INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

ANÁLISIS DE ALGORITMOS

PRUEBAS DE NP-COMPLETOS

EQUIPO 08

INTEGRANTES:

- GARCÍA MIGUEL LUIS CHRISTIAN
- HERNÁNDEZ CASTELLANOS CÉSAR URIEL.
- AGUILAR GARCÍA MAURICIO

DOCENTE:

- DR. BENJAMÍN LUNA BENOSO

22 DE NOVIEMBRE DEL 2018

Contenido

- El problema de la satisfactibilidad proposicional.
- Cláusulas.
- Un problema SAT.
- Importancia del problema SAT.
- Un lema base para mostrar que L es NP-completo.
- Pasos para probar que L es NP-completo.
- 3-CNF

Problema SAT

El problema de la satisfactibilidad proposicional es el problema de determinar, dada una fórmula de cálculo proposicional en forma FNC, sí existe una asignación de valores de verdad por los cuales la fórmula sea verdadera.

Ejemplo:

$$(x_1 \vee \bar{x}_2) \wedge (\bar{x}_2 \vee \bar{x}_3) \wedge (x_1 \vee x_3)$$

Problema SAT

Comenzamos con una lista de variables booleanas x1, …, xn. Un literal es una de las variables xi (o la negación de una de las variables ¬xi). Hay 2n literales posibles. Una cláusula es un conjunto de literales.

Las reglas del juego son las siguientes: Asignamos valores booleanos Verdadero (V) o Falso (F) a cada una de las variables. De este modo a cada uno de los literales se le asigna un valor booleano

Problema SAT

Las literales Xj aparecen con valor verdadero o falso.

Ejemplo de un problema SAT

$$((\sim p) \vee r \vee (\sim t)) \wedge (p \vee (\sim r) \vee s)$$

Las cláusulas Fi pueden tener diferente número de literales dependiendo del problema SAT de que se trate.

Para distinguir entre el número de literales en la instancia, se usa a veces la siguiente denominación:

Cláusulas

2 -Cláusula: Cláusula con dos literales.

3 -Cláusula: Cláusula con tres literales.

k -Cláusula: Cláusula con k literales.

De esta forma, se tienen instancias de problemas SAT denominados como sigue:

2 -SAT: Sólo se permiten cláusulas con dos literales.

3 -SAT: Sólo se permiten cláusulas con tres literales.

k -SAT: Sólo se permiten cláusulas con k literales.

3 -SAT: Es la instancia del problema k-SAT el cual es NP-Completo.

Prueba

Sea una instancia SAT dada. Mostraremos cómo transformar una instancia 3-SAT que sea satisfactible si y sólo si el problema SAT original fue satisfactible.
Vamos a reemplazar cláusulas que contienen más de tres literales con colecciones de cláusulas que contienen exactamente tres literales y que tienen la misma satisfactibilidad que el original.

Prueba

Suponga que nuestra instancia SAT contiene la cláusula

{ x1, x2, …, xk } { k >= 4 }

Entonces esta cláusula será reemplazada por k-2 nuevas cláusulas, utilizando k-3 nuevas variables, zj(j = 1,..., k — 3). Las k-2 nuevas cláusulas son:

{x1, x2, z1}, {x3, !z1, z2}, {x4, !z2, z3}, …, {x k-1, xk, !zk-3}

Prueba

Si x1*, …, xk* es una asignación de valores de verdad para las x*s por la cual la cláusula {x1, x2, …, xk} {k>=4} es verdadera, entonces existen asignaciones z1* …, zk-3* de valores de verdad para las z*s tal que todas las cláusulas {x1,x2,z1}, {x3, !z1,z2}, {x4,!z2, z3},..., {xk-1,xk, !zk-3} son simultáneamente satisfechas por (x*,z*). Contrariamente, si (x*,z*) es alguna asignación que satisface todas las cláusulas {x1,x2,z1}, {x3,!z1,z2}, {x4, !z2,z3},..., {xk-1,xk,!zk-3}, entonces x* sola satisface {x1,x2, ...,xk}(k > 4).

Para probar esto, primero suponga que {x1,x2, ...,xk}(k > 4) es satisfecha por alguna asignación x*.

Prueba

Entonces una, al menos, de las k literales x1,..,xk, digamos xr, tiene el valor 'verdadero'. Entonces podemos satisfacer todas las k — 2 de las cláusulas transformadas {x1,x2,z1},{x3,!z1,z2},{x4,!z2,z3},...,{xk-1,xk, !zk-3} asignando zs* := 'verdadero' para s <= r — 2 y zs* ='falso' para s > r — 2.

Ejemplo

Consideramos el conjunto de variables x1, x2, x3. Podemos construir la siguiente lista de cláusulas.

$$\{x_1, \neg x_2\}\ \{x_1, x_3\}\ \{x_2, \neg x_3\}\ \{\neg x_1, x_3\}$$

Si elegimos los valores (V,V,F) para las variables (x1, x2, 3) respectivamente, entonces los valores de las cuatro cláusulas será (V,V,V,F), así que no podría ser una asignación válida para satisfacer el conjunto de cláusulas.

Al final obtenemos como asignación satisfactoria a (T,T,T).

Importancia del problema SAT

El problema SAT es el primer problema NP-Completo, y todos los problemas del conjunto de NP-Completos pueden ser transformados al problema SAT.

Importancia del problema SAT

Al demostrar que estos problemas son NP-completos lo que se pretende es ofrecer una gama suficiente de problemas NP-completos, que puedan ser utilizados para demostrar que otros también lo son, por ello la importancia del problema SAT

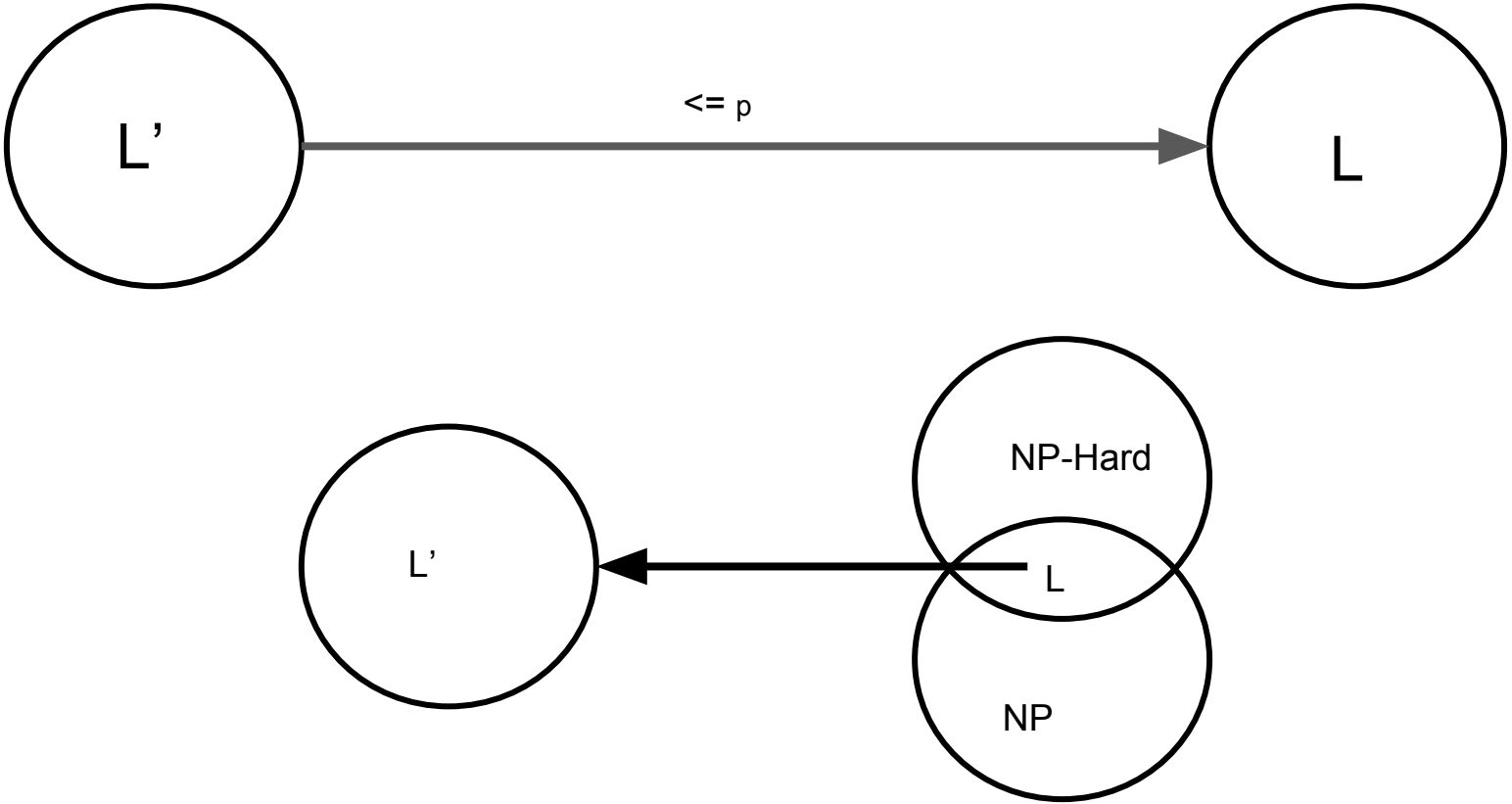
Lema

Si L es un lenguaje tal que $L' \leq_P$ L para algunos $L' \in \text{NPC}$, entonces L es NP-hard. Y sí además $L \in \text{NP}$, entonces $L \in \text{NPC}$.

Prueba

Ya que L' es NP-Completo, para todos $L'' \in \text{NP}$, tenemos $L'' \leq_P L'$. , por suposición $L' \leq_P L$ y por lo tanto por transitividad, tenemos $L'' \leq_P L$.

Lo cual muestra que L es NP-hard, y si $L \in \text{NP}$, también tenemos que $L \in \text{NPC}$.


L’
<= p
L
NP-Hard
L’
L
NP

En otras palabras, al reducir un lenguaje conocido NP-completo L’ a L, implícitamente se reduce cada lenguaje de NP a L.

Por lo tanto, el Lema nos da un método para probar que una lenguaje L es NP-completo

Esta metodología de reducción de un solo lenguaje NP conocido es mucho más simple que mostrar directamente cada lenguaje en NP

El problema SAT

Una vez que se ha probado que un problema π es NP-Completo, el procedimiento para probar problemas adicionales NP-Completos se simplifica.

Dado un problema π que pertenece a NP, todo lo que necesitamos hacer es mostrar que un problema NP-Completo π', puede ser transformado a π. Por lo tanto, el proceso de prueba de NP-Completez para un problema π consistirá de los siguientes pasos:

Pasos para probar NP-Completez

1. Mostrar que L está en NP.
2. Seleccionar un problema NP-Completo L’ ya conocido.
3. Construir una transformación f de L' a L.
4. Probar que f es una transformación polinomial.

3-CNF

Podemos probar muchos problemas NP-completos al reducir la satisfacción de la fórmula.Sin embargo, el algoritmo de reducción debe manejar cualquier fórmula de entrada, y esto requerimiento puede conducir a una gran cantidad de casos que debemos considerar. A menudo se prefiere para reducir de un lenguaje restringido de fórmulas booleanas, por lo que necesitamos considerar menos casos.

Por supuesto, no debemos restringir tanto el lenguaje que se convierte en tiempo de solución polinomial. Un lenguaje conveniente es 3-CNF satisfiability, o 3-CNF-SAT.

Se define la 3-CNF satisfacibilidad usando los siguientes términos:

Una literal booleana.

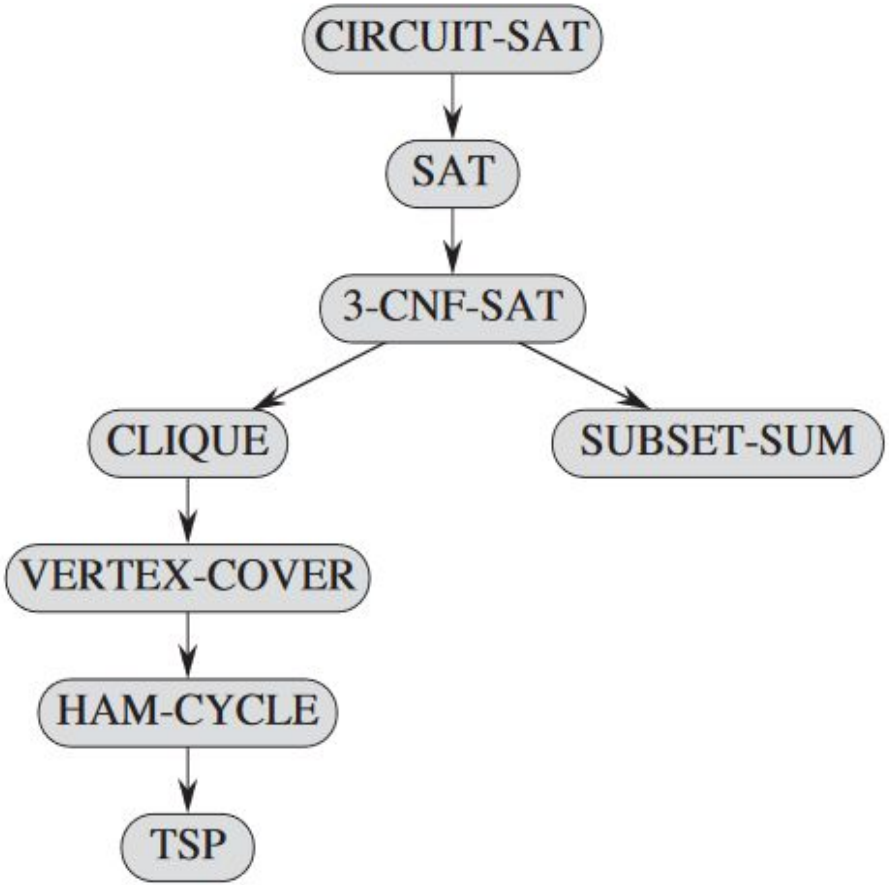
Una fórmula booleana está en CNF, la cual está expresada como un AND de cláusulas, cada una con OR en una o más literales.

Una formula boleana en 3-CNF es cuando tienen exactamente tres distintas literales.

$$(x_1 \vee \neg x_1 \vee \neg x_2) \wedge (x_3 \vee x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_3 \vee \neg x_4)$$

Teorema

Si la fórmula en su tercera forma conjuntiva normal es satisfacible es un NP-Completo


CIRCUIT-SAT
SAT
3-CNF-SAT
CLIQUE
SUBSET-SUM
VERTEX-COVER
HAM-CYCLE
TSP

Referencias

Cormen, T. and Cormen, T. (2001). *Introduction to algorithms*. 3rd ed. Cambridge, Mass.: MIT Press.

Pretolani. Gallo G. A new algorithm for the propositional satisfiability problem.Departament di informática,University of

Pisa,Corso, Italia. 1992. S. Horie and O.Watanabe. Hard instance generation for SAT.Technical ReportT97 TR-0007,Dept.of Computer Science, Tokyo Inst.of Tech,1997.