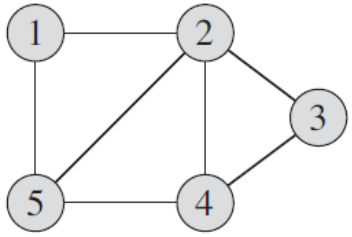
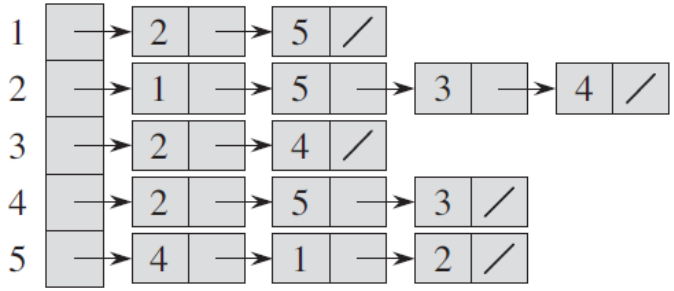
Graph Algorithms

Representations of graphs



(a)

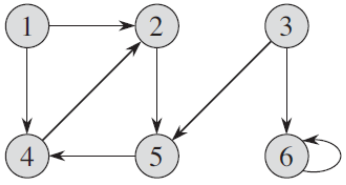


(b)

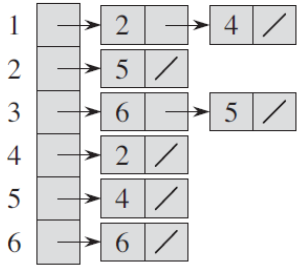
	1	2	3	4	5
1	0	1	0	0	1
2	1	0	1	1	1
3	0	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

(c)

Figure 22.1 Two representations of an undirected graph. **(a)** An undirected graph G with 5 vertices and 7 edges. **(b)** An adjacency-list representation of G. **(c)** The adjacency-matrix representation of G.



(a)



(b)

	1	2	3	4	5	6
1	0	1	0	1	0	0
2	0	0	0	0	1	0
3	0	0	0	0	1	1
4	0	1	0	0	0	0
5	0	0	0	1	0	0
6	0	0	0	0	0	1

(c)

Figure 22.2 Two representations of a directed graph. **(a)** A directed graph G with 6 vertices and 8 edges. **(b)** An adjacency-list representation of G. **(c)** The adjacency-matrix representation of G.

Breadth-first search

- Breadth-first search is one of the simplest algorithms for searching a graph and the archetype for many important graph algorithms.
- Prim’s minimum-spanning tree algorithm and Dijkstra’s single-source shortest-paths algorithm use ideas similar to those in breadth-first search.

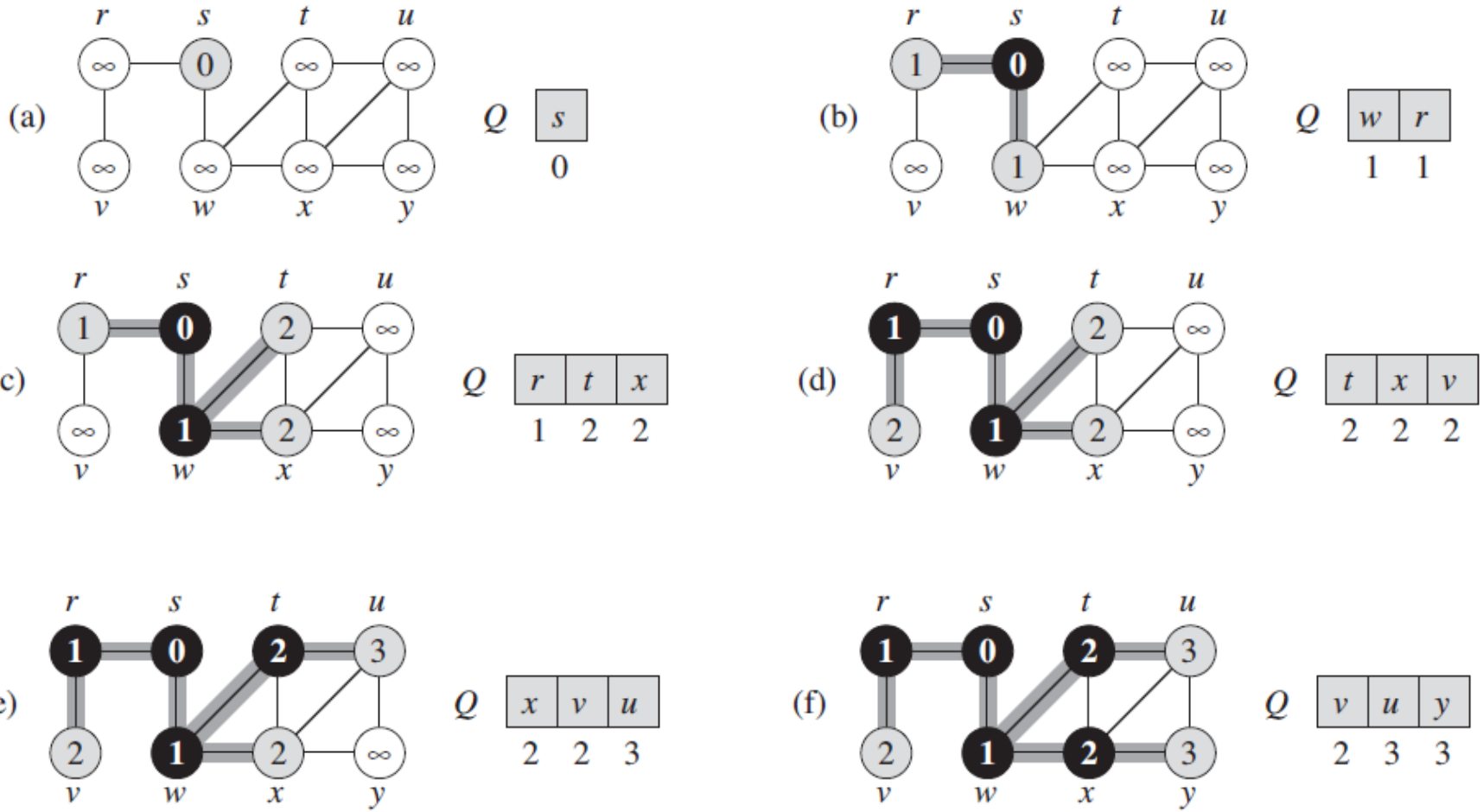
Breadth-first search

```
BFS(G, s)
O(V)  1  for each vertex u ∈ G.V − {s}
      2      u.color = WHITE
      3      u.d = ∞
      4      u.π = NIL
      5  s.color = GRAY
      6  s.d = 0
      7  s.π = NIL
      8  Q = ∅
      9  ENQUEUE(Q, s)        O(1)
     10  while Q ≠ ∅
     11      u = DEQUEUE(Q)   O(1)
     12      for each v ∈ G.Adj[u]   Θ(E)
     13          if v.color == WHITE
     14              v.color = GRAY
     15              v.d = u.d + 1
     16              v.π = u
     17          O(1)  ENQUEUE(Q, v)
     18      u.color = BLACK
```

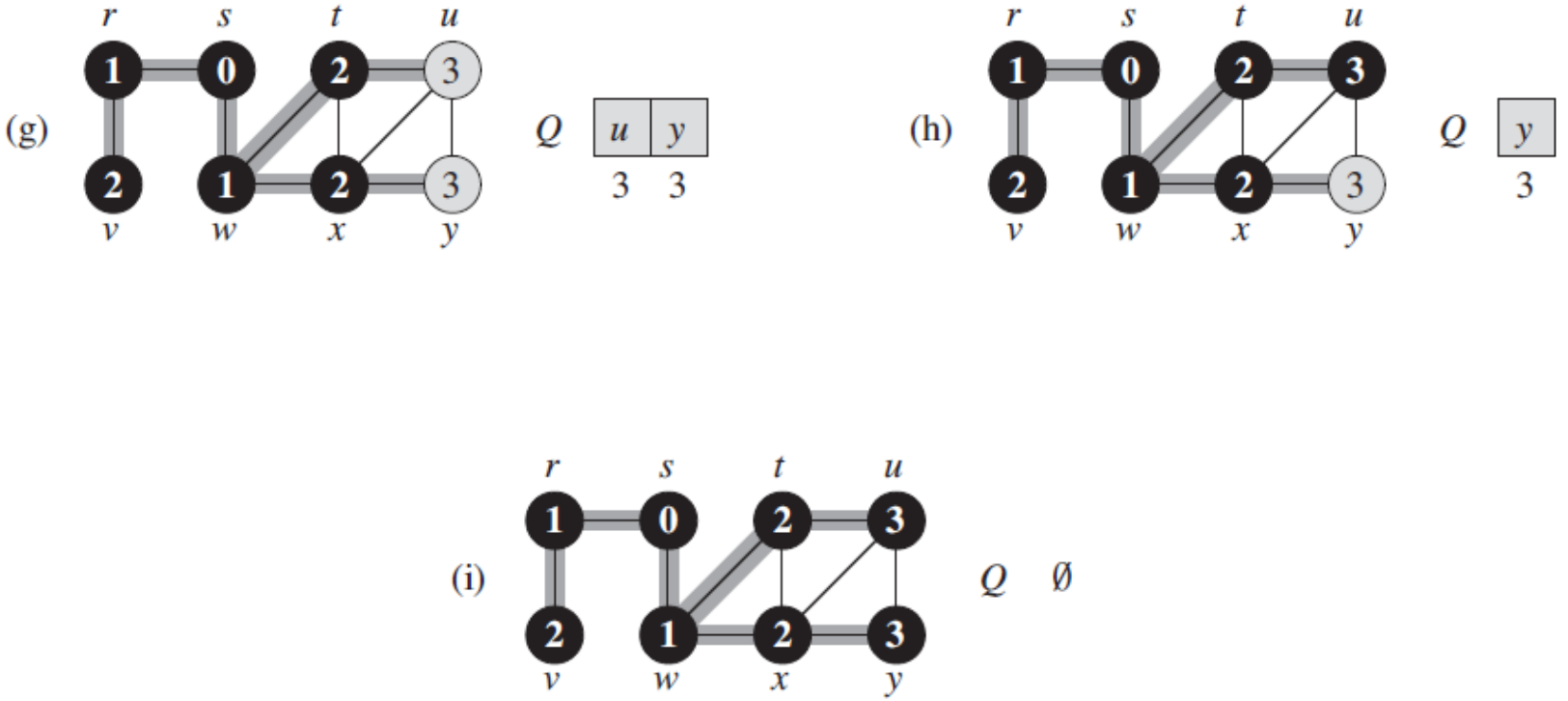
$O(V + E)$

The overhead for initialization is O(V) +The sum of the lengths of all the adjacency lists is E, the total time spent in scanning adjacency lists is O(E)

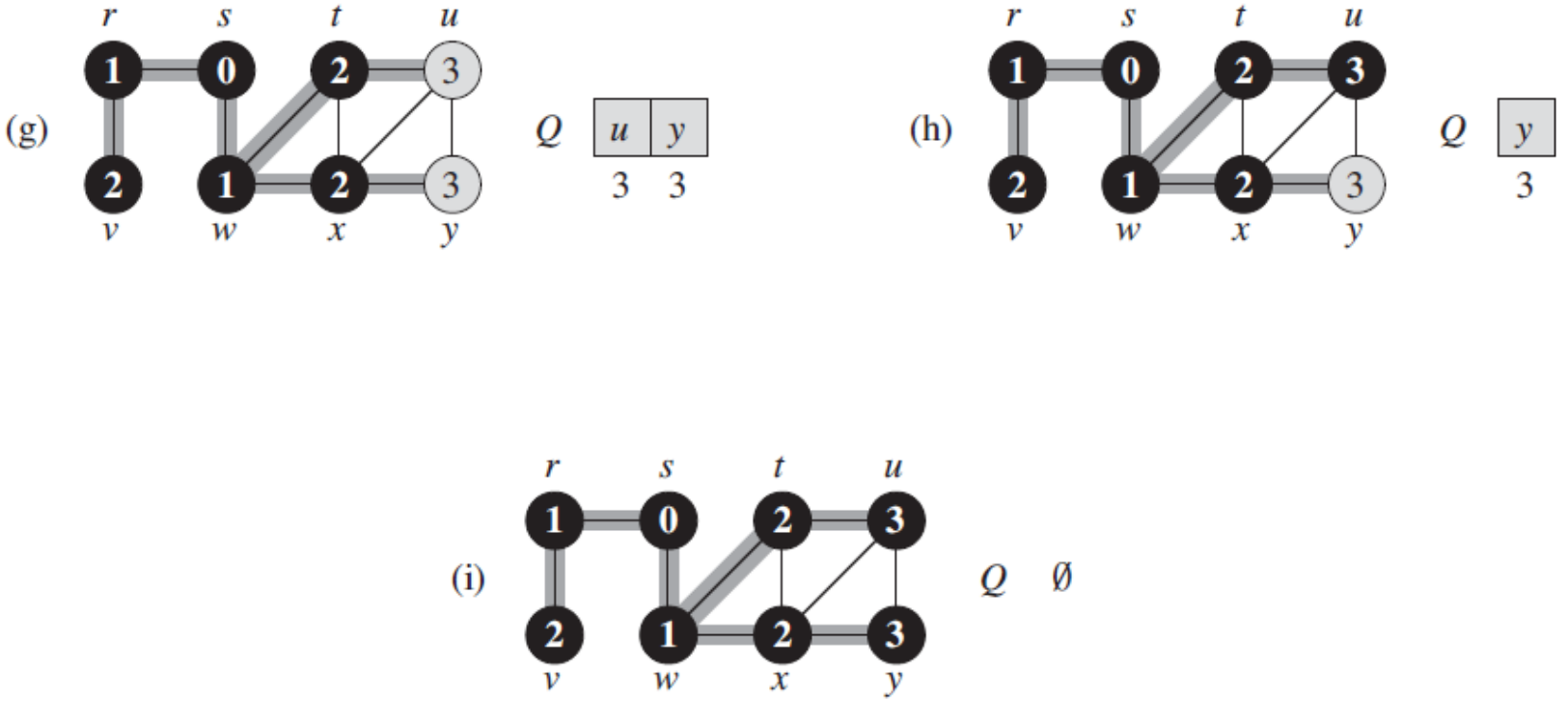
Breadth-first search



Breadth-first search



Breadth-first search



Depth-first search

- The strategy followed by depth-first search is, as its name implies, to search “deeper” in the graph whenever possible.

```
DFS(G)
1  for each vertex u ∈ G.V                    Θ(V)
2      u.color = WHITE
3      u.π = NIL
4  time = 0
5  for each vertex u ∈ G.V                    Θ(V)
6      if u.color == WHITE
7          DFS-VISIT(G, u)
```

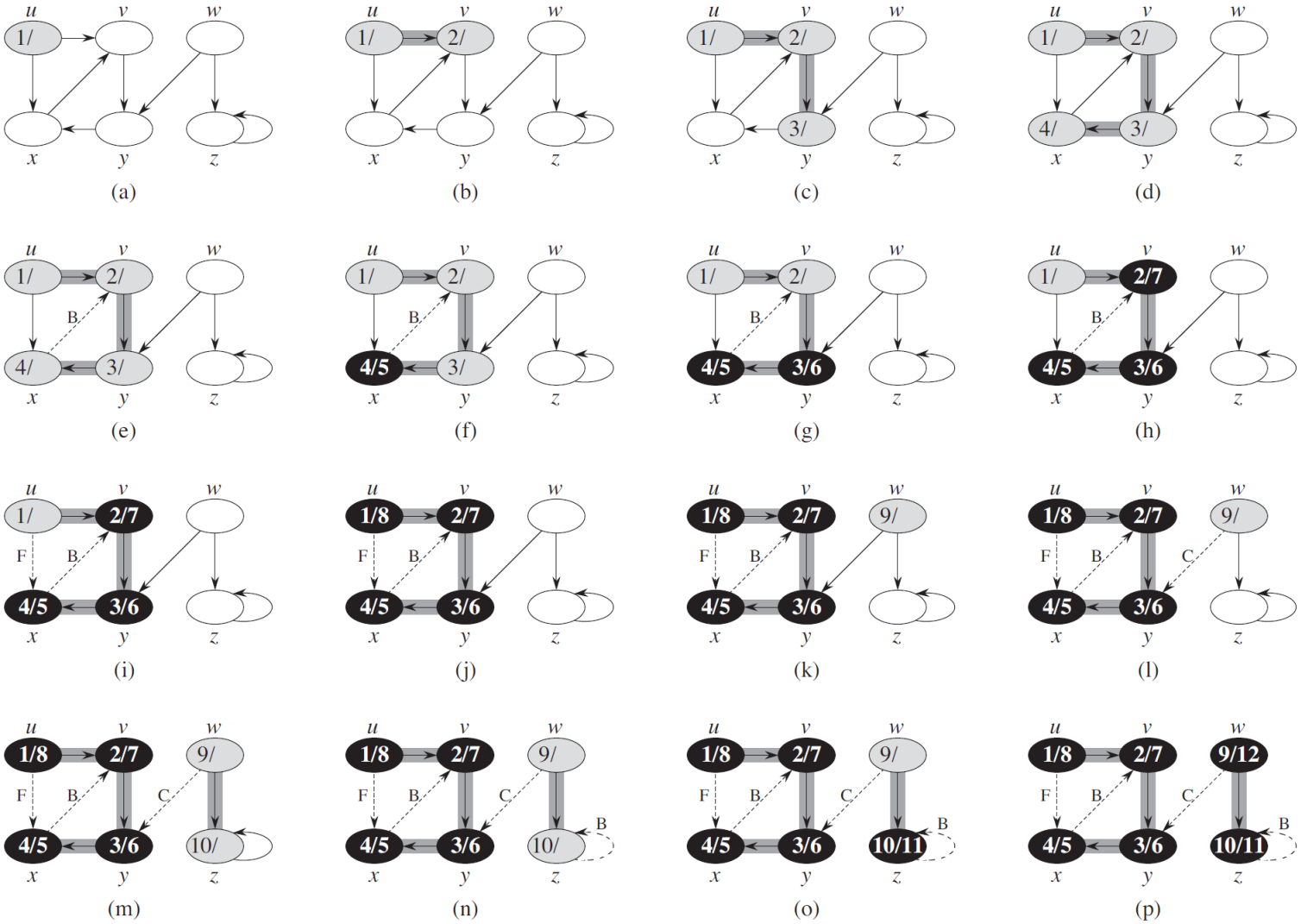
Depth-first search

```
DFS-VISIT(G, u)
 1  time = time + 1                  // white vertex u has just been discovered
 2  u.d = time
 3  u.color = GRAY
 4  for each v ∈ G.Adj[u]            // explore edge (u, v)
 5      if v.color == WHITE
 6          v.π = u
 7          DFS-VISIT(G, v)
 8  u.color = BLACK                  // blacken u; it is finished
 9  time = time + 1
10  u.f = time
```

$$\sum_{v \in V} |Adj[v]| = \Theta(E)$$

$$\Theta(V + E)$$

Depth-first search



Thank you