COMP582 Midterm Exam Hsuan-You(Shaun) Lin Student ID: S01435165

<Quick Answers>

1. For the union/find problem of size N, what is the worst-case complexity for any single union operation when using weighted union but no path compression? Note: the cost of initialization does not enter in to the calculation
My Answer: O(log(n))

2. For the union/find problem of size N, what is the worst-case complexity for any single union operation when using both weighted union and path compression?
Note: the cost of initialization does not enter in to the calculation
My Answer: O(log*n)

3. For the union/find problem of size N, what is the worst-case complexity for a series of O(N) union/find operations when using weighted union and path compression?
My Answer: O(n log*n)

4. What is the worst-case complexity for selection sort n items?
My Answer: O(n^2)

5. What is the best-case complexity for selection sort of n items?
My Answer: O(n^2)

6. What is the worst-case complexity for insertion sort n items?
My Answer: O(n^2)

7. What is the best-case complexity for insertion sort of n items?
My Answer: O(n)

8. What is the worst-case complexity of quicksort of n items?
My Answer: O(n^2)

9. What is the average-case for quicksort of n items?
My Answer: O(n log(n))

10. What is the worst-case for merge sort of n items?
My Answer: O(n log(n))

11. You are comparing 2 algorithms A, and B. You have exact running times
$T_A(n) = 22 \log(n!) + 5n + 2205$, and
$T_B(n) = 75n \log(n) + 70000n + 200 \log(n) + 100000$.
You are interested in a Big Oh comparison. Is $T_A(n) = O(T_B(n))$, or is
$T_B(n) = O(T_A(n))$ or are these 2 complexities essentially the same in the Big Oh sense?
My Answer: $T_A(n) = O(T_B(n))$, also $T_B(n) = O(T_A(n))$

12. What is the worst-case complexity for quickselect to find the 1st quartile (the n/4-smallest) item in a set of n items?
My Answer: O(n^2)

13. What is the average-case complexity for quickselect to find the 1st quartile (the n/4-smallest) item in a set of n items?
My Answer: O(n)

14. Suppose you are looking for an item with key k in a set of n unsorted items. What is the worst-case complexity to find the item (assuming it is in the set)?
My Answer: O(n)

15. In a 0-based heap array of size N, where N > 2, what is the highest index of a non-leaf element?
My Answer: n-2/2 flow

16. For Shellsort, let the h-gap sequence be {1, 4, 9}. Let the input sequence be
-27, 9, 0, -4, 12, 17, -100, -8, 42, 6404, 4
Show the sequence after the h = 9 gap is completed
My Answer: -27, 4, 0, -4, 12, 17, -100, -8, 42, 6404, 9

17. Suppose you are looking for an item with key k in a set of n sorted items. What is the worst-case complexity to determine the item is not in the set?
My Answer: O(log(n))

18. What is the worst-case complexity to delete the minimum element from a min- heap of size n?
My Answer: O(log(n)

19. Given a number f and a nonnegative integer L, what is the worst-case complexity to compute f ^L?
My Answer: O(log(n)

20. You are comparing the worst-case complexities of 2 algorithms A and B. The worst-case time for A is $T_A(n) = 4000000n + 3504n \log(n) + 100000$. The worst-case time for B is $T_B(n) = n2 + 2n + 4$ Is $O(T_A) \leq O(T_B)$, or is $O(T_B) \leq O(T_A)$, or both?
My Answer: $O(T_A) \leq O(T_B)$

<Not-so-Quick Answers>

1. My Answer: No, it's not a algorithm. Because there is no break point in the program, and we don't know that is halts and when will finally satisfy the condition.

2. My Answer: $O(n^3 \log^3(n))$
My reason:

$a = 3, b = 2, k = 3, p = 3$

$a < b^k, p \geq 0$

$\therefore T(n) = O(n^k \log^p(n)) = O(n^3 \log^3(n))$ #

3.

3.1

My pseudocode:

I didn't complete this part during the exam.

3.2

My Answer: $O(n \log(n))$

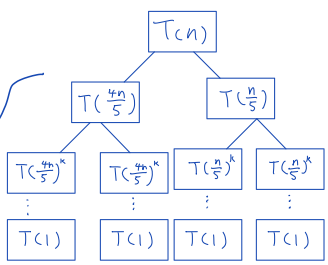
My reason:

$T(n) = T(\frac{n}{5}) + T(\frac{4n}{5}) + O(n) + O(1)$

$T(\frac{n}{5}) = T(\frac{4n}{25}) + T(\frac{n}{25})$

$T(\frac{4n}{5}) = T(\frac{16n}{25}) + T(\frac{4n}{5})$

tree height k



$T(n) = T(\frac{n}{5}) + T(\frac{4n}{5}) + O(n) + O(1)$

$= T(\frac{4n}{25}) + T(\frac{n}{25}) + T(\frac{16n}{25}) + T(\frac{4n}{5}) + O(n) + O(1)$

$= \ldots$ (As shown on Tree figure, the tree height is k, $k = \log_{\frac{4}{5}} n$, so complexity is $O(\log(n)) + O(n)$)

$= O(n \log(n))$ #

3.3

My Answer: $O(n \log(n))$

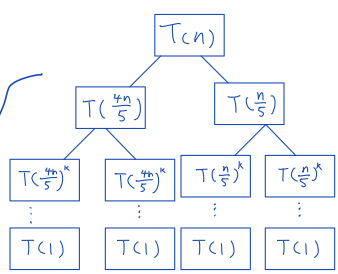
My reason:

$T(n) = T(\frac{n}{5}) + T(\frac{4n}{5}) + O(n) + O(\log n)$

$T(\frac{n}{5}) = T(\frac{4n}{25}) + T(\frac{n}{25})$

$T(\frac{4n}{5}) = T(\frac{16n}{25}) + T(\frac{4n}{5})$

tree height k



$T(n) = T(\frac{n}{5}) + T(\frac{4n}{5}) + O(n) + O(\log n)$

$= T(\frac{4n}{25}) + T(\frac{n}{25}) + T(\frac{16n}{25}) + T(\frac{4n}{5}) + O(n) + O(\log n)$

$= \ldots$ (As shown on Tree figure, the tree height is k, $k = \log_{\frac{4}{5}} n$, so complexity is $O(\log(n)) + O(n)$)

$= O(n \log(n))$ #

4.

My Answer: $O(n^2)$

My reason:

$$\Theta n^2 = cn^2$$

$$T(n) = T\left(\frac{3n}{4}\right) + T\left(\frac{n}{4}\right) + cn^2$$

$$= T\left(\frac{9n}{16}\right) + 2\cdot T\left(\frac{3n}{16}\right) + T\left(\frac{n}{16}\right) + cn^2 + \frac{9cn^2}{16} + \frac{3cn}{16} + \frac{10cn^2}{16}$$

$$= \left(T\left(\frac{27n}{64}\right) + T\left(\frac{9n}{64}\right) + c\left(\frac{9n}{16}\right)^2\right) + 2\cdot\left(T\left(\frac{9n}{64}\right) + T\left(\frac{3n}{64}\right) + c\left(\frac{3n}{16}\right)^2\right)$$

$$+ \left(T\left(\frac{3n}{64}\right) + T\left(\frac{n}{64}\right) + c\left(\frac{n}{16}\right)^2\right) + cn^2 + \frac{10cn^2}{16}$$

$$= \sum_{i=0}^{k}\left(C_i^k \cdot T\left(\frac{3^i n}{4^k}\right) + \left(\frac{5}{8}\right)^i \cdot cn^2\right) = \left(\frac{8}{3}\right)\cdot cn^2 = O(n^2) \#$$

5.

My Answer: O(L log(M))

My pseudocode:

```python
# Hsuan-You Lin
# Midterm exam: Question 5
import heapq

def Largest_M(arr, m):
    root = []
    for i in range(len(arr)):
        heapq.heappush(root, arr[i])
        if len(root) > m:
            heapq.heappop(root)
    return root

if __name__ == "__main__":
    arr = [3, 6, 1, 8, 10, 14, 21, 42, 61, 52]
    print(Largest_M(arr, 5))

#        14
#       /   \
#     21     52
#    / \
#   61  42
```

outer loop complexity is $O(L)$

inner loop complexity is $O(\log(M))$

∴ Overall complexity is $O(L \log(M))$ #

```
Midterm_Exam — -bash — 80×2
(base) pisces:Midterm_Exam pisces$ python Q5.py
[14, 21, 52, 61, 42]
(base) pisces:Midterm_Exam pisces$
```

6.
6.1
My Answer: The largest number of total operations is 42
My reason:

$S_4 = bmerge(S_2, S_3) \rightarrow 4+7=11$
$S_5 = bmerge(S_1, S_4) \rightarrow 3+11=14$
$S_6 = bmerge(S_0, S_5) \rightarrow 3+14=17$
$\therefore S_4+S_5+S_6 = 11+14+17 = \underline{42}$ #

6.2
My Answer: The smallest number of total operations is 33
My reason:

$S_4 = bmerge(S_0, S_1) \rightarrow 3+3=6$
$S_5 = bmerge(S_2, S_4) \rightarrow 4+6=10$
$S_6 = bmerge(S_3, S_5) \rightarrow 7+10=17$
$\therefore S_4+S_5+S_6 = 6+10+17 = \underline{33}$ #

6.3
My Answer: O(n log(n))
My pseudocode:

```python
# Hsuan-You Lin
# Midterm exam: Question 6.3
import heapq

def bmerge_min(arr_sizes):
    heap = []
    for i in range(len(arr_sizes)):
        heapq.heappush(heap, (arr[i], i))

    L = len(arr_sizes)
    while len(heap) > 1:
        number_1, index_1 = heapq.heappop(heap)
        number_2, index_2 = heapq.heappop(heap)
        print(f"{L} = bmerge({index_1}, {index_2})")
        heapq.heappush(heap, (number_1 + number_2, L))
        L += 1

if __name__ == "__main__":
    arr = [0, 1, 2]
    bmerge_min(arr)
```

→ complexity is $O(\log n)$

→ complexity is $O(n-1) = O(n)$

∴ Overall complexity is $\underline{O(n \log n)}$ #

```
Midterm_Exam — -bash — 80×24
(base) pisces:Midterm_Exam pisces$ python Q6.py
3 = bmerge(0, 1)
4 = bmerge(3, 2)
(base) pisces:Midterm_Exam pisces$
```

7.

My Answer: The maximum of L's complexity is O(n), the 2nd largest element of L's complexity is O(1)

My pseudocode:

```python
# Hsuan-You Lin
# Midterm exam: Question 7

def Find_Two_Largest(arr):
    max_1 = 0
    max_2 = 0
    for i in range(len(arr)):
        if arr[i] > max_1:
            max_1, max_2 = arr[i], max_1
        elif arr[i] > max_2:
            max_2 = arr[i]
    return max_1, max_2

if __name__ == "__main__":
    arr = [0, 1, 2, 3, 3, 5, 7, 10, 21]
    L1, L2 = Find_Two_Largest(arr)
    print(L1, L2)
```

```
Midterm_Exam — -bash — 80×24
(base) pisces:Midterm_Exam pisces$ python Q7.py
21 10
(base) pisces:Midterm_Exam pisces$
```

"On my honor, I have neither given nor received any unauthorized aid on this exam.
Start Time: Oct. 15, 2022, P.M 14:30
End Time: Oct. 15, 2022, P.M 18:30