Tarea No. 3

Tomás Balderas Contreras
Instituto Nacional de Astrofísica, Óptica y Electrónica
Curso: Estructuras de Datos

26 de julio, 2002

1. ¿Qué es un *grafo*? ¿Qué es un *camino*? ¿Qué es un *grafo fuertemente conexo*? ¿Qué es un *grafo acíclico*?

   Solución:

   Un *grafo no dirigido* $G = (V, E)$ consiste de un conjunto V de vértices y un conjunto E de arcos. Cada arco $e \in E$ está asociado a un único par no ordenado de vértices u y v ($e = (u, v)$ ó $e = (v, u)$). Si G es un *grafo dirigido* cada arco $e \in E$ está asociado a un par ordenado de vértices u y v ($e = (u, v)$).

   Un *camino* de longitud n de v_0 a v_n es una sucesión de n arcos distintos entre sí

   $$\{(v_0, v_1), (v_1, v_2), \ldots, (v_{n-1}, v_n)\}$$

   lo que puede abreviarse simplemente como la secuencia de vértices

   $$(v_0, v_1, v_2, \ldots, v_{n-1}, v_n).$$

   Un grafo dirigido es *fuertemente conexo* si es posible encontrar un camino entre cualesquiera dos de sus vértices $u, v \in V$.

   Un *ciclo* es un camino que parte de un vértice v y llega a v. Un grafo que no contiene ciclos es un *grafo acíclico*.

2. Explique como representar un grafo a través de

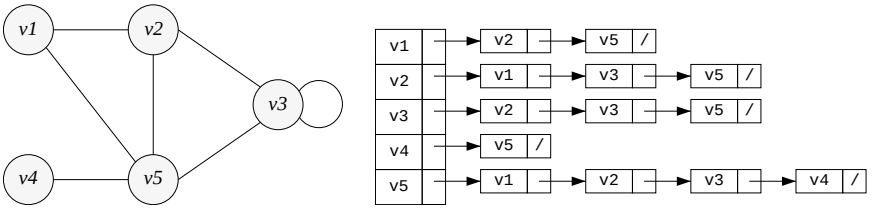
   (a) una lista de adyacencia

   (b) una matriz de adyacencia.

   Explique las ventajas de cada representación.

   Solución:

   (a) Una lista de adyacencia para un grafo $G = (V, E)$ es un arreglo de longitud $|V|$ donde cada entrada representa un vértice $v \in V$, dicha entrada es además un apuntador a una lista cuyos nodos representan los vértices del grafo que son adyacentes a v.



(a) (b)

$$
\begin{array}{c} \\ \mathbf{V1} \\ \mathbf{V2} \\ \mathbf{V3} \\ \mathbf{V4} \\ \mathbf{V5} \end{array}
\begin{array}{c}
\begin{array}{ccccc} \mathbf{V1} & \mathbf{V2} & \mathbf{V3} & \mathbf{V4} & \mathbf{V5} \end{array} \\
\left(\begin{array}{ccccc}
0 & 1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 & 0
\end{array} \right)
\end{array}
$$

(c)

Figura 1: Un grafo no dirigido y sus diferentes representaciones.

Una ventaja de esta forma de representación es su bajo costo en términos de espacio de almacenamiento cuando el grafo G contiene pocos arcos.

(b) En un grafo G que tiene $n = |V|$ vértices, es posible construir la matriz de adyacencia $A_{n \times n} = (a_{ij})$ de la siguiente forma

$$a_{ij} = \begin{cases} 1 & \text{si } v_i \text{ es adyacente a } v_j \\ 0 & \text{si } v_i \text{ no es adyacente a } v_j. \end{cases}$$

En un grafo no dirigido se cumple que $A = A^T$ (A es simétrica) y, por lo tanto, es posible almacenar únicamente los valores almacenados en la diagonal de A y los elementos situados por encima de la diagonal, con la finalidad de ahorrar espacio. Además esta forma, en extremo simple, de representación es adecuada para determinar rápidamente si dos vértices son adyacentes.

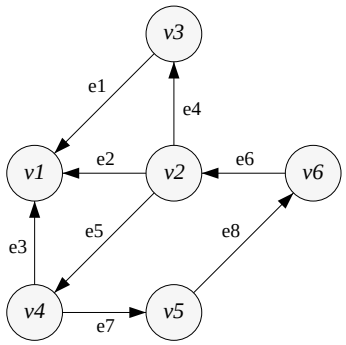
La figura 1a muestra un grafo no dirigido, la figura 1b ilustra su representación como una lista de adyacencia y la figura 1c la representación del grafo como una matriz de adyacencia.

3. Dibuje el grafo correspondiente a la siguiente *matriz de incidencia*.

$$A = \begin{array}{c} \\ \mathbf{V1} \\ \mathbf{V2} \\ \mathbf{V3} \\ \mathbf{V4} \\ \mathbf{V5} \\ \mathbf{V6} \end{array} \begin{array}{c} \begin{array}{cccccccc} \mathbf{E1} & \mathbf{E2} & \mathbf{E3} & \mathbf{E4} & \mathbf{E5} & \mathbf{E6} & \mathbf{E7} & \mathbf{E8} \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & -1 & -1 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix} \end{array}$$

Solución:

El grafo dirigido correspondiente a la matriz de incidencia A es el siguiente.



4. Escriba el pseudocódigo para calcular el *grado de salida* (*out-degree*) de cada vértice de un grafo representado por una matriz de incidencia.

   Solución:

   El algoritmo 1 ilustra un procedimiento simple para determinar el grado de salida de cada vértice en un grafo representado por su matriz de incidencia. El algoritmo recibe como entrada una matriz de incidencia $A_{m \times n}$, donde m es el número de vértices en el grafo y n es el número de arcos. El parámetro d es un arreglo de m contadores tal que al final del algoritmo $d[i]$ es el grado de salida del vértice v_i.

   Obviamente este algoritmo es $O(m \times n)$ y es ineficiente para valores de m y n arbitrariamente grandes.

5. Explique en qué consisten la *búsqueda primero en anchura* (*breadth-first search*) y la búsqueda primero en profundidad (*depth-first search*).

   Solución:

   Los algoritmos de búsqueda primero en anchura y de búsqueda primero en profundidad tienen un mismo objetivo, recorrer los arcos que forman un grafo y visitar o descubrir los vértices del mismo. Los métodos, sin embargo, difieren en el mecanismo empleado para recorrer los arcos.

Algoritmo 1 OUT_DEGREE(A, m, n, d)

```
for i ← 1 to m do
  d[i] ← 0;
  for j ← 1 to n do
    if A[i, j] = −1 then
      d[i] ← d[i] + 1;
    end if
  end for
end for
return d;
```

El algoritmo de recorrido primero en anchura en un grafo G se asegura de descubrir todos los vértices adyacentes a un vértice v antes de descubrir algún otro vértice que sea, a su vez, adyacente a alguno de los vértices adyacentes a v; expandiendo el espacio de búsqueda de vértices a todo lo ancho del grafo. Este algoritmo produce un *árbol primero en anchura* que tiene la propiedad de que la ruta entre la raíz s y cualquier vértice u del árbol contiene el mínimo número de arcos desde s a u en G.

El algoritmo de recorrido primero en profundidad generalmente se diseña de forma recursiva. El algoritmo se interna por un camino del grafo, partiendo de un vértice v, hasta llegar a un vértice sin vértices adyacentes o con vértices adyacentes ya visitados; en este momento regresa, mediante *backtracking*, para continuar por otra dirección en algún vértice anterior que aún tenga vértices adyacentes sin visitar. El procedimiento anterior se puede repetir para otros vértices iniciales que no hayan sido visitados previamente. Como resultado tenemos un *bosque primero en profundidad* que consta de varios *árboles primero en profundidad*, usualmente muy profundos.

6. Muestre gráficamente, paso a paso, la búsqueda en anchura sobre el grafo correspondiente a la matriz de incidencia del ejercicio 3, iniciando la búsqueda en el vértice 1.

   Solución:

   Como en el grafo dirigido solicitado no hay arcos que salgan de v_1 se presentan las soluciones a otros dos problemas. Primero, se ilustra el proceso de aplicar el algoritmo de recorrido primero en anchura en el grafo no dirigido correspondiente partiendo de v_1. Segundo, se muestra el recorrido en el grafo dirigido pero tomando como vértice inicial a v_2.

   La figuras 2a–2g ilustran el proceso de recorrido primero en anchura del grafo solcitado, considrándolo como un grafo no dirigido y partiendo de v_1. Se muestra la configuración del grafo en cada momento de la evaluación de la condición del lazo principal, esta configuración incluye el color de cada vértice, su distancia al vértice inicial, el contenido de la cola Q y los arcos que conforman el *árbol primero en anchura* (*breadth-first tree*) resultante.

El proceso de recorrido desde el vértice v_2 en el grafo dirigido se muestra en las figuras 3a–3g. También están indicados los valores de la distancia desde v_2 para cada vértice, los arcos que forman parte del árbol primero en anchura y los que no, los colores de los vértices en cada etapa del proceso y el contenido de la cola Q.

7. Muestre gráficamente, paso a paso, la búsqueda en profundidad sobre el grafo correspondiente a la matriz de incidencia del ejercicio 3, iniciando la búsqueda en el vértice 1.

   Solución:

   En las figuras 4a–4k está ilustrado el proceso de recorrido primero en profundidad del grafo dirigido solicitado partiendo del vértice v_1. El algoritmo produce un *bosque primero en profundidad* (*depth-first forest*) que consiste en dos *árboles primeros en profundidad*, uno con raíz en v_1 y el otro con raíz en v_2. Dentro de cada vértice están las marcas (*timestamps*) de los tiempos en que cada vértice fue descubierto y finalizado, los diagramas muestran también tanto los arcos que componen los árboles como los que no forman parte de árbol alguno y, finalmente, se muestran los colores que el algoritmo asigna a los vértices durante el proceso.

   Las figuras 5a–5l ilustran el recorrido en un grafo no dirigido tomándo a v_1 como vértice fuente. En este caso el algoritmo produce un bosque que contiene un único árbol. Se muestran los colores de cada vértice durante el proceso, los arcos del árbol resultante y los arcos del grafo que no forman parte del árbol. Además se indican, dentro de cada vértice, los tiempos de descubrimiento y finalización.

8. Ejecute el algoritmo de Dijkstra sobre el grafo dirigido de la figura 6, utilizando el vértice s como origen. Basándose en la descripción para este algoritmo, muestre gráficamente, paso a paso, el recorrido sobre el grafo indicando los valores para d y π y los vértices en el conjunto S después de cada iteración del ciclo **while**.

   Solución:

   Las figuras 7a–7f ilustran el progreso del algoritmo de Dijkstra sobre el grafo dirigido solicitado. En el interior de cada vértice en las figuras está indicado el valor del *estimador de la ruta mas corta* (atributo d); la relación de predecesor establecida por el atributo π está indicada en las figuras por los arcos sombreados entre los vértices que forman el *árbol de rutas mas cortas*, el contenido del conjunto S y de la cola Q en cada paso del algoritmo se muestran debajo de cada figura. Adicionalmente se colorearon los vértices para diferenciar entre los que pertenecen a S (en negro) y los que están aún en Q (en blanco).

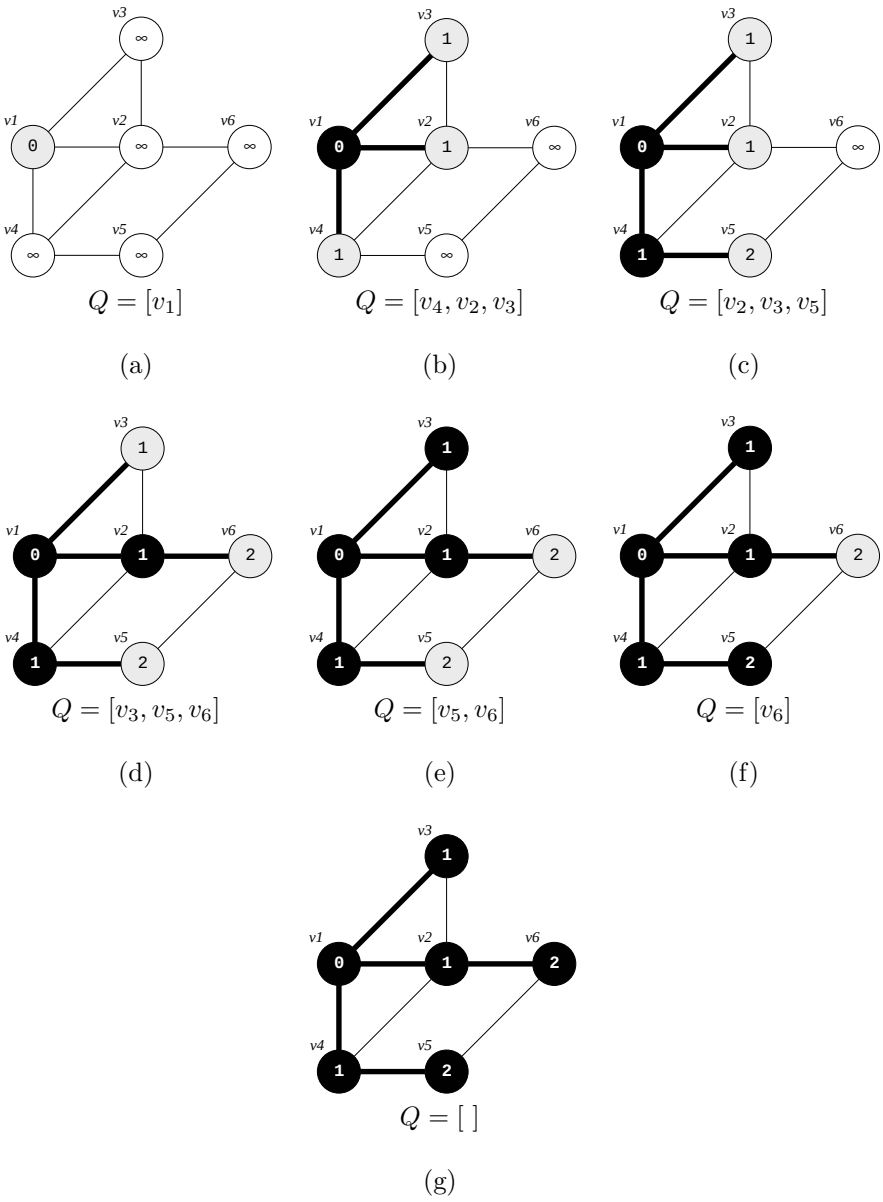


Figura 2: Proceso de recorrido primero en anchura para un grafo no dirigido.

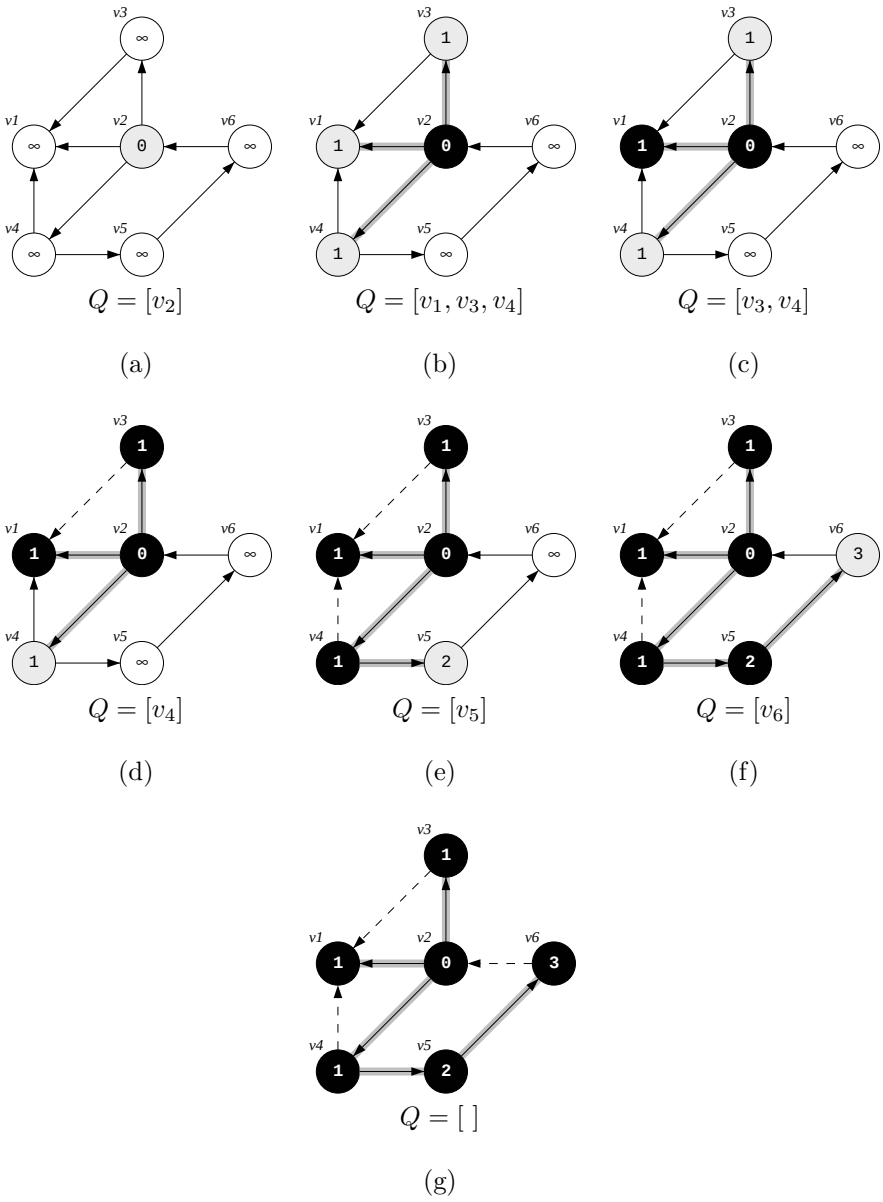


Figura 3: Proceso de recorrido primero en anchura para un grafo dirigido.

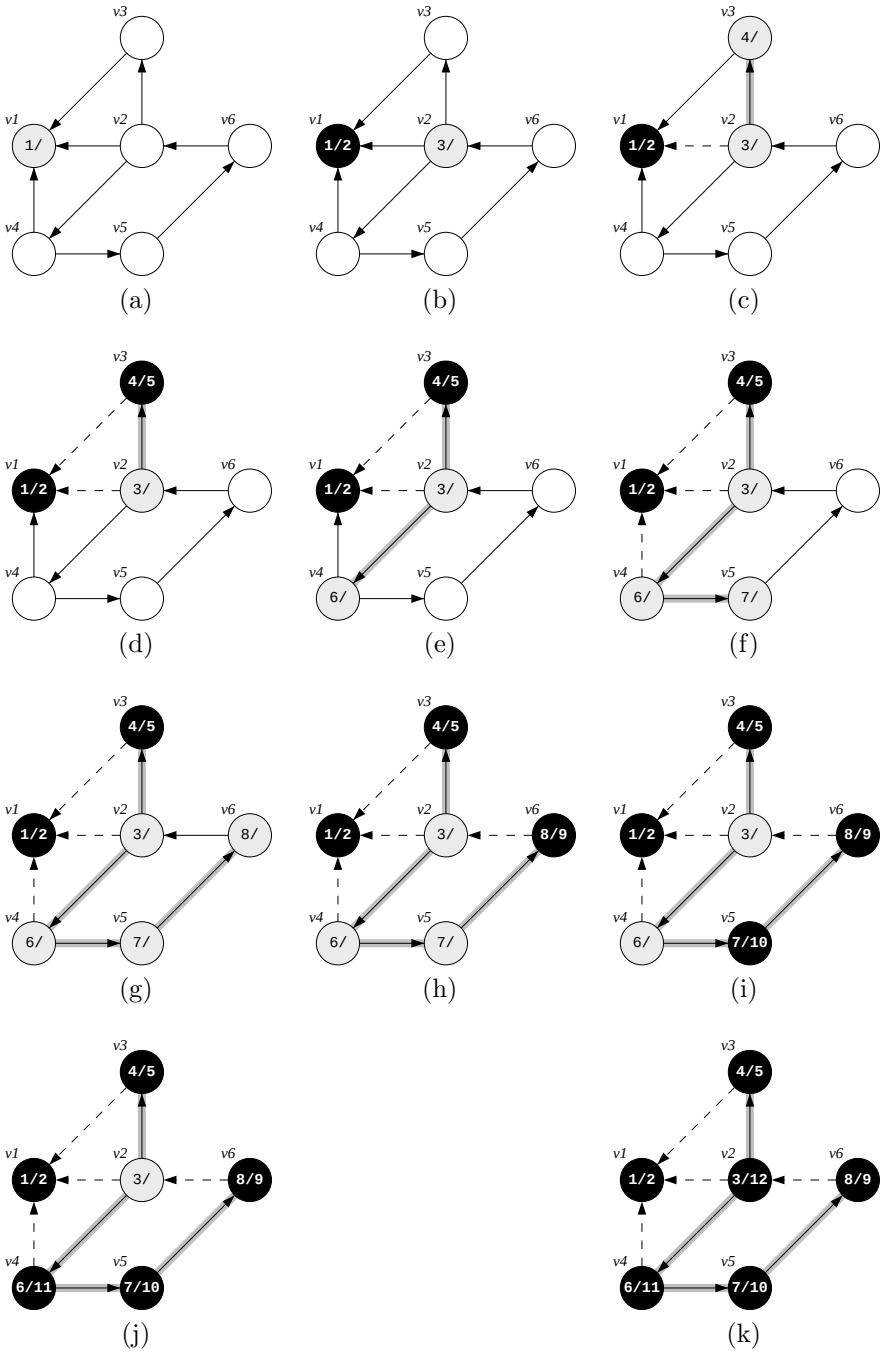


Figura 4: Proceso de recorrido primero en profundidad para un grafo dirigido.

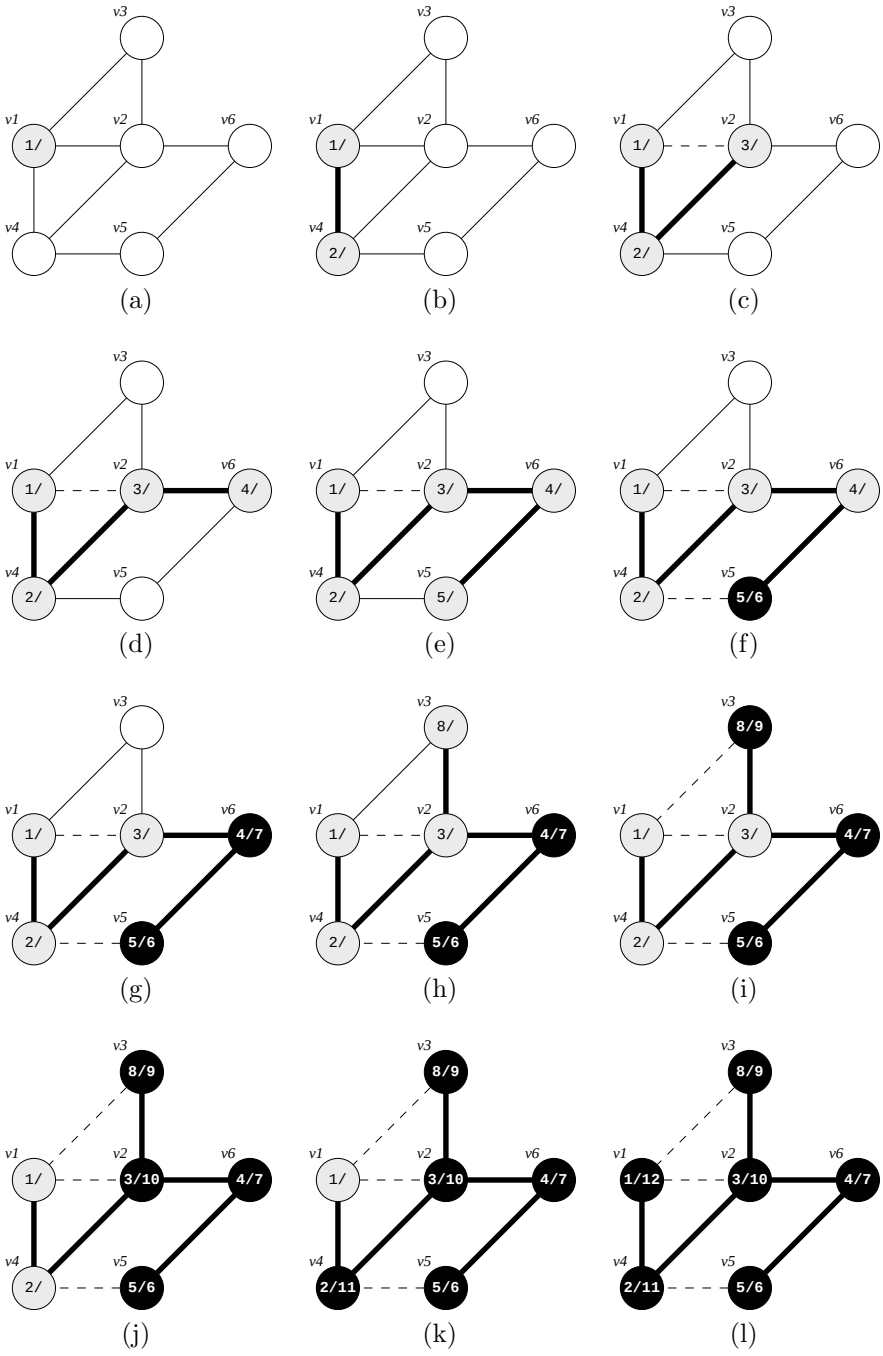


Figura 5: Proceso de recorrido primero en profundidad para un grafo no dirigido.

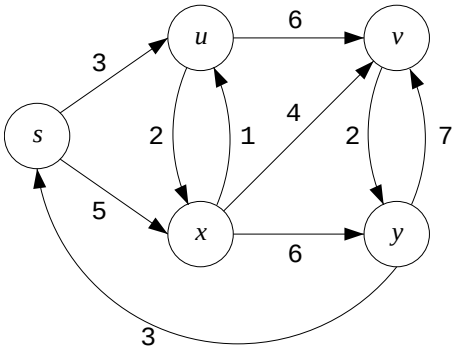


Figura 6: Grafo dirigido para el problema 8.

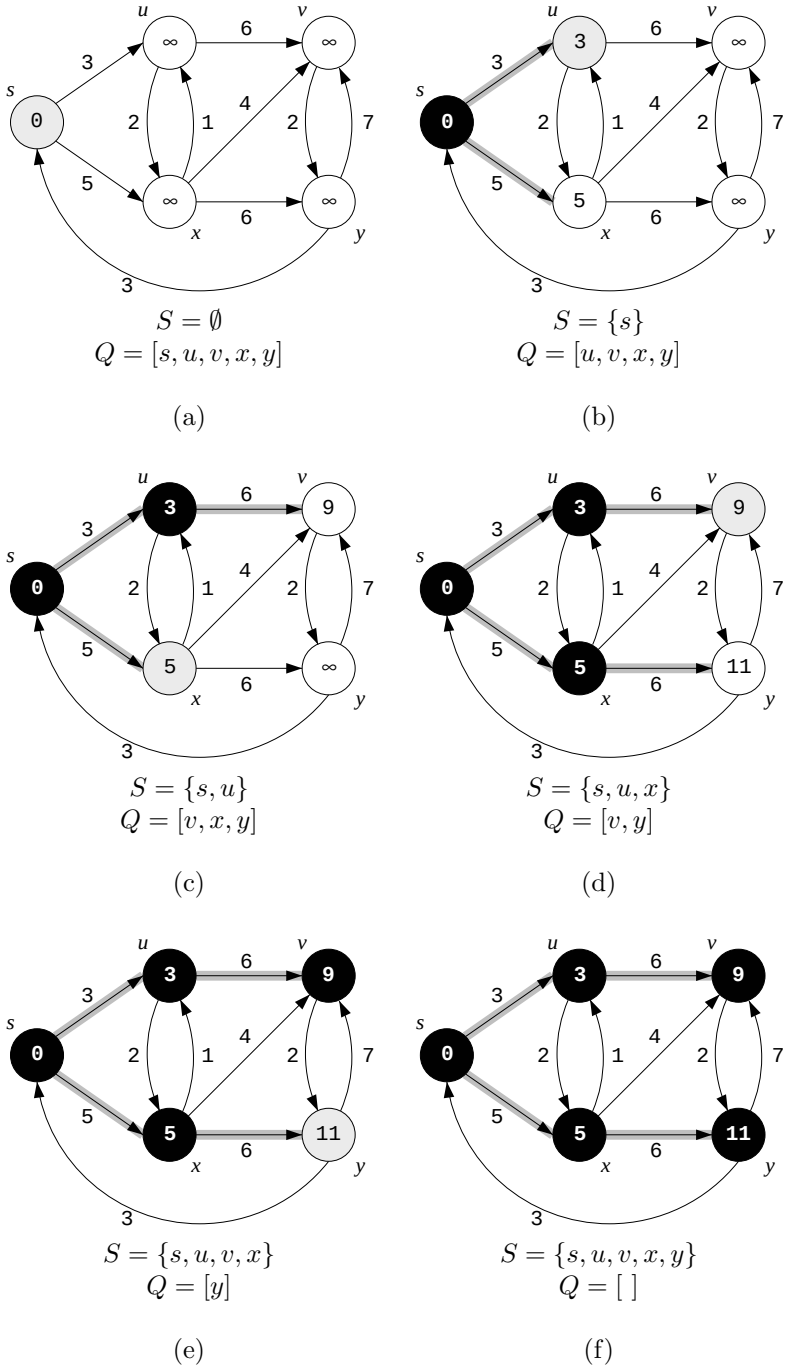


Figura 7: Proceso del algoritmo de Dijkstra.