HEURÍSTICAS CONSTRUTIVAS

DCE770 - Heurísticas e Metaheurísticas

Atualizado em: 25 de agosto de 2025

Iago Carvalho

Departamento de Ciência da Computação

HEURÍSTICAS

Uma heurística é um procedimento mental simples que ajuda a encontrar respostas adequadas, embora várias vezes imperfeitas, para perguntas difíceis

Em computação, uma heurística tem outro sentido

- É um algoritmo aplicado a problemas exponenciais
  - NP-Completos
- Não garante a solução ótima do problema
- Entretanto, visa obter soluções de boa qualidade

EXPLOSÃO COMBINATÓRIA

Problemas NP-Completos normalmente possuem um número exponencial de soluções

- Não possuem sub-estrutura ótima

É inviável realizarmos a exploração deste conjunto completo

- Seria problema $\#P$-Completo

Heurísticas são algoritmos que realizam uma busca (parcial) neste espaço de soluções

- Uma heurística difere da outra pela maneira como ela realiza esta busca

CLASSES DE HEURÍSTICAS

Existem 4 grandes classes de algoritmos heurísticos

1. **Heurísticas construtivas**
2. Heurísticas de busca local
3. Heurísticas populacionais
4. Algoritmos aproximativos

Estas 4 abordagens são complementares. Normalmente, um bom algoritmo heurístico se utiliza de dois ou mais destas classes.

HEURÍSTICAS CONSTRUTIVAS

Constroem uma solução para um problema de otimização

- Muitas vezes, constroem a solução do zero
- Utilizam a definição do problema e os parâmetros da instância

Complexidade polinomial

Na maioria das vezes, garantem uma solução viável

- Nunca garantem a otimalidade

HEURÍSTICA CONSTRUTIVA

Em geral, uma heurística construtiva é um algoritimo guloso

Algoritmo 1: Estrutura geral de uma heurística construtiva

```
   Entrada: C          // Conjunto de elementos candidatos
1  s ← ∅                          // Inicializa a solução vazia
2  enquanto s não for uma solução completa faça
3  |  c ← melhor(C)          // Obtém melhor elemento de C
4  |  s ← s ∪ c                 // Insere o elemento c em s
5  retorne s
```

ELEMENTOS DE UM ALGORITMO GULOSO

Um algoritmo guloso pode ser decomposto em cinco componentes

1. **Conjunto candidato**: O universo de todos os elementos básicos a partir dos quais a solução final será montada
2. **Função de seleção**: É o critério que, a cada passo, avalia os candidatos disponíveis e escolhe o "melhor"ou "mais promissor"para ser adicionado à solução parcial.
3. **Função de viabilidade**: Um mecanismo de verificação que garante que a adição de um candidato selecionado não viola nenhuma das restrições fundamentais do problema
4. **Função objetivo**: função que mede a qualidade de uma solução, seja ela parcial ou completa, e que a função de seleção tenta otimizar localmente
5. **Função de solução**: Um critério de parada que determina quando a solução parcial se tornou uma solução completa e viável para o problema

CONJUNTO CANDIDATO E SOLUÇÃO

O conjunto candidato C é de extrema importância

- Ele é utilizado para representar uma solução para o problema

Um conjunto candidato contém algum elemento do problema

- Um vértice
- Uma aresta
- Algum outro elemento

Já o conjunto s representa uma solução parcial ou completa

Solução parcial

- Um subgrafo
- Parte de um caminho
- Componentes desconectados

HEURÍSTICAS CONSTRUTIVAS

Começa de uma solução inicial

- Na maioria das vezes, uma solução vazia

Adiciona elemento por elemento de forma gulosa

- Cada adição de um elemento aumenta a solução parcial

Finaliza quando obtiver uma solução completa

DECISÕES LOCAIS E MÍOPES

Algoritmos gulosos no gerais são *míopes*

- Tomam decisões ótimas locais
- Estas podem ser ruins globalmente

Além disso, elas também são dependentes da solução inicial

- São algoritmos determinísticos
- Uma mesma solução inicial leva a uma mesma solução heurística

ALGORITMOS GULOSOS SEMI-ALEATÓRIOS

Podemos usar uma aleatoriedade controlada para construir heurísticas não-determinísticas

E se, em vez de sempre escolher o **melhor** candidato, nós déssemos uma chance a outros candidatos **bons**?

A ideia central de uma heurística semi-aleatória (ou semi-gulosa) é que ela seleciona o melhor candidato com uma probabilidade r

- Da mesma forma, com uma probabilidade de $1 - r$, ela seleciona um elemento candidato aleatório
  - Este deve respeitar a função de viabilidade

HEURÍSTICA CONSTRUTIVA SEMI-ALEATÓRIA

Algoritmo 2: Heurística semi-aleatoriezada

```
   Entrada: C          // Conjunto de elementos candidatos
   Entrada: r     // Nível de aleatoriedade da heurística
 1 s ← ∅                          // Inicializa a solução vazia
 2 enquanto s não for uma solução completa faça
 3 |   a ← U(0,1)                 // Número uniforme entre 0 e 1
 4 |   se a < r então
 5 |   |   c ← melhor(C)      // Obtém melhor elemento de C
 6 |   caso contrário
 7 |   |   c ← aleatório(C)       // Obtém elemento aleatório
 8 |   c ← melhor(C)             // Obtém melhor elemento de C
 9 |   s ← s ∪ c                    // Insere o elemento c em s
10 retorne s
```

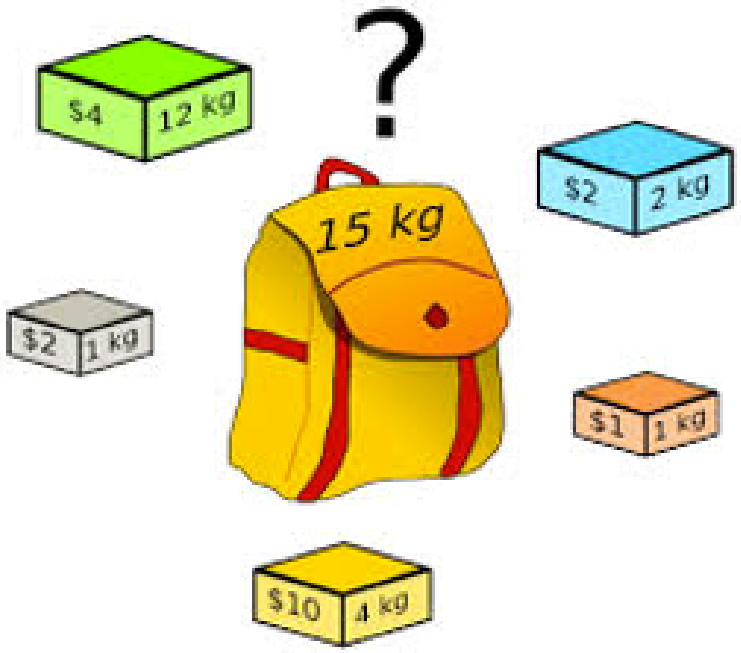
EXEMPLOS

APLICAÇÃO - PROBLEMA DA MOCHILA BINÁRIA

Seja I um conjunto de elementos

- Cada elemento $i \in I$ é associado a um peso p_i e a um benefício b_i

Problema da mochila binária: Encontrar $X \subseteq I$ tal que $\sum_{i \in X} b_i$ é máximo e que $\sum_{i \in X} p_i \leq c$, para uma constante c qualquer

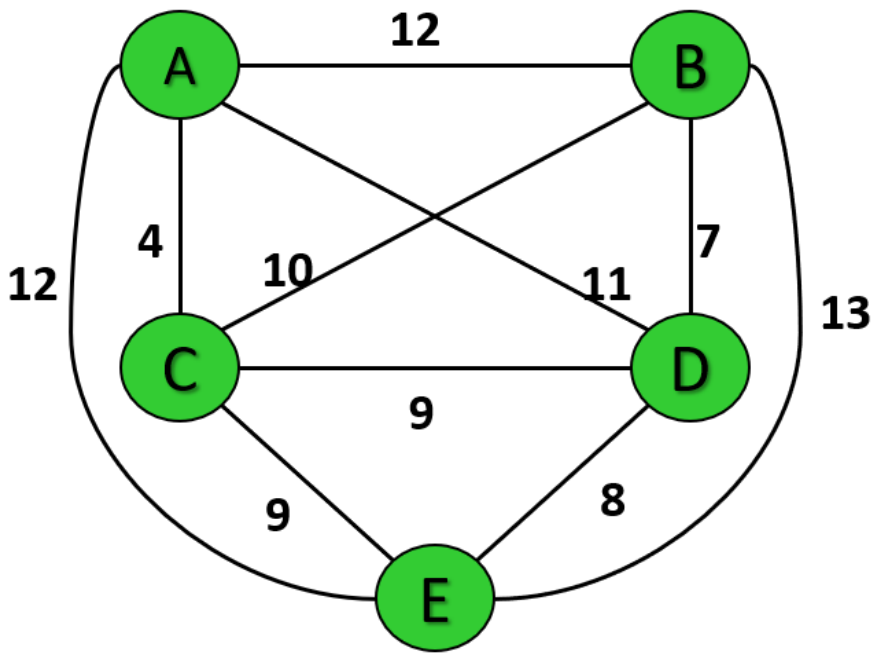


APLICAÇÃO - PROBLEMA DO CAIXEIRO VIAJANTE

Seja $G = (V, E)$ um grafo

- Cada aresta $e \in E$ é associada a um peso w_e

Problema do caixeiro viajante: Encontrar um subgrafo $G' = (V, X)$, onde $X \subseteq E$, tal que o grau de todos os vértices seja igual a 2 e que $\sum_{e \in X} w_e$ é mínimo



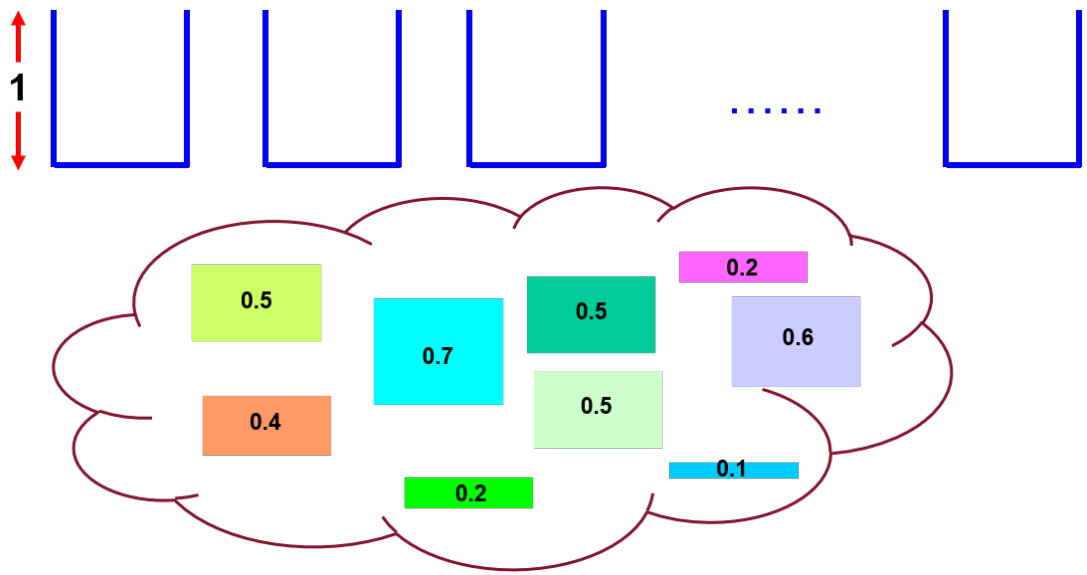
APLICAÇÃO - PROBLEMA DO EMPACOTAMENTO

Seja I um conjunto de elementos

- Cada elemento tem um tamanho $i \in I$ tem um tamanho t_i

Problema do empacotamento: Alocar todos os elementos de I em "potes"de tamanho 1

- Utilizar o menor número de "potes"o possível



FINALIZANDO...

VANTAGENS E DESVANTAGENS

Heurísticas construtivas são **eficientes**

- Baixa complexidade computacional
- Capazes de encontrar uma solução viável para problemas NP-Completos

Elas também são **simples**

- Fáceis de programar e de entender

Úteis para gerar **soluções iniciais**

- Espera-se que sejam de boa qualidade

VANTAGENS E DESVANTAGENS

Suas decisões *míopes* podem levar a resultados indesejados

- Decisões míopes são ótimas para problemas com sub-estrutura ótima
- Não garantem a otimalidade de problemas NP-Completos

Suas escolhas são **irreversíveis**

- Caso um elemento seja adicionado, ele não pode mais ser removido
- Válido tanto para a determinística quanto para a semi-aleatória