

Complejidad Computacional

Tarea 2

Karla Adriana Esquivel Ramírez
Andrea Itzel González Vargas
Luis Pablo Mayo Vega
Carlos Gerardo Acosta Hernández

Entrega: 14/03/17
Facultad de Ciencias UNAM

Ejercicios

1. Supón que $L_1, L_2 \in \mathbf{NP}$. ¿Qué pasa con $L_1 \cup L_2$ y $L_1 \cap L_2$
- 2.
- 3.

Demuestra que el lenguaje

$SPACETM = \{ \langle M \rangle \langle \alpha \rangle 1^n \mid M \text{ es una MT que acepta } \alpha \text{ en espacio } n \}$
es PSPACE-completo.

P.D.

I) $SPACETM \in PSPACE$.

II) Cualquier $L \in PSPACE$ se puede reducir a SPACETM

Entrada $\langle M, \alpha, 1^n \rangle$

Construimos $M' \in TM$ para Simular M , M' con dos contadores.

I) Un contador para contar el espacio utilizado por M , el otro contador contará el número de pasos que va a ejecutar M . Cada vez después de que M' Simule un paso de M , actualiza los contadores y comprueba si el espacio utilizado es mayor que n o si las etapas que se ejecutan en M son mayores que 2^{cn} para alguna constante c que está relacionada con M .

Si cualquiera de las anteriores ocurre entonces se rechaza inmediatamente. Observemos que el número total de configuraciones posibles de M es como máximo 2^{cn} , por lo tanto, si M no se detiene en este número de pasos, entonces debe haber entrado en un bucle infinito y M' Rechaza, después de simular M corriendo en w , si M acepta w en espacio n , M' Acepta, en otro caso M' rechaza. Por definición de SPACETM, M' decide a SPACE. Para simular M solamente utilizamos espacio $O(n)$. Entonces $SPACETM \in PSPACE$

II) Ahora Observemos que para cualquier $L \in PSPACE$ donde L es un lenguaje existe una Máquina de Turing Determinista M que decide L usando espacio $s(n)$ el cual es una entrada polinomial de tamaño n . Consideremos la función $f: \{0, 1\}^n \Rightarrow \{0, 1\}^m$ tal que $f(x) = \langle M, x, 1^{s(n)} \rangle$. Donde m es polinomial de n . La función f puede ser computada en espacio y tiempo polinomial. Por definición de SPACETM $x \in L$ si y solo si $f(x) \in SPACETM$. Así que SPACETM es PSPACE-hard entonces $SPACETM \in PSPACE - completo$

$\therefore EsPSPACE - Completo.$

...

4.

Demuestra $2SAT \in NL$

Primero veamos que para todo F en 2-SAT podemos construir una grafica dirigida G de la siguiente manera:

$V(G)$ son las literales de F y hay una arista de a a b en $E(G)$ si y solo si $(a' \vee b)$ esta en F

Con esta construccion veamos que:

F es satisfacible si y solo si G no tiene un ciclo donde exista una literal y su negacion.

\Rightarrow)

veamos que $a' \vee b$ es equivalente a $a \rightarrow b$

Supongamos que tenemos un ciclo que contiene a b y b' en G . Si esto pasa entonces por transitividad significa que $b \rightarrow \dots \rightarrow b'$ y que $b' \rightarrow \dots \rightarrow b$ por lo que tendríamos $b \leftrightarrow b'$ lo cual es una contradiccion.

Por lo que si F es satisfacible entonces G no tiene un ciclo donde exista una literal y su negacion.

\Leftarrow)

Veamos estos haciend induccion sobre el numero de variables Cuando se tienen 0 variables

Si F no tiene variables entonces F es satisfacible y G no tiene ciclos.

Hip. Ind

Supongamos que se cumple cuando se tiene n variables

Ahora vemos que pasa cuando se tiene un variables mas:

Tomamos una literal b que en la grafica G generada no tiene un camino de b a b' .

Ahora supongamos que hay un camino de b a c y un camino de b a c'

Por contraposicion tendríamos un camino de c' a b' y por transitividad tendríamos un camino b a b' lo cual sera una contradiccion.

Ahora hagamos todas las literales a las que llegamos a partir de b verdaderas, esta conjunto esta bien definido por lo anterior visto. Entonces todas las clausulas que contenga a un literal de este conjunto se satisfacen. Y nos queda una grafica G' dada por las clausulas restantes las cuales tampoco tienen un ciclo con una literal y su negacion y por Hipotesis de induccion esta formula tambien es satisfacible.

∴ Si G no tiene ciclos donde existe una literal y su negacion F es satisfacible

En libro tenemos que NL= coNL veamos que 2-SAT' en NL implica 2-SAT en NL

Para ver si algo cumple con 2-SAT' basta con ver si hay una camino de alguna literal b a b' y de b' a b esto es que exista un ciclo en la grafica G generada. Por lo que podemos reducir el problema a PATH que de igual manera por lo visto en el libro PATH esta en NL.

∴ 2-SAT' esta en NL ∴ 2-SAT esta en NL

5.

6.

Anexo

Tarea 1.1 Ejercicio 1. En clase se presentó una descripción general de cómo codificar una máquina de Turing como una cadena binaria. Completa la descripción de tal forma que una máquina universal pueda utilizarla para imitar la máquina codificada

Comencemos con la codificación de la máquina de Turing como una cadena binaria.

Sea $M = (Q, \Sigma, \Gamma, \delta, s, r, t, \vdash, \sqcup)$ una máquina de Turing, con

- $Q = i$
- $\Sigma = j$
- $\Gamma = k$
- s el m-ésimo estado
- t el n-ésimo estado
- r el p-ésimo estado
- \vdash el q-ésimo caracter de Σ
- \sqcup el r-ésimo caracter de Σ

Dicha especificación será codificada por la siguiente cadena binaria:

$$0^i 10^j 10^k 10^m 10^n 10^p 10^q 10^r 1. \quad (1)$$

Continuamos con la codificación de una transición en delta de la forma $\delta(s_u, a_v) = (s_w, a_x, \rightarrow)$ como (2), mientras que una transición de la forma $\delta(s_u, a_v) = (s_w, a_x, \leftarrow)$ como (3):

$$0^u 10^v 10^w 10^x 101 \quad (2)$$

$$0^u 10^v 10^w 10^x 111 \quad (3)$$

De esta manera, podemos codificar completamente la tabla original de M .

Finalmente, una cadena de entrada en Γ^* de la forma $a_{n_1} \dots a_{n_z}$, quedará codificada por la cadena binaria:

$$0^{n_1} 10^{n_2} 1 \dots 0^{n_z} \quad (4)$$

Recordemos que la Máquina Universal de Turing U que simulará a M , consta de tres cintas, además de la de entrada y salida. La primera de ellas será utilizada para la descripción de M , es decir, la tabla que quedó codificada para cada transición δ como en (3); la segunda, contendrá lo que corresponde con la cinta original de M ; y la tercera el estado en el que la M se encontraría.

Al iniciar U , en la primera cinta ya debe tener la tabla de M codificada, mientras que en la tercera cinta aparecerá un 0, indicando que se encuentra en el estado inicial de la máquina original.

Separados por unos, podemos considerar una cadena binaria proveniente de una codificación de transición en δ , como una quinteta, por los 5 elementos en total que componen una transición. Para emular a M , la Máquina Universal de Turing U , deberá hacer lo siguiente por cada transición de la TM original:

1. Localiza en la primera cinta aquella quinteta en la que el primer componente sea igual al que se encuentra en la tercera cinta y el segundo componente sea igual al que se encuentra en la segunda cinta.
2. Copia a la tercera cinta el tercer componente de la primera cinta (el estado al que se transfiere).
3. Sustituye la segunda cinta con lo que tiene el cuarto componente.
4. Interpreta el movimiento de la cabeza que se encuentra en el quinto componente, moviendo acorde la cabeza de la segunda cinta.