Introducción

En la primera parte de la clase de hoy trataremos sobre los llamados SAT-solvers, que son programas basados en la lógica de proposiciones que nos permiten resolver muchos problemas prácticos. Veremos entonces algunos ejemplos de problemas que pueden ser resueltos por los SAT-solvers. Y a continuación, mostraremos el método en el que está basado el diseño de los SAT-solvers, que es el llamado método de Davis-Putnam.

El Problema SAT

El problema SAT. Es el problema de decidir si una fórmula proposicional en FNC es satisfactible.

El interés del problema SAT radica en que hay muchos problemas prácticos que se pueden representar mediante una fórmula de un lenguaje de proposiciones. Entonces, el que el problema tenga o no solución depende de si la fórmula asociada al problema es satisfactible.

Un SAT-solver es un programa que resuelve el problema SAT.

Los SAT-solvers están muy estudiados y son capaces de tratar fórmulas grandes.

Ejemplos de problemas que pueden ser resueltos por SAT-solvers

(1) El problema de distribuir ordenadores portátiles a los profesores de una escuela, de manera que profesores que tienen alguna hora de clase en común no compartan portátil.

(2) El problema de colorear un mapa con cuatro colores de manera que no haya dos países vecinos que tengan el mismo color.

(3) El problema de instalar paquetes de actualización en un ordenador satisfaciendo una serie de restricciones.

(4) El problema de formar un equipo de personas lo más pequeño posible para realizar una serie de tareas.

(5) El problema de trazar rutas de transporte o comunicaciones.

(6) El problema de asignar recursos en procesos industriales.

(7) Problemas para confeccionar horarios de hospitales, escuelas o líneas aéreas.

(8) La resolución de sudokus.

Ejemplo 1

Mostramos el siguiente problema, que puede ser resuelto mediante un SAT-solver.
Queremos colorear el mapa de un continente con cuatro colores de manera que no haya dos países vecinos que tengan el mismo color. Sea P el conjunto de países del continente. Representamos a los cuatro colores por los números 1,2,3,4. Representamos entonces el problema mediante una fórmula proposicional en forma normal conjuntiva de manera que pueda ser resuelto por un SAT-solver. Para ello, para $i \in P$ y $j \in \{1, 2, 3, 4\}$, consideramos la proposición Rij que significa que al país i se le asigna el color j.

Tenemos que formalizar lo siguiente:

Ejemplo 1

(1) A cada país se le asigna un color.

Para cada $i \in P$ ponemos la cláusula

$Ri1 \vee Ri2 \vee Ri3 \vee Ri4$.

(2) Ningún país tiene asignado más de un color.

Para todo $i \in P$ y para todo $j, j' \leq 4$ con $j \neq j'$, ponemos la cláusula

$\neg Rij \vee \neg Rij'$.

(3) Los países vecinos no comparten color.

Para todo $i, i' \in P$ tales que i, i' son países vecinos y para todo $j \leq 4$, ponemos la cláusula

$\neg Rij \vee \neg Ri'j$.

La fórmula buscada es entonces la conjunción de las cláusulas de (1), (2) y (3).

Ejemplo 2

Mostramos el siguiente problema, que puede ser resuelto mediante un SAT-solver.
Tenemos un país con n aeropuertos y queremos que en cada vuelo haya un control antidrogas en el aeropuerto de salida o en el aeropuerto de llegada. Tenemos la lista L de los vuelos existentes formada por pares (i, j) donde i es el aeropuerto de salida del vuelo y j es el aeropuerto de llegada. Además disponemos de k equipos de policía. El problema consiste en determinar los aeropuertos donde se han de situar los equipos de policía.
Se pide entonces representar el problema mediante una fórmula proposicional en forma normal conjuntiva de manera que pueda ser resuelto por un SAT-solver. Para ello, para $i \in \{1, \ldots, k\}$ y $j \in \{1, \ldots, n\}$, considerar la proposición Pij que significa que el "i-ésimo equipo de policía ha de ir al aeropuerto j".

Tenemos que formalizar lo siguiente:

Ejemplo 2

(1) Cada vuelo tiene un equipo de policía en el aeropuerto de origen o en el aeropuerto de destino.

Para cada vuelo (i, j) de la lista L ponemos la cláusula

$$P1i \vee P1j \vee P2i \vee P2j \vee \ldots \vee Pki \vee Pkj.$$

(2) Ningún equipo de policía puede estar en dos aeropuertos.

Para todo $i \leq k$ y para todo $j, j' \leq n$ con $j \neq j'$, ponemos la cláusula

$$\neg Pij \vee \neg Pij'.$$

(3) En ningún aeropuerto puede haber dos equipos de policía.

Para todo $i, i' \leq k$ con $i \neq i'$ y para todo $j \leq n$, ponemos la cláusula

$$\neg Pij \vee \neg Pi'j.$$

La fórmula buscada es entonces la conjunción de las cláusulas de (1), (2) y (3).

Resolución de un sudoku mediante un SAT-solver

Para cada fila i $(1 \leq i \leq 9)$, para cada columna j $(1 \leq j \leq 9)$ y para cada valor k $(1 \leq k \leq 9)$, consideramos el átomo $Rijk$ con el significado "en la fila i columna j del sudoku está el valor k".
Representamos entonces el problema mediante una fórmula en FNC con las siguientes cláusulas:

(1) En toda casilla del sudoku hay un valor.

Para cada $i, j \in \{1, \ldots, 9\}$, ponemos la cláusula $Rij1 \vee Rij2 \ldots \vee Rij9$.

(2) En ninguna casilla del sudoku hay más de un valor.

Para cada $i, j \in \{1, \ldots, 9\}$ y para cada $k, k' \in \{1, \ldots, 9\}$ con $k \neq k'$, ponemos la cláusula $\neg(Rijk \wedge Rijk') \equiv \neg Rijk \vee \neg Rijk'$.

Resolución de un sudoku mediante un SAT-solver

(3) En cada fila (o columna o cuadrado de 3×3) ningún valor se repite.

Consideremos el caso de las filas. Para cada $i \in \{1, \ldots, 9\}, j, j' \in \{1, \ldots 9\}$ con $j \neq j'$ y $k \in \{1, \ldots, 9\}$, ponemos la cláusula $\neg(Rijk \wedge Rij'k) \equiv \neg Rijk \vee \neg Rij'k$. Análogamente, se procede para las columnas.

Consideremos ahora el cuadrado $[1, 3] \times [1, 3]$. Para cada $i, i', j, j' \in [1, 3]$ con $(i, j) \neq (i', j')$ y para cada $k \in \{1, \ldots, 9\}$, ponemos la cláusula $\neg(Rijk \wedge Ri'j'k) \equiv \neg Rijk \vee \neg Ri'j'k$. Análogamente, procedemos para el resto de los cuadrados de 3×3.

(4) Ponemos los átomos correspondientes a los números puestos de antemano en el sudoku.

Si en la fila i columna j del sudoku tenemos el valor k puesto de antemano, ponemos el átomo $Rijk$.

Resolución de un sudoku mediante un SAT-solver

Ahora, tomamos la conjunción φ de las cláusulas consideradas en (1), (2), (3) y (4). Se tiene entonces que el sudoku tiene solución si y sólo si la fórmula φ es satisfactible. Además, si el sudoku tiene solución, la interpretación que hace cierta la fórmula φ es la solución del sudoku.

Un programa en JAVA para un SAT-solver

No es difícil escribir en Java un programa para determinar si una fórmula de un lenguaje de proposiciones es satisfactible. El programa genera mediante un bucle "for" anidado todas las posibles interpretaciones, y determina entonces si alguna de ellas satisface la fórmula. En concreto, para fórmulas de tres variables, podemos escribir el siguiente programa en JAVA, en el que ponemos como entrada la fórmula $(P_1 \vee P_2) \wedge (\neg P_1 \vee P_3)$.

Un programa en JAVA para un SAT-solver

```
class Satisfactible
{ public static void main(String [ ] args)
{ boolean b = false;
  boolean [ ] booleanos = {true, false};
  for (boolean P1 : booleanos)
  { for (boolean P2 : booleanos)
  { for (boolean P3 : booleanos)
  { boolean formula = ((P1 || P2) && ( ! P1 || P3));
    if (formula) { b = true;
                   System.out.println("la formula es satisfactible para:");
                   System.out.println("P1=" + P1);
                   System.out.println("P2=" + P2);
                   System.out.println("P3=" + P3);
                   break; }}
    if (b) break; }
    if (b) break; }
    if (!b) System.out.println("la formula es insatisfactible"); }}
```

Un programa en JAVA para un SAT-solver

El programa anterior sirve únicamente para fórmulas que tengan pocas cláusulas, ya que el número de interpretaciones crece exponencialmente con respecto al número de símbolos de proposición. Para poder escribir programas que resuelvan el problema SAT y que puedan tratar con fórmulas que tengan muchas cláusulas, hemos de utilizar el algoritmo de Davis y Putnam, que es el algoritmo en el que están basados los diseños de los SAT-solvers. Este método de Davis y Putnam para determinar si una fórmula proposicional es satisfactible es mucho más eficiente que el algoritmo consistente en generar todas las interpretaciones y ver si alguna de ellas satisface la fórmula.

El método de Davis-Putnam

Para introducir el método de Davis y Putnam, necesitamos algunas definiciones previas.

Si ψ es un literal, escribimos $\sim\psi = \neg\psi$ si ψ es un átomo, y escribimos $\sim\psi = \chi$ si $\psi = \neg\chi$ donde χ es un átomo.

Denotamos por $\square$ a la cláusula vacía. Como $\square$ es una disyunción vacía, $\square$ es una contradicción, ya que una interpretación I satisface una disyunción de fórmulas $\varphi_1 \vee \ldots \vee \varphi_n$ si y sólo si hay un $i \in \{1, \ldots, n\}$ tal que I satisface φ_i. Por tanto, como $\square$ es una disyunción vacía, no hay ninguna interpretación I que satisface $\square$.

Y denotamos por $\blacksquare$ a la conjunción vacía. Por ser una conjunción vacía, $\blacksquare$ es una tautología, ya que una interpretación I satisface una conjunción de fórmulas $\varphi_1 \wedge \ldots \wedge \varphi_n$ si y sólo si para todo $i \in \{1, \ldots, n\}$, I satisface φ_i. Por tanto, como $\blacksquare$ es una conjunción vacía, toda interpretación I satisface $\blacksquare$.

El método de Davis-Putnam

El método de Davis y Putnam consiste en añadir a la regla de resolución otras tres reglas, que preservan la satisfactibilidad de la fórmula de la entrada. Las cuatro reglas reciben entonces como entrada una fórmula en FNC y dan como salida otra fórmula en FNC. Se trata entonces de aplicar dichas reglas a la fórmula de entrada φ hasta llegar o bien a $\square$ o bien a $\blacksquare$. Si llegamos a $\blacksquare$ se tiene que φ es satisfactible, y si llegamos a $\square$ se tiene que φ es una contradicción.

Las reglas de Davis y Putnam son entonces las siguientes:

Regla de la tautología

La denotaremos por (TAU). La entrada es una fórmula φ en FNC. Y la salida es la fórmula φ^* que resulta de eliminar en φ las cláusulas que contengan un par complementario.

Se tiene que φ es satisfactible si y sólo si φ^* es satisfactible.

Por ejemplo, si $\varphi = (P \vee Q \vee \neg S) \wedge (P \vee \neg P \vee R) \wedge \neg Q$, al aplicar la regla de la tautología a φ , obtenemos la fórmula $\varphi^* = (P \vee Q \vee \neg S) \wedge \neg Q$.

Regla de la cláusula elemental

La denotaremos por $(\mathrm{CE})_\psi$ donde ψ es un literal. La entrada es una fórmula φ en FNC tal que ψ es una cláusula de φ. Y la salida es la fórmula φ^* que se calcula de la siguiente forma. En primer lugar, se construye la fórmula χ^* que resulta de eliminar las cláusulas de φ en las que aparece ψ. Si $\chi^* = \blacksquare$, entonces $\varphi^* = \blacksquare$. Si no, φ^* es la fórmula que resulta de eliminar $\sim \psi$ de las cláusulas de χ^*.

Se tiene que φ es satisfactible si y sólo si φ^* es satisfactible.

Por ejemplo, si $\varphi = P \wedge (P \vee \neg Q) \wedge (\neg P \vee \neg Q \vee R)$ y aplicamos la regla $(\mathrm{CE})_P$ a φ, obtenemos la fórmula $\varphi^* = \neg Q \vee R$.

Regla del literal puro

La denotaremos por $(\text{PU})_\psi$ donde ψ es un literal. La entrada es una fórmula φ en FNC tal que ψ es miembro de alguna cláusula de φ, pero $\sim \psi$ no es miembro de ninguna cláusula de φ. La salida es la fórmula φ^* que resulta de eliminar en φ las cláusulas en las que ψ aparece.

Se tiene que φ es satisfactible si y sólo si φ^* es satisfactible.

Por ejemplo, si $\varphi = (\neg P \vee Q) \wedge (\neg P \vee R \vee S) \wedge (\neg Q \vee T) \wedge S$ y aplicamos la regla $(\text{PU})_{\neg P}$ a φ, obtenemos la fórmula $\varphi^* = (\neg Q \vee T) \wedge S$.

Ejemplo 1

Aplicamos el método de Davis-Putnam para determinar si la fórmula $\varphi = (P \vee Q) \wedge (P \vee \neg Q) \wedge (R \vee Q) \wedge (R \vee \neg Q)$ es satisfactible.

Aplicando, en primer lugar, $(\text{PU})_P$ a φ, obtenemos la fórmula $\varphi_1 = (R \vee Q) \wedge (R \vee \neg Q)$.

Aplicando ahora $(\text{PU})_R$ a φ_1, obtenemos $\blacksquare$.

Por tanto, φ es satisfactible.

Ejemplo 2

Aplicamos el método de Davis-Putnam para determinar si la fórmula
$\varphi = (\neg P \vee \neg Q \vee \neg R) \wedge (Q \vee \neg Q \vee R) \wedge (P \vee \neg Q \vee R) \wedge (\neg P \vee Q) \wedge P \wedge R$
es satisfactible.

Aplicando, en primer lugar, la regla de la tautología a φ obtenemos la fórmula
$\varphi_1 = (\neg P \vee \neg Q \vee \neg R) \wedge (P \vee \neg Q \vee R) \wedge (\neg P \vee Q) \wedge P \wedge R.$

Aplicando ahora $(\text{CE})_P$ a φ_1 obtenemos la fórmula
$\varphi_2 = (\neg Q \vee \neg R) \wedge Q \wedge R.$

Aplicando entonces $(\text{CE})_Q$ a φ_2 obtenemos la fórmula
$\varphi_3 = \neg R \wedge R.$

Por último, aplicando $(\text{CE})_R$ a φ_3 obtenemos $\square$.

Por tanto, φ es una contradicción.

Ejemplo 3

Aplicamos el étodo de Davis-Putnam para determinar si la fórmula $\varphi = (P \vee \neg Q) \wedge (\neg P \vee Q) \wedge (R \vee Q) \wedge (\neg R \vee \neg Q)$ es satisfactible.

Se observa que únicamente se puede aplicar la regla de resolución a φ. Se obtiene entonces la fórmula $\varphi_1 = (\neg Q \vee Q) \wedge (R \vee Q) \wedge (\neg R \vee \neg Q)$.

Aplicando de nuevo la regla de resolución a φ_1, obtenemos la fórmula $\varphi_2 = (\neg Q \vee Q) \wedge (R \vee \neg R)$.

Aplicando ahora la regla de la tautología a φ_2, obtenemos la fórmula $\varphi_3 = R \vee \neg R$.

Y aplicando de nuevo la regla de la tautología a φ_3, obtenemos $\blacksquare$.

Por tanto, φ es satisfactible.

Bibliografía para el tema

- Lógica para informáticos (R. Farré)

  Capítulos 2 y 3.

- Lógica computacional (E. Paniagua)

  Capítulo 5.