

1

Algoritmos em Grafos: Introdução

R. Rossetti, A.P. Rocha, J. Pascoal Faria
CAL, MIEIC, FEUP
Março de 2014

Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014

2

Índice

- ◆ **Revisão de conceitos e definições**
- ◆ **Exemplificar aplicações**
- ◆ **Representação**
- ◆ **Pesquisa em profundidade e em largura**
- ◆ **Ordenação topológica**

Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014

3

Revisão de conceitos e definições

Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014

4

Conceito de grafo

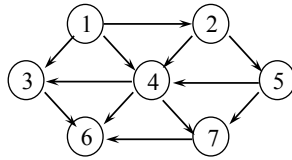
◆ Grafo $G = (V, E)$

- V – conjunto de vértices (ou nós)
- E – conjunto de arestas (ou arcos)
 - cada aresta é um par de vértices (v, w) , em que $v, w \in V$
 - se o par for ordenado, o grafo é dirigido, ou digrafo
 - um vértice w é adjacente a um vértice v se e só se $(v, w) \in E$
 - num grafo não dirigido com aresta (v, w) e, logo, (w, v) , w é adjacente a v e v adjacente a w
 - as arestas têm por vezes associado um custo ou peso

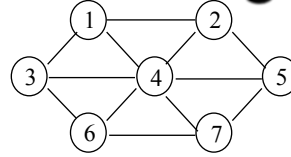
Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014

5

Grafos dirigidos e não dirigidos



G1 = (Cruzamentos, Ruas)



G2 = (Cidades, Estradas)

◆ Algumas aplicações

- Tráfego, transporte (controlo, gestão)
- Navegação GPS
- Abastecimento de água e redes de saneamento (gestão de carga)
- Gestão de redes de energia
- Workflows e cadeias de decisão
- Planeamento e gestão de projectos
- Compiladores, sistemas de ficheiros, jogos, criptografia, redes, Internet
- Redes Bayesianas e probabilísticas (Processo de Manchester)

Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014

6

Caminhos e ciclos

◆ Caminho - sequência de vértices v_1, \dots, v_n tais que $(v_i, v_{i+1}) \in E, 1 \leq i < n$

- comprimento do caminho é o número de arestas, $n-1$
- se $n = 1$, caminho reduz-se a um vértice e tem comprimento 0
- anel - caminho $v, v \Rightarrow (v, v) \in E$, comprimento 1; raro
- caminho simples - todos os vértices distintos excepto possivelmente o primeiro e o último

◆ Ciclo - caminho de comprimento ≥ 1 com $v_1 = v_n$

- num grafo não dirigido requer-se que as arestas sejam diferentes
- DAG (Grafo Dirigido Acíclico) é um grafo dirigido sem ciclo. Para qualquer vértice v , não há nenhuma ligação dirigida começando e acabando em v .

Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014

7

Conectividade e densidade

◆ Conectividade:

- Grafo não dirigido é conexo sse houver um caminho a ligar qualquer par de vértices
- Digrafo com a mesma propriedade: fortemente conexo, se p/ todo $v, w \in V$ existir em G um caminho de v para w e w para v .
- Digrafo fracamente conexo (ou não fortemente conexo): se, para toda bipartição (X, Y) de seu conjunto de vértices V , alguma aresta (x, y) tem uma ponta em X e outra em Y . Ou seja, não há bipartição vazia!

◆ Densidade:

- Grafo denso – $|E| = \Theta(V^2)$
 - Grafo completo – existe uma aresta entre qualquer par de nós
- Grafo esparso – $|E| = \Theta(V)$

Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014

8

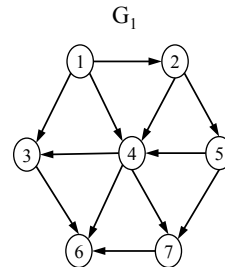
Representação

Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014

9

Matriz de adjacências

	1	2	3	4	5	6	7
1	0	1	1	1	0	0	0
2	0	0	0	1	1	0	0
3	0	0	0	0	0	1	0
4	0	0	1	0	0	1	1
5	0	0	0	1	0	0	1
6	0	0	0	0	0	0	0
7	0	0	0	0	0	1	0



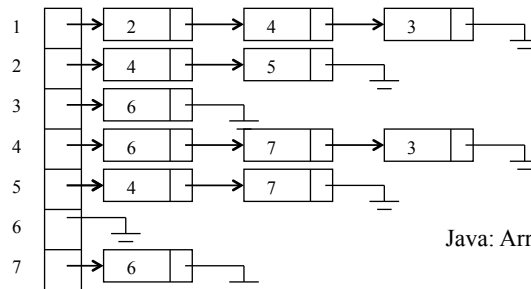
- $a[u][v] = 1$ sse $(u, v) \in E$
- elementos da matriz podem ser os pesos (0 - não há aresta)
- grafo não dirigido - matriz simétrica
- apropriada para grafos densos
 - 3000 cruzamentos x 12 000 troços de ruas (4 por cruzamento) = 9 000 000 de elementos na matriz!

Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014

10

Lista de adjacências

- ◆ Estrutura típica para grafos esparsos
 - para cada vértice, mantém-se a lista dos vértices adjacentes
 - vector de cabeças de lista, indexado pelos vértices
 - espaço é $O(|E| + |V|)$
 - pesquisa de adjacentes em tempo proporcional ao número destes
- ◆ Grafo não dirigido: lista com dobro do espaço

Java: `ArrayList<LinkedList<Integer>>`

Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014

11

Representação

- ◆ Normalmente precisamos de guardar informação adicional em cada vértice e em cada aresta (nome, peso, etc.), pelo que se opta por uma representação mais complexa, como por exemplo:

```
class Graph {  
    ArrayList<Vertex> vertexSet;  
}  
  
class Vertex {  
    String name;  
    LinkedList<Edge> adj; //arestas a sair deste vértice  
}  
  
class Edge {  
    Vertex dest;  
    double weight;  
}
```

Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014

12

Referências e mais informação

- ◆ “Data Structures and Algorithm Analysis in Java”, Second Edition, Mark Allen Weiss, Addison Wesley, 2006
- ◆ “Introduction to Algorithms”, Second Edition, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, The MIT Press, 2001

Algoritmos em Grafos: Introdução • CAL - MIEIC/FEUP, Março de 2014