

UFPI - CCN - DC
Ciência da Computação
Estrutura de Dados

Análise de Algoritmos



Prof. Raimundo Moura
rsm@ufpi.edu.br

67

Análise de Algoritmos

- ▶ Computadores são usados para resolver problemas difíceis ou para processar grande quantidade de dados, então:
 - Quanto tempo o programa demorará?
 - Por que o programa fica sem memória?
- ▶ Mais informações:
 - <https://algs4.cs.princeton.edu/14analysis/>

68

Análise de Algoritmos

► **Método Científico:** a mesma abordagem que os cientistas usam para entender o mundo natural é eficaz para estudar o tempo de execução dos programas.

69

Método Científico

1. **Observar** algumas características do mundo natural, geralmente com medições precisas.
2. **Hipotetizar** um modelo que seja consistente com as observações.
3. **Prever** eventos usando a hipótese.
4. **Verificar** as previsões fazendo outras observações.
5. **Validar** repetindo até que a hipótese e as observações estejam de acordo.

70

Método Científico

- ▶ Os experimentos que planejamos devem ser reproduzíveis e as hipóteses que formulamos devem ser verificáveis.
- ▶ DESAFIO: Como definir medidas quantitativas para tempo de execução de programas?
 - Classe StopwatchCPU() mede o tempo de execução total de um programa

71

Temporizador

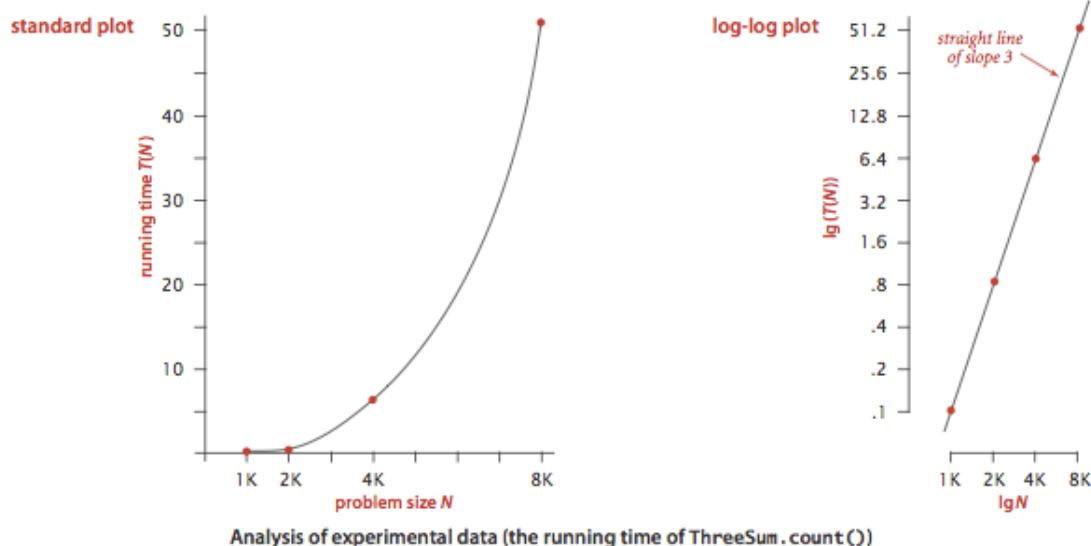
```
public class StopwatchCPU()
```

StopwatchCPU()	Cria um temporizador
double elapsedTime()	Retorna o tempo gasto desde a criação, em segundos

- ▶ Exemplo: aplicação `ThreeSum.java`, que conta o número de triplas em um arquivo de N inteiros que somam ZERO (ignorando *overflow* de inteiros)

72

Análise de Dados Experimentais



73

Modelos Matemáticos

- O tempo de execução total de um programa é determinado por dois fatores principais:
 - Custo de execução de cada instrução
 - Frequência de execução de cada instrução

74

Modelos Matemáticos

- Aproximações til (~): Descarta-se termos de baixa ordem que complicam as fórmulas.
 - ~ $f(N)$ para representar qualquer função que quando dividida por $f(N)$ se aproxima de 1 à medida que N cresce.
 - $g(N) \sim f(N)$ para indicar que $g(N)/f(N)$ se aproxima de 1 à medida que N cresce.

75

Ordem de Crescimento

function	tilde approximation	order of growth
$N^3/6 - N^2/2 + N/3$	$\sim N^3/6$	N^3
$N^2/2 - N/2$	$\sim N^2/2$	N^2
$\lg N + 1$	$\sim \lg N$	$\lg N$
3	~ 3	1

76

Classificações

description	order of growth	typical code framework	description	example
<i>constant</i>	1	<code>a = b + c;</code>	<i>statement</i>	<i>add two numbers</i>
<i>logarithmic</i>	$\log N$	[see page 47]	<i>divide in half</i>	<i>binary search</i>
<i>linear</i>	N	<pre>double max = a[0]; for (int i = 1; i < N; i++) if (a[i] > max) max = a[i];</pre>	<i>loop</i>	<i>find the maximum</i>

77

Classificações

<i>linearithmic</i>	$N \log N$	[see ALGORITHM 2.4]	<i>divide and conquer</i>	<i>mergesort</i>
<i>quadratic</i>	N^2	<pre>for (int i = 0; i < N; i++) for (int j = i+1; j < N; j++) if (a[i] + a[j] == 0) cnt++;</pre>	<i>double loop</i>	<i>check all pairs</i>
<i>cubic</i>	N^3	<pre>for (int i = 0; i < N; i++) for (int j = i+1; j < N; j++) for (int k = j+1; k < N; k++) if (a[i] + a[j] + a[k] == 0) cnt++;</pre>	<i>triple loop</i>	<i>check all triples</i>
<i>exponential</i>	2^N	[see CHAPTER 6]	<i>exhaustive search</i>	<i>check all subsets</i>

78