CURSO DE COMPLEJIDAD COMPUTACIONAL 2021

Práctico 1

Temas: La Máquina de Turing (MT) como reconocedor de lenguajes y como computador de funciones. Variantes de la Máquina de Turing. La clase DTIME. Problemas de la clase P.

Para los ejercicios 1.1 a 1.5 emplear el modelo de MT de cinta única de Sipser. Para los ejercicios 1.6 a 1.11 emplear el modelo de MT multicinta de Arora-Barak.

1.1 Ejercicio 3.8 de Sipser.

Give implementation-level descriptions of Turing machines that decide the following languages over the alphabet {0,1}.

a. $\{w \mid w$ contains an equal number of 0s and 1s$\}$
b. $\{w \mid w$ contains twice as many 0s as 1s$\}$
c. $\{w \mid w$ does not contain twice as many 0s as 1s$\}$

1.2 Dar una descripción a nivel de implementación y una descripcion formal de la MT que decide el lenguaje $B = \{0^n 1^n 2^n \mid n \geq 0\}$.

1.3 Sea una función f : {1,0}* → 1* / f (w) = x siendo w la representación binaria de un entero n ≥ 1 (sin ceros no significativos), y x su representación unaria ("n" unos).

Ejemplos:

f(1)	1
f(101)	11111
f(1000)	11111111

Construya una MT que compute la función f.

Nota: en la configuración final de la MT, en la cinta sólo debe quedar la tira resultado x .

1.4 Ejercicio 3.12 Sipser

A ***Turing machine with left reset*** is similar to an ordinary Turing machine, but the transition function has the form

$$\delta: Q \times \Gamma \longrightarrow Q \times \Gamma \times \{\mathrm{R}, \mathrm{RESET}\}.$$

If $\delta(q, a) = (r, b, \mathrm{RESET})$, when the machine is in state q reading an a, the machine's head jumps to the left-hand end of the tape after it writes b on the tape and enters state r. Note that these machines do not have the usual ability to move the head one symbol left. Show that Turing machines with left reset recognize the class of Turing-recognizable languages.

1.5 Ejercicio 3.13 Sipser

A ***Turing machine with stay put instead of left*** is similar to an ordinary Turing machine, but the transition function has the form

$$\delta: Q \times \Gamma \longrightarrow Q \times \Gamma \times \{\mathrm{R}, \mathrm{S}\}.$$

At each point the machine can move its head right or let it stay in the same position. Show that this Turing machine variant is *not* equivalent to the usual version. What class of languages do these machines recognize?

1.6 Probar que la función $T(n) = n$ es time-constructible.

Asumir para la prueba que $n = 2^p \quad p \geq 0$

1.7 Ejercicio 1.5 Arora-Barak

Define a TM M to be *oblivious* if its head movements do not depend on the input but only on the input length. That is, M is oblivious if for every input $x \in \{0, 1\}^*$ and $i \in \mathbb{N}$, the location of each of M's heads at the ith step of execution on input x is only a function of $|x|$ and i. Show that for every time-constructible $T: \mathbb{N} \to \mathbb{N}$, if $L \in \mathbf{DTIME}(T(n))$, then there is an oblivious TM that decides L in time $O(T(n)^2)$. Furthermore, show that there is such a TM that uses only *two tapes*: one input tape and one work/output tape.

1.8 Ejercicio 1.7 Arora-Barak

Define a *two-dimensional* Turing machine to be a TM where each of its tapes is an infinite grid (and the machine can move not only Left and Right but also Up and Down). Show that for every (time-constructible) $T: \mathbb{N} \to \mathbb{N}$ and every Boolean function f, if f can be computed in time $T(n)$ using a two-dimensional TM then $f \in \mathbf{DTIME}(T(n)^2)$.

Considere que la malla infinita es una matriz de celdas $M(i, j)$ con $i, j \geq 0$ (la máquina no se mueve Left si $i = 0$ y no se mueve Down si $j = 0$, permaneciendo en la misma celda).

Asumir que la entrada de tamaño n se almacena en el rango de celdas de (0,0) a (n-1,0).

1.9 Ejercicio 1.9 Arora-Barak

Define a *RAM Turing machine* to be a Turing machine that has *random access memory*. We formalize this as follows: The machine has an infinite array A that is initialized to all blanks. It accesses this array as follows. One of the machine's work tapes is designated as the *address tape*. Also the machine has two special alphabet symbols denoted by R and W and an additional state we denote by q_{access}. Whenever the machine enters q_{access}, if its address tape contains $\llcorner i \lrcorner$R (where $\llcorner i \lrcorner$ denotes the binary representation of i) then the value $A[i]$ is written in the cell next to the R symbol. If its tape contains $\llcorner i \lrcorner$Wσ (where σ is some symbol in the machine's alphabet) then $A[i]$ is set to the value σ.

Show that if a Boolean function f is computable within time $T(n)$ (for some time-constructible T) by a RAM TM, then it is in **DTIME**$(T(n)^2)$.

1.10 Un procesador de Turing paralelo (PTP) es una red de $f(n)$ máquinas de Turing diferentes, cada una de los cuales tiene una cinta de trabajo que es pública. En un paso de tiempo de ejecución dado cada MT del PTP puede leer una celda de la cinta pública de cualquier otra MT del PTP, mediante la escritura del número de celda y del número de máquina a leer en una cinta de direcciones. Una MT se designa como máquina líder, es la que recibe la entrada al principio del cómputo y produce la salida al final del mismo. Se define **paralelo-P** como la clase de lenguajes X tales que dada una entrada de tamaño n, existe un PTP M con $f(n)$ procesadores y con una cota de tiempo de ejecución $t(n)$, siendo f y t polinomios, de modo que $L(M) = X$ (la clase de los lenguajes reconocidos por M es X).

Probar que **paralelo-P = P**.

Considerar que la operación de tiempo dominante en la simulación de las $f(n)$ máquinas de Turing del PTP con una MT convencional es la lectura de celdas en las cintas públicas del PTP.

1.11 Ejercicio 1.10 Arora-Barak

Consider the following simple programming language. It has a single infinite array A of elements in $\{0, 1, \square\}$ (initialized to $\square$) and a single integer variable i. A program in this language contains a sequence of lines of the following form:

label: If A[i] equals σ then *cmds*

where $\sigma \in \{0, 1, \square\}$ and *cmds* is a list of one or more of the following commands: (1) Set A[i] to τ where $\tau \in \{0, 1, \square\}$, (2) Goto *label*, (3) Increment i by one, (4) Decrement i by one, and (5) Output b and halt, where $b \in \{0, 1\}$. A program is executed on an input $x \in \{0, 1\}^n$ by placing the ith bit of x in A[i] and then running the program following the obvious semantics.

Prove that for every functions $f : \{0, 1\}^* \to \{0, 1\}$ and (time-constructible) $T : \mathbb{N} \to \mathbb{N}$, if f is computable in time $T(n)$ by a program in this language, then $f \in$ **DTIME**$(T(n))$.

1.12 Ejercicio 1.8 Arora-Barak

Let LOOKUP denote the following function: on input a pair $\langle x, i \rangle$ (where x is a binary string and i is a natural number), LOOKUP outputs the ith bit of x or 0 if $|x| < i$. Prove that LOOKUP $\in$ **P**.

1.13 Ejercicio 1.14 Arora-Barak

Prove that the following languages/decision problems on graphs are in **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 connected distinct vertices).

(c) BIPARTITE—The set of all bipartite graphs. That is, $G \in$ BIPARTITE if the vertices of G can be partitioned to 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).

(d) TREE—The set of all trees. A graph is a *tree* if it is connected and contains no cycles. Equivalently, a graph G is a tree if every two distinct vertices u, v in G are connected by exactly one simple path (a path is simple if it has no repeated vertices).