

Si hay cambios raras a lo largo del TP es pg' es la 1ª vez g' uso una tablero gráfico !!

Práctica 8

Ej 1:

1. Un lenguaje \mathcal{L} es esparso si existe un polinomio p tal que $|\mathcal{L} \cap \{0, 1\}^n| \leq p(n)$ para todo $n \in \mathbb{N}$.
Probar que todo lenguaje esparso está en P/poly.

Consultar, una vez me dijo q' era así para no estaba segura.

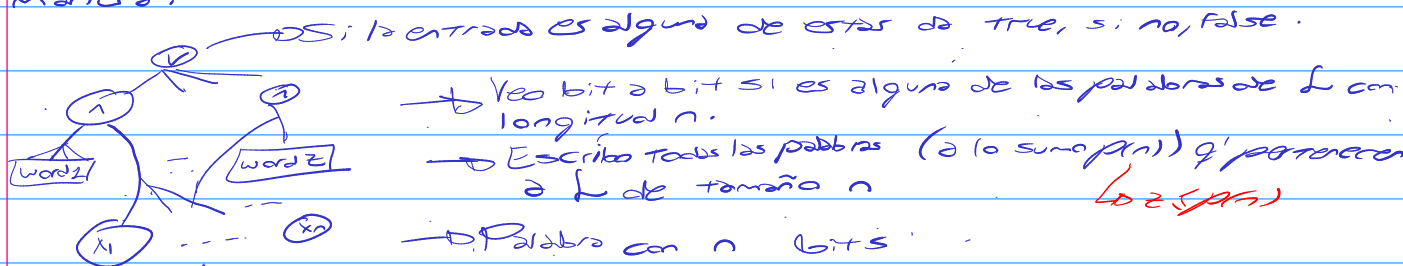
Idea: Para probar esto podría hacer una familia de circuitos los cuales tengan cierta estructura la cual me permita demostrar que para una entrada x con $|x|=n$, $C_n = 1$ si $x \in \mathcal{L}$ con \mathcal{L} esparso.

Voy a reescribir la def. de esparso para entenderlo mejor:

$\exists p$. es un polinomio tq' $|\mathcal{L} \cap \{0, 1\}^n| \leq p(n)$ para todo $n \in \mathbb{N}$.

o sea, hay una cantidad polinomial de palabras de longitud n

Declaro entonces una fila de circuitos de la siguiente manera:



Estos circuitos tienen un tamaño polinomial, ya q' depende la cantidad de palabras de longitud n q' hayan en \mathcal{L} (el cual ya sabemos q' es $p(n)$).

Dep. verif el exccpoly (creo q' sería $n \cdot p(n)$).

son poly conjunciones (ver todas las palabras de longitud n de \mathcal{L})

Luego poly disyunciones (veo q' la entrada coincide con alguna palabra de \mathcal{L} con tamaño n .)

Creería q' esta es suficiente prueba de q' existe una familia de circuitos $(C_n)_{n \in \mathbb{N}}$ donde se compute todo \mathcal{L}

★ Es más visual con advice (desp. relleno) (con \mathcal{L} esparso) !!

Ej 2:

2. Probar que existen lenguajes fuera de P/poly.

Idea: Sale por diagonalización. La cantidad de lenguajes en P/poly es enumerable y sabemos q' la cant. de lenguajes es no numerable.

Más que nada debemos justificar q' la cant. de circuitos ó máq. poly con consejo es enumerable, esto se ve en q' se tiene una codificación para cada máq., por lo cual, cada una es codificable finitamente y por ende enumerable.

Como no se puede enumerar lo no numerable necesariamente existen lenguajes q' no estén en P/poly.

Ej 3:

3. Definimos la clase P_{advice} como la clase de lenguajes que se pueden resolver en tiempo polinomial asumiendo que se cuenta con un consejo a para cada tamaño n de tamaño polinomial en n . Es decir, $\Pi \in P_{\text{advice}}$ si y solamente si existe una función $\text{adv} : \mathbb{N} \rightarrow \{0,1\}^*$ y una máquina polinomial M tal que

$$x \in \Pi \iff M(x, \text{adv}(|x|)) = 1$$

Ver q' el adv. es el mismo para toda cadena de = longitud.

donde aparte existe un polinomio p con $|\text{adv}(n)| \leq p(n)$ (es decir, el consejo es chico).

Probar que $P_{\text{advice}} = P/\text{poly}$.

Idea: Para \subseteq se puede ver medio fácil por q' P_{advice} tiene una máq. q' genera circuitos básicamente y habría q' ver q' estos son de altura poly.

Para \supseteq usa q' $\text{adv}(n) := C_n$ y luego q' la máq. simule C_n dado el input x (donde $|x| = n$), o sea $\text{adv}(n)$

$$\subseteq) P_{\text{advice}} \subseteq P/\text{poly}$$

$M :=$ Móg. det. polinomial

$\text{adv}() :=$ Función de $\mathbb{N} \rightarrow \{0,1\}^*$ y $|\text{adv}(n)| \leq p(n) \cdot \forall n \in \mathbb{N}$.

$$x \in \Pi \text{ sii } M(x, \text{adv}(|x|)) = 1$$

Como $|\text{adv}(|x|)| \leq p(|x|)$ puedo insertar este advice como constante en el circuito C_m (con $|x|=m$). Esta práctica es válida pq' agrandaría a lo sumo polinomialmente el circuito, pues su tamaño es $\leq p(n)$. Además como la familia $(C_n)_{n \in \mathbb{N}}$ no tiene q' ser generable

$M(n) \leq C_n$ con M det. \leftarrow uniformemente puedo incorporar info dependiente de n .

Luego, el funcionamiento de este C_n sería el mismo q' el de la M polinomial mencionada antes.

Ahora, ¿cómo sé que la simulación de esta M/poly resulta en un circuito poly ?

Bueno, esta demo es más difícil de lo anticipado, así q' lo voy a justificar 'con esto: Como sé q' $P \subseteq P/\text{poly}$ (ver Teo 1.1) puedo decir q' cada móg. q' corre en tiempo poly está en P/poly , ya q' puedo interpretar q' cada móg. det. poly . con salida $\in \{1, 0\}$ decida un $L \in P$ (esto sale por def. de P básicamente).

✓

$$\supseteq) P/\text{poly} \subseteq P_{\text{advice}}$$

$(C_n)_{n \in \mathbb{N}} :=$ Familia de circuitos de tamaño polinomial.

$$x \in \Pi \text{ sii } C_{|x|}(x) = 1$$

Puedo crear a $\text{adv}(n)$ como la codificación de C_n , de manera q' la móg. M simule $C_{|x|}(x)$ polinomialmente.

¿Por qué y cómo lo hace poly ?

M corre polinomialmente porque es evaluar un circuito.
O sea, se ve de la siguiente manera la máquina

$M \langle x, \langle C, |x| \rangle \rangle$:

Evalúa C_n con entrada $x \rightarrow$ Esto es poly pq' es simular las compuertas lógicas codificadas en C_n . Esto sería \approx prox. $\leq n \cdot p(n)$ compuertas a evaluar.

Como el advice cambia dependiendo del tamaño de la entrada no hay problema en cambiar la configuración de circuito q' se esté usando dependiendo de $|x|$.

Ej 4:

4. Definimos $P/f(n)$ como la clase de problemas que se resuelven con un consejo de tamaño $f(n)$ (y entonces $P/\text{poly} = \bigcup_{k \in \mathbb{N}} P/n^k$). Probar que $P \neq P/1 \cap R$.

Primero lo primero:

• ¿Qué es R ? R hace referencia a los problemas recursive, o sea \subset todas los problemas computables.

Idea: A ver, q'vq $P \neq (P/1 \cap R)$, o sea, tendría sentido q' $P \subseteq P/1 \cap R$, ya q' tendría sentido q' agregarle 1 bit de advice no le saque poder de decisión (sería muy raro sino). Entonces, el problema debe ser q' $(P/1 \cap R) \not\subseteq P$.

Entonces, por qué $(P/1 \cap R) \not\subseteq P$? Debe haber un Π tal que $\Pi \in P/1 \cap R$ y $\Pi \notin P$, esto lo debemos ver por diagonalización.

Q'vq $(P/1 \cap R) \not\subseteq P$ por diagonalización:

Enumeramos M_1, M_2, \dots a todas las mág. det. polinomiales

Construimos $\Pi \in P/1 \cap R$ tal que:

- Por cada $n \in \mathbb{N}$ ($n = |x|$) difiere del lenguaje aceptado por M_n en algún x de tamaño n usando el bit de consejo.

Visual:

	M_1	M_2	M_3	M_4	...
Tamaño de entrada	1	$S_{M_1}^1$	$S_{M_2}^1$...	
	2	$S_{M_1}^2$			
	3				
	4				

$S_{M_n}^i = \{ \text{C/ta de salidas de todas las posibles entradas de } M_n \text{ de tamaño } n \}$

○ —○ Niega algún elemento de $S_{M_n}^i$ y lo paga de advice (Medio q' así estoy asumiendo)

Esto sucede así por lo q' todas las máq. det. poly son de problemas vista en la teoría/práctico de de decisión)

q' cualquier problema de decisión puede ser de salida normal y viceversa en tiempo poly

EjS:

5. Probar que $NP = P$ si y solamente si $NP \subseteq P/\log(n)$.

Idea: Bueno acá va a haber q' probar la doble implicación (se viene largo)

$\Rightarrow NP = P \Rightarrow NP \subseteq P/\log(n)$

Si $NP = P$, entonces puedo probar q' $P \subseteq P/\log(n)$ y me basta, esto sale sencillamente porque si $L \in P$ y M decide L , puedo tener una M' q' le entre un advice q' no haga nada y luego q' la máquina funcione igual a como lo hacía M entonces M' decide L , y cumple con la def. de lenguaje en $P/\log(n)$:

$$x \in L \text{ si: } M'(x, \text{adv}(|x|))$$

$$|\text{adv}(n)| \leq \log(n)$$

$\Leftarrow NP \subseteq P/\log(n) \Rightarrow NP = P$

Como no lo uso puede ser cualquier cosa \emptyset de tamaño menor o igual a $\log(n)$

Significa q' NP es resoluble por una máq. M det. poly con un consejo $\text{adv}(n)$ de tamaño $\log n$ (Lademo no la hice, me cuesta verlo Preguntar)

Ej 6:

6. Probar que si $NP \not\subseteq P/poly$ entonces $NP \neq P$.

Sé q' $P \subseteq P/poly$ por lo cual si $NP \not\subseteq P/poly$, lógicamente $NP \neq P$.

Aunq' es eso, me da miedo q' la rta. lleve 2 renglones.

✓

Ej 7: (tengo miedo, cifre la explicación aparte a este 0.0)
Lo edito y con razón...

7. Probar que si $EXP \subseteq P/poly$ entonces $\Sigma_2^P = EXP$.

Ayuda: Probar que si $\Pi \in EXP$ y M es una máquina exponencial con Q estados que lo resuelve en $c2^{n^k}$ pasos entonces el lenguaje $\Pi_M = \{ \langle x, i, t, p, q \rangle : i, t, p \leq c2^{|x|^k}, q \leq Q, \text{ y en el timestep } t \text{ el } i\text{-ésimo bit de la memoria de } M \text{ es } 1, \text{ el puntero está en la posición } p \text{ y la máquina está en el estado } q \} \text{ está en } EXP$. Usar el \exists para adivinar el circuito que resuelve Π_M , y luego el \forall para verificar que es el correcto.

$$\Sigma_2^P = \exists u_1 \forall u_2. M(x, u_1, u_2)$$

Ver después (el Teo 29 (Rarp-Lipton) creo q' ayuda a la idea)

Ej 8:

8. Probar que si $PSPACE \subseteq P/poly$ entonces $PSPACE = \Sigma_2^P \cap \Pi_2^P$.

Es similar a la prueba anterior ^^.

Ej 9:

9. Probar que los lenguajes

- $AND = \{x_1 \dots x_n : \forall 1 \leq i \leq n, x_i = 1\}$
- $OR = \{x_1 \dots x_n : \exists 1 \leq i \leq n, x_i = 1\}$

están en NC^1 . Usar esto para probar que $AC^d \subseteq NC^{d+1}$ para todo $d \geq 0$.

Recordar:

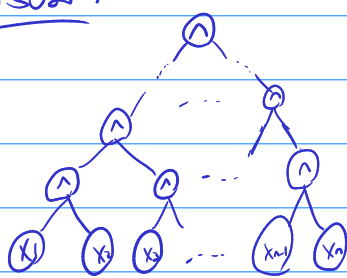
$$NC^d \subseteq AC^d \subseteq NC^{d+1}$$

NC^2 : Lenguajes decidibles por una familia de circuitos L -uniforme $(C_n)_{n \in \mathbb{N}}$ tq' $|C_n|$ es poly en n y $\text{prof}(C_n) = O(\log n)$

Idea: Para probar esto debería hacer las familias de circuitos q' representan estos lenguajes.

$$\bullet \text{ AND} = \{x_1, \dots, x_n : \forall 1 \leq i \leq n, x_i = 1\}$$

La familia de circuitos L -Uniformes puede ser la q' haga lo siguiente
Visual:



$O(\log n)$ (Es ir dividiendo el problema a 2 hasta q' quede 1 (para casos impares se puede intuir una sol. sencilla))

Entonces lo que haría (asumo q' $n = 2^c$ pero funcionaría con menores modificaciones para el resto de casos) sería q' cada 2 nodos del nivel $i-1$ q' no tienen flechas de salida los conecta con un 1 y continuo hasta q' no queden nodos de nivel $i-1$ sin flechas de salida



nivel $\log(n)$




nivel 2



nivel 1

Comentario al margen:

Crea q' si hago  de tantos

$$\bullet \text{ OR} = \{x_1, \dots, x_n : \exists 1 \leq i \leq n, x_i = 1\}$$

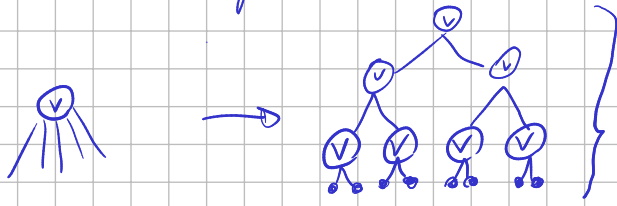
Es muy similar a la anterior solución pero con \vee

Si, acabo de descubrir q' hay distintas tijas de hojas !!

Ahora q'vq $AC^d \subseteq NC^{d+1}$ para todo $d \geq 0$

Idea: Esto debería salir viendo que cada operación g' se haga en AC^d puede reproducirse de forma g' quede en NC^{d+1}

Para hacer lo planteado en la Idea comienza definiendo el \odot :



Esto lleva a g' por cada \odot arbitrario pero se tiene un árbol de \odot no arbitrario de altura a lo sumo $\log(n)$

Entonces esta sería la forma para representarlo y toma $\log(n)$ por cada \odot arbitrario, por lo cual queda a lo sumo $O(\log^d(n))$ de profundidad

Para definir el \oplus es la misma idea g' para el \odot .

Luego para el caso de \ominus no es necesario, pues no tendría sentido un \ominus arbitrario.

Ej 10:

10. Probar que $NC^1 \subseteq L$.

(Es el Teo 33)

Idea: Tomo un $L \in NC^1$ y pruebo g' se encuentra en L .

$\pi \in NC^1$ sii π es decidible por una familia de circuitos g' son L -uniformes, $|C_n|$ es $p(n)$ y $\text{prof}(C_n) = O(\log(n))$.

$x \in \pi$ sii $C_{|x|}(x) = 1$

Q'vq $\pi \in L$, o sea, g' π sea decidible por una máq. det. M g' corre en espacio logarítmico.

$M \langle x \rangle$

Se puede hacer g' la familia de circuitos es L -uniforme.

1- Con M' genero el i -ésimo bit del circuito $C_{|x|}$ ingresándole $\langle |x|, i \rangle$

2- Voy evaluando recursivamente el circuito C_n con entrada x usando M'

Es similar, pero en la teórico esta mejor la demo.

E; LL:

11. Decidir si las clases AC^k y NC^k están cerradas por unión, intersección y complemento.

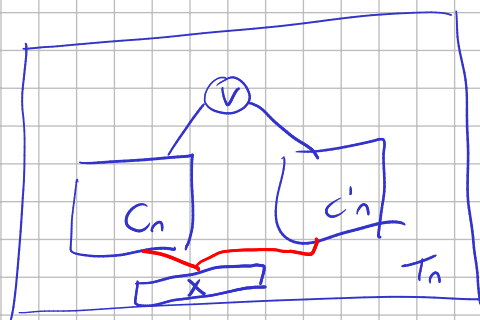
Unión AC^k :

Si tengo $L \in AC^k$ y $\pi \in AC^k \Rightarrow L \cup \pi \in AC^k$.

Sé que L y π tienen sus respectivas familias de circuitos $(C_n)_{n \in \mathbb{N}}$ y $(C'_n)_{n \in \mathbb{N}}$.

Entonces puedo formar la familia de circuitos de la siguiente manera:

• $(T_n)_{n \in \mathbb{N}}$: Es la familia de circuitos q' por cada n se pone a correr C_n y C'_n y ve si alguno de estos da 1



→ Que las circuitos estén en paralelo no le agrega altura al circuito.

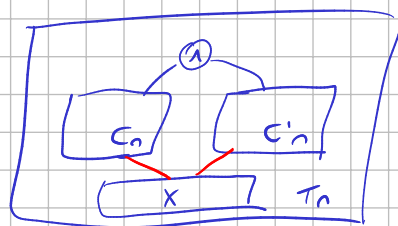
Considerar las líneas rojas como una conexión de todos los bits de x a los circuitos.

Unión NC^k :

Es el mismo concepto q' para la unión de NC^k .

Intersección AC^k y NC^k :

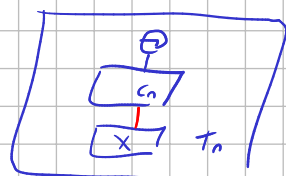
Es similar al concepto de la unión, pero en vez de un \vee es un \wedge , para indicar q' la palabra está en ambos lenguajes.



Complemento AC^k :

Si $L \in AC^k$ tal q' sea decidido por la familia de circuitos $(C_n)_{n \in \mathbb{N}}$

Luego, $\bar{L} = \{x : x \notin L\}$, entonces si el circuito $C_{|x|}(x) = 1$ la nueva familia de circuitos $(T_n)_{n \in \mathbb{N}}$ por cada $C_{|x|}(x)$ devuelve $\neg C_{|x|}(x)$.



Complemento NC^k:

Misma idea q' para AC^k.

✓

Ej 12:

12. Dadas dos matrices $A, B \in \{0, 1\}^{n \times n}$ definimos el producto booleano como

$$(A \cdot B)_{ij} = \bigvee_{k=1}^n (A_{ik} \wedge B_{kj})$$

Disyunción de la conjunción de elementos de la Fila i de A y la col j de B (uno a uno)

Considerar el lenguaje

- $\text{PBM} = \{ \langle A, B, n, i, j \rangle : A, B \in n \times n, 0 \leq i, j < n, (A \cdot B)_{ij} = 1 \}$

Probar que:

✓ Chequea q' en cierta posición del producto booleano de A y B tenga un 1.

a) $\text{PBM} \in \text{AC}^0$.

b) El lenguaje

- $\text{EBM} = \{ \langle A, n, k, i, j \rangle : A \in \{0, 1\}^{n \times n}, k \leq \log n, (A^{2^k})_{ij} = 1 \}$

✓ Calculo q' esta potencia es con productos booleanos

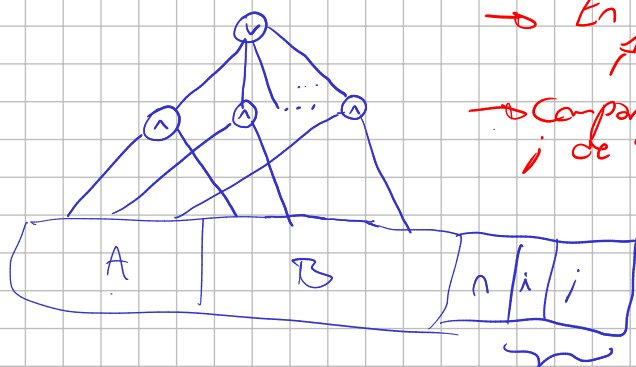
está en AC^1 .

c) Concluir que $\text{NL} \subseteq \text{AC}^1$.

2) $\text{PBM} \in \text{AC}^0$

Preguntar si es esta la idea.

Idea: Para q' esté en AC^0 tiene q' tener una familia de circuitos $(C_n)_{n \in \mathbb{N}}$ q' decida PBM y q' sea L -uniforme tal q' $|C_n| \leq \text{poly}(n)$ y $\text{pos}(C_n) = \text{cite}$.



→ En alguna pos. ambas son 1 → Retorna 1.

→ Compara 1 a 1 la Fila i de A con la columna j de B .

→ Fila i de A , col j de B , elementos de 1 a n .

me indica qué posiciones debo tomar

6) $\text{EBM} \in \text{AC}^1$

Esto sale relativamente fácil al tener a), $k \leq \log n$ y $(A^{2^k})_{ij} = 1$.

$A^{2^k} = ((A^2)^{2^{k-1}})^{2^{k-1}}$, entonces esto sería aplicar el circuito de a) k veces, ya q' $A^2 = A \cdot A$ (y también $X^2 = X \cdot X$ con $X = A^{2^c}$ con $c \leq k$).

Luego, tengo un circuito de AC^0 q' se repite a lo sumo $\log n$ veces, por lo cual sé que $\text{EBM} \in \text{AC}^1$ pues es $\log n$ veces un circuito de AC^0 , entonces queda $\log n \times \text{cite}$ (en este caso $\text{cite} = 1$).

C) $NL \subseteq AC^1$

Idea:

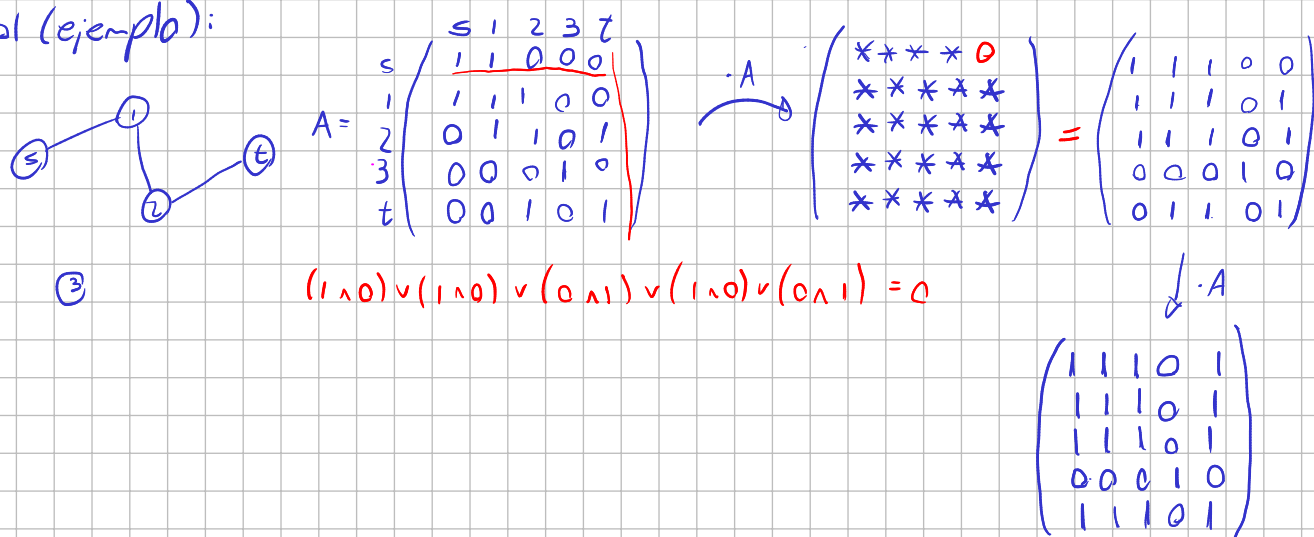
¿Sé q' PATH es NL-completo, o sea q' si $PATH \in AC^1$, entonces $NL \subseteq AC^1$?

• Cómo resolver PATH con una familia de circuitos $(C_n)_{n \in \mathbb{N}}$; $|C_n| \leq poly$ de n ;
 $C \cdot prof(C_n) \leq \log n$

Ojo, a ver, si yo tengo una matriz de adyacencia representando el grafo, entonces el producto booleano me permite obtener el siguiente resultado de qué nodos están conectados con cuáles otros con a lo sumo 2 aristas de distancia.

Bueno, siguiendo la idea planteada anteriormente podemos usar EDM (el cual sé q' está en AC^1) con la entrada de $\langle A, n, k, s, t \rangle$ donde A es la matriz de adyacencia de la entrada de PATH, s es el starting point de PATH, t es el terminal y n es la cantidad de nodos.

Visual (ejemplo):



¿Sé q' luego de k multiplicaciones a A , van a aparecer todas las conexiones entre nodos q' estén a lo sumo a una k distancia de cada nodo.

Como debo ver todas las caminos de a lo sumo n distancia, debo tener A^n para ver si se conectan de alguna forma dos nodos. Entonces el k que voy a estar utilizando es $k = \log n$, pues $A^n = A^{2^{\log n}}$.

Ahora mi pregunta es, esta que estoy planteando de la multiplicación de la matriz de adyacencia, ¿tiene sentido o solo me autoconvienci de q' funciona?

Vamos a hacer una demo informal al respecto:

Si yo tengo A me muestra las conexiones entre nodos directas. Ahora bien, si yo tengo A^k , ¿qué significa?

Por cada $Z \cdot Y$ uno obtiene en cada posición $(Z \cdot Y)_{ij} = \bigvee_{k=1}^n Z_{ik} \wedge Y_{kj}$.

Si uno lleva el anterior caso a matrices con significados de conexiones de nodos, nos quedamos con que en la posición ij está en 1 si el nodo i está conectado a algún nodo k y el nodo j está conectado al mismo nodo k (Estamos viendo $A \cdot A$ pero aplica para $A^k \cdot A^k$ también).

Por último, esto se encuentra en AC^1 , pues sería básicamente usar el circuito de EDM con la entrada $\langle A, n, k, s, t \rangle$ descrito anteriormente.

✓

Ej 13:

13. En este ejercicio se demuestra que el lenguaje

$$\blacksquare \text{ MAJORITY} = \{x : x \text{ tiene mas 1s que 0s}\}$$

está en NC^1 .

- Diseñar un circuito NC^0 que haga lo siguiente: dados 3 números de n bits x, y, z devuelve dos números u, v tales que $x + y + z = u + v$. **Ayuda:** Tomar $v_{i+1}u_i = x_i + y_i + z_i$.
- Probar que el lenguaje $\mathcal{L} = \{\langle s_1, s_2, \dots, s_k, r \rangle : s_i, r \subseteq \{0, 1\}^*, s_1 + s_2 + \dots + s_k = r\}$ está en NC^1 .
- Probar $\text{MAJORITY} \in \text{NC}^1$.

Voy a repasar otras guías q' hay hay consultas ^.^