Computación y Estructuras Discretas I

Andrés A. Aristizábal P.
aaaristizabal@icesi.edu.co

Departamento de Computación y Sistemas Inteligentes

UNIVERSIDAD ICESI

2024-2

Agenda del día

1. **Caminos de longitud mínima**
   - Introducción
   - Algoritmos para calcular caminos de longitud mínima
   - El problema del viajante
   - Ejercicios

Introducción

¿Qué son los grafos ponderados?

Introducción

¿Qué son los grafos ponderados?

- Son aquellos en los que se asigna un número a cada una de las aristas.

Introducción

¿Qué son los grafos ponderados?

- Son aquellos en los que se asigna un número a cada una de las aristas.

¿Para que se utilizan este tipo de grafos?

Introducción

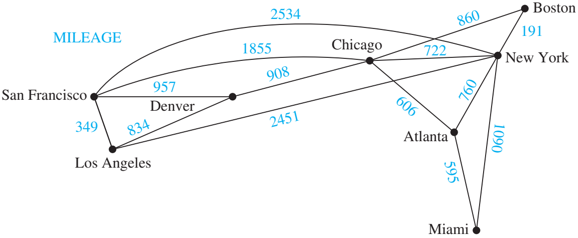
¿Qué son los grafos ponderados?

- Son aquellos en los que se asigna un número a cada una de las aristas.

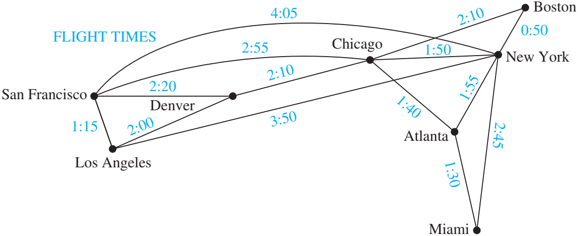
¿Para que se utilizan este tipo de grafos?

- Se utilizan para representar redes informáticas y pueden emplearse para estudiar los costes de comunicación, los tiempos de respuesta de los computadores o la distancia entre ellos.
- También para representar el sistema de vuelos de una línea aérea.

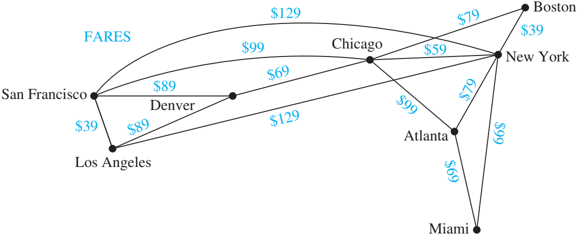
Introducción



Introducción



Introducción



Introducción

¿Cómo es la definición de un grafo simple ponderado?

Introducción

¿Cómo es la definición de un grafo simple ponderado?

- Se dice que G que es un grafo ponderado si tiene asociado una función $W : E \to R$ llamada función de ponderación.
- La imagen de cada arista determinada por los vértices v_i y v_j se llama peso de una arista y se denota por W_{ij}.
- Sea $G = (V, E)$ un grafo ponderado finito tal que $V = \{v_1, \ldots, v_n\}$, se denomina matriz de peso del grafo G a la matriz $n \times n$ $W = [a_{ij}]$ tal que

$$a_{ij} = \begin{cases} w_{ij}, \text{ si } \{v_i, v_j\} \in E, \\ \infty \text{ en caso contrario} \end{cases}$$

- Es decir, se pone el valor del peso cuando lo tenga, y el símbolo infinito cuando no exista tal valor.

Introducción

¿Cuál es uno de los problemas relacionados con grafos ponderados que aparece con mayor frecuencia?

Introducción

¿Cuál es uno de los problemas relacionados con grafos ponderados que aparece con mayor frecuencia?

- El determinar un camino de longitud mínima entre dos vérices de una red.
- Más concretamente, se define la longitud de un camino en un grafo ponderado como la suma de los pesos de las aristas de ese camino.

Introducción

¿Qué puede representar el camino más corto en los ejemplos previos?

Introducción

¿Qué puede representar el camino más corto en los ejemplos previos?

- La distancia más corta en millas entre una ciudad y otra.
- El menor tiempo de viaje entre una ciudad y otra.
- El menor costo de viaje entre ciudades.

Introducción

¿Cuál es otro problema que involucra grafos ponderados?

Introducción

¿Cuál es otro problema que involucra grafos ponderados?

- Es aquel que pide determinar un circuito de longitud total mínima que visite exactamente una vez cada uno de los vértices de un grafo completo.
- Éste es el famoso problema del viajante, que pregunta cuál es el orden en el que un viajante debería visitar exactamente una vez cada una de las ciudades de su ruta de manera que la distancia total recorrida sea mínima.

Agenda del día

1. **Caminos de longitud mínima**
   - Introducción
   - Algoritmos para calcular caminos de longitud mínima
   - El problema del viajante
   - Ejercicios

Algoritmos para calcular caminos de longitud mínima

¿Cuál es el algoritmo de Dijkstra?

Algoritmos para calcular caminos de longitud mínima

¿Cuál es el algoritmo de Dijkstra?

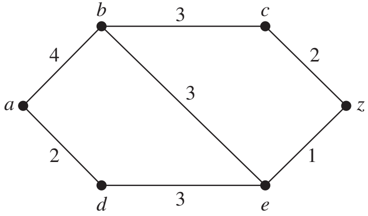
- Es uno de los más importantes algoritmos para hallar un camino de longitud mínima entre dos vértices de un grafo ponderado.
- Fue descubierto por el matemático holandés Edsger Dijkstra en 1959.
- Resuelve este problema para grafos ponderados no dirigidos si todos los pesos son positivos.
- Este algoritmo puede adaptarse fácilmente para resolver problemas de caminos de longitud mínima en grafos dirigidos.

Algoritmos para calcular caminos de longitud mínima

¿Cuál es la longitud del camino más corto entre a y z en el siguiente grafo ponderado?

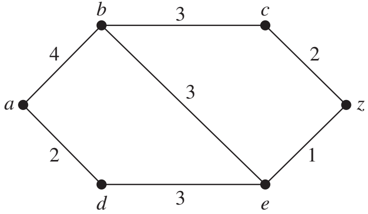
Algoritmos para calcular caminos de longitud mínima

¿Cuál es la longitud del camino más corto entre *a* y *z* en el siguiente grafo ponderado?

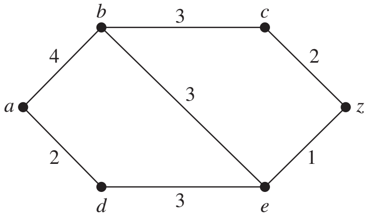


- Aunque puede hallarse fácilmente el camino más corto por simple inspección, vamos a desarrollar algunas ideas que nos serán útiles para entender el algoritmo de Dijkstra.
- Resolveremos este problema hallando la longitud de un camino de longitud mínima de *a* a cada uno de los vértices sucesivos hasta que lleguemos a *z*.

Algoritmos para calcular caminos de longitud mínima

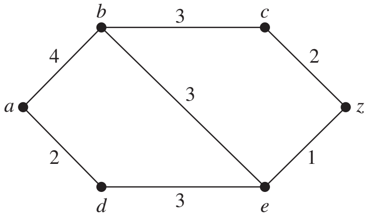


Algoritmos para calcular caminos de longitud mínima



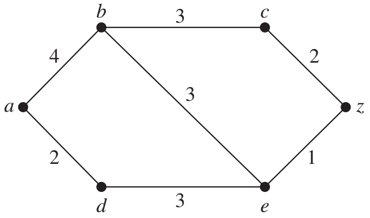
- *a*.

Algoritmos para calcular caminos de longitud mínima



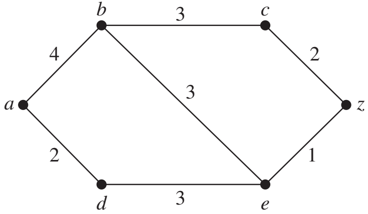
- a. $S = \{a\}$

Algoritmos para calcular caminos de longitud mínima



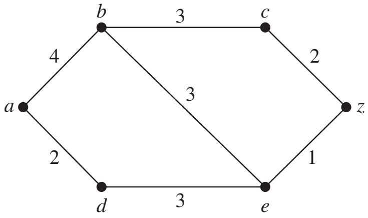
- a. $S = \{a\}$
- b ó d.

Algoritmos para calcular caminos de longitud mínima



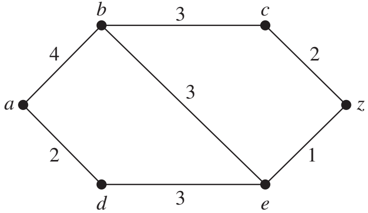
- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$.

Algoritmos para calcular caminos de longitud mínima



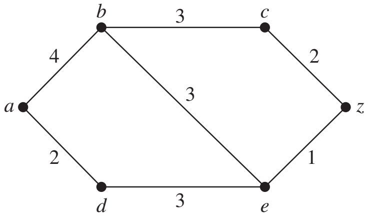
- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$

Algoritmos para calcular caminos de longitud mínima



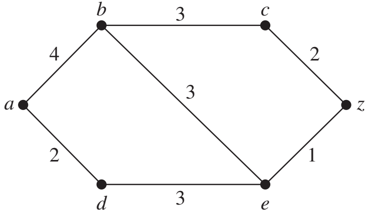
- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$
- b ó e.

Algoritmos para calcular caminos de longitud mínima



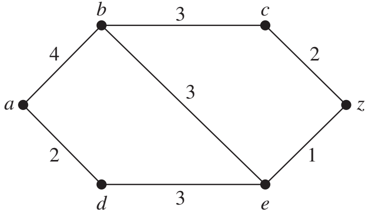
- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$
- b ó e. $a \xrightarrow{2} d \xrightarrow{3} e$ ó $a \xrightarrow{4} b$.

Algoritmos para calcular caminos de longitud mínima



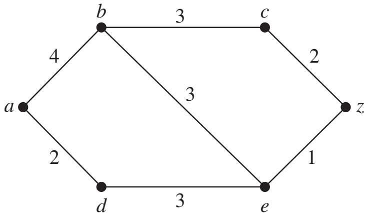
- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$
- b ó e. $a \xrightarrow{2} d \xrightarrow{3} e$ ó $a \xrightarrow{4} b$. $S = \{a, d, b\}$

Algoritmos para calcular caminos de longitud mínima



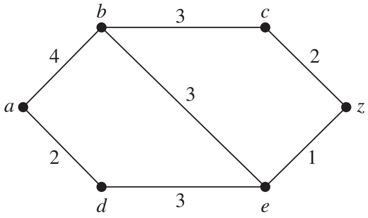
- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$
- b ó e. $a \xrightarrow{2} d \xrightarrow{3} e$ ó $a \xrightarrow{4} b$. $S = \{a, d, b\}$
- c ó e.

Algoritmos para calcular caminos de longitud mínima



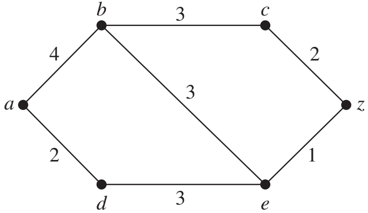
- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$
- b ó e. $a \xrightarrow{2} d \xrightarrow{3} e$ ó $a \xrightarrow{4} b$. $S = \{a, d, b\}$
- c ó e. $a \xrightarrow{4} b \xrightarrow{3} c$ ó $a \xrightarrow{2} d \xrightarrow{3} e$.

Algoritmos para calcular caminos de longitud mínima



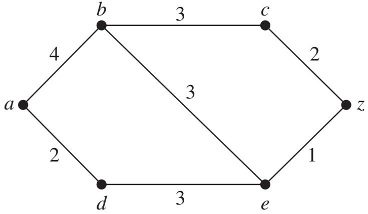
- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$
- b ó e. $a \xrightarrow{2} d \xrightarrow{3} e$ ó $a \xrightarrow{4} b$. $S = \{a, d, b\}$
- c ó e. $a \xrightarrow{4} b \xrightarrow{3} c$ ó $a \xrightarrow{2} d \xrightarrow{3} e$. $S = \{a, d, b, e\}$

Algoritmos para calcular caminos de longitud mínima



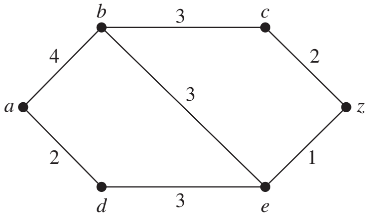
- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$
- b ó e. $a \xrightarrow{2} d \xrightarrow{3} e$ ó $a \xrightarrow{4} b$. $S = \{a, d, b\}$
- c ó e. $a \xrightarrow{4} b \xrightarrow{3} c$ ó $a \xrightarrow{2} d \xrightarrow{3} e$. $S = \{a, d, b, e\}$
- c ó z.

Algoritmos para calcular caminos de longitud mínima



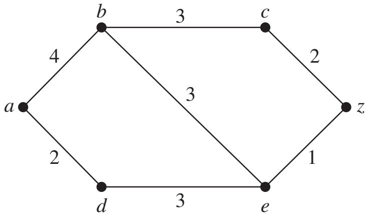
- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$
- b ó e. $a \xrightarrow{2} d \xrightarrow{3} e$ ó $a \xrightarrow{4} b$. $S = \{a, d, b\}$
- c ó e. $a \xrightarrow{4} b \xrightarrow{3} c$ ó $a \xrightarrow{2} d \xrightarrow{3} e$. $S = \{a, d, b, e\}$
- c ó z. $a \xrightarrow{4} b \xrightarrow{3} c$ ó $a \xrightarrow{2} d \xrightarrow{3} e \xrightarrow{1} z$.

Algoritmos para calcular caminos de longitud mínima



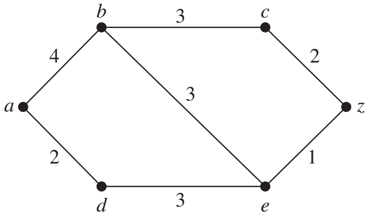
- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$
- b ó e. $a \xrightarrow{2} d \xrightarrow{3} e$ ó $a \xrightarrow{4} b$. $S = \{a, d, b\}$
- c ó e. $a \xrightarrow{4} b \xrightarrow{3} c$ ó $a \xrightarrow{2} d \xrightarrow{3} e$. $S = \{a, d, b, e\}$
- c ó z. $a \xrightarrow{4} b \xrightarrow{3} c$ ó $a \xrightarrow{2} d \xrightarrow{3} e \xrightarrow{1} z$. $S = \{a, d, b, e, z\}$

Algoritmos para calcular caminos de longitud mínima



- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$
- b ó e. $a \xrightarrow{2} d \xrightarrow{3} e$ ó $a \xrightarrow{4} b$. $S = \{a, d, b\}$
- c ó e. $a \xrightarrow{4} b \xrightarrow{3} c$ ó $a \xrightarrow{2} d \xrightarrow{3} e$. $S = \{a, d, b, e\}$
- c ó z. $a \xrightarrow{4} b \xrightarrow{3} c$ ó $a \xrightarrow{2} d \xrightarrow{3} e \xrightarrow{1} z$. $S = \{a, d, b, e, z\}$
- El costo del camino más corto de $a - z$ es 6

Algoritmos para calcular caminos de longitud mínima



- a. $S = \{a\}$
- b ó d. $a \xrightarrow{4} b$ ó $a \xrightarrow{2} d$. $S = \{a, d\}$
- b ó e. $a \xrightarrow{2} d \xrightarrow{3} e$ ó $a \xrightarrow{4} b$. $S = \{a, d, b\}$
- c ó e. $a \xrightarrow{4} b \xrightarrow{3} c$ ó $a \xrightarrow{2} d \xrightarrow{3} e$. $S = \{a, d, b, e\}$
- c ó z. $a \xrightarrow{4} b \xrightarrow{3} c$ ó $a \xrightarrow{2} d \xrightarrow{3} e \xrightarrow{1} z$. $S = \{a, d, b, e, z\}$
- El costo del camino más corto de $a - z$ es 6 y el camino es $a \xrightarrow{2} d \xrightarrow{3} e \xrightarrow{1} z$.

Algoritmos para calcular caminos de longitud mínima

¿De qué otra forma si hubiese podido encontrar el camino más corto entre a y z?

Algoritmos para calcular caminos de longitud mínima

¿De qué otra forma si hubiese podido encontrar el camino más corto entre a y z?

Se podría haber encontrado a través de fuerza bruta, examinando la longitud de cada camino de a a z.

Algoritmos para calcular caminos de longitud mínima

¿De qué otra forma si hubiese podido encontrar el camino más corto entre a y z?

Se podría haber encontrado a través de fuerza bruta, examinando la longitud de cada camino de a a z.

¿Por qué ese enfoque no sería adecuado para el algoritmo?

Algoritmos para calcular caminos de longitud mínima

¿De qué otra forma si hubiese podido encontrar el camino más corto entre a y z?

Se podría haber encontrado a través de fuerza bruta, examinando la longitud de cada camino de a a z.

¿Por qué ese enfoque no sería adecuado para el algoritmo?

Este enfoque es impráctico para los seres humanos e incluso para las máquinas cuando se tiene un gran número de aristas.

Algoritmos para calcular caminos de longitud mínima

De manera general, ¿cómo funciona el algoritmo de Dijkstra?

Algoritmos para calcular caminos de longitud mínima

De manera general, ¿cómo funciona el algoritmo de Dijkstra?

- Se considera ahora el problema general de determinar la longitud de un camino de longitud mínima entre a y z en un grafo ponderado simple no dirigido.
- El algoritmo de Dijkstra procede determinando la longitud de un camino de longitud mínima entre a y un primer vértice, la longitud de un camino de longitud mínima entre a y un segundo vértice, y así sucesivamente, hasta determinar la longitud de un camino de longitud mínima entre a y z.
- El algoritmo actúa realizando una serie de iteraciones.

Algoritmos para calcular caminos de longitud mínima

- Se construye un conjunto distinguido de vértices al que se añade un vértice en cada iteración.
- También se lleva a cabo en cada iteración un proceso de etiquetado.
- En este proceso, a cada vértice w se le pone una etiqueta que es la longitud de un camino cuya longitud es mínima entre la de los caminos de a a w que sólo pasan por vértices que ya estén en el conjunto distinguido.
- El vértice que se añade al conjunto distinguido es cualquiera cuya etiqueta sea mínima entre los vértices que aún no están en el conjunto.

Algoritmos para calcular caminos de longitud mínima

¿Cuál es el pseudocódigo del algoritmo de Dijkstra?

Algoritmos para calcular caminos de longitud mínima

¿Cuál es el pseudocódigo del algoritmo de Dijkstra?

procedure *Dijkstra*(G: weighted connected simple graph, with
    all weights positive)
{G has vertices $a = v_0, v_1, \dots, v_n = z$ and lengths $w(v_i, v_j)$
    where $w(v_i, v_j) = \infty$ if $\{v_i, v_j\}$ is not an edge in G}
for $i := 1$ **to** n
    $L(v_i) := \infty$
$L(a) := 0$
$S := \emptyset$
{the labels are now initialized so that the label of a is 0 and all
    other labels are ∞, and S is the empty set}
while $z \notin S$
    $u :=$ a vertex not in S with $L(u)$ minimal
    $S := S \cup \{u\}$
    for all vertices v not in S
        if $L(u) + w(u, v) < L(v)$ **then** $L(v) := L(u) + w(u, v)$
        {this adds a vertex to S with minimal label and updates the
        labels of vertices not in S}
return $L(z)$ {$L(z) =$ length of a shortest path from a to z}

Algoritmos para calcular caminos de longitud mínima

¿Cuál es la complejidad temporal del algoritmo de Dijkstra?

Algoritmos para calcular caminos de longitud mínima

¿Cuál es la complejidad temporal del algoritmo de Dijkstra?

- El algoritmo de Dijkstra realiza $O(n^2)$ operaciones (sumas y comparaciones) para determinar la longitud del camino más corto entre dos vértices de un grafo ponderado simple, conexo y no dirigido con n vértices.

Algoritmos para calcular caminos de longitud mínima

¿Cuál es el pseudocódigo del algoritmo de Dijkstra utilizando una cola de prioridad y arreglos para las distancias y los vértices previos?

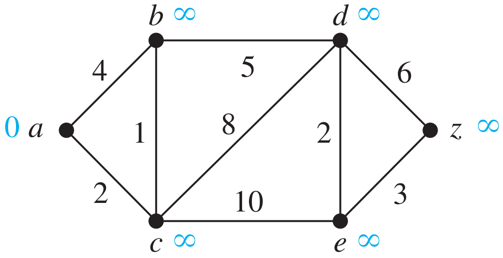
Algoritmos para calcular caminos de longitud mínima

¿Cuál es el pseudocódigo del algoritmo de Dijkstra utilizando una cola de prioridad y arreglos para las distancias y los vértices previos?

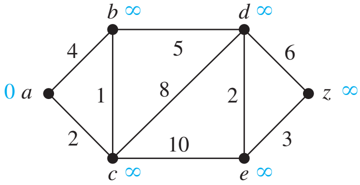
```
function Dijkstra(Graph, source):
    dist[source] ← 0

    create vertex set Q

    for each vertex v in Graph:
        if v ≠ source
            dist[v] ← INFINITY
        prev[v] ← UNDEFINED

        Q.add_with_priority(v, dist[v])


    while Q is not empty:
        u ← Q.extract_min()
        for each neighbor v of u:
            alt ← dist[u] + length(u, v)
            if alt < dist[v]
                dist[v] ← alt
                prev[v] ← u
                Q.decrease_priority(v, alt)

    return dist, prev
```

Algoritmos para calcular caminos de longitud mínima

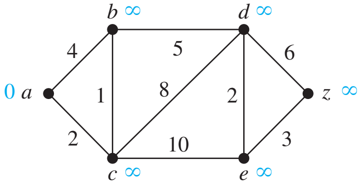
¿Cómo sería el seguimiento del algoritmo de Dijkstra para el siguiente grafo ponderado?



Algoritmos para calcular caminos de longitud mínima

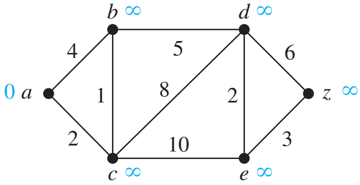


Algoritmos para calcular caminos de longitud mínima



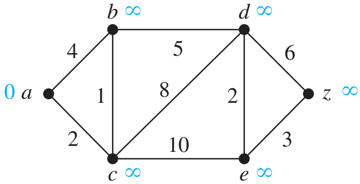
$PQ = \{a, b, c, d, e, z\}$

Algoritmos para calcular caminos de longitud mínima



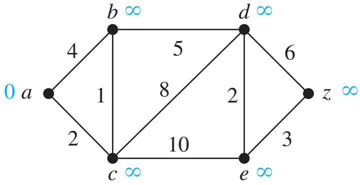
$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$

Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

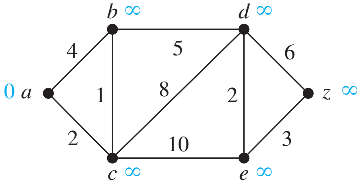
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$

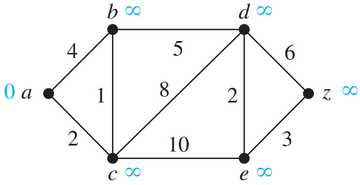
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$ $\quad dist = \{0, 4, 2, \infty, \infty, \infty\}$

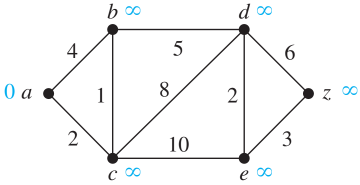
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$ $\quad dist = \{0, 4, 2, \infty, \infty, \infty\}$ $\quad prev = \{nil, a, a, nil, nil, nil\}$

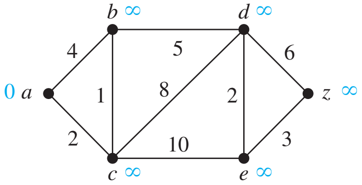
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$ $dist = \{0, 4, 2, \infty, \infty, \infty\}$ $prev = \{nil, a, a, nil, nil, nil\}$ $PQ = \{c, b, d, e, z\}$

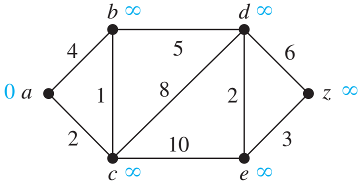
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$ $\quad dist = \{0, 4, 2, \infty, \infty, \infty\}$ $\quad prev = \{nil, a, a, nil, nil, nil\}$ $\quad PQ = \{c, b, d, e, z\}$
$u = c$

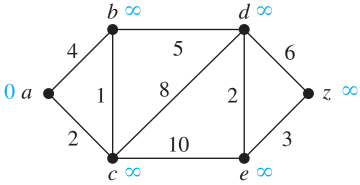
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$ $dist = \{0, 4, 2, \infty, \infty, \infty\}$ $prev = \{nil, a, a, nil, nil, nil\}$ $PQ = \{c, b, d, e, z\}$
$u = c$ $dist = \{0, 3, 2, 10, 12, \infty\}$

Algoritmos para calcular caminos de longitud mínima

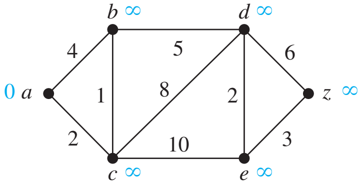


$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$ $dist = \{0, 4, 2, \infty, \infty, \infty\}$ $prev = \{nil, a, a, nil, nil, nil\}$ $PQ = \{c, b, d, e, z\}$

$u = c$ $dist = \{0, 3, 2, 10, 12, \infty\}$ $prev = \{nil, c, a, c, c, nil\}$

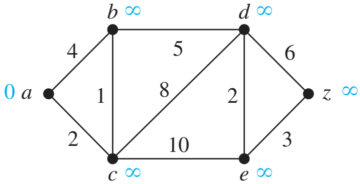
Algoritmos para calcular caminos de longitud mínima



$$PQ = \{a, b, c, d, e, z\} \;\; dist = \{0, \infty, \infty, \infty, \infty, \infty\} \; prev = \{nil, nil, nil, nil, nil, nil\}$$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$

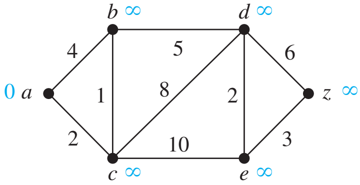
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$ $dist = \{0, 4, 2, \infty, \infty, \infty\}$ $prev = \{nil, a, a, nil, nil, nil\}$ $PQ = \{c, b, d, e, z\}$
$u = c$ $dist = \{0, 3, 2, 10, 12, \infty\}$ $prev = \{nil, c, a, c, c, nil\}$ $PQ = \{b, d, e, z\}$
$u = b$

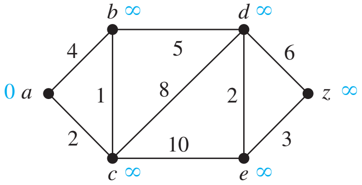
Algoritmos para calcular caminos de longitud mínima



$$PQ = \{a, b, c, d, e, z\} \;\; dist = \{0, \infty, \infty, \infty, \infty, \infty\} \; prev = \{nil, nil, nil, nil, nil, nil\}$$

$u = a$ $\quad dist = \{0, 4, 2, \infty, \infty, \infty\}$ $\quad prev = \{nil, a, a, nil, nil, nil\}$ $\quad PQ = \{c, b, d, e, z\}$
$u = c$ $\quad dist = \{0, 3, 2, 10, 12, \infty\}$ $\quad prev = \{nil, c, a, c, c, nil\}$ $\quad PQ = \{b, d, e, z\}$
$u = b$ $\quad dist = \{0, 3, 2, 8, 12, \infty\}$

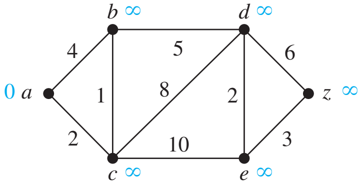
Algoritmos para calcular caminos de longitud mínima



$$PQ = \{a, b, c, d, e, z\} \;\; dist = \{0, \infty, \infty, \infty, \infty, \infty\} \; prev = \{nil, nil, nil, nil, nil, nil\}$$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	

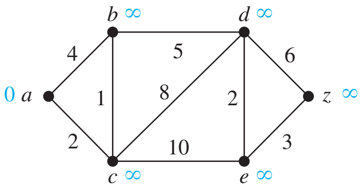
Algoritmos para calcular caminos de longitud mínima



$$PQ = \{a, b, c, d, e, z\} \quad dist = \{0, \infty, \infty, \infty, \infty, \infty\} \; prev = \{nil, nil, nil, nil, nil, nil\}$$

$$\begin{array}{llll}
u = a & dist = \{0, 4, 2, \infty, \infty, \infty\} & prev = \{nil, a, a, nil, nil, nil\} & PQ = \{c, b, d, e, z\} \\
u = c & dist = \{0, 3, 2, 10, 12, \infty\} & prev = \{nil, c, a, c, c, nil\} & PQ = \{b, d, e, z\} \\
u = b & dist = \{0, 3, 2, 8, 12, \infty\} & prev = \{nil, c, a, b, c, nil\} & PQ = \{d, e, z\}
\end{array}$$

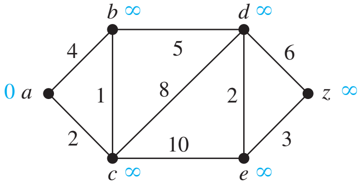
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$			

Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

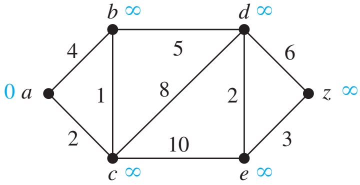
$u = a$ $dist = \{0, 4, 2, \infty, \infty, \infty\}$ $prev = \{nil, a, a, nil, nil, nil\}$ $PQ = \{c, b, d, e, z\}$

$u = c$ $dist = \{0, 3, 2, 10, 12, \infty\}$ $prev = \{nil, c, a, c, c, nil\}$ $PQ = \{b, d, e, z\}$

$u = b$ $dist = \{0, 3, 2, 8, 12, \infty\}$ $prev = \{nil, c, a, b, c, nil\}$ $PQ = \{d, e, z\}$

$u = d$ $dist = \{0, 3, 2, 8, 10, 14\}$

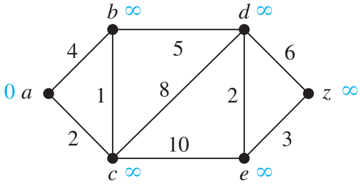
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$	$dist = \{0, 3, 2, 8, 10, 14\}$	$prev = \{nil, c, a, b, d, d\}$	

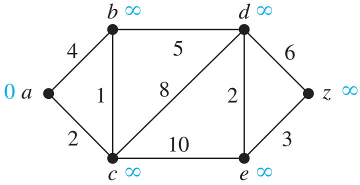
Algoritmos para calcular caminos de longitud mínima



$$PQ = \{a, b, c, d, e, z\} \;\; dist = \{0, \infty, \infty, \infty, \infty, \infty\} \; prev = \{nil, nil, nil, nil, nil, nil\}$$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$	$dist = \{0, 3, 2, 8, 10, 14\}$	$prev = \{nil, c, a, b, d, d\}$	$PQ = \{e, z\}$

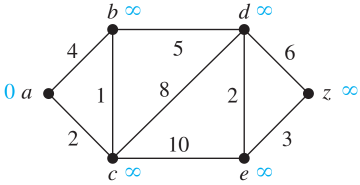
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$	$dist = \{0, 3, 2, 8, 10, 14\}$	$prev = \{nil, c, a, b, d, d\}$	$PQ = \{e, z\}$
$u = e$			

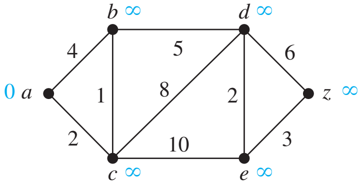
Algoritmos para calcular caminos de longitud mínima



$$PQ = \{a, b, c, d, e, z\} \;\; dist = \{0, \infty, \infty, \infty, \infty, \infty\} \; prev = \{nil, nil, nil, nil, nil, nil\}$$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$	$dist = \{0, 3, 2, 8, 10, 14\}$	$prev = \{nil, c, a, b, d, d\}$	$PQ = \{e, z\}$
$u = e$	$dist = \{0, 3, 2, 8, 10, 13\}$		

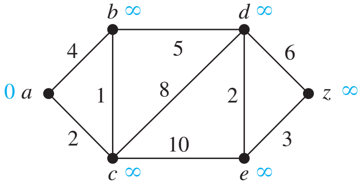
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$	$dist = \{0, 3, 2, 8, 10, 14\}$	$prev = \{nil, c, a, b, d, d\}$	$PQ = \{e, z\}$
$u = e$	$dist = \{0, 3, 2, 8, 10, 13\}$	$prev = \{nil, c, a, b, d, e\}$	

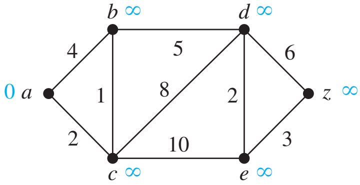
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$	$dist = \{0, 3, 2, 8, 10, 14\}$	$prev = \{nil, c, a, b, d, d\}$	$PQ = \{e, z\}$
$u = e$	$dist = \{0, 3, 2, 8, 10, 13\}$	$prev = \{nil, c, a, b, d, e\}$	$PQ = \{z\}$

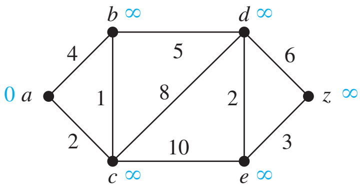
Algoritmos para calcular caminos de longitud mínima



$$PQ = \{a, b, c, d, e, z\} \;\; dist = \{0, \infty, \infty, \infty, \infty, \infty\} \; prev = \{nil, nil, nil, nil, nil, nil\}$$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$	$dist = \{0, 3, 2, 8, 10, 14\}$	$prev = \{nil, c, a, b, d, d\}$	$PQ = \{e, z\}$
$u = e$	$dist = \{0, 3, 2, 8, 10, 13\}$	$prev = \{nil, c, a, b, d, e\}$	$PQ = \{z\}$
$u = z$			

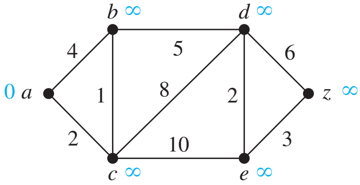
Algoritmos para calcular caminos de longitud mínima



$$PQ = \{a, b, c, d, e, z\} \;\; dist = \{0, \infty, \infty, \infty, \infty, \infty\} \; prev = \{nil, nil, nil, nil, nil, nil\}$$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$	$dist = \{0, 3, 2, 8, 10, 14\}$	$prev = \{nil, c, a, b, d, d\}$	$PQ = \{e, z\}$
$u = e$	$dist = \{0, 3, 2, 8, 10, 13\}$	$prev = \{nil, c, a, b, d, e\}$	$PQ = \{z\}$
$u = z$	$dist = \{0, 3, 2, 8, 10, 13\}$		

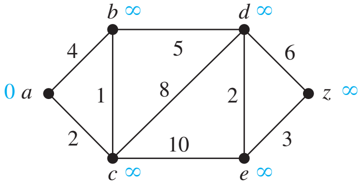
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$	$dist = \{0, 3, 2, 8, 10, 14\}$	$prev = \{nil, c, a, b, d, d\}$	$PQ = \{e, z\}$
$u = e$	$dist = \{0, 3, 2, 8, 10, 13\}$	$prev = \{nil, c, a, b, d, e\}$	$PQ = \{z\}$
$u = z$	$dist = \{0, 3, 2, 8, 10, 13\}$	$prev = \{nil, c, a, b, d, e\}$	

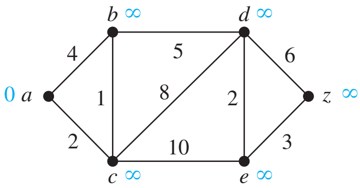
Algoritmos para calcular caminos de longitud mínima



$PQ = \{a, b, c, d, e, z\}$ $dist = \{0, \infty, \infty, \infty, \infty, \infty\}$ $prev = \{nil, nil, nil, nil, nil, nil\}$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$	$dist = \{0, 3, 2, 8, 10, 14\}$	$prev = \{nil, c, a, b, d, d\}$	$PQ = \{e, z\}$
$u = e$	$dist = \{0, 3, 2, 8, 10, 13\}$	$prev = \{nil, c, a, b, d, e\}$	$PQ = \{z\}$
$u = z$	$dist = \{0, 3, 2, 8, 10, 13\}$	$prev = \{nil, c, a, b, d, e\}$	$PQ = \{\}$

Algoritmos para calcular caminos de longitud mínima



$$PQ = \{a, b, c, d, e, z\} \;\; dist = \{0, \infty, \infty, \infty, \infty, \infty\} \; prev = \{nil, nil, nil, nil, nil, nil\}$$

$u = a$	$dist = \{0, 4, 2, \infty, \infty, \infty\}$	$prev = \{nil, a, a, nil, nil, nil\}$	$PQ = \{c, b, d, e, z\}$
$u = c$	$dist = \{0, 3, 2, 10, 12, \infty\}$	$prev = \{nil, c, a, c, c, nil\}$	$PQ = \{b, d, e, z\}$
$u = b$	$dist = \{0, 3, 2, 8, 12, \infty\}$	$prev = \{nil, c, a, b, c, nil\}$	$PQ = \{d, e, z\}$
$u = d$	$dist = \{0, 3, 2, 8, 10, 14\}$	$prev = \{nil, c, a, b, d, d\}$	$PQ = \{e, z\}$
$u = e$	$dist = \{0, 3, 2, 8, 10, 13\}$	$prev = \{nil, c, a, b, d, e\}$	$PQ = \{z\}$
$u = z$	$dist = \{0, 3, 2, 8, 10, 13\}$	$prev = \{nil, c, a, b, d, e\}$	$PQ = \{\}$

$$a \xrightarrow{2} c \xrightarrow{1} b \xrightarrow{5} d \xrightarrow{2} e \xrightarrow{3} z.$$

Agenda del día

1. **Caminos de longitud mínima**
   - Introducción
   - Algoritmos para calcular caminos de longitud mínima
   - El problema del viajante
   - Ejercicios

El problema del viajante

¿Cuál es el problema del viajante?

El problema del viajante

¿Cuál es el problema del viajante?

Un viajante quiere visitar exactamente una vez cada ciudad de un conjunto de n ciudades para finalmente regresar al punto de partida.

El problema del viajante

¿Cuál es el problema del viajante?

Un viajante quiere visitar exactamente una vez cada ciudad de un conjunto de n ciudades para finalmente regresar al punto de partida.

¿Cómo sería un ejemplo de este problema?

El problema del viajante

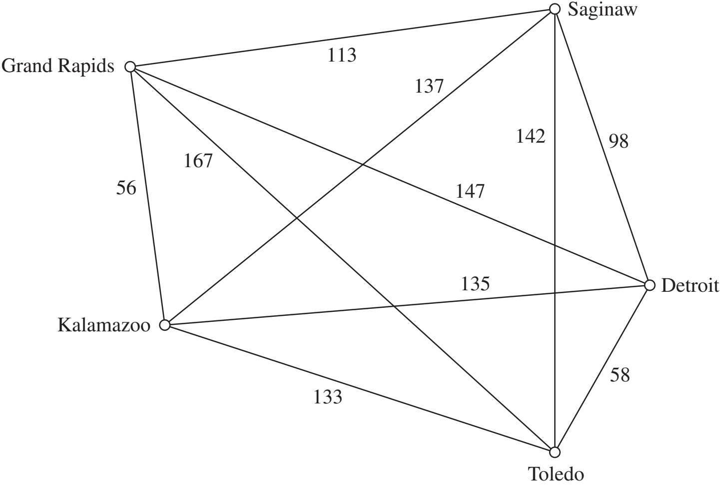
¿Cuál es el problema del viajante?

Un viajante quiere visitar exactamente una vez cada ciudad de un conjunto de *n* ciudades para finalmente regresar al punto de partida.

¿Cómo sería un ejemplo de este problema?

- Supongamos un caso particular del problema del viajante, en el que éste quiere visitar Detroit, Toledo, Saginaw, Grand Rapids y Kalamazoo.
- ¿En qué orden debería visitar estas ciudades para recorrer la mínima distancia total?

El problema del viajante



El problema del viajante

¿Cómo se resuelve este ejemplo?

El problema del viajante

¿Cómo se resuelve este ejemplo?

- Para resolver el problema suponemos que el viajante comienza por Detroit (ya que Detroit tiene que formar parte del circuito) y examinamos todas las posibles formas en que puede visitar las otras cuatro ciudades y regresar a Detroit (comenzar por cualquier otra ciudad produciría los mismos circuitos).
- Hay un total de 24 circuitos posibles, pero como al recorrer un circuito en orden inverso se recorre la misma distancia, basta considerar 12 circuitos diferentes para determinar la mínima distancia total que el viajante debe recorrer.
- Enumeramos estos 12 circuitos distintos y la distancia total que se recorre en cada circuito.

El problema del viajante

Route	*Total Distance (miles)*
Detroit–Toledo–Grand Rapids–Saginaw–Kalamazoo–Detroit	610
Detroit–Toledo–Grand Rapids–Kalamazoo–Saginaw–Detroit	516
Detroit–Toledo–Kalamazoo–Saginaw–Grand Rapids–Detroit	588
Detroit–Toledo–Kalamazoo–Grand Rapids–Saginaw–Detroit	458
Detroit–Toledo–Saginaw–Kalamazoo–Grand Rapids–Detroit	540
Detroit–Toledo–Saginaw–Grand Rapids–Kalamazoo–Detroit	504
Detroit–Saginaw–Toledo–Grand Rapids–Kalamazoo–Detroit	598
Detroit–Saginaw–Toledo–Kalamazoo–Grand Rapids–Detroit	576
Detroit–Saginaw–Kalamazoo–Toledo–Grand Rapids–Detroit	682
Detroit–Saginaw–Grand Rapids–Toledo–Kalamazoo–Detroit	646
Detroit–Grand Rapids–Saginaw–Toledo–Kalamazoo–Detroit	670
Detroit–Grand Rapids–Toledo–Saginaw–Kalamazoo–Detroit	728

El problema del viajante

¿Cuál es el resultado?

El problema del viajante

¿Cuál es el resultado?

- Como puede verse en la lista anterior, la distancia total mínima de 458 millas se recorre eligiendo el circuito Detroit-Toledo-Kalamazoo-Grand Rapids-Saginaw-Detroit (o el circuito inverso).

El problema del viajante

¿Qué pide determinar el problema del viajante?

El problema del viajante

¿Qué pide determinar el problema del viajante?

- El problema del viajante pide determinar el circuito de peso total mínimo de un grafo ponderado, completo y no dirigido que visita cada vértice exactamente una vez y regresa al punto de partida.
- Esto es equivalente a hallar un circuito hamiltoniano con peso total mínimo en el grafo completo, ya que cada vértice se visita exactamente una vez en el circuito.

El problema del viajante

¿Cuál es la forma más directa para resolver un caso particular?

El problema del viajante

¿Cuál es la forma más directa para resolver un caso particular?

- La forma más directa de resolver un caso particular del problema del viajante es examinar todos los posibles circuitos hamiltonianos y elegir uno de longitud total mínima.

El problema del viajante

¿Cuál es la forma más directa para resolver un caso particular?

- La forma más directa de resolver un caso particular del problema del viajante es examinar todos los posibles circuitos hamiltonianos y elegir uno de longitud total mínima.

¿Cuántos circuitos tenemos que examinar para resolver el problema si hay n vértices en el grafo?

El problema del viajante

¿Cuál es la forma más directa para resolver un caso particular?

- La forma más directa de resolver un caso particular del problema del viajante es examinar todos los posibles circuitos hamiltonianos y elegir uno de longitud total mínima.

¿Cuántos circuitos tenemos que examinar para resolver el problema si hay n vértices en el grafo?

- Una vez elegido el punto de partida, hay $(n-1)!$ circuitos hamiltonianos distintos que examinar, puesto que hay $n-1$ posibles elecciones para el segundo vértice, $n-2$ para el tercer vértice y así sucesivamente.
- Como un circuito hamiltoniano se puede recorrer en orden inverso, sólo tenemos que examinar $(n-1)!/2$ circuitos para hallar la solución.

El problema del viajante

¿Cuál es el problema con este enfoque exhaustivo?

El problema del viajante

¿Cuál es el problema con este enfoque exhaustivo?

- $(n-1)!/2$ crece muy rápidamente.
- Tratar de resolver de esta forma un problema del viajante cuando hay sólo unas pocas docenas de vértices es ya poco menos que imposible.
- Por ejemplo, con 25 vértices habría que considerar un total de $24!/2$ (aproximadamente, $3{,}1 \times 10^{23}$) circuitos hamiltonianos diferentes.
- Aunque sólo llevase un nanosegundo examinar cada circuito hamiltoniano, haría falta un total de aproximadamente diez millones de años para encontrar el circuito hamiltoniano de peso mínimo en este grafo empleando técnicas exhaustivas de búsqueda.

El problema del viajante

¿Qué se ha realizado para enfrentar este problema?

El problema del viajante

¿Qué se ha realizado para enfrentar este problema?

- Dado que el problema del viajante tiene importancia tanto desde el punto de vista práctico como desde el teórico, se ha hecho un gran esfuerzo para idear algoritmos eficientes para resolverlo.
- Sin embargo, no se conoce ningún algoritmo de complejidad polinómica en el peor caso que lo resuelva.
- Además, si se descubriese un algoritmo de complejidad polinómica en el peor caso para el problema del viajante, muchos otros problemas difíciles serían también resolubles empleando algoritmos de complejidad polinómica en el peor caso.

El problema del viajante

¿Cuál sería un enfoque práctico para resolver este problema?

El problema del viajante

¿Cuál sería un enfoque práctico para resolver este problema?

- Un enfoque práctico para resolver el problema del viajante cuando hay muchos vértices que visitar es utilizar un algoritmo de aproximación.
- Éstos son algoritmos que no necesariamente producen la solución exacta del problema, pero sí garantizan que la solución que producen está cerca de la solución exacta.
- Esto es, pueden producir un circuito hamiltoniano con peso total W' tal que $W \leq W' \leq cW$, siendo W la longitud total de la solución exacta y c una constante.

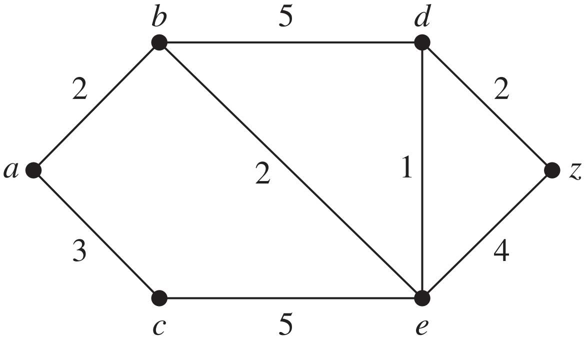
Agenda del día

1. **Caminos de longitud mínima**
   - Introducción
   - Algoritmos para calcular caminos de longitud mínima
   - El problema del viajante
   - Ejercicios

Ejercicios

Ejercicio

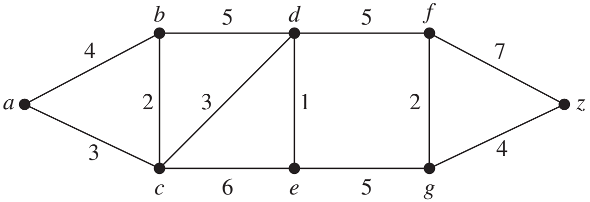
Determine la longitud de un camino de longitud mínima desde a hasta los demás nodos en el siguiente grafo ponderado. Realice el seguimiento al algoritmo.



Ejercicios

Ejercicio

Determine la longitud de un camino de longitud mínima desde a hasta los demás nodos en el siguiente grafo ponderado. Realice el seguimiento al algoritmo.



Ejercicios

Ejercicio

Continúe su implementación genérica de grafos (matriz de adyacencia y listas de adyacencia) e implemente el algoritmo de Dijkstra (incluya el BFS, DFS). Recuerde que la interface debe incluir todos los métodos del grafo.