Exercise Sheet 1
COMS10018 Algorithms 2024/2025

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

Example Question: Big-O Notation

Question. Give a formal proof of the following statement using the definition of Big-O from the lecture (i.e., identify positive constants c, n_0 for which the definition holds):

$$5\sqrt{n} \in O(n) \ .$$

Solution. We need to show that there are positive constants c, n_0 such that $5\sqrt{n} \le c \cdot n$ holds, for every $n \ge n_0$. This is equivalent to showing that $(\frac{5}{c})^2 \le n$ holds.
We choose $c = 5$, which implies $1 \le n$. We can thus select $n_0 = 1$, since then $1 \le n$ holds for every $n \ge n_0$. This prove that $5\sqrt{n} \in O(n)$.

Remark: Observe that there are many other combinations of values for c and n_0 that satisfy the inequality we need to prove. For example, if we pick $c = 1$ then we obtain $25 \le n$ (which follows from $(\frac{5}{c})^2 \le n$). In this case, we would have to choose a value for n_0 that is greater or equal to 25, in particular, $n_0 = 25$ would do. ✓

1 O-notation: Part I

Give formal proofs of the following statements using the definition of Big-O from the lecture (i.e., identify positive constants c, n_0 for which the definition holds):

1. $n^2 + 10n + 8 \in O(\frac{1}{2}n^2)$.
2. $n^3 + n^2 + n = O(n^3)$.
3. $10 \in O(1)$.
4. $\sum_{i=1}^{n} i \in O(4n^2)$.

2 Racetrack Principle

Use the racetrack principle to prove the following statement:

$$n \le e^n \text{ holds for every } n \ge 1 \ .$$

3 O-notation: Part II

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

1. $f \in O(h_1), g \in O(h_2)$ then $f \cdot g \in O(h_1 \cdot h_2)$.
2. $2^n \in O(n!)$.
3. $2^{\sqrt{\log n}} \in O(n)$.

4 Fast Peak Finding

Consider the following variant of FAST-PEAK-FINDING where the "$\geq$" sign in the condition in instruction 4 is replaced by a "$<$" sign:

1. **if** A is of length 1 **then return** 0
2. **if** A is of length 2 **then** compare $A[0]$ and $A[1]$ and **return** position of larger element
3. **if** $A[\lfloor n/2 \rfloor]$ is a peak **then return** $\lfloor n/2 \rfloor$
4. Otherwise, **if** $A[\lfloor n/2 \rfloor - 1] < A[\lfloor n/2 \rfloor]$ **then**
   return FAST-PEAK-FINDING($A[0, \lfloor n/2 \rfloor - 1]$)
5. **else**
   return $\lfloor n/2 \rfloor + 1+$ FAST-PEAK-FINDING($A[\lfloor n/2 \rfloor + 1, n-1]$)

Give an input array of length 8 on which this algorithm fails.

5 Optional and Difficult

Exercises in this section are intentionally more difficult and are there to challenge yourself.

5.1 Advanced Racetrack Principle

Use the racetrack principle and determine a value n_0 such that

$$\frac{2}{\log n} \leq \frac{1}{\log \log n} \text{ holds for every } n \geq n_0 \ .$$

Hint: Transform the inequality and eliminate the log-function from one side of the inequality before applying the racetrack principle. If needed, apply the racetrack principle twice!
Recall that $(\log n)' = \frac{1}{n \ln(2)}$. The inequality $\ln(2) \geq 1/2$ may also be useful.

5.2 Finding Two Peaks

We are given an integer array A of length n that has exactly two peaks. The goal is to find both peaks. We could do this as follows: Simply go through the array with a loop and check every array element. This strategy has a runtime of $O(n)$ (requires $c \cdot n$ array accesses, for some constant c). Is there a faster algorithm for this problem (e.g. similar to FAST-PEAK-FINDING)? If yes, give such an algorithm. If no, justify why there is no such algorithm.