Universidad Nacional Autónoma de México
Complejidad Computacional

SAT (MaxSAT)

Emmanuel Cruz Hernández
314272588

Índice

1. Problema

Dada una fórmula f en su forma normal conjuntiva con variables booleanas x_1, x_2, ..., x_n y pesos no negativos llamados w_c para cada cláusula c en la fórmula *f*. El problema consiste en dar una asignación de verdad que maximice el peso total de satisfacibilidad de las cláusulas. Sea C el conjunto de cláusulas de *f*. Como f es una fórmula en forma normal conjuntiva, cada una de ellas es una disyunción de literales y cada literal es una variable booleana o su negación. Denotemos *size(c)* el tamaño de la cláusula c, es decir, la cantidad de literales contenidas en la cláusula. Se supone que la cantidad de literales en una cláusula es arbitraria.

Se denota MAX-kSAT a la restricción de MAX-SAT, para cualquier entero positivo k, a las instancias cuya longitud sea de al menos k. Se tiene que MAX-SAT es un problema NP-Hard.

A contunuación se van a mostrar dos algoritmos de apoximación para Max-SAT. El primero funciona mejor con cláusulas grandes con una garantía de *1/2* y el segundo funciona mejor con cláusulas pequeñas con una garantía de *1-1/e*.

La notación que se usará en los algoritmos es la siguiente. W denota el peso total de cláusulas satisfacibles y para cada cláusula c la variable W_c denota el peso contribuido por la cláusula c en W y decimos que $E[W_c] = w_c$ **Pr**[c es satisfacible].

2. Algoritmos de aproximación

2.1. Algoritmo para cláusulas grandes

Primero se asigna un valor *true* a cada variable independientemente una de otra con una probabilidad de *1/2* y la asignación de verdad resultante la llamaremos τ. Finalmente, definimos α_k=1-2^{-k} para k $\geq$ 1. Para el algoritmo se usa el lema que se muestra y demuestra a continuación.

Lema 1: *Si size(c)=k, entonces* $E[W_c]=\alpha_k w_c$

Demostración. Una cláusula c no es satisfacible por ρ si y sólo si todas las literales son *false*. La probabilidad de este evento es 2^{-k}.

□

Para alguna $k \geq 1$, tenemos que $\alpha_k \geq 1/2$.

Además, por la expresión lineal $\mathrm{E}[\mathrm{W}] = \sum_{e \in C} \mathrm{E}[\mathrm{W}_c] \geq \frac{1}{2} \sum_{c \in C} \mathrm{w}_c \geq \frac{1}{2}$ OPT, donde se ha usado un límite superior que es OPT. Este valor es el peso total de las cláusulas en C.

Con lo anterior se muestra como desaleatorizar el procedimiento en vez de convertirlo en una solución de alta probabilidad. El algoritmo resultante entonces puede computar determinísticamente una asignación de verdad tal que se cumple que los pesos de las cláusulas satisfacibles es $\geq \mathrm{E}[\mathrm{W}] \geq$OPT. Se puede notar también que α_k aumenta con k y la garantía del algoritmo es $\frac{3}{4}$ si cada cláusula tiene dos o más literales.

2.2. Quitando aleatoriedad con el método de *expectativa condicional*

Para este algoritmo se usará la autoreducción del problema del SAT. Sea T el árbol de reducción para la fórmula f. Cada nodo interno en el nivel i corresponde a la asignación de variables booleanas x_1, ..., x_i y cada hoja representa una asignación completa a las n variables. Lo que haremos ahora es asignar una etiqueta a cada variable, de tal forma que cada etiqueta a_1, ..., a_i es una asignación de verdad a las variables x_1, ..., x_i. El nodo al que le corresponde esta asignación será etiquetado con $\mathrm{E}[\mathrm{W} \mid \mathrm{x}_1{=}\mathrm{a}_1, \ldots, \mathrm{x}_i{=}\mathrm{a}_i]$. Si $i{=}n$, significa que este es un nodo hoja y su expectativa condicional es el total de pesos de las cláusulas satisfacibles por su asignación de verdad.

Lema 2: *La expectativa condicional de cualquier nodo en T puede ser computado en tiempo polinomial.*

Demostración. Consideremos el nodo $\mathrm{x}_1{=}\mathrm{a}_1$, ..., $\mathrm{x}_i{=}\mathrm{a}_i$. Sea ϕ la fórmula booleana, con las variables x_{i+1}, ..., x_n, obtenidas de este nodo por la vía de auto-reducción. El peso esperado de las cláusulas satisfacibles de ϕ bajo un asignación de verdad aleatoria a las variables x_{i+1}, ..., x_n puede ser computado en tiempo polinomial. Al añadir este total de pesos de la cláusula f ya se satisface por la asignación parcial $\mathrm{x}_1{=}\mathrm{a}_1$, ..., $\mathrm{x}_i{=}\mathrm{a}_i$ dada por la respuesta. □

Teorema 1: *Se puede computar una ruta desde la raíz a una hoja de un árbol, tal que la expectativa condicional de cada nodo en esta ruta es $\geq$ E[W], en tiempo polinomial.*

Demostración. La expectativa condicional de un nodo es el promedio de las expectativas condicionales de sus dos hijos. Esto es:

$E[W \mid x_1{=}a_1, ..., x_i{=}a_i] = E[W \mid x_1{=}a_1, ..., x_i{=}a_i, x_{i+1}{=}\mathit{True}]/2 + E[W \mid x_1{=}a_1, ..., x_i{=}a_i, x_{i+1}{=}\mathit{False}]/2$.

La razón es porque x_{i+1} es igual de probable que se establezca en *True* o en *False*. Como resultado de lo anterior, el hijo con el valor mayor tiene una expectativa condicional al menos tan grande como el padre, así que esto establece la existencia del camino que se quiere encontrar. Como consecuencia del lema 2, este camino se puede encontrar en tiempo polinomial. □

El algoritmo determinista se sigue a manera de corolario del teorema 1. Lo que se hará es generar la asignación de verdad del nodo hoja de la ruta calculada. El peso total de cláusulas satisfacibles por este nodo es $\geq E[W]$.

Se mostrará a continuación que la técnica mostrada anteriormente se puede usar para quitar aleatoridad a algoritmos aleatorios más complejos. Supongamos que el algoritmo no asigna un valor booleano a las variables independientemente. Es decir,

$$E[W \mid x_1{=}a_1, ..., x_i{=}a_i] =$$
$$E[W \mid x_1{=}a_1, ..., x_i{=}a_i, x_{i+1}{=}\mathit{True}] \Pr[x_{i+1}{=}\mathit{True} \mid x_1{=}a_1, ..., x_i{=}a_i] +$$
$$E[W \mid x_1{=}a_1, ..., x_i{=}a_i, x_{i+1}{=}\mathit{False}] \Pr[x_{i+1}{=}\mathit{False} \mid x_1{=}a_1, ..., x_i{=}a_i].$$

La suma de dos probabilidades condicionales es igual a 1, en este caso, porque ambos eventos son exhaustivos, así que la expectativa condicional del padre sigue siendo una combinación convexa de las expectativas condicionales de ambos hijos. Si se puede determinar en tiempo polinomial cuál de los dos hijos es tiene un valor mayor, se puede nuevamente quitar aleatoridad al algoritmo, pero computar la expectativa condicional pude volverse muy laborioso.

En general, un algoritmo aleatorio puede elegir entre un conjunto más grande de opciones que no necesariamente toman la misma probabilidad, pero una combinación convexa de la expectativa condicional de las elecciones dadas por la probabilidad de elegirlas es igual a la expectativa condicional del padre, así que debe haber una opción que tenga al menos una expectativa condicional tan grande como la del padre.

2.3. Algoritmo para cláusulas pequeñas por LP-rounding

En esta sección se muestra un programa de enteros para el problema MAX-SAT. Para cada cláusula $C \in C$, sea S_c^{+} (S_c^{-}) denotar el conjunto de variables boleanas no negadas en c. La asignación de verdad es codificada por y. Tomamos $y_i{=}1$ ($y_i{=}0$) que denota la asignación x_i a *True* (*False*).

La restricción para la cláusula c asegura que z_c puede ser 1 sólo si al menos una de las literales que está en c se establece en *True*, es decir, si la cláusula c es satisfacible cambiando la asignación de verdad.

Maximizar:
$\sum_{c \in C} W_c z_c$
Sujeto a:
$\forall$ c $\in$ C: $\sum_{i \in S_c^+} y_i + \sum_{i \in S_c^-} (1 - y_i) \geq z_c$
$\forall$ c $\in$ C: $z_c \in 0, 1$
$\forall$ i : $y_i \in 0, 1$

La relación LP es la siguiente
Maximizar:
$\sum_{c \in C} W_c z_c$
Sujeto a:
$\forall$ c $\in$ C: $\sum_{i \in S_c^+} y_i + \sum_{i \in S_c^-} (1 - y_i) \geq z_c$
$\forall$ c $\in$ C: $1 \geq z_c \geq 0$
$\forall$ i : $1 \geq y_i \geq 0$

Sea (y*, z*) lo que denota la solución óptima. Independientemente se define x_i con *True* con una probabilidad de y_i*, para $1 \leq i \leq n$. A partir de lo anterior se genera una asignación de verdad resultante llamada ρ.

Se usan las variables aleatorias W y W_c definidas anteriormente. Además, para $k \geq 1$, se define:

$\beta_k = 1 - \left(1 - \frac{1}{k}\right)^k$

Lema 3: *Si size(c)=k, entonces E[W_c] $\leq \beta_k \; w_c \; z_c$**

Demostración. Podemos asumir, sin pérdida de generalidad, que todas las literales en c aparecen no negadas. En caso de que x_i aparezca negada, podemos reemplazar x_i con $\overline{x}_i$ a lo largo de la fórmula f y además, modificar *LP* como consecuencia, pero sin modificar o afectar z_c* o W_i. Finalmente, al cambiar el nombre de la variables, podemos asumir que c=($x_1 \vee ... \vee x_k$).

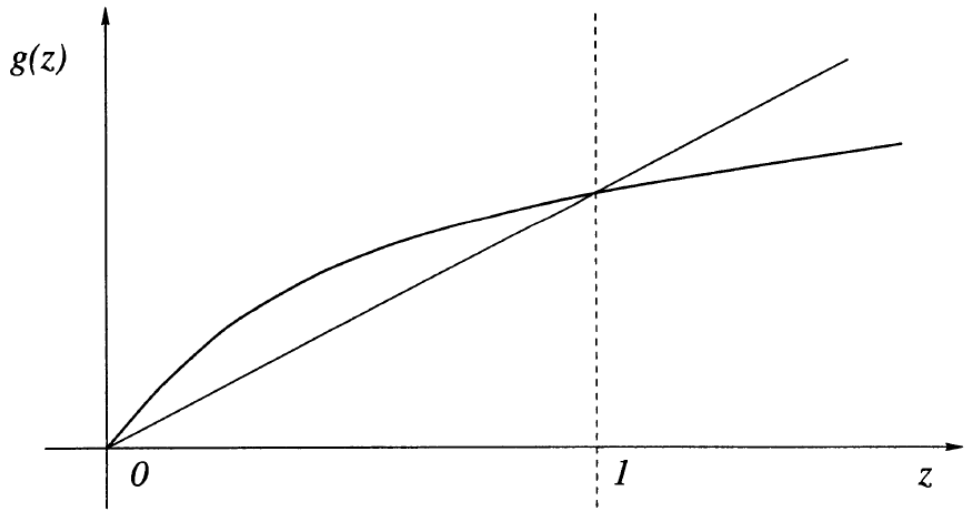
Decimos que la cláusula c es satisfacible si x_1, ..., x_k no todas las variables tienen valor *False.* La probabilidad de que este evento ocurra está dado por:

$$1 - \prod_{i=1}^{k} (1 - y_i) \geq 1 - \left(\frac{\sum_{i=1}^{k} (1 - y_i)}{k}\right)^k = 1 - \left(1 - \frac{\sum_{i=1}^{k} y_i}{k}\right)^k \geq 1 - \left(1 - \frac{z_c*}{k}\right)^k$$

donde la primera desigualdad se sigue de la desigualdad media aritmético-geométrica que establece que para números no negativos a_1, ..., a_k

$\frac{a_1+...+a_k}{k} \geq \sqrt[k]{a_1 \times ... \times a_k}$.

La segunda desigualdad se obtiene de usar la restricción en LP que $y_1 + ... + y_k \leq z_c$.



Se define la función g por:

$g(z) = 1 - \left(1 - \frac{z}{k}\right)^k$

La función anterior es cóncava con g(0)=0 y g(1)=β_k. Por lo tanto, para z ∈ [0,1], g(z)≥ β_kz. Por lo que **Pr**[c es satisfacible]≥ $\beta_k z_c$*. Así que el lema se cumple. □

Notemos que β_k es una función decreciente de k. Por lo que, si todas las cláusulas tienen un tamaño máximo de k, entonces tenemos que:

$E[W] = \sum_{c \in C} E[W_c] \geq \beta_k \sum_{c \in C} w_c z_c* = \beta_k OPT_f \geq \beta_k OPT$

donde OPT_f es el valor de la solución óptima de LP. Tenemos que $OPT_f \geq OPT$. Este algoritmo puede quitar aleatoridad usando el método de la expectativa condicional. Por lo tanto, para las instancias del problema

MAX-SAT con cláusulas de tamaño al menos k, hay un algoritmo de apoximación de factores β_k, ya que

$\forall k \in \mathbb{Z}^+ : \left(1 - \frac{1}{k}\right)^k < \frac{1}{e}$

este es un algoritmo de factor 1- 1/e para MAX-SAT.

2.4. Algoritmo con factor de 3/4

Para que este algoritmo funcione se hace una combinación de los dos anteriores. Sea b el lanzamiento de una moneda justa, si b=0, lo que se hará es correr el primer algoritmo aleatorio, pero si b=1, entonces se correrá el segundo algoritmo aleatorio.

Como se están usando los algoritmos aleatorios, se está modificando la variable x_i a *true* con una probabilidad de $\frac{1}{4}+\frac{1}{2}y_i$*. Observemos que x_i's no se les da un valor de forma independiente. Sea (y*, z*) una solución óptima de LP sobre la instancia dada del algoritmo.

Lema 4: $\mathrm{E}[\mathrm{W}_c] \geq \frac{3}{4}\mathrm{w}_c\mathrm{z}_c$*.

Demostración. Sea *size(c)=k*. Por el lema 1, tenemos que

$\mathrm{E}[\mathrm{W}_c \mid b = 0] = \alpha_k \mathrm{w}_c \geq \alpha_k \mathrm{w}_c \mathrm{z}_c$*.

Para lo anterior se uso el hecho de que z_c*≤ 1 y por el lema 4 tenemos que

$\mathrm{E}[\mathrm{W}_c \mid \mathrm{b}=1] \geq \beta_k \mathrm{w}_c \mathrm{z}_c$*.

Combinando lo anterior, obtenemos lo siguiente:

$E[W_c] = \frac{1}{2}(E[W_c \mid b = 0] + E[W_c \mid b = 1]) \geq w_c z_c * \frac{(\alpha_k + \beta_k)}{2}$

Ahora, tenemos que $\alpha_1 + \beta_1 = \alpha_2 + \beta_2 = 3/2$, y para k ≥ 3, $\alpha_k + \beta_k \geq$ 7/8 + (1-1/e) $\geq 3/2$. Por esto, se cumple el lema. □

Por linealidad de expectativa tenemos que

$E[W] = \sum_{c \in C} E[W_c] \geq \frac{3}{4} \sum_{c \in C} w_c z_c* = \frac{3}{4} OPT_f \geq \frac{3}{4} OPT$

Donde OPT_f es la solución óptima a LP. Finalmente, consideramos el siguiente algoritmo determinista:

Algoritmo MAX-SAT- factor 3/4

1. Utilice el algoritmo de factor 1/2 para quitar aleatoriedad para obtener una asignación de verdad, que llamaremos ρ_1.
2. Utilice el algoritmo de factor para quitar aleatoriedad 1-1/e para obtener una asignación de verdad que llamaremos ρ_2.
3. Generar la mejor de las dos soluciones.

Teorema 2: *El algoritmo MAX-SAT- factor 3/4 es un algoritmo de aproximación de factor 3/4 determinista para MAX-SAT.*

Demostración. Por el lema 4, tenemos que el promedio de los pesos de las cláusulas satisfechas bajo ρ_1 y ρ_2 es mayor o igual a $\frac{3}{4}$OPT. Por lo tanto, la mejor de estas dos asignaciones también lo es al menos tan bien como la otra asignación. □

3. Ejemplos

Ejemplo 1:

Consideremos la fórmula SAT $f = (x_1 \vee x_2) \wedge (\overline{x_1} \vee x_2) \wedge (x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee \overline{x_2})$ donde cada cláusula tiene un peso unitario. Podemos notar que la asignación y_i=1/2 y z_c=1 para todo i y c es una solución óptima de LP para cualquier ejemplar teniendo cláusulas de longitud 2. Por lo tanto, OPT_f=4. Por otro lado, OPT = 3, y por lo tanto, para este caso, LP tiene un espacio de garantía de 3/4.

Ejemplo 2:

Para ejemplificar el algoritmo de aproximación propuesto, definimos una fórmula $f = (x \vee y) \wedge (x \vee \overline{y}) \wedge (\overline{x} \vee z)$. Además, cada cláusula debe contar con un peso. Los pesos de las cláusulas serán 1, 1 y 2+ϵ, respectivamente. Por la observación hecha en el ejemplo 1, en este caso, el algoritmo de factor 1-1/e establecerá cada una de las variables en *True* con probabilidad 1/2, así que será el mismo que el algoritmo con garantía de 1/2.

Durante el proceso de quitar aleatoriedad, supongamos que la variable x se define primero. Las expectativas condicionales resultantes de esto son:

- $E[W \mid x = True] = 3 + \epsilon/2$
- $E[W \mid x = False] = 3 + \epsilon$

Por lo tanto, x se definirá en *False*. Con esto, se obtiene un peso total de 3+ϵ, mientras que al darle el valor a x de *True*, podemos obtener un peso de 4+ϵ. Con esto se pude obtener una familia infinita de posibles soluciones aproximadas al replicar las 3 cláusulas pero con nuevas variables.

4. Referencias

- Vi jay V. Vazirani, *Approximation Algorithms*, Springer, 2003.