

# Aula 6.1

## Roteiro

- Algoritmos randomizados
- Exemplos
- Algoritmo de Monte Carlo
- Algoritmos de Las Vegas
- Comparação
- Aplicações

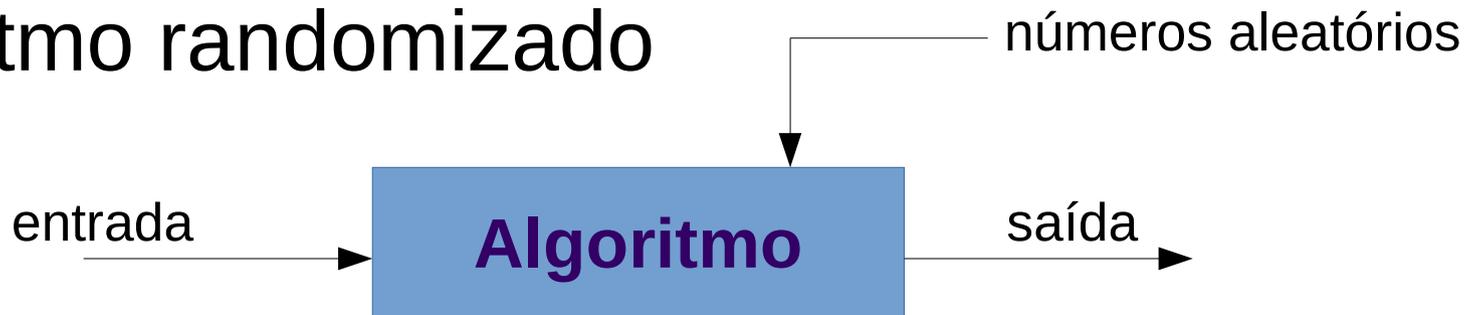
# Algoritmos Randomizados

- Algoritmo clássico (determinístico)



- Mostrar que algoritmo **sempre** gera saída correta em determinado tempo necessário (complexidade)

- Algoritmo randomizado



- Mostrar que algoritmo geralmente gera saída correta em um tempo razoável
  - saída depende também dos números aleatórios

# Exemplo 1

- Buscar por um elemento em um vetor
  - assumir  $V$  possui  $x > 0$  elementos com valor  $s$

`busca_D(V, s)`

Para  $i=1, \dots, |V|$

Se  $V[i] == s$

Retorna  $i$

Retorna  $\emptyset$

- Resposta sempre correta, complexidade  $\Theta(|V|)$

`busca_A(V, s, n)`

Para  $i=1, \dots, n$

$k = \text{random}(1, |V|)$

Se  $V[k] == s$

Retorna  $k$

Retorna  $\emptyset$

- Resposta errada com probabilidade  $(1 - x/|V|)^n$ , complexidade  $\Theta(n)$
- Tempo de execução não depende de  $|V|$

# Exemplo 2

- Testar igualdade entre dois polinômios (nos inteiros)
  - $p(x) = (a_1 - x)(a_2 - x)\dots(a_d - x)$
  - $q(x) = b_d x^d + b_{d-1} x^{d-1} + \dots + b_0 x^0$
- Algoritmo determinístico: expandir  $p(x)$ , comparar constantes com  $q(x)$ , complexidade  $O(d^2)$

igualdade\_A(r, n)

Para  $i=1, \dots, n$

$k = \text{random}(-r, r)$

Se  $p(k) \neq q(k)$

Retorna 0

Retorna 1

- Se  $p \neq q$  então  $p(x) = q(x)$  se  $x$  é raiz de  $z(x) = p(x) - q(x)$
- $z(x)$  possui no máximo  $d$  raízes
- Resposta errada com probabilidade  $(d/2r)^n$ , complexidade  $\Theta(n)$
- Probabilidade diminui com  $r$  e  $n$  (exponencialmente)

# Algoritmo de Monte Carlo

- Algoritmo randomizado com complexidade (tempo de execução) que independe da aleatoriedade
- Gera resposta errada com uma certa probabilidade (em geral,  $< 1/2$ )
- Não podemos determinar se resposta está correta
  - mas podemos executar várias vezes para reduzir probabilidade de erro (em geral, exponencialmente)
- Exemplos anteriores são algoritmos de Monte Carlo
- Algoritmos baseados em gerar amostras e calcular média amostral são algoritmos de Monte Carlo
  - média amostral possui precisão e erro

# Voltando na Busca

- Outro tipo de algoritmo randomizado para o problema de busca em vetor
  - assumir  $V$  possui  $x > 0$  elementos com valor  $s$

```
busca_LV(V, s)
```

```
  Enquanto(1)
```

```
    k = random(1, |V|)
```

```
    Se  $V[k] == s$ 
```

```
      Retorna k
```

- Sempre gera a resposta correta!
- Tempo de execução é aleatório, pior caso é infinito
- Analisar complexidade do caso médio
- Número de tentativas até sucesso  $\sim \text{Geom}(x/|V|)$
- Valor esperado  $\Theta(|V|/x)$

# Quicksort com Aleatoriedade

- Algoritmo recursivo para ordenação de um vetor

$QS(V)$

$k = \text{random}(1, |V|)$

$V1 = \{\text{elementos de } V \text{ menores que } V[k]\}$

$V2 = \{\text{elementos de } V \text{ maiores que } V[k]\}$

$V = QS(V1) \cdot V[k] \cdot QS(V2)$

- Resposta sempre correta (ordenação)
- Tempo de execução é aleatório, pois depende das escolhas aleatórias
  - caso médio  $\Theta(|V| \log |V|)$ , pior caso  $\Theta(|V|^2)$
- Com alta probabilidade, tempo de execução será próximo do caso médio

# Algoritmos de Las Vegas

- Algoritmo randomizado que sempre gera a resposta correta
  - ou pode dizer que a resposta correta não foi encontrada
- Tempo de execução do algoritmo é aleatório
  - pior caso é infinito, mas podemos impor tempo máximo
- Calcular tempo médio de execução, e probabilidade do algoritmo parar próximo ao tempo médio
- Exemplo de busca e Quicksort são algoritmos Las Vegas
- Termo *Las Vegas* definido por László Babai (1979) para diferenciar da ideia de Monte Carlo

# Monte Carlo x Las Vegas

	Tempo de Execução	Corretude
<b>Monte Carlo</b>	Determinado a priori (fixo). Não depende das escolhas aleatórias do algoritmo	Gera resposta errada com alguma probabilidade; análise da probabilidade de erro
<b>Las Vegas</b>	Depende das escolhas aleatórias do algoritmo; análise do tempo médio de execução	Sempre gera resposta correta (ou diz que resposta correta não foi encontrada)

- Teoria da computação (algoritmos) prefere algoritmos de Las Vegas
  - dar resposta correta é mais importante
- Engenharia prefere algoritmos de Monte Carlo
  - erro é tolerável, tempo de execução é importante

# Monte Carlo == Las Vegas ?

- Convertendo entre os tipos de algoritmos
- Algoritmo de Las Vegas → Algoritmo de Monte Carlo
  - fixar limite de tempo de execução, gerar resposta aleatória ao final do limite
  - prob. de erro pode ser calculada a partir da prob. do tempo exceder o limite
- Algoritmo de Monte Carlo → Algoritmo de Las Vegas
  - ao final da execução, outro algoritmo determinístico avalia se resposta está correta
  - em geral, tal algoritmo não está disponível (ou não é eficiente, computacionalmente)
- Las Vegas → Monte Carlo, mas não o contrário

# Algoritmos Randomizados

- Aplicação de Monte Carlo e Las Vegas em diferentes temas e problemas
- Estrutura de dados: árvores de buscas, tabelas *hash*
- Algoritmos em grafos: MST, caminhos mais curto, corte mínimo
- Contagem e enumeração: contagem de estruturas combinatoriais
- Algoritmos distribuídos: consenso, eleição de líder
- Computação quântica: todos os algoritmos são randomizados!

**Importante Área da Computação  
(na teoria e na prática)**