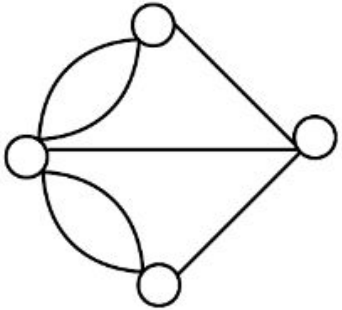
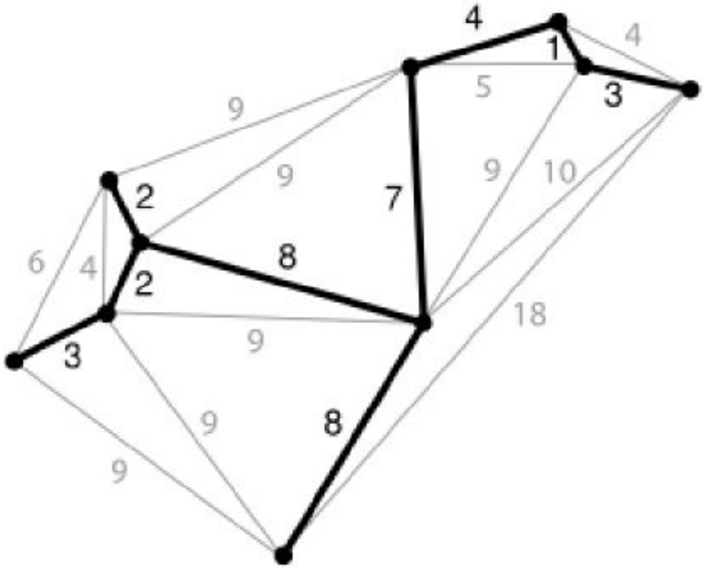


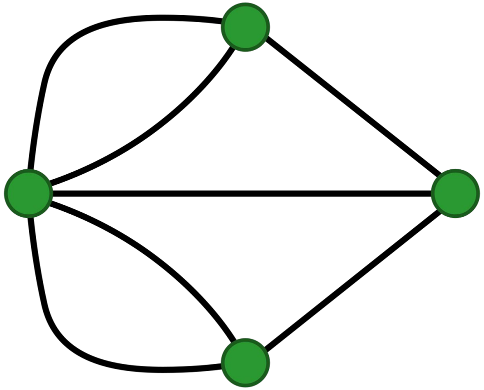
Grafos

Ciclos Hamiltonianos

Árvores Espalhadas Mínimas vs Eulerianos



Árvores Espalhadas Mínimas
Subgrafo conectado acíclico que liga todos os vértices do grafo



Ciclo Euleriano
Achar um ciclo que passa por todas as arestas, sem repetir aresta

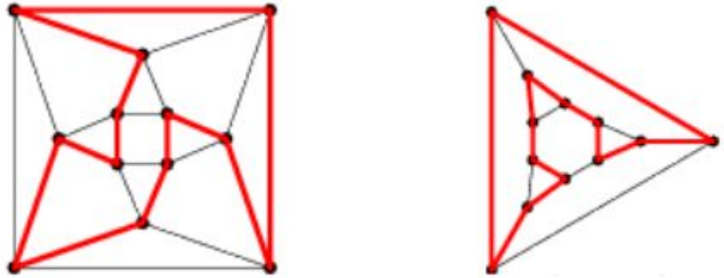
Definições

Ciclo Hamiltoniano

- Caminho que passa exatamente uma vez por cada vértice e retorna ao vértice inicial

Grafo Hamiltoniano

- Todo grafo que contém um ciclo Hamiltoniano



Dois grafos Hamiltonianos
Em vermelho, o **ciclo Hamiltoniano**

Eulerianos x Hamiltonianos

Um **Grafo Hamiltoniano** tem um ciclo que visita **todos os vértices** exatamente uma vez

- Pode não visitar todas as arestas

Um **Grafo Euleriano** tem um ciclo que visita **todas as arestas** exatamente uma vez

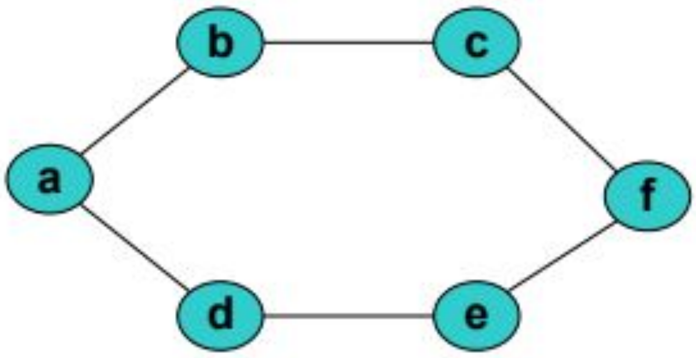
- Pode não visitar ou até repetir vértices

São conceitos parecidos, mas independentes

- Um grafo pode ser de um dos tipos e não ser do outro

Exemplo

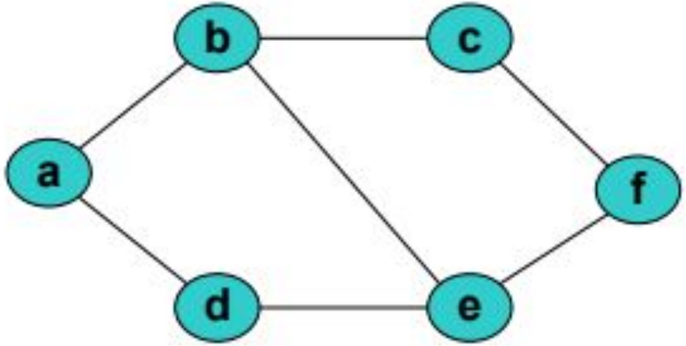
Grafo Hamiltoniano e Euleriano



Ciclo Hamiltoniano e Euleriano
a → b → c → f → e → d → a

Exemplo

Grafo Hamiltoniano não-Euleriano

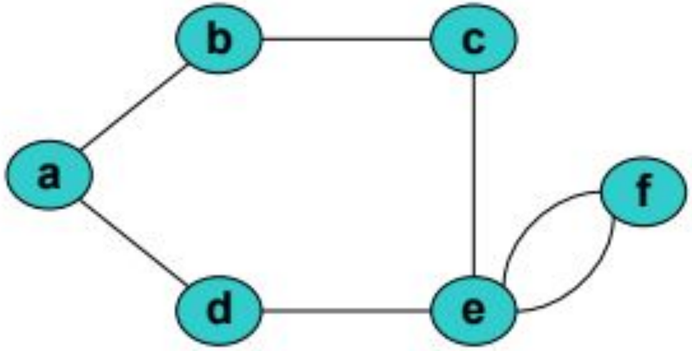


Ciclo Hamiltoniano

a → b → c → f → e → d → a

Exemplo

Grafo Hamiltoniano não-Euleriano



Ciclo Hamiltoniano

a → b → c → **e** → f → **e** → d → a

Grafos Hamiltonianos

Algumas condições suficientes para um grafo não-direcionado <u>conectado</u> G ser Hamiltoniano

- G é completo, com $|V| > 2$
- Ou para todo par de vértices u e v não-adjacentes, vale $\mathrm{grau}(u) + \mathrm{grau}(v) > |V|$

Grafo Conectado

Se existir pelo menos um caminho entre cada par de vértices

Grafo Completo

Todo vértice é adjacente a todos os outros vértices

Porém, essas condições não são **necessárias**

- Um grafo pode não satisfazer essas condições e ainda assim ser Hamiltoniano...

Algoritmos

Diferente do problema de achar um ciclo Euleriano, não existem (ainda) algoritmos eficientes para o problema de achar um ciclo Hamiltoniano

Este é um problema classificado como **NP-Completo**

- Classe de problemas para os quais não são conhecidos algoritmos eficientes (e talvez não existam)
- Os algoritmos conhecidos são exponenciais

Problema Relacionado

Um problema relacionado ao problema de achar um ciclo Hamiltoniano é o

Problema do Caixeiro-Viajante

Problema do Caixeiro-Viajante

- Em um grafo completo e valorado representando cidades e estradas entre elas
- Um vendedor (caixeiro) planeja viajar por todas as cidades e retornar
- Para minimizar seus custos, o caminho **não deve repetir vértices** e **ser o de menor custo possível**

Problema do Caixeiro-Viajante

- O problema consiste em achar um **ciclo Hamiltoniano de custo mínimo** em um grafo completo e valorado
- Este é um problema de grande importância teórica na Computação

Algoritmos

- Também para o problema do caixeiro-viajante (**achar um ciclo Hamiltoniano de custo mínimo**) não existem algoritmos eficientes
- É um problema **NP-Difícil**
    - Em teoria, os melhores algoritmos possíveis são, pelo menos, tão ineficientes quanto os algoritmos de problemas NP-Completos

Algoritmos de Aproximação

Ao invés dos algoritmos exatos (que são ineficientes) podem ser usados **algoritmos de aproximação**

Porém, as respostas dos algoritmos de aproximação não são perfeitas: acha uma solução “boa”, mas não a “melhor”

Para o caixeiro-viajante, são usadas técnicas de otimização de soluções, estudadas em IA

Algoritmos de Aproximação

As técnicas de **otimização** de soluções partem de uma solução inicial qualquer, possivelmente de qualidade “ruim”

Depois, tentam modificá-la pouco a pouco para achar variações melhores, até certo limite

- ***Hill-Climbing*** (“escalando encosta”)

Hill-Climbing

Começa gerando solução inicial aleatória

- No caso, um ciclo qualquer que use todos os vértices

Depois, **faz um loop**: a cada iteração, aplica um "operador" que gera uma nova solução ligeiramente modificada

- Por exemplo, pode trocar a posição de dois vértices no ciclo para criar um novo ciclo

Ao final da iteração, descarta todas as soluções e fica só com a menor delas

O algoritmo para quando nenhuma das novas soluções for melhor do que a anterior

Hill-Climbing

O algoritmo é bastante eficiente, podendo ser usado com grafos relativamente grandes

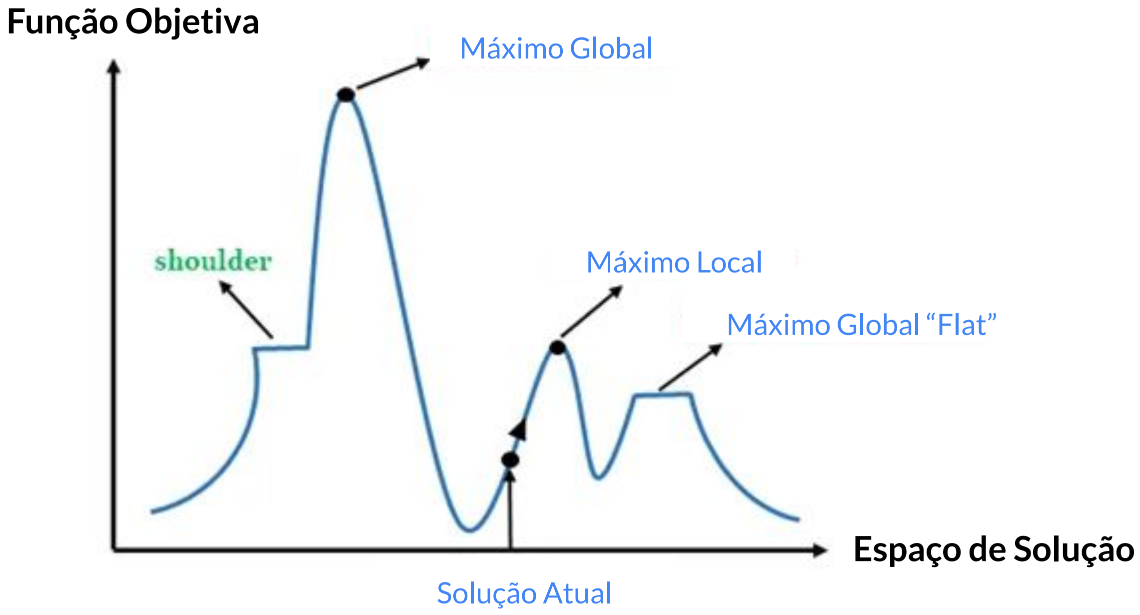
Porém, a qualidade da solução final pode variar de acordo com a solução inicial escolhida

Em alguns casos, a solução final é muito distante da “melhor”

- **Diz-se que o algoritmo ficou preso em um “mínimo local”**

Recomenda-se reiniciá-lo várias vezes mudando a solução inicial, para evitar esse problema

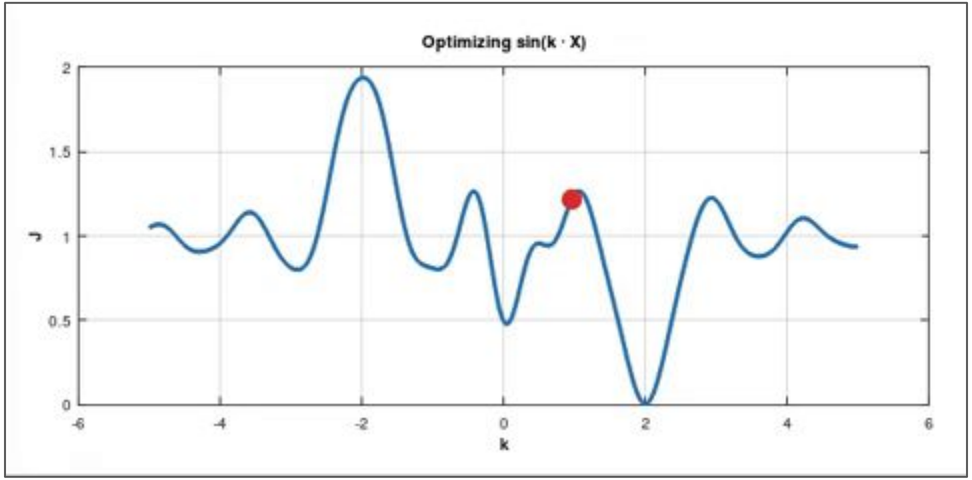
Hill-Climbing



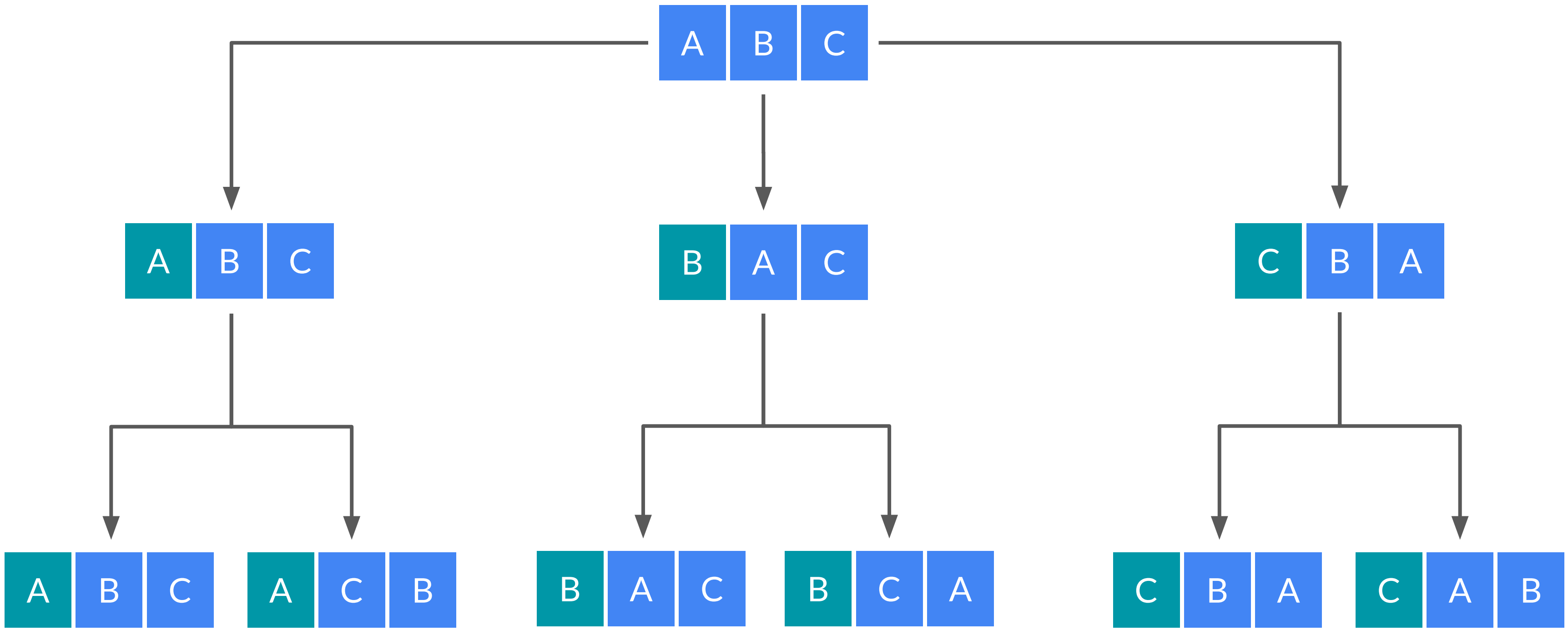
Hill-Climbing

Em alguns casos, a solução final é muito distante da "melhor"

- **Diz-se que o algoritmo ficou preso em um "mínimo local"**



Análise Combinatória



Análise Combinatória