# BCC204 - Teoria dos Grafos

Marco Antonio M. Carvalho

Departamento de Computação
Instituto de Ciências Exatas e Biológicas
Universidade Federal de Ouro Preto

# Conteúdo

## Fonte

Este material é baseado no livro

- Goldbarg, M., & Goldbarg, E. (2012). *Grafos: conceitos, algoritmos e aplicações*. Elsevier.

## Licença

# O Problema do Carteiro Chinês

## Introdução

Considere serviços como coleta de lixo ou correios. Os cruzamentos das ruas são vértices do grafo e as arestas são as ruas. Cada rua tem um custo de percurso associado que representa a distância, tempo ou outro fator.

É necessário percorrer **todas as ruas** e retornar ao **ponto inicial** com custo mínimo.

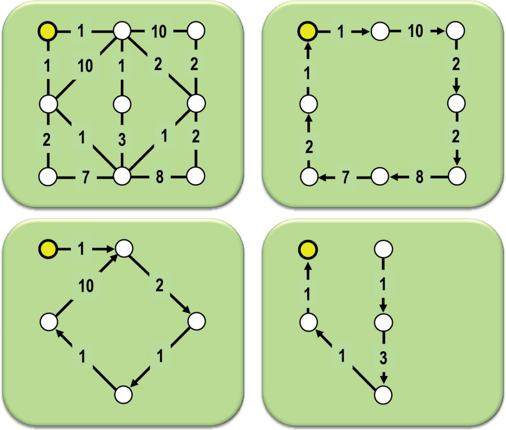
## Aplicações

- Coleta de lixo;
- Vendas em domicílio;
- Entrega do correio;
- Recenseamento;
- Nebulização contra dengue; etc...

# O Problema do Carteiro Chinês

## Definição

O Problema do Carteiro Chinês – PCC, consiste em determinar um passeio fechado de **custo mínimo** que passe por cada aresta de um grafo $G$ conectado pelo menos uma vez.




Grafo de exemplo e dificuldade em resolver o problema em grafos não eulerianos.

# Complexidade

## PCC em Grafos Não Orientados

Para o caso de grafos não orientados, a solução exata deste problema pode ser obtida em $O(n^3)$[a], portanto, em tempo polinomial.

[a] Papadimitriou & Steiglitz (1982)

## PCC em Grafos Orientados

Para o caso de grafos orientados, a solução exata deste problema pode ser obtida utilizando um algoritmo de fluxo em redes, portanto, em tempo polinomial.

## PCC em Grafos Mistos

Para o caso de grafos que possuem ambos os tipos de arestas, orientadas e não orientadas, o problema é NP-Difícil.

# Complexidade

## PCC em Grafos – Caso Simétrico

Para o caso de grafos nos quais as distâncias são iguais independente do sentido de travessia, o problema possui solução em tempo polinomial.

## PCC em Grafos – Caso Íngreme

Para o caso de grafos nos quais as distâncias dependem do sentido de travessia, o problema é NP-Difícil.

## Solução

Em grafos eulerianos e não orientados, a solução consiste em determinar o ciclo euleriano.

A solução deverá consistir em um *itinerário único*, de modo que caso o grafo não seja euleriano, algumas arestas serão percorridas *mais de uma vez*.

O processo de solução, no caso de trabalharmos com um grafo não euleriano, adiciona arestas até que se obtenha um grafo euleriano. Desta forma, indicamos quais arestas serão percorridas duas vezes.

# O Problema do Carteiro Chinês

## Abordagem para Grafos Não Orientados e Conexos

1. Verifique se $G$ é euleriano;
2. Caso positivo vá para 5;
3. Caso negativo, o grafo possui vértices de grau ímpar;
4. Adicione arestas ao grafo, duplicando as arestas que formam o caminho mais curto entre os vértices de grau ímpar, de modo que se tornem vértices de grau par;
5. Aplique um algoritmo de determinação de ciclos eulerianos.

## Fleury

A determinação da composição de ciclos eulerianos pode ser realizada em tempo determinístico polinomial.

Estudaremos o algoritmo de Fleury, que parte do princípio que o grafo é euleriano, ou seja, obedece ao Teorema de Euler.

# Algoritmo de Fleury

## Princípio

O algoritmo de Fleury, proposto em 1883, também utiliza um grafo reduzido induzido pelas arestas ainda não marcadas pelo algoritmo.

Inicialmente todas as arestas estão não marcadas, e, a partir de um vértice aleatório, uma aresta que obedeça a regra da ponte é escolhida para ser percorrida e inserida no ciclo.

## Regra da Ponte

Se uma aresta $\{v, w\}$ é uma ponte no grafo reduzido, então $\{v, w\}$ só deve ser escolhida pelo algoritmo de Fleury caso não haja outra opção.

## Terminologia

- $C$: conjunto das arestas que definem um ciclo euleriano no grafo.

# Algoritmo de Fleury

**Entrada:** Grafo $G = (V, A)$

**Escolha** qualquer vértice $v \in V$;

$C \leftarrow \{v\}$;

**repita**

> **Escolha** uma aresta $\{v, w\}$ não marcada usando a regra da ponte;
>
> **Atravessar** $\{v, w\}$;
>
> $C \leftarrow C \cup \{w\}$;
>
> **Marcar** $\{v, w\}$;
>
> $v \leftarrow w$;

**até** *que todas as arestas estejam marcadas*;

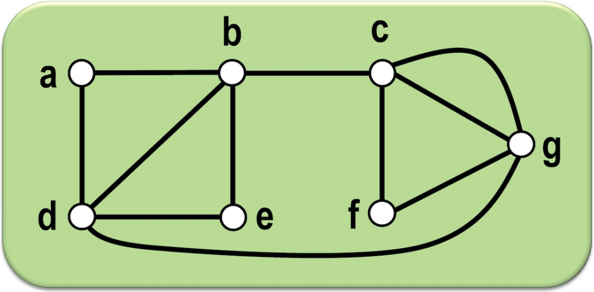
$C \leftarrow C \cup \{v\}$;

# Algoritmo de Fleury

## Complexidade

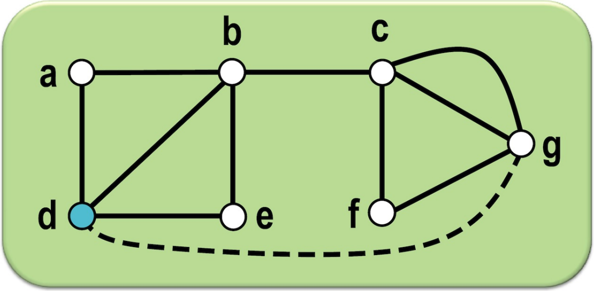
- $O(m)$ remoções de arestas;
- $O(m)$ para detecção de pontes (usando um algoritmo ingênuo);
- Resultando em complexidade $O(m^2)$.

# Exemplo



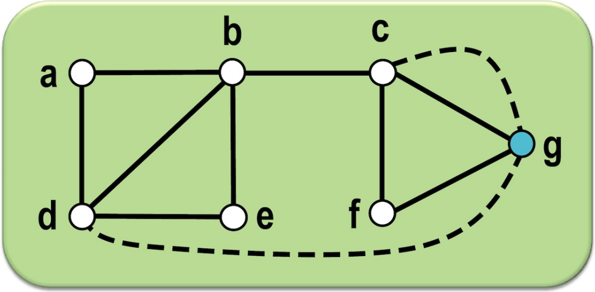

Grafo $G$.

# Exemplo



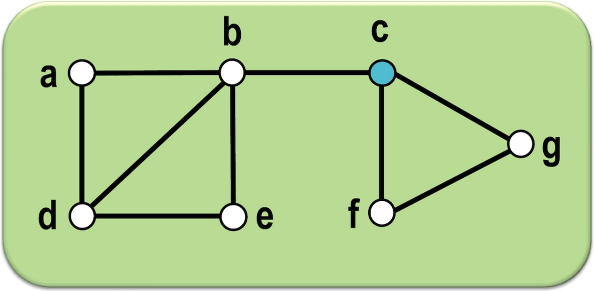

$$(v, w) = (d, g).$$

# Exemplo



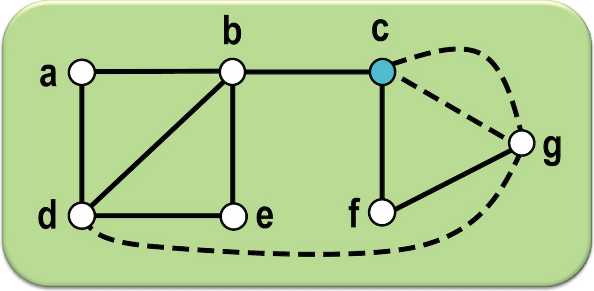

$$(v, w) = (g, c).$$

# Exemplo



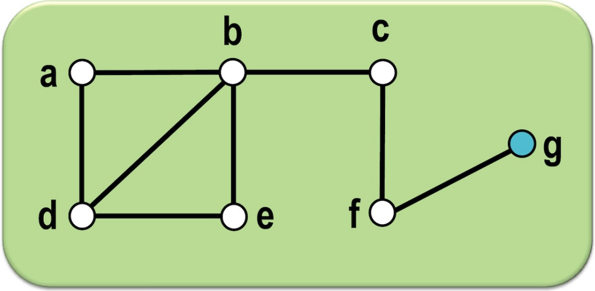

Grafo reduzido na terceira iteração.

# Exemplo



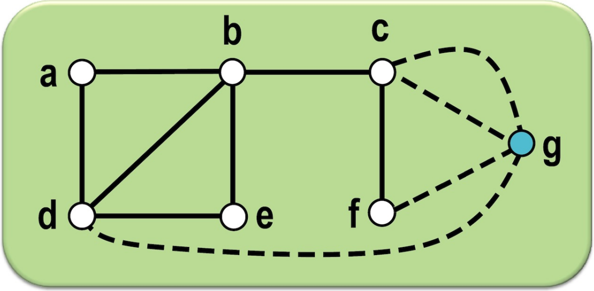

$$(v, w) = (c, g).$$

# Exemplo



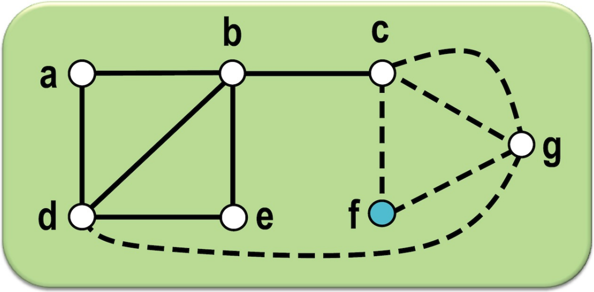

Grafo reduzido na quarta iteração.

# Exemplo



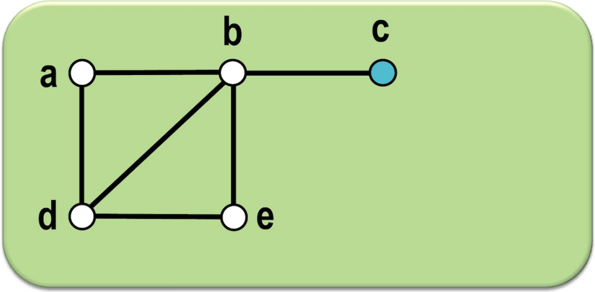

$$(v, w) = (g, f).$$

# Exemplo

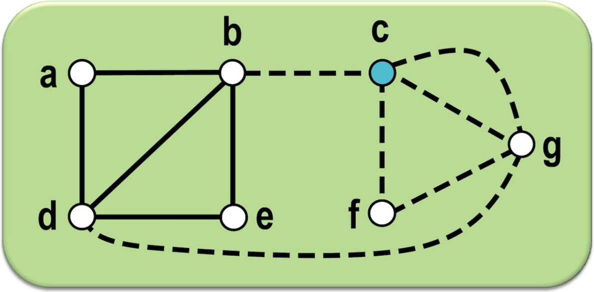



$$(v, w) = (f, c).$$

# Exemplo

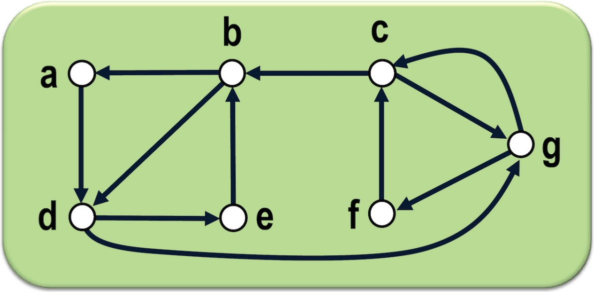



Grafo reduzido na sexta iteração.




$$(v, w) = (c, b).$$

# Exemplo




$$C = \{d, g, c, g, f, c, b, a, d, e, b, d\}.$$

# Dúvidas?