ESTRUTURAS DE DADOS II

MSC. DANIELE CARVALHO OLIVEIRA

DOUTORANDA EM CIÊNCIA DA COMPUTAÇÃO - USP

MESTRE EM CIÊNCIA DA COMPUTAÇÃO – UFU

BACHAREL EM CIÊNCIA DA COMPUTAÇÃO - UFJF

PROGRAMAÇÃO DINÂMICA

ALL-PAIRS SHORTEST PATHS

- E se quisermos descobrir o menor caminho entre quaisquer pares de vértices?
  - Aplicar o Algoritmo geral de Caminho Mínimo $|V|$ vezes? -- $O(|V|^2|E|)$
  - Algoritmo Floyd-Warshall

4 ALGORITMO FLOYD-WARSHALL

- O menor caminho $u \rightarrow w_1 \rightarrow \ldots \rightarrow w_l \rightarrow v$ possui l vértices, possivelmente $l = 0$;

- Iremos, então, suprimir nós intermediários
  - Logo $dist(u, v)$ é simplesmente a aresta que liga u a v

- A partir daí, gradualmente será expandido o conjunto de vértices intermediários permitidos

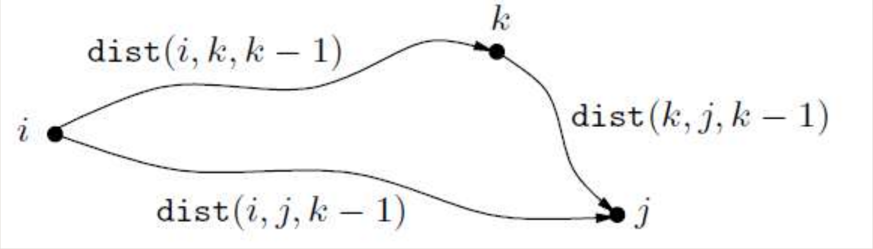
- Eventualmente todos os vértices serão permitidos em todos os caminhos.

5 ALGORITMO FLOYD-WARSHALL

- Definição:
  - Seja $V = \{1,2, \dots, n\}$
  - $dist(i, j, k)$ – tamanho do menor caminho entre i e j, no qual apenas os nós $\{1, 2, \dots, k\}$ podem ser usados como intermediários.
  - Inicialmente:
    - $dist(i, j, 0) = \begin{cases} C_{ij}, & se\ (i, j) \in E \\ \infty, & se\ (i, j) \notin E \end{cases}$

6 ALGORITMO FLOYD-WARSHALL

- Nosso subproblema, então será expandir o conjunto de nós intermediários adicionando o nó k
  - Neste ponto precisaremos reexaminar todos os pares i, j, e verificar se usar k como um nó intermediário produzirá um caminho mais curto entre i e j.
  - Considerando que nós já calculamos o tamanho do menor caminho de i à k e de k à j
  - $dist(i, j, k) = \min\{\, dist(i, k, k-1) + dist(k, j, k -$



7 ALGORITMO FLOYD-WARSHAL

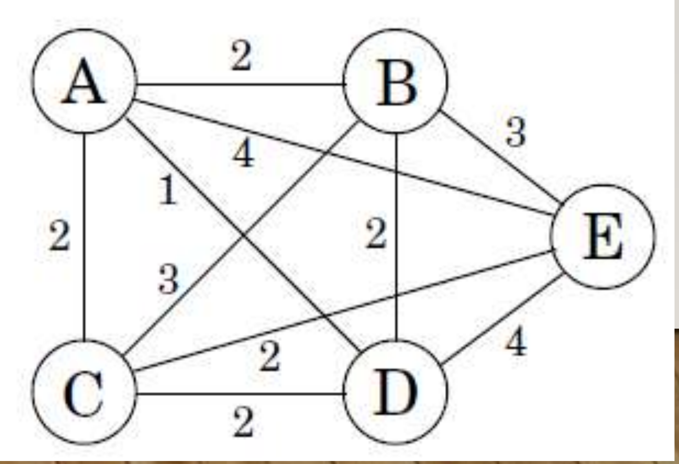
```
for i = 1 to n:
    for j = 1 to n:
        dist(i; j; 0) = ∞
for all (i, j) ∈ E:
    dist(i; j; 0) = Cij
for k = 1 to n:
    for i = 1 to n:
        for j = 1 to n:
            dist(i; j; k) = min{dist(i; k; k-1) + dist(k; j; k-1); dist(i; j;
            k-1)}
```

O PROBLEMA DO CAIXEIRO VIAJANTE

Traveling Salesman Problem

- "Um Caixeiro deve visitar apenas uma vez todas as n diferentes cidades partindo de uma cidade base, e retornando à ela. Qual caminho minimiza a distancia total percorrida pelo caixeiro?"

- É um problema difícil inclusive para computadores



9 O PROBLEMA DO CAIXEIRO VIAJANTE

- Usualmente o problema é dado por:
  - $G = (V, D)$, onde:
    - $V = \{1,2,\dots,n\}$ é o conjunto de cidades
    - D é a matriz de distancias entre cidades, com $\forall\ i, i \in V,\ i \neq j,$ $d_{ij} > 0$
  - O problema consiste em encontrar o percurso com distância mínima, começando em 1, passando por todas as n cidades e retornando a 1.

O PROBLEMA DO CAIXEIRO VIAJANTE

- Como Resolver?
  - Por Força – Bruta ??
    - Seria necessário verificar todas as possibilidades – (n-1)! Possibilidades → O(n!)
    - Tratável para problemas < 13
  - Programação Dinâmica
    - Nos trás uma solução bem mais rápida, mas ainda não é polinomial
    - Tratável para problemas de +- 30 cidades

O PROBLEMA DO CAIXEIRO VIAJANTE

- Como resolver por Programação Dinâmica?
  - Algoritmo Held-Karp

- Definindo o Subproblema:
  - Suponha que iniciamos na cidade 1, visitamos algumas cidades e agora alcançamos a cidade i, e faltam k cidades para visitar antes de retornar a 1.

  - Quais informações são necessárias para ampliarmos nosso caminho?
    - Precisamos conhecer i – pois nos ajudará a determinar qual cidade é mais conveniente visitar agora
    - Precisamos saber quais cidades já foram visitadas – de forma a não repeti-las

O PROBLEMA DO CAIXEIRO VIAJANTE

- Um subproblema apropriado seria, então:
  - Dado um subconjunto de cidades $S \subseteq V$, que inclui a cidade 1, e uma cidade $i \in S,\ j \neq 1$, seja $C(S, j)$ o tamanho do menor caminho que passa por cada nó de S exatamente uma vez, começando em 1 e terminando em j.

  - Considere:
    - Se $|S| = 2$, então $C(S, j) = d_{1,j}$, para $j = 2, 3, \dots, n$
    - Se $|S| > 2$, então $C(S, j) = o\ caminho\ ótimo\ de\ 1\ até\ i + d_{ij}\ ,\ \exists\ i \in S - \{j\}$

- $C(S, j) = min_{i \in S, i \neq j}\{C(S - \{j\}, i) + d_{ij}\}$

13

O PROBLEMA DO CAIXEIRO VIAJANTE

- $C(\{1\},1) = 0$
- $for\ s = 2\ to\ n$:
    - $para\ todos\ os\ subconjuntos\ S \subseteq \{1,2,\dots,n\}\ de\ tamanho\ s\ e\ contendo\ 1$:
        - $C(S,1) = \infty$
        - $for\ all\ j \in S,\ \ j \neq 1$:
        - $C(S,j) = min_{i \in S, i \neq j}\{C(S - \{j\}, i) + d_{ij}\}$
- $return\ min_j\ C(\{1,\dots,n\},j) + d_{j1}$

Existem no máximo $2^n \cdot n$ subproblemas, cada um com tempo linear para resolver. O Tempo total é $O(n^2 2^n)$

FIM DA AULA 21

Próxima aula:

Problemas NP-Completos