CSC317:Data Structures and Algorithm Design

Homework 1

Aaron Jesus Valdes

February 22, 2021

Problem 1

Part a

For this problem, we simply need to follow section 6.5 of the book where they describe how to create a max-heapify and instead use it to develop a min-heapify.
First we need to develop a heap from the array, then we need to called the heap-extract-min(A)

Algorithm 1 heap-extract-min(A)

$min \leftarrow A[1]$
$A[1] \leftarrow A[heap - size[A]]$
$heap - size[A] \leftarrow heap - size[A] - 1$
$min - Heapify(A, 1)$
return min

Algorithm 2 min-Heapify(A,i)

$l \leftarrow LEFT(i)$ ▷ Setting left child node A[i]
$r \leftarrow RIGHT(i)$ ▷ Setting right child node A[i]
if $l \leq |A|$ **and** $A[l] < A[i]$ **then** ▷ Checking for the smallest element
    $min \leftarrow l$
else min $\leftarrow i$
if $r \leq |A|$ **and** $A[r] < A[min]$ **then**
    $min \leftarrow r$
if min $\neq i$ **then** ▷ Updating the min-heap tree
    exchange $A[i] \leftrightarrow A[min]$
    min-Heapify(A,min)

Part b

Building the heap takes O(n) while heap-extract-min takes O(log(n)) therefore the total time complexity is O(n+klog(n)) as previously mentioned. Therefore given the fact that the upper bound for build a heap is O(n) and for extracting the min-value is O(k log(n)) then therefore the combination of both worst-case scenario must be O(n+klog(n))

Problem 2

Algorithm 3 Modified Quicksort using Select and Partition from Section 9.3

QUICKSORT(A,p,r)
if $p > r$ **then**
    $i \leftarrow \left\lfloor \frac{r-p+1}{2} \right\rfloor$
    $x \leftarrow SELECT(A, p, r, i)$ ▷ Finds the smallest value from the array
    $q \leftarrow$ SELECT-PARTITION(A, p, r, x)
    $QUICKSORT(A, p, q - 1)$
    $QUICKSORT(A, q + 1, r)$

Algorithm 4 SELECT-PARTITION(A,p,r,x)

```
i ← x
exchange A[r] ↔ A[i]
return PARTITION(A,p,r)
```

Algorithm 5 PARTITION(A,p,r)

```
x ← A[r]
i ← p − 1
for j ← p to r − 1 do
    if A[j] ≤ x then
        i ← i + 1
        exchange A[i] ↔ A[j]
exchange A[i + 1] ↔ A[r]
return i + 1
```

Problem 3

Solving the recurrence using the substitution method therefore we must guess the form of our solution.
Claim: The recurrence $T(n) = 2T(n/2) + c_1n + c_2$ has solution $T(n) \leq c_3nlog(n)$
Inductive Step:

$$
\begin{aligned}
T(n) =& 2T(n/2) + c_1n + c_2 && (1)\\
T(n) \leq& 2[c_3(\frac{n}{2})(log(\frac{n}{2}))] + c_1n + c_2 && (2)\\
=& c_3nlog(\frac{n}{2}) + c_1n + c_2 && (3)\\
=& c_3n(log(n) - log(2)) + c_1n + c_2 && (4)\\
=& c_3nlog(n) + c_1n - c_3n + c_2 && (5)\\
=& c_3nlog(n) - (-c_1n + c_3n - c_2) && (6)\\
& && (7)
\end{aligned}
$$

Now we want the last term to be $\leq c_3nlog(n)$ so $-c_1n + c_3n - c_2 \leq 0$

$$
\begin{aligned}
-c_1n + c_3n - c_2 \leq& 0 && (8)\\
c_3n \leq& c_1n + c_2 && (9)\\
c_3 \leq& \frac{c_1n + c_2}{n} && (10)\\
c_3 \leq& c_1 + \frac{c_2}{n} && (11)
\end{aligned}
$$

Therefore as long as c_3 is a less than $\frac{c_2}{n} + c_1$ our claim is true
Now we need to find n_0

$$c_3 \leq c_1 + \frac{c_2}{n_0} \tag{12}$$

$$c_3 - c_1 \leq \frac{c_2}{n_0} \tag{13}$$

$$n_0 \leq \frac{c_2}{c_3 - c_1} \tag{14}$$

$$\tag{15}$$

Therefore n_0 must be greater than $\frac{c_2}{c_3 - c_1}$