Exercise Sheet 2
COMS10007 Algorithms 2018/2019

19.02.2019

Reminder: $\log n$ denotes the binary logarithm, i.e., $\log n = \log_2 n$.

1 Proofs by Induction

Prove the following statements by induction:

1. For every integer $n \geq 0$, the following holds:

$$\sum_{i=0}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} \ .$$

2. For every $n \geq 1$, the following holds:

$$11^n - 6 \text{ is divisible by } 5.$$

3. Consider the following sequence: $s_1 = 1, s_2 = 2, s_3 = 3$, and $s_n = s_{n-1} + s_{n-2} + s_{n-3}$, for every $n \geq 4$. Prove that the following holds:

$$s_n \leq 2^n \ .$$

2 Bubblesort

Bubblesort is a popular, but inefficient, sorting algorithm. It works by repeatedly swapping adjacent elements that are out of order:

Algorithm 1 BUBBLESORT

Require: Array A of n integers

```
for i = 0 to n − 2 do
    for j = n − 1 downto i + 1 do
        if A[j] < A[j − 1] then
            exchange A[j] with A[j − 1]
        end if
    end for
end for
```

1. What is the worst-case runtime of BUBBLESORT?

2. Consider the loop in lines $2-6$. Prove that the following invariant holds at the beginning of the loop:

$$A[j] \leq A[k], \text{ for every } k \geq j \ .$$

Give a suitable termination property of the loop.

3. Consider now the loop in lines $1-7$. Prove that the following invariant holds at the beginning of the loop:

The subarray $A[0, i]$ is sorted.

Give a suitable termination property that shows that A is sorted upon termination.

3 More on Big-O

Give formal proofs of the following statements using the definition of Big-O from the lecture.

1. $20n \in O(\frac{1}{4}n^2)$.

2. Suppose that $f(n) \in O(g(n))$. Show that: $2^{f(n)} \in O(2^{g(n)})$.

Proof. This statement is wrong and cannot be proved!
For example, consider the functions $f(n) = \log n$ and $g(n) = \frac{1}{2}\log n$. Then:

$$2^{f(n)} = 2^{\log n} = n \text{ and } 2^{g(n)} = 2^{\frac{1}{2}\log n} = n^{\frac{1}{2}} = \sqrt{n} \ .$$

Observe that $n \notin O(\sqrt{n})$. The statement is thus false. □

3. Rank the following functions by order of growth: (no proof needed)

$$(\sqrt{2})^{\log n}, n^2, n!, (\log n)!, (\frac{3}{2})^n, n^3, \log^2 n, \log(n!), 2^{2^n}, n \log n$$

Hint: Stirling's approximation for the factorial function can be helpful:

$$e(\frac{n}{e})^n \leq n! \leq en(\frac{n}{e})^n$$

4 Heap Sort

Consider the following array A:

4	3	9	10	14	8	7	2	1	7

1. Interpret A as a binary tree as in the lecture (on heaps).

2. Heapify() the tree. Give the sequence of node exchanges. Draw the resulting heap.

3. What is the worst-case runtime of Heapify() on a binary tree of n elements?

4. Explain how heap sort uses the heap for sorting. Explain why the algorithm has a worst-case runtime of $O(n \log n)$.

5. Give an array of length n so that heap-sort runs in $O(n)$ time on A.

5 Merge Sort

This is an open-ended exercise without a single correct solution. This exercise has the potential to lead to further discussions.

Recall the merge operation in Merge Sort: Given is an array A so that the left $\lfloor n/2 \rfloor$ elements and the right $\lceil n/2 \rceil$ elements are sorted in increasing order. The task is to merge the two halves so that the entire array is sorted. The merge operation as discussed in the lecture does not merge in place. Can you think of a merge operation that merges in place? What is the runtime of your suggested merge operation? What is the runtime of merge sort when your merge operation is applied?