Análise de Algoritmos

Prof. Sérgio Carlos Portari Júnior

2016

Plano de Ensino

- **EMENTA:** Desenvolvimento de Algoritmos. Medidas de Complexidade. Estratégias Básicas. Divisão e Conquista. Método Guloso. Programação Dinâmica. Método de Recuo, Ramificação e Poda. Outros Tópicos Relacionados.

Bibliografia

- KNUTH, D. E. The Art of Computer Programming. Massachusetts: Addison-Wesley Longman, 1997. v. 1 e 2.
- SALVETTI, D. D. & BARBOSA L M. Algoritmos. São Paulo: Makron Books, 1998.
- TERADA, R. Desenvolvimento de Algoritmo e Estruturas de Dados. São Paulo: Makron Books, 1991.
- ZIVIANI, N. Projeto de Algoritmos - Com Implementações em PASCAL e C. São Paulo. Editora Pioneira, 1999.

Objetivos

- Capacitar o aluno a realizar métricas em diferentes algoritmos que resolvam um mesmo problema, permitindo assim escolher o tipo de algoritmo adequado a cada situação, com melhor custo x benefício de utilização de tempo e processamento.

Método de Ensino

- Aulas expositivas;
- Aulas Práticas em Laboratório;
- Atividades individuais e/ou grupo.

Avaliação

- Trabalhos individuais ou em grupos, intra ou extra sala, totalizando até 30% do total semestral de pontos;
- Aplicação de 2 avaliações parciais, totalizando 40% do total semestral de pontos;
- Avaliação Semestral, totalizando 30% do total semestral de pontos.

Informações e Contatos

E-mail: **portari.uemgituiutaba@gmail.com**

Utilizem este e-mail para evitar outros endereços com SPAM que não acesso mais;

Página: **www.sergioportari.com.br**

Na página você encontrará um link para informações da disciplina, datas importantes como entrega de trabalhos e provas, além de download dos materiais apresentados em sala.

Dúvidas

- ??????????????????????????????????????
  ??????????????????????????????????????
  ??????????????????????????????????????
  ??????????????????????????????????????
  ??????????????????????????????????????
  ??????????????????????????????????????
  ??????????????????????????????????????
  ??????????????????????????????????????
  ??????????????????????????????????????

Desenvolvimento de Algoritmos

Introdução

Introdução

-
- Dado um problema, como encontramos um algoritmo eficiente para sua solução?
- Encontrado um algoritmo, como comparar este algoritmo com outros algoritmos que solucionam o mesmo problema?
- Como deveríamos julgar a qualidade de um algoritmo?
- Qual é o algoritmo de menor custo possível para resolver um problema particular?

Introdução

- Questões desta natureza são de interesse para programadores e cientistas da computação.
- Algoritmos podem ser avaliados por uma variedade de critérios.
- Na maioria das vezes estamos interessados na taxa de crescimento do tempo ou de espaço necessário para a solução de grandes problemas.

O que é um Algoritmo?

- Um algoritmo pode ser visto como uma sequência de ações executáveis para a obtenção de uma solução para um determinado tipo de problema.
- Segundo **Dijkstra**, um algoritmo corresponde a uma descrição de um padrão de comportamento, expresso em termos de um conjunto finito de ações.
- Segundo **Terada**, um algoritmo é, em geral, uma descrição passo a passo de como um problema é solucionável. A descrição deve ser finita, e os passos devem ser bem definidos,

Medidas de Complexidade

- Como selecionar um algoritmo quando existem vários que solucionam o problema?
- Uma resposta pode ser, escolher um algoritmo de fácil entendimento, codificação e depuração ou então uma outra resposta pode ser, um algoritmo que faz uso eficiente dos recursos do computador.
- Qual a melhor solução?
- Como escolher?

Medidas de Complexidade

- Vários critérios podem ser utilizados para escolher o algoritmo, mas vai depender das pretensões de utilização do algoritmo.
- Pode-se estar selecionando o algoritmo somente para um experimento, ou será um programa de grande utilização, ou será utilizado poucas vezes e será descartado, ou ainda, terá aplicações futuras que podem demandar alterações no código.
- Para cada uma das respostas anteriores, pode-se pensar em uma solução diferente.
- Calcular o tempo de execução e o espaço

Medidas de Complexidade

- Para o cálculo de complexidade, pode-se medir o número de passos de execução em um modelo matemático denominado maquina de Turing, ou medir o número de segundos gastos em um computador específico.
- Ao invés de calcular os tempos de execução em máquinas específicas, a maioria das análises conta apenas o número de operações “elementares” executadas.
- A medida de complexidade é o mais próximo do assintótico* de

Que tal escolher um valor bem próximo do número de contagens de operações

Medidas de Complexidade

- O menor custo possível para resolver problemas de uma classe nos dá a dificuldade inerente para resolver o problema.
- Quando o custo de um algoritmo é igual ao menor custo possível, o algoritmo é **ótimo** para a medida de custo considerada.
- Podem existir vários algoritmos ótimos para resolver o mesmo Problema.
- Se a mesma medida de custo é aplicada a diferentes algoritmos, então é possível compará-los e escolher o mais adequado.

Medidas de Complexidade

- Utilizaremos um modelo matemático baseado em um computador idealizado.
- Deve ser especificado o conjunto de operações e seus custos de execuções.
- É mais usual ignorar o custo de algumas operações e considerar apenas as mais significativas.
- Ex.: Em algoritmos de ordenação consideramos o número de comparações entre os elementos do conjunto a ser ordenado e ignoramos as demais operações.

Medidas de Complexidade

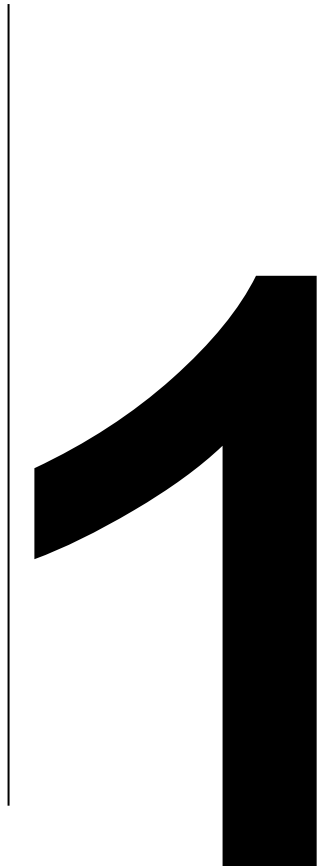
- Para medir o custo de execução de um algoritmo definiremos uma função de complexidade ou função de custo **f** .
- Função de complexidade de tempo: **f (n)** mede o tempo necessário para executar um algoritmo em um problema de tamanho **n**.
- Função de complexidade de espaço: **f (n)** mede a memória necessária para executar um algoritmo em um problema de tamanho **n**.

Medidas de Complexidade

- Utilizaremos **f** para denotar uma função de complexidade de tempo daqui para frente.
-
- A complexidadede tempo na realidade não representa tempo diretamente:
  - Representa o número de vezes que determinadas operações, ditas relevantes, são executadas.

Medidas de Complexidade

- Considere o algoritmo ao lado para encontrar o maior número dentro de um vetor de inteiros A[n]; onde n>1:



Medidas de Complexidade

- Seja **f** uma função de complexidade tal que **f(n)** é o número de comparações envolvendo os elementos de A, se A contiver n elementos.
- Qual é a função **f(n)**?
-
-

- **f(n) = n – 1**

Medidas de Complexidade

- **Teorema**: Qualquer algoritmo para encontrar o maior elemento de um conjunto com n elementos, n >= 1, faz pelo menos n − 1 comparações.
- **Prova**: Cada um dos n − 1 elementos tem de ser investigado por meio de comparações, que é menor do que algum outro elemento.
- **Logo**, n − 1 comparações são necessárias.

Medidas de Complexidade

- O teorema nos diz que, se o número de comparações for utilizado como medida de custo, então a função **Max** do programa anterior é ótima.
- A medida do custo de execução de um algoritmo depende principalmente do tamanho da entrada dos dados.

Medidas de Complexidade

- É comum considerar o tempo de execução de um programa como uma função do tamanho da entrada.
- Para alguns algoritmos, o custo de execução é uma função da entrada particular dos dados, não apenas do tamanho da entrada.
  - Ou seja, o custo pode ser diferente para entradas distintas, mas de mesmo tamanho.

Medidas de Complexidade

- No caso da função **Max** do programa do exemplo, o custo é uniforme sobre todos os problemas de tamanho **n**.
-

Exercício

- Avalie os dois códigos fonte abaixo e responda:
- O resultado será o mesmo? Justifique sua resposta.
- Qual a função de complexidade de cada um dos procedimentos? Defina as operações relevantes.
- Caso o resultado seja o mesmo, qual dos dois você escolheria?

```
1 void Procedimento1() {
2   int i = 0, a = 0;
3   while(i < n) {
4     a += i;
5     i += 2;
6   }
7 }
```

```
1 void Procedimento2() {
2   int i, j, a = 0;
3   for(i = 0; i < n; i++)
4     for(j = 0; j < i; j++)
5       a += i + j;
6 }
```

- Avalie o córigo, faça o teste de mesa, depois responda as perguntas. Em sequida, implemente os códigos acime, execute-os e confira seus resultados