Ma5201 Complejidad Computacional	**01 de Junio de 2013**

Pauta Control 2

Prof. Cátedra: M. Kiwi — *Prof. Auxiliar: P. Muñoz, D. Salas*

PROBLEMA 1:

(i).- Si establecemos que A_{mT} mucho a uno reduce a H (respectivamente, a $\overline{H}$), tendremos que H no puede ser co-reconocible (respectivamente, reconocible) dado que en caso contrario nos quedaría que A_{mT} sería co-reconocible, luego decidible, contradiciendo la indecidibilidad de A_{mT}.

Veamos primero que $A_{mT} \leq_m H$. Para ello, consideremos la reducción $f(\cdot)$ que a una instancia $\langle M, \omega\rangle$ de A_{mT} le asocia $\langle R_{M,\omega}\rangle$ donde $R_{M,\omega}$ procede de la siguiente forma:

> **input**: σ.
> Vía el Teorema de la Recurrencia, obtener descripción $\langle R_{M,\omega}\rangle$ de si misma;
> **if** $\sigma \neq \langle R_{M,\omega}\rangle$ **then** RECHAZAR;
> **if** *simulación de M en* ω *acepta* **then**
> | ACEPTAR;
> **else**
> ⌊ RECHAZAR;

Es fácil ver que $L_{R_{M,\omega}}$ es igual a $\emptyset$ o a $\{\langle R_{M,\omega}\rangle\}$. Además se verifica que $\langle M, \omega\rangle \in A_{mT}$ si y sólo si $L_{R_{M,\omega}} = \{\langle R_{M,\omega}\rangle\}$ (en particular, $\langle R_{M,\omega}\rangle \in H$). Falta establecer que $f(\cdot)$ es totalmente recursiva. Para ello, hay que argumentar que $R_{M,\omega}$ como se definió, es efectivamente una máquina de Turing. El Teorema de la Recurrencia garantiza que es posible para una máquina de Turing obtener una descripción de si misma. Más aún, el referido teorema es constructivo, en el sentido que su demostración señala como calcular la descripción de una máquina de Turing que utilice su propia descripción para calcular alguna función totalmente recursiva que dependa de una descripción de una máquina de Turing y de otra entrada. Resumiendo, $f(\cdot)$ es totlamente recursiva, por lo que A_{mT} mucho-a-uno reduce a H.

Veamos ahora que $A_{mT} \leq_m \overline{H}$. Procedemos como en el párrafo anterior, pero ahora definimos $R'_{M,\omega}$ como:

> **input**: σ.
> Vía el Teorema de la Recurrencia, obtener descripción $\langle R'_{M,\omega}\rangle$ de si misma;
> **if** $\sigma = \langle R'_{M,\omega}\rangle$ **then** RECHAZAR;
> **if** *simulación de M en* ω *acepta* **then**
> | ACEPTAR;
> **else**
> ⌊ RECHAZAR;

Ahora se verifica que $\langle M, \omega\rangle \in A_{mT}$ si y sólo si $L_{R'_{M,\omega}} = \left\{\sigma : \sigma \neq \langle R'_{M,\omega}\rangle\right\}$ (en particular, $\langle R'_{M,\omega}\rangle \in \overline{H}$). La aplicación que a $\langle M, \omega\rangle$ le asocia $\langle R'_{M,\omega}\rangle$ es totalmente recursiva por las mismas razones por las cuales la reducción del párrafo anterior lo era. Resumiendo, A_{mT} mucho-a-uno reduce a $\overline{H}$.

(ii).- Veamos que 3SAT $\leq_P$ OneIn3SAT. Consideremos una instancia $\langle\varphi\rangle$ de 3SAT, donde $\varphi = \wedge_{i=1}^{m} C_i$ es fórmula booleana en forma 3CNF en las variables $x_1,\ldots,x_n$. Sin pérdida de generalidad, podemos asumir que en ninguna claúsula aparecen literales repetidos (de lo contrario, se eliminan las repeticiones dejando sólo una ocurrencia del literal). También podemos suponer que ninguna claúsula contiene una variable y su negación (de lo contrario la claúsula es siempre cierta y se puede eliminar). Sea C'_i la claúsula que se obtiene al reemplazar el j-ésimo literal de la claúsula C_i, $1 \le j \le 3$ e $i \in [m]$, por la variable $z_{i,j}$. Sea $i \in [m]$, $1 \le j \le 3$. Asociamos a cada variable $z_{i,j}$ una variable auxiliar $z'_{i,j}$. Si el j-ésimo literal de C_i es y ($y = x_k$ o $y = \overline{x_k}$ para algún $k \in [n]$), entonces definimos

$$C''_{i,j} = \overline{y} \vee z_{i,j} \vee z'_{i,j}.$$

Observar que $C''_{i,j}$ evalúa a verdadero ssi cuando $y = 0$, se tiene que $z_{i,j} = z'_{i,j} = 0$, y cuando $y = 1$, se tiene que $z'_{i,j} = \overline{z_{i,j}}$ con $z_{i,j}$ cualquiera. Finalmente, definimos $\Psi = \wedge_{i=1}^{m} C'_i \wedge \bigwedge_{i,j} C''_{i,j}$. Notar que Ψ está en forma 3CNF, y que sus variables son x_i con $i \in [n]$, $z_{i,j}$ y $z'_{i,j}$ con $i \in [m]$ y $1 \le j \le 3$.

Afirmamos que $\langle\varphi\rangle \in$ 3SAT ssi $\langle\Psi\rangle \in$ OneIn3SAT. En efecto, sea $a = (a_1,\ldots,a_n) \in \{0,1\}^n$ una asignación que satisface φ. Sigue que por cada claúsula C_i de φ existe un literal y, digamos el j_i-ésimo literal de C_i, tal que al evaluarlo en la asignación a toma el valor 1. Fijamos entonces $z_{i,j_i} = 1$ y $z'_{i,j_i} = 0$. Para todo $j' \ne j_i$, fijamos $z_{i,j'} = 0$ y hacemos $z'_{i,j'} = 1$ ssi el j'-ésimo literal de C_i evalúa a 1. A la variable x_i de Ψ le asignamos el valor a_i. Por construcción, el único literal que evalúa a verdadero en la claúsula C'_i es el j_i-ésimo. Falta ver que las claúsulas $C''_{i,j}$ también se satisfacen y que sólo uno de sus literalres evalúa a verdadero. Supongamos que $C''_{i,j} = \overline{y} \vee z_{i,j} \vee z'_{i,j}$. Si $j = j_i$, entonces $\overline{y} = 0$, $z_{i,j_i} = 1$ y $z'_{i,j_i} = 0$, por lo que uno y sólo un literal de $C''_{i,j}$ será verdadero. Si $j \ne j_i$ y $\overline{y} = 1$, es porque el j-ésimo literal de C_i evalúa a 0, por lo que se tendrá que $z_{i,j} = 0$ y $z'_{i,j} = 0$, i.e. uno y sólo un literal de $C''_{i,j}$ será verdadero. Si $j \ne j_i$ y $\overline{y} = 0$, es porque el j-ésimo literal de C_i evalúa a 1, por lo que se tendrá que $z_{i,j} = 0$ y $z'_{i,j} = 1$, i.e. uno y sólo un literal de $C''_{i,j}$ será verdadero. En resumen, $\langle\Psi\rangle \in$ OneIn3SAT. Veamos ahora que también se tiene el converso, es decir, que si $\langle\Psi\rangle \in$ OneIn3SAT, entonces $\langle\varphi\rangle \in$ 3SAT. Consideremos a_i, $b_{i,j}$, y $b'_{i,j}$, asignaciones booleanas a las variables x_i, $z_{i,j}$, y $z'_{i,j}$, respectivamente, y que hacen verdadera a Ψ. Sean $a = (a_i)_i$, $b = (b_{i,j})_{i,j}$, y $b' = (b'_{i,j})_{i,j}$. Afirmamos que a es una asignación que satisface φ. En efecto, consideremos $i \in [m]$ cualquiera. Como C'_i evalua a verdadero en (a,b,b'), se tiene que existe algún $1 \le j_i \le 3$ tal que $z_{i,j_i} = 1$. Sea y el j_i-ésimo literal de C_i. Como uno y sólo un literal de $C'_{i,j_i} = \overline{y} \vee z_{i,j_i} \vee z'_{i,j_i}$ evalúa a verdadero en (a,b,b'), debe tenerse que y evalúa a verdadero en a, por lo que C_i también evalúa a verdadero en a. Dado que i es arbitrario, hemos establecido que a satisface φ.

Falta ver que la reducción que a una instancia $\langle\varphi\rangle$ de 3SAT le asocia una instancia $\langle\Psi\rangle$ de OneIn3SAT es a tiempo polinomial. Para ello, basta notar que la construcción de Ψ requiere ciclar sobre los literales de φ (a lo más $|\langle\varphi\rangle|$) reemplazándolos por las variables $z_{i,j}$ adecuadas y generar la claúsula $C''_{i,j}$ (que toma $O(1)$ en una RAM). Luego, la reducción es a tiempo $O(|\langle\varphi\rangle|)$, en particular a tiempo polinomial, en una RAM.

En resumen, hemos establecido que 3SAT reduce en tiempo polinomial a OneIn3SAT, como se deseaba.

(iii.1).- El problema de factibilidad enunciado lo podemos formalizar como un problema de decisión de pertenencia en:

$$\text{PlanClases} = \left\{ \langle \{d_i\}_{i\in[\ell]}, \{t_i\}_{i\in[m]}, \{P_i\}_{i\in[n]}, \{T_i\}_{i\in[n]} \rangle : \exists (s_1,\ldots,s_\ell) \in [n]^\ell,\, d_i \in P_{s_i},\, \cup_{i\in[\ell]} T_{s_i} = \{t_i\}_{i\in[m]} \right\}.$$

(iii.2).- Sea $\langle G = (V,E), \ell \rangle$ una instancia de VC. Informalmente, la reducción considera los arcos de G como los temas que se requiere cubrir. Un nodo v de G se interpretará como un "profesor", que cubre los temas correspondientes a los carcos incidentes en v. Habrán ℓ días para agendar clases, y todos los profesores estarán disponibles todos los días. Formalmente, definimos $d_i = i$ para todo $i \in [\ell]$. Definimos los $P_v = \{ d_j : j \in [\ell] \}$ para todo $v \in V(G)$. Finalmente, definimos $T_v = \delta(v)$ para todo $v \in V(G)$. Sea $f(\cdot)$ la aplicación que a $\langle G, \ell \rangle$ le asocia $\langle \{d_i\}_{i\in[\ell]}, E, \{P_v\}_{v\in V}, \{T_v\}_{v\in V} \rangle$.

Afirmamos que $f(\cdot)$ es una reducción a tiempo polinomial de VC a PlanClases. En efecto, si $S \subseteq V(G)$ es un recubrimiento de nodos de G de tamaño a lo más ℓ, se tiene que asignando $\{P_s\}_{s\in S}$ a los ℓ días, en cualquier orden, se obtiene una calendarización factible. Por otro lado, si $(s_1,\ldots,s_\ell)$ es un testigo de la pertenencia de $f(\langle G,\ell\rangle)$ en PlanClases, entonces se verifica fácilmente que $S=\{s_i : i\in[\ell]\}$ es un recubrimiento de nodos de G de tamaño ℓ.

Finalmente, para ver que $f(\cdot)$ se puede calcular en tiempo polinomial, basta observar que su determinación requiere esencialmente ciclar sobre los nodos v de G y para cada uno de ellos, ciclar sobre los arcos de G para calcular $\delta(v)$. Sigue que $f(\cdot)$ se pude calcular en tiempo $O(|V|\cdot|E|)$ en una RAM.

(iii.3).- Sea $\langle\varphi\rangle$ una instancia de 3SAT donde $\varphi=\varphi(x_1,\ldots,x_n)=\wedge_{i=1}^{m}C_i(x_1,\ldots,x_n)$. La idea de la reducción es ver las claúsulas como los "temas" que requieren ser cubiertos. Además, cada uno de los $2n$ posibles literales, se considerarán como un "profesor" que cubre los "temas" correspondientes a las claúsulas que contienen al literal (i.e. las claúsulas que se satifacen al hacer verdadero el literal). Finalmente, la reducción enfuerza que sólo uno de los dos profesores asociados a una variable pueda ser seleccionado, lo anterior vía asignarles un idéntico día a los dos referidos profesores como el único factible donde pueden hacer clases. Formalmente, definimos $d_i=i$ para todo $i\in[n]$. Definimos los $P_{x_i}=P_{\overline{x_i}}=\{i\}$ y finalmente, definimos T_{x_i} (respectivamente, $T_{\overline{x_i}}$) como la colección de claúsulas C_j donde aparece el literal x_i (respectivamente, $\overline{x_i}$). Sea $f(\cdot)$ la aplicación que a $\langle\varphi\rangle$ le asocia $\langle\{d_i\}_{i\in[n]},\{C_j\}_{j\in[m]},\cup_{i\in[n]}\{P_{x_i},P_{\overline{x_i}}\},\cup_{i\in[n]}\{T_{x_i},T_{\overline{x_i}}\}\rangle$.

Afirmamos que $f(\cdot)$ es una reducción a tiempo polinomial de 3SAT a PlanClases. En efecto, si $a=(a_1,\ldots,a_n)$ es una asignación que satisface $\varphi=\wedge_{i=1}^{m}C_i(x_1,\ldots,x_n)$, entonces una calendarización factible consiste en programar para el i-ésimo día al profesor P_{x_i} si $a_i=1$, y al profesor $P_{\overline{x_i}}$ en caso contrario. Por otro lado, si $(y_1,\ldots,y_n)$ es un testigo de la pertenencia de $f(\langle\varphi\rangle)$ en PlanClases, entonces se verifica fácilmente que la asignación $a=(a_1,\ldots,a_n)$ tal que $a_i=1$ si $y_i=x_i$, y $a_i=0$ si $y_i=\overline{x_i}$, es tal que satisface φ.

Finalmente, para ver que $f(\cdot)$ se puede calcular en tiempo polinomial, basta observar que su determinación requiere esencialmente ciclar sobre las n variables de φ y para cada uno de ellos, ciclar sobre las m claúsulas φ para calcular T_{x_i} y $T_{\overline{x_i}}$, lo que toma $O(n\cdot m)$ en una RAM. Posteriormente, en $O(n)$, también en una RAM, se pueden generar los conjuntos P_{x_i} y $P_{\overline{x_i}}$. Sigue que $f(\cdot)$ se pude calcular en tiempo polinomial en una RAM.

(iv).- Bastará ver que Indep $\leq_P$ SetPack. En efecto, sea $\langle G,k\rangle$ una instancia de Indep, donde $G=(V,E)$ es grafo y $k\in\mathbb{N}$. Para $v\in V$, sea S_v el conjunto de arcos de G incidentes en v, i.e. $S_v=\delta(v)$. Sea $f(\cdot)$ la aplicación que a $\langle G,k\rangle$ le asocia la instancia $\langle S_v : v\in V,E,k\rangle$ de SetPack. Afirmamos que

$$\langle G,k\rangle \in \text{Indep} \Longleftrightarrow f(\langle G,k\rangle)=\langle S_v : v\in V,V,k\rangle \in \text{SetPack}.$$

En efecto, si $I\subseteq V(G)$ es un conjunto independiente en G, entonces $S_u\cap S_v=\emptyset$ para todo $u,v\in I$, $u\neq v$ (de lo contrario existiría un arco e de G cuyos extremos serían u y v, contradiciendo que I es conjunto independiente). Si $|I|\geq k$, entonces I sería testigo de la pertenencia de $f(\langle G,k\rangle)=\langle S_v : v\in V,E,k\rangle$ en SetPack.

Para calcular $f(\langle G,k\rangle)$, basta esencialmente ciclar sobre los nodos de G determinando para cada nodo v el conjunto $S_v=\delta(v)$. Una codificación de este último se puede calcular ciclando sobre los arcos de G identificando aquellos en que uno de sus extremos sea igual a v. Sigue que $f(\langle G,k\rangle)$ puede calculares en tiempo $O(|V|\cdot|E|)$ en una RAM.

En resumen $f(\cdot)$ es una reducción a tiempo polinomial de Indep a SetPack.

PROBLEMA 2:

(i).- Ver apuntes de cátedra o el Sipser.

(ii.1).- Un ejemplo de lenguaje auto-reducible es 3SAT. En efecto, dado $\langle\varphi\rangle$ instancia de 3SAT de largo n, donde $\varphi = \varphi(x_1,\ldots,x_n) = \wedge_{i=1}^{m} C_i(x_1,\ldots,x_n)$, es fácil ver que $\langle\varphi(x_1,\ldots,x_{n-1},1)\rangle$ y $\langle\varphi(x_1,\ldots,x_{n-1},0)\rangle$ son instancias de 3SAT de largo estrictamente menor que $s = |\langle\varphi\rangle|$, ambas calculables eficientemente por una máquina de Turing M que utilizando el oráculo $3SAT_{s-1}$ acepta si y sólo si $\langle\varphi(x_1,\ldots,x_{n-1},1)\rangle$ o $\langle\varphi(x_1,\ldots,x_{n-1},0)\rangle$ están en $3SAT_{s-1}$.

(ii.2).- Sean L, L_n, y M como en el enunciado. Sea ω una potencial entrada tal que $n = |\omega|$. Sea $p(\cdot)$ polinomio tal que M es a tiempo $p(n) = Cn^c$ para C y c constantes suficientemente grandes. Consideremos el siguiente algoritmo recursivo $\mathcal{A}$:

```
input: ω.
if |ω| = 0 then
(*)   if ε ∈ L then
          Aceptar;
      else
          Rechazar;
else
      m ← p(|ω|);
      for α = α_1 ... α_m ∈ {Aceptar, Rechazar}^m do
          s ← 1;
          Simular M en ω de forma que:
          begin
              La s-ésima vez que M requiera consultar su oráculo por la pertenencia de digamos ω_s,
              proseguir la simulación suponiendo que la respuesta fue α_s;
              s ← s + 1;
          end
          for i = 1, ..., s do
(**)          if A(ω_i) ≠ α_i then Rechazar;
      Aceptar;
```

Notar que siempre podemos asumir que M, puede responder acerca de la membresía en L de un número constante de palabras, por lo que el paso (*) puede implementarse eficientemente (de hecho, en tiempo constante). Observar además, que por las características de M y L, si $\mathcal{A}$ es ejecutado en la entrada ω, entonces cada invocación del algoritmo en el paso (**) es sobre ω_i's de largo estrictamente menor al de ω. Luego, la profundidad de la recursión cuando se ejecuta $\mathcal{A}$ en ω es a lo más $|\omega|$. Por inducción fuerte en el largo de la entrada con que se invoca el algoritmo $\mathcal{A}$, es relativamente sencillo verificar que $\mathcal{A}$ decide L. Más aún, lo hace en espacio $S(\cdot)$, donde $S(\cdot)$ se puede asumir monótona no-decreciente y tal que satisface la siguente recurrencia:

$$S(n) \leq O(p(n)) + S(n-1),$$

donde $S(0) = O(1)$. Resolviendo la recurrencia, se tiene que $\mathcal{A}$ puede ser implementado en una máquina de Turing en espacio $S(n) = O(n \cdot p(n))$. Resumiendo, se tiene que $L \in$ PESPACIO, como se quería establecer.