

## UNIT IV ITERATIVE IMPROVEMENT

### 1. What do you mean by optimal solution?

Given a problem with  $n$  inputs, we obtain a subset that satisfies some constraints. Any subset that satisfies these constraints is called a feasible solution.

A feasible solution, which either maximizes or minimizes a given objective function is called optimal solution.

### 2. What is feasible solution?

It is obtained from given  $n$  inputs. Subsets that satisfy some constraints are called feasible solution. It is obtained based on some constraints.

### 3. Compare feasible and optimal solution.

A solution (set of values for the decision variables) for which all of the constraints in the Solver model are satisfied is called a feasible solution. In some problems, a feasible solution is already known; in others, finding a feasible solution may be the hardest part of the problem.

An optimal solution is a feasible solution where the objective function reaches its maximum (or minimum) value – for example, the most profit or the least cost. A globally optimal solution is one where there are no other feasible solutions with better objective function values. A locally optimal solution is one where there are no other feasible solutions “in the vicinity” with better objective function values.

### 4. What is LPP?

*Linear programming problem* (LPP) is to optimize a linear function of several variables subject to linear constraints:

Maximize (or minimize)  $c_1 x_1 + \dots + c_n x_n$

Subject to  $a_{i1}x_1 + \dots + a_{in}x_n \leq (\text{or } \geq \text{ or } =) b_i, i = 1, \dots, m$   
 $x_1 \geq 0, \dots, x_n \geq 0$

The function  $z = c_1 x_1 + \dots + c_n x_n$  is called the *objective function*;

Constraints  $x_1 \geq 0, \dots, x_n \geq 0$  are called *nonnegativity constraints*.

### 5. What is Simplex Method?

- The classic method for solving Linear programming problem (LPP).
- One of the most important algorithms ever invented to solve LPP.
- Invented by George Dantzig in 1947.
- Based on the iterative improvement idea.
- Generates a sequence of adjacent points of the problem's feasible region with improving values of the objective function until no further improvement is possible.

### 6. Write the steps to solve LPP problems by Simplex Method / Procedure for Simplex Method.

- **Step 0** [Initialization] Present a given LP problem in standard form and set up initial tableau.
- **Step 1** [Optimality test] If all entries in the objective row are nonnegative then stop: the tableau represents an optimal solution.
- **Step 2** [Find entering variable] Select the most negative entry in the objective row. Mark its column to indicate the entering variable and the **pivot column**.

- **Step 3** [Find departing (leaving) variable] For each positive entry in the pivot column, calculate the  $\theta$ -ratio by dividing that row's entry in the rightmost column (solution) by its entry in the pivot column. (If there are no positive entries in the pivot column then stop: the problem is unbounded.) Find the row with the smallest  $\theta$ -ratio, mark this row to indicate the departing variable and the **pivot row**.
- **Step 4** [Form the next tableau] Divide all the entries in the pivot row by its entry in the pivot column. Subtract from each of the other rows, including the objective row, the new pivot row multiplied by the entry in the pivot column of the row in question. Replace the label of the pivot row by the variable's name of the pivot column and go back to Step 1.

### 7. Write the time complexity of the Simplex Method.

- Finding an initial basic feasible solution may pose a problem.
- Theoretical possibility of cycling.
- Typical number of iterations is between  $m$  and  $3m$ , where  $m$  is the number of equality constraints in the standard form.
- Worse-case efficiency is **exponential**.
- More recent interior-point algorithms such as Karmarkar's algorithm (1984) have polynomial worst-case efficiency and have performed competitively with the simplex method in empirical tests.

### 8. Write the Standard form of LP problem.

- Must be a **maximization** problem
- All constraints (except the nonnegativity constraints) must be in the form of linear equations
- All the variables must be required to be nonnegative
- Thus, the general linear programming problem in standard form with  $m$  constraints and  $n$  unknowns ( $n \geq m$ ) is
- Maximize  $c_1 x_1 + \dots + c_n x_n$
- Subject to  $a_{i1}x_1 + \dots + a_{in}x_n = b_i, i = 1, \dots, m, x_1 \geq 0, \dots, x_n \geq 0$

### 9. Formulate the LPP for the given data.

A farmer has a 320 acre farm on which she plants two crops: corn and soybeans. For each acre of corn planted, her expenses are \$50 and for each acre of soybeans planted, her expenses are \$100. Each acre of corn requires 100 bushels of storage and yields a profit of \$60; each acre of soybeans requires 40 bushels of storage and yields a profit of \$90. If the total amount of storage space available is 19,200 bushels and the farmer has only \$20,000 on hand.

#### Linear Programming Problem Formulation

	Corn	Soybean	Total
Expenses	\$50	\$100	\$20,000
Storage(bushels)	100	40	19,200
Profit	60	90	Maximize profit

A farmer has a 320 acre farm is unwanted data but  $c+s \leq 320$ .

#### Canonical form of LPP

Maximize:  $60c + 90s$

Subject to  $50c + 100s = 20000$

$100c + 40s = 19200$

$c \geq 0, s \geq 0$

**10. What are the possible outcomes in solving an LP problem?**

- has a finite optimal solution, which may not be unique
- *unbounded*: the objective function of maximization (minimization) LP problem is unbounded from above (below) on its feasible region
- *infeasible*: there are no points satisfying all the constraints, i.e. the constraints are contradictory

**11. Solve the LPP by algebraic geometry technique.**

Maximize:  $60c + 90s$

Subject to  $50c + 100s = 20000$  (1)

$100c + 40s = 19200$  (2)

$$(1)/50 \Rightarrow c + 2s = 400$$

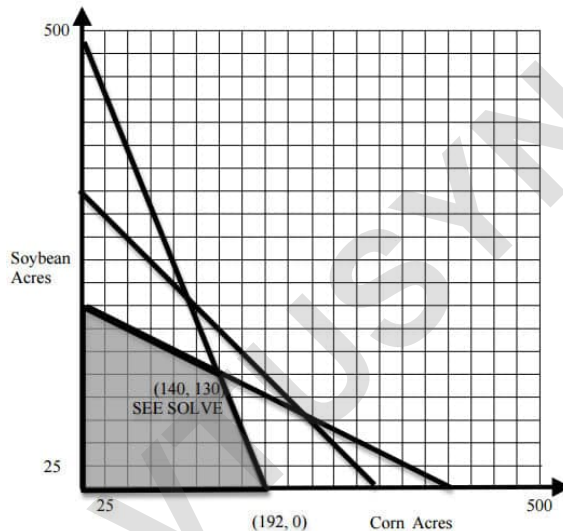
$$(2)/20 \Rightarrow 5c + 2s = 960$$

$$(2) - (1) \Rightarrow 4c = 560$$

$$c = 140$$

Substitute  $c = 140$  in (1) then  $s = 130$

Profit:  $p = 60c + 90s = 60(140) + 90(130) = 20,100$

**12. Solve the LPP by Graphical method.**

Profit at  $(0, 200) = 60c + 90s = 60(0) + 90(200) = \$18,000$

Profit at  $(192, 0) = 60c + 90s = 60(192) + 90(0) = \$11,520$

Profit at  $(140, 130) = 60c + 90s = 60(140) + 90(130) = \$20,100$

**13. How can you convert Primal to Dual of LPP?**

Primal  $\text{Maximize } z = \sum_{j=1}^n c_j x_j$ ,  
 subject to:  $\sum_{j=1}^n a_{ij} x_j \leq b_i \quad (i = 1, 2, \dots, m)$ ,  
 $x_j \geq 0 \quad (j = 1, 2, \dots, n)$ .

Dual  $\text{Minimize } z' = \sum_{i=1}^m b_i y_i$ ,  
 subject to:  $\sum_{i=1}^m a_{ij} y_i \geq c_j \quad (j = 1, 2, \dots, n)$ ,  
 $y_i \geq 0 \quad (i = 1, 2, \dots, m)$ .

**14. How will you calculate new pivot row and remaining rows in new iteration of simplex method?**

Pivot row:

New Pivot Row = Current Pivot Row / Pivot Element

All other rows including z:

New Row = Current Row – (Its Pivot column coefficient) \* New Pivot Row

**15. Convert the given primal problem into dual problem.**

The Primal problem

$$\begin{aligned} &\text{Minimize } 4x_1 + 2x_2 - x_3 \\ &\text{subject to } x_1 + x_2 + 2x_3 \geq 3 \\ &\quad \quad \quad 2x_1 - 2x_2 + 4x_3 \leq 5 \\ &\quad \quad \quad x_1, x_2, x_3 \geq 0. \end{aligned}$$

The dual problem

$$\begin{aligned} &\text{Maximize } 3y_1 + 5y_2 \\ &\text{subject to } y_1 + 2y_2 \leq 4 \\ &\quad \quad \quad y_1 - 2y_2 \leq 2 \\ &\quad \quad \quad 2y_1 + 4y_2 \leq -1 \\ &\quad \quad \quad y_1 \geq 0, y_2 \geq 0 \end{aligned}$$

**16. Compare Explicit and Implicit Constraints.**

## 1) Explicit constraints:

Explicit constraints are rules that restrict each  $X_i$  to take values only from a given set. Some examples are,

$X_i > 0$  or  $S_i = \{\text{all non-negative real nos.}\}$

$X_i = 0$  or 1 or  $S_i = \{0, 1\}$ .

$L_i X_i U_i$  or  $S_i = \{a: L_i \leq a \leq U_i\}$

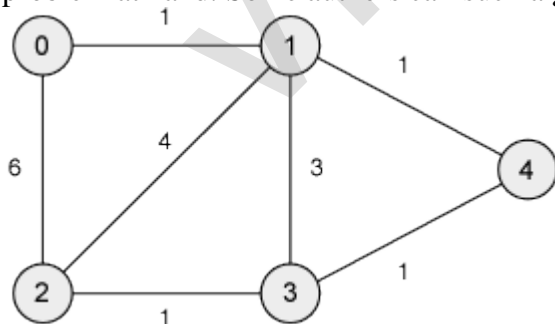
All tuples that satisfy the explicit constraint define a possible solution space.

## 2) Implicit constraints:

The implicit constraint determines which of the tuples in the solution space can actually satisfy the criterion functions.

**17. Define weighted graph.**

A weighted graph is a graph in which a number (the weight) is assigned to each edge. such weights might represent for example costs, lengths or capacities, depending on the problem at hand. Some authors call such a graph a network.

**18. Define multistage graph.**

A multistage graph is a graph

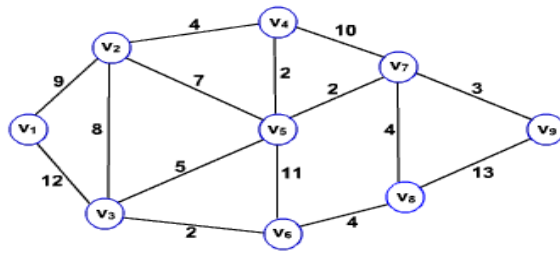
- $G=(V,E)$  with  $V$  partitioned into  $K \geq 2$  disjoint subsets such that if  $(a,b)$  is in  $E$ , then  $a$  is in  $V_i$ , and  $b$  is in  $V_{i+1}$  for some subsets in the partition;
- and  $|V_1| = |V_K| = 1$ .

The vertex  $s$  in  $V_1$  is called the source; the vertex  $t$  in  $V_K$  is called the sink.

$G$  is usually assumed to be a **weighted graph**.

The cost of a path from node  $v$  to node  $w$  is sum of the costs of edges in the path.

The "multistage graph problem" is to find the minimum cost path from  $s$  to  $t$ .



### 19. Define source and sink node of graph.

A Flow graph contains 1 source node and 1 sink node.

Source node: Unique vertex with no entering edges.

Sink node: Unique vertex with no leaving edges.

### 20. What is Maximum Flow Problem?

Problem of maximizing the flow of a material through a transportation network (e.g., pipeline system, communications or transportation networks)

Formally represented by a connected weighted digraph with  $n$  vertices numbered from 1 to  $n$  with the following properties:

- contains exactly one vertex with no entering edges, called the *source* (numbered 1)
- contains exactly one vertex with no leaving edges, called the *sink* (numbered  $n$ )
- has positive integer weight  $u_{ij}$  on each directed edge  $(i,j)$ , called the *edge capacity*, indicating the upper bound on the amount of the material that can be sent from  $i$  to  $j$  through this edge.
- A digraph satisfying these properties is called a **flow network** or simply a network.

### 21. Define Flow

A *flow* is an assignment of real numbers  $x_{ij}$  to edges  $(i,j)$  of a given network that satisfy the following:

- *flow-conservation* *requirements*  
The total amount of material entering an intermediate vertex must be equal to the total amount of the material leaving the vertex
- *capacity constraints*

$$0 \leq x_{ij} \leq u_{ij} \text{ for every edge } (i,j) \in E$$

### 22. What is Flow value and Maximum Flow Problem

Since no material can be lost or added to by going through intermediate vertices of the network, the total amount of the material leaving the source must end up at the sink:

$$\sum_{j: (1,j) \in E} x_{1j} = \sum_{j: (j,n) \in E} x_{jn}$$

The *value* of the flow is defined as the total outflow from the source (= the total inflow into the sink). The *maximum flow problem* is to find a flow of the largest value (maximum flow) for a given network.

### 23. Write the Maximum-Flow Problem as LP problem.

$$\text{Maximize } v = \sum_{j: (1,j) \in E} x_{1j}$$

subject to

$$\sum_{j: (j,i) \in E} x_{ji} - \sum_{j: (i,j) \in E} x_{ij} = 0 \quad \text{for } i = 2, 3, \dots, n-1$$

$$0 \leq x_{ij} \leq u_{ij} \text{ for every edge } (i,j) \in E$$

**24. Explain Augmenting Path (Ford-Fulkerson) Method**

- Start with the zero flow ( $x_{ij} = 0$  for every edge).
- On each iteration, try to find a *flow-augmenting path* from source to sink, which a path along which some additional flow can be sent.
- If a flow-augmenting path is found, adjust the flow along the edges of this path to get a flow of increased value and try again.
- If no flow-augmenting path is found, the current flow is maximum.

**25. Write Shortest-Augmenting-Path Algorithm**

Generate augmenting path with the least number of edges by BFS as follows.

Starting at the source, perform BFS traversal by marking new (unlabeled) vertices with two labels:

- first label – indicates the amount of additional flow that can be brought from the source to the vertex being labeled
- second label – indicates the vertex from which the vertex being labeled was reached, with “+” or “-” added to the second label to indicate whether the vertex was reached via a forward or backward edge.

**26. Write about Vertex labeling**

- The source is always labeled with  $\infty$ ,
- All other vertices are labeled as follows:
  - If unlabeled vertex  $j$  is connected to the front vertex  $i$  of the traversal queue by a directed edge from  $i$  to  $j$  with positive unused capacity  $r_{ij} = u_{ij} - x_{ij}$  (forward edge), vertex  $j$  is labeled with  $l_j, i^+$ , where  $l_j = \min\{l_i, r_{ij}\}$
  - If unlabeled vertex  $j$  is connected to the front vertex  $i$  of the traversal queue by a directed edge from  $j$  to  $i$  with positive flow  $x_{ji}$  (backward edge), vertex  $j$  is labeled  $l_j, i^-$ , where  $l_j = \min\{l_i, x_{ji}\}$
- If the sink ends up being labeled, the current flow can be augmented by the amount indicated by the sink's first label.
- The augmentation of the current flow is performed along the augmenting path traced by following the vertex second labels from sink to source; the current flow quantities are increased on the forward edges and decreased on the backward edges of this path.
- If the sink remains unlabeled after the traversal queue becomes empty, the algorithm returns the current flow as maximum and stops.

**27. Define Cut**

Let  $X$  be a set of vertices in a network that includes its source but does not include its sink, and let  $X^c$ , the complement of  $X$ , be the rest of the vertices including the sink. The *cut* induced by this partition of the vertices is the set of all the edges with a tail in  $X$  and a head in  $X^c$ .

*Capacity of a cut* is defined as the sum of capacities of the edges that compose the cut.

- We'll denote a cut and its capacity by  $C(X, X^c)$  and  $c(X, X^c)$
- Note that if all the edges of a cut were deleted from the network, there would be no directed path from source to sink
- *Minimum cut* is a cut of the smallest capacity in a given network

**28. Write Max-Flow Min-Cut Theorem**

1. The value of maximum flow in a network is equal to the capacity of its minimum cut
2. The shortest augmenting path algorithm yields both a maximum flow and a minimum cut:
  - Maximum flow is the final flow produced by the algorithm
  - Minimum cut is formed by all the edges from the labeled vertices to unlabeled vertices on the last iteration of the algorithm.

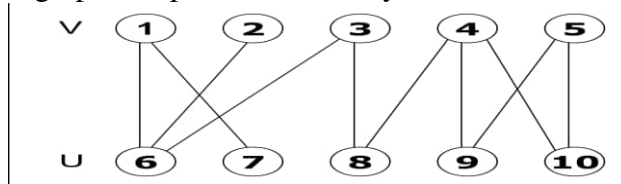


- All the edges from the labeled to unlabeled vertices are full, i.e., their flow amounts are equal to the edge capacities, while all the edges from the unlabeled to labeled vertices, if any, have zero flow amounts on them.

Maximum flow=minimum cut

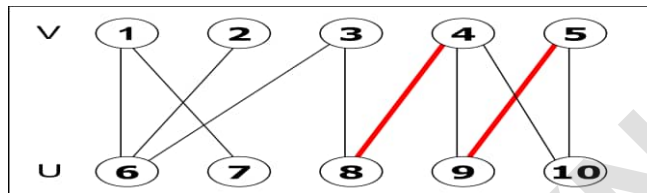
## 29. Define Bipartite Graphs

*Bipartite graph*: a graph whose vertices can be partitioned into two disjoint sets  $V$  and  $U$ , not necessarily of the same size, so that every edge connects a vertex in  $V$  to a vertex in  $U$ . A graph is bipartite if and only if it does not have a cycle of an odd length.



## 30. Define Matching in a Graph

A *matching* in a graph is a subset of its edges with the property that no two edges share a vertex



a matching in this graph  $M = \{(4,8), (5,9)\}$

## 31. Define maximum matching or maximum cardinality

A *maximum* (or *maximum cardinality*) *matching* is a matching with the largest number of edges

- always exists
- not always unique

## 32. Define Stable Marriage Problem

- There is a set  $Y = \{m_1, \dots, m_n\}$  of  $n$  men and a set  $X = \{w_1, \dots, w_n\}$  of  $n$  women. Each man has a ranking list of the women, and each woman has a ranking list of the men (with no ties in these lists).
- A *marriage matching*  $M$  is a set of  $n$  pairs  $(m_i, w_j)$ .
- A pair  $(m, w)$  is said to be a *blocking pair* for matching  $M$  if man  $m$  and woman  $w$  are not matched in  $M$  but prefer each other to their mates in  $M$ .
- A marriage matching  $M$  is called *stable* if there is no blocking pair for it; otherwise, it's called *unstable*.
- The *stable marriage problem* is to find a stable marriage matching for men's and women's given preferences.

## 33. Analyze the Gale-Shapley Algorithm

- The algorithm terminates after no more than  $n^2$  iterations with a stable marriage output.
- The stable matching produced by the algorithm is always *man-optimal*: each man gets the highest rank woman on his list under any stable marriage. One can obtain the *woman-optimal* matching by making women propose to men.
- A man (woman) optimal matching is unique for a given set of participant preferences.
- The stable marriage problem has practical applications such as matching medical-school graduates with hospitals for residency training.