El Problema de la Suma de Subconjuntos es NP-Completo

Carla Sunami Pérez Valera

Septiembre 2024

El problema de la Suma de Subconjuntos es el siguiente: dado n enteros no negativos $w_1, \ldots, w_n$ y una suma objetivo W, la pregunta es decidir si existe un subconjunto $I \subset \{1, \ldots, n\}$ tal que $\sum_{i \in I} w_i = W$. Este es un caso muy especial del problema de la Mochila: en el problema de la Mochila, los elementos también tienen valores v_i, y el problema es maximizar $\sum_{i \in I} v_i$ sujeto a $\sum_{i \in I} w_i \leq W$. Si establecemos $v_i = w_i$ para todos i, la Suma de Subconjuntos es un caso especial del problema de la Mochila que discutimos al considerar la programación dinámica. En esa sección, dimos un algoritmo para el problema que se ejecuta en tiempo $O(nW)$. Este algoritmo funciona bien cuando W no es demasiado grande, pero notamos que este algoritmo no es un algoritmo de tiempo polinómico. Para escribir un entero W, solo necesitamos $\log W$ dígitos. Es natural suponer que todos $w_i \leq W$, y así la longitud de la entrada es $(n+1) \log W$, y el tiempo de ejecución de $O(nW)$ no es polinómico en esta longitud de entrada.

En este documento, mostramos que, de hecho, la Suma de Subconjuntos es NP-completa. Primero mostramos que la Suma de Subconjuntos está en NP.

1 Afirmación 1.

La Suma de Subconjuntos está en NP.

Demostración. Dado un conjunto propuesto I, todo lo que tenemos que hacer es probar si efectivamente $\sum_{i \in I} w_i = W$. Sumar como máximo n números, cada uno de tamaño W, toma $O(n \log W)$ tiempo, lineal en el tamaño de la entrada.

Para establecer que la Suma de Subconjuntos es NP-completa, probaremos que es al menos tan difícil como SAT.

2 Teorema 1.

SAT $\leq$ Subset Sum.

Demostración. Para probar la afirmación, necesitamos considerar una fórmula Φ, una entrada para SAT, y transformarla en una entrada equivalente

para la Suma de Subconjuntos. Supongamos que Φ tiene n variables $x_1, \ldots, x_n$, y m cláusulas $c_1, \ldots, c_m$, donde la cláusula c_j tiene k_j literales.

Definiremos nuestro problema de Suma de Subconjuntos utilizando una base muy grande B, por lo que escribiremos números como $\sum_{j=0}^{n+m} a_j B^j$, y estableceremos la base B como $B = 2^{\max_j k_j}$, lo que asegurará que las sumas entre nuestros números nunca causen un acarreo.

Escrito en base B, los dígitos $i = 1, \ldots, n$ corresponderán a las n variables $x_1, \ldots, x_n$, y el objetivo de estos dígitos será asegurarse de que establezcamos cada variable como verdadera o falsa (y no ambas). Tendremos dos números w_i y w_{i+n} correspondientes a la variable x_i establecida como verdadera o falsa, y el dígito i se asegurará de que usemos uno de w_i y w_{i+n} en cualquier solución. Para hacer esto, establecemos los dígitos i-ésimos de W, w_i y w_{i+n} como 1, y establecemos este dígito en todos los demás números como 0.

Los siguientes m dígitos corresponderán a las m cláusulas, y el dígito objetivo $n + j$ es asegurarse de que la j-ésima cláusula sea satisfecha por nuestra configuración de las variables.

El valor objetivo será $W = \sum_{i=1}^{n} B^i + \sum_{j=1}^{m} k_j B^{n+j}$.

Comenzamos definiendo $2n$ números, uno para cada uno de los literales x_i y $\neg x_i$. Los dígitos $1, \ldots, n$ se asegurarán de que cualquier subconjunto que sume W usará exactamente 1 de los dos números x_i y $\neg x_i$, y los siguientes m dígitos tendrán como objetivo garantizar que cada cláusula sea satisfecha. Necesitaremos algunos números adicionales que definiremos más adelante.

El número correspondiente al literal x_i es el siguiente: $w_i = B^i + \sum_{j:x_i \in c_j} B^{n+j}$, mientras que el número correspondiente al literal $\neg x_i$ es $w_{i+n} = B^i + \sum_{j:\neg x_i \in c_j} B^{n+j}$. Si sumamos un conjunto de n números $W = k_m k_{m-1} \ldots k_2 k_1 11 \ldots 110$ con los dígitos $n + 1, \ldots, n + m$ y los dígitos $1, \ldots, n$:

Dígitos	$n+1, \ldots, n+m$	**Dígitos**	$1, \ldots$
w_i	0	0	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$
1 en la posición $n + j$ si x_i es verdadero	1 en la posición i si i está en la cláusula c_j		

La figura 1 muestra el total W y los números w_i y w_{i+n}.

Correspondiente a una asignación de verdad satisfactoria para Φ, obtenemos una suma de la forma $\sum_{i=1}^{n} B^i + \sum_{j=1}^{m} b_j B^{n+j}$ donde b_j es el número de literales verdaderos en la cláusula c_j. Dado que esta fue una asignación satisfactoria, debemos tener $b_j \geq 1$.

Como detalle final, agregaremos $k_j - 1$ copias del número B^{n+j} para todas las cláusulas c_j. Esto define ahora nuestro problema de suma de subconjuntos, con el objetivo W y los $2n + \sum_j (k_j - 1)$ números definidos, utilizando estos números adicionales nos permitirá alcanzar exactamente W.

Para probar que esta es una reducción válida, necesitamos establecer dos afirmaciones a continuación que establecen la dirección "si" y "solo si" de la prueba respectivamente.

3 Afirmación 2.

Si el problema SAT definido por la fórmula Φ es solucionable, entonces el problema de la Suma de Subconjuntos que acabamos de definir con $2n - m + \sum_j k_j$ números también es solucionable.

Demostración. Supongamos que tenemos una asignación satisfactoria para la fórmula Φ, primero consideramos agregar los números que corresponden a los literales verdaderos. Usamos exactamente uno de w_i y w_{i+n}, por lo que tendremos 1 en el dígito i, y obtenemos una suma que es de la forma $\sum_{i=1}^{n} B^i + \sum_{j=1}^{m} a_j B^{n+j}$.

Además, tendremos $1 \leq a_j \leq k_j$, donde a_j es al menos 1, ya que la asignación satisface la fórmula, por lo que al menos uno de los números sumados tiene un 1 en el dígito $n+j$, y como máximo k_j ya que incluso sumando todos los números, como máximo k_j de ellos tiene un 1 en el dígito $n+j$. En particular, con $B > k_j$, no habrá acarreos.

Para hacer que esta suma sea exactamente W, agregamos $k_j - a_j$ copias del número B^{n+j} que agregamos al final de la construcción.

A continuación, necesitamos probar la otra dirección:

4 Afirmación 3.

Si el problema de la Suma de Subconjuntos que acabamos de definir con $2n - m + \sum_j k_j$ números es solucionable, entonces el problema SAT definido por la fórmula Φ es solucionable.

Demostración. Primero notamos que para cualquier subconjunto que podamos agregar, nunca habrá un acarreo en ningún dígito. Para ver por qué, notemos que todos los números a sumar tienen todos los dígitos 0 o 1; para el dígito $i = 1, \ldots, n$ tenemos dos números con un 1 en ese dígito w_i y w_{i+n}; el dígito 0 siempre es 0; y para el dígito $n + j$ tenemos exactamente $2 \max_j k_j - 1$ números que tienen un 1 en ese dígito: k_j correspondientes a los k_j literales en la cláusula, y $k_j - 1$ números adicionales B^{n+j} que agregamos al final. Así que incluso si sumamos todos los números, no podemos causar un acarreo en ninguno de los dígitos.

Basado en la observación anterior sobre no tener acarreos, para obtener el número W, necesitamos encontrar un subconjunto I que tenga exactamente el número correcto de 1's en cada dígito. Primero nos enfocamos en los dígitos $1, \ldots, n$. Este dígito en W es un 1, y los dos números que tienen un 1 en este dígito son w_i y w_{i+n}; para sumar W, debemos usar exactamente uno de estos, dejando $I_0 \subset I$ correspondiente a los literales. Esto muestra que los números seleccionados entre los primeros $2n$ de ellos corresponden a una asignación de verdad de las variables $x_1, \ldots, x_n$.

Finalmente, necesitamos mostrar que esta asignación de verdad satisface la fórmula Φ. Consideremos la suma $W_0 = \sum_{j \in I_0} w_j$, simplemente agregando el subconjunto I que corresponde a las variables. Notemos que $W_0 = \sum_{i=1}^{n} B^i + \sum_{j=1}^{m} a_j B^{n+j}$ con $a_j \leq k_j$. Necesitamos mostrar que $a_j \geq 1$, lo que probará que

tenemos una asignación satisfactoria. Recordemos que el subconjunto I suma exactamente W. Para poder extender I_0 con un subconjunto de los números adicionales para sumar W, debemos tener $a_j \geq 1$ ya que solo hay $k_j - 1$ copias de B^{n+j}.