Diseño y Análisis de Algoritmos. Problema 3: La Pregunta

Jesús Santos Capote y Kenny Villalobos Morales

Facultad de Matemática y Computación, Universidad de La Habana, La Habana, Cuba

1. Definición del Problema

Se tiene una expresión booleana y se quiere conocer si existe una asignación a las variables de esta, tal que la expresión se torne falsa.

2. Primera Aproximación

Dada la expresión brindada, sea esta f, si existe una asignación s de las variables de f tal que la expresión sea falsa, entonces dicha asignación hace verdadera la expresión $\neg f$, por lo que el problema de encontrar una asignación de variables que hagan falsa f sea soluciona hallando una asignación de variables que satifaga $\neg f$. Para verificar la satisfacibilidad de la fórmula f es más sencillo trabajar sobre una fórmula en CNF, por lo cual hacemos la transformación de f a f', siendo f' una fórmula en CNF equisatisfacible con f. Para esta transformación utilizamos el algoritmo Tseitin, que nos garantiza que dicha transformación se desarrolla en tiempo polinomial, y con un crecimiento de la fórmula polinomial con respecto al tamaño de la fórmula original.(Ver algoritmo de Tseitin en ...)

En este punto buscamos una asignación de variables tal que la fórmula f' quede satisfecha, por tanto, procedemos a verificar todas las posibles asignaciones de variables

2.1. Idea del Algoritmo

La idea de esta aproximación para encontrar una distribución de valores de las variables, que satisfagan la CNF es mediante Backtrack, probar todas las posibles asignaciones de las variables y comprobar si alguna de ellas hace que la CNF sea satisfacible. Si no existe tal distribución el algoritmo lo notifica.

2.2. Correctitud

Este algoritmo es correcto pues se comprueban todas las posibles asignaciones de las variables. Por tanto, si existe una distribución de valores que hace que la CNF se satisfaga, el algoritmo la detecta. También detecta si no existe forma de satisfacer la CNF.

2.3. Complejidad Temporal

Sea n la cantidad de variables. Sea c la cantidad de cláusulas de la CNF. La cantidad de distribuciones de valores booleanos de las variables de la CNF es 2^n. Luego para cada distribución se comprueba si satisface la CNF. La comprobación de satisfacibilidad tiene complejidad $O(n*c)$ pues se recorren todas las variables de todas las cláusulas asignando sus valores y evaluando las cláusulas durante el recorrido y luego se verifica el valor de las evaluciones de todas las cláusulas en $O(c)$ la cnf durante el recorrido. Por tanto el algoritmo tiene complejidad $O(n * c * 2^n)$

3. Reducción del Problema

Sea A el problema que estamos tratando de resolver, y sea B el conocido problema NP-Completo 3-Sat.(Ver problema 3-Sat) Supongamos que tenemos un algoritmo f que soluciona A en tiempo polinomial. Para toda instancia β de B podemos transformar esta en una instancia α de A, de forma que la solución de las instancias α de A y β de B son la misma, es decir la respuesta de α es "si"si y solo si la respuesta de β es si. Este procedimiento de reducción polinomial es tan sencillo como tomar para cada instancia β de 3-Sat y convertirla a α, siendo $\alpha = \neg$. Si el algoritmo f retorna $True$, es porque existe una asignación de variables para α tal que está es $False$, luego para esta misma asignación de variables $lnot\beta = False$, por lo cual $\beta = True$

4. Algoritmo Walk-Sat

5. Utilización del algoritmo Walk-Sat

Walk-Sat es un algoritmo estocástico de búsqueda local(focused search) ...breve explicacion de xq usar walksat...

Para llevar una instancia de nuestro problema a una instancia de Walk-Sat utilizamos la transformación de Tseitin, dado que esta nos permite llevar una expresión booleana a 3-CNF en tiempo polinomial y con un crecimiento de la fórmula polinomial con respecto al tamaño de la fórmula original.

5.1. Idea del algoritmo

Comienza asignando un valor aleatorio a cada variable de la fórmula. Si la asignación satisface todas las cláusulas, el algoritmo termina devolviendo la asignación. De lo contrario, se invierte una variable seleccionada bajo un criterio de una cláusula insatisfecha elegida aleatoriamente y se repite lo anterior hasta que todas las cláusulas estén satisfechas. El algoritmo puede utilizar diferentes criterios, puede seleccionar una variable de la cláusula de forma aleatoria, o bien puede seguir un criterio greedy de los implementados para seleccionar la variable.

5.2. Criterios greedy

Minimizar la insatisfacción de cláusula que ya estaban satisfechas. Este criterio escoge la variable de la cláusula que aparezca menos veces como única variable que satisface una cláusula, dado que cambiar esta haría que menos cláusulas se insatisfagan.

Maximizar la cantidad de cláusulas que se satisfacen Este criterio escoge la variable de la cláusula que aparezca en más cláusulas insatisfechas, dado que cambiar esta satisfacería dichas cláusulas.

5.3. Eficacia del algoritmo

6. Contenidos preliminares.

6.1. Problema de satisfacibilidad booleana. 3-SAT.

El problema SAT es el problema de saber si, dada una expresión booleana con variables y sin cuantificadores, hay alguna asignación de valores para sus variables que hace que la expresión sea verdadera. El problema 3-SAT restringe la expresión de entrada a Forma Normal Conjuntiva(CNF) con 3 variables por cláusula (3-CNF).

6.2. Algoritmo de Tseitin

Dada una fórmula ϕ identificamos todas las subfórmulas ϕ_i que la componen (subfórmulas entre dos variables a lo sumo) e introducimos una nueva variable y_i por cada una de estas. Luego reescribimos la fórmula original como la consecución de *and* entre la la variable correspondiente a la subfórmula mayor y las clásulas de la forma: $y_i \Leftrightarrow \phi_i$. Luego, cada una de estas cláusulas debe ser llevada a CNF, utilizando el algoritmo de transformación basado en el uso de la tabla verititativa, como cada cláusula tiene tres variables a lo sumo, su representación en FNC dada por este método es un expresión en con menos de tres variables por cláusula. Luego para conseguir que cada clásula C_i tenga exactamente 3 variables, se sigue el procedimiento:

•SiC$_i$ tiene 3 literales distintos, simplemente incluimos C_i como una cláusula de la fórmula original.

•SiC$_i$ tiene 2 literales distintos, dígase $C_i = (l_1 \vee l_2)$, entonces incluimos como cláusulas $(l_1 \vee l_2 \vee p) \wedge (l_1 \vee l_2 \vee \neg p)$

•SiC$_i$ tiene soll un literal l, entonces incluimos las cláusulas $(l \vee p \vee q) \wedge (l \vee p \vee \neg q) \wedge (l \vee \neg p \vee q) \wedge (l \vee \neg p \vee \neg q)$