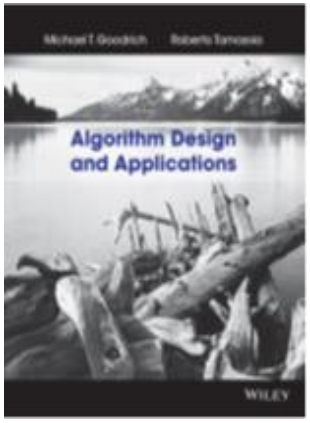
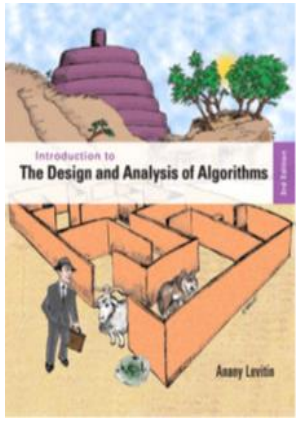
Unidade 2 - Modelo de Knuth

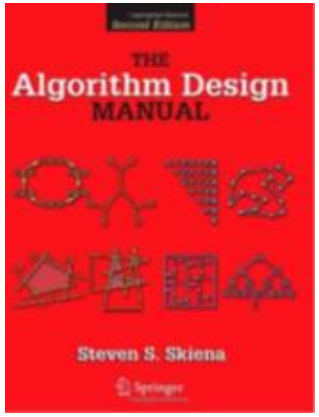
Prof. Aparecido V. de Freitas
Doutor em Engenharia
da Computação pela EPUSP
aparecidovfreitas@gmail.com

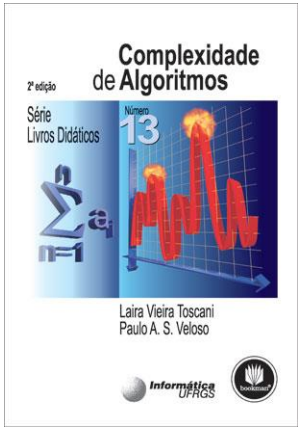
Bibliografia

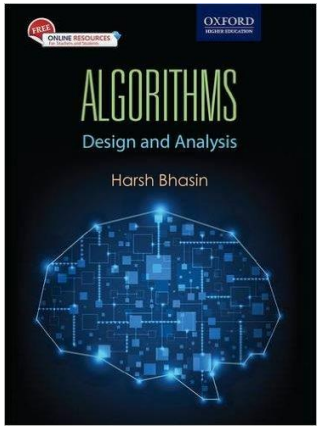
- Algorithm Design and Applications – Michael T. Goodrich, Roberto Tamassia, Wiley, 2015
- Introduction to the Design and Analysis of Algorithms – Anany Levitin, Pearson, 2012
- The Algorithm Design Manual – Steven S. Skiena, Springer, 2008
- Complexidade de Algoritmos – Série Livros Didáticos – UFRGS
- Algorithms – Design and Analysis – Harsh Bhasin – Oxford University Press - 2015

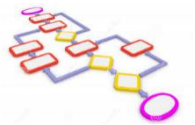










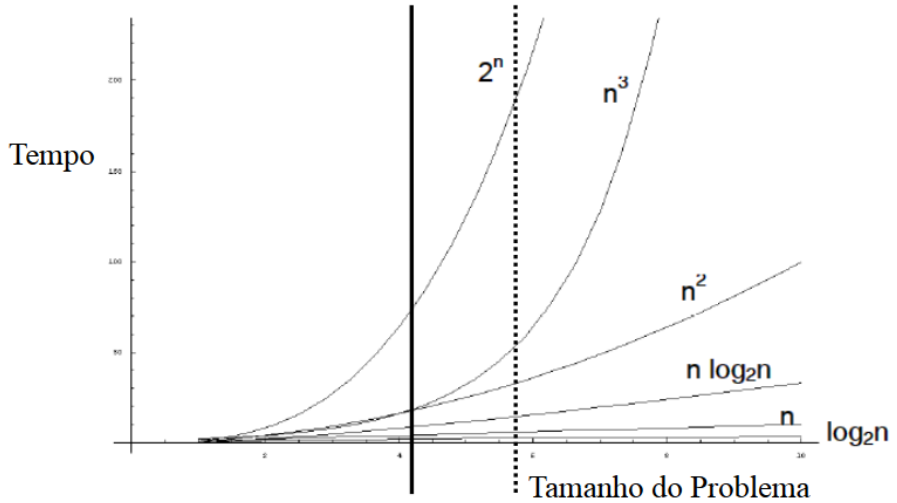


Aula 2

- ✓ Introdução à **Análise** de **Algoritmos**
- ✓ Modelo de **Knuth** - Análise de Algoritmos Iterativos

Método Analítico

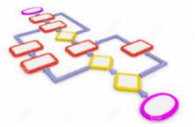
- Embora o método experimental tem um importante papel em análise de algoritmos, quando tratado de forma isolada **não** é suficiente;
- É necessário um método analítico que:
  - ✓ Considere **todas** as entradas possíveis;
  - ✓ Seja **independente** de ambientes de hardware e software;
  - ✓ Seja obtido **sem** a execução do algoritmo.



Modelo de Knuth

- Modelo matemático, desenvolvido por **Donald Knuth**, Stanford University, 1968;
- Baseia-se na **contabilização** do conjunto de **operações** executadas pelo algoritmo;
- Associa-se um custo à cada operação executada;
- Em geral, ignora-se o custo de algumas operações e se contabiliza apenas as operações mais significativas;
- As operações mais significativas são denominadas **operações básicas**;

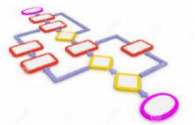
Stanford University



Modelo Detalhado

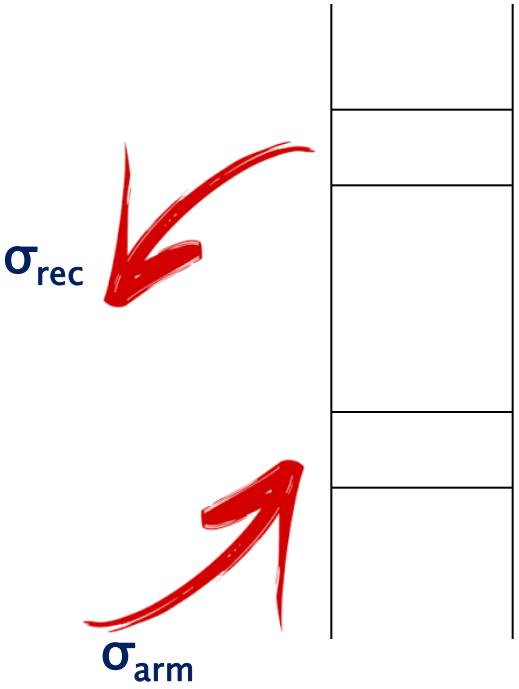
- O tempo de execução de um algoritmo será calculado pela somatória do tempo necessário para a execução das operações básicas;

- O tempo de processamento das operações básicas é definido por um conjunto de **axiomas**.



Axioma 1

- Os tempos requeridos para recuperar um operando da memória e para armazenar o resultado na memória são constantes: σ_{rec} , σ_{arm} , respectivamente.



Exemplo - Axioma 1

Qual o tempo de processamento da atribuição?

y = x ;

- ✓ Será necessário recuperar-se em memória o conteúdo da variável x. Este tempo é σ_{rec} .
- ✓ O tempo requerido para se armazenar o valor na variável y é σ_{arm} .

Resposta: $\sigma_{rec} + \sigma_{arm}$

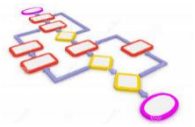
Outro Exemplo - Axioma 1

Qual o tempo de processamento do comando ?

y = 1 ;

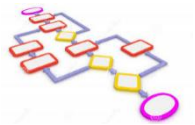
- ✓ A constante 1 (chamada de literal numérico) também precisa ser armazenada na memória (tabela de literais gerada pelo compilador).
- ✓ Assim, o custo para se recuperar em memória a constante 1 também será σ_{rec} .
- ✓ O tempo requerido para se armazenar o valor na variável y é σ_{arm}.

Resposta: $\sigma_{rec} + \sigma_{arm}$



Axioma 2

- ✓ Os tempos necessários para se realizar operações aritméticas elementares, tais como: adição, subtração, multiplicação, divisão e comparação são todos constantes.
- ✓ Estes tempos são denotados por: σ_{+}, σ_{-}, σ_{x}, $\sigma_{/}$, $\sigma_{\leq}$, respectivamente.



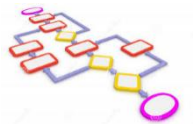
Exemplo – Axioma 2

Qual o tempo de processamento da atribuição ?

y = y + 1 ;

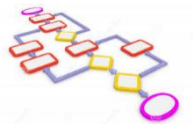
- ✓ Neste caso, temos a necessidade de recuperar dois valores em memória: y e 1.
- ✓ Assim, o tempo para se recuperar estes valores será dado por: $2\sigma_{rec}$.
- ✓ O tempo para se efetuar a soma é σ_{+}.
- ✓ O tempo requerido para se armazenar o resultado na variável y é σ_{arm}.

Resposta: $2\sigma_{rec} + \sigma_{+} + \sigma_{arm}$



Axioma 3

- O tempo necessário para se chamar um método é constante: $\sigma_{chamada}$ e o tempo necessário para se retornar de um método é constante: $\sigma_{retorno}$.
- Quando um método é chamado, algumas operações de bastidores são necessárias: save de endereços de retorno, chaveamento de contexto, etc.
- Estas operações são desfeitas no momento de retorno.



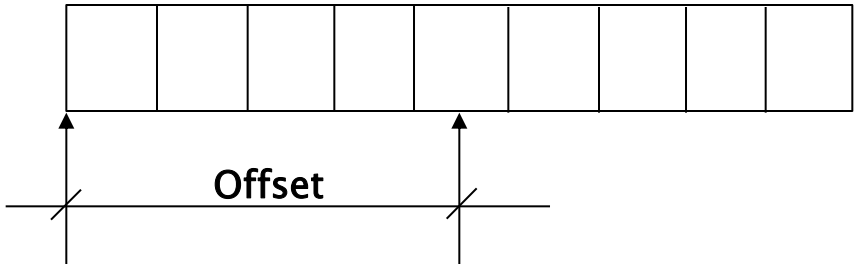
Axioma 4

- O tempo necessário para se passar um parâmetro a um método é o mesmo tempo para se armazenar um valor em memória: σ_{arm}.

- Conceitualmente, o esforço computacional necessário para o tratamento da passagem de parâmetros é o mesmo que se atribuir ao parâmetro formal do método o valor do argumento.

Operações com índices de Arrays

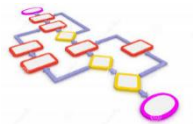
- Em geral, os elementos de um array são armazenados em locais contíguos de memória.
- Assim, dado o endereço do primeiro elemento do array, uma simples operação de adição é suficiente para se determinar o endereço de um elemento arbitrário do array.



Axioma 5

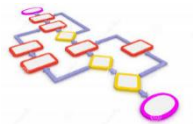
- O tempo requerido para o cálculo do endereço advindo de uma operação de índice de um array, por exemplo, a[i], é constante: $\sigma_.$;
- Esse tempo não inclui o tempo para calcular a expressão do índice, nem inclui o tempo de acesso (ou seja, o tempo de recuperação ou armazenamento) ao elemento do array;
- Exemplo: **y = a[i] =>** Tempo: $3\sigma_{rec} + \sigma_. + \sigma_{arm}$
- Serão necessárias três recuperações: a primeira para recuperar a (o endereço base do array), a segunda para recuperar i e a terceira para recuperar o elemento a[i].

Modelo Simplificado de KNUTH



Modelo Simplificado

- O modelo detalhado fornece uma boa previsão do desempenho de execução de um algoritmo;
- No entanto, tal modelo é oneroso e trabalhoso;
- No modelo simplificado, eliminamos a dependência de tempo, considerando-se um tempo constante e igual ao ciclo do processador (**T=1**);
- Assim, neste modelo contabiliza-se apenas a quantidade de operações efetuadas pelo algoritmo.



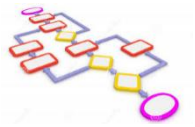
Modelo Simplificado - Axioma 1

- O total de ciclos de processador requeridos para recuperar um operando da memória e para armazenar o resultado na memória são constantes (1 ciclo = 1 operação para recuperar e 1 ciclo = 1 operação para armazenar)

Exemplo: **Qual o total de operações para executar a atribuição?**

$$y = x ; \quad (\sigma_{rec} + \sigma_{arm})$$

Resposta: 1 + 1 = 2 operações



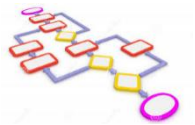
Modelo Simplificado - Axioma 2

- As operações necessárias para se realizar operações aritméticas elementares, tais como: adição, subtração, multiplicação, divisão e comparação são todas constantes e iguais a 1 ciclo cada (1 operação).

Exemplo: Qual o total de operações para processar a atribuição?

$$y = y + 1 \; ; \quad (2\sigma_{rec} + \sigma_{+} + \sigma_{arm})$$

Resposta: 4 operações



Modelo Simplificado – Axioma 3

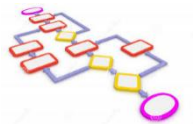
- Gasta-se 1 ciclo de processador (1 operação) para se chamar um método e 1 ciclo (1 operação) para se providenciar o retorno do método.

Resposta: 2 operações

Modelo Simplificado – Axioma 4

- Gasta-se 1 ciclo de processador (1 operação) para se passar um parâmetro a um método.

Resposta: 1 operação



Modelo Simplificado – Axioma 5

- y = a[i] => Tempo: $3\sigma_{rec} + \sigma_{.} + \sigma_{arm}$
- Serão necessárias três recuperações: a primeira para recuperar **a** (o endereço base do array), a segunda para recuperar **i** e a terceira para recuperar o elemento a[i] ;
- Teremos, portanto um total de 5 operações básicas.