Análisis de Algoritmos II

Jesús Alexander Aranda Ph.D Robinson Duque, Ph.D
Juan Francisco Díaz, Ph. D

Universidad del Valle

jesus.aranda@correounivalle.edu.co
robinson.duque@correounivalle.edu.co
juanfco.diaz@correounivalle.edu.co

Programa de Ingeniería de Sistemas
Escuela de Ingeniería de Sistemas y Computación

Overview

SAT: Primer Problema NP-Completo

Problema de Satisfactibilidad Booleana (SAT)

Para probar que un problema es NP-Completo es necesario un primer problema NP-Completo. Para esto, utilizaremos el problema de satisfactibilidad (SAT) para el cual se ha demostrado que es NP-Completo (Stephen Cook, 1971).

Esto quiere decir que cualquier problema NPC puede ser reducido a SAT en tiempo polinomial.

SAT: Primer Problema NP-Completo

Problema de Satisfactibilidad Booleana (SAT)

Para probar que un problema es NP-Completo es necesario un primer problema NP-Completo. Para esto, utilizaremos el problema de satisfactibilidad (SAT) para el cual se ha demostrado que es NP-Completo (Stephen Cook, 1971).

Esto quiere decir que cualquier problema NPC puede ser reducido a SAT en tiempo polinomial.

SAT: Primer Problema NP-Completo

Problema de Satisfactibilidad Booleana (SAT)

El problema consiste en un conjunto V de n variables booleanas $v_1, v_2, v_3 ... v_n$ y un conjunto C de m clausulas $c_1, c_2, c_3 ... c_m$ en forma normal conjuntiva (FNC).

Ejemplo de instancia SAT

$C = (x \vee \neg y \vee z \vee w) \wedge (\neg y \vee z) \wedge (\neg x \vee \neg y \vee \neg z) \wedge (z \vee w) \wedge y$
¿Existe una asignación de valores para (x, y, z, w), donde C sea verdadera?

$C = (x \vee y) \wedge (x \vee \neg y) \wedge \neg x$
¿Existe una asignación de valores para (x, y), donde C sea verdadera?

SAT: Primer Problema NP-Completo

(Ser NP) SAT es NP

- Si la entrada es satisfactible, es porque existe al menos una asignación de valor a las variables que hacen que la fórmula sea cierta. Dada esa asignación, como certificado, es muy fácil (polinomial) verificar (reemplazando en cada cláusula) si cada cláusula evalúa a verdadero.
- Si la entrada es insatisfactible, ninguna asignación de variables que me entreguen servirá para verificar que es una instancia positiva. Es decir, no se puede construir un certificado con el que me puedan engañar.
- Se concluye entonces que $SAT \in NP$ (Nótese que hay que argumentar para las instancias positivas y para las negativas)

SAT: Primer Problema NP-Completo

(Ser NP) SAT es NP

- Si la entrada es satisfactible, es porque existe al menos una asignación de valor a las variables que hacen que la fórmula sea cierta. Dada esa asignación, como certificado, es muy fácil (polinomial) verificar (reemplazando en cada cláusula) si cada cláusula evalúa a verdadero.
- Si la entrada es insatisfactible, ninguna asignación de variables que me entreguen servirá para verificar que es una instancia positiva. Es decir, no se puede construir un certificado con el que me puedan engañar.
- Se concluye entonces que $SAT \in NP$ (Nótese que hay que argumentar para las instancias positivas y para las negativas)

SAT: Primer Problema NP-Completo

(Ser NP) SAT es NP

- Si la entrada es satisfactible, es porque existe al menos una asignación de valor a las variables que hacen que la fórmula sea cierta. Dada esa asignación, como certificado, es muy fácil (polinomial) verificar (reemplazando en cada cláusula) si cada cláusula evalúa a verdadero.
- Si la entrada es insatisfactible, ninguna asignación de variables que me entreguen servirá para verificar que es una instancia positiva. Es decir, no se puede construir un certificado con el que me puedan engañar.
- Se concluye entonces que $SAT \in NP$ (Nótese que hay que argumentar para las instancias positivas y para las negativas)

SAT: Primer Problema NP-Completo

(Ser NP duro) SAT es NP-duro (intuición)

- Dada una máquina de Turing no determinista cualquiera, que resuelva un problema NPC, y una entrada, se produce una fórmula en FNC de SAT tal que (1) si la máquina aceptaba la entrada, la fórmula es satisfactible, y (2) si la máquina no aceptaba la entrada, la fórmula es insatisfactible.
- El número de variables necesitadas para codificar la máquina en una fórmula es polinomial con respecto al tamaño de la máquina.
- El número de cláusulas de la fórmula es polinomial en el tamaño de la máquina.

SAT: Primer Problema NP-Completo

(Ser NP duro) SAT es NP-duro (intuición)

- Dada una máquina de Turing no determinista cualquiera, que resuelva un problema NPC, y una entrada, se produce una fórmula en FNC de SAT tal que (1) si la máquina aceptaba la entrada, la fórmula es satisfactible, y (2) si la máquina no aceptaba la entrada, la fórmula es insatisfactible.
- El número de variables necesitadas para codificar la máquina en una fórmula es polinomial con respecto al tamaño de la máquina.
- El número de cláusulas de la fórmula es polinomial en el tamaño de la máquina.

SAT: Primer Problema NP-Completo

(Ser NP duro) SAT es NP-duro (intuición)

- Dada una máquina de Turing no determinista cualquiera, que resuelva un problema NPC, y una entrada, se produce una fórmula en FNC de SAT tal que (1) si la máquina aceptaba la entrada, la fórmula es satisfactible, y (2) si la máquina no aceptaba la entrada, la fórmula es insatisfactible.
- El número de variables necesitadas para codificar la máquina en una fórmula es polinomial con respecto al tamaño de la máquina.
- El número de cláusulas de la fórmula es polinomial en el tamaño de la máquina.

Esquema General de Demostración de NP-Completitud

Lema de NP-Completitud

Lema: Si B es un problema tal que $A \prec_p B$ para algún $A \in NPC$, entonces B es NP-Hard. Si además $B \in NP$ entonces $B \in NPC$.

De esta manera, para demostrar que un problema B es NP-Completo, se debe realizar lo siguiente:

1. (Ser NP) Probar que $B \in NP$
2. (Ser NP-duro) Probar que $B \in NP - Hard$
   1. Seleccionar un problema NP-Completo A conocido
   2. Describir un algoritmo que transforme cada instancia de A a una instancia de B
   3. Probar que el algoritmo anterior corre en tiempo polinomial
   4. Probar que el algoritmo es correcto

Demostración de NP-Completitud de 3SAT

Definición 3SAT

Una instancia de 3-SAT es una colección de C cláusulas en forma normal conjuntiva (FNC) donde cada cláusula contiene exactamente 3 literales (i.e., variables booleanas $v_i \in V$ o su negación).

$V = (x, y, z, w)$
$C = (x \vee \neg y \vee z) \wedge (\neg x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg w) \wedge (y \vee z \vee w)$
¿Existe una asignación de valores para (x, y, z, w), donde la instancia sea verdadera?

Demostración de NP-Completitud de 3SAT

Para probar que 3SAT es NP-Completo se necesita:

1. (Ser NP) Probar que $3SAT \in NP$
2. (Ser NP-duro) Probar que $3SAT \in NP - Hard$
   1. Seleccionar un problema NP-Completo A conocido
   2. Describir un algoritmo que transforme cada instancia de A a una instancia de 3SAT. Esto es $A \prec_p 3SAT$.
   3. Probar que el algoritmo anterior corre en tiempo polinomial
   4. Probar que el algoritmo es correcto

Demostración de NP-Completitud de 3SAT

Para probar que 3SAT es NP-Completo se necesita:

1. (Ser NP) Probar que $3SAT \in NP$
2. (Ser NP-duro) Probar que $3SAT \in NP - Hard$
   1. Seleccionar un problema NP-Completo A conocido
   2. Describir un algoritmo que transforme cada instancia de A a una instancia de 3SAT. Esto es $A \prec_p 3SAT$.
   3. Probar que el algoritmo anterior corre en tiempo polinomial
   4. Probar que el algoritmo es correcto

¿Está 3SAT en NP?

Recordemos: la clase NP es aquella cuyos problemas son verificables en tiempo polinomial. Si se tuviera alguna clase de certificado de una solución, entonces, es posible verificar en tiempo polinomial que el certificado es correcto.

Para este caso el certificado es la asignación de variables que hace satisfactible la fórmula.

3SAT $\in$ NP

Dada una instancia positiva de 3SAT y el certificado, sólo se debe chequear cada cláusula para evaluar que se satisfaga. Evidentemente la verificación se puede hacer en tiempo polinomial $3 * |C|$ equivalente a $O(|C|)$.

Dada una instancia negativa de 3SAT, ningún certificado puede hacer que el algoritmo verifique la instancia.

Por tanto $3 - SAT \in NP$

¿Está 3SAT en NP?

Recordemos: la clase NP es aquella cuyos problemas son verificables en tiempo polinomial. Si se tuviera alguna clase de certificado de una solución, entonces, es posible verificar en tiempo polinomial que el certificado es correcto.
Para este caso el certificado es la asignación de variables que hace satisfactible la fórmula.

3SAT $\in$ NP

Dada una instancia positiva de 3SAT y el certificado, sólo se debe chequear cada cláusula para evaluar que se satisfaga. Evidentemente la verificación se puede hacer en tiempo polinomial $3 * |C|$ equivalente a $O(|C|)$.
Dada una instancia negativa de 3SAT, ningún certificado puede hacer que el algoritmo verifique la instancia.
Por tanto $3 - SAT \in NP$

¿Está 3SAT en NP?

Recordemos: la clase NP es aquella cuyos problemas son verificables en tiempo polinomial. Si se tuviera alguna clase de certificado de una solución, entonces, es posible verificar en tiempo polinomial que el certificado es correcto.
Para este caso el certificado es la asignación de variables que hace satisfactible la fórmula.

3SAT $\in$ NP

Dada una instancia positiva de 3SAT y el certificado, sólo se debe chequear cada cláusula para evaluar que se satisfaga. Evidentemente la verificación se puede hacer en tiempo polinomial $3 * |C|$ equivalente a $O(|C|)$.
Dada una instancia negativa de 3SAT, ningún certificado puede hacer que el algoritmo verifique la instancia.
Por tanto $3 - SAT \in NP$

¿Está 3SAT en NP?

Recordemos: la clase NP es aquella cuyos problemas son verificables en tiempo polinomial. Si se tuviera alguna clase de certificado de una solución, entonces, es posible verificar en tiempo polinomial que el certificado es correcto.
Para este caso el certificado es la asignación de variables que hace satisfactible la fórmula.

3SAT $\in$ NP

Dada una instancia positiva de 3SAT y el certificado, sólo se debe chequear cada cláusula para evaluar que se satisfaga. Evidentemente la verificación se puede hacer en tiempo polinomial $3 * |C|$ equivalente a $O(|C|)$.
Dada una instancia negativa de 3SAT, ningún certificado puede hacer que el algoritmo verifique la instancia.
Por tanto $3 - SAT \in NP$

¿Es 3SAT NP-Hard? Escogiendo un problema para reducir

Se procede a realizar una reducción desde SAT. Nótese que 3SAT es un problema más restringido que SAT, y se debe describir un algoritmo que transforme cada instancia de SAT a 3SAT en tiempo polinomial:

$$SAT \prec_p 3SAT$$

¿Es 3SAT NP-Hard? Definiendo la reducción

idea: Transformar cada cláusula de una instancia de SAT en un conjunto de cláusulas de $3-SAT$ lógicamente equivalentes

Se transformará cada cláusula de forma independiente basado en su longitud. Suponga que una cláusula C_i de SAT contiene k literales:

- Si $k = 1$, significa que $C_i = z_1$. Se crean dos variables v_1, v_2 y cuatro nuevas cláusulas con 3 literales cada una:

$$(v_1 \vee v_2 \vee z_1) \wedge (v_1 \vee \neg v_2 \vee z_1) \wedge (\neg v_1 \vee v_2 \vee z_1) \wedge (\neg v_1 \vee \neg v_2 \vee z_1)$$

Observe que esta nueva expresión con 4 cláusulas se satisface ssi z_1 es verdadera.

¿Es 3SAT NP-Hard? Definiendo la reducción

idea: Transformar cada cláusula de una instancia de SAT en un conjunto de cláusulas de $3-SAT$ lógicamente equivalentes
Se transformará cada cláusula de forma independiente basado en su longitud. Suponga que una cláusula C_i de SAT contiene k literales:

- Si $k = 1$, significa que $C_i = z_1$. Se crean dos variables v_1, v_2 y cuatro nuevas cláusulas con 3 literales cada una:

$$(v_1 \vee v_2 \vee z_1) \wedge (v_1 \vee \neg v_2 \vee z_1) \wedge (\neg v_1 \vee v_2 \vee z_1) \wedge (\neg v_1 \vee \neg v_2 \vee z_1)$$

Observe que esta nueva expresión con 4 cláusulas se satisface ssi z_1 es verdadera.

¿Es 3SAT NP-Hard? Definiendo la reducción

- Si $k = 2$, significa $(z_1 \vee z_2)$. Se crea una nueva variable v_1 y dos nuevas cláusulas con 3 literales cada una:

$$(v_1 \vee z_1 \vee z_2) \wedge (\neg v_1 \vee z_1 \vee z_2)$$

Observe que independientemente del valor de verdad que tome la nueva variable v_1, la nueva expresión con 2 cláusulas se satisface ssi z_1 o z_2 son verdaderas.

¿Es 3SAT NP-Hard? Definiendo la reducción

- Si $k = 3$, significa $(z_1 \vee z_2 \vee z_3)$. La cláusula ya se encuentra en 3SAT y se utiliza como está.
- Si $k > 3$, , significa $(z_1 \vee z_2 ... \vee z_k)$. Cree $k - 3$ variables $v_1, v_2 ... v_{k-3}$ y agregue $k - 2$ cláusulas, siguiendo un encadenamiento de la forma:

$$(z_1 \vee z_2 \vee v_1) \wedge (\neg v_1 \vee z_3 \vee v_2) \wedge (\neg v_2 \vee z_4 \vee v_3) \wedge \ldots$$

$$(\neg v_i \vee z_{i+2} \vee v_{i+1}) \wedge \ldots (\neg v_{k-4} \vee z_{k-2} \vee v_{k-3}) \wedge (\neg v_{k-3} \vee z_{k-1} \vee z_k)$$

¿Es 3SAT NP-Hard? Corrección de la reducción

[Instancias positivas] Observe que si la cláusula es satisfactible (algún literal es verdadero, en rojo), entonces todas las cláusulas nuevas son satisfactibles:

$$(z_1 \vee z_2 \vee v_1) \wedge (\neg v_1 \vee z_3 \vee v_2) \wedge (\neg v_2 \vee z_4 \vee v_3) \wedge \ldots$$

$$(\neg v_i \vee z_{i+2} \vee v_{i+1}) \wedge \ldots (\neg v_{k-4} \vee z_{k-2} \vee v_{k-3}) \wedge (\neg v_{k-3} \vee z_{k-1} \vee z_k)$$

[Instancias negativas] Y que si la cláusula no es satisfactible (todos los literales son falsos), entonces no se pueden satisfacer todas las cláusulas nuevas:

$$(z_1 \vee z_2 \vee v_1) \wedge (\neg v_1 \vee z_3 \vee v_2) \wedge (\neg v_2 \vee z_4 \vee v_3) \wedge \ldots$$

$$(\neg v_i \vee z_{i+2} \vee v_{i+1}) \wedge \ldots (\neg v_{k-4} \vee z_{k-2} \vee v_{k-3}) \wedge (\neg v_{k-3} \vee z_{k-1} \vee z_k)$$

¿Es 3SAT NP-Hard? Corrección de la reducción

[Instancias positivas] Observe que si la cláusula es satisfactible (algún literal es verdadero, en rojo), entonces todas las cláusulas nuevas son satisfactibles:

$$(z_1 \vee z_2 \vee v_1) \wedge (\neg v_1 \vee z_3 \vee v_2) \wedge (\neg v_2 \vee z_4 \vee v_3) \wedge \ldots$$

$$(\neg v_i \vee z_{i+2} \vee v_{i+1}) \wedge \ldots (\neg v_{k-4} \vee z_{k-2} \vee v_{k-3}) \wedge (\neg v_{k-3} \vee z_{k-1} \vee z_k)$$

[Instancias negativas] Y que si la cláusula no es satisfactible (todos los literales son falsos), entonces no se pueden satisfacer todas las cláusulas nuevas:

$$(z_1 \vee z_2 \vee v_1) \wedge (\neg v_1 \vee z_3 \vee v_2) \wedge (\neg v_2 \vee z_4 \vee v_3) \wedge \ldots$$

$$(\neg v_i \vee z_{i+2} \vee v_{i+1}) \wedge \ldots (\neg v_{k-4} \vee z_{k-2} \vee v_{k-3}) \wedge (\neg v_{k-3} \vee z_{k-1} \vee z_k)$$

¿Es 3SAT NP-Hard? Corrección de la reducción

[Instancias positivas] Observe que si la cláusula es satisfactible (algún literal es verdadero, en rojo), entonces todas las cláusulas nuevas son satisfactibles:

$$(z_1 \vee z_2 \vee v_1) \wedge (\neg v_1 \vee z_3 \vee v_2) \wedge (\neg v_2 \vee z_4 \vee v_3) \wedge \ldots$$

$$(\neg v_i \vee z_{i+2} \vee v_{i+1}) \wedge \ldots (\neg v_{k-4} \vee z_{k-2} \vee v_{k-3}) \wedge (\neg v_{k-3} \vee z_{k-1} \vee z_k)$$

[Instancias negativas] Y que si la cláusula no es satisfactible (todos los literales son falsos), entonces no se pueden satisfacer todas las cláusulas nuevas:

$$(z_1 \vee z_2 \vee v_1) \wedge (\neg v_1 \vee z_3 \vee v_2) \wedge (\neg v_2 \vee z_4 \vee v_3) \wedge \ldots$$

$$(\neg v_i \vee z_{i+2} \vee v_{i+1}) \wedge \ldots (\neg v_{k-4} \vee z_{k-2} \vee v_{k-3}) \wedge (\neg v_{k-3} \vee z_{k-1} \vee z_k)$$

¿Es 3SAT NP-Hard? Corrección y Complejidad de la reducción

Corrección de la reducción

Utilizando las reglas mencionadas, según el número de variables en cada cláusula de la instancia de SAT, se genera una instancia de 3SAT con un conjunto de cláusulas equivalentes: instancias positivas de SAT se codifican en instancias positivas de 3SAT e instancias negativas de SAT se codifican en instancias negativas de 3SAT.

Complejidad de la reducción

Si se tienen m cláusulas y n literales en una instancia de SAT, y la cláusula más grande tiene k literales, esta reducción se realiza en tiempo $O(n + mk)$ y el tamaño de la instancia en 3SAT es $O(mk)$.

3SAT es NP-Completo

Se demostró que 3SAT es NP. También se mostró que 3SAT es NP-Hard a través de una reducción de SAT. Adicionalmente la reducción es correcta como se evidenció en cada caso y se puede realizar en tiempo polinomial, luego se concluye que:

3SAT es NP-Completo

3SAT es NP-Completo

Con pequeñas variaciones de la construcción de 3SAT, se puede demostrar que 4SAT, 5SAT,... también son NP-Completos. Sin embargo, esta construcción se rompe cuando se intenta utilizar 2-SAT. ¿Puede ver porqué?

Con clausulas formadas por dos literales, no es posible encadenarlas utilizando variables adicionales. 2SAT se puede resolver en tiempo polinomial y es a partir de 3 literales por clausula lo que hace el problema difícil. 1SAT es un problema trivial de resolver, ¿Puede ver porqué?

Queda demostrado que 3SAT es NP-Completo, ahora lo podemos usar en futuras reducciones:

$$SAT \prec_p 3SAT \prec_p X$$

3SAT es NP-Completo

Con pequeñas variaciones de la construcción de 3SAT, se puede demostrar que 4SAT, 5SAT,... también son NP-Completos. Sin embargo, esta construcción se rompe cuando se intenta utilizar 2-SAT. ¿Puede ver porqué?

Con clausulas formadas por dos literales, no es posible encadenarlas utilizando variables adicionales. 2SAT se puede resolver en tiempo polinomial y es a partir de 3 literales por clausula lo que hace el problema difícil. 1SAT es un problema trivial de resolver, ¿Puede ver porqué?

Queda demostrado que 3SAT es NP-Completo, ahora lo podemos usar en futuras reducciones:

$$SAT \prec_p 3SAT \prec_p X$$

Ejercicios para la casa

Ejercicio 1

Realice la reducción de la siguiente instancia SAT a 3SAT:

$$C = (a_1 \vee a_2 \vee \neg a_3) \wedge (a_2 \vee \neg a_3) \wedge \neg a_1$$

- Encuentre una asignación de valores donde SAT sea satisfactible y se refleje en la reducción.
- Encuentre una asignación de valores donde SAT no sea satisfactible y se refleje en la reducción.

Ejercicios para la casa

Ejercicio 2

Realice la reducción de la siguiente instancia SAT a 3SAT:

$$C = (a_1 \vee a_2 \vee \neg a_4) \wedge (a_2 \vee \neg a_3) \wedge (a_1 \vee \neg a_2 \vee a_3 \vee a_5 \vee a_6 \vee \neg a_7)$$

- Encuentre una asignación de valores donde SAT sea satisfactible y se refleje en la reducción.
- Encuentre una asignación de valores donde SAT no sea satisfactible y se refleje en la reducción.

Ejercicios para la casa

Ejercicio 3

Demuestre que:

- 4SAT está en NP
- 4SAT es NP-Hard
- 4SAT es NP-Complete
- 5SAT es NP-Complete
- Generalice que X-SAT es NP-Complete para $X \geq 3$

Fin de la Presentación

¿Preguntas?