Complexidade de Tempo de Algoritmos Recursivos

Prof. André Vignatti

Problema: Dados $A[p \ldots q]$ e $A[q{+}1 \ldots r]$ crescentes, rearranjar $A[p \ldots r]$ de modo que ele fique em ordem crescente.

Entrada:

	p				q				r
A	22	33	55	77	99	11	44	66	88

Saída:

	p				q				r
A	11	22	33	44	55	66	77	88	99

```
INTERCALA(A, p, q, r)
para i ← p até q faça
    B[i] ← A[i]
para j ← q + 1 até r faça
    B[r + q + 1 − j] ← A[j]
i ← p
j ← r
para k ← p até r faça
    se B[i] ≤ B[j]
       então A[k] ← B[i]
                   i ← i + 1
         senão A[k] ← B[j]
                   j ← j − 1
```

Complexidade de Intercala

Tamanho da entrada: $n = r - p + 1$

Consumo de tempo: $\Theta(n)$

Corretude do Intercala

Invariantes do Intercala: no começo de cada iteração do laço das linhas 7–12:

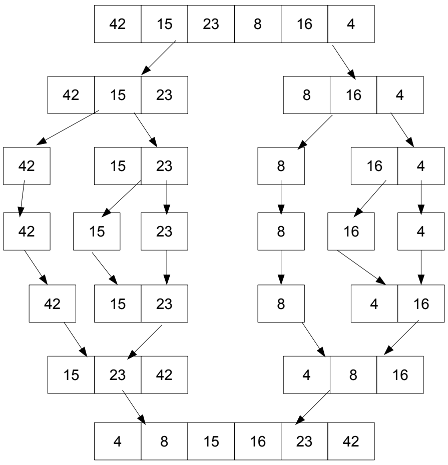
1. $A[p \ldots k-1]$ está ordenado,
2. $A[p \ldots k-1]$ contém todos os elementos de $B[p \ldots i-1]$ e de $B[j+1 \ldots r]$,
3. $B[i] \geq A[k-1]$ e $B[j] \geq A[k-1]$.

Exercício. Prove que a afirmação acima é de fato um invariante de INTERCALA.

Exercício. (fácil) Mostre usando o invariante acima que INTERCALA é correto.

MergeSort

MERGESORT(A, p, r)
se $p < r$
    então $q \leftarrow \lfloor (p+r)/2 \rfloor$
    MERGESORT(A, p, q)
    MERGESORT$(A, q+1, r)$
    INTERCALA(A, p, q, r)



O MergeSort é projetado com a técnica de **divisão e conquista**.

Algoritmos de divisão-e-conquista possuem (geralmente) três etapas:

1. **Divisão:** a instância do problema é dividida em instâncias de tamanho menor, gerando subproblemas.
2. **Conquista:** cada subproblema é resolvido recursivamente.
3. **Combinação:** as soluções dos subproblemas são combinadas para obter uma solução do problema original.

No caso do MergeSort:

1. **Divisão**: divida o vetor com n elementos em dois subvetores de tamanho $\lfloor n/2 \rfloor$ e $\lceil n/2 \rceil$
2. **Conquista**: ordene os dois subvetores **recursivamente** usando o Mergesort
3. **Combinação**: intercale os dois subvetores para obter um vetor ordenado usando o algoritmo Intercala

Análise de Algoritmos Recursivos

- Como mostrar a corretude de um algoritmo recursivo?
- Como analisar o tempo de um algoritmo recursivo?
  - O que é uma **relação de recorrência**?
  - O que significa *resolver* uma relação de recorrência?

Corretude

A corretude do Mergesort apoia-se na corretude do Intercala, e segue por indução em $n = r - p + 1$.

Complexidade

Seja $T(n)$ o consumo de tempo em função de $n = r - p + 1$

linha	consumo de tempo
1	$\Theta(1)$
2	$\Theta(1)$
3	$T(\lceil n/2 \rceil)$
4	$T(\lfloor n/2 \rfloor)$
5	$\Theta(n)$

Obtemos uma **relação de recorrência**:

$$T(n) = \begin{cases} \Theta(1) & \text{se } n \leq 1 \\ T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) & \text{se } n > 1 \end{cases}$$

Em geral, a análise de algoritmos recursivos levam a uma relação de recorrência.

- É necessário então **resolver** a recorrência!
- Ou seja, obter uma "fórmula não-recursiva" ("fórmula fechada") para $T(n)$.

Nas próximas aulas, veremos que $T(n) = \Theta(n \lg n)$ (no pior ou melhor caso?).