DIJKSTRA'S ALGORITHM

- Dijkstra's Algorithm solves the **single-source shortest-pathsproblem**.
- For a given vertex called the ***source*** in a weighted connected graph, find shortest paths to all its othervertices.
- The single-source shortest-paths problem asks for a family of paths, each leading from the source to a different vertex in the graph, though some paths may, of course, have **edges in common**.
- The most widely used **applications** are transportation planning and packet routing in communication networks including the Internet.
- It also includes **finding shortest paths** in social networks, speech recognition, document formatting, robotics, compilers, and airline crew scheduling.
- In the world of **entertainment**, one can mention pathfinding in video games and finding best solutions to puzzles using their state-spacegraphs.
- Dijkstra's algorithm is the best-known algorithm for the single-source shortest-paths problem.

ALGORITHM *Dijkstra(G,s)*

//Dijkstra's algorithm for single-source shortest paths
//Input: A weighted connected graph $G = (V, E)$ with nonnegative weights and its vertex s
//Output: The length *dv* of a shortest path from s to v and its penultimate vertex *pv* for every
// vertex v in V
Initialize(Q) //initialize priority queue to empty

for every vertex v in V

dv ← ∞; *pv* ← **null**

Insert (Q, v, dv) //initialize vertex priority in the priority queue

Ds ← 0; *Decrease(Q, s, d_s)* //update priority of s with d_s V_T← Φ

for i ←0 **to** $|V| - 1$ **do**

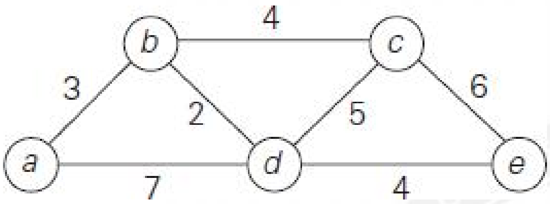
u^*← *DeleteMin(Q)* //delete the minimum priority element

V_T←$V_T \cup \{u^*\}$

for every vertex u in $V - VT$ that is adjacent to u^* **do if** $d_{u^*} + w(u^*, u) < d_u$

$d_u \leftarrow d_{u^*} + w(u^*, u)$;
$p_u \leftarrow u^*$ Decrease(Q,
u, d_u)

The time efficiency of Dijkstra's algorithm depends on the data structures used for implementing the priority queue and for representing an input graph itself. It is in Θ (|V |^2) for graphs represented by their weight matrix and the priority queue implemented as an unordered array. For graphs represented by their adjacency lists and the priority queue implemented as a min- heap, it is in O(|E| log |V |).



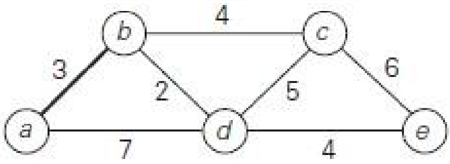
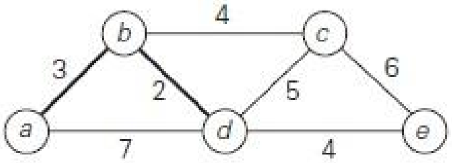
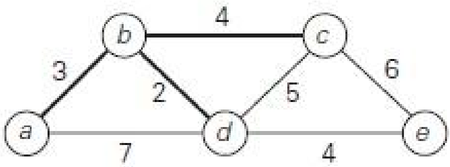
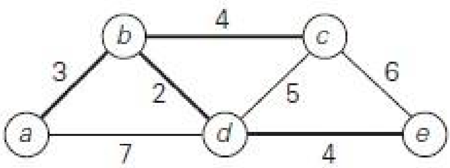
Tree vertices	Remaining vertices	Illustration
a(−, 0)	**b(a, 3)** c(−, ∞) d(a, 7) e(−, ∞)	
b(a, 3)	c(b, 3 + 4) **d(b, 3 + 2)** e(−, ∞)	
d(b, 5)	**c(b, 7)** e(d, 5 + 4)	
c(b, 7)	**e(d, 9)**	
e(d, 9)		

FIGURE 3.16 Application of Dijkstra's algorithm. The next closest vertex is shown in bold

The shortest paths (identified by following nonnumeric labels backward from a destination vertex in the left column to the source) and their lengths (given by numeric labels of the tree vertices) are as follows:

From a to b : a − b of length 3
From a to d : a − b − d of length 5
From a to c : a − b − c of length 7
From a to e : a − b − d − e of length 9