

## 5.2 Shortest Path Problem

The problem investigated in this section is one of finding the collection of arcs (or path) that comprises the shortest path from a specified node  $s$ , called the source, to a second specified node  $t$ , called the destination. Figure 11 shows a typical network where  $s = 1$  and  $t = 10$ . The parameter on the arcs is enclosed in parenthesis and represents its length. Each arc is numbered sequentially.

A closely related problem, which we also consider, finds the set of shortest paths from the source node to all other nodes in the network. This *shortest path tree problem* is solved with very little additional difficulty. Figure 12 shows the shortest path tree with the source at node 1.

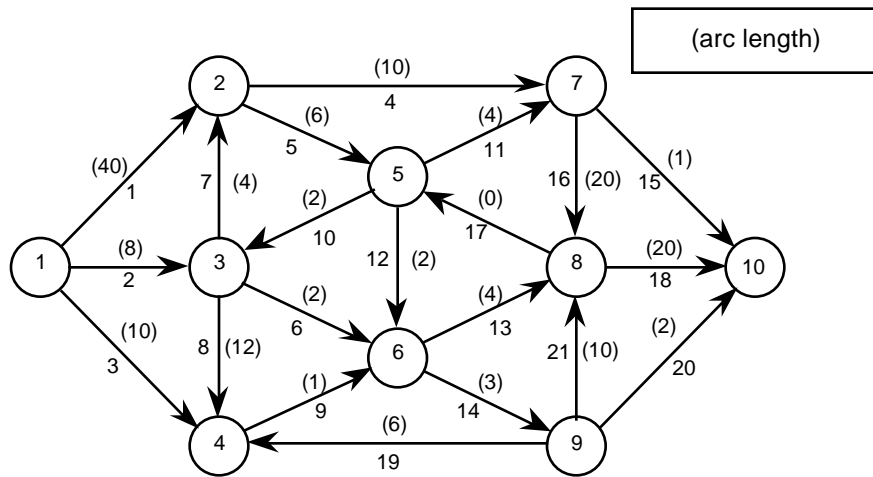


Figure 11. Network showing arc lengths

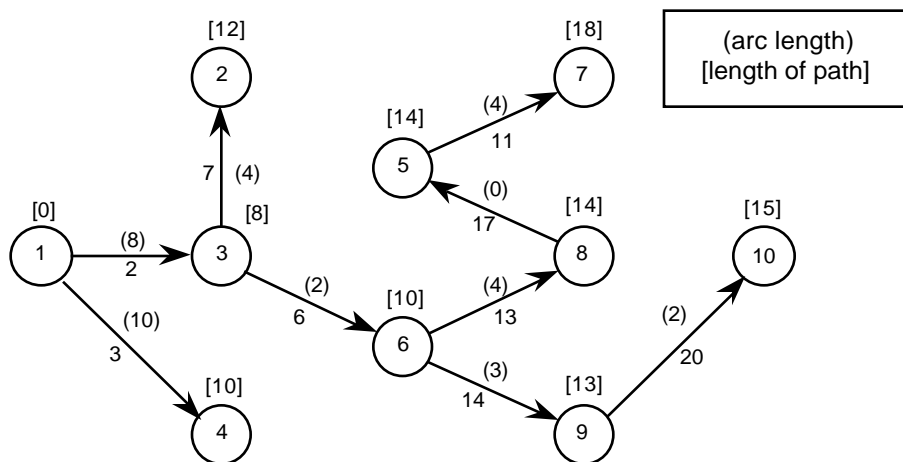


Figure 12. Shortest path tree rooted at node 1

In the following sections, we present two algorithms for the shortest path problem. The first is a greedy approach that was developed by Dijkstra and yields the optimal solution when all arc lengths are positive. If some lengths are negative but no negative cycles exist, a primal

simplex solution algorithm may be used. This is the second approach discussed.

### Dijkstra's Algorithm

When all arc lengths are nonnegative, as in Fig. 11, an efficient greedy algorithm can solve the shortest path problem. Starting with the source node alone, the algorithm finds the shortest path from the source node to one additional node in each subsequent iteration. The procedure requires  $m - 1$  iterations to find the shortest path tree.

The algorithm uses a set  $S$ , called the set of solved nodes. This set includes the nodes for which the shortest path has already been determined at the current point in the algorithm. The unsolved nodes are the nodes not in  $S$ , defined as the set  $\bar{S}$ . While iterating, the algorithm assigns the numbers  $d_i$  to each node in the network, where  $d_i$  is the length of the shortest path to node  $i$  from the source node  $s$  through the members of  $S$ . Note that the  $d_i$ -values are equivalent to the dual variables associated with the mathematical programming formulation of the problem. At the end of the algorithm  $d_i$  is the length of the shortest path to node  $i$ . In the following  $M$  is the set of all arcs.

#### Algorithm

Initially, let  $S = \{s\}$ ,  $d_s = 0$ .

Repeat until  $S$  is the set of all nodes:

Find an arc  $k(i, j)$  that passes from a solved node to an unsolved node such that:

$$k(i, j) = \operatorname{argmin}\{d_{i'} + c_{k'} : k'(i', j') \in M, i' \in S, j' \in \bar{S}\}$$

Add node  $j$  and arc  $k$  to the tree. Add node  $j$  to the solved set  $S$ .

Let  $d_j = d_i + c_k$ .

At each iteration the algorithm computes the length of the path to every unsolved node through only solved nodes. The unsolved node with the shortest path length is added to the solved set. The process terminates when a spanning tree is obtained. Since a node is added to the tree at each step the algorithm requires  $m - 1$  iterations. The algorithm works because of the assumption of nonnegative arc lengths.

#### Example

Consider the network in Fig. 11. The problem is to find the shortest path tree rooted at node 1. For hand computations, the algorithm is easily accomplished on the figure describing the problem. Figure 13 shows the intermediate situation with five nodes assigned to the tree,  $S = \{1, 3, 4, 6, 2\}$  with arcs  $\{2, 3, 6, 7\}$  (note that the order of the items in the sets shows the order in which they were added in the algorithm). The bold numbers in brackets indicate the values of  $d_i$  associated with the nodes in the set  $S$ . For example, the shortest path to node 6 is 10. The bracketed numbers on the nodes in  $\bar{S}$  show the length of the shortest paths to unsolved nodes passing through only nodes in  $S$ . For example, the shortest path passing through  $S$  to node 8 is 14. The

algorithm now selects the arc and node associated with the smallest  $j$  value for  $i \in \bar{S}$ . In reduced terms, the choice is  $\min\{18, 22, 14, 13\} = 13$  so node 9 and arc 14 join the tree.

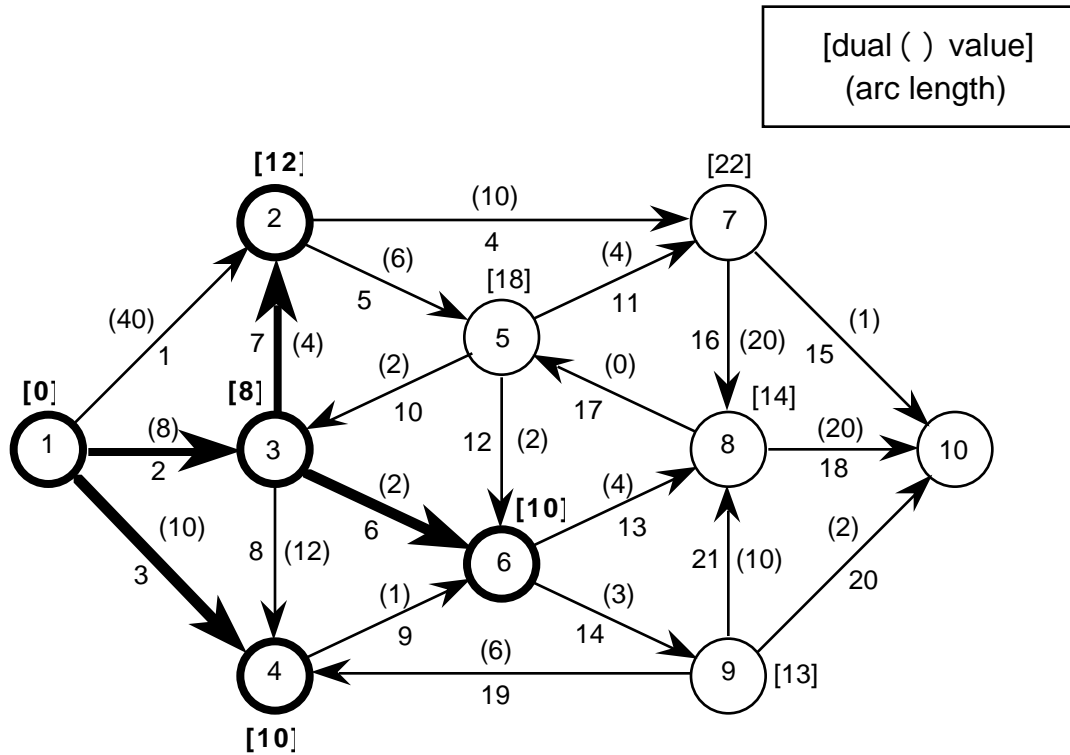


Figure 13. Intermediate point in Dijkstra's Algorithm with  $S = \{1, 3, 4, 6, 2\}$

*Tabular Implementation*

An alternative statement of the algorithm constructs a table with seven columns.

*Initialization:* Let  $h = 1$  and let  $s$  be the origin node. Let  $S = \{s\}$ , let  $s = 0$ .

*Iterative step:* Repeat until all nodes are in the set  $S$

a. Add to the table

Column 1: Show the value of  $h$ . At any step,  $h$  is the number of nodes in the  $S$ .

Column 2: List the members of  $S$  (the solved set) that have at least one arc connected to an unsolved node.

Column 3: For each node listed in column 2, find the closest unsolved node.

Column 4: Let  $i$  be the index of the node listed in column 2, let  $j$  be the index of the node listed in column 3, and let  $k$  be the index of the arc connecting nodes  $i$  and  $j$ . Compute for each case

$$j' = i + c_k.$$

Show these numbers in column 4.

Column 5: Select the smallest number in column 4. Let  $i$  be the node in column 2 and  $j$  the node in column 3 from which this number was computed. Show node  $j$  in column 5.

Column 6: Show the length of the shortest path to the node added. This is the minimum obtained from column 4.

Column 7: Show the arc  $k(i, j)$ . Add node  $j$  and arc  $k$  to the shortest path tree.

b. Add node  $j$  to  $S$ . Let  $j = j'$ .

The steps of the algorithm for the example of Fig. 11 are shown in Table 3. The optimal tree is displayed in Fig. 12.

Table 3. Shortest path computations for network in Fig. 11

$h$	Solved nodes	Closest unsolved node	Length of path to unsolved node	Node added to solved set	Length of shortest path	Arc added to tree
1	1	3	8	3	8	2
2	1	4	10	4	10	3
	3	6	10			
3	1	2	40	6	10	6
	3	6	10			
	4	6	11			
4	1	2	40	2	12	7
	3	2	12			
	6	9	13			
5	2	5	18	9	13	14
	6	9	13			
6	2	5	18	8	14	13
	6	8	14			
	9	10	15			
7	2	5	18	5	14	17
	8	5	14			
	9	10	15			
8	2	7	22	10	15	20
	5	7	18			
	8	10	34			
	9	10	15			
9	2	7	22	7	18	11
	5	7	18			

### Primal Simplex Algorithm

The primal simplex algorithm can also be used to solve the shortest path tree problem but is not limited to the case of all nonnegative arc lengths. It required, however, that there exist no directed cycles with negative total length. If such a cycle were present, the simplex algorithm would yield an unbounded solution.

The algorithm described below is a variant on the pure network flow programming algorithm discussed in Section 5.4 of this chapter. As with all simplex-based procedures, the optimal solution to the shortest path problem is a basic solution. The algorithm for the shortest path problem makes use of the critical property that any collection of arcs that forms a spanning tree is a basic solution.

### Algorithm

1. Start with a basis described by the arcs of a spanning tree. Assign the dual value  $u_s = 0$ .
2. Repeat until all marginal costs are nonnegative:
  - a. Compute the dual values for all nodes but the source node  $s$  such that if arc  $k(i, j)$  is in the basis
 
$$u_j = u_i + c_k.$$
  - b. Compute the marginal costs,  $d_k$ , for each nonbasic arc  $k(i, j)$ , where
 
$$d_k = u_i + c_k - u_j.$$
  - c. Select some nonbasic arc for which  $d_k < 0$  and call it the entering arc. The leaving arc is the arc in the tree that currently enters node  $j$ .
  - d. Change the basis by removing the leaving arc from the tree and adding the entering arc. Compute the dual solution associated with the new basis.

### Example

We use the example problem in Fig. 11 to illustrate the computations but assume arc 4(2, 7) has length  $-10$  rather than 10. The arcs shown in Fig. 12 form the initial basis. The node labels in that figure are equivalent to the values of  $u_i$  computed in Step 2a of the algorithm.

With  $c_4 = -10$ , we compute for arc 4

$$d_4 = u_2 + c_4 - u_7 = 12 - 10 - 18 = -6.$$

Because  $d_4$  is negative, we select arc 4 to enter the basis. According to Step 2c of the algorithm the arc that currently enters node 7 must leave the basis. We change the basis and obtain the new spanning tree in Fig. 14. The dual values are shown adjacent to the nodes.

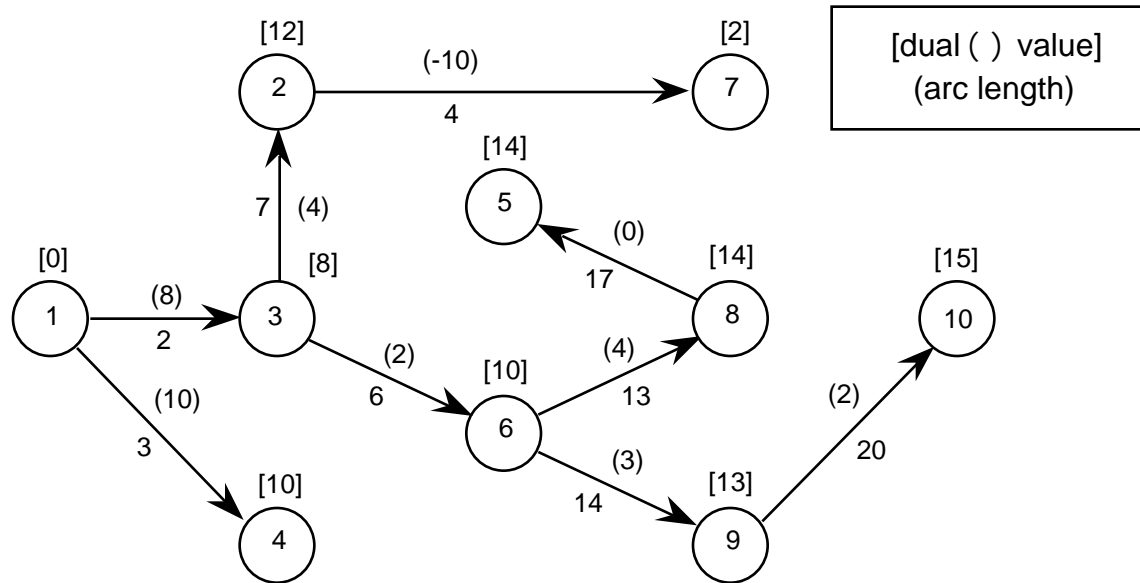


Figure 14. New basis after an iteration of the primal simplex algorithm

At the next iteration we find that arcs 15 and 16 are candidates to enter the basis, so the solution in Fig. 14 is not optimal.