Module 2 Connected graphs and shortest paths

Contents

Any network (communication or pipe line or transportation) consists of nodes and physical links connecting certain pairs of nodes. One of the network problems is to move objects (messages/liquids/vehicles) between two given nodes in shortest possible time (or distance).

This real world problem can be easily modeled as a graph theoretic problem. Figure 2.1 shows a communication network with five nodes, each of which is represented by a vertex. An edge represents a direct link. The integer along an edge represents the time to send a message along that link.

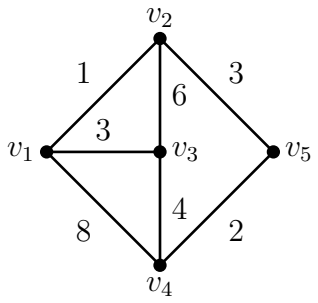


Figure 2.1: A communication network.

By examining all possible paths from v_1 to v_4, we find that the shortest route to send a message from v_1 to v_4 is along the path (v_1, v_2, v_5, v_4) and it takes six units of time. There exist good algorithms to find shortest paths which avoid brute force method of examining all paths between two vertices.

2.1 Walks, trails, paths, cycles

The real world concept of "moving" objects between two nodes is captured in the following terminology.

Definitions. Let G be a graph and let $v_0, v_t \in V(G)$.

- A (v_0, v_t)-***walk*** is a finite alternating sequence

$$W(v_0, v_t) = (v_0, e_1, v_1, e_2, v_2, \ldots, e_t, v_t)$$

of vertices and edges such that e_i is an edge incident with vertices v_{i-1} and $v_i, i = 1, 2, \ldots, t$.

- v_0 is called the ***origin*** and v_t is called the ***terminus***. Other vertices are called the ***internal vertices***. Note that v_0 and v_t can also be internal vertices.
- The ***length*** of W is the number of edges it contains where an edge is counted as many times as it occurs.
- W is called a ***closed walk***, if $v_0 = v_t$.

Remarks.

- In a walk, vertices and edges may appear any number of times.
- If there exists a (v_0, v_t)-walk, then there exists a (v_t, v_0)-walk.
- If G is a simple graph, W is denoted as a sequence of vertices $(v_0, v_1, \ldots, v_t)$ with the understanding that (v_i, v_{i+1}) is an edge, for $i = 0, 1, \ldots, t-1$.

Definitions.

- A (v_0, v_t)-walk is called a (v_0, v_t)-***trail***, if no edge is repeated (but vertices may get repeated). It is called a ***closed trail*** if $v_0 = v_t$.
- A (v_0, v_t)-walk is called a (v_0, v_t)-***path***, if no vertex is repeated (and therefore no edge is repeated).

  By definition, every path is a trail and every trail is a walk. However, a walk need not be a trail and a trail need not be a path.
- A closed walk $W(v_0, v_t)$ is called a ***cycle***, if all its vertices are distinct except that $v_0 = v_t$.
- A cycle with k vertices is called a k-cycle and it is denoted by C_k. A C_3 is also a K_3 and it is referred to as a ***triangle***. A 1-cycle is a loop and a 2-cycle consists of two multiple edges.

Any subsequence $W^1(v_i, v_j) = (v_i, e_{i+1}, v_{i+1}, \ldots, v_j)$ of W is called a subwalk of W. We illustrate these concepts by taking a graph.

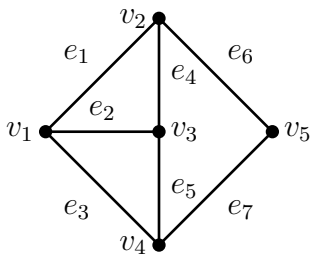


Figure 2.2: A graph G.

Define:

(1) $W_1(v_1, v_5) = (v_1, e_1, v_2, e_4, v_3, e_2, v_1, e_3, v_4, e_5, v_3, e_4, v_2, e_6, v_5)$.

(2) $W_2(v_1, v_5) = (v_1, e_1, v_2, e_4, v_3, e_2, v_1, e_3, v_4, e_7, v_5)$.

(3) $W_3(v_1, v_5) = (v_1, e_1, v_2, e_4, v_3, e_5, v_4, e_7, v_5)$.

(4) $W_4(v_1, v_1) = (v_1, e_1, v_2, e_4, v_3, e_2, v_1, e_3, v_4, e_5, v_3, e_2, v_1)$.

(5) $W_5(v_1, v_1) = (v_1, e_1, v_2, e_4, v_3, e_5, v_4, e_3, v_1)$.

Here, W_1 is a (v_1, v_5)-walk of length 7. It is not a trail. W_2 is a trail of length 5 but it is not a path. W_3 is a path of length 4. W_4 is a closed walk of length 6 but it is not a cycle. W_5 is a cycle of length 4. As per our convention, W_5 is also denoted by $(v_1, v_2, v_3, v_4, v_1)$.

Theorem 2.1. *Every (v_0, v_t)-walk W contains a (v_0, v_t)-path.*

Proof. Among all (v_0, v_t)-subwalks of W, let $P(v_0, v_t)$ be a subwalk of W which has minimum length. We claim that P is a path. Otherwise, there exist i and j (say $i < j$) such that $v_i = v_j$. That is,

$$P(v_0, v_t) = (v_0, \ldots, v_i, e_{i+1}, v_{i+1}, \ldots, v_j(= v_i), e_{j+1}, v_{j+1}, \ldots, v_t).$$

By deleting the subsequence $(e_{i+1}, v_{i+1}, \ldots, v_j)$, we obtain a (v_0, v_t)-subwalk of W which has lesser length, which is a contradiction to the minimality of P. So, $P(v_0, v_t)$ is a path. □

Theorem 2.2. *If G is simple and $\delta(G) \geq 2$, then there exists a cycle of length of at least $\delta(G) + 1$ in G.*

Proof. Let P be a path of maximum length in G. Let $P = (v_1, v_2, \ldots, v_t)$. If v is a vertex adjacent with v_1, then $v \in \{v_2, v_3, \ldots, v_t\}$; else $(v, v_1, v_2, \ldots, v_t)$ is a path of greater length, which is a contradiction to the maximality of P. So, $N(v_1) \subseteq \{v_2, v_3, \ldots, v_t\}$. Let v_k be the last vertex in P to which v_1 is adjacent; see Figure 2.3. Then the subpath $Q = (v_1, v_2, \ldots, v_k)$ contains at least $\deg(v_1) + 1 \geq \delta(G) + 1$

Figure 2.3: A maximum path $\mathrm{P}(v_1, v_t)$ and the resultant cycle.

vertices, and so $(v_1, v_2, \ldots, v_k, v_1)$ is a cycle of length $\geq \delta(G) + 1$. □

Theorem 2.3. *Every graph G with $m(G) \geq n(G)$ contains a cycle.*

Proof. If G contains a loop (= a 1-cycle) or a multiple edge (= a 2-cycle) we are through. So, we prove the theorem for simple graphs. This we do by induction on n. If $n \leq 3$, then there is only one graph with $m \geq n$, namely C_3. So we proceed to the induction step. If $\delta(G) \geq 2$, then G contains a cycle by Theorem 2.2. Next assume that $\delta(G) \leq 1$, and let v be a vertex of degree ≤ 1 in G. Then $G - v$ is a graph with $m(G - v) \geq n(G - v)$. Therefore, by induction hypothesis, $G - v$ contains a cycle. Hence G too contains a cycle. □

Definitions. If G contains a cycle, then the following are defined.

- The length of a shortest cycle in G is called its ***girth***.
- The length of a longest cycle in G is called its ***circumference***.

If G is acyclic, then girth and circumference are defined to be ∞.

Graph	P_n	C_n	K_n	$K_m^c + K_n^c$	Q_n	P
Girth	∞	n	3	4	4	5
Circumference	∞	n	n	$2\min\{m,n\}$	2^n	9

Table 2.1: Girths and Circumferences; $m, n \geq 3$ and P is the Petersen graph.

Corollary. *If G is simple and $\delta(G) \geq 2$, then $circumference(G) \geq \delta(G) + 1$.*

Proof. A consequence of Theorem 2.2 □

Theorem 2.4. *If G is a simple graph on least six vertices, then either*

(i) G contains at least three vertices which are mutually adjacent, or

(ii) G contains at least three vertices whcih are mutually non-adjacent.

Proof. Let v be a vertex. Since $n \geq 6$,

either (a) there are at least three vertices, say v_1, v_2, v_3, which are adjacent to v, or (b) there are at least three vertices, say u_1, u_2, u_3 which are non-adjacent to v.

Suppose (a) holds (see Figure 2.4):

If there are two vertices in $\{v_1, v_2, v_3\}$ which are adjacent, say v_1, v_2, then v, v_1, v_2 are three mutually adjacent vertices. On the other hand, if no two vertices of $\{v_1, v_2, v_3\}$ are adjacent, then v_1, v_2, v_3 are three mutually non-adjacent vertices. So, (i) or (ii) holds as claimed in the theorem.

Suppose (b) holds:

If there are two vertices in $\{u_1, u_2, u_3\}$, which are non-adjacent say u_1, u_2, then v, u_1, u_2 are three mutually non-adjacent vertices. On the other hand, if any

Figure 2.4: Adjacency of v.

two vertices in $\{u_1, u_2, u_3\}$ are adjacent, then u_1, u_2, u_3 are three mutually adjacent vertices. So, (i) or (ii) holds. □

The above theorem can be reformulated as follows:

Theorem 2.5. *If G is a simple graph on at least six vertices, then either $K_3 \subseteq G$ or $K_3 \subseteq G^c$.* □

Remarks.

- The assumption $n \geq 6$ made in Theorem 2.4 is necessary. For example, C_5 is a graph on five vertices which satisfies neither (i) nor (ii).
- Which of the problems stated at the beginning of this course is now solved?

2.2 Connected graphs

Clearly, we can move objects between two nodes if they are "connected".

Definitions.

- In a graph G, two vertices u and v are said to be ***connected***, if there exists a (u, v)-path.
- G is said to be a ***connected graph*** if any two vertices are connected; else, G is said to be a ***disconnected graph***.
- A maximal connected subgraph H of a graph G is called a ***component*** of G; maximal in the sense that if H_1 is a connected subgraph of G such that $H \subseteq H_1$, then $H = H_1$.

- The number of components in a graph G is denoted by $c(G)$. So, $c(G) = 1$ if and only if G is connected.

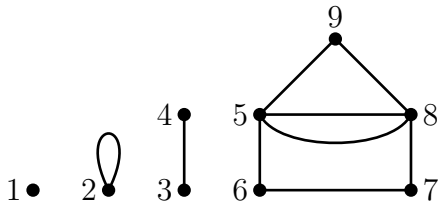
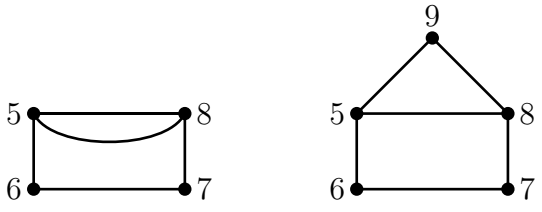


Figure 2.5: A graph G with four components.



(a) H_1 (b) H_2

Figure 2.6: Two subgraphs of G which are not its components; they are connected subgraphs but not ***maximal*** connected subgraphs.

Theorem 2.6. *For any graph G, $m(G) \geq n(G) - c(G)$.*

Proof. We prove the theorem by induction on n.

Basic step: If $n = 1$, the result is obvious.

Induction step: If $\delta(G) \geq 2$, then $m(G) = \frac{1}{2}\left(\sum_{v \in V(G)} d(v)\right) \geq \frac{1}{2}(2n(G)) = n(G) > n(G) - c(G)$. If $\delta(G) = 0$, let v be a vertex of degree 0. Then

$$\begin{aligned} m(G) &= m(G - v) \\ &\geq n(G - v) - c(G - v), \text{ by induction hypothesis,} \\ &= (n(G) - 1) + (c(G) - 1) \\ &= n(G) - c(G). \end{aligned}$$

Next assume that $\delta(G) = 1$, and let v be a vertex of degree 1. Then $c(G-v) = c(G)$. So,

$$\begin{aligned} m(G) &= m(G-v) + 1 \\ &\geq n(G-v) - c(G-v) + 1, \text{ by induction hypothesis,} \\ &= (n(G)-1) - c(G-v) + 1 \\ &= n(G) - c(G). \end{aligned}$$

□

Corollary. *For any connected graph G, $m(G) \geq n(G)-1$.* □

Theorem 2.7. *Every graph with $m \geq n-1$ is either connected or contains a cycle.*

Proof. It is enough if we prove the theorem for simple graphs. Assume the contrary and let G be a simple graph with $m \geq n-1$ which is neither connected nor contains a cycle. Let $G_1, G_2, \ldots, G_t$ be its component where $t \geq 2$. Let G_i have n_i vertices and m_i edges, $i = 1, 2, \ldots, t$. Since G_i is acyclic, using Theorem 2.3, we deduce that $m_i \leq n_i - 1$. So,

$$\begin{aligned} m &= m_1 + m_2 + \cdots + m_t \\ &\leq (n_1 - 1) + (n_2 - 1) + (n_t - 1) \\ &= n - t \\ &\leq n - 2. \end{aligned}$$

This is a contradiction to our assumption that $m \geq n-1$. □

Theorem 2.8. *A connected simple graph G contains a cycle if and only if $m \geq n$.*

Proof. If $m \geq n$, then G contains a cycle by Theorem 2.3. The reverse implication can be proved by induction on n by following the proof of Theorem 2.7. □

The above theorem characterizes the graphs with a cycle. However, it is a difficult open problem to find sufficient conditions (or necessary conditions) for the existence of a cycle C_k of specified length k. The following theorem gives a sufficient condition for a simple graph to contain a C_3.

Theorem 2.9. *Every simple graph with $m > \frac{n^2}{4}$ contains a cycle of length 3(= triangle).*

Proof. We prove that if G is a simple graph which has no triangles, then $m \leq \frac{n^2}{4}$. This we do by induction n.

Case 1: n is even.

If $n = 2$, then the inequality is obvious. So, we proceed to the induction step assuming that G has $n + 2$ vertices and that the theorem holds for all graphs on n vertices. There exists an edge (u, v) in G; else, $m = 0$. We partition E(G) into three parts as follows:

(1) $E(G) = E(G - \{u, v\}) \cup \{e \in E(G) : e$ has one end vertex in $\{u, v\}$ and other end vertex in $G - \{u, v\}\} \cup \{(u, v)\}$.

Since $G-\{u, v\}$ has no triangles, $m(G-\{u, v\}) \leq \frac{n^2}{4}$, by induction hypothesis. No vertex in $G - \{u, v\}$ is adjacent to both the vertices u and v, since G has no triangles. So

$$|\{e \in E(G) : e \text{ has one end vertex in } \{u, v\} \text{ and other end vertex in } G - \{u, v\}\}| \leq n(G - \{u, v\}) = n.$$

Hence, $m(G) \leq m(G - \{u, v\}) + n + 1 \leq \frac{n^2}{4} + n + 1 = \frac{(n+2)^2}{4}$.

Case 2: n is odd. Proof is exactly as above. □

• Distance

The concept of "distance" occurs in every branch of mathematics; graph theory is no exception.

Definitions. Let G be a graph and $u, v \in V(G)$.

- The ***distance*** $d_G(u,v)$ or $d(u,v)$ between u and v is defined as follows:

$$d_G(u,v) = \begin{cases} \text{length of a shortest } (u,v) \text{ path}, & \text{if } u \text{ and } v \text{ are connected,} \\ \infty, & \text{if } u \text{ and } v \text{ are not connected.} \end{cases}$$

- The ***diameter*** of G, $\text{diam}(G)$, is defined by

$$diam(G) = \begin{cases} \max\{d(u,v) : u, v \in V(G)\}, & \text{if } G \text{ is connected,} \\ \infty, & \text{if } G \text{ is disconnected.} \end{cases}$$

If G is connected, then the function $d : V(G) \times V(G) \to \mathbb{R}$ is a metric on $V(G)$. Formally, we state this fact as a theorem. Its proof is easy and hence it is left as an exercise.

Theorem 2.10. *If G is a connected graph, then $(V(G), d)$ is a metric space. That is:*

(i) $d(u,v) \geq 0$, *for every* $u, v \in V(G)$.
(ii) $d(u,v) = 0$ *iff* $u = v$.
(iii) $d(u,v) = d(v,u)$, *for every* $u, v \in V(G)$.
(iv) $d(u,v) \leq d(u,w) + d(w,v)$, *for every* $u, v, w \in V(G)$. □

Theorem 2.11. *Let $A = [a_{ij}]$ be the adjacency matrix of a simple graph G. Then the (i,j) th entry $[A^p]_{ij}$ in A^p is the number of walks of length p from v_i to v_j.*

Proof. We prove the theorem by induction on p. If $p = 1$, then A^p is A and the result is obvious. Suppose that the result is true for $p = r$ and let $p = r + 1$. We

have,

$$[A^{r+1}]_{ij} = \sum_{k=1}^{n} [A^r]_{ik}\ a_{kj}.$$

Now

$$[A^r]_{ik}\ a_{kj} = \begin{cases} [A^r]_{ik}, & \text{if } a_{kj} = 1, \text{ that is, if } v_k \text{ and } v_j \text{ are adjacent.} \\ 0 & \text{if } a_{kj} = 0, \text{ that is, if } v_k \text{ and } v_j \text{ are not adjacent.} \end{cases}$$

By induction, $[A^r]_{ik}$ is the number of walks of length r connecting v_i and v_k. Each of these walks is extendable to a walk of length $r+1$ connecting v_i and v_j iff v_k and v_j are adjacent, that is, iff $a_{kj} = 1$. So, by the above equations, $[A^{r+1}]_{ij}$ is the number of walks of length $r+1$ connecting v_i and v_j. □

Theorem 2.12. *If G is a connected simple graph, then the distance between v_i and v_j is the smallest integer $p(\geq 0)$ such that $[A^p]_{ij} \neq 0$.*

Proof. By the minimality of p, $[A^r]_{ij} = 0$, for every $r, 0 \leq r \leq p-1$. So, by Theorem 2.11, there is no walk of length $\leq p-1$ connecting v_i and v_j; hence, $d(v_i, v_j) \geq p$. Since $[A^p]_{ij} \neq 0$, there does exist a walk, say $W(v_i, v_j)$, from v_i to v_j. Since, $[A^r]_{ij} = 0$, for all r, $0 \leq r \leq p-1$, $W(v_i, v_j)$ is a walk of minimum length and hence it is a path. So, $d(v_i, v_j) \leq p$. Combining the two inequalities we get $d(v_i, v_j) = p$. □

• Cut-vertices and cut-edges

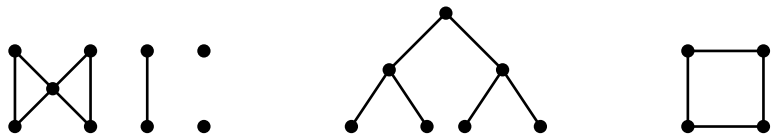
The following two concepts make precise the notions of faulty nodes and faulty links. At the outset, observe that the number of components in $G - v$ may increase or decrease or remain the same, when compared to the number of components in G. In fact, $c(G-v) < c(G)$ iff $deg(v) = 0$.

Let G be a graph, v be a vertex and e be an edge.

Definition. *v is said to be a **cut-vertex** if $c(G-v) > c(G)$.*

Remarks.

- If G is connected, then v is a cut-vertex if $G - v$ is disconnected.
- v is a cut-vertex of G if and only if v is a cut-vertex of a component of G.



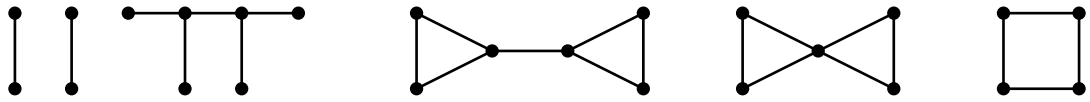
(a) G_1 (b) G_2 (c) G_3

Figure 2.7: G_1 contains exactly one cut-vertex, G_2 contains three cut-vertices and G_3 contains no cut-vertices.

Definition. *e is said to be a cut-edge of G if $c(G - e) > c(G)$.*

Remarks.

- If $e(u, v)$ is a cut-edge of G, then u and v are in two different components of $G - e$. Moreover, $c(G - e) = c(G) + 1$.
- The two remarks made above with respect to a cut-vertex hold good for a cut-edge also.



(a) G_1 (b) G_2 (c) G_3 (d) G_4

Figure 2.8: Every edge in G_1 is a cut-edge, G_2 contains exactly one cut-edge and two cut-vertices, G_3 contains a cut-vertex but contains no cut-edge, G_4 contains neither a cut-vertex nor a cut edge.

Theorem 2.13. *A vertex v of a connected graph is a cut-vertex if and only if there exist vertices x and y ($\neq v$) such that every (x, y)-path contains v.*

Proof. (1) Suppose v is a cut-vertex. $G - v$ is disconnected so it contains at least two components, say C and D. Let $x \in V(C)$ and $y \in V(D)$. Since there is no (x, y)-path in $G - v$, it follows that every (x, y)-path in G contains v.
(2) Suppose there exist x and y $(\neq v)$ such that every (x, y)-path contains v. It follows that there is no (x, y)-path in $G - v$. Hence $G - v$ is disconnected, that is v is a cut-vertex. □

Theorem 2.14. *Let G be a connected graph with at least three vertices. If e(u, v) is a cut-edge in G, then either u or v is a cut-vertex.*

Proof. In $G - e$, there exist two components C and D such that $u \in V(C)$ and $v \in V(D)$. Since $n \geq 3$, either $n(C) \geq 2$ or $n(D) \geq 2$, say $n(C) \geq 2$. Then u is a cut-vertex of G. □

Remarks.

- By the above theorem, it follows that if G is connected, $n(G) \geq 3$ and G has a cut-edge, then G has a cut-vertex. However, the converse is false: The graph G_3 shown in Figure. 2.8 contains a cut-vertex but contains no cut-edges.
- If $e(u, v)$ is a cut-edge, then it is not necessary that both u ***and*** v are cut-vertices; if $d(u) \geq 2$ and $d(v) \geq 2$, then both u and v are cut-vertices.

Theorem 2.15. *An edge $e(u, v)$ is not a cut-edge of G if and only if e belongs to a cycle in G.*

Proof. (1) Suppose e is not a cut-edge.

So, $G - e$ is connected, and u, v are in the same components of $G - e$. So, there exists a path $P(u, v)$ in $G - e$. But then, $(P(u, v), v, u)$ is a cycle in G containing e.
(2) Suppose e belongs to cycle C in G.

So, $C-e$ is a (u,v)-path in $G-e$. Hence, u and v are in the same components of $G-e$, that is $G-e$ is connected. Therefore, e is not a cut-edge. □

Theorem 2.16. *Every connected graph G with $n \geq 2$, contains at least 2 vertices which are not cut-vertices.*

Proof. Let P be a path of maximum length in G; let $P = P(x,y)$. Our claim is that neither x nor y is a cut-vertex of G. On the contrary, suppose x is a cut-vertex and consider $G-x$; see Figure 2.9. It contains at least 2 components say C and D where $y \in D$. Let v be any vertex in C. Then every (v,y)-path in G contains x.

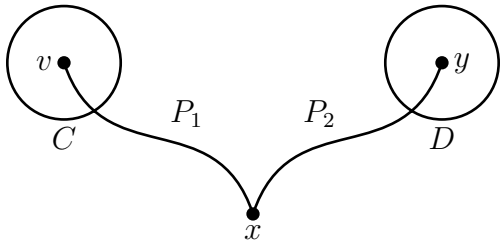


Figure 2.9: Existence of two paths P_1 and P_2 in G.

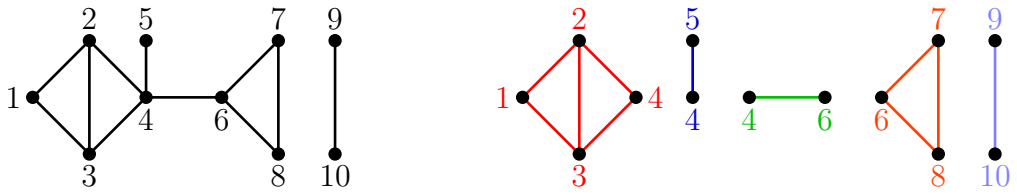
But then $Q = (P_1(v,x), P_2(x,y))$ is a path of length greater than the length of P, a contradiction to the maximality of P. □

• Blocks

Definition. A maximal connected subgraph B of a graph G that has a no cut-vertex of its own is called a ***block*** of G; maximal in the sense that if H is a connected subgraph of G that has no cut-vertex and $H \subseteq B$, then $H = B$; see Figure 2.10.

If G has no cut-vertices, then G is called a block.

Remarks.



(a) G (b) Blocks of G.

Figure 2.10: A graph G and its five blocks.

- A block of a graph does not have a cut-vertex of its own. However, it may contain cut-vertices of the whole graph. In example 2.10a, the edge joining 4 and 6 is a block and both the end points are cut-vertices of the whole graph.
- By definition, G itself is a block if it is connected and it has no cut-vertex.
- Two blocks in a graph share at most one vertex; else, the two blocks together form a block, thus contradicting the maximality of blocks.
- If two distinct blocks of G share a vertex v, then v is a cut-vertex of G.
- Any two distinct blocks are edge disjoint; so the edge sets of blocks partition the edge set of G.
- To establish a property P of a graph G, often it is enough to establish P for each of its blocks, and thereby simplify the proofs.

Definition. Two paths $P(x, y)$ and $Q(x, y)$ are said to be ***internally disjoint*** if they have no internal vertex common.

The following result is a special case of a theorem proved by Menger (1932).

Theorem 2.17. *Let G be a graph with $n(G) \geq 3$. Then G is a block if and only if given any two vertices x and y of G, there exist at least two internally disjoint (x, y)-paths in G.*

Proof. (1) Given any two vertices x, y of G, there exist at least two internally disjoint (x, y)-paths $\Rightarrow$ G is a block.

By the hypothesis, G is connected. So, we have to only show that G has no cut-vertices. On the contrary, suppose v is a cut-vertex of G. Then by Theorem 2.13, there exist vertices x, y such that every (x, y)-path passes through v. This implies that there do not exist two internally disjoint (x, y)-paths, which is a contradiction to the hypothesis.

(2) G is a block $\Rightarrow$ Given any two vertices x, y of G, there exist at least two internally disjoint (x, y)-paths.

We prove the implication by induction on the distance $d(x, y)$.

Basic step: $d(x, y) = 1$. Let e be an edge joining x and y. Then $G - e$ is connected; else, x or y is a cut-vertex which is a contradiction, since G is a block. Hence, there exists a path $P(x, y)$ in $G-e$. But then (x, e, y) and $P(x, y)$ are two internally disjoint (x, y)-paths.

Induction step: $d(x, y) > 1$. Let $P(x, y)$ be a shortest (x, y)-path; so length of $P = d(x, y)$. Let w be a vertex that precedes y in P. So, $d(x, w) = d(x, y) - 1$. Hence, by induction hypothesis, there exist two internally disjoint (x, w)-paths, say $Q(x, w)$ and $R(x, w)$; see Figure 2.11.

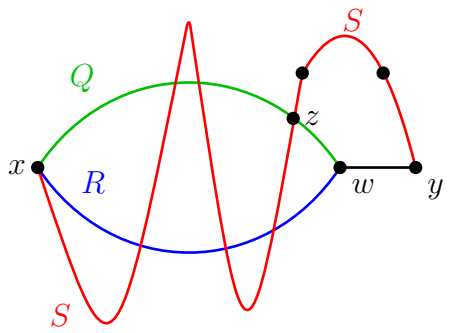


Figure 2.11: Construction of two internally disjoint (x, y)-paths in G.

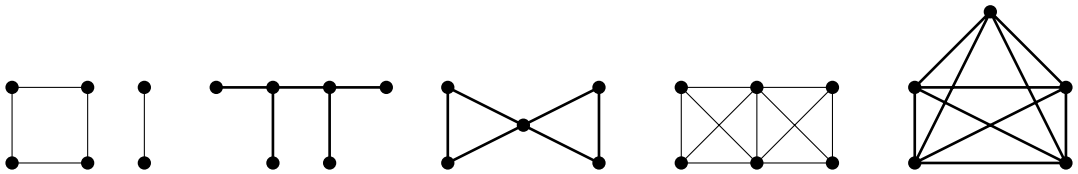
Since G is a block, w is not a cut-vertex. So, there exists a path $S(x, y)$ in $G - w$. Let z be the last vertex in $S(x, y)$ that lies in $V(Q) \cup V(R)$. For definiteness, let $z \in V(Q)$. Then $(Q(x, z), S(z, y))$ and $(R(x, w), w, y)$ are two internally disjoint (x, y)-paths. □

A reformulation of the above theorem yields an alternative characterization of a block.

Corollary. *A graph G with at least three vertices is a block if and only if given any two vertices x and y, there exists a cycle containing x and y.* □

2.3 Connectivity

Consider the graphs shown in Figure 2.12 representing communication or transportation networks. They are successively more robust. k-vertex-connectivity and k-edge-connectivity are two basic parameters that measure the robustness of a graph/network.



(a) G_1 (b) G_2 (c) G_3 (d) G_4 (e) G_5

Figure 2.12: Successively more robust graphs.

Definitions (k-vertex-connectivity)**.**

- A vertex subset $W \subseteq V(G)$ is called a ***vertex-cut*** if $G - W$ is disconnected or $G - W$ is a single vertex graph.
- The integer

$$k_0(G) = \min\{|W| : W \subset V(G), W \text{ is a vertex-cut}\}.$$

is called the ***vertex-connectivity*** of G. That is, $k_0(G)$ is the minimum number of vertices whose deletion disconnects G or results in a graph with a single vertex.

- Any vertex-cut $K \subseteq V(G)$ such that $|K| = k_0(G)$ is called a ***minimum vertex-cut*** of G.
- A graph G is said to be p-***vertex-connected***, if $k_0(G) \geq p$.

Remarks.

- $k_0(G) = 0$ if and only if G is disconnected or G is a single vertex graph.
- $k_0(G) = 1$ if and only if G is connected and it has a cut-vertex.
- $k_0(G) \geq 2$ if and only if G is a block.
- $k_0(G) = n(G) - 1$ if and only if $G \supseteq K_n$.
- A graph G may have many minimum vertex-cuts but $k_0(G)$ is unique.
- If G is p-vertex-connected, then it is t-vertex-connected for every t, $1 \leq t \leq p$. Thus a 3-connected graph is also a 2-connected graph and a connected graph.

Definitions (k-edge-connectivity)**.**

- An edge subset $F \subseteq E(G)$ is called an ***edge-cut*** if $G - F$ is disconnected or $G - F$ is a single vertex graph.
- The integer

$$k_1(G) = \min\{|F| : F \subseteq E(G) \text{ and } F \text{ is an edge cut}\}$$

is called the ***edge-connectivity*** of G. That is, $k_1(G)$ is the minimum number of edges whose deletion disconnects G or results in a single vertex graph.

- Any edge-cut $F \subseteq E(G)$ such that $|F| = k_1(G)$ is called a ***minimum edge-cut***.
- A graph G is said to be s-***edge-connected*** if $k_1(G) \geq s$.

Remarks.

- $k_1(G) = 0$ if and only if G is disconnected or G is a single vertex graph.
- $k_1(G) = 1$ if and only if G is connected and G has a cut-edge.
- $k_1(G) \leq \delta(G)$; this follows since by deleting all the edges incident with a vertex of minimum degree, we disconnect the graph.
- A graph G may have many minimum edge-cuts but $k_1(G)$ is unique.
- If G is s-edge-connected, then it is t-edge-connected, for every t, $1 \leq t \leq s$.
- If F is a minimum edge-cut, then $G - F$ contains exactly two components A and B such that $[V(A), V(B)] = F$.

Table 2.2 shows the above two parameters for the graphs shown in Figure 2.12, some standard graphs ($n \geq 3$) and the Petersen graph P.

	G_1	G_2	G_3	G_4	G_5	K_n	P_n	C_n	P
k_0	0	1	1	2	4	$n-1$	1	2	3
k_1	0	1	2	3	4	$n-1$	1	2	3

Table 2.2: Vertex and edge connectivity of graphs

Theorem 2.18. *For any graph G on at least two vertices, $k_0(G) \leq k_1(G) \leq \delta(G)$.*

Proof. We have already remarked above that $k_1(G) \leq \delta(G)$. Below we prove that $k_0(G) \leq k_1(G)$. If G is disconnected or G is a single vertex graph, then $k_0(G) = 0 = k_1(G)$, $\delta(G) \geq 0$. So, next assume that G is connected. If $n(G) = 2$, then $k_0(G) = 1$, $k_1(G) =$ number of edges joining the two vertices, and $\delta(G) \geq$ number of edges joining the two vertices . Therefore the result follows. Next assume that G is connected and $n(G) \geq 3$.

Case 1: G is simple.

Let $k_1(G) = \lambda$ and let $F = \{e_1(u_1, v_1), \ldots, e_\lambda(u_\lambda, v_\lambda)\}$ be a minimum edge-cut where u_i and v_j need not be distinct. Let $H = G - (\{u_1, u_2, \ldots, u_{\lambda-1}\} - \{u_\lambda, v_\lambda\})$. Then e_λ is a cut-edge of H. So, u_λ or v_λ is a cut-vertex of H; say u_λ. But then

$\{u_1, \ldots, u_{\lambda-1}, u_\lambda\}$ is a vertex-cut of G. Therefore, $k_0(G) \leq \lambda = k_1(G)$.

Case 2: G is any graph.

Let H be an underlying simple spanning subgraph of G. Then $k_0(H) = k_0(G)$ and $k_1(H) \leq k_1(G)$. We deduce that $k_0(G) = k_0(H) \leq k_1(H) \leq k_1(G)$. □

Remark. The inequalities shown in Theorem 2.18 are best possible in the following sense. Given any three integers r, s, t such that $0 \leq r \leq s \leq t$, there exists a simple graph G with $k_0(G) = r$, $k_1(G) = s$, $\delta(G) = t$. (We leave it as an exercise to construct such a graph G. You can take a hint from the two graphs shown in Figure 2.13.)

(a) G_1 (b) G_2

Figure 2.13: $k_0(G_1) = 1$, $k_1(G_1) = 2$, $\delta(G_1) = 3$; $k_0(G_2) = 3$, $k_1(G_2) = 4$, $\delta(G_2) = 4$.

Two central theorems on connectivity are due to K. Menger (1932).

Theorem 2.19. *A graph G is k-vertex connected $(1 \leq k \leq n-1)$ if and only if given any two distinct vertices u and v, there exist k internally disjoint (u, v)-paths (that is, no two of the paths have an internal vertex common).*

Theorem 2.20. *A graph G is k-edge connected if and only if given any two distinct vertices u and v, there exists k edge-disjoint (u, v)-paths (that is, no two paths have an edge common).*

There are many sufficient conditions for a graph to be k-vertex-connected or k-edge-connected. We state and prove two results, and a few are included in the exercise list.

Theorem 2.21. *Let $1 \leq k \leq n$. If $deg_G(v) \geq \left\lceil \frac{n+k-2}{2} \right\rceil$, for every vertex v, then G is k-vertex-connected.*

Proof. Let $W \subseteq V(G)$ be a k-vertex-cut. Then $G - W$ contains at least two components and so it has a component D such that $d := |V(D)| \leq \frac{n-k}{2}$. Let x be a vertex in D; see Figure 2.14.

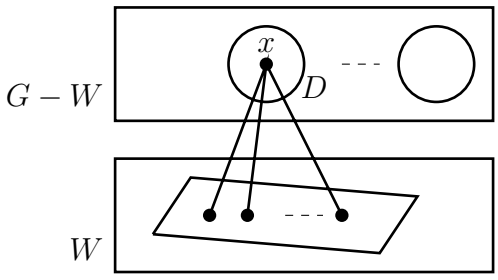


Figure 2.14: Estimate for $deg_G(x)$.

Then x can be adjacent to at most $d-1$ vertices in $G-W$ and k vertices in W. So, $deg_G(x) \leq d - 1 + k \leq \frac{n-k}{2} - 1 + k = \frac{n+k-2}{2}$, which is a contradiction to the hypothesis. □

If G is simple and $diam(G) = 1$, then $G = K_n$ and so $k_1(G) = n - 1 = \delta(G)$. The next results shows that simple graphs with diameter 2 also attain maximum possible edge-connectivity.

Theorem 2.22 (J. Plesink, 1975)**.** *If G is simple and $diam(G) = 2$, then $k_1(G) = \delta(G)$.*

Proof. **(Technique of two-way counting)** Let F be a a minimum edge-cut; so $|F| = k_1(G)$. Then G contains exactly two components, say $[A]$ and $[B]$ such that $[A, B] = F$. Let $|A| = a$ and $|B| = b$.

If there exists some pair of vertices $x \in A$ and $y \in B$ such that $[x, B] = \emptyset$ and $[A, y] = \emptyset$, then $\text{dist}(x, y) \geq 3$, contrary to the hypothesis. So, $[x, B] \neq \emptyset$, for every $x \in A$ or $[A, y] \neq \emptyset$, for every $y \in B$. For definiteness, let $[x, B] \neq \emptyset$, for every $x \in A$.

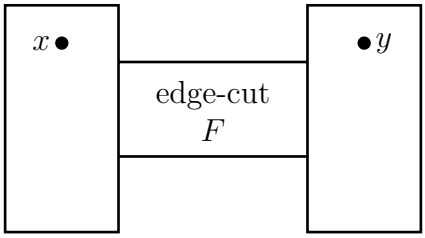


Figure 2.15: Two components $[A]$ and $[B]$ of $G - F$.

Therefore,

$$k_1 = |F| = \sum_{x \in A} |[x, B]| \geq \sum_{x \in A} 1 = a. \tag{2.1}$$

We next estimate $|F|$, using the following equation and deduce that $k_1 \geq \delta$.

$$d_G(x) = d_{[A]}(x) + |[x, B]|, \tag{2.2}$$

$$\begin{aligned}
k_1 = |F| &= \sum_{x \in A} |[x, B]| \\
&= \sum_{x \in A} d_G(x) - d_{[A]}(x), \text{by (2.2)} \\
&\geq \delta a - a(a-1), \text{ since } d_{[A]}(x) \leq a - 1, \text{ for every } x \in A \\
&= \delta + \delta(a-1) - a(a-1) \\
&= \delta + (a-1)(\delta - a) \\
&\geq \delta + (a-1)(k_1 - a), \text{ since } \delta \geq k_1, \text{ by Theorem 2.18} \\
&\geq \delta, \text{ since } a \geq 1, \text{ and } k_1 \geq a, \text{ by (2.1).}
\end{aligned}$$

Since $k_1 \leq \delta$, for every graph (Theorem 2.18), the result follows. □

2.4 Weighted graphs and shortest paths

In any network, the links are associated with "band widths" or "lengths" or "capacities". In discrete mathematics all these parameters are called the "weights" of the links.

• Weighted graphs

Definitions.

- A pair $(G, \mathcal{W})$, where G is a graph and $\mathcal{W} : E(G) \to \mathbb{R}$ is any function is called a ***weighted graph***; $\mathcal{W}(e)$ is called the weight of e. We assume that $\mathcal{W}(e) \geq 0$, for every edge e.
- Convention: Any unweighted graph G is treated as a weighted graph with $\mathcal{W}(e) = 1$, for every edge e.
- If H is a subgraph of G, then its weight $\mathcal{W}(H)$ is defined to be $\sum_{e \in E(H)} \mathcal{W}(e)$. So, the weight of a path P in G is $\sum_{e \in E(P)} \mathcal{W}(e)$. It is called the *weighted length* of P.
- The ***weighted distance*** $dist(u, v)$ between two given vertices u and v in a weighted graph $(G, \mathcal{W})$ is defined as

$$dist(u,v) = \begin{cases} \min\{\mathcal{W}(P) : \ P \text{ is a } (u,v)\text{-path}\}, & \text{if } u \text{ and } v \text{ are connected,} \\ \infty, & \text{if } u \text{ and } v \text{ are not connected.} \end{cases}$$

- We often drop the adjective "weighted" to reduce the writing.

  In the graph of Figure 2.16:

  - The weight of G is 14.5.
  - The weight of the path (u, a, v, c, x, e, y) is 12.
  - The weight of the path (u, a, v, d, x, e, y) is 4.5.
  - The weight of the path (u, b, x, e, y) is 5.

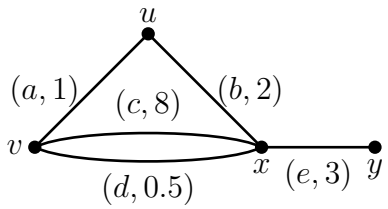


Figure 2.16: A weighted graph $(G, \mathcal{W})$, where the weight of an edge is shown next to its label.

– The shortest (u, y)-path is (u, a, v, d, x, e, y); so $d(u, y) = 4.5$; however, in the underlying "unweighted" graph, $d(u, y) = 2$.

If G is a connected weighted graph, define $d : V(G) \times V(G) \to \mathbb{R}$ by $d(u, v) = dist(u, v)$, for every $u, v \in V(G)$.

Theorem 2.23. *If G is a connected weighted graph with non-negative edge weights, then the following hold.*

1. $d(u, v) \geq 0$, *for every* u, $v \in V(G)$.
2. $d(u, v) = 0$, *if and only if* $u = v$ *or* $\mathcal{W}(u, v) = 0$.
3. $d(u, v) = d(v, u)$, *for every* $u, v \in V(G)$.
4. $d(u, v) \leq d(u, x) + d(x, v)$, *for every* $u, v, x \in V(G)$.

Proof. Exercise. □

In many real world network problems, one is often required to find the shortest distance and a shortest path between two given nodes. In graph theoretic terminology, these two problems can be modeled as follows:

Design an algorithm to find the shortest distance and a shortest path between two given vertices in a given weighted graph.

We describe two well-known such algorithms. It is assumed that the edge weights are non-negative.

• Dijkstra's shortest path algorithm (1959)

It is a "one-to-all" algorithm in the sense that given any vertex $u_0 \in V(G)$, the algorithm outputs the distance between u_0 and every other vertex. However, it can be easily modified into an "all-to-all" algorithm or to a "many-to-many" algorithm. It is based on the principle that if $(u_0, e_1, u_1, e_2, u_2, \ldots, u_{k-1}, e_k, u_k)$ is a shortest (u_0, u_k)-path, then $(u_0, e_1, u_1, e_2, u_2, \ldots, u_{k-1})$ is a shortest (u_0, u_{k-1})-path.

◦ Description of the algorithm

Input: A weighted graph $(G, \mathcal{W})$ and a vertex $u_0 \in V(G)$.

If a pair of vertices u and v are non-adjacent it is assumed that they are joined by an edge of large weight, say $2^{\mathcal{W}(G)}$, denoted ∞.

Output: The weighted distance $d(u_0, x)$, for every $x \in V(G)$.

Remark.

- ◦ It is a labeling process where every vertex x is dynamically labeled $l(x)$ and at the termination of the algorithm, $l(x)$ indicates $d(u_0, x)$.
- ◦ Each vertex x is associated with an array of 3 fields $\boxed{x \mid l(x) \mid p(x)}$, where $p(x)$ is the parents of x.

Step 1: Initialization

$l(u_0) \leftarrow 0; p(u_0) = u_0; l(x) \leftarrow \infty$ and $p(x) = NULL$, for every vertex $x \neq u_0$;

$S \leftarrow \{u_0\}; i \leftarrow 0$.

Step 2: Computation

If $S = V$ goto Step 4;

Else, for each $x \in V - S$ do:

(i) If $l(x) > l(u_i) + \mathcal{W}(u_i, x)$, replace $l(x)$ by $l(u_i) + \mathcal{W}(u_i, x)$; $p(x) \leftarrow u_i$.

Else, retain $l(x)$ and $p(x)$.

(ii) Compute $\min\{l(x) : x \in V - S\}$.

(iii) Designate any vertex for which minimum is attained in (ii) as u_{i+1}.

(If there is more than one vertex for which the minimum is attained, you can designate any such vertex as u_{i+1}.)

Step 3: Recursion

$S \leftarrow S \cup \{u_{i+1}\}$; $i \leftarrow i + 1$; goto step 2.

Step 4: Output $l(x)$ as $d(u_0, x)$, and the array $P(x, u_0) = (x, p(x), p^2(x), \ldots, u_0)$, for each $x \in V(G)$ and STOP.

In the following, we illustrate the algorithm by taking a weighted graph.

Initialization:

$Vertexx$	a	b	c	d	e
$l(x)$	0	∞	∞	∞	∞

$u_0 = a$, $p(a) = a$, $S = \{a\}$; $V - S = \{b, c, d, e\}$.

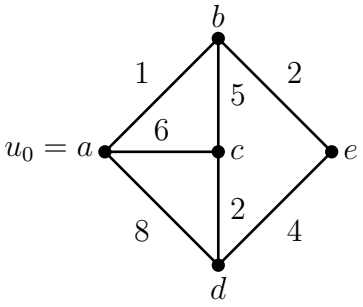


Figure 2.17: The input graph and its initialization.

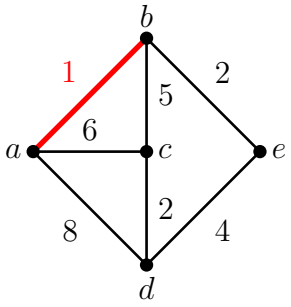
Iteration 1:

x	a	b	c	d	e
$l(x)$	0	1	6	8	∞

$\min\{l(x) : x \in V - S\} = 1$;

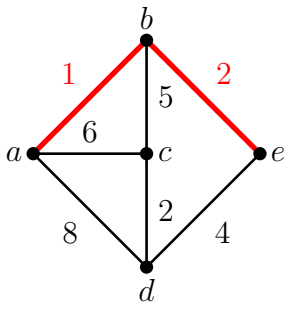
$l(b) = 1$; $p(b) = a$, $S = \{a, b\}$;

$V - S = \{c, d, e\}$, $P = (b, p(b)) = (b, a)$.



Iteration 2:

x	a	b	c	d	e
$l(x)$	0	1	6	8	3



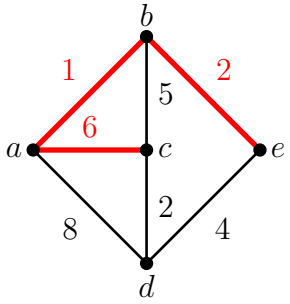
$\min\{l(x) : x \in V - S\} = 3$;

$l(e) = 3$; $p(e) = b$, $S = \{a, b, e\}$;

$V - S = \{c, d\}$, $P = (e, b, a)$.

Iteration 3:

x	a	b	c	d	e
$l(x)$	0	1	6	7	3



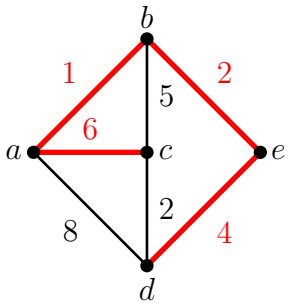
$\min\{l(x) : x \in V - S\} = 6$;

$l(c) = 6$; $p(c) = b$, $S = \{a, b, e, c\}$;

$V - S = \{d\}$, $P = (c, a)$.

Iteration 4:

x	a	b	c	d	e
$l(x)$	0	1	6	7	3



$\min\{l(x) : x \in V - S\} = 7$;

$l(d) = 7$; $p(d) = e$; $S = \{a, b, e, c, d\}$;

$V - S = \emptyset$, $P = (d, e, b, a)$.

We stop the algorithm and declare the labels $l(x)$ shown in the above table as $d(a, x)$.

- The shortest (a, b)-path is (a, b).
- A shortest (a, c)-path is (a, c). The reader may notice that (a, b, c) is also a shortest (a, c)-path.
- The shortest (a, d)-path is (a, b, e, d).
- The shortest (a, e)-path is (a, b, e).

• Floyd-Warshall shortest path algorithm (1962)

It is a "all-to-all" algorithm in the sense that given a weighted graph (G, $\mathcal{W}$), the algorithm outputs the shortest distance $d(x, y)$, for every pair (x, y) of vertices. The algorithm makes use of a recursion formula proved below.

Theorem 2.24 ((Floyd and Warshall, 1962)). *Let (G, $\mathcal{W}$) be a weighted graph on vertices labeled* $1, 2, \ldots, n$. *Define*

$$d_{ij}^k = \begin{cases} \textit{The weighted length of a shortest } (i,j)\textit{-path with all its internal vertices} \\ \textit{from } \{1,\ldots,k\}\textit{; this path need not contain every vertex from } \{1,\ldots,k\}. \end{cases}$$

Then,

$$d_{ij}^k = \min\{d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1}\}.$$

Proof. Let P be a shortest (i, j)-path with all its internal vertices from $\{1, \ldots, k\}$; so its length $l(P) = d_{ij}^k$. Two cases arise.

Case 1: k is not an internal vertex of P.

So P is a (i, j)-path with all its internal vertices from $\{1, 2, ..., k-1\}$. Hence, by defintion $d_{ij}^k = l(P) = d_{ij}^{k-1}$.

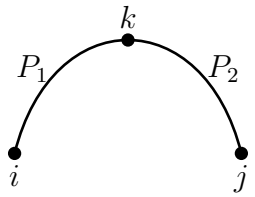


Figure 2.18: k is an internal vertex of P(i, j).

Case 2: k is an internal vertex of P.

In this case, P consists of two subpaths $P_1(i, k)$ and $P_2(k, j)$, where P_1 is a shortest (i, k)-path with all its internal vertices from $\{1, 2, \ldots, k-1\}$ and P_2 is a

shortest (k, j) path with all its internal vertices from $\{1, 2, \ldots, k-1\}$. So, $l(P_1) = d_{ik}^{k-1}$, $l(P_2) = d_{kj}^{k-1}$. Hence,

$$d_{ij}^{k} = l(P) = l(P_1) + l(P_2) = d_{ik}^{k-1} + d_{kj}^{k-1}.$$

The two cases imply that $l(P)$ is either d_{ij}^{k-1} or $d_{ik}^{k-1} + d_{kj}^{k-1}$, whichever is minimum. So, $d_{ij}^{k} = \min\{d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1}\}$. □

Remarks. By the definition of d_{ij}^{k}, it may be observed that

- d_{ij}^{0} := The weighted length of a shortest (i, j)-path with no internal vertices; so the path is (i, j).
- d_{ij}^{n} := The weighted length of a shortest (i, j)-path in G.
- $d_{ii}^{n} := 0$.

The input for the algorithm is an $n \times n$ matrix $W(G) = [\mathcal{W}_{ij}]$, called the ***weighted matrix***, where

$$\mathcal{W}_{ij} = \begin{cases} 0, & \text{if } i = j, \\ \mathcal{W}(i, j), & \text{if } i \neq j \text{ and } i, j \text{ are adjacent}, \\ \infty, & \text{if } i \neq j \text{ and } i, j \text{ are non-adjacent}. \end{cases}$$

The output is the $n \times n$ matrix $[d_{ij}^{n}]$, whose (i, j)th entry is the length of a shortest (i, j)-path.

◦ **Description of Floyd-Warshall algorithm**

Input: A weighted graph $(G, \mathcal{W})$ on vertices 1, 2,..., n.

Step 1: Initial: $D^0 \leftarrow W(G)$

Step 2: Recursion:

for $k = 1$ to n do

for $i = 1$ to n do

for $j = 1$ to n do

$d_{ij}^k \leftarrow \min\{d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1}\}$.

Step 3: Output $D^n = [d_{ij}^n]$.

Remarks.

- An advantage of the algorithm is that it requires very little knowledge of program coding.
- The algorithm can be modified to output a shortest (x, y)-path; see exercises.

We again take the graph shown in Fig. 2.17 and illustrate the algorithm with vertices labeled $a = 1$, $b = 2$, $c = 3$, $d = 4$ and $e = 5$. In the following, we show the sequence of matrices D^0, D^1, D^2, D^3, D^4, D^5 successively generated by the algorithm.

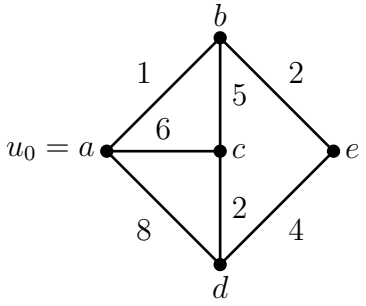


Figure 2.19: The input graph G.

Input matrix $\mathcal{W}(G) = D^0 = [d_{ij}^0]$.

	a	b	c	d	e
$a=1$	0	1	6	8	∞
$b=2$	1	0	5	∞	2
$c=3$	6	5	0	2	∞
$d=4$	8	∞	2	0	4
$e=5$	∞	2	∞	4	0

Output matrix $D^1 = [d^1_{ij}]$ **after one** (i,j)**-loop**, where $d^1_{ij} = \min\{d^0_{ij}, d^0_{i1} + d^0_{1j}\}$ indicates the length of a shortest (i,j)-path with internal vertices from $\{1 = a\}$.

	a	b	c	d	e
$a=1$	0	1	6	8	∞
$b=2$	1	0	5	9	2
$c=3$	6	5	0	2	∞
$d=4$	8	9	2	0	4
$e=5$	∞	2	∞	4	0

Output matrix after 2^{nd} (i,j) **loop**, where $d^2_{ij} = \min\{d^1_{ij}, d^1_{i2} + d^1_{2j}\}$ indicates the length of a shortest (i,j)-path with internal vertices from $\{1 = a, 2 = b\}$.

	a	b	c	d	e
$a=1$	0	1	6	8	3
$b=2$	1	0	5	9	2
$c=3$	6	5	0	2	7
$d=4$	8	9	2	0	4
$e=5$	3	2	7	4	0

Output matrix after the 3^{rd} (i,j) **loop**, $D^3 = [d^3_{ij}]$, where $d^3_{ij} = \min\{d^2_{ij}, d^2_{i3} + d^2_{3j}\}$ indicates the length of a shortest (i,j)-path with internal vertices from $\{1 = a, 2 = b, 3 = c\}$.

	a	b	c	d	e
$a=1$	0	1	6	8	3
$b=2$	1	0	5	7	2
$c=3$	6	5	0	2	7
$d=4$	8	7	2	0	4
$e=5$	3	2	7	4	0

Output matrix after the 4^{th} (i,j) **loop**, $D^4=[d^4_{ij}]$, where $d^4_{ij}=\min\{d^3_{ij}, d^3_{i4}+d^3_{4j}\}$ indicates the length of a shortest (i,j)-path with internal vertices from $\{1=a, 2=b, 3=c, 4=d\}$.

	a	b	c	d	e
$a=1$	0	1	6	8	3
$b=2$	1	0	5	7	2
$c=3$	6	5	0	2	6
$d=4$	8	7	2	0	4
$e=5$	3	2	6	4	0

Output matrix after the 5^{th} (i,j) **loop,** $D^5=[d^5_{ij}]$, where $d^5_{ij}=\min\{d^4_{ij}, d^4_{i4}+d^4_{4j}\}$ indicates the length of a shortest (i,j)-path with internal vertices from $\{1=a, 2=b, 3=c, 4=d, 5=e\}$ = The length of a shortest (i,j)-path.

	a	b	c	d	e
$a=1$	0	1	6	7	3
$b=2$	1	0	5	6	2
$c=3$	6	5	0	2	6
$d=4$	7	6	2	0	4
$e=5$	3	2	6	4	0

Exercises

1. Draw 3 mutually non-isomorphic graphs on 6 vertices and 5 edges, which do not contain cycles.

2. Give an example of a 3-regular graph on 10 vertices in which the minimum length of a cycle is 5.

3. Show that any two longest paths in a connected graph have a vertex in common. Does this assertion hold for a disconnected graph?

4. If C is a cycle in G, then an edge of G which joins two non-consecutive vertices of C is a called a ***chord*** of C. For example, $(2, 5)$ is a chord of the 5-cycle $(1, 2, 3, 4, 5, 1)$. Show that if G has an odd cycle then it has an odd cycle without chords.

5. Let G be a triangle-free graph and let C be a cycle of minimum length in G. Show that every vertex in $V(G) - V(C)$ is adjacent with at most two vertices of C.

6. If G is simple, connected, incomplete and $n \geq 3$, then show that G has three vertices u, v, w such that $(u, v), (v, w) \in E(G)$ but $(u, w) \notin E(G)$.

7. Give an example of a graph G such that
   (a) every two adjacent vertices lie on a common cycle,
   (b) there exists two adjacent edges that do not lie on a common cycle.

8. Let G be a graph. Define a relation R on $V(G)$ as follows. If $u, v \in V(G)$, then $u \ R \ v$ iff there exists a path connecting u and v. Show that R is an equivalence relation on G. What are the induced subgraphs $[V_1], [V_2], \ldots, [V_p]$, where $V_1, V_2, \ldots, V_p$ are the equivalence classes?

9. If G is simple and $\delta(G) \geq \frac{n-1}{2}$, then show that G is connected. Give an example of a disconnected simple graph G on 8 vertices with $\delta(G) = 3$.

10. If $m > \binom{n-1}{2}$ and G is simple, then show that G is connected. For each $n \geq 1$, find a disconnected graph with $m = \binom{n-1}{2}$.

11. If $deg(u) + deg(v) \geq p - 1$, for every pair of non-adjacent vertices in a simple graph G , then show that (i) G is connected, (ii) $diam(G) \leq 2$ and (iii) G has no cut-vertices.

12. Let G be a simple graph on vertices $v_1, v_2, \ldots, v_n$. Show that

(i) If every $G - v_i$ is disconnected, then G is also disconnected.

(ii) If $n \geq 3$ and $G - v_i$, $G - v_j$ are connected, for some $i, j \in \{1, 2, \ldots, n\}$, $i \neq j$, then G is connected. Does this hold if $n = 2$?

13. Show the following for a connected graph G:
    (i) $c(G - v) \leq deg(v)$, for every $v \in V(G)$.
    (ii) If every vertex in G is of even degree, then $c(G - v) \leq \frac{1}{2}deg(v)$, for every $v \in V(G)$.

14. Let G ba a 2-connected simple graph. Let H be the simple graph obtained from G by adding a new vertex y and joining it with at least 2 vertices of G. Show that H is 2 connected.

15. Show that if G contains a closed walk of odd length, then it contains a cycle.

16. Show that G is connected iff every entry in $I + A + A^2 + \cdots + A^{n-1}$ is non-zero.

17. Find the maximum number of edges in a simple graph G which has girth 6.

18. Show that a k-regular simple graph with girth four has at least $2k$ vertices; draw such a graph on $2k$ vertices.

19. Show that a k-regular simple graph of girth five has at least $k^2 + 1$ vertices. In addition, if G has diameter two, then show that it has exactly $k^2 + 1$ vertices. Draw such a graph on $2k$ vertices.

20. Prove or disprove: If G is a connected graph with cut-vertices and if u and v are vertices of G such that $d(u, v) = diam(G)$, then no block of G contains both u and v.

21. Show that every graph with a cut-vertex has at least two end-blocks. (A block H of G is called an **end-block** if it contains exactly one cut-vertex of G.)

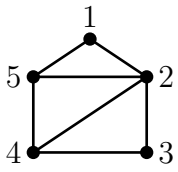
22. Draw the following simple graphs:
    (a) A graph on 10 vertices with 3 components that has maximum number of edges.
    (b) A connected graph (on at most 10 vertices) which has exactly 3 cut-vertices and exactly 2 cut-edges.

23. Show that every k-connected graph G has at least $\frac{nk}{2}$ edges.

24. If G is a 2-connected graph, then show that for some $(x, y) \in E(G)$, $G - x - y$ is connected.

25. If the average degree of a connected graph G is greater than two, prove that G has at least two cycles.

26. Draw connected spanning subgraphs H_1, H_2 and H_3 of the graph G (shown below) having diameters 2,3 and 4 respectively and with minimum number of edges.



G

27. Give an example of a graph with the following properties or explain why no such example exists.
    (i) A 4-connected graph that is not 3-connected.
    (ii) A 3-connected graph that is not 4-connected.
    (iii) A 2-edge-connected graph that is not 3-edge-connected.
    (iv) A 3-edge-connected graph that is not 2-edge-connected.

28. (a) If G is disconnected, then show that G^c is connected. Is the converse true?
    (b) If G is simple, show that G or G^c has diameter ≤ 3.
    (c) If G is self-complimentary, show that $2 \leq diam(G) \leq 3$.

29. For any two graphs G_1 and G_2, show the following.
    (a) $k_0(G_1 \cup G_2) = 0$.
    (b) $k_0(G_1 + G_2) = \min\{k_0(G_1) + n(G_2), k_0(G_2) + n(G_1)\}$.
    (c) $k_1(G_1 \cup G_2) = 0$.
    (d) $k_1(G_1 + G_2) = \delta(G_1 + G_2)$.

30. Give an example of a graph G with the following properties or state why such examples do not exist.
    (a) $k_0(G) = 3$, $k_1(G) = 4$ and $\delta(G) = 5$

(b) $k_0(G) = 2$, $k_1(G) = 2$ and $\delta(G) = 4$

(c) $k_0(G) = 3$, $k_1(G) = 3$ and $\delta(G) = 2$

(d) $k_0(G) = 3$, $k_1(G) = 2$ and $\delta(G) = 4$

31. Prove or disprove:
    (i) If G is a graph with $k_0(G) = k \geq 1$, then $G - U$ is disconnected, for every set U of k vertices.
    (ii) If G is a connected graph and U is a minimum vertex-cut, then $G - U$ contains exactly two components.
    (iii) If G is a graph on n vertices and $W = \{u \in V(G) : deg(u) = n - 1\}$, then
        (a) $W \subseteq K(G)$, for every vertex-cut $K(G)$ of G.
        (b) every edge-cut of G contains an edge incident with a vertex in W.

32. Find the connectivity and edge-connectivity of the d-cube Q_d.

33. Prove that if G is a simple r-regular graph, where $0 \leq r \leq 3$, then $k_0(G) = k_1(G)$. Does it hold for multi-graphs? Find the minimum positive integer r for which there exists an r-regular graph G such that $k_0(G) \neq k_1(G)$.

34. Find the minimum positive integer r for which there exists a r-regular graph such that $k_1(G) \geq k_0(G) + 2$.

35. If G is simple and $\delta(G) \geq n - 2$, then show that $k_0(G) = \delta(G)$.

36. If G is simple and $\delta(G) \geq \frac{n}{2}$, then show that $k_1(G) = \delta(G)$.

37. If G is a simple graph with $\delta(G) \geq \frac{n+1}{2}$, then show that G is 3-vertex connected.

38. If G is simple and has no even cycles, then show that each block of G is either K_2 or an odd cycle.

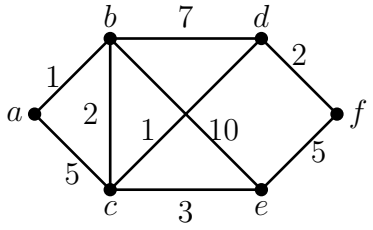
39. Draw a network N with 7 nodes and with minimum number of links satisfying the following conditions. Justify that your network has minimum number of links.
    (i) N contains no self-loops and multiple links.
    (ii) If any two nodes fail, the remaining 5 healthy nodes can communicate along themselves.

40. Let G be an incomplete simple graph on n vertices with vertex connectivity k. If $deg(v) \geq \frac{1}{3}(n + 2k - 2)$, for every vertex v, and S is a vertex-cut with k vertices, then find the number of components in $G - S$.

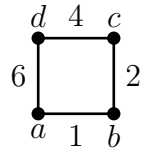
41. If G is a connected graph, then G^2 has vertex set V(G) and two vertices u, v are adjacent in G^2 iff the distance between u and v in G is 1 or 2.
    (a) Draw $(P_6)^2$, where P_6 is the path on 6 vertices.
    (b) If G is connected, show that G^2 is 2-connected.

42. Let G be a simple graph with exactly one cycle. Find all possible values of the vertex connectivity and edge connectivity of G. Justify your answer.

43. Use Dijkstra's algorithm to compute a shortest path from the vertex a to every other vertex in the edge weighted graph shown below. Show the weighted paths generated at each step of the algorithm.



44. Find the entry in the first row and second column of the matrix generated after applying one iteration of the Floyd-Warshall algorithm to the weighted graph shown below.



45. If $k_0(G) \geq 3$, then show that $k_0(G - e) \geq 2$, for every edge e.