Tutorial 11

Exercise 1: Reminder on Big-O notation 1

Which of these definitions of the O-notation are correct?

1. $f = \mathcal{O}(g)$ if and only if
there exist positive integers c and n_0 such that for all $n \geq n_0$ we have that $f(n) \geq c \cdot g(n)$
2. $f = \mathcal{O}(g)$ if and only if
for all positive integers c and n_0 it is the case that for all $n \geq n_0$ we have that $f(n) \leq c \cdot g(n)$
3. $f = \mathcal{O}(g)$ if and only if
there exist positive integers c and n_0 such that for all $n \geq n_0$ we have that $f(n) \leq c \cdot g(n)$
4. $f = \mathcal{O}(g)$ if and only if
for all positive integers c and n_0 it is the case that there exists an $n \geq n_0$ such that $f(n) \leq c \cdot g(n)$

Solution:

1. WRONG: f should be bounded by g, i.e., $f(n) \leq c \cdot g(n)$.
2. WRONG: The requirement for all c and all n_0 is too strong.
3. RIGHT.
4. WRONG: Again, the quantifiers are flipped, which yields an incorrect definition.

Exercise 2: Reminder on Big-O notation 2

Which of the following claims are true? Give precise arguments for your answers.

1. $3n^2 + 2n + 7 = \mathcal{O}(n^2)$
2. $n^2 = \mathcal{O}(n \log n)$
3. $3^n = \mathcal{O}(2^n)$
4. $3^n = \mathcal{O}(2^{n^2})$ (Hint: $3 = 2^{\log_2 3}$)

Solution:

1. RIGHT: Choose $c = 12$ and $n_0 = 1$. Then clearly $3n^2 + 2n + 7 \leq 3n^2 + 2n^2 + 7n^2 = 12n^2$, and so $3n^2 + 2n + 7 \leq c \cdot n^2$ for all $n \geq n_0 = 1$.
2. WRONG: By contradiction. Assume that there are constants c and n_0 such that $n^2 \leq c \cdot n \log n$ for all $n \geq n_0$. This would mean that $n \leq c \cdot \log n$, and hence that $\frac{n}{\log n} \leq c$ for all $n \geq n_0$. However, this cannot be the case, since $\frac{n}{\log n}$ goes to ∞ as n goes to ∞, and hence the expression $\frac{n}{\log n}$ will eventually be larger than any chosen constant c.
3. WRONG: By contradiction. Assume that there are constants c and n_0 such that $3^n \leq c \cdot 2^n$ for all $n \geq n_0$. This would mean that hence $\frac{3^n}{2^n} = (\frac{3}{2})^n = (1.5)^n \leq c$ for all $n \geq n_0$. However, this cannot be the case, since $(1.5)^n$ goes to ∞ as n goes to ∞, and hence the expression $(1.5)^n$ will eventually be larger than any chosen constant c.
4. RIGHT: We have $3^n = (2^{\log_2 3})^n = 2^{n \log_2 3}$. Since $\log_2 3 \leq 2$, we have $n \log_2 3 \leq 2n \leq n^2$ for all $n \geq 2$. Thus, we can choose $c = 1$ and $n_0 = 2$, and so $3^n = 2^{n \log_2 3} \leq c \cdot 2^{n^2}$ for all $n \geq n_0 = 2$.

Exercise 3: Test your understanding

1. Which of the following statements about the class P are correct?

   (a) P is the class of all languages that are computable by single-tape DTMs running in polynomial time.

   (b) P is the class of all languages such that if $w \in$ P, then there is a single-tape DTM which accepts the word w in polynomial time.

   (c) P is the class of all languages that are computable by multi-tape DTMs running in polynomial time.

   (d) A language L belongs to P if and only if there is a constant k and a halting DTM M running within time $O(n^k)$ such that $L = L(M)$.

   (e) A language L belongs to P if and only if $L \in \textsc{Time}(2^n)$.

2. Give a language in P that we have not discussed in the course.

3. Does the halting problem HP belong to P?

4. Does the complement $\overline{\mathrm{HP}}$ belong to P?

Solution:

1. (a) RIGHT.

   (b) WRONG: The elements of P are languages, not words! Hence, the expression "$w \in$ P" does not make sense.

   (c) RIGHT. Recall that any multi-tape TM running in polynomial can be simulated by a single-tape TM also running in polynomial time (it is only quadratically slower).

   (d) RIGHT.

   (e) WRONG: The implication from left to right holds, but the one from right to left does not hold.

2. The following languages (for example) belong to P:

   - $\emptyset$
   - every regular language
   - every context-free language
   - $\{a^k b^k c^k \mid k \geq 0\}$
   - $\{G \mid G$ is a connected graph $\}$
   - $\{\ulcorner M \urcorner \mid M$ is a TM that has more than 10 states $\}$

3. The language HP does not belong to P because the language HP is not computable and P contains only computable languages (as it is defined as a class of languages accepted by halting DTMs).

4. The complement $\overline{\mathrm{HP}}$ does not belong to P (for the same reason as HP).

Exercise 4: NFA nonemptiness

Consider the following decision problem:

> "Does a given NFA $\mathcal{A}$ have a nonempty language?"

1. Define this problem as a language N_{NFA}.
2. Argue that N_{NFA} is in P.

Solution:

1. To define this problem as a language, we first have to think about how to encode NFAs as words over some fixed alphabet. Let $\mathcal{A} = (Q, \Sigma, s, T, \Delta)$ be an NFA with terminal/accepting states T and transitions $\Delta\colon Q \times \Sigma \to 2^Q$. (You may be used to a different notation!) Here, we proceed similarly to the encoding of DTMs.

   We assume without loss of generality that $Q = \{1, 11, \ldots, 1^n\}$ for some $n \geq 1$ with $s = 1$, and $\Sigma = \{1, 11, \ldots, 1^t\}$ for some $t \geq 1$. Furthermore, let $\Delta = \{\delta_0, \ldots, \delta_k\} = \{(q, a, q') \in Q \times \Sigma \times Q \mid q' \in \delta(q, a)\}$ be the set of transitions.

   Then, we encode $\mathcal{A}$ by the word

   $$\ulcorner \mathcal{A} \urcorner = 1^n 0 1^t 0 w_T 0 w_\Delta \in \{0,1\}^*$$

   where

   - $w_T \in \{0,1\}^n$ is the bit vector of length n such that w_T is 1 at position j if and only if $1^j \in T$ (i.e., the state 1^j is accepting), and
   - $w_\Delta = w_{\delta_0} 0 w_{\delta_1} 0 \cdots 0 w_{\delta_k}$ encodes the transitions such that for each $\delta_j = (q, a, q')$ we have $w_{\delta_j} = q0a0q'$.

   Then, we have

   $$N_{\text{NFA}} = \{\ulcorner \mathcal{A} \urcorner \mid \mathcal{A} \text{ is an NFA with nonempty language}\} \subseteq \{0,1\}^*.$$

2. We construct a multi-tape halting DTM for N_{NFA} with polynomial time complexity.

   On input $w \in \{0,1\}^*$:

   1. First check whether $w = \ulcorner \mathcal{A} \urcorner$ for some NFA $\mathcal{A}$. If not, reject.
   2. Otherwise, let n be the number of states encoded in $w = \ulcorner \mathcal{A} \urcorner$.
   3. Write the adjacency matrix of $\mathcal{A}$ on the tape, i.e., the matrix of the directed graph (V, E) with $V = \{1, \ldots, 1^n\}$ and $E = \{(1^i, 1^j) \in V \times V \mid (1^i, 1^k, 1^j) \text{ is a transition of } \mathcal{A} \text{ for some letter } 1^k\}$.
   4. For each accepting state 1^j check whether 1^j is reachable from $s = 1$ in (V, E). If yes, accept.
   5. Reject.

   Each step can be done in polynomial time (in particular the graph reachability, which was mentioned in the lecture). Hence, the overall running time is polynomial as well. Thus, N_{NFA} is in P.

Exercise 5: Challenge

Describe (in sufficient detail) a one-tape halting DTM for the language $\{0^m1^m \mid m \geq 1\}$ that runs within time $\mathcal{O}(n \log_2 n)$.

Solution:
Consider the following halting DTM:
On input w:

1. Scan the tape and reject if w is not of the form 0^*1^*.
2. Repeat lines 3 and 4 as long as at least one 0 and at least one 1 are on the tape.
3. If there is an odd number of cells containing either a 0 or a 1, then reject.
4. Otherwise, replace every second 0 by an X and every second 1 by an X.
5. If every 0 and every 1 is replaced by an X, then accept and otherwise reject.

Let us analyze the time complexity: Line 1 takes $\mathcal{O}(|w|)$ steps and is executed only once. Checking whether the tape contains at least one 0 and at least one 1 takes again $\mathcal{O}(|w|)$ steps. Let us say Line 2 is executed i times. We will later bound i.

Lines 3 and 4 also each take $\mathcal{O}(|w|)$ steps and are executed i times. Finally, Line 5 takes $\mathcal{O}(|w|)$ steps and is executed once.

So, if we show that $i \leq \log |w|$, then we are done. Note that in each iteration of Line 4, half of the 0's and half of the 1's on the tape are replaced by X's. An induction shows that after j executions, only $\frac{|w|}{2^j}$ many non-X symbols are left on the tape. Thus, after $\log |w|$ executions, none are left and the loop terminates. Hence, $i \leq \log |w|$ as required.