Reducción de Partición a Bin Packing y Bin Packing a Batalla Naval

Juan Miguel Pérez Martínez c412

September 23, 2024

1 Reducción de Partición a Bin Packing

El problema de Bin Packing es NP-completo. Más específicamente:

Teorema 8.1. Es NP-completo decidir si una instancia de Bin Packing admite una solución con dos contenedores.

Demostración: Reducimos desde Partición, la cual sabemos que es NP-completa. Recordemos que en el problema de Partición, se nos dan n números $c_1, \ldots, c_n \in \mathbb{N}$ y se nos pide decidir si existe un conjunto $S \subseteq \{1, \ldots, n\}$ tal que

$$\sum_{i \in S} c_i = \sum_{i \notin S} c_i.$$

Dada una instancia de Partición, creamos una instancia para Bin Packing estableciendo $s_i = \frac{2c_i}{\sum_{j=1}^{n} c_j} \in (0, 1]$ para $i = 1, \ldots, n$. Obviamente, dos contenedores son suficientes si y solo si existe un $S \subseteq \{1, \ldots, n\}$ tal que

$$\sum_{i \in S} c_i = \sum_{i \notin S} c_i.$$

Esto nos permite derivar una cota inferior sobre la aproximabilidad de Bin Packing.

Corolario 8.2. No existe un algoritmo de aproximación con factor $\rho < \frac{3}{2}$ para Bin Packing a menos que $P = NP$.

2 Reducción de Bin Packing a Batalla Naval

El problema de Batalla Naval es NP-completo.

Definición: El problema se define de la siguiente manera. Un rompecabezas de Batalla Naval consiste en una cuadrícula de rompecabezas, un conteo de filas y columnas, y una flota que contiene un cierto número de barcos de longitudes variables. La cuadrícula inicial está parcialmente llena con segmentos de barcos o agua. El objetivo del rompecabezas es demostrar que los barcos proporcionados pueden colocarse en la cuadrícula de acuerdo con las siguientes cuatro condiciones:

- Todos los barcos de la flota se colocan en la cuadrícula;
- Se respetan las indicaciones en la cuadrícula inicial;
- Ningún barco ocupa casillas adyacentes (ni ortogonal ni diagonalmente);
- El número de segmentos de barco en la columna (fila) i es igual al valor i-ésimo del conteo de columnas (filas).

Podemos reducir el problema de Bin Packing al rompecabezas de Batalla Naval de la siguiente manera:

Entrada: n enteros positivos $a_1, \ldots, a_n$ (los objetos), dos enteros C (la capacidad) y B (el número de contenedores) tal que $a_1 + \cdots + a_n = CB$.

Pregunta: ¿Se puede particionar $a_1, \ldots, a_n$ en B subconjuntos, cada uno de los cuales tiene una suma total exactamente igual a C?

Transformamos una instancia de Bin Packing en un rompecabezas de Batalla Naval reservando una franja vertical de longitud a_i por cada par de objeto a_i y contenedor $b \in \{1, \ldots, B\}$. Todas las demás casillas se llenan inicialmente con agua. Nótese que esta reducción asume que los números $a_1, \ldots, a_n$ están representados en unario, ya que las franjas mismas son representaciones unarias de estos números. Esta suposición no afecta el resultado, ya que el problema de Bin Packing sigue siendo NP-completo si sus números se representan en unario. Esta propiedad hace que el problema sea NP-completo fuerte.

La idea es que cada franja abierta representa la posibilidad de que el objeto a_i sea colocado en el contenedor b. El objeto mismo está representado por un barco de longitud a_i. Para asegurarnos de que cada objeto a_i se coloque en solo un contenedor, tenemos 1's en el conteo de filas correspondiente. Esto también asegura que cada barco de longitud a_i se coloque en una franja de longitud exactamente a_i.

Esta reducción prueba que Batalla Naval es NP-difícil. Es fácil ver que una conjetura no determinista puede verificarse como una solución en tiempo polinomial. Por lo tanto, Batalla Naval es NP-completo.

Concluciones del primer intento de reduccion:

Esta reduccion no me convence ya que en la primera parte de Partición a Bin Packing solo reduzco para una instancia del problema por lo que seguí investigando y llegué a la realiacion de otra reducción

3 Reducción de 3-SAT a 3,4-SAT

Comience con cualquier instancia de 3-SAT. Para cada variable x que aparece en más de tres cláusulas (por 'aparece' nos referimos a que ella o su complemento está en la cláusula), realice el siguiente procedimiento: Suponga que x aparece en k cláusulas. Cree k nuevas variables $x_1, \ldots, x_k$ y reemplace la i-ésima ocurrencia de x con x_i, $i = 1, \ldots, k$. Añada la cláusula $\{x_i \vee \neg x_{i+1}\}$ para $i = 1, \ldots, k-1$ y la

cláusula $\{x_k \vee \neg x_1\}$. En la nueva instancia, la cláusula $\{x_i \vee \neg x_{i+1}\}$ implica que si x_i es falso, x_{i+1} también debe ser falso. La estructura cíclica de las cláusulas, por lo tanto, obliga a que los x_i sean todos verdaderos o todos falsos, por lo que la nueva instancia es satisfacible si la original lo es. Además, la transformación requiere tiempo polinomial.

Teorema 2.1. La satisfacibilidad booleana es NP-completa cuando se restringe a instancias con 2 o 3 variables por cláusula y como máximo 3 ocurrencias por variable.

Un corolario interesante es:

Corolario 2.2. Para cualquier $s \geq 3$, o bien (i) toda expresión booleana con exactamente 3 variables por cláusula y no más de s ocurrencias por variable es satisfacible, o (ii) 3,s-SAT es NP-completo.

Prueba. Suponga que (i) no se cumple, por lo que existe una expresión insatisfacible en las variables $x_1, y, z, \ldots$. Sin pérdida de generalidad, la primera cláusula de la expresión incluye una 'x' sin complementar. Sea B el resto de la expresión; claramente podemos asumir que B es satisfacible. B ahora tiene las propiedades de que x aparece como máximo $s-1$ veces, y que solo puede ser satisfecho cuando x es falso.

Ahora considere una instancia arbitraria de 3-SAT y realice el procedimiento en la construcción del Teorema 2.1. Para la i-ésima cláusula, $(a \vee b)$ que contiene dos variables, añada una i-ésima copia de B usando las variables $x_i, y, z, \ldots$, y cambie la cláusula a $(a \vee b \vee x_i)$. Así que si (i) es falso, entonces (ii) es verdadero. Note que (i) y (ii) no pueden ser ambos verdaderos a menos que $P = NP$.

Teorema 2.3. 3,4-SAT es NP-completo.

Prueba. Lo único que falta en la construcción del Teorema 2.1 es que las cláusulas $(x_i \vee x_{i+1})$ contienen solo dos variables. Para cada una de estas cláusulas, introduzca una nueva variable y_i, de modo que la cláusula se convierta en $(x_i \vee x_{i+1} \vee y_i)$. Ahora note que podemos forzar a que cada y_i sea verdadero mediante las siguientes cláusulas en las que y_i aparece solo tres veces. La construcción es, sospechamos, lo más pequeña posible. Las primeras tres cláusulas requieren que y_i sea verdadero si cualquiera de los pares a_i, b_i son ambos falsos; las otras diez cláusulas fuerzan a que esto suceda:

$$\{y_i \vee a_i \vee b_i\}, \quad i = 1, \ldots, 3,$$

$$\{d_i \vee D_i \vee b_i\}, \quad \{d_i \vee O_i \vee b_i\}, \quad \{d_i \vee a_i \vee b_i\}, \quad j = 1, \ldots, 3,$$

$$\{G_i \vee D_i \vee b_i\}.$$

En la reducción real, habría una copia de lo anterior para cada y_i; hemos suprimido el subíndice 'i' de cada variable para mayor legibilidad. Si la instancia original de 3-SAT tiene m cláusulas, la instancia 3,4-SAT tendrá $m+3m+39m = 43m$ cláusulas. El Corolario 2.2 y el algoritmo en la Sección 3 demuestran que no hay un nivel intermedio de complejidad en los problemas 3,s-SAT entre lo virtualmente trivial y lo NP-completo. El único caso no resuelto es 3,3-SAT.

Teorema 2.4.1

Toda instancia de (r, r)-SAT es satisfactoria.

Demostración. Denotemos las variables por $x_1, x_2, \ldots, x_n$ y las cláusulas por $c_1, \ldots, c_m$, donde $m \leq n$ (y $m = n$ solo si cada variable aparece en exactamente r cláusulas). Sea $A = A_1, \ldots, A_m$ la familia de conjuntos (no necesariamente distintos) de los x_i definidos por $A_i = \{x_j \mid x_j \in c_i \text{ o } x_j \in c_j\}$.

Consideremos cualquier unión de k de los conjuntos A_i. Dado que cada A_i contiene r elementos distintos y ningún x_j está contenido en más de r conjuntos, la unión contiene al menos k elementos distintos. Entonces, por el Teorema de Philip Hall, existe un sistema de representantes distintos de A. Es decir, existe una inyección de las cláusulas c_i a las variables x_j tal que cada cláusula contiene la variable (o el complemento de la variable) a la que está asignada. Es trivial, dado tal mapeo, satisfacer cada cláusula, y por lo tanto, la instancia de (r, r)-SAT.

Conjetura 2.5

Si $s \leq 2^{r-1} - 1$, entonces toda instancia de (r, s)-SAT es satisfactoria.

El significado intuitivo del término $2^{r-1}-1$ es el número mínimo de cláusulas de tamaño r requerido para forzar que una variable sea verdadera, si las variables 'forzadoras' están restringidas a la mitad de sus posibles valores de verdad. Por ejemplo, solo 7 de las 16 combinaciones de verdad de $y_1, \ldots, y_4$ restringen a un valor en las cláusulas $\{t \vee y_1 \vee y_2, t \vee y_3 \vee y_4\}$. En el caso $(3, 4)$, la fracción análoga es $37/64$, que es más de $1/2$, por lo que la 'ola' de implicaciones de los y a z se vuelve más fuerte y es posible forzar una contradicción.

4 Existencia de una reducción parsimoniosa de 3-SAT a nuestro problema de decisión de Battleships

Teorema: Existe una reducción de 3-SAT a BATTLESHIPS.

Demostración: Probaremos esto proporcionando dos reducciones: una de 3-SAT a un problema intermedio 3, {3, 4}-SAT, y la otra de 3, {3, 4}-SAT a BATTLESHIPS. 3-SAT es el conjunto de fórmulas booleanas satisfacibles que están en forma normal conjuntiva y tienen tres literales por cláusula. Como es habitual, asumimos que ninguna variable aparece dos veces en una sola cláusula. 3, {3, 4}-SAT es el subconjunto de 3-SAT que contiene todas las fórmulas satisfacibles con exactamente tres variables por cláusula y cada variable aparece tres o cuatro veces. Decidir si una instancia de 3, {3, 4}-SAT es satisfacible fue probado como NP-completo anteriormente; el resultado se estableció mediante una reducción de 3-SAT.

Dedicaremos el resto de esta demostración a una reducción de 3, {3, 4}-SAT a BATTLESHIPS. Para ello, sea $\phi \equiv C_1 \wedge \ldots \wedge C_m$ una instancia de 3, {3, 4}-

SAT sobre las variables $x_1, \ldots, x_n$. Asociamos con cada cláusula C_i y variable x_j una región X_{ij}. Además, cada cláusula C_i (variable x_j, respectivamente) está asociada con una región marcada Y_i (Z_j, respectivamente). Construimos el rompecabezas colocando dispositivos en las regiones marcadas con X, Y y Z junto con la flota y los conteos de filas y columnas, usando ϕ. Terminamos la construcción fijando los valores de conteo restantes. Procedemos de la siguiente manera:

- Para X_{ij}: si x_j aparece positivamente en C_i, colocamos un dispositivo específico; si x_j aparece negativamente en C_i, colocamos otro dispositivo; de lo contrario, si x_j no aparece en C_i, colocamos un dispositivo de agua. Todos los dispositivos tienen una altura de $8i + 3$. Si x_j aparece en C_i, añadimos un barco de longitud 1, el barco X_{ij}, a la flota.

- Para Y_i: colocamos un dispositivo de altura $8i + 3$; añadimos un barco de longitud $4i + 1$, un barco de longitud $4i$ y un barco de longitud 1. Los barcos añadidos los llamamos barcos Y_i. Este dispositivo también determina el conteo de filas para las filas que intersectan Y_i.

- Para Z_j: si x_j aparece cuatro veces en ϕ, colocamos un dispositivo específico y añadimos un barco de longitud 4; de lo contrario, si x_j aparece tres veces, colocamos otro dispositivo y añadimos un barco de longitud 3. El barco añadido lo llamamos barco Z_j. Este dispositivo también determina el conteo de filas para las filas que intersectan Z_j.

- Concluimos la construcción asignando valores de conteo a las filas más bajas y las columnas más a la derecha, que hasta ahora permanecían abiertas:

Conteo de filas	**Conteo de columnas**
$9m + 1$	0
$3n + 1$	$\sum_{1 \leq i \leq m}(4i + 1)$
$9m + 2$	n_4
$3n + 2$	0
$9m + 3$	n
$3n + 3$	$\sum_{1 \leq i \leq m}(4i + 1)$
$9m + 4$	n
$9m + 5$	n

Table 1: Conteo de filas y columnas.

donde n_4 es igual al número de variables que aparecen cuatro veces en ϕ.

Basta con mostrar que el número de asignaciones de verdad que satisfacen ϕ es igual al número de soluciones del rompecabezas de Battleships anterior. De hecho, mostraremos que cada asignación de verdad satisfactoria de ϕ corresponde exactamente a una solución de la reducción de ϕ, y viceversa.

Supongamos que ϕ tiene una asignación de verdad satisfactoria $t : \{x_1, \ldots, x_m\} \to \{true, false\}$. Entonces resolveremos el rompecabezas reducido usando t de la siguiente manera. Para cada variable x_j que aparece en la cláusula C_i, colocamos el barco X_{ij} en la ranura noreste, si x_j aparece positivamente en C_i y $t(x_j) = true$; en la ranura suroeste, si x_j aparece positivamente en C_i y $t(x_j) = false$; en la ranura sureste, si x_j aparece negativamente en C_i y $t(x_j) = true$; en la ranura noroeste, si x_j aparece negativamente en C_i y $t(x_j) = false$. Es inmediato que t está codificado de manera única por la posición de los barcos X_{ij}.

Si la asignación de verdad de x_j contribuye a la verdad de su cláusula C_i depende de si x_j aparece negativamente o positivamente en C_i. Por ejemplo, si x_j aparece negativamente en C_i y se asigna como verdadero, entonces el barco X_{ij} debe colocarse en la ranura abierta derecha. En consecuencia, se coloca en la fila inferior. Un barco X_{ij} se coloca en la ranura abierta inferior si no contribuye a que C_i sea verdadero. Dado que t es una asignación satisfactoria, al menos uno de los literales de C_i es verdadero. En el rompecabezas de Battleships, esto corresponde a al menos un barco X_{ij} colocado en la fila superior. El dispositivo en Y_i y su conteo pueden satisfacerse para cualquier disposición de los barcos X_{ij} donde el número de barcos en la fila superior (inferior) sea al menos uno (dos). En cuanto al número de barcos X en la fila superior, hacemos una distinción de casos. Supongamos que la fila superior contiene tres barcos X, entonces el conteo de filas correspondiente ya está satisfecho. El resto del dispositivo Y_i solo puede resolverse organizando los barcos Y_i de manera específica.

Si la fila superior contiene dos barcos X, entonces hay varias posibles soluciones. Descartamos una de las soluciones exigiendo que la primera y tercera columna de los dispositivos Y contengan $\sum_{1 \leq i \leq m} 4i + 1$ segmentos de barco. Si se eligiera un patrón incorrecto, nunca se podría resolver el rompecabezas debido a esta restricción.

Supongamos que la fila superior contiene solo un barco X, entonces debe respetarse la solución única del rompecabezas, que garantiza que los dispositivos Y_i se resuelvan correctamente.

A partir de la definición de t, se sigue que cada variable x_j se asigna a un valor de verdad único. Si $t(x_j) =$ true (es decir, verdadero), entonces el barco se coloca en la columna derecha. Si $t(x_j) =$ false (falso), entonces el barco se coloca en la columna izquierda.

Ahora, supongamos que $t(x_j) =$ true, entonces el conteo de columnas derecho de 3 (o 4, en caso de que x_j aparezca cuatro veces en ϕ) se satisface; el conteo de columnas izquierdo de 3 (o 4) se satisface colocando el barco Z_j en la ranura izquierda, y así sucesivamente. De esta manera, el posicionamiento de los barcos asegura que la asignación de verdad se refleje correctamente en el tablero del rompecabezas.

Esto completa la construcción del rompecabezas de BATTLESHIPS y muestra que cualquier solución válida al rompecabezas corresponde a una asignación de verdad satisfactoria para ϕ. Por lo tanto, hemos demostrado la existencia de una reducción de 3-SAT a BATTLESHIPS.

5 En búsqueda de una solución

5.1 Backtracking

El **backtracking** es una técnica de búsqueda que explora todas las configuraciones posibles para encontrar una solución válida. Esta es la primera opción que nos viene a la mente. Este método se basa en construir soluciones parciales y, si se determina que una solución parcial no puede conducir a una solución completa válida, se retrocede y se prueba otra opción.

5.1.1 Características del Backtracking

- **Exploración Exhaustiva:** El backtracking intenta todas las combinaciones posibles hasta encontrar una solución o determinar que no existe.
- **Condiciones de Parada:** Se detiene cuando encuentra una solución válida o cuando ha explorado todas las opciones posibles.
- **Aplicaciones en Battleship:** En el contexto del rompecabezas de Battleship, el backtracking puede ser utilizado para colocar barcos en el tablero respetando las restricciones del juego (como no superponerse o tocarse).

Este método es efectivo para resolver el rompecabezas, aunque puede ser ineficiente para tableros grandes o flotas complejas debido a su naturaleza exponencial.

5.2 Algoritmos Genéticos

Los **algoritmos genéticos** son una técnica inspirada en la evolución natural. Se utilizan para encontrar soluciones óptimas mediante un proceso de selección, cruce y mutación de soluciones candidatas.

5.2.1 Funcionamiento

1. Se inicia con un conjunto inicial de soluciones (población).
2. Las soluciones se evalúan según un criterio de aptitud (fitness).
3. Se seleccionan las mejores soluciones para crear nuevas generaciones a través del cruce y la mutación.
4. Este proceso se repite hasta que se encuentra una solución satisfactoria o se alcanza un número máximo de generaciones.

Los algoritmos genéticos son útiles cuando el espacio de búsqueda es muy grande, ya que permiten explorar múltiples configuraciones simultáneamente. Este método siempre proporciona una solución, aunque no necesariamente la ideal. Decidí implementar este enfoque, ya que, aunque en muchos casos no da soluciones óptimas, considero que puede ofrecer algunos buenos resultados.