Introducción a la Informática Teórica
Tarea 6
Hamiltonian Guy

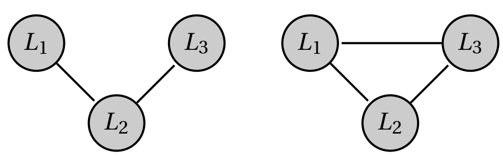
Hernán Vargas Leighton
201073009-3

9 de julio de 2014

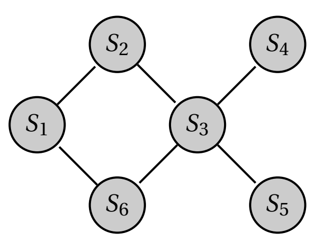
Respuestas

1. Buscamos el menor subconjunto de nodos tal que todas las aristas sean adyacentes a estos. Para ello debemos recorrer cada nodo y verificar sus aristas, por lo tanto la evaluación caso a caso será en tiempo polinomial. Como buscamos el menor subconjunto debemos siempre recorrer el grafo hasta el final, encontrar todos los subconjuntos que son *vertex cover*, y elegir de ellos el más pequeño, entonces el problema es `NP`.
   Para que el problema sea `NP`-completo debe ser `NP` y `NP`-duro, esta última condición se satisface haciendo la reducción de 3-SAT a nuestro problema:

   - 3-SAT nos dice que el problema de verificar si existen valores booleanos tal que una expresión compuesta por clausulas de tres literales sea verdadera es `NP`-completo, es decir: $S_1 \times S_2 \times ..S_n =$ `True`, con $\times$ una operación booleana (`or, and, not`) y S_i una clausula de la forma $(L_1 \times L_2 \times L_3)$
   - Digamos las clausulas S_i de manera que L_1, L_2, L_3 sean valores `True` o `False` arbitrarios, así $\times$ puede ser reducido a `and` u `or` (el `not` no es necesario pues basta cambiar el valor de L_i por su complemento).
   - Ahora, si la expresión está compuesta por dos `or`, le asignamos un grafo con tres nodos y dos aristas de manera de que un nodo sea adyacente a los otros dos. En cualquier otro caso asignamos un grafo con tres nodos y tres aristas entonces cada nodo estará conectado con los demás. Así, asignamos a cada clausula grafos de tres nodos, de alguna de estas formas:

   - Cada uno de estos grafos tendrá al menos un nodo con dos aristas, llamemos a éste el nodo principal.
   - Crearemos un nuevo grafo pero ahora con la expresión que evalúa las clausulas, para ello cuando las expresiones estén con un `and` pondrán los nodos linealmente mientras que las con `or` unirán solo uno de sus nodos a los `and`, por ejemplo una expresión como $S_1 \wedge S_2 \wedge (S_3 \vee S_4 \vee S_5) \wedge S_6$ será:

   - Ahora a cada nodo S_i se le agrega una arista y se conecta con el nodo principal de los literales de cada clausula. Con este proceso se logra transformar una expresión booleana 3-SAT a un grafo.
   - Buscar si la expresión booleana satisface el 3-SAT será equivalente a solucionar el problema *Vertex Cover*, por lo tanto se logra la reducción.

2. Reducción de *Vertex Cover* a *Set Cover*:

   - Digamos nuestro grafo para el *Vertex Cover* como $G = (V, E)$, con V el conjunto de los nodos y E el conjunto de las aristas.
   - Digamos los elementos de V como $V_i \forall i \in 1..p$, entonces V_i representará al nodo i de un total de p.
   - Digamos los elementos de E como $E_i \forall i \in 1..q$, E_i será la arista i de un total de q.
   - Definamos el conjunto U como la unión de los nodos y las aristas de nuestro grafo, es decir $U = V \cup E$.
   - Los subconjuntos S_i serán elegidos de manera que se contengan un nodo y todos los nodos y aristas adyacentes, por ejemplo para un nodo V_1 que es adyacente a E_1, E_2, V_2, V_3 tendremos el subconjunto $\{V_1, E_1, E_2, V_2, V_3\}$. Hacer esta transcripción implica recorrer los nodos uno por uno y revisar sus vecinos, por lo tanto su tiempo es polinomial.
   - Tendremos entonces que el número de elementos $n = p + q$ y el número de subconjuntos $m = n$.
   - Ahora encontrar el *Vertex Cover* de G será equivalente a encontrar un mínimo k tal que su unión sea U, por lo tanto se logra la reducción.

3. En primer lugar el algoritmo para resolver el *Half-Clique* será polinomial pues nos basta con tomar toda combinación posible de nodos que sean la mitad de los nodos totales y verificar si todos están conectados entre si, así el problema es `NP`.
   Para probar que el problema es `NP`-duro y por ello `NP`- completo basta reducir el problema *Clique* al *Half-Clique*:

   - *Clique* nos dice que el problema de encontrar un k tal que un grafo G tenga un subconjunto G' con k nodos de manera que estos estén todos conectados entre si, es `NP` -completo.
   - Digamos un grafo G con n nodos.
   - Por definición de *Half-Clique* sabemos que n será par, entonces $n/2$ es un natural mayor o igual a 1.
   - Basta con decir $k = n/2$, ahora el problema *Clique* fue reducido a un problema *Half-Clique*.
   - Notamos que *Half-Clique* es un caso particular de *Clique*.

   Con esto se concluye que *Clique* es `NP`-completo.

HV/$\LaTeX\,2_\varepsilon$