UNIT – II: TRANSPORTATION PROBLEM

FORMULATING TRANSPORTATION PROBLEMS

In general, a transportation problem is specified by the following information:

- A set of *m* ***supply points*** from which a good/service is shipped. Supply point *i* can supply at most s_i units.
- A set of *n* ***demand points*** to which the good/service is shipped. Demand point *j* must receive at least d_j units.
- Each unit produced at supply point *i* and shipped to demand point *j* incurs a variable cost of c_{ij}.

The relevant data can be formulated in a ***transportation tableau***:

	Demand point 1	Demand point 2	.....	Demand point *n*	SUPPLY
Supply point 1	c_{11}	c_{12}		c_{1n}	s_1
Supply point 2	c_{21}	c_{22}		c_{2n}	s_2
.....					
Supply point *m*	c_{m1}	c_{m2}		c_{mn}	s_m
DEMAND	d_1	d_2		d_n	

If total supply equals total demand then the problem is said to be a ***balanced transportation problem***.

Let x_{ij} = number of units shipped from supply point *i* to demand point *j*

- Decision variable x_{ij}: number of units shipped from supply point *i* to demand point *j*

then the general LP representation of a transportation problem is

$$\min \sum_i \sum_j c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_j x_{ij} \le s_i \ (i=1,2, \ldots, m) \qquad \text{Supply constraints}$$

www.Jntufastupdates.com

$\sum_i x_{ij} \geq d_j$ (j=1,2, ..., n) Demand constraints

$x_{ij} \geq 0$

If a problem has the constraints given above and is a *maximization* problem, it is still a transportation problem.

Formulating Balanced Transportation Problem Example 1.

Powerco

Powerco has three electric power plants that supply the needs of four cities. Each power plant can supply the following numbers of kwh of electricity: plant 1, 35 million; plant 2, 50 million; and plant 3, 40 million. The peak power demands in these cities as follows (in kwh): city 1, 45 million; city 2, 20 million; city 3, 30 million; city 4, 30 million. The costs of sending 1 million kwh of electricity from plant to city is given in the table below. To minimize the cost of meeting each city's peak power demand, formulate a balanced transportation problem in a transportation tableau and represent the problem as a LP model.

	To			
From	City 1	City 2	City 3	City 4
Plant 1	$8	$6	$10	$9
Plant 2	$9	$12	$13	$7
Plant 3	$14	$9	$16	$5

Answer

Representation of the problem as a LP model

x_{ij}: number of (million) kwh produced at plant *i* and sent to city *j*.

min z = 8 x11 + 6 x12 + 10 x13 + 9 x14 + 9 x21 + 12 x22 + 13 x23 + 7 x24 + 14 x31 + 9 x32 + 16 x33 + 5 x34

s.t. x11 + x12 + x13 + x14 ≤ 35 (supply constraints)

x21 + x22 + x23 + x24 ≤ 50

x31 + x32 + x33 + x34 ≤ 40

x11 + x21 + x31 ≥ 45 (demand constraints)

x12 + x22 + x32 ≥ 20

x13 + x23 + x33 ≥ 30

x14 + x24 + x34 ≥ 30

$x_{ij} \geq 0$ (i = 1, 2, 3; j = 1, 2, 3, 4)

www.Jntufastupdates.com

Formulation of the transportation problem

	City 1	City 2	City 3	City 4	SUPPLY
Plant 1	8	6	10	9	35
Plant 2	9	12	13	7	50
Plant 3	14	9	16	5	40
DEMAND	45	20	30	30	125

Total supply & total demand both equal 125: "balanced transport'n problem".

Balancing an Unbalanced Transportation Problem

Excess Supply

If total supply exceeds total demand, we can balance a transportation problem by creating a ***dummy demand point*** that has a demand equal to the amount of excess supply. Since shipments to the dummy demand point are not real shipments, they are assigned a cost of zero. These shipments indicate unused supply capacity.

Unmet Demand

If total supply is less than total demand, actually the problem has no feasible solution. To solve the problem it is sometimes desirable to allow the possibility of leaving some demand unmet. In such a situation, *a penalty is often associated with unmet demand.* This means that a ***dummy supply point*** should be introduced.

Example 2. Modified Powerco for Excess Supply

Suppose that demand for city 1 is 40 million kwh. Formulate a balanced transportation problem.

Answer

Total demand is 120, total supply is 125.

To balance the problem, we would add a dummy demand point with a demand of 125 – 120 = 5 million kwh.

From each plant, the cost of shipping 1 million kwh to the dummy is 0. For details see Table 4.

Table 4. Transportation Tableau for Excess Supply

	City 1	City 2	City 3	City 4	Dummy	SUPPLY
Plant 1	8	6	10	9	0	35
Plant 2	9	12	13	7	0	50
Plant 3	14	9	16	5	0	40
DEMAND	40	20	30	30	5	125

Example 3. Modified Powerco for Unmet Demand

Suppose that demand for city 1 is 50 million kwh. For each million kwh of unmet demand, there is a penalty of 80$. Formulate a balanced transportation problem. **Answer**

We would add a dummy supply point having a supply of 5 million kwh representing shortage.

	City 1	City 2	City 3	City 4	SUPPLY
Plant 1	8	6	10	9	35
Plant 2	9	12	13	7	50
Plant 3	14	9	16	5	40
Dummy (Shortage)	80	80	80	80	5
DEMAND	50	20	30	30	130

FINDING BFS FOR TRANSPORT'N PROBLEMS

For a balanced transportation problem, general LP representation may be written as: min $\sum_i$

$\sum_j c_{ij} x_{ij}$

s.t. $\sum_j x_{ij} = s_i$ (i=1,2, ..., m) Supply constraints

$\sum_i x_{ij} = d_j$ (j=1,2, ..., n) Demand constraints

$x_{ij} \geq 0$

To find a bfs to a balanced transportation problem, we need to make the following important observation:

If a set of values for the xij's satisfies all but one of the constraints of a balanced transportation problem, the values for the xij's will automatically satisfy the other constraint.

This observation shows that when we solve a balanced transportation, we may omit from consideration any one of the problem's constraints and solve an LP having $m+n-1$ constraints. We arbitrarily assume that the first supply constraint is omitted from consideration. In trying to find a bfs to the remaining $m+n-1$ constraints, you might think that any collection of $m+n-1$ variables would yield a basic solution. But this is not the case: If the $m+n-1$ variables yield a basic solution, the cells corresponding to this set contain ***no loop***.

An ordered sequence of at least four different cells is called a loop if

- Any two consecutives cells lie in either the same row or same column
- No three consecutive cells lie in the same row or column
- The last cell in the sequence has a row or column in common with the first cell in the sequence

There are three methods that can be used to find a bfs for a balanced transportation problem:

1. Northwest Corner method
2. Minimum cost method
3. Vogel's method

Northwest Corner Method

We begin in the upper left corner of the transportation tableau and set x_{11} as large as possible (clearly, x_{11} can be no larger than the smaller of s_1 and d_1).

- If $x_{11}=s_1$, cross out the first row of the tableau. Also change d_1 to d_1-s_1.
- If $x_{11}=d_1$, cross out the first column of the tableau. Change s_1 to s_1-d_1.
- If $x_{11}=s_1=d_1$, cross out either row 1 or column 1 (but not both!).
    - If you cross out row, change d_1 to 0.
    - If you cross out column, change s_1 to 0.

Continue applying this procedure to the most northwest cell in the tableau that does not lie in a crossed out row or column.

Eventually, you will come to a point where there is only one cell that can be assigned a value. Assign this cell a value equal to its row or column demand, and cross out both the cell's row or column.

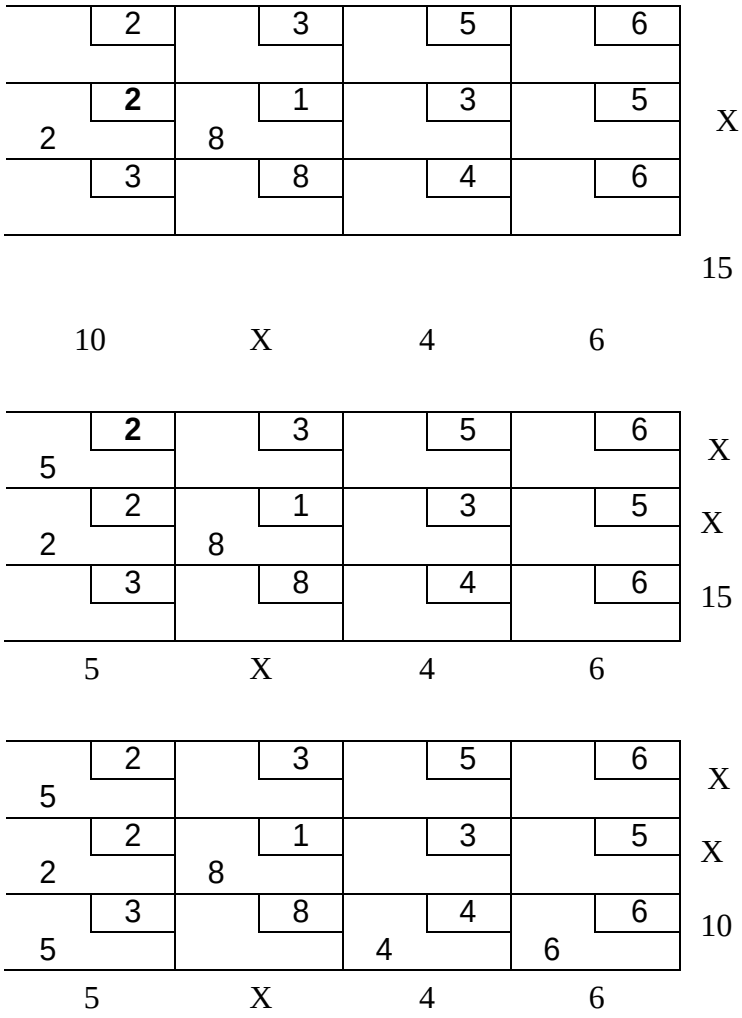
Minimum Cost Method

Northwest Corner method does not utilize shipping costs, so it can yield an initial bfs that has a very high shipping cost. Then determining an optimal solution may require several pivots.

To begin the minimum cost method, find the variable with the smallest shipping cost (call it x_{ij}). Then assign x_{ij} its largest possible value, min $\{s_i, d_j\}$.

As in the NWC method, cross out row i or column j and reduce the supply or demand of the noncrossed-out of row or column by the value of x_{ij}.

Continue like NWC method (instead of assigning upper left corner, the cell with the minimum cost is assigned). See Northwest Corner Method for the details!



Vogel's Method

Begin by computing for each row and column a penalty equal to the difference between the two smallest costs in the row and column. Next find the row or column with the largest penalty. Choose as the first basic variable the variable in this row or column that has the smallest cost. As described in the NWC method, make this variable as large as possible, cross out row or column, and change the supply or demand associated with the basic variable (See Northwest Corner Method for the details!). Now recomputed new penalties (using only cells that do not lie in a crossed out row or column), and repeat the procedure until only one uncrossed cell remains. Set this variable equal to the supply or demand associated with the variable, and cross out the variable's row and column.

www.Jntufastupdates.com

Example 3.

Supply					Row penalty
	6	7	8	10	7-6=1
	15	80	78	15	78-15=63
Demand	15	5	5		
Column penalty	15-6=9	80-7=**73**	78-8=70		

Supply					Row penalty
	6	**7**	8	5	8-6=2
		5			
	15	80	78	15	78-15=63
Demand	15	X	5		
Column penalty	15-6=9	----------	78-8=**70**		

Supply					Row penalty
	6	7	**8**	X	-
		5	5		
	15	80	78	15	-
Demand	15	X	0		
Column penalty	15-6=9	-	-		

	6	7	8	X
		5	5	
	15	80	78	
15			0	
				15
Demand	15	X	0	

THE TRANSPORTATION SIMPLEX METHOD

Steps of the Method

1. If the problem is unbalanced, balance it
2. Use one of the methods to find a bfs for the problem
3. Use the fact that $u_1 = 0$ and $u_i + v_j = c_{ij}$ for all basic variables to find the u's and v's for the current bfs.
4. If $u_i + v_j - c_{ij} \leq 0$ for all nonbasic variables, then the current bfs is optimal. If this is not the case, we enter the variable with the most positive $u_i + v_j - c_{ij}$ into the basis using the *pivoting procedure*. This yields a new bfs. Return to Step 3.

For a maximization problem, proceed as stated, but replace Step 4 by the following step:

If $u_i + v_j - c_{ij} \geq 0$ for all nonbasic variables, then the current bfs is optimal. Otherwise, enter the variable with the most negative $u_i + v_j - c_{ij}$ into the basis using the *pivoting procedure*. This yields a new bfs. Return to Step 3.

Pivoting procedure

1. Find the loop (there is only one possible loop!) involving the entering variable (determined at step 4 of the transport'n simplex method) and some or all of the basic variables.
2. Counting *only cells in the loop*, label those that are an even number (0, 2, 4, and so on) of cells away from the entering variable as *even cells*. Also label those that are an odd number of cells away from the entering variable as *odd cells*.
3. Find the odd cell whose variable assumes the smallest value. Call this value □ . The variable corresponding to this odd cell will leave the basis. To perform the pivot, decrease the value of each odd cell by □ and increase the value of each even cell by □ . The values of variables not in the loop remain unchanged. The pivot is now complete. If □ = 0, the entering variable will equal 0, and odd variable that has a current value of 0 will leave the basis.

www.Jntufastupdates.com

Example 1. Powerco

The problem is balanced (total supply equals total demand).

When the NWC method is applied to the Powerco example, the bfs in the following table is obtained (check: there exist $m+n-1=6$ basic variables).

	City 1	City 2	City 3	City 4	SUPPLY
Plant 1	8 / 35	6	10	9	35
Plant 2	9 / 10	12 / 20	13 / 20	7	50
Plant 3	14 / 10	9 / 30	16	5	40
DEMAND	45	20	30	30	125

u1 = 0

u1 + v1 = 8 yields v1 = 8

u2 + v1 = 9 yields u2 = 1

u2 + v2 = 12 yields v2 = 11

u2 + v3 = 13 yields v3 = 12

u3 + v3 = 16 yields u3 = 4 u3

+ v4 = 5 yields v4 = 1

For each nonbasic variable, we now compute $\hat{c}_{ij} = u_i + v_j - c_{ij}$ $\hat{c}_{12}$ =

0 + 11 – 6 = 5

$\hat{c}_{13}$ = 0 + 12 – 10 = 2

$\hat{c}_{14}$ = 0 + 1 – 9 = -8

$\hat{c}_{24}$ = 1 + 1 – 7 = -5

$\hat{c}_{31}$ = 4 + 8 – 14 = -2

$\hat{c}_{32}$ = 4 + 11 – 9 = 6

Since $\hat{c}_{32}$ is the most positive one, we would next enter x_{32} into the basis: Each unit of x_{32} that is entered into the basis will decrease Powerco's cost by \$6. The loop involving x_{32} is (3,2)-(3,3)-(2,3)-(2,2). □ = 10 (see table)

	City 1	City 2	City 3	City 4	SUPPLY
Plant 1	8 / 35	6	10	9 / 35	
Plant 2	9 / 10	12 / **20–Φ**	13 / **20+Φ**	7	50
Plant 3	14	9 / **Φ**	16 / **10–Φ**	5 / 30	40
DEMAND	45	20	30	30	125

x_{33} would leave the basis. New bfs is shown at the following table:

u_i/v_j	8	11	12	7	SUPPLY
0	8: 35	6	10	9	35
1	9: 10	12: 10	13: 30	7	50
-2	1	1: 5	9	6: 30	40
DEMAND	45	10	20	30	125

$\hat{c}_{12} = 5$, $\hat{c}_{13} = 2$, $\hat{c}_{14} = -2$, $\hat{c}_{24} = 1$, $\hat{c}_{31} = -8$, $\hat{c}_{33} = -6$

Since $\hat{c}_{12}$ is the most positive one, we would next enter x_{12} into the basis. The loop involving x_{12} is (1,2)-(2,2)-(2,1)-(1,1). □ = 10 (see table)

	City 1	City 2	City 3	City 4	SUPPLY
Plant 1	8: **35–Φ**	6: **Φ**	10	9	35
Plant 2	9: **10+Φ**	12: **10–Φ**	13: 30	7	50
Plant 3	14	9: 10	16	5: 30	40
DEMAND	45	20	30	30	125

x_{22} would leave the basis. New bfs is shown at the following table:

u_i/v_j	8	6	12	2	SUPPLY
0	8: 25	6: 10	10	9	35
1	9: 20	12	13: 7		50
3	30		9: 14	16: 5	40
DEMAND	45	20	30	30	125
		10		30	

$\hat{c}_{13} = 2$, $\hat{c}_{14} = -7$, $\hat{c}_{22} = -5$, $\hat{c}_{24} = -4$, $\hat{c}_{31} = -3$, $\hat{c}_{33} = -1$

Since $\hat{c}_{13}$ is the most positive one, we would next enter x_{13} into the basis. The loop involving x_{13} is (1,3)-(2,3)-(2,1)-(1,1). □ = 25 (see table)

	City 1	City 2	City 3	City 4	SUPPLY
Plant 1	8	6	10	9	35
	25–Φ	10	Φ	13	
Plant 2				7	50
	20+Φ	14	**30–Φ**	16	
Plant 3		9		5	40
		10		30	
DEMAND	45	20	30	30	125

x_{11} would leave the basis. New bfs is shown at the following table:

u_i/v_j	6	6	10	2	SUPPLY
0	8	6	10	9	35
1		10		25	
	9		2	13	50
3			7		
	45		5		40
3			9	16	
14				5	
DEMAND	45	20	30	30	125
		10		30	

$\hat{c}_{11} = -2$, $\hat{c}_{14} = -7$, $\hat{c}_{22} = -3$, $\hat{c}_{24} = -2$, $\hat{c}_{31} = -5$, $\hat{c}_{33} = -3$

Since all $\hat{c}_{ij}$'s are negative, an optimal solution has been obtained.

Report

45 million kwh of electricity would be sent from plant 2 to city 1.

10 million kwh of electricity would be sent from plant 1 to city 2. Similarly, 10 million kwh of electricity would be sent from plant 3 to city 2.

25 million kwh of electricity would be sent from plant 1 to city 3. 5 million kwh of electricity would be sent from plant 2 to city 3.

30 million kwh of electricity would be sent from plant 3 to city 4 and

Total shipping cost is:

z = .9 (45) + 6 (10) + 9 (10) + 10 (25) + 13 (5) + 5 (30) = $ 1020

TRANSSHIPMENT PROBLEMS

Sometimes a point in the shipment process can both receive goods from other points and send goods to other points. This point is called as **transshipment point** through which goods can be transshipped on their journey from a supply point to demand point.

Shipping problem with this characteristic is a transshipment problem.

The optimal solution to a transshipment problem can be found by converting this transshipment problem to a transportation problem and then solving this transportation problem.

Remark

As stated in "Formulating Transportation Problems", we define a **supply point** to be a point that can send goods to another point but cannot receive goods from any other point.

Similarly, a **demand point** is a point that can receive goods from other points but cannot send goods to any other point.

Steps

1. If the problem is unbalanced, balance it

   Let *s* = total available supply (or demand) for balanced problem
2. Construct a transportation tableau as follows

   A row in the tableau will be needed for each supply point and transshipment point A column will be needed for each demand point and transshipment point

   Each supply point will have a supply equal to its original supply Each demand point will have a demand equal to its original demand

   Each transshipment point will have a supply equal to "that point's original supply + *s*"

   Each transshipment point will have a demand equal to "that point's original demand + *s*"
3. Solve the transportation problem

Example 1. Bosphorus

(Based on Winston 7.6.)

Bosphorus manufactures LCD TVs at two factories, one in Istanbul and one in Bruges. The Istanbul factory can produce up to 150 TVs per day, and the Bruges factory can produce up to 200 TVs per day. TVs are shipped by air to customers in London and Paris. The customers in each city require 130 TVs per day. Because of the deregulation of air fares, Bosphorus believes that it may be cheaper to first fly some TVs to Amsterdam or Munchen and then fly them to their final destinations.

www.Jntufastupdates.com

The costs of flying a TV are shown at the table below. Bosphorus wants to minimize the total cost of shipping the required TVs to its customers.

€	To					
From	Istanbul	Bruges	Amsterdam	Munchen	London	Paris
Istanbul	0	-	8	13	25	28
Bruges	-	0	15	12	26	25
Amsterdam	-	-	0	6	16	17
Munchen	-	-	6	0	14	16
London	-	-	-	-	0	-
Paris	-	-	-	-	-	0

Answer:

In this problem Amsterdam and Munchen are *transshipment points*.

Step 1. Balancing the problem Total

supply = 150 + 200 = 350

Total demand = 130 + 130 = 260

Dummy's demand = 350 – 260 = 90

s = 350 (total available supply or demand for balanced problem)

Step 2. Constructing a transportation tableau

Transshipment point's demand = Its original demand + *s* = 0 + 350 = 350 Transshipment point's supply = Its original supply + *s* = 0 + 350 = 350

	Amsterdam	Munchen	London	Paris	Dummy	Supply
Istanbul	8	13	25	28	0	150
Bruges	15	12	26	25	0	200
Amsterdam	0	6	16	17	0	350
Munchen	6	0	14	16	0	350
Demand	350	350	130	130	90	

Step 3. Solving the transportation problem

	Amsterdam	Munchen	London	Paris	Dummy	Supply
Istanbul	8	13	25	28	0	
	130				20	150
Bruges	15		12	26	25	
	0					200
Amsterdam	130					
						350
Munchen	70					
	0		6	16	17	
				130	90	1050

Report:

Bosphorus should produce 130 TVs at Istanbul, ship them to Amsterdam, and transship them from Amsterdam to London.

The 130 TVs produced at Bruges should be shipped directly to Paris. The total shipment is 6370 Euros.

ASSIGNMENT PROBLEMS

There is a special case of transportation problems where each supply point should be assigned to a demand point and each demand should be met. This certain class of problems is called as "assignment problems". For example determining which employee or machine should be assigned to which job is an assignment problem.

LP Representation

An assignment problem is characterized by knowledge of the cost of assigning each supply point to each demand point: c_{ij}

On the other hand, a 0-1 integer variable x_{ij} is defined as follows

x_{ij} = 1 if supply point i is assigned to meet the demands of demand point j x_{ij} = 0 if supply point i is not assigned to meet the demands of point j

In this case, the general LP representation of an assignment problem is min $\sum_i \sum_j c_{ij} x_{ij}$

s.t. $\sum_j x_{ij} = 1$ (i=1,2, ..., m) Supply constraints

$\sum_i x_{ij} = 1$ (j=1,2, ..., n) Demand constraints

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

Hungarian Method

Since all the supplies and demands for any assignment problem are integers, all variables in optimal solution of the problem must be integers. Since the RHS of each constraint is equal to 1, each x_{ij} must be a nonnegative integer that is no larger than 1, so each x_{ij} must equal 0 or 1.

Ignoring the $x_{ij} = 0$ or $x_{ij} = 1$ restrictions at the LP representation of the assignment problem, we see that we confront with a balanced transportation problem in which each supply point has a supply of 1 and each demand point has a demand of 1.

However, the high degree of degeneracy in an assignment problem may cause the Transportation Simplex to be an inefficient way of solving assignment problems.

For this reason and the fact that the algorithm is even simpler than the Transportation Simplex, the Hungarian method is usually used to solve assignment problems.

Remarks

1. To solve an assignment problem in which the goal is to maximize the objective function, multiply the profits matrix through by –1 and solve the problem as a **minimization** problem.
2. If the number of rows and columns in the cost matrix are unequal, the assignment problem is **unbalanced**. Any assignment problem should be balanced by the addition of one or more dummy points before it is solved by the Hungarian method.

Steps

1. Find the minimum cost each row of the *m***m* cost matrix.
2. Construct a new matrix by subtracting from each cost the minimum cost in its row
3. For this new matrix, find the minimum cost in each column
4. Construct a new matrix (reduced cost matrix) by subtracting from each cost the minimum cost in its column
5. Draw the minimum number of lines (horizontal and/or vertical) that are needed to cover all the zeros in the reduced cost matrix. If *m* lines are required, an optimal solution is available among the covered zeros in the matrix. If fewer than *m* lines are needed, proceed to next step
6. Find the smallest cost (*k*) in the reduced cost matrix that is uncovered by the lines drawn in Step 5
7. Subtract *k* from each uncovered element of the reduced cost matrix and add *k* to each element that is covered by two lines. Return to Step 5

Example 1. Flight Crew

(Based on Winston 7.5.)

Four captain pilots (CP1, CP2, CP3, CP4) has evaluated four flight officers (FO1, FO2, FO3, FO4) according to perfection, adaptation, morale motivation in a 1-20 scale (1: very good, 20: very bad). Evaluation grades are given in the table. Flight

Company wants to assign each flight officer to a captain pilot according to these

evaluations. Determine possible flight crews.

	FO1	FO2	FO3	FO4
CP1	2	4	6	10
CP2	2	12	6	5
CP3	7	8	3	9
CP4	14	5	8	7

Answer:

Step 1. For each row in the table we find the minimum cost: 2, 2, 3, and 5 respectively

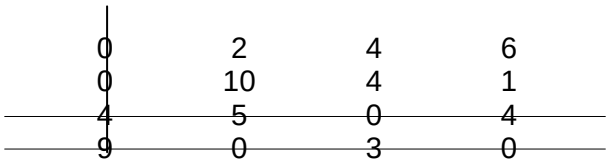
Step 2 & 3. We subtract the row minimum from each cost in the row. For this new matrix, we find the minimum cost in each column

	0	2	4	8
	0	10	4	3
	4	5	0	6
	9	0	3	2
Column minimum	0	0	0	2

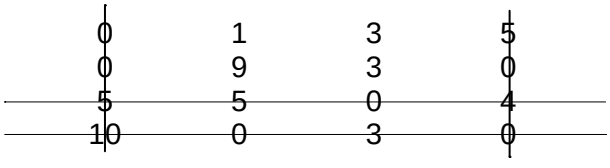
Step 4. We now subtract the column Minimum from each cost in the column obtaining reduced cost matrix.

0	2	4	6
0	10	4	1
4	5	0	4
9	0	3	0

Step 5. As shown, lines through row 3, row 4, and column 1 cover all the zeros in the reduced cost matrix. The minimum number of lines for this operation is 3. Since fewer than four lines are required to cover all the zeros, solution is not optimal: we proceed to next step.



Step 6 & 7. The smallest uncovered cost equals 1. We now subtract 1 from each uncovered cost, add 1 to each twice-covered cost, and obtain



Four lines are now required to cover all the zeros: An optimal s9olution is available. Observe that the only covered 0 in column 3 is x_{33}, and in column 2 is x_{42}. As row 5 can not be used again, for column 4 the remaining zero is x_{24}. Finally we choose x_{11}.

Report:

CP1 should fly with FO1; CP2 should fly with FO4; CP3 should fly with FO3; and CP4 should fly with FO4.

Example 2. Maximization problem

	F	G	H	I	J
A	6	3	5	8	10
B	2	7	6	3	2
C	5	8	3	4	6
D	6	9	3	1	7
E	2	2	2	2	8

Report:

Optimal profit = 36

Assigments: A-I, B-H, C-G, D-F, E-J

Alternative optimal sol'n: A-I, B-H, C-F, D-G, E-J

Traveling Salesperson Problems

"Given a number of cities and the costs of traveling from any city to any other city, what is the cheapest round-trip route (tour) that visits each city once and then returns to the starting city?"

This problem is called the traveling salesperson problem (TSP), not surprisingly.

An itinerary that begins and ends at the same city and visits each city once is called a ***tour***.

Suppose there are N cities.

Let c_{ij} = Distance from city i to city j (for $i \neq j$) and

Let c_{ii} = M (a very large number relative to actual distances) Also

define x_{ij} as a 0-1 variable as follows:

x_{ij} = 1 if s/he goes from city i to city j;

x_{ij} = 0 otherwise

The formulation of the TSP is:

$$\min \quad \sum_i \sum_j c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_i x_{ij} = 1 \qquad \text{for all } j$$

$$\sum_j x_{ij} = 1 \qquad \text{for all } i$$

$$u_i - u_j + N x_{ij} \leq N - 1 \qquad \text{for } i \neq j;\ i, j >$$

1 All $x_{ij} = 0$ or 1, All $u_i \geq 0$

The first set of constraints ensures that s/he arrives once at each city. The second set of constraints ensures that s/he leaves each city once. The third set of constraints ensure the following:

Any set of x_{ij}'s containing a subtour will be infeasible Any set of x_{ij}'s that forms a tour will be feasible

$u_i - u_j + N\, x_{ij} \leq N - 1$ for $i \neq j;\ i, j >$

1 Assume N=5

Subtours: 1-5-2-1, 3-4-3 ???

Choose the subtour that does not contain city 1:

$$u_3 - u_4 + 5\, x_{34} \leq 4$$

$$u_4 - u_3 + 5\, x_{43} \leq 4\ 5$$

$$(x_{34} + x_{43}) \leq 8$$

This rules out the possibility that $x_{34} = x_{43} = 1$

The formulation of an IP whose solution will solve a TSP becomes unwieldy and inefficient for large TSPs.

When using branch and bound methods to solve TSPs with many cities, large amounts of computer time may be required. For this reason, heuristics, which quickly lead to a good (but not necessarily optimal) solution to a TSP, are often used.

SOLVING IP

We have gone through a number of examples of IPs at the "Formulating IP Problems" section.

"How can we get solutions to these models?" There are two common approaches: The technique based on dividing the problem into a number of smaller problems in a *tree search* method called ***branch and bound***.

The method based on ***cutting planes*** (adding constraints to force integrality). Solving IP

Actually, all these approaches involve solving a series of LP.

For solving LP's we have *general purpose* (independent of the LP being solved) and *computationally effective* (able to solve large LP's) algorithms (simplex or interior point).

For solving IP's *no* similar general purpose and computationally effective algorithms exist

Categorization Categorization (w.r.t.

Purpose)

- General purpose methods will solve any IP but potentially computationally ineffective (will only solve relatively small problems); or
- Special purpose methods are designed for one particular type of IP problem but potentially computationally more effective.

Categorization (w.r.t. Algorithm)

- Optimal algorithms mathematically guarantee to find the optimal solution

- Heuristic algorithms are used to solve a problem by trial and error when an optimal algorithm approach is impractical. They hopefully find a good feasible solution that, in objective function terms, is close to the optimal solution.

Why Heuristics?

Because the size of problem that we want to solve is beyond the computational limit of known optimal algorithms within the computer time we have available.

We could solve optimally but feel that this is not worth the effort (time, money, etc) we would expend in finding the optimal solution.

In fact it is often the case that a well-designed heuristic algorithm can give good quality (near-optimal) results.

Solution Algorithms Categories

- General Purpose, Optimal

  Enumeration, branch and bound, cutting plane
- General Purpose, Heuristic

  Running a general purpose optimal algorithm and terminating after a specified time
- Special Purpose, Optimal

  Tree search approaches based upon generating bounds via dual ascent, lagrangean relaxation
- Special Purpose, Heuristic

  Bound based heuristics, tabu search, simulated annealing, population heuristics (e.g. genetic algorithms), interchange

LP Relaxation

For any IP we can generate an LP by taking the same objective function and same constraints but with the requirement that variables are integer replaced by appropriate continuous constraints:

"$x_i = 0$ or 1" $\square$ $x_i \geq 0$ and $x_i \leq 1$

"$x_i \geq 0$ and integer" $\square$ $x_i \geq 0$

The LP obtained by omitting all integer and 0-1 constraints on variables is called the ***LP Relaxation of the IP*** (***LR***). We can then solve this LR of the original IP.

Naturally Integer LP

If LR is optimized by integer variables then that solution is feasible and optimal for IP. In other words, if the solution is turned out to have all variables taking integer values at the optimal solution, it is also optimal solution for IP:

LR – IP Relation

Since LR is less constrained than IP:

- If IP is a maximization problem, the optimal objective value for LR is greater than or equal to that of IP.
- If IP is a minimization problem, the optimal objective value for LR is less than or equal to that of IP.
- If LR is infeasible, then so is IP.

So solving LR does give some information. It gives a bound on the optimal value, and, if we are lucky, may give the optimal solution to IP.

Enumeration

Unlike LP (where variables took continuous values) in IP's (where all variables are integers) each variable can only take a finite number of discrete (integer) values. Hence the obvious solution approach is simply to ***enumerate*** all these possibilities - calculating the value of the objective function at each one and choosing the (feasible) one with the optimal value.

Example 1. Multi-period Capital Budgeting

Maximize $0.2\,x_1 + 0.3\,x_2 + 0.5\,x_3 + 0.1\,x_4$

Subject to $0.5\,x_1 + 1\quad x_2 + 1.5\,x_3 + 0.1\,x_4 \leq 3.1$

$0.3\,x_1 + 0.8\,x_2 + 1.5\,x_3 + 0.4\,x_4 \leq 2.5$

$0.2\,x_1 + 0.2\,x_2 + 0.3\,x_3 + 0.1\,x_4 \leq 0.4$

$x_j = 0 \text{ or } 1 \qquad j = 1, \ldots 4$

Review

Hence for our example, we merely have to examine 16 possibilities before we know precisely what the best possible solution is. This example illustrates a general truth about integer programming.

What makes solving the problem easy when it is small is precisely what makes it hard very quickly as the problem size increases.

This is simply illustrated: suppose we have 100 integer variables each with two possible integer values then there are 2x2x2x ... x2 = 2^{100} (approximately 10^{30}) possibilities which we have to enumerate (obviously many of these possibilities will be infeasible, but until we generate one we cannot check it against the constraints to see if it is feasible).

For 100 integer variable - ***conceptually*** there is not a problem - simply enumerate all possibilities and choose the best one. But ***computationally*** (numerically) this is just impossible.

The Branch-and-Bound Method

The most effective general purpose optimal algorithm is an LP-based tree search approach called as ***branch and bound*** (B&B).

The method was first put forward in the early 1960's by Land and Doig.

This is *a way of systematically (implicitly) enumerating* feasible solutions such that the optimal integer solution is found.

Where this method differs from the enumeration method is that *not all* the feasible solutions are enumerated but only a part (hopefully a small part) of them. However we can still *guarantee* that we will find the optimal integer solution.

By solving a single sub-problem, many possible solutions may be eliminated from consideration.

Sub-problems are generated by branching on an appropriately chosen fractional- valued variable.

Suppose that in a given sub-problem (call it subp.1), assumes a fractional value between the integers i and $i+1$. Then the two newly generated sub-problems:

Subp.2 = Subp.1 + Constraint "$x_i \geq i+1$" Subp.3

= Subp.1 + Constraint "$x_i \leq I$"

If all variables have integer values in the optimal solution to the sub-problem then the solution is a ***feasible*** solution for the original IP.

If the current feasible solution for the IP has a better z-value than any previously obtained feasible solution, then it becomes a ***candidate solution***, and its z-value becomes the current ***Lower Bound*** (***LB***) on the optimal z-value (for a max problem). If it is unnecessary to branch on a sub-problem, we say that it is fathomed (inactive):

- The sub-problem is infeasible
- The sub-problem yields an optimal solution in which all variables have integer values
- The optimal z-value for the sub-problem does not exceed the current LB, so it cannot yield the optimal solution of the IP

Two general approaches are used to determine which sub-problem should be solved next:

- Backtracking (LIFO)

  Leads us down one side of the B&B tree and finds a candidate solution. Then we backtrack our way up to the top of the other side of the tree.
- Jumptracking

  Solves all the problems created by branching. Then it branches again on the node with the best z-value. Often jumps from one side of the tree to the other.

A display of the sub-problems that have been created is called a ***tree***. Each sub-problem is referred to as a ***node*** of the tree.

Each additional constraint is referred to as a line (***arc***) connecting two nodes (old sub-problem and one of the new sub-problems) of the tree.

B&B for Solving Pure IP Problems Example 2.

Pure IP

(*Winston 9.3., p. 513*)

max $z = 8\,x_1 + 5\,x_2$

s.t. $x_1 + x_2 \leq 6$

$9\,x_1 + 5\,x_2 \leq 45$

$x_1, x_2 \geq 0$ and integer

Answer

Suppose that we were to solve the LR of the problem [replace "$x_1, x_2 \geq 0$ and integer" by "$x_1, x_2 \geq 0$"]

Then using any LP package or utilizing simplex or graphical solution method we get

$z = 165/4$, $x_1 = 15/4$, $x_2 = 9/4$

As a result of this we now know something about the optimal integer solution, namely that it is $\leq 165/4$, i.e. this value of 165/4 is an ***Upper Bound*** on the optimal integer solution

This is because when we relax the integrality constraint we (as we are maximizing) end up with a solution value at least that of the optimal integer solution (and maybe better)

We arbitrarily choose a variable that is fractional in the optimal solution to the LR (subp.1): say x_1.

We need x_1 to be integer. We branch on x_1 and create two new sub-problems: Subp.2:

LR + "$x_1 \geq 4$"

Subp.3: LR + "$x_1 \leq 3$"

Observe that neither subp.2 nor subp.3 includes any points with $x_1 = 15/4$. This means that the optimal solution to LR can not recur when we solve these new sub- problems.

We now arbitrarily choose to solve subp.2.

We see that the optimal solution to subp.2 is

$z = 41$, $x_1 = 4$, $x_2 = 9/5$

We choose x_2 that is fractional in the optimal solution to subp.2.

We need x_2 to be integer. We branch on x_2 and create two new sub-problems:

Subp.4: LR + $x_1 \geq 4$ and $x_2 \geq 2$ = Subp.2 + $x_2 \geq 2$

Subp.5: LR + $x_1 \geq 4$ and $x_2 \leq 1$ = Subp.2 + $x_2 \leq 1$

The set of unsolved sub-problems are consists of subp.3, 4, and 5.

We choose to solve the most recently created sub-problem (This is called LIFO): The LIFO rule implies that we should next solve subp.4 or 5.

We now arbitrarily choose to solve subp.4.

We see that subp.4 is infeasible. Thus subp.4 can not yield the optimal solution to the IP.

Because any branches emanating from subp.4 will yield no useful information, it is fruitless to create them.

LIFO rule implies that we should next solve subp.5.

The optimal solution to subp.5 is

$z = 365/9$, $x_1 = 40/9$, $x_2 = 1$

We branch on fractional-valued x_1:

Subp.6: Subp.5 + $x_1 \geq 5$

Subp.7: Subp.5 + $x_1 \leq 4$

Subp.3, 6, and 7 are now unsolved.

The LIFO rule implies that we next solve subp.6 or 7. We now arbitrarily choose to solve subp.7.

The optimal solution to subp.7 is

$z = 37$, $x_1 = 4$, $x_2 = 1$

As both variables assume integer values, this solution is ***feasible*** for the original IP

◇ this solution is a ***candidate solution***

We must keep this candidate solution until a better feasible solution to the IP (if any exists) is found.

We may conclude that the optimal z-value for the IP ≥ 37 □ ***Lower Bound*** (LB) LIFO rule implies that we should next solve subp.6.

The optimal solution to subp.6 is

$z = 40$, $x_1 = 5$, $x_2 = 0$

Its z-value of 40 is larger than LB.

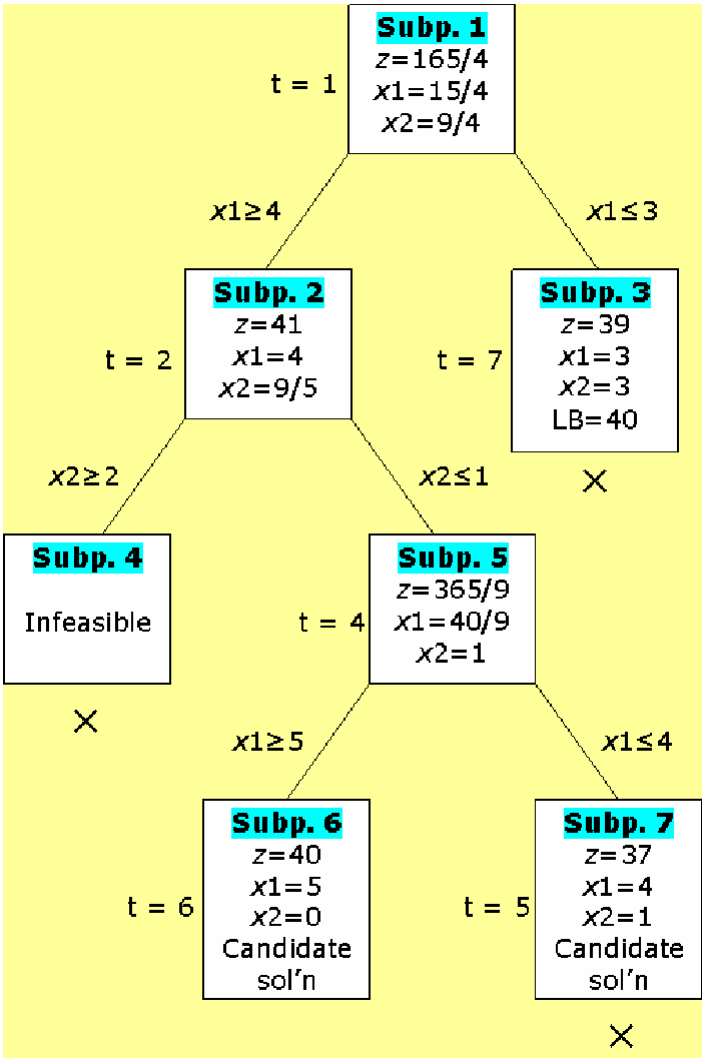
Thus subp.7 cannot yield the optimal solution of the IP. We update our LB to 40.

Subp.3 is the only remaining unsolved problem. The optimal solution to subp.3 is

$z = 39, x_1 = 3, x_2 = 3$

Subp.3 cannot yield a z-value exceeding the current LB, so it cannot yield the optimal solution to the IP.

Final B&B Tree



Optimal Sol'n

Thus, the optimal solution to the IP

$z = 40, x_1 = 5, x_2 = 0$

B&B for Solving Mixed IP Problems

In MIP, some variables are required to be integers and others are allowed to be either integer or nonintegers.

To solve a MIP by B&B method, modify the method by branching only on variables that are required to be integers.

For a solution to a sub-problem to be a candidate solution, it need only assign integer values to those variables that are required to be integers

Example 3. Mixed IP

(*Winston 9.4., p. 523*)

max $z = 2x_1 + x_2$

s.t. $5x_1 + 2x_2 \leq 8$

$x_1 + x_2 \leq 3$

$x_1, x_2 \geq 0$; $x1$ integer

Answer

We solve the LR (subp.1) of the problem [replace

"$x_1 \geq 0$ and integer" by "$x_1 \geq 0$"]

Then using any LP package or utilizing simplex or graphical solution method we get

$z = 11/3$, $x_1 = 2/3$, $x_2 = 7/3$

Because x_2 is allowed to be fractional, we do not branch on x_2. We

branch on x_1 and create two new sub-problems:

Subp.2: LR + $x_1 \geq 1$

Subp.3: LR + $x_1 \leq 0$

We see that the optimal solution to subp.2 is

$z = 7/2$, $x_1 = 1$, $x_2 = 3/2$

As only x_1 assume integer value, this solution is feasible for the original MIP □

Candidate solution; LB = 7/2 The

optimal solution to subp.3 is

$z = 3$, $x_1 = 0$, $x_2 = 3$

Subp.3 cannot yield a z-value exceeding the current LB, so it cannot yield the optimal solution to the MIP.

Optimal Sol'n

Thus, the optimal solution to the MIP

$z = 7/2$, $x_1 = 1$, $x_2 = 3/2$

B&B for Solving Binary IP Problems

One aspect of the B&B method greatly simplify:

Due to each variable equaling 0 or 1, branching on x_i will yield in

$x_i = 0$ and $x_i = 1$

Example 4. Binary IP

max z = $0.2\,x_1 + 0.3\,x_2 + 0.5\,x_3 + 0.1\,x_4$

s.t. $0.5\,x_1 + 1\quad x_2 + 1.5\,x_3 + 0.1\,x_4 \leq 3.1$

$0.3\,x_1 + 0.8\,x_2 + 1.5\,x_3 + 0.4\,x_4 \leq 2.5$

$0.2\,x_1 + 0.2\,x_2 + 0.3\,x_3 + 0.1\,x_4 \leq 0.4$

$x_j = 0 \text{ or } 1 \qquad j = 1, \ldots 4$

Answer

Replace "$x_j = 0$ or 1 (j=1,...,4)" by "$0 \leq x_j \leq 1$ (j=1,...,4)" □ LR of the problem

Optimal solution to the LR:

z=0.65, x_2=0.5, x_3=1, x_1=x_4=0

The variable x_2 is fractional. To resolve this we can generate two new problems: P1: LR + x_2=0

P2: LR + x_2=1

We now have two new sub-problem to solve (*jumptracking*). If we do this we get

P1 solution: z=0.6, x_1=0.5, x_3=1, x_2=x_4=0 P2 solution: z=0.63, x_2=1, x_3=0.67, x_1=x_4=0

Choosing sub-problem P2 (the best z–value), we branch on x_3 and get P3 (P2 + x_3=0) sol'n: z=0.5, x_1=x_2=1, x_3=x_4=0

P4 (P2 + x_3=1) sol'n: infeasible

P3 solution is feasible for the original binary IP □ Candidate solution; LB = 0.5

Choosing the only remaining sub-problem P1, we branch on x_1 and get

P5 (P1 + x_1=0) sol'n: z=0.6, x_3=x_4=1, x_1=x_2=0

P6 (P1 + x_1=1) sol'n: z=0.53, x_1=1, x_3=0.67, x_2=x_4=0

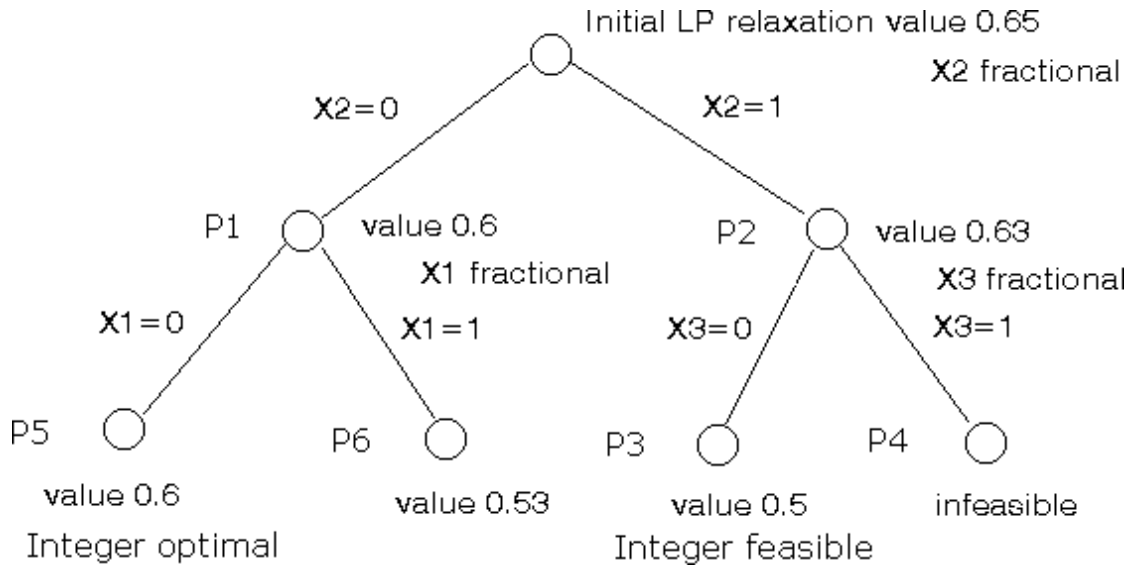
P5 solution is feasible for the original binary IP □ New candidate solution; updated LB = 0.6

P6 cannot yield a z-value exceeding the current LB, so it cannot yield the optimal solution to the binary IP.

Thus, the optimal solution to the binary IP

$z = 0.6, x_1 = 0, x_2 = 0, x_3 = 1, x_4 = 1$

www.Jntufastupdates.com



Review

Note here that B&B, like complete enumeration, also involves powers of 2 as we progress down the (binary) tree.

However also note that we did not enumerate all possible integer solutions (of which there are 16). Instead here we solved 7 LP's.

This is an important point, and indeed why tree search works at all. We do not need to examine as many LP's as there are possible solutions.

While the computational efficiency of tree search differs for different problems, it is this basic fact that enables us to solve problems that would be completely beyond us where we to try complete enumeration

B&B for Solving Knapsack Problems

Please recall that a knapsack problem is an IP, in which each variable must be equal to 0 or 1, with a single constraint:

max $z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$

s.t. $a_1x_1 + a_2x_2 + \cdots + a_nx_n \leq$

b $x_i = 0$ or 1 ($i = 1, 2, \ldots, n$)

Two aspects of the B&B method greatly simplify:

- Due to each variable equaling 0 or 1, branching on x_i will yield in $x_i = 0$ and $x_i = 1$
- The LP relaxation may be solved by ***inspection*** instead of using any LP package or utilizing simplex or graphical solution method

Inspection

Recall that

c_i is the benefit obtained if item i is chosen

b is the total amount of an available resource

a_i is the amount of the available resource used by item i

Observe that ratio r_i (c_i/a_i) may be interpreted as the benefit item i earns for each unit of the resource used by item i.

Thus, the best items have the largest value of r and the worst items have the smallest values of r.

To solve any sub-problem resulting from a knapsack problem, compute all the ratios. Then put the best item in the knapsack.

Then put the second best item in the knapsack.

Continue in this fashion until the best remaining item will overfill the knapsack. Then fill the knapsack with as much of this item as possible.

Example 5. Knapsack

max $z =$ $8\,x_1 + 11\,x_2 + 6\,x_3 + 4\,x_4$

s.t. $5\,x_1 + 7\,x_2 + 4\,x_3 + 3\,x_4 \leq 14$

$x_j = 0 \text{ or } 1 \qquad j = 1, \ldots 4$

Answer

We compute the ratios:

$r_1 = 8 / 5 = 1.6$

$r_2 = 11 / 7 = 1.57$

$r_3 = 6 / 4 = 1.5$

$r_4 = 4 / 3 = 1.33$

Using the ratios, LR solution is

$x_1 = 1, x_2 = 1, x_3 = 0.5, x_4 = 0, z = 22$

We branch on x_3 and get

P1 (LR + x_3=0) sol'n: x_3=0, x_1=x_2=1, x_4=2/3, z=21.67 P2

(LR + x_3=1) sol'n: x_3=x_1=1, x_2=5/7, x_4=0, z=21.85

Choosing sub-problem P2 (the best z–value), we branch on x_2 and get P3 (P2 + x_2=0) sol'n: x_3=1, x_2=0, x_1=1, x_4=1, z=18

P4 (P2 + x_2=1) sol'n: x_3=x_2=1, x_1=3/5, x_4=0, z=21.8

P3 solution is feasible for the original knapsack problem □ Candidate solution; LB = 18

Choosing sub-problem P4, we branch on x_1 and get P5 (P4

+ x_1=0) sol'n: x_3=x_2=1, x_1=0, x_4=1, z=21

P6 (P4 + x_1=1) sol'n: Infeasible (x_3=x_2=x_1=1: LHS=16)

P5 solution is feasible for the original knapsack problem □ New candidate solution; updated LB = 21

The only remaining sub-problem is P1 with solution value 21.67

There is no better solution for this sub-problem than 21. But we already have a solution with value 21.

It is not useful to search for another such solution. We can fathom P1 based on this bounding argument and mark P1 as inactive.

Optimal sol'n and Report

Thus, the optimal solution is

z=21, x_1=0, x_2=1, x_3=1, x_4=1

Items 2, 3, and 4 should be put in the knapsack. In this

case, the total value would be 21.

