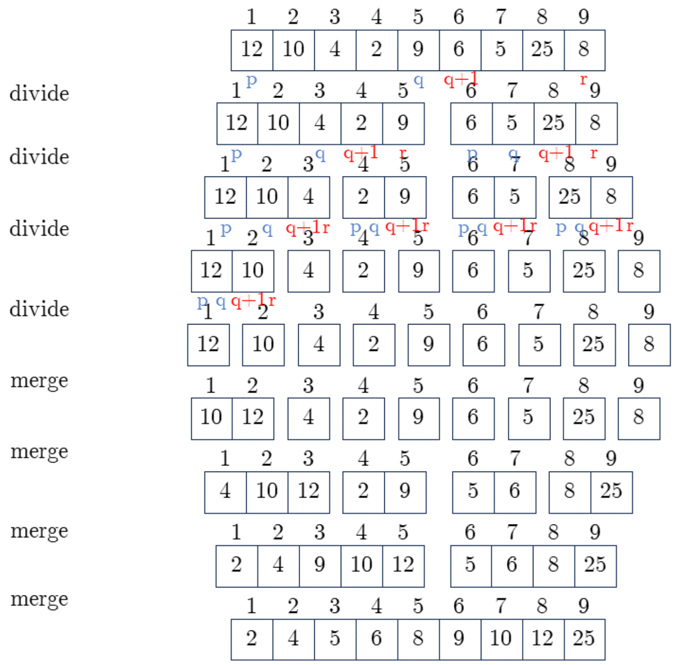
Data Structure and Algorithm Analysis(H)
Southern University of Science and Technology
Mengxuan Wu
12212006

Work Sheet 3

Mengxuan Wu

Question 3.1



Question 3.2

Proof.
Since $n = 2^k$, we can rewrite the equation as:

$$T(2^k) = \begin{cases} d & \text{if } k = 0 \\ 2T(2^{k-1}) + c2^k & \text{if } k \geq 1 \end{cases}$$

And the term we want to proof now becomes

$$T(2^k) = d2^k + c2^k \log 2^k = d2^k + ck2^k = (d + ck)2^k$$

Base case:
$k = 0$, $T(2^0) = d = (d + c \cdot 0)2^0$

Inductive step:
Assume when $k = k_0$, $T(2^{k_0}) = (d + ck_0)2^{k_0}$ holds. For $k = k_0 + 1$, we have

$$\begin{aligned} LHS &= T(2^{k_0+1}) \\ &= 2T(2^{k_0}) + c2^{k_0+1} \\ &= 2(d + ck_0)2^{k_0} + c2^{k_0+1} \\ &= (d + ck_0)2^{k_0+1} + c2^{k_0+1} \\ &= (d + c(k_0 + 1))2^{k_0+1} \\ &= RHS \end{aligned}$$

Thus, $T(2^k) = (d + ck)2^k$ holds for all $k \geq 0$. $\square$

Question 3.3

The watershed function for all questions is $n^{\log_4 2} = n^{\frac{1}{2}}$.

1.

Since $f(n) = 1$, we have $f(n) = O(n^{\log_b a - \epsilon}) = O(n^{\frac{1}{2} - \epsilon})$ for all ϵ that satisfies $0 < \epsilon \leq \frac{1}{2}$. Hence, $T(n) = \Theta(n^{\frac{1}{2}})$

2.

Since $f(n) = \sqrt{n} = n^{\frac{1}{2}}$, we have $f(n) = \Theta(n^{\log_b a} \log^k n) = \Theta(n^{\frac{1}{2}} \log^0 n) = \Theta(n^{\frac{1}{2}})$ when $k = 0$. Hence, $T(n) = \Theta(n^{\frac{1}{2}} \log n)$

3.

Since $f(n) = \sqrt{n} \log^2 n = n^{\frac{1}{2}} \log^2 n$, we have $f(n) = \Theta(n^{\log_b a} \log^k n) = \Theta(n^{\frac{1}{2}} \log^2 n)$ when $k = 2$. Hence, $T(n) = \Theta(n^{\frac{1}{2}} \log^3 n)$

4.

Since $f(n) = n$, we have $f(n) = \Omega(n^{\log_b a + \epsilon}) = \Omega(n^{\frac{1}{2} + \epsilon})$ for all ϵ that satisfies $0 < \epsilon \leq \frac{1}{2}$. Also, $af(n/b) = 2f(\frac{n}{4}) = \frac{n}{2} \leq cf(n) = cn$ holds for all c that satisfies $\frac{1}{2} \leq c < 1$. Hence, $T(n) = \Theta(f(n)) = \Theta(n)$

Question 3.4

$\textsc{BinarySearch}(A, x, \text{low}, \text{high})$	runtime(for each iteration)
1. **if** low $>$ high **then**	$\Theta(1)$
2. **return** false	$\Theta(1)$
3. mid $= \lfloor(\text{low} + \text{high})/2\rfloor$	$\Theta(1)$
4. **if** $A[\text{mid}] = x$ **then**	$\Theta(1)$
5. **return** true	$\Theta(1)$
6. **else if** $A[\text{mid}] > x$ **then**	$\Theta(1)$
7. **return** $\textsc{BinarySearch}(A, x, \text{low}, \text{mid} - 1)$	$T(n/2)$
8. **else**	$\Theta(1)$
9. **return** $\textsc{BinarySearch}(A, x, \text{mid} + 1, \text{high})$	$T(n/2)$

The recurrence equation is $T(n) = T(n/2) + \Theta(1)$.
Using Master Theorem, we have $a = 1$, $b = 2$, $f(n) = \Theta(1)$ and $n^{\log_b a} = n^{\log_2 1} = n^0 = 1$. Since that $f(n) = \Theta(n^{\log_b a} \log^k n) = \Theta(1 \cdot \log^0 n) = \Theta(1)$ when $k = 0$, we have $T(n) = \Theta(n^{\log_b a} \log^{k+1} n) = \Theta(\log n)$.