Week – 1 Graded Homework (Solutions)

Solution 1:

(a+b), ad, bc.

Solution 2:

- Yes.

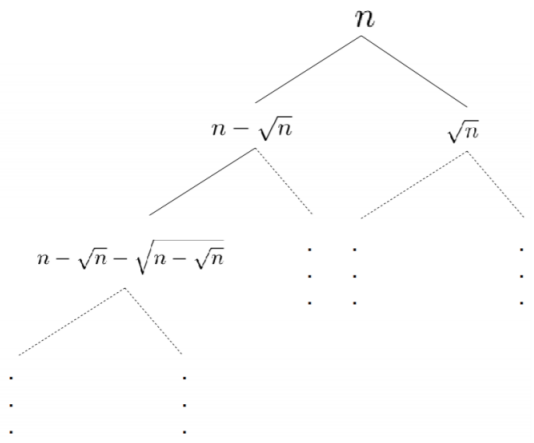
Recurrence relation: $S(n) = S(\frac{3n}{4}) + S(\frac{n}{4}) + cn$, for $n > 1$; $S(1) = 1$.
Proof by substitution method.

- Guess: $S(n) = O(n \log n)$.
- Induction goal: $S(n) < dn \log(n) + d'$, for some constants $d, d' > 0$.
- Base case: $n = 1$, then $S(1) = 1 < d'$, if $d' \geq 1$.
- Induction hypothesis: $S(\frac{3n}{4}) < d(\frac{3n}{4}) \log(\frac{3n}{4}) + d'$ and $S(\frac{n}{4}) < d(\frac{n}{4}) \log(\frac{n}{4}) + d'$.
- Proof of Induction step: $S(n) = S(\frac{3n}{4}) + S(\frac{n}{4}) + cn$
  $\rightarrow S(n) < d(\frac{3n}{4}) \log(\frac{3n}{4}) + d' + d(\frac{n}{4}) \log(\frac{n}{4}) + d' + cn$
  $\rightarrow S(n) < d(\frac{3n}{4})(\log(n) - \log(\frac{4}{3})) + d(\frac{n}{4})(\log n - \log 4) + +2d' + cn$
  $\rightarrow S(n) < dn \log(n) - d(\frac{3n}{4}) \log(\frac{4}{3}) - d(\frac{n}{4}) \log 4 + cn + 2d'$
  $\rightarrow S(n) < dn \log(n) - n(d(\frac{3}{4} \log(4/3) + \frac{\log(4)}{4}) - c - \frac{2d'}{n})$
  $\rightarrow S(n) < dn \log(n)$ if $d(\frac{3}{4} log(\frac{4}{3}) + \frac{\log(4)}{4}) > c + 2d'$: Hence we can pick $d' = 1$ and $d > c + 2$ and the proof follows.

- No.

To justify this answer, all we need is to show a lower bound on the running time for this split that is greater than $O(n \log n)$.

Lower bound:



The figure above shows the recursion tree where each node represents the number of points passed at the corresponding recursive function call. The function call on the leftmost branch at level i must have $\geq n - i\sqrt{n}$. We have that $n - i\sqrt{n} \geq \frac{n}{2}$ for $i \leq \frac{\sqrt{n}}{2}$, and hence that any call at level $i \leq \frac{\sqrt{n}}{2}$ on the leftmost branch has $\geq \frac{n}{2}$ points. Then, at each of the topmost $\frac{\sqrt{n}}{2}$ levels on the leftmost branch, the function call takes at least $\frac{cn}{2}$ amount of work. This also implies that the total amount of work for all the topmost $\frac{\sqrt{n}}{2}$ calls is $\geq \frac{cn}{2} \times \frac{\sqrt{n}}{2} = \Theta(n^{3/2})$. Since we have only considered work on part of the leftmost branch so far, the total time complexity for all the recursive calls made by the algorithm must be $\Omega(n^{3/2})$.

For completeness, we show that the lower bound above is tight (you did not need to show this part to get full marks for this question):

Upper bound:
Recurrence relation: $S(n) = S(\sqrt{n} + S(n - \sqrt(n)) + cn$, if $n > 1$; $S(n) = 1$, if $n \leq 1$
Proof by substitution method.

- Guess: $S(n) = O(n^{3/2})$.
- Induction goal: $S(n) < dn^{3/2} + d'$ for some constants $d, d' > 0$.
- Base Case: $n = 1$, then $S(1) = 1 < d'$, if $d' \geq 1$.
- Induction hypothesis: $S(\sqrt{n}) < dn^{3/4} + d'$ and $S(n - \sqrt{n}) < d(n - \sqrt{n})^{3/2} + d'$.
- Proof of Induction step: $S(n) = S(\sqrt{n}) + S(n - \sqrt{n}) + cn$.
  $\rightarrow S(n) < dn^{3/4} + d(n - \sqrt{n})^{3/2} + 2d' + cn$
  $\rightarrow S(n) < dn^{3/2}$, if $d > (2(c + 2)/3)$ and $d' = 1$.

Solution 3:

Say A and B are the two databases and $A(i)$, $B(i)$ are i^{th} smallest elements of A, B.

First, let us compare the medians of the two databases. Let k be $\lceil \frac{1}{2}n \rceil$, then $A(k)$ and $B(k)$ are the medians of the two databases. Suppose $A(k) < B(k)$ (the case when $A(k) > B(k)$ would be the same with interchange of the role of A and B). Then one can see that $B(k)$ is greater than the first k elements of A. Also $B(k)$ is always greater than the first $k-1$ elements of B. Therefore $B(k)$ is at least $2k^{th}$ element in the combine databases. Since $2k \geq n$, all elements that are greater than $B(k)$ are greater than the median and we can eliminate the second part of the B database. Let B' be the half of B (i.e., the first k elements of B).

Similarly, the first $\lfloor \frac{1}{2}n \rfloor$ elements of A are less than $B(k)$, and thus, are less than the last $n-k+1$ elements of B. Also they are less than the last $\lceil \frac{1}{2}n \rceil$ elements of A. So, they are less than at least $n-k+1+\lceil \frac{1}{2}n \rceil = n+1$ elements of the combine database. It means that they are less than the median and we can eliminate them as well. Let A' be the remaining parts of A (i.e., the $\left[\lfloor \frac{1}{2}n \rfloor + 1; n\right]$ segment of A).

Now we eliminate $\lfloor \frac{1}{2}n \rfloor$ elements that are less than the median, and the same number of elements that are greater than median. It is clear that the median of the remaining elements is the same as the median of the original set of elements. We can find a median in the remaining set using recursion for A' and B'. Note that we can't delete elements from the databases. However, we can access i^{th} smallest elements of A' and B': the i^{th} smallest elements of A' is $i + \lfloor \frac{1}{2}n \rfloor^{th}$ smallest elements of A, and the i^{th} smallest elements of B' is i^{th} smallest elements of B.

Formally, the algorithm is the following. We write recursive function `median(`n`,`a`,`b`)` that takes integers n, a and b and find the median of the union of the two segments $A[a+1; a+n]$ and $B[b+1; b+n]$.

```
median(n, a, b)
   if n=1 then return min(A(a+k), B(b+k)) // base case
   k=⌈½n⌉
   if A(a+k)<B(b+k)
     then return median (k, a+⌊½n⌋ ,b)
     else return median (k, a, b+⌊½n⌋)
```

To find median in the whole set of elements we evaluate `median(`n`,0,0)`.

Let $Q(n)$ be the number of queries asked by our algorithm to evaluate `median(`n`,`a`,`b`)`. Then it is clear that $Q(n) = Q(\lceil \frac{1}{2}n \rceil) + 2$. Therefore $Q(n) = 2\lceil \log n \rceil$.

A final note. In order to prove this algorithm correct, note that it is not enough to prove simply that, in the recursive call, the median remains in the set of numbers considered; one must prove the stronger statement that the median value in the recursive call will in fact be the same as the median value in the original call. Also, the algorithm cannot invoke the recursive call by simply saying, "Delete half of each database." The only way in which the algorithm can interact with the database is to pass queries to it; and so a conceptual "deletion" must in fact be implemented by keeping track of a particular interval under consideration in each database.