Tablas Semánticas - Árboles de Refutación

➢Método alternativo a tablas de verdad para averiguar satisfacibilidad de fórmulas.

➢Su uso se basa en la estrategia de refutación.

✓deducción por refutación

$\models B$ sí y sólo sí $\neg B$ es contradicción

$A \models B$ sí y sólo sí $\{A, \neg B\}$ es insatisfacible $(A = \{A_1, A_2, ..., A_n\})$

➢<u>Tabla semántica</u>:

Secuencia de fórmulas proposicionales construidas de acuerdo a ciertas reglas, usualmente representada gráficamente como un árbol.

Árboles de Refutación

➢Dadas A, B fórmulas proposicionales:

<u>Regla 1</u>:	<u>Regla 2</u>:	<u>Regla 3</u>:	<u>Regla 4</u>:	
$\neg\neg A$	$A \wedge B$	$\neg(A \vee B)$	$\neg(A \rightarrow B)$	REGLAS de TIPO A
A	A	$\neg A$	A	
	B	$\neg B$	$\neg B$	

<u>Regla 5</u>:	<u>Regla 6</u>:	<u>Regla 7</u>:		
$A \vee B$	$\neg(A \wedge B)$	$A \rightarrow B$	premisas	REGLAS de TIPO B
A B	$\neg A$ $\neg B$	$\neg A$ B	conclusiones	

Árboles de Refutación

- Las reglas de tipo ***A*** son las que tienen una o dos conclusiones en la misma rama.

> Una valuación *v* satisface a la premisa sí y sólo sí *v* satisface a **todas** las conclusiones.

- Las reglas de tipo ***B*** son las que tienen dos conclusiones separadas en distintas ramas.

> Una valuación *v* satisface a la premisa si y solo si *v* satisface **al menos una** de las conclusiones.

Heurística para aplicar reglas:
Aplicar las reglas de tipo A antes que las de tipo B.

Árboles de Refutación

Lema:
Toda fórmula que NO es una variable proposicional o negación de una variable proposicional, es de la forma de la premisa de una (y sólo una) de las reglas.

Estas reglas proveen un método alternativo a las Tablas de Verdad para determinar satisfacibilidad de fórmulas.

Ejemplos:
Usando árboles de refutación, determinar si las siguientes fórmulas son tautologías o no.

1) $F1 = (((p \vee q) \rightarrow r) \rightarrow ((p \rightarrow r) \wedge (q \rightarrow r)))$

2) $F2 = ((p \wedge q) \rightarrow \neg(p \vee \neg q))$

Árboles de Refutación

Definiciones:

- Un **árbol de refutación** de una **fórmula F** se obtiene haciendo un número finito de **aplicaciones inmediatas de las reglas**, partiendo del **árbol** cuyo único nodo es **F**.

- Un **árbol de refutación** de un **conjunto finito C de fórmulas** es un á**rbol de refutación** de la **conjunción** de todas las **fórmulas de C.**

- Una **rama** de un árbol de fórmulas se dice **cerrada** si **contiene** a la vez una **fórmula y su negación**.
En **caso contrario**, se dice **abierta**.

- Un **árbol** se dice **cerrado** si **todas** sus **ramas** son **cerradas.**

Árboles de Refutación

-Un conjunto C de fórmulas se dice **saturado** sí y sólo sí satisface las siguientes condiciones:

i) Una fórmula y su negación no pueden estar simultáneamente en C.
ii) Si una fórmula de tipo A está en C, entonces sus dos conclusiones están en C, excepto para la regla 1 que es una sola conclusión.
iii) Si una fórmula de tipo B está en C, entonces al menos una de sus dos conclusiones están en C.

- Una rama de un árbol de fórmulas se dice saturada si el conjunto formado por las fórmulas de sus nodos es un conjunto saturado.

Observación:

Una rama saturada no puede ser cerrada.

Árboles de Refutación

- Un **árbol de refutación** se dice **completo** si cada una de sus ramas es **cerrada o saturada**.

Observación:

Todo **árbol de refutación cerrado** es un **árbol completo**.

Lema 1:

Toda fórmula F tiene por lo menos un **árbol de refutación completo**.

Lema 2:
Si F es una **fórmula satisfacible**, entonces **todo árbol de refutación de F** tiene **al menos una rama abierta**.

Lema 3:
Todo **conjunto saturado** de fórmulas es **satisfacible**.

Árboles de Refutación

Teorema:
Las siguientes propiedades son equivalentes para todo conjunto finito y no vacío $C \subseteq F_m$

i) C es satisfacible.

ii) Todo árbol de refutación de C tiene por lo menos una rama abierta.

iii) Existe un árbol de refutación completo de C con una rama abierta.

Árboles de Refutación

Corolario 1:
Las siguientes propiedades son equivalentes para todo conjunto finito y no vacío $C \subseteq F_m$

i) C es insatisfacible.

ii) C tiene un árbol de refutación cerrado.

iii) Todo árbol de refutación completo de C es cerrado.

Corolario 2:
Las siguientes propiedades son equivalentes para toda fórmula F:

i) F es tautología

ii) ¬F tiene un árbol de refutación cerrado.

iii) Todo árbol de refutación completo de ¬F es cerrado.

Árboles de Refutación

Ejercicios: (del Trabajo Práctico Nº 2)

Usando árboles de refutación, probar las siguientes tautologías:

1.a) $\models (A \rightarrow (B \rightarrow A \wedge B))$

1.b) $\models (A \rightarrow B) \rightarrow (\neg B \rightarrow \neg A)$

Usando árboles de refutación, determinar cuáles de las siguientes consecuencias semánticas son válidas:

2.a) $\{ A \vee B \} \models A \rightarrow B$

2.c) $\{A \rightarrow B, \neg C \rightarrow \neg B \} \models (A \rightarrow C)$

Sistemas Deductivos

- Naturaleza sintáctica, combinatoria
- En general axiomas + reglas de inferencia → teorema
- Demostración o prueba: secuencia finita de pasos, de aplicaciones de reglas de inferencia.
- Conexión con el concepto semántico de consecuencia
- Corrección, completitud, decidibilidad

Sistemas Deductivos

Demostración de fórmulas:

Demostración de fórmula A es una sucesión finita de fórmulas F_1, F_2, ..., F_n, todas válidas, tales que:

1) Cada fórmula F_i es un axioma o teorema ya demostrado, o una fórmula obtenida de las anteriores aplicando alguna regla de inferencia.
2) La última fórmula de la sucesión F_n es la fórmula A a demostrar.

A es entonces un teorema del sistema

Sistemas Deductivos

Demostración de deducciones:

- **Un argumento o deducción tiene la forma $P_1, P_2, ..., P_n \rightarrow Q$**
- **P_i son hipótesis y Q es la conclusión**

Demostración de la validez de un argumento es una sucesión finita de fórmulas $F_1 = P_1, F_2 = P_2, ..., F_n = P_n, F_{n+1}, F_{n+2}, F_m = Q$, todas válidas, tales que:

1) Cada fórmula F_i es una hipótesis, un axioma o teorema ya demostrado, o una fórmula obtenida de las anteriores aplicando alguna regla de inferencia.

2) La última fórmula de la sucesión $F_m = Q$ es la conclusión del argumento.

Se dice entonces que Q se deduce de $P_1, P_2, ..., P_n$

$P_1, P_2, ..., P_n \vdash Q$

Sistemas Deductivos

- Axiomático: conjunto de axiomas y conjunto de reglas de inferencia

Ejemplo:

Axiomas:

$\vdash A \rightarrow (B \rightarrow A)$

$\vdash (\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A)$

$\vdash (A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$

Regla de inferencia:

$$\frac{A \qquad A \rightarrow B}{B}$$

Sistemas Deductivos

- Deducción natural (Gentzen): sin axiomas y con varias reglas de inferencia

Reglas de inferencia básicas: Dadas $A, B, C \in F_m$

Introducción conjunción

$$\frac{A \quad B}{A \wedge B}$$

Eliminación conjunción

$$\frac{A \wedge B}{A} \qquad \frac{A \wedge B}{B}$$

Introducción disyunción

$$\frac{A}{A \vee B}$$

Eliminación disyunción

$$\frac{A \vee B \qquad \begin{matrix} A \\ \dots \\ C \end{matrix} \qquad \begin{matrix} B \\ \dots \\ C \end{matrix}}{C}$$

Sistemas Deductivos

Introducción implicación

$$\frac{\begin{matrix} A \\ \dots \\ C \end{matrix}}{A \rightarrow C}$$

Eliminación implicación (Modus Ponens)

$$\frac{A \qquad A \rightarrow C}{C}$$

Eliminación doble negación

$$\frac{\neg\neg A}{A}$$

Introducción negación

$$\frac{\begin{matrix} A \\ \dots \\ B \wedge \neg B \end{matrix}}{\neg A}$$

Sistemas Deductivos

- Resolución:

✓única regla de deducción (sin axiomas)

✓fácil implementación

✓deducción por refutación

$\models B$ sí y sólo sí $\neg B$ es contradicción

$A \models B$ sí y sólo sí $\{A, \neg B\}$ es insatisfacible $(A = \{A_1, A_2, ..., A_n\})$

✓fórmulas en forma normal conjuntiva

Formas Normales

✓ FORMA NORMAL DISYUNTIVA (FND)

Una fórmula A está en FND si $A = \bigvee_{i=1,...n} C_i$ C_i cláusulas duales

Cada C_i tiene la forma

$C_i = \bigwedge_{j=1,...m} l_{ij}$ l_{ij} literales

Ejemplo:

$A = (p \rightarrow q) \rightarrow r$

$\equiv \neg(\neg p \vee q) \vee r$

$\equiv p \wedge \neg q \vee r$

✓ FORMA NORMAL CONJUNTIVA (FNC)

Resolución

FORMA NORMAL CONJUNTIVA (FNC)

Una fórmula A está en FNC si $A = \bigwedge_{i=1,...n} C_i$ C_i cláusulas

Cada C_i tiene la forma

$C_i = \bigvee_{j=1,...m} l_{ij}$ l_{ij} literales

Ejemplo

$A = (p \rightarrow q) \rightarrow r$

$\equiv \neg(\neg p \vee q) \vee r$

$\equiv p \wedge \neg q \vee r$

$FNC(A) = (p \vee r) \wedge (\neg q \vee r)$

Resolución

FORMA CLAUSULAR

A en FNC se puede escribir $cl(A) = \{C_1, C_2, ..., C_n\}$ C_i cláusulas

Cláusula $C = l_1 \vee l_2 \vee ... \vee l_m$ se escribe $C = \{l_1, l_2, ..., l_m\}$

Cláusula unitaria: cláusula con un solo literal $C = \{ l \}$

Ejemplos: $\neg p$ es cláusula unitaria y q también

$FNC(A) = (p \vee r) \wedge (\neg q \vee r)$

$Cl(A) = \{C_1, C_2\} = \{p \vee r, \neg q \vee r\}$ siendo $C_1 = p \vee r$ y $C_2 = \neg q \vee r$

También se puede escribir

$C_1 = \{p, r\}$ y $C_2 = \{\neg q, r\}$ p, r, $\neg q$ literales

$Cl(A) = \{\{p, r\}, \{\neg q, r\}\}$

Resolución

Cláusula satisfacible

C = $\{l_1, l_2, ..., l_m\}$ es satisfacible por una valuación v sí y sólo sí la fórmula $l_1 \vee l_2 \vee ... \vee l_m$ es satisfacible por v.

Es decir $v(l_i) = 1$ para algún $l_i \in C$

Fórmula en FNC satisfacible

A en FNC, A = $\{C_1, C_2, ..., C_n\}$, es satisfacible por una valuación v sí y sólo sí la fórmula $C_1 \wedge C_2 \wedge ... \wedge C_n$ es satisfacible por v

Resolución

Fórmula satisfacible

Una fórmula A es satisfacible sí y sólo sí la forma clausular asociada, cl(A) = $\{C_1, C_2, ..., C_n\}$, es satisfacible.

Cláusula vacía

C = { } (conjunto de literales es vacío)

Se denota también con $\perp$ y es insatisfacible

Conjunto de cláusulas S

-Si S = $\varnothing$ entonces S es satisfacible (S conjunto vacío de cláusulas)

-Si S = {{ }} = { $\perp$ } entonces S es insatisfacible (S contiene únicamente la cláusula vacía)

Resolución

Resolvente

Dada la forma clausular $A = \{p \vee \neg q, q \vee \neg t\}$
Sea v una valuación

$v(q) = 0$ para que $v(q \vee \neg t) = 1$ debe ser $v(t) = 0$ (depende de t)

$v(q) = 1$ para que $v(p \vee \neg q) = 1$ debe ser $v(p) = 1$ (depende de p)

$A = \{p \vee \neg q, q \vee \neg t\}$ es satisfacible sí y sólo sí

$p \vee \neg q$ y $q \vee \neg t$ son satisfacibles sí y sólo sí

p ó $\neg t$ es satisfacible

Satisfacibilidad de A depende de $p \vee \neg t$ A se puede reducir a $p \vee \neg t$

Resolución

Definición de Resolvente

Sean C_1 y C_2 cláusulas. Supongamos que $l \in C_1$ y $l^c \in C_2$

La resolvente de C_1 y C_2 es la cláusula

$Res(C_1, C_2) = (C_1 - \{ l \}) \cup (C_2 - \{ l^c \})$

Ejemplos

$Res(p \vee \neg q, q \vee \neg t) = p \vee \neg t$

$Res(\neg q, q) = \perp$

Cuidado

$Res(p \vee \neg q, q \vee \neg p) = p \vee \neg p$ resolviendo sobre q

$Res(p \vee \neg q, q \vee \neg p) = \neg q \vee q$ resolviendo sobre p

Resolución

Regla o principio de resolución

Determinación de la resolvente de un par de cláusulas

Dadas C_1 y C_2 cláusulas tales que $l \in C_1$ y $l^c \in C_2$ se denomina resolución a la regla

$$\frac{C_1 \qquad C_2}{(C_1 - \{ l \}) \cup (C_2 - \{ l^c \})}$$

Teorema

La regla de resolución preserva la satisfacibilidad. Es decir, si una valuación satisface un conjunto de cláusulas, también satisface cualquier resolvente de un par de ellas.

Resolución

1) Dados los siguientes conjuntos de cláusulas, calcule en cada caso todas las resolventes posibles

a) $S = \{ p \vee q, \neg p \vee r, \neg q \vee r, \neg r, p \vee t \}$

b) $S = \{ q, \neg p \vee r, \neg q \vee t, \neg r \vee p \}$

2) ¿Qué conclusión puede sacar sobre la satisfacibilidad de cada conjunto de cláusulas?

Resolución

Deducción por Resolución:

Sea S un conjunto de cláusulas y C una cláusula.

Una deducción por resolución de C a partir de S, $S \vdash_R C$, es una sucesión finita de cláusulas $C_1, C_2, \ldots, C_n$ tales que

1) $C_n = C$

2) Para $1 \leq i \leq n$ se cumple que

   a) $C_i \in S$ ó

   b) Existen cláusulas C_j, C_k con $j, k < n$ tal que $Res(C_j, C_k) = C_i$

Refutación de un conjunto de cláusulas:

Sea S un conjunto de cláusulas. Una refutación de S es una deducción por resolución de $\perp$ a partir de S.

En símbolos $S \vdash_R \perp$

Resolución

Teorema:

Sea S un conjunto de cláusulas. Entonces $S \vdash_R \perp$ sí y sólo sí S es insatisfacible.

Ejemplo:

Determinar si $S = \{ p \vee q, \neg p \vee r, \neg q \vee r, \neg r, p \vee t \}$ es satisfacible o no

1) $p \vee q$
2) $\neg p \vee r$
3) $\neg q \vee r$
4) $\neg r$
5) $p \vee t$
6) $Res(1, 2) = q \vee r$
7) $Res(3, 4) = \neg q$
8) $Res(7, 6) = r$
9) $Res(8, 4) = \perp$

Como $S \vdash_R \perp$

$\Downarrow$

S es insatisfacible

Resolución

En una deducción por resolución:

- **una cláusula puede ser utilizada más de una vez**
- **puede haber cláusulas que no se utilicen**
- **siempre se usa un conjunto finito de cláusulas**

Teorema:

Sea S un conjunto de cláusulas y C una cláusula.

Entonces $S \vdash_R C$ sí y sólo sí existe un subconjunto finito S_0

$S_0 \subseteq S$ tal que $S_0 \vdash_R C$

Resolución

✓ **El método de resolución es completo, correcto y decidible.**

Dados $\Gamma \cup \{ A \} \subseteq F_m$

Teorema de Completitud:

Si $\Gamma \models A$ entonces $cl(\Gamma \cup \{\neg A\}) \vdash_R \bot$

Teorema de Corrección:

Si $cl(\Gamma) \vdash_R cl(A)$ entonces $\Gamma \models A$

Decidibilidad:

Existe un algoritmo para decidir si una fórmula es deducible a partir de otras (puede aplicarse Algoritmo de Davis-Putnam)

Resolución

Corolario:

$\models A$ sí y sólo sí $cl(\neg A) \vdash_R \bot$

Corolario:

Sea S un conjunto de cláusulas.

$S \vdash_R \bot$ sí y sólo sí S es insatisfacible

Método para determinar si una fórmula A es tautología:

1) Negar la fórmula.

2) Calcular $cl(\neg A)$.

3) Usar resolución para determinar si $cl(\neg A) \vdash_R \bot$

Ejemplo: Determinar si $\models (p \rightarrow r) \wedge (q \rightarrow s) \rightarrow (p \wedge q \rightarrow r \wedge s)$

Resolución

Método para determinar si una deducción semántica es válida:

Dados $\Gamma \cup \{ A \} \subseteq F_m$

Para determinar si $\Gamma \models A$

1) Negar la conclusión (A)

2) Calcular $cl(\Gamma \cup \{\neg A \})$

3) Usar resolución para determinar si $cl(\Gamma \cup \{\neg A \}) \vdash_R \bot$

Si $cl(\Gamma \cup \{\neg A \}) \vdash_R \bot$ entonces $\Gamma \models A$ (deducción semántica es válida)

Si $cl(\Gamma \cup \{\neg A \}) \not\vdash_R \bot$ entonces $\Gamma \not\models A$ (deducción semántica no es válida)

Ejemplo:

Determinar si $p \rightarrow q \wedge r, \neg(s \vee t), q \leftrightarrow s \vee t \models \neg p$

Resolución

Cláusulas de Horn:

- Como máximo un literal positivo
- Pueden ser de la forma:

1) $\neg p_1 \vee \neg p_2 \vee ... \vee \neg p_n \vee q \equiv p_1 \wedge p_2 \wedge ... \wedge p_n \rightarrow q$ (reglas)

2) q (hechos)

3) $\neg p_1 \vee \neg p_2 \vee ... \vee \neg p_n \equiv \neg(p_1 \wedge p_2 \wedge ... \wedge p_n)$

Ejemplos:

1) $\neg p \vee \neg q \vee \neg r \vee s$ 2) s 3) $\neg p \vee \neg q$

Resolución unitaria: en cada paso interviene una cláusula unitaria

Si S es un conjunto de cláusulas de Horn insatisfacible entonces existe una refutación unitaria de S.

Resolución

Algoritmo de Davis-Putnam (sistematiza proceso de resolución):

Sea S conjunto finito de cláusulas y $\{p_1, p_2, ..., p_n\}$ conjunto de variables proposicionales que ocurren en cláusulas de S.

1) Sea $S_1 = S$

2) Sea $i = 1$

3) Repetir hasta que $i = n + 1$ (n nro. var. proposicionales)

4) Sea $S_i' = S_i - \{C\}$ C es cláusula que tiene l y l^c, l literal

5) Elegir una variable p_i y definir el conjunto de cláusulas que contienen p_i

$T_i = \{ C \in S_i' : p_i \in C \text{ o } \neg p_i \in C \}$

6) Calcular el conjunto de resolventes de pares de cláusulas en T_i

$R_i = \{ D:$ existen $C_1 \cup \{p_i\}, C_2 \cup \{\neg p_i\} \in T_i$ y

$D = Res(C_1 \cup \{p_i\}, C_2 \cup \{\neg p_i\}) \}$

7) Definir el conjunto $S_{i+1} = (S_i' - T_i) \cup R_i$

8) Sea $i = i + 1$

9) Volver a 3)

Resolución

- Como la cantidad de variables en S es finita (n), en n+1 pasos, como máximo, se eliminan todas las variables.

- S_{n+1} es el conjunto final o solución y puede ser sólo uno de los siguientes:

$S_{n+1} = \{ \bot \}$ es decir $S \vdash_R \bot$ $\Rightarrow$ S es insatisfacible

$S_{n+1} = \varnothing$ es decir $S \nvdash_R \bot$ $\Rightarrow$ S es satisfacible

Ejemplo:

Dado $S = \{ \neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q, \neg s, r, \neg s \vee p \vee s \}$

Aplicando el ADP determinar si S es satisfacible o no

Resolución

Ejemplo:

Dado $S = \{ \neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q, \neg s, r, \neg s \vee p \vee s \}$

1) $S_1 = S$

2) $i = 1$

3) $i < n + 1$

4) $S_1' = S_1 - \{\neg s \vee p \vee s\} = \{ \neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q, \neg s, r \}$

5) Elijo p

$T_1 = \{ \underset{1}{\neg p \vee \neg q \vee r}, \underset{2}{p \vee r \vee s}, \underset{3}{\neg p \vee q} \}$

6) (1, 2) $\neg q \vee r \vee s$ (2, 3) $r \vee s \vee q$

$R_1 = \{ \neg q \vee r \vee s, r \vee s \vee q \}$

7) $S_2 = (S_1' - T_1) \cup R_1 = \{\neg s, r\} \cup \{ \neg q \vee r \vee s, r \vee s \vee q \}$

$S_2 = \{\neg s, r, \neg q \vee r \vee s, r \vee s \vee q \}$

Resolución

4) $S_2' = S_2 = \{\neg s, r, \neg q \vee r \vee s, r \vee s \vee q\}$

5) Elijo s

$T_2 = \{\neg s, \neg q \vee r \vee s, r \vee s \vee q\}$

6) $R_2 = \{\neg q \vee r, r \vee q\}$

7) $S_3 = (S_2' - T_2) \cup R_2 = \{r, \neg q \vee r, r \vee q\}$

4) $S_3' = S_3 = \{r, \neg q \vee r, r \vee q\}$

5) Elijo r

$T_3 = \{r, \neg q \vee r, r \vee q\}$

6) $R_2 = \varnothing$

7) $S_4 = (S_3' - T_3) \cup R_3 = \varnothing$

Para $S = \{\neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q, \neg s, r, \neg s \vee p \vee s\}$

el resultado de aplicar el ADP es ∅, por lo tanto

$S \not\vdash_R \perp \Rightarrow$ S es satisfacible

Resolución

Algoritmo de Davis-Putnam (sistematiza proceso de resolución):

Sea S conjunto finito de cláusulas y $\{p_1, p_2, ..., p_n\}$ conjunto de variables proposicionales que ocurren en cláusulas de S.

1) Sea $S_1 = S$

2) Sea i = 1

3) Repetir hasta que i = n + 1 (n nro. var. proposicionales)

$S = \{\neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q, \neg s, r, \neg s \vee p \vee s\}$

1) $S_1 = S$

2) i = 1

3) i < n + 1

Resolución

Algoritmo de Davis-Putnam (sistematiza proceso de resolución):

Sea S conjunto finito de cláusulas y $\{p_1, p_2, ..., p_n\}$ conjunto de variables proposicionales que ocurren en cláusulas de S.

1) Sea $S_1 = S$

2) Sea i = 1

3) Repetir hasta que i = n + 1 (n nro. var. proposicionales)

4) Sea $S_i' = S_i - \{C\}$ C es cláusula que tiene l y l^c, l literal

$S = \{ \neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q, \neg s, r, \neg s \vee p \vee s \}$

1) $S_1 = S$

2) i = 1

3) i < n + 1

4) $S_1' = S_1 - \{\neg s \vee p \vee s\} = \{ \neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q, \neg s, r \}$

Resolución

Algoritmo de Davis-Putnam (sistematiza proceso de resolución):

Sea S conjunto finito de cláusulas y $\{p_1, p_2, ..., p_n\}$ conjunto de variables proposicionales que ocurren en cláusulas de S.

1) Sea $S_1 = S$

2) Sea i = 1

3) Repetir hasta que i = n + 1 (n nro. var. proposicionales)

4) Sea $S_i' = S_i - \{C\}$ C es cláusula que tiene l y l^c, l literal

5) Elegir una variable p_i y definir el conjunto de cláusulas que contienen p_i

$T_i = \{ C \in S_i' : p_i \in C \text{ o } \neg p_i \in C \}$

4) $S_1' = S_1 - \{\neg s \vee p \vee s\} = \{ \neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q, \neg s, r \}$

5) Elijo p

$T_1 = \{ \neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q \}$

Resolución

4) Sea $S_i' = S_i - \{C\}$ C es cláusula que tiene l y l^c, l literal

5) Elegir una variable p_i y definir el conjunto de cláusulas que contienen p_i

$T_i = \{ C \in S_i' : p_i \in C \text{ o } \neg p_i \in C \}$

6) Calcular el conjunto de resolventes de pares de cláusulas en T_i

$R_i = \{ D:$ existen $C_1 \cup \{p_i\}, C_2 \cup \{\neg p_i\} \in T_i$ y

$D = Res(C_1 \cup \{p_i\}, C_2 \cup \{\neg p_i\}) \}$

4) $S_1' = S_1 - \{\neg s \vee p \vee s\} = \{ \neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q, \neg s, r \}$

5) Elijo p

$T_1 = \{ \underbrace{\neg p \vee \neg q \vee r}_{1}, \underbrace{p \vee r \vee s}_{2}, \underbrace{\neg p \vee q}_{3} \}$

6) (1, 2) $\neg q \vee r \vee s$ (2, 3) $r \vee s \vee q$

$R_1 = \{ \neg q \vee r \vee s, r \vee s \vee q \}$

Resolución

4) Sea $S_i' = S_i - \{C\}$ C es cláusula que tiene l y l^c, l literal

5) Elegir una variable p_i y definir el conjunto de cláusulas que contienen p_i

$T_i = \{ C \in S_i' : p_i \in C \text{ o } \neg p_i \in C \}$

6) Calcular el conjunto de resolventes de pares de cláusulas en T_i

$R_i = \{ D:$ existen $C_1 \cup \{p_i\}, C_2 \cup \{\neg p_i\} \in T_i$ y

$D = Res(C_1 \cup \{p_i\}, C_2 \cup \{\neg p_i\}) \}$

7) Definir el conjunto $S_{i+1} = (S_i' - T_i) \cup R_i$

4) $S_1' = S_1 - \{\neg s \vee p \vee s\} = \{ \neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q, \neg s, r \}$

5) Elijo p

$T_1 = \{ \neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q \}$

6) $R_1 = \{ \neg q \vee r \vee s, r \vee s \vee q \}$

7) $S_2 = (S_1' - T_1) \cup R_1 = \{\neg s, r\} \cup \{ \neg q \vee r \vee s, r \vee s \vee q \}$

$S_2 = \{\neg s, r, \neg q \vee r \vee s, r \vee s \vee q \}$

Resolución

Algoritmo de Davis-Putnam (sistematiza proceso de resolución):

Sea S conjunto finito de cláusulas y $\{p_1, p_2, ..., p_n\}$ conjunto de variables proposicionales que ocurren en cláusulas de S.

1) Sea $S_1 = S$

2) Sea $i = 1$

3) Repetir hasta que $i = n + 1$ (n nro. var. proposicionales)

4) Sea $S_i' = S_i - \{C\}$ C es cláusula que tiene l y l^c, l literal

5) Elegir una variable p_i y definir el conjunto de cláusulas que contienen p_i

$T_i = \{ C \in S_i' : p_i \in C \text{ o } \neg p_i \in C \}$

6) Calcular el conjunto de resolventes de pares de cláusulas en T_i

$R_i = \{ D:$ existen $C_1 \cup \{p_i\}, C_2 \cup \{\neg p_i\} \in T_i$ y

$D = Res(C_1 \cup \{p_i\}, C_2 \cup \{\neg p_i\}) \}$

7) Definir el conjunto $S_{i+1} = (S_i' - T_i) \cup R_i$

8) Sea $i = i + 1$

9) Volver a 3)

Resolución

4) $S_2' = S_2 = \{\neg s, r, \neg q \vee r \vee s, r \vee s \vee q \}$

5) Elijo s

$T_2 = \{\neg s, \neg q \vee r \vee s, r \vee s \vee q\}$

6) $R_2 = \{\neg q \vee r, r \vee q\}$

7) $S_3 = (S_2' - T_2) \cup R_2 = \{ r, \neg q \vee r, r \vee q \}$

4) $S_3' = S_3 = \{ r, \neg q \vee r, r \vee q \}$

5) Elijo r

$T_3 = \{r, \neg q \vee r, r \vee q \}$

6) $R_3 = \emptyset$

7) $S_4 = (S_3' - T_3) \cup R_3 = \emptyset$

Resolución

- Como la cantidad de variables en S es finita (n), en n+1 pasos, como máximo, se eliminan todas las variables.

- S_{n+1} es el conjunto final o solución y puede ser sólo uno de los siguientes:

$S_{n+1} = \{ \bot \}$ es decir $S \vdash_R \bot \Rightarrow$ S es insatisfacible

$S_{n+1} = \varnothing$ es decir $S \not\vdash_R \bot \Rightarrow$ S es satisfacible

Dado $S = \{ \neg p \vee \neg q \vee r, p \vee r \vee s, \neg p \vee q, \neg s, r, \neg s \vee p \vee s \}$

Como el resultado de aplicar el ADP es $\varnothing$

$S \not\vdash_R \bot \Rightarrow$ S es satisfacible