HW#2 solution

1 Show that the solution to $T(n) = 2T\left(\left\lfloor\frac{n}{2}\right\rfloor + 17\right) + n$ is $O(n \lg n)$.

This problem can be solved by *domain transformation*.

Let $S(n) = T(n+\alpha)$ (i.e. transform the domain n to $n+\alpha$)

where α is a unknown constant, chosen so that $S(n)$ satisfies

$S(n) \le 2S(n/2) + O(n)$

that can be solved directly by the master theorem.

First of all, write

$$T(n) = 2T\left(\left\lfloor\frac{n}{2}\right\rfloor + 17\right) + n \le 2T\left(\frac{n}{2} + 17\right) + n$$

From these, we obtain

$$T(n+\alpha) = S(n) \le S(n/2) + O(n) = T(n/2+\alpha) + O(n) \qquad \cdots (1)$$

and

$$T(n+\alpha) \le 2T\left(\frac{n+\alpha}{2} + 17\right) + n + \alpha = 2T\left(\frac{n+\alpha}{2} + 17\right) + O(n) \quad \cdots (2)$$

From (1) and (2), we have

$$n/2 + \alpha = \frac{n+\alpha}{2} + 17 \Rightarrow \alpha = 34$$

Since $S(n) = O(n \log n)$ by the master theorem, it follows that

$$\begin{aligned}
T(n) &= S(n-34) \\
&= O\big((n-34)\lg(n-34)\big) \\
&= O(n-34)O(\lg(n-34)) \\
&= O(n)O(\lg n) \qquad \because \lg(n-34) \le \lg n \\
&= O(n \lg n)
\end{aligned}$$

2 We can try $T(n) \le cn^{\log_3 4}$, then we will get following :

$$T(n) \le 4\left(c\left(\frac{n}{3}\right)^{\log_3 4}\right) + n$$
$$= 4c\left(\frac{n^{\log_3 4}}{3^{\log_3 4}}\right) + n$$
$$= 4c\left(\frac{n^{\log_3 4}}{4}\right) + n$$
$$= cn^{\log_3 4} + n$$

The $T(n)$ is greater than $cn^{\log_3 4}$. It is fail. So we try to subtract off a lower-order term and assume that $T(n) \le cn^{\log_3 4} - dn$. Then we can get following :

$$T(n) \le 4\left(c\left(\frac{n}{3}\right)^{\log_3 4} - \frac{dn}{3}\right) + n$$
$$= 4\left(\frac{cn^{\log_3 4}}{4} - \frac{dn}{3}\right) + n$$
$$= cn^{\log_3 4} - \frac{4dn}{3} + n$$

We want $T(n) = cn^{\log_3 4} - dn \le cn^{\log_3 4} - \frac{4dn}{3} + n$, so that it need $d \ge 3$

So $T(n)$ will less than or equal to $cn^{\log_3 4} - dn$ if $d \ge 3$

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

This is a divide-and-conquer recurrence with a = 2, b = 2 and *f(n)* = n^4.
Thus $n^{log_b a} = n^{log_2 2} = n$.
Since $n^4 = \Omega\left(n^{\log_2 2 + 3}\right)$, and $a/b^i = 2/2^4 = 1/8 < 1$, this problem can be solved by the **case 3** of the master theorem.
$\Rightarrow T(n) = \theta(f(n)) = \theta(n^4)$

b) $T(n) = T(7n/10) + n$:

This is a divide-and-conquer recurrence with a = 1, b = 10/7 and *f(n)* = n.
Thus $n^{log_b a} = n^{log_{10/7} 1} = n^0 = 1$.
Since $n = \Omega\left(n^{\log_{10/7} 1 + 1}\right)$, and $a/b^i = 1/(10/7)^1 = 1/8 < 1$, this problem can be solved by the **case 3** of the master theorem.
$\Rightarrow T(n) = \theta(f(n)) = \theta(n)$

c) $T(n) = 16T(n/4) + n^2$:

This is a divide-and-conquer recurrence with a = 16, b = 4 and *f(n)* = n^2.

Thus $n^{log_b a} = n^{log_4 16} = n^2$.

Since $n^2 = \theta(n^{log_4 16})$, this problem can be solved by the **case 2** of the master theorem. (*k*=0)

$\Rightarrow T(n) = \theta(f(n)\lg n) = \theta(n^2 \lg n)$

d) $T(n) = 7T(n/3) + n^2$:

This is a divide-and-conquer recurrence with a = 7, b = 3 and *f(n)* = n^2.

Since $n^2 = \Omega\left(n^{\log_b a + \varepsilon}\right) = \Omega\left(n^{\log_3 7 + \varepsilon}\right)$ for any $0 < \varepsilon \leq 2 - \log_3 7$, we can solve this by the case 3 of the master theorem.

$\Rightarrow T(n) = \theta(f(n)) = \theta(n^2)$

e) $T(n) = 7T(n/2) + n^2$:

This is a divide-and-conquer recurrence with a = 7, b = 2 and *f(n)* = n^2.

Since $n^2 = O\left(n^{\log_b a - \varepsilon}\right) = O\left(n^{\log_2 7 - \varepsilon}\right)$ for any $0 < \varepsilon \leq \log_2 7 - 2$, we can solve this by the case 1 of the master theorem.

$\Rightarrow T(n) = \theta(n^{log_b a}) = \theta(n^{log_2 7})$

f) $T(n) = 2T(n/4) + \sqrt{n}$:

This is a divide-and-conquer recurrence with a = 2, b = 4 and *f(n)* = $\sqrt{n}$.

Since $\sqrt{n} = \theta\left(n^{\log_b a} lg^k n\right) = \theta(n^{\log_4 2})$ (with k = 0), we can solve this by the case 1 of the master theorem

$$\Rightarrow T(n) = \theta\left(n^{log_b a}\, lg^{k+1} n\right) = \theta\left(n^{\frac{1}{2}} \lg n\right)$$

4 a) $T(n) = T\left(n/2 + \sqrt{n}\right) + n$

Let $S(n) = S(n/2) + n$

$U(n) = U(2n/3) + n$

Then, by the master theorem, $S(n) = \Theta(n)$ and $U(n) = \Theta(n)$

Since $S(n) \leq T(n) \leq U(n)$ for n large enough, we have $T(n) = \Theta(n)$.

b) $T(n) = T(n/2) + T\left(\sqrt{n}\right) + n$

Since $\sqrt{n}$ is much smaller than $n/2$, it is reasonable to ignore it and guess that $T(n) = \Theta(n)$.

Clearly , $T(n) = \Omega(n)$ and we need only show that $T(n) = O(n)$

Assume that $T(n) \leq cn$

Then,

$$\begin{aligned} T(n) &= T(n/2) + T(\sqrt{n}) + n \\ &\le cn/2 + c\sqrt{n} + n \\ &= cn - (cn/2 - c\sqrt{n} - n) \\ &\le cn \end{aligned}$$

as long as

$cn/2 - c\sqrt{n} - n \ge 0$

Clearly, the inequality holds for $c > 2$ and sufficiently large n.

For example, pick $c = 4$.

The inequality holds for $n - 4\sqrt{n} \ge 0 \Rightarrow n \ge 16$

5 a) Key observation

If $A[j] < j$, then $A[i] \ne i$ for all $i \le j \Rightarrow$ search $A[j+1..n]$

If $A[j] > j$, then $A[i] \ne i$ for all $i \ge j \Rightarrow$ search $A[1..j-1]$

Thus, we may use binary search,

SEARCH(A, l, h)

if $(l > h)$ **return** -1

$m = \lfloor (l+h)/2 \rfloor$

if $(A[m] == m)$ **return** m

if $(A[m] < m)$ **return** SEARCH$(A, l, m-1)$

else **return** SEARCH$(A, m+1, h)$

b) Let $T(n)$ be the worst-case running time of SEARCH.

Then,

$$T(n) = \begin{cases} \Theta(1), & n = 0 \\ T\left(\left\lceil \frac{n-1}{2} \right\rceil\right) + \Theta(1), & n > 0 \end{cases}$$

c) We may simplify the recurrence to

$T(n) = T(n/2) + \Theta(1)$

Then, by case 2 of the master theorem, $T(n) = \Theta(\lg n)$.

Alternative solution

We may stick on the original recurrence and use *domain transformation* to solve it. (See the solution to the first problem of this homework.)

6 a) Show that the running time of MERGE$(A[1..n])$ is $O(n \lg \sqrt{n})$.

Step 2 takes a time in $O(\lg \sqrt{n})$.

Step 4 takes a time in $O(n)$.

Analysis of step 3

There are $O(n)$ EXTRACT-MIN and INSERT operations, each taking $O(\lg|Q|)$

running time. Since $|Q| \leq \sqrt{n}$, step 3 takes a time in
$\mathrm{O}(n) \times O(\lg \sqrt{n}) = O(n \lg \sqrt{n})$
Thus, the total time is
$O(\lg \sqrt{n}) + O(n) + O(n \lg \sqrt{n}) = O(n \lg \sqrt{n})$

b) $T(n) = \sqrt{n}T(\sqrt{n}) + O(n \lg \sqrt{n})$
We shall solve this recurrence in two steps.
Step 1: Range transformation
Let $S(n) = T(n)/n$
Then,

$$T(n) = \sqrt{n}T(\sqrt{n}) + O(n \lg \sqrt{n}) \Rightarrow T(n)/n = T(\sqrt{n})/\sqrt{n} + O(\lg \sqrt{n})$$
$$\Rightarrow S(n) = S(\sqrt{n}) + O(\lg \sqrt{n})$$

Step 2: Change of variable
Next, let $U(m) = S(2^m)$ (i.e. rename $m = \lg n$)
Then,
$U(m) = S(2^m) = S(2^{m/2}) + O(\lg 2^{m/2}) = U(m/2) + O(m)$
By the master theorem, $U(m) = O(m)$

Finally, we have
$T(n) = nS(n) = nU(\lg n) = n \times O(\lg n) = O(n \lg n)$