算法设计与分析

CS2008 班 U202015533 徐瑞达

2022.03.08

1 作业 1: 抄写伪代码

1.1 INSERTIONSORT

Algorithm 1: INSERTION-SORT(A)

```
for j=2 to A.length do
    $key = A[j]$
    // INSERT $A[j]$ into the sorted sequence $A[1..j-1]$
    $i = j - 1$
    while i>0 and A[i]>key do
        $A[i+1] = A[i]$
        $i = i - 1$
    $A[i+1] = key$
```

1.2 MERGESORT

Algorithm 2: MERGE-SORT(A, p, r)

```
if p<r then
    $q = \lfloor (p+r)/2 \rfloor$
    MERGE-SORT$(A, p, q)$
    MERGE-SORT$(A, q+1, r)$
    MERGE$(A, p, q, r)$
```

1.3 MERGE

Algorithm 3: MERGE-SORT(A,p,r)

```
$n_1 = q - p + 1$
$n_2 = r - q$
let $L[1..n_1 + 1]$ and $R[1..n_2 + 1]$ be new arrays
for $i$=1 to $n_1$ do
    $L[i] = A[p + i - 1]$
for $j$=1 to $n_2$ do
    $R[j] = A[q + j]$
$L[n_1 + 1] = \infty$
$R[n_2 + 1] = \infty$
$i = 1$
$j = 1$
for $k = ptor$ do
    if $L[i] \leq R[j]$ then
        $A[k] = L[i]$
        $i = i + 1$
    else $A[k] = R[j]$
        $j = j + 1$
```

2 作业 2:*Q&A*

2.1 1.2-2

Question:

Suppose We are comparing implementations of insertion sort and merge sort on the same machine. For inputs of size n,insertion sort runs in $8n^2$ steps, while merge sort runs in $64n\lg n$ steps. For which values of n does insertion sort beat merge sort?

Answer:

$8n^2 < 64n\lg n$

$2^n < n^8$

$2 \leq n \leq 43$

2.2 1.2-3

What is the smallest value of n such that an algorithm whose running time is $100n^2$ runs faster than an algorithm whose running time is 2^n on the same machine?

Answer:

$100n^2 < 2^n$

$n \geq 15$

3 思考题: 选择算法

Question:

Consider sorting n numbers stored in array A by first finding the smallest element of A and exchanging it with the element in $A[1]$. Then find the second smallest element of A, and exchange it with $A[2]$. Continue in this manner for the first $n-1$ elements of A.Write pseudocode for this algorithm, which is known as selection sort. What loop invariant does this algorithm maintain? Why does it need to run for only the first $n-1$ elements, rather than for all n elements? Give the best-case and worst-case running times of selection sort in Θ-notation.

Pseudocode:

Algorithm 4: SELCTION-SORT

```
$n = A.length$ for $i$=1 to $n$-1 do
    $index = i$
    for $j$=$i$+1 to $n$ do
        if $A[j]$<$A[index]$ then
            $index = j$
    $swap(A[i], A[index])$
```

Loop invariant:

在循环初始，子数组 $A[1..i-1]$ 由最小的 $i-1$ 个元素有序排列而成

Why does it need to run for only the first $n-1$ elements, rather than for all n elements?

经过 n-1 次迭代，子数组 $A[1..n-1]$ 由最小的 $n-1$ 个元素有序排列而成, 因此,$A[n]$ 仍然是最大的元素

Running time:

$\Theta(n^2)$