Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Ernst W. Mayr
Chris Pinkau

Spring Semester 2013
Problem Set 1
April 16, 2013

Complexity Theory

Due date: April 23, 2013 **before** *class!*

Problem 1 (10 Points)

Recall the definition of the Landau notation for $f, g : \mathbb{N} \to \mathbb{N}$:

$$\begin{aligned}
f = \mathcal{O}(g) &\quad :\Longleftrightarrow \quad \exists c > 0\, \exists n_0 \in \mathbb{N}\, \forall n \geq n_0 : f(n) \leq c \cdot g(n),\\
f = \Omega(g) &\quad :\Longleftrightarrow \quad g = \mathcal{O}(f)\\
f = \Theta(g) &\quad :\Longleftrightarrow \quad f = \mathcal{O}(g) \wedge f = \Omega(g),\\
f = o(g) &\quad :\Longleftrightarrow \quad \forall c > 0\, \exists n_0 \in \mathbb{N}\, \forall n \geq n_0 : f(n) \leq c \cdot g(n),\\
f = \omega(g) &\quad :\Longleftrightarrow \quad g = o(f).
\end{aligned}$$

Remark: Depending on the author, you will see the notations $f = \mathcal{O}(g)$ or $f \in \mathcal{O}(g)$, respectively. Both notations are tolerated, just be consistent with yours!

(a) For strictly positive functions f, g, i.e. $f(n), g(n) > 0$ for all $n \in \mathbb{N}$, show or disprove:

   (i) $f = \Theta(g)$ if and only if there exist $c_1, c_2 > 0$ such that $c_1 \leq \frac{f(n)}{g(n)} \leq c_2$ for almost all $n \in \mathbb{N}$. ("almost all" is equivalent to "except for finitely many").

   (ii) $f = o(g)$ if and only if $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$.

(b) Show that polynomial growth is dominated by exponential growth, i.e. for every $d > 0, b > 1$ it holds that $n^d = o(b^n)$.

(c) For each of the following pairs of functions f, g determine whether $f = o(g), g = o(f)$ or $f = \Theta(g)$.

   (i) $f(n) = n^2, \quad g(n) = 2n^2 + 100\sqrt{n}$,

   (ii) $f(n) = 1000n, \quad g(n) = n \log n$,

   (iii) $f(n) = 2^{2^{n+1}}, \quad g(n) = 2^{2^n}$,

   (iv) $f(n) = n^n, \quad g(n) = 2^{2^n}$.

Problem 2 (10 Points)

Prove that the following languages/decision problems on graphs are in $\mathcal{P}$: (You may pick either the adjacency matrix or adjacency list representation for graphs, it will not make a difference — can you see why?)

(a) Connected — the set of all connected graphs. That is, $G \in$ Connected if every pair of vertices u, v in G are connected by a path.

(b) Trianglefree — the set of all graphs that do not contain a triangle (i.e., a triplet u, v, w of pairwise connected distinct vertices).

(c) Bipartite — the set of all bipartite graphs. That is, $G \in$ Bipartite if the vertices of G can be partitioned into two sets A, B such that all edges in G are from a vertex in A to a vertex in B (there is no edge between two members of A or two members of B).

Problem 3 (10 Points)

(a) We are given a 1-tape Turing machine with alphabet $\Gamma = \{0, 1, \sqcup\}$, a set of states $Q = \{q_1, q_2, q_3, q_4\}$, and the transition function δ, defined by

$q \in Q$	$s \in \Gamma$	$\delta(q, s)$		
q_1	$\sqcup$	q_2	0	R
q_2	$\sqcup$	q_3	$\sqcup$	R
q_3	$\sqcup$	q_4	1	R
q_4	$\sqcup$	q_1	$\sqcup$	R

On every other possible input for δ, the machine does nothing in this step.
The TM is started with an empty tape (i.e., only $\sqcup$ symbols on it). What does this TM do?

(b) Give an example of a 1-tape Turing machine for identifying palindromes over $\{0, 1\}$. (A palindrom is a word that can be read the same way in either direction, i.e. Palindromes $= \{x \in \{0, 1\}^* : x = x^R\}$.)

(c) A Turing machine is called *oblivious* if the position of its heads at the i-th step of its computation on input x only depend on i and $|x|$, not on the input x itself.

Let L be a language that is decided by a Turing machine M in time $t(n)$. Show that there exists an oblivious Turing machine M' that decides L in time $\mathcal{O}(t(n) \log t(n))$.

Problem 4 (10 Points)

Consider a variant of the Knapsack problem: Given a set of natural numbers $A = \{a_1, \ldots, a_n\}$ and a natural number b, is there a subset $A' \subseteq A$ such that $\sum_{a \in A'} a = b$?

Show that in unary representation this problem can be solved in polynomial time. (Unary representation uses only one digit, 1. The representation of a natural number N is therefore 1 repeated N times.)