Universitat Oberta de Catalunya

Estudios de Informática, Multimedia y Telecomunicación

ASIGNATURA: Grafos y Complejidad

Tercera PEC. Módulos 6 y 7.

Semestre de primavera de 2012 (del 9 de mayo al 30 de mayo).

Por favor, seguid las instrucciones siguientes:

- Enviad la solución en un fichero con el nombre:
  PEC3_Apellido1Apellido2Nombre.pdf
- Debe enviarse al apartado "Entrega y registro de EC" del aula.
- Enumerad las respuestas de acuerdo con la numeración de las preguntas y los apartados.
- Las respuestas a los problemas propuestos no deben limitarse a dar un resultado. Añadid, también, una explicación que justifique la respuesta.

1. (Valoración de un 20 %) Considerar las fórmulas booleanas siguientes:

   - $f_1 = (x \vee \bar{y}) \wedge (\bar{x} \vee y)$.
   - $f_2 = (x \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee z)$.
   - $f_3 = (x \wedge y) \vee (\bar{x} \wedge z)$.
   - $f_4 = (\bar{x} \vee y \vee z) \wedge (x \vee \bar{y} \vee z) \wedge (x \vee y \vee \bar{z})$.

   a) Decir qué fórmulas están en FNC.

   b) Justificar si alguna de ellas son instancias del problema SAT o 3SAT.

   c) Considerar el problema 3SAT-EQUILIBRADO: dada una fórmula booleana en FNC donde cada cláusula contiene 3 literales y cada variable aparece negada y no negada el mismo número de veces en la fórmula, decidir si hay una asignación de variables en la fórmula que la satisface.
   Decir si alguna de las fórmulas anteriores es una instancia del problema 3SAT-EQUILIBRADO.

d) Demostrar que 3SAT-EQUILIBRADO $\in NP$.

e) Demostrar que 3SAT-EQUILIBRADO es NP-completo.

Solución:

a) Están en FNC las fórmulas f_1, f_2, f_4.

b) Las tres que estan en FNC son instancias del problema SAT pero sólo f_2 y f_4 son instancias del problema 3SAT.

c) La fórmula f_2 es una instancia del problema 3SAT-EQUILIBRADO.

d) El problema 3SAT-EQUILIBRADO es un subconjunto del problema 3SAT. Igual que el SAT y el 3SAT podemos verificar una asignación de valores de verdad a cada variable en tiempo polinomial (repasar el ejemplo 29 del módulo 6).

e) Puesto que 3SAT-EQUILIBRADO $\in NP$ es suficiente demostrar que 3SAT-EQUILIBRADO es NP-Difícil. Es fácil construir una reducción polinómica de 3SAT $\leq_p$ 3SAT-EQUILIBRADO. Sólo hace falta, para cada variable x, contar cuántas veces aparece negada y no negada y añadiremos las cláusulas $(x \vee t \vee \bar{t})$ o $(\bar{x} \vee t \vee \bar{t})$ que hagan falta, donde t es una nueva variable.

2. (Valoración de un 20 %)

Sean los problemas siguientes:
CONEXO: Dado $G = (V, A)$, determinar si G es conexo.
COMPONENTES: Dado $G = (V, A)$, decidir si G tiene exactamente k componentes conexos.

a) Demostrar que CONEXO $\in P$.

b) Demostrar que CONEXO $\leq_p$ COMPONENTES.

c) ¿Podemos afirmar a partir del apartado anterior que COMPONENTES $\in P$? ¿Y que COMPONENTES $\notin P$?

Solución:

a) Un posible algoritmo en tiempo polinómico sería: escogemos un vértice inicial cualquiera y hacemos una exploración BFS. Si acabamos obteniendo todos los vértices es que el grafo es conexo. Recordamos que el algoritmo BFS tiene complejidad $O(n + m)$.

b) La función de reducción sería $f(G) = (G, 1)$. Sólo es necesario observar que CONEXO$(G) \iff$ COMPONENTES$(G, 1)$.

c) No podemos afirmar que pertenezca a P, podríamos si la reducción fuese COMPONENTES $\leq_p$ CONEXO. Tampoco podemos afirmar que no pertenezca, ya que los dos problemas podrían ser polinómicament equivalentes.

3. (Valoración de un 20 %)

Sean los dos problemas siguientes:

MCD(n, m, x): Dados n, m y x enteros, $0 < n \leq m$, decidir si el mcd(n, m) (o sea, el máximo común divisor de n y m) es igual a x.
COPRIMOS(n, m): Dados n i m enteros, $0 < n \leq m$, determinar si n y m no tienen ningún divisor común mayor que 1.

a) Sea el algoritmo siguiente para hallar el mcd de n y m:

función $MCD(n, m)$
  inicio
    $mcd \leftarrow 1$
    para $i = 1$ **hasta** n
      si $(n \bmod i = 0 \ \wedge \ m \bmod i = 0)$ **entonces**
        $mcd = i$
      finsi
    finpara
    retorno mcd
  fin

Demostrar que este algoritmo tiene complejidad exponencial respecto de la medida de la entrada.

b) A partir del apartado anterior, ¿podemos afirmar que $MCD(n, m, x)$ es intratable?

c) Demostrar que COPRIMOS$(n, m) \leq_p$ MCD(n, m, x).

Solución:

a) Dentro del bucle se produce un número constante de operaciones. El bucle se ejecuta n veces. La medida de la entrada es $N = \log_2(n) + \log_2(m) = \log_2(nm)$. Por lo tanto, el algoritmo tiene complejidad $O(2^N)$ respecto la medida de la entrada.

b) A partir del apartado anterior, sólo podem afirmar que MCD$(n, m, x) \in EXP$, pero no excluye que exista un algoritmo en tiempo polinómico que resuelva el problema.

c) La función de reducción sería $f(n, m) = (n, m, 1)$. Observar que COPRIMOS(n, m) $\Longleftrightarrow$ MCD$(n, m, 1)$.

4. (Valoración de un 20 %)

   Considerar los problemas siguientes:
   SUMA_SUB: Dado un conjunto C de enteros y un entero t, decidir si existe C', subconjunto de C, tal que la suma de los elementos de C' es exactamente t.

   SUMA_RESTA_SUB: Dado un conjunto C de enteros positivos y un entero t, decidir si existe C', subconjunto de C, tal que sumando y/o restando los elementos de C' obtenemos exactamente t. (Por ejemplo, si $C = \{1, 3, 6, 10\}$ y $t = 8$, C' podría ser $\{1,3,6\}$, porque $3 + 6 - 1 = 8$)

   a) ¿De que tipo es cada uno de los problemas? (decisional, de cálculo, de optimización).

   b) Demostrar que SUMA_RESTA_SUB $\leq_p$ SUMA_SUB, usando la función de reducción $f(C, t) = (\bar{C}, t)$, donde $\bar{C} = C \cup \{-x \mid x \in C\}$. (siguiendo con el ejemplo anterior, $\bar{C}$ sería $\{1, -1, 3, -3, 6, -6, 10, -10\}$).

   Solución:

   a) Son problemas decisionales ambos.

   b) Observemos que f es una función polinómica (de hecho, constante), ya que sólo hay que añadir, para cada $x \in C$, x y $-x$ a $\bar{C}$. Hay que ver que SUMA_RESTA_SUB(C) $\iff$ SUMA_SUB($\bar{C}$):
   Si SUMA_RESTA_SUB(C) es cierto, todo elemento de C' y su opuesto están en $\bar{C}$, por lo que $C' \cup \{-x \mid x \in C'\}$ contiene un subconjunto de $\bar{C}$ que nos garantiza que SUMA_SUB($\bar{C}$) es cierto (en el ejemplo del enunciado sería $\{-1, 3, 6\}$).
   Supongamos ahora que SUMA_SUB($\bar{C}$) es cierto. Sea $\bar{C}'$ el subconjunto que nos da t. Consideremos $C' = \{x \in C \mid x \in \bar{C}'\} \cup \{x \in C \mid -x \in \bar{C}'\}$. Por definición de $\bar{C}$, si $x \in \bar{C}'$, entonces o bien x o bien $-x$ pertenece a C. Sumando los términos de C' que se encuentran en el primer caso y restando los que se encuentran en el segundo obtenemos t. Por lo tanto, SUMA_RESTA_SUB(C) es cierto.

5. (Valoración de un 20 %)

   Considerar un conjunto de n ficheros $S_1, \ldots, S_n$, donde el fichero S_j tiene longitud c_j ($j = 1, \ldots, n$) y un conjunto, $\{x_1, \ldots, x_m\}$, de m peticiones de unidades de información. Cada unidad de información está almacenada en, al menos, un fichero. Queremos encontrar un subconjunto de ficheros de coste total mínimo tal que permitan responder a todas las peticiones de unidades de información.

   a) Modelar este problema utilizando la teoría de grafos y definir el problema de optimización asociado.

b) Relacionadlo con alguno de los problemas estudiados en el módulo 7 y justificar si es un problema intratable.

c) Si la tabla siguiente representa una instancia del problema, calcular una solución óptima:

	S_1	S_2	S_3	S_4	S_5
x_1	1	0	1	1	0
x_2	1	0	0	1	0
x_3	1	1	0	1	0
x_4	0	1	1	1	0
x_5	1	0	0	0	1
x_6	0	0	1	0	0
x_7	0	1	0	1	0
Coste	3	6	2	5	7

Solución:

a) Definimos un grafo bipartido $G(S \cup X, A)$ donde $S = \{S_1, \ldots, S_n\}$ y $X = \{x_1, \ldots, x_m\}$. S_i y x_j son adyacentes si el fichero S_i contiene la petición x_j. Además, en el conjunto de vértices S hay definidos el conjunto de costes $C = \{c_1, \ldots, c_n\}$. El problema de optimización sería, ¿cuál es el subconjunto S' de S de coste mínimo que verifica que para todo $x \in X$ es adyacente a algún vértice de S'?

b) Se trata de una variante del problema del conjunto de dominación que es un problema NP-Completo. Por lo tanto, también es un problema intratable.

c) Podemos resolver la instancia del problema como un problema de cobertura:

	S_1	S_2	S_3	S_4	S_5
x_1	1	0	1	1	0
x_2	1	0	0	1	0
x_3	1	1	0	1	0
x_4	0	1	1	1	0
x_5	1	0	0	0	1
x_6	0	0	1	0	0
x_7	0	1	0	1	0
Coste	3	6	2	5	7

La fila x_6 sólo tiene un 1 en la columna S_3. Por lo tanto, S_3 tiene que formar parte de la solución.

Ahora podemos eliminar las filas x_1, x_4 y x_6:

	S_1	S_2	S_3	S_4	S_5
x_2	1	0	0	1	0
x_3	1	1	0	1	0
x_5	1	0	0	0	1
x_7	0	1	0	1	0
Coste	3	6	2	5	7

Ahora, x_3 cubre x_2 y x_7. Por lo tanto, podemos eliminar x_3.

	S_1	S_2	S_3	S_4	S_5
x_2	1	0	0	1	0
x_5	1	0	0	0	1
x_7	0	1	0	1	0
Coste	3	6	2	5	7

Ahora, la fila x_2 está cubierta por S_1 y S_4. Tenemos que elegir S_1 que también cubre x_5. Finalmente, para cubrir x_7 tenemos que elegir S_4 que tiene un coste menor que S_2.

la solución final será $S' = \{S_1, S_3, S_4\}$ con un coste 10.