.00	1.20	12.00	-.83	.2033
5	1-2-4-5	13.00	1.20	14.00	.83	.7967
6	1-2-4-5-6	25.00	1.37	26.00	.73	.7673

Unit-III

Integer Linear Programming: Introduction- Applications-Integer Programming Solutions-Algorithms.

Chapter 3: Sec 3.1-3.3

INTEGER LINEAR PROGRAMMING

Integer linear programming (ILP) are linear programs in which same or all the variables are restricted to integer value.

3.1. Illustrative Applications:

This section presents a number of ILP applications. The applications generally fall into two categories: *direct* and *transformed.* In the *direct* category, the variables are naturally integer and may assume binary (0 or 1) or general discrete values. For example, the problem may involve determining whether or not a project is selected for execution (binary) or finding the optimal number of machines needed to perform a task (general discrete value). In the transformed category, the original problem, which may not involve any integer variables, is analytically intractable. Auxiliary integer variables (usually binary) are used to make it tractable. For example, in sequencing two jobs, *A* and *B,* on a single machine, job *A* may precede job *B* *or* job *B* may precede job *A.* The "or" nature of the constraints is what makes the problem analytically intractable, because all mathematical programming algorithms deal with "and" constraints only. The situation is remedied by using auxiliary binary variables to transform the "or" constraints into equivalent "and" constraints.

Pure Linear Program

A pure integer program is one in which all the variables are integer

Mixed Integer program

A mixed Integer Program is one in which some of the variable are integer.

Capital budgeting Problem

There are m projects to be evaluated over a ‘n’ year planning horizon. The following table gives the expected returns to each project available funds for each year and the associated yearly.

Expenditures

Project	Expenditures per year	Returns
	1 2 3 ….j………n	
1	c_{11} c_{12} ....c_{ij} .... c_{in}	a_1
2	c_{21} c_{22} ....c_{2j}…….c_{2n}	a_2
..	.. ..	..
..	.. ..	..
..	.. ..	..
i	c_{i1} c_{i2} ....c_{ij}…….c_{in}	a_i
..	.. ..	..
..	.. ..	..
..	.. ..	..
m	c_{m1} c_{m2} ....c_{mj}…….c_{mn}	A_m
Available funds	b_1 b_2 ....b_j............ b_n	

The problem is to determine the project to be executed over the n-year horizon so that the total return is maximum. This is called the capital budgeting problem.

Hence the problem reduced to a " yes -no" for its project.

Defined x_i as follows x_i=1 if project I is selected

0 if project I is not selected

Since the ILP model is max $z=a_1x_1+a_2x_2+\ldots+a_m x_m$

Subject to

$C_{11}x_1+c_{21}x_2+\ldots+c_{m1}x_m \leq b_1$

$C_{12}x_1+c_{22}x_2+\ldots+c_{m2}x_m \leq b_2$

..

..

$C_{1n}x_1+c_{2n}x_2+\ldots+c_{mn} x_m \leq b_n$

$x_1, x_2 \ldots\ldots x_m = (0, 1)$

This is a pure ILP Model.

Example 1:

Five projects are being evaluated over a three year planning horizon. The following table gives the expected returns for each project and the associated yearly expenditure.

Project	Expenditure(million$)			Returns (million$)
	1	2	3	
1	5	1	8	20
2	4	7	10	40
3	3	9	2	20
4	7	4	1	15
5	8	6	10	30
Available funds (million $)	25	25	25	

Solution:

Determine the project to be executed over the 3-year horizon. The problem reduces to a "Yes-No" decision for each project. Defined the binary variable xj as

Defined x_j as follows x_i=1 if project I is selected

0 if project I is not selected

Since the DIP model is gives as

max z = $20x_1+40x_2+\ldots.+20x_3+15x_4+30x_5$

Subject to

$5x_1+4x_2+3x_3+7x_4+8x_5\leq 25$

$x_1+7x_2+9x_3+4x_4+6x_5\leq 25$

$8x_1+10x_2+2x_3+x_4+10x_5\leq 25$

$x_1,x_2,\ldots x_m=(0,1)$ This is a

pure ILP model

For convenience, a pure integer problem is defined to have *all* integer variables. Otherwise, a problem is a mixed integer program if it deals with both continuous and integer variables.

Remarks.

It is interesting to compare the continuous LP solution with the ILP solution. The LP optimum, obtained by replacing $x_j = (0,1)$ with $0 \leq x_j \leq 1$ for all j, yields

$x_1 = .5789$, $x_2 = x_3 = x_4 = 1$, $x_5 = .7368$, and $z = 108.68$ (million $).

The solution is meaningless because two of the variables assume fractional values. We may *round* the solution to the closest integer values, which yields $x_1 = x_5 = 1$. However, the resulting solution is infeasible because the constraints are violated. More important, the concept of rounding is meaningless here because x_i rep-resents a "yes-no" decision.

Set-Covering Problem

In this class of problems, overlapping services are offered by a number of installations to a number of facilities. The objective is to determine the minimum number of installations that will *cover* (i.e., satisfy the service needs) of each facility. For example, water treatment plants can be constructed at various locations, with each plant serving different sets of cities. The overlapping arises when a given city can receive service from more than one plant.

Example 2: (Installing Security Telephones)

To promote on-campus safety, the U of A Security Department is in the process of installing emergency telephones at selected locations. The department wants to install the minimum number of telephones, provided that each of the campus main streets is served by at least one telephone. Figure 3.1 maps the principal streets (A to K) on campus.

It is logical to place the telephones at street intersections so that each telephone will serve at least two streets. Figure 3.1 shows that the layout of the streets requires a maximum of eight telephone locations.

Define

$$x_j = \begin{cases} 1, \text{ a telephone is installed in location } j \\ 0, \text{ otherwise} \end{cases}$$

The constraints of the problem require installing at least one telephone on each of the 11 streets (A to K). Thus, the model becomes

$$\text{Minimize } z = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8$$

subject to

$$\begin{aligned} x_1 + x_2 &\geq 1 \quad \text{(Street } A\text{)} \\ x_2 + x_3 &\geq 1 \quad \text{(Street } B\text{)} \\ x_4 + x_5 &\geq 1 \quad \text{(Street } C\text{)} \end{aligned}$$

The constraints of the problem require installing at least one telephone on each of the 11 streets (A to K).

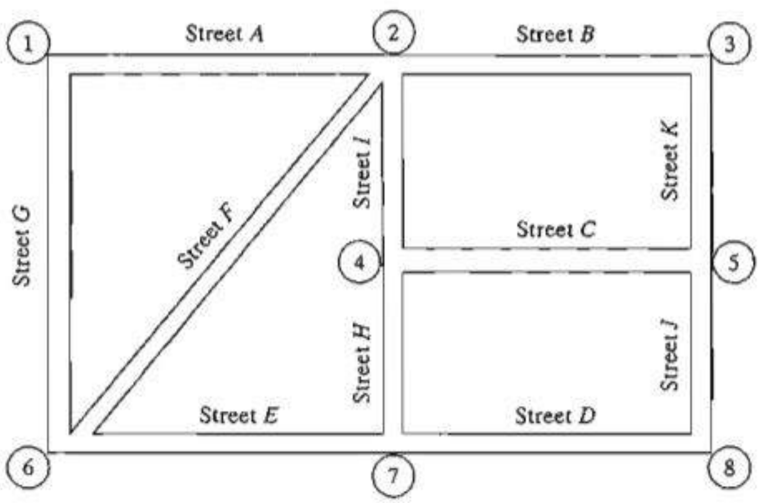


Figure 3.1

Street Map of the U of A Campus

$$
\begin{array}{lr}
x_7 + x_8 \geq 1 & \text{(Street } D\text{)} \\
x_6 + x_7 \geq 1 & \text{(Street } E\text{)} \\
x_2 + x_6 \geq 1 & \text{(Street } F\text{)} \\
x_1 + x_6 \geq 1 & \text{(Street } G\text{)} \\
x_4 + x_7 \geq 1 & \text{(Street } H\text{)} \\
x_2 + x_4 \geq 1 & \text{(Street } I\text{)} \\
x_5 + x_8 \geq 1 & \text{(Street } J\text{)} \\
x_3 + x_5 \geq 1 & \text{(Street } K\text{)} \\
x_j = (0, 1), j = 1, 2, \ldots, 8 &
\end{array}
$$

The optimum solution of the problem requires installing four telephones at intersections 1,2,5, and 7.

Remarks. In the strict sense, set-covering problems are characterized by (1) the variables X_j, $j = 1,2, \ldots, n$, are binary, (2) the left-hand-side coefficients of the constraints are 0 or 1, (3) the right-hand side of each constraint is of the form (≥ 1), and (4) the objective function minimizes $c_1x_1 + c_2x_2 + \ldots + c_nx_m$ where $cj > 0$ for all $j = 1,2, \ldots, n$. In the present example, $c_j = 1$ for all j. If cj represents the installation cost in location j, then these coefficients may assume values other than 1. Variations of the set-covering problem include additional side conditions, as some of the situations in Problem Set 9.lb show.

Fixed-Charge Problem

The fixed-charge problem deals with situations in which the economic activity incurs two types of costs: an initial "flat" fee that must be incurred to start the activity and a variable cost that is directly proportional to the level of the activity. For example, the initial tooling of a machine prior to starting production incurs a fixed setup cost regardless of how many units are manufactured. Once the setup is done, the cost of labour and material is proportional to the amount produced. Given that F is the fixed charge, e is the variable unit cost, and x is the level of production, the cost function is expressed as $C(x) = \begin{cases} F + cx, if\ x > 0 \\ 0,\ otherwise \end{cases}$

The function *C(x)* is intractable analytically because it involves a discontinuity at $x = 0$. The next example shows how binary variables are used to remove this intractability.

Example 3:(Choosing a Telephone Company)

There are three telephone companies in which a person can subscribes to their long distance service. The following table gives the flat monthly charge and charge per minutes for a long distance service fixed by the three companies.

Company	Flat monthly charge (Rs)	Charge per minute(Rs)
A	16	0.25
B	25	0.21
C	18	0.22

Mr. X usually makes an average of 200 minutes of long distance calls a month. He need not pay the flat monthly fees unless he makes calls and he can call among all three companies.

The problem is how should Mr. X use the three companies to minimize his monthly telephone bill. This problem is called fixed charge problem.

Formulation of ILP

Define x_1= company A, long distance minutes per month. x_2= company B, long distance minutes per month. x_3= company C, long distance minutes per month.

$$y_1 = \begin{cases} 1 & \text{if } x_1>0 \\ 0 & \text{if } x_1=0 \end{cases}$$

$$y_2 = \begin{cases} 1 & \text{if } x_2>0 \\ 0 & \text{if } x_2=0 \end{cases}$$

$$y_3 = \begin{cases} 1 & \text{if } x_3>0 \\ 0 & \text{if } x_3=0 \end{cases}$$

Since Mr. X makes about 200 minutes per month $x_1+x_2+x_3=200$ The

ILP model is minimize z

(i.e.) min $z=0.25x_1+0.21x_2+0.22x_3+16y_1+25y_2+18y_3$

subject to

$x_1+x_2+x_3<200$

$x_1 \leq 200y_1$

$x_2 \leq 200y_2$

$x_3 \leq 200\ y_3$

$x_1, \quad x_2, \quad x_3 \geq \quad 0$

$y_1,y_2,y_3=(0,1)$

This is a mixed ILP model.

The formulation shows that the *jth* monthly flat fee will be part of the objective function *z* only if $y_j = 1$, which can happen only if $x_j > 0$ (per the last three constraints of the model). If $x_j = 0$ at the optimum, then the minimization of *z,* together with the fact that the objective coefficient of y_j is strictly positive. will force y_j to equal zero, as desired.

The optimum solution yields $x_3 = 200$, $y_3 = 1$. and all the remaining variables equal to zero, which shows that BabyBell should be selected as my long-distance carrier. Remember that the information conveyed by $y_3 = 1$ is redundant because the same result is implied by $x_3 > 0$ (= 200). Actually. the main reason for using y_1. y_2, and y_3 is to account for the monthly fiat fee. In effect, the three binary variables convert an ill-behaved (nonlinear) model into an analytically tractable formulation. This conversion has resulted in introducing the integer (binary) variables in an otherwise continuous problem.

Either-Or and If-Then Constraints

In the fixed-charge problem, we used binary variables to handle the discontinuity in the objective cost function. In this section, we deal with models in which constraints are not satisfied simultaneously (either-or) or are dependent (if-then), again using binary variables. The transformation does not change the "or" or "dependence" nature of the constraints. It simply uses a mathematical trick to present them in the desired format of "and" constraints.

Example 4: (Job-Sequencing Model)

Jobco uses a single machine to process three jobs. Both the processing time and the due date (in days) for each job are given in the following table. The due dates are measured from zero, the assumed start time of the first job.

Job	Processing time (days)	Due date (days)	Late penalty $/day
1	5	25	19
2	20	22	12
3	15	35	34

The objective of the problem is to determine the minimum late-penalty sequence for processing the three jobs.

Define x_j = Start date in days for job j (measured from zero)

The problem has two types of constraints: the non-interference constraints (guaranteeing that no two jobs are processed concurrently) and the due dates constraints. Consider the non-interference constraints first.

Two jobs *i* and *j* with processing time *pi* and *pj* will not be processed concurrently if either $x_i \geq x_j + p_j$ or $x_j \geq x_i + p_i$, depending on whether job *j* precedes job i, or vice versa. Because all mathematical programs deal with *simultaneous* constraints only, we transform the either-or constraints by introducing the following auxiliary binary variable: $y_{ij} = \begin{cases} 1, if\ i\ precedes\ j \\ 0, if\ j\ precedes\ i \end{cases}$

For *M* sufficiently large, the either-or constraint is converted to the following two *simultaneous* constraints

$$My_{ij} + (x_i - x_j) \geq p_j \text{ and } M(1 - y_{ij}) + (x_j - x_i) \geq p_i$$

The conversion guarantees that only one of the two constraints can be active at anyone time. If $y_{ij} = 0$, the first constraint is active, and the second is redundant (because its left-hand side will include M, which is much larger than pi). If $y_{ij} = 1$, the first constraint is redundant, and the second is active.

Next, the due-date constraint is considered. Given that d_j is the due date for job j, let s_j be an unrestricted variable. Then, the associated constraint is

$$x_j + p_j + s_j = d_j$$

If $s_j \geq 0$, the due date is met, and if $s_j < 0$, a late penalty applies. Using the substitution

$$s_j = s_j^- - s_j^+, s_j^-, s_j^+ \geq 0$$

the constraint becomes

$$x_j + s_j^- - s_j^+ = d_j - p_j$$

The late-penalty cost is proportional to s_j^+.

The model for the given problem is

$$\text{Minimize } z = 19s_1^+ + 12s_2^+ + 34s_3^+$$

subject to

$$\begin{aligned}
x_1 - x_2 + My_{12} &\geq 20 \\
-x_1 + x_2 - My_{12} &\geq 5 - M \\
x_1 - x_3 + My_{13} &\geq 15 \\
-x_1 + x_3 - My_{13} &\geq 5 - M \\
x_2 - x_3 + My_{23} &\geq 15 \\
-x_2 + x_3 - My_{23} &\geq 20 - M \\
x_1 + s_1^- - s_1^+ &= 25 - 5 \\
x_2 + s_2^- - s_2^+ &= 22 - 20 \\
x_3 + s_3^- - s_3^+ &= 35 - 15
\end{aligned}$$

$$x_1, x_2, x_3, s_1^-, s_1^+, s_2^-, s_2^+, s_3^-, s_3^+ \geq 0$$

$$y_{12}, y_{13}, y_{23} = (0, 1)$$

The integer variables, y_{12}, y_{13}, and y_{23}, are introduced to convert the either-or constraints into simultaneous constraints. The resulting model is a *mixed* ILP.

To solve the model, we choose $M = 100$, a value that is larger than the sum of the processing times for all three activities.

The optimal solution is $x_1 = 20$, x_2, =0, and $x_3 = 25$, This means that job 2 starts at time 0, job 1 starts at time 20, and job 3 starts at time 25, thus yielding the optimal processing sequence 2 -> 1 -> 3. The solution calls for completing job 2 at time 0 + 20 = 20, job 1 at time = 20 + 5 = 25, and job 3 at 25 + 15 = 40 days. Job 3 is delayed by 40 - 35 = 5 days past its due date at a cost of 5 X $34 = $170.

Example 5: (Job Sequencing Model Revisited)

In Example 4, suppose that we have the following additional condition: If job *i* precedes job *j* then job *k* must precede job m. Mathematically, this if-then condition is translated as

$$\text{if } x_i + p_i \leq x_j \text{ then } x_k + p_k \leq x_m$$

Given $\varepsilon > 0$ and infinitesimally small and *M* sufficiently large, this condition is equivalent to the following two simultaneous constraints:

$$x_j - (x_i + p_i) \leq M(1 - w) - \varepsilon$$

$$(x_k + p_k) - x_m \leq Mw$$

$$w = (0, 1)$$

If $x_i + p_i \leq x_j$, then $x_j - (x_i + p_i) \geq 0$, which requires $w = 0$, and the second constraint becomes $x_k + p_k \leq x_m$, as desired. Else, *w* may assume the value 0 or 1, in which case the second constraint mayor may not be satisfied, depending on other conditions in the model

3.2. Integer Programming Algorithms:

The ILP algorithms are based on exploiting the tremendous computational success of LP. The strategy of these algorithms involves three steps.

Step 1. Relax the solution space of the ILP by deleting the integer restriction on all integer variables and replacing any binary variable *y* with the continuous range $0 \leq Y \leq 1$. The result of the relaxation is a regular LP

Step 2. Solve the LP, and identify its continuous optimum.

Step 3. Starting from the continuous optimum point, add special constraints that iteratively modify the LP solution space in a manner that will eventually render an optimum extreme point satisfying the integer requirements.

Two general methods have been developed for generating the special constraints in step 3.

1. Branch-and-bound (B&B) method
2. Cutting-plane method

Although neither method is consistently effective computationally, experience shows that the B&B method is far more successful than the cutting-plane method. This point is discussed further in this chapter.

3.2.1. Branch-and-Bound (B&B) Algorithm:

The first B&B algorithm was developed in 1960 by A. Land and G. Doig for the general mixed and pure ILP problem. Later, in 1965, E. Balas developed the additive algorithm for solving ILP problems with pure binary (zero or one) variable. The additive algorithm's computations were so simple (mainly addition and subtraction) that it was hailed as a possible breakthrough in the solution of general ILP. Unfortunately, it failed to produce the desired computational advantages. Moreover, the algorithm, which initially appeared unrelated to the B&B technique, was shown to be but a special case of the general Land and Doig algorithm.

This section will present the general Land-Doig B&B algorithm only. A numeric example is used to explain the details.

Example 1:

Consider the following ILP (B&B) method max z = $5x_1 + 4x_2$

Subject to

$x_1+x_2 \leq 5$

$10x_1+6x_2 \leq 45$

$x_1, x_2 \geq 0$ and integer.

Solution:

We consider to given LP as Po, max z =

$5x_1+4x_2$

Subject to

$x_1+x_2 \leq 5$

$10x_1+6x_2 \leq 45$

$x_1,x_2 \geq 0$ and integer.

Consider the first constraint as

$x_1+x_2=5$

put$x_1=0$

$x_2=5$

point (0,5)

put $x_2= 0$ $x_1= 5$

point (5,0)

Consider the second constraint

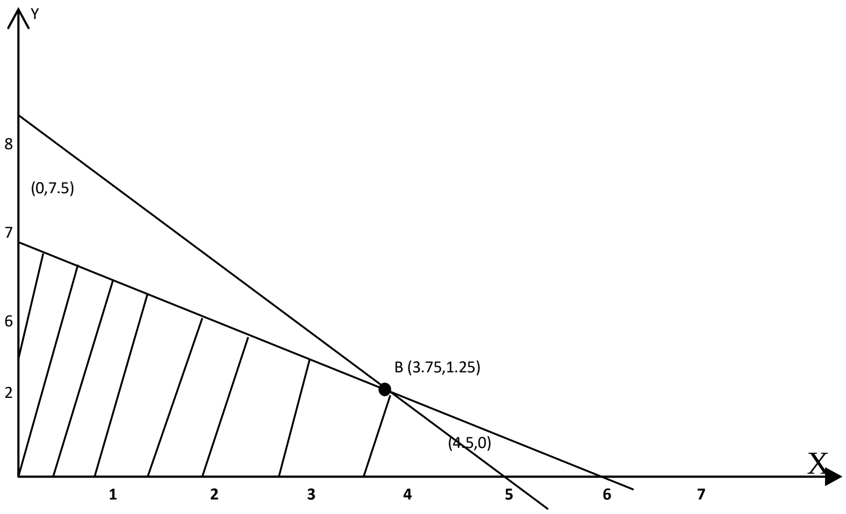
$10x_1+6x_2=45$

Put $x_1=0$, $x_2=45/6$

Point $(0, 45/6)$ (0,7.5)

Put $x_2=0$, $10x_1=45$, $x_1=4.5$

point (4.5,0)



$x_1+x_2=5$ (1)

$10x_1+6x_2=45$ (2)

$10x_1+10x_2=50$

$4x_2=5$

$x_2=\frac{5}{4} = 1.25$

$x_1=5-1.25$

$=3.75$

$x_1=3.75$, $x_2=1.25$

The point is (3.75,1.25)

At (0,0)

Max $z=5x_1+4x_2 z = 0$

At (4.5,0)

Max $z=5x_1+4x_2$

z = 5.45 + 0=22.5

At (3.75,1.25)

Max z=5 x 3.75+4 x 1.25

=18.75+5.00

=23.75

At (0,5)

Max z=0+4 x 5

Z=20

max z=23.75at B.

optimal Solutions are x_1=3.75, x_2=1.25 and x=23.75.

Here $x_1$and $x_2$values are not integer

We choose $x_1$as the branching variable. LP_o is

subdivided into two branches$LP_1 = LP_o + x_1 \leq 3$

$LP_2=LP_o+x_1\geq4$

Solve: LP_1

Consider the LP1 max

$z=5x_1+4x_2$

Subject to

$x_1+x_2\leq5$

$10x_1+6x_2\leq45$

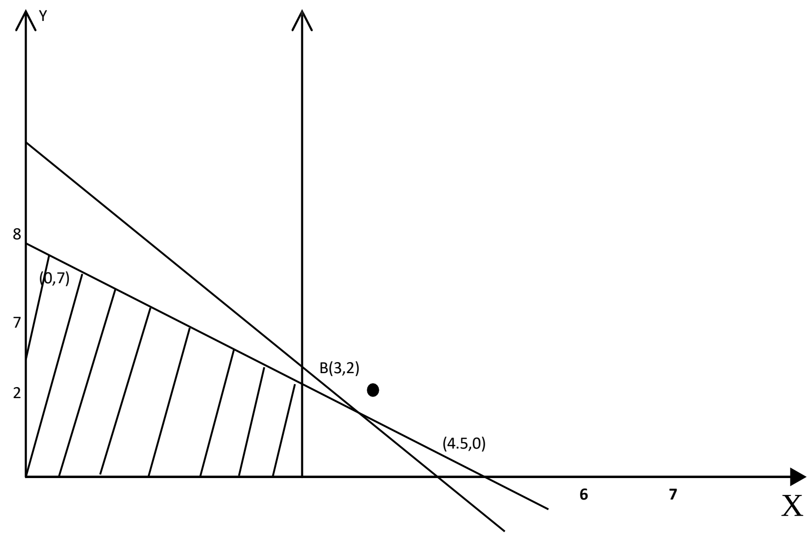
$x_1\leq3$

Consider the equation $x_1+ x_2= 5$

The point is (0,5) and (5, 0)

Consider the equation$10x_1+6x_2=45$

The point is the (0,7.5) and (4.5,0)



$x_1+x_2=5$ $x_1= 3$

$x_2=5-3=2$

The point is (3,2)

At O (0,0)

$x_1=0$, $x_2=0$

$z=5x_1+4x_2=0$

At A (3,0)

$x_1=3$, $x_2=0$ $z=5x3=15$

At B (3,2)

$x_1=3$, $x_2=0$

$z=5x_1+4x_2=5 \text{ x } 3+4 \text{ x } 2=23$

At C (0,5)

$z=5 \text{ x } 0+4 \text{ x } 5=20$

Max $z=23$

The optimal Solutions are $x_1=3$, $x_2=3$ and $z=23$ Here

$x_1$and $x_2$are integer

Solve LP_2

Consider the max $z = 5x_1+ 4x_2$

Subject to

$x_1 + x_2 \leq 5$

$10x_1 + 6x_2 \leq 45$

$x_1 \geq 4$

Consider the equation $x_1 + x_2 = 5$

the point is (0, 5) and (5,0)

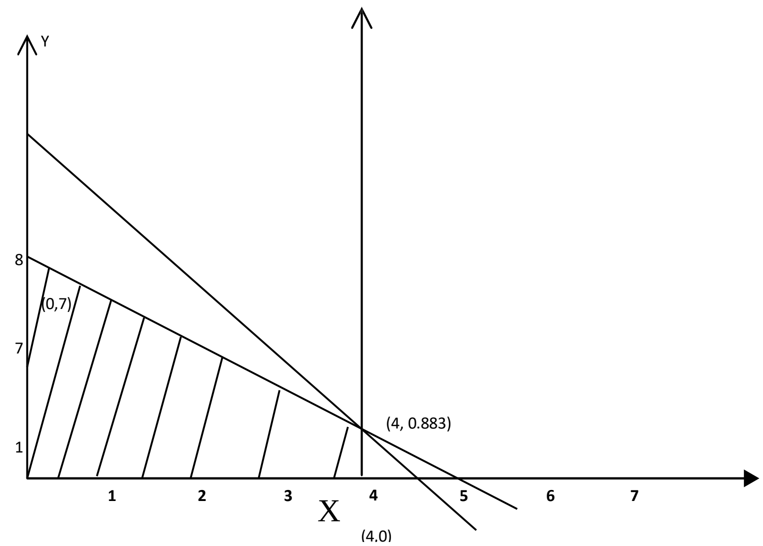
consider the equation $10x_1 + 6x_2 = 45$

The point is (0, 7.5) (4.5, 0)

$10x_1 + 6x_2 = 45$

$x_1 = 4$ $10x4 + 6x_2 = 45$

$x_2 = 0.833$



At 0(0,0)

z=0

At A (4,0)

z=20

At B (4.50)

z=22.5

At C (4,0.833)

$z = 5x_1 + 4x_2$

max z = 23.332

Here x_1 is integer and x_2 is not integer.

The Solution is not optimal

We choose x_2 as a branching variable. We subdivided LP_2 in sub problem in 2 sub problem.

$LP_3 = LP_2 + x_2 \leq 0$

$LP_4 = LP_2 + x_2 \geq 1$

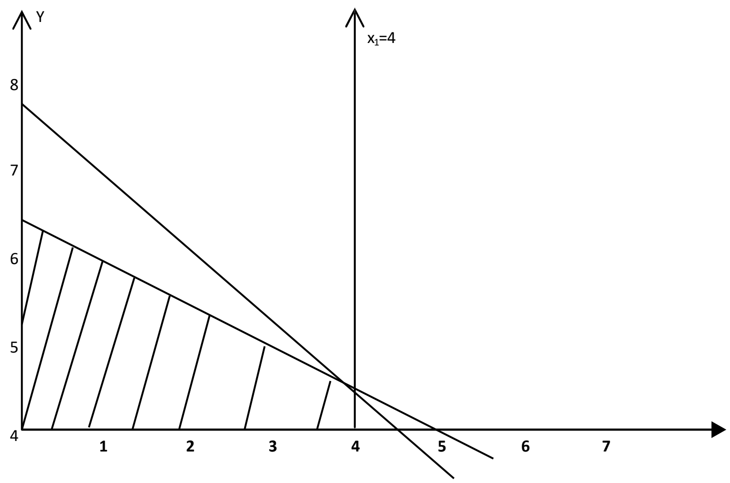
Solve LP_3

$\max z=5x_1+4x_2$ S.to

$x_1+x_2\leq 5$

$10x_1+6x_2\leq 45$

$x_1\geq 4$, $x_2\leq 0$ $x_1, x_2\geq 0$



At O (0,0)

z=0

At A (4,0)

z=5 x 4=20

At B (4.5, 0)

z=5 x 4.5=22.5

max z=22.5

The optimal Solution $x_1=4.5$, $x_2=0$ and z=22.5

x is not integer x_2 is integer. The Solution is not optimal The

LP_3 can be subdivided into two program.

$LP_5=LP_3+x_1\leq 4$

$LP_6=LP_3+x_1\geq 5$

Solve LP_5

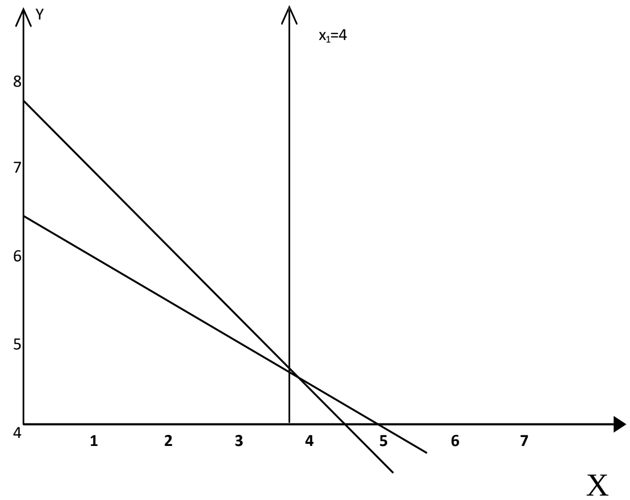
$\text{Max } z=5x_1+4x_2$

$x_1+x_2\leq 5$

$100x_1+6x_2\leq 45$

$x_1\geq 4$, $x_2\leq 0$, $x_1\leq 4$, $x_2\geq 0$

$x_1,\ x_2\geq 0$



At O (0,0)

$z=5(0) + 4(0) =0$

At A (4,6)

$z=20$

The optimal Solution $x_1=4$, $x_2=0$, $z=20$

The Solution is optimal

Solve LPs

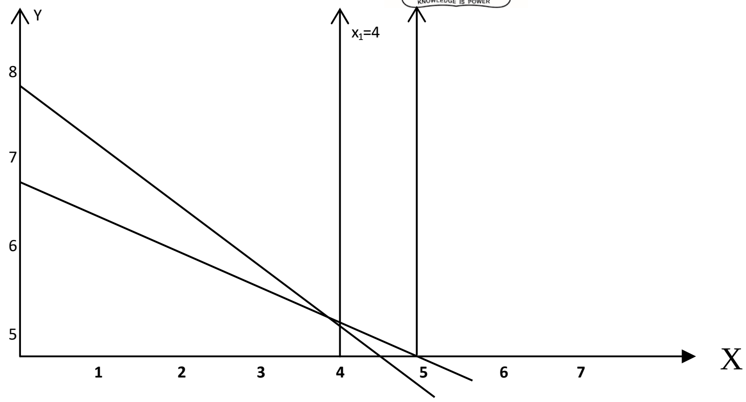
$\text{Max } z=5x_1+4x_2$

$x_1+ x_2\leq 5$

$10x_1+6x_2\leq 5$

$x_1\geq 4$ $x_2\leq 0$ $x_1\geq 5$

$x_2\geq 0$ $x_1,x_2 \geq 0$



LP$_6$ is fathomed because is it has no solution.

Solve LP$_4$

Max $z=5x_1+4x_2$

Subject to

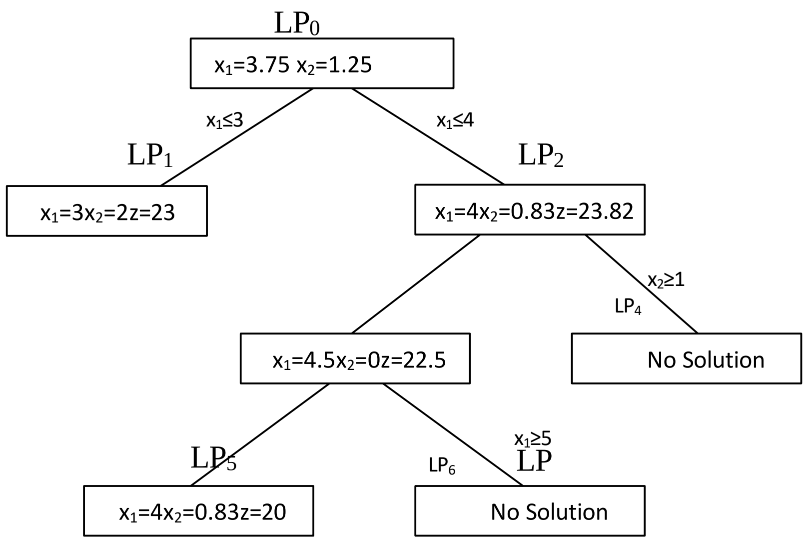
$x_1+x_2\leq 5$

$0x_1+6x_2\leq 45$, $x_1\geq 4$, $x_2\leq 1$ LP has no Solution LP$_4$ is fathomed

In LP$_5$the decision variable x_1, $x_2$are integer. z = 20

is a lower bound . The optimum value of z in LP$_1$as

23 The lower bound is z = 23



Lower Bound

3.2.2. Cutting-Plane Algorithm:

Algebra Development of cuts

The cutting plane algorithm start by solving the continuous Lp problem. In the optimum LP table be select one of the rows called the source row, for which the basic variable is non-integer. The desire cut is constructed from the fractional for componence of the coefficient of the source row. For this reason, it is reputed to as to fractional cut

Example 2:

Consider the following ILP.

Maximize $z = 7x_1 + 10x_2$

Subject to,

$-x_1 + 3x_2 \leq 6$

$7x_1 + x_2 \leq 35$

$x_1, x_2 \geq 0$ and integer.

The cutting-plane algorithm modifies the solution space by adding *cuts* that produce an optimum integer extreme point. Figure 3.6 gives an example of two such cuts.

Initially, we start with the continuous LP optimum z = 66(1/2), x_1 = 4(1/2), x_2 = 3(4/7). Next, we add cut I, which produces the (continuous) LP optimum solution z = 62, x_1= 4(4/7), x_2= 3. Then, we add cut II, which, together with cut I and the original constraints, produces the LP optimum z = 58, x_1 = 4, x_2 = 3. The last solution is all integer, as desired.

The added cuts do not eliminate any of the original feasible integer points, but must pass through at least one feasible or infeasible integer point. These are basic requirements of any cut.

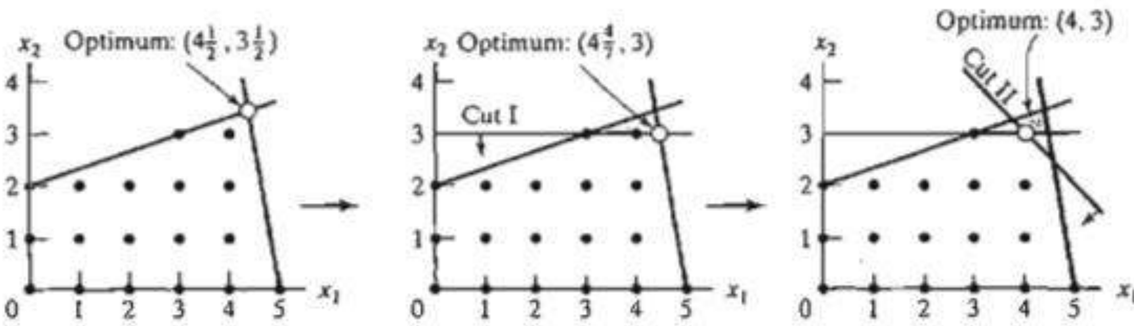


Figure 3.6

Illustration of the use of cuts in ILP

It is purely accidental that a 2-variable problem used exactly 2 cuts to reach the optimum integer solution. In general, the number of cuts, though finite, is independent of the size of the problem, in the sense that a problem with a small number of variables and constraints may re-quire more cuts than a larger problem.

Next, we use the same example to show how the cuts are constructed and implemented algebraically.

Given the slacks *x3* and *x4* for constraints 1 and 2, the optimum LP tableau is given as

Basic	x_1	x_2	x_3	x_4	Solution
z	0	0	$\frac{63}{22}$	$\frac{31}{22}$	$66\frac{1}{2}$
x_2	0	1	$\frac{7}{22}$	$\frac{1}{22}$	$3\frac{1}{2}$
x_1	1	0	$-\frac{1}{22}$	$\frac{3}{22}$	$4\frac{1}{2}$

The optimum continuous solution is $z = 66(1/2)$, $x_1 = 4(1/2)$, $x_2 = 3(1/2)$, $x_3 = 0$, $x_4 = 0$. The cut is developed under the assumption that *all* the variables (including the slacks x_3 and x_4*)* are integer.

Note also that because all the original objective coefficients are integer in this example, the value of *z* is integer as well.

The information in the optimum tableau can be written explicitly as

$$z + \frac{63}{22}x_3 + \frac{31}{22}x_4 = 66\frac{1}{2} \qquad (z\text{-equation})$$

$$x_2 + \frac{7}{22}x_3 + \frac{1}{22}x_4 = 3\frac{1}{2} \qquad (x_2\text{-equation})$$

$$x_1 - \frac{1}{22}x_3 + \frac{3}{22}x_4 = 4\frac{1}{2} \qquad (x_1\text{-equation})$$

A constraint equation can be used as a source row for generating a cut, provided its right-hand side is fractional. We also note that the z-equation can be used as a source row because *z* happens to be integer in this example. We will demonstrate how a cut is generated from each of these source rows, starting with the z-equation.

First, we factor out all the non-integer coefficients of the equation into an integer value and a fractional component, *provided that the resulting fractional component is strictly positive*. For example,

$$\frac{5}{2} = \left(2 + \frac{1}{2}\right)$$

$$-\frac{7}{3} = \left(-3 + \frac{2}{3}\right)$$

The factoring of the z-equation yields

$$z + \left(2 + \frac{19}{22}\right)x_3 + \left(1 + \frac{9}{22}\right)x_4 = \left(66 + \frac{1}{2}\right)$$

Moving all the integer components to the left-hand side and all the fractional components to the right-hand side, we get

$$z + 2x_3 + 1x_4 - 66 = -\frac{19}{22}x_3 - \frac{9}{22}x_4 + \frac{1}{2} \qquad (1)$$

Because x3 and *x4* are nonnegative and all fractions are originally strictly positive, the right-hand side must satisfy the following inequality:

$$-\frac{19}{22}x_3 - \frac{9}{22}x_4 + \frac{1}{2} \leq \frac{1}{2} \qquad (2)$$

Next, because the left-hand side in Equation (1), $z + 2x_3 + 1x_4 - 66$, is an integer value by construction, the right-hand side, $-\frac{19}{22}x_3 - \frac{19}{22}x_4 + \frac{1}{2}$, must also be integer. It then follows that (2) can be replaced with the inequality:

$$-\frac{19}{22}x_3 - \frac{9}{22}x_4 + \frac{1}{2} \leq 0$$

This result is justified because an integer value ≤ 1/2 must necessarily be ≤ 0.

The last inequality is the desired cut and it represents a *necessary* (but not sufficient) condition for obtaining an integer solution. It is also referred to as the **fractional cut** because all its coefficients are fractions.

Because $x_3 = x_4 = 0$ in the optimum continuous LP tableau given above, the current continuous solution violates the cut (because it yields 1/2 ≤ 0). Thus, if we add this cut to the optimum tableau, the resulting optimum extreme point moves the solution toward satisfying the integer requirements.

Before showing how a cut is implemented in the optimal tableau, we will demonstrate how cuts can also be constructed from the constraint equations. Consider the x_1-row:

$$x_1 - \frac{1}{22}x_3 + \frac{3}{22}x_4 = 4\frac{1}{2}$$

Factoring the equation yields

$$x_1 + \left(-1 + \frac{21}{22}\right)x_3 + \left(0 + \frac{3}{22}\right)x_4 = \left(4 + \frac{1}{2}\right)$$

The associated cut is

$$-\frac{21}{22}x_2 - \frac{3}{22}x_4 + \frac{1}{2} \leq 0$$

Similarly, the x_2-equation

$$x_2 + \frac{7}{22}x_3 + \frac{1}{22}x_4 = 3\frac{1}{2}$$

is factored as

$$x_2 + \left(0 + \frac{7}{22}\right)x_3 + \left(0 + \frac{1}{22}\right)x_4 = 3 + \frac{1}{2}$$

Hence, the associated cut is given as

$$-\frac{7}{22}x_3 - \frac{1}{22}x_4 + \frac{1}{2} \leq 0$$

Anyone of three cuts given above can be used in the first iteration of the cutting-plane a1gorithm. It is not necessary to generate all three cuts before selecting one.

Arbitrarily selecting the cut generated from the x_2-row, we can write it in equation form as

$$-\frac{7}{22}x_3 - \frac{1}{22}x_4 + s_1 = -\frac{1}{2}, \quad s_1 \geq 0 \qquad \text{(Cut I)}$$

This constraint is added to the LP optimum tableau as follows:

Basic	x_1	x_2	x_3	x_4	S_1	Solution
z	0	0	$\frac{63}{22}$	$\frac{31}{22}$	0	$66\frac{1}{2}$
x_2	0	1	$\frac{7}{22}$	$\frac{1}{22}$	0	$3\frac{1}{2}$
x_1	1	0	$-\frac{1}{22}$	$\frac{3}{22}$	0	$4\frac{1}{2}$
S_1	0	0	$-\frac{7}{22}$	$-\frac{1}{22}$	1	$-\frac{1}{2}$

The tableau is optimal but infeasible. We apply the dual simplex method (Section 4.4.1) to recover feasibility, which yields

Basic	x_1	x_2	x_3	x_4	S_1	Solution
z	0	0	0	1	9	62
x_2	0	1	0	0	1	3
x_1	1	0	0	$\frac{1}{7}$	$-\frac{1}{7}$	$4\frac{1}{7}$
x_3	0	0	1	$\frac{1}{7}$	$-\frac{22}{7}$	$1\frac{4}{7}$

The last solution is still non-integer in xl and *x3*. Let us arbitrarily select xl as the next source row-that is, $x_1 + \left(0 + \frac{1}{7}\right) x_4 + \left(-1 + \frac{6}{7}\right) s_1 = 4 + \frac{4}{7}$

The associated cut is $-\frac{1}{7}x_4 - \frac{6}{7}s_2 = -\frac{4}{7}$, $s_2 \geq 0$ (cut II)

Basic	x_1	x_2	x_3	x_4	S_1	S_2	Solution
z	0	0	0	1	9	0	62
x_2	0	1	0	0	1	0	3
x_1	1	0	0	$\frac{1}{7}$	$-\frac{1}{7}$	0	$4\frac{1}{7}$
x_3	0	0	1	$\frac{1}{7}$	$-\frac{22}{7}$	0	$1\frac{4}{7}$
S_2	0	0	0	$-\frac{1}{7}$	$-\frac{6}{7}$	1	$-\frac{4}{7}$

The dual simplex method yields the following tableau:

Basic	x_1	x_2	x_3	x_4	S_1	S_2	Solution
z	0	0	0	0	3	7	58
x_2	0	1	0	0	1	0	3
x_1	1	0	0	0	-1	1	4
x_3	0	0	0	0	-4	1	1
x_4	0	0	1	1	6	-7	4

The optimum solution

(x_1= 4, x_2= 3, *z* = 58) is all integer. It is not accidental that all the coefficients of the last tableau are integers, a property of the implementation of the fractional cut.

Remarks. It is important to point out that the fractional cut assumes that *all* the variables, *including slack and surplus*, are integer. This means that the cut deals with pure integer problems only. The importance of this assumption is illustrated by an example.

Consider the constraint $x_1 + \frac{1}{3}x_2 \leq \frac{13}{2}$

$x_1, x_2 \geq 0$ and integer

From the standpoint of solving the associated ILP, the constraint is treated as an equation by using the nonnegative slack s_1-that is,

$$x_1 + \frac{1}{3}x_2 + s_1 = \frac{13}{2}$$

The application of the fractional cut assumes that the constraint has a feasible integer solution in all xl, *x2*, and sl. However, the equation above will have a feasible integer solution in *xl* and *x2 only if* x1 *is non-integer*. This means that the cutting-plane algorithm will show that the problem has no feasible integer solution, even though the variables of concern, xl and *x2*, can assume feasible integer values.

There are two ways to remedy this situation.

1. Multiply the entire constraint by a proper constant to remove all the fractions. For example, multiplying the constraint above by 6, we get $6x_1 + 2x_2 \leq 39$

Any integer solution of *x1* and *x2* automatically yields integer slack. However, this type of conversion is appropriate for only simple constraints, because the magnitudes of the integer coefficients may become excessively large in some cases.

2.Use a special cut, called the mixed cut, which allows only a subset of variables to assume integer values, with all the other variables (including slack and surplus) remaining continuous.

3.2.3. Computational Considerations in ILP:

To date, and despite over 40 years of research, there does not exist a computer code that can solve ILP consistently. Nevertheless, of the two solution algorithms presented in this chapter, B&B is more reliable. Indeed, practically a U commercial ILP codes are B&B based. Cutting-plane methods are generally difficult and uncertain, and the round off error presents a serious problem. This is true because the "accuracy" of the cut depends on the accuracy of a true representation of its fractions on the computer. For instance, in Example 2, the fraction 1/7 cannot be represented exactly as a floating point regardless of the level of precision that may be used. Though attempts have been made to improve the cutting-plane computational efficacy, the end results are not encouraging. In most cases, the cutting-plane method is used in a secondary capacity to improve B&B performance at each sub problem by eliminating a portion of the solution space associated with a sub problem.

The most important factor affecting computations in integer programming is the number of integer variables and the feasible range in which they apply. Because avail-able algorithms are not consistent in producing a numeric ILP solution, it may be advantageous computationally to reduce the number of integer variables in the ILP model as much as possible. The following suggestions may prove helpful:

1. Approximate integer variables by continuous ones wherever possible.
2. For the integer variables, restrict their feasible ranges as much as possible.

3. Avoid the use of nonlinearity in the model.

The importance of the integer problem in practice is not yet matched by the development of reliable solution algorithms. The nature of discrete mathematics and the fact that the integer solution space is a nonconvex set make it unlikely that new theoretical breakthroughs will be achieved in the area of integer programming. Instead, new technological advances in computers (software and hardware) remain the best hope for improving the efficiency of ILP codes.

3.3. Traveling Salesperson (TSP) Problem:

Historically, the TSP problem deals with finding the shortest (closed) tour in an n-city situation where each city is visited exactly once. The problem, in essence, is an assignment model that excludes sub tours. Specifically, in an n-city situation, define

$$x_{ij} = \begin{cases} 1, \text{ if city } j \text{ is reached from city } i \\ 0, \text{ otherwise} \end{cases}$$

Given that *dij* is the distance from city *i* to city *j*, the TSP model is given as

$$\text{Minimize } z = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} x_{ij},\ d_{ij} = \infty \text{ for all } i = j$$

subject to

$$\sum_{j=1}^{n} x_{ij} = 1,\ i = 1, 2, \ldots, n \qquad (1)$$

$$\sum_{i=1}^{n} x_{ij} = 1,\ j = 1, 2, \ldots, n \qquad (2)$$

$$x_{ij} = (0, 1) \qquad (3)$$

$$\text{Solution forms an } n\text{-city tour} \qquad (4)$$

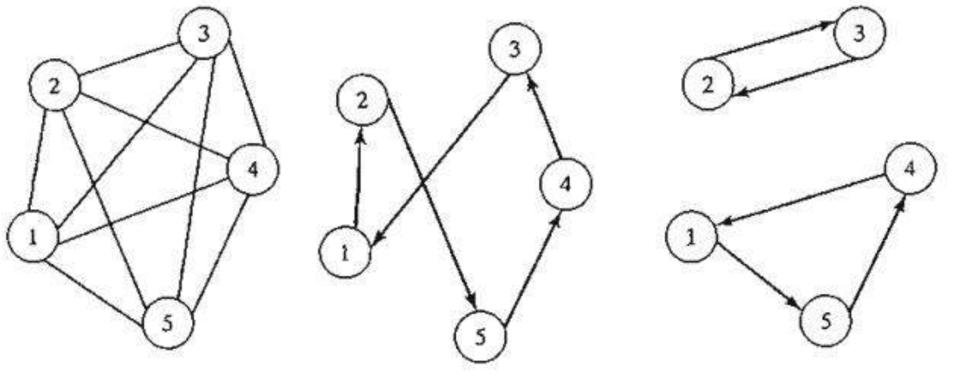


Figure 3.7

A 5-city TSP example with a tour and subtour solutions of the associated assignment model

Constraints (1), (2), and (3) define a regular assignment model. Figure 3.7. demonstrates a 5-city problem. The arcs represent two-way routes. The figure also illustrates a tour and a sub tour solution of the associated assignment model. If the optimum solution of the assignment model (i.e., excluding constraint 4) happens to produce a tour, then it is also optimum for the TSP. Otherwise, restriction (4) must be accounted for to ensure a tour solution.

Example 1:

The daily production schedule at the Rainbow Company includes batches of white *(W),* yellow *(Y),* red *(R),* and black *(B)* paints. Because Rainbow uses the same facilities for all four types of paint, proper cleaning between batches is necessary. The table below summarizes the clean-up time in minutes. Because each color is produced in a single batch, diagonal entries in the table are assigned infinite setup time. The objective is to determine the optimal sequencing for the daily production of the four colors that will minimize the associated total clean-up time.

Current paint	Cleanup min given next paint is White	Yellow	Black	Red
White	∞	10	17	15
Yellow	20	∞	19	18
Black	50	44	∞	25
Red	45	40	20	∞

Each paint is thought of as a "city" and the "distances" are represented by the clean-up time needed to switch from one paint batch to the next. The situation reduces to determining the *shortest loop* that starts with one paint batch and passes through each of the remaining three paint batches exactly once before returning back to the starting paint.

We can solve this problem by exhaustively enumerating the six [(4 - 1)! = 3! = 6] possible loops of the network. The following table shows that *W* -> *Y* -> *R* -> *B* -> *W* is the optimum loop.

Production loop	Total clean-up time
$W \to Y \to B \to R \to W$	10 + 19 + 25 + 45 = 99
$W \to Y \to R \to B \to W$	10 + 18 + 20 + 50 = 98
$W \to B \to Y \to R \to W$	17 + 44 + 18 + 45 = 124
$W \to B \to R \to Y \to W$	17 + 25 + 40 + 20 = 102
$W \to R \to B \to Y \to W$	15 + 20 + 44 + 20 = 99
$W \to R \to Y \to B \to W$	15 + 40 + 19 + 50 = 124

Exhaustive enumeration of the loops is not practical in general. Even a modest size 11-city problem will require enumerating 10! = 3,628,800 tours, a daunting task indeed. For this reason, the problem must be formulated and solved in a different manner, as we will show later in this section.

To develop the assignment-based formulation for the paint problem, define

X_{ij} = 1 if paint *j* follows paint *i* and zero otherwise

Letting *M* be a sufficiently large positive value, we can formulate the Rainbow problem as

$$\text{Minimize } z = Mx_{WW} + 10x_{WY} + 17x_{WB} + 15x_{WR} + 20x_{YW} + Mx_{YY} + 19x_{YB} + 18x_{YR} + 50x_{BW} + 44x_{BY} + Mx_{BB} + 25x_{BR} + 45x_{RW} + 40x_{RY} + 20x_{RB} + Mx_{RR}$$

subject to

$$x_{WW} + x_{WY} + x_{WB} + x_{WR} = 1$$
$$x_{YW} + x_{YY} + x_{YB} + x_{YR} = 1$$
$$x_{BW} + x_{BY} + x_{BB} + x_{BR} = 1$$
$$x_{RW} + x_{RY} + x_{RB} + x_{RR} = 1$$
$$x_{WW} + x_{YW} + x_{BW} + x_{RW} = 1$$
$$x_{WY} + x_{YY} + x_{BY} + x_{RY} = 1$$
$$x_{WB} + x_{YB} + x_{BB} + x_{RB} = 1$$
$$x_{WR} + x_{YR} + x_{BR} + x_{RR} = 1$$
$$x_{ij} = (0, 1) \quad \text{for all } i \text{ and } j$$

Solution is a tour (loop)

The use of *M* in the objective function guarantees that a paint job cannot follow itself. The same result can be realized by deleting *Xww*, *XYY*, *XBB*, and *XRR* from the entire model.

3.3.1. Heuristic Algorithms:

This section presents two heuristics: the *nearest-neighbor* and the *sub tour-reversal* algorithms. The first is easy to implement and the second requires more computations. The trade-off is that the second algorithm generally yields better results. Ultimately, the two heuristics are combined into one heuristic, in which the output of the nearest-neighbor algorithm is used as input to the reversal algorithm.

The Nearest-Neighbor Heuristic. As the name of the heuristic suggests, a "good" solution of the TSP problem can be found by starting with any city (node) and then connecting it with the closest one. The just-added city is then linked to its nearest unlinked city (with ties broken arbitrarily). The process continues until a tour is formed.

Example 1:

The matrix below summarizes the distances in miles in as-city TSP problem.

$$\|d_{ij}\| = \begin{pmatrix} \infty & 120 & 220 & 150 & 210 \\ 120 & \infty & 100 & 110 & 130 \\ 220 & 80 & \infty & 160 & 185 \\ 150 & \infty & 160 & \infty & 190 \\ 210 & 130 & 185 & \infty & \infty \end{pmatrix}$$

The heuristic can start from any of the five cities. Each starting city may lead to a different tour. The following table provides the steps of the heuristic starting at city 3.

Step	Action	(Partial) tour
1	Start with city 3	3
2	Link to city 2 because it is closest to city 3 ($d_{32} = \min\{220, \mathbf{80}, \infty, 160, 185\}$)	3-2
3	Link to node 4 because it is closest to node 2 ($d_{24} = \min\{120, \infty, —, \mathbf{110}, 130\}$)	3-2-4
4	Link to node 1 because it is closest to node 4 ($d_{41} = \min\{\mathbf{150}, \infty, —, —, 190\}$)	3-2-4-1
5	Link to node 5 by default and connect back to node 3 to complete the tour	3-2-4-1-5-3

Notice the progression of the steps: Comparisons exclude distances to nodes that are part of a constructed partial tour. These are indicated by (-) in the *Action* column of the table.

The resulting tour 3-2-4-1-5-3 has a total length of 80 + 110 + 150 + 210 + 185 = 735 miles. Observe that the quality of the heuristic solution is starting-node dependent. For example, starting from node 1, the constructed tour is 1-2-3-4-5-1 with a total length of 780 miles (try it!).

Sub tour Reversal Heuristic. In an n-city situation, the sub tour reversal heuristic starts with a feasible tour and then tries to improve on it by reversing 2-city sub tours, followed by 3-city sub tours, and continuing until reaching sub tours of size n - 1.

Example 2:

Consider the problem of Example 1. The reversal steps are carried out in the following table using the feasible tour 1-4-3-5-2-1 of length 745 miles:

Type	Reversal	Tour	Length
Start	—	(1-4-3-5-2-1)	**745**
Two-at-a-time reversal	4-3	1-3-4-5-2-1	820
	3-5	(1-4-5-3-2-1)	**725**
	5-2	1-4-3-2-5-1	730
Three-at-a-time reversal	4-5-3	1-3-5-4-2-1	∞
	5-3-2	1-4-2-3-5-1	∞
Four-at-a-time reversal	4-5-3-2	1-2-3-5-4-1	∞

The two-at-a-time reversals of the initial tour 1-4-3-5-2-1 are 4-3, 3-5, and 5-2, which leads to the given tours with their associated lengths of 820, 725, and 730. Since 1-4-5-3-2-1 yields a smaller length (= 725), it is used as the starting tour for making the three-at-a-time reversals. As shown in the table, these reversals produce no better results. The same result applies to the four-at-a-time reversal. Thus, 1-4-5-3-2-1 (with length 725 miles) provides the best solution of heuristic.

Notice that the three-at-a-time reversals did not produce a better tour, and, for this rea-son, we continued to use the best two-at-a-time tour with the four-at-a-time reversal. Notice also that the reversals do not include the starting city of the tour (= 1 in this example) because the process does not yield a tour. For example, the reversal 1-4 leads to 4-1-3-5-2-1, which is not a tour.

The solution determined by the reversal heuristic is a function of the initial feasible tour used to start the algorithm. For example, if we start with 2-3-4-1-5-2 with length 750 miles, the heuristic produces the tour 2-1-4-3-5-2 with length 745 miles (verify!), which is inferior to the solution we have in the table above. For this reason, it may be advantageous to first utilize the nearest-neighbor heuristic to determine *all* the tours that result from using each city as a starting node and then select the best as the starting tour for the reversal heuristic. This combined heuristic should, in general, lead to superior solutions than if either heuristic is applied separately. The following table shows the application of the composite heuristic to the present example.

Heuristic	Starting city	Tour	Length
	1	1-2-3-4-5-1	780
	2	2-3-4-1-5-2	750
Nearest neighbor	3	**(3-2-4-1-5-3)**	**735**
	4	4-1-2-3-5-4	∞
	5	5-2-3-4-1-5	750
	2-4	3-4-2-1-5-3	∞
	4-1	(3-2-1-4-5-3)	**725**
	1-5	3-2-4-5-1-3	810
Reversals	2-1-4	3-4-1-2-5-3	745
	1-4-5	3-2-5-4-1-3	∞
	2-1-4-5	3-5-4-1-2-3	∞

3.3.2. B&B Solution Algorithm:

The idea of the B&B algorithm is to start with the optimum solution of the associated assignment problem. If the solution is a tour, the process ends. Otherwise, restrictions are imposed to remove the sub tours. This can be achieved by creating as many branches as the number of xij variables associated with one of the sub tours. Each branch will correspond to setting one of the variables of the sub tour equal to zero (recall that all the variables associated with a sub tour equal 1). The solution of the resulting assignment problem may or may not produce a tour. If it does, we use its objective value as an upper bound on the true minimum tour length. If it does not, further branching is necessary, again creating as many branches as the number of variables in one of the sub tours. The process continues until all unexplored sub problems have been fathomed, either by producing a better (smaller) *upper bound* or because there is evidence that the sub problem cannot produce a better solution. The optimum tour is the one associated with the best upper bound.

The following example provides the details of the TSP B&B algorithm.

Example 1:

Consider the following 5-city TSP problem:

$$\|d_{ij}\| = \begin{pmatrix} \infty & 10 & 3 & 6 & 9 \\ 5 & \infty & 5 & 4 & 2 \\ 4 & 9 & \infty & 7 & 8 \\ 7 & 1 & 3 & \infty & 4 \\ 3 & 2 & 6 & 5 & \infty \end{pmatrix}$$

We start by solving the associated assignment, which yields the following solution:

$z = 15$, $(x_{13} = x_{31} = 1)$, $(x_{25} = x_{54} = x_{42} = 1)$, all others = 0

This solution yields two sub tours: (1-3-1) and (2-5-4-2), as shown at node 1 in Figure 3.8. The associated total distance is $z = 15$, which provides a lower bound on the optimal length of the 5-city tour.

A straightforward way to determine an upper bound is to select any tour and use its length as an upper bound estimate. For example, the tour 1-2-3-4-5-1 (selected totally arbitrarily) has a total length of 10 + 5 + 7 + 4 + 3 = 29. Alternatively, a better upper bound can be found by applying the heuristic of Section 3.3.1. For the moment, we will use the upper bound of length 29 to apply the B&B algorithm. Later, we use the "improved" upper bound obtained by the heuristic to demonstrate its impact on the search tree.

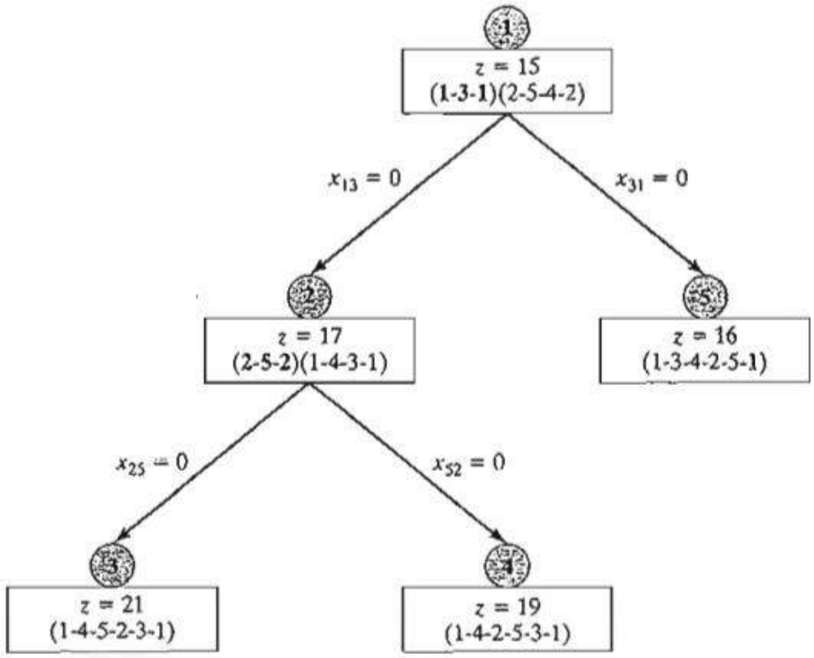


Figure 3.8

The computed lower and upper bounds indicate that the optimum tour length lies in range (15, 29). A solution that yields a tour length larger than (or equal to) 29 is discarded as non-promising.

To eliminate the sub tours at node **1,** we need to "disrupt" its loop by forcing its member variables, *xij,* to be zero. Sub tour 1-3-1 is disrupted if we impose the restriction xl3 = 0 or *x31* = 0 (i.e., one at a time) on the assignment problem at node 1. Similarly, sub tour 2-5-4-2 is eliminated by imposing one of the restrictions *x25* = 0, *x54* = 0, or *x42* = 0. In terms of the B&B tree, each of these restrictions gives rise to a branch and hence a new sub problem. It is important to notice that branching *both* sub tours at node 1 is *not* necessary. Instead, only *one* sub tour needs to be disrupted at anyone node. The idea is that a breakup of one sub tour automatically alters the member variables of the other sub tour and hence produces conditions that are favourable to creating a tour. Under this argument, it is more efficient to select the sub tour with the smallest number of cities because it creates the smallest number of branches.

Targeting sub tour (1-3-1), two branches $x13 = 0$ and $x31 = 0$ are created at node 1. The associated assignment problems are constructed by removing the row and column associated with the zero variable, which makes the assignment problem smaller. Another way to achieve the same result is to leave the size of the assignment problem unchanged and simply assign an infinite distance to the branching variable. For example, the assignment problem associated with $xl3 = 0$ requires substituting $d13 = \infty$ in the assignment model at node 1. Similarly, for $x31 = 0$, we substitute $d31 = \infty$.

In Figure 3.8., we arbitrarily start by solving the sub problem associated with $x13 = 0$ by setting $dl3 = \infty$, Node 2 gives the solution $z = 17$ but continues to produce the sub tours (2-5-2) and (1-4-3-1). Repeating the procedure, we applied at node 1 gives rise to two branches: $x25 = 0$ and $x52 = 0$.

We now have three unexplored sub problems, one from node 1 and two from node 2, and we are free to investigate any of them at this point. Arbitrarily exploring the sub problem associated with $x25 = 0$ from node 2, we set $d13 = \infty$ and $d25 = \infty$ in the *original* assignment problem, which yields the solution $z = 21$ and the tour solution 1-4-5-2-3-1 at node 3. The tour solution at node 3 lowers the upper bound from $z == 29$ to $z == 21$. This means that any unexplored sub problem that can be shown to yield a tour length larger than 21 is discarded as non-promising.

We now have two unexplored sub problems. Selecting the sub problem 4 for exploration, we set $dl3 = \infty$ and $d52 = \infty$ in the *original* assignment, which yields the tour solution 1-4-2-5-3-1 with $z = 19$. The new solution provides a better tour than the one associated with the current upper bound of 21. Thus, the new upper bound is updated to $z = 19$ and its associated tour, 1-4-2-5-3-1, is the best available so far.

Only sub problem 5 remains unexplored. Substituting $d31 = \infty$ in the *original* assignment problem at node 1, we get the tour solution 1-3-4-2-5-1 with $z = 16$, at node 5. Once again, this is a better solution than the one associated with node 4 and thus requires updating the upper bound to $z = 16$.

There are no remaining unfathomed nodes, which completes the search tree. The optimal tour is the one associated with the current upper bound: 1-3-4-2-5-1 with length 16 miles.

Remarks. The solution of the example reveals two points:

1. Although the search sequence 1 -> 2 -> 3 -> 4 -> 5 was selected deliberately to demonstrate the mechanics of the B&B algorithm and the updating of its upper bound, we generally have no way of predicting which sequence should be adopted to improve the efficiency of the search. Some rules of thumb can be of help. For example, at a given node we can start with the branch associated with the *largest dij* among all the created branches. By cancelling the tour leg with the largest *dij*, the hope is that a "good" tour with a smaller total length will be found. In the present example, this rule calls for exploring branch *x31* == 0 to node 5 be-fore branch *x13* to node 2 because *(d31* == 4) > *(d13* == 3), and this would have produced the upper bound *z* == 16, which automatically fathoms node 2 and, hence, eliminates the need to create nodes 3 and 4. Another rule calls for sequencing the exploration of the nodes in a horizontal tier (rather than vertically). The idea is that nodes closer to the starting node are *more likely* to produce a tighter upper bound because the number of additional constraints (of the type *xij* = 0) is smaller. This rule would have also discovered the solution at node 5 sooner.

2. The B&B should be applied in conjunction with the heuristic in Section 3.3.1. The heuristic provides a "good" upper bound which can be used to fathom nodes in the search tree. In the present example, the heuristic yields the tour 1-3-4-2-5-1 with a length of 16 distance units.

3.3.3. Cutting-Plane Algorithm:

The idea of the cutting plane algorithm is to add a set of constraints to the assignment problem that prevent the formation of a sub tour. The additional constraints are defined as follows. In an *n-city* situation, associate a continuous variable *uj* (≥ 0) with cities 2, 3, ..., and *n*. Next, define the required set of additional constraints as

$$u_i - u_j + nx_{ij} \leq n - 1, i = 2, 3, \ldots, n; j = 2, 3, \ldots, n; i \neq j$$

These constraints, when added to the assignment model, will automatically remove all sub tour solutions.

Example 1:

Consider the following distance matrix of a 4-city TSP problem.

$$\|d_{ij}\| = \begin{pmatrix} - & 13 & 21 & 26 \\ 10 & - & 29 & 20 \\ 30 & 20 & - & 5 \\ 12 & 30 & 7 & - \end{pmatrix}$$

The associated LP consists of the assignment model constraints plus the additional constraints in the table below. All *xij* = (0, 1) and all *uj* ≥ 0.

No.	x_{11}	x_{12}	x_{13}	x_{14}	x_{21}	x_{22}	x_{23}	x_{24}	x_{31}	x_{32}	x_{33}	x_{34}	x_{41}	x_{42}	x_{43}	x_{44}	u_2	u_3	u_4	
1							4										1	−1		≤3
2								4									1		−1	≤3
3										4							−1	1		≤3
4												4						1	−1	≤3
5														4			−1		1	≤3
6															4			−1	1	≤3

The optimum solution is

$$u_2 = 0, u_3 = 2, u_4 = 3, x_{12} = x_{23} = x_{34} = x_{41} = 1, \text{tour length} = 59.$$

This corresponds to the tour solution 1-2-3-4-1. The solution satisfies all the additional constraints in *uj* (verify!).

To demonstrate that sub tour solutions do not satisfy the additional constraints, consider (1-2-1,3-4-3), which corresponds to $x_{12} = x_{21} = 1, x_{34} = x_{43} = 1$. Now, consider constraint 6 in the tableau above:

$$4x_{43} + u_4 - u_3 \leq 3$$

Substituting $x_{43} = 1$, $u_3 = 2$, $u_4 = 3$ yields $5 \leq 3$, which is impossible, thus disallowing $x_{43} = 1$and sub tour 3-4-3.

The disadvantage of the cutting-plane model is that the number of variables grows exponentially with the number of cities, making it difficult to obtain a numeric solution for practical situations. For this reason, the B&B algorithm (coupled with the heuristic) may be a more feasible alternative for solving the problem.

Unit IV

Inventory Theory: Basic Elements of an Inventory Model-Deterministic Models: Single item Stock Model with and without Price Breaks-Multiple Items Stock Model with Storage Limitations- Probabilistic Models: Continuous Review Model-Single Period Models.

Chapter 4 - Sections 4.1- 4.9

4.1. Basic Elements of an Inventory Model:

An inventory means a physical stock of idle resources of any kind having some economic value kept for the purpose of meeting future demand. It indicates the raw material required before production, the finished goods afterproduction ready for delivery to consumers, human resources, financial resources, etc., which are stocked in order to meet an expected demand in the future. Almost every business must maintain an inventory for running its operations efficiently and smoothly. If an enterprise does not maintain an inventory, it may suddenly find at some point in its operations that it has no materials or goods to supply to its customers. Then on receiving a sales order,it will first have to place order for purchase of raw materials, wait for their receipt and then start production. The customer will, thus, have to wait for a long time for the delivery of the goods and may turn to other suppliers, resulting in loss of business/goodwill for the enterprise.

Maintaining an inventory is necessary because of the following reasons:

i) It helps in smooth and efficient running of an enterprise.

ii) It provides service to the customer at short notice. Timely delivery canfetch more goodwill and orders.

iii) In the absence of the inventory, an enterprise may have to pay high prices because of piecemeal purchasing. Maintaining an inventory may earn pricediscounts because of bulk purchasing. Such purchases entail less orders and, therefore, less clerical costs.

iv) It also takes advantage of favourable market.

v) It acts as a buffer stock when raw materials are received late and shop rejections are too many.

vi) Process and movement inventories (also called pipeline stocks) are quite necessary in big enterprises wherein a significant amount of time is required to ship items from one location to another.

Though inventories are essential, their maintenance also costs money by wayof expenses on stores, equipment, personnel, insurance, etc. Thus, excess inventories are undesirable. So, only that quantity should be kept in stock, which balances the costs of holding too much stock vis-à-vis the costs of ordering in small quantities. This calls for controlling the inventories in the most profitable way and that is why we need inventory analysis. We now discuss various factors involved in inventory analysis.

Factors Influencing Inventories:

The major problem of inventory control is to answer two questions:

1. How much to order?
2. When to order?

These are answered by developing a model. An inventory model is based on the consideration of the mainaspects of inventory. The varieties of factors related to these are placed below:

1. **Inventory related costs**

Various costs associated with inventory control are often classified asfollows:

i) **Set-up cost**: This is the cost associated with the setting up of machinery before starting production. The set-up cost is generallyassumed to be independent of the quantity ordered for.

ii) **Ordering cost**: This is the cost incurred each time an order is placed.This cost includes the administrative costs (paper work, telephone calls, postage), transportation, receiving and inspection of goods, etc.

iii) **Purchase (or production) cost**: It is the actual price at which an itemis purchased (or produced). It may be constant or variable. It becomesvariable when quantity discounts are allowed for purchases above a certain quantity

iv) **Carrying (or holding) cost**: The cost includes the following costs for maintaining the inventory: i) Rent for the space; ii) cost of equipment or any other special arrangement for storage; iii) interest of the moneyblocked; iv) the expenses on stationery; v) wages of the staff required for the purpose; vi) insurance and depreciation; and vii) deterioration and obsolescence, etc.

v) **Shortage (or Stock-out) cost**: This is the penalty cost for running out of stock, i.e., when an item cannot be supplied on the customer's demand. These costs include the loss of potential profit through sales of items demanded and loss of goodwill in terms of permanent loss of the customer

2. Demand

Demand is the number of units required per period and may either be known exactly or known in terms of probabilities. Problems in which demand is known and fixed are called **deterministic problems** whereasproblems in which demand is known in terms of probabilities are called **probabilistic problems.**

3. Selling Price

The amount which one gets on selling an item is called its selling price.The unit selling price may be constant or variable, depending upon whether quantity discount is allowed or not.

4. Order Cycle

The period between placement of two successive orders is referred to asan order cycle. The order may be placed on the basis of either of the following two types of inventory review systems:

a) The record of the inventory level is checked continuously until a specified point is reached where a new order is placed. This is calledcontinuous review.

b) The inventory levels are reviewed at equal intervals of time and ordersare placed accordingly at such levels. This is called periodic review

5. Time Horizon

The period over which the time cost will be minimized and inventory level will be controlled is termed as time horizon. This can be finite orinfinite depending on the nature of demand.

6. Stock Replenishment

The rate at which items are added to the inventory is called the rate of replenishment. The actual replenishment of items may occur at a uniform rate or be instantaneous over time. Usually uniform replacement occurs incases when the item is manufactured within the factory while instantaneous replacement occurs in cases when the items are purchased from outside sources.

7. **Lead Time**

The time gap between placing an order for an item and actually receivingthe item into the inventory is referred to as lead time.

8. **Reorder Level**

The lower limit for the stock is fixed at which the purchasing activitiesmust be started for replenishment. With this replenishment, the stock reached at a level is known as maximum stock. The level between maximum and minimum stock is known as the reorder level.

9. **Economic Order Quantity (EOQ)**

The order in quantity that balances the costs of holding too much stock vis-à-vis the costs of ordering in small quantities too frequently is calledEconomic Order Quantity (or Economic lot size).

10. **Reorder Quantity**

The quantity ordered at the level of minimum stock is known as the reorder quantity. In certain cases it is the 'Economic Order Quantity'.

we shall discuss the following inventory models forobtaining economic order quantity:

i) EOQ Model with Uniform Demand

ii) EOQ Model with Different Rates of Demand in Different Cycles

iii) EOQ Model when Shortages are Allowed

iv) EOQ Model with Uniform Replenishment

v) EOQ Model with Price (or Quantity) Discounts

However, before discussing these models, we give the notations that we shalluse in the development of the models.

The notation used in the Models

Q= Number of units ordered (supplied) per order

D= Demand in units of inventory per year

N= Number of orders placed per year

TC = Total Inventory cost

C_O = Ordering cost per order

C = Purchase or manufacturing price per unit inventory

C_h = Carrying or holding cost per unit per period of time the inventory is kept

C_s = Shortage cost per unit of inventory

t = The elapsed time between placement of two successive orders

r_p = Replenishment rate at which lot size Q is added to inventory.

4.2. Economic Order Quantity (EOQ) Model With Uniform Demand:

The objective of the **EOQ model with uniform demand** is to determine anoptimum economic order quantity such that the total inventory cost is minimized. We make the following assumptions for this model:

1. Demand rate (D) is constant and known;
2. Replenishment rate (r_p) is instantaneous;
3. Lead time is constant and zero;
4. Purchase price is constant, i.e., discounts are not allowed;
5. Carrying cost and ordering cost are known and constant; and
6. Shortages are not allowed.

The situation can be graphically represented as shown in Figure 4.1.

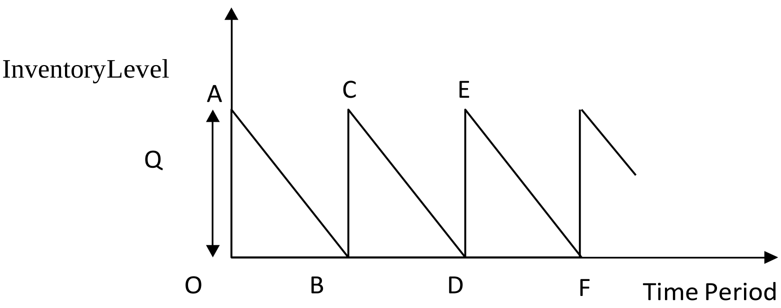


Figure 4.1.

The graph in Fig. 4.1 shows that initially there were Q units in the stock. The number of units goes on decreasing with respect to time to meet the demand and this is represented by the line AB in the graph. When the stock vanishes, i.e., the point B is reached, the stock level rises to Q instantaneously as the replenishment is instantaneous. Since the demand is uniform, the rate of decrease of the quantity remains the same as earlier. Therefore, this is represented by the line CD in the graph, which is parallel to AB. Similarly, EFis parallel to AB and CD due to uniform demand, and so on.

Since the demand is uniform, the average inventory is simply the arithmeticmean of the maximum and the minimum levels of inventory. Let Q be the quantity ordered (or replenished) when the minimum level, i.e., zero is reached.

$$= \frac{1}{2} \text{(maximum level + minimum level)} = \frac{1}{2}(Q+O) = \frac{Q}{2}$$

Since average inventory during any cycle period is $\frac{Q}{2}$, the average inventory during the entire period is also $\frac{Q}{2}$.

So, carrying cost = average units in inventory × carrying cost per unit $= \frac{Q}{2}C_h$

Ordering cost = number of orders × ordering cost per order

$$= N \times C_o$$

$$= \frac{D}{Q}C_o$$

Total variable inventory cost is then given by

$$TC = \frac{D}{Q}C_o + \frac{Q}{2}C_h$$

Differentiating w.r.t. Q, we get

$$\frac{d}{dQ}(TC) = -\frac{DC_o}{Q^2} + \frac{C_h}{2}$$

$$\frac{d^2}{dQ^2}(TC) = \frac{2DC_o}{Q^3}$$

$$\frac{d}{dQ}(TC) = -\frac{DC_o}{Q^2} + \frac{C_h}{2}$$

$$\frac{d^2}{dQ^2}(TC) = \frac{2DC_o}{Q^3}$$

For maxima or minima,

$$\frac{d}{dQ}(TC) = 0$$

i.e. $$-\frac{DC_o}{Q^2} + \frac{C_h}{2} = 0$$

i.e. $$\frac{C_o D}{Q} = \frac{C_h Q}{2}$$

i.e. $$Q = \sqrt{\frac{2DC_o}{C_h}}$$

At $Q = \sqrt{\frac{2DC_o}{C_h}}, \frac{d^2}{dQ^2}(TC) > 0$

Hence TC is minimum when $Q = \sqrt{\frac{2DC_o}{C_h}}$.

∴ EOQ is given as $Q^* = \sqrt{\frac{2DC_0}{C_h}}$

Optimum number of orders placed per time period $(N^*) = \frac{D}{Q^*}$

Minimum total variable inventory cost

$$= \frac{D}{Q^*}C_o + \frac{Q^*}{r^2}C_h = \sqrt{2DC_oC_h}$$

$$= \sqrt{2 \times \text{demand rate} \times \text{ordering cost} \times \text{holding cost}}$$

Optimum length of time between orders $= \frac{Q^*}{D}$

$$= \frac{1}{D}\sqrt{\frac{2DC_o}{C_h}} = \sqrt{\frac{2C_o}{DC_h}}$$

Note: If the carrying cost is given as a percentage of average value of inventory held, then total annual carrying cost C_h may be expressed as

$C_h =$ cost of one unit × inventory carrying cost in percentage = CI

Hence optimum order quantity is given by

$$Q^* = \sqrt{\frac{2DC_o}{CI}}$$

Example 1:

An enterprise requires 1000 units per month. The ordering cost is estimated to be ` 50 per order. In addition to ` 1, the carrying costs are 5% per unit of average inventory per year. The purchase price is ` 20 per unit. Find theeconomic lot size to be ordered and the total minimum cost.

Solution:

We are given that

D = Monthly demand $\times 12$

= 1000 × 12 = 12000 units per year, $C_O = 50$ per order;

C = 20 per unit $C_h = 1 + 5\%$ of 20

= 1 + 1 = 2 per unit of average inventoryThe economic lot size,

therefore, is given by $Q^* = \sqrt{\frac{2D\,C_0}{C_h}} = \sqrt{\frac{2\times12000\times50}{2}} = 775\ units$

Total minimum cost = TC* + Cost of material

$= \sqrt{2D\ C_0 C_h} + (12000 \times 20)$

$= \sqrt{2 \times 12000 \times 50 \times 2} + 240000$

$= \sqrt{2400000}$ +240000

$= \sqrt{240} \times 100 + 240000$

$= 15.5 \times 100 + 240000$

=241550

You may now like to solve the following problems to assess yourunderstanding.

Example 2:

The demand rate of a particular item is 12000 units per year. The set-up cost per runis Rs. 350 and the holding cost is Rs.20 per unit per month. If no shortages are allowed and the replacement is instantaneous, determine

(i)the optimum run size,

(ii)the optimum scheduling period, and

(iii) minimum total expected annual cost.

Solution:

Here, D= 12000 per year, C_O = Rs.. 350,

C_h=Rs. 0.2 per unit per month= Rs. 2.4 per unit per year

(i) Optimum lot size $= Q^* = \sqrt{\frac{2DC_o}{C_h}} = \sqrt{\frac{2\times12000\times350}{2.4}}$

$= \sqrt{3500000} = 1870.8 \simeq 1871$ units

(ii) Optimum scheduling period

$= t^* = \frac{Q^*}{D} = \frac{1871}{12000}$ years $= 1.87$ months

(iii) minimum total expected annual cost $= \sqrt{2DC_oC_h}$

$= \sqrt{2\times12000\times350\times2.4}$

$= \sqrt{20160000} =$ Rs. 4490

Example 3:

The annual requirement for a product is 3000 units. The ordering cost is Rs. 100 perorder. The cost per unit is Rs. 10. The carrying cost per unit per year is 50% of the unit cost. Find the EOQ. If a new EOQ is found by using ordering cost as Rs. 80, what would be thefurther savings in cost?

Solution:

Here D = 300 units per year

C_o = Rs. 100

C = Rs. 10, I = 30%

$C_h = CI = 10 \times \frac{30}{100}$ = Rs. 3 per unit per year.

Optimal lot size, $Q^* = \sqrt{\frac{2DC_o}{C_h}} = \sqrt{\frac{2 \times 3000 \times 100}{3}}$ = 447 units

Total inventory cost $= \sqrt{2DC_0C_h}$

$= \sqrt{2 \times 3000 \times 100 \times 3}$ = Rs. 1342 per year

In the second part, we have D = 3000, C_o = Rs. 100/80, C_h = Rs. 3 per unit per year

$Q^* = \sqrt{\frac{2DC_0}{C_h}} = \sqrt{\frac{2 \times 3000 \times 80}{2}}$ = 400 units.

Total inventory cost $= \sqrt{2DC_oC_h} = \sqrt{2 \times 3000 \times 80 \times 3}$ = Rs. 1200 per year

Net savings=1342-1200=Rs.142 per year.

Example 4:

A company requires 1000 units per month. Order cost is estimated to be Rs. 50 per order. In addition to Re 1.00, the carrying costs are 10% per unit of average inventory per year. The purchase price is Rs. 10 per unit. Find the economic lot size to be ordered and the total minimum cost.

Solution. Here D = 1000 units per month

= 12000 units per year

C_0 = Rs. 50 per order

C = Rs. 10 per unit

C_h = 1.00 + 10% of Rs. 10

= Rs. 2 per unit of average inventory

The economic lot size is given by

$Q^* = \sqrt{\frac{2DC_o}{C_h}} = \sqrt{\frac{2 \times 12000 \times 50}{2}}$ = 775 units

Total minimum cost = Total minimum inventory cost + Cost of material

$= \sqrt{2DC_oC_h} + 12000 \times 10$

$= \sqrt{2 \times 12000 \times 50 \times 2} + 1,20,000$

= 1549 + 1,20,000 = Rs. 1,21,549.

Example 5:

The XYZ manufacturing company has determined from an analysis of its accountingand production data for 'part number alpha', that its cost to purchase is Rs. 36 per order and Rs. 2 per part. Its inventory carrying charge is 9 per cent of the average inventory. The demand of this part is 10,000 units per annum. Determine

i)What should be the economic order quantity?

ii)What is the optimum number of orders?

iii)What is the optimum number of days supply per optimum order

Solution:

Here, demand per annum, D = 10,000

Ordering cost per order, C_0= Rs. 36

Cost of one part, C=Rs. 2, I = 9%

Inventory carrying cost C_h =Rs 0.09 x 2=Rs. 0.18

Total inventory cost, TC=Ordering cost + Carrying cost

$$=\left(\frac{10000}{Q}\right)36+0.09\,Q$$

Differentiate w.r.t. Q,

$$\frac{d(TC)}{dQ}=-\frac{360000}{Q^2}+0.09$$

Equating it to zero, we have $-\frac{360000}{Q^2}+0.09=0$

$$\therefore Q^2=\frac{360000}{0.09}=4000000$$

$$\Rightarrow Q=2000$$

$$\frac{d^2(TC)}{dQ^2}=\frac{720000}{Q^3}$$

At $Q=2000$, $\frac{d^2(TC)}{dQ^2}=\frac{720000}{(2000)^3}>0$

$\therefore TC$ is minimum when $Q=2000$ units

Thus, EOQ is $Q^*=2000$ units .

(ii) Optimal number of orders $=\frac{\text{Demand}}{\text{EOQ}}=\frac{10000}{2000}=5$

(iii) Optimal number of days supply per optimum order $=\frac{365}{5}=73$ days

Exercises:

1. A company uses annually 12000 units of raw material costing Rs. 1.25 per unit. Placing each order costs Rs. 15 and the carrying costs are 15% per year per unit of average inventory. Find the economic order quantity?

Answer. $Q^* = \sqrt{\frac{2DC_o}{C_h}} = \sqrt{\frac{2\times12000\times15}{0.15\times1.25}}$ = 1385 units.

2. A manufacturer uses Rs. 10,000 worth of an item during the year. He has estimated the ordering costs is Rs. 25 per order and carrying costs as 12.5% of average inventory value. Find the optimal order size, number of orders per year, time period per order and total cost.

Answer. $Q^* = \sqrt{\frac{2DC_o}{C_h}} = \sqrt{\frac{2\times10000\times25}{0.125}}$ = Rs. 2000

Number of orders per year $= \frac{D}{Q^*} = 5$

Time period per order $= \frac{365}{5} = 73$ days

Total cost, $TC^* = \sqrt{2\cdot DC_oC_h} = \sqrt{2\times10000\times25\times0.125}$ = Rs. 250 (variable cost)

Total annual cost = Rs. 10000 + Rs. 250= Rs. 10,250

Model II:

4.3. EOQ with Finite Rate of Replenishment or EOQ with Uniform Replenishment

For this model, it is assumed that the production run may take a significant time to complete. Let r¢ bethe demand rate in units per unit of time and q be the replenishment rate per unit time. Assume that each cycle time › of two parts t, and 2 such that

(a) production is continuous and constant until *Q* units are produced to stock, then it stops;

(b) the production rate p is greater than demand rater,

(c) there is no replenishment (or production) during time t2 and the inventory is decreasing at the rater¢ per unit of time.

Let Q be the number of units produced per order cycle. Then, $t_1 = \frac{Q}{r_p}$

Inventory is building up at the rate of $(r_p - r_d)$

Maximum inventory level $= (r_p - r_d)t$

Minimum inventory level = 0

Average inventory $= \frac{1}{2}[(r_p r_d)t_1 + 0] = \frac{(r_p - r_d)Q}{2r_p} = \frac{Q}{2}\left(1 - \frac{r_d}{r_p}\right)$

Ordering cost (or set up cost) $= \frac{D}{Q} C_o$

Carrying cost $= \frac{Q}{2}\left(1 - \frac{r_d}{r_p}\right) C_h$

Total inventory cost is

$$TC = \frac{D}{Q}C_o + \frac{Q}{2}\left(1 - \frac{r_d}{r_p}\right)C_h$$

This cost will be minimum if

$$\frac{D}{Q}C_o = \frac{Q}{2}\left(1 - \frac{r_d}{r_p}\right)C_h$$

$$\frac{d}{dQ}(TC) = \frac{-D}{Q^2}C_0 + \frac{1}{2}\left(1 - \frac{r_d}{r_p}\right)$$

$$\frac{D}{Q^2}C_o = \frac{1}{2}\left(1 - \frac{r_d}{r_p}\right)C_h$$

i.e. $$Q^2 = \frac{2DC_o}{C_h}\left(\frac{r_p}{r_p - r_d}\right)$$

i.e. $$Q^* = \sqrt{\frac{2DC_o}{C_h}\left(\frac{r_p}{r_p - r_d}\right)}$$

Characteristics of the model:

1. Optimum length of each lot size production run

$$t_1^* = \frac{Q^*}{r_p} = \sqrt{\frac{2DC_o}{C_h r_p (r_p - r_d)}}$$

2. Optimum number of production runs per year

$$N^* = \frac{D}{Q^*} = \sqrt{\frac{DC_h (r_p - r_d)}{2C_o r_p}}$$

3. Optimum production cycle time, $t^* = \frac{Q^*}{D}$

4. Total minimum inventory cost

$$TC^* = \frac{D}{Q^*} C_o + \frac{Q^*}{2}\left(1 - \frac{r_d}{r_p}\right) C_h$$

$$= \frac{DC_0 \sqrt{C_h (r_p - r_d)}}{\sqrt{2DC_0 r_p}} + \frac{1}{2}\sqrt{\frac{2DC_o}{C_h} \cdot \frac{r_p}{r_p - r_d}} - \left(1 - \frac{r_d}{r_p}\right) C_h$$

$$= \sqrt{2DC_o C_h \left(1 - \frac{r_d}{r_p}\right)}$$

Example 1:

A tyre producer makes 1200 tyres per day and sells them at approximately half that rate. Accounting figures show that the production set up cost is Rs. 1000 and carrying cost per unit is Rs.5. If annual demand is 120000 tyres, what is the optimal lot size and how many production runs shouldbe scheduled per year?

Solution:

Annual demand, D=12000 tyres

C_h=Rs.5

C_0=Rs.1000

Production rate, r_p=1200 tyres per day

Demand rate, r_d= 600 tyres per day

Optimal lot size, $Q^* = \sqrt{\frac{2DC_o}{C_h} \cdot \frac{rp}{r_p - r_d}}$

$$= \sqrt{\frac{2 \times 120000 \times 1000}{5} \times \frac{1200}{1200 - 600}}$$

$$= 2000\sqrt{24} = 9797.96 = 9798 \text{ tyres.}$$

Optimal number of production runs per year

$$N^* = \frac{D}{Q^*} = \frac{120000}{9798} \simeq 13 \text{ runs/year}$$

Example 2:

A contractor has to supply 10000 paper cones per day to a textile unit. He finds thatwhen he starts production run, he can produce 25000 paper cones per day. The cost of holding a paper cone in stock for one year is 2 paisa and the set up cost of production run is Rs. 18. How frequently should production run be made?

Solution. Here r_d = 10000 paper cones per day

r_p= 25000 paper cones per day

$D = 10000 \times 365 = 3{,}65{,}0000$ cones

C_H= Rs. 0.02 per paper cone per year

C_o= Rs. 18

Now, $$Q^* = \sqrt{\frac{2DC_o}{C_h}\left(\frac{r_p}{r_p - r_d}\right)} = \sqrt{\frac{2 \times 3650000}{0.02} \times \left(\frac{25000}{25000 - 10000}\right) \times 18}$$

$$= \sqrt{3650000 \times 100 \times \frac{25}{15} \times 18} = 104642 \text{ paper cones}$$

Frequency of production runs is given by

$$t^* = \frac{Q^*}{r_d} = \frac{104642}{10000} = 10.46 \text{ days}$$

Thus, production run can be made after every 10.46 days

Exercises:

1. A product is produced at the rate of 50 items per day. The demand occurs at the rate of 30 items per day. Given that set up cost per order is Rs. 1000 and holding cost per unit time is Rs. 0.05. Find the economic let size and the associated total cost per cycle assuming that no shortage is allowed.

Answer. $Q^* = 1732$ units and total inventory cost = Rs. 34.64

2. The annual demand for a product is 100000 units. The rate of production is 200000 units per year. The set up cost per production run is Rs. 500 and the variable production cost of each item is Rs. 10. The annual holding cost per unit is 20% of its value. Find the optimum production lot size and the length of the production run.

Answer. $Q^* = 10000$ units, $t_1^* = \frac{10000}{200000} = 0.05$years

3. An item is produced at the rate of 50 items per day. The demand occurs at the rate of 25 items per day. If the set-up cost is Rs. 100 per set up and holding cost is Re 0.01 per unit of item per day, find the economic lot size for one run, assuming that shortages are not allowed. Also find the time of cycle, length of each production run, minimum total cost per day and maximum inventory level.

Answer. $Q^* = 1000$ items, Cycle time, $t_l{}^* = 40$ days

Length of each production run, $t_1{}^* = 20$ days, Maximum inventory level = 500 items

Minimum daily cost = Rs. 5

4. A company uses 100000 units of a particular item per year. Each item costs Rs. 2. The production engineering department estimates set up cost at Rs. 25 and the accounting department estimates the holding cost as 12.5% of the value of the inventory per day.

Replenishment rate is uniform 500 units per day.

Assuming 250 working days (per replenishment purpose), calculate

(a) Optimal setup quantity

(b) Total cost on the basis of optimal policy

(c) Optimal number of set up

Answer. (a) $Q^* = 10000$ units

(b) Total minimum cost = $TC^* + 200000 = 200500$

(c) Optimum number of set ups = $\frac{D}{Q^*} = 10$

Model III:

4.4. Economic Order Quantity with Different Rates of Demands in Different Cycles

Here the stock will vanish at different time periods with a policy of ordering same quantity forreplenishment of inventory.

Here replenishment rate is infinite, replenishment is instantaneous and shortage is not allowed. The total demand D is specified as demand during total time period T and stock level Q is fixed.

Number of production cycles, $n = \frac{D}{Q}$

Let the demand in different time periods be $D = D_1, D_{2,\ldots\ldots\ldots} D_n$ respectively so that total demand in time T is

$D = D_1 + D_{2,} + \cdots .. D_n$

Where, $T = t_1 + t_2 + \cdots + t_n$

Cost of ordering in time T is given by $\frac{D}{Q} C_o$

Let C_h be the holding cost per item per unit time

Then carrying cost for time T is

$$\frac{Qt_1}{r} C_h + \frac{Qt_2}{2} C_h + \ldots + \frac{Qt_n}{2} C_h$$

$$= \frac{Q}{2} C_h (t_1 + t_2 + \ldots + t_n) = \frac{Q}{2} C_h T$$

Total inventory cost, $TC = \frac{DC_0}{Q} + \frac{Q}{2} C_h T$

Total cost is minimum, when $\frac{DC_0}{Q} = \frac{Q}{2} C_h T$

i.e. $Q = \sqrt{\frac{2DC_o}{C_h T}} = \sqrt{\frac{2C_o}{C_h} \cdot \frac{D}{T}}$

$\therefore Q^* = \sqrt{\frac{2C_o}{C_h} \cdot \frac{D}{T}}$

This result is similar to the result of Model I with the only different that uniform demand is replaced by average demand.

Here, $TC^* = \sqrt{\frac{2D}{T} C_0 C_h}$

and total minimum cost $= \sqrt{\frac{2D}{T} C_0 C_h}$ + cost of material

Remark: If $T = 1$ year, then results of this model are exactly same as that of model I.

Model IV:

4.5. Economic Order Quantity when Shortages are Allowed

The assumptions of this model are same as that of model I except that shortages are allowed and shortages may occur regularly. Let C_S be the shortage cost per unit of time per unit quantity.

Ordering cost $= \frac{D}{Q} C_o$

Total inventory over the time period, $t = \frac{1}{2} M t_1$

Average inventory at any time $= \frac{1}{2} M t_1 / t$

Inventory holding cost $= \frac{M t_1}{2t} C_h$

Total amount of shortage over time period $t = \frac{1}{2} S t_2$

Average shortage at any time $= \frac{1}{2} \frac{S t_2}{t}$

Shortage cost $= \frac{1}{2} \frac{S t_2}{t^2} C_s$

From (1),

$$-2DC_0 - M^2 C_h + C_s\left(2Q^2 - 2QM - Q^2 + 2QM - M^2\right) = 0$$

i.e., $C_s Q^2 - M^2 (C_h + C_s) - 2DC_o = 0$

i.e., $C_s Q^2 - \frac{Q^2 C_s^2}{(C_h + C_s)^2}(C_h + C_s) = 2DC_o$ [using (3)]

i.e. $C_s Q^2 - \frac{Q^2 C_s^2}{C_h + C_s} = 2DC_o$

i.e. $Q^2 \left(\frac{C_s C_h + C_s^2 - C_s^2}{C_h + C_s}\right) = 2DC_o$

i.e. $Q^2 = \frac{2DC_o}{C_3 C_h}(C_3 + C_h)$

$$\therefore Q^* = \sqrt{\frac{2DC_o}{C_h} \cdot \frac{C_h + C_s}{C_s}}$$

and $M^* = \sqrt{\frac{2DC_0}{C_h} \cdot \frac{C_s}{C_h + C_s}}$

Total minimum cost

$$= \frac{D}{Q^*} C_0 + \frac{M^{*2}}{2Q^*} C_h + \frac{(Q^* - M^*)}{2Q^*} C_s$$

$$= \sqrt{2DC_o C_h \frac{C_s}{C_h + C_s}}$$

Total cost, $TC = \frac{D}{Q} C_o + \frac{1}{2} M \frac{t_1}{t} C_h + \frac{1}{2} S \frac{t_2}{t} C_s$

Using the relationship for similar triangles, we have

$$\frac{t_1}{t} = \frac{M}{Q} \text{ and } \frac{t_2}{t} = \frac{S}{Q}$$

$$t_1 = \frac{M}{Q} t \text{ and } t_2 = \frac{S}{Q} t$$

So $\quad TC = \frac{D}{Q} C_o + \frac{1}{2} \frac{M^2}{Q} C_h + \frac{(Q-M)^2}{2Q} C_s$ since $S = a - M$

Since TC is function of two variables Q and M, so in order to determine the optimal order size and optimal shortage level, we put $\frac{\partial}{\partial Q}(TC)$ and $\frac{\partial}{\partial M}(TC)$, both equal to zero so that

$$-\frac{DC_o}{Q^2} - \frac{M^2 C_h}{2Q^2} + \frac{C_s}{2}\left(\frac{Q \cdot 2(Q-M) - (Q-M)^2 \cdot 1}{Q^2}\right) = 0 \qquad (1)$$

and

$$\frac{M}{Q} C_h + \frac{2C_s}{2Q}(Q-M)(-1) = 0 \qquad (2)$$

From (2),

$$MC_h + C_s(-Q) + MC_s = 0$$

$$\Rightarrow M = Q\left(\frac{C_s}{C_h + C_s}\right) \qquad (3)$$

Example 1:

A contractor undertakes to supply Diesel engines to a truck manufacturer at a rate of 20 engines per day. The penalty in the contract is Rs. 100 per engine per day late for missing the scheduled delivery date. The cost of holding an engine in stock for one month is Rs. 150. His production process is such that each month (30 days) he starts producing a batch of engines through the shops and all these are available for supply after the end of the month. Determine the maximum inventory level at the beginning of each month.

Solution. Here, $C_h = \text{Rs.}\frac{150}{30}$ per engine per day= Rs. 5 per engine per day

C_s = Rs. 100 per engine per day

$D = 20$ engines per day

$t^* = 30$ days

∴ Optimum inventory level at the beginning of each month

$$= M^* = \frac{C_s}{C_h + C_s}, Q^* = \frac{C_s}{C_h + C_s} \cdot Dt^* = \frac{100}{5+100} \times 20 \times 30 = 571 \text{ engines}$$

Example 2:

A dealer supplies you the following information with regard to a product dealt in by him.

Annual demand =10,000 units Ordering cost = Rs.10 per order, Price=Rs.20per unit Inventory carrying cost = 20% of the value of inventory per year

The dealer is considering the possibility of allowing some back order (stock ordered) to occur. He hasestimated that the annual cost of back ordering will be 25% of the value of inventory.

(i) What is the optimal number of units of product he should buy in one lot?

(ii) What quantity of product should be allowed to be back ordered, if any?

(iii) What would be maximum quantity of inventory at any time of the year?

(iv) Would you recommended to allow back-ordering? If so, what would be the annual cost savingby adopting the policy of back ordering?

Sol. Here, D = 10,000 units

C_o = Rs. 10 per order

C_h = 20% of Rs. 20 = Rs. 4 per unit per year

C_s = 25% of Rs. 20 = Rs. Per unit per year

(i) (a) When stock outs are not permitted:

$$Q^* = \sqrt{\frac{2DC_o}{C_h}} = \sqrt{\frac{2 \times 10000 \times 10}{4}} = 223.6 \text{ units}$$

(b) When back ordering is permitted

$$Q^* = \sqrt{\frac{2DC_o}{C_h} \cdot \frac{C_h + C_s}{C_s}}$$

$$= \sqrt{\frac{2 \times 10000 \times 10}{4}\left(\frac{4+5}{5}\right)} = \sqrt{90000} = 300 \text{ units}$$

(ii) Optimal quantity of the product to be back-ordered is given by

$$S^* = Q^*\left(\frac{C_h}{C_h + C_s}\right)$$

$$300\left(\frac{4}{4+5}\right) = 133 \text{ units}$$

(iii) Maximum inventory level, $M^* = Q^* - S^* = 300 - 133 = 167$ units

(iv) Minimum total variable inventory cost in case shortages is not allowed

$$= TC(223.6) = \sqrt{2DC_0C_h}$$

$$= \sqrt{2 \times 10000 \times 10 \times 4} = \text{Rs. } 894.43$$

Minimum total variable inventory cost in case shortage is allowed

$= TC$ (300)

$$= \sqrt{2DC_oC_h \cdot \frac{C_s}{C_h + C_s}} = \sqrt{2 \times 10,000 \times 10 \times 4 \cdot \left(\frac{5}{4+5}\right)}$$

$= $ Rs. 666.67

Since *TC* (300) <*TC* (223.6), the dealer should accept the proposal for back ordering as this will result in asaving of Rs. (894.43 -666.67) = Rs. 22776 per year.

Exercise.

A contractor undertakes to supply diesel engines to a trick manufacturer at a rate of 25 engines per day. He finds that the cost of holding a completed engine in stock is Rs. 16 per month and there is a clause in the contract penalizing him Rs. 10 per engine per day late for missing the scheduled delivery date production of engines is in batches and each time a new batch is started, there are setup costsof Rs. 10,000. How frequently should batches be started and what should be the initial inventory level at the time, each batch is completed?

Solution. $Q^* = \sqrt{\frac{2DC_o}{C_h} \cdot \frac{C_h + C_s}{C_s}} = \sqrt{\frac{2 \times 25 \times 10,000}{\frac{11}{30}} \left(\frac{\frac{16}{30} + 10}{10}\right)}$ =994 units (app.)

$$t^* = \frac{Q^*}{D} = \frac{994}{25} = 39.76 \simeq 40 \text{ days}$$

4.6. Problems of EOQ with Price Breaks

In the real world, it is not always true that the unit cost of an item is independent of the quantity procured.Often discounts are offered for the purchase of large quantities. These discounts take the form of price breaks.

Let us consider a manufacturer, who is encountered with a problem of determining an optimum production quantity for each production run and an optimal interval between successive runs. The following conditions are assumed to hold

(i) Demand is known and uniform

(ii) Shortages are not allowed

(iii) Production for supply commodities is instantaneous

Let Q be the lot size in each production run, D, total number of units produced or supplied overthe time period, C_0 the cost per production run, R cost of manufacturing (or purchasing) per unit and I
inventory carrying charge expressed as a % of the value of the average inventory. Then total cost is givenby

$$TC = \text{Purchase cost + Holding cost + Ordering cost}$$

$$= Dk + \frac{1}{2}QkI + \frac{D}{Q}C_o$$

$$\frac{d(TC)}{dQ} = \frac{1}{2}kI - \frac{DC_o}{Q^*}$$

$$\frac{d^2}{dQ^2}(TC) = \frac{2DC_0}{Q^3}$$

For maxima or minima, $\frac{d}{dQ}(TC) = 0$

i.e., $\frac{1}{2}kI - \frac{DC_0}{Q^2} = 0$

i.e., $Q = \sqrt{\frac{2DC_o}{kI}}$

At $Q = \sqrt{\frac{2DC_o}{kI}}, \frac{d^2(TC)}{dQ^2} > 0$

Hence total cost is minimum when $Q = \sqrt{\frac{2DC_o}{kI}}$

So total minimum cost, $TC(Q^*) = Dk + \frac{1}{2}Q^*kI + \frac{D}{Q^*}C_o$

$$= Dk + \frac{1}{2}\sqrt{\frac{2DC_o}{kI}}kI + \frac{DC_o\sqrt{kI}}{\sqrt{2DC_o}}$$

$$= Dk + \sqrt{2DC_okI}$$

4.6.1. Purchase Inventory Model with One Price Break:

The purchase inventory model with only one price break may be represented as follows:

Range of quantity	Purchase cost (per unit)
$0 \le Q < b$	P_1
$b \le Q$	P_2

Where b is the quantity at and beyond which the quantity discount applies and $P_2 < P_1$

The procedure for obtaining EOQ may be summarized in the following steps:

Step 1. Calculate optimum order quantity Q_2^* for the lowest price (highest discount) i.e.

$$Q_2^o = \sqrt{\frac{2DC_o}{P_2 I}}$$

and compare it with the quantity b

If $Q_2^o < b$, calculate optimum order quantity Q_1^o for price P_1 and compare total inventory cost for $Q = Q_1^o$ with $Q = b$ which given by

$$TC\left(Q_1^o\right) = DP_1 + \frac{D}{Q_1^o}C_o + \frac{Q_1^o}{2}IP_1$$

$$TC(b) = DP_2 + \frac{D}{b}C_o + \frac{b}{2}IP_2$$

If $TC\left(Q_1^o\right) > TC(b)$, then $Q^o = b$ otherwise $Q^o = Q_1^o$.

Example 1:

Find the optimum order quantity for a product for which the price breaks are as follows:

Quantity	Unit cost (Rs.)
$0 \le Q < 500$	10.00
$500 \le Q$	9.25

The monthly demand for the product is 200 units, the cost of storage is 2% of the unit cost and ordering cost is Rs. 350.

Solution. We are given

C_o = Rs. 350

D = 200 units per month

$I = 2\% = 0.02$

P_1 = Rs. 10

P_2 = Rs. 9.25

Highest discount available is Rs. 9.25. So we compute Q_2^o as

$$Q_2^o = \sqrt{\frac{2DC_o}{IP_2}} = \sqrt{\frac{2 \times 200 \times 350}{0.02 \times 9.25}} = 870 \text{ units}$$

Since $Q_2^o > b = 500$, the optimum purchase quantity is given by $Q^o = Q_2^o$ = 870 units.

Example 2:

Find the optimal order quantity for a product having the following characteristics:

Annual demand = 2400 units

Ordering cost = Rs. 100

Cost of storage = 24% of the unit cost

Price break

Quantity	Unit cost (Rs.)
$0 \le Q < 500$	10
$500 \le Q$	9

Solution. We have $D = 2400$ units per year

$I = 0.24$

C_o = Rs. 100

P_1 = Rs. 10

P_2 = Rs. 9

Highest discount available is Rs. 9, so we compute Q_2^* as

$$Q_s^* = \sqrt{\frac{2DC_o}{P_2 I}} = \sqrt{\frac{2\times 2400\times 100}{9\times 0.24}} = 471 \text{ units}$$

Since $Q_2^* < 500$, Q_2^* is not feasible

We calculate Q_1^* as

$$Q_1^* = \sqrt{\frac{2DC_o}{P_1 I}} = \sqrt{\frac{2\times 2400\times 100}{10\times 0.24}} = 447 \text{ units.}$$

Now total cost corresponding to order size 447 is

$$TC(Q_1^*) = DP_1 + \frac{D}{Q_1^*}C_o + \frac{Q_1^*}{2}IP_1$$

$$= 2400\times 10 + \frac{2400\times 100}{447} + \frac{447}{2}\times 0.24\times 10$$

= 2400 + 536.91 + 536.4 = Rs. 25073.31

Total cost at price break is

$TC(b) = TC(500)$

$$= DP_2 + \frac{D}{b}\times C_o + \frac{b}{2}IP_2$$

$$= 2400\times 9 + \frac{2400}{500}\times 100 + \frac{500}{2}\times 0.24\times 9$$

= Rs. 22620

Since $TC(b) < TC(Q_1^*)$, the optimal order quantity is the price discount quantity is 500 units.

4.6.2. EOQ Problems with Two Price Breaks

When there are two price breaks i.e. two quantity discounts, the situation can be represented as

Range of quantity	Purchase cost per unit
$0 \leq Q < b_1$	P_1
$b_1 \leq Q < b_2$	P_2
$b_2 \leq Q$	P_3

where b_1 and b_2 are the quantities which determine this price discount. The procedure for obtaining EOQ may be summarized in the following steps:

1. Compute the optimal order quantity for the lowest price. Let it be Q_3^*.
2. If $Q_3^* \geq b_2$, the optimum order quantity is Q_3^*.
3. If $Q_3^* < b_2$, calculate Q_2^*, the optimal order quantity for the next lowest price.
4. If $b_1 \leq Q_2^* < b_2$, then compare $TC\left(Q_2^*\right)$ and $TC\left(b_2\right)$ to determine the optimum purchase quantity.
5. If $Q_2^* < b_1$, calculate Q_1^* and compare $TC\left(b_1\right)$, $TC\left(b_2\right)$ and $TC\left(Q^*\right)$ to determine the optimum, purchase quantity.

Example 1:

Find the optimal order quantity for a product for which the price discounts are as:

Range of quantity	Unit price (Rs.)
$0 \leq Q < 500$	10.00
$500 \leq Q < 750$	9.25
$750 \leq Q$	8.75

The monthly demand for the product is 200 units, storage cost is 2% of unit cost and ordering cost is Rs. 100.

Solution. Here $D = 200$ units

$I = 2\%$ of unit cost

C_0 = Rs. 100 per order

P_1 = Rs. 10

P_2 = Rs. 9.25

P_3 = Rs. 8.75

We calculate Q_3^* as

$$Q_3^* = \sqrt{\frac{2DC_o}{IP_3}} = \sqrt{\frac{2 \times 200 \times 100}{8.75 \times 0.02}} = 475 \text{ units}$$

Since $Q_3^* < b_2 = 750$, Q_3^* is not feasible.

$$\text{Now } Q_2^* = \sqrt{\frac{2DC_o}{IP_2}} = \sqrt{\frac{2 \times 200 \times 100}{0.02 \times 9.25}} = 465 \text{ units}$$

Since $Q_2^* = b_1 = 500$, we compute Q_1^*

$$Q_1^* = \sqrt{\frac{2DC_o}{IP_1}} = \sqrt{\frac{2 \times 200 \times 100}{0.02 \times 10}} = 447 \text{ units.}$$

Now,

$$TC\left(Q_1^*\right) = DP_1 + \frac{D}{Q_1^*}C_o + \frac{Q_1^* IP_1}{2}$$

$$= 200\times10 + \frac{200}{447}\times100 + \frac{447}{2}\times0.02\times10$$

$= 2000 + 44.74 + 44.70 =$ Rs. 2089.44

$$TC(b_1) = TC(500)$$

$$= DP_2 + \frac{D}{b_1}C_o + \frac{b_1}{2}IP_2$$

$$= 200\times9.25 + \frac{200}{500}\times100 + \frac{500}{2}\times0.02\times9.25$$

$= 1850 + 40 + 46.25 =$ Rs. 1936.25

$$TC(b_2) = TC(750)$$

$$= DP_3 + \frac{D}{b_2}C_o + \frac{b_2}{2}IP_3$$

$$= 200\times8.75 + \frac{200}{750}\times100 + \frac{750}{2}\times0.02\times8.15$$

$= 1750 + 26.67 + 65.62 =$ Rs. 1842.29

The lowest total inventory cost is Rs. 1842.24

So optimal order quantity is $Q^* = b_2 = 750$

4.6.3. Purchase Inventory Model with *n* Price Breaks

When there are n price breaks, the situation may be illustrated as

Range of quantity	Purchase cost per unit
$0 \le Q < b_1$	P_1
$b_1 \le Q < b_2$	P_2
$\vdots$	$\vdots$
$b_{n-1} \le Q$	P_n

where $b_1, b_2, \ldots b_{n-1}$ are the quantities which determine the price breaks. Let $Q_1^*, Q_2^*, \ldots, Q_n^*$ be EOQ corresponding to prices $P_1, P_2, \ldots, P_n$ respectively. The procedure for obtaining optimum order quantity is:

1. Compute Q_n^*. If $Q_n^* \ge b_{n-1}$, then the optimum purchase quantity is Q_n^*.
2. If $Q_n^* < b_{n-1}$, then compute Q_{n-1}^*.
   If $Q_{n-1}^* \ge b_{n-2}$, then optimum order quantity is determined by comparing $TC\left(Q_{n-1}^*\right)$ with $TC\left(b_{n-1}\right)$.
3. If $Q_{n-1}^* < b_{n-2}$, compute Q_{n-2}^*. If $Q_{n-2}^* \ge b_{n-3}$, then optimum order quantity is determined by comparing $TC\left(Q_{n-2}^*\right)$ with $TC\left(b_{n-2}\right)$ and $TC\left(b_{n-1}\right)$.
4. If $Q_{n-2}^* < b_{n-3}$, compute Q_{n-3}^*
   If $Q_{n-3}^* \ge b_{n-4}$, then optimum order quantity is determined by comparing $TC\left(Q_{n-3}^*\right)$ with $TC\left(b_{n-3}\right)$, $TC\left(b_{n-2}\right)$ and $TC\left(b_{n-1}\right)$.
5. Continue in this way until $Q_{n-j}^* \ge b_{n-(j+1)}; 0 \le j \le n-1;$ and then compare $TC\left(Q_{n-j}^*\right)$ with $TC\left(b_{n-j}\right)$, $TC\left(b_{n-j+1}\right), TC\left(b_{n-j+2}\right), \ldots, TC\left(b_{n-1}\right)$.

This procedure involves a finite number of steps.

Example 1:

The annual demand for a product is 500 units. The cost of storage per unit per year is 10% of the unit cost. The ordering cost is Rs.180 for each order. The unit cost depends upon the amount ordered. The range of amount ordered and the unit cost price are as follows:

Range of amount ordered	Price per unit
$0 \le Q < 500$	Rs. 25
$500 \le Q < 1500$	Rs. 24.80
$1500 \le Q < 3000$	Rs. 24.60
$3000 \le Q$	Rs. 24.40

Find the optimal order quantity.

Solution. Here $D = 500$ units

C_o = Rs. 180 per order

$I = 0.10$

$b_1 = 500,\ b_2 = 1500,\ b_3 = 3000$

P_1 = Rs. 25, P_2 = Rs. 24.80

P_3 = Rs. 24.60, P_4 = Rs. 24.40

Step 1 $Q_4^* = \sqrt{\dfrac{2DC_o}{IP_4}} = \sqrt{\dfrac{2\times500\times180}{0.10\times24.40}} = \sqrt{\dfrac{1000\times10\times18000}{2440}}$

$= 1000\sqrt{\dfrac{180}{2440}} = 271.6 \simeq 272$ units

Since $Q_4^* < b_3$, we compute Q_3^*.

Step 2 $Q_3^* = \sqrt{\dfrac{2DC_o}{IP_3}} = \sqrt{\dfrac{2\times500\times180}{0.16\times24.60}}$

$= 100\sqrt{\dfrac{18}{246}} = 270$ units

Since $Q_3^* < b_2$, we compute Q_2^*

Step 3 $Q_2^* = \sqrt{\dfrac{2DC_o}{IP_2}} = \sqrt{\dfrac{2\times500\times180}{0.10\times24.80}} = 269$ units

Since $Q_2^* < b_1$, we compute Q_1^*

Step 4 $Q_1^* = \sqrt{\dfrac{2\times500\times180}{0.10\times25}} = \sqrt{\dfrac{180}{25}\times10000} = \sqrt{72000} = 268$ units.

Now we compute $TC\left(Q_1^*\right), TC\left(b_1\right), TC\left(b_2\right), TC\left(b_3\right)$ and compare them to get optimal order quantity.

$$TC\left(Q_1^*\right) = DP_1 + \frac{D}{Q_1^*}C_o + \frac{1}{2}Q_1^* IP_1$$

$$= 500\times25 + \frac{500}{268}\times180 + \frac{268}{2}\times0.10\times25$$

$$= 12500 + 335.82 + 335 = \text{Rs. } 13170.82$$

$$TC(b_1) = DP_2 + \frac{D}{b_1}C_o + \frac{1}{2}b_1IP_2$$

$$= 500 \times 24.80 + \frac{500}{500} \times 180 + \frac{500}{2} \times 0.10 \times 24.80$$

$$= 12400 + 1800 + 620 = \text{Rs. } 14820$$

$$TC(b_2) = DP_3 + \frac{D}{b_2}C_o + \frac{1}{2}b_2IP_3$$

$$= 500 \times 24.60 + \frac{500}{1500} \times 180 + \frac{1}{2} \times 1500 \times 0.10 \times 24.60$$

$$= 12300 + 60 + 1845 = \text{Rs. } 14205$$

$$TC(b_3) = DP_4 + \frac{D}{b_3}C_o + \frac{1}{2}b_3IP_4$$

$$= 500 \times 24.40 + \frac{500}{3000} \times 180 + \frac{1}{2} \times 3000 \times 0.10 \times 2440$$

$$= 12200 + 30 + 3660 = \text{Rs. } 15890$$

Since $TC(Q_1^*) < TC(b_2) < TC(b_1) < TC(b_3)$,

Optimum order quantity is Q_1^* i.e. 268 units.

4.7. Probabilistic Inventory Models:

In previous sections, we have discussed simple deterministic inventory models where each and every influencing factor is completely known. Generally, in actual business environment complete certainty never occurs. Therefore, here we will discuss some practical situations of inventory problems by relaxing the condition of certainty for some of the factors.

The major influencing factors for the inventory problems are Demand, Price and Lead Time. There are also other factors like Ordering Cost, Carrying Cost or Holding Cost and Stock out Costs, but their nature is not so much disturbing. Because of this their estimation provides almost, on the average, as known as values. Even Price can also be averaged out to reflect the condition of certainty. But there are situations where Price fluctuations are too much in the market and hence they influence the inventory decisions. Similarly, the demand variations or consumption variation of an item as well as the lead time variation influence the overall inventory policy. In this section we will discuss single period probabilistic models.

4.8. Continuous Review Models:

This section presents two models: (1) a “probabilitized” version of the deterministic EOQ that uses a buffer stock to account for probabilistic demand, and (2) a more exact probabilistic EOQ model that includes the probabilistic demand directly in the formulation.

4.8.1. "Probabilitized" EOQ Model:

Some practitioners have sought to adapt the deterministic EOQ model to reflect the probabilistic nature of demand by using an approximation that superimposes a constant buffer stock on the inventory level throughout the entire planning horizon. The size of the buffer is determined such that the probability of running out of stock *during lead time* (the period between placing and receiving an order) does not exceed a prespecified value.

Let

L = Lead time between placing and receiving an order

x_L = Random variable representing demand during lead time

μ_L = Average demand during lead time

σ_L = Standard deviation of demand during lead time

B = Buffer stock size

α = Maximum allowable probability of running out of stock during lead time

The main assumption of the model is that the demand, x_L, during lead time L is normally distributed with mean μ_L and standard deviation σ_L—that is, $N(\mu_L, \sigma_L)$.

Buffer stock imposed on the classical EOQ model

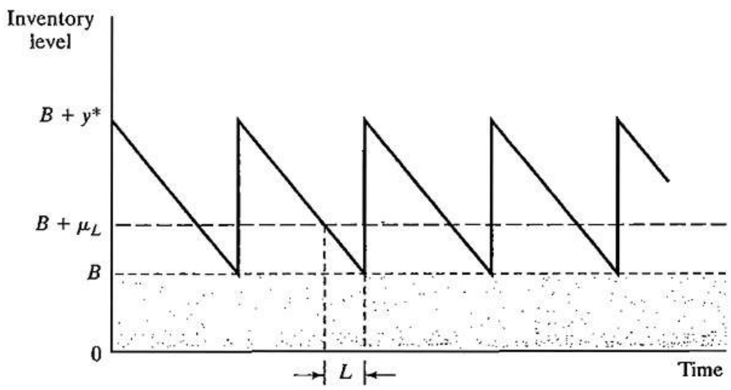


Figure.1

Example 1:

In Example 11.2-1 dealing with determining the inventory policy of neon lights, EOQ = 1000 *units*. If the *daily* demand is normal with mean $D = 100$ lights and standard deviation $\sigma = 10$ lights—that is, $N(100, 10)$—determine the buffer size so that the probability of running out of stock is below $\alpha = .05$.

From Example 11.2-1, the effective lead time is $L = 2$ days. Thus,

$$\mu_L = DL = 100 \times 2 = 200 \text{ units}$$

$$\sigma_L = \sqrt{\sigma^2 L} = \sqrt{10^2 \times 2} = 14.14 \text{ units}$$

Given $K_{.05} = 1.645$, the buffer size is computed as

$$B \geq 14.14 \times 1.645 \approx 23 \text{ neon lights}$$

Thus, the optimal inventory policy with buffer B calls for ordering 1000 units whenever the inventory level drops to 223 ($= B + \mu_L = 23 + 2 \times 100$) units.

4.8.2. Probabilistic EOQ Model:

There is no reason to believe that the "probabilitized" EOQ model is section 4.8.1 will produce an optimal inventory policy. The fact that pertinent information regarding the probabilistic nature of demand is initially ignored, only to be "revived" in a totally independent manner at a later stage of the calculations, is sufficient to refute optimality.

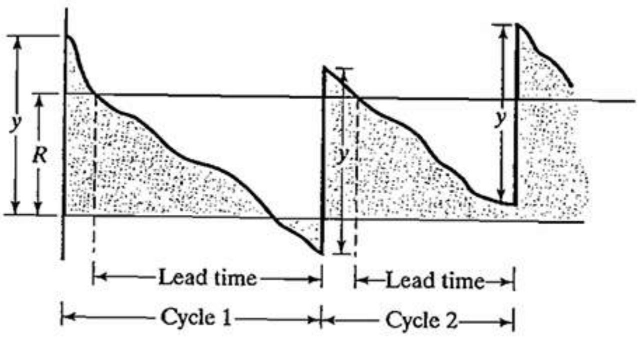


Figure.3

Probabilistic inventory model with shortage

The model has three assumptions.

1. Unfilled demand during lead time is backlogged.
2. No more than one outstanding order is allowed.
3. The distribution of demand during lead time remains stationary (unchanged) with time.

To develop the total cost function per unit time, let

$$f(x) = \text{pdf of demand, } x\text{, during lead time}$$
$$D = \text{Expected demand per unit time}$$
$$h = \text{Holding cost per inventory unit per unit time}$$
$$p = \text{Shortage cost per inventory unit}$$
$$K = \text{Setup cost per order}$$

Based on these definitions, the elements of the cost function are now determined.

1. *Setup cost.* The approximate number of orders per unit time is $\frac{D}{y}$, so that the setup cost per unit time is approximately $\frac{KD}{y}$.

2. *Expected holding cost.* The average inventory is

$$I = \frac{(y + E\{R - x\}) + E\{R - x\}}{2} = \frac{y}{2} + R - E\{x\}$$

The formula is based on the average of the beginning and ending expected inventories of a cycle, $y + E\{R - x\}$ and $E\{R - x\}$, respectively. As an approximation, the expression ignores the case where $R - E\{x\}$ may be negative. The expected holding cost per unit time thus equals hI.

3. *Expected shortage cost.* Shortage occurs when $x > R$. Thus, the expected shortage quantity per cycle is

$$S = \int_R^{\infty} (x - R)f(x)\,dx$$

Because p is assumed to be proportional to the shortage quantity only, the expected shortage cost per cycle is pS, and, based on $\frac{D}{y}$ cycles per unit time, the shortage cost per unit time is $\frac{pDS}{y}$.

The resulting total cost function per unit time is

$$\text{TCU}(y, R) = \frac{DK}{y} + h\left(\frac{y}{2} + R - E\{x\}\right) + \frac{pD}{y}\int_R^{\infty} (x - R)f(x)\,dx$$

The solutions for optimal y^* and R^* are determined from

$$\frac{\partial \text{TCU}}{\partial y} = -\left(\frac{DK}{y^2}\right) + \frac{h}{2} - \frac{pDS}{y^2} = 0$$

$$\frac{\partial \text{TCU}}{\partial R} = h - \left(\frac{pD}{y}\right)\int_R^{\infty} f(x)\,dx = 0$$

We thus get

$$y^* = \sqrt{\frac{2D(K + pS)}{h}} \tag{1}$$

$$\int_{R^*}^{\infty} f(x)\,dx = \frac{hy^*}{pD} \tag{2}$$

Because y^* and R^* cannot be determined in closed forms from (1) and (2), a numeric algorithm, developed by Hadley and Whitin (1963, pp. 169–174), is used to find the solutions. The algorithm converges in a finite number of iterations, provided a feasible solution exists.

For $R = 0$, (1) and (2) above yield

$$\hat{y} = \sqrt{\frac{2D(K + pE\{x\})}{h}}$$

$$\tilde{y} = \frac{PD}{h}$$

If $\tilde{y} \geq \hat{y}$, unique optimal values of y and R exist. The solution procedure recognizes that the smallest value of y^* is $\sqrt{\frac{2KD}{h}}$, which is achieved when $S = 0$.

The steps of the algorithm are

Step 0. Use the initial solution $y_1 = y^* = \sqrt{\frac{2KD}{h}}$, and let $R_0 = 0$. Set $i = 1$, and go to step i.

Step i. Use y_i to determine R_i from Equation (2). If $R_i \approx R_{i-1}$, stop; the optimal solution is $y^* = y_i$, and $R^* = R_i$. Otherwise, use R_i in Equation (1) to compute y_i. Set $i = i + 1$, and repeat step i.

Example 2:

Electro uses resin in its manufacturing process at the rate of 1000 gallons per month. It costs Electro \$100 to place an order for a new shipment. The holding cost per gallon per month is \$2, and the shortage cost per gallon is \$10. Historical data show that the demand during lead time is uniform over the range (0, 100) gallons. Determine the optimal ordering policy for Electro.

Using the symbols of the model, we have

$$D = 1000 \text{ gallons per month}$$
$$K = \$100 \text{ per order}$$
$$h = \$2 \text{ per gallon per month}$$
$$p = \$10 \text{ per gallon}$$
$$f(x) = \tfrac{1}{100},\ 0 \le x \le 100$$
$$E\{x\} = 50 \text{ gallons}$$

First, we need to check whether the problem has a feasible solution. Using the equations for $\hat{y}$ and $\tilde{y}$ we get

$$\hat{y} = \sqrt{\frac{2 \times 1000(100 + 10 \times 50)}{2}} = 774.6 \text{ gallons}$$
$$\tilde{y} = \frac{10 \times 1000}{2} = 5000 \text{ gallons}$$

Because $\tilde{y} \ge \hat{y}$, a unique solution exists for y^* and R^*.

The expression for S is computed as

$$S = \int_R^{100} (x - R)\frac{1}{100}dx = \frac{R^2}{200} - R + 50$$

Using S in Equations (1) and (2), we obtain

$$y_i = \sqrt{\frac{2 \times 1000(100 + 10S)}{2}} = \sqrt{100{,}000 + 10{,}000S} \text{ gallons} \qquad (3)$$

$$\int_R^{100} \frac{1}{100}dx = \frac{2y_i}{10 \times 1000}$$

The last equation yields

$$R_i = 100 - \frac{y_i}{50}$$

We now use Equations (3) and (4) to determine the solution.

Iteration 1

$$y_1 = \sqrt{\frac{2KD}{h}} = \sqrt{\frac{2 \times 1000 \times 100}{2}} = 316.23 \text{ gallons}$$

$$R_1 = 100 - \frac{316.23}{50} = 93.68 \text{ gallons}$$

Iteration 2

$$S = \frac{R_1^2}{200} - R_1 + 50 = .19971 \text{ gallons}$$

$$y_2 = \sqrt{100{,}000 + 10{,}000 \times .19971} = 319.37 \text{ gallons}$$

Hence,

$$R_2 = 100 - \frac{319.39}{50} = 93.612$$

Iteration 3

$$S = \frac{R_2^2}{200} - R_2 + 50 = .20399 \text{ gallon}$$

$$y_3 = \sqrt{100{,}000 + 10{,}000 \times .20399} = 319.44 \text{ gallons}$$

Thus,

$$R_3 = 100 - \frac{319.44}{50} = 93.611 \text{ gallons}$$

4.9. Single-Period Models:

Single-item inventory models occur when an item is ordered only once to satisfy the demand for the period. For example, fashion items become obsolete at the end of the season. This section presents two models representing the no-setup and the setup cases.

The symbols used in the development of the models include

K = Setup cost per order

h = Holding cost per held unit during the period

p = Penalty cost per shortage unit during the period

D = Random variable representing demand during the period

$f(D)$ = pdf of demand during the period

y = Order quantity

x = Inventory on hand before an order is placed.

The model determines the optimal value of y that minimizes the sum of the expected holding and shortage costs. Given optimal y $(= y^*)$, the inventory policy calls for ordering $y^* - x$ if $x < y$; otherwise, no order is placed.

4.9.1. No-Setup Model (Newsvendor Model):

This model has come to be known in the literature as the *newsvendor* model (the original classical name is the *newsboy* model) because it deals with items with short life such as newspapers.

The assumptions of this model are

1. Demand occurs instantaneously at the start of the period immediately after the order is received.
2. No setup cost is incurred.

If $D < y$, the quantity $y - D$ is held during the period. Otherwise, a shortage amount $D - y$ will result if $D > y$.

The expected cost for the period, $E\{C(y)\}$, is expressed as

$$E\{C(y)\} = h\int_0^y (y - D)f(D)\,dD + p\int_y^\infty (D - y)f(D)\,dD$$

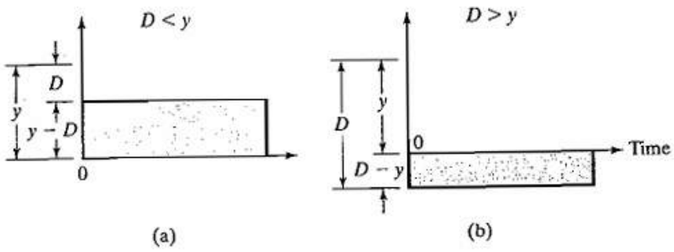


Figure.3

The function $E\{C(y)\}$ can be shown to have a unique minimum because it is convex in y. Taking the first derivative of $E\{C(y)\}$ with respect to y and equating it to zero, we get

$$h\int_0^y f(D)\,dD - p\int_y^\infty f(D)\,dD = 0$$

or

$$hP\{D \leq y\} - p(1 - P\{D \leq y\}) = 0$$

or

$$P\{D \leq y^*\} = \frac{p}{p + h}$$

The preceding development assumes that the demand D is continuous. If D is discrete, then $f(D)$ is defined only at discrete points and the associated cost function is

$$E\{C(y)\} = h\sum_{D=0}^{y}(y - D)f(D) + p\sum_{D=y+1}^{\infty}(D - y)f(D)$$

The necessary conditions for optimality are

$$E\{C(y - 1)\} \geq E\{C(y)\} \text{ and } E\{C(y + 1)\} \geq E\{C(y)\}$$

These conditions are sufficient because $E\{C(y)\}$ is a convex function. After some algebraic manipulations, the application of these conditions yields the following inequalities for determining y^*:

$$P\{D \leq y^* - 1\} \leq \frac{p}{p + h} \leq P\{D \leq y^*\}$$

4.9.2. Setup Model (s-S policy):

The present model differs from the one in section 4.8.1 in that a setup cost K is in curred. Using the same notation, the total expected cost per period is

$$E\{\bar{C}(y)\} = K + E\{C(y)\}$$
$$= K + h\int_0^y (y - D)f(D)\,dD + p\int_y^\infty (D - y)f(D)\,dD$$

$$P\{y \leq y^*\} = \frac{p}{p+h}$$

$$E\{C(s)\} = E\{\bar{C}(S)\} = K + E\{C(S)\}, s < S$$

The equation yields another value $s_1 (> S)$, which is discarded.

Given that the amount on hand before an order is placed is x units, how much should be ordered? This equation is investigated under three conditions:

1. $x<s$
2. $s \leq x \leq S$
3. $x>S$

This condition indicates that it is not advantageous to order in this case—that is, $y^* = x$.

The optimal inventory policy, frequently referred to as the s-S policy, is summarized as

$$\text{If } x < s, \text{ order } S - x$$
$$\text{If } x \geq s, \text{ do not order}$$

The optimality of the s-S policy is guaranteed because the associated cost function is convex.

Example 3:

The daily demand for an item during a single period occurs instantaneously at the start of the period. The pdf of the demand is uniform between 0 and 10 units. The unit holding cost of the item during the period is \$.50, and the unit penalty cost for running out of stock is \$4.50. A fixed cost of \$25 is incurred each time an order is placed. Determine the optimal inventory policy for the item.

To determine y^*, consider

$$\frac{p}{p+h} = \frac{4.5}{4.5+.5} = .9$$

Also,

$$P\{D \leq y^*\} = \int_0^{y^*} \frac{1}{10} dD = \frac{y^*}{10}$$

Thus, $S = y^* = 9$.

The expected cost function is given as

$$E\{C(y)\} = .5\int_0^{y} \frac{1}{10}(y - D)\,dD + 4.5\int_y^{10} \frac{1}{10}(D - y)\,dD$$

$$= .25y^2 - 4.5y + 22.5$$

The value of s is determined by solving

$$E\{C(s)\} = K + E\{C(S)\}$$

This yields

$$.25s^2 - 4.5s + 22.5 = 25 + .25S^2 - 4.5S + 22.5$$

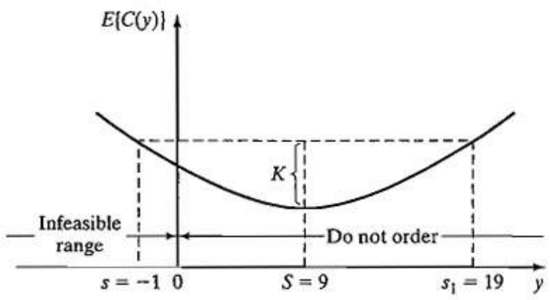


Figure.4

s-S policy applied to Example

Given S=9, the proceeding equation reduces to S^2-18s-19=0

The solution of this equation is s=-1 or s=19. The value of s>S is discarded. Because the remaining value is negative (=-1), s has no feasible value (Figure.4). This conclusion usually happens when the cost function is flat or when the setup cost is high relative to the other costs of the model.

Unit V

Queuing Theory: Basic elements of Queuing Model-Role of Poisson and Exponential Distributions- Pure Birth and Death Models-Specialised Poisson Queues-(M/G/1):GD/∞/∞)-Pollaczek-Khintechine Formula.

Chapter 5: Sections 5.1-5.5

5.1. Introduction:

Queueing or waiting in a line is a common situation occurring in everyday life. We wait in queues in ticket booths, bus stops, post offices, banks, traffic lights and so on. We can think of many more situations. In general, a queue is formed when there are customers who require some sort of services and the queuing problem is identified by the presence of a group of customers who arrive randomly to receive some service.

The customer may be a person, machine, vehicle or anything else which requires service. The objectiveof a queuing model is to find out the optimum service rate and the number of servers so that the average cost of being in queuing system and the cost of service are minimized.

The objective of a queueing system is to find ways of reducing the time spent in waiting by the customer and at the same time optimizing the cost to the service provider. Sometimes Queueing Theory, also reference as the waiting line theory. It was developed in 1909 when A.K. Erlang made an effort to analyses telephone traffic congestion. It can be applied to wide variety of operational situations whenever customer's expectations do not match with objectives of semers due to one's inability to predict accurately the arrival and service time of customers. The purpose of queueing analysis is to provide information to evaluate an acceptable level of service and service capacity since providing too much service capacity is costly (because of employees or equipment). In fact, providing too little service capacity is also costly.

In this Chapter, we discuss the concepts of stochastic processes, Poisson process and birth -death process. We explain the basic component, fundamental structure and operating characteristics of a queueing system. Also, we describe the M/M/1, queueing models.

5.2. Basic Concepts of Queueing Theory:

For studying queueing systems, one should be familiar with the probability theory, specially the conceptof random variable and its probability distribution such as Poisson distribution, Exponential distribution Let us have a glance on some basic definitions.

Consider an experiment whose outcome is not uniquely determined. In such a situation an observed

outcome of the experiment is one from a set of possible

outcomes. **Sample point:** An outcome of an experiment is

called a sample point.

Sample Space: The set of all possible outcomes of a random experiment is called sample space.**Events:** Subsets of the sample space are called events.

Random Variable: A random variable is a function that associates a point of the sample space with a real number.

Random Process/Stochastic Process: A random process or stochastic process is a family (or collection) of random variables.

For example, if a die with six faces numbered 1, 2... 6 is thrown, the set S is a sample space where the set of all possible outcomes is S= (1, 2, 3, 4, 5, 6). A function X that assigns to an outcome or sample point, the number written on it, is the random variable. The event that an odd number is observed corresponds to the set {1, 3, 5} which is a subset of S.

Let us take more examples of stochastic process.

i) If $n \geq 1$, Suppose X_n is the outcome of the n^{th} throw. Then $\{X_n, n \geq 1\}$ is a collection of random variables, such that for a distinct value of n, one gets a distinct random variable X_n. The sequence $\{X_n, n \geq 1\}$ Constitutes a random (stochastic) process called the Bernoulli process.

ii) Let X_n represents the number of sixes in the first n throws. For each value of n=1, 2, ..., we get a distinct Binomial random variable. The sequence X_n. $(X_n, n \geq 1\}$, which gives a collection of random variables, is a random or stochastic process.

iii) Suppose a telephone call is received at switchboard. Let X, be the random variable, which represents the number of incoming calls in an interval (0, t). Then X, is a random variable and the family $\{X_t\ t\ \varepsilon\ T\}$ constitutes a stochastic process, where T is the interval $0 \leq t \leq \infty$

5.3. Poisson Process:

Suppose X(t) represent the maximum temperature at a particular place. Here we deal with discrete state, continuous time stochastic processes. The Poisson process is one of the representatives of this type of stochastic processes. The Poisson process may be explain as follows:

Let E be a random event such as (i) incoming telephone calls at a switchboard, (ii) arrival of patients for treatment at a clinic, (iii) occurrence of accidents at a certain place.

We consider the total number N(t) of occurrences of an event E in an interval of time 't'. Suppose $P_n(t)$ is the probability that the random variable N(t) assumes the value n, i.e.,

$P_n(t) = P[N(t)=n]$ .....1.

For n=0, 1, 2, 3.......... We have,

$\sum_{n=0}^{\infty} P(t) = 1$, for each fixed t 2

We can thus say from equation (2) that $P_n(t)$ is a probability mass function of the random variable N(t) and the family of random variables $[N_t, t > 0]$ is a stochastic process. From our earlier discussion, you mayunderstand that this family is continuous parameter (in this case, time) stochastic process with a discrete state space. This is called a Poisson process. Under certain conditions, N(t) follows a Poisson distribution with mean λt (λ is being constant). This holds for most practical situations.

Assumptions in Poisson Process

Events must be independent, in other words the number of the customers which is arrive in disjoint time intervals are statistically independent e.g. the number of goals scored by a team should not make the number of goals scored by another team more or less likely and the mean number of goals scored is assumed to be the same for all teams.

The probability of two events occurring between time t and t + Δt is 0(Δ (t)), i.e. negligible. Thus., $P_o(\Delta t) + P_1(\Delta t) + 0(\Delta t) = 1$

The probability that event E occurs between time t and Δt is equal to $\lambda\Delta t + 0(\Delta)t$). Thus, $P_i(\Delta t) = \lambda\Delta t + 0(\Delta t)$, Under the assumptions stated above, N(t) follows a Poisson distribution with mean λt, i.e., Pn(t) is given by

$$P_n(t) = \frac{e^{-\lambda t(\lambda t)}}{n!} \text{ , n=0,1, 2, 3 , ...}$$

To formulate a queueing model, we have to specify the assumed form of the probability distributions of both inter arrival times and service e times. You have learnt that inter- arrival times follow the Poisson distribution. Similarly, most of the times, the service times in a queueing system follow the exponential distribution. Hence, the exponential distribution is the most important distribution in queueing theory

Markov Process

Around the beginning of the 20^{th} century, the methodology discussed here was developed by the Russian mathematician. A.A. Markov. The Markov Process forms a sub-class of the set of all random processes.It is a sub class with enough simplifying assumptions to make them easy to handle.

A stochastic (or random) process is called a Markov process if the occurrence of a future state depends on

the immediately preceding state and only on it. If, for all tn , tn-1 ... . to, satisfying tn >tn-1>......t0 and non- negative integer j, the family of random variables {X(t), t≥0} is said to be a Markov process if it satisfy the Markovian property:

P[X(tn)= j_n/X (tn-1)= j_{n-1} ... ... X(to)= j_0]=P[X(tn)= j_n/X (tn-1)= j_{n-1}]

The process has the Markov property and is called a continuous time Markov process.

5.4. Birth and Death Process:

The arrival process assumes that the customers arrive at the queueing system and never leave it. Such a system is down as pure birth process. On the other hand, the departure process assumes that no customer joins the system while service is continued for those who are already in the system. If at time t=0, N is greater than or equal to one i.e. at stating time number of customers is N. Since service is being provided at the rate of y , therefore customers leave the system at the rate of y after being serviced. This type of process is known as pure death process.

Let the state at time t would correspond to the number of arrivals by time t. *Pn(t)* denotes the probability that there are n customers in the system at any time t, both waiting and being served. Arrivals can be considered as births. If the system is in state E_n and arrival occurs, the state is changed to E_{n+1}. Similarly, a departure can be looked upon as death. A departure occurring while the system is in state *En* changes the system to the state E_{n-1}. This type of process is generally referred to as a birth death process.

Assumptions in the Birth-Death Process

The assumptions of the birth-death process are as follows:

- If the system is in state *En*, the current probability distribution of the time t until the next arrival(birth) is exponential with parameter λ_n, where n = 0, 1, 2,
- If the system is in state *En*, the current probability distribution of the time t until the next departure or service completion (death) is exponential with parameter pp, where n =0, 1, 2 ,....
- Only one birth or death can occur in a small interval of time, i.e., Δt.

5.4.1. Fundamental Structure of a Queuing System

Queuing theory is related with the mathematical study of queues or waiting lines, a queue is formed when there are customers who require some sort of services and the current demand for a service exceeds the current capacity to provide the service.

Generally, the customer's arrival and their service time are not known in advance or can't be predicted accurately. Since arrival-departure process are random. So, queuing models developed to reduce waiting time/excessive costs and work for maintaining balance between service capacity and waiting time.

A simple queuing system can be described as follows:

1. Input or arrival process of customers
2. Service mechanism (or process)
3. Queue discipline

The fundamental structure of a queueing system shown in the figure given below

Now we explain all the components of queueing system.

1. Input or Arrival Process of Customers

The rate at which the customers arrive at the service facility is determined by the arrival process. An input/arrival process can be defined completely by its size, the arrival time distribution, and the attitude of the customers. We describe these, in brief.

i) **Size:** It may be finite or infinite according as the arrival rate is affected or not affected by the number of customers in the service system.

ii) **The arrival time distribution:** Mostly, the arrival time distribution is approximated by Poisson

distribution.

iii) Customer or arrivals behavior:

- A customer who stay in the system until served no matter how much he has to wait for service. Such a customer is called Patient Customer.
- The customer who wait for a certain time in the queue and leave the system without getting service. This kind of customer is known as impatient or reneging behavior.
- If a customer before joining the system get discouraged by seeing the number of customers already in the queue is too large and does not join the queue. This behavior of the customer is called Balking behavior.
- Customers who move from one queue to another because they think that their queue ismoving slower, the behavior of the customer is known as queue jockeying.

Remark: Generally, it is assumed that the customers arrive into the system one at a time. But, sometimes, customers may arrive in groups and such arrival is called Bulk arrival.

2. Service Mechanism (or Process):

Service time distributions are generally exponential distributions. It may be any one of the following types:

1. Single channel facility
2. One queue-several station facilities
3. Several queues-one service station
4. Multi-channel facility and
5. Multistage channel facility

3. Queue Discipline:

Queue discipline is the order or the manner in which the service station selects the next customer from thewaiting line to be served. There are many ways in which a customer to be selected for service. Some of these are described as follows:

i) FIRST IN, FIRST OUT (FIFO) or in other words First Come First Served (FCFS);

ii) Last in First Out (LTO); and

iii) Service in Random Order (S&O)

Throughout the chapter, we consider the FCFS queue discipline.

5.4.2. Operating Characteristics of a Queueing System

For studying queuing system, we have to set up a set of equations. We have to solve these equations to determine the operating characteristics (or performance measures). There are two types of solutions of these equations: (i) **Transient** (ii)**Steady state**.

Transient solutions: The time dependent solutions are known as transient solutions.

Steady **state solutions:** These solutions are independent of time and represent the probability of the

system being in a particular state in the long run.

Note: Here we shall be analysing the system under the steady state condition.

5.4.3. Operating characteristics/ Performance Measures

Performance Measures of a queuing system are determined by two statistical properties, namely, theprobability distribution of inter-arrival times and the probability distribution of service times.

Some of the operating characteristics/performance measures of any queueing system that are of generalinterest for analysing the system are listed below:

1. **Ls:** The average number of customers in the queuing system (those waiting to be served and those being served).

2. **W_s:** The average time each customer spends in the queuing system from entry into the queue to completion of the service (the time spent waiting in the queue and during the service).

3. **L_q:** The average number of customers in the queue waiting to get service (this excludes customersundergoing service).

4. **Wq:** The average time each customer spends in the queue waiting to get service (this excludes customers time spent during the service).

5. **Service idle time:** The relative frequency with which the service system is idle.

Now we describe the equations for each of the operating characteristics listed above under steady statesolutions. By definition of expectation, we have

$$L_s = E(n) = \sum_{n=1}^{\infty} nP_n \text{ (Expected number of customers in the system)} \quad (1)$$

$$L_q = \sum_{n=C+1}^{\infty} (n-c)P_n \text{ (Expected number of customers in queue)} \quad (2)$$

where there are c parallel servers so that c customers can be served simultaneously with λ arrival rates for those who join the system, we have

$$L_s = \lambda W_s \quad (3)$$

and

$$L_q = \lambda W_q \quad (4)$$

If μ is the service rate, the expected service time is $\frac{1}{\mu}$ and we have

$$W_s = W_q + \frac{1}{\mu} \quad (5)$$

Where, W_s denotes expected waiting time in system and W_q represents expected waiting time in queue

Multiplying both sides of equation (5) with λ, we get

$$\lambda W_s = \lambda W_q + \frac{\lambda}{\mu}$$

$$\Rightarrow L_s = L_q + \frac{\lambda}{\mu} \text{ (By (3) and (4))} \quad (6)$$

5.5. Classification of Queueing Models:

Generally, queueing models are described by five symbols such as a/b//c: d/e or a/b/c/d/e. The first symbol'a' describes the arrival process. The second symbol 'b' describes the service time distribution. The third

symbols 'c' stands for the number of servers. The symbols 'd' and 'e' stand for the system capacity and queue discipline respectively.

First three symbols i.e. a/b/c in the above notation were described by D. Kendall in 1953. Later, A. Lee in1966 added the fourth (d) and fifth (e) to the Kendall notation.

Remark: The fifth symbol (e) from the notation can be omitted if the system has FCFS queue discipline and for this case it can be described as a/b/c/d. Moreover, if the system has infinite capacity with FCFS queue discipline then we can use simply a/b//c or a/b/c: ∞ /FCFS for describing the queueing system.

Note: If arrivals follow Poisson distribution and departures follow exponential distribution, symbol M is used in place of 'a' and 'b' e.g. suppose there is a single server then we can use the notation M/M/1(here we consider infinite capacity and FCFS queue discipline).

5.5.1. M/M/1 OR M/M/1: ∞ /FCFS QUEUEING MODEL

This model deals with a queueing system having a single server channel and no limit on system capacity with Poisson input and exponential distribution for services while the customers are served on a "First Come First Served" basis. For the given model, arrivals follow Poisson distribution and hence inter-arrival times have an exponential distribution. The service time for this model follows an exponential distribution.If λ *and* μ denote the average arrival rate and average service rate respectively then $1/\lambda$ *and* $1/\mu$ show mean arrival time and mean service time correspondingly. Then the ratio. $\rho = \frac{\lambda}{\mu}$ is called the traffic intensity or the utilization factor. It is the measure of the degree to which the capacity of the service stationis utilized. For example, if customer arrives at a rate of 10 per minute and the service rate 20 per minute, the utilization of the service capacity is $\frac{10}{50} = 50\%$, i.e., the service facility is kept busy 50% of the time and remains idle 50% of the time.

Arrival-Departure Equations for the M/M/1 Queuing Model (or)

System of Differential-Difference Equations

If n ≥ 1, the probability that the system has n customers at time (t+ Δt) can be expressed as:

P_n(t+ Δt) = [{Probability of n-1 customers in the system at time › and (1 arrival, no departure in time Δt or 2 arrivals, one departure in time Δt or 3 arrivals, two departure in time Δt or ...)} + {Probability of *n* 1 customers in the system at time › and (2 arrivals, no departure in time Δt or 3 arrivals, one departure in time Al or 4 arrivals, two departures in time Δt or ...)} + {Probability of n 3 customers in

the system at time t and (3 arrivals, no departure in time Δt or 4 arrivals, one departure in time Δt or 5

arrivals, two departures in time Δt or ...)} + ...] + [{Probability of n + 1 customers in the system at timer and (1 departure, no arrival in time Δt or 2 departures, one arrival in time Δt or 3 departures, two arrivals in time Δt or ...)} + {Probability of *n* + 2 customers in the system at time › and (2 departures, no arrivals in time Δt or 3 departures, one arrival in time Δt or ...)} + {Probability of n + 3 customers in the system at time t and (3 departures, no arrival in time Δt or …)} + …] + [{Probability of n customers in the system at time t and (no arrival, no departure in time Δt or 1 arrival, one departure in time Δt or 2 arrivals, two departures in time Δt or …)]

$$= P_{n-1}(t)\left[\{\lambda\Delta t + O(\Delta t)\}\{1-\mu\Delta t + O(\Delta t)\} + \text{The terms equal to } O(\Delta t)\right]$$

$$+P_{n-2}(t)\left[\{O(\Delta t)\}\{1-\mu\Delta t + O(\Delta t)\} + \text{The terms equal to } O(\Delta t)\right]$$

$$+P_{n-3}(t)O(\Delta t) + \text{The terms equal to } O(\Delta t)$$

$$+P_{n+1}(t)\left[\{(\mu\Delta t + O(\Delta t))\}\{1-\lambda\Delta t + O(\Delta t)\} + \text{The terms equal to } O(\Delta t)\right]$$

$$+P_{n+2}(t)\left[\{O(\Delta t)\}\{1-\lambda\Delta t + O(\Delta t)\} + \text{The terms equal to } \mathrm{O}(\Delta t)\right]$$

$$+P_{n+3}(t)O(\Delta t) + \text{The terms equalto } O(\Delta t)$$

$$+P_{n}(t)\left[\{1-\mu\Delta t + O(\Delta t)\}\{1-\lambda\Delta t + O(\Delta t)\}\right.$$

$$\left.+\{\mu\Delta t + O(\Delta t)\}\{\lambda\Delta t + O(\Delta t)\} + \text{The terms which ultimately become equal to } O(\Delta t)\right]$$

Notice that $\Delta t \cdot O(\Delta t), O(\Delta t)\cdot O(\Delta t)$, $O(\Delta t)+O(\Delta t), O(\Delta t)-O(\Delta t)$, $\Delta t \cdot \Delta t$ are very small and may be taken as equal to $O(\Delta t)$

We deal with the case $n = 0$ separately because there cannot be any possibility of less than zero customers. However, in the above equation, the case of less than n customers is included.

The probability of 0(Zero)customers in the system at time $t+\Delta t$ = [{Probability of 1 customer in the system at time t and (1 departure, no customer/arrival in time Δt or 2 departure, 1 arrivals in time Δt or 3 departures, two arrivals in time Δt)} + {Probability of 2 customers in the system at time t and (2 departures, no arrival in time Δt or 3 departures, one arrival in time Δt or 4 departures, 2 arrivals in time Δt)} + {Probability of 3 customers in the system at time t and (3 departures, no arrival in time Δt or 4 departures, one arrival in time Δt or 5 departures, two arrivals in time Δt)} + …] + [{Probability of O customers in the system at time t and (no arrival, no departure in time Δt or 1 arrival, one departure in time Δt or 2 arrivals, 2 departures in time Δt)}

$$= P_1(t)\left[\{\mu\Delta t + O(\Delta t)\}\{1-\lambda\Delta t + O(\Delta t)\} + \text{The terms which ultimately become equal to } O(\Delta t)\right]$$

$$+P_2(t)\left[\{0(\Delta t)\}\{1-\lambda\Delta t + O(\Delta t)\} + \text{The terms which ultimately become equal to } O(\Delta t)\right]$$

$$+P_3(t)O(\Delta t) + \text{The terms which ultimately equal to } O(\Delta t)\Big]$$

$P_O(t)$ [{Probability of no departure which 1 as there is no customer in the system and hence no chance of departure}

$\{1-\lambda\Delta t+O(\Delta t)\}\{\{\mu\Delta t+O(\Delta t)\}\{\lambda\Delta t+O(\Delta t)\}$ + The terms which ultimately become equal to $O(\Delta t)]$

Therefore, the arrival-departure equations for the M/M/1 Queuing model are:

$$P_n(t+\Delta t)=P_{n-1}(t)\left[\lambda\Delta t+O(\Delta t)\right]+P_{n+1}(t)\left[\mu\Delta t+O(\Delta t)\right]$$

$$+P_n(t)\left[\lambda\Delta t+O(\Delta t)\right]\left[\mu\Delta t+O(\Delta t)\right]$$

$$+P_n(t)[1-\lambda\Delta t][1-\mu\Delta t]+O(\Delta t),\ n\geq 1$$

$$P_O(t+\Delta t)=P_1(t)\left[\mu\Delta t+O(\Delta t)\right]+P_O(t)\left[\lambda\Delta t+O(\Delta t)\right]$$

$$+P_O(t)[1-\lambda\Delta t]\cdot 1+O(\Delta t),\ n=0 \qquad (1)$$

$$P_n(t+\Delta t)=P_{n-1}(t)\lambda\Delta t+P_{n+1}(t)\mu\Delta t$$

$$+P_n(t)[1-\lambda\Delta t-\mu\Delta t]+O(\Delta t),\ n\geq 1$$

$$P_O(t+\Delta t)=P_1(t)\mu\Delta t+P_0(t)[1-\lambda\Delta t]+O(\Delta t) \qquad (2)$$

Dividing equation (2) by Δt and taking the limit as Δt tends to zero, we have

$$\underset{\Delta t\to 0}{\text{limit}}\frac{P_n(t+\Delta t)-P_n(t)}{\Delta t}=\lambda P_{n-1}(t)+\mu P_{n+1}(t)-(\lambda+\mu)P_n(t)$$

$$\Rightarrow P_n^1(t)=\lambda P_{n-1}(t)+\mu P_{n+1}(t)-(\lambda+\mu)P_n(t) \qquad (3)$$

and

$$P_O^1(t)=-\lambda P_O(t)+\mu P_1(t) \qquad (4)$$

When steady state, i.e., the equilibrium state, is reached, $P_n^1(t)$ becomes independent of time, say p_n, and the rate of its change with respect to time becomes zero, i.e.,

$P_n^1(t)=0$.

Therefore, the **steady-state solution** is given by

$$\lambda P_{n-1}+\mu P_{n+1}-(\lambda+\mu)P_n=0,\ n\geq 1$$

$$\mu P_1=\lambda P_0 \qquad (5)$$

Now, from equation (5), we have

$$\mu P_{n+1} - \lambda P_n = \mu P_n - \lambda P_{n-1} \quad (6)$$

This implies that

$\mu P_n - \lambda P_{n-1} = \mu P_{n-1} - \lambda P_{n-2}$ (changing n to $n-1$)

$\mu P_{n-1} - \lambda P_{n-2} = \mu P_{n-2} - \lambda P_{n-3}$ (again, changing n to $n-1$)

$\mu P_{n-2} - \lambda P_{n-3} = \mu P_{n-3} - \lambda P_{n-4}$ (again, changing n to $n-1$)

…

…

…

$\mu P_2 - \lambda P_1 = \mu P_1 - \lambda P_0$

But $\mu P_1 - \lambda P_0 = 0$ (from (5))

Therefore, $\mu P_{n+1} - \lambda P_n = \mu P_n - \lambda P_{n-1}$

$= \mu P_{n-1} - \lambda P_{n-2}$

$\vdots$

$= \mu P_1 - \lambda P_0 = 0$

$$\Rightarrow P_n = \frac{\lambda}{\mu} P_{n-1}$$

$$= \frac{\lambda}{\mu}\left(\frac{\lambda}{\mu} P_{n-2}\right) \quad (\because \mu P_{n-1} - \lambda P_{n-2} = 0 \Rightarrow P_{n-1} = \frac{\lambda}{\mu} P_{n-2}$$

$$= \frac{\lambda}{\mu}\frac{\lambda}{\mu}\cdot\left(\frac{\lambda}{\mu} P_{n-3}\right)\left(\because \mu P_{n-2} - \lambda P_{n-3} = 0 \Rightarrow P_{n-2} = \frac{\lambda}{\mu} P_{n-3}\right)$$

$$\vdots$$

$$= \frac{\lambda \cdot \lambda \cdot \lambda \ldots \lambda (n \text{ times})}{\mu \cdot \mu \cdot \mu \ldots \mu (n \text{ times})} P_0 = \frac{\lambda^n}{\mu^n} P_0$$

$$= \left(\frac{\lambda}{\mu}\right)^n P_0 = P^n P_0.$$

Since $\sum_{n=0}^{\infty} P_n = 1$, it follows that

$$\sum_{n=0}^{\infty} P_n = \sum_{n=0}^{\infty} P_0 P^n = 1 \quad (\because P_n = P\ldots\ldots\ldots\ldots)$$

$$\Rightarrow P_0\left(1+HP+P^2+\ldots+P^n+\ldots\right)=1$$

$$\Rightarrow P_0\left(\frac{1}{1-P}\right)=1$$

$$\Rightarrow P_0=1-P \text{ Where } P=\frac{\lambda}{\mu}$$

Therefore, the probability of n customers (units) in the system is given by

$$P_n=(1-P)P^n, n\geq 0$$

Model → M/M/1 or M/M/1: ∞/FCFS

Probability of n customers in the system is given by

$$P_n=(1-\rho)\rho^n, n\geq 1$$

Probability of zero customers (units) in the queue/system is given by $P_0=1-\rho$

5.5.2. Operating Characteristics or Measures of Performances of the Model M/M/1:

1. The average number of customers (units) in the system is given by

$$L_s=E(n)=\sum_{n=0}^{\infty}nP_n=\sum_{n=0}^{\infty}n(1-\rho)\rho^n=(1-\rho)\sum_{n=0}^{\infty}n\rho^n \quad \ldots(*)$$

Now, consider $\sum_{n=0}^{\infty}n\rho^n=0+\rho+2\rho^2+3\rho^3+\ldots$

Let $S=\sum_{n=0}^{\infty}n\rho^n=\rho+2\rho^2+3\rho^3+\ldots$

$$\Rightarrow \rho S=\rho^2+2\rho^3+3\rho^4+\ldots$$

$$\Rightarrow S-\rho S=\left(\rho+2\rho^2+3\rho^3+\ldots\right)-\left(\rho^2+2\rho^3+3\rho^4+\ldots\right)$$

$$\Rightarrow S(1-\rho)=\rho+\rho^2+\rho^3+\ldots$$

$$\Rightarrow S(1-\rho)=\frac{\rho}{1-\rho}$$

$$\Rightarrow S=\frac{\rho}{(1-\rho)^2}$$

Thus from (*), we have $L_s=(1-\rho)S=\frac{(1-\rho)\rho}{(1-\rho)^2}=\frac{\rho}{1-\rho}$

Now, ρ is the traffic intensity or utilisation factor, where $\rho = \frac{\lambda}{\mu}$

Thus, $L_s = \frac{\rho}{1-\rho} = \frac{\frac{\lambda}{\mu}}{1-\frac{\lambda}{\mu}} = \frac{\lambda}{\mu-\lambda}$ (1)

Where, λ denotes the average arrival rate and μ denote the average service rate.

2. The average queue length (Expected number of customers) is L_q

And $L_q = L_s -$ Traffic intensity

$$= \frac{\lambda}{\mu-\lambda} - \frac{\lambda}{\mu} = \frac{\lambda^2}{\mu(\mu-\lambda)} \quad (2)$$

3. The average time an arrival spends in the system (Expected waiting time in system) is $W_s = \frac{L_s}{\lambda} = \frac{\lambda}{\lambda(\mu-\lambda)} = \frac{1}{\mu-\lambda}$ (from (1))

Thus $W_s = \frac{1}{\mu-\lambda}$ (3)

4. The average waiting time of an arrival in the queue is W_q.

And $W_q = \frac{L_q}{\lambda} = \frac{\lambda^2}{\lambda\mu(\mu-\lambda)} = \frac{\lambda}{\mu(\mu-\lambda)}$ (4)

from (4), we can also obtain the relation

$$W_q = W_s - \frac{1}{\mu}$$

5. The probability that the number of customers (units) waiting in the queue and the number of units being serviced is greater than k is

$$P[n > k] = \rho^{k+1} \quad (5)$$

6. The probability of having a queue i.e. $1 - P_0$

Thus $1 - P_0 = 1 - (1-\rho) = \rho$ (6)

Now, we explain the M/M/1 queuing model with the help of following examples.

Example 1:

A TV repairman finds that the time spent on his job has an exponential distribution with mean 30 minutes. If he repairs sets in the order in which these come in, and if the arrival of sets is approximately Poisson with an average rate of 10 per 8-hour day, what is the repairman's expected idle time each day? How many jobs are ahead of the average set just brought in?

Solution: Here, we have

Arrival rate $(\lambda) = \frac{10}{8}$ arrivals/hr

Service rate $(\mu) = \frac{60}{30} = 2$ services/hr

∴ The probability of the repairman being idle is

$$P_0 = 1 - \rho = 1 - \frac{\lambda}{\mu} = 1 - \frac{10}{8} \times \frac{1}{2} = 1 - \frac{5}{8} = \frac{3}{8}$$

Thus, $P_0 = \frac{3}{8}$.

Hence the idle time in the 8-hour day $= \frac{3}{8} \times 8$ hours $= 3$ hours

Now, the average number of jobs that are ahead of the set (person) just brought in

= Average number of units in the system means who are in queue and at counter, i.e.

$$L_s = \frac{\lambda}{\mu - \lambda} = \frac{\frac{10}{8}}{2 - \frac{10}{8}} = \frac{10}{6} = \frac{5}{3} = 1.666$$

Thus, $L_s = 1.67$.

Example 2.

Customers arrive at a window in a bank, according to a Poisson distribution with mean 10 per hour. Service time per customer is exponential with mean 5 minutes. The space in front of the window including that for the serviced customers can accommodate a maximum of three customers. Other customers can wait outside this space.

(a) What is the probability that an arriving customer can go directly to the space in front of the window?

(b) What is the probability that an arriving customer will have to wait outside the indicated space?

(c) How long is an arriving customer expected to wait before being served?

Solution:

Here, arrival rate $\lambda = 10$ per hour and service rate $(\mu) = \frac{60}{5}$ per hour=12 per hour

Thus, traffic intensity $(\rho) = \frac{\lambda}{\mu} = \frac{10}{\frac{60}{5}} = \frac{5}{6}$

$$\Rightarrow \rho = \frac{5}{6}$$

(a) The probability that an arriving customer can go directly to the space in front of the window i.e. the probability of n customers in the system here $2 \geq n \geq 0$, therefore the required probability $= P_0 + P_1 + P_2$

$$= \left(1-\frac{5}{6}\right) + \left(1-\frac{5}{6}\right)\frac{5}{6} + \left(1-\frac{5}{6}\right)\left(\frac{5}{6}\right)^2$$

$$= \frac{1}{6} + \frac{1}{6}\cdot\frac{5}{6} + \frac{1}{6}\cdot\frac{25}{36} \approx 0.42$$

(b) The probability that an arriving customer will have to wait outside the indicated space, i.e. the required probability $= P_3 + P_4 + P_5 + \ldots$

$$= 1 - \left(P_0 + P_1 + P_2\right)$$

$$= 1 - 0.42 = 0.58$$

(c) Expected waiting time before being served

$$= W_q = \frac{\lambda}{\mu(\mu - \lambda)} = \frac{10}{12(12-10)} = 0.417 \text{ hours}$$

Exercises:

1. Customers arrive at a box office window being served by a single individual according to a Poisson input process with a mean rate of 30 per hour. The time required to serve a customer has an exponential distribution with a mean of 90 seconds. Find the average waiting time of a customer in the queue.

Solution. $W_q = \frac{\lambda}{\mu(\mu-\lambda)} = \frac{30}{40(40-30)} = \frac{3}{10} hr$

$= \frac{3}{4} \times 60 \text{ minutes} = 4.5 \text{ minutes}$

2. A fertilizer company distributes its products by trucks loaded at its only loading station. Both company trucks and contractor's truck are used for this purpose. It was noticed that on an average on truck arrived every 5 minutes and the average loading time was 3 minutes. Forty percent of the trucks belong to the contractor. Determine the expected waiting time of contractor's trucks per day.

Solution. The expected waiting time of contractor's truck per day=8.64 hrs.

3. In a railway marshalling yard, goods train arrives at a rate of 36 trains per day. Assuming that inter-arrival time follows exponential distribution and the service time distribution is also exponential with an average of 30 minutes, calculate the following:

i. The mean line length
ii. The probability that the queue size exceeds 10.
iii. If the input increases to an average of 42 per day, what will the change in (a) and (b) be?

5.6. Multiple Server Models:

1.M/M/C or M/M/C: ∞/FCFS Queueing Model:

This model deals with a queueing system having a multiple server channel and no limit on system capacitywith Poisson input and exponential distribution for services while the customers are served on a "First Come First Served" basis. We consider a queue with Poisson input (having arrival rate, say, I) and with C ($1 \leq C < \infty$) parallel service channels having independently and identically distributed exponential service time distribution, each with rate, say, μ.

If there are n units in the system, and n is less than C, then, in all, n channels, are busy. If there are n ($\geq$ C) in the system, then all the F channels are busy. Thus, we have a birth death model having constant arrival (birth) rate λ and state-dependent service (death) rate.

$$\mu_n = \begin{cases} n\mu, n = 0, 1, 2, \ldots, C \\ C\mu, n = C+1, C+2, \ldots \end{cases}$$

Hence, the steady state probability exist if

$\rho = \dfrac{\lambda}{C\mu} < 1$ and are given by

$$P_n = \begin{cases} \dfrac{\lambda \cdot \lambda \cdot \ldots \lambda}{\mu \cdot 2\mu \ldots n\mu} P_0, \ 0 \le n < C \\ \dfrac{\lambda \cdot \lambda \ldots \lambda}{(\mu \cdot 2\mu \ldots C\mu)(C\mu \ldots C\mu)} P_0, \ n \ge C \end{cases}$$

where, the term $(c\mu \ldots c\mu)$ is repeated $(n-c)$ times.

$$= \begin{cases} \dfrac{\left(\dfrac{\lambda}{\mu}\right)^n}{n!} P_0, & 0 < n \le C \\ \dfrac{\left(\dfrac{\lambda}{\mu}\right)^n}{C!\, C^{n-C}} P_0, & n \ge C \end{cases}$$

Now, $\sum_{n=0}^{\infty} P_n = 1$

$$\Rightarrow \sum_{n=0}^{C-1} P_n + \sum_{n=C}^{\infty} P_n = 1 \Rightarrow P_0 \left[\sum_{n=0}^{C-1} \frac{\left(\frac{\lambda}{\mu}\right)^n}{n!} + \sum_{n=0}^{C-1} \frac{\left(\frac{\lambda}{\mu}\right)^n}{C!\, C^{n-C}} \right] = 1$$

$$\Rightarrow P_0 \left[\sum_{n=0}^{C-1} \frac{C^n \rho^n}{n!} + \sum_{n=C}^{\infty} \frac{C^C \cdot \rho^n}{C!} \right] = 1, \text{ where } \rho = \frac{\lambda}{C\mu}$$

$$\Rightarrow P_0 \left[\sum_{n=0}^{C-1} \frac{C^n \rho^n}{n!} + \sum_{n=C}^{\infty} \frac{C^C \rho^n}{C!} \right]^{-1}$$

$$= \left[\sum_{n=0}^{C-1} \frac{C^n \rho^n}{n!} + \frac{C^C}{C!} \rho^C \frac{1}{1-\rho} \right]^{-1}$$

Operating characteristics/Performance measures of M/M/C Model

The meaning of performances measures such as $L_q, L_s, W_q,$ and W_s, here we only evaluate mathematical expression for all these as given below:

1. $L_q = \sum_{n=C}^{\infty}(n-C)P_n$

$= \sum_{j=1}^{\infty} jP_{C+j}$, Now put $n - C = j$, we get

$= P_0 \sum_{j=1}^{\infty} \frac{j\left(\frac{\lambda}{\mu}\right)^{c+j}}{C!C^j}$

$= \frac{P_0\left(\frac{\lambda}{\mu}\right)^C}{C!} \sum_{j=1}^{\infty} j\rho^j$

$= \frac{P_0\left(\frac{\lambda}{\mu}\right)^C}{C!} \frac{\rho}{(1-\rho)^2}$

$\left[\because \rho\sum_{j=1}^{\infty} j\rho^{j-1} = \rho\sum_{j=1}^{\infty}\frac{d}{d\rho}(\rho^j) = \rho\frac{d}{d\rho}\left(\frac{\rho}{1-\rho}\right) = \rho\left[\frac{(1-\rho)-\rho(-1)}{(1-\rho)^2}\right]\right]$

2. $L_s = L_q + \frac{\lambda}{\mu}$ (Expected number of customers in the system)

3. Expected waiting time in the queue

$W_q = \frac{1}{\lambda}\cdot L_q$

4. Expected waiting time in the system

$W_s = W_q + \frac{1}{\mu}$ or $W_s = \frac{L_s}{\lambda}$

The following examples illustrate M/M/C queueing Model.

Example 1:

A petroleum company is considering expansion of its unloading facility at its refinery. Due to random of variations in weather, loading delays and other factors, ships arriving at the refinery to unload crude oil arrive at an average rate of 5 ships per week. Service rate on an average is 10 ships per week. Assuming Poisson arrivals and exponential service distributions, find

(a) The average time a ship must wait before beginning to deliver its cargo to the refinery;

(b) If a second birth is rented, what will be the average number of ships waiting before being unloaded?

(c) What would be the average time a ship would wait before being unloaded with two berths?

(d) What is the average number of idle berths at any specified time?

Solution: (a) Here, we have the case of M/M/1 model

$\lambda = 5$ per week, $\mu = 10$ per week

$$W_q = \frac{\lambda}{\mu(\mu - \lambda)} = \frac{1}{10} \text{ week}$$

(b) For this case, we have M/M/2 model

Here $\rho = \frac{\lambda}{C\mu} = \frac{5}{2 \times 10} = \frac{1}{4}$

$$P_0 = \left[1 + \frac{\lambda}{\mu} + \frac{\left(\frac{\lambda}{\mu}\right)^2}{2!} \cdot \frac{1}{1 - \frac{1}{4}}\right]^{-1}$$

$$= \left[1 + \frac{1}{2} + \frac{1}{8} \times \frac{4}{3}\right] = \left(\frac{3}{2} + \frac{1}{6}\right)^{-1} = \left(\frac{9+1}{6}\right)^{-1} = \frac{3}{5}$$

$$L_q = \frac{P_0 \left(\frac{\lambda}{\mu}\right)^C \rho}{C!(1-\rho)^2} = \frac{1}{30}$$

(c) $W_q = \frac{L_q}{\lambda} = 150$ week

(d)
$$\begin{array}{lll} X: & 1 & 2 \\ P: & P_1 & P_0 \end{array}$$

Where X be the number of idle berths and probability of number of ships in the system

$$E(X) = \sum PX = P_1 + 2P_0 = 1.5$$

Hence 2 births are idle.

Example 2:

Arrival of machinists at a tool crib are considered to be Poisson distributed at an average rate 6 per hour. The length of time the machinists must remain at the tool crib is exponentially with average time of 0.05 hours.

a) What is the probability that a machinist arriving at the tool crib will have to wait?

b) What is the average number of machinists at the tool crib?

c) The company will install a second tool crib when convinced that a machinist would have to spend 6 minutes in waiting and being served at the tool crib. At what rate should the arrival of machinist to the tool crib increase to justify the addition of a second crib?

Solution. a) Here the arrival rate λ = 6/hr and the service rate μ = 20/hr

Therefore, the probability of zero customers in queue is

$p_0 = 1 - \rho$ where $\rho = \frac{\lambda}{\mu} = \frac{6}{20}$

$p_0 = 1 - \frac{6}{20} = \frac{14}{20} = \frac{7}{10} = 0.7$

The probability that a machinist arriving at the tool crib will have to wait

= Probability that there is at least one machinist at the tool crib

= 1- Probability that there is no machinist at the tool crib

= 1- p_0 = 1- 0.7 = 0.3

b) The average number of machinists at the tool crib is given by

$$L_s = \frac{\lambda}{\mu - \lambda} = \frac{6}{20-6} = \frac{6}{14} = 0.428$$

c) The company is ready to install a second tool crib when convinced that a machinist would have to spend 6 min. in waiting and being served. Let the increased arrival rate be λ'.

Waiting time in the system = 6 min. $= \frac{1}{10}$ hr.

We have $\frac{1}{\mu - \lambda'} = \frac{1}{10}$

or $\frac{1}{20 - \lambda'} = \frac{1}{10}$

or $10 = 20 - \lambda'$

or $\lambda' = 20 - 10 = 10$/hr

The increase is, therefore, (10 – 6)/hr = 4/hr.

Example 3:

A repairman is to be hired to repair machines which break down at an average rate of 3 per hour. The breakdown follows a Poisson distribution. Non-productive time of a machine is considered to cost 10 Rs per hour. Two repairmen have been interviewed of whom one is slow but charges less and the other is fast but more expensive. The slow repairman charges 5 Rs per hour and services breakdown machines at the rate of 4 per hour. The fast repairman demands 7 Rs per hour, but services breakdown machines at an average rate of 6 per hour. Which repairman should be hired?

Solution. The data given is summarized below:

Slow/less expensive Repairman:

$\lambda = 3/\text{hr}$, $\mu = 4/\text{hr}$, Labour cost = 5 Rs/ hr

Fast/more expensive Repairman:

$\lambda = 3/\text{hr}$, $\mu = 6/\text{hr}$, Labour cost = 7 Rs/ hr

Case of Slow/less expensive Repairman:

Cost of engaging slow repairman for 8 hours

= Breakdown cost + Labour cost for 8 hour working day

= (No. of breakdowns per hr) $\times$ 8 hours $\times$ Av. Time spent in the system $\times$ (Breakdown cost/ cost for non-productive time) +

Labour cost per hr $\times$ 8 hours

$= \lambda \times 8 \times w_s \times 10 + 5 \times 8$

Since the average time spent in the system is $W_s = \frac{1}{\mu-\lambda} = \frac{1}{4-3} = 1$,

The total cost for engaging the slow repairman is equal to

$3 \times 8 \times 1 \times 10 + 5 \times 8 = 240 + 40 = 280$ Rs.

Case of Fast/more expensive Repairman:

Cost of engaging fast repairman for 8 hours

= Breakdown cost + Labour cost in 8 hour working day.

= (No. of breakdowns per hr) × 8 hours × Av. Time spent in the System × (Breakdown cost/ cost for non-productive time) + Labour cost per hr × 8 hours

$= \lambda \times 8 \times W_s \times 10 + 7 \times 8$

Since the average time spent in the system is $W_s = \dfrac{1}{\mu - \lambda}$

$$= \frac{1}{6-3} = \frac{1}{3},$$

The total cost for engaging the Fast repairman is

$$3 \times 8 \times \frac{1}{3} \times 10 + 7 \times 8 = 80 + 56 = 136 \text{ Rs}$$

Since the total cost of engaging the fast repairman is less, we should engage the fast repairman.

2. M/M/1/K Queueing Model:

In this model, the capacity of the system is limited, say *k*, hence the maximum size of a queue is *k*. Here, we have single service channel. The server, serving each customer according to the exponential distribution with an average of μ customers per unit of time.

Steady State Difference Equations:

The simplest way of starting this is to treat the model as a special case of birth death process, where

$$\lambda_n = \begin{cases} \lambda, & 0 \le n < k \\ 0, & \text{elsewhere} \end{cases}$$

and $\mu_n = \mu$ for n=1,2,3, ...

Now, following the similar arguments as given in M/M/1 model, we obtain

$$P_0(t+\Delta t) = P_0(t)[1-\lambda\Delta t] + P_1(t)\mu\Delta t + 0(\Delta t),\ n=0$$

$$P_n(t+\Delta t) = P_n(t)[1-(\lambda+\mu)\Delta t] + P_{n-1}(t)\lambda\Delta t + P_{n+1}(t)\mu\Delta t + 0(\Delta t),$$

for $n = 1, 2, \ldots, k-1$ and

$$P_k(t+\Delta t) = P_k(t)[1-(0+\mu)\Delta t] + P_{k-1}(t)\lambda\Delta t + 0x\mu\Delta t + 0(\Delta t)$$

$$= P_k(t)[1-\mu\Delta t] + P_{k-1}(t)\lambda\Delta t + 0$$

Now, dividing above three equations by Δt and taking limit as $\Delta t \to 0$, these equations transform into

$$P_0^1(t) = -\lambda P_0(t) + \mu P_1(t) \text{ for } n = 0$$

$$P_n^1(t) = -(\lambda+\mu)P_n(t) + \lambda P_{n-1}(t) + \mu P_{n+1}(t) \text{ for } n = 1, 2, \ldots, k-1$$

and $P_k^1(t) = -\mu P_k(t) + \lambda P_{k-1}(t)$, for $n = k$

Thus, the steady-state equations are:

$$\lambda P_{n-1} + \mu P_{n+1} - (\lambda+\mu)P_n = 0, 1 \le n < k \quad (1)$$

$$-\lambda P_0 + \mu P_1 = 0 \quad (2)$$

$$\mu P_k - \lambda P_{k-1}, n = k \quad (3)$$

for $1 \le n < k$, by (1) we have

$$\mu P_{n+1} - \lambda P_n = \mu P_n - \lambda P_{n-1}$$

$$= \mu P_{n-1} - \lambda P_{n-2} \text{ (on replacing } n \text{ by } n-1)$$

...

...

...

$$= \mu P_1 - \lambda P_0$$

Now, from (2), (3) and the above equation, we have

$$\mu P_{n-}\lambda P_{n-1} = 0; 0 \le n \le k$$

$$\Rightarrow P_n = \frac{\lambda}{\mu} P_{n-1}$$

$$= \frac{\lambda}{\mu} \cdot \frac{\lambda}{\mu} P_{n-2}$$

$$= \left(\frac{\lambda}{\mu}\right)^n P_0 = \rho^n P_0, \; n = 0, 1, 2, \ldots, k$$

But $\sum_{n=0}^{k} P_n = 1$

Therefore, $\sum_{n=0}^{k} \rho^n P_0 = 1$

$$\Rightarrow P_0\left(1 + \rho + \rho^2 + \ldots + \rho^k\right) = 1$$

$$\Rightarrow P_0 = \frac{1}{1 + \rho + \rho^2 + \ldots \rho k}$$

$$= \begin{cases} \dfrac{1-\rho}{1-\rho^{k+1}}, & \text{if } \rho \neq 1 \\ \dfrac{1}{k+1}, & \text{if } \rho = 1 \end{cases}$$

$$\therefore P_n = \begin{cases} \dfrac{\rho^n(1-\rho)}{1-\rho^{k+1}} & \text{if } \rho \neq 1 \\ \dfrac{1}{k+1} & \text{if } \rho = 1 \end{cases}$$

Operating Characteristics/Performances Measures of M/M/1/K

or M/M/1: K/FCFS Queuing Model

i. Expected number of customers in the system i.e.

$$L_s = E(n) = \sum_{n=0}^{k} nP_n$$

$$= \begin{cases} \sum_{n=0}^{k} n \dfrac{\rho^n(1-\rho)}{1-\rho^{k+1}}, & \text{if } \rho \neq 1 \\ \sum_{n=0}^{k} n \cdot \dfrac{1}{k+1}, & \text{if } \rho = 1 \end{cases}$$

$$= \begin{cases} \dfrac{1-\rho}{1-\rho^{k+1}} \rho \sum_{n=1}^{k} n\rho^{n-1}, & \text{if } \rho \neq 1 \\ \dfrac{1}{k+1}\left[1+2+3+\ldots+k\right], \text{if } \rho = 1 \end{cases}$$

But $\sum_{n=1}^{k} n\rho^{n-1} = \frac{d}{d\rho}\left[\sum_{n=1}^{k} \rho^n\right]$

$$=\frac{d}{d\rho}\left(\rho+\rho^2+\rho^3+\ldots+\rho^k\right)=\frac{d}{d\rho}\left(\frac{\rho\left(1-\rho^k\right)}{1-\rho}\right)$$

$$=\frac{(1-\rho)\left\{\rho\left(-k\rho^{k-1}\right)+\left(1-\rho^k\right)\right\}-\rho\left(1-\rho^k\right)(-1)}{(1-\rho)^2}$$

$$=\frac{-k\rho^k}{1-\rho}+\frac{\left(1-\rho^k\right)(1-\rho+\rho)}{(1-\rho)^2}=\frac{1-\rho^k}{(1-\rho)^2}-\frac{k\rho^k}{1-\rho}$$

$$\therefore L_s=\begin{cases}\frac{1-\rho}{1-\rho^{k+1}}\left[\frac{\rho\left(1-\rho^k\right)}{(1-\rho)^2}-\frac{k\rho^{k+1}}{1-\rho}\right], & \text{if } \rho\neq 1\\ \frac{k}{2}, & \text{if } \rho=1\end{cases}$$

$$=\begin{cases}\frac{\rho}{1-\rho^{k+1}}\left[\frac{\left(1-\rho^k\right)}{1-\rho}-\frac{k\rho^k}{1}\right], & \text{if } \rho\neq 1\\ \frac{k}{2}, & \text{if } \rho=1\end{cases}$$

$$=\begin{cases}\frac{\rho}{1-\rho^{k+1}}\left[\frac{1-\rho^{k+1}+\rho^{k+1}-\rho^k}{1-\rho}-k\rho^k\right], & \text{if } \rho\neq 1\\ \frac{k}{2}\end{cases}$$

$$=\begin{cases}\frac{\rho}{1-\rho^{k+1}}\left[\frac{1-\rho^{k+1}}{1-\rho}+\frac{\rho^{k+1}-\rho^k-k\rho^k+k\rho^{k+1}}{1-\rho}\right], & \text{if } \rho\neq 1\\ \frac{k}{2}, & \text{if } \rho=1\end{cases}$$

$$=\begin{cases}\frac{\rho}{1-\rho^{k+1}}\left[\frac{1-\rho^{k+1}}{1-\rho}+\frac{-\rho^k(1-\rho)-k\rho^k(1-\rho)}{1-\rho}\right], & \text{if } \rho\neq 1\\ \frac{k}{2}, & \text{if } \rho=1\end{cases}$$

$$=\begin{cases}\frac{\rho}{1-\rho}-\frac{(k+1)\rho^{k+1}}{1-\rho^{k+1}}, & \text{if } \rho\neq 1\\ \frac{k}{2}, & \text{if } \rho=1\end{cases}$$

ii. Expected queue length i.e.

$$L_q=\sum_{n=1}^{k}(n-1)P_n=\sum_{n=1}^{k}nP_n-\left(1-P_0\right)$$

$$= L_s - (1 - P_0)$$

Note: The relationships $W_s = \frac{L_s}{\lambda}$ and $W_q = \frac{L_q}{\lambda}$ as already given for model M/M/1 are not valid here. This is because no arrivals are allowed to join the system once the maximum allowable length is reached. However, if λ is replaced by $\lambda^1 = \lambda(1 - P_k)$ which is the effective rate of arrival, then

$$W_s = \frac{L_s}{\lambda^1}, W_q = W_s - \frac{1}{\mu}$$

Remark: For M/M/1/K model, the assumption $\lambda < \mu$ is not necessary for deriving steady state results. The steady-state probabilities can be obtained for $\lambda = \mu$ also and in this case,

$$P_n = \frac{1}{k+1}, n = 1, 2, \ldots, N$$

and

$$L_s = \frac{k}{2}, W_s = \frac{k+1}{2\mu}, W_q = \frac{k-1}{2\mu}, L_q = \frac{k(k-1)}{2(k+1)}$$

Example 1:

At a railway station, only one train is handled at a time. The railway yard is sufficient only for two trains to wait while others are given a signal to leave the station. Trains arrive at an average rate of 6 per hour and the railway station can handle them on an average of 12 per hour. Assuming Poissonarrivals and exponential service time distributions, find the steady-state probabilities for various number of trains in the system. Also find the average waiting time of a new train coming in the system.

Solution. Here $\lambda = 6$ per hour, $\mu = 12$ per hour, $k = 2$, $\rho = \frac{\lambda}{\mu} = \frac{1}{2}$

Steady-state probabilities are:

$$P_0 = \frac{1-\rho}{1-\rho^{2+1}} = \frac{1-\rho}{1-\rho^3} = \frac{4}{7}$$

$$P_n = \rho^n \cdot \rho_0; n = 0, 1, 2, \ldots$$

$$= \frac{1}{2^n} \cdot \frac{4}{7}$$

Now, $\sum_{n=0}^{2} P_n = 1 \Rightarrow P_0\left[1 + \frac{1}{2} + \frac{1}{4}\right] = 1 \Rightarrow P_0 = \frac{4}{7}$

$$\therefore P_n = \frac{1}{2^n} \cdot \frac{4}{7}; n = 0, 1, 2$$

The average waiting time of an incoming train

$$W_s = \frac{L_s}{\lambda^1}, \text{ where } L_s = \frac{\rho}{1-\rho} - \frac{(k+1)\rho^{k+1}}{1-\rho^{k+1}}$$

$$= \frac{\frac{1}{2}}{1-\frac{1}{2}} - \frac{3\left(\frac{1}{2}\right)^3}{1-\left(\frac{1}{2}\right)^3}$$

$$= 1 - \frac{\frac{3}{8}}{\frac{7}{8}} = \frac{4}{7}$$

$$\lambda' = \lambda[1-P_2] = 6\left[1-\frac{1}{4}\cdot\frac{4}{7}\right] = \frac{36}{7}$$

$$\therefore W_s = \frac{\frac{4}{7}}{\frac{36}{7}} = \frac{1}{9} \text{ hours}$$

3.M/M/C/K Queueing Model:

In this system, the maximum size of a queue is k and the number of servers is C. For example, a car servicing station may have only facilities for offering services to C cars at a time. However, because of space limitation, the station can accept only *k* cars at any one point of time of servicing C (<k) cars.

The queuing system can be treated as a special case of the birth death process where

$$\lambda_n = \begin{cases} \lambda, 0 \le n < k \\ 0, \text{elsewhere} \end{cases} \quad \text{and} \quad \mu_n = \begin{cases} n\mu, 0 \le n < C \\ C\mu, C \le n \le k \end{cases}$$

Hence, the steady state probabilities are given by

$$P_n = \begin{cases} \dfrac{\lambda\cdot\lambda\cdot\lambda\ldots\lambda}{\mu\cdot 2\mu\cdot\ldots n\mu}\cdot P_0; & 0 \le n < C \\ \dfrac{\lambda\cdot\lambda\cdot\ldots\lambda}{\mu\cdot 2\mu\ldots C\mu\cdot\underbrace{(C\mu\cdot C\mu\cdot - C\mu)}_{n-C \text{ times}}}\, P_0, & C \le n \le k \end{cases}$$

$$= \begin{cases} \dfrac{(\lambda/\mu)^n}{n!} P_0, & C \le n \le k \\ \dfrac{\left(\frac{\lambda}{\mu}\right)^n}{C!C^{n-C}}, & n > k \end{cases} \qquad \text{Where, } P_0 \text{ is given by } \sum_{n=0}^{k} P_n = 1$$

$$\Rightarrow \sum_{n=0}^{C} \frac{\left(\frac{\lambda}{\mu}\right)^n}{n!} P_0 + \sum_{n=C+1}^{k} \frac{\left(\frac{\lambda}{\mu}\right)^n}{C!C^{n-C}} P_0 = 1$$

$$\Rightarrow P_0 = \left[\sum_{n=0}^{C} \frac{\left(\frac{\lambda}{\mu}\right)^n}{n!} + \sum_{n=C+1}^{k} \frac{C^C}{C!} \left(\frac{\lambda}{C\mu}\right)^n \right]^{-1}$$

$$= \left[\sum_{n=0}^{C} \frac{C^n \rho^n}{n!} + \sum_{n=C+1}^{k} \frac{C^C \rho^n}{C!} \right]^{-1}, \text{ where } \rho = \frac{\lambda}{C\mu}$$

Operating Characteristics/Performance Measures of the M/M/C/K Queueing Model:

1. $L_s = \sum_{n=1}^{k} n P_n$
2. $L_q = \sum_{n=C+1}^{k} (n-C) P_n$
3. $W_s = \frac{L_s}{\lambda'}$, where $\lambda' = \lambda(1-P_k)$
4. $W_q = \frac{L_q}{\lambda'}$ W, where $\lambda' = \lambda(1-P_k)$

Example 1:

A car servicing station has two bays where service can be offered simultaneously. Dueto a limitation in space, only four cars are accepted for servicing. The arrival pattern is Poisson with 12 cars per day. The service time for both the bays is exponentially distributed with y 8 cars per day per bay. Find the average number of cars in the service station, the average number of cars waiting to be serviced and the average time a car spends in the system.

Solution. Here, $\lambda = 12, \mu = 8, C = 2$

$$\rho = \frac{\lambda}{C\mu} = \frac{12}{2 \times 8} = \frac{3}{4}, k = 4$$

Therefore, $P_0 = \left[\sum_{n=0}^{C} \frac{C^n \rho^n}{n!} + \sum_{n=C+1}^{k} \frac{C^C \cdot \rho^n}{C!} \right]^{-1}$

$$= \left[\sum_{n=0}^{2} \frac{2^n \cdot \left(\frac{3}{4}\right)^n}{n!} + \sum_{n=3}^{4} \frac{2^2 \left(\frac{3}{4}\right)^n}{2!} \right]^{-1}$$

$$= \left[\left(1 + \frac{3}{2} + \frac{\frac{9}{4}}{2!} \right) + 2\left(\frac{27}{64} + \frac{81}{256} \right) \right]^{-1}$$

$$= \left[1 + \frac{3}{2} + \frac{9}{8} + \frac{27}{32} + \frac{81}{128} \right]^{-1}$$

$$=\left[\frac{128+192+144+108+81}{128}\right]^{-1}=\frac{128}{653}$$

$$P_1=\frac{\left(\frac{3}{2}\right)^1}{1!}\cdot P_0=\frac{3}{2}\cdot\frac{128}{653}=\frac{192}{653}$$

$$P_2=\frac{\left(\frac{3}{2}\right)^2}{2!}\cdot P_0=\frac{\left(\frac{3}{2}\right)^2}{2!}\cdot\frac{128}{653},$$

$$P_3=\frac{\left(\frac{3}{2}\right)^3}{2!\cdot 2^{3-2}}P_0=\frac{\left(\frac{3}{2}\right)^3}{2!\cdot 2}\frac{128}{653},$$

$$P_4=\frac{\left(\frac{3}{2}\right)^4}{2!\cdot 2^{4-2}}\frac{128}{653}.$$

Now, we can easily evaluate L_s, L_q and W_s

5.7. (M/G/1):GD/∞/∞)-Pollaczek-Khintechine Formula:

Queuing models in which the arrivals and departures do not follow the Poisson distribution are complex. In general, it is advisable to use simulation as an alternative tool for analysing these situations.

This section presents one of the few non-Poisson queues for which analytic results are available. It deals with the case in which the service time t, is represented by any probability distribution with mean E{t} and variance var{t}. The results of the model include the basic measures of performance L_s, L_q, W_s and W_q .The model does not provide a closed –form expression for P_n because of analytic intractability.

Let λ be the arrival rate at the single-server facility. Given E{t} and var{t} of the service time distribution and that λ E{t}<1 , it can be shown using sophisticated probability/ Markov chain analysis that

$$L_s=\lambda\, E\{t\}+\frac{\lambda^2(E^2\{t\}+var\{t\})}{2(1-\lambda\, E\{t\})},\lambda\, E\{t\}<1$$

The probability that the facility is empty (idle) is computed as

$P_0 = 1 - \lambda E\{t\} = 1 - \rho$

Because $\lambda_{eff} = \lambda$, the remaining measures of performance L_q, W_s and W_q can be derived from as explained in section 5.4

Example 1:

Automata car wash facility operates with only one bay. Cars arrive according to a Poisson distribution with a mean of 4 cars per hour, and may wait in the facility's parking lot if the bay is busy. The time for washing and cleaning a car is exponential, with a mean of 10 minutes. Cars that cannot park in the lot can wait in the street bordering the wash facility. This means that, for all practical purpose, there is no limit on the size of the system. The manager of the facility wants to determine the size of the parking lot.

For this situation, we have $\lambda = 4$ cars per hour.How does the new system affect the operation of the facility?

Solution:

Let $\lambda_{eff} = \lambda = 4$ cars per hour .

The service time is constant so that E{t}= $\frac{10}{60} = \frac{1}{6}$ hour and var{t}=0. Thus,

$$L_s = 4\left(\frac{1}{6}\right) + \frac{4^2\left(\left(\frac{1}{6}\right)^2 + 0\right)}{2\left(1 - \frac{4}{6}\right)} = 1.33\ cars$$

$$L_q = 1.333 - \left(\frac{4}{6}\right) = 0.667 \text{ cars.}$$

$$W_s = \frac{1.333}{4} = 0.333 \text{ hour}$$

$$W_q = \frac{0.667}{4} = 1.67 \text{ hour}$$

It is intersecting that even though the arrival and departure rates are the same as in the Poisson case, the expected waiting time is lower in the current model because the service time is constant,

	(M/M/1): (GD/∞/∞)	(M/D/1): (GD/∞/∞)
W_s (hr)	0.500	0.333
W_q(hr)	0.333	0.167

The results make sense because a constant service time indicates more certainty in the operation of the facility.

Study Learning Material Prepared by

Dr. S. KALAISELVI M.SC., M.Phil., B.Ed., Ph.D.,

Assistant Professor,

Department of Mathematics,

Sarah Tucker College (Autonomous), Tirunelveli-627007.