Uma introdução à lógica assistida por computador

Flávio L. C. de Moura
Departamento de Ciência da Computação
Universidade de Brasília[1]

11 de novembro de 2024

[1] flaviomoura@unb.br

Capítulo 1

O algoritmo *mergesort* (versão com *merge* iterativo)

Algoritmos recursivos desempenham um papel fundamental em Computação. O algoritmo de ordenação *mergesort* é um exemplo de algoritmo recursivo, que se caracteriza por dividir o problema original em subproblemas que, por sua vez, são resolvidos recursivamente. As soluções dos subproblema são então combinadas para gerar uma solução para o problema original. Este paradigma de projeto de algoritmo é conhecido com *divisão e conquista*. Este algoritmo foi inventado por J. von Newmann em 1945.

O algoritmo *mergesort* é um algoritmo de ordenação que utiliza a técnica de divisão e conquista, que consiste das seguintes etapas:

1. **Divisão**: O algoritmo divide a lista (ou vetor) l recebida como argumento ao meio, obtendo as listas l_1 e l_2;

2. **Conquista**: O algoritmo é aplicado recursivamente às listas l_1 e l_2 gerando, respectivamente, as listas ordenadas l'_1 e l'_2;

3. **Combinação**: O algoritmo combina as listas l'_1 e l'_2 através da função *merge* que então gera a saída do algoritmo.

Por exemplo, ao receber a lista $(4 :: 2 :: 1 :: 3 :: nil)$, este algoritmo inicialmente divide esta lista em duas sublistas, a saber $(4 :: 2 :: nil)$ e $(1 :: 3 :: nil)$. O algoritmo é aplicado recursivamente às duas sublistas para ordená-las, e ao final deste processo, teremos duas listas ordenadas $(2 :: 4 :: nil)$ e $(1 :: 3 :: nil)$. Estas listas são, então, combinadas para gerar a lista de saída $(1 :: 2 :: 3 :: 4 :: nil)$.

if $p < r$ **then**
    $q = \lfloor \frac{p+r}{2} \rfloor$;
    mergesort(A, p, q);
    mergesort$(A, q+1, r)$;
    merge(A, p, q, r); ← $T_{merge}(n)$
end

$$T_{ms}(n) = T_{ms}(n/2) + T_{ms}(n/2) + T_{merge}(n)$$
$$= 2 . T_{ms}(n/2) + T_{merge}(n).$$

Algoritmo 1: mergesort(A, p, r)

A etapa de combinar dois vetores ordenados (algoritmo *merge*) é a etapa principal do algoritmo *mergesort*. O procedimento *merge*(A, p, q, r) descrito a seguir recebe como argumentos o vetor A, e os índices p, q e r tais que $p \leq q < r$. O procedimento assume que os subvetores $A[p..q]$ e $A[q+1..r]$ estão ordenados.

$$\boxed{T_{ms}(n) = 2 . T_{ms}(n/2) + \theta(n)}$$

$$\rightarrow \begin{cases} T_{ms}(n) = 2 \cdot T_{ms}(n/2) + c \cdot n \quad (c>0) \end{cases}, \quad n = 2^K$$

$$T_{ms}(1) = 0$$

$$T_{ms}(2^K) = 2 \cdot T_{ms}(2^{K-1}) + c \cdot 2^K$$

$$= 2 \cdot \left(2 \cdot T_{ms}(2^{K-2}) + c \cdot 2^{K-1}\right) + c \cdot 2^K$$

$$= 2^2 \cdot T_{ms}(2^{K-2}) + c \cdot 2^K + c \cdot 2^K$$

$$= 2^2 \cdot \left(2 \cdot T_{ms}(2^{K-3}) + c \cdot 2^{K-2}\right) + 2 \cdot c \cdot 2^K$$

$$= 2^3 \cdot T_{ms}(2^{K-3}) + 3 \cdot c \cdot 2^K$$

?

$$= \dots$$

$$= 2^K \cdot T_{ms}(2^{K-K}) + K \cdot c \cdot 2^K$$

$$= 2^K \cdot \underbrace{T_{ms}(1)}_{0} + K \cdot c \cdot 2^K$$

$$(n = 2^K) \Downarrow K = \lg n$$

$$\rightarrow \quad = K \cdot c \cdot 2^K$$

$$\rightarrow T_{ms}(n) = (\lg n) \cdot c \cdot n$$

$$\rightarrow \quad = c \cdot n \cdot \lg n$$

$$\begin{cases} T_{ms}(2^K) = 2 \cdot T_{ms}(2^{K-1}) + c \cdot 2^K \\ T_{ms}(1) = 0 \end{cases}$$

$$\boxed{T_{ms}(2^K) = c \cdot K \cdot 2^K}$$

$n_1 = q - p + 1$; // Qtd. de elementos em $A[p..q]$
$n_2 = r - q$; // Qtd. de elementos em $A[q+1..r]$
let $L[1..n_1+1]$ and $R[1..n_2+1]$ be new arrays;
for $i = 1$ *to* n_1 **do**
    $L[i] = A[p+i-1]$;
end
for $j = 1$ *to* n_2 **do**
    $R[j] = A[q+j]$;
end
$L[n_1+1] = \infty$;
$R[n_2+1] = \infty$;
$i = 1$;
$j = 1$;
for $k = p$ *to* r **do**
    if $L[i] \leq R[j]$ **then**
        $A[k] = L[i]$;
        $i = i + 1$;
    end
    else
        $A[k] = R[j]$;
        $j = j + 1$;
    end
end

$|A[p..r]| = n.$

$T_{merge}(n) = n \Rightarrow T_{merge}(n) = \Theta(n).$

Algoritmo 2: merge(A, p, q, r)

Exercício 1. *Prove que o algoritmo merge é correto.*

Exercício 2. *Prove que o algoritmo mergesort é correto.*

Exercício 3. *Faça a análise assintótica do algoritmo merge.*

Exercício 4. *Faça a análise assintótica do algoritmo mergesort.*