

CS483 Fall03 - Sample Homework 1, Solutions, Jana Košecká

1. Consider sorting n numbers in an array by first finding the smallest element of A and exchanging it with $A[1]$, then finding second smallest element and exchanging it with $A[2]$ and continue in this manner for the first $n - 1$ elements. Write a pseudocode of this algorithm. Why does the algorithm need to run for only $n - 1$ elements? Give the best-case and worst case running time in terms of Θ notation.

Pseudo-code of the algorithm :

```

Selection Sort(A,n)
for i=1 to n-1 do
    % find the smallest element in A[i] ... A[n]
    min = A[i] ; loc = i ;
    for j = i+1 to n do
        if A[j] < min then
            min = A[j]; loc = j;
        endif
    end
    swap A[i] and A[loc]
end
    
```

Analysis:

We have two loops one for $1 \dots n - 1$ and one for $i \dots n$, and suppose that number of operations with a loop will take constant time c_1 and c_2 . Then total time in the worst case will be

$$\begin{aligned}
 T(n) &= \sum_{i=1}^{n-1} (c_1 + \sum_{j=i+1}^n c_2) = \sum_{i=1}^{n-1} (c_1 + c_2(n-i)) \\
 &= c_1(n-1) + c_2 \sum_{i=1}^n i = c_1(n-1) + c_2 n(n-1)/2 \Rightarrow T(n) = \Theta(n^2)
 \end{aligned}$$

2. Compare the functions $(\frac{3}{2})^n, n^3, \lg^2 n, 2^{2^n}, 1$.
 Answer (using either limit or definition) :
 $1 = O(\lg^2 n), \lg^2 n = O(n^3), n^3 = O((\frac{3}{2})^n), (\frac{3}{2})^n = O(2^{2^n})$.
3. Compute the solutions of the following recurrences:

a. $T(n) = 4T(n/2) + n^2$

Using Master's theorem: $a = 4, b = 2, f(n) = n^2, \log_2 4 = 2$

Case 2 applies: $f(n) = \Theta(n^2) \Rightarrow T(n) = \Theta(n^2 \lg n)$.

b. $T(n) = T(n-1) + n$ for $n \geq 2$ and $T(1) = 1$

$$T(n) = T(n-2) + n - 1 + n = \dots = 1 + 2 + 3 + \dots + (n-1) + n = n(n-1)/2 \Rightarrow T(n) = \Theta(n^2)$$

c. $T(n) = 1$ for $n = 1$ and $T(n) = 3T(\frac{n}{2}) + n \lg n$ for $n > 1$

Master's theorem: $a = 3, b = 2, f(n) = n \lg n, \log_2 3 = 1.585\dots > 1$. Then $f(n) = O(n^{\lg_2 3 - \epsilon})$ for some $\epsilon > 0$ (e.g. using limits). Case 1 applies $T(n) = \Theta(n^{\lg_2 3})$.

d. $T(n) = 6T(\frac{n}{5}) + n \lg^6 n$

$a = 6, b = 5; f(n) = n \lg^6 n, \log_5 6 = 1.11\dots$

Case 1 applies: $f(n) = O(n^{1.1-\epsilon}) \Rightarrow T(n) = \Theta(n^{\log_5 6})$.

4. Given the recurrence equation $T(n) = 2T(\frac{n}{2}) + 2$ and $T(1) = 1$. Prove by induction that $T(n) = 3n - 2$.
 Base case: $T(1) = 3 \cdot 1 - 2 = 1$. Induction hypothesis: $T(k) = 2T(\frac{k}{2}) - 2$ for $0 < k < n$. To prove:
 $T(n) = 2T(\frac{n}{2}) + 2$ use induction hypothesis for $T(\frac{n}{2})$. Then $T(n) = 2 \cdot (3 \cdot \frac{n}{2} - 2) + 2 = 3n - 2$.
5. (10) Solve the following recurrence by appealing to the recurrence tree.

$$T(n) = T(n/3) + T(2n/3) + cn$$

Lower bound is provided by the length of the shortest path in the recurrence tree. In this case the shortest branch has length $\lg_3 n$ and since at each level the cost is cn then $T(n) = \Omega(n \lg_3 n) = \Omega(n \lg n)$.