COMPLEXIDADE
PARTE III

Ana Neri (ana.i.neri@inesctec.pt)

DI,
Universidade do Minho

6 de Março 2023

Conteúdo

1 Relembrar a aula anterior 2

2 Análise amortizada 3

2.1 Análise Agregada 5

2.2 Método contabilístico 8

2.3 Método Potencial 11

2.4 Tabelas Dinâmicas 14

Relembrar a aula anterior

O programa seguinte gera todas as combinações de N bits:

```
def bitSeq(n, c='', r=[]):
  if n == 0:
      r.append(c)
  else:
      bitSeq(n-1, c+'0', r)
      bitSeq(n-1, c+'1', r)
  return r
```

1. define uma recorrência que exprima complexidade.
2. desenhe a árvore de recursão.
3. apresente uma solução para a recurrência

Análise amortizada

```
def inc(b):
  N=len(b)
  i=N-1
  while((i>=0) and (b[i] == 1)):
    b[i] = 0
    i-=1
  if (i>=0) :
    b[i] = 1
  return b
```

A complexidade:

1. no melhor caso é 1.
2. no pior caso é $N - 1$.
3. no caso médio é
   $\overline{T}(N) = \left(\sum_{k=1}^{N} \frac{k}{2^k}\right) + \frac{N}{2^N} = 2 - \frac{1}{2^{N-1}}$

Como $\lim \overline{T}(N) = 2$ temos $\overline{T}(N) = \Theta(1)$
Nesta caso, há uma grande diferença entre o caso médio e o pior caso porque o pior caso tem pouca probabilidade de acontecer.

Análise amortizada

Este tipo de situações leva-nos à **análise amortizada**, onde invés de analisar o custo de uma operação vamos analisar o custo de uma sequência de operações.

Tipicamente analisamos a sequência com o pior custo.

Com N operações $\{op_i\}_{1 \leq i \leq N}$ cada uma com custo c_i.
Queremos o custo amortizado da operação i, denotado por $\hat{c}_i$, de forma a que a soma dos custos amortizados seja um limite superior da soma dos custos reais:

$$\sum_{i=1}^{N} \hat{c}_i \geq \sum_{i=1}^{N} c_i \tag{1}$$

Análise Agregada

Primeiro passo na análise amortizada é calcular a soma dos custos reais das operações da sequência.
Com isso calculamos o custo amortizado da sequência:

$$\hat{c}_i = \frac{1}{N} \sum_{i=1}^{N} c_i \tag{2}$$

Vamos voltar à função `inc`.
Vamos considerar uma sequência de N invocações desta função a um mesmo vetor `b` com número suficientemente grande elementos, inicialmente todo preenchido a 0.

Análise Agregada

i	input	output	c_i
1	$\{\ldots,0,0,0,0,0\}$	$\{\ldots,0,0,0,0,1\}$	$1=1$
2	$\{\ldots,0,0,0,0,1\}$	$\{\ldots,0,0,0,1,0\}$	$2=1+1$
3	$\{\ldots,0,0,0,1,0\}$	$\{\ldots,0,0,0,1,1\}$	$1=1$
4	$\{\ldots,0,0,0,1,1\}$	$\{\ldots,0,0,1,0,0\}$	$3=1+1+1$
5	$\{\ldots,0,0,1,0,0\}$	$\{\ldots,0,0,1,0,1\}$	$1=1$
6	$\{\ldots,0,0,1,0,1\}$	$\{\ldots,0,0,1,1,0\}$	$2=1+1$
7	$\{\ldots,0,0,1,1,0\}$	$\{\ldots,0,0,1,1,1\}$	$1=1$
8	$\{\ldots,0,0,1,1,1\}$	$\{\ldots,0,1,0,0,0\}$	$4=1+1+1+1$
$\ldots$			

O objetivo é calcular a soma dos elementos da última coluna $\sum c_i$, i.e. o custo C de fazer N incrementos (sem overflow).

$$C_N = N + \frac{N}{2} + \frac{N}{4} + \cdots = \sum_{i=0}^{log_2 N} \frac{N}{2^i} = N \sum_{i=0}^{log_2 N} \frac{1}{2^i} = N(2 - \frac{1}{2^{log_2 N}}) = N(2 - \frac{1}{N}) = 2N - 1 \quad (3)$$

Análise Agregada

Para chegarmos aos custo amortizado temos de dividir o custo de N incrementos pelo número de operações em causa.

$$\hat{c}_i = \frac{C_N}{N} = \frac{1}{N}(2N - 1) = 2 - \frac{1}{N} = \Theta(1) \tag{4}$$

- Neste caso o custo médio calculado é diferente, mas em termos assimptóticos temos o mesmo resultado.
  A diferença deve-se ao fator excecional do caso de overflow.
- Focarmo-nos na análise assimptótica torna irrelevante começarmos ou não por um valor 0.
  Assumirmos outro valor inicial dá resultados de custo diferentes, mas assimptoticamente iguais.

Método contabilístico

No método contabilístico queremos estimar um custo $\hat{c}_i$ para a operação i de tal forma que a soma dos custos das operações da sequência seja menor do que a soma destes custos esperados, isto é,

$$\sum_{i=1}^{N} c_i \leq \sum_{i=1}^{N} \hat{c}_i \tag{5}$$

ou

$$\left(\sum_{i=1}^{N} \hat{c}_i -\right) \left(\sum_{i=1}^{N} c_i\right) \geq 0 \tag{6}$$

Para garantirmos que esta inequação é válida:

$$B_K \doteq \left(\sum_{i=1}^{K} \hat{c}_i\right) - \left(\sum_{i=1}^{K} c_i\right) \geq 0 \quad , \forall K \tag{7}$$

Método contabilístico

Podemos pensar nisto como uma conta bancária na qual se pretende garantir um saldo positivo.

- $D_K \doteq \sum_{i=1}^{K} \hat{c}_i$ soma de todos os depósitos até à operação K.
- $W_K \doteq \sum_{i=1}^{K} c_i$ soma de todos os levantamentos até à operação K.

Assim o saldo até à operação k é:

$$\begin{aligned} B_k &= D_k - W_k \\ &= \left(\sum_{i=0}^{k} \hat{c}_i\right) - \left(\sum_{i=0}^{k} c_i\right) \\ &= \left(\hat{c}_k \sum_{i=0}^{k-1} \hat{c}_i\right) - \left(c_k \sum_{i=0}^{k-1} c_i\right) \\ &= \left(\sum_{i=0}^{k-1} \hat{c}_i - \sum_{i=0}^{k-1} c_i\right) + (\hat{c}_k - c_k) \\ B_{k-1} &+ (\hat{c}_k - c_k) \end{aligned} \tag{8}$$

Por razões de completudo e dado que temos uma recorrência devemos indicar o valor inicial B_0. É costume usar 0 mas não é necessário para uma análise assimptótica.

Método contabilístico

Voltando ao programa `inc`:
O custo de cada operação é constante $\hat{c}_i$. Cada operação transforma uma sequência de 1s em 0s e finalmente transforma um 0 em 1. O custo de passar os 1s a 0 se faz à custa do valor *amealhado*, devemos amealhar 1, para mais tarde o podermos usar para converter esse 1.

i	input	c_i	$\hat{c}_i$	$\mathbf{B}_i$
0				0
1	$\{\ldots,0,0,0,0,0,0\}$	1	2	1
2	$\{\ldots,0,0,0,0,0,1\}$	2	2	1
3	$\{\ldots,0,0,0,0,1,0\}$	1	2	2
4	$\{\ldots,0,0,0,0,1,1\}$	3	2	1
5	$\{\ldots,0,0,0,1,0,0\}$	1	2	2
6	$\{\ldots,0,0,0,1,0,1\}$	2	2	2
7	$\{\ldots,0,0,0,1,1,0\}$	1	2	3
8	$\{\ldots,0,0,0,1,1,1\}$	4	2	1

i	input	c_i	$\hat{c}_i$	$\mathbf{B}_i$
9	$\{\ldots,0,0,1,0,0,0\}$	1	2	2
10	$\{\ldots,0,0,1,0,0,1\}$	2	2	2
11	$\{\ldots,0,0,1,0,1,0\}$	1	2	3
12	$\{\ldots,0,0,1,0,1,1\}$	3	2	2
13	$\{\ldots,0,0,1,1,0,0\}$	1	2	3
14	$\{\ldots,0,0,1,1,0,1\}$	2	2	3
15	$\{\ldots,0,0,1,1,1,0\}$	1	2	4
16	$\{\ldots,0,0,1,1,1,1\}$	5	2	1
17	$\{\ldots,0,1,0,0,0,0\}$	1	2	2

O valor estimado ($\hat{c}_i = 2$) é suficiente para garantir que o saldo nunca é negativo.
Em metade dos casos o custo real é 1, o que permite *poupar* 1 por cada uma dessas operações.

Método do Potencial

No método do potencial queremos calcular estes custos estimados a partir de uma função (dita de potencial) sobre a estrutura de dados (estado) em questão.
Considere uma função Φ que mapeia cada estado num número que satisfaz as seguinte propriedades:

- $\Phi(S) \geq 0$ para todos os estados S
- $\Phi(S_0) = 0$ no qual S_0 é o estado inicial.

O custo estimado de cada operações da sequência, usando a função sequencial é:

$$\hat{c}_i = c_i + (\Phi(S_i) - \Phi(S_{i-1})) \tag{9}$$

Onde:

- c_i é o custo real da operação i
- S_{i-1} e S_i são os estados antes e depois de executar a operação i.

Vamos simplificar $\Phi(S_i)$ por Φ_i.

Método Potencial

Estas definições permitem-nos obter um majorante para o custo de N operações.

$$\begin{aligned}\sum &= \hat{c}_1 + \hat{c}_2 + \cdots + \hat{c}_{N-1} + \hat{c}_N \\ &= (c_1 + \Phi_1 - \Phi_0) + (c_2 + \Phi_2 - \Phi_1) + \cdots + (c_N + \Phi_N - \Phi_{N-1}) \\ &= c_1 + \Phi_1 - \Phi_0 + c_2 + \Phi_2 - \Phi_1 + \cdots + c_N + \Phi_N - \Phi_{N-1} \\ &= c_1 + c_2 + \cdots + c_N + \Phi_N - \Phi_0 \\ &(\sum c_i) + \Phi_N - \Phi_0\end{aligned} \tag{10}$$

Com a igualdade:

$$\sum_{i=1}^{N} \hat{c}_i = \left(\sum_{i=1}^{N} c_i + \Phi_N - \Phi_0 \right) \tag{11}$$

e as propriedades da função potencial, concluímos:

$$\sum_{i=1}^{N} \hat{c}_i - \left(\sum_{i=1}^{N} c_i \right) = \Phi_N - \Phi_0 = \Phi_N - 0 \geq 0 \tag{12}$$

Neste contexto temos de caracterizar o custo real de cada operação:

Método Potencial

Voltado ao nosso exemplo: `inc`.
Vamos definir o potencial de um array de bits como o número de 1s desse array.
Esta função é independente de `inc`, só depende do valor do seu argumento (*estado*).
Para chegarmos aos valor esperado do custo de cada operação:

$$\hat{c}_i = c_i + \Phi_i - \Phi_{i-1} \tag{13}$$

Agora temos de caracterizar o custo real de cada operação.
A função `inc` está converter em 0s o maior prefixo de 1s do vetor, seguindo de converter um bit adicional.

- se o array b tem pelo menos um bit a 0, e se k é a posição do bit 0 mais à direita, então $\Phi(b) = l + r$, onde l é a soma do número de 1s até á posição k, e r é o número de 1s da posição k em diante.
- No estado após a execução de `inc`, o potencial passa a $\Phi(b') = l + 1$, uma vez que os r bits a 1 foram substituídos por 0

Logo o custo estimado de cada operação é:

$$\hat{c}_i = (l + 1) + (r + 1) - (l + r) = 2 \tag{14}$$

Tabelas Dinâmicas

Vamos agora ver um exemplo de análise amortizada considerando que temos uma estrutura de dados implementada num array em que o custo de fazer uma inserção é constante (digamos que é 1).
Vamos enriquecer esta implementação dizendo que no caso de querermos de inserir um novo elemento e o array esgotou a sua capacidade:

1. é realocado um novo array com o dobro da capacidade,
2. é copiado o conteúdo do array antigo para o novo array,
3. é adicionado o novo elemento

- Qual é o melhor caso?
- Qual é o pior caso?

Vamos ver como chegar à análise amortizada.

Próximo episódio

- Classes
- Máquinas de Turing