PRÁCTICA 3
MAC

Reducciones

Juan Antonio Velasco Gómez
Miguel Sánchez Maldonado

1. **El problema de la parada** (una vez más): Definimos la función $HALT(p, x)$ como una función booleana que nos indica si la máquina de Turing M_p representada por p para en un número finito de pasos para la entrada x. Por otro lado, sabemos (por diagonalzación) que existe una función no computable UC definida como se describe a continuación. La función $UC(s)$ es 0 si la máquina de Turing codificada por s devuelve 1 cuando recibe s como entrada; esto es, $UC(s) = 0$ cuando $M_s(s) = 1$. En cualquier otro caso, cuando la máquina de Turing devuelve cualquier otro valor o, simplemente, no para, $UC(s) = 1$. Dada la existencia de la función no computable UC, demostrar por reducción que la función $HALT$ tampoco es computable.

   NOTA: Para demostrar que UC no es computable, basta con que supongamos que UC es computable. Entonces, existiría una máquina de Turing M tal que $M(s) = UC(s)$ para cualquier cadena de bits s. En particular, debería verificarse que $M(M) = UC(M)$, lo cual, por la propia definición de UC, es imposible.

[Arora & Barak: "Computational Complexity", 1.5.1, pp. 22-23]

Solución

Podemos definir la función **HALT** como una función booleana que nos indica si la máquina de Turing M_p representada por p para un número finito dada una entrada x.

```
UC(Mp(x)){
        Halt(p,x)
}
```

Tenemos que:

UC <= HALT

Si Halt fuera computable ⇒ UC sería computable, pero sabemos por el enunciado del problema que la función UC no es computable

⇒ por reducción tenemos que el problema de la parada tampoco sería computable.

2. **Caminos disjuntos** [DISJOINT PATHS]: Dos caminos en un grafo se dicen disjuntos [edge-disjunct] si no comparten aristas en común. Sin embargo, sí permitimos que compartan vértices. Demuestre cómo resolver el siguiente problema de decisión a partir de un algoritmo que nos permita calcular el flujo máximo en una red [MAX FLOW]: Dados un grafo dirigido (G), dos vértices (s, t) y un número entero (k), determinar si existen k caminos disjuntos de s a t.

[Moore & Mertens: "The Nature of Computation", problem 3.48, pp. 88]

[Kleinberg & Tardos: "Algorithm Design", Section 7.6, pp. 373-378]

Solución

Tenemos que reducir el problema del flujo a nuestro problema de caminos disjuntos, para ello vamos a partir de un grafo ponderado con ponderación 1 en todas sus aristas, por lo que el flujo tendrá que ser igual al número de caminos disjuntos.

1º) Vamos a probar que **Flujo >= nº de caminos:**
<u>En efecto</u>

Supongamos que no lo es, existe un camino cuyo flujo es <1, lo cual no es posible porque implicaría que tenemos un camino con flujo < 1, lo cual no puede ser.

2º) Vamos a probar que **Flujo <= nº de caminos:**
<u>En efecto</u>

Supongamos que no lo es, existe un camino cuyo flujo es >1 porque hemos supuesto que el grafo es de ponderación 1.

⇒ Flujo=nº caminos ⇒ Calculando el flujo, podemos calcular el número de caminos disjuntos.

Por tanto podemos reducir nuestro problema al siguiente: Crear un programa que nos calcule a partir de un grafo ponderado, su flujo máximo. Tomamos ahora el grafo del que queremos sacar el número de caminos disjuntos y ponderamos todas sus aristas a 1. Se lo metemos ahora al programa de cálculo del flujo. Como, para un grafo con aristas ponderadas en 1,

Flujo = nº caminos

(como hemos demostrado antes) tenemos calculado el número de caminos disjuntos del grafo.

3. **Spin glass** (`http://en.wikipedia.org/wiki/Spin_glass`): Un "vidrio de espín" es un sistema magnético en el que el acoplamiento entre los momentos magnéticos de los distintos átomos es aleatorio (algo así como un imán desordenado). Formalmente, puede verse como un grafo en el que el vértice i-ésimo tiene un espín $s_i = \pm 1$ que representa si el campo magnético del átomo i-ésimo apunta hacia arriba o hacia abajo. Cada arista (i, j) del grafo tiene asociada una fuerza de interacción J_{ij} que indica hasta qué punto interactúan s_i y s_j. Además, cada vértice i está sometido a un campo externo h_i. Dada una configuración del sistema (un conjunto de valores para los s_i), su energía es

$$E(\{s_i\}) = -\sum_{ij} J_{ij} s_i s_j - \sum_i h_i s_i$$

La arista (i, j) es ferromagnética si $J_{ij} > 0$ y antiferromagnética si si $J_{ij} < 0$; esto es, el ferromagnetismo tiende a alinear los espines. El campo externo es positivo o negativo en función de si queremos que s_i sea $+1$ o -1, respectivamente.

Un sistema como el descrito tiende a minimizar su energía, al menos a temperatura cero. Muestre cómo calcular el estado de mínima energía del sistema ('ground state' para los físicos) para materiales ferromagnéticos ($J_{ij} \geq 0$) reduciendo el problema al de calcular un corte mínimo en un grafo [MIN CUT], problema que se puede resolver en tiempo polinómico.

[Moore & Mertens: "The Nature of Computation", problem 3.50, pp. 88-89]

Solución

Para hacer una solución más fácil de entender vamos a suponer un grafo ponderado con forma de malla, en cada vértice tendremos los valores de S_j y en cada arista los J_{ij}, es decir los vértices de cada arista. Ponemos dos vértices más, Q y W, ambos con valor 1 y que va a unirse de la siguiente forma:

- Si $H_i > 0$ Q se unirá con S_i y por tanto la arista resultante tiene el valor H_i
- Si $H_i < 0$ W se unirá con S_i y por tanto la arista resultante tiene el valor H_i

Nuestro problema se ha convertido en un problema de grafos ponderados sabemos por teoría que el flujo máximo, es decir el flujo de un punto a otro de un grafo ponderado así como el corte mínimo son equivalente, por tanto usaremos el corte mínimo y nos basamos en que son equivalentes. El corte mínimo lo hacemos sobre Q y W

La estructura para cualquier camino entre Q y W tiene como valor:

$$H_k + \Sigma J_{ij} S_i S_j - H_p \text{ con k y p fijo}$$

Veamos que significa cada término de la expresión anterior.

· H_k el valor de la primera arista
· $\Sigma J_{ij}S_iS_j$ es la suma finita de las aristas intermedias en el camino ponderado
· H_p valor de la última arista

Maximizar el flujo será tomar todos los caminos posibles que no repitan ninguna arista, y cuando tomamos todos los caminos con su valor podemos observar que tenemos :

si queremos maximizar el flujo debemos tomar todos los caminos posibles que no repitan ninguna arista. Si nos fijamos, al tomar el total de caminos con su valor nos queda algo como: $\Sigma H_k + \Sigma J_{ij}S_iS_j - \Sigma H_p$ ya que sabemos que en todos los caminos no tenemos aristas repetidas. Esto es lo mismo que $\Sigma J_{ij}S_iS_j + \Sigma H_pS_p$
y del enunciado tenemos :

$$E(\{s_i\}) = - \Sigma J_{ij}S_iS_j - \Sigma H_pS_i = - \Sigma J_{ij}S_iS_j - \Sigma H_pS_p = - (\Sigma J_{ij}S_iS_j + \Sigma H_pS_p)$$

Podemos concluir por tanto que cuanto mayor es el flujo, menor es la energía, por tanto si buscamos el flujo máximo vamos a obtener la energía mínima.
Por tanto lo que vamos a hacer es hacer un programa que calcule a partir de un grafo ponderado su flujo máximo que en nuestro caso será lo mismo que calcular el corte mínimo.

A continuación se toman los vértices y creamos un grafo con los pasos que describimos anteriormente. Se le pasamos al programa de cálculo de flujo y entonces lo que obtenemos es que la energía que denotamos (E) es opuesta al flujo máximo(F_{max}), por tanto la expresión de la energía mínima será:

$$E = - F_{MAX}$$

4. **Problemas equivalentes:** Dos problemas A y B son equivalentes cuando A se puede reducir a B y B también se puede reducir a A. Muestre la equivalencia entre los elementos de los siguientes conjuntos de problemas:

2 problemas definidos sobre una lista de enteros:

- INTEGER PARTITIONING: Dada una lista de enteros positivos $S = \{x_1..x_k\}$, ¿existe una partición balanceada? Esto es, un subconjunto $A \subseteq \{1..k\}$ tal que $\sum_{i \in A} x_i = \sum_{i \notin A} x_i$.
- SUBSET SUM: Dada una lista de enteros positivos $S = \{x_1..x_k\}$ y un entero t, ¿existe un subconjunto de S tal que sus elementos sumen t? Esto es, un subconjunto $A \subseteq \{1..k\}$ tal que $\sum_{i \in A} x_i = t$.

[Moore & Mertens: "The Nature of Computation", 4.2.3, pp. 105-107 & exercise 4.13, p. 122]

3 problemas definidos sobre un grafo $G = (V, E)$:

- INDEPENDENT SET: Dado un grafo G y un entero k, ¿tiene G un conjunto independiente de tamaño k o mayor? Un conjunto de vértices $S \subseteq V$ es independiente si no incluye dos vértices adyacentes.
- VERTEX COVER: Dado un grafo G y un entero k, ¿tiene G una cobertura de tamaño k o menor? Un conjunto de vértices $S \subseteq V$ es una cobertura del grafo G si, para toda arista $e \in E$, al menos uno de los extremos de e está en S.
- CLIQUE: Dado un grafo G y un entero k, ¿tiene G un clique de tamaño k o mayor? Un clique es un subgrafo completo: un conjunto de vértices $S \subseteq V$ forma un clique si todos sus vértices son adyacentes.

[Moore & Mertens: "The Nature of Computation", 4.2.4, pp. 107-109]

Solución

a) Llamamos s = $\sum_{i=1}^{n} xi$

Veamos en primer lugar que "Integer Partitioning" se puede reducir a "Subset Sum"
En efecto

Si podemos encontrar un subconjunto de A de {1,2,...,k} tal que:

$$\sum_{i \in A} x_i = \sum_{i \notin A} x_i.$$

entonces es claro que $s/2=\sum_{i \varepsilon A} xi$. Recíprocamente, si podemos encontrar un subconjunto A de {1,2,...,k} tal que: $\sum_{i \varepsilon A} xi = s/2$ entonces la suma de los xi tales que i no pertenece a A ha de ser $\sum_{i\in A} x_i = \sum_{i\notin A} x_i.$

Hemos demostrado por tanto que encontrar una partición de S que sumen lo mismo equivale a encontrar un subconjunto cuyos elementos sumen s/2.

Ahora veamos que "Subset Sum" se puede reducir a "Integer Partitioning"

En efecto

Buscamos un subconjunto A de {1,2,...,k} tal que $\sum_{i \varepsilon A} xi$. = t . Para ello vamos a considerar el subconjunto S' definido así S' = $\{x_1,x_2,..,x_k,t+s,2s-t\}$. Si podemos encontrar un subconjunto A de S cuyos elementos sumen t ⇒ considerando los subconjuntos A U {2s-t} y (S intersección $\overline{A}$) U {t+s} verifican ambos que la suma de sus elementos vale 2s, y forman una partición de S'.

Por otra parte, si podemos particionar S' en dos conjuntos A y B tales que A intersección B sea el vacío y $\sum_{i \varepsilon A} xi = \sum_{i \varepsilon B} xi_i$ como (s+t) + (2s-t)=3s y tanto los elementos de A como los de B suman 2s, (s+t) y 2s-t pertenecen a diferentes conjuntos de la partición de S'.

Si utilizamos el mismo razonamiento que anteriormente, podemos ver que el resto de elementos de A pertenecen a A y suman t. Con esto hemos demostrado que poder encontrar un conjunto A que esté en {1,2,...,k} tal que $\sum_{i \varepsilon A} xi$. = t es equivalente a poder particionar S' en dos conjuntos cuyos elementos sumen lo mismo.

b) Veamos en primer lugar que "Independent Set" se reduce a "Clique"

En efecto

Dados un grafo G y un entero x, considerando su grafo complementario, se tiene que en el grafo original hay un clique de tamaño x o mayor ⇔ en el complementario hay un clique de ese mismo tamaño. De hecho, el conjunto de vértices será el mismo en ambos grafos.

Por el mismo razonamiento, dado un grafo G, se tiene que un clique es equivalente a un conjunto independiente de vértices en el grafo complementario. Por lo que existe un clique de tamaño mayor o igual que x ⇔ existe un conjunto independiente de vértices en el grafo complementario.

Es decir, el problema del Clique se reduce a Independent Set ⇒ Clique e Independent Set son equivalentes puesto que ya hemos probado ambas implicaciones.

Para acabar el ejercicio debemos demostrar que los tres problemas son equivalentes usando las equivalencias vistas y ya demostradas hasta ahora.

Si ahora tomamos un grafo G y un entero x, llamamos n al número de vértices del grafo. Como sabemos que existe un conjunto independiente de vértices de tamaño mayor o igual que x, esto equivale a que existe una cobertura de tamaño menor o igual a n-x, lo cuál equivale, a su vez, a que exista un conjunto independiente de vértices de tamaño n-x. Reduciendo el problema de “Independent set” en el de “Vertex Cover”. Es decir, “Independent Set” se reduce a “Vertex Cover” y viceversa, siendo entonces también equivalentes.

⇒ Hemos probado las tres equivalencias:

$$Independent\ Set \Leftrightarrow Clique \Leftrightarrow Vertex\ Cover$$