Análise de Algoritmos

Introdução

Profa. Sheila Morais de Almeida

DAINF-UTFPR-PG

abril - 2018

Modelo Computacional

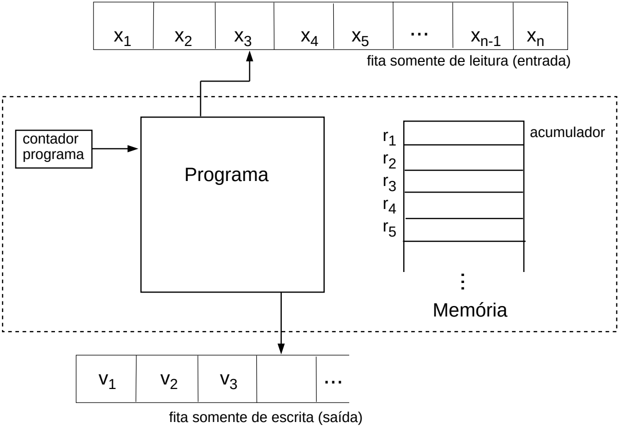
A análise de um algoritmo depende do modelo computacional adotado.

É de acordo com o modelo computacional que se define quais são os recursos disponíveis e quanto custam.

Modelo adotado: RAM - *Random Access Machine*

- Simula máquinas convencionais.
- Um único processador que executa instruções sequencialmente.
- Operações aritméticas básicas (somas, subtrações, multiplicações e divisões), atribuições e comprações são feitas em tempo constante.

Modelo Computacional RAM



Fonte: **A. Aho; J. Hopcroft; J. Ullman**. *The Design and Analysis of Computer Algorithms*, 1974

Modelo Computacional

No modelo RAM o programa não é armazenado na memória, então o espaço que ele ocupa não é contabilizado no gasto de memória.

Instruções existentes em computadores reais podem ser incorporadas ao modelo RAM sem alterar a ordem de grandeza da complexidade dos problemas[1].

[1] **A. Aho; J. Hopcroft; J. Ullman**. *The Design and Analysis of Computer Algorithms*, 1974

Análise de Algoritmos

Para fazer uma previsão do tempo necessário para a execução de um algoritmo em determinado modelo computacional, considerando uma determinada instância do problema, devemos:

- Identificar o conjunto Q das instruções computacionais básicas do modelo.
- Determinar $t(q)$, o tempo necessário para a execução de cada instrução $q \in Q$.
- Determinar $f(q)$, a quantidade de vezes que cada instrução $q \in Q$ é executada.
- Calcular o tempo total de execução:

$$\sum_{\forall q \in Q} f(q) \cdot t(q)$$

Análise de Algoritmos

Vamos analisar o Algoritmo Ordenação por Inserção.

Quantas operações de comparação são executadas?

Algoritmo Ordenação por Inserção
Entrada: vetor $v[1..n]$ de inteiros; n número de elementos no vetor
    Para i de 2 a n **faça:**
        $j \leftarrow i - 1$
        Enquanto $j > 0$ && $v[i] < v[j]$ **faça:**
            $j \leftarrow j - 1$
        $t \leftarrow v[i]$
        Para k de $i - 1$ a $j + 1$ **faça:**
            $v[k+1] \leftarrow v[k]$
        $v[j+1] \leftarrow t$

Análise de Algoritmos

Vamos analisar o Algoritmo Ordenação por Inserção.

Quantas operações de comparação são executadas?
Depende da instância.

Algoritmo Ordenação por Inserção
Entrada: vetor $v[1..n]$ de inteiros; n número de elementos no vetor

  Para i de 2 a n **faça:**
    $j \leftarrow i - 1$
    Enquanto $j > 0$ && $v[i] < v[j]$ **faça:**
      $j \leftarrow j - 1$
    $t \leftarrow v[i]$
    Para k de $i - 1$ a $j + 1$ **faça:**
      $v[k+1] \leftarrow v[k]$
    $v[j+1] \leftarrow t$

Análise de Algoritmos

Podemos considerar todas as instâncias possíveis!

Seria uma tarefa bastante árdua! Quantas entradas possíveis existem?

Vamos usar uma medida da entrada, que chamamos de **tamanho da entrada**.

- A análise dos algoritmos será em função do tamanho da entrada.
- O tamanho da entrada é normalmente denotado por n.
- Em geral, o tamanho da entrada é uma medida da quantidade de memória necessária para armazenar a entrada.

Análise de Algoritmos

Tamanho da entrada

- Deve-se considerar as unidades básicas das estruturas de dados que serão manipuladas pelo algoritmo:
    - em uma ordenação em um vetor, o número de elementos;
    - no cálculo com números muito grandes (para astronomia, por exemplo), a quantidade de dígitos dos números;
    - em um grafo, a quantidade de vértices e arestas.

Análise de Algoritmos

O algoritmo pode não se comportar exatamente da mesma forma para todas as instâncias de tamanho n.

Nós teremos que escolher, dentre todas as instâncias possíveis, qual queremos usar como indicativo do tempo de execução do algoritmo.

Análise de Algoritmos

Escolha das instâncias para análise

Normalmente, considera-se:

- a entrada do melhor caso, aquela que faz com que o algoritmo execute o menor número de instruções para dar a resposta;
- a entrada de caso médio, aquela que exige um tempo de execução que é aproximadamente a média dos tempos de execução de todas as possíveis entradas; ou
- a entrada de pior caso, aquela que faz o algoritmo demorar mais (executar mais instruções) para dar a resposta.

Análise de Algoritmos

Sobre a entrada de melhor caso

A análise com entrada de melhor caso geralmente não representa o que ocorre com o tempo de execução de um algoritmo na maior parte dos casos.

A maioria dos problemas tem uma instância para a qual a sua solução é trivial.

Análise de Algoritmos

Sobre a entrada de caso médio

Existem alguns casos famosos em que a análise realizada com entradas de caso médio apresenta resultados bem melhores que a análise com entradas de pior caso.

Alguns problemas são intratáveis no pior caso, mas as entradas que explicitam esse comportamento podem raramente ocorrer na prática.

- A complexidade de caso médio pode ser uma medida mais precisa da performance desses algoritmos.

Análise de Algoritmos

Sobre a entrada de caso médio

Poderia ser uma boa escolha! Mas...

O que é uma entrada do caso médio? Nem sempre é claro o que é uma entrada de caso médio.

Que parâmetros serão usados para se tirar a média?

Se não tomarmos cuidado, podemos considerar como entrada de caso médio um tipo de entrada que nunca ocorre na prática.

Análise de Algoritmos

Sobre a entrada de caso médio

É difícil avaliar o desempenho de entradas de caso médio. Geralmente, exigem maior habilidade matemática.

Nós veremos alguns exemplos de análise com entradas de caso médio, mas vamos nos concentrar em análises com entradas de pior caso.

Análise de Algoritmos

Sobre a entrada de pior caso

A análise com entrada de pior caso fornece um **limite superior** para o tempo de execução do algoritmo. Conhecer esse limite é ter uma garantia de que o algoritmo não vai demorar mais do que essa medida.

Em alguns casos, o **pior caso ocorre com frequência**: na busca em um banco de dados, o pior caso é quando o dado não está no banco.

Análise de Algoritmos

Sobre a entrada de pior caso

Frequentemente, os resultados da análise com entradas de pior caso **são muito próximos dos obtidos com entradas de caso médio** ou através de observações experimentais.

Mesmo quando a análise com entradas de pior caso tem resultados diferentes da análise com entradas de caso médio, **o algoritmo que tem o melhor desempenho no pior caso também tem desempenho muito bom com as demais instâncias**.

Análise de Algoritmos

Nós chamamos de

- Análise de pior caso, aquela que considera as instâncias de pior caso;
- Análise de caso médio, aquela que considera as instâncias de caso médio;
- Análise de melhor caso, aquela que considera as instâncias de melhor caso.

Análise de Algoritmos

Para o Algoritmo de Ordenação por Inserção, apresente uma instância

- de melhor caso;
- de pior caso;
- e de caso médio.

Análise de Algoritmos

Algoritmo Ordenação por Inserção
Entrada: vetor $v[1..n]$ de inteiros; n número de elementos no vetor

Para i de 2 a n **faça:**
    $j \leftarrow i - 1$
    Enquanto $j > 0$ && $v[i] < v[j]$ **faça:**
        $j \leftarrow j - 1$
    $t \leftarrow v[i]$
    Para k de $i - 1$ a $j + 1$ **faça:**
        $v[k + 1] \leftarrow v[k]$
    $v[j + 1] \leftarrow t$

Análise de Algoritmos

Para o Algoritmo de Ordenação por Inserção, apresente uma instância

- de melhor caso: vetor em ordem crescente.
- de pior caso: vetor em ordem decrescente.
- e de caso médio: quando $v[i]$ é considerado, cada número de $v[0]$ a $v[i-1]$ tem 50% de chance de ser maior que $v[i]$ e 50% de chance de ser menor que $v[i]$. Para um exemplo, podemos supor:
  - os números em posições de 0 a $\lfloor \frac{i}{2} \rfloor$ são menores que $v[i]$ e
  - os números em posições de $\lfloor \frac{i}{2} \rfloor + 1$ a $i - 1$ são maiores que $v[i]$.

Análise de Algoritmos

Para o Algoritmo de Ordenação por Inserção, apresente uma instância

- de melhor caso: vetor em ordem crescente.
- de pior caso: vetor em ordem decrescente.
- e de caso médio: quando $v[i]$ é considerado, cada número de $v[0]$ a $v[i-1]$ tem 50% de chance de ser maior que $v[i]$ e 50% de chance de ser menor que $v[i]$. Para um exemplo, podemos supor:
  - os números em posições de 0 a $\lfloor \frac{i}{2} \rfloor$ são menores que $v[i]$ e
  - os números em posições de $\lfloor \frac{i}{2} \rfloor + 1$ a $i-1$ são maiores que $v[i]$.

Ex.: 90 12 18 83 24 71 67 35 58 40.

Análise de complexidade de tempo no pior caso

Considere o Algoritmo de Ordenação por Inserção:

Algoritmo Ordenação por Inserção
Entrada: vetor $v[1..n]$ de inteiros; n número de elementos no vetor
  Para i de 2 a n **faça:**
    $j \leftarrow i - 1$
    Enquanto $j > 0 \ \&\& \ v[i] < v[j]$ **faça:**
      $j \leftarrow j - 1$
    $t \leftarrow v[i]$
    Para k de $i - 1$ a $j + 1$ **faça:**
      $v[k + 1] \leftarrow v[k]$
    $v[j + 1] \leftarrow t$

Análise de complexidade de tempo no pior caso

Pergunta: Quantas instruções básicas do modelo computacional RAM (operações aritméticas básicas, atribuições e comparações) são executadas pelo Algoritmo de Ordenação por Inserção, considerando uma entrada de tamanho n de pior caso?

Análise de complexidade de tempo no pior caso

Considere o Algoritmo de Ordenação por Inserção:

Algoritmo Ordenação por Inserção
Entrada: vetor $v[1..n]$ de inteiros; n número de elementos no vetor
  Para i de 2 a n **faça:** $2 + 3(n-1) = 3n - 1$
    $j \leftarrow i - 1$ $2(n-1) = 2n - 2$
    Enquanto $j > 0$ && $v[i] < v[j]$ **faça:** $2\sum_{c=1}^{n-1} c + \ (n-1)^*$
      $j \leftarrow j - 1$ $2\sum_{c=1}^{n-1} c = n^2 - n$
    $t \leftarrow v[i]$ $n - 1$
    Para k de $i-1$ a $j+1$ **faça:** $4\frac{n(n-1)}{2} + 4(n-1)^\dagger$
      $v[k+1] \leftarrow v[k]$ $n(n-1) = n^2 - n$
    $v[j+1] \leftarrow t$ $2(n-1) = 2n - 2$

$^*2\sum_{c=1}^{n-1} c + \ (n-1) = n(n-1) + (n-1) = (n+1)(n-1) = n^2 - 1$

$^\dagger 4\frac{n(n-1)}{2} + 4(n-1) = 2n^2 + 2n - 4$

Análise de complexidade de tempo no pior caso

Considere o Algoritmo de Ordenação por Inserção:

Algoritmo Ordenação por Inserção
Entrada: vetor $v[1..n]$ de inteiros; n número de elementos no vetor

  Para i de 2 a n **faça:** $2 + 3(n-1) = 3n - 1$
    $j \leftarrow i - 1$ $2(n-1) = 2n - 2$
    Enquanto $j > 0 \;\&\&\; v[i] < v[j]$ **faça:** $n^2 - 1$
      $j \leftarrow j - 1$ $2\sum_{c=1}^{n-1} c = n^2 - n$
    $t \leftarrow v[i]$ $n - 1$
    Para k de $i - 1$ a $j + 1$ **faça:** $2n^2 + 2n - 4$
      $v[k+1] \leftarrow v[k]$ $n(n-1) = n^2 - n$
    $v[j+1] \leftarrow t$ $2(n-1) = 2n - 2$

Total: $5n^2 + 8n - 11$