Basic Theory of Graph (Part 4)

Shortest Path Problem and Dijkstra's Algorithm, Traveling Salesperson Problem, and Chinese Postman Problem

MZI

School of Computing
Telkom University

SoC Tel-U

May 2023

Acknowledgements

This slide is composed based on the following materials:

1. *Discrete Mathematics and Its Applications*, 8th Edition, 2019, by K. H. Rosen (main).
2. *Discrete Mathematics with Applications*, 5th Edition, 2018, by S. S. Epp.
3. *Mathematics for Computer Science*. MIT, 2010, by E. Lehman, F. T. Leighton, A. R. Meyer.
4. Slide for Matematika Diskret 2 (2012). Fasilkom UI, by B. H. Widjaja.
5. Slide for Matematika Diskret 2 at Fasilkom UI by Team of Lecturers.
6. Slide for Matematika Diskret. Telkom University, by B. Purnama.

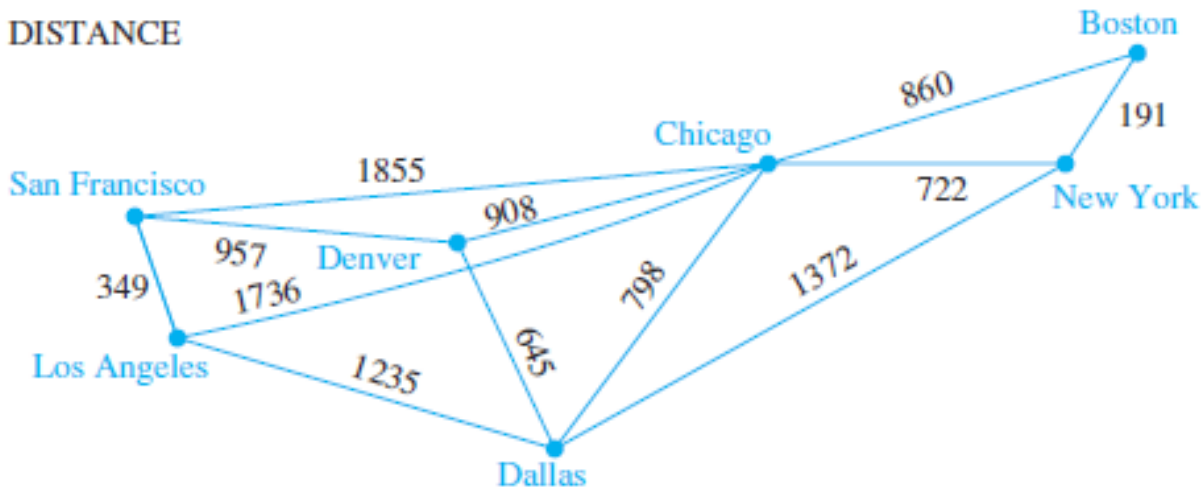
Some of the pictures are taken from the above resources. This slide is intended for academic purpose at FIF Telkom University. If you have any suggestions/comments/questions related with the material on this slide, send an email to `<pleasedontspam>@telkomuniversity.ac.id`.

Contents

Contents

Motivation: Determining The Shortest Path

Observe the graph whose edges have the following distance information.



We are interested to determine the shortest path (**with minimum distance**) between two particular cities. The problem of determining the path with minimum distance is called as the shortest path problem.

Weighted Graph

To determine the shortest path between two vertices, we use weighted graph model.

Definition (Weighted graph)

Suppose $G = (V, E, f)$ is an undirected graph. Graph G is called as a weighted graph if each edge is given weight label. Usually the weight is a positive real number. Weight of an edge $\{u, v\}$ is denoted as $w(u, v)$. In this case $w(u, v) = w(v, u)$.

An example of a weighted graph is as follows.

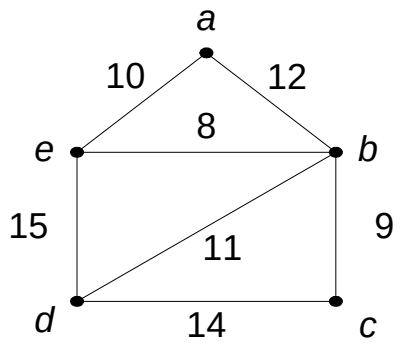
Weighted Graph

To determine the shortest path between two vertices, we use weighted graph model.

Definition (Weighted graph)

Suppose $G = (V, E, f)$ is an undirected graph. Graph G is called as a weighted graph if each edge is given weight label. Usually the weight is a positive real number. Weight of an edge $\{u, v\}$ is denoted as $w(u, v)$. In this case $w(u, v) = w(v, u)$.

An example of a weighted graph is as follows.



On the above graph, weight of the edge $\{a, b\}$ is $w(a, b) = w(b, a) =$

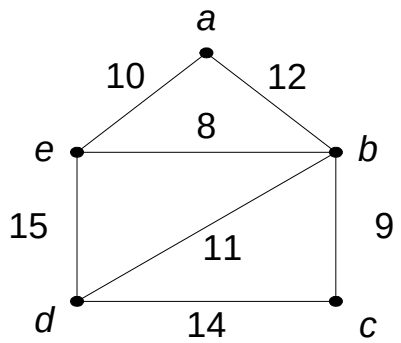
Weighted Graph

To determine the shortest path between two vertices, we use weighted graph model.

Definition (Weighted graph)

Suppose $G = (V, E, f)$ is an undirected graph. Graph G is called as a weighted graph if each edge is given weight label. Usually the weight is a positive real number. Weight of an edge $\{u, v\}$ is denoted as $w(u, v)$. In this case $w(u, v) = w(v, u)$.

An example of a weighted graph is as follows.



On the above graph, weight of the edge $\{a, b\}$ is $w(a, b) = w(b, a) = 12$. Weight of the edge $\{b, c\}$ is $w(b, c) = w(c, b) =$

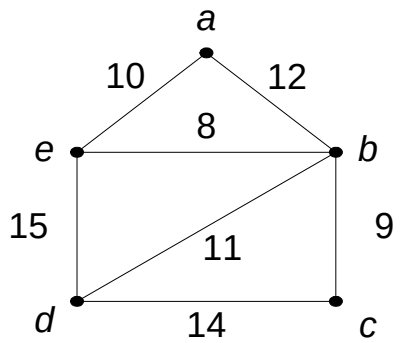
Weighted Graph

To determine the shortest path between two vertices, we use weighted graph model.

Definition (Weighted graph)

Suppose $G = (V, E, f)$ is an undirected graph. Graph G is called as a weighted graph if each edge is given weight label. Usually the weight is a positive real number. Weight of an edge $\{u, v\}$ is denoted as $w(u, v)$. In this case $w(u, v) = w(v, u)$.

An example of a weighted graph is as follows.



On the above graph, weight of the edge $\{a, b\}$ is $w(a, b) = w(b, a) = 12$. Weight of the edge $\{b, c\}$ is $w(b, c) = w(c, b) = 9$.

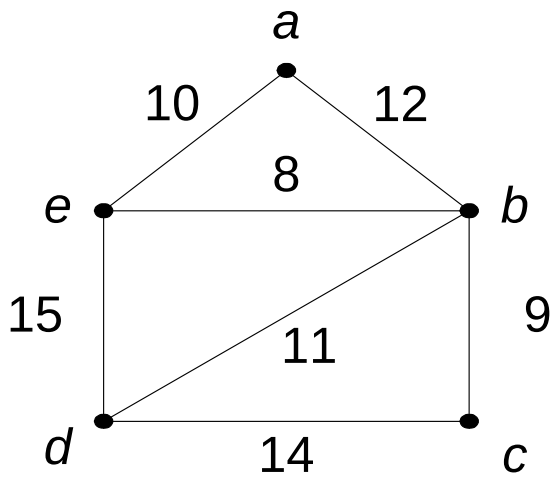
Distance Matrix of The Weighted Graph

Definition

Suppose $G = (V, E)$ is an undirected weighted graph with no multiple edges and no loop. If the weight of an edge $\{i, j\}$ is $w(i, j)$, then the distance matrix for graph G is $\mathbf{D}_G$ defined as

$$\mathbf{D}_G[i, j] = \begin{cases} w(i, j), & \text{if vertices } i \text{ and } j \text{ are adjacent,} \\ \infty, & \text{if vertices } i \text{ and } j \text{ are not adjacent.} \end{cases}$$

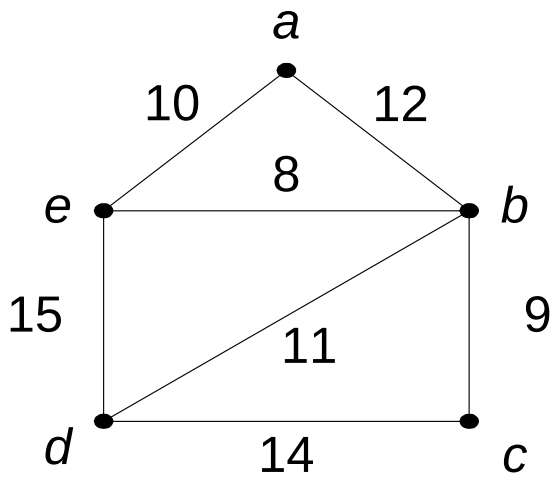
For example, for the following graph G



Then

$$\mathbf{D}_G =$$

For example, for the following graph G



Then

$$\mathbf{D}_G = \begin{array}{c|ccccc} & a & b & c & d & e \\ \hline a & \infty & 12 & \infty & \infty & 10 \\ b & 12 & \infty & 9 & 11 & 8 \\ c & \infty & 9 & \infty & 14 & \infty \\ d & \infty & 11 & 14 & \infty & 15 \\ e & 10 & 8 & \infty & 15 & \infty \end{array}$$

Types of Shortest Path Problems

There are several types of shortest path problems that can be explored:

1. shortest path between two particular vertices,
2. shortest path between all pairs of vertices on the graph,
3. shortest path from one particular vertex to all vertices on the graph,
4. shortest path between two vertices that must traverse some particular vertices.

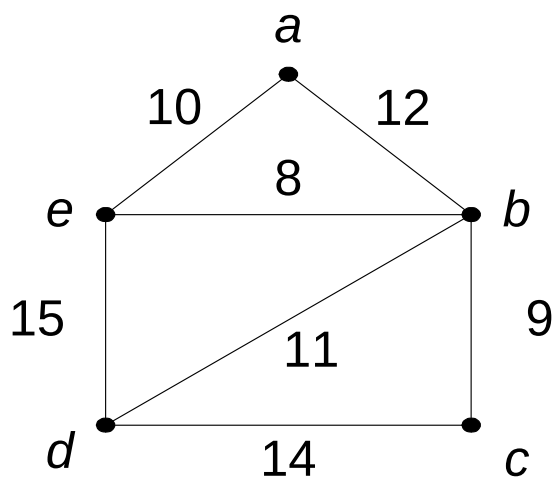
Contents

Weight of a Path

Definition (Path's Length)

Suppose $G = (V, E, f)$ is an undirected weighted connected graph and $u, v \in V$. The length of path from u to v, $L(u, v)$, is the sum of weight on the edges on the path from u to v.

For example, on the following graph



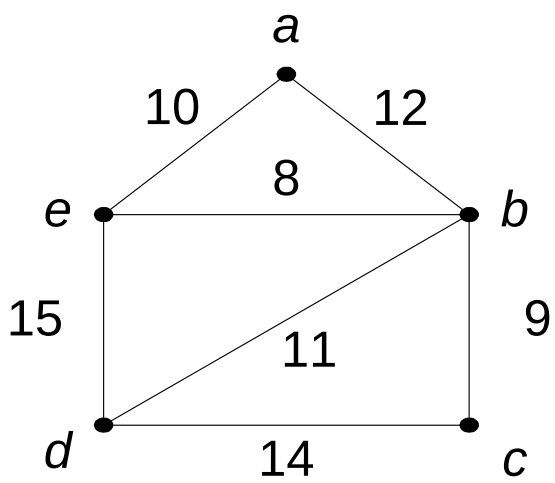
The length of the shortest path from a to c is

Weight of a Path

Definition (Path's Length)

Suppose $G = (V, E, f)$ is an undirected weighted connected graph and $u, v \in V$. The length of path from u to v, $L(u, v)$, is the sum of weight on the edges on the path from u to v.

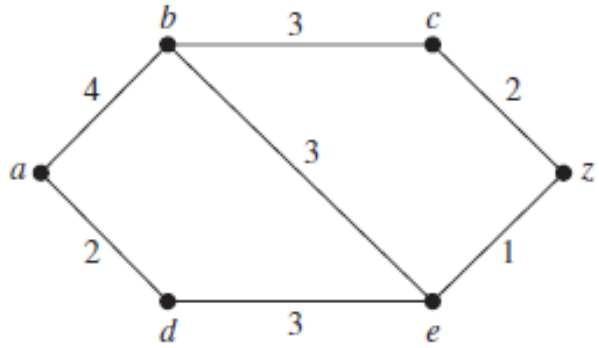
For example, on the following graph



The length of the shortest path from a to c is $12 + 9 = 21$, so $L(a, c) = 21$ for the path $\langle a, b, c \rangle$.

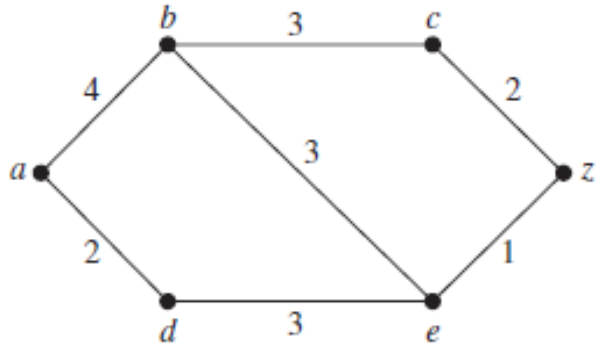
Example of Determining Simple Shortest Path

We will find the shortest path from a to z on the following graph.



Example of Determining Simple Shortest Path

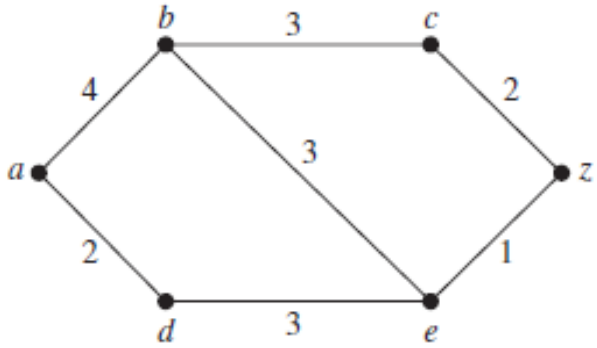
We will find the shortest path from a to z on the following graph.



1. There are two paths that contain two vertices with a as initial vertex, namely $\langle a, b\rangle$ and $\langle a, d\rangle$. Notice that $L(a, b) = 4$ and $L(a, d) = 2$. Therefore, d is the nearest vertex from a.

Example of Determining Simple Shortest Path

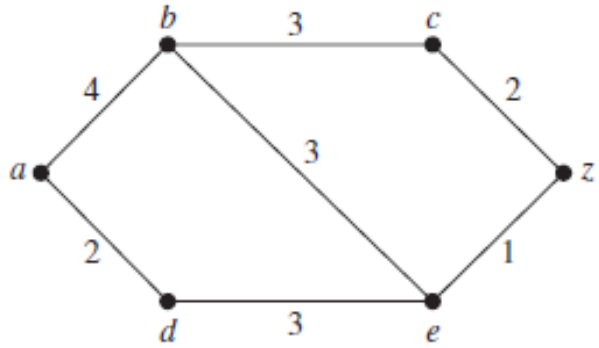
We will find the shortest path from a to z on the following graph.



1. There are two paths that contain two vertices with a as initial vertex, namely $\langle a, b\rangle$ and $\langle a, d\rangle$. Notice that $L(a, b) = 4$ and $L(a, d) = 2$. Therefore, d is the nearest vertex from a.
2. Afterwards, we will find the next nearest vertex. Notice that the shortest path from a to b is $\langle a, b\rangle$ with $L(a, b) = 4$. Then, the shortest path from a to e is $\langle a, d, e\rangle$, with $L(a, e) = 5$.

Example of Determining Simple Shortest Path

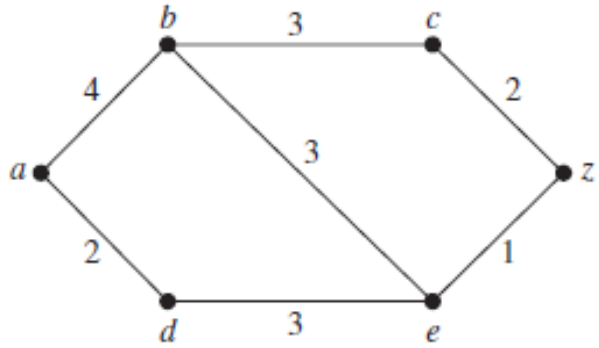
We will find the shortest path from a to z on the following graph.



1. There are two paths that contain two vertices with a as initial vertex, namely $\langle a, b\rangle$ and $\langle a, d\rangle$. Notice that $L(a, b) = 4$ and $L(a, d) = 2$. Therefore, d is the nearest vertex from a.
2. Afterwards, we will find the next nearest vertex. Notice that the shortest path from a to b is $\langle a, b\rangle$ with $L(a, b) = 4$. Then, the shortest path from a to e is $\langle a, d, e\rangle$, with $L(a, e) = 5$.
3. The shortest path from a to c is $\langle a, b, c\rangle$ with $L(a, c) = 7$. The shortest path from a to z is $\langle a, d, e, z\rangle$, with $L(a, z) = 6$.

Example of Determining Simple Shortest Path

We will find the shortest path from a to z on the following graph.



1. There are two paths that contain two vertices with a as initial vertex, namely $\langle a, b\rangle$ and $\langle a, d\rangle$. Notice that $L(a, b) = 4$ and $L(a, d) = 2$. Therefore, d is the nearest vertex from a.
2. Afterwards, we will find the next nearest vertex. Notice that the shortest path from a to b is $\langle a, b\rangle$ with $L(a, b) = 4$. Then, the shortest path from a to e is $\langle a, d, e\rangle$, with $L(a, e) = 5$.
3. The shortest path from a to c is $\langle a, b, c\rangle$ with $L(a, c) = 7$. The shortest path from a to z is $\langle a, d, e, z\rangle$, with $L(a, z) = 6$.

So the shortest path from a to z is $\langle a, d, e, z\rangle$ of length 6. Notice that not all vertices on the graph G must be visited.

Dijkstra's Algorithm: Initialization

- The following Dijkstra's Algorithm is used to find the shortest path from a vertex (i.e., a) to another vertex (i.e., z).
- The notation $L_k(v)$ denotes $L_k(a, v)$, namely the length of the shortest path from vertex a to v after k-th iteration.
- On the 0-th iteration (initialization step), $L_0(a) = 0$ and $L_0(v) = \infty$ for every vertices except a.
- The notation S_k denotes a set of vertices that have been checked on a path after k-th iteration.
  1. For the initialization $S_0 = \emptyset$.
  2. The set S_k is obtained from set S_{k-1} by adding a vertex u that is not in S_{k-1} whose $L_{k-1}(u)$ is minimum.

Important Properties of Dijkstra's Algorithm

Dijkstra's algorithm use the following important properties.

Suppose $v \notin S_k$. The shortest path from a to v that contains vertices on S_k is one of the following paths

Important Properties of Dijkstra's Algorithm

Dijkstra's algorithm use the following important properties.

Suppose $v \notin S_k$. The shortest path from a to v that contains vertices on S_k is one of the following paths

1. a path that connecting a to v that contains every vertex on S_{k-1}, or

Important Properties of Dijkstra's Algorithm

Dijkstra's algorithm use the following important properties.

Suppose $v \notin S_k$. The shortest path from a to v that contains vertices on S_k is one of the following paths

1. a path that connecting a to v that contains every vertex on S_{k-1}, or
2. a path that consists of a path connecting a to u (for a vertex u on S_{k-1}) and edge $\{u, v\}$.

From these two properties, we obtain

$$L_k(a, v) \quad =$$

Important Properties of Dijkstra's Algorithm

Dijkstra's algorithm use the following important properties.

Suppose $v \notin S_k$. The shortest path from a to v that contains vertices on S_k is one of the following paths

1. a path that connecting a to v that contains every vertex on S_{k-1}, or
2. a path that consists of a path connecting a to u (for a vertex u on S_{k-1}) and edge $\{u, v\}$.

From these two properties, we obtain

$$\begin{aligned} L_k(a,v) &= \min\left\{L_{k-1}(a,v), L_{k-1}(a,u) + w(u,v)\right\}, \text{ or} \\ L_k(v) &= \end{aligned} \tag{1}$$

Important Properties of Dijkstra's Algorithm

Dijkstra's algorithm use the following important properties.

Suppose $v \notin S_k$. The shortest path from a to v that contains vertices on S_k is one of the following paths

1. a path that connecting a to v that contains every vertex on S_{k-1}, or
2. a path that consists of a path connecting a to u (for a vertex u on S_{k-1}) and edge $\{u, v\}$.

From these two properties, we obtain

$$L_k(a,v) = \min\{L_{k-1}(a,v), L_{k-1}(a,u) + w(u,v)\}, \text{ or} \quad (1)$$

$$L_k(v) = \min\{L_{k-1}(v), L_{k-1}(u) + w(u,v)\}. \quad (2)$$

In the formulation (1) as well as (2), u is a vertex with the following properties

$$L_{k-1}(u) \leq L_{k-1}(x), \text{ for all } x \in S_{k-1}.$$

In other words $L_{k-1}(u)$ denotes the length of the shortest path from a to u after $(k-1)$-th iteration.

Dijkstra's Algorithm: Pseudocode

Dijkstra's Algorithm: Initialization

1. Input: $G = (V, E)$ that is simply connected with positive edge weights.
2. The vertices on G are $a = v_0, v_1, v_2, \ldots, v_{n-1}, v_n = z$.
3. Initialize $L_0(a) = 0$ and $L_0(v) = \infty$ for all $v \neq a$, $v \in V$.
4. Initialize $S = \emptyset$. The set S is a set of vertices that been checked. One of the subset of S is a set of vertices that are in the shortest path from a to z. Hence, the set of vertices that are in the path is not always identical to S.
5. `for` $i := 1$ `to` n
6. $\quad L(v_i) := \infty$
7. `end for`
8. $L(a) := 0$.

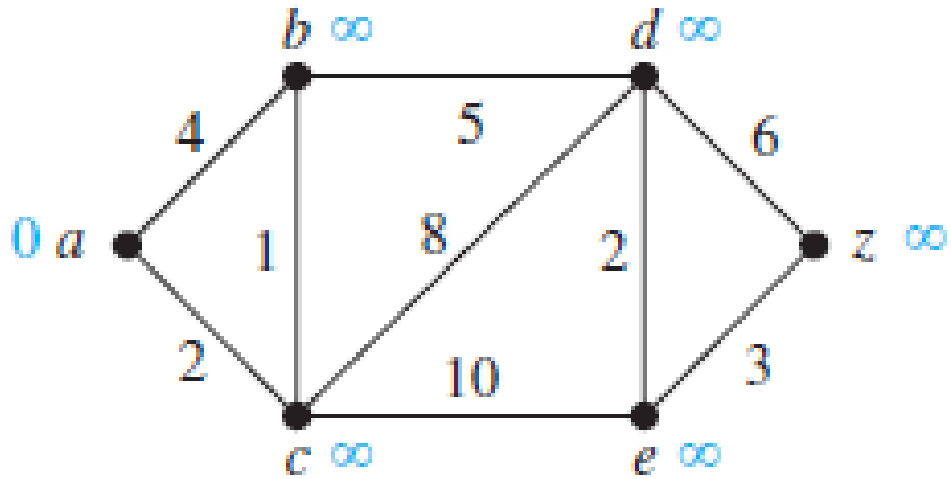
Dijkstra's Algorithm: Iteration

1. `while` $z \notin S$
2. `begin`
3. $u :=$ a vertex that is not in S with $L(u)$ minimal
4. $S := S \cup \{u\}$
5. `for all` $v \notin S$
6. $L(v) := \min\{L(v), L(u) + w(u, v)\}$
   {this adds a vertex to S with minimum label
   and updates the labels of vertices that are not in S}
7. `end for`
8. `end while`
9. $L(z)$ is the length of the shortest path from a to z.
10. If elements $S = \{a = u_0, u_1, \ldots, u_k = z\}$ are sorted, then a subset of S with a particular order is the shortest path from a to z.

Contents

Dijkstra's Algorithm Example

We will use Dijkstra's algorithm to determine the shortest path from a to z on the following graph.

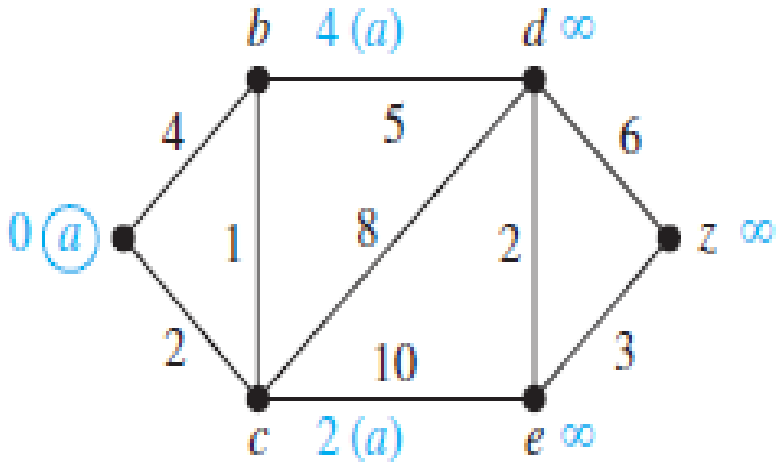


Graph G

Initialization

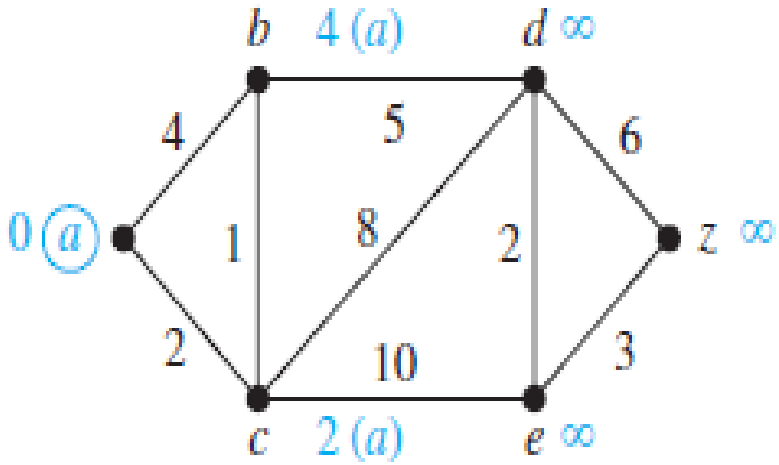
1. $S_0 = \emptyset$.
2. $L_0(a) = 0$.
3. $L_0(b) = L_0(c) = L_0(d) = L_0(e) = L_0(z) = \infty$.
4. Because $z \notin S_0$, the iteration is continued.

First Iteration



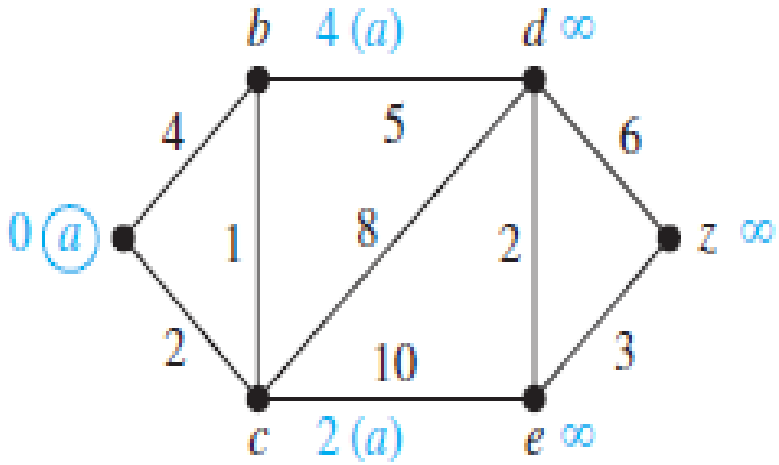
1. $u := a$ (because $a \notin S_0$ and $L_0(a)$ is minimal).

First Iteration



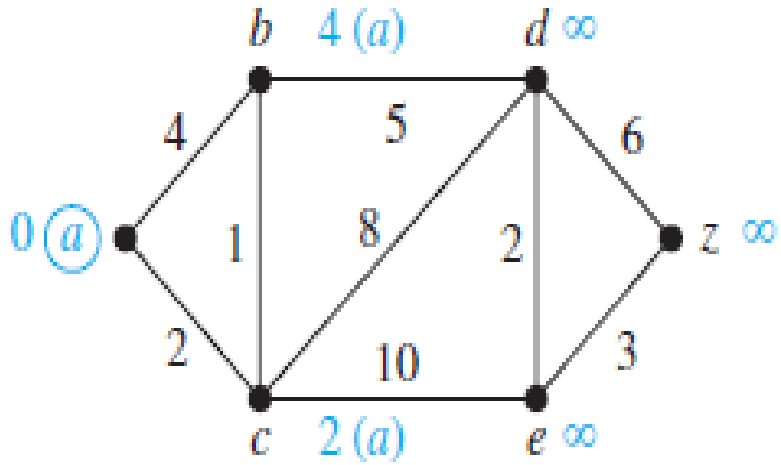
1. $u := a$ (because $a \notin S_0$ and $L_0(a)$ is minimal).
2. $S_1 = S_0 \cup \{a\} = \emptyset \cup \{a\} = \{a\}$.
3. $L_1(b) = \min\{L_0(b), L_0(a) + w(a,b)\} = \min\{\infty, 0+4\} =$

First Iteration



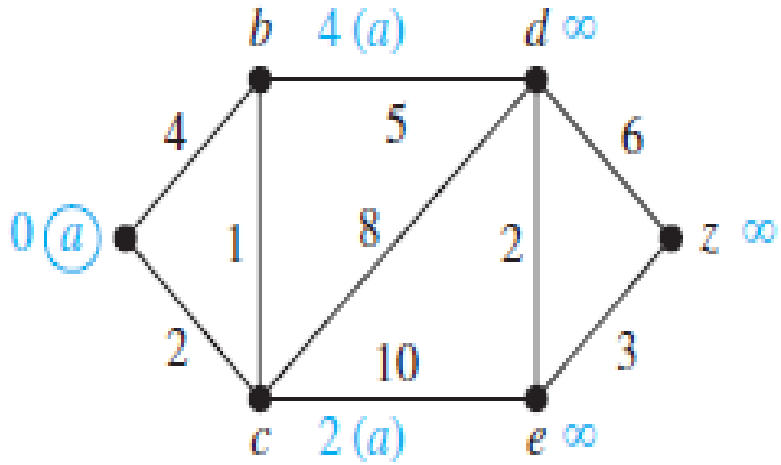
1. $u := a$ (because $a \notin S_0$ and $L_0(a)$ is minimal).
2. $S_1 = S_0 \cup \{a\} = \emptyset \cup \{a\} = \{a\}$.
3. $L_1(b) = \min\{L_0(b), L_0(a) + w(a,b)\} = \min\{\infty, 0+4\} = 4$. Path:

First Iteration



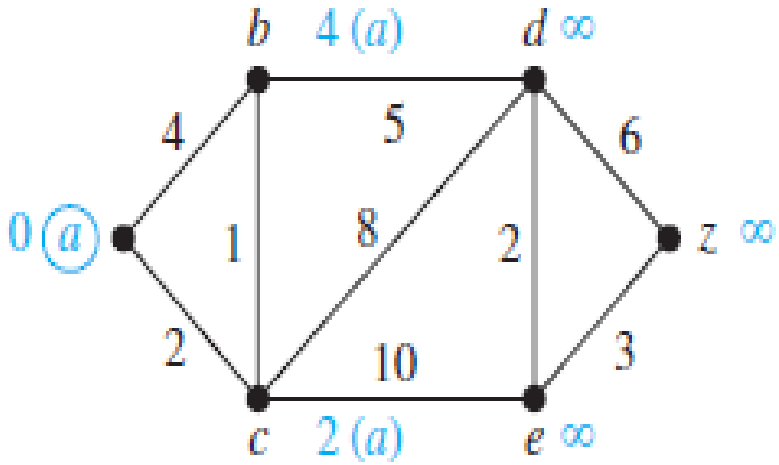
1. $u := a$ (because $a \notin S_0$ and $L_0(a)$ is minimal).
2. $S_1 = S_0 \cup \{a\} = \emptyset \cup \{a\} = \{a\}$.
3. $L_1(b) = \min\{L_0(b), L_0(a) + w(a,b)\} = \min\{\infty, 0+4\} = 4$. Path: $\langle a, b \rangle$.
   $L_1(c) = \min\{L_0(c), L_0(a) + w(a,c)\} = \min\{\infty, 0+2\} =$

First Iteration



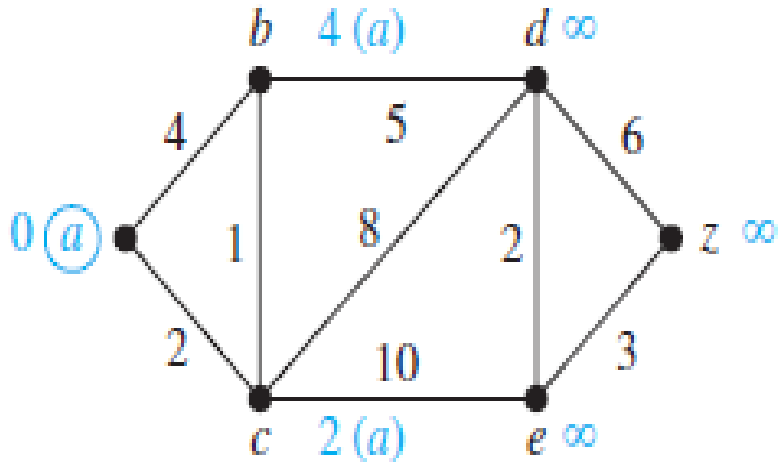
1. $u := a$ (because $a \notin S_0$ and $L_0(a)$ is minimal).
2. $S_1 = S_0 \cup \{a\} = \emptyset \cup \{a\} = \{a\}$.
3. $L_1(b) = \min\{L_0(b), L_0(a) + w(a,b)\} = \min\{\infty, 0+4\} = 4$. Path: $\langle a, b \rangle$.
   $L_1(c) = \min\{L_0(c), L_0(a) + w(a,c)\} = \min\{\infty, 0+2\} = 2$. Path:

First Iteration



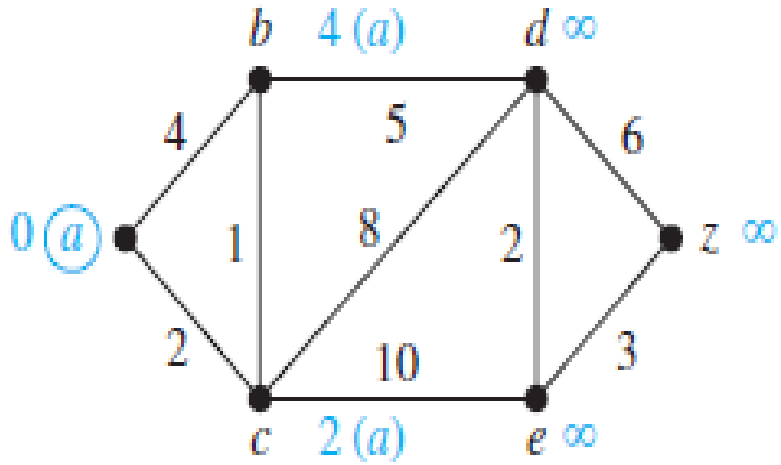
1. $u := a$ (because $a \notin S_0$ and $L_0(a)$ is minimal).
2. $S_1 = S_0 \cup \{a\} = \emptyset \cup \{a\} = \{a\}$.
3. $L_1(b) = \min\{L_0(b), L_0(a) + w(a,b)\} = \min\{\infty, 0+4\} = 4$. Path: $\langle a, b \rangle$.
   $L_1(c) = \min\{L_0(c), L_0(a) + w(a,c)\} = \min\{\infty, 0+2\} = 2$. Path: $\langle a, c \rangle$.

First Iteration



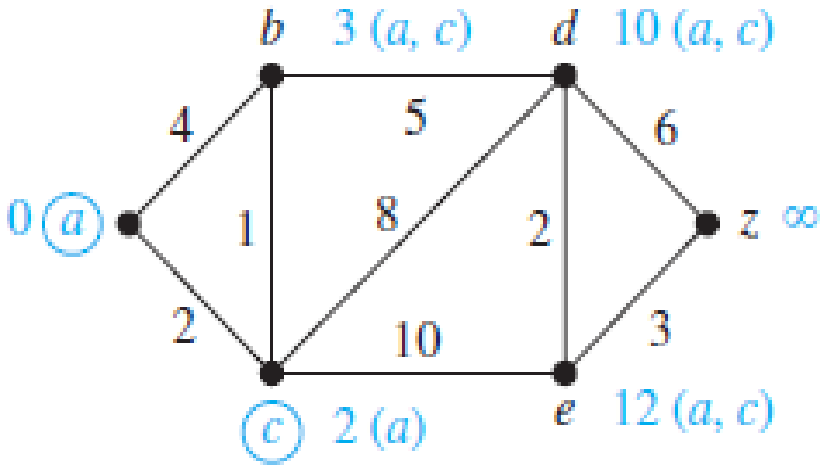
1. $u := a$ (because $a \notin S_0$ and $L_0(a)$ is minimal).
2. $S_1 = S_0 \cup \{a\} = \emptyset \cup \{a\} = \{a\}$.
3. $L_1(b) = \min\{L_0(b), L_0(a) + w(a,b)\} = \min\{\infty, 0+4\} = 4$. Path: $\langle a, b\rangle$.
   $L_1(c) = \min\{L_0(c), L_0(a) + w(a,c)\} = \min\{\infty, 0+2\} = 2$. Path: $\langle a, c\rangle$.
   $L_1(d) = L_1(e) = L_1(z) = \infty$ because $w(a,d) = w(a,e) = w(a,z) = \infty$.

First Iteration



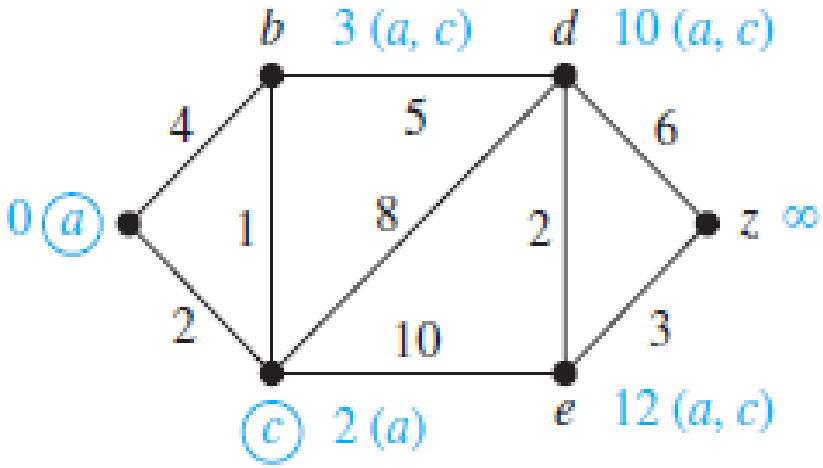
1. $u := a$ (because $a \notin S_0$ and $L_0(a)$ is minimal).
2. $S_1 = S_0 \cup \{a\} = \emptyset \cup \{a\} = \{a\}$.
3. $L_1(b) = \min\{L_0(b), L_0(a) + w(a,b)\} = \min\{\infty, 0+4\} = 4$. Path: $\langle a, b\rangle$.
   $L_1(c) = \min\{L_0(c), L_0(a) + w(a,c)\} = \min\{\infty, 0+2\} = 2$. Path: $\langle a, c\rangle$.
   $L_1(d) = L_1(e) = L_1(z) = \infty$ because $w(a,d) = w(a,e) = w(a,z) = \infty$.
4. Because $z \notin S_1$, the iteration is continued.

Second Iteration



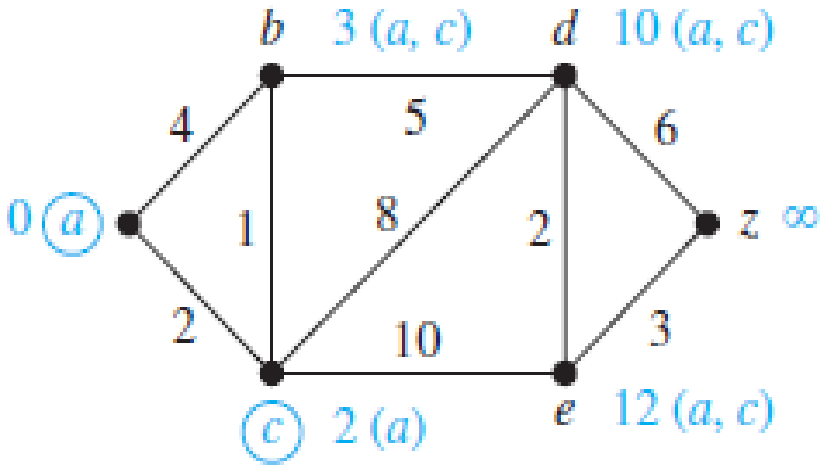
1. $u := c$ (because $c \notin S_1$ and $L_1(c)$ is minimal).

Second Iteration



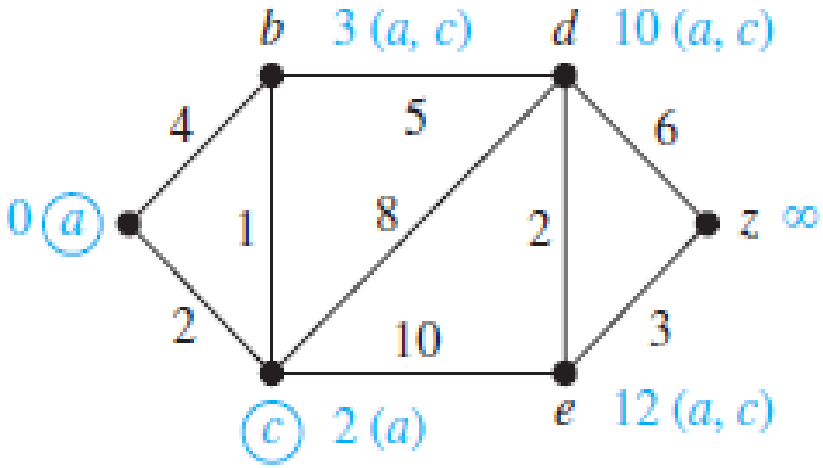
1. $u := c$ (because $c \notin S_1$ and $L_1(c)$ is minimal).
2. $S_2 = S_1 \cup \{c\} = \{a\} \cup \{c\} = \{a, c\}$.
3. $L_2(b) = \min\{L_1(b), L_1(c) + w(c, b)\} = \min\{4, 2 + 1\} =$

Second Iteration



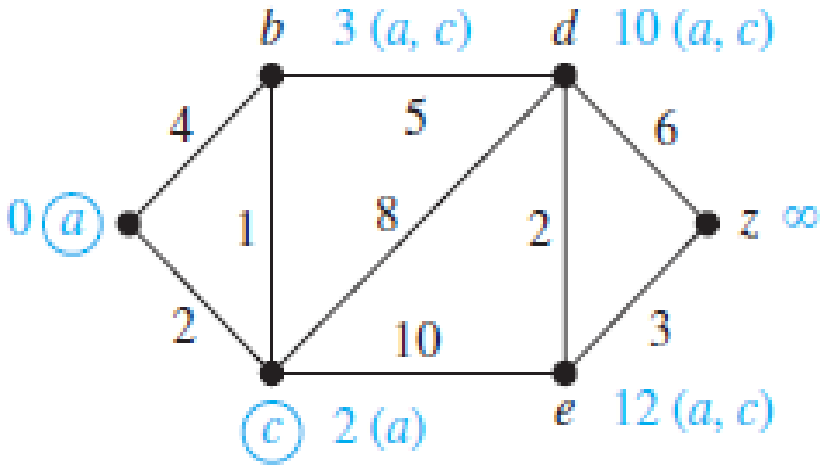
1. $u := c$ (because $c \notin S_1$ and $L_1(c)$ is minimal).
2. $S_2 = S_1 \cup \{c\} = \{a\} \cup \{c\} = \{a, c\}$.
3. $L_2(b) = \min\{L_1(b), L_1(c) + w(c, b)\} = \min\{4, 2 + 1\} = 3$. Path:

Second Iteration



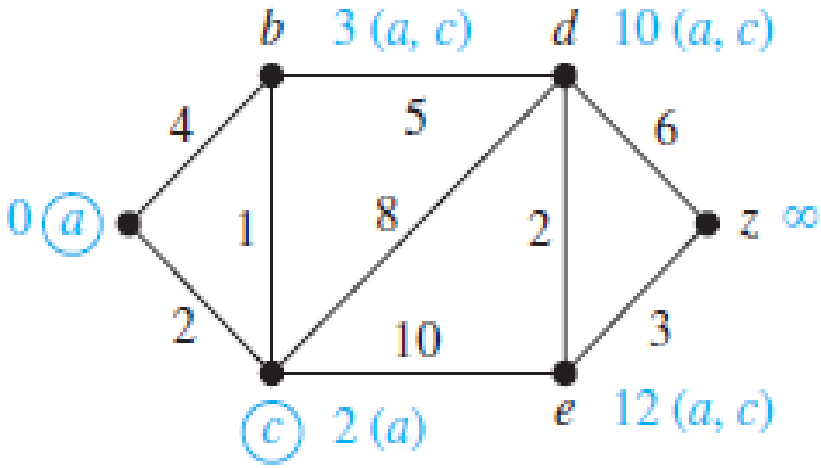
1. $u := c$ (because $c \notin S_1$ and $L_1(c)$ is minimal).
2. $S_2 = S_1 \cup \{c\} = \{a\} \cup \{c\} = \{a, c\}$.
3. $L_2(b) = \min\{L_1(b), L_1(c) + w(c, b)\} = \min\{4, 2 + 1\} = 3$. Path: $\langle a, c, b \rangle$.
   $L_2(d) = \min\{L_1(d), L_1(c) + w(c, d)\} = \min\{\infty, 2 + 8\} =$

Second Iteration



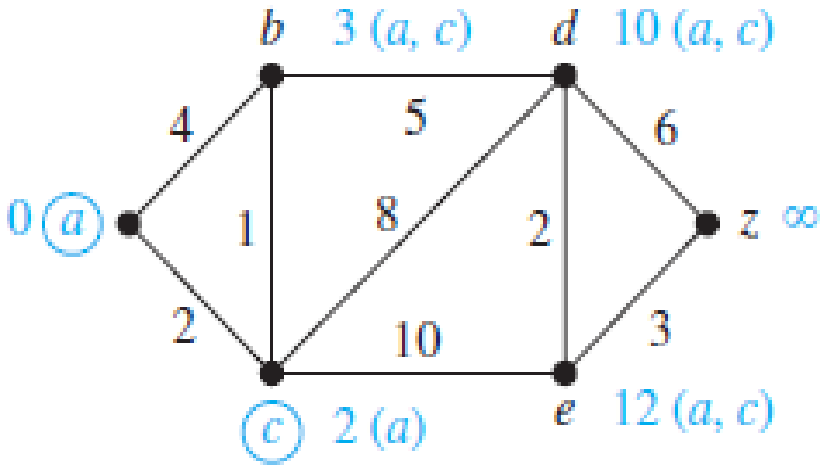
1. $u := c$ (because $c \notin S_1$ and $L_1(c)$ is minimal).
2. $S_2 = S_1 \cup \{c\} = \{a\} \cup \{c\} = \{a, c\}$.
3. $L_2(b) = \min\{L_1(b), L_1(c) + w(c, b)\} = \min\{4, 2 + 1\} = 3$. Path: $\langle a, c, b \rangle$.
   $L_2(d) = \min\{L_1(d), L_1(c) + w(c, d)\} = \min\{\infty, 2 + 8\} = 10$. Path:

Second Iteration



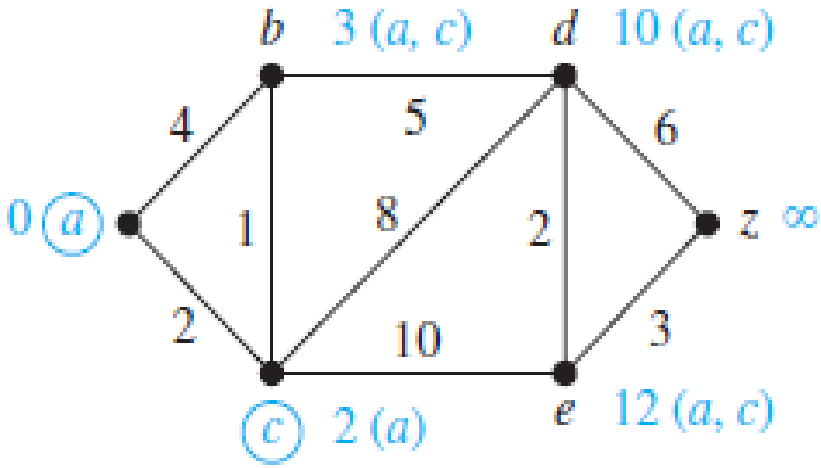
1. $u := c$ (because $c \notin S_1$ and $L_1(c)$ is minimal).
2. $S_2 = S_1 \cup \{c\} = \{a\} \cup \{c\} = \{a, c\}$.
3. $L_2(b) = \min\{L_1(b), L_1(c) + w(c, b)\} = \min\{4, 2 + 1\} = 3$. Path: $\langle a, c, b\rangle$.
   $L_2(d) = \min\{L_1(d), L_1(c) + w(c, d)\} = \min\{\infty, 2 + 8\} = 10$. Path: $\langle a, c, d\rangle$.
   $L_2(e) = \min\{L_1(e), L_1(c) + w(c, e)\} = \min\{\infty, 2 + 10\} =$

Second Iteration



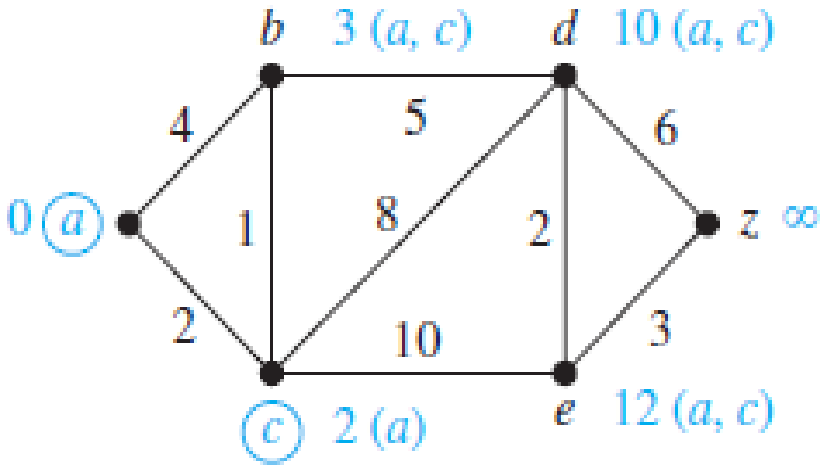
1. $u := c$ (because $c \notin S_1$ and $L_1(c)$ is minimal).
2. $S_2 = S_1 \cup \{c\} = \{a\} \cup \{c\} = \{a, c\}$.
3. $L_2(b) = \min\{L_1(b), L_1(c) + w(c, b)\} = \min\{4, 2 + 1\} = 3$. Path: $\langle a, c, b\rangle$.
   $L_2(d) = \min\{L_1(d), L_1(c) + w(c, d)\} = \min\{\infty, 2 + 8\} = 10$. Path: $\langle a, c, d\rangle$.
   $L_2(e) = \min\{L_1(e), L_1(c) + w(c, e)\} = \min\{\infty, 2 + 10\} = 12$. Path:

Second Iteration



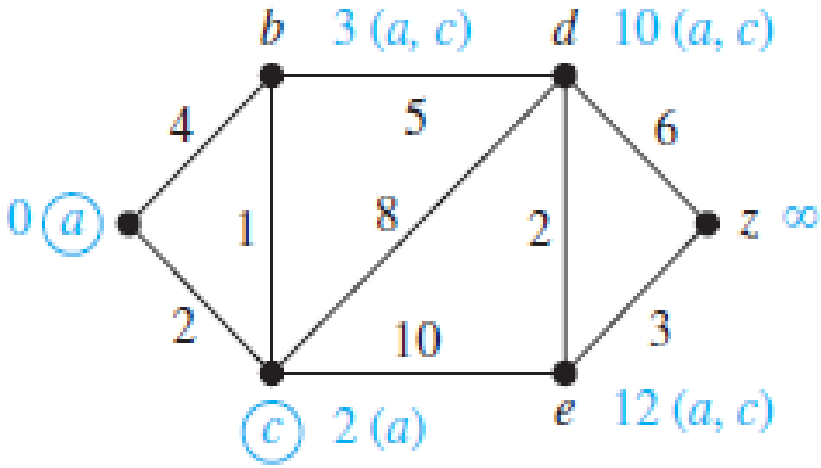
1. $u := c$ (because $c \notin S_1$ and $L_1(c)$ is minimal).
2. $S_2 = S_1 \cup \{c\} = \{a\} \cup \{c\} = \{a, c\}$.
3. $L_2(b) = \min\{L_1(b), L_1(c) + w(c, b)\} = \min\{4, 2 + 1\} = 3$. Path: $\langle a, c, b\rangle$.
   $L_2(d) = \min\{L_1(d), L_1(c) + w(c, d)\} = \min\{\infty, 2 + 8\} = 10$. Path: $\langle a, c, d\rangle$.
   $L_2(e) = \min\{L_1(e), L_1(c) + w(c, e)\} = \min\{\infty, 2 + 10\} = 12$. Path: $\langle a, c, e\rangle$.

Second Iteration



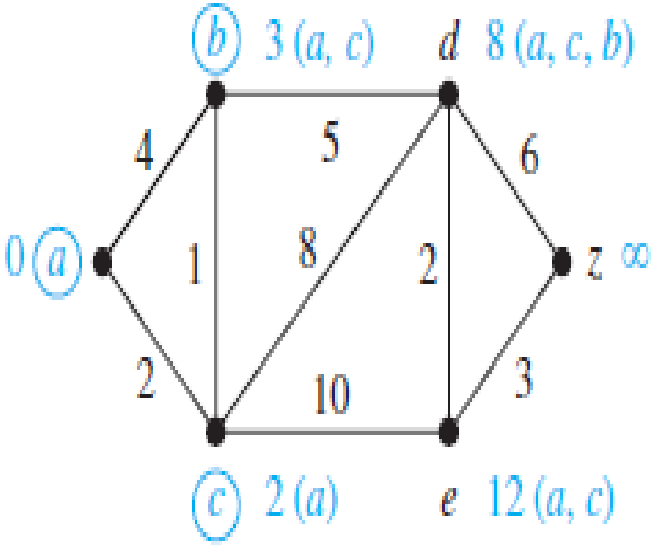
1. $u := c$ (because $c \notin S_1$ and $L_1(c)$ is minimal).
2. $S_2 = S_1 \cup \{c\} = \{a\} \cup \{c\} = \{a, c\}$.
3. $L_2(b) = \min\{L_1(b), L_1(c) + w(c, b)\} = \min\{4, 2 + 1\} = 3$. Path: $\langle a, c, b\rangle$.
   $L_2(d) = \min\{L_1(d), L_1(c) + w(c, d)\} = \min\{\infty, 2 + 8\} = 10$. Path: $\langle a, c, d\rangle$.
   $L_2(e) = \min\{L_1(e), L_1(c) + w(c, e)\} = \min\{\infty, 2 + 10\} = 12$. Path: $\langle a, c, e\rangle$.
   $L_2(z) = \min\{L_1(z), L_1(c) + w(c, z)\} = \infty$, because $\{c, z\} \notin E$.

Second Iteration



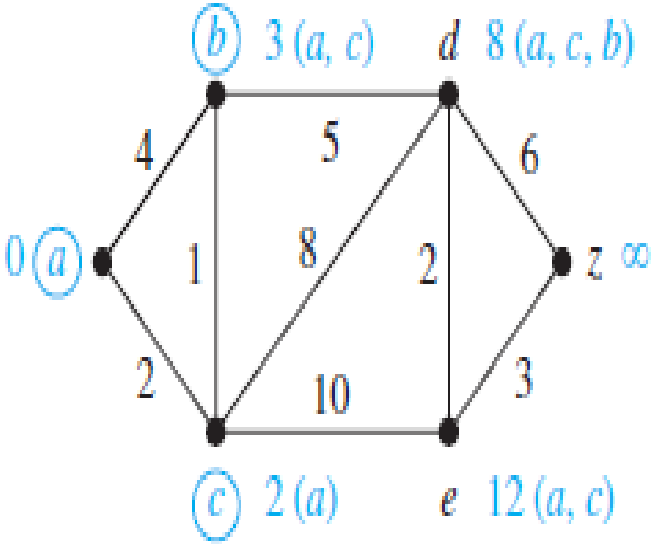
1. $u := c$ (because $c \notin S_1$ and $L_1(c)$ is minimal).
2. $S_2 = S_1 \cup \{c\} = \{a\} \cup \{c\} = \{a, c\}$.
3. $L_2(b) = \min\{L_1(b), L_1(c) + w(c, b)\} = \min\{4, 2 + 1\} = 3$. Path: $\langle a, c, b\rangle$.
   $L_2(d) = \min\{L_1(d), L_1(c) + w(c, d)\} = \min\{\infty, 2 + 8\} = 10$. Path: $\langle a, c, d\rangle$.
   $L_2(e) = \min\{L_1(e), L_1(c) + w(c, e)\} = \min\{\infty, 2 + 10\} = 12$. Path: $\langle a, c, e\rangle$.
   $L_2(z) = \min\{L_1(z), L_1(c) + w(c, z)\} = \infty$, because $\{c, z\} \notin E$.
4. Because $z \notin S_2$, the iteration is continued.

Third Iteration



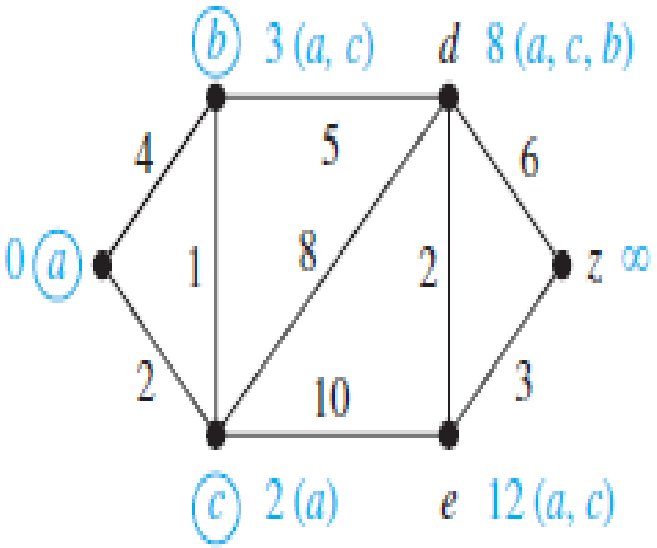
1. $u := b$ (because $b \notin S_2$ and $L_2(b)$ is minimal).

Third Iteration



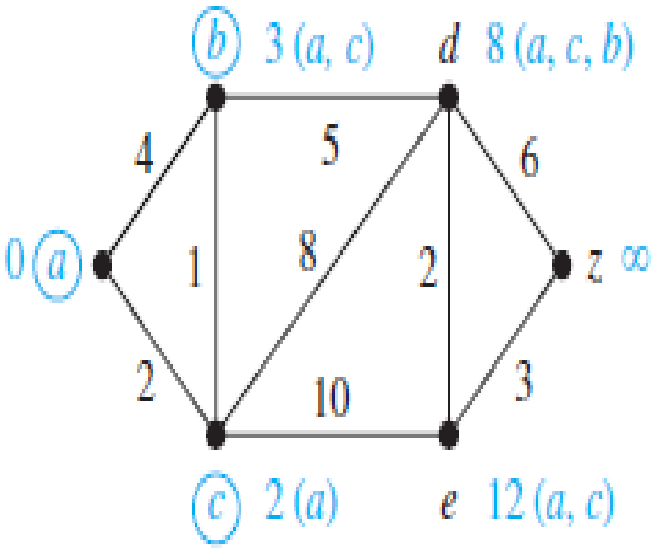
1. $u := b$ (because $b \notin S_2$ and $L_2(b)$ is minimal).
2. $S_3 = S_2 \cup \{b\} = \{a, c\} \cup \{b\} = \{a, c, b\}$.
3. $L_3(d) = \min\{L_2(d), L_2(b) + w(b, d)\} = \min\{10, 3 + 5\} =$

Third Iteration



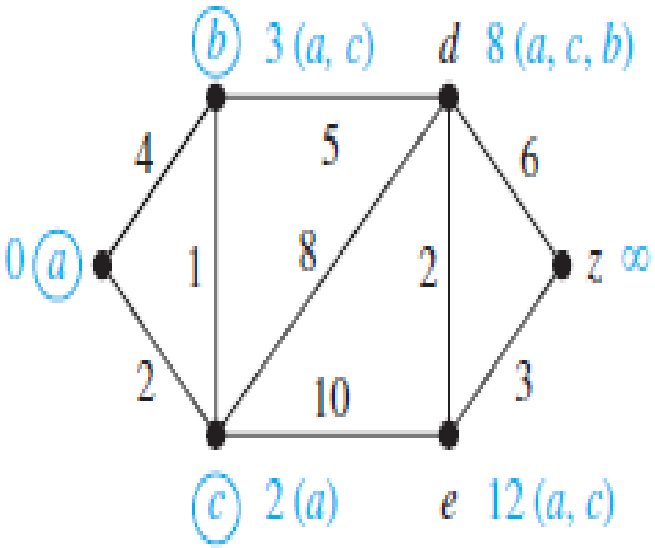
1. $u := b$ (because $b \notin S_2$ and $L_2(b)$ is minimal).
2. $S_3 = S_2 \cup \{b\} = \{a, c\} \cup \{b\} = \{a, c, b\}$.
3. $L_3(d) = \min\{L_2(d), L_2(b) + w(b, d)\} = \min\{10, 3 + 5\} = 8$. Path:

Third Iteration



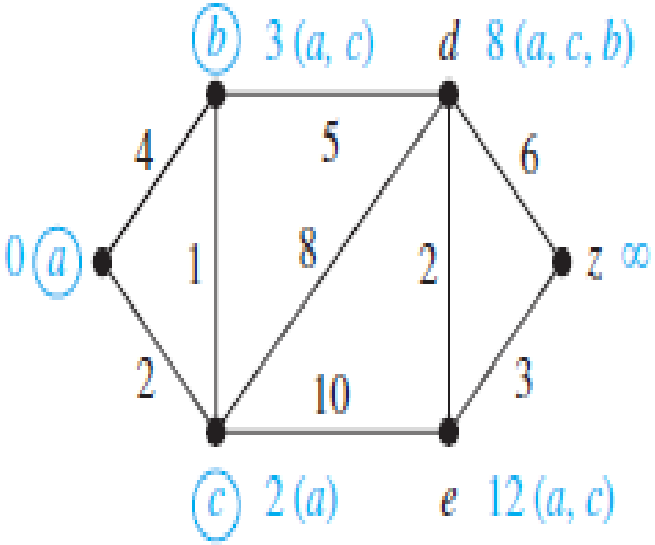
1. $u := b$ (because $b \notin S_2$ and $L_2(b)$ is minimal).
2. $S_3 = S_2 \cup \{b\} = \{a, c\} \cup \{b\} = \{a, c, b\}$.
3. $L_3(d) = \min\{L_2(d), L_2(b) + w(b, d)\} = \min\{10, 3 + 5\} = 8$. Path: $\langle a, c, b, d \rangle$.
   $L_3(e) = \min\{L_2(e), L_2(b) + w(b, e)\} = \min\{12, 3 + \infty\} =$

Third Iteration



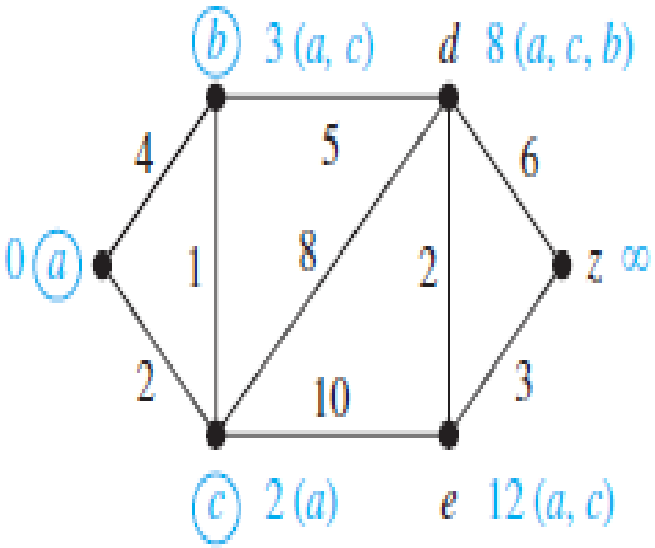
1. $u := b$ (because $b \notin S_2$ and $L_2(b)$ is minimal).
2. $S_3 = S_2 \cup \{b\} = \{a, c\} \cup \{b\} = \{a, c, b\}$.
3. $L_3(d) = \min\{L_2(d), L_2(b) + w(b, d)\} = \min\{10, 3 + 5\} = 8$. Path: $\langle a, c, b, d \rangle$.
   $L_3(e) = \min\{L_2(e), L_2(b) + w(b, e)\} = \min\{12, 3 + \infty\} = 12$. Path:

Third Iteration



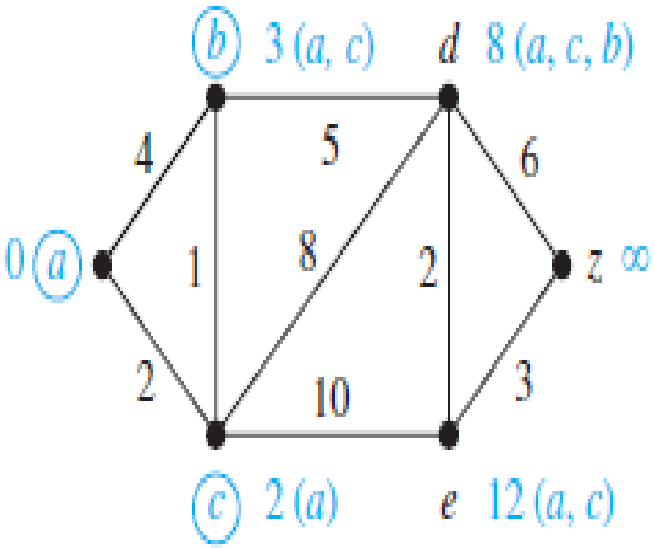
1. $u := b$ (because $b \notin S_2$ and $L_2(b)$ is minimal).
2. $S_3 = S_2 \cup \{b\} = \{a, c\} \cup \{b\} = \{a, c, b\}$.
3. $L_3(d) = \min\{L_2(d), L_2(b) + w(b, d)\} = \min\{10, 3 + 5\} = 8$. Path: $\langle a, c, b, d\rangle$.
   $L_3(e) = \min\{L_2(e), L_2(b) + w(b, e)\} = \min\{12, 3 + \infty\} = 12$. Path: $\langle a, c, e\rangle$.

Third Iteration



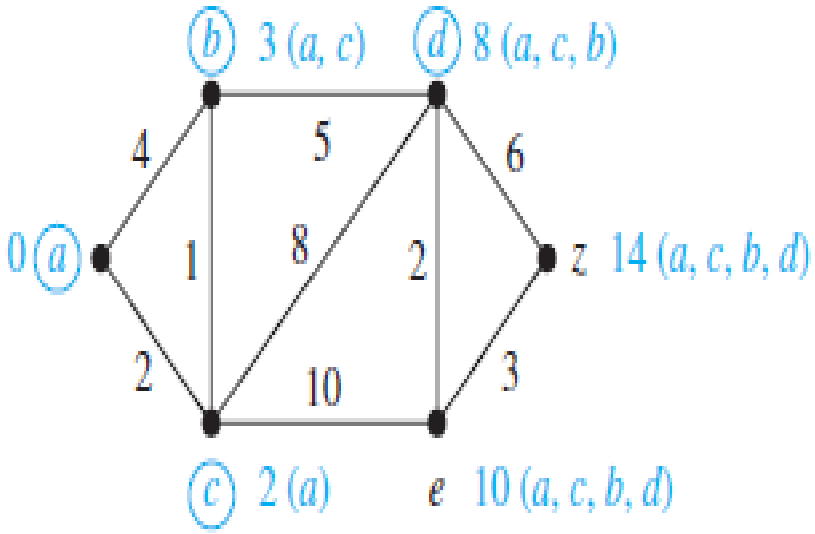
1. $u := b$ (because $b \notin S_2$ and $L_2(b)$ is minimal).
2. $S_3 = S_2 \cup \{b\} = \{a, c\} \cup \{b\} = \{a, c, b\}$.
3. $L_3(d) = \min\{L_2(d), L_2(b) + w(b, d)\} = \min\{10, 3 + 5\} = 8$. Path: $\langle a, c, b, d\rangle$.
   $L_3(e) = \min\{L_2(e), L_2(b) + w(b, e)\} = \min\{12, 3 + \infty\} = 12$. Path: $\langle a, c, e\rangle$.
   $L_3(z) = \min\{L_2(z), L_2(b) + w(b, z)\} = \infty$, because $\{b, z\} \notin E$.

Third Iteration



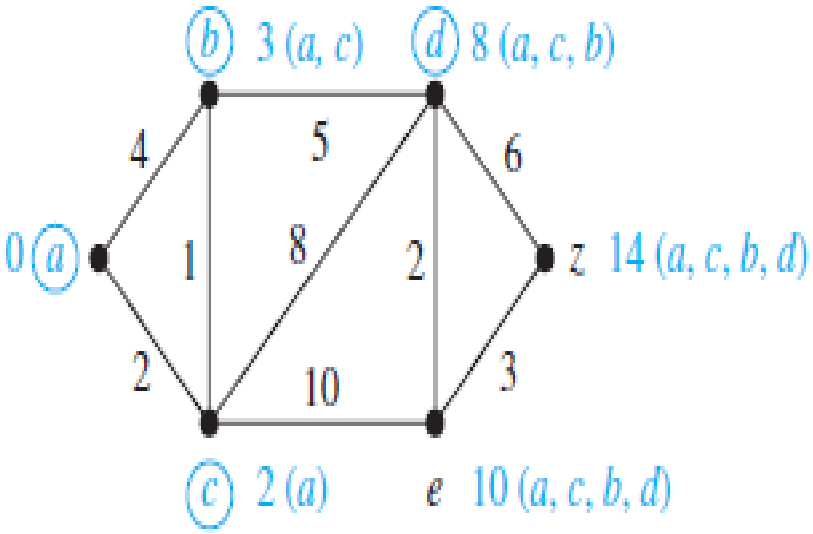
1. $u := b$ (because $b \notin S_2$ and $L_2(b)$ is minimal).
2. $S_3 = S_2 \cup \{b\} = \{a, c\} \cup \{b\} = \{a, c, b\}$.
3. $L_3(d) = \min\{L_2(d), L_2(b) + w(b, d)\} = \min\{10, 3 + 5\} = 8$. Path: $\langle a, c, b, d\rangle$.
   $L_3(e) = \min\{L_2(e), L_2(b) + w(b, e)\} = \min\{12, 3 + \infty\} = 12$. Path: $\langle a, c, e\rangle$.
   $L_3(z) = \min\{L_2(z), L_2(b) + w(b, z)\} = \infty$, because $\{b, z\} \notin E$.
4. Because $z \notin S_3$, the iteration is continued.

Fourth Iteration



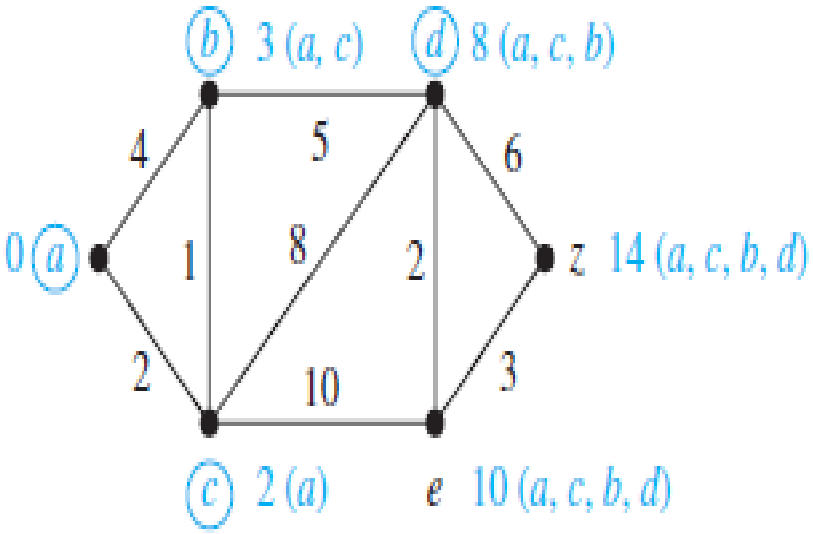
1. $u := d$ (because $d \notin S_3$ and $L_3(d)$ is minimal).

Fourth Iteration



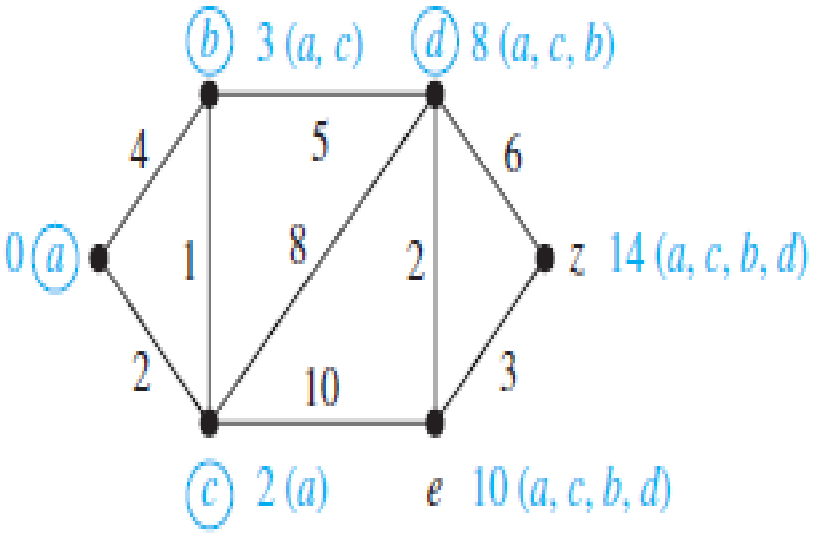
1. $u := d$ (because $d \notin S_3$ and $L_3(d)$ is minimal).
2. $S_4 = S_3 \cup \{d\} = \{a, c, b\} \cup \{d\} = \{a, c, b, d\}$.
3. $L_4(e) = \min\{L_3(e), L_3(d) + w(d, e)\} = \min\{12, 8 + 2\} =$

Fourth Iteration



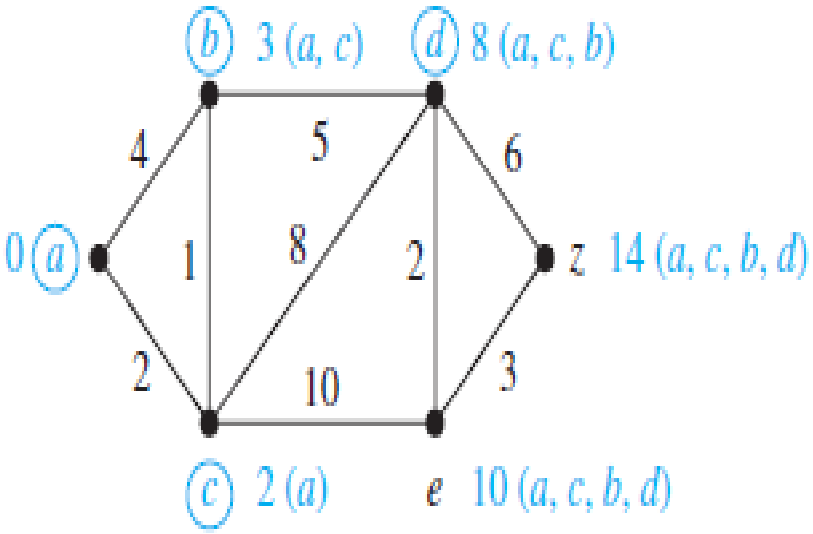
1. $u := d$ (because $d \notin S_3$ and $L_3(d)$ is minimal).
2. $S_4 = S_3 \cup \{d\} = \{a, c, b\} \cup \{d\} = \{a, c, b, d\}$.
3. $L_4(e) = \min\{L_3(e), L_3(d) + w(d, e)\} = \min\{12, 8 + 2\} = 10$. Path:

Fourth Iteration



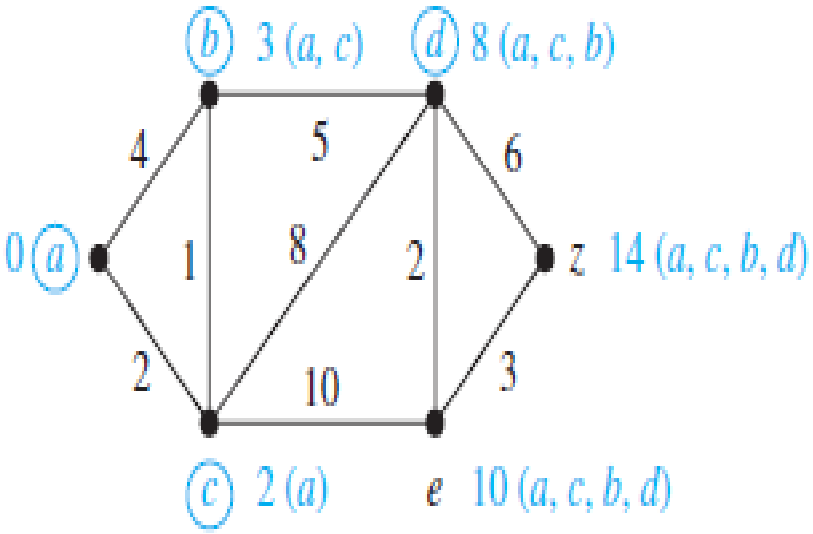
1. $u := d$ (because $d \notin S_3$ and $L_3(d)$ is minimal).
2. $S_4 = S_3 \cup \{d\} = \{a, c, b\} \cup \{d\} = \{a, c, b, d\}$.
3. $L_4(e) = \min\{L_3(e), L_3(d) + w(d, e)\} = \min\{12, 8 + 2\} = 10$. Path: $\langle a, c, b, d, e \rangle$.
   $L_4(z) = \min\{L_3(z), L_3(d) + w(d, z)\} = \min\{\infty, 8 + 6\} =$

Fourth Iteration



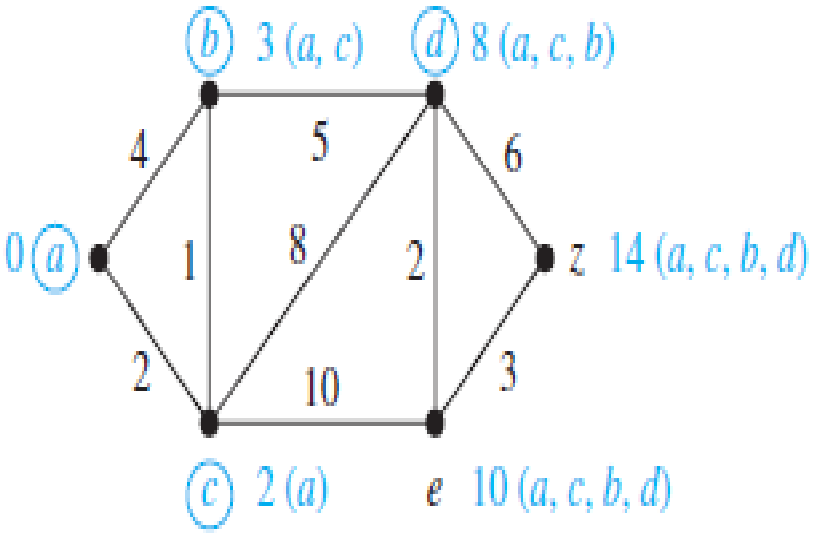
1. $u := d$ (because $d \notin S_3$ and $L_3(d)$ is minimal).
2. $S_4 = S_3 \cup \{d\} = \{a, c, b\} \cup \{d\} = \{a, c, b, d\}$.
3. $L_4(e) = \min\{L_3(e), L_3(d) + w(d, e)\} = \min\{12, 8 + 2\} = 10$. Path: $\langle a, c, b, d, e \rangle$.
   $L_4(z) = \min\{L_3(z), L_3(d) + w(d, z)\} = \min\{\infty, 8 + 6\} = 14$. Path:

Fourth Iteration



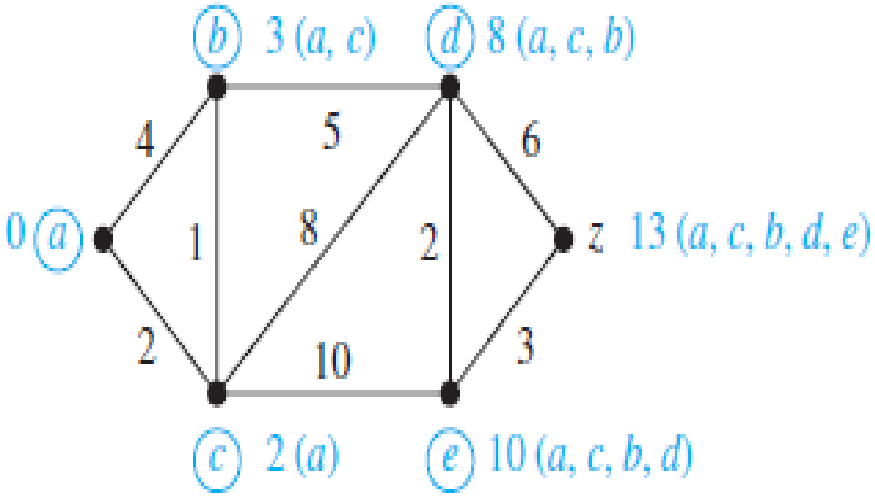
1. $u := d$ (because $d \notin S_3$ and $L_3(d)$ is minimal).
2. $S_4 = S_3 \cup \{d\} = \{a, c, b\} \cup \{d\} = \{a, c, b, d\}$.
3. $L_4(e) = \min\{L_3(e), L_3(d) + w(d, e)\} = \min\{12, 8 + 2\} = 10$. Path: $\langle a, c, b, d, e\rangle$.
   $L_4(z) = \min\{L_3(z), L_3(d) + w(d, z)\} = \min\{\infty, 8 + 6\} = 14$. Path: $\langle a, c, b, d, z\rangle$.

Fourth Iteration



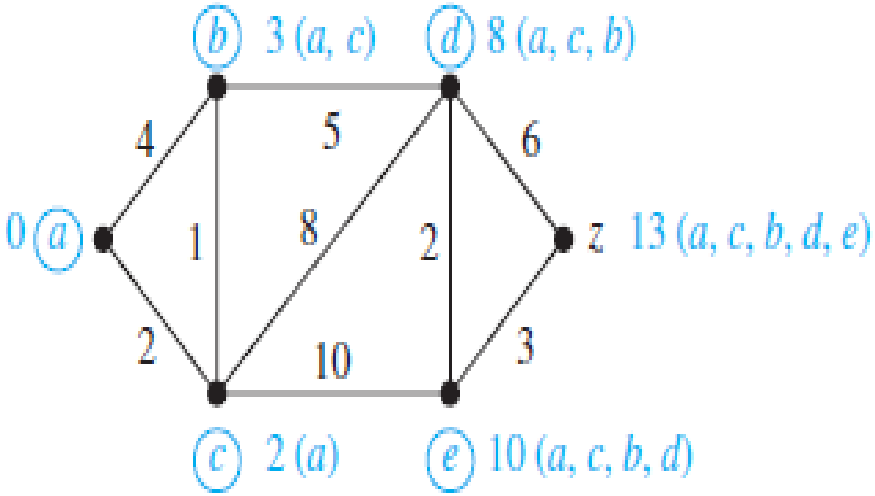
1. $u := d$ (because $d \notin S_3$ and $L_3(d)$ is minimal).
2. $S_4 = S_3 \cup \{d\} = \{a, c, b\} \cup \{d\} = \{a, c, b, d\}$.
3. $L_4(e) = \min\{L_3(e), L_3(d) + w(d, e)\} = \min\{12, 8 + 2\} = 10$. Path: $\langle a, c, b, d, e\rangle$.
   $L_4(z) = \min\{L_3(z), L_3(d) + w(d, z)\} = \min\{\infty, 8 + 6\} = 14$. Path: $\langle a, c, b, d, z\rangle$.
4. Because $z \notin S_4$, the iteration is continued.

Fifth Iteration



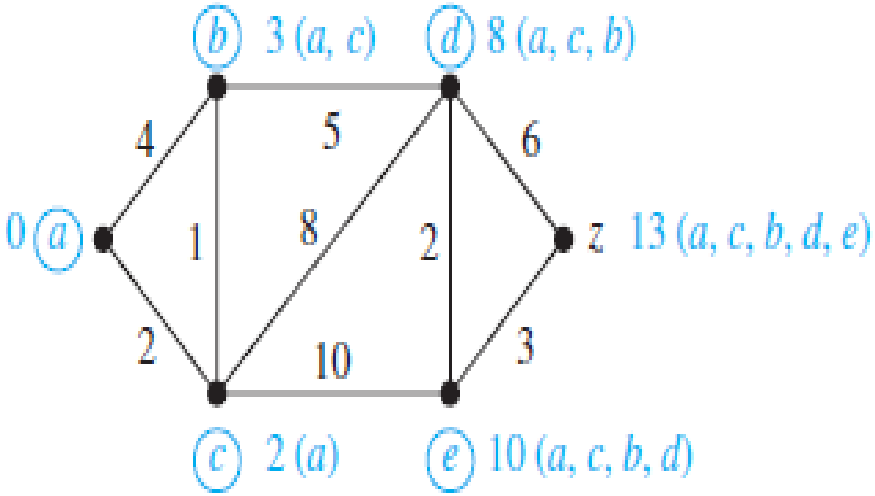
1. $u := e$ (because $e \notin S_4$ and $L_4(e)$ is minimal).

Fifth Iteration



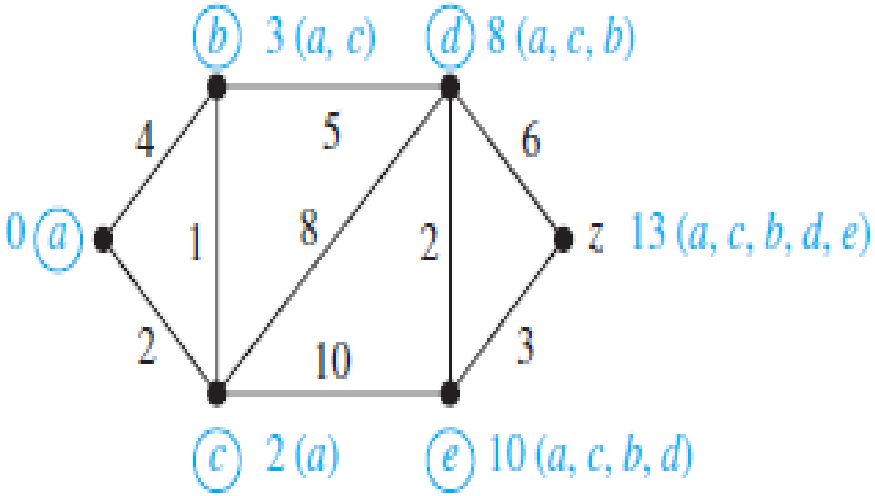
1. $u := e$ (because $e \notin S_4$ and $L_4(e)$ is minimal).
2. $S_5 = S_4 \cup \{e\} = \{a, c, b, d\} \cup \{e\} = \{a, c, b, d, e\}$.
3. $L_5(z) = \min\{L_4(z), L_4(e) + w(e, z)\} = \min\{14, 10 + 3\} =$

Fifth Iteration



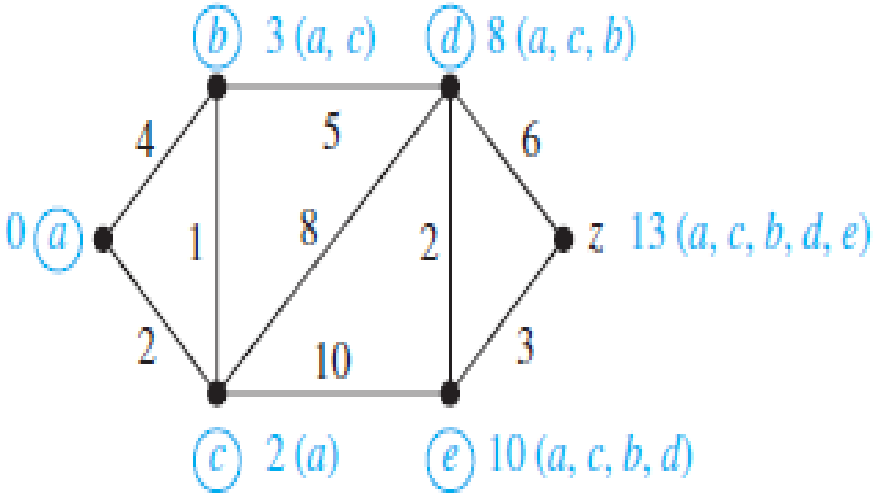
1. $u := e$ (because $e \notin S_4$ and $L_4(e)$ is minimal).
2. $S_5 = S_4 \cup \{e\} = \{a, c, b, d\} \cup \{e\} = \{a, c, b, d, e\}$.
3. $L_5(z) = \min\{L_4(z), L_4(e) + w(e, z)\} = \min\{14, 10 + 3\} = 13$. Path:

Fifth Iteration



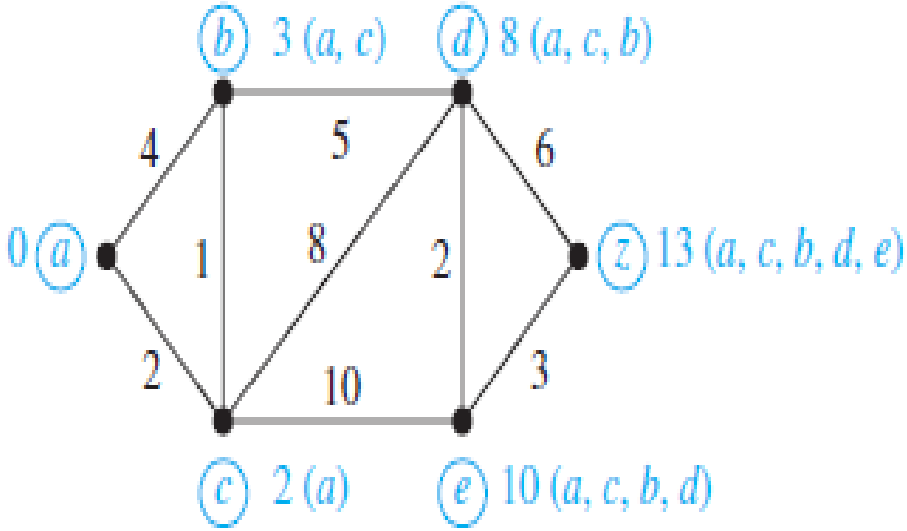
1. $u := e$ (because $e \notin S_4$ and $L_4(e)$ is minimal).
2. $S_5 = S_4 \cup \{e\} = \{a, c, b, d\} \cup \{e\} = \{a, c, b, d, e\}$.
3. $L_5(z) = \min\{L_4(z), L_4(e) + w(e, z)\} = \min\{14, 10 + 3\} = 13$. Path: $\langle a, c, b, d, e, z \rangle$.

Fifth Iteration



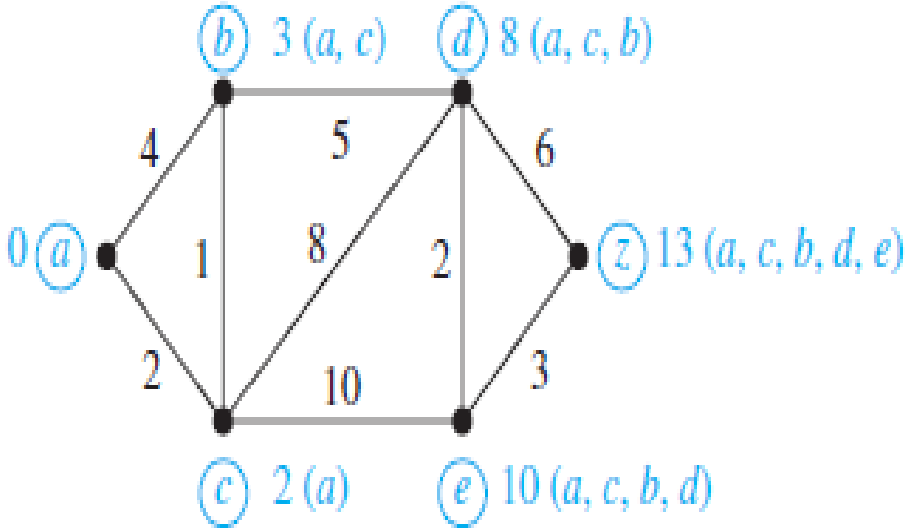
1. $u := e$ (because $e \notin S_4$ and $L_4(e)$ is minimal).
2. $S_5 = S_4 \cup \{e\} = \{a, c, b, d\} \cup \{e\} = \{a, c, b, d, e\}$.
3. $L_5(z) = \min\{L_4(z), L_4(e) + w(e, z)\} = \min\{14, 10 + 3\} = 13$. Path: $\langle a, c, b, d, e, z \rangle$.
4. Because $z \notin S_5$, the iteration is continued.

Sixth Iteration



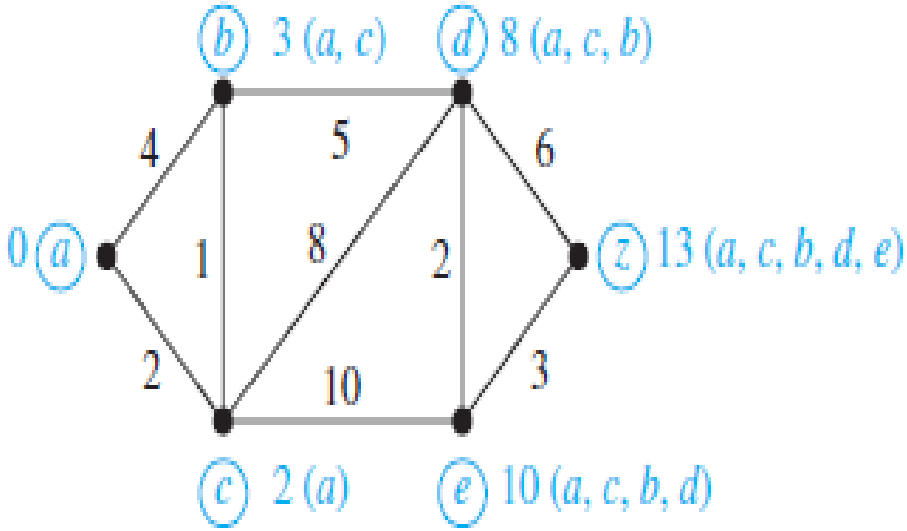
1. $u := z$ (because $z \notin S_5$ and $L_5(z)$ is minimal).

Sixth Iteration



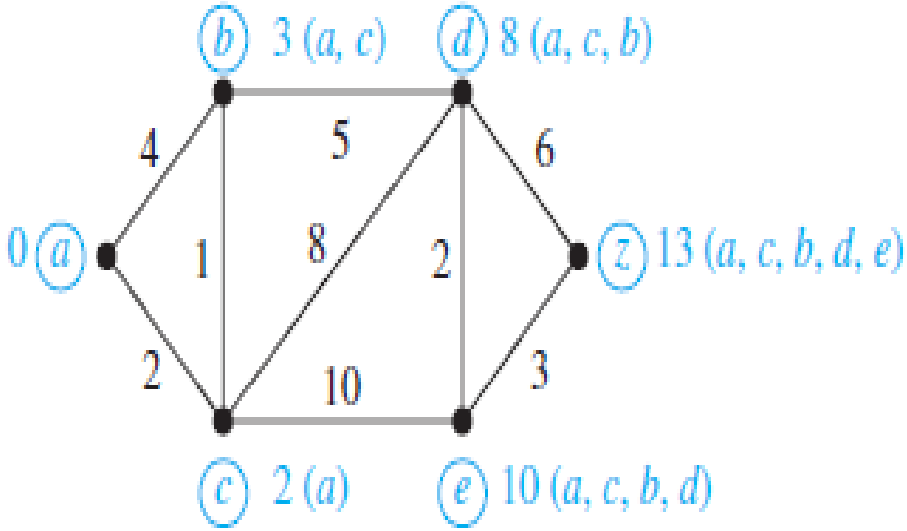
1. $u := z$ (because $z \notin S_5$ and $L_5(z)$ is minimal).
2. $S_6 = S_5 \cup \{z\} = \{a, c, b, d, e\} \cup \{z\} = \{a, c, b, d, e, z\}$.

Sixth Iteration



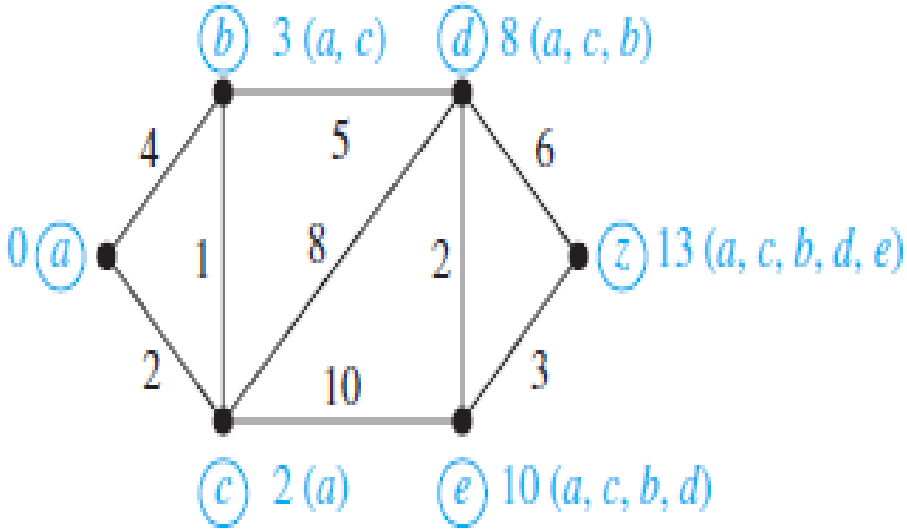
1. $u := z$ (because $z \notin S_5$ and $L_5(z)$ is minimal).
2. $S_6 = S_5 \cup \{z\} = \{a, c, b, d, e\} \cup \{z\} = \{a, c, b, d, e, z\}$.
3. There is no $v \in V$ with $v \notin S$. Therefore, `for all` $v \notin S$ loop in line 5, 6, and 7 of iteration process in Dijkstra's Algorithm cannot be executed.

Sixth Iteration



1. $u := z$ (because $z \notin S_5$ and $L_5(z)$ is minimal).
2. $S_6 = S_5 \cup \{z\} = \{a, c, b, d, e\} \cup \{z\} = \{a, c, b, d, e, z\}$.
3. There is no $v \in V$ with $v \notin S$. Therefore, `for all` $v \notin S$ loop in line 5, 6, and 7 of iteration process in Dijkstra's Algorithm cannot be executed.
4. Because $z \in S_6$ then `while` loop of Dijkstra's algorithm ends.

Sixth Iteration



1. $u := z$ (because $z \notin S_5$ and $L_5(z)$ is minimal).
2. $S_6 = S_5 \cup \{z\} = \{a, c, b, d, e\} \cup \{z\} = \{a, c, b, d, e, z\}$.
3. There is no $v \in V$ with $v \notin S$. Therefore, `for all` $v \notin S$ loop in line 5, 6, and 7 of iteration process in Dijkstra's Algorithm cannot be executed.
4. Because $z \in S_6$ then `while` loop of Dijkstra's algorithm ends.

Dijkstra's algorithm give the path $\langle a, c, b, d, e, z \rangle$ as the shortest path from a to z of length (weight) 13.

Contents

Traveling Salesperson Problem

Traveling Salesperson Problem, TSP

Suppose $G = (V, E)$ is a simple weighted graph that model some cities and their distance. A traveling salesperson **must visit all the cities exactly once, except for the first and last city** (the first and the last city are identical). How do we determine the most efficient traveling for this salesperson?

Traveling Salesperson Problem

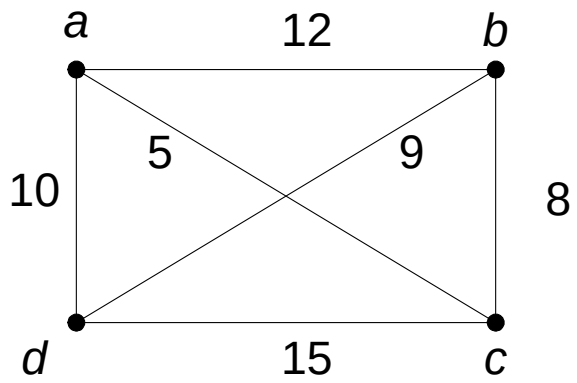
Traveling Salesperson Problem, TSP

Suppose $G = (V, E)$ is a simple weighted graph that model some cities and their distance. A traveling salesperson **must visit all the cities exactly once, except for the first and last city** (the first and the last city are identical). How do we determine the most efficient traveling for this salesperson?

Traveling salesperson problem is a problem of determining Hamilton circuit on a simple weighted graph with minimum length.

An Example of Traveling Salesperson Problem

We can determine the solution of Traveling Salesperson Problem if initial vertex and terminal vertex that is the vertex a of the following graph.

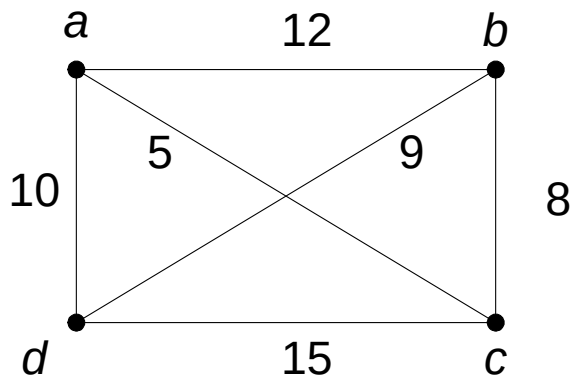


Because the above graph is a complete graph, then there are $\frac{(4-1)!}{2} = 3$ different Hamilton circuit. Notice that:

1. Circuit 1:

An Example of Traveling Salesperson Problem

We can determine the solution of Traveling Salesperson Problem if initial vertex and terminal vertex that is the vertex a of the following graph.

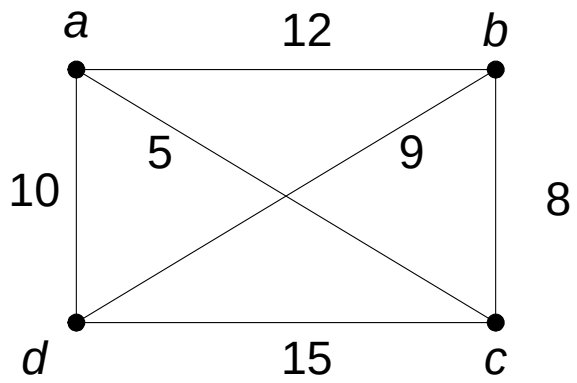


Because the above graph is a complete graph, then there are $\frac{(4-1)!}{2} = 3$ different Hamilton circuit. Notice that:

1. Circuit 1: $\langle a, b, c, d, a \rangle$ or $\langle a, d, c, b, a \rangle$ has length:

An Example of Traveling Salesperson Problem

We can determine the solution of Traveling Salesperson Problem if initial vertex and terminal vertex that is the vertex a of the following graph.

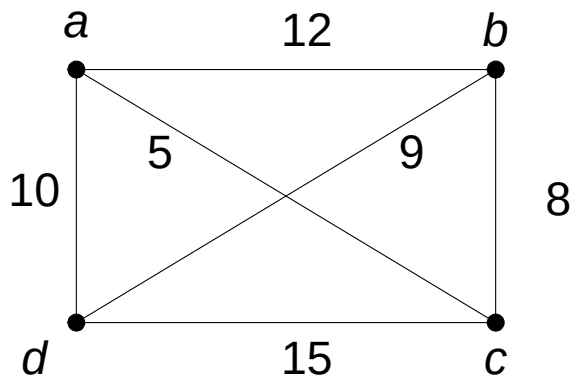


Because the above graph is a complete graph, then there are $\frac{(4-1)!}{2} = 3$ different Hamilton circuit. Notice that:

1. Circuit 1: $\langle a, b, c, d, a \rangle$ or $\langle a, d, c, b, a \rangle$ has length: $12 + 8 + 15 + 10 = 45$.
2. Circuit 2:

An Example of Traveling Salesperson Problem

We can determine the solution of Traveling Salesperson Problem if initial vertex and terminal vertex that is the vertex a of the following graph.

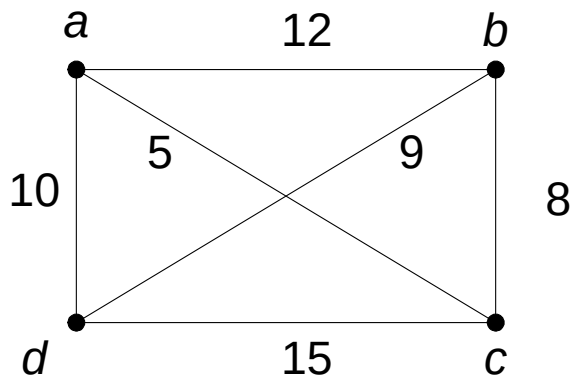


Because the above graph is a complete graph, then there are $\frac{(4-1)!}{2} = 3$ different Hamilton circuit. Notice that:

1. Circuit 1: $\langle a, b, c, d, a\rangle$ or $\langle a, d, c, b, a\rangle$ has length: $12 + 8 + 15 + 10 = 45$.
2. Circuit 2: $\langle a, c, d, b, a\rangle$ or $\langle a, b, d, c, a\rangle$ has length:

An Example of Traveling Salesperson Problem

We can determine the solution of Traveling Salesperson Problem if initial vertex and terminal vertex that is the vertex a of the following graph.

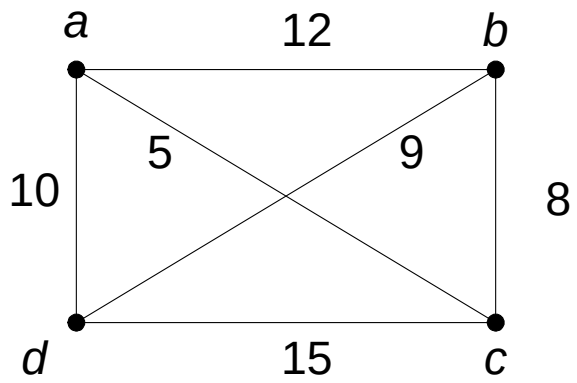


Because the above graph is a complete graph, then there are $\frac{(4-1)!}{2} = 3$ different Hamilton circuit. Notice that:

1. Circuit 1: $\langle a, b, c, d, a\rangle$ or $\langle a, d, c, b, a\rangle$ has length: $12 + 8 + 15 + 10 = 45$.
2. Circuit 2: $\langle a, c, d, b, a\rangle$ or $\langle a, b, d, c, a\rangle$ has length: $5 + 15 + 9 + 12 = 41$.
3. Circuit 3:

An Example of Traveling Salesperson Problem

We can determine the solution of Traveling Salesperson Problem if initial vertex and terminal vertex that is the vertex a of the following graph.

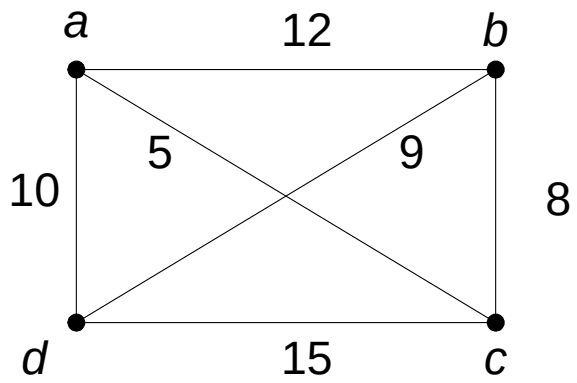


Because the above graph is a complete graph, then there are $\frac{(4-1)!}{2} = 3$ different Hamilton circuit. Notice that:

1. Circuit 1: $\langle a, b, c, d, a\rangle$ or $\langle a, d, c, b, a\rangle$ has length: $12 + 8 + 15 + 10 = 45$.
2. Circuit 2: $\langle a, c, d, b, a\rangle$ or $\langle a, b, d, c, a\rangle$ has length: $5 + 15 + 9 + 12 = 41$.
3. Circuit 3: $\langle a, c, b, d, a\rangle$ or $\langle a, d, b, c, a\rangle$ has length:

An Example of Traveling Salesperson Problem

We can determine the solution of Traveling Salesperson Problem if initial vertex and terminal vertex that is the vertex a of the following graph.



Because the above graph is a complete graph, then there are $\frac{(4-1)!}{2} = 3$ different Hamilton circuit. Notice that:

1. Circuit 1: $\langle a, b, c, d, a\rangle$ or $\langle a, d, c, b, a\rangle$ has length: $12 + 8 + 15 + 10 = 45$.
2. Circuit 2: $\langle a, c, d, b, a\rangle$ or $\langle a, b, d, c, a\rangle$ has length: $5 + 15 + 9 + 12 = 41$.
3. Circuit 3: $\langle a, c, b, d, a\rangle$ or $\langle a, d, b, c, a\rangle$ has length: $10 + 5 + 9 + 8 = 32$.

So the most efficient circuit is $\langle a, c, b, d, a\rangle$ of length 32.

Contents

Chinese Postman Problem

Chinese Postman Problem

In China, a postman must deliver all mails given to him. To save time and travel cost, he plans a traveling route that only traverse each road exactly once. If the initial place and the final/terminal place of the postman are identical, how do we determine the most efficient traveling route?

Chinese Postman Problem

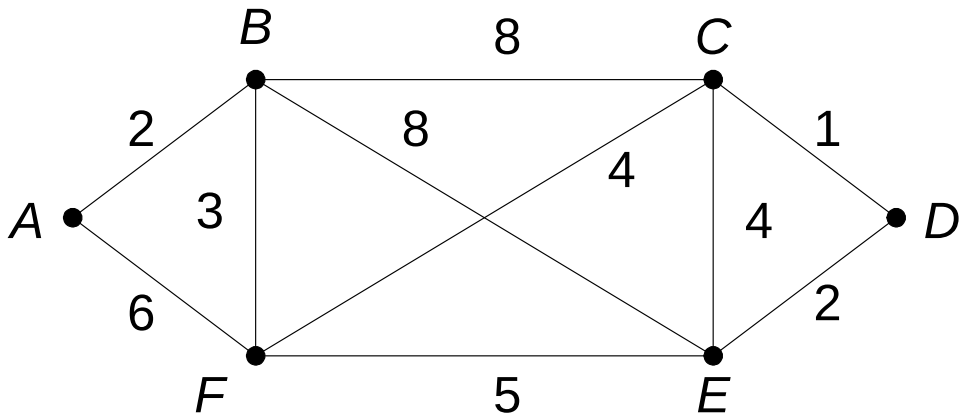
Chinese Postman Problem

In China, a postman must deliver all mails given to him. To save time and travel cost, he plans a traveling route that only traverse each road exactly once. If the initial place and the final/terminal place of the postman are identical, how do we determine the most efficient traveling route?

This problem was firstly delivered by Guan Meigu in 1962. Basically, the Chinese Postman Problem (CPP) is a problem of finding an Euler circuit on a graph.

An Example of Chinese Postman Problem

On the following graph, we will determine the initial vertex and terminal vertex for an Euler path.



One of them is: $\langle A, B, C, D, E, F, C, E, B, F, A \rangle$.