Universidad del Valle de Guatemala Christopher Sandoval 13660
Análisis y Diseño de Algoritmos - Sección 10 17/03/2020

Tarea 3
Complejidad

1. **Se define el problema LONGEST-PATH como un problema de decisión cuyas instancias son tuplas $\langle G,u,v,k\rangle$, donde G es un grafo no-dirigido, u y v son un par de vértices; y k es un número mayor o igual a cero. El problema pregunta si entre los vértices u,v existe un camino simple (sin ciclos) que pase por al menos k vértices. Luego se define el problema LONGEST-PATH-LENGTH como el problema con instancia $\langle G,u,v\rangle$, cuya solución es el número de vértices que hay en el camino simple más largo entre u,v. Demuestre que LONGEST-PATH-LENGTH puede ser resuelto en tiempo polinomial si y sólo si LONGEST-PATH pertenece a P.**

   Asumiendo que LONGEST-PATH pertenece a P entonces podemos ejecutar el algoritmo para cada vértice en el grafo, cuando el algoritmo nos resulte falso se encontró el camino más largo. Este algoritmo se ejecutaría como máximo n veces por lo que es de tamaño polinomial. Para resolver el LONGEST-PATH-LENGHT en tiempo polinomial lo que se tiene que hacer es ir utilizando los resultados del algoritmo anterior para calcular así la longitud del más largo.

2. **Considere el conjunto $GI=\{\langle G_1,G_2\rangle\}$ donde G_1 y G_2 son grafos isomorfos. Un par de grafos son isomorfos si existe un isomorfismo entre ellos, que es una función biyectiva $f:V(G_1)\rightarrow V(G_2)$ de los vértices de G_1 a los vértices de G_2 tal que $u,v \in G_1$ son adyacentes sí y sólo sí $f(u),f(v) \in G_2$ son adyacentes también (i.e., G_1 y G_2 tienen la misma cantidad de vértices, conectados de la misma forma). Demuestre que el problema de determinar si dos grafos son isomorfos es NP, proveyendo un algoritmo que permita verificar el isomorfismo en tiempo polinomial.**

   Para empezar se tiene que obtener todas las posibles parejas del primer grafo, esto sería un algoritmo $O(n^2)$. Después en cada pareja de vértices se debe comprobar que tengan un equivalente en el segundo grafo. Sabiendo que cada una de las parejas si tiene un equivalente en el otro grafo entonces se tiene que comprobar que tiene las mismas conexiones en ambos sentidos. Esta comprobación también sería $O(n^2)$.

3. **Sea X un problema nuevo e Y un problema demostrado NP-complete. ¿En qué sentido se debe hacer la reducción para demostrar que X es también NP-complete? ¿Por qué?**

   La reducción se haría de X a Y y sabiendo que todos los problemas NP-complete también son problemas NP-hard entonces X pasaría también a ser NP-hard. Pero cómo

sabemos que todos los problemas NP-hard son NP-complete entonces X también sería NP-complete.

4. **Se puede determinar la satisfactibilidad de una fórmula booleana en tiempo polinomial si la fórmula está en forma normal disyuntiva. Investigue sobre forma normal disyuntiva (DNF) y forma normal conjuntiva (CNF) y responda: ¿por qué, pese a que determinar la satisfactibilidad de una fórmula DNF toma tiempo polinomial, SAT sigue siendo NP-complete? ¿Cómo es que determinar la satisfactibilidad de una fórmula en DNF toma tiempo polinomial?**

   La razón recae en el hecho de que en la forma DNF consiste de varios ANDs unidos por ORs entonces para demostrar que es válida solo es necesario comprobar que uno es válido. Pero con los problemas CNF si hay que revisar todas las cláusulas y convertir un problema CNF a DNF no toma un tiempo polinomial, por lo que SAT sigue siendo NP-complete.

5. **El longest-simple-cycle problem es el problema de hallar el ciclo simple (visita cada uno de sus nodos una única vez) más largo que existe en un grafo dado. Demuestre que este problema es NP-complete.**

   El problema de probar todos los caminos posibles toma un tiempo n en una máquina de Turing no determinística porque podemos probar todos los caminos simultáneamente de cada nodo. A esto debemos aplicarle el ciclo Hamiltoniano que consigue una ruta que pasa únicamente por cada uno de los vértices. Este problema es NP-complete, entonces el longest-simple-cycle también debe pertenecer a NP-complete.

6. **Demuestre que el problema de las torres de Hanoi para tres torres y n discos es NP-hard pero no NP-complete.**

   El problema consiste en pasar los discos de la primera torre a la última procurando que nunca quede un disco más grande por debajo de un disco más pequeño. Esta comparación toma un tiempo exponencial en verificarse por lo que este problema no puede ser clasificado como NP-complete.