ISIS 211 Diseño de Algoritmos
Semestre 2001-1 - Parcial 3
Mayo 15, 2001
Prof. Rodrigo Cardoso

Sea S_n un conjunto de n números naturales, $n>0$. Considere el problema de determinar si existen dos subconjuntos disyuntos de S_n, tales que la suma de los elementos del uno sea igual a la suma de los elementos del otro.

Por ejemplo, para $S_{10} = \{3,28,32,35,49,59,60,66,88,95\}$, los conjuntos $\{28,95\}$ y $\{35,88\}$ solucionan afirmativamente el problema. En cambio, para $S_3 = \{1,3,5\}$, la respuesta es negativa.

Se quiere desarrollar un algoritmo de agenda para resolver el problema.

Para simplificar el manejo de subconjuntos de S_n, conviene tener una representación fácil de manipular. Sea $S_n = \{r_0, r_1, \ldots, r_{n-1}\}$. Con un arreglo:

$$x : \textbf{array}[0..n-1] \textbf{ of } \{0,1\}$$

se representa un subconjunto de S_n, de manera que, para $0 \le i < n$:

$$x[i]=1 \equiv \text{"}r_i \text{ es elemento del subconjunto"}.$$

El arreglo x se puede interpretar, además, como la representación binaria de un número natural en $[0..2^n-1]$. Más exactamente, x es la representación binaria de

$$\nu x = (+i \mid 0 \le i < n \wedge x[i]=1 : 2^i).$$

Por otro lado, interesa la suma de los elementos de x, para resolver el problema. Defínase:

$$\sigma x = (+i \mid 0 \le i < n \wedge x[i]=1 : r_i).$$

Para determinar si dos arreglos representan conjuntos disyuntos, sea

$$\delta(x,y) = (+i \mid 0 \le i < n \wedge x[i]=y[i]=1 : 1).$$

1 **[35/100]** Exprese los diferentes elementos de una solución con algoritmo de agenda (SOLPOS, sat, …) utilizando la notación anterior.

2 **[10/100]** Argumente si su algoritmo amerita o no:
- **2a** Manejo de nodos marcados
- **2b** Predicado dominó

3 **[10/100]** Estime, en términos de n, el orden de complejidad de la verificación del predicado de satisfacción.

4 **[10/100]** Estime, en términos de n, el orden de complejidad del paso

$$\text{AGENDA} := \text{AGENDA} \cup \text{SUC}(x)$$

5 **[15/100]** Si es posible, estime, en términos de n, el orden de complejidad de su algoritmo.

---

6 **[20/100]** Para cada una de las siguientes afirmaciones, juzgue la veracidad de la misma, y explique su respuesta:
- **6a** **[5/10]** El problema del agente viajero es un problema NP.
- **6b** **[5/10]** Si un problema no está en NP, tampoco puede estar en P.
- **6c** **[10/10]** SAT-1 está en P.

ISIS 211 Diseño de Algoritmos
Semestre 2001-1 - Solución Parcial 3
Mayo 15, 2001
Prof. Rodrigo Cardoso

---

1 ***[35/100]*** *Expresse los diferentes elementos de una solución con algoritmo de agenda (SOLPOS, sat, …) utilizando la notación anterior.*

El conjunto vacío se puede representar con un arreglo de 0's. Llámese 0 a este arreglo. Además, llámese x+1 al arreglo x' tal que $\nu x' = 1 + \nu x$.

SOLPOS = {(x,y) | x : **array**[0..n-1] **of** {0,1}, $x \neq 0$, $y \neq 0$} [5/10]
sat(x,y) $\equiv \sigma x = \sigma y \wedge \delta(x,y) = 0$ [3/10]
SOL = {(x,y):SOLPOS | $\sigma x = \sigma y \wedge \delta(x,y) = 0$} [2/10]

El espacio de búsqueda es el mismo SOLPOS:
BUSQ = SOLPOS. [5/10]

La relación de sucesión entre nodos del espacio de búsqueda se puede definir así:
$(x,y) \rightarrow (x,y+1)$, si $2^n - 1 > \nu y$ [15/10]
$(x,y) \rightarrow (w,0+1)$, si $2^n - 1 = \nu y$, $2^n - 1 > \nu x$.
En esta solución cada elemento tiene exactamente un sucesor. La agenda se reduce a un elemento.

La búsqueda empieza en:
(x,y) = (0+1,0+1). [5/30]

Variantes:
Cualquier enumeración de $[0..2^n-1] \times [0..2^n-1]$ (o bien, de $S^n \times S^n$) logra el mismo resultado (agenda de un elemento).
Ejemplos:
- Enumerar S_n, listando primero los conjuntos con menos elementos. Para $S^n \times S^n$, hacer orden lexicográfico sobre parejas.
- Diagonalizar.
- etc. [15/10]*

2 ***[10/100]*** *Argumente si su algoritmo amerita o no:*

2a *Manejo de nodos marcados*
No es necesario: nunca se repite una pareja (x,y) en la enumeración. [5/10]

2b *Predicado dominó*
No hay. [5/10]

3 ***[10/100]*** *Estime, en términos de n, el orden de complejidad de la verificación del predicado de satisfacción.*

O(sat(x,y)) = O($\sigma x = \sigma y$) + O($\delta(x,y) = 0$) [10/10]
= O(n) + O(n) + O(1) + O(n)
= O(n)

4 ***[10/100]*** *Estime, en términos de n, el orden de complejidad del paso*

AGENDA:=AGENDA $\cup$ *SUC(x)*

El algoritmo corresponde a calcular el sucesor de número natural, en notación binaria. Esto equivale a buscar el `0` menos significativo, cambiarlo por `1` y cambiar por `0`'s todos los `1`'s en posición menos significativa. En total $O(n)$. [10/10]

5 ***[15/100]*** *Si es posible, estime, en términos de n, el orden de complejidad de su algoritmo.*
El algoritmo itera sobre $[0..2^n-1]\times[0..2^n-1]$, efectuando cada vez $O(n)$ operaciones (verificación de `sat` y cálculo de sucesores).
En total $O(n2^{n-1})$. O bien: $O(n2^n)$. [15/10]

6 ***[20/100]*** *Para cada una de las siguientes afirmaciones, juzgue la veracidad de la misma, y explique su respuesta:*

6a ***[5/10]*** *El problema del agente viajero es un problema NP.*
Verdadero. [5/10]
Certificado para una MTAV: un tour. En tiempo lineal se chequea si el costo del tour es menor o igual a la cota del problema.

6b **[5/10]** Si un problema no está en NP, tampoco puede estar en P.
Verdadero. [5/10]
Es la contrarrecíproca de $P \subseteq NP$.

6c **[10/10]** SAT-1 está en P.
Verdadero. [10/10]
Una fórmula de SAT-1 es una conjunción de literales. Si hay n literales, se puede revisar en $O(n^2)$ si un literal y su negación aparecen en la fórmula. Si esto es así, la fórmula no es satisfacible. En otro caso, sí es satisfacible: basta asignar cada literal positivo como true y cada literal negativo como false.