ALGORITMOS DE SATURACIÓN.

DEF. Sean C_1 y C_2 cláusulas y ℓ una literal. Definimos la REGLA DE RESOLUCIÓN BINARIA como

$$\frac{C_1 \vee \ell \qquad\qquad C_2 \vee \ell^c}{C_1 \vee C_2}\ \text{RES}$$

↖ Resolvente

NOTA.

$$\frac{p \qquad\qquad \neg p}{\square}\ \text{RES}$$

Cláusula Vacía. ⟶

DEF. Sea S un conjunto de cláusulas. Definimos a la RESOLUCIÓN DE S, denotada por $R(S)$, como el conjunto que contiene <u>todos</u> los resolventes de cláusulas de S, ie,

$$R(S) = S \cup \{ E \mid \text{existen } C, D \in S \text{ t.q. } E \text{ es un resolvente de } C \text{ y } D \}$$

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

$$R(S) = S \cup \{ p \vee q \vee p,\ p \vee q,\ q \vee p,\ r \vee r,\ r \}$$

DEF. La n-ésima resolución de S se define recursivamente como sigue:

* $Res_0(S) = S$
* $Res_{n+1}(S) = R(Res_n(S))$

Ej. $Res_0(S) = S$

$Res_1(S) = R(Res_0(S)) = R(S)$

$Res_2(S) = R(Res_1(S)) = R(R(S)) = R(S) \cup \{ \text{resolventes} \}$

Proposición *Sea $\mathbb{S}$ es un conjunto finito de cláusulas. Entonces $\mathbb{S}$ es no satisfacible si y sólo si $\square \in Res_n(\mathbb{S})$ para alguna $n \in \mathbb{N}$.*

Algoritmo de Saturación.

Así, dado un conjunto de cláusulas S, podemos construir los conjuntos $Res_n(S)$ hasta alcanzar uno de los sig. casos

1. En algún momento $\square$ es generada, es decir, $\square \in Res_n(S)$ para algún $n \in \mathbb{N}$.

   En este caso Γ es insatisfacible.

2. El algoritmo termina sin generar $\square$, es decir, para algún $n \in \mathbb{N}$, $Res_n(S) = Res_{n+1}(S)$ y $\forall i \leq n$, $\square \notin Res_i(S)$.

   Ya no se generan nuevos resolventes

   En este caso S es satisfacible.

3. La saturación <u>no</u> termina; el algoritmo se ejecuta hasta agotar sus recursos y <u>no</u> se genera $\square$.

   En este caso se desconoce si S sea satisfacible

Ej. $S =$

1. $p \vee r$
2. $\neg p \vee \neg r$
3. p
4. r

0. $Res_0(S) = S$

1. $Res_1(S) = R(Res_0(S)) = R(S) = S \cup$

5.	$p \vee \neg p$	res (1, 2)
6.	$r \vee \neg r$	res (1, 2)
7.	$\neg r$	res (2, 3)
8.	$\neg p$	res (3, 4)

2. $Res_2(S) = R(Res_1(S)) = Res_1(S) \cup$ 9. $\square$ res (4, 7)

$\therefore$ S <u>no</u> es satisfacible.