MCTA003-17 - Análise de Algoritmos
Centro de Matemática, Computação e Cognição
Universidade Federal do ABC
Prof. Maycon Sambinelli

Recorrências

1. Seja F_n o n-ésimo número da sequência de Fibonacci. Temos $F_1 = 1$, $F_2 = 1$ e $F_n = F_{n-1} + F_{n-2}$ para $n \geq 3$. Use indução e prove que $F_n \geq 2^{0.5n}$ para todo $n \geq 6$. Qual o significado desse resultado?

2. Considere os seguintes algoritmos recursivos que resolvem um mesmo problema em uma entrada de tamanho n:

Algoritmo A: Divide o problema em 2 partes de tamanho $n/4$ cada, resolve-as e gasta tempo adicional n.

Algoritmo B: Divide o problema em 9 partes de tamanho $n/3$ cada, resolve-as e gasta tempo adicional n.

Algoritmo C: Divide o problema em 6 partes de tamanho $n/5$ cada, resolve-as e gasta tempo adicional $\sqrt{n}$.

No que segue, assuma $T(1) = 1$. Para cada um dos três algoritmos, monte uma recursão para seu tempo de execução $T(n)$ e a resolva com o melhor limitante possível (notação O ou Θ) usando:

(a) **Algoritmo A**: Método de substituição.
(b) **Algoritmo B**: Método de iteração.
(c) **Algoritmo C**: Método mestre.

Qual algoritmo é mais eficiente?

3. Suponha que $T(1) = 1$ e $T(2) = 1$. Resolva as seguintes recorrências com notação O com o resultado mais justo possível (quando não especificado, use o método que preferir):

(a) $T(n) = T(\frac{n}{3}) + n^2$ (método de iteração)
(b) $T(n) = 2T(n-1) + n$ (método de iteração)
(c) $T(n) = 4T(\frac{n}{2}) + n$ (método de substituição, suponha $T(n) = \Theta(n^2)$)
(d) $T(n) = T(n-1) + T(n-2) + 3$ (método de substituição, suponha $T(n) = \Theta(2^n)$)
(e) $T(n) = 4T(\frac{n}{2}) + \sqrt{n}$ (árvore de recursão e método mestre)
(f) $2T(\lfloor \frac{n}{2} \rfloor) + n$ (método de substituição)
(g) $T(n) = 7T(\frac{n}{3}) + n^2$ (árvore de recursão e método de substituição)
(h) $T(n) = 2T(\frac{n}{4}) + 1$ (método mestre)
(i) $T(n) = 2T(\frac{n}{4}) + \sqrt{n}$ (método mestre)
(j) $2T(\lfloor \frac{n}{2} + 17 \rfloor) + n$ (método de substituição)
(k) $T(n) = 2T(\frac{n}{4}) + n$ (método mestre)
(l) $T(n) = 2T(\frac{n}{4}) + n^2$ (método mestre)
(m) $T(n) = T(\sqrt{n}) + 1$ (faça uma mudança de variáveis, usando $m = \log n$, escreva uma recorrência equivalente e resolva-a)

(n) $T(n) = T(\frac{n}{2}) + T(\frac{n}{4}) + n$

(o) $T(n) = 2T(\frac{3n}{4}) + n$

(p) $T(n) = T(\frac{2n}{3}) + 2T(\frac{n}{3}) + 1$ (método de substituição)

(q) $T(n) = 64T(\frac{n}{8}) + 7n^3$

(r) $T(n) = 4T(\frac{n}{8}) + \sqrt{n}$

(s) $T(n) = 2T(\frac{n}{2}) + n \log n$ (chute: $T(n) = \Theta(n \log^2 n)$)

(t) $T(n) = T(\frac{n}{3}) + T(\frac{n}{6}) + T(\frac{n}{9}) + n$

4. Um algoritmo A tem o seu consumo de tempo descrito através da recorrência

$$T(n) = 7T(n/2) + n^2 .$$

Um outro algoritmo B tem o seu consumo de tempo descrito através da recorrência

$$T'(n) = aT'(n/4) + n^2 ,$$

onde a é uma constante. Qual é o maior valor inteiro possível para a de tal forma que o algoritmo B seja assintoticamente mais eficiente que o algoritmo A? Justifique a sua resposta.