Teoremas y proposiciones evaluadas en el primer parcial (pero con mis formas de ver la demo)

Proposición 1. *Sea $f : \{0,1\}^* \to \{0,1\}^*$, sea T una función construible en tiempo y sea Γ un alfabeto. Si f es computable en tiempo $T(n)$ por una máquina $M = (\Gamma, Q, \delta)$, entonces f es computable en tiempo $O(\log|\Gamma| \cdot T(n))$ por una máquina $M' = (\Sigma, Q', \delta')$ donde $\Sigma = \{0, 1, \triangleright, \square\}$ es el alfabeto estándar.*

Demo:

$|\Gamma|$ es la cantidad de estados que hay originalmente, para codificar cada estado de Q en Q' se usa el alfabeto Σ (binario), por lo cual conlleva $\log_2|\Gamma|$ bits por estado ($\log_2|\Gamma| = \log|\Gamma|$ bits por convención, aunque en big O no importa).

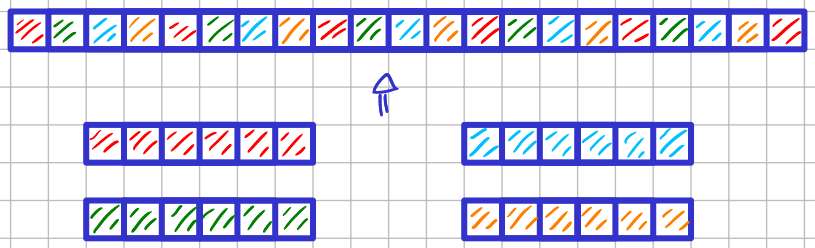
Esto se traslada a δ' también, por lo que lo que antes llevaba un símbolo de Γ ahora lleva $\log|\Gamma|$ de Σ.

Proposición 2. *Sea $f : \{0,1\}^* \to \{0,1\}^*$ y sea T una función construible en tiempo. Si f es computable en tiempo $T(n)$ por una máquina estándar de $k \geq 3$ cintas (entrada, salida y $k-2$ cintas de trabajo), entonces f es computable en tiempo $O(T(n)^2)$ por una máquina de cinta única.*

Demo:

Puedo alternar los símbolos de las k cintas en la cinta única y usar símbolos característicos para indicar dónde está la cabeza de cada cinta.

En la posición i está el caracter $\lceil i/3 \rceil$ de la cinta $i \bmod k$.



Queda en $O(T(n)^2)$ debido a que por cada paso de δ debo primero ubicar cada cabeza para ver que accionar y después modificar cada cabeza como lo indique δ.

Proposición 3. *Sea $f : \{0,1\}^* \to \{0,1\}^*$ y sea T una función construible en tiempo. Si f es computable en tiempo $T(n)$ por una máquina estándar entonces hay una máquina oblivious que computa f en tiempo $O(T(n)^2)$.*

Demo:

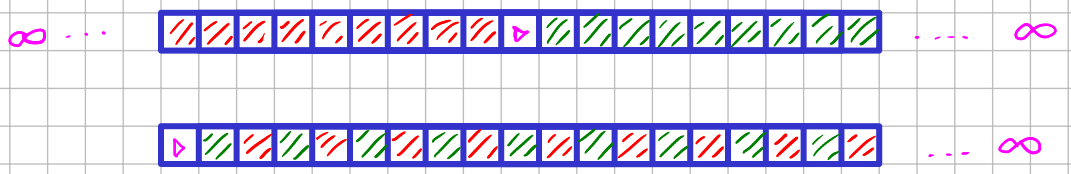
Esto es similar a Prop. 2.

La máquina oblivious se mueve un patrón fijo en cada paso (barre toda la cinta) y cambia la cinta con los cambios que se piden.

Proposición 4. *Sea $f : \{0,1\}^* \to \{0,1\}^*$ y sea T una función construible en tiempo. Si f es computable por una máquina con cintas bi-infinitas en tiempo $T(n)$, entonces f es computable por una máquina estándar en tiempo $O(T(n))$.*

Demo:

Se puede doblar la cinta bi-infinita de manera que rebota en el símbolo $\triangleright$ para ver ambos infinitos.



Teorema 1. *[Turing 1936] halt no es computable.*

$$halt(x): \begin{cases} 1 & \text{si la } x\text{-ésima máquina con entrada } x \text{ termina } (M_x(x)) \\ 0 & \text{si no} \end{cases}$$

Demo:

Sale por diagonalización, tengo que:

	M_1	M_2	...
1	$M_1(1)$		
2		$M_2(2)$	
3			⋱
⋮			

Defino entonces M tal q' $M(x)$ termina s.i. $halt(x)=0$.

$M(\langle M\rangle)$ termina s.i. $halt(\langle M\rangle)=0$ s.i. $M(\langle M\rangle)$ no termina abs!

Teorema 2. *Existe una máquina U que computa la función $u(\langle i,x\rangle) = M_i(x)$. Más aún, si M_i con entrada x termina en t pasos, entonces U con entrada $\langle i,x\rangle$ termina en $c\cdot t\cdot \log t$ pasos, donde c depende solo de i.*

Demo:

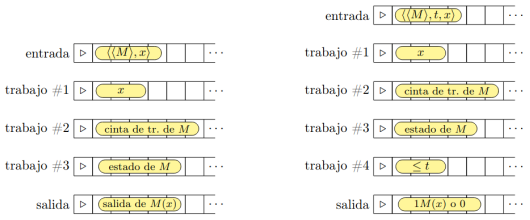


Figura 12: Izquierda: la simulación que hace U (con 3 cintas de trabajo) de M (con una única cinta de trabajo) y entrada x. Derecha: la simulación que hace $\tilde{U}$ (con 4 cintas de trabajo) de M (con una única cinta de trabajo), entrada x hasta el tiempo t.

Pone en cada cinta de trabajo la simulación de la máq. estándar de M. O sea, en #1: La entrada x, #2: Cinta de trabajo de M, #3: Estado de M.
Si #3 = $[q_f]$ termina la ejecución M.
Por cada paso de M busco δ en la entrada de la máquina U.

Teorema 3. *Existe una máquina $\tilde{U}$ que computa la función $\tilde{u}(\langle i,t,x\rangle)$ en tiempo $c\cdot t\cdot \log t$, donde c depende solo de i.*

Demo:

Muy similar a Teo 2 pero con una cinta de trabajo más para llevar registro de i (cant. de pasos en la simulación).

Teorema 4. $\mathbf{P} \subseteq \mathbf{NP}$.

Demo:

Sea $L \in P$. $L(M)$, M máq. det. poly.

Tomás $p(|x|) = 0$. Defino M' det. poly y el certificado $c = \varepsilon$ (cadena vacía), entonces tengo que M' con entrada $\langle x, c \rangle$ copia el comportamiento de M con entrada x.

$x \in L$ sii $M(x) = 1$

sii $M'(\langle x, c \rangle) = 1$

sii $\exists c.\ c \in \{0,1\}^{p(|x|)}$ tal q' $M'(\langle x, c \rangle) = 1$

$\Leftarrow$ Def. de NP.

Teorema 6. *Existe una máquina no-determinística NU que tal que NU acepta $(\langle i, x \rangle)$ sii N_i acepta x y si N_i corre en tiempo $T(n)$ entonces $NU(\langle i, x \rangle)$ decide si N_i acepta o rechaza x en $c \cdot T(|x|)$ pasos, donde c depende solo de i.*

Demo: Similar Teo. 2. No tiene el logaritmo porque...

Teorema 7. *La relación $\leq_p$ es transitiva.*

Demo:

Sea $L \leq_p L'$ vía f y $L' \leq_p L''$ vía g q.q. $L \leq_p L''$

$x \in L$ sii $f(x) \in L'$ sii $g(f(x)) \in L''$

Ahora q.q. $g \circ f$ es computable en tiempo poly a $|x|$.

Sea M_f, M_g tal q' M_f computa f en poly y M_g que computa g en poly.

$M_{g \circ f}$: $\langle x \rangle$

$y := M_f(x)$ $\rightarrow O(n^c)$ $\rightarrow$ A lo sumo su salida es polinomial respecto $n = |x|$

ret $M_g(y)$ $\rightarrow O((n^c)^d) = O(n^{cd})$ $\rightarrow$ A lo sumo es poly respecto $|y|$.

Teorema 8. *Si $\mathbf{NP\text{-hard}} \cap \mathbf{P} \neq \emptyset$, entonces $\mathbf{P} = \mathbf{NP}$.*

Demo:

Si $L \in$ NP-hard $\cap P$, entonces puedo tomar cualquier $L' \in NP$ y reducirlo a L, por lo cual tenga una f computable poly. tal que:

$x \in L'$ sii $f(x) \in L$

Entonces a la $L'(M_{L'})$ la decido como:

$M_{L'}: \langle x \rangle$

$y_i = M_f(x)$ $\rightarrow O(n^c)$

$M_L(y)$ $\rightarrow O(n^{cd})$

} Es una máq. det. poly por lo q' $L' \in P$.

O sea $P = NP$.

Teorema 9. *Si $\mathcal{L} \in$ NP-completo, entonces $\mathcal{L} \in$ P sii P = NP.*

Dem:

$\Rightarrow$) $L \in P \Rightarrow P = NP$

Si L es NP-C y a su vez $L \in P$, lo única que agrega que $L \in$ NP-C (como L ya era NP (porque $P \subseteq NP$)) es que $L \in$ NP-hard!

Por esto, sale por el Teo. 8.

$\Leftarrow$) $P = NP \Rightarrow L \in P$

Si P = NP, entonces todos los problemas de NP se pueden resolver en tiempo poly, en particular también todos los NP-C.

Problema: TMSAT

$$\text{TMSAT} = \{\langle y, x, 1^n, 1^t \rangle \mid \exists u \in \{0,1\}^n \ M_y(xu) = 1 \text{ en a lo sumo } t \text{ pasos}\}$$

Proposición 6. TMSAT $\in$ NP-completo.

Dem: $L \in NP$

$x \in L$ sii $\exists u \in \{0,1\}^{p(|x|)} \ M(x,u) = 1$ (def. de L en NP)

$x \in L$ sii $f(x) \in TMSAT$ (def $L \leq_p$ TMSAT)

La f de la reducción sería:

$f(x) = \{\langle M \rangle, x, 1^{p(|x|)}, 1^{q(|x|+p(|x|))}\}$ $\rightarrow$ Computable en tiempo poly.

$f(x) \in TMSAT$ sii $\langle \langle M \rangle, x, 1^{p(|x|)}, 1^{q(|x|,p(|x|))} \rangle \in TMSAT$

sii $\exists u \in \{0,1\}^{p(|x|)} . M(x,u) = 1$ en $\leq q(|xu|) = q(|x|+p(|x|))$ pasos

sii $x \in L$

Proposición 7. *Sea $\varphi(p_1, \ldots, p_n)$ una fórmula booleana. Si v y v' son dos valuaciones tales que $v(p_i) = v'(p_i)$ para $i \in \{1, \ldots, n\}$, entonces $v \models \varphi(p_1, \ldots, p_n)$ sii $v' \models \varphi(p_1, \ldots, p_n)$.*

Dem:

$V(p_i) = V'(p_i)$ significa que la valuación evalúa a las variables proposicionales con el mismo valor. La prop. sale inmediata.

Teorema 10. SAT, 3SAT $\in$ NP.

Demo:

$M: \langle x, u \rangle$

- x representa una fórmula booleana
- Si: $x = \langle \varphi \rangle$ y φ tiene n var. proposicionales:
  - ret $u_{\wedge n} \models \varphi$
    tomo las primeras n elementos de u.

Proposición 8. *Para toda $F : \{0,1\}^\ell \to \{0,1\}$ existe una fórmula booleana $\varphi_F(p_1, \ldots, p_\ell)$ en CNF tal que $v \models \varphi_F$ sii $F(v) = 1$ para todo $v \in \{0,1\}^\ell$. Más aún, φ_F se computa en tiempo polinomial a partir de $\langle F \rangle$ y tiene tamaño $O(\ell 2^\ell)$.*

Demo:

Concateno todas las posibles combinaciones de ℓ bits ($\ell 2^\ell$) (ℓ bits por valuación, todas las valuaciones) de manera que si $F(v) = 0$, haya un $\vee(p_1 \wedge p_2 \ldots \wedge p_n)$ y $(p_1 \wedge \ldots \wedge p_n)$ está en φ_F.

Corolario 1. *Para toda $F : \{0,1\}^\ell \to \{0,1\}^k$ existe una fórmula booleana $\varphi_F(p_1, \ldots, p_{\ell+k})$ en CNF tal que $uv \models \varphi_F$ sii $F(u) = v$ para todo $u \in \{0,1\}^\ell$, $v \in \{0,1\}^k$. Más aún, φ_F se computa en tiempo polinomial a partir de $\langle F \rangle$ y tiene tamaño $O((\ell + k)2^{\ell+k})$.*

Demo:

Defino $G: \{0,1\}^{\ell+k} \to \{0,1\}$ tal que $G(u,v) = 1$ sii $F(u) = v$ y aplicar Prop 8.

Teorema 11. SAT $\in$ NP-hard. → Teorema de Cook Levin

Idea de la demo (no explico toda pq es muy técnica):

$x \in L$ sii $\exists u \in \{0,1\}^{p(|x|)} M(x,u) = 1$

sii Existe $u \in \{0,1\}^{p(|x|)}$ y existe un cómputo $C_0, \ldots, C_{t(|xu|)}$ a partir de la entrada xu

Con el uso de mini-configuraciones $z_0, \ldots, z_m \in \{0,1\}^K$ con K dependiente de la máq.

Con la entrada (xu) quiero ver con fórmulas de tamaño polinomial a la entrada puedo simular el cómputo de la M det. que decide a L.

Las fórmulas son:

1. La que representa la cinta de entrada. (xu)
2. Configuración inicial de M a partir de (xu) $(q_0, \text{símbolo})$
3. Evolución en un paso de z_i a z_{i+1} (δ)
4. Configuración final de M aceptadora. $(q_f$, símbolo en la cinta de trabajo $= 1)$

Corolario 2. SAT, 3SAT ∈ **NP-completos.** → Sale del Teo. 10 y 11

Proposición 9. INDSET ∈ **NP-completo.**

Demo:

Ver que INDSET ∈ NP es medianamente fácil, ahora ver q' INDSET es NP-hard tenés que hacerlo por def (nunca) ó por reducción poly de un NP-hard q' sepamos.

Veamos que $3SAT \leq_p INDSET$:

$\varphi = (l_{11} \vee l_{12} \vee l_{13}) \wedge \dots \wedge (l_{m1} \vee l_{m2} \vee l_{m3})$

A la suma tengo 3m vértices (si todos los literales son ≠), uno por cada variable

Defino una arista entre z y z' si: están en la misma cláusula de la fórmula ó la variable ligada al nodo z es la negación de la variable ligada al nodo z'

Esta sería la f de que:

$$x \in 3SAT \text{ sii } f(x) \in INDSET$$

(la demo de que funciona queda en tu imaginación)

Proposición 10. CAMHAM ∈ **NP-completo.**

Teorema 12. TSP ∈ **NP-completo.**

Proposición 11. KNAPSACK ∈ **NP-completo.**

Teorema 13. *Si* **P = NP** *entonces* **ExpTime = NExpTime.**

Demo:

1) $Exptime \subseteq NExptime$

Esta ya se cumple sin que P=NP.

2) $NExptime \subseteq Exptime$

Toma $L \in NExptime$ y $L(N)$ tal q' N es no-det y corre en $c2^{n^c}$.

Considero $L_{pad} = \{\langle x, 1^{2^{|x|^c}} \rangle : x \in L\}$. (L con padding)

Veamos q' $L_{pad} \in NP$:

Como la entrada de L_{pad} es una cadena exp. con respecto a |x|, puedo correr a N con x por $2^{|x|^c}$ pasos pq' es poly con respecto $|\langle x, 1^{2^{|x|^c}} \rangle|$

Jerarquía de Tiempos Determinísticos

Teorema 14. *Si f, g son construibles en tiempo y cumplen que $f(n) \cdot \log f(n) = o(g(n))$ entonces* $\mathbf{DTIME}(f(n)) \subsetneq \mathbf{DTIME}(g(n))$.

Demo:

A ver, esta va a ser lenta y detallada pq' me cuesta entenderla.

La idea principal es demostrarlo por diagonalización que hay lenguajes que están en $DTime(g(n))$ pero no en $DTime(f(n))$.

Declaro D det. tal que:

$D: \langle x \rangle$

Calcula $n = |x|$ y $t = g(n)$ → $O(g(n) + n) = O(g(n))$

Simula $U^*(\langle x, x \rangle)$ por t pasos → $O(g(n))$

Si $U(\langle x, x \rangle)$ no terminó en a lo sumo t pasos: ret 0 → $O(1)$

Si no: ret $\neg(U(\langle x, x \rangle))$ → $O(1)$

$L(D) \in DTime(g(n))$ es inmediato.

Ahora vamos a ver por absurdo la prueba:

Supongo q' $L(D) \in DTime(f(n))$. Sea $L(M) = L(D)$, donde M termina en a lo sumo $O(f(|x|))$ pasos ($c \cdot f(|x|)$) (acá $|x| \geq n_0$ q' es el punto donde siempre se vuelve mayor)

Por el Teo 2 sé que $U(\langle x, x \rangle)$ simula $M_x(x)$ en $c' \cdot c \cdot f(|x|) \cdot \log(c \cdot f(|x|))$ pasos (teo 2 dice q' $U(\langle x, x \rangle)$ corre en $k \cdot z \cdot \log z$ donde z es el tiempo de cómputo de x)

Luego, existen d y $n_1 \geq n_0$ tal q' $\forall n \geq n_1$:

$$c' c f(n) \cdot \log(c \cdot f(n)) \leq d f(n) \log(f(n)) \leq g(n)$$

Tomo x tal q' $|x| \geq n_1$ y $M_x = M$. La existencia de este x es garantizada pq' se pueden hacer ∞ máq. que hagan lo mismo.

Ahora bien, $U(\langle x, x \rangle)$ termina en $g(|x|)$ pasos. Luego, $D(x) = \neg(M_x(x)) = \neg(M(x)) = 0$

máq x $L(D) = L(M)$

OK, tiene sentido, te armas una máq. q' corra en $g(|x|)$ pasos, forzas a una máq. a identificar el mismo lenguaje pero en $\log(f(|x|)) \cdot f(|x|)$ pasos y como siempre termina, siempre pasa que da una contradicción con el lenguaje que suponía que decidía.

¿Que pasa si le meto D_x? No me da una contradicción también!

$D(\langle D_x \rangle) = U(\langle D_x \rangle, \langle D_x \rangle)$ por t pasos = $g(|x|)$ pasos.. Opa! Esto no termina, requiere para $\log(g(|x|)) \cdot g(|x|)$ pasos para terminar

Jerarquía de Tiempos no-determinísticos

Teorema 15. *Si f, g son construibles en tiempo, no decrecientes y cumplen que $f(n+1) = o(g(n))$, entonces* $\mathbf{NDTIME}(f(n)) \subsetneq \mathbf{NDTIME}(g(n))$.

Demo:

Deja espacio para verlo después.

Teorema 16 (Ladner)**.** *Si* **P** $\neq$ **NP** *entonces existe* $\mathcal{L}$ *tal que* $\mathcal{L} \in$ **NP**, $\mathcal{L} \notin$ **P**, *y* $\mathcal{L} \notin$ **NP-completo**.