MC102 – Algoritmos e Programação de Computadores

Instituto de Computação

UNICAMP

Primeiro Semestre de 2013

Roteiro

Fundamentos de análise de algoritmos

Como analisar um algoritmo:

- contar o número de operações realizadas pelo algoritmo para uma dada entrada.
- expressar este número em função do tamanho da entrada.

Exemplos:

$f_1(n) = 5n$
$f_2(n) = 2n^2$
$f_3(n) = n \log n$

Fundamentos de análise de algoritmos

A análise é sempre realizada em relação a um dado modelo computacional. No nosso caso, vamos considerar um modelo simplificado:

- o computador tem um único processador.
- todos os acessos à memória têm o mesmo custo.
- instruções são executadas sequencialmente.
- não há instruções nem operações paralelas.
- todas as instruções têm custo similar (uma unidade).

Geralmente estamos interessados em medir o uso dos recursos computacionais (tempo de processamento, uso de memória, largura de banda, etc). Nesta aula vamos considerar como custo computacional apenas o tempo de processamento de um programa.

Fundamentos de análise de algoritmos

Para exemplificar a diferença entre funções de custo, suponha:

- um computador com 1GHz.
- uma instrução executada a cada ciclo da máquina (1GHz $= 10^9$ instruções por segundo).

- algoritmo com custo $f(n) = n$ e entrada de tamanho $n = 1000000$:
  o computador gastará um 1 milissegundo.

- algoritmo com custo $f(n) = 100n$ e entrada de tamanho $n = 1000000$:
  o computador gastará um décimo de segundo.

- algoritmo com $f(n) = n^2$ e entrada de tamanho $n = 1000000$:
  o computador gastará aproximadamente 17 minutos.

- algoritmo com $f(n) = n^3$ e entrada de tamanho $n = 1000000$:
  o computador gastará aproximadamente 32 anos.

Cálculo da função de custo

Exemplos:

1. custo: operação de atribuição

```
void troca(int *x, int *y) {
  int aux;

  aux = *x;
  *x = *y;
  *y = aux;
}
```

$f(n) = 3$

Cálculo da função de custo

2. custo: inicialização de um vetor

```
for (i = 0; i < n; i++)
  v[i] = 0;
```

- comando `i = 0`: $f(n) = 1$
- comando `i < n`: $f(n) = n + 1$
- comando `i++`: $f(n) = n$
- comando `v[i] = 0`: $f(n) = n$

- total: $f(n) = 3n + 2$

Cálculo da função de custo

3. custo: número de multiplicações

```
int fatorial(int n) {
  int fat = 1;

  for (i = 2; i <= n; i++)
    fat = fat * i;
  return fat;
}
```

$$f(n) = \sum_{i=2}^{n} 1 = n - 1$$

Cálculo da função de custo

4. custo: soma de matrizes quadradas (operação relevante: adição)

```
for (i = 0; i < n; i++)
  for (j = 0; j < n; j++)
    c[i][j] = a[i][j] + b[i][j];
```

$$f(n) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} 1 = \sum_{i=0}^{n-1} n = n^2$$

Cálculo da função de custo

5. custo: produto de matrizes (operação relevante: multiplicação)

```
for (i = 0; i < n; i++)
  for (j = 0; j < n; j++) {
    c[i][j] = 0;
    for (k = 0; k < n; k++)
      c[i][j] = c[i][j] + a[i][k]*b[k][j];
  }
```

$$f(n) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} 1 = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} n = \sum_{i=0}^{n-1} n^2 = n^3$$

Cálculo da função de custo

6. custo: inversão da ordem dos elementos de um vetor

```
for (i = 0; i < n/2; i++)
  troca(a[i], a[n-i-1]);
```

$$f(n) = \sum_{i=0}^{\lfloor n/2 \rfloor - 1} 1 = \lfloor n/2 \rfloor$$

Cálculo da função de custo

7. custo: comando `soma++` nas funções a seguir:

```
soma = 0;
for (i = 0; i < n; i++)
  soma++;
```

$$f(n) = \sum_{i=0}^{n-1} 1 = n$$

```
soma = 0;
for (i = 0; i < n; i++)
  for (j = 0; j < n; j++)
    soma++;
```

$$f(n) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} 1 = \sum_{i=0}^{n-1} n = n^2$$

Cálculo da função de custo

```
soma = 0;
for (i = 0; i < n; i++)
  for (j = 0; j < n*n; j++)
    soma++;
```

$$f(n) = \sum_{i=0}^{n-1} \sum_{j=0}^{n^2-1} 1 = \sum_{i=0}^{n-1} n^2 = n^3$$

```
soma = 0;
for (i = 0; i < n; i++)
  for (j = 0; j < i; j++)
    soma++;
```

$$f(n) = \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1 = \sum_{i=0}^{n-1} i = \frac{n(n-1)}{2}$$

Cálculo da função de custo

```
soma = 0;
k = 1;
for (i = 1; i <= n; i++) {
  for (j = 1; j <= k; j++)
    soma++;
  k = k * 2;
}
```

$$f(n) = \sum_{i=1}^{n} \sum_{j=1}^{2^{(i-1)}} 1 = \sum_{i=1}^{n} 2^{(i-1)} = \sum_{i=0}^{n-1} 2^{i} = 2^{n} - 1$$

Cálculo da função de custo

```
soma = 0;
for (i = 1; i <= n; i++) {
  for (j = 1; j <= n/i; j++)
    soma++;
}
```

$$f(n) = \sum_{i=1}^{n} \sum_{j=1}^{\lfloor n/i \rfloor} 1 = \sum_{i=1}^{n} \lfloor n/i \rfloor \leq \sum_{i=1}^{n} n/i = n \sum_{i=1}^{n} 1/i \leq n(\ln n + 1) = n \ln n + n$$