

Caminhos de custo mínimo em grafos – Algoritmo de Dijkstra

Luciano Antonio Digiampietri

Histórico

- Histórico:
 - Especificado em 1956 pelo holandês Edsger Wybe Dijkstra e publicado em 1959^[1];
 - Originalmente concebido para encontrar o **caminho de menor distância** entre dois nós em um grafo com arestas ponderadas (com pesos não negativos, podendo ou não ser direcionadas);
 - Diversas variações foram produzidas ao longo dos anos.

Características

- Problema: encontrar os **caminhos de menor custo/distância** entre um nó de **origem** e todos os demais em um grafo ou digrafo com arestas ponderadas.
- Restrições:
 - os pesos das arestas são **não negativos**;
 - existe ao menos um caminho entre o nó de origem e cada um dos demais nós*

Algoritmo^[2]

- Entrada:
 - um grafo ou um digrafo (conjunto de nós + conjunto de arestas ponderadas);
 - o nó de origem.
- Saída:
 - d : arranjo com **os valores das menores distâncias** do nó de origem para cada um dos nós do grafo;
 - π : arranjo com o **predecessor** de cada um dos nós no caminho de menor distância entre o nó de origem e os demais nós.

Distância Mínima - Dijkstra

```
DIJKSTRA( $G, w, s$ ) {  
1. INITIALIZE-SINGLE-SOURCE( $G, s$ )  
2.  $S \leftarrow \emptyset$   
3.  $Q \leftarrow V[G]$   
4. while  $Q \neq \emptyset$   
5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$   
6.      $S \leftarrow S \cup \{u\}$   
7.     for each vertex  $v \in \text{Adj}[u]$   
8.         do RELAX( $u, v, w$ )
```

Distância Mínima - Dijkstra

INITIALIZE-SINGLE-SOURCE(G, s)

1. **for** each vertex $v \in V[G]$
2. **do** $d[v] \leftarrow \infty$
3. $\pi[v] \leftarrow \text{NIL}$
4. $d[s] \leftarrow 0$

RELAX(u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$
2. **then** $d[v] \leftarrow d[u] + w(u, v)$
3. $\pi[v] \leftarrow u$

Distância Mínima - Djikstra

EXTRACT-MIN (Q)

1. $min = Q.first$
2. **for** each vertex $v \in Q$
3. **do if** $d[v] < d[min]$
4. **then** $min = v$
5. $Q \leftarrow Q - \{min\}$
6. **return** min

Distância Mínima - Dijkstra

- Corretude

Distância Mínima - Dijkstra

- Corretude

Ao executarmos o algoritmo de Dijkstra sobre um digrafo ponderado $G=(V,E)$ com função peso não negativa w e origem s , ele terminará com $d[u]$ = distância mínima de s até u para todo $u \in V$

Distância Mínima - Dijkstra

- Corretude

Loop invariante

1. Inicialização
2. Manutenção
3. Término

Distância Mínima - Dijkstra

DIJKSTRA(G, w, s) {

1. INITIALIZE-SINGLE-SOURCE(G, s)

2. $S \leftarrow \emptyset$

3. $Q \leftarrow V[G]$

4. **while** $Q \neq \emptyset$

$d[u]$ = distância mínima
de s para todo $u \in S$

5. **do** $u \leftarrow \text{EXTRACT-MIN}(Q)$

6. $S \leftarrow S \cup \{u\}$

7. **for** each vertex $v \in \text{Adj}[u]$

8. **do** RELAX(u, v, w)

Distância Mínima - Dijkstra

- Complexidade assintótica
 - Pior caso

Distância Mínima - Dijkstra

```
DIJKSTRA( $G, w, s$ ) {  
1. INITIALIZE-SINGLE-SOURCE( $G, s$ )  
2.  $S \leftarrow \emptyset$   
3.  $Q \leftarrow V[G]$   
4. while  $Q \neq \emptyset$   
5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$   
6.      $S \leftarrow S \cup \{u\}$   
7.     for each vertex  $v \in \text{Adj}[u]$   
8.         do RELAX( $u, v, w$ )
```

Distância Mínima - Djikstra

Número de
vezes:

```
DIJKSTRA( $G, w, s$ ) {  
1. INITIALIZE-SINGLE-SOURCE( $G, s$ )  
2.  $S \leftarrow \emptyset$   
3.  $Q \leftarrow V[G]$   
4. while  $Q \neq \emptyset$   
5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$   
6.      $S \leftarrow S \cup \{u\}$   
7.     for each vertex  $v \in \text{Adj}[u]$   
8.         do RELAX( $u, v, w$ )
```

Distância Mínima - Dijkstra

```
DIJKSTRA( $G, w, s$ ) {  
1. INITIALIZE-SINGLE-SOURCE( $G, s$ )  
2.  $S \leftarrow \emptyset$   
3.  $Q \leftarrow V[G]$   
4. while  $Q \neq \emptyset$   
5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$   
6.      $S \leftarrow S \cup \{u\}$   
7.     for each vertex  $v \in \text{Adj}[u]$   
8.         do RELAX( $u, v, w$ )
```

Número de
vezes:
1

Distância Mínima - Dijkstra

DIJKSTRA (G, w, s) {

1. INITIALIZE-SINGLE-SOURCE (G, s)

2. $S \leftarrow \emptyset$

3. $Q \leftarrow V[G]$

4. **while** $Q \neq \emptyset$

5. **do** $u \leftarrow \text{EXTRACT-MIN}(Q)$

6. $S \leftarrow S \cup \{u\}$

7. **for** each vertex $v \in \text{Adj}[u]$

8. **do** RELAX (u, v, w)

Número de
vezes:

1

1

Distância Mínima - Djikstra

DIJKSTRA (G, w, s) {

1. INITIALIZE-SINGLE-SOURCE (G, s)

2. $S \leftarrow \emptyset$

3. $Q \leftarrow V[G]$

4. **while** $Q \neq \emptyset$

5. **do** $u \leftarrow \text{EXTRACT-MIN}(Q)$

6. $S \leftarrow S \cup \{u\}$

7. **for** each vertex $v \in \text{Adj}[u]$

8. **do** RELAX (u, v, w)

Número de
vezes:

1

1

1

Distância Mínima - Djikstra

DIJKSTRA (G, w, s) {

1. INITIALIZE-SINGLE-SOURCE (G, s)

2. $S \leftarrow \emptyset$

3. $Q \leftarrow V[G]$

4. **while** $Q \neq \emptyset$

5. **do** $u \leftarrow \text{EXTRACT-MIN}(Q)$

6. $S \leftarrow S \cup \{u\}$

7. **for** each vertex $v \in \text{Adj}[u]$

8. **do** RELAX (u, v, w)

Número de
vezes:

1

1

1

$|V| + 1$

Distância Mínima - Djikstra

DIJKSTRA (G, w, s) {

1. INITIALIZE-SINGLE-SOURCE (G, s)

2. $S \leftarrow \emptyset$

3. $Q \leftarrow V[G]$

4. **while** $Q \neq \emptyset$

5. **do** $u \leftarrow \text{EXTRACT-MIN}(Q)$

6. $S \leftarrow S \cup \{u\}$

7. **for** each vertex $v \in \text{Adj}[u]$

8. **do** RELAX (u, v, w)

Número de
vezes:

1

1

1

$|V| + 1$

$|V|$

Distância Mínima - Djikstra

DIJKSTRA (G, w, s) {

1. INITIALIZE-SINGLE-SOURCE (G, s)

2. $S \leftarrow \emptyset$

3. $Q \leftarrow V[G]$

4. **while** $Q \neq \emptyset$

5. **do** $u \leftarrow \text{EXTRACT-MIN}(Q)$

6. $S \leftarrow S \cup \{u\}$

7. **for** each vertex $v \in \text{Adj}[u]$

8. **do** RELAX (u, v, w)

Número de
vezes:

1

1

1

$|V| + 1$

$|V|$

$|V|$

Distância Mínima - Dijkstra

DIJKSTRA (G, w, s) {

1. INITIALIZE-SINGLE-SOURCE (G, s)

2. $S \leftarrow \emptyset$

3. $Q \leftarrow V[G]$

4. **while** $Q \neq \emptyset$

5. **do** $u \leftarrow \text{EXTRACT-MIN}(Q)$

6. $S \leftarrow S \cup \{u\}$

7. **for** each vertex $v \in \text{Adj}[u]$

8. **do** RELAX(u, v, w)

Número de
vezes:

1

1

1

$|V| + 1$

$|V|$

$|V|$

$|E| + |V|$

Distância Mínima - Djikstra

DIJKSTRA (G, w, s) {

1. INITIALIZE-SINGLE-SOURCE (G, s)

2. $S \leftarrow \emptyset$

3. $Q \leftarrow V[G]$

4. **while** $Q \neq \emptyset$

5. **do** $u \leftarrow \text{EXTRACT-MIN}(Q)$

6. $S \leftarrow S \cup \{u\}$

7. **for** each vertex $v \in \text{Adj}[u]$

8. **do** RELAX (u, v, w)

Número de
vezes:

1

1

1

$|V| + 1$

$|V|$

$|V|$

$|E| + |V|$

$|E|$

Distância Mínima - Dijkstra

Número de

INITIALIZE-SINGLE-SOURCE (G, s) **vezes:**

1. **for** each vertex $v \in V[G]$
2. **do** $d[v] \leftarrow \infty$
3. $\pi[v] \leftarrow \text{NIL}$
4. $d[s] \leftarrow 0$

RELAX (u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$
2. **then** $d[v] \leftarrow d[u] + w(u, v)$
3. $\pi[v] \leftarrow u$

Distância Mínima - Dijkstra

Número de

INITIALIZE-SINGLE-SOURCE (G, s) vezes:

1. **for** each vertex $v \in V[G]$ $|V| + 1$
2. **do** $d[v] \leftarrow \infty$
3. $\pi[v] \leftarrow \text{NIL}$
4. $d[s] \leftarrow 0$

RELAX (u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$
2. **then** $d[v] \leftarrow d[u] + w(u, v)$
3. $\pi[v] \leftarrow u$

Distância Mínima - Djikstra

Número de

INITIALIZE-SINGLE-SOURCE (G, s) vezes:

1. **for** each vertex $v \in V[G]$ $|V| + 1$
2. **do** $d[v] \leftarrow \infty$ $|V|$
3. $\pi[v] \leftarrow \text{NIL}$
4. $d[s] \leftarrow 0$

RELAX (u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$
2. **then** $d[v] \leftarrow d[u] + w(u, v)$
3. $\pi[v] \leftarrow u$

Distância Mínima - Dijkstra

Número de

INITIALIZE-SINGLE-SOURCE (G, s) **vezes:**

1. **for** each vertex $v \in V[G]$ $|V| + 1$
2. **do** $d[v] \leftarrow \infty$ $|V|$
3. $\pi[v] \leftarrow \text{NIL}$ $|V|$
4. $d[s] \leftarrow 0$

RELAX (u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$
2. **then** $d[v] \leftarrow d[u] + w(u, v)$
3. $\pi[v] \leftarrow u$

Distância Mínima - Dijkstra

Número de

INITIALIZE-SINGLE-SOURCE (G, s) vezes:

1. **for** each vertex $v \in V[G]$ $|V| + 1$
2. **do** $d[v] \leftarrow \infty$ $|V|$
3. $\pi[v] \leftarrow \text{NIL}$ $|V|$
4. $d[s] \leftarrow 0$ 1

RELAX (u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$
2. **then** $d[v] \leftarrow d[u] + w(u, v)$
3. $\pi[v] \leftarrow u$

Distância Mínima - Dijkstra

Número de

INITIALIZE-SINGLE-SOURCE (G, s) vezes:

1. **for** each vertex $v \in V[G]$ $|V| + 1$
2. **do** $d[v] \leftarrow \infty$ $|V|$
3. $\pi[v] \leftarrow \text{NIL}$ $|V|$
4. $d[s] \leftarrow 0$ 1

RELAX (u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$ 1
2. **then** $d[v] \leftarrow d[u] + w(u, v)$
3. $\pi[v] \leftarrow u$

Distância Mínima - Dijkstra

Número de

INITIALIZE-SINGLE-SOURCE (G, s) vezes:

1. **for** each vertex $v \in V[G]$ $|V| + 1$
2. **do** $d[v] \leftarrow \infty$ $|V|$
3. $\pi[v] \leftarrow \text{NIL}$ $|V|$
4. $d[s] \leftarrow 0$ 1

RELAX (u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$ 1
2. **then** $d[v] \leftarrow d[u] + w(u, v)$ 1
3. $\pi[v] \leftarrow u$

Distância Mínima - Dijkstra

Número de

INITIALIZE-SINGLE-SOURCE (G, s) vezes:

1. **for** each vertex $v \in V[G]$ $|V| + 1$
2. **do** $d[v] \leftarrow \infty$ $|V|$
3. $\pi[v] \leftarrow \text{NIL}$ $|V|$
4. $d[s] \leftarrow 0$ 1

RELAX (u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$ 1
2. **then** $d[v] \leftarrow d[u] + w(u, v)$ 1
3. $\pi[v] \leftarrow u$ 1

Distância Mínima - Dijkstra

Número de
vezes:

EXTRACT-MIN(Q)

1. $min = Q.first$
2. **for** each vertex $v \in Q$
3. **do if** $d[v] < d[min]$
4. **then** $min = v$
5. $Q \leftarrow Q - \{min\}$
6. **return** min

Distância Mínima - Dijkstra

EXTRACT-MIN(Q)

1. $min = Q.first$
2. **for** each vertex $v \in Q$
3. **do if** $d[v] < d[min]$
4. **then** $min = v$
5. $Q \leftarrow Q - \{min\}$
6. **return** min

Número de
vezes:
1

Distância Mínima - Dijkstra

EXTRACT-MIN(Q)

1. $min = Q.first$
2. **for** each vertex $v \in Q$
3. **do if** $d[v] < d[min]$
4. **then** $min = v$
5. $Q \leftarrow Q - \{min\}$
6. **return** min

Número de
vezes:

1

$|Q| + 1$

Distância Mínima - Djikstra

EXTRACT-MIN (Q)

1. $min = Q.first$
2. **for** each vertex $v \in Q$
3. **do if** $d[v] < d[min]$
4. **then** $min = v$
5. $Q \leftarrow Q - \{min\}$
6. **return** min

Número de
vezes:

1

$|Q| + 1$

$|Q|$

Distância Mínima - Djikstra

EXTRACT-MIN (Q)

1. $min = Q.first$

2. **for** each vertex $v \in Q$

3. **do if** $d[v] < d[min]$

4. **then** $min = v$

5. $Q \leftarrow Q - \{min\}$

6. **return** min

Número de
vezes:

1

$|Q| + 1$

$|Q|$

$|Q|$

Distância Mínima - Dijkstra

EXTRACT-MIN(Q)

1. $min = Q.first$

2. **for** each vertex $v \in Q$

3. **do if** $d[v] < d[min]$

4. **then** $min = v$

5. $Q \leftarrow Q - \{min\}$

6. **return** min

Número de
vezes:

1

$|Q| + 1$

$|Q|$

$|Q|$

1

Distância Mínima - Dijkstra

EXTRACT-MIN(Q)

1. $min = Q.first$

2. **for** each vertex $v \in Q$

3. **do if** $d[v] < d[min]$

4. **then** $min = v$

5. $Q \leftarrow Q - \{min\}$

6. **return** min

Número de
vezes:

1

$|Q| + 1$

$|Q|$

$|Q|$

1

1

Distância Mínima - Dijkstra

INITIALIZE-SINGLE-SOURCE(G, s) custo

1. **for** each vertex $v \in V[G]$ $|V| + 1$
2. **do** $d[v] \leftarrow \infty$ $|V|$
3. $\pi[v] \leftarrow \text{NIL}$ $|V|$
4. $d[s] \leftarrow 0$ 1

RELAX(u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$ 1
2. **then** $d[v] \leftarrow d[u] + w(u, v)$ 1
3. $\pi[v] \leftarrow u$ 1

Distância Mínima - Dijkstra

```
INITIALIZE-SINGLE-SOURCE (G, s) custo
1. for each vertex  $v \in V[G]$  c1  $|V| + 1$ 
2.     do  $d[v] \leftarrow \infty$   $|V|$ 
3.          $\pi[v] \leftarrow \text{NIL}$   $|V|$ 
4.  $d[s] \leftarrow 0$   $1$ 

RELAX ( $u, v, w$ )
1. if  $d[v] > d[u] + w(u, v)$   $1$ 
2.     then  $d[v] \leftarrow d[u] + w(u, v)$   $1$ 
3.          $\pi[v] \leftarrow u$   $1$ 
```

Distância Mínima - Dijkstra

```
INITIALIZE-SINGLE-SOURCE (G, s) custo
1. for each vertex  $v \in V[G]$  c1  $|V| + 1$ 
2.     do  $d[v] \leftarrow \infty$  c2  $|V|$ 
3.          $\pi[v] \leftarrow \text{NIL}$   $|V|$ 
4.  $d[s] \leftarrow 0$  1

RELAX ( $u, v, w$ )
1. if  $d[v] > d[u] + w(u, v)$  1
2.     then  $d[v] \leftarrow d[u] + w(u, v)$  1
3.          $\pi[v] \leftarrow u$  1
```


Distância Mínima - Dijkstra

```
INITIALIZE-SINGLE-SOURCE (G, s) custo
1. for each vertex  $v \in V[G]$  c1  $|V| + 1$ 
2.     do  $d[v] \leftarrow \infty$  c2  $|V|$ 
3.      $\pi[v] \leftarrow \text{NIL}$  c3  $|V|$ 
4.  $d[s] \leftarrow 0$  1

RELAX ( $u, v, w$ )
1. if  $d[v] > d[u] + w(u, v)$  1
2.     then  $d[v] \leftarrow d[u] + w(u, v)$  1
3.      $\pi[v] \leftarrow u$  1
```

Distância Mínima - Dijkstra

INITIALIZE-SINGLE-SOURCE (G, s) **custo**

1. **for** each vertex $v \in V[G]$ **c1** $|V| + 1$
2. **do** $d[v] \leftarrow \infty$ **c2** $|V|$
3. $\pi[v] \leftarrow \text{NIL}$ **c3** $|V|$
4. $d[s] \leftarrow 0$ **c4** 1

RELAX (u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$ **1**
2. **then** $d[v] \leftarrow d[u] + w(u, v)$ **1**
3. $\pi[v] \leftarrow u$ **1**

Distância Mínima - Dijkstra

INITIALIZE-SINGLE-SOURCE (G, s) **custo**

1. **for** each vertex $v \in V[G]$ **c1** $|V| + 1$
2. **do** $d[v] \leftarrow \infty$ **c2** $|V|$
3. $\pi[v] \leftarrow \text{NIL}$ **c3** $|V|$
4. $d[s] \leftarrow 0$ **c4** 1

RELAX (u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$ **c5** 1
2. **then** $d[v] \leftarrow d[u] + w(u, v)$ 1
3. $\pi[v] \leftarrow u$ 1

Distância Mínima - Dijkstra

```
INITIALIZE-SINGLE-SOURCE (G, s) custo
1. for each vertex  $v \in V[G]$  c1  $|V| + 1$ 
2.     do  $d[v] \leftarrow \infty$  c2  $|V|$ 
3.      $\pi[v] \leftarrow \text{NIL}$  c3  $|V|$ 
4.  $d[s] \leftarrow 0$  c4  $1$ 

RELAX ( $u, v, w$ )
1. if  $d[v] > d[u] + w(u, v)$  c5  $1$ 
2.     then  $d[v] \leftarrow d[u] + w(u, v)$  c6  $1$ 
3.      $\pi[v] \leftarrow u$   $1$ 
```

Distância Mínima - Dijkstra

```
INITIALIZE-SINGLE-SOURCE (G, s) custo
1. for each vertex  $v \in V[G]$  c1  $|V| + 1$ 
2.     do  $d[v] \leftarrow \infty$  c2  $|V|$ 
3.      $\pi[v] \leftarrow \text{NIL}$  c3  $|V|$ 
4.  $d[s] \leftarrow 0$  c4  $1$ 

RELAX ( $u, v, w$ )
1. if  $d[v] > d[u] + w(u, v)$  c5  $1$ 
2.     then  $d[v] \leftarrow d[u] + w(u, v)$  c6  $1$ 
3.      $\pi[v] \leftarrow u$  c7  $1$ 
```

Distância Mínima - Dijkstra

INITIALIZE-SINGLE-SOURCE(G, s) custo

1. **for** each vertex $v \in V[G]$ c1 $|V| + 1$
2. **do** $d[v] \leftarrow \infty$ c2 $|V|$
3. $\pi[v] \leftarrow \text{NIL}$ c3 $|V|$
4. $d[s] \leftarrow 0$ c4 1

$O(|V|)$

RELAX(u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$ c5 1
2. **then** $d[v] \leftarrow d[u] + w(u, v)$ c6 1
3. $\pi[v] \leftarrow u$ c7 1

$O(1)$

Distância Mínima - Djikstra

EXTRACT-MIN(Q)

custo

1. $min = Q.first$

1

2. **for** each vertex $v \in Q$

$|Q| + 1$

3. **do if** $d[v] < d[min]$

$|Q|$

4. **then** $min = v$

$|Q|$

5. $Q \leftarrow Q - \{min\}$

1

6. **return** min

1

Distância Mínima - Djikstra

EXTRACT-MIN(Q)

1. $min = Q.first$

2. **for** each vertex $v \in Q$

3. **do if** $d[v] < d[min]$

4. **then** $min = v$

5. $Q \leftarrow Q - \{min\}$

6. **return** min

custo

c8

1

$|Q| + 1$

$|Q|$

$|Q|$

1

1

Distância Mínima - Dijkstra

EXTRACT-MIN(Q)

1. $min = Q.first$

2. **for** each vertex $v \in Q$

3. **do if** $d[v] < d[min]$

4. **then** $min = v$

5. $Q \leftarrow Q - \{min\}$

6. **return** min

custo

c8

1

c9

$|Q| + 1$

$|Q|$

$|Q|$

1

1

Distância Mínima - Djikstra

EXTRACT-MIN (Q)

1. $min = Q.first$

2. **for** each vertex $v \in Q$

3. **do if** $d[v] < d[min]$

4. **then** $min = v$

5. $Q \leftarrow Q - \{min\}$

6. **return** min

custo

c8 1

c9 $|Q| + 1$

c10 $|Q|$

$|Q|$

1

1

Distância Mínima - Djikstra

EXTRACT-MIN (Q)

1. $min = Q.first$

2. **for** each vertex $v \in Q$

3. **do if** $d[v] < d[min]$

4. **then** $min = v$

5. $Q \leftarrow Q - \{min\}$

6. **return** min

custo

c8 1

c9 $|Q| + 1$

c10 $|Q|$

c11 $|Q|$

1

1

Distância Mínima - Djikstra

EXTRACT-MIN (Q)

1. $min = Q.first$

2. **for** each vertex $v \in Q$

3. **do if** $d[v] < d[min]$

4. **then** $min = v$

5. $Q \leftarrow Q - \{min\}$

6. **return** min

custo

c8 1

c9 $|Q| + 1$

c10 $|Q|$

c11 $|Q|$

$O(|Q|)$ 1

1

Distância Mínima - Dijkstra

EXTRACT-MIN (Q)

1. $min = Q.first$

2. **for** each vertex $v \in Q$

3. **do if** $d[v] < d[min]$

4. **then** $min = v$

5. $Q \leftarrow Q - \{min\}$

6. **return** min

custo

c8 1

c9 $|Q| + 1$

c10 $|Q|$

c11 $|Q|$

$O(|Q|)$ 1

c12 1

Distância Mínima - Djikstra

EXTRACT-MIN (Q)

1. $min = Q.first$

2. **for** each vertex $v \in Q$

3. **do if** $d[v] < d[min]$

4. **then** $min = v$

5. $Q \leftarrow Q - \{min\}$

6. **return** min

custo

c8 1

c9 $|Q| + 1$

c10 $|Q|$

c11 $|Q|$

$O(|Q|)$ 1

c12 1

$O(|Q|)$

Distância Mínima - Djikstra

```
DIJKSTRA ( $G, w, s$ ) {  
1. INITIALIZE-SINGLE-SOURCE ( $G, s$ )  
2.  $S \leftarrow \emptyset$   
3.  $Q \leftarrow V[G]$   
4. while  $Q \neq \emptyset$   
5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$   
6.      $S \leftarrow S \cup \{u\}$   
7.     for each vertex  $v \in \text{Adj}[u]$   
8.         do RELAX ( $u, v, w$ )
```

custo

1
1
1
 $|V| + 1$
 $|V|$
 $|V|$
 $|E| + |V|$
 $|E|$

Distância Mínima - Djikstra

```
DIJKSTRA ( $G, w, s$ ) {  
1. INITIALIZE-SINGLE-SOURCE ( $G, s$ )           1   custo  $O(|V|)$   
2.  $S \leftarrow \emptyset$                        1  
3.  $Q \leftarrow V[G]$                          1  
4. while  $Q \neq \emptyset$                       $|V| + 1$   
5.   do  $u \leftarrow \text{EXTRACT-MIN}(Q)$           $|V|$   
6.      $S \leftarrow S \cup \{u\}$                   $|V|$   
7.     for each vertex  $v \in \text{Adj}[u]$           $|E| + |V|$   
8.       do RELAX( $u, v, w$ )                    $|E|$ 
```


Distância Mínima - Djikstra

```
DIJKSTRA ( $G, w, s$ ) {  
1. INITIALIZE-SINGLE-SOURCE ( $G, s$ )  
2.  $S \leftarrow \emptyset$   
3.  $Q \leftarrow V[G]$   
4. while  $Q \neq \emptyset$   
5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$   
6.      $S \leftarrow S \cup \{u\}$   
7.     for each vertex  $v \in \text{Adj}[u]$   
8.         do RELAX ( $u, v, w$ )
```

custo
1 $O(|V|)$
1 $c13$
1
 $|V| + 1$
 $|V|$
 $|V|$
 $|E| + |V|$
 $|E|$

Distância Mínima - Dijkstra

```
DIJKSTRA( $G, w, s$ ) {  
    1. INITIALIZE-SINGLE-SOURCE( $G, s$ )  
    2.  $S \leftarrow \emptyset$   
    3.  $Q \leftarrow V[G]$   
    4. while  $Q \neq \emptyset$   
    5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$   
    6.      $S \leftarrow S \cup \{u\}$   
    7.     for each vertex  $v \in \text{Adj}[u]$   
    8.         do RELAX( $u, v, w$ )
```

custo

1.	INITIALIZE-SINGLE-SOURCE(G, s)	1	$O(V)$
2.	$S \leftarrow \emptyset$	1	c13
3.	$Q \leftarrow V[G]$	1	$O(V)$
4.	while $Q \neq \emptyset$	$ V + 1$	
5.	do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	
6.	$S \leftarrow S \cup \{u\}$	$ V $	
7.	for each vertex $v \in \text{Adj}[u]$	$ E + V $	
8.	do RELAX(u, v, w)	$ E $	

Distância Mínima - Dijkstra

```
DIJKSTRA( $G, w, s$ ) {  
    1. INITIALIZE-SINGLE-SOURCE( $G, s$ )           1   O(|V|)  
    2.  $S \leftarrow \emptyset$                    1   c13  
    3.  $Q \leftarrow V[G]$                        1   O(|V|)  
    4. while  $Q \neq \emptyset$                     $|V| + 1$  c14  
    5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$       $|V|$   
    6.      $S \leftarrow S \cup \{u\}$               $|V|$   
    7.     for each vertex  $v \in \text{Adj}[u]$   $|E| + |V|$   
    8.     do RELAX( $u, v, w$ )                  $|E|$   
}
```

custo

Distância Mínima - Djikstra

```
DIJKSTRA( $G, w, s$ ) {  
    1. INITIALIZE-SINGLE-SOURCE( $G, s$ )           1   O( $|V|$ )  
    2.  $S \leftarrow \emptyset$                        1   c13  
    3.  $Q \leftarrow V[G]$                            1   O( $|V|$ )  
    4. while  $Q \neq \emptyset$                         $|V| + 1$  c14  
    5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$           $|V|$   O( $|Q|$ )  
    6.        $S \leftarrow S \cup \{u\}$                 $|V|$   
    7.       for each vertex  $v \in \text{Adj}[u]$   $|E| + |V|$   
    8.         do RELAX( $u, v, w$ )                  $|E|$ 
```

custo

Distância Mínima - Djikstra

```
DIJKSTRA ( $G, w, s$ ) {  
    1. INITIALIZE-SINGLE-SOURCE ( $G, s$ )           1   custo  $O(|V|)$   
    2.  $S \leftarrow \emptyset$                        1   c13  
    3.  $Q \leftarrow V[G]$                           1    $O(|V|)$   
    4. while  $Q \neq \emptyset$                         $|V| + 1$  c14  
    5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$           $|V|$    $O(|Q|)$   
    6.      $S \leftarrow S \cup \{u\}$                   $|V|$   c15  
    7.     for each vertex  $v \in \text{Adj}[u]$             $|E| + |V|$   
    8.     do RELAX ( $u, v, w$ )                    $|E|$ 
```

Distância Mínima - Djikstra

```
DIJKSTRA( $G, w, s$ ) {  
    1. INITIALIZE-SINGLE-SOURCE( $G, s$ )           1   O(|V|)  
    2.  $S \leftarrow \emptyset$                      1   c13  
    3.  $Q \leftarrow V[G]$                          1   O(|V|)  
    4. while  $Q \neq \emptyset$                     $|V| + 1$  c14  
    5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$         $|V|$  O(|Q|)  
    6.      $S \leftarrow S \cup \{u\}$                 $|V|$  c15  
    7.     for each vertex  $v \in \text{Adj}[u]$   $|E| + |V|$  c16  
    8.     do RELAX( $u, v, w$ )                    $|E|$   
}
```

custo

Distância Mínima - Djikstra

```
DIJKSTRA( $G, w, s$ ) {  
    1. INITIALIZE-SINGLE-SOURCE( $G, s$ )           1   O(|V|)  
    2.  $S \leftarrow \emptyset$                      1   c13  
    3.  $Q \leftarrow V[G]$                          1   O(|V|)  
    4. while  $Q \neq \emptyset$                     $|V| + 1$  c14  
    5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$         $|V|$  O(|Q|)  
    6.      $S \leftarrow S \cup \{u\}$                 $|V|$  c15  
    7.     for each vertex  $v \in \text{Adj}[u]$   $|E| + |V|$  c16  
    8.     do RELAX( $u, v, w$ )                    $|E|$  O(1)  
}
```

Distância Mínima - Dijkstra

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. $Q \leftarrow V[G]$	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	c16
8. do RELAX(u, v, w)	$ E $	$O(1)$

Distância Mínima - Dijkstra

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	$c13$
3. $Q \leftarrow V[G]$	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	$c14$
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	$c15$
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	$c16$
8. do RELAX (u, v, w)	$ E $	$O(1)$
$c13 + c14 + (c14+c15+c16)* V + 2*O(V) + V *O(Q)$ $+ c16* E + E *O(1)$		

Distância Mínima - Dijkstra

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c_{13}
3. $Q \leftarrow V[G]$	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c_{14}
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c_{15}
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	c_{16}
8. do RELAX(u, v, w)	$ E $	$O(1)$
$c_{13} + c_{14} + (c_{14} + c_{15} + c_{16}) * V + 2 * O(V) + V * O(Q)$ $+ c_{16} * E + E * O(1)$		

Distância Mínima - Dijkstra

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c_{13}
3. $Q \leftarrow V[G]$	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c_{14}
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c_{15}
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	c_{16}
8. do RELAX(u, v, w)	$ E $	$O(1)$
$c_{13} + c_{14} + (c_{14} + c_{15} + c_{16}) * V + 2 * O(V) + O(V ^2) + c_{16} * E + E * O(1)$		

Distância Mínima - Dijkstra

	número de vezes	custo
DIJKSTRA(G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE(G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. $Q \leftarrow V[G]$	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow$ EXTRACT-MIN(Q)	$ V $	$O(Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in$ Adj[u]	$ E + V $	c16
8. do RELAX(u, v, w)	$ E $	$O(1)$
	$O(E + V ^2)$	

Distância Mínima - Dijkstra

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. $Q \leftarrow V[G]$	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	c16
8. do RELAX (u, v, w)	$ E $	$O(1)$

$O(|V|^2)$

Distância Mínima - Dijkstra

Qual o trecho do algoritmo que determina sua complexidade?

Distância Mínima - Dijkstra

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. $Q \leftarrow V[G]$	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	c16
8. do RELAX(u, v, w)	$ E $	$O(1)$
		$O(V ^2)$

Distância Mínima - Dijkstra

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. $Q \leftarrow V[G]$	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	c16
8. do RELAX(u, v, w)	$ E $	$O(1)$
		$O(V ^2)$

Distância Mínima - Dijkstra

É possível melhorar esse algoritmo?

Distância Mínima - Dijkstra

		número de vezes	custo
DIJKSTRA (G, w, s) {			
1. INITIALIZE-SINGLE-SOURCE (G, s)		1	$O(V)$
2. $S \leftarrow \emptyset$		1	c13
3. $Q \leftarrow V[G]$		1	$O(V)$
4. while $Q \neq \emptyset$		$ V + 1$	c14
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$		$ V $	$O(Q)$
6. $S \leftarrow S \cup \{u\}$		$ V $	c15
7. for each vertex $v \in \text{Adj}[u]$		$ E + V $	c16
8. do RELAX(u, v, w)		$ E $	$O(1)$
			$O(V ^2)$

Distância Mínima - Dijkstra

```
DIJKSTRA( $G, w, s$ ) {  
1. INITIALIZE-SINGLE-SOURCE( $G, s$ )  
2.  $S \leftarrow \emptyset$   
3. INITIALIZE-Q( $Q, V[G], s$ )  
4. while  $Q \neq \emptyset$   
5.     do  $u \leftarrow \text{EXTRACT-MIN}(Q)$   
6.      $S \leftarrow S \cup \{u\}$   
7.     for each vertex  $v \in \text{Adj}[u]$   
8.         do RELAX( $u, v, w$ )
```

Distância Mínima - Djikstra

RELAX (u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$
2. **then** $d[v] \leftarrow d[u] + w(u, v)$
3. $\pi[v] \leftarrow u$
4. *UPDATE* ($Q, v, d[v]$)

Distância Mínima - Dijkstra

RELAX(u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$
2. **then** $d[v] \leftarrow d[u] + w(u, v)$
3. $\pi[v] \leftarrow u$
4. *DECREASE_KEY(Q, v, d[v])*

Uso de um fila de prioridades mínima

- Funções / métodos:
 - Inicialização – estrutura vazia;
 - Inicialização – conjunto de elementos;
 - Extração do elemento de menor prioridade;
 - Exclusão de um elemento arbitrário;
 - Inserção com prioridade;
 - Diminuição do valor da prioridade de um elemento;
 - Retorno do elemento de menor prioridade;

Uso de um fila de prioridades mínima

- Funções / métodos:
 - Inicialização – estrutura vazia;
 - **Inicialização – conjunto de elementos;**
 - **Extração do elemento de menor prioridade;**
 - Exclusão de um elemento arbitrário;
 - Inserção com prioridade;
 - **Diminuição do valor da prioridade de um elemento;**
 - Retorno do elemento de menor prioridade;

Uso de um fila de prioridades mínima

- Implementação utilizando um **arranjo ordenado** de forma decrescente

Usando um arranjo ordenado

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q ($Q, V[G], s$)	1	?
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow$ EXTRACT-MIN (Q)	$ V $?
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in$ Adj [u]	$ E + V $	c16
8. do RELAX (u, v, w)	$ E $?

Usando um arranjo ordenado

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q ($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow$ EXTRACT-MIN (Q)	$ V $?
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in$ Adj [u]	$ E + V $	c16
8. do RELAX (u, v, w)	$ E $?

Usando um arranjo ordenado

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q ($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow$ EXTRACT-MIN (Q)	$ V $	$O(1)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in$ Adj [u]	$ E + V $	c16
8. do RELAX (u, v, w)	$ E $?

Distância Mínima - Dijkstra

RELAX(u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$

1 c5

2. **then** $d[v] \leftarrow d[u] + w(u, v)$

1 c6

3. $\pi[v] \leftarrow u$

1 c7

4. DECREASE_KEY($Q, v, d[v]$)

1 ?

Distância Mínima - Dijkstra

RELAX(u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$

1 c5

2. **then** $d[v] \leftarrow d[u] + w(u, v)$

1 c6

3. $\pi[v] \leftarrow u$

1 c7

4. DECREASE_KEY($Q, v, d[v]$)

1 $O(|Q|)$

Distância Mínima - Dijkstra

RELAX(u, v, w)

1. **if** $d[v] > d[u] + w(u, v)$ 1 c5
2. **then** $d[v] \leftarrow d[u] + w(u, v)$ 1 c6
3. $\pi[v] \leftarrow u$ 1 c7
4. DECREASE_KEY($Q, v, d[v]$) 1 O(|Q|)

$O(|Q|)$

Usando um arranjo ordenado

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q ($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow$ EXTRACT-MIN (Q)	$ V $	$O(1)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in$ Adj [u]	$ E + V $	c16
8. do RELAX (u, v, w)	$ E $?

Usando um arranjo ordenado

	número de vezes	custo
DIJKSTRA(G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE(G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(1)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	c16
8. do RELAX(u, v, w)	$ E $	$O(Q)$

Usando um arranjo ordenado

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q ($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow$ EXTRACT-MIN (Q)	$ V $	$O(1)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in$ Adj [u]	$ E + V $	c16
8. do RELAX (u, v, w)	$ E $	$O(Q)$

$O(|E|*|V|)$

Usando um arranjo ordenado

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q ($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(1)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	c16
8. do RELAX (u, v, w)	$ E $	$O(Q)$

$O(|E|*|V|)$ – corresponde a uma melhoria?

Usando um arranjo ordenado

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q ($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(1)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	c16
8. do RELAX (u, v, w)	$ E $	$O(Q)$

$O(|E|*|V|)$ – não para $|E| \in \Omega(|V|)$

Recapitulando

- Para a fila de prioridade mínima:
 - Inicialização – conjunto de elementos: ocorre **1 vez**
 - Extração do elemento de menor prioridade: ocorre **|V| vezes**
 - Diminuição do valor da prioridade de um elemento: ocorre até **|E| vezes**

Uso de um fila de prioridades mínima

- Implementação utilizando um **arranjo ordenado** de forma decrescente
 - Inicialização pode ser feita em $O(|V|)$
 - Extração do elemento de menor valor em $O(1)$
 - Diminuição do valor da prioridade de um elemento é custoso:
 - Encontrar um elemento arbitrário: $O(\lg|Q|)$
 - Encontrar o local para colocar o elemento com distância atualizada: $O(\lg|Q|)$
 - Mover os elementos para reordenar o arranjo: $O(|Q|)$

Uso de um fila de prioridades mínima

- Implementação utilizando um **lista ligada ordenada** de forma crescente

Uso de um fila de prioridades mínima

- Implementação utilizando um **lista ligada ordenada** de forma crescente
 - Inicialização:
 - Extração do elemento de menor valor:
 - Diminuição do valor da prioridade de um elemento:

Uso de um fila de prioridades mínima

- Implementação utilizando um **lista ligada ordenada** de forma crescente
 - Inicialização pode ser feita em $O(|V|)$
 - Extração do elemento de menor valor em $O(1)$
 - Diminuição do valor da prioridade de um elemento é custoso:
 - Encontrar um elemento arbitrário: $O(|Q|)$
 - Encontrar o local para colocar o elemento com distância atualizada: $O(|Q|)$
 - Atualização exige apenas a atualização de ponteiros: $O(1)$

Uso de um fila de prioridades mínima

- Implementação utilizando um **lista duplamente ligada ordenada** de forma crescente e com **acesso direto** aos seus elementos

Uso de um fila de prioridades mínima

- Implementação utilizando um **lista duplamente ligada ordenada** de forma crescente e com **acesso direto** aos seus elementos
 - Inicialização pode ser feita em $O(|V|)$
 - Extração do elemento de menor valor em $O(1)$
 - Diminuição do valor da prioridade de um elemento é custoso:
 - Encontrar um elemento: $O(1)$
 - Encontrar o local para colocar o elemento com distância atualizada: $O(|Q|)$
 - Atualização exige apenas a atualização de ponteiros: $O(1)$

Uso de um fila de prioridades mínima

- Implementação utilizando um **heap binário mínimo**

Uso de um fila de prioridades mínima

- Implementação utilizando um **heap binário mínimo**
 - Inicialização:
 - Extração do elemento de menor valor:
 - Diminuição do valor da prioridade de um elemento:

Uso de um fila de prioridades mínima

- Implementação utilizando um **heap binário mínimo**
 - Inicialização pode ser feita em $O(|V|)$
 - Extração do elemento de menor valor em $O(\lg|Q|)$
 - Diminuição do valor da prioridade de um elemento:
 - Reorganizar os elementos, colocando no local correto o elemento com distância atualizada: $O(\lg|Q|)$

Usando um heap binário mínimo

	número de vezes	custo
DIJKSTRA(G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE(G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow$ EXTRACT-MIN(Q)	$ V $	$O(\lg Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in$ Adj[u]	$ E + V $	c16
8. do RELAX(u, v, w)	$ E $	$O(\lg Q)$

Usando um heap binário mínimo

	número de vezes	custo
DIJKSTRA(G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE(G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	$c13$
3. INITIALIZE-Q($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	$c14$
5. do $u \leftarrow$ EXTRACT-MIN(Q)	$ V $	$O(\lg Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	$c15$
7. for each vertex $v \in$ Adj[u]	$ E + V $	$c16$
8. do RELAX(u, v, w)	$ E $	$O(\lg Q)$
$O((V + E) * \lg V)$		

Usando um heap binário mínimo

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q ($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow$ EXTRACT-MIN (Q)	$ V $	$O(\lg Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in$ Adj [u]	$ E + V $	c16
8. do RELAX (u, v, w)	$ E $	$O(\lg Q)$
		$O((V + E) * \lg V)$

corresponde a uma potencial melhoria?

Usando um heap binário mínimo

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q ($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow$ EXTRACT-MIN (Q)	$ V $	$O(\lg Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in$ Adj [u]	$ E + V $	c16
8. do RELAX (u, v, w)	$ E $	$O(\lg Q)$

$O((|V| + |E|) \cdot \lg|V|)$

potencialmente sim, se $|E| \in o(|V|^2/\lg|V|)$

Uso de um fila de prioridades mínima

- Implementação utilizando um **árvore binária de busca**

Uso de um fila de prioridades mínima

- Implementação utilizando um **árvore binária balanceada de busca**

Uso de um fila de prioridades mínima

- Implementação utilizando um **heap binário mínimo**
 - Inicialização pode ser feita em **$O(|V|)$**
 - Extração do elemento de menor valor em **$O(\lg|Q|)$**
 - Diminuição do valor da prioridade de um elemento:
 - Reorganizar os elementos, colocando no local correto o elemento com distância atualizada: **$O(\lg|Q|)$**

Uso de um fila de prioridades mínima

- Implementação utilizando um **Heap de Fibonacci Mínimo**^[3]

Uso de um fila de prioridades mínima

- Implementação utilizando um **Heap de Fibonacci Mínimo**^[3]
 - Inicialização pode ser feita em $O(|V|)$
 - Extração do elemento de menor valor em $O(\lg|Q|)$ *
 - Diminuição do valor da prioridade de um elemento:
 - Reorganizar os elementos, colocando no local correto o elemento com distância atualizada: $O(1)$ *

* cálculo amortizado

Usando um heap de Fibonacci mínimo

	número de vezes	custo
DIJKSTRA(G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE(G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow$ EXTRACT-MIN(Q)	$ V $	$O(\lg Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in$ Adj[u]	$ E + V $	c16
8. do RELAX(u, v, w)	$ E $	$O(1)$

Usando um heap de Fibonacci mínimo

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	c13
3. INITIALIZE-Q ($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	c14
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(\lg Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	c15
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	c16
8. do RELAX (u, v, w)	$ E $	$O(1)$

$$O(|E| + |V| \lg |V|)$$

Usando um heap de Fibonacci mínimo

	número de vezes	custo
DIJKSTRA (G, w, s) {		
1. INITIALIZE-SINGLE-SOURCE (G, s)	1	$O(V)$
2. $S \leftarrow \emptyset$	1	$c13$
3. INITIALIZE-Q ($Q, V[G], s$)	1	$O(V)$
4. while $Q \neq \emptyset$	$ V + 1$	$c14$
5. do $u \leftarrow \text{EXTRACT-MIN}(Q)$	$ V $	$O(\lg Q)$
6. $S \leftarrow S \cup \{u\}$	$ V $	$c15$
7. for each vertex $v \in \text{Adj}[u]$	$ E + V $	$c16$
8. do RELAX (u, v, w)	$ E $	$O(1)$

$$O(|E| + |V| \cdot \lg|V|) \in O(|V|^2)$$

Distância Mínima - Djikstra

E se o grafo for esparso?

E se o grafo for muito grande?

Considerações Finais

- Utilizando uma **fila de prioridade mínima** implementada com um **Heap de Fibonacci** a complexidade assintótica (amortizada) do algoritmo será: $O(|E| + |V|\log_2|V|)$
- $O(|E| + |V|\log_2|V|)$ é a **menor complexidade conhecida** para encontrar a menor distância entre um nó e todos os outros para grafos com arestas com pesos arbitrários não negativos.

Referências

1. DIJKSTRA, E.W.. A note on two problems in connexion with graphs. *Numerische Mathematik*, Volume 1, Issue 1, pp 269-271, 1959.
2. CORMEN, T.H.; LEISERSON, C.E.; RIVEST, R.L.; STEIN, C.. Algoritmos Teoria e Prática (tradução da 2ª edição americana), 2002.
3. FREDMAN, M.L.; TARJAN, R.E.. Fibonacci heaps and their uses in improved network optimization algorithms. 25th IEEE Annual Symposium on Foundations of Computer Science. pp. 338-346, 1984.

Distância Mínima - Djikstra

Primeira implementação