# Tarea 2: Complejidad Computacional

Profesores: Sergio Rajsbaum y Ricardo Strausz
Ayudante: Sebastián Bejos

28 de febrero de 2009

1. Diseñar una Máquina de Turing que al inicio tenga escrita una secuencia $w$ de 0s y 1s, y otra secuencia $u$ de $n$ 1s, donde $n \geq 0$. La entrada seria $\#w\#u\#$, es decir, tendríamos las dos secuencias $w$ y $u$ separadas por un símbolo $\#$, y delimitadas en ambos extremos por un símbolo $\#$. La $MT$ debe copiar $w$ a una posición, $n$ celdas después de la secuencia de entrada y dejarla encerrada por dos símbolos $\#$. Por ejemplo, si $n = 0$, la cinta al final debe tener $\#w\#u\#\#w\#$, si $n = 2$, $\#w\#u\#bb\#w\#$, donde $b$ es el símbolo blanco.

2. Muestra que para toda Máquina de Turing $S$ con $k$-cintas, existe una Máquina de Turing $T$ que remplaza a $S$, en el siguiente sentido: Para toda palabra $x \in \Sigma^*$, la máquina $S$ con entrada $x$ se detiene en un número finito de pasos, si y solo si $T$ con entrada $x$ se detiene y cuando paran, lo mismo esta escrito en la ultima cinta de $S$ que en la cinta de $T$. Muestra además que si $S$ realiza $n$ pasos, entonces $T$ hace $O(n^2)$ pasos.

3. Demuestra que existe una función $\varphi$ que determina la duración más larga, de un computo que se detiene, para una Máquina de Turing de tamaño específico. Es decir una función $\varphi(\mid Q \mid, \mid \Gamma \mid, n) = \tau$ ($Q$ es el conjunto de estados y $\Gamma$ es el conjunto de simbolos que utiliza la máquina), tal que si una $MT$ visita $n$ celdas y no ha terminado en tiempo $\tau$, esta máquina ya nunca terminará.

4. Con el modelo de cómputo de DNA visto en clase. Da un programa, que sume dos bits.

5. Programa la función *monus* con instrucciones del modelo de computo $RAM$. La función monus se define como sigue:

$$\dot{-}(x, y) = \begin{cases} x - y & \text{si } y < x \\ 0 & \text{si } y \geq x. \end{cases}$$

6. Demuestra que $REGULAR_{MT}$ es indecidible.

   $REGULAR_{MT} = \{< M >| \; M \quad \text{es una } MT \text{ y } L(M) \text{ es un lenguaje regular.}\}$ .

7. Leer el artículo *Who Can Name the Bigger Number?* Habrá un examen sorpresa sobre este artículo[1].

[1] La liga al artículo la puedes encontrar en http://atenea.matem.unam.mx/moodle dentro del curso de Teoría de la Complejidad Computacional