**Hoja 1 de ejercicios de TALF 2**

Entrega el 4 de noviembre de 2011

Máquinas de Turing

1. Diseñar una máquina de Turing que divide dos números "unarios" La cinta comienza con una secuencia 000D111…1B111…1d y acaba con una secuencia de 1s delante de la D que codifica el cociente y otra detrás de la d que codifica el resto. E.g. …000D11111B11d000… → …0011D11111B11d100… Representarla con un diagrama de estados.

2. Dada una cinta con distintas cadenas de 1s separadas por As (e.g. A111A1111A11A11111A). Diseñar una máquina de Turing que escriba al comienzo de la cinta la cadena más corta y luego pare. Modular la máquina de Turing como composición de submáquinas.

3. Sea una fila de *N* AFs soldados. Cada AF recibe una entrada de cada uno de sus dos AFs adyacentes. En un extremo, un AF "general" da la orden "FUEGO" al primer AF de la fila. Al cabo de unos instantes todos pasan simultáneamente al estado "FUEGO". Diseñar someramente el AF que valga para cualquier *N*.

   NOTA: se puede demostrar que el diseño para que el tiempo que tardan en disparar todos los AFs sea mínimo provoca un retraso de $2N$-1 lapsos de tiempo, pero la solución es muy compleja. Las soluciones más sencillas suelen estar entre $3N$ y $8N$

Problema de la parada

4. Explicar razonadamente si hay una máquina de Turing que a partir de una cadena de caracteres de la forma M:w, donde $w \in (a+b)^*$ y M es la codificación mediante letras de la c a la z de otra máquina de Turing con alfabeto $(a+b)^*$ haga lo siguiente:
   a. Si M se para al ejecutarse a partir de w en la cinta, la máquina que se pide se comporta durante su ejecución igual que M en cuanto a los caracteres que escribe sobre la cinta.
   b. Si M no se para al ejecutarse a partir de w en la cinta, la máquina que se pide no se para, y durante su ejecución escribe infinitos ceros sobre la cinta.
5. Suponemos que M es una máquina de Turing que tiene la propiedad de que si acepta la codificación de otra máquina de Turing arbitraria y una palabra, entonces la máquina de Turing codificada al ejecutarse sobre la palabra nunca se para. Construir una máquina de Turing que no se para sobre alguna palabra, pero la máquina M no es capaz de detectarlo.

Complejidad

6. Demostrar que

   $L_0 = \{M \mid M \textit{ no para sobre } M\}$

no es computable sabiendo que

$L_1 = \{M;N \mid M$ *no para sobre* $N\}$ no es computable

7. Demostrar que resolver un Sudoku es un problema NP

## Cálculo lambda

Teniendo en cuenta la siguiente codificación de los números naturales:

número n = λf.λx.f(f(…(f(fx))…)) (e.g. 0 = λf.λx.x ; 1 = λf. λx.fx)

*λf. λ x.* ← *n f's* →

y de los valores booleanos:

TRUE: λx.λy.x

FALSE: λx.λy.y

NOTA: Poner, en todos los ejercicios, todos los paréntesis para que quede claro y unívoco.

8. Definir la función lambda "xnor" y probarla para TRUE,FALSE y para TRUE,TRUE.

9. Hallar una expresión λ en forma normal equivalente a la expresión *(λx.x((λy.x)x))v*

10. Dar una función λ *f* tal que su punto fijo *Yf* tenga dominio vacío, es decir que para cualquier otra función λ *g, Yfg* no tengo forma normal.

## Funciones recursivas

11. Dando por supuesto que la suma de números naturales es una función recursiva primitiva y que también lo son las funciones *Px* cuyo valor es *x-1* salvo cuando *x* es nulo, en cuyo caso vale 0, y *si(x,y,z)*, cuyo valor es *y* si *x* no es nulo y *z* si *x* es nulo, demostrar que la función *f(x)* cuyo valor es la suma de los números pares menores o iguales que *x* también es recursiva primitiva.

12. Demostrar que la función que a cada cadena de caracteres w de (a+b)* le hace corresponder otra formada por N aes, donde N es el producto del número de aes de w por el de bes, es primitiva recursiva.

Optativos:

1. Definir la función lambda "esNulo" que devuelve la función TRUE al aplicarla al número 0 y la FALSE al aplicarla a cualquier otro.

2. Hallar una expresión λ en forma normal equivalente a la expresión *(λu.uu)(λx.λy.xy)v*. ¿Se puede simplificar más?

3. Escribir una función lambda de un argumento que, si el argumento es impar devuelve el número par inmediatamente superior, y si es par devuelve el número multiplicado por dos. Suponer que se cuenta con la función PAR que acepta un argumento y devuelve la función TRUE si es par y la función FALSE si es impar. [PRODUCTO = λf.λg.λx.(f((g)(x))) ]. Probarla para los valores 2 y 3.

4. Dando por supuesto que hay una expresión lambda, $e_\lambda$, que representa la potenciación ($x^y$), describir la forma en que se puede representar en el Cálculo Lambda la función
$$f(x, y) = (y=0) \ ? \ 1 : x^{f(x, y-1)}$$

5. Demostrar que la función $f(x,y)=x*y+x+y$ es recursiva primitiva.

6. Suponiendo que f(m, n) es una función recursiva primitiva, demostrar que también lo es la función g(x) cuyo valor es la suma de los valores de f(m, n) con $m \leq x$ y $n \leq x$.