Universidad de Concepción
Facultad de Ingeniería
Departamento de Ingeniería Informática y Ciencias de la Computación

Práctica 12 Solución

Teoría de Computación (503306)

Profesor: María Angélica Pinninghoff
Ayudante: Diego Palma

1. Los problemas de la clase P son aquellos problemas que pueden resolverse utilizando algoritmos que se ejecuten en tiempo polinomial. En otras palabras, el algoritmo se ejecuta en tiempo polinomial en una Máquina de Turing (Determinista).

2. Los problemas NP son aquellos problemas que tienen algoritmos que no necesariamente se ejecutan en tiempo polinomial en un computador regular. Sin embargo, en una Máquina de Turing no determinista, se pueden ejecutar en tiempo polinomial (de ahí que NP significa *nondeterministic polynomial time*).

   Una forma equivalente de definir NP es hacer referencia a problemas que se pueden verificar en tiempo polinomial. Esto significa que no necesariamente existe una forma de encontrar la solución al problema en tiempo polinomial, pero que una vez se encuentra la solución, toma un tiempo polinomial verificar que es correcta.

   Como nota aparte, está el problema abierto de determinar si $P = NP$, lo que significa que cualquier problema que se puede verificar en tiempo polinomial se puede resolver también en tiempo polinomial y vice versa. Si esto se pudiese demostrar, revolucionaría las ciencias de la computación, pues se podrían diseñar algoritmos más rápidos para resolver una gran variedad de problemas importantes.

3. Se deben demostrar dos cosas:

   - El problema es NP
   - El problema es NP-Hard. Esto quiere decir que para cualquier lenguaje L' en NP exista una reducción de L' a L en tiempo polinomial. (Ver libro guía, página 429 del pdf).

   Por otro lado, si se sabe que un problema P_1 es *NP-completo*, y se quiere demostrar si un problema P_2 es *NP-completo*, entonces si P_2 es NP, y existe una reducción en tiempo polinomial de P_1 a P_2, entonces se puede concluir que P_2 es *NP-completo*.

4. Los siguientes problemas son *NP-completos*:

   a) Satisfacibilidad booleana (SAT)
   b) Vendedor Viajero
   c) Clique
   d) Knapsack
   e) Muchos más, probablemente vistos en clases...

5. Falso. Si la reducción fuese de *3-coloring* al problema X entonces se podría concluir que X es *NP-completo*. Pero sólo sabiendo que X es NP, no es suficiente, pues X podría ser P. Se debería demostrar que X es $NP - hard$.

6. Primero se debe demostrar que B está en NP. Esto es simple de observar, pues si se nos entrega una solución al problema B, podemos verificar dicha solución en tiempo polinomial (Además, una MT no determinista podría encontrar la solución en tiempo polinomial). Luego, B es NP.

Por otro lado, debemos encontrar una reducción de HC a B. Supongamos que existe un programa imaginario que resuelve B. Creamos un nuevo grafo G' el cual está compuesto por dos copias de G; las copias no están conectadas entre sí. Luego, el grafo original G tiene un ciclo de Hamilton sí y solo sí el nuevo grafo G' tiene un ciclo simple de longitud igual a la mitad del número de vértices en G'. Luego, existe una reducción en tiempo polinomial de HC a B, y por lo tanto B es *NP-completo*.