INF 02779 Análise de Algoritmos - 2002/2	01/21/03
Aula 7: Aula 07	
Instrutor: Berilhes Borges Garcia	Escriba: Hilario Seibel Junior

1 Relações de Recorrência

Muitos dos algoritmos que nos interessam neste curso podem ser expressos como procedimentos recursivos, dessa forma o custo de resolver um problema particular pode ser expresso como uma relação de recorrência. Por exemplo, nós vimos que o tempo de execução no caso médio do procedimento de Pesquisa Binária pode ser descrito pela recorrência:

$$T(n) = \left\{ \begin{array}{ll} T(n) = T(\frac{n}{2}) + \Theta(1) & se\ n > 1, \\ 1 & se\ n = 1. \end{array} \right\} \tag{1}$$

cuja solução nós afirmamos ser $T(n) = \Theta(lg(n))$ O desenvolvimento de uma relação de recorrência descrevendo a performance de um algoritmo é, por si só, um grande passo para a sua análise, uma vez que a própria recorrência seja uma grande quantidade de informação. Propriedades específicas do algoritmo estão encapsuladas em uma expressão matemática relativamente simples. Esta parte do curso visa estudar alguns dos métodos utilizados para resolver relações de recorrência.

2 Método da Substituição

Adivinhando a forma das soluções, e então utilizando indução matemática para encontrar as constantes e mostrar que a solução está correta.
Este método funciona bem quando é fácil adivinhar. Mas, não existe nenhuma forma geral de adivinhar a solução correta.
Example:

$$T(n) = 3T(\lfloor \frac{n}{3} \rfloor) + n \tag{2}$$

Nós adivinhamos que a solução é $T(n) = O(nlg(n))$. Dessa forma nós devemos provar que $T(n) \leq cnlg(n)$ para escolha apropriada da constante c.

$$\begin{aligned} T(\lfloor \frac{n}{3} \rfloor) &\leq c\lfloor \frac{n}{3} \rfloor lg(\lfloor \frac{n}{3} \rfloor) && (3) \\ T(n) &\leq 3c\lfloor \frac{n}{3} \rfloor lg(\lfloor \frac{n}{3} \rfloor) + n && (4) \\ &\leq cnlg(\frac{n}{3}) + n && (5) \\ &= cnlg(n) - cnlg(3) + n && (6) \\ &= cnlg(n) - n(clg(3) - 1) && (7) \\ &\leq cnlg(n) && (8) \end{aligned}$$

O último passo é verdade para $c \leq \frac{1}{3}$ ⊠

Condições de Fronteira ....

Example: Considere agora a recorrência obtida no algoritmo 5 do problema de Fibonacci e no algoritmo de Pesquisa Binária:

$$T(n) = T(\frac{n}{2}) + \Theta(1) \tag{9}$$

ou

$$T(n) = T(\frac{n}{2}) + 1$$

Suponha que a solução desta recorrência seja $T(n) = O(lg(n))$. Vamos mostrar agora que esta adivinhação está correta.
Assumindo que $T(\frac{n}{2}) \leq c\ lg(\frac{n}{2})$, nós temos então que:

$$T(n) \leq clg(\frac{n}{2}) + 1 = c(lg(n) - lg(2)) + 1 = clg(n) - c + 1 \leq clg(n) \tag{11}$$

Onde o último passo é verdade para $c \geq 1$. ⊠

Fazer uma boa adivinhação consiste em grande parte em ser capaz de encontrar uma relação de recorrência similar, e então utilizar a solução desta relação de recorrência como uma solução candidata. Por exemplo, considere a relação de recorrência

$$T(n) = 2T(\lfloor\frac{n}{2}\rfloor + 5) + n, \tag{12}$$

esta relação é similar à $T(n) = 3T(\lfloor\frac{n}{3}\rfloor) + n$, quando n é grande, a diferença entre $\frac{n}{3}$ e $\frac{n}{3} + 5$ é insignificante. Dessa forma nós podemos advinhar que a solução da recorrência é $T(n) = O(nlg(n))$.

3 Algumas Sutilezas

Quando a prova matemática não pode ser obtida por indução, você pode tentar ajustar a sua adivinhação com um tempo de ordem mais baixa.
Example: Considere a seguinte recorrência:

$$T(n) = 3T(\lfloor\frac{n}{3}\rfloor) + 4 \tag{13}$$

Suponha que nós adivinhamos que $T(n) = O(n)$. Logo, nós devemos provar por indução que:

$$T(n) \leq cn, \tag{14}$$

para alguma constante positiva c. Substituindo nossa adivinhação na recorrência, nós obtemos

$$T(n) \leq 3\lfloor\frac{n}{3}\rfloor + 4 = cn + 4 \tag{15}$$

o que não implica que $T(n) \leq cn$, como era exigido. Contudo nós podemos ajustar nossa adivinhação para $T(n) \leq cn - b$, para alguma constante positiva b. Nós agora temos:

$$T(n) \leq 3(c\lfloor\frac{n}{3}\rfloor - b) + 4 = cn - 3b + 4 = cn - b - (2b - 4), \tag{16}$$

Portanto, $T(n) \leq cn - b$ se $2b - 4 \geq 0$, ou seja se $b \geq 2$ ⊠