Guía 2

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

Ej 1:

1. Probar que los siguientes lenguajes están en P.
   a) COPRIME $= \{\langle a, b\rangle : (a : b) = 1$, es decir, a y b son coprimos$\}$
   b) POWER $= \{\langle a, e, b\rangle : a^e = b\}$
   c) TREE $= \{\langle G\rangle : G$ 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: <a,b>
  while (b ≠ 0):          → Algoritmo de Euclides para calcular MCD (gcd en inglés).
    temp := b             → Hace a la suma log b iteraciones
    b := a mod b
    a := temp
  ret (a = 1)
```

b) $a^e = b$

```
M': <a, e, b>
  c := 1
  base := a
  while (e > 0):
    if (e mod 2 = 1):
      c := c * base
    base := base * base
    e >>2
  ret (c = b)
```

c) Grafo conexo sin ciclos.

DFS ó BFS y chequeo si en algún momento quiero visitar un nodo que ya visité $O(|V|^2)$

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

¡Gracias Mauricio por la corrección!

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 (creería yo) 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 cumplan 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: ⟨x⟩

ret ($x \in L$ || $x \in \Pi$) → x'

Como L y $\Pi \in P$ puedo usar máquinas det. poly para computar si x pertenece a cada una, y como es polinomial sobre la entrada (x) queda de complejidad 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 $L \cup \Pi$:

M: ⟨x⟩

ret ($x \in L$ && $x \in \Pi$) → x'

Complemento:

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

M: ⟨x⟩

ret($\neg$M'(x)) → x'

Ej 3:

3. Probar que los siguientes lenguajes están en NP.

   a) HAMPATH = $\{\langle G, s, t\rangle : G$ es un grafo con dos nodos s y t tales que hay un camino hamiltoniando de s a $t\}$:

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

   c) CLIQUE= $\{\langle G, k\rangle : G$ es un grafo con subgrafo completo de tamaño mayor o igual a $k\}$

   d) k-COLORING = $\{\langle G\rangle : G$ es un grafo que se puede particionar en k conjuntos indepenientes$\}$

   e) ISOMORPHISM = $\{\langle G, H\rangle : G$ y H son dos grafos isomorfos$\}$

   f) SUBGRAPH ISOMORPHISM= $\{\langle G, H\rangle : G$ es un grafo y H es isomorfo a un subgrafo inducido de $G\}$

   g) $\neg$SAT = $\{\langle \phi\rangle : \neg\phi$ 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.

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

⟨V, E⟩ → V=Vertex E=Edges

Certificado: Conjunto de |V| nodos.

Algoritmo: ⟨ G, s, t, u⟩ ← certificado

- Verifico que |u| = |V| → O(|V|)
- Verifico que todos los nodos de u pertenezcan a V → O(|V|²)
- Verifico que el primer nodo sea s y el último sea t. → O(2log|V|)

- Veo que no haya nodos repetidos en u. $\rightarrow O(|V|^2)$
- Veo que $\forall i < n \ (u[i], u[i+1]) \in E$. O sea, que cada nodo esté 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 lo cual es poly y el verificador q' corre el algoritmo también es poly.

b) K-CLIQUE: $\{\langle G \rangle : G$ 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 nodos (si tiene una clique de $K+1$ nodos, tiene una de K). O sea, q.q.d. G tenga una clique de K nodos.

Luego, sé que K es ctte. 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 $\binom{|V|}{K} = O(|V|^K)$ (K ctte.) subconjuntos de K nodos.

Por último para ver que un subconjunto de K nodos 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 ctte 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: $\{\langle G, K \rangle : G$ 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 $K = V/2$, entonces esto me queda $\binom{n}{n/2} = O(n^{n/2})$. 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 nodos. $\rightarrow O(K \cdot \log|V|) = O(|V|)$

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

- Verifica 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 nodos repetidos en c $\rightarrow O(|c|^2) = O(|V|)$

- Chequeo que cada nodo de C esté conectado con todo otro nodo del conjunto C. → $O(|V|^4)$

d) K-COLORING: {⟨G⟩: G es un grafo que se puede particionar en K c/tas independientes (no tienen aristas directas entre nodos)}.

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

Algoritmo: ⟨G, c⟩

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

e) ISOMORPHISM: {⟨G, H⟩ (¿Gran Hermano?): G y H son dos grafos isomorfos (o sea tienen la misma estructura, existe una correspondencia entre sus vértices donde se preserva la adyacencia y no adyacencia)}

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

Algoritmo: ⟨G, H, c⟩

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

f) SUBGRAPH_ISOMORPHISM: {⟨G, H⟩: G es un grafo y H es isomorfo a un subgrafo inducido de G (o sea, tiene un subconjunto de nodos de G con todas las aristas correspondientes a esos nodos en G que los dos extremos estén en H)}.

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

Algoritmo: ⟨G, H, c⟩

- c tiene el doble de elementos que V_H. → $O(|V_H|)$
- $\forall (v,w).\ (v,w) \in C \ \&\& \ v \in V_G \ \&\& \ w \in V_H$. → $O(|V|^2)$
- No hay nodos repetidos en c → $O(|V|^3)$
- 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. → Poly

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

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

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

ret ($\neg(c \models \phi)$). → n es la cant. de var. proposicionales de ϕ.

Ej 4:

4. Probar que los siguientes problemas están en coNP.

   a) PRIME= $\{n : n \in \mathbb{N}$ es primo$\}$
   b) GIRTH= $\{\langle G, k \rangle : G$ es un grafo tal que todos sus ciclos simples tienen k o menos vértices$\}$
   c) TAUTOLOGY= $\{\langle \phi \rangle : \phi$ es tautología$\}$

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

a) PRIME: $\{n : n \in \mathbb{N}$ es primo$\}$ → Ojo! n es un número por lo que poly sería algo $\log n = |n|$

$\overline{PRIME}$: $\{n : n \in \mathbb{N}$ no es primo$\}$

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

Algoritmo: $\langle n, c \rangle$

ret (n mod c == 0)

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

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

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

Algoritmo: $\langle G, K \rangle$

- Las nodos de C deben estar en V, $|C| \leq |V|$, $C[1] = C[z]$, (z: nro. de ult elemento) → $O(|V|^2)$
- No deben haber elementos repetidos (exceptuando el último nodo). → $O(|V|^2)$
- Verificar que $\forall i . 1 \leq i < n, (C[i], C[i+1]) \in E_G$. → $O(|V|^3)$

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

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

Es hacer $\neg SAT$ (lo cual fue demostrado en 3.g).

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

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

Ej 5:

5. Considerar los siguientes lenguajes:

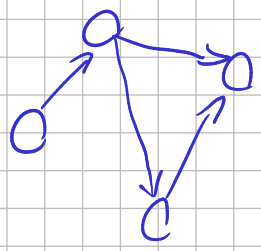
   - PATH = $\{\langle G, s, t, k\rangle : G$ es un digrafo con dos nodos s y t tales que hay un recorrido de longitud menor o igual a k de s a $t\}$:
   - EVEN PATH= $\{\langle G, s, t, k\rangle : G$ es un digrafo con dos nodos s y t tales que hay un recorrido de longitud par y menor o igual a k de s a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

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

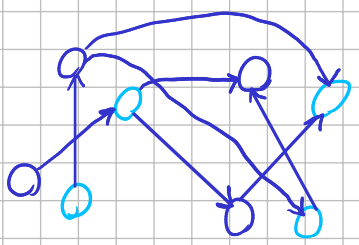
   a) Demostrar que $\langle G, s, t, k\rangle \in$ EVEN-PATH $\iff \langle G', (s, p), (t, p), k\rangle \in$ PATH.

   b) Mostrar que la reducción de EVEN-PATH a PATH implicada por el punto anterior es polinomial.

a) G:



G':



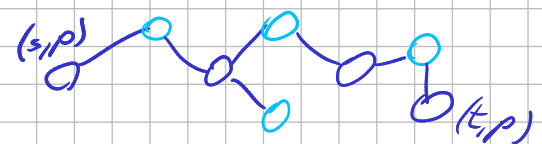
$\Rightarrow$) $\langle G, s, t, k\rangle \in$ EVEN-PATH $\Rightarrow \langle G', (s,p), (t,p), k\rangle \in$ PATH

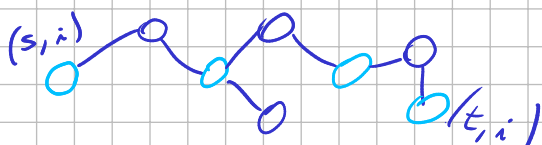
Ahora bien, veamos por construcción cómo se cumple esto.

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

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 cómo se colocan las edges en G' que hay algo de la pinta $\{(s,p), (x_1, i), (x_2, p), \ldots, (x_{k-1}, i), (t,p)\}$ y también otro $\{(s,i), \ldots, (t,i)\}$. O sea que hay un PATH de (s,p) a (t,p) de longitud k.





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

Esta vuelta sale media trivial al haber probado la ida, pero sería algo así:

Tengo G' que tiene un camino de (s,p) a (t,p) de k pasos, ahora ¿cómo se que existe uno de s a t en G? Bastante sencillo! Como la construcción 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, qué significa que $x=y$? Si $x=y$, entonces el camino tiene longitud par ya que por construcción vemos que no hay impares pq si 2 nodos están conectados el segundo elemento de la tupla DEBE tener el otro símbolo en su segunda componente.

b) $\leq_p$?

¿Cómo demuestra si la reducción es polinomial? Veo que f (descrito anteriormente) sea computable polinomialmente.

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

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

M: ⟨G⟩

- Por cada $v \in V$: → $O(|V|)$
  - Crea (v,p) y (v,i) y los seto en la cinta de salida
- Por cada $(v,w) \in E$: → $O(|V|^2)$
  - Crea $((v,i),(w,p))$ y $((v,p),(w,i))$ y los seto en la salida

Ej 6:

6. Considerar los siguientes lenguajes:

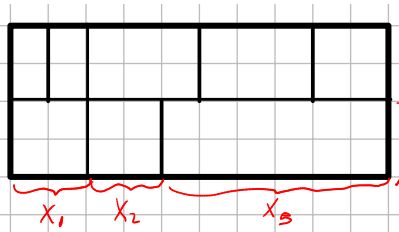
- 2-PARTITION = $\{\langle X\rangle : X$ es un subconjunto finito de los números naturales con $\min_{x\in X}(x) > 2$ que se puede particionar en dos conjuntos X_1, X_2 tales que $X_1 \cap X_2 = \emptyset$, $X_1 \cup X_2 = X$ y $\sum_{x\in X_1} x = \sum_{x\in X_2} x\}$
- 3-PARTITION= $\{\langle X, t\rangle : X$ es un subconjunto finito de los números naturales tal que $\sum_{x\in X} x = \frac{|X|t}{3}$, $\max_{x\in X} x < \lfloor t/2 \rfloor$ y X se puede particionar en $\frac{|X|}{3}$ triplas donde cada una suma $t\}$
- RECTANGLE PACKING= $\{\langle R, r_1, \ldots, r_k\rangle : R$ es un rectángulo que se puede cubrir completamente y sin superposición usando los rectángulos $r_1, \ldots, r_k$ con traslaciones y/o rotaciones$\}$[1] [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 $\langle R, r_1, \ldots, r_{|X|}\rangle$ 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-PARTITION $\iff$ $\langle R, r_1, \ldots, r_{|X|}\rangle \in$ RECTANGLE PACKING.

b) Dada una instancia $\langle X, t\rangle$ de 3-PARTITION, se define la instancia $\langle R, r_1, \ldots, r_{|X|}\rangle$ 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-PARTITION $\iff$ $\langle R, r_1, \ldots, r_{|X|}\rangle \in$ RECTANGLE PACKING.

c) Mostrar que las reducciones implicadas por los puntos anteriores son polinomiales en función de los tamaños de las entradas.

a) $\{x_1, x_2, \ldots, x_{|X|}\}$ sii



→ Z_1 (Rectángulo superior)

→ Z_2 (Rectángulo inferior)

⇒) $\langle X\rangle \in$ 2-PARTITION ⇒ $\langle R, r_1, \ldots, r_{|X|}\rangle \in$ RECTANGLE PACKING

Si $X \in$ 2-P entonces tengo 2 subconjuntos Y_1, Y_2 donde $\sum_{i \in Y_1} y_{1i} = \sum_{i \in Y_2} y_{2i}$.

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 $\sum x_i/2$ y altura 2 como resultado, lo cual es lo que quería y que pertenece a R-P.

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

⇐) $\langle R, r_1, \ldots, r_{|X|}\rangle \in$ R-P ⇒ $\langle X\rangle \in$ 2-P

Por cómo se construye R y r_i (explicada más a detalle en ⇒) 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

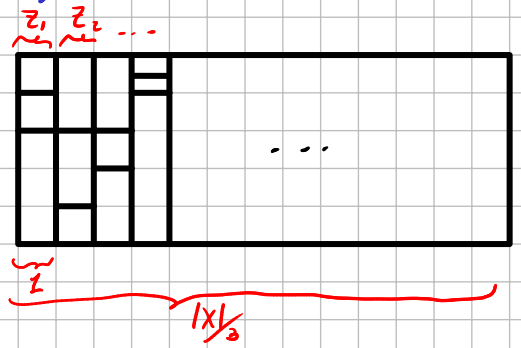
$X_1 \cap X_2 = \emptyset$: Ya que cada x_i se asocia a un r_i diferente y el rectángulo tiene 2 subrectángulos más chicas Z_1 y Z_2 donde se conforman por r_i's distintos

$X_1 \cup X_2 = X$: Como x_i se relaciona a un r_i directamente y se usan todas las r_i's para formar R puedo usar todos los 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 los r_i de Z_1 y a X_2 los r_i de Z_2 y obtener que tengan igual sumatoria de elementos.

b)

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



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

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

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

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

Por construcción r_i corresponde a un x_i, no hay repetidos, se cumple que suman t las triplas 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 triangulitos asociados de a 3 cada Z_i.

c)

$2\text{-}P \leq_p R\text{-}P$

$M: \langle X \rangle$

- 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.

$3\text{-}P \leq_p R\text{-}P$

$M: \langle X, t \rangle$

- 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 $\frac{t}{3} + x_i$.

γ

Ej 7:

complemento.

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?

G "Si pertenece" o "No pertenece" que devuelve la máquina

$$x \in \Pi \leq_p id(x) \in \Pi^c \equiv x \in \Pi^c$$

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

Luego, tengo que las nociones de NP y coNP son sensibles a la etiqueta porque el que acepte un x en $L^c \in coNP$ requiere que haya un $L \in NP$ que rechace esa entrada y lo mismo para cuando rechaza L^c, 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 $= \{\langle G, s, t\rangle : G$ es un digrafo y s y t dos nodos de G tales que hay un recorrido de s 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 $(S, v) \to (R, w)$ es una arista de H si y solo si $w \notin S$, $R = S \cup \{w\}$ y $v \to w$ es una arista de G.

   a) Demostrar que $\langle G, s, t\rangle \in$ HAMPATH $\iff \langle H, (\{s\}, s), (V(G), t)\rangle \in$ CONNECTED.

   b) 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 $\mathcal{P}(V)$ (partes de $V(G)$) por cada nodo de G, o sea tenes un total de $n.2^n$ nodos para H.

Omg!! Hi exponencial!

a)

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

Para ver esto me enfoco 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 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 atravezados por el recorrido una única vez en G.

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

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

b)

Por el comentario hecho en *' puedo ver que la 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:

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

[2] RECURSIVE es el conjunto de problemas decidibles, o bien *recursivos*.

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

¿Qué hacía HALTING?

Recibe un programa X y corre $U(\langle X, X \rangle)$ → Si se cuelga 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: ⟨x⟩ → La f de M devuelve la codificación de una máquina M' construida con x.

ret ⟨M'⟩

M': ⟨z⟩ → Va a ser ignorado (Acá entra la codificación de M' cdo. se hace $U(\langle x, x \rangle)$

```
if (N(x) == 1):
    ret true
else:
    while(true)
ret false
```

→ Te imaginás que llegue acá? Re loco sería.

Ej 10:

10. Probar que NP ⊆ RECURSIVE. Concluir que HALTING ∉ NP.

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

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

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

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

Por lo anterior sé que:

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

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 NP⊆R.
↳(decidir)

Puedo concluir que HALTING ∉ NP porque NP⊆R y HALTING ∉ R.

γ