

Guía 2

Ej 1:

Si sos del futuro y ves esto, considerá que ya no sé más cuanto tiempo estoy estudiando.

1. Probar que los siguientes lenguajes están en P.

- a) $\text{COPRIME} = \{ \langle a, b \rangle : (a, b) = 1, \text{ es decir, } a \text{ y } b \text{ son coprimos} \}$
- b) $\text{POWER} = \{ \langle a, e, b \rangle : a^e = b \}$
- c) $\text{TREE} = \{ \langle G \rangle : G \text{ es un grafo conexo sin ciclos} \}$
- d) \mathcal{L} donde $|\mathcal{L}| < \infty$ (es decir, probar que todo lenguaje finito está en P).

a)

a y b son coprimos: Significa que su MCD = 1 máx. común divisor

Para probar que está en P escribo el código de la máquina determinística polinomial en el siguiente pseudocódigo:

M: $\langle a, b \rangle$

while ($b \neq 0$):

temp = b
b = a mod b
a = temp

ret(a = 1)

→ Algoritmo de Euclides para calcular MCD (gcd en inglés).

→ Hace a la suma $\log b$ iteraciones

b) $a^e = b$

M: $\langle a, e, b \rangle$

c = 1
base = a

while ($e > 0$):

if ($e \bmod 2 = 1$):

c = c * base

base = base * base
e >> 2

ret(c = b)

c) Grafo conexo sin ciclos.

DFS o BFS y chequeo si en algún momento quiero visitar un nodo que ya visité $O(|V|)$

d) \mathcal{L} donde $|\mathcal{L}| < \infty$

El lenguaje es el que reciben lenguaje \mathcal{L} y ve que este sea finito, esto sale porque \mathcal{L} es representado por un conjunto de palabras (cuerpo ya) y no por una expresión.

Entonces tengo que \mathcal{L} pertenece si: puedo leer la entrada, si no se cuelga (pues sería infinito).

Ej 2:

2. Probar que la clase P está cerrada por unión, intersección y complemento.

Idea: Tomar 2 o 1 \mathcal{L} genérico en P y probar q' se cumplen las propiedades

Unión:

Sea $L \in P$ y $\Pi \in P$ la M. máq. det. poly. que decide $L \cup \Pi$ tendría esta forma

$M: \langle x \rangle$

$\text{ret}(x \in L \parallel x \in \Pi) \rightarrow$ * Como L y $\Pi \in P$ puedo usar máquinas det. poly para computar si x pertenece a cada una, y cómo es polinomial sobre la entrada (x) quedo de computar la suma de complejidades de estas máquinas, y la suma de polinomios es un polinomio

Intersección:

$L, \Pi \in P$. M. máq. det. poly. que decide $L \cap \Pi$:

$M: \langle x \rangle$

$\text{ret}(x \in L \&\& x \in \Pi) \rightarrow$ *

Complemento:

Sea $L \in P$ y M' la máq. det. poly tal que $M'(L)$, construyo M para \bar{L} como:

$M: \langle x \rangle$

$\text{ret}(\neg M'(x)) \rightarrow$ *

Ej 3:

3. Probar que los siguientes lenguajes están en NP.

- HAMPATH = $\{\langle G, s, t \rangle : G \text{ es un grafo con dos nodos } s \text{ y } t \text{ tales que hay un camino hamiltoniano de } s \text{ a } t\}$
- k -CLIQUE = $\{\langle G \rangle : G \text{ es un grafo con subgrafo completo de tamaño mayor o igual a } k\}$
- CLIQUE = $\{\langle G, k \rangle : G \text{ es un grafo con subgrafo completo de tamaño mayor o igual a } k\}$
- k -COLORING = $\{\langle G \rangle : G \text{ es un grafo que se puede particionar en } k \text{ conjuntos independientes}\}$
- ISOMORPHISM = $\{\langle G, H \rangle : G \text{ y } H \text{ son dos grafos isomorfos}\}$
- SUBGRAPH ISOMORPHISM = $\{\langle G, H \rangle : G \text{ es un grafo y } H \text{ es isomorfo a un subgrafo inducido de } G\}$
- $\neg\text{SAT} = \{\langle \phi \rangle : \neg\phi \text{ es satisfacible}\}$

Para economizar palabras por ahí escribo directamente el certificado y el algoritmo, pero se podría y debería formalizar/justificar por ahí mejor por qué el algoritmo funciona.

$\rightarrow \langle V, E \rangle \rightarrow V = \text{Vertex } E = \text{Edges}$

a) HAMPATH: $\{\langle G, s, t \rangle : \text{Camino hamiltoniano (no repite nodos y los recorre todos) de } s \text{ a } t\}$

Certificado: Conjunto de $|V|$ nodos.

Algoritmo: $\langle G, s, t, u \rangle$ ^{o certificado}

- Verifico que $|u| = |V| \rightarrow O(|V|)$

- Verifico que todos los nodos de u pertenezcan a $V \rightarrow O(|V|^2)$

- Verifico que el primer nodo sea s y el último sea t . $\rightarrow O(2 \log |V|)$

- Veo que no haya nodos repetidos en u . $\rightarrow O(|V|^2)$

- Veo que de $1 \leq i < n$ $(u[i], u[i+1]) \in E$. O sea, que cada nodo este conectado con el que le sigue. $\rightarrow O(|V|^3)$

Básicamente el algoritmo b que hace es que en el certificado tengo el supuesto camino hamiltoniano y verifico que este sea hamiltoniano efectivamente.
 u es de tamaño $|V|$ por b cual es poly y el verificador q corre el algoritmo también es poly .

b) K-CLIQUE: $\{ \langle G \rangle : G \text{ es un grafo con un subgrafo completo de tamaño mayor o igual a } K \}$

A ver, ¿qué nos piden? QVQ G tenga una clique de al menos K nodes (si tiene una clique de $K+1$ nodes, tiene una de K). O sea, que G tenga una clique de K nodes.

Luego, sé que K es cte. pues no depende de la entrada, si no de el nombre (sea 2-CLIQUE, o 3-CLIQUE, o etc.).

Entonces puedo ver que G tiene a lo sumo $\binom{|V|}{K} = O(|V|^K)$ ^{correcto.} subconjuntos de K nodes.

Por último para ver que un subconjunto de K nodes esté conectado cada nodo a todo el resto sería de costo $O(K^2 |V|)$ por cada nodo busco en la matriz de adyacencia $O(|V|^2)$ que esté conectado a todo el resto.

Ahora, ¿qué notamos acá? Esto está en P ! (Como $P \subseteq NP \rightarrow$ El algoritmo está en NP).

Algoritmo: $\langle G \rangle \rightarrow O(|V|^K \cdot K^2 \cdot |V|^2) = O(|V|^{K+2})$ porque K es cte esto queda polinomial.

- Genero todos los subconjuntos de tamaño K posibles y por cada s hago $\rightarrow O(|V|^K)$

- Verifico que cada nodo del s esté conectado a todo otro: $\rightarrow O(K^2 |V|^2)$

- ret true si lo está, si no continue.

- ret false.

c) CLIQUE: $\{ \langle G, K \rangle : G \text{ es un grafo con una clique de tamaño } \geq K \}$

Muy similar al anterior, pero... El algoritmo que usamos antes, ^{Deja de ser poly!}

Por qué? Bueno, cuando queremos generar todos los subconjuntos de tamaño K posibles, si $K = \frac{n}{2}$, entonces esto me queda $\binom{n}{\frac{n}{2}} = O(n^{\frac{n}{2}})$. Lo cual es exponencial!

Bien, ahora como ya no me voy a poner a probar si $P = NP$ (o si (!)), armo un algoritmo de la forma de NP (con certificados y verificador).

Certificado: Subconjunto de V con K nodes. $\rightarrow O(K \cdot \log |V|) = O(|V|)$

Algoritmo: $\langle G, K, c \rangle$

- Verifico que $|c| = K \cdot \log |V|$. $\rightarrow O(K \log |V|) = O(|V| \log |V|)$

- Verifico que cada nodo en c sea parte de $V \rightarrow O(|c|) = O(|V|)$

- Verifico que no haya nodes repetidos en $c \rightarrow O(|c|^2) = O(|V|^2)$

- Chequea que cada nodo de C esté conectado con todos los otros nodos del conjunto C . $\rightarrow O(|V|^4)$

d) **K-COLORING**: $\{ \langle G \rangle : G \text{ es un grafo que se puede particionar en } K \text{ ctes independientes (no tienen aristas directas entre nodos)} \}$.

Certificado: Conjunto de K conjuntos de nodos. $\rightarrow O(|V|)$

Algoritmo: $\langle G, c \rangle$

- Cada nodo de todos los subconjuntos estén en V . $\rightarrow O(|V|^2)$
- La cant. de nodos en C sea la misma que en V . $\rightarrow O(|V|)$
- No haya nodos repetidos en C . $\rightarrow O(|V|^2)$
- Por cada subconjunto de C veo que no haya conexiones directas a los otros nodos $\rightarrow O(|V|^4)$

e) **ISOMORPHISM**: $\{ \langle G, H \rangle : G \text{ y } H \text{ son dos grafos isomorfos (o sea tienen la misma estructura, existe una correspondencia entre sus vértices donde se preserve la adyacencia y no adyacencia)} \}$

Certificado: Una correspondencia de nodos tal que sea un conjunto de tuplas donde $\forall v \in V, w \in V_H \Rightarrow (v, w) \in C$.

Algoritmo: $\langle G, H, c \rangle$

Veo que:

- C tenga el doble de la cantidad de elementos de V_G y de V_H . $\rightarrow O(|V|)$
- Todas las primeras componentes pertenecen a V_G y las segundas a V_H . $\rightarrow O(2|V|)$
- No hay repetidos de primeras ni de segundas componentes $\rightarrow O(2|V|^2)$
- Por cada componente de C veo que se correspondan las conexiones que tenga en E_G con las de E_H siguiendo la asociación de los nodos $\rightarrow O(|V|^4)$. \rightarrow O por ahí, no me lo puse a calcular pero es polinomial seguro.

f) **SUBGRAPH-ISOMORPHISM**: $\{ \langle G, H \rangle : G \text{ es un grafo y } H \text{ es isomorfo a un subgrafo inducido de } G \text{ (o sea, tiene un subconjunto de nodos de } G \text{ con todas las aristas correspondientes a esos nodos en } G \text{ que los dos extremos estén en } H) \}$

Certificado: Conjunto de tuplas de pares donde los nodos de V_H estén asociados a uno de V_G . La longitud del conjunto es igual a la cant. de elementos de V_H . $\rightarrow O(2|V_H|)$

Algoritmo: $\langle G, H, c \rangle$

- C tiene el doble de elementos que V_H . $\rightarrow O(|V_H|)$
- $\forall (v, w), (v, w) \in C \ \&\& \ v \in V_G \ \&\& \ w \in V_H$. $\rightarrow O(|V|^2)$
- No hay nodos repetidos en C $\rightarrow O(|V|^3)$
- Para cada nodo de V_H verifico que sus nodos estén conectados con los que corresponden siguiendo la relación con los nodos de V_G . \rightarrow Poly

g) $\neg \text{SAT} : \{ \langle \phi \rangle : \neg \phi \text{ es satisfacible} \}$

Certificado: Valución de tamaño $|\phi|$.

Algoritmo: $\langle \phi, c \rangle$

$\text{ret}(\neg(c \models \phi))$ $\rightarrow m$ es la cont. de var. proposicionales de ϕ .

Ej 4:

4. Probar que los siguientes problemas están en coNP.

a) $\text{PRIME} = \{n : n \in \mathbb{N} \text{ es primo}\}$

b) $\text{GIRTH} = \{ \langle G, k \rangle : G \text{ es un grafo tal que todos sus ciclos simples tienen } k \text{ o menos vértices} \}$

c) $\text{TAUTOLOGY} = \{ \langle \phi \rangle : \phi \text{ es tautología} \}$

Para estas problemas lo que tengo que hacer es ver que sus complementos están en NP, o sea, que $\text{PRIME}, \text{GIRTH}, \text{TAUTOLOGY} \in \text{NP}$.

a) $\text{PRIME} : \{n : n \in \mathbb{N} \text{ es primo}\} \rightarrow \text{Ojo! } n \text{ es un número por lo que poly sería algo } \log n = |\ln|$
 $\overline{\text{PRIME}} : \{n : n \in \mathbb{N} \text{ no es primo}\}$

Certificado: Un número tal que ese número sea divisor de $n \rightarrow O(|\ln|)$

Algoritmo: $\langle n, c \rangle$

$\text{ret}(n \bmod c \neq 0)$

b) $\text{GIRTH} : \{ \langle G, k \rangle : G \text{ es un grafo tal que todos sus ciclos simples tienen } k \text{ o menos vértices} \}$

$\overline{\text{GIRTH}} : \{ \langle G, k \rangle : \text{Hay un ciclo simple con } > k \text{ vértices} \}$

Certificado: Conjuntos de nodos (Representan el ciclo en cuestión) de longitud mayor a k y menor a $|V| \rightarrow O(|V|)$

Algoritmo: $\langle G, k \rangle$

- Las nodos de c deben estar en V , $|c| \leq |V|$, $c[1] = c[2]$, $\rightarrow O(|V|^4)$ erro. de ult elemento

- No deben haber elementos repetidos (exceptuando el último nodo). $\rightarrow O(|V|^2)$

- Verificar que $\forall i. 1 \leq i < n, (c[i], c[i+1]) \in E_G$, $\rightarrow O(|V|^3)$

c) $\text{TAUTOLOGY} : \{ \langle \phi \rangle : \phi \text{ es tautología} \}$

TAUTOLOGY es básicamente decir que $\neg \phi$ tiene una valuación que lo satisface (o sea que ϕ no es una tautología por lo cual $\neg \phi$ no es una contradicción).

Es hacer $\neg \text{SAT}$ (lo cual fue demostrado en 3.9).

$\text{TAUTOLOGY} = \neg \forall v. v \models \phi = \exists v. v \not\models \phi = \exists v. v \models \neg \phi$

Lo si hay una v que no satisface a ϕ entonces, si niego ϕ , esta v debería satisfacerlo.

Ej 5:

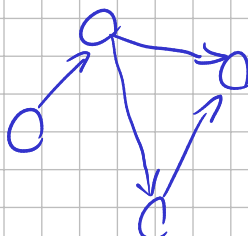
5. Considerar los siguientes lenguajes:

- $\text{PATH} = \{ \langle G, s, t, k \rangle : G \text{ es un digrafo con dos nodos } s \text{ y } t \text{ tales que hay un recorrido de longitud menor o igual a } k \text{ de } s \text{ a } t \}$
- $\text{EVEN PATH} = \{ \langle G, s, t, k \rangle : G \text{ es un digrafo con dos nodos } s \text{ y } t \text{ tales que hay un recorrido de longitud par y menor o igual a } k \text{ de } s \text{ a } t \}$

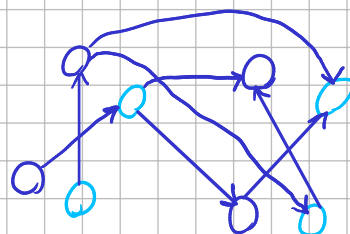
Dado un digrafo G , definimos G' como el digrafo que tiene dos copias (v, p) y (v, i) de cada v rtice $v \in V(G)$ donde $(v, x) \rightarrow (w, y)$ es una arista de G' si y solo si $v \rightarrow w \in E(G)$ y $x \neq y$.

- Demostrar que $\langle G, s, t, k \rangle \in \text{EVEN-PATH} \iff \langle G', (s, p), (t, p), k \rangle \in \text{PATH}$.
- Mostrar que la reducci n de EVEN-PATH a PATH implicada por el punto anterior es polinomial.

a) G :



G' :



$\Rightarrow \langle G, s, t, k \rangle \in \text{EVEN-PATH} \Rightarrow \langle G', (s, p), (t, p), k \rangle \in \text{PATH}$

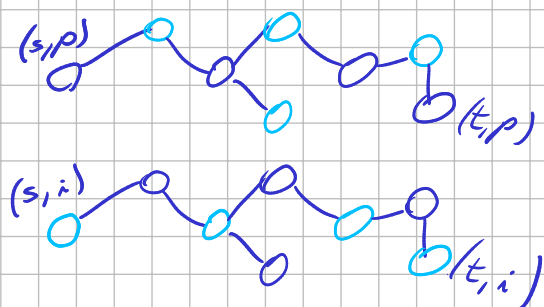
Ahora bien, veamos por construcci n c mo se cumple esto.

Tengo un s y t tal que hay un camino par entre ellos:



Luego, construyo G' donde duplico la cantidad de nodos y coloco las aristas de manera que:

Si G tiene un EVEN-PATH de s a t significa, por c mo se colocan las edges en G' que hay algo de la pinta $\{(s, p), (x_1, i), (x_2, p), \dots, (x_{k-1}, i), (t, p)\}$ y tambi n otro $\{(s, i), \dots, (t, i)\}$. O sea que hay un PATH de (s, p) a (t, p) de longitud k .



$\Leftarrow \langle G', (s, p), (t, p), k \rangle \in \text{PATH} \Rightarrow \langle G, s, t, k \rangle \in \text{EVEN-PATH}$

Esta vuelta sale medio trivial al haber probado la ida, pero ser  algo as :

Tengo G' que tiene un camino de (s, p) a (t, p) de k pasos, ahora c mo se que existe uno de s a t en G ? Bastante sencillo! Como la construcci n de G' depende de G , puedo ver que si hay un camino de s a t tiene que haber uno de (s, x) a (t, y) donde x puede llegar a ser igual a i .

Ahora, qu  significa que $x=y$? Si $x=y$, entonces el camino tiene longitud par ya que por construcci n vemos que no hay impares por si 2 nodos est n conectados el segundo elemento de la tupla DEBE tener el otro s mbolo en su segundo componente.

b) $\{p\}$

Cómo demuestra si la reducción es polinomial? Vea que f (descrita anteriormente) sea computable polinomialmente.

$L \leq_p L'$ def: $\exists f$ computable poly tal que

$$x \in L \text{ si: } f(x) \in L'$$

$M: \langle G \rangle$

- Por cada $v \in V$:

$\rightarrow O(|V|)$

- Crea (v, p) y (v, i) y los reto en la cinta de salida

- Por cada $(v, w) \in E$:

$\rightarrow O(|V|^2)$

- Crea $((v, i), (w, p))$ y $((v, p), (w, i))$ y los reto en la salida

Ej 6:

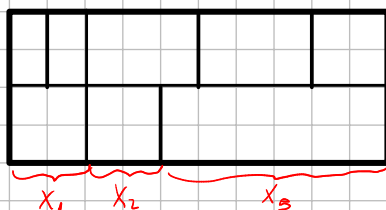
6. Considerar los siguientes lenguajes:

- 2-PARTITION = $\{\langle X \rangle : X \text{ es un subconjunto finito de los números naturales con } \min_{x \in X} (x) > 2 \text{ que se puede particionar en dos conjuntos } X_1, X_2 \text{ tales que } X_1 \cap X_2 = \emptyset, X_1 \cup X_2 = X \text{ y } \sum_{x \in X_1} x = \sum_{x \in X_2} x\}$
 - 3-PARTITION = $\{\langle X, t \rangle : X \text{ es un subconjunto finito de los números naturales tal que } \sum_{x \in X} x = \frac{|X|t}{3}, \text{ máx}_{x \in X} x < \lfloor t/2 \rfloor \text{ y } X \text{ se puede particionar en } \frac{|X|}{3} \text{ triplas donde cada una suma } t\}$
 - RECTANGLE PACKING = $\{\langle R, r_1, \dots, r_k \rangle : R \text{ es un rectángulo que se puede cubrir completamente y sin superposición usando los rectángulos } r_1, \dots, r_k \text{ con traslaciones y/o rotaciones}\}^1$ Un rectángulo en este contexto se representa con dos números naturales indicando su ancho y alto.
- a) Dada una instancia $\langle X \rangle$ de 2-PARTITION, se define la instancia $\langle R, r_1, \dots, r_{|X|} \rangle$ de RECTANGLE PACKING donde R tiene base $\sum_{x \in X} x/2$ y altura 2, y r_i es un rectángulo de base x_i y altura 1, para cada $1 \leq i \leq |X|$. Demostrar que $\langle X \rangle \in 2\text{-PARTITION} \iff \langle R, r_1, \dots, r_{|X|} \rangle \in \text{RECTANGLE PACKING}$.
- b) Dada una instancia $\langle X, t \rangle$ de 3-PARTITION, se define la instancia $\langle R, r_1, \dots, r_{|X|} \rangle$ de RECTANGLE PACKING donde R tiene base $\frac{|X|}{3}$ y altura $t + |X|$ y r_i es un rectángulo de base 1 y altura $\frac{|X|}{3} + x_i$, para cada $1 \leq i \leq |X|$. Demostrar que $\langle X, t \rangle \in 3\text{-PARTITION} \iff \langle R, r_1, \dots, r_{|X|} \rangle \in \text{RECTANGLE PACKING}$.
- c) Mostrar que las reducciones implicadas por los puntos anteriores son polinomiales en función de los tamaños de las entradas.

a)

$$\{x_1, x_2, \dots, x_{|X|}\}$$

si:



$\rightarrow Z_1$ (Rectángulo superior)

$\rightarrow Z_2$ (Rectángulo inferior)

$\Rightarrow \langle X \rangle \in 2\text{-PARTITION} \Rightarrow \langle R, r_1, \dots, r_{|X|} \rangle \in \text{RECTANGLE PACKING}$

Si $X \in 2\text{-P}$ entonces tengo 2 subconjuntos Y_1, Y_2 donde $\sum_i x_i = \sum_i x_i$.

Luego, puedo ver que para construir R se usan las $x_i \in X$. Si uso las $x_i \in Y_1$ para formar Z_1 con la base x_i y altura 1 y luego uso las $x_i \in Y_2$ para formar Z_2 y los apilo me queda un rectángulo con base $\sum_i x_i/2$ y altura 2 como resultado, lo cual es lo que quería y que pertenece a R-P.

Las rectángulitas $r_1, \dots, r_{|X|}$ son las que forme de base x_i y altura 1 a medida que declaraba la anterior.

$\Leftarrow \langle R, r_1, \dots, r_{|X|} \rangle \in \text{R-P} \Rightarrow \langle X \rangle \in 2\text{-P}$

Por cómo se construye R y r_i (explicada más a detalle en \Rightarrow) veo que puedo asociar cada r_i a un x_i tomando el valor de x_i como la base de r_i .

Luego, sé que van a haber 2 subconjuntos de X q' cumplan que

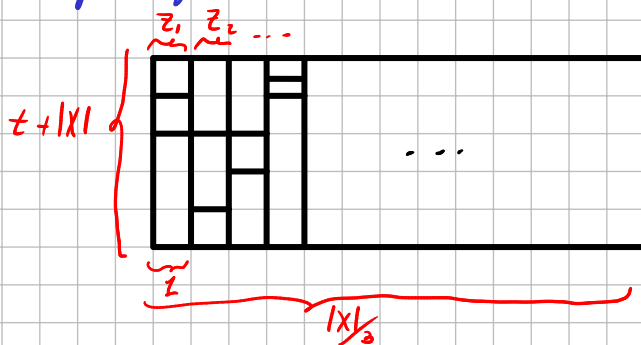
$X_1 \cap X_2 = \emptyset$: Ya que cada x_i se asocia a un r_i diferente y el rectángulo tiene 2 subrectángulos más chicos Z_1 y Z_2 donde se conforman por r_i 's distintos

$X_1 \cup X_2 = X$: Como x_i se relaciona a un r_i directamente y se usan todas las r_i 's para formar R puedo usar todas las x_i para X_1 y X_2 y que cumplan las otras propiedades.

$\sum_{x \in X_1} x = \sum_{x \in X_2} x$: Como son 2 rectángulos de = base Z_1 y Z_2 puedo dividir a X_1 y X_2 asociando a X_1 las r_i de Z_1 y a X_2 las r_i de Z_2 y obtener que tengan igual sumatoria de elementos.

b)

$(\{x_1, \dots, x_{|X|}\}, t) \Leftrightarrow$



La demo es muy similar a la anterior. Así que lo voy a hacer breve

$\Rightarrow \langle X \rangle \in 3-P \Rightarrow \langle R, r_1, \dots, r_{|X|} \rangle \in R-P$

Por cada tripla de X tengo a un Z_i (columna del rectángulo de base 1 y altura $t + |X|$) ya q' cada rectángulo en Z_i tiene altura $\frac{|X|}{3} + x_i (= \text{altura } r_i)$ por lo cual coincide con $t + |X|$ la altura de estas r_i apiladas.

$\Leftarrow \langle R, r_1, \dots, r_{|X|} \rangle \in R-P \Rightarrow \langle X \rangle \in 3-P$

Por construcción r_i corresponde a un x_i , no hay repetidos, se cumple que sumen t las triplos pq' si no la altura del rectángulo no da y como la base de R es $\frac{|X|}{3}$ y la base de r_i es 1, entonces son triplitas asociados de a 3 cada Z_i .

c)

$2-P \leq_p R-P$

$M: \langle X \rangle$

- A R lo declaro como un rectángulo de base $|X|$ y altura 2.

- Por cada $x_i \in X$ construyo r_i con base x_i y altura 1.

$3-P \leq_p R-P$

$M: \langle X, t \rangle$

- Declaro R como rectángulo de base $\frac{|X|}{3}$ y altura $t + |X|$

- Por cada $x_i \in X$ construyo r_i con base 1 y altura $\frac{t}{3} + x_i$.

✓

Ej 7:

7. Explicar por qué la identidad no es una reducción polinomial de un lenguaje Π a Π^c . Concluir que las nociones de NP y coNP son altamente sensibles a la "etiqueta" de la respuesta.

¿Qué me están diciendo?

↳ "sí pertenece" o "no pertenece" que devuelve la máquina

$$x \in \Pi \iff \text{id}(x) \in \Pi^c \iff x \in \Pi^c$$

A ver, no es una reducción porque si Π acepta a x $\text{id}(x)$ (que es x) no va a ser aceptado por Π^c (o sea su complemento) porque justamente Π acepta lo que Π no acepta y rechaza lo que Π acepta (por definición).

Luego, tengo que las nociones de NP y coNP son sensibles a la etiqueta porque el que acepte un x en $L^c \in \text{coNP}$ requiere que haya un $L \in \text{NP}$ que rechace esa entrada y lo mismo para cuando rechaza L , L acepta.

Notar que en los problemas coNP es fácil certificar la no pertenencia, al revés que en NP.

Ej 8:

8. Considerar el siguiente lenguaje:

- CONNECTED = $\{\langle G, s, t \rangle : G \text{ es un digrafo y } s \text{ y } t \text{ dos nodos de } G \text{ tales que hay un recorrido de } s \text{ a } t\}$

Para un digrafo G , sea H el digrafo que tiene un vértice (S, v) para cada $S \subseteq V(G)$ y cada $v \in V(G)$, donde $(S, v) \rightarrow (R, w)$ es una arista de H si y solo si $w \notin S$, $R = S \cup \{w\}$ y $v \rightarrow w$ es una arista de G .

- Demostrar que $\langle G, s, t \rangle \in \text{HAMPATH} \iff \langle H, (\{s\}, s), (V(G), t) \rangle \in \text{CONNECTED}$.
- Mostrar que la reducción de HAMPATH a CONNECTED implicada por el punto anterior no es polinomial.

*'

O sea que tenés una cant. de elem. igual a $P(V)$ (potencia de $V(G)$) por cada nodo de G , o sea tenés un total de $n \cdot 2^n$ nodos para H .
Omg!!! Es exponencial!

a)

$$\Rightarrow \langle G, s, t \rangle \in \text{HAM} \Rightarrow \langle H, (\{s\}, s), (V(G), t) \rangle \in \text{CON}$$

Para ver esto me enfoca en el armado de H y el recorrido que implica que x pertenezca a CON.

Si hay camino hamiltoniano de s a t en G particularmente en la creación de H va a existir un camino de $(\{s\}, s)$ a $(V(G), t)$ en H , ya que por cada arista del recorrido que hace en G , existe un nodo que pertenece al recorrido que hace de $(\{s\}, s)$ a $(V(G), t)$ de la forma (L, l) donde l es el nodo correspondiente a s el nodo que se lo relaciona en G y L sería el conjunto de nodos que ya pasó del recorrido por 1 vez.

O sea, que si tiene $V(G)$ al final del recorrido (habiendo arrancado con $\{s\}$) significa que todos los nodos de $V(G)$ fueron atravesados por el recorrido una única vez en G .

$$\Leftarrow \langle H, (\{s\}, s), (V(G), t) \rangle \in \text{CON} \Rightarrow \langle G, s, t \rangle \in \text{HAM}$$

Bastante similar, por construcción de H se ve que sucede.

b)

Por el comentario hecho en *' puede ver que lo f que reduciría el problema de una entrada x de CON a una de HAM debería ser exponencial para que respete el armado de los nodos de la H .

Ej 9:

²RECURSIVE es el conjunto de problemas decibles, o bien *recursivos*.

9. Sea $\mathcal{L} \in \text{RECURSIVE}$.² Probar que $\mathcal{L} \leq_p \text{HALTING}$.

$$x \in \mathcal{L} \text{ ssi } f(x) \in \text{HALTING}$$

¿Qué hacía HALTING?

Recibe un programa x y corre $U(\langle x, x \rangle) \rightarrow$ Si se cueda retorna 0, si no ret 1.

Voy a asumir que tengo la máquina que decide \mathcal{L} determinística, se llama N .

Quiero una f computable en tiempo $poly$ tal que si $x \in \mathcal{L}$ devuelva algo computable y si no algo que no lo sea.

$M: \langle x \rangle \rightarrow$ La f de M devuelve la codificación de una máquina M' construida con x .
ret $\langle M' \rangle$

$M': \langle z \rangle \rightarrow$ Va a ser ignorado (Acá entra la codificación de M' cada se hace $U(\langle x, x \rangle)$)

if ($N(x) == 1$):

ret true

else:

while(true)

ret false

¿Te imaginás que llegue acá? De loco sería.

Ej 10:

10. Probar que $\text{NP} \subseteq \text{RECURSIVE}$. Concluir que $\text{HALTING} \notin \text{NP}$.

Para todo $\mathcal{L} \in \text{NP}$, sé que $\mathcal{L} \in \mathcal{R}$ por lo siguiente:

Arranquemos viendo ¿qué significa que $\mathcal{L} \in \text{NP}$?

$$x \in \mathcal{L} \text{ ssi } \exists u. M(\langle x, u \rangle) = 1$$

Con M det. $poly$ y $u \in \{0, 1\}^{p(|x|)}$ con p un polinomio.

Por lo anterior sé que:

$$x \notin \mathcal{L} \text{ ssi } \forall u. M(\langle x, u \rangle) = 0.$$

Al ser una cantidad polinomial y finita de certificados posibles ($2^{p(|x|)}$ más precisamente) puedo computar la no pertenencia de x y la pertenencia de x en \mathcal{L} , o sea $\text{NP} \subseteq \mathcal{R}$.
(decidir)

Puedo concluir que $\text{HALTING} \notin \text{NP}$ porque $\text{NP} \subseteq \mathcal{R}$ y $\text{HALTING} \notin \mathcal{R}$.

✓