Ma50b Complejidad Computacional **10 de Mayo de 2010**

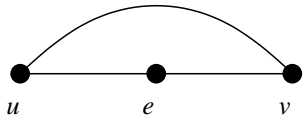
Pauta Control 2

Prof. Cátedra: M. Kiwi *Prof. Auxiliar: R. Bricenño*

PROBLEMA 1:

(i).- Veamos primero que DS $\in$ NP. En efecto, dada una instancia $\langle G=(V,E),k\rangle$ de DS, basta considerar el certificado $D \subseteq V$ y el proceso de verificación que consiste en comprobar que $|D| \leq k$ y que para todo $u \in V \setminus D$ existe un nodo $v \in D$ tal que $vu \in E$. En una RAM, el proceso de verificación se puede implementar facilmente en tiempo $O(|V| \cdot |D| \cdot |E|) = O(|V|^2 \cdot |E|)$, es decir polinomial en el tamaño de $\langle G=(V,E),k\rangle$.

Veamos ahora que DS es NP-duro. Bastará probar que VC $\leq_P$ DS, donde VC corresponde al lenguaje *vertex-cover*. Dado $\langle G=(V,E),k\rangle$ instancia de VC, consideramos el grafo $G'=(V',E')$ sobre el conjunto de nodos $V'=V\cup E$ y donde uv, eu, y ev están en E' si $e=uv$ está en E. En otros palabras, construimos G' a partir de G reemplazando cada arco $e=uv \in E$ por la siguiente gadget:



En una RAM, la construcción de G' dado G se puede hacer facilmente en tiempo $O(|V|+|E|)$, i.e. en tiempo polinomial en el tamaño de la entrada $\langle G,k\rangle$.

Afirmamos que G posee un recubrimiento de nodos $S \subseteq V$ de tamaño a lo más k si y solo si G' posee un conjunto dominante de nodos $D \subseteq V'$ de tamaño a lo más $k+a$, donde a es el número de vértices aislados de G. En efecto, si S es un recubrimiento de nodos en G y A el conjunto de nodos aislados de G, entonces $S\cup A$ es un conjunto dominante en G'. Lo anterior, porque si u es un nodo de G' tal que $u \notin S\cup A$, entonces existe un $v \in S$ tal que $e=uv \in E$ (porque S es un recubrimiento de G). Luego, $v \in S$ domina a u, i.e. u está dominado por $S\cup A$. Supongamos ahora que D es un conjunto dominante en G'. Si existe un nodo $e=uv \in D\cap E$, reemplazamos D por $D\cup\{u\}\setminus\{e\}$, hasta eleminar todos los nodos e de G' que estaban en $D\cap E$. El conjunto D' que se obtiene sigue siendo dominante en G' porque cualquiera de los nodos de un gadget domina a todo el resto de los nodos del gadget. Observar que el tamaño del conjunto que se obtiene es a lo mas el tamaño del conjunto original, es decir a lo mas k. Afirmamos que $D'\setminus A$ es un recubrimiento de nodos en G, donde A es el conjunto de nodos aislados de G. En efecto, sea $e=uv$ un arco de G. Como el nodo e esta dominado en G', debe estarlo por los nodos u o v, i.e. u o v están en $D'\setminus A$.

En resumen, la transformación que a $\langle G,k\rangle$ le asocia $\langle G',k+a\rangle$, donde a es el número de vértices aislados de G, es una reducción a tiempo polinomial de VC a DS, i.e. VC $\leq_P$ DS.

(ii).- Afirmamos que el algoritmo recursivo CICLOENTRADA que se detalla más abajo, retorna Aceptar en la entrada $(C,0^n,0^n,2^{n+1})$ si y solo si el circuito C posee ciclos de entrada.

Algorithm 1 Algoritmo para decidir si un circuito Booleano C posee ciclos de entrada.

```
1: procedure CICLOENTRADA(C, x⃗, y⃗, t)
2:     ▷ C es circuito Booleano en las variables x⃗, y⃗ ∈ {0,1}^n y t ∈ ℕ.
3:     if t = 0 ∧ x⃗ = y⃗ then
4:         return aceptar
5:     else
6:         return rechazar
7:     end if
8:     if t = 1 ∧ C(x⃗, y⃗) = 1 then
9:         return aceptar
10:    else
11:        return rechazar
12:    end if
13:    for m⃗ ∈ {0,1}^n do
14:        if CICLOENTRADA(C(x⃗, m⃗), ⌈t/2⌉) = 1 ∧ CICLOENTRADA(C(m⃗, y⃗), ⌊t/2⌋) = 1 then
15:            return aceptar
16:        end if
17:        return rechazar
18:    end for
19: end procedure
```

Para probar la afirmación, observar primero que si CICLOENTRADA$(C,0^n,0^n,2^{n+1})$ retorna *aceptar* es porque C posee un ciclo de entrada. Supongamos que C posee un ciclo de entrada $\omega_0 = 0^n, \omega_1, \ldots, \omega_{m+1} = 0^n$. Sin pérdida de generalidad, asumimos que m es mínimo. Luego, todos los ω_i con $1 \leq i \leq m$ deben ser distintos, i.e. $m \leq 2^n$. Por un argumento similar al usado en la demostración del Teorema de Savitch, se tiene que si CICLOENTRADA hace una recursión de profundidad $n+1$, entonces necesariamente determinará que existe un ciclo de entrada de largo $m+1 \leq 2^{n+1}$. Sigue que CICLOENTRADA$(C,0^n,0^n,2^{n+1})$ retorna *aceptar* si C posee un ciclo de entrada.

Finalmente, observar que en la entrada $(C,0^n,0^n,2^{n+1})$, el algoritmo CICLOENTRADA hace una recursión de profundida a lo más $n+1$, y ocupa espacio $O(n)$ en cada nivel de la recursión.

Resumiendo, se tiene que Ciclo $\in$ PESPACIO.

(iii).- Supongamos que DESPACIO$(n) =$ P. Veremos que PESPACIO $\subseteq$ P. En efecto, sea $L \in$ PESPACIO y M la máquina de Turing a espacio polinomial $p(n)$ que decide L. Sin pérdida de generalidad podemos suponer que $p(n) \geq n$. Sea # un caracter que no está en el alfabeto de L. Definimos

$$L' \quad = \quad \{\omega\#^m : \omega \in L, m = p(n) - n\} .$$

Sea M' la máquina de Turing que en la entrada $\omega\#^m$ verifica que $m = p(n) - n$ y posteriormente simula M en ω, aceptando si M acepta, y rechazando si M rechaza. Observar que M' decide L' en espacio $S(n) = n$. Sigue que $L' \in$ DESPACIO$(n) =$ P. Hemos concluido que PESPACIO $\subseteq$ P, luego PESPACIO $=$ P. Por nuestro supuesto inicial se concluye que PESPACIO $=$ DESPACIO(n), contradicción.

PROBLEMA 2:

(i).- Veamos que PESPACIO $\subseteq \mathrm{P}^{\mathrm{TQBF}}$. Lo anterior es inmediato del hecho que TQBF es PESPACIO-duro bajo reducciones en tiempo polinomial. En efecto, sea $L \in$ PESPACIO, sabemos que existe una máquina de

Turing M a tiempo polinomial que calcula una función de reducción f tal que $\omega \in L$ si y solo si $f(\omega) \in \text{TQBF}$. Sigue que la máquina de Turing con oráculo TQBF que en la entrada ω simula M, calcula $f(\omega)$, utiliza el oráculo para determinar si $f(\omega)$ está en TQBF y acepta en caso afirmativo, decide L en tiempo polinomial. Por definición de P^{TQBF} se concluye que $L \in \text{P}^{\text{TQBF}}$.

Veamos ahora que $\text{P}^{\text{TQBF}} \subseteq \text{NP}^{\text{TQBF}}$. En efecto, sea $L \in \text{P}^{\text{TQBF}}$. Sigue que existe una máquina de Turing determinista con oráculo TQBF que en tiempo polinomial decide L. Toda máquina de Turing determinista a tiempo polinomial puede ser simulada por una máquina de Turing no–determinista en el mismo tiempo, y es fácil ver que lo anterior se mantiene si la máquina de Turing determinista tiene acceso a un oráculo y la máquina de Turing no–determinista puede acceder al mismo oráculo. Luego, por definición de NP^{TQBF} se concluye que $L \in \text{NP}^{\text{TQBF}}$.

Finalmente, veamos que $\text{NP}^{\text{TQBF}} \subseteq \text{PESPACIO}$. En efecto, si $L \in \text{NP}^{\text{TQBF}}$, entonces existe una máquina de Turing no–determinista con oráculo TQBF que en tiempo polinomial decide L. Toda máquina de Turing no–determinista a tiempo polinomial puede ser simulada por una máquina de Turing determinista a espacio polinomial, y es fácil ver que lo anterior se mantiene si la máquina de Turing no–determinista tiene acceso a un oráculo y la máquina de Turing a espacio polinomial puede acceder al mismo oráculo. Pero como TQBF $\in$ PESPACIO, una máquina de Turing a espacio polinomial puede decidir por si misma TQBF. Sigue que toda máquina de Turing a espacio polinomial con oráculo en PESPACIO puede ser simulada por una máquina de Turing a espacio polinomial. De la anterior discusión, se concluye $L \in \text{PESPACIO}$.

De lo demostrado sigue que $\text{P}^{\text{TQBF}} = \text{PESPACIO} = \text{NP}^{\text{TQBF}}$.

(ii.1).- Sea $M^?$ la máquina de Turing no–determinista con oráculo A que en la entrada 0^n adivina de manera no–determinista un $x \in \Sigma_A^*$ de largo n, consulta al oráculo acerca de la pertenencia de x en A, acepta si la respuesta del oráculo es *si*, y rechaza en caso contrario. Es fácil ver que M^A decide L_A, y que $M^?$ es una máquina de Turing no–determinista a tiempo polinomial (de hecho, a tiempo lineal).

(ii.2).- Sea $\widetilde{M}_i^?$ la máquina de Turing que simula $M_i^?$ por $(n+2)^i$ pasos (utilizando una cinta auxiliar donde mantiene un contador). Si $M_i^?$ acepta (respectivamente rechaza) en a lo más $(n+2)^i$ pasos, entonces $\widetilde{M_i^?}$ acepta (respectivamente rechaza). Si $M_i^?$ no se detiene en $(n+2)^i$ pasos, entonces $\widetilde{M}_i^?$ rechaza. Como $\widetilde{M}_i^?$ es una máquina de Turing con acceso a un oráculo y a tiempo polinomial, también debe estar en la lista $M_1^?, M_2^?, \ldots$. Sigue que $L(\widetilde{M}_i^A) \in \mathcal{C}_A$. Por otro lado, sea $L = L(M_i^A) \in \mathcal{C}_A$ donde la máquina de Turing M_i^A es a tiempo polinomial $p_i(n)$. Como hay un número arbitrario de formas de codificar la misma máquina de Turing, hay una infinidad de valores de j para los que $L(M_j^A) = L$. Para alguno de dichos j se debe tener que $(n+2)^j \geq p_i(n)$ para todo $n \in \mathbb{N}$. Luego, la simulación de M_i^A que realiza $\widetilde{M}_j^A$ llega a término y por lo tanto $L(\widetilde{M}_j^A) = L$. En resumen, $\mathcal{C}_A = \left\{ L(\widetilde{M}_i^A) : i \in \mathbb{N} \right\}$.

(ii.3).- Observemos primero que la secunecia de n_i's es estrictamente creciente en i. Por la forma en que se definió el oráculo A sigue que para $\omega \in \{0,1\}^*$ tal que $|\omega| \leq n_{i-1}$ se tiene que $\omega \in A$ si y solo si $\omega \in A_{i-1}$. Además, notar que cualquier consulta ω tal que $|\omega| \geq n_{i-1}$ que haga M_i^A en la entrada 0^{n_i} es tal que $\omega \notin A$. Sigue que M_i^A en la entrada 0^{n_i} hace exactamente lo mismo que $M_i^{A_{i-1}}$ en la entrada 0^{n_i}.

(ii.4).- Por contradicción. Supongamos que $L_A \in \text{P}^A$. Sigue que existe un $i \geq 1$ tal que M_i^A decide L_A en tiempo $(n+2)^i$.

Supongamos que $0^{n_i} \in L_A$, i.e. M_i^A acepta 0^{n_i}. Por (ii.3) sigue que $M_i^{A_{i-1}}$ también acepta 0^{n_i}. Pero entonces A_i no contiene palabras de largo n_i. Por definición de A y dado que los n_i's son estrictamente crecientes, sigue que A no contine palabras de largo n_i. Luego, $0^{n_i} \notin L_A$, contradicción.

Supongamos ahora que $0^{n_i} \notin L_A$, i.e. M_i^A rechaza 0^{n_i}. Por (ii.3) sigue que $M_i^{A_{i-1}}$ también rechaza 0^{n_i}. Afirmamos que A_i contiene palabras de largo n_i. En efecto, como $M_i^{A_{i-1}}$ rechaza 0^{n_i} el conjunto A_i se obtiene a partir de A_{i-1} agregando todas aquellas palabras en $\{0,1\}^{n_i}$ que no fueron consultadas al oráculo durante la ejecución de $M_i^{A_{i-1}}$ en la entrada 0^{n_i}. Como $\{0,1\}^{n_i}$ tiene cardinal 2^{n_i} y $M_i^{A_{i-1}}$ en la entrada 0^{n_i} puede haber consultado al oráculo a lo más $(n_i+2)^i < 2^{n_i}$ palabras, necesariamente se agregaron a A_{i-1} palabras de largo n_i al obtener A_i. Como $A_i \subseteq A$, sigue que A también contiene palabras de largo n_i. Luego, $0^{n_i} \in L_A$, contradicción.

En resumen, $L_A \notin \mathrm{P}^A$.