IA012 - Tarefa 6

Rodrigo Seiji Piubeli Hirao (186837)

16 de dezembro de 2021

01) A complexidade de algoritmos dita o crescimento de uma função, o que se mostra muito importante para valores muito grandes. Por exemplo $O(\log_2 n)$ não será tão distante de $O(n)$ para valores muito pequenos, como $n = 2$, mas para valores muito grandes como os usados na criptografia um algoritmo se mostra muito mais viável que o outro, por exemplo $n = 2^{30000}$ onde teríamos que executar 3×10^4 operações em um e aproximadamente 10^{10000} na outra.

02) Complexidade espacial é o crescimento do espaço físico utilizado por um algoritmo, enquanto complexidade temporal é o crescimento do tempo de execução necessário para executar um algoritmo.

03) Primeiro é necessário descobrir a função que dita o algoritmo, por exemplo um algoritmo que executa 2 loops encadeados, e 3 loops individuais logo a seguir (sem nenhum tipo de otimização) provavelmente seguirá a função de execuções atômicas $n^2 + 3n$, que então podemos comparar com uma outra função para descobrir a que grupo essa função pertence usando a fórmula da equação 1, seguindo a regra da equação 2 e 3.

$$c = \lim_{n \to \infty} \frac{f_1}{f_2} \tag{1}$$

$$\begin{cases} f_1 \in O(f_2) & , c = 0 \\ f_1 \in \Theta(f_2) & , 0 < c < \infty \\ f_1 \in \Omega(f_2) & , c \to \infty \end{cases} \tag{2}$$

$$\begin{cases} f_1 \in \Theta(f_2) & \to f_1 \in O(f_2) \ \& \ f_1 \in \Omega(f_2) \\ f_1 \notin \Theta(f_2) \ \& \ f_1 \in O(f_2) & \to f_1 \in o(f_2) \\ f_1 \notin \Theta(f_2) \ \& \ f_1 \in \Omega(f_2) & \to f_1 \in \omega(f_2) \end{cases} \tag{3}$$

Assim, no nosso exemplo podemos ter que $\lim_{n \to \infty} \frac{n^2+3n}{n^2} = 1$, logo $n^2 + 3n \in \Theta(n^2)$

04) Pode ser estimado comparando o algoritmo com outro já existente ou imaginando quantos loops e recursões existirão no algoritmo. Deve ser também levado em conta as otimizações que podem fazer loops serem menos de n e resursões que podem ter comportamentos variados.

05) Big-O é o grupo de todas as funções que crescem mais devagar ou igual á função de parâmetro, Big-Ω são as que crescem mais rápido ou igual, e Θ são as que crescem igual, pode ser visto isso nas equações 2 e 3 respondidas no exercício 3.

06) Big-O e little-o são os grupos de funções que crescem menos que a função parâmetro, sendo o Big-O incluso a própria função de parâmetro. Assim qualquer função que pertença a $O(f(n))$ terá o crescimento menor ou igual a $f(n)$, e no caso de $o(f(n))$ o crescimento será sempre menor. Isso pode ser visto melhor pelas equações 2 e 3 respondidas no exercício 3.

07)

- **bubblesort** - $O(n^2)$ - o algoritmo irá percorrer a lista inteira (n) para cada elemento, ou seja n vezes n.
- **mergesort** - $O(nlog_w(n))$ - isso se deve ao fato de ele dividir a lista em 2 sempre e ordenar essas 2 metades, fazendo isso recusrivamente. Assim tendo uma árvore de altura $log_2(n)$ e em cada nível da árvore percorrerá n.
-]**fatoração** - $O(n)$ - se o algoritmo for implementado testando todos os números anterior a $\sqrt{n}$ para descobrir os primos e fazer a mesma coisa com todos esses $\sqrt{n}$ números, temos uma complexidade de n.

08) Ambos mergesort e quicksort podem ser implementados com compexidades de melhor caso e caso médio de $O(n \log_2(n))$, mas o quicksort possui uma complexidade de pior caso de $O(n^2)$, pois neste caso ele não funcionará muito diferente de um bubble sort, enquanto o mergesort sempre terá a mesma complexidade pois sua implementação independe do formato da entrada.

09) Podemos sim, a complexidade dita apenas o crescimento da função, sendo importante para valores muito grandes, porém com valores pequenos podemos ter a situação de $200n \in O(n)$ e $2^n \in O(2^n)$, onde se tivermos um $n = 10$ por exemplo a função exponencial executará mais rápida que a função polinomial.

10) Pelo ataque do aniversário temos um caso médio de colisão pertencente a $O(2^{\frac{m}{2}})$, podemos fazer o teste de novo, mas para números muito distantes, resultando na mesma complexidade, logo o total seria $O(2^{\frac{m}{2}+1})$ que pertence a $O(2^{\frac{m}{2}})$.