1 Recorrência

Uma recorrência é uma expressão que dá o valor de uma função em termos dos valores "anteriores"da mesma função.

Para analisar o consumo de tempo de um algoritmo recursivo é necessário resolver uma recorrência.

O exemplo clássico de recorrência, provavelmente o mais fomoso, é a formula de Fibonacci:

$$F(n) = \begin{cases} 0 & \text{se } n = 0 \\ 1 & \text{se n = 1} \\ F(n-1) + F(n-2) & \text{se } n \geq 2 \end{cases}$$

Outro exemplo clássico de recorrência é:

$$F(n) = \begin{cases} 1 & \text{se n = 0} \\ n.F(n-1) & \text{se } n \geq 1 \end{cases}$$

Um fórmula fechada para F(n) é dada por:

$$F(n) = n.F(n-1)$$

$$= n.(n-1).F(n-2) = ... =$$

$$= n.(n-1).(n-2)....2.1.1$$

$$= n!$$

Resolver uma recorrência é encontrar uma "fórmula fechada"que dê o valor da função diretamente em termos do seu argumento.
Vamos analisar o consumo de tempo do algoritmo da busca binária.
Em cada iteração, o algoritmo descarta metade do valor. Se denotamos por $T(n)$ o número máximo de iterações realizadas pela busca binária sobre um vetor com n elementos, a função $T(n)$ pode ser expressa pela seguinte recorrência:

$$T(n) = T(\lfloor \frac{n}{2} \rfloor) + 1,\ T(1) = 1.$$

Para obter uma solução, supomos inicialmente que $n = 2^k$.

$$\begin{aligned} T(n) &= T(\lfloor \frac{n}{2} \rfloor) + 1 \\ = (T(\frac{n}{4}) + 1) + 1\ &= T(\frac{n}{2^2}) = 2 \\ &= T(\frac{n}{2^3}) + 3 \\ = T(\frac{n}{2^k}) + k = T(1) + k &= k + 1 \\ &= log\ n + 1 \end{aligned} \tag{1}$$

Podemos agora tentar mostrar que $T(n) \leq \log n + 1, \ \forall n \geq 1$, por indução em n.
Para $n = 1, T(1) = 1 = \log 1 + 1$ Para $n > 1$, temos

continua...

2 Recorrência - Exercicios

1. Resolva a recorrência $T(n) = T(\frac{n}{2}) + 7n + 2$, $T(1) = 1$.
2. Mostre que a solução de $T(n) = T(n-1) + n$ é $O(n^2)$.
3. Mostre que a solução de $T(n) = T(\lceil \frac{n}{2} \rceil) + 1$ é $O(\log n)$.
4. Mostre que a solução de $T(n) = 2T(\lfloor \frac{n}{2} \rfloor + 17) + n$ é $O(n \log n)$.
5. Resolva a recorrência $T(n) = 3T(\lfloor \frac{n}{2} \rfloor) + n$, $T(1) = 1$.
6. Resolva a recorrência $T(n) = 3T(\lfloor \frac{n}{3} \rfloor) + 1$, $T(1) = 1$.
7. Mostre que a solução de $T(n) = 3T(\frac{n}{2}) + n$ é $O(n^{\log_2 3})$.
8. Mostre que a solução de $T(n) = 4T(\frac{n}{2}) + n$ é $O(n^2)$.

3 Respostas

1. $T(n) = T(\frac{n}{2}) + 7n + 2$, $T(1) = 1$.
   $T(n) \leq O(n^2), \forall n \geq 8$, por indução em n

   $T(1) = (1)^2 = 1$
   $T(2) = T(\frac{2}{2}) + 7 * 2 + 2 = 17 \not\leq n^2(4)$
   $T(4) = T(\frac{4}{2}) + 7 * 4 + 2 = 32 \not\leq n^2(16)$
   $T(8) = T(\frac{8}{2}) + 7 * 8 + 2 = 62 \leq n^2(64)$