Chapter 1

Full SAT

En esta sección se presentará un nuevo enfoque distinto a los anteriores, el cual se basa en definir un lenguaje al cual pertenecen todos los problemas SAT que son satisfacible y al cual se le denominará *Full SAT*.

1.1 Transformación de una fórmula booleana a una cadena

Primeramente para definir Full-SAT se debe definir una transformación de una fórmula booleana a una cadena de símbolos. Para ello se utilizará una fórmula en fornma normal conjuntiva (*CNF*), esto es extensible a cualquier fórmula booleana, ya que existe un algoritmo polinomial que transforma cualquier fórmula booleana a una fórmula en *CNF*.

Entonces dada una fórmula en *CNF*:

$$F = X_1 \wedge X_2 \wedge \ldots \wedge X_n$$

donde cada cláusula X_i es una disyunción de literales:

$$X_i = L_{i1} \vee L_{i2} \vee \ldots \vee L_{im}$$

y cada literal L_{ij} es una variable booleana o su negación. En cada cláusula X_i las variables que aparecen en F, puede tener cada una 3 estados posibles: a si la variable aparece positiva, b si la variable aparece negada y c si la variable no pertenece a ninguno de los literales de la cláusula.

Ahora dada la afirmación anterior, se puede definir una cadena de símbolos w que representa a la cláusula X_i sobre una secuencia de variables $v_1, v_2, \ldots, v_p$ de la siguiente manera:

- w cuenta con exactamente p símbolos.
- Si la variable v_j aparece positiva en X_i, entonces el j-ésimo símbolo es a.

- Si la variable v_j aparece negada en X_i, entonces el j-ésimo símbolo es b.
- Si la variable v_j no aparece en X_i, entonces el j-ésimo símbolo es c.

Si se toma la secuencia de variables correspondiente a F, y se le aplica el procedimiento anterior a cada cláusula se obtendrá una cadena de símbolos que representa a dicha cláusula en F.

Si ya se tiene una representación para cada cláusula de F solo resta obtener una cadena de símbolos que represente a F, esto se puede lograr concatenando las cadenas de símbolos de cada cláusula de F en el orden que aparecen con un separador en este caso se eligió el símbolo d.

Por ejemplo la siguiente fórmula booleana en CNF:

$$F = (x_1 \vee x_2) \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee x_3)$$

puede ser expresada como la cadena de símbolos:

$$w = aacdbaadabad$$

tomando como secuencia de variables x_1, x_2, x_3 como se describió anteriormente.

1.2 Transductor finito

Un transductor finito es un modelo computacional que extiende los autómatas finitos al incluir tanto entradas como salidas. Formalmente, un transductor finito es un autómata finito determinista o no determinista con una función de transición extendida que asocia una salida a cada transición.

Un transductor finito puede representarse como una tupla:

$$T = (Q, \Sigma, \Gamma, \delta, q_0, F),$$

donde:

- Q es el conjunto finito de estados.
- Σ es el alfabeto de entrada.
- Γ es el alfabeto de salida.
- $\delta : Q \times \Sigma \to Q \times \Gamma^*$ es la función de transición, que mapea una combinación de estado actual y símbolo de entrada a un nuevo estado y una salida.
- $q_0 \in Q$ es el estado inicial.
- $F \subseteq Q$ es el conjunto de estados finales.

1.2.1 Transductor Full-SAT

Ahora la idea para definir Full-SAT es construir un transductor finito que acepte como entrada cadenas del lenguaje $L_{0,1} = \{wd\}^*$ donde $w \in \{0,1\}^*$ y tenga como salida cadenas, donde cada cadena e representa una fórmula booleana en *CNF* y que acepte si y solo si la fórmula es satisfacible. Detrás de esta construcción se busca asociar cada carácter 0 ó 1 en la cadena de entrada al valor de la variable booleana correspondiente en la cadena de salida y verificar que para dichos valores al evaluar la fórmula booleana se obtenga un valor de verdad. Observe que mediante esta construcción se mantiene la invariante fundamental del SAT que a dos instancias de la misma variable se les asocia el mismo valor de verdad, esto es posible por como está definido el formato de la cadena de entrada.

A continuación se define el transductor finito $T_{Full-SAT}$ que sigue la construcción definida anteriormente, para ello se define el transductor T_{SAT} que hace el proceso de transducción para los valores de verdad de una cláusula w, donde $w \in 0, 1$:

$$T_{SAT} = (Q_{SAT}, \Sigma_{SAT}, \Gamma_{SAT}, \delta_{SAT}, q_{0_{SAT}}, F_{SAT}),$$

donde:

- $Q_{SAT} = q_0, q_p, q_n$.
- $\Sigma_{SAT} = 0, 1$.
- $\Gamma_{SAT} = a, b, c$.
- $\delta_{SAT} : Q \times \Sigma \to Q \times \Gamma^*$ función de transición.
- $q_{0_SAT} = q_0$ estado inicial.
- $F = q_p$ conjunto de estados finales.

se define la función de transición δ de la siguiente manera:

- transiciones para el estado q_0: representa el estado inicial.
  - $\delta_{SAT}(q_0, 1) = (q_p, a)$
  - $\delta_{SAT}(q_0, 0) = (q_p, b)$
  - $\delta_{SAT}(q_0, 0) = (q_n, a)$
  - $\delta_{SAT}(q_0, 1) = (q_0, c)$
  - $\delta_{SAT}(q_0, 1) = (q_n, b)$
  - $\delta_{SAT}(q_0, 0) = (q_0, c)$
- transiciones para el estado q_p (estado positivo de T_{SAT}): representa que para los valores de verdad de la cláusula obtiene un valor de verdad positivo.

- $\delta_{SAT}(q_p, 1) = (q_p, a)$
- $\delta_{SAT}(q_p, 0) = (q_p, a)$
- $\delta_{SAT}(q_p, 1) = (q_p, b)$
- $\delta_{SAT}(q_p, 0) = (q_p, b)$
- $\delta_{SAT}(q_p, 1) = (q_p, c)$
- $\delta_{SAT}(q_p, 0) = (q_p, c)$

- transiciones para el estado q_n (estado negativo de T_{SAT}): representa que para los valores de verdad la cláusula obtiene un valor de verdad negativo.

- $\delta_{SAT}(q_n, 1) = (q_p, a)$
- $\delta_{SAT}(q_n, 0) = (q_n, a)$
- $\delta_{SAT}(q_n, 1) = (q_n, b)$
- $\delta_{SAT}(q_n, 0) = (q_p, b)$
- $\delta_{SAT}(q_n, 1) = (q_n, c)$
- $\delta_{SAT}(q_n, 0) = (q_n, c)$

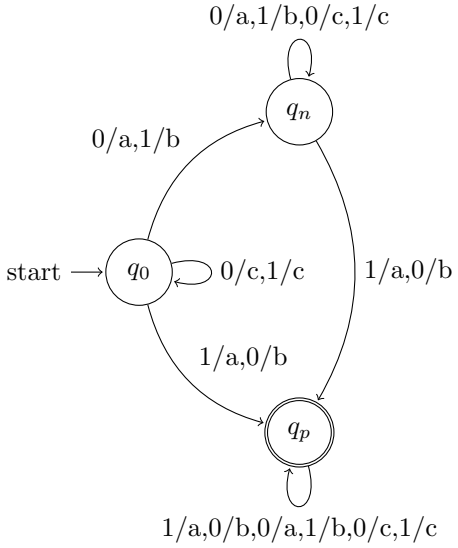

Figure 1.1: Transductor T_{SAT}.

Ahora para definir el transductor $T_{Full-SAT}$ se toman dos instancias del transductor T_{SAT} (T_1 y T_2 respectivamente) y se concatenan añadiendo una transición del estado q_{p_1} (estado positivo de T_1) al estado q_{0_2} (estado inicial de T_2) con el símbolo d (tanto de lectura como de escritura) y además se lenguaje agrega una cláusura a T_2 con una transición del estado q_{p_2} (estado positivo de T_2) al estado q_{0_2} con el símbolo d (tanto de lectura como de escritura). Entonces solo resta definir el estado inicial y el estado final de $T_{Full-SAT}$, los cuales serían q_{0_1} (estado inicial de T_1) y q_{0_2} (estado inicial de T_2), respectivamente.

1.3 Definición del lenguaje Full-SAT

Finalmente se define el lenguaje Full-SAT ($L_{Full-SAT}$) como el lenguaje de todas las cadenas e que son aceptadas por el transductor $T_{Full-SAT}$, a partir del lenguaje de cadenas de entrada $L_{0,1} = \{wd\}^*$ donde $w \in \{0,1\}^*$.

$$L_{Full-SAT} = \{e \,|\, e \in T_{Full-SAT}(w) \wedge e \in L_{0,1}\}$$

Luego $L_{Full-SAT}$ contiene todas las fórmulas booleanas satisfacibles, pero este conjunto por si solo no sirve de mucho sin un formalismo que permita conocer si una cadena que representa una fórmula booleana pertenece al lenguaje o no, para ello se necesita encontrar un formalismo que sea capaz de generar el lenguaje $L_{0,1}$ y al aplicarle el transductor $T_{Full-SAT}$ a dicho formalismo se obtenga un formalismo que cuente con un algoritmo de parsing para reconocer si una cadena pertenece a dicho formalismo o no.