Projeto e Análise de Algoritmos
2024.2

Análise Amortizada

Prof. Marcio Costa Santos DCC/UFMG

Contador Binário

- Assume que temos um vetor de *n* bits.
- Este vetor representa um número.
- Vamos criar uma função para incrementar uma unidade no número representado.

Contador Binário

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000

Algoritmo Incrementa Bit

Entrada: vetor de bits A e tamanho n.

$i \leftarrow 0$;

while $i \leq n$ *e* $A[i] = 1$ **do**

  $A[i] \leftarrow 0$;

  $i \leftarrow i + 1$;

end

if $i \leq n$ **then**

  $A[i] \leftarrow 0$;

end

Algoritmo 1: IncrementaBit()

Contador Binário - Complexidade

- Complexidade de Pior caso: $O(n)$.
- Mas esse pior caso acontece muito raramente...
- As operações tem uma relação clara entre elas.
- Seria interessante ter uma ligação entre a complexidade e as operações.

Contador Binário - Análise Amortizada

- Considere o número de operações para se realizar uma sequencia de n operações : $T(n)$.
- Desejamos calcular

$$\frac{T(n)}{n}$$

- Complexidade Amortizada.

Contador Binário

0	0000	0
1	0001	1
2	0010	3
3	0011	4
4	0100	7
5	0101	8
6	0110	10
7	0111	11
8	1000	15

Contador Binário - Análise Amortizada

- Vamos realizar n operações de incremento.
- Calcular o custo de cada chamada é complicado.
- Vamos pensar em quantas vezes cada bit é trocado de 0 para 1 ou vice e versa.
- Seja $F(i)$ o número de vezes que o bit na posição i é flipado.

Contador Binário - F(0)

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000

Contador Binário - F(1)

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000

Contador Binário - Análise Amortizada

- $F(0) = 8$.
- $F(1) = 4$.
- $F(i) = \lfloor \frac{n}{2^i} \rfloor$.
- $$\sum_{i=0}^{n-1} F(i) = \sum_{i=0}^{n-1} \lfloor \frac{n}{2^i} \rfloor$$

  .

Contador Binário - Análise Amortizada

- $\sum_{i=0}^{n-1} F(i) = \sum_{i=0}^{n-1} \lfloor \frac{n}{2^i} \rfloor.$
- $\sum_{i=0}^{n-1} F(i) \leq n \sum_{i=0}^{n-1} \frac{n}{2^i} \leq 2n.$
- $$T(n) = \frac{\sum_{i=0}^{n-1} F(i)}{n} \leq \frac{2n}{n} \leq 2 = O(1)$$

Análise Amortizada

- Esta forma de realizar a análise amortizada é conhecida como **análise agregada**.
- Existem duas outras formas de realizar essa análise:
    - Método Contável.
    - Método do Potencial.

Método do Potencial

- Vamos atribuir uma **energia potencial** para a estrutura de dados.
- Temos uma função que calcula essa potencial $\Phi()$.
- Seja D_0 uma estrutura de dados, temos $\Phi(D_0)$.
- Vamos realizar n operações com essa estrutura.
- Seja D_i é a estrutura após a i-ésima operação.

Método do Potencial

- Seja c_i o custo real de realizar a operação i na estrutura.
- O *custo amortizado por operação* $\hat{c}_i$ é dado por

$$\hat{c}_i = c_i + \Phi(D_i) - \Phi(D_{i-1})$$

- O custo amortizado total é

$$\sum_{i=1}^{n} \hat{c}_i = \sum_{i=1}^{n} c_i + \sum_{i=1}^{n} \Phi(D_i) - \sum_{i=1}^{n} \Phi(D_{i-1})$$

$$\sum_{i=1}^{n} \hat{c}_i = \sum_{i=1}^{n} c_i + \Phi(D_n) - \Phi(D_0)$$

Método do Potencial

- Vamos usar esse método para calcular o custo amortizado da operação de incremento.
- Assuma que a operação i reseta t_i bits.
- O custo real é então $t_i + 1$ (um bit é setado).
- Vamos definir o potencial da estrutura como sendo o número de bits setados em 1, b_i.

Método do Potencial

- $b_0 = 0$ e $b_n = k$.
- $b_i = b_{i-1} - t_i + 1$ isso significa que
  $b_i - b_{i-1} = -t_i + 1$
- $\hat{c}_i = c_i + \Phi(D_i) - \Phi(D_{i-1}) = t_i + 1 + (-t_i + 1) = 2$

Método Contável

- Vamos analisar o nosso contador binário.
- Vamos modificar o custos do nosso programa.
- Vamos assumir que flipar um bit para 1 custa 2.
- Porque?
- Porque mudar um bit para 0 sai de graça!