GROUP HOMEWORK 3, CPSC 421/501, FALL 2023

JOEL FRIEDMAN

Please note:

(1) You must justify all answers; no credit is given for a correct answer without justification.
(2) Proofs should be written out formally.
(3) You do not have to use LaTeX for homework, but **homework that is too difficult to read will not be graded.**
(4) You may work together on homework in groups of up to four, **but you must submit a single homework as a group submission under Gradescope.**

---

(1) Exercise 7.2.26(a,b,d,f) on the handout "Uncomputability in CPSC421/501." You may use any result proven in class, but nothing more.

(2) (a) Show that if $L \subset \Sigma^*_{\mathrm{ASCII}}$ is recognizable but undecidable, then the complement of L, i.e., $\Sigma^*_{\mathrm{ASCII}} \setminus L$, is unrecognizable. You may use any result proven in class, but nothing more.
(b) **Extra Credit:** Exercise 7.2.26(g) on the handout "Uncomputability in CPSC421/501." You may use any result proven in class, and part (a) above, but nothing more.

You may use the following identities in the problem below:

$$\binom{2}{2} + \binom{3}{2} + \cdots + \binom{n}{2} = \binom{n+1}{3},$$

(where $\binom{n}{k}$ is the binomial coefficient "n choose k," i.e. $n!/(k!(n-k)!)$), and

$$1 + 2^2 + 3^2 + \cdots + n^2 = \frac{n(n+1)(2n+1)}{6}.$$

Research supported in part by an NSERC grant.

(3) Let $i_1, i_2, \ldots$ be a sequence elements of $\Sigma^*_{\mathrm{ASCII}}$ such that each element of $\Sigma^*_{\mathrm{ASCII}}$ appears exactly once in this sequence.[1] Say that p is a Python program, and we want to know if p accepts at least one input. We can do this by the following algorithm:

Phase 1: simulate p for one step on input i_1;

Phase 2: simulate p for two steps on i_1 and one step on i_2;

Phase 3: simulate p for three steps on i_1, for two steps on i_2, and for one step on i_3;

etc.:

Phase k: on the k-th phase, for $j = 1, 2, \ldots, k$ we simulate p for $k-j+1$ steps on i_j;

Consider the total number of steps run in each phase; for example, Phase 3 has 6 steps total, and the total number of steps in Phases 1 to 3 is $1+3+6 = 10$. (Our convention is that when you simulate p on an input for some number of steps, you forget all previous simulations of p on any input.) Say that p accepts only one input, namely i_ℓ, and that p requires m program steps to do so.

(a) Show that the total number of steps until the above algorithm stops (i.e., when it detects that p accepts i_ℓ after m steps) is **exactly**
$$(1/6)(\ell+m)^3 + O(1)(\ell+m)^2,$$
where the $O(1)$ refers to an "order 1 term," i.e., a function of ℓ, m that is bounded by a constant for $\ell+m$ sufficiently large. By **exactly** we mean that $(1/6)(\ell+m)^3 + O(1)(\ell+m)^2$ is both a lower bound and an upper bound (for different values of $O(1)$).

(b) Say that we use the following variant: for all $k \in \mathbb{N}$, the k-the phase consists of simulating k steps of p on each of $i_1, \ldots, i_k$. Show that the total number of steps needed is **exactly**
$$(1/3)(\max(\ell,m))^3 + O(1)(\max(\ell,m))^2.$$

(c) Say that we use the following variant: for all $k \in \mathbb{N}$, the k-the phase consists of simulating $5k$ steps of p on each of $i_1, \ldots, i_{5k}$. Show that the total number of steps needed is **exactly** $c(\max(\ell,m))^3 + O(1)(\max(\ell,m))^2$ for some constant, c. What is c ?

(d) Using the previous part, for any constant $c > 0$, give a variant of the above algorithm that takes **at most** $c(\max(\ell,m))^3 + O(1)(\max(\ell,m))^2$ steps.

(4) **Extra Credit:** Continuing with the setup and notation as in the previous problem:

(a) Describe a variant of the above algorithm that uses no more than $O(1)(\max(\ell,m))^2$ steps.

(b) Prove that there is a constant $c > 0$ such that any such algorithm requires at least $c(\max(\ell,m))^2$ steps for $\max(\ell,m)$ sufficiently large, and give such a constant, c. [This implies that there is a $c > 0$ for

[1] In CPSC 421/501, we typically do this by listing the strings according to their length (and lexicographical order for strings of equal length), so that $i_1 = \epsilon$ (which is the single string of length 0), $i_2, \ldots, i_{129}$ are the elements of Σ_{ASCII}, $i_{130}, \ldots, i_{1+128+128^2}$ are the elements of $\Sigma^2_{\mathrm{ASCII}}$, etc.

which this holds for all $\ell, m \in \mathbb{N}$, but it is simpler to find a c that holds when $\max(\ell, m)$ is sufficiently large.]

Department of Computer Science, University of British Columbia, Vancouver, BC V6T 1Z4, CANADA.
E-mail address: jf@cs.ubc.ca
URL: http://www.cs.ubc.ca/~jf