Shortest Paths:

The problem can be described in graph theoretic terms as the search for the shortest path on a weighted graph. Recall that a path is a sequence of vertices in which there is an edge between consecutive vertices and no vertex is repeated. As with the algorithms for the Traveling Salesman Problem, the weight associated with an edge may represent more than just distance (e.g., cost or time) and the shortest path really indicates the path of least total weight.

- We will only investigate how to find the shortest path since determining if the shortest path exists is quickly answered by simply knowing if the graph is connected.

Dijkstra's Algorithm:

Dijkstra's Algorithm is a bit more complex than the algorithms we have studied so far. Each vertex is given a two-part label $L(v) = (x, (w(v))$. The first portion of the label is the name of the vertex used to travel to v. The second part is the weight of the path that was used to get to v from the designated starting vertex. At each stage of the algorithm, we will consider a set of *free* vertices, denoted by an F below. Free vertices are the neighbors of previously visited vertices that are themselves not yet visited.

2.3.5. Algorithm. (Dijkstra's Algorithm—distances from one vertex.)
Input: A graph (or digraph) with nonnegative edge weights and a starting vertex u. The weight of edge xy is $w(xy)$; let $w(xy) = \infty$ if xy is not an edge.
Idea: Maintain the set S of vertices to which a shortest path from u is known, enlarging S to include all vertices. To do this, maintain a tentative distance $t(z)$ from u to each $z \notin S$, being the length of the shortest u,z-path yet found.
Initialization: Set $S = \{u\}$; $t(u) = 0$; $t(z) = w(uz)$ for $z \neq u$.
Iteration: Select a vertex v outside S such that $t(v) = \min_{z \notin S} t(z)$. Add v to S. Explore edges from v to update tentative distances: for each edge vz with $z \notin S$, update $t(z)$ to $\min\{t(z), t(v) + w(vz)\}$.

The iteration continues until $S = V(G)$ or until $t(z) = \infty$ for every $z \notin S$. At the end, set $d(u, v) = t(v)$ for all v. ■

Dijkstra's Algorithm

Input: Weighted connected simple graph $G = (V, E, w)$ and designated $Start$ vertex.

Steps:

1. For each vertex x of G, assign a label $L(x)$ so that $L(x) = (-, 0)$ if $x = Start$ and $L(x) = (-, \infty)$ otherwise. Highlight $Start$.

2. Let $u = Start$ and define F to be the neighbors of u. Update the labels for each vertex v in F as follows:

$$\text{if } w(u) + w(uv) < w(v), \text{ then redefine } L(v) = (u, w(u) + w(uv))$$

$$\text{otherwise do not change } L(v)$$

3. Highlight the vertex with lowest weight as well as the edge uv used to update the label. Redefine $u = v$.

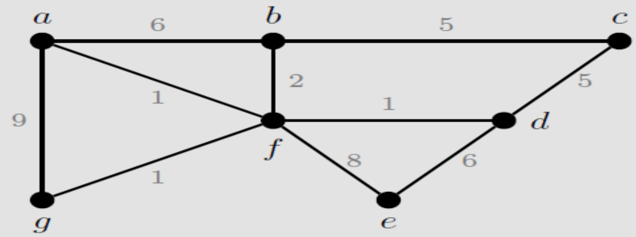
4. Repeat Steps (2) and (3) until each vertex has been reached. In all future iterations, F consists of the un-highlighted neighbors of all previously highlighted vertices and the labels are updated only for those vertices that are adjacent to the last vertex that was highlighted.

5. The shortest path from $Start$ to any other vertex is found by tracing back using the first component of the labels. The total weight of the path is the weight given in the second component of the ending vertex.

Output: Highlighted path from $Start$ to any vertex x of weight $w(x)$.

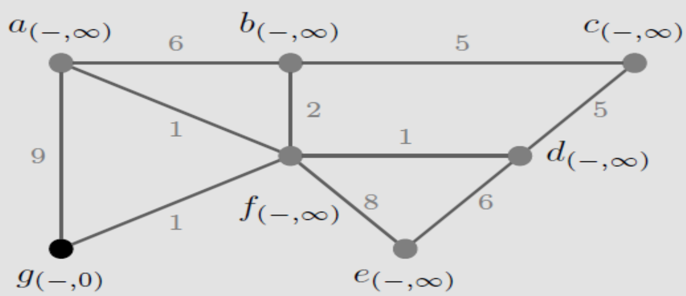
EXAMPLE:

Example 2.17 Apply Dijkstra's Algorithm to the graph below where $Start = g$.



Solution: In each step, the label of a vertex will be shown in the table on the right.

Step 1: Highlight g. Define $L(g) = (-, 0)$ and $L(x) = (-, \infty)$ for all $x = a, \cdots, f$.



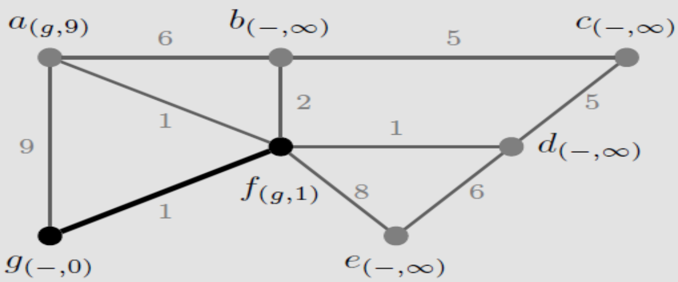
$F = \{\}$

a	$(-, \infty)$
b	$(-, \infty)$
c	$(-, \infty)$
d	$(-, \infty)$
e	$(-, \infty)$
f	$(-, \infty)$
g	$(-, 0)$

Step 2: Let $u = g$. Then the neighbors of g comprise $F = \{a, f\}$. We compute

$$w(g) + w(ga) = 0 + 9 = 9 < \infty = w(a)$$
$$w(g) + w(gf) = 0 + 1 = 1 < \infty = w(f)$$

Update $L(a) = (g, 9)$ and $L(f) = (g, 1)$. Since the minimum weight for all vertices in F is that of f, we highlight the edge gf and the vertex f.

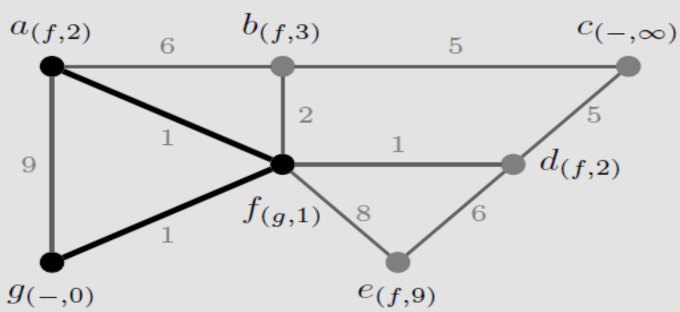


$$F = \{a, f\}$$
$$\begin{array}{c|l} a & (-,\infty) \to (g,9) \\ b & (-,\infty) \\ c & (-,\infty) \\ d & (-,\infty) \\ e & (-,\infty) \\ f & (-,\infty) \to (g,1) \\ g & (-,0) \end{array}$$

Step 3: Let $u = f$. Then the neighbors of all highlighted vertices are $F = \{a, b, d, e\}$. We compute

$$\begin{aligned} w(f) + w(fa) &= 1 + 1 = 2 < 9 = w(a) \\ w(f) + w(fb) &= 1 + 2 = 3 < \infty = w(b) \\ w(f) + w(fd) &= 1 + 1 = 2 < \infty = w(d) \\ w(f) + w(fe) &= 1 + 8 = 9 < \infty = w(e) \end{aligned}$$

Update $L(a) = (f, 2)$, $L(b) = (f, 3)$, $L(d) = (f, 2)$ and $L(e) = (f, 9)$. Since the minimum weight for all vertices in F is that of a or d, we choose to highlight the edge fa and the vertex a.

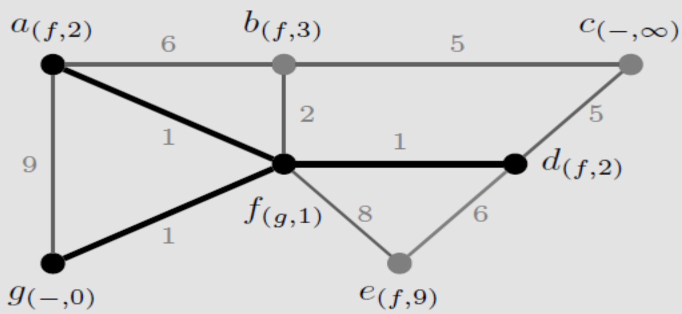


$$F = \{a, b, d, e\}$$
$$\begin{array}{c|l} a & (g,9) \to (f,2) \\ b & (-,\infty) \to (f,3) \\ c & (-,\infty) \\ d & (-,\infty) \to (f,2) \\ e & (-,\infty) \to (f,9) \\ f & (g,1) \\ g & (-,0) \end{array}$$

Step 4: Let $u = a$. Then the neighbors of all highlighted vertices are $F = \{b, d, e\}$. Note, we only consider updating the label for b since this is the only vertex adjacent to a, the vertex highlighted in the previous step.

$$w(a) + w(ba) = 2 + 6 = 8 \not< 2 = w(b)$$

We do not update the label for b since the computation above is not less than the current weight of b. The minimum weight for all vertices in F is that of d, and so we highlight the edge fd and the vertex d.



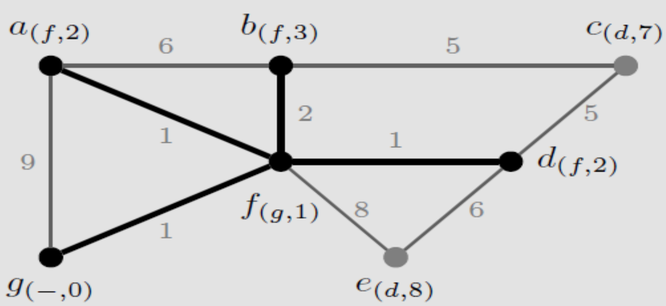
$$F = \{b, d, e\}$$

$$\begin{array}{c|l} a & (f,2) \\ b & (f,3) \\ c & (-,\infty) \\ d & (f,2) \\ e & (f,9) \\ f & (g,1) \\ g & (-,0) \end{array}$$

Step 5: Let $u = d$. Then the neighbors of all highlighted vertices are $F = \{b, c, e\}$. We compute

$$w(d) + w(dc) = 2 + 5 = 7 < \infty = w(c)$$
$$w(d) + w(de) = 2 + 6 = 8 < 9 = w(e)$$

Update $L(c) = (d, 7)$ and $L(e) = (d, 8)$. Since the minimum weight for all vertices in F is that of b, we highlight the edge bf and the vertex b.



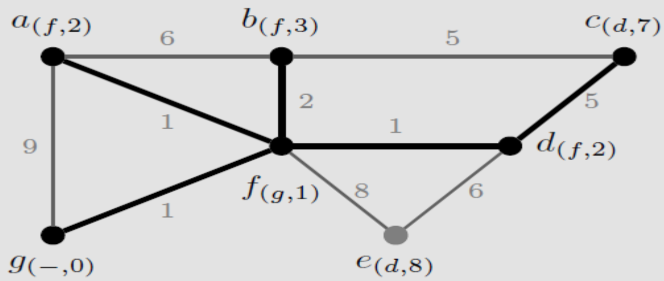
$$F = \{b, c, e\}$$

$$\begin{array}{c|l} a & (f,2) \\ b & (f,3) \\ c & (-,\infty) \to (d,7) \\ d & (f,2) \\ e & (f,9) \to (d,8) \\ f & (g,1) \\ g & (-,0) \end{array}$$

Step 6: Let $u = b$. Then the neighbors of all highlighted vertices are $F = \{c, e\}$. However, we only consider updating the label of c since e is not adjacent to b. Since

$$w(b) + w(bc) = 3 + 5 = 8 \not< 7 = w(c)$$

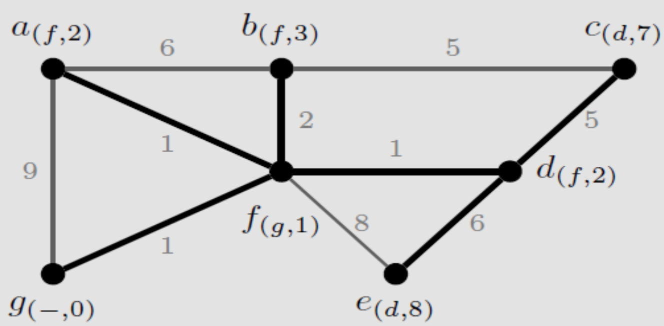
we do not update the labels of any vertices. Since the minimum weight for all vertices in F is that of c we highlight the edge dc and the vertex c. This terminates the iterations of the algorithm since our ending vertex has been reached.



$F = \{c, e\}$

a	$(f, 2)$
b	$(f, 3)$
c	$(d, 7)$
d	$(f, 2)$
e	$(d, 8)$
f	$(g, 1)$
g	$(-, 0)$

Step 7: Let $u = c$. Then the neighbors of all highlighted vertices are $F = \{e\}$. However, we do not need to update any labels since c and e are not adjacent. Thus we highlight the edge de and the vertex e. This terminates the iterations of the algorithm since all vertices are now highlighted.



$F = \{e\}$

a	$(f, 2)$
b	$(f, 3)$
c	$(d, 7)$
d	$(f, 2)$
e	$(d, 8)$
f	$(g, 1)$
g	$(-, 0)$

Output: The shortest paths from g to all other vertices can be found highlighted above. For example the shortest path from g to c is $g\,f\,d\,c$ and has a total weight 7, as shown by the label of c.

QUESTIONS:

4. Apply Dijkstra's algorithm to find the shortest paths (and the respective weights) from u_1 to all other vertices in the following weighted graph.

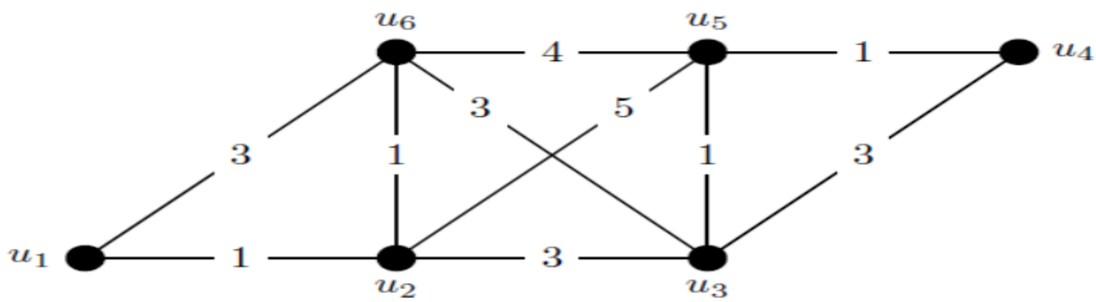


Fig. 3.6.10

5. Apply Dijkstra's algorithm to find the shortest paths (and the respective weights) from u_1 to all other vertices in the following weighted graph.

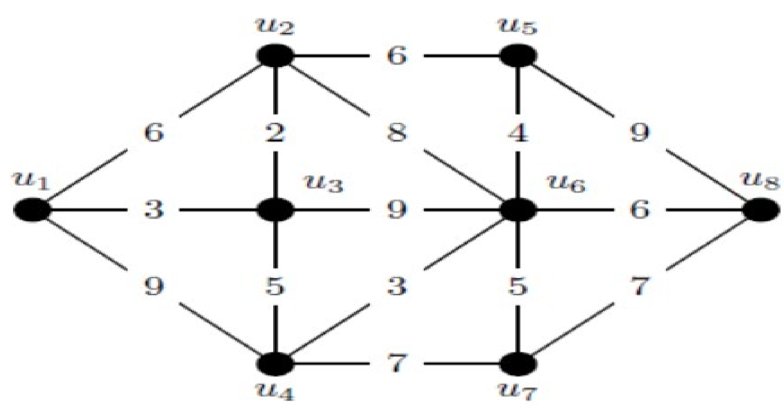


Fig. 3.6.11

6. Apply Dijkstra's algorithm to find the shortest paths (and the respective weights) from a to all other vertices in the following weighted graph.

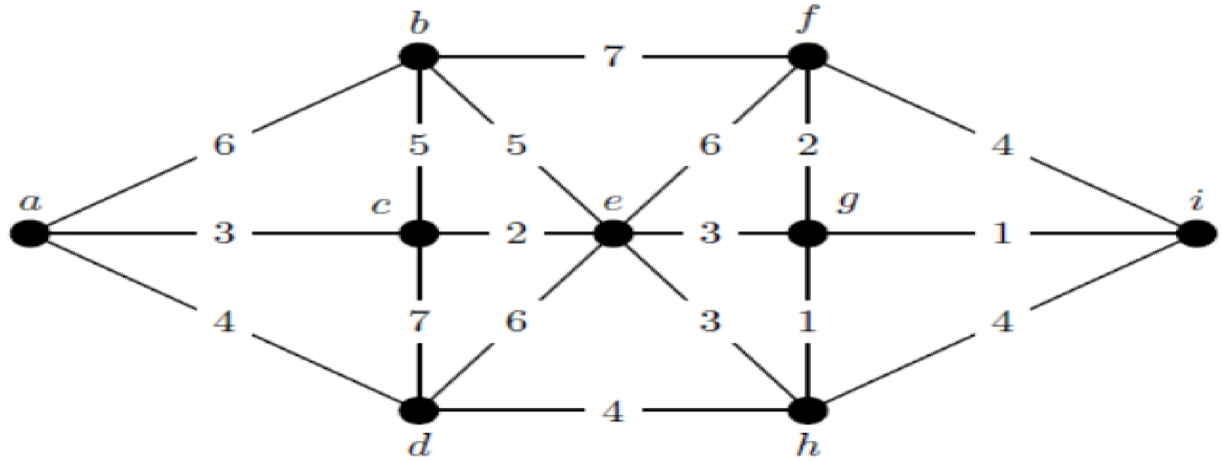


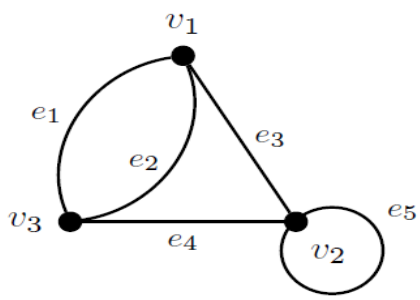
Fig. 3.6.12

Walks Using Matrices:

- We saw how to model a graph using an adjacency matrix.
- Matrix representations of graphs are useful when using a computer program to investigate certain features or processes on a graph.
- Another use for the adjacency matrix is to count the number of walks between two vertices within a graph.

Intuitive Idea:

Consider the graph shown below with its adjacency matrix A on the right.



$$\begin{bmatrix} 0 & 1 & 2 \\ 1 & 1 & 1 \\ 2 & 1 & 0 \end{bmatrix}$$

If we want a walk of length 1, we are in essence asking for an edge between two vertices. So to count the number of walks of length 1 from v_1 to v_3, we need only to count the number of edges (namely 2) between these vertices. What if we want the walks of length 2? By inspection, we can see there is only one, which is

$$v_1 \underset{e_3}{\rightarrow} v_2 \underset{e_4}{\rightarrow} v_3$$

Now consider the walks from v_1 to v_2. There is only one walk of length 1, and yet three of length 2:

$$v_1 \underset{e_3}{\rightarrow} v_2 \underset{e_5}{\rightarrow} v_2$$
$$v_1 \underset{e_1}{\rightarrow} v_3 \underset{e_4}{\rightarrow} v_2$$
$$v_1 \underset{e_2}{\rightarrow} v_3 \underset{e_4}{\rightarrow} v_2$$

- How could we count this?
- If we know how many walks there are from v1 to v2 (1) and then the number from v2 to itself (1), we can get one type of walk from v1 to v2.
- Also, we could count the number of walks from v1 to v3 (2) and then the number of walks from v3 to v2 (1).
- In total we have 1 *1 +2*1 =3 walks from v1 to v2.
- Note that we did not include any walks of the form v1v1v2 since there are no edges from v1 to itself.

Viewing this as a multiplication of vectors, we have

$$\begin{bmatrix} 0 & 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = 0 * 1 + 1 * 1 + 2 * 1 = 3$$

If we do this for the entire adjacency matrix, we have

$$A^2 = \begin{bmatrix} 5 & 3 & 1 \\ 3 & 3 & 3 \\ 1 & 3 & 5 \end{bmatrix}$$

Thus the entry a_{ij} in A^2 represents the number of walks between vertex v_i and v_j of length 2. If we multiplied this new matrix by A again, we would simply be counting the number of ways to get from v_i to v_j using 3 edges. The theorem below summarizes this for walks of any length n.

Theorem 2.24 Let G be a graph with adjacency matrix A. Then for any integer $n > 0$ the entry a_{ij} in A^n counts the number of walks from v_i to v_j.

Distance, Diameter, and Radius:

- Dijkstra's Algorithm provides the method for determining the shortest path between two points on a graph, which we define as the distance between those vertices.
- There are many theoretical implications for this distance. We will begin with defining the diameter and radius of a graph and the eccentricity of a vertex.

Definition 2.25 Given two vertices x, y in a graph G, define the ***distance*** $d(x, y)$ as the length of the shortest path from x to y. The ***eccentricity*** of a vertex x is the maximum distance from x to any other vertex in G; that is $\epsilon(x) = \max_{y \in V(G)} d(x, y)$.

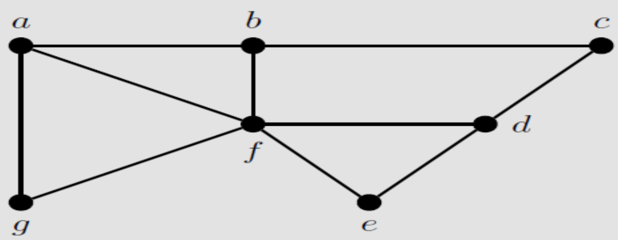
The ***diameter*** of G is the maximum eccentricity among all vertices, and so measures the maximum distance between any two vertices; that is $diam(G) = \max_{x,y \in V(G)} d(x, y)$. The ***radius*** of a graph is the minimum eccentricity among all vertices; that is $rad(G) = \min_{x \in V(G)} \epsilon(x)$.

Remarks:

- If a graph is connected, all of these parameters will be non-negative integers.

- What happens if the graph is disconnected? If x and y are in separate components of G then there is no shortest path between them and d(x; y) = ∞.
- This would then make diam(G) = rad(G) = ∞ since ε(v) = ∞ for all vertices in G.

Example 2.18 Find the diameter and radius for the graph below.



Solution: Note that f is adjacent to all vertices except c, but there is a path of length 2 from f to c. As no vertex is adjacent to all other vertices, we know the radius is 2. The longest path between two vertices is from g to c, and is of length 3, so the diameter is 3.

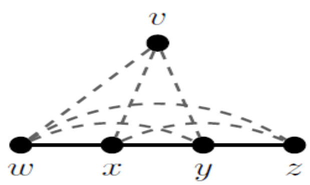
✓ While graphs can exist with arbitrarily large diameter and radius, there is a direct relationship between the diameter and radius of a graph and that of its complement.

Theorem 2.26 If G is disconnected then $\overline{G}$ is connected and $diam(\overline{G}) \leq 2$.

Proof: Assume G is disconnected. To prove $\overline{G}$ is connected, we must show there is an $x - y$ path for any pair of vertices x, y. If x and y are not adjacent in G then $xy \in E(\overline{G})$. Thus the edge xy is itself a $x - y$ path. Otherwise, $xy \in E(G)$ and so x and y are in the same component of G and not adjacent in $\overline{G}$. Since G is not connected, there must exist some vertex z in a different component from x and y, and so z cannot be adjacent to either of x or y. This implies $xz, yz \in E(\overline{G})$, and so xzy is a $x - y$ path in $\overline{G}$. Note that every pair of vertices in $\overline{G}$ fall into one of these two cases and so satisfy $d(x, y) \leq 2$. Thus $\overline{G}$ is connected and $diam(\overline{G}) \leq 2$.

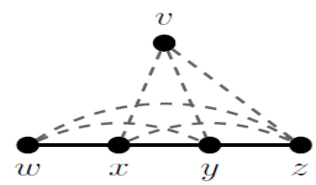
Theorem 2.27 For a simple graph G if $rad(G) \geq 3$ then $rad(\overline{G}) \leq 2$.

Proof: Since G is simple and $rad(G) = r \geq 3$, we know that $\epsilon(v) \geq r$ for all vertices in G. In particular, there exists some path $wxyz$ such that w is not adjacent to y and z, and x is not adjacent to z. Since $r \geq 3$ we know $\epsilon(x) \geq 3$ and so there must exist another vertex v (not one of w, y, z) such that $d(x, v) \geq 3$. Thus v is not adjacent to either of x or y. Moreover, v cannot be adjacent to both w and z since otherwise $d(w, z) < 3$. Thus at least one of the edges vw or vz, but possibly both, cannot exist in G.

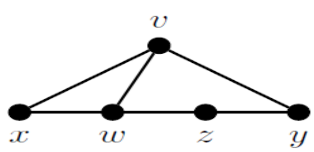


G

or

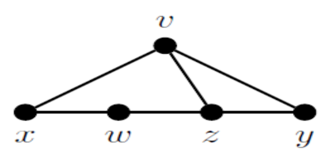


G



$\overline{G}$

or



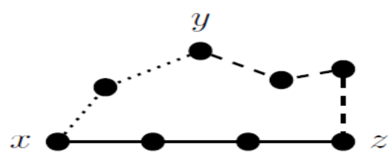
$\overline{G}$

In either case, we have that the distance in $\overline{G}$ between any two of these vertices is at most 2, and since this holds for any collection of vertices in G, we see that $rad(\overline{G}) \leq 2$.

Theorem 2.28 For any simple graph G, $rad(G) \leq diam(G) \leq 2rad(G)$.

Proof: First note that if the radius of a graph G is r then there is some vertex v with $\epsilon(v) = r$. Thus there cannot be a longest path from v to any other vertex that is shorter than r. Thus $rad(G) \leq diam(G)$.

Next, suppose x, y, and z are vertices of G. Then the shortest path from x to z may or may not travel along the shortest paths from x to y and then from y to z.



In either case, we know that $d(x, y) + d(y, z) \geq d(x, z)$. So suppose x and z are chosen so that $d(x, z) = diam(G)$. Let y be a vertex so that $\epsilon(y) = rad(G) = r$. Then we know that $d(x, y) \leq r$ and $d(y, z) \leq r$ since no minimum path from y can be longer than the longest minimum path from y, which has length r. Therefore we have $d(x, z) \leq r + r$ and so $diam(G) \leq 2rad(G)$.

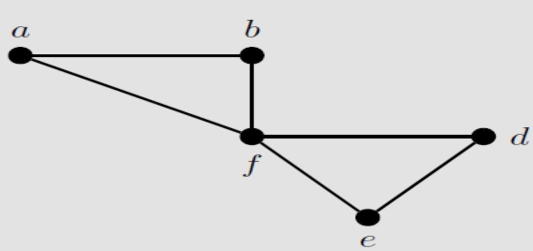
Once the radius of a graph is known, a natural question would be which vertex (or vertices) have the minimum eccentricities that result in that radius.

- A vertex of this type is called a central vertex and the collection of central vertices is called the center of a graph.

Definition 2.29 Let G be a graph with $rad(G) = r$. Then x is a ***central vertex*** if $\epsilon(x) = r$. Moreover, the ***center*** of G is the graph $C(G)$ that is induced by the central vertices of G.

Example 2.19 Find the center of the graph from Example 2.18.

Solution: Vertices a, b, d, e, and f all have eccentricities of 2. The center of G is the graph induced by these vertices, as shown below.



- ✓ Radius and diameter rely on (shortest) paths in their definition. If we turn instead to cycles, we can ask a similar question as to the length of both the shortest and longest cycles within a graph.

Definition 2.30 Given a graph G, the ***girth*** of G, denoted $g(G)$, is the minimum length of a cycle in G. The ***circumference*** of G is the maximum length of a cycle.

Example 2.20 Find the girth and circumference for the graph from Example 2.18.

Solution: Since the graph is simple, we know the girth must be at least 3, and since we can find triangles within the graph we know $g(G) = 3$. Moreover, the circumference is 7 since we can find a cycle containing all the vertices (try it!).

Remark:

- Note that the circumference of any graph is at most n, the number of vertices since no vertex can be repeated in a cycle. Moreover, if a graph does not have any cycles (what we will call a tree in Chapter 3) then we define g(G) = ∞ and the circumference to be 0.
- It shouldn't be too surprising that the diameter and girth of a graph are related, since removing one edge xy from a cycle creates a path and would increase the distance between x and y.

Theorem 2.31 If G is a graph with at least one cycle then $g(G) \leq 2diam(G) + 1$.

- While girth, radius, and diameter have clear relationships to one another, the same cannot be said about their relationship to degree measures of a graph.
- In fact, we can find connected graphs with a large diameter when the minimum degree is quite small or quite large.
- however, to maintain a small minimum degree with a large diameter, we will need to have a large number of vertices in the graph.
- Conversely, if a graph has a small diameter and maximum degree, it cannot have too many vertices.

Theorem 2.32 Let G be a graph with n vertices, radius at most k, and maximum degree at most d, with $d \geq 3$. Then $n < \frac{d}{d-2}(d-1)^k$.