

SuperComputação

Aula 06 – Algoritmos aleatorizados

2021 – Engenharia

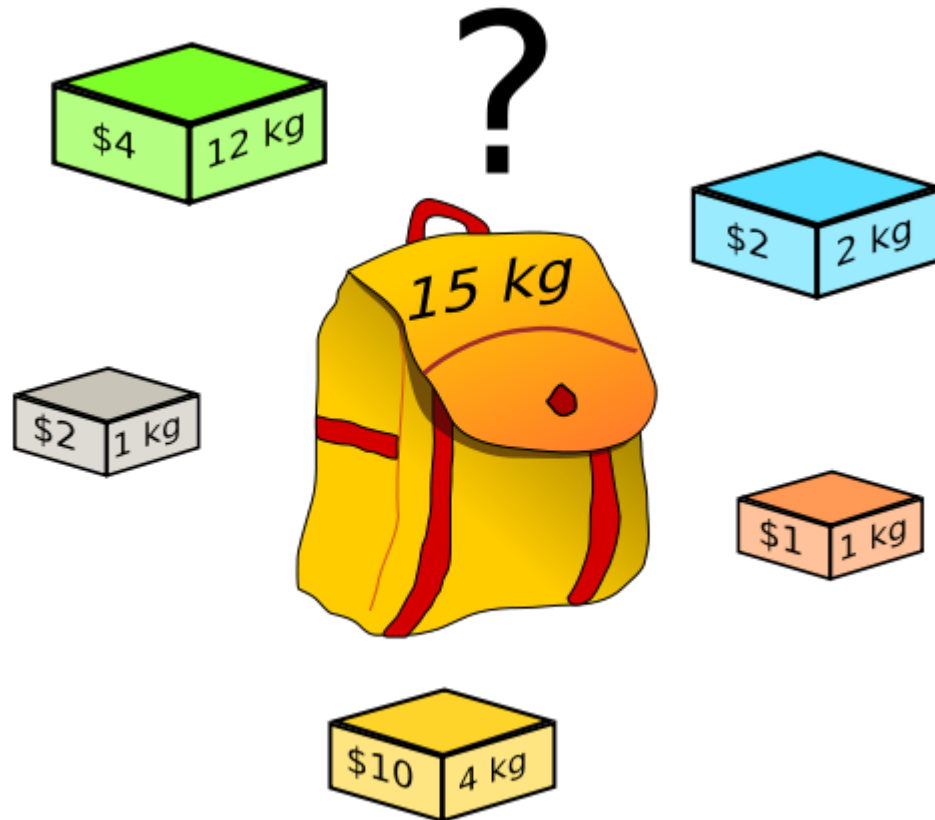
André Filipe de Moraes Batista <andrefmb@insper.edu.br>

Hoje

- Algoritmos aleatorizados
- Exploration / Exploitation

Algoritmos aleatorizados

A mochila binária



<https://commons.wikimedia.org/wiki/File:Knapsack.svg>

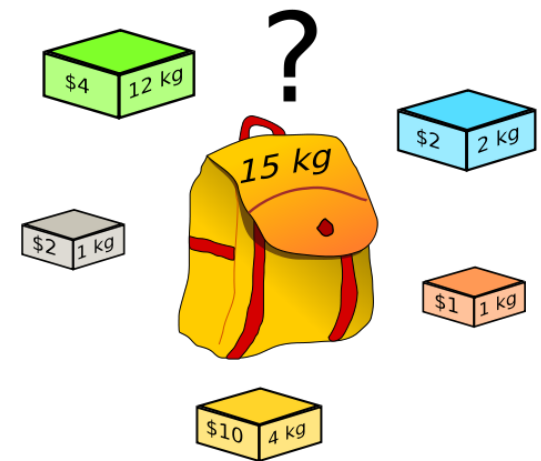
A mochila binária

Quais escolhas podem ser feitas?

- Quais produtos pegar?

Qual é a função objetivo?

- Maximizar valor dos objetos guardados



Quais são as restrições?

- Peso dos objetos não pode exceder capacidade da mochila

Como resolver esse problema?

Algumas opções:

- tentar tudo e ver qual é melhor
- pegar o mais caro primeiro
- pegar o mais leve primeiro

Heurística

"truque" usado para resolver um problema rapidamente

Ainda assim, uma boa heurística é suficiente para obter resultados aproximados ou ganhos de curto prazo.

- Não garante resultados ótimos
- Nem resultados bons em todas situações

Heurísticas - limitações

1. E se a solução gerada não for boa? Consigo "tentar" de novo e gerar outras parecidas?
2. Será que é possível melhorar a solução gerada? Como?

Exploration x Exploitation



Exploration

Exploration:

- decisão não localmente ótima feita "de propósito"
- visa adicionar variabilidade nas soluções geradas



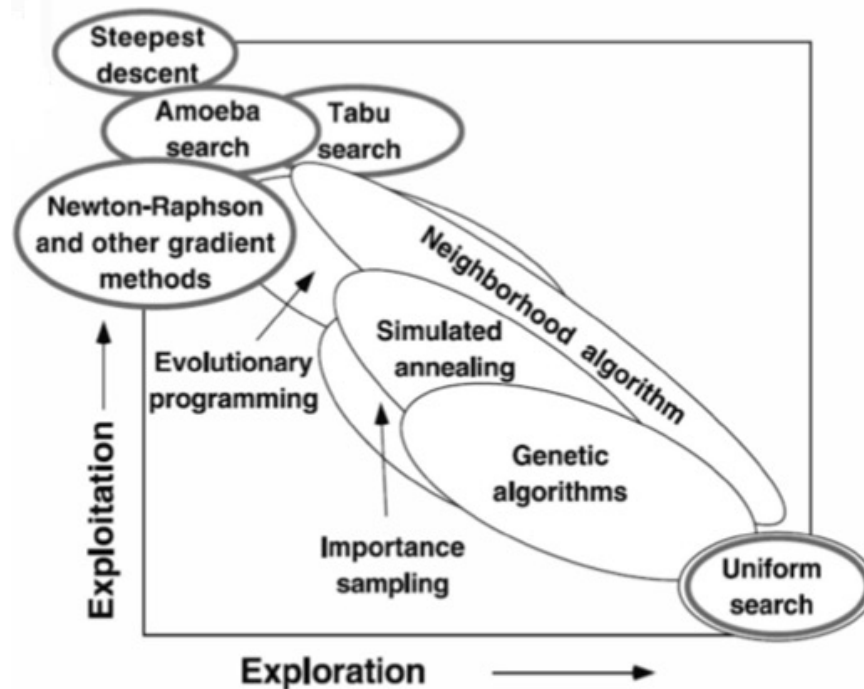
Exploitation

Exploitation:

- explorar alguma **propriedade do problema**
- pode ser uma intuição que leve a bons resultados em curto prazo

Exploration x Exploitation

Exploration vs. exploitation

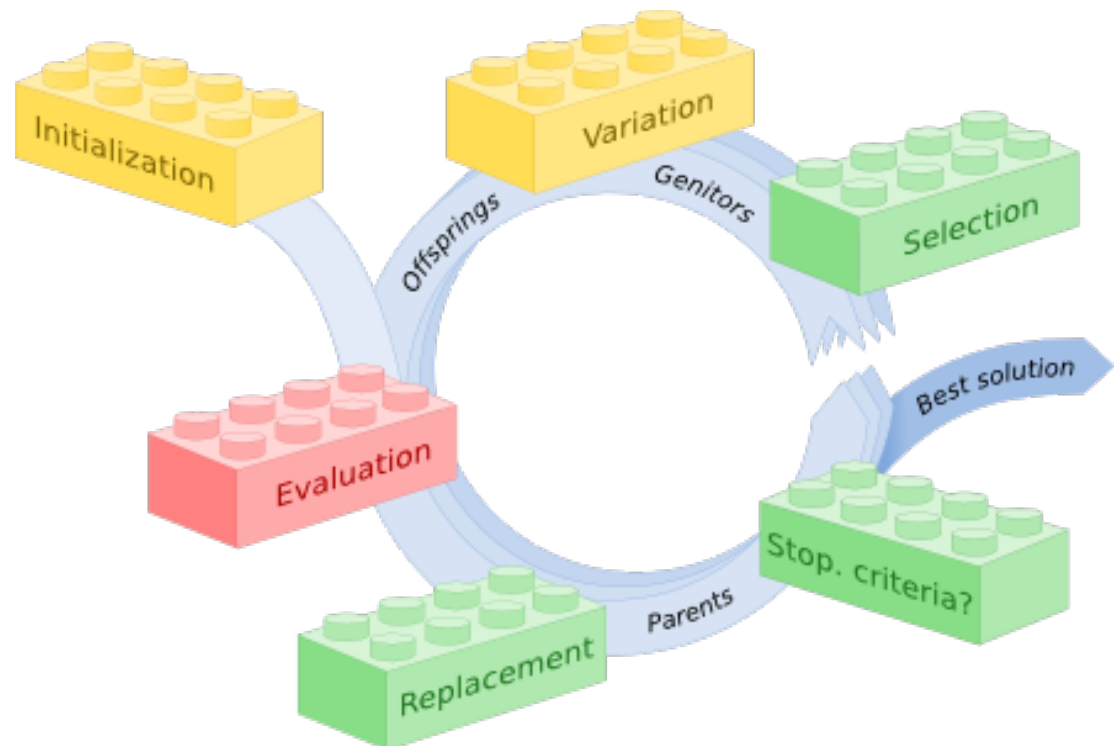


picture courtesy of Heriot-Watt university

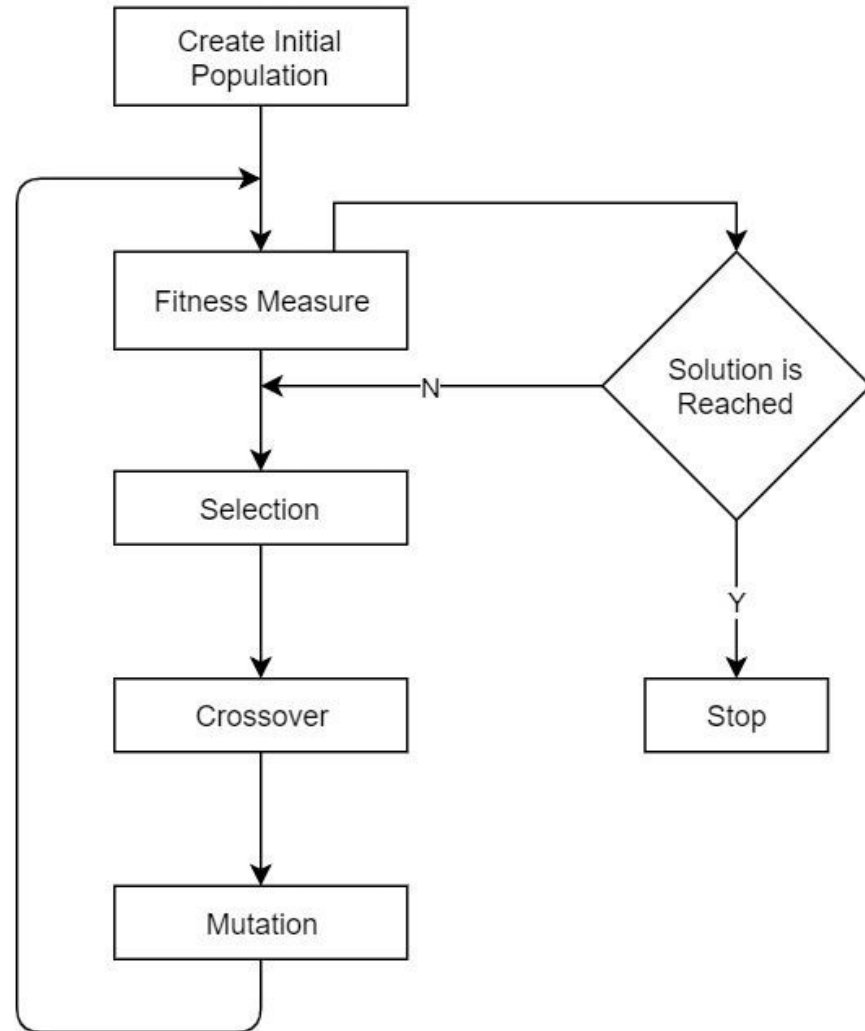
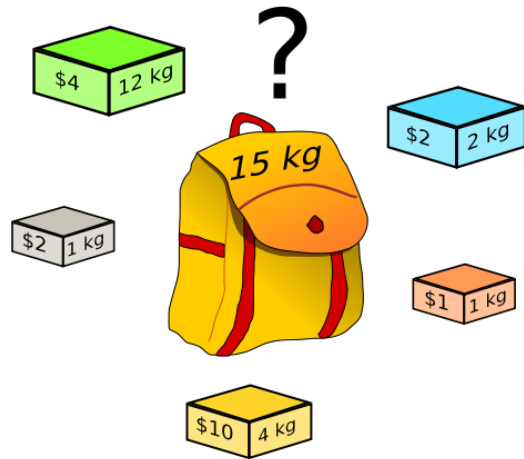
Exemplo de Exploration vs. Exploitation



Algoritmos Genéticos



Algoritmos Genéticos para a Mochila Binária



Exploration vs Exploitation em Algoritmos Genéticos

- Estratégia de Reprodução (cross-over)
- Taxa de mutação
- Estratégia de elitismo

Possíveis itens:

Item	Peso	Valor
1	12	32
2	2	29
3	1	29
4	5	4
5	6	2
6	4	20
7	7	34
8	1	39
9	1	48
10	2	44

Capacidade da mochila: 35

População Inicial
(n=8)

Tamanho da população = (8, 10)

População inicial:

```
[[1 0 1 1 0 0 0 1 1 1]
 [1 0 0 0 0 1 1 0 1 1]
 [0 0 0 0 1 0 1 1 1 1]
 [0 0 0 0 0 1 0 1 0 0]
 [1 1 1 1 0 0 1 0 1 0]
 [1 0 0 0 1 1 1 1 0 0]
 [1 1 1 0 0 0 1 0 0 0]
 [0 1 1 0 1 1 0 1 0 1]]
```

array([[196],
[178],
[167],
[59],
[176],
[127],
[124],
[163]])

Fitness

Seleção

```
[1 0 1 1 0 0 0 1 1 1] [1 0 0 0 0 1 1 0 1 1]
[0 0 0 0 1 0 1 1 1 1] [1 1 1 1 0 0 1 0 1 0]
```

Cross-Over

```
[1 0 1 1 0 | 0 0 1 1 1] [1 0 1 1 0 | 1 1 0 1 1]
[1 0 0 0 0 | 1 1 0 1 1] [1 0 0 0 0 | 0 0 1 1 1]
```

Mutação

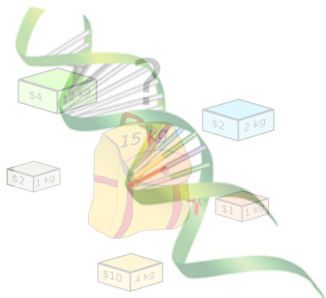
```
[1 0 0 0 0 0 0 1 1 1] [1 0 0 0 1 0 0 1 1 1]
```

Repetir
por N
gerações

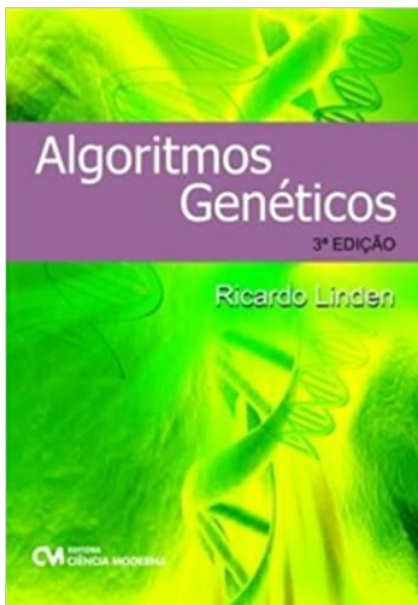
$$fitness = \sum_{i=1}^n c_i v_i; \text{ if } \sum_{i=1}^n c_i w_i \leq kw$$

fitness = 0; otherwise

Mochila Binária por Algoritmos Genéticos



Para saber mais sobre algoritmos genéticos



<https://www.algoritmosgeneticos.com.br>

No mundo real...

NASA ST5 spacecraft antenna

Automated Antenna Design with Evolutionary Algorithms

Gregory S. Hornby* and Al Globus

University of California Santa Cruz, Mailtop 269-3, NASA Ames Research Center, Moffett Field, CA

Derek S. Linden

JEM Engineering, 8683 Cherry Lane, Laurel, Maryland 20707

Jason D. Lohn

NASA Ames Research Center, Mail Stop 269-1, Moffett Field, CA 94035

G. S. Hornby, J. D. Lohn, and D. S. Linden

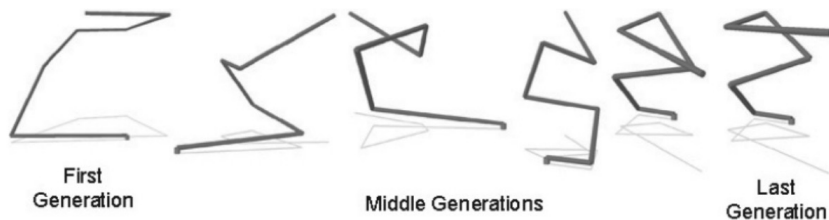


Figure 10: Sequence of evolved antennas leading up to antenna ST5-33.142.7.



Exploration x Exploitation

Nossa heurística é **100% Exploitation**.

Como podemos adicionar **Exploration**?

Exploration x Exploitation

Nossa heurística é **100% Exploitation**.

Como podemos adicionar **Exploration**?

1. Alternar heurísticas de vez em quando
2. De vez em quando faço uma escolha qualquer
3. Inverto a heurística de vez em quando

Exploration x Exploitation

Nossa heurística é **100% Exploitation**.

Como podemos adicionar **Exploration**?

1. Alternar heurísticas **de vez em quando**
2. **De vez em quando** faço uma escolha qualquer
3. Inverto a heurística **de vez em quando**

Exploration

Exploration requer a capacidade de criar um programa que execute de maneira diferente a cada execução.

Precisamos

1. de uma fonte de aleatoriedade;
2. uma maneira de gerar sequências de números aleatórios

Números aleatórios

Um gerador de números aleatórios é impossível de ser criado usando um computador:

- 1.É impossível prever qual será o próximo número aleatório "de verdade"
- 2.Um computador executa uma sequência de comandos conhecida baseada em dados guardados na memória.
Execução é **Determinística**.

Números (pseudo-)aleatórios

Gerador de números pseudo-aleatórios (**pRNG**): algoritmo determinístico que gera sequências de números que parecem aleatórias

1. **Determinístico:** produz sempre a mesma sequência.
2. **Sequências que parecem aleatórias:** não conseguiríamos distinguir uma sequência gerada por um pRNG e uma sequência aleatória de verdade.

Números (pseudo-)aleatórios

Sorteio de números aleatórios

1. **Gerador:** produz bits aleatórios a partir de um parâmetro **seed**. Cada **seed** gera uma sequência diferente de bits.
2. **Distribuição de probabilidade:** gera sequência de números a partir de um conjunto de parâmetros



Atividade prática

Aleatorizando a mochila binária (45 minutos)

1. Adicionar aleatoriedade em nossas heurísticas



Comentários sobre RNGs



Atividade prática

E se fosse tudo aleatório?

1. Criando uma solução completamente aleatória

Fechamento

Adicionar aleatoriedade melhorou os resultados?

Qual a qualidade das soluções aleatórias?

7

Insper

www.insper.edu.br