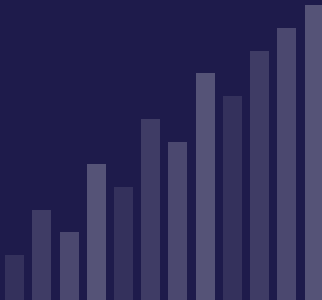


AULA 32 · BLOCO 4: GRAFOS

Caminhos mínimos entre todos os pares: Floyd-Warshall

Seção 25.2

Estruturas de dados e seus algoritmos · José Dihego · IFBA



Roteiro da aula

Ao final, você será capaz de

- ▶ Formular caminhos mínimos entre todos os pares por programação dinâmica
- ▶ Executar Floyd-Warshall e reconstruir caminhos
- ▶ Calcular o fecho transitivo
- ▶ Medir V^3 contra V execuções de Dijkstra em grafos esparsos



Teoria

A recorrência dos intermediários, o algoritmo, o fecho transitivo



Prática em C

Floyd-Warshall no grafo da Figura 25.1



Experimento

Tempo de Floyd-Warshall contra $V \times$ Dijkstra



Conexões

Redes, bioinformática, mapas e análise de dependências

A recorrência

Numere os vértices $1..n$. Seja $d_{ij}^{(k)}$ o peso do caminho mínimo de i a j cujos vértices **intermediários** estão todos em $\{1, \dots, k\}$:

$$d_{ij}^{(k)} = \begin{cases} w_{ij} & k = 0 \\ \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}) & k \geq 1 \end{cases}$$

- ▶ Ou o caminho mínimo não passa por k , ou passa **uma vez**, e as duas metades são mínimas.
- ▶ A resposta é $D^{(n)}$. Pesos negativos são permitidos; ciclos negativos, não.

O algoritmo

- ▶ Três laços, $\Theta(n^3)$, constante pequena.
- ▶ Uma só matriz basta (exercício 25.2-4).
- ▶ Predecessores π_{ij} reconstroem os caminhos.
- ▶ **Fecho transitivo:** troque min e + por ou e e.

FLOYD-WARSHALL(W)

```
1  $n \leftarrow W.linhas$ ;  $D \leftarrow W$ 
2 para  $k \leftarrow 1$  até  $n$  faça
3   para  $i \leftarrow 1$  até  $n$  faça
4     para  $j \leftarrow 1$  até  $n$  faça
5        $d_{ij} \leftarrow \min(d_{ij}, d_{ik} + d_{kj})$ 
6 retorne  $D$ 
```

As matrizes $D^{(k)}$ e $\Pi^{(k)}$

$$\begin{aligned}
 D^{(0)} &= \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} & \Pi^{(0)} &= \begin{pmatrix} \text{NIL} & 1 & 1 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL} & 3 & \text{NIL} & \text{NIL} & \text{NIL} \\ 4 & \text{NIL} & 4 & \text{NIL} & \text{NIL} \\ \text{NIL} & \text{NIL} & \text{NIL} & 5 & \text{NIL} \end{pmatrix} \\
 D^{(1)} &= \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} & \Pi^{(1)} &= \begin{pmatrix} \text{NIL} & 1 & 1 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL} & 3 & \text{NIL} & \text{NIL} & \text{NIL} \\ 4 & 1 & 4 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 5 & \text{NIL} \end{pmatrix} \\
 D^{(2)} &= \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} & \Pi^{(2)} &= \begin{pmatrix} \text{NIL} & 1 & 1 & 2 & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL} & 3 & \text{NIL} & 2 & 2 \\ 4 & 1 & 4 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 5 & \text{NIL} \end{pmatrix} \\
 D^{(3)} &= \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} & \Pi^{(3)} &= \begin{pmatrix} \text{NIL} & 1 & 1 & 2 & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL} & 3 & \text{NIL} & 2 & 2 \\ 4 & 3 & 4 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 5 & \text{NIL} \end{pmatrix} \\
 D^{(4)} &= \begin{pmatrix} 0 & 3 & -1 & 4 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix} & \Pi^{(4)} &= \begin{pmatrix} \text{NIL} & 1 & 4 & 2 & 1 \\ 4 & \text{NIL} & 4 & 2 & 1 \\ 4 & 3 & \text{NIL} & 2 & 1 \\ 4 & 3 & 4 & \text{NIL} & 1 \\ 4 & 3 & 4 & 5 & \text{NIL} \end{pmatrix} \\
 D^{(5)} &= \begin{pmatrix} 0 & 1 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix} & \Pi^{(5)} &= \begin{pmatrix} \text{NIL} & 3 & 4 & 5 & 1 \\ 4 & \text{NIL} & 4 & 2 & 1 \\ 4 & 3 & \text{NIL} & 2 & 1 \\ 4 & 3 & 4 & \text{NIL} & 1 \\ 4 & 3 & 4 & 5 & \text{NIL} \end{pmatrix}
 \end{aligned}$$

Figura 25.4 de Cormen et al. (2009) — Floyd-Warshall no grafo da Figura 25.1: de $D^{(0)}$ a $D^{(5)}$, com as matrizes de



Prática em C

- ▶ O grafo da Figura 25.1, com pesos negativos
- ▶ Três laços: k por fora
- ▶ Imprime $D^{(5)}$ (Figura 25.4)

Floyd-Warshall em C (1/2)

```
1  #include <stdio.h>
2
3  /* 0 grafo da Figura 25.1 (5 vértices, pesos negativos, sem ciclo negativo).
4      $D^{(k)}[i][j]$ : menor caminho de i a j com intermediários só entre os vértices 1..k. */
5  #define V 5
6  #define INF 1000000
7
8  int main(void) {
9     int qualquer;
10     if (scanf("%d", &qualquer) != 1) return 1;
11     int D[V][V];
12     for (int i = 0; i < V; i++)
13         for (int j = 0; j < V; j++) D[i][j] = i == j ? 0 : INF;
```

Floyd-Warshall em C (2/2)

```
14  int arestas[][3] = {{0, 1, 3}, {0, 2, 8}, {0, 4, -4}, {1, 3, 1}, {1, 4, 7},
15                          {2, 1, 4}, {3, 0, 2}, {3, 2, -5}, {4, 3, 6}};
16  for (int e = 0; e < 9; e++) D[arestas[e][0]][arestas[e][1]] = arestas[e][2];
17  for (int k = 0; k < V; k++)           // k: novo intermediário permitido
18      for (int i = 0; i < V; i++)
19          for (int j = 0; j < V; j++)
20              if (D[i][k] < INF && D[k][j] < INF && D[i][k] + D[k][j] < D[i][j])
21                  D[i][j] = D[i][k] + D[k][j];    // passar por k é melhor
22  for (int i = 0; i < V; i++) {           // a matriz  $D^{(5)}$  da Figura 25.4
23      for (int j = 0; j < V; j++) printf("%d%c", D[i][j], j < V - 1 ? ' ' : '\n');
24  }
25  return 0;
26 }
```



Experimento

- ▶ Grafo com $E = 8V$; as duas respostas são conferidas
- ▶ Floyd-Warshall: $\Theta(V^3)$
- ▶ $V \times$ Dijkstra com heap: $\Theta(V(V + E) \lg V) = \Theta(V^2 \lg V)$

EXPERIMENTO

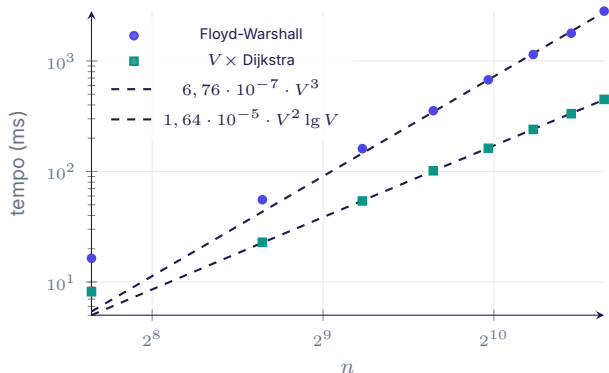
V^3 contra $V \times$ Dijkstra

Hipótese. Floyd-Warshall custa $\Theta(V^3)$, sem depender de E . Rodar Dijkstra de cada vértice custa $\Theta(V(V + E) \lg V)$: em grafos esparsos, $\Theta(V^2 \lg V)$. Apesar da constante menor do Floyd-Warshall, a diferença de ordem deve aparecer já para algumas centenas de vértices.

Método

1. V de 200 a 1.600; $E = 8V$; pesos de 1 a 100
2. Floyd-Warshall com matriz em um vetor contíguo
3. Dijkstra com heap binário a partir de cada vértice
4. Soma das distâncias conferida entre os dois métodos

Tempo para todos os pares



n	medido	razão
200,0	16,4	$2,05 \cdot 10^{-6}$
400,0	55,5	$8,68 \cdot 10^{-7}$
600,0	160,9	$7,45 \cdot 10^{-7}$
800,0	354,6	$6,93 \cdot 10^{-7}$
1.000	675,2	$6,75 \cdot 10^{-7}$
1.600	2.826	$6,90 \cdot 10^{-7}$

razão = medido $\div V^3$

✓ desvio máx. 3,8%

Em log-log, inclinação 3 contra pouco mais de 2. Para $V = 1.600$: 1,5 s contra 0,27 s.

🧪 EXPERIMENTO

O que o experimento mostrou

As duas formas de crescimento se confirmam: V^3 para Floyd-Warshall e $V^2 \lg V$ para V execuções de Dijkstra, com as mesmas respostas. Em grafos **esparsos**, Dijkstra repetido vence; em grafos **densos** ou pequenos, a simplicidade do Floyd-Warshall compensa, e ele aceita pesos negativos.

➤ Reproduza: `gcc -O2 aula32_experimento.c -o exp -lm && ./exp`



Conexões

- ▶ Redes
- ▶ Bioinformática
- ▶ Mapas e logística
- ▶ Análise de dependências

Onde isso aparece na computação

Redes

Matrizes de latência entre todos os servidores de um data center ou de uma CDN orientam onde colocar réplicas e como balancear carga.

Bioinformática

Distâncias entre todas as espécies (ou genes) alimentam a construção de árvores filogenéticas por métodos de distância.

Mapas e logística

Tabelas de distância entre todas as cidades (o “mapa rodoviário”) são pré-calculadas para planejar rotas e entregas.

Análise de dependências

O fecho transitivo responde “A depende de B, direta ou indiretamente?” em compiladores, bancos de dados (consultas recursivas) e controle de acesso.

Para casa

Exercícios

1. (25.2-1) Execute Floyd-Warshall no grafo da Figura 25.2 e mostre cada $D^{(k)}$.
2. (25.2-4) Mostre que basta uma única matriz (sem os índices k).
3. (25.2-6) Use a saída de Floyd-Warshall para detectar um ciclo negativo.
4. (25.2-9) Calcule o fecho transitivo de um grafo direcionado em $O(V^3)$ com bits.
5. Troque a ordem dos laços para i, j, k . O resultado continua correto? Por quê?

Leitura

Cormen et al., seção 25.2.

Código

josedihego.net/estruturas