Ejercicio 7.44. del libro de Sipser: UNARY-SSUM

Hecho por Marcos Hidalgo Baños a día 3 de mayo del 2022

Enunciado del problema.

Sea UNARY-SSUM el problema de la suma de subconjuntos en el cual todos los números son representados en unario. ¿Por qué la demostración NP-Completo para SUBSET-SUM falla para demostrar que UNARY-SSUM es NP-Completo? Prueba que UNARY-SSUM ∈ P.

Definición de conceptos.

- **Sistema de numeración unario.**
  - Es el sistema de representación numérica más sencillo. Consiste en representar cualquier número natural N mediante una cadena de unos de longitud N. De esta manera, al número 5 le corresponde la cadena 11111.
- **Problema de la suma de subconjuntos.**
  - También conocido cómo el problema SUBSET-SUM, se pregunta si dado un conjunto no vacío de enteros existe algún subconjunto cuya suma sea otro número objetivo t.

    S = {-7, -3, -2, 5, 8} ➜ Tiene solución para *t = 0* ya que 5 + (-2) + (-3) = 0

<u>Pregunta 1.</u> ¿Por qué la demostración NP-Completo para SUBSET-SUM falla para demostrar que UNARY-SSUM es NP-Completo?

Demostración: SUBSET-SUM ∈ NP-Completo.

1) SUBSET-SUM está en NP.

Dado una instancia del problema y su solución (certificado) podemos **verificar** en tiempo polinómico si es correcta o no, tal y como hemos visto en las transparencias de teoría. Esto lo podemos demostrar mediante el uso de un verificador V que para una instancia del problema SUBSET-SUM con la entrada <<S,t>, c>:

1. Compruebe si c es una colección de números que suma t.
2. Compruebe si S contiene todos los números en c.
3. Si 1 y 2 se cumplen, acepte que y en cualquier otro caso rechace.

También existe una demostración alternativa a la anterior que nos resultará útil en futuras demostraciones, la cual emplea una **mTnD** en tiempo polinómico que se comporte de la siguiente manera:

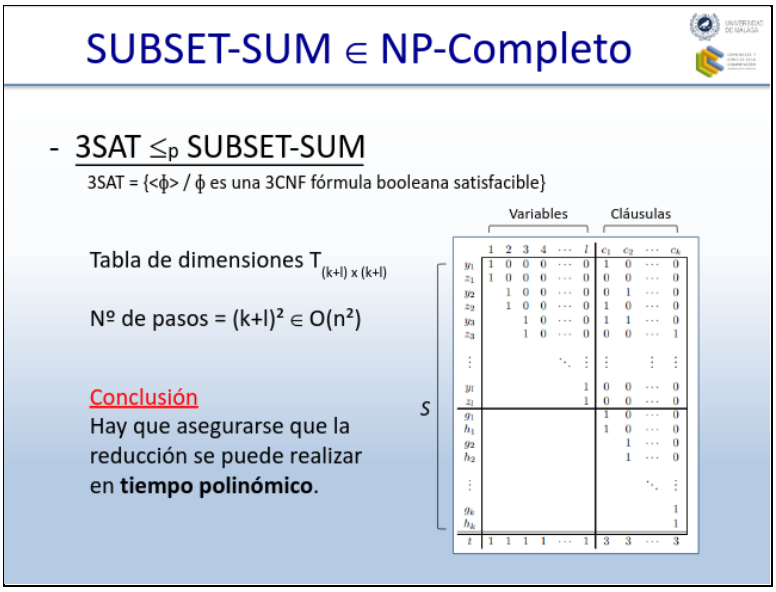
N = “Con la entrada <S,t>

1.- De forma no determinista seleccionar un subconjunto c de los números de S.

2.- Comprobar si c es una colección de números que suma t.

3.- Si es así, aceptar; sino, rechazar."

2) 3SAT ≤p SUBSET-SUM



Esta demostración daría para hacer otra exposición nueva. No es fácil de explicar ni de entender, así que he tomado la decisión de no explicarla en detalle, solamente comentar los aspectos que afectan a la complejidad temporal.

Que la reducción sea posible en tiempo polinómico permite afirmar SUBSET-SUM ∈ NPC.

¿Por qué falla entonces la demostración para UNARY-SSUM?

En SUBSET-SUM podemos representar los elementos del conjunto S en espacio logarítmico gracias a que están representados en binario o cualquier otra base mayor.

*Se necesita de espacio **log(n)** para representar el valor n.*

Sin embargo, si la entrada es de tamaño N su representación en unario es 2^N. Como este crecimiento es **exponencial**, la demostración de que 3SAT ≤p UNARY-SSUM no nos sirve porque la Máquina de Turing requiere de un tiempo mayor que el polinómico.

Pregunta 2. Prueba que UNARY-SSUM ∈ P

Demostración: UNARY-SSUM ∈ P.

Idea general: Como sabemos que SUBSET-SUM ∈ NP, podemos determinar que UNARY-SSUM ∈ P si transformamos el problema para que sea un caso particular de suma de conjuntos en binario.

Entradas:

- Conjunto de números naturales en unario, S.
- Valor objetivo representado en unario, t.

  Si realizamos la transformación unario-binario para cada uno de estos valores veremos que estaremos empleando una sucesión de divisiones enteras empleando un método que no resulta relevante para esta demostración.

  Lo que sí nos concierne es que este proceso tarda **tiempo polinómico** lo cual es imprescindible para demostrar que todo el problema pertenece a la clase de complejidad lineal. Recordemos que este es un mero paso para aligerar la carga computacional durante el siguiente paso, ya que estaremos reduciendo el tamaño de la entrada a **O(log n)**.

Funcionamiento:

Simular la mTnD en una mTD sobre la entrada en binario. Teniendo en cuenta el aumento de complejidad al pasar del modelo no determinista al determinista, es imprescindible transformar la entrada como hemos hecho en el apartado anterior.

$O(2^{\log n}) = O(n)$ ➜ **¡También en tiempo polinómico!**

Referencias.

- Demostración NPC para el problema de la suma de subconjuntos. . https://www.geeksforgeeks.org/subset-sum-is-np-complete/#:~:text=Subset%20Sum%20is%20NP%2DHard,i%2C%20ai%3Di.
- Demostración UNARY-SSUM pertenece a la clase de complejidad P. http://www.cs.virginia.edu/~evans/cs3102-s10/ps/ps6/ps6-comments.pdf