Ciclos eulerianos e o problema do carteiro chinês

Marco A L Barbosa
malbarbo.pro.br

Departamento de Informática
Universidade Estadual de Maringá

http://github.com/malbarbo/na-grafos

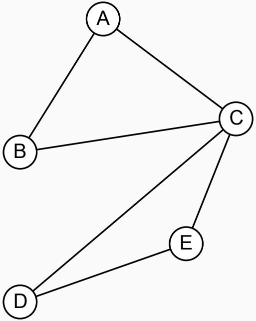
Um **caminho euleriano** é um caminho que usa cada aresta do grafo exatamente uma vez

Um **ciclo euleriano** é um ciclo que usa cada aresta do grafo exatamente uma vez

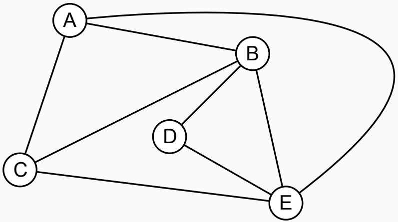
Um grafo que contém um ciclo euleriano é chamado de **grafo euleriano**

Um grafo que contém um caminho euleriano, mas não contém um ciclo euleriano é chamado de **grafo semi-euleriano**

Exemplos



Ciclo Euleriano: A, C, D, E, C, B, A



Caminho Euleriano: A, B, D, E, B, C, E, A, C

Lema 1

Dado um grafo não orientado conexo $G = (V, E)$ com todos os vértices de grau par, então qualquer par de vértices $u, v \in G$ faz parte de um ciclo sem arestas repetidas.

Prova (por contradição)

Suponha que exista um par de vértices $u, v \in G$ que não admita um ciclo em comum. Como o grafo é conexo, então existe um caminho p de u para v. Isto implica que deve existir uma aresta (x, y) no caminho p cuja a remoção torna o grafo desconexo, caso contrário existiria um outro caminho alternativo p' de u para v disjunto de p (formando um ciclo com u e v). A remoção da aresta (x, y) gera dois componentes, sendo que x e y pertencem a componentes distintos. Desta forma, x e y são os únicos vértices de grau ímpar em seus componentes, mas isto é uma contradição, pois o número de vértices de grau ímpar em um (sub)grafo deve ser par.

Teorema 1

Um grafo não orientado conexo G é um grafo euleriano se e somente se todo vértice de G tem grau par.

Prova (ida)

Seja $G = (V, E)$ um grafo euleriano e seja p um ciclo euleriano de G. Cada ocorrência de um vértice $v \in V$ em p, implica uma aresta que chega em v e uma aresta que sai de v. Como todas as arestas de E fazem parte de p, o número de arestas incidentes em cada vértice é par.

Prova (volta)

Seja $G = (V, E)$ um grafo com todos os vértices de grau par. Na construção de um caminho em G sempre é possível chegar e sair de um vértice por arestas ainda não utilizadas. Ou seja, é possível construir um ciclo arbitrário C a partir de um vértice qualquer v (Lema 1). Se C contém todas as arestas de G, temos um ciclo euleriano. Senão, construímos um grafo G', tal que $G'.E = G.E - \{\text{arestas de } C\}$. Em G' todos os vértices tem grau par, e pelo menos um vértice de C está em $G'.V$ e tem grau maior que 0 (senão o grafo não seria conexo). Recomeçamos este processo para o grafo G', começando com um vértice $v' \in C$ com grau maior que 0 e construímos um ciclo C'. Os ciclos C e C' podem ser unidos para formar um único ciclo. Continuando este processo até acabar as arestas do grafo, obteremos necessariamente um ciclo único que contém todas as arestas de G.

HIERHOLZER-1(G)

```
1  G' = (G.V, G.E)
2  v_0 = um vértice de G
3  C = caminho contendo apenas v_0
4  while G'.E ≠ ∅
5      u = vértice em C tal que d(u) > 0 em G'
6      U = ciclo em G' que contém u
7      C = C substituindo u por U
8      G'.E = E - { arestas de U}
9  return C
```

Análise do tempo de execução

- As operações da linhas 5 e 7 podem ser implementadas em tempo constante (usando lista duplamente encadeada), portanto o total destas linhas é $O(E)$
- O tempo total das linhas 6 e 8 (usando análise agregada) é $O(E)$
- Portanto, o tempo de execução do algoritmo é $O(E)$

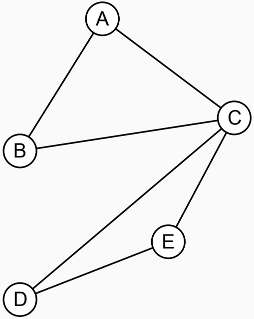
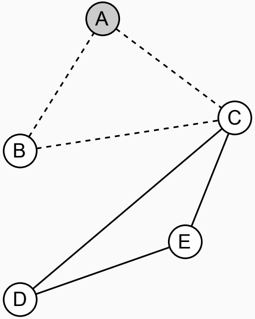
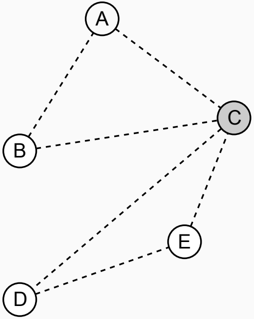


Figura 1



Vértice selecionado: A
Ciclo atual: A
Ciclo criado: A, B, C, A
Junção dos ciclos: A, B, C, A



Vértice selecionado: C
Ciclo atual: A, B, C, A
Ciclo criado: C, E, D, C
Junção dos ciclos:
A, B, C, E, D, C, A

Algoritmo de Hierholzer (primeira versão)

O procedimento HIERHOLZER-1 foi derivado diretamente da prova do Teorema 1, e por isto, podemos verificar facilmente que ele é correto. No entanto, a sua implementação é um pouco trabalhosa

A seguir, apresentamos uma versão do algoritmo de Hierholzer que utiliza uma pilha para auxiliar na construção do ciclo, o que facilita a implementação. No entanto a prova de corretude não é tão simples (fica como exercício)

Algoritmo de Hierholzer (segunda versão)

```
HIERHOLZER-2(G)
s = { um vértice de G } // pilha
C = {} // lista
while s ≠ ∅
    u = POP(s)
    PUSH-FRONT(C, u)
    while G.adj[u] ≠ ∅
        v = POP-FRONT(G.adj[u])
        if (u, v) não foi visitada
            PUSH(S, u)
            marcar a aresta (u, v) como visitada
            u = v
return C
```

Funcionamento

- s é o ciclo temporário
- C é o ciclo definitivo
- O laço da linha 6 constrói um ciclo que começa e termina com o vértice desempilhado na linha 4
- No caso de grafos não orientados, a remoção da aresta (u, v) da lista de adjacências u na linha 7 não garante que ela não será mais visitada (ela ainda está na lista de v). Utilizamos a marcação da linha 10 e a verificação da linha 8 para evitar que arestas sejam visitadas mais que uma vez

Análise do tempo de execução

- Usamos análise agregada
- Cada aresta é removida da lista de adjacências duas vezes. A operação de remoção pode ser implementada em tempo constante, desta forma, o tempo total das operações de remoção é $\Theta(E)$
- As operações POP, PUSH, PUSH-FRONT e POP-FRONT podem ser implementadas com tempo constante. Como o grafo de entrada é conexo e portanto $E > V - 1$, a quantidade de vezes que estas operações são realizadas é limitado por E, desta forma, o tempo total gasto com estas operações é $O(E)$
- Portanto, o tempo de execução do algoritmo é $O(E)$

Dado um grafo conexo com peso nas arestas, o **problema do carteiro chinês** consiste em encontrar um ciclo de peso mínimo que passe por cada aresta pelo menos uma vez

Aplicações

- Entrega de correspondência
- Coleta de lixo
- Nebulização no combate a dengue

Grafo euleriano

- Aplicar o algoritmo de Hierholzer

Grafo não euleriano

- Transformar o grafo em euleriano adicionando arestas artificiais e aplicar o algoritmo de Hierholzer

Se o grafo for semi-euleriano, adicionar uma aresta artificial que representa o caminho mínimo entre os dois vértices de grau ímpar (o caminho mínimo pode ser encontrado usando o algoritmo de Dijkstra)

Se o grafo tiver 4 ou mais vértices de grau ímpar

- Montar um grafo completo com os vértices de grau ímpar, onde cada aresta representa o menor caminho entre o par de vértices (algoritmo de Floyd-Warshall)
- Encontrar a melhor combinação de pares de vértices (emparelhamento perfeito, algoritmo de Edmonds de complexidade polinomial)

Referências

Caminho euleriano. Wikipédia. https://en.wikipedia.org/wiki/Eulerian_path

Problema do carteiro chinês. Wikipédia. https://en.wikipedia.org/wiki/Route_inspection_problem