Introducción

Los presentes apuntes contienen una introducción a la lógica proposicional y sus aplicaciones orientada principalmente a carreras técnicas.

Los apuntes se utilizan en la primera parte de la asignatura “Lógica” de la Escuela Universitaria de Ingeniería Técnica Informática de Oviedo impartida por los autores.

Para cualquier consulta o sugenrencia, puede ponerse en contacto con los autores en: labra@lsi.uniovi.es ó anaisabel@lsi.uniovi.es

J. E. Labra G
Ana I. Fernández M.
Octubre, 1998

Contenido

1. Lenguaje de la Lógica Proposicional

La lógica Proposicional pretende estudiar las frases declarativas simples (enunciados o proposiciones) que son los elementos básicos de transmisión de conocimiento humano.

De manera informal, una proposición se define como una frase que puede ser considerada Verdadera o Falsa y que no se puede descomponer en otras frases Verdaderas o Falsas.

Para relacionar las distintas proposiciones se utilizan las siguientes conectivas:

Nombre de la conectiva	**Representación**	**Ejemplos de frases en las que aparece**
Negación	$\neg p$	no *p* es falso *p* no es cierto *p*
Conjunción	$p \wedge q$	*p* y *q* *p* pero *q* *p* sin embargo *q* *p* no obstante *q* *p* a pesar de *q*
Disyunción	$p \vee q$	o *p* o *q* o ambos al menos *p* o *q* como mínimo *p* o *q*
Condicional (Implicación)	$p \rightarrow q$	si *p* entonces *q* *p* sólo si *q* *q* si *p* *q* cuando *p* *q* es necesario para *p* para *p* es necesario *q* *p* es suficiente para *q* para *q* es suficiente *p* no *p* a menos que *q*
Bicondicional (Equivalencia)	$p \leftrightarrow q$	*p* es necesario y suficiente para *q* *p* si y sólo si *q*

1.1. Alfabeto de la Lógica Proposicional

El lenguaje de la lógica proposicional trabajará con los siguientes conjuntos de símbolos:

Constantes:	**V F**
Variables o letras proposicionales:	p, q, r, ...
Símbolos de Conectivas:	$\neg \wedge \vee \rightarrow \leftrightarrow$
Signos de puntuación:	( )

1.2. Sintaxis de la Lógica Proposicional

Las reglas de formación de frases en el lenguaje de la lógica proposicional (LPROP) son:

1.- Las constantes **V** (Verdadero) y **F** (Falso) pertenecen a LPROP
2.- Las letras de proposición *p,q,r*,.. pertenecen a LPROP

3.- Si A y B pertenecen a LPROP entonces $(\neg A), (\neg B), (A \wedge B), (A \vee B), (A \rightarrow B), (A \leftrightarrow B)$ pertenecen a LPROP

4.- Sólo pertenecen a LPROP las fórmulas que cumplan los requisitos 1, 2 y 3.

Con el fin de evitar el exceso de paréntesis se establece la siguiente jerarquía de prioridades:

$\neg$
$\wedge$ $\vee$
$\rightarrow$ $\leftrightarrow$

Con dicha tabla, la fórmula $\neg p \vee q \rightarrow p \wedge r$ se reconocería como: $((\neg p) \vee q) \rightarrow (p \wedge r)$

1.3. Semántica de la Lógica Proposicional

La teoría semántica de la lógica proposicional trata de atribuir significados (Verdadero o Falso) a las distintas fórmulas del lenguaje. Dichos significados dependen del contexto particular en el que se utilice la fórmula. Cada contexto se denomina Interpretación.

Definición 1: Una **interpretación** de una fórmula *F* en lógica proposicional es una asignación de valores {**V, F**} a cada una de las letras proposicionales de *F.* El valor de una proposición *p* bajo una interpretación *I* se denota como $V_I(p)$.

Definición 2: Dada una fórmula *F* y una interpretación *I*, el **valor de *F* bajo *I* ,** denotado por $V_I(F)$ es:

° Si *F* está formada por una proposición *p*, entonces $V_I(F)= V_I(p)$

° Si F es de la forma $\neg$G entonces $V_I(F)=\begin{cases}\mathbf{V} & si\, V_I(G)=\mathbf{F}\\ \mathbf{F} & si\, V_I(G)=\mathbf{V}\end{cases}$

° Si F es de la forma G $\wedge$ H entonces $V_I(F)=\begin{cases}\mathbf{V} & si\, V_I(G)=V_I(H)=\mathbf{V}\\ \mathbf{F} & \text{en caso contrario}\end{cases}$

° Si F es de la forma G $\vee$ H entonces $V_I(F)=\begin{cases}\mathbf{F} & si\, V_I(G)=V_I(H)=\mathbf{F}\\ \mathbf{V} & \text{en caso contrario}\end{cases}$

° Si F es de la forma G $\rightarrow$ H entonces $V_I(F)=\begin{cases}\mathbf{F} & si\, V_I(G)=\mathbf{V} \text{ y } V_I(H)=\mathbf{F}\\ \mathbf{V} & \text{en caso contrario}\end{cases}$

° Si F es de la forma G $\leftrightarrow$ H entonces $V_I(F)=\begin{cases}\mathbf{V} & si\, V_I(G)=V_I(H)\\ \mathbf{F} & \text{en caso contrario}\end{cases}$

Ejemplo 1: Sea la fórmula F= $(p\rightarrow q)\leftrightarrow\neg q\vee\neg p$ y la interpretación I que asigna $V_I(p)=\mathbf{F}$ y $V_I(q)=\mathbf{V}$. Entonces $V_I(F)=\mathbf{V}$

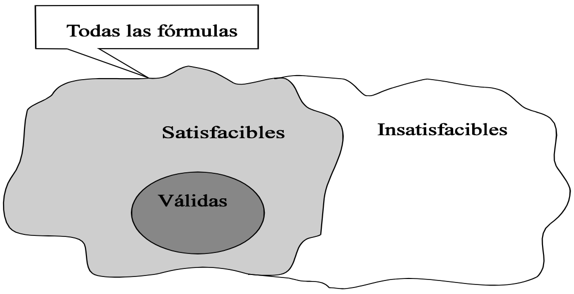
Definición 3: Una interpretación *I* es un **modelo** para una fórmula *F* si $V_I(F)=\mathbf{V}$

Es posible establecer una clasificación de las fórmulas proposicionales en función de los valores que tomen bajo las diferentes interpretaciones, de esta forma una fórmula *F* se clasifica en:

Válida ó Tautología: Todas las interpretaciones son un modelo (Para toda interpretación *I*, $V_I(F)=\mathbf{V}$)

Satisfacible: Alguna interpretación es un modelo (Existe una interpretación *I* tal que $V_I(F)=\mathbf{V}$)

Insatisfacible: Ninguna interpretación es un modelo (No existe una interpretación *I* tal que $V_I(F)=\mathbf{V}$)



Una fórmula puede ser: satisfacible o insatisfacible. Un tipo especial de fórmula satisfacible, es aquella que toma **siempre** valor **V** (es válida). Por tanto, las fórmulas válidas son un subconjunto de las satisfacibles.

Teorema 1: Una fórmula *F* es válida si y sólo si su negación $\neg F$ es insatisfacible.

Dem: F es válida

$\Leftrightarrow$ { Def. válida}

$\forall I\ V_I(F) = \mathbf{V}$

$\Leftrightarrow$ { Def. Interpretación $\neg$ }

$\forall I\ V_I(\neg F) = \mathbf{F}$

$\Leftrightarrow$ { Def. Insat. }

$\neg$F es Insatisfacible

NOTA: A lo largo de estos apuntes se utilizará un formato lineal para las demostraciones promovido por E. W. Dijkstra [Dijkstra, 90]. En este formato, las líneas impares contienen los principales pasos de la demostración y las líneas pares, comentarios para pasar de un paso a otro.

En algunas ocasiones, el comentario recurre a la regla de Leibniz que dice lo siguiente:

Si se cumple F(X) y X = Y entonces también se cumple F(Y)

2. Equivalencia lógica

Definición 4: Se dice que dos fórmulas A y B son **equivalentes lógicamente** (se denota por $A \equiv B$ ó $A \Leftrightarrow B$) si para toda interpretación I, se cumple que $V_I(A) = V_I(B)$

Teorema 2: $A \equiv B$ si y sólo si la fórmula $A \leftrightarrow B$ es válida

Dem: $A \equiv B$

$\Leftrightarrow$ { Def. ≡ }

$\forall I\ V_I(A) = V_I(B)$

$\Leftrightarrow$ { Def. Interpretación ↔ }

$\forall I\ V_I(A \leftrightarrow B) = \mathbf{V}$

$\Leftrightarrow$ { Def. Válida }

$A \leftrightarrow B$ es válida

El teorema anterior reduce la demostración de equivalencia entre fórmulas a la demostración de validez de una fórmula.

A continuación se presenta una tabla con una serie de equivalencias de uso común y de fácil demostración

Supresión de Implicación: $A \to B \equiv \neg A \lor B$
Contraposición: $A \to B \equiv \neg B \to \neg A$
Supresión de Doble Implicación: $A \leftrightarrow B \equiv (A \to B) \land (B \to A)$

Absorción	$A \land (B \lor A) \equiv A$	$A \lor (B \land A) \equiv A$
Dominación	$A \land \mathbf{F} \equiv A$	$A \lor \mathbf{V} \equiv \mathbf{V}$
Elemento neutro	$A \land \mathbf{V} \equiv A$	$A \lor \mathbf{F} \equiv A$
E. Complementario	Contradicción $A \land \neg A \equiv \mathbf{F}$	Medio Excluido $A \lor \neg A \equiv \mathbf{V}$
Idempotencia	$A \land A \equiv A$	$A \lor A \equiv A$
Commutativa	$A \land B \equiv B \land A$	$A \lor B \equiv B \lor A$
Asociativa	$A\land (B\land C)\equiv (A\land B)\land C$	$A\lor (B\lor C)\equiv (A\lor B)\lor C$
Distributiva	$A\lor (B\land C)\equiv(A\lor B)\land (A\lor C)$	$A\land(B\lor C)\equiv(A\land B)\lor(A\land C)$
De Morgan	$\neg(A\lor B)\equiv\neg A\lor\neg B$	$\neg(A\land B)\equiv\neg A\lor\neg B$
Doble Negación (Involución)	$\neg\neg A\equiv A$	

Teorema 3: Si A es válida y $A \equiv B$ entonces B es válida

Dem: A es válida

$\Leftrightarrow$ { Def. Válida }

$\forall I\ V_I(A) = \mathbf{V}$

$\Leftrightarrow$ {Si $A\equiv B$ entonces $\forall I\ V_I(A) = V_I(B)$, Leibniz }

$\forall I\ V_I(B) = \mathbf{V}$

$\Leftrightarrow$ { Def. Válida }

B es válida

Con el teorema anterior, si se sabe que X es válida, para demostrar que Z es válida se podrá utilizar el formato: X

≡ {...}

Y

≡ {...}

Z

3. Consecuencia Lógica

Definición 5: Sea *C* un conjunto de fórmulas $\{P_1, P_2, ...P_n\}$ y sea *Q* una fórmula. Se dice que *Q* es **consecuencia lógica** del conjunto C (se denotará $C \Rightarrow Q$) si toda interpretación que es un modelo de *C* es también un modelo de *Q*.

Es decir, si para toda interpretación *I se* cumple que si $V_I(P_1) = V_I(P_2) = ... = V_I(P_n) = \mathbf{V}$ entonces $V_I(Q) = \mathbf{V}$

Intuitivamente, se podría considerar cada interpretación como una posible situación. Decir que *Q* es consecuencia lógica de unas premisas es equivalente a pensar que *Q* toma valor **V** en cualquier situación en la que las premisas tomen valor **V**.

Una estructura de la forma $\{P_1, P_2, ...P_n\} \Rightarrow Q$ se denomina **razonamiento.** Donde $\{P_1, P_2, ...P_n\}$ es el conjunto de premisas y *Q* la conclusión.

Se dice que un razonamiento es **correcto** si la conclusión es consecuencia lógica de las premisas.

Teorema 4: $\{P_1, P_2, ...P_n\} \Rightarrow Q$ es correcto si y sólo si $P_1 \wedge P_2 \wedge ... \wedge P_n \rightarrow Q$ es válida

Dem: $\{P_1, P_2, ...P_n\} \Rightarrow Q$ es correcto

$\Leftrightarrow$ { Def. Razonamiento }

$\forall I$ Si $V_I(P_1) = V_I(P_2) = ... = V_I(P_n) = \mathbf{V}$ entonces $V_I(Q) = \mathbf{V}$

$\Leftrightarrow$ { Def. Interpretación de conjunción }

$\forall I$ Si $V_I(P_1 \wedge P_2 \wedge ... \wedge P_n) = \mathbf{V}$ entonces $V_I(Q) = \mathbf{V}$

$\Leftrightarrow$ { Def. Interpretación de Implicación }

$\forall I$ $V_I(P_1 \wedge P_2 \wedge ... \wedge P_n \rightarrow Q) = \mathbf{V}$

$\Leftrightarrow$ { Def. Válida }

$P_1 \wedge P_2 \wedge ... \wedge P_n \rightarrow Q$ es válida

4. Técnicas Semánticas de Estudio de Validez Proposicional

4.1. Tablas de Verdad

Definición 6: Una **tabla de verdad** es una representación tabular del valor de una fórmula en todas las posibles interpretaciones.

Por ejemplo, para calcular el valor de verdad de la fórmula $F = p \rightarrow q \leftrightarrow \neg p \vee q$, la tabla de verdad consiste en representar las 4 posibles interpretaciones y evaluar la fórmula en dichas interpretaciones

p	q	$p \rightarrow q \leftrightarrow \neg p \vee q$
F	**F**	**V**
F	**V**	**V**
V	**F**	**V**
V	**V**	**V**

El número de posibles interpretaciones de una fórmula F es 2^n donde *n* es el número de variables proposicionales de F. Por tanto, este método tiene una complejidad exponencial que complica su utilización para fórmulas complejas

4.2. Árboles Semánticos

Definición 7: Un **árbol semántico** es una técnica similar a las tablas de verdad que puede simplificar la evaluación de algunas fórmulas.

Inicialmente, se forma el conjunto *LP* de letras proposicionales de la fórmula. Se construye un nodo inicial del árbol que se tomará como nodo actual y se aplica el siguiente procedimiento:

1.- Se intenta evaluar la fórmula en el nodo actual.

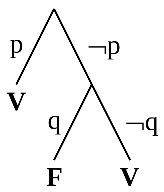
2.- Si es posible asignar a *F* un valor $\{\mathbf{V},\mathbf{F}\}$ se etiqueta el nodo con dicho valor y se finaliza el tratamiento del nodo actual.

3.-En caso.contrario:

- Se Selecciona la primera letra proposicional *p* del conjunto *LP*
- Se Borra *p* de *LP*.
- Se Construyen dos ramas, una correspondiente a *p* interpretado con valor **V** (identificada como *p*) y la otra correspondiente a *p* con valor **F** (identificada como $\neg p$).
- Repetir el procedimiento por cada uno de los dos nuevos nodos.

Definición 8: Los nodos del árbol semántico en los que el conjunto de significados atribuidos hasta ellos hacen Falsa la fórmula, se denominan **nodos de fallo** y los que la hacen verdadera, **nodos de éxito**

Ejemplo 2: Dada la fórmula $(p \rightarrow q) \rightarrow (\neg p \rightarrow \neg q)$. Seleccionando los literales por orden alfabético, se obtiene el árbol semántico:



Como puede observarse, no ha sido necesario evaluar las interpretaciones p=**V,** q=**V** y p=**V**, q=**F**.

4.3. Demostraciones por Contradicción

Para demostrar que una fórmula F es válida por contradicción se realiza lo siguiente:

> 1.- Se supone que existe una interpretación I tal que $V_I(F) = \mathbf{F}$ y se intentan calcular los diversos valores de la fórmula.
>
> 2.- Si se llega a una contradicción:
>
> Entonces: $\neg\exists I\ V_I(F) = \mathbf{F} \Rightarrow \forall I\ V_I(F) = \mathbf{V} \Rightarrow$ F es válida
>
> En Caso Contrario: $\exists I\ V_I(F) = \mathbf{F} \Rightarrow$ F no es válida

Este tipo de demostraciones se suelen representar etiquetando la fórmula con valor **F** y evaluando posibles valores hasta que se llegue la contradicción.

Ejemplo 3. A continuación se demuestra que la fórmula $\neg p \vee \neg q \rightarrow \neg(p \wedge q)$ es válida

$$\underbrace{\underbrace{\underbrace{\underbrace{\neg \underbrace{p}_{V}}_{F} \vee \underbrace{\neg \underbrace{q}_{V}}_{F}}_{F}}_{\mathbf{V}} \rightarrow \underbrace{\neg \underbrace{(\underbrace{p}_{V} \wedge \underbrace{q}_{V})}_{V}}_{\mathbf{F}}}_{\mathbf{F}}$$

Contradicción

A la hora de evaluar una conectiva pueden aparecer varias alternativas. Conviene recordar que:

> Para poder asegurar que F es válida debe llegarse a contradicción por **todas** las alternativas
>
> Si no se llega a contradicción por alguna alternativa se puede decir que F no es válida

Ejemplo 4. A continuación se demuestra la transitividad de la equivalencia lógica. Es decir que:

$\{A \leftrightarrow B, B \leftrightarrow C) \Rightarrow (A \leftrightarrow C)$

Para ello, basta con demostrar que la fórmula $(A \leftrightarrow B) \wedge (B \leftrightarrow C) \rightarrow (A \leftrightarrow C)$ es válida. En dicha demostración aparecen dos alternativas y, como se llega a contradicción por ambas, puede concluirse que la fórmula es válida.

$\underbrace{\underbrace{\underbrace{\underbrace{A}_{\mathbf{V}} \leftrightarrow \underbrace{B}_{\mathbf{V}}}_{\mathbf{V}} \wedge \underbrace{\underbrace{B}_{\mathbf{V}} \leftrightarrow \underbrace{C}_{\mathbf{V}}}_{\mathbf{V}}}_{\mathbf{V}} \rightarrow \underbrace{\underbrace{\underbrace{A}_{\mathbf{V}} \leftrightarrow \underbrace{C}_{\mathbf{V}}}_{\mathbf{V}}}_{F}}_{F}$ Contradicción	$\underbrace{\underbrace{\underbrace{\underbrace{A}_{\mathbf{F}} \leftrightarrow \underbrace{B}_{\mathbf{F}}}_{V} \wedge \underbrace{\underbrace{B}_{\mathbf{F}} \leftrightarrow \underbrace{C}_{\mathbf{F}}}_{V}}_{V} \rightarrow \underbrace{\underbrace{\underbrace{A}_{\mathbf{F}} \leftrightarrow \underbrace{C}_{\mathbf{F}}}_{V}}_{\mathbf{F}}}_{\mathbf{F}}$ Contradicción

4.4. Resolución Proposicional

El método de resolución es un algoritmo fácilmente mecanizable propuesto por J.A. Robinson en 1965. La entrada del algoritmo no es una fórmula, sino un conjunto de cláusulas y el algoritmo chequea si son insatisfacibles. Antes de presentar el algoritmo de resolución, se define qué es una cláusula y cómo transformar una fórmula en un conjunto de cláusulas mediante las formas normales.

4.4.1. Formas Normales

Definición 9: Una fórmula *F* es una **conjunción** si es de la forma $F_1 \wedge F_2 \wedge \cdots \wedge F_n$ n≥0

Definición 10: Una fórmula *F* es una **disyunción** si es de la forma $F_1 \vee F_2 \vee \cdots \vee F_n$ n≥0

Definición 11: Un **literal** es una proposición (p) o una proposición negada $(\neg p)$.

Definición 12: Una fórmula *F* está en **Forma Normal Conjuntiva (FNC)** si es una conjunción de la forma $F_1 \wedge F_2 \wedge \cdots \wedge F_n$ donde cada F_i es una disyunción de literales. Se representa como $\overset{m}{\underset{i=1}{\wedge}}\left(\overset{n_i}{\underset{j=1}{\vee}} l_{ij}\right)$

Ejemplo 5: La siguiente fórmula está en Forma Normal Conjuntiva: $(\neg p \vee q) \wedge (\neg p \vee r \vee \neg s) \wedge p$

Definición 13: Una fórmula *F* está en **Forma Normal Disyuntiva (FND)** si es una disyunción de la forma $F_1 \vee F_2 \vee ... \vee F_n$ donde cada F_i es una conjunción de literales. Se representa como $\overset{m}{\underset{i=1}{\vee}}\left(\overset{n_i}{\underset{j=1}{\wedge}} l_{ij}\right)$

Ejemplo 6: La siguiente fórmula está en Forma Normal Disyuntiva: $(p \wedge \neg q \wedge r) \vee \neg p \vee (r \wedge \neg s)$

Ejemplo 7: Obsérvese que la fórmula $\neg p$ está a la vez en FNC y FND

Teorema 5: Toda fórmula de la lógica de proposiciones puede ser transformada en una fórmula lógicamente equivalente a ella en Forma Normal Conjuntiva (Disyuntiva).

Dem:

La demostración consiste en indicar los pasos del algoritmo de transformación a forma normal conjuntiva. Puesto que estos pasos mantienen la equivalencia y dado que la equivalencia cumple la propiedad transitiva (ejemplo **4**), la fórmula resultante es equivalente a la fórmula original. Para demostrar formalmente que el algoritmo termina, se requiere el estudio de sistemas de re-escritura de términos que puede consultarse en [Abramsky, 92]. Los pasos de transformación son:

1. Eliminar conectiva $\leftrightarrow$

   $A \leftrightarrow B \equiv (A \rightarrow B) \wedge (B \rightarrow A)$
2. Eliminar conectiva $\rightarrow$

   $A \rightarrow B \equiv \neg A \vee B$
3. Introducir negaciones hasta que afecten a literales mediante las leyes de Morgan.

   $\neg(A \wedge B) \equiv \neg A \vee \neg B \qquad \neg(A \vee B) \equiv \neg A \wedge \neg B$
4. Eliminar negaciones múltiples.

   $\neg\neg A \equiv A$
5. Aplicar propiedades distributivas para eliminar las posibles conjunciones (disyunciones) dentro de disyunciones (conjunciones) obteniendo Forma Normal Conjuntiva (Disyuntiva).

   $A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C) \qquad A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$

Puesto que las fórmulas resultantes de aplicar cada uno de los pasos anteriores mantienen la equivalencia, la fórmula obtenida será equivalente a la fórmula original.

En muchas ocasiones se añaden otros tres pasos que simplifican la fórmula resultante:

6. Eliminar conjunciones/disyunciones con un literal y su opuesto.

   $(p \wedge \neg p \wedge X) \vee Y \equiv Y$

   $(p \vee \neg p \vee X) \wedge Y \equiv Y$
7. Eliminar literales repetidos

   $p \wedge p \equiv p$

$p \vee p \equiv p$

8. Eliminar subsunciones. Una subsunción se produce cuando una conjunción (o disyunción) C está incluida en otra D. En dicho caso se elimina la cláusula D
   $(A \vee B) \wedge A \equiv A$
   $(A \wedge B) \vee A \equiv A$

Ejemplo 8: Para transformar la fórmula $\neg(p\rightarrow q)\leftrightarrow p\vee r$ a Forma Normal Conjuntiva, se pueden emplear los siguientes pasos:

$\neg(p\rightarrow q)\leftrightarrow p\vee r$

$\equiv$ { Eliminación $\leftrightarrow$ }

$(\neg (p \rightarrow q) \rightarrow p \vee r) \wedge ((p \vee r) \rightarrow \neg(p \rightarrow q))$

$\equiv$ { Eliminación $\rightarrow$ }

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

$\equiv$ { Eliminación doble negación }

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

$\equiv$ { Eliminación disyunción con literal y su opuesto }

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

$\equiv$ { De Morgan }

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

$\equiv$ { Eliminación doble negación }

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

$\equiv$ { Distributiva $\vee$ }

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

$\equiv$ { Distributiva $\vee$ }

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

Definición 14: Una **cláusula** es una disyunción de literales.

Definición 15: Una fórmula está en **Forma Clausal** si se expresa como un conjunto de cláusulas.

La transformación de una fórmula en Forma Normal Conjuntiva a Forma Clausal es inmediata sustituyendo las conectivas $\wedge$ por comas y englobando las disyunciones entre llaves.

Ejemplo 9: La fórmula $(\neg p \vee q) \wedge (\neg p \vee q \vee \neg r) \wedge (p)$ en Forma Normal Conjuntiva equivale a $\{\neg p \vee q, \neg p \vee q \vee \neg r, p\}$ en Forma Clausal

Definición 16: Una cláusula sin literales se denomina **cláusula vacía**, se representa por π y su valor es siempre Falso.

Definición 17: Una cláusula que tiene a lo sumo un literal positivo, se denomina **cláusula Horn**. Una cláusula Horn será de la forma: $A\vee\neg B_1\vee\neg B_2\vee...\vee\neg B_n$

Si n=0, se denomina **hecho,** si no existe literal positivo (no existe A) entonces se denomina **objetivo** y, finalmente, si $n>0$ y existe literal positivo, se denomina **regla**.

4.4.2. Algoritmo de Resolución Proposicional

El algoritmo se basa en una regla de inferencia sencilla y, a la vez, de gran potencia: la **regla de resolución**. Puesto que se utiliza una sola regla, el algoritmo es fácil de analizar e implementar.

La idea del principio de resolución es simple: Si se sabe que se cumple: "P ó Q" y también se sabe que se cumple "no P ó R" entonces se puede deducir que se cumplirá "Q ó R".

Ejemplo 10: Si se tiene: "Gana o Pierde o Empata" y "Si Gana entonces da una Fiesta o Va de Viaje". Se puede deducir que: "O Pierde o Empata o da una Fiesta o va de Viaje".

Formalizando, la primera frase sería: $G \vee P \vee E$ y la segunda: $G \rightarrow F \vee V \equiv \neg G \vee F \vee V$

La regla de resolución inferirá: $P \vee E \vee F \vee V$

Definición 18: Dadas dos cláusulas C_1 y C_2 tales que exista un literal *l* de forma que $l \in C_1$ y $\neg l \in C_2$, se denomina **resolvente de** C_1 **y** C_2 **respecto a** l a la cláusula:

$$R_l(C_1, C_2) = \left(C_1 - \{l\}\right) \cup \left(C_2 - \{\neg l\}\right).$$

Se dice que C_1 y C_2 son **cláusulas resolubles**.

Teorema 6 (Consistencia de la regla de resolución): El resolvente de dos cláusulas es consecuencia lógica de ellas. Es decir $\{C_1, C_2\} \Rightarrow R(C_1, C_2)$

Demostración: Se demuestra por contradicción:

Sea $C_1 = l \vee l_{11} \vee l_{12} \vee \cdots \vee l_{1m}$ y $C_2 = \neg l \vee l_{21} \vee l_{22} \vee \cdots \vee l_{2n}$.

El resolvente de C_1 y C_2 respecto a *l* será $R_l(C_1, C_2) = l_{11} \vee l_{12} \vee \cdots \vee l_{1m} \vee l_{21} \vee l_{22} \vee \cdots \vee l_{2n}$

Por el teorema 2, probar que $\{C_1, C_2\} \Rightarrow R(C_1, C_2)$ es equivalente a probar que $C_1 \wedge C_2 \rightarrow R_l(C_1, C_2)$ es válida. Supóngase que existe una interpretación que la hace Falsa, la asignación de valores será:

$$\underbrace{\underbrace{\underbrace{\underbrace{l}_{V} \vee \underbrace{l_{11}}_{F} \vee \cdots \vee \underbrace{l_{1m}}_{F}}_{V} \wedge \underbrace{\underbrace{\underbrace{\underbrace{\neg \underbrace{l}_{V}}_{F} \vee \underbrace{l_{21}}_{F} \vee \cdots \vee \underbrace{l_{2n}}_{F}}_{F}}_{V}}}_{V} \rightarrow \underbrace{\underbrace{l_{11}}_{F} \vee \cdots \vee \underbrace{l_{1m}}_{F} \vee \underbrace{l_{21}}_{F} \vee \cdots \vee \underbrace{l_{2n}}_{F}}_{F}}_{F}$$

Contradicción

Puesto que se llega a una contradicción, la fórmula no puede ser Falsa y será siempre verdadera, es decir, la fórmula es Válida.

Teorema 7: Dadas dos cláusulas C_1 y C_2 pertenecientes a un conjunto C *y* resolubles respecto un literal *l*, entonces: $C \equiv C \cup R_l(C_1, C_2)$.

Demostración**:** Sea $C = C_1 \wedge C_2 \wedge \ldots \wedge C_n$, entonces, el conjunto $C \cup R_l(C_1, C_2)$ será $C_1 \wedge C_2 \wedge \ldots \wedge C_n \wedge R_l(C_1, C_2)$. La demostración se reduce a comprobar que la fórmula $C_1 \wedge C_2 \wedge \ldots \wedge C_n \leftrightarrow C_1 \wedge C_2 \wedge \ldots \wedge C_n \wedge R_l(C_1, C_2)$ es válida. Para ello, se tiene en cuenta el teorema anterior y se demuestra por contradiccón.

$$\underbrace{\underbrace{\underbrace{C_1}_{V} \wedge \underbrace{C_2}_{V} \wedge \cdots \wedge \underbrace{C_n}_{V}}_{V} \leftrightarrow \underbrace{\underbrace{\underbrace{C_1}_{V} \wedge \underbrace{C_2}_{V} \wedge \cdots \wedge \underbrace{C_n}_{V} \wedge \underbrace{R_l(C_1, C_2)}_{V}}_{V}}_{F}}_{F}$$

Obsérvese el valór **V** se obtiene a partir del teorema anterior

$$\underbrace{\underbrace{\underbrace{\underset{V}{C_1}\wedge\underset{V}{C_2}\wedge\cdots\wedge\underset{V}{C_n}}_{V}}_{F} \leftrightarrow \underbrace{\underset{V}{C_1}\wedge\underset{V}{C_2}\cdots\wedge\underset{V}{C_n}\wedge\underset{V}{R_l(C_1,C_2)}}_{V}}_{F}$$

Teorema 8: Si el resolvente de dos cláusulas C_1 y C_2 pertenecientes a un conjunto *C* es la cláusula vacía, entonces *C* es insatisfacible.

Demostración**:** En primer lugar, se demuestra que el conjunto *C* es equivalente a **F**

C
≡ { Teorema **7** }
C ∧ $R_l(C_1,C_2)$
≡ { Hipótesis }
C ∧ π
≡ {Definición, π = **F**}
C ∧ **F**
≡ { Dominación }
F

Finalmente, se demuestra que cualquier fórmula equivalente a **F** es insatisfacible

C ≡ **F**
⇔ { Def. Interpretación ≡ }
∀I V_I(C) = V_I(**F**)
⇔ { Def. Interpretación: V_I(**F**) = **F** }
∀I V_I(C) = **F**
⇔ { Def. Insatisfacible }
C es insatisfacible

A partir de los teoremas anteriores, se define el algoritmo de resolución que chequeará si un conjunto de cláusulas es insatisfacible.

Algoritmo de resolución proposicional

Entrada: Un conjunto de cláusulas *C*

Salida: Detecta si *C* es insatisfacible

1.- Buscar dos cláusulas $C_1, C_2 \in C$ tales que exista un literal *l* que cumple que $l \in C_1$ y $\neg l \in C_2$

2.- Si se encuentran:

3.- Calcular $R_l(C_1, C_2)$ y añadirlo al conjunto *C*

4.- Si $R_l(C_1, C_2) = \square$ entonces **SALIR** indicando que *C* es **insatisfacible.**

5.- Si no, Volver a 1

3.- Si no se encuentran: **SALIR** indicando que *C* **no es insatisfacible.**

Ejemplo 11: Sea *C* el siguiente conjunto de cláusulas $\{p, \neg p \vee q, \neg r, \neg p \vee \neg q \vee r\}$, se puede demostrar que *C* es insatisfacible por resolución. Para ello:

- Se resuelve la tercera cláusula ($\neg r$) con la cuarta ($\neg p \vee \neg q \vee r$), obteniendo $\neg p \vee \neg q$.

- Se resuelve ahora la cláusula anterior con la segunda cláusula ($\neg p \vee q$) obteniendo: $\neg p$

- Se resuelve ahora la cláusula anterior con la primera y se llega a la cláusula vacía o

Puesto que se llega a la cláusula vacía, *C* es insatisfacible.

Teorema 9: Un razonamiento de la forma $P_1, P_2, \cdots, P_n \Rightarrow Q$ es correcto si y sólo si el conjunto de cláusulas $\{P_1^c, P_2^c, \cdots P_n^c, \neg Q^c\}$ es insatisfacible. Cada P_i^c es el resultado de transformar la premisa P_i a forma clausal y $\neg Q^c$ es el resultado de transformar la negación de la conclusión a forma clausal.

Dem: $P_1, P_2, \cdots, P_n \Rightarrow Q$

$\Leftrightarrow$ { Teorema **4**}

$P_1 \wedge P_2 \wedge \cdots \wedge P_n \rightarrow Q$ es válida

$\Leftrightarrow$ { Teorema **1**}

$\neg(P_1 \wedge P_2 \wedge \cdots \wedge P_n \rightarrow Q)$ es insatisfacible

$\Leftrightarrow$ { Pasando a Forma Normal Conjuntiva cada premisa y operando }

$\{P_1^c, P_2^c, \cdots P_n^c, \neg Q^c\}$ es insatisfacible

Ejemplo 12: Para estudiar si el razonamiento $\{p \wedge q \rightarrow r \wedge s, p \rightarrow \neg s\} \Rightarrow \neg p \vee \neg q$ es correcto por resolución, es necesario transformar cada premisa a forma clausal y añadir el resultado de transformar la negación de la conclusión a forma clausal. El conjunto obtenido sería $\{\neg p \vee \neg q \vee r, \neg p \vee \neg q \vee s, \neg p \vee \neg s, p, q\}$. Aplicando el algoritmo de resolución:

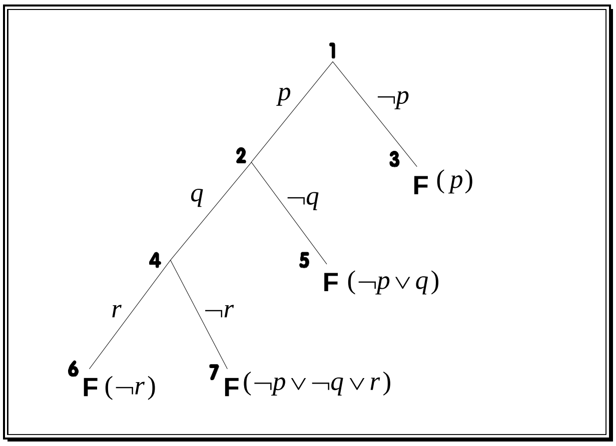
- Se resuelve la segunda cláusula ($\neg p \vee \neg q \vee s$) con la tercera ($\neg p \vee \neg s$), obteniendo $\neg p \vee \neg q$
- Se resuelve ahora la cláusula anterior con la cuarta cláusula (p) obteniendo: $\neg q$
- Se resuelve ahora la cláusula anterior con la quinta y se llega a la cláusula vacía o

Puesto que se llega a la cláusula vacía, el conjunto de cláusulas es insatisfacible y el razonamiento es correcto.

Demostración de la completud del algoritmo de resolución

Se presentan las ideas generales de la demostración de la completud del algoritmo de resolución proposicional. Una demostración formal requiere técnicas que se salen del ámbito de estos apuntes y puede consultarse en [Fitting, 96]

Ejemplo 13: Sea el conjunto de cláusulas $C = \{p, \neg p \vee q, \neg r, \neg p \vee \neg q \vee r\}$, para construir el árbol semántico para C se recuerda que un conjunto de cláusulas equivale a una fórmula en Forma Normal Conjuntiva, en este caso, $p \wedge (\neg p \vee q) \wedge (\neg r) \wedge (\neg p \vee \neg q \vee r)$. En la siguiente figura se muestra el árbol semántico correspondiente marcando la cláusula falsificada en los nodos de fallo. El árbol semántico será:



Lema 1: Si un conjunto de cláusulas es insatisfacible, entonces el árbol semántico es finito y está limitado por nodos de fallo, se denomina, en ese caso, **árbol de fallo**.

Lema 2: Cada nodo de fallo *n* falsifica al menos a una de las cláusulas del conjunto que será la **cláusula asociada a *n*.**

Lema 3: La cláusula *C* asociada a un nodo de fallo *n* contiene un subconjunto de los complementos de los literales que aparecen en la rama que va desde la raíz del árbol semántico hasta *n*.

Demostración**:** Puesto que la cláusula *C* es falsificada en el nodo *n*, todos sus literales deben tener asignado un valor en la interpretación parcial correspondiente a *n*. Además, el valor de esos literales debe ser **F** (puesto que *C* es una disyunción). El valor asignado debe ser el complementario.

Definición 19: Se denomina **nodo de inferencia** a un nodo del árbol semántico cuyos dos hijos son nodos de fallo.

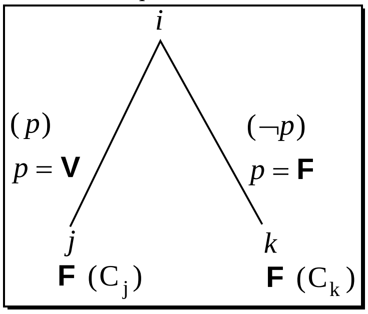
Lema 4: En un árbol de fallo, salvo que sólo tenga un nodo, debe existir al menos un nodo de inferencia.

Demostración**:** Puesto que el árbol de fallo es finito y las ramas se desarrollan de dos en dos, necesariamente tendremos un último nodo desarrollado con dos hijos.

Lema 5: Si el árbol semántico de un conjunto de cláusulas es de fallo y contiene un sólo nodo, entonces dicho conjunto contiene la cláusula vacía.

Lema 6: Un nodo de inferencia *i* indica un paso de resolución de las cláusulas asociadas a sus dos hijos. El resolvente de dichas cláusulas es falsificado por el nodo *i* y, ocasionalmente, por alguno de sus antecesores.

Demostración**:** En un nodo de inferencia *i* cualquiera, se tendrá un esquema como el que sigue:



- Puesto que el nodo *i* no falsificó C_j y lo único que cambia en el nodo *j* respecto a *i* es el valor de *p*, la cláusula C_j debe contener el literal $\neg p$ (complementado para que sea Falso).

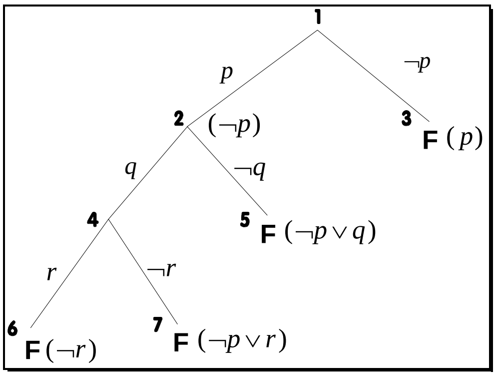
- Por la misma razón anterior, la cláusula C_k debe contener el literal *p* (sin complementar para que sea Falso)

Por tanto C_j y C_k son resolubles respecto a *p*. El esquema será:

$$\left.\begin{array}{r} C_j = \neg p \vee resto\_C_j \\ C_k = p \vee resto\_C_k \end{array}\right\} R_p(C_j, C_k) = resto\_C_j \vee resto\_C_k$$

En el nodo *j*, C_j toma valor Falso, por tanto $resto\_C_j$ tomará también valor Falso, como $resto\_C_j$ no contiene el literal *p* también tomarán valor Falso en el nodo *i*. De la misma forma, $resto\_C_k$ tomará valor Falso en *el nodo i*. Por tanto, $R_p(C_j, C_k) = resto\_C_j \vee resto\_C_k$ tomará valor Falso en el nodo *i*, es decir, el nodo *i*, es un nodo de fallo para el resolvente de C_j y C_k

En ocasiones, puede ocurrir que el resolvente sea falsificado también por alguno de los padres del nodo de inferencia, como ejemplo, considérese el conjunto de cláusulas $\{p, \neg p \vee q, \neg r, \neg p \vee r\}$**,** el árbol semántico, junto con los resolventes sería:



El resolvente de los nodos 6 y 7 es ($\neg p$) que falsifica al nodo 4 pero también falsifica a su antecesor, el nodo 2.

Teorema 10 (Completud del Algoritmo de Resolución Proposicional): Si un conjunto de cláusulas es insatisfacible entonces, aplicando el algoritmo de resolución, se alcanza la cláusula vacía.

Demostración**:** C es un conjunto de cláusulas insatisfacibles
$\Leftrightarrow$ {Lema 1}
El árbol semánttico de C será un árbol de fallo
$\Leftrightarrow$ {Lema 4}
Existe un nodo de inferencia i
$\Leftrightarrow$ {Lema 6, un nodo de inferencia indica un paso de resolución }
Se puede formar el resolvente con las cláusulas asociadas a los dos hijos i
El resolvente puede añadirse al conjunto C
Se construye un nuevo árbol semántico
$\Leftrightarrow$ { Hipótesis, C es insatisfacible, Consistencia Resolución }
El nuevo árbol seguirá siendo insatisfacible, pero contendrá menos nodos

Repitiendo el proceso se llegará a un árbol semántico con un solo nodo que corresponderá a la cláusula vacía {Lema 5}. Quedando demostrado que se alcanza la cláusula vacía por resolución.

4.4.3. Estrategias de resolución

El método de resolución es un algoritmo no determinista ya que pueden encontrarse múltiples formas de alcanzar la cláusula vacía en un conjunto insatisfacible. Muchas veces, siguiendo un determinado camino se alcanzará la cláusula vacía con muchos menos pasos de resolución que por otro camino.

Durante el desarrollo del algoritmo es necesario responder las siguientes preguntas: ¿Qué dos cláusulas se seleccionan? y ¿sobre qué literales se realiza la resolución?.

Las distintas estrategias de resolución tratan de responder a ambas preguntas de forma que se mantenga la completud (si el conjunto es insatisfacible, alcanzar la cláusula vacía) y que se obtenga un comportamiento eficiente.

Una de las desventajas de la utilización de la reglas de resolución sin ninguna restricción consiste en que se pueden seleccionar cláusulas cuyo resolvente no sea útil en el camino de búsqueda de la cláusula vacía. Se observa que muchas veces los resolventes son redundantes o no aportan ninguna utilidad para la búsqueda. A continuación se mencionan una serie de estrategias que servirán para eliminar el trabajo inútil.

4.4.3.1. Estrategias de Borrado

Una estrategia de borrado será una técnica en la cual se eliminan una serie de cláusulas antes de que sean utilizadas. Si dichas cláusulas no van a aportar nada para la búsqueda de la cláusula vacía, su eliminación permitirá un ahorro computacional.

4.4.3.1.1. Eliminación de cláusulas con literales puros

Definición 20: Un literal es **puro** si y sólo si no existe un literal complementario a él en el conjunto de cláusulas.

Una cláusula que contenga un literal *puro* es inútil en la búsqueda de la cláusula vacía, puesto que el literal *puro* no podrá ser eliminado nunca mediante resolución. Por tanto, una estrategia de borrado consiste en la eliminación de cláusulas con literales puros.

Ejemplo 14: El conjunto $C=\{\neg p\vee\neg q\vee r,\neg p\vee s,\neg q\vee s,p,q,\neg r\}$ es insatisfacible, sin embargo, para demostrarlo, se puede ignorar la segunda y la tercera cláusula, puesto que ambas contienen el literal puro *s*.

4.4.3.1.2. Eliminación de tautologías

Definición 20: Una **tautología** es una cláusula que contiene el mismo literal en su forma directa e inversa.

Ejemplo 15: La cláusula $p\vee\neg q\vee r\vee\neg p$ es una *tautología*.

La presencia o ausencia de tautologías en un conjunto de cláusulas no afecta la condición de satisfacibilidad del conjunto. Un conjunto de cláusulas permanecerá satisfacible independientemente de que se le añadan tautologías. De la misma forma, un conjunto de cláusulas insatisfacible seguirá siendo insatisfacible aunque se eliminen todas sus tautologías. Es posible, por tanto, eliminar las tautologías de un conjunto de cláusulas para que no intervengan en el proceso de búsqueda sin alterar la satisfacibilidad del conjunto.

4.4.3.1.3. Eliminación de Subsunciones

Definición 21: Una cláusula *C* **subsume** a una cláusula *D* si y sólo si todo literal de *C* pertenece también a *D*, es decir, $C\subseteq D$.

Ejemplo 16: La cláusula $p\vee\neg q$ subsume a la cláusula $p\vee\neg q\vee r$.

Debido a la ley de absorción, un conjunto de cláusulas en el que se eliminan todas las cláusulas subsumidas es equivalente al conjunto original. Las cláusulas subsumidas pueden ser, por tanto, eliminadas.

Es necesario observar que, durante el desarrollo del proceso de resolución, se pueden generar resolventes de cláusulas que sean *tautologías* o *cláusulas subsumidas.* Las estrategias de borrado deberán chequear el conjunto de cláusulas original así como los distintos resolventes generados en cada resolución.

4.4.3.2. Resolución unitaria

Definición 22: Un resolvente **unitario** es un resolvente en el cual al menos uno de sus padres es una cláusula unitaria (con un sólo literal).

Una **estrategia de resolución unitaria** es una aplicación del algoritmo de resolución en la cual todos los resolventes son unitarios.

Ejemplo 17: Sea $C=\{p\vee q,\neg p\vee r,\neg q\vee r,\neg r\}$. A continuación se aplicará la estrategia de resolución unitaria, para ello, se seleccionan siempre dos cláusulas resolubles tales que una de ellas tenga un literal.

1.- $p\vee q$		7.- q $R_p(1,5)$
2.- $\neg p\vee r$		8.- p $R_q(1,6)$
3.- $\neg q\vee r$		9.- r $R_q(3,7)$
4.- $\neg r$		10.- ❐ $R_q(6,7)$
5.- $\neg p$	$R_r(2,4)$	
6.- $\neg q$	$R_r(3,4)$	

Obsérvese que los resolventes generados son un subconjunto de los que se podrían generar mediante la resolución sin restricciones. Por ejemplo, las cláusulas 1 y 2 podrían haberse seleccionado para obtener $q\vee r$. Sin embargo ni esa cláusula ni sus descendientes podrán ser generados porque ninguna de las cláusulas que la generan es unitaria.

Los procedimientos de resolución basados en resolución unitaria son sencillos de implementar y, normalmente, bastante eficientes. Obsérvese que si una cláusula es resuelta con una cláusula unitaria, su resolvente tiene menos literales que la cláusula original. De esa forma los procedimientos siguen una búsqueda directa hacia la cláusula vacía ganando en eficiencia.

Desafortunadamente, los procedimientos de inferencia basados en resolución unitaria no son, en general, completos. Por ejemplo, el conjunto $C=\{p\vee q,\neg p\vee q,p\vee\neg q,\neg p\vee\neg q\}$ es insatisfacible, sin embargo, la resolución unitaria no encontrará la cláusula vacía porque ninguna de las cláusulas es unitaria.

Por otro lado, restringiendo el formato de cláusulas a cláusulas Horn (cláusulas con un literal positivo como máximo) se puede demostrar que si un conjunto de cláusulas Horn es insatisfacible, entonces se llegará a la cláusula vacía aplicando la estrategia de resolución unitaria.

4.4.3.3. Resolución de Entrada

Definición 23: Un **resolvente de entrada** es un resolvente en el cual al menos uno de sus padres es una cláusula del conjunto original de entrada.

Una **estrategia de resolución de entrada** es una aplicación del algoritmo de resolución en la cual todos los resolventes son de entrada.

Ejemplo 18: Sea $C=\{p\vee q,\neg p\vee r,\neg q\vee r,\neg r\}$. A continuación se aplicará la estrategia de resolución de entrada, para ello, se seleccionan siempre dos cláusulas resolubles tales que una de ellas pertenezca al conjunto inicial de cláusulas:

1.- $p\vee q$			7.- $\neg p$	$R_r(2,4)$
2.- $\neg p\vee r$			8.- r	$R_p(2,6)$
3.- $\neg q\vee r$			9.- ❒	$R_r(4,8)$
4.- $\neg r$				
5.- $q\vee r$	$R_p(1,2)$			
6.- $p\vee r$	$R_q(1,3)$			

Se puede demostrar que la resolución unitaria y la resolución de entrada tienen el mismo poder de inferencia en el sentido de que si con una estrategia se puede alcanzar la cláusula vacía, con la otra también.

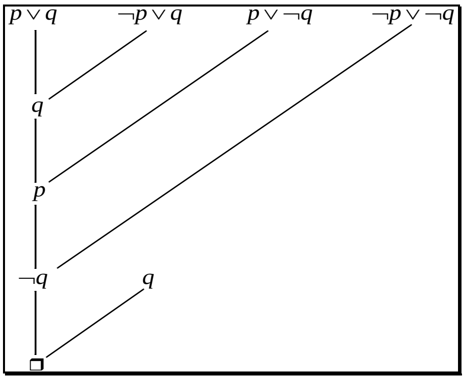
Una consecuencia de lo anterior es que la resolución de entrada es completa para cláusulas Horn, pero incompleta en general. Como contraejemplo, se puede tomar el del apartado anterior.

4.4.3.4. Resolución Lineal

La resolución lineal (también conocida como resolución con filtrado de antepasados) es una ligera generalización de la resolución de entrada. Se escoge una cláusula inicial o cláusula cabeza C_0 y se forma una cedena de resolventes $R_0,R_1,R_3,\cdots,R_n$ donde:

$$R_0=C_0$$
$$R_{i+1}=R(R_i,C_i)\quad \text{tal que } C_i\in C \text{ ó } C_i=R_j\ \ (\text{j}\le\text{i})$$

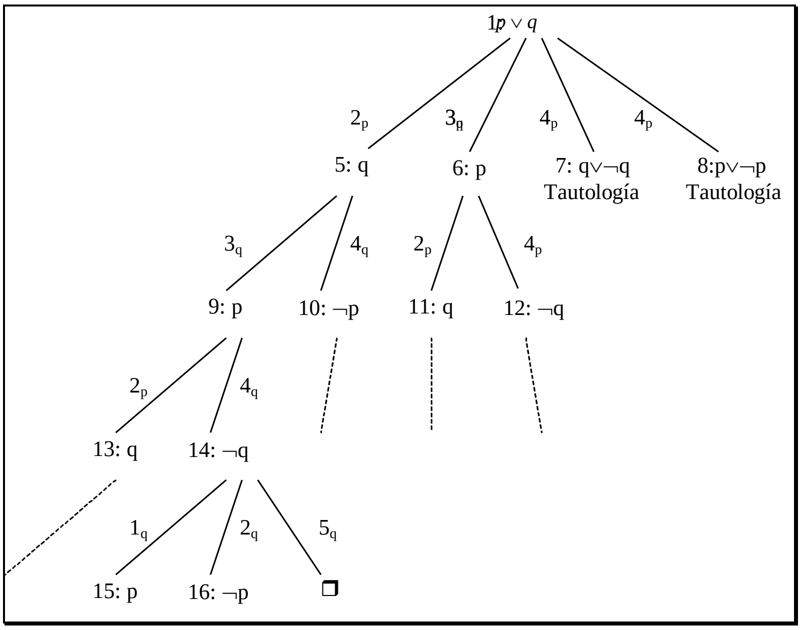
La resolución lineal toma su nombre del aspecto lineal que presentan las inferencias realizadas. Una resolución lineal comienza con una cláusula del conjunto inicial y produce una cadena lineal de resoluciones como la que se muestra en la figura para el conjunto de cláusulas $C=\{p\vee q,\neg p\vee q,p\vee\neg q,\neg p\vee\neg q\}$. Obsérvese que cada resolvente, después del primero, se obtiene del resolvente anterior y de alguna otra cláusula del conjunto.



Resolución Lineal

La resolución lineal evita muchas resoluciones inútiles centrándose en cada paso en los antepasados de una cláusula y en los elementos del conjunto inicial.

Los resultados obtenidos aplicando resolución para una determinada cláusula cabeza se pueden mostrar en forma de **árbol de resolución**. La raíz del árbol es la cláusula cabeza y se forman los nodos descendientes según las cláusulas con las que se pueda resolver. El árbol de resolución para el ejemplo anterior sería:



Árbol de resolución

En la figura se representan las resoluciones indicando el número de cláusula y el literal por el que se resuelve. A cada resolvente se le asigna un nuevo número. Obsérvese que pueden existir caminos infinitos (el camino más a la izquierda), caminos que llevan a tautologías y caminos de éxito que alcanzan la cláusula vacía.

Se puede demostrar que la resolución lineal es completa. Para cualquier conjunto de cláusulas insatisfacibles, aplicando resolución lineal, se alcanza la cláusula vacía.

Debido al siguiente teorema, no siempre es necesario probar con todas las cláusulas del conjunto inicial como cláusulas cabeza.

Teorema 11: Si un conjunto de cláusulas S es satisfacible y $S \cup C$ es insatisfacible, entonces se encuentra la cláusula vacía mediante resolución lineal tomando como cláusula cabeza una cláusula del conjunto C.

El teorema anterior tiene aplicación al estudio de los razonamientos, en los cuales las premisas son, por lo general, satisfacibles. Si al añadir las cláusulas resultantes de negar la conclusión el conjunto resultante es insatisfacible (y el razonamiento es correcto) entonces, según el teorema anterior basta con probar como cláusula cabeza con las que resultaron de negar la conclusión.

4.4.3.5. Resolución Ordenada

La resolución *ordenada* o *selectiva* es una estrategia de resolución muy restrictiva en la cual cada cláusula se toma como un conjunto de literales ordenados. La resolución sólo se realiza con el primer literal de cada cláusula. Los literales del resolvente mantienen el orden de las cláusulas padre con los literales del padre positivo (la cláusula que contenía el literal por el que se resuelve afirmado) seguidos de los literales del padre negativo (la cláusula que contenía el literal por el que se resuelve negado).

Ejemplo 19: Sea $C=\{p\vee q,\neg p\vee r,\neg q\vee r,\neg r\}$. A continuación se aplicará la estrategia de resolución ordenada (se han ordenado los literales de cada cláusula por orden alfabético):

1.- $p\vee q$	5.- $q\vee r$	$R_p(1,2)$
2.- $\neg p\vee r$	6.- r	$R_q(3,5)$
3.- $\neg q\vee r$	7.- ❒	$R_r(4,6)$
4.- $\neg r$		

La cláusula 5 es el único resolvente ordenado entre las cláusulas 1 y 4. Las cláusulas 1 y 3 no resuelven puesto que sus literales complementarios no son los primeros. Por la misma razón tampoco resuelven las cláusulas 2 y 4 ni las cláusulas 3 y 4. Una vez generada la cláusula 5, resuelve con la cláusula 3 para producir la cláusula 6, la cual resuelve con la cláusula 4 para producir la cláusula vacía.

La resolución ordenada es la más eficiente (en el ejemplo, se obtuvo la cláusula vacía en el tercer paso de resolución). Desafortunadamente, la resolución ordenada no es completa. Sin embargo, se ha demostrado que la resolución ordenada sí es completa para cláusulas Horn.

Tras este breve repaso de las principales estrategias de resolución, cabe reseñar que los principales sistemas de demostración automática basados en el principio de resolución (por ejemplo, los sistemas Prolog) utilizan una combinación de las dos últimas estrategias restringidas a conjuntos de cláusulas Horn[1].

[1]Los sistemas Prolog utilizan la resolución lineal ordenada para cláusulas Horn en lógica de predicados. Conocida como resolución SLD (***S****elective* ***L****inear Resolution for* ***D****efinite Clauses).*

5. Teoría de la Prueba: Deducción Natural

En las secciones anteriores se han utilizado técnicas que estudian la corrección de los razonamientos en base al significado de las fórmulas que contienen. Este conjunto de técnicas se engloban en lo que se denomina **teoría semántica.** Por el contrario, existe otro conjunto de técnicas, conocido como **teoría de la prueba**, que prescinde de los posibles valores de las fórmulas y se centra únicamente en la manipulación sintáctica de fórmulas. Existen diversos estilos como el sistema de Hilbert, la deducción natural, etc. Todos ellos utilizan un conjunto de axiomas y una serie de reglas de inferencia que permiten obtener teoremas a partir de dichos axiomas o de otros teoremas previamente derivados.

En esta sección se presenta el estilo de deducción natural, desarrollado por Gentzen en 1935 y cuyo principal objetivo es ofrecer un sistema que se acerque a las técnicas de demostración habituales. La deducción natural no contiene axiomas y ofrece una serie de reglas de inferencia por cada tipo de conectiva. Las reglas de inferencia se presentan en la siguiente tabla.

Reglas de Introducción		Reglas de Eliminación	
$\wedge$-I	$\dfrac{A \quad B}{A \wedge B}$	$\wedge$-E	$\dfrac{A \wedge B}{A}$ $\quad$ $\dfrac{A \wedge B}{B}$
$\vee$-I	$\dfrac{A}{A \vee B}$ $\quad$ $\dfrac{B}{A \vee B}$	Prueba por casos $\vee$-E	$\dfrac{A \vee B \quad A \rightarrow C \quad B \rightarrow C}{C}$
Deducción $\rightarrow$-I	$\dfrac{\boxed{\begin{array}{c} A \\ \vdots \\ B \end{array}}}{A \rightarrow B}$	Modus Ponens $\rightarrow$-E	$\dfrac{A \quad A \rightarrow B}{B}$
$\leftrightarrow$-I	$\dfrac{A \rightarrow B \quad B \rightarrow A}{A \leftrightarrow B}$	$\leftrightarrow$-E	$\dfrac{A \leftrightarrow B}{A \rightarrow B}$ $\quad$ $\dfrac{A \leftrightarrow B}{B \rightarrow A}$
Dem. por Contradicción $\neg$-I	$\dfrac{\boxed{\begin{array}{c} A \\ \vdots \\ B \wedge \neg B \end{array}}}{\neg A}$	Dem. por Contradicción $\neg$-E	$\dfrac{\boxed{\begin{array}{c} \neg A \\ \vdots \\ B \wedge \neg B \end{array}}}{A}$
$\mathbf{V}$-I	$\dfrac{\neg A \vee A}{\mathbf{V}}$	$\mathbf{V}$-E	$\dfrac{\mathbf{V}}{\neg A \vee A}$
$\mathbf{F}$-I	$\dfrac{A \wedge \neg A}{\mathbf{F}}$	$\mathbf{F}$-E	$\dfrac{\mathbf{F}}{A}$

Tabla 1: Reglas de Inferencia para Deducción Natural

Las reglas de la forma • - I se refieren a la inclusión del símbolo • y las reglas de la forma • - E se refieren a la eliminación de dicho símbolo.

Ejemplo 20: Demostrar que $p \wedge q \rightarrow q \wedge p$

1	$p \wedge q$	Premisa
2	q	$\wedge$-E 1
3	p	$\wedge$-E 1
4	$q \wedge p$	$\wedge$-I 2,3

Para el estudio de razonamientos de la forma $\{P_1, P_2, ...P_n\} \Rightarrow Q$ se parte de las premisas y se intenta llegar a la conclusión.

Ejemplo 21: Demostrar que $p \wedge q \Rightarrow p \wedge (q \vee r)$

1	$p \wedge q$	Premisa
2	p	$\wedge$-E 1
3	q	$\wedge$-E 1
4	$q \vee r$	$\vee$-I 3
5	$p \wedge (q \vee r)$	$\wedge$-I 2,4

La utilización de cuadros permite visualizar la idea de **pruebas subordinadas**. En una prueba subordinada, se realiza un supuesto y, una vez llegado a un resultado, se descarta el supuesto (se cierra el cuadro) obteniendo un resultado libre de supuestos. Un ejemplo es la regla de deducción:

$$\rightarrow\text{- I} \quad \frac{\boxed{\begin{array}{c} A \\ \vdots \\ B \end{array}}}{A \rightarrow B}$$

Esta regla enuncia que, si se supone A y se llega a demostrar B, entonces, se puede deducir la fórmula $A \rightarrow B$.

Ejemplo 22: Demostrar que $p \rightarrow (q \rightarrow r) \Rightarrow p \wedge q \rightarrow r$

1	$p \rightarrow (q \rightarrow r)$	Premisa
2	$p \wedge q$	Supuesto
3	p	$\wedge$-E 2
4	$q \rightarrow r$	$\rightarrow$ E 1,3
5	q	$\wedge$-E 2
6	r	$\rightarrow$ E 4,5
7	$p \wedge q \rightarrow r$	$\rightarrow$ I 2-6

Ejemplo 23: Demostrar que $p \wedge q \rightarrow r \Rightarrow p \rightarrow (q \rightarrow r)$

1	$p \wedge q \rightarrow r$	Premisa
2	p	Supuesto
3	q	Supuesto
4	$p \wedge q$	$\wedge$-I 2,3
5	r	$\rightarrow$ E 1,4
6	$q \rightarrow r$	$\rightarrow$ I 3-5
7	$p \rightarrow (q \rightarrow r)$	$\rightarrow$ I 2-6

Ejemplo 24: Demostrar que $\neg p \Rightarrow p \rightarrow q$

1	$\neg p$	Premisa
2	p	Supuesto
3	$p \wedge \neg p$	$\wedge$-I 1,2
4	**F**	**F**-I 3
5	q	**F-**E 4
6	$p \rightarrow q$	$\rightarrow$ I 2,5

Ejemplo 25: Demostrar $p \leftrightarrow \neg\neg p$

1	p	Supuesto
2	$\neg p$	Supuesto
3	$p \wedge \neg p$	$\wedge$-I 1,2
4	$\neg\neg p$	$\neg$I 2-3
5	$p \rightarrow \neg\neg p$	$\rightarrow$ I 2-4
6	$\neg\neg p$	Supuesto
7	$\neg p$	Supuesto
8	$\neg p \wedge \neg\neg p$	$\wedge$-I 6,7
9	p	$\neg$-E 7-8
10	$\neg\neg p \rightarrow p$	$\rightarrow$-I 6-9
11	$p \leftrightarrow \neg\neg p$	$\leftrightarrow$-I 5,10

Ejemplo 26: Demostrar que $\neg p \vee q \Rightarrow p \rightarrow q$

1	$\neg p \vee q$	Premisa
3	$\neg p$	Supuesto
4	$p \rightarrow q$	$\neg A \Rightarrow A \rightarrow B$ (Demostrado en Ej. **24**)
5	$\neg p \rightarrow (p \rightarrow q)$	$\rightarrow$ I 3-4
6	q	Supuesto
7	p	Supuesto
8	q	6
9	$p \rightarrow q$	$\rightarrow$-I 7-8
10	$q \rightarrow (p \rightarrow q)$	$\rightarrow$-I 6-10
11	$p \rightarrow q$	$\vee$-E 1,5,10

Ejemplo 27: Demostrar que $p \rightarrow q \Rightarrow \neg p \vee q$

1	$p \rightarrow q$	Premisa
2	$\neg (\neg p \vee q)$	Supuesto
3	$\neg p$	Supuesto
4	$\neg p \vee q$	$\vee$-I 3
5	$(\neg p \vee q) \wedge \neg(\neg p \vee q)$	$\wedge$-I 4,2
6	p	$\neg$-E 3-5
7	q	$\rightarrow$-E 1,6
8	$\neg p \vee q$	$\vee$-I 7
10	$(\neg p \vee q) \wedge \neg(\neg p \vee q)$	$\wedge$-I 8,2
11	$\neg p \vee q$	$\neg$-E 2-10

6. Aplicación al diseño de Circuitos: Álgebra de Boole

6.1. Introducción

George Boole (1815-1864) presentó el primer tratamiento sistemático de la lógica y para ello, desarrolló un sistema algebraico, conocido ahora como **Álgebra de Boole**. Además de sus aplicaciones al campo de la lógica, el álgebra de Boole ha tenido dos aplicaciones importantes: el tratamiento de conjuntos mediante las operaciones de unión e intersección que ha servido de base a la teoría de la probabilidad y el diseño de **circuitos digitales combinacionales**.

6.2. Definición de álgebra de Boole y Teoremas

Definición 24: Un **álgebra de Boole** es una estructura de la forma **{A, +, ×, -, 0,1}** siendo A un conjunto en el que se definen las siguientes operaciones:

$+$ y × son leyes de composición binaria sobre A: $a + b \in A$ y $a \times b \in A$ $\forall a,b \in$ A

- es una ley de composición unaria sobre A: $\overline{a} \in A$ $\forall$ a $\in$ A

verificándose los postulados:

1. Conmutativa:	a + b = b + a	∀a,b ∈ A	{conmutativa +}
	a × b = b × a	∀a,b ∈ A	{conmutativa ×}
2. Distributiva:	a + (b × c) = (a + b) × (a + c)	∀a,b,c ∈ A	{distributiva +}
	a × (b + c) = (a × b) + (a × c)	∀a,b,c ∈ A	{distributiva ×}
3. Elemento neutro:	a + 0=a	∀ a ∈ A	{neutro +}
	a × 1=a	∀ a ∈ A	{neutro ×}
4. Elemento inverso:	$a + \overline{a} = 1$	∀ a ∈ A	{inverso +}
	$a * \overline{a} = 0$	∀ a ∈ A	{inverso ×}

En el caso más sencillo, el conjunto *A* tiene como únicos elementos a los neutros de las operaciones, *A={ 0 , 1 }*. Esto quiere decir que las variables sólo pueden tomar los valores 0 o 1. En este caso, el álgebra de Boole se dice que es ***bivaluada***.

Ejemplo 28: La lógica proposicional LPROP, tiene estructura de álgebra de Boole.

Basta con demostrar que la tupla ({**F**,**V**}, ∨, ∧, $\overline{\ }$, **F**, **V**) cumple la definición de álgebra de Boole

Los elementos de LPROP, las fórmulas de la lógica proposicional, sólo pueden tomar los valores **F** o **V**, o lo que es lo mismo, 0 o 1.

La disyunción y la conjunción son operaciones binarias (intervienen dos operandos) e internas.

El complementario es una operación unaria interna.

Como ya se ha demostrado, se cumplen los postulados del álgebra de Boole:

Commutativa: a ∨ b = b ∨ a y a ∧ b = b ∧ a ∀a,b∈A

Distributiva: a ∨ (b ∧ c) = (a ∨ b) ∧ (a ∨ c)

a ∧ (b ∨ c) = (a ∧ b) ∨ (a ∧ c) ∀a,b∈A

Elemento neutro: a ∨ **F** = a

a ∧ **V** =a ∀a∈A

Elemento inverso: a ∨ ¬a = **V** (medio excluido) y a ∧ ¬a = **F** (contradicción) ∀a∈A

Ejemplo 29: Dado un conjunto C, la estructura $\{2^C, \cup, \cap, -, C, \varnothing\}$, donde 2^C es el conjunto de todos los subconjuntos de C, $\cup$, $\cap$ y - son las operaciones de unión, intersección y complementario entre conjuntos y $\varnothing$ es el conjunto vacío, tiene estructura de álgebra de Boole

Ejemplo 30: La estructura {A, ⊗, ⊕, ', 0, I} donde A contiene los elementos: {0, 1, S, S' } y las operaciones se definen mediante las siguientes tablas, también tiene estructura de álgebra de Boole

⊗	0	S	S'	I
0	0	0	0	0
S	0	S	0	S
S'	0	0	S'	S'
I	0	S	S'	I

⊕	0	S	S'	I
0	0	S	S'	I
S	S	S	I	I
S'	S'	I	S'	I
I	I	I	I	I

X	X'
0	I
S	S'
S'	S
I	0

Ejemplo 31: Dado un número natural *n*, la estructura $\{D_n$, mcm, mcd, (n/), n, 1 } donde D_n es el conjunto de divisores de *n*, mcm y mcd son el mínimo común múltiplo y (n/) x = n / x tiene estructura de álgebra de Boole.

A partir de la definición de álgebra de Boole, se deducen los teoremas siguientes:

Teorema 12 (Principio de dualidad): Cada identidad deducida de los postulados del álgebra de Boole permanece válida si se intercambian las operaciones + y ×, y los valores 0 y 1.

De manera informal, este teorema puede demostrarse indicando que, puesto que los postulados son todos simétricos y cumplen la propiedad de dualidad, todo lo que se deduzca de ellos, cumplirá también dicha propiedad.

Teorema 13 (**Dominación** ó Elemento Cero) $a + 1 = 1$ $\forall a \in A$

$a \times 0 = 0$ [Dual]

Demostración:

1
= { inverso + }
$a + \overline{a}$
= {neutro ×, $a/\overline{a}$ }
$a + \overline{a} \times 1$
= { distributiva + }
$(a + \overline{a}) \times (a + 1)$
= { inverso + }
$1 \times (a + 1)$
= { neutro × }
$a + 1$

A partir del teorema anterior se pueden construir las tablas de verdad de las operaciones boolenas + y ×

Teorema 14 (**Idempotencia**) $a + a = a$ $\forall a \in A$

$a \times a = a$ [Dual]

Demostración

a
= *{ neutro + }*
$a + 0$

= *{ inverso × }*
$a + a \times \overline{a}$
= { distributiva + }
$(a + a) \times (a + \overline{a})$
= { inverso + }
$(a + a) \times 1$
= { neutro × }
$a + a$

Teorema 15 (Absorción) $a + a \times b = a$ $\forall a,b \in A$

$a \times (a + b) = a$ [Dual]

Demostración
a
= { neutro × }
$1 \times a$
= { dominación + }
$(1 + b) \times a$
= { distributiva × }
$1 \times a + b \times a$
= { neutro × }
$a + a \times b$

Teorema 16 (Asociativa). $a + (b + c) = (a + b) + c = a + b + c$ $\forall a,b,c \in A$

$a \times (b \times c) = (a \times b) \times c = a \times b \times c$ [Dual]

Demostración: Se demostrarán dos teoremas auxiliares TA1 y TA2:

TA1: $\mathbf{a \times ((a + b) + c) = a \times (a + (b + c))}$
$a \times ((a + b) + c)$
= { distributiva × }
$a \times (a + b) + a \times c$
= { absorción × }
$a + a \times c$
= { absorción + }
a
= { absorción ×, b / b+c }
$a \times (a + (b + c))$

TA2: $\mathbf{\overline{a} \times ((a + b) + c) = \overline{a} \times (a + (b + c))}$
$\overline{a} \times ((a + b) + c)$
= { distributiva × }
$\overline{a} \times (a + b) + \overline{a} \times c$
= { distributiva × }
$(\overline{a} \times a + \overline{a} \times b) + \overline{a} \times c$
= { inverso × }
$(0 + \overline{a} \times b) + \overline{a} \times c$
= { neutro+ }
$\overline{a} \times b + \overline{a} \times c$
= { distributiva × }
$\overline{a} \times (b + c)$
= { neutro + }
$0 + \overline{a} \times (b + c)$
= { inverso × }
$\overline{a} \times a + \overline{a} \times (b + c)$
= { distributiva × }
$\overline{a} \times (a + (b + c))$

A partir de dichos teoremas auxiliares, la demostración se obitiene fácilmente:

$$
\begin{aligned}
& (a + b) + c \\
= \quad & \{\text{neutro } \times\} \\
& ((a + b) + c) \times 1 \\
= \quad & \{\text{ inverso } + \} \\
& ((a + b) + c) \times (a + \bar{a}) \\
= \quad & \{\text{ distributiva } \times\text{, conmutativa }\} \\
& a \times ((a + b) + c) + \bar{a} \times ((a + b) + c) \\
= \quad & \{\text{ TA1 }\} \\
& a \times (a + (b + c)) + \bar{a} \times ((a + b) + c) \\
= \quad & \{\text{ TA2 }\} \\
& a \times (a + (b + c)) + \bar{a} \times (a + (b + c)) \\
= \quad & \{\text{ distributiva } \times \} \\
& (a + \bar{a}) \times (a + (b + c)) \\
= \quad & \{\text{ inverso } + \} \\
& 1 \times (a + (b + c)) \\
= \quad & \{\text{ neutro } \times \} \\
& a + (b + c)
\end{aligned}
$$

Teorema 17. (Unicidad del complementario) El elemento $\bar{a}$ asociado a un elemento *a* en un álgebra de Boole es único, es decir, existe un único elemento, *x*, que cumple la propiedad de elemento inverso, es decir, que cumpla que : $a + x = 1$ y $a \times x = 0$

Demostración:

Supóngase que existen dos elementos *x* e *y* que cumplen la propiedad:

$a + x = 1$ (H1)

$a \times x = 0$ (H2)

$a + y = 1$ (H3)

$a \times y = 0$ (H4)

$$
\begin{aligned}
& x \\
= \quad & \{\text{ neutro } \times \} \\
& 1 \times x \\
= \quad & \{\text{ } H3 \text{ }\} \\
& (a + y) \times x \\
= \quad & \{\text{ distributiva } \times \} \\
& a \times x + y \times x \\
= \quad & \{\text{ H2 }\} \\
& 0 + y \times x \\
= \quad & \{\text{ H4 }\} \\
& a \times y + y \times x \\
= \quad & \{\text{ distributiva } \times\text{, conmutativa } \times \} \\
& (a + x) \times y \\
= \quad & \{\text{ H1 }\} \\
& 1 \times y \\
= \quad & \{\text{ neutro } \times \} \\
& y
\end{aligned}
$$

Concluyendo que *x e* y son el mismo elemento

Teorema 18 (Involución). $\bar{\bar{a}} = a \qquad \forall a \in A$

Demostración: A partir del teorema anterior, cualquier x que cumpla que $\bar{a} + x = 1$ y que $\bar{a} \times x = 0$ es igual a $\bar{\bar{a}}$. Suponiendo que dicho *x* es *a*, se demuestra:

$\bar{a} + a$ $=$ { conmutativa + } $a + \bar{a}$ $=$ { inverso + }	$\bar{a} \times a$ $=$ { conmutativa $\times$ } $a \times \bar{a}$ $=$ { inverso $\times$ }

1	0

Por tanto, $a = \overline{\overline{a}}$

Teorema 19. (De Morgan) $\overline{a+b+c+\ldots} = \overline{a}\times\overline{b}\times\overline{c}\times\ldots$

$\overline{a\times b\times c\times\ldots} = \overline{a}+\overline{b}+\overline{c}+\ldots$ [Dual]

Demostración

Se realizará en dos partes: En la primera parte se demostrará para dos variables, y la segunda parte, se generaliza el resultado para *n* variables.

1.- Demostración para dos variables: $\overline{a+b} = \overline{a}\times\overline{b}$

Por la unicidad del complementario, el único *x* que cumple que $(a + b) + x = 0$ y $(a + b) \times x = 1$ es $\overline{a+b}$.

Si se demuestra que

$a + b + \overline{a}\times\overline{b} = 0$

y que

$(a + b) \times \overline{a}\times\overline{b} = 1$

entonces quedará demostrado

$\overline{a+b} = \overline{a}\times\overline{b}$

Las demostraciones de ambas igualdades son sencillas:

$a + b + \overline{a}\times\overline{b}$

= { Distributiva + }

$(a + b + \overline{a}) \times (a + b + \overline{b})$

= { Conmutativa +, inverso + }

$(1 + b) \times (1 + a)$

= { dominación + }

1×1

= { neutro × }

1

$(a + b)\times \overline{a}\times\overline{b}$

= { distributiva × }

$a\times\overline{a}\times\overline{b} + b\times\overline{a}\times\overline{b}$

= { conmutativa ×, inverso × }

$0 \times \overline{b} + 0 \times \overline{a}$

= { dominación × }

$0 + 0$

= { neutro + }

0

La demostración para *n* variables se realizaría de la siguiente forma:

$\overline{a+b+c+\ldots}$

= { Sea $p = b + c + \ldots$ }

$\overline{a+p}$

= { De Morgan (2 variables) }

$\overline{a} \times \overline{p}$

= { Deshaciendo }

$\overline{a} \times \overline{b+c+\ldots}$

= { repitiendo el proceso anterior hasta sacar todas las variables }

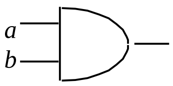
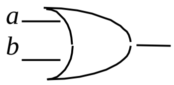
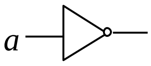
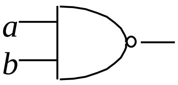
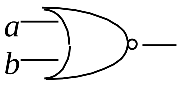
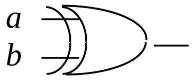
$\overline{a}\times\overline{b}\times\overline{c}\times\ldots$

6.3. Puertas Lógicas

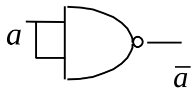
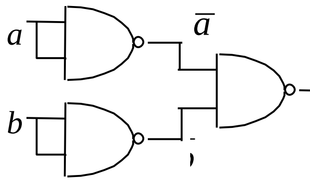
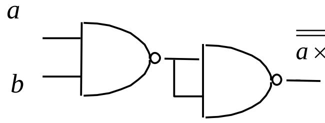
Un *circuito digital* es un circuito electrónico cuyas entradas y salidas sólo pueden tomar dos niveles distintos de tensión. Desde el punto de vista del diseño, estos niveles de tensión se representan como 1 (verdadero) ó 0 (falso). Un *circuito combinacional* se caracteriza por ser un sistema sin *memoria:* el valor de las salidas en cada instante depende sólo del valor de las entradas en ese momento.

Un circuito de estas características puede representarse analíticamente, mediante una *función booleana*, o gráficamente, mediante un *diagrama de puertas lógicas*. En estos diagramas se representan las entradas, las salidas, las operaciones o puertas lógicas y sus conexiones.

Las diferentes conectivas pueden representarse mediante las siguientes puertas lógicas.

Puerta AND  $a\times b$	Puerta OR  $a+b$	Puerta NO (Inversor)  $\overline{a}$
Puerta NAND  $\overline{a\times b}$	Puerta NOR  $\overline{a+b}$	Puerta XOR (O-Exclusiva)  $a\oplus b$

Mediante la utilización de puertas NAND ó NOR pueden implementarse el resto de operaciones. A modo de ejemplo, se muestra cómo se implementa mediante puertas NAND las operaciones $\overline{a}, a+b, a\times b$

$\overline{a}$ = { idempotencia $\times$} $\overline{a\times a}$	 $\overline{a} = \overline{a\times a}$
$a + b$ = { involución } $\overline{\overline{a+b}}$ = { De Morgan + } $\overline{\overline{a}\times\overline{b}}$	 $\overline{a}$ $\overline{\overline{a}\times\overline{b}}$
$a \times b$ = { involución } $\overline{\overline{a\times b}}$	 $\overline{\overline{a\times b}}$

6.4. Funciones Booleanas

Definición 25. Una **variable booleana** es una variable que toma únicamente dos valores 0 ó 1.

Definición 26. Una **función Booleana** es una expresión algebraica que relaciona **variables Booleanas** por medio de las operaciones +, ×, y –.

Ejemplo. 32: $f(a,b,c) = (\overline{a+b})\times c + \overline{a}\times(b+c)$

6.4.1. Formas Canónicas

Definición 27. Un **término canónico** de una función booleana f es una expresión formada por el producto (o la suma) de todas las variables de f en su forma directa o inversa. Cuando el término canónico es un producto, se conoce como **MINTERM** o **producto canónico**. Cuando es una suma, se conoce como **MAXTERM** o **suma canónica**.

Ejemplo.**33** : $ab\overline{c}$ es un producto canónico de *f(a,b,c)*, mientras que $\overline{a}+\overline{b}+\overline{c}+d$ es una suma canónica de la función *f(a,b,c,d)*

Una función de *n* variables tiene a lo sumo 2^n sumas canónicas y 2^n productos canónicos distintos.

Para representar los términos canónicos de una función se utiliza el siguiente convenio: se asigna un 1 a las variables en forma directa y un 0 a las variables en forma inversa. Cada término se representa utilizando el valor decimal de la combinación binaria resultante.

Ejemplo. 34 $\overline{a}bc\overline{d} \equiv 0110_{2|} \equiv 6_{10|}$

$$a+\overline{b}+\overline{c}+d \equiv 1001_{2|} \equiv 9_{10|}$$

Definición 28. Una función está en **forma canónica** si es una suma de productos **canónicos** o un producto de sumas **canónicas**.

Cuando la función es una suma de productos canónicos, se utiliza el símbolo ∑, mientras que para un producto de sumas canónicas se utiliza el símbolo ∏.

Ejemplo. 35 : A continuación se presentan dos funciones en forma canónica:

$$f(a,b,c) = \overline{a}b\overline{c} + ab\overline{c} + \overline{a}\,\overline{b}\,\overline{c} = \sum_{3}(2,6,0)$$

$$g(a,b,c) = (\overline{a}+\overline{b}+\overline{c})(a+\overline{b}+c)(a+b+c) = \prod_{3}(0,5,7)$$

Teorema 20. Toda función Booleana puede expresarse como:

$$f(a,b,c,\ldots) = a \times f(1,b,c,\ldots) + \overline{a} \times f(0,b,c,\ldots)$$
$$f(a,b,c,\ldots) = \big(a + f(0,b,c,\ldots)\big) \times \big((\overline{a} + f(1,b,c,\ldots)\big)$$

Demostración:

Puesto que una función booleana trabaja únicamente con variables Booleanas y estas variables sólo pueden tomar los valores 0 ó 1, es suficiente demostrar la igualdad para $a = 0$ y luego para $a = 1$

1.- Sea $a = 0$

$a \times f(1,b,c...) + \overline{a} \times f(0,b,c...)$

= $\{ a = 0,\ \overline{a} = 1 \}$

$0 \times f(1,b,c...) + 1 \times f(0,b,c...)$

= { *neutro* ×, *dominación* × }

$0 + f(0,b,c...)$

= { *neutro* + }

$f(0,b,c...)$

= $\{ a = 0 \}$

$f(a,b,c...)$

2.- Sea $a = 1$

$a \times f(1,b,c...) + \overline{a} \times f(0,b,c...)$

= $\{ a = 1,\ \overline{a} = 0 \}$

$1 \times f(1,b,c...) + 0 \times f(0,b,c...)$

= { *neutro* ×, *dominación* × }

$f(1,b,c...) + 0$

= { *neutro* + }

$f(1,b,c...)$

= $\{ a = 1 \}$

$f(a,b,c...)$

La otra igualdad se demuestra por dualidad

Multiplicando (o sumando) las expresiones anteriores por *a (ó* $\bar{a}$) se obtienen las siguientes igualdades:

$a \times f(a,b,c...) = a \times f(1, b, c...)$

$\bar{a} \times f(a,b,c...) = \bar{a} \times f(0, b, c...)$

$a + f(a,b,c...) = a + f(0, b,c...)$

$\bar{a} + f(a,b,c...) = \bar{a} + f(0, b,c...)$

Las anteriores igualdades tienen una aplicación importante para la simplificación de funciones

Ejemplo 36: La $f(a,b,c,d)= abc + \bar{a} (a \times \bar{b} + \bar{a} \times c + a \times b \times \bar{c})$ puede simplificarse, utilizando la segunda igualdad, resultando en:

$abc + \bar{a} (a \times \bar{b} + \bar{a} \times c + a \times b \times \bar{c})$

= $\{ \bar{a} f(a,b,c...) = \bar{a} f(0,b,c....) \}$

$abc + \bar{a} (0 \times b + \bar{0} \times c + 0 \times b \times \bar{c})$

= { neutro ×, $\bar{0}=1$ }

$abc + \bar{a} (0 + 1\times c + 0)$

= { neutro +, neutro × }

$abc + \bar{a} c$

Transformación en forma canónica

Teorema 21. Toda función lógica puede transformarse en una función equivalente en forma canónica.

Demostración.

$f(a,b,c...)$

= { Teorema **20**, sacando *a*}

$a \times f(1,b,c...) + \bar{a} \times f(0,b,c...)$

= { Teorema **20**, sacando *b* }

$a \times (b \times f(1,1,c...) + \bar{b} \times f(1,0,c...)) + \bar{a} \times (b \times f(0,1,c...) + \bar{b} \times f(0,0,c...))$

= { Distributiva × }

$a \times b \times f(1,1,c...) +a \times \bar{b} \times f(1,0,c...) + \bar{a} \times b \times f(0,1,c...) + \bar{a} \times \bar{b} \times f(0,0,c...)$

= ...{ repitiendo el proceso con el resto de variables }

$a\times b\times c\times f(1,1,1...) + a\times b\times \bar{c} \times f(1,1,0...) +$

$a\times \bar{b} \times c\times f(1,0,1...) + a\times \bar{b} \times \bar{c} \times f(1,0,0...) +$

$\bar{a} \times b\times c\times f(0,1,1...) + \bar{a} \times b\times \bar{c} \times f(0,1,0...) +$

$\bar{a} \times \bar{b} \times c\times f(0,0,1...) + \bar{a} \times \bar{b} \times \bar{c} \times f(0,0,0...)$

Las expresiones *f(...)* toman valores 0 ó 1 dependiendo de la función particular. Cuando toman valor 1, el término canónico correspondiente permanece, mientras que si toman valor cero, el término desaparece. Con lo cual, cualquier función puede expresarse en forma canónica.

La expresión de producto de sumas, dual de la anterior, sería:

$f(a,b,c...) =$

$(a + b + c + f(0,0,0...)) \times (a + b + \bar{c} + f(1,1,0...)) \times$

$(a + \bar{b} + c + f(1,0,1...)) \times (a + \bar{b} + \bar{c} + f(1,0,0...)) \times$

$$(\bar{a} + b + c + f(0,1,1...)) \times \quad (\bar{a} + b + \bar{c} + f(0,1,0...)) \times$$
$$(\bar{a} + \bar{b} + c + f(0,0,1...)) \times \quad (\bar{a} + \bar{b} + \bar{c} + f(0,0,0...))$$

Obsérvese que en producto de sumas, los términos canónicos permanecen cuando la función toma valor cero y desaparecen cuando toma valor 1. Además, los términos no corresponden de forma directa, sino que cuando la variable está en forma directa, le corresponde un 0 y cuando está en forma inversa, un 1.

El teorema anterior ofrece un método para obtener la expresión canónica de una función a partir de la tabla de verdad

Suma de productos: Toman términos en los que la función vale 1 numerando de arriba abajo

Producto de Sumas: Tomar términos en los que la función vale 0 numerando de abajo a arriba

Ejemplo 37. A partir de la siguiente tabla de verdad, expresar en forma de suma de productos y producto de sumas:

	a	b	c	f(a,b,c)	
0	0	0	0	0	7
1	0	0	1	0	6
2	0	1	0	1	5
3	0	1	1	0	4
4	1	0	0	0	3
5	1	0	1	1	3
6	1	1	0	1	1
7	1	1	1	1	0

$$f(a,b,c)=\sum_3(2,3,5,6,7)= \bar{a}b\bar{c}+a\bar{b}c+ab\bar{c}+abc$$

$$f(a,b,c)=\prod_3(3,4,6,7)= (a+b+c)(a+b+\bar{c})(a+\bar{b}+\bar{c})(\bar{a}+b+c)$$

En ocasiones, desea obtenerse la expresión canónica de una función definida mediante una expresión algebraica. Para ello, se transforma en suma de productos (o producto de sumas) y se multiplica (o suma) cada término no canónico por la suma (o producto) de las variables que faltan y sus inversas.

Ejemplo 38: Se desea obtener la expresión canónica de $f(a,b,c)=a\,(\bar{b} + c) + c$

$a \times (\bar{b} + c) + c$

= { distributiva × }

$a \times \bar{b} + a \times c + c$

= { neutro × , inverso + }

$a \times \bar{b} \times (c + \bar{c}) + a \times c \times (b + \bar{b}) + c \times (a + \bar{a})\,(b + \bar{b})$

= { distributiva × , conmutativa ×}

$a\times\bar{b}\times c + a\times\bar{b}\times\bar{c} + a\times b\times c + a\times\bar{b}\times c + a\times b\times c + \bar{a}\times b\times c + a\times\bar{b}\times c + \bar{a}\times\bar{b}\times c$

= { idempotencia +, conmutativa + }

$\bar{a}\times\bar{b}\times c + \bar{a}\times b\times c + a\times\bar{b}\times\bar{c} + a\times\bar{b}\times c + a\times b\times c$

= { convenio }

$\sum_3(1,3,4,5,7)$

En producto de sumas, el procedimiento sería:

$a \times (\bar{b} + c) + c$

= { distributiva + }

$(a + c) \times (\overline{b} + c + c)$

= { idempotencia }

$(a + c) \times (\overline{b} + c)$

= { neutro +, inverso × }

$(a + c + b\times\overline{b}) \times (\overline{b} + c + a \times \overline{a})$

= { distributiva +, conmutativa + }

$(a + b + c)\times (a + \overline{b} + c)\times (a + \overline{b} + c)\times(\overline{a} + \overline{b} + c)$

= { idempotencia ×, conmutativa }

$(\overline{a} + \overline{b} + c) \times (a + \overline{b} + c) \times (a + b + c)$

= { convenio }

$\prod_3(1,5,7)$

6.4.2. Simplificación de funciones lógicas

En el diseño de circuitos, se utilizan los siguientes pasos:

- Especificación del circuito
- Expresión analítica (normalmente, en forma de Tabla de verdad)
- Simplificación.
- Implementación del diseño.

En estos apuntes se tratan los tres primeros pasos, dejando las técnicas de implementación a otros libros especializados. En cuanto a la simplificación, existen diversos métodos dependiendo del tipo de puertas lógicas disponibles. Uno de los métodos de simplificación más comunes consiste en la aplicación reiterada del siguiente teorema de simplificación

Teorema 22 (Simplificación) $a \times b\times c\times ... + \overline{a} \times b\times c\times ... = b\times c\times ...$

$(a + b + c + ...) \times (\overline{a} + b + c + ...) = b + c + ...$ [Dual]

Demostración.

$a \times b\times c\times ... + \overline{a} \times b\times c\times ...$

= { distributiva × }

$(a + \overline{a}) \times b\times c\times ...$

= { inverso + }

$1 \times b\times c\times ...$

= { neutro × }

$b\times c\times ...$

Definición 29: Dos términos son adyacentes si sólo difieren en el valor de una variable. Su representación binaria sólo diferirá en un bit.

A partir del teorema de simplificación, dos términos adyacentes pueden agruparse eliminando la variable respecto a la que difieren y quedando únicamente la parte común.

Los principales métodos de simplificación reiteran este proceso de agrupamiento hasta obtener una expresión que ya no contenga más términos adyacentes.

Ejemplo 39. $f = \underbrace{\underbrace{a\bar{b}cd + \bar{a}\bar{b}cd}_{\bar{b}cd} + \underbrace{a\bar{b}\bar{c}d + \bar{a}\bar{b}\bar{c}d}_{\bar{b}\bar{c}d}}_{\bar{b}d} \Rightarrow f = \bar{b}d$

Utilizando las representaciones binarias de los términos canónicos, los agrupaciones anteriores serían:

Grupos			
	11 - 1011 3 - 0011	$\}$ 3-11 - X011	
	9 - 1001 1 - 0001	$\}$ 1-9 - X001	1-3-9-11 - X0X1 = $\bar{b}d$

Uno de los métodos más utilizados para funciones de menos de 6 variables es el método de Karnaugh que permite visualizar los términos adyacentes en una cuadrícula. Sin embargo, para funciones de un mayor número de variables, el método algebraico más general es el método de Quine-McCluskey.

Método de Karnaugh

Es un método gráfico que visualiza los términos adyacentes en cuadrículas intentando que dos términos adyacentes estén próximos entre sí. Los pasos del método son:

1. Construir una tabla de 2^n casillas, siendo n el número de variables de la función. Cada casilla corresponderá a un término canónico, producto o suma. Cada casilla se etiqueta con el término asociado intentando que los términos adyacentes estén próximos entre sí. Cuando esto no es posible (funciones de más de tres variables) se disponen los términos en los límites de la tabla y se supone que los términos de un límite son adyacentes a los del límite opuesto. A continuación aparecen las tablas para funciones entre 2 y 5 variables con el número decimal asociado a los términos canónicos.

2 variables, *f(a,b)*

a \ b	0	1
0	0	1
1	2	3

3 Variables, *f(a,b,c)*

a \ bc	00	01	11	10
0	0	1	3	2
1	4	5	7	6

4 Variables, *f(a,b,c,d)*

ab \ cd	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

5 Variables, *f(a,b,c,d,e)*

bc \ de	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

a = 0

bc \ de	00	01	11	10
00	16	17	19	18
01	20	21	23	22
11	28	29	31	30
10	24	25	27	26

a = 1

Son cuadros *adyacentes*:

- Los que tienen un lado común.
- Los de la fila superior con los respectivos de la fila inferior.
- Los de la columna de la izquierda con los respectivos de la columna derecha.
- Los que están en la misma posición en ambas tablas (5 variables)

Ejemplo 40 Los términos adyacentes al 10 son 8,11, 14, 2, 26.

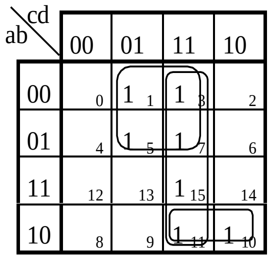
Los términos adyacentes al 13 son: 12, 5, 15, 9, 29.

2. En el caso de la expresión canónica en forma de suma de productos, las casillas asociadas a términos que aparecen en ella se completan con 1 y el resto se dejan en blanco. O lo que es lo mismo, aquellas casillas etiquetadas con una entrada de la tabla de verdad para la cual la función toma el valor 1 se cubren con unos y el resto se deja vacío. Si la expresión está dada en forma de productos de sumas las casillas se completan con 0 en vez de 1. Esta tabla es lo que se conoce como **mapa de Karnaugh de la función.**

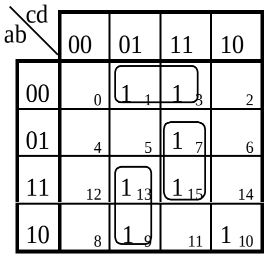
3. **Proceso de simplificación**. Se realizan agrupamientos reiterados de términos adyacentes hasta que todos los términos hayan sido agrupados. Los grupos deben contener el mayor número posible de términos (el número de términos en cada grupo será siempre potencia de 2).

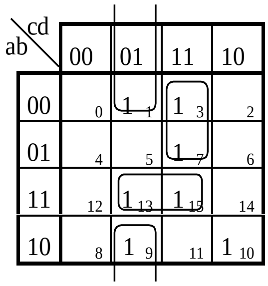
4. **Construir la expresión reducida**. Por cada agrupamiento se obtiene una expresión formada por las variables comunes a los términos adyacentes.

Nota 1: Para construir grupos mayores pueden utilizarse casillas que ya han sido previamente agrupadas. Ejemplo:



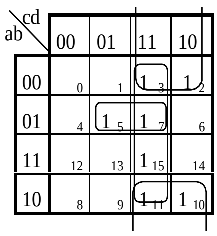
Nota 2: Algunas funciones pueden agruparse de varias formas. En dichos casos, existe más de una solución. Ejemplo:





Ejemplo 41. Simplificar la función *f(a,b,c,d)*= $\sum_4$(2,3,5,7,10,11,15)

El cuadro de Karnaugh correspondiente es:



Grupos

3 - 0011
7 - 0111 → 3-7 - 0X11
15 - 1111
11 - 1011 → 15-11 - 1X11
→ 3-7-15-11 - XX11 = *cd*

2 - 0010
3 - 0011 → 2-3 - 001X
11 - 1011
10 - 1010 → 10-11 - 101X
→ 2-3-10-11 - X01X = $\bar{b}c$

5 - 0101
7 - 0111 → 5-7 - 01X1 = $\bar{a}bd$

El resultado es: $f(a,b,c,d) = \bar{a}bd + \bar{b}c + cd$

Para simplificar en forma de producto de sumas, es necesario transformar la forma canónica a producto de sumas. Para ello, puede construirse la tabla de verdad y tomar los elementos de valor cero numerando de abajo a arriba. El resultado sería: *f(a,b,c,d)*= $\sum_4$(1,2,3,6,7,9,11,14,15)

ab \ cd	00	01	11	10
00	$_{0}$	0_{1}	0_{3}	0_{2}
01	$_{4}$	$_{5}$	0_{7}	0_{6}
11	$_{12}$	$_{13}$	0_{15}	0_{14}
10	$_{8}$	0_{9}	0_{11}	$_{10}$

Grupos

1 - 0001, 3 - 0011 → 1-3 - 00X1
9 - 1001, 11 - 1011 → 9-11 - 10X1
→ 1-3-9-11 - X0X1 = $\bar{b}+d$

2 - 0010, 3 - 0011 → 2-3 - 001X
6 - 0110, 7 - 0111 → 6-7 - 011X
→ 2-3-6-7 - 0X1X = $a+c$

6 - 0110, 7 - 0111 → 6-7 - 011X
14 - 1110, 15 - 1111 → 14-15 - 111X
→ 6-7-14-15 - X11X = $b+c$

Resultado: $f(a,b,c,d) = (\bar{a}+c)\times(b+c)\times(\bar{b}+d)$

Funciones incompletas

Existen funciones que no están totalmente definidas, llamadas **funciones incompletas**. Las razones son:

1. Combinaciones de entrada imposibles, pudiendo ser su valor de salida 1 ó 0 indistintamente.

2. Con salidas inhibidas: el valor de salida es indiferente.

En la forma canónica de la función se representan de forma separada los términos indefinidos del resto. Los términos indefinidos se agrupan mediante el símbolo $\varnothing$.

Ejemplo 42. Sea *f(a,b,c,d)* = $\sum_4(1,3,10,11)+\sum_\varnothing(0,2,4,13)$

En la minimización de funciones incompletas los valores indefinidos se representan mediante X y actúan como *comodines*. No es necesario agruparlos, pero, si al agruparlos se obtiene un grupo mayor, se pueden agrupar.

ab \ cd	00	01	11	10
00	X_{0}	1_{1}	1_{3}	X_{2}
01	X_{4}	$_{5}$	$_{7}$	$_{6}$
11	$_{12}$	X_{13}	$_{15}$	$_{14}$
10	$_{8}$	$_{9}$	1_{11}	1_{10}

Grupos

0 - 0000, 1 - 0001 → 0-1 - 000X
2 - 0010, 3 - 0011 → 2-3 - 001X
→ 0-1-2-3 - 00XX = $\bar{a}\bar{b}$

2 - 0010, 3 - 0011 → 2-3 - 001X
11 - 1011, 10 - 1010 → 10-11 - 101X
→ 2-3-10-11 - X01X = $\bar{b}c$

Resultado: $f(a,b,c,d) = \bar{b}c + \bar{a}\bar{b}$

7. Ejercicios

Los siguientes ejercicios se han recopilado de diversas fuentes. Una de las fuentes fundamentales, ha sido el "Boletín de Ejercicios" ofrecido en la asignatura "Lógica Informática" de la Escuela Universitaria de Ingeniería Técnica Informática de Gijón.

[1] Formalizar las siguientes expresiones:

a) q si p
b) p pero q
c) como mínimo p
d) p no obstante q
e) q necesario para p
f) q suficiente para p
g) p a pesar de q
h) no p a menos que q
i) p sólo si q
j) p sin embargo q
k) p suficiente para q
l) p siempre que q
m) a veces p, siempre q
n) p a no ser que q

[2] Formalizar los razonamientos:

" Si el resultado obtenido es superior al previsto en 5 unidades, será debido a no haber realizado el proceso a la temperatura adecuada o a la existencia de errores en los cálculos finales."

" El análisis realizado, innecesario si nos dejamos llevar por la precipitación, se torna necesario si nos paramos a reflexionar sobre el mensaje que se pretende transmitir."

" El cáncer no logrará curarse a no ser que se logre determinar su causa y se consiga encontrar fármacos adecuados o bien para prevenirlo o para curarlo."

[3] Simplificar mediante Karnaugh las siguientes funciones lógicas:

a) $x + x\bar{y} + \bar{y}$

b) $xy + x\bar{y}z + y(\bar{x} + z) + \bar{y}\,\bar{z}$

c) $wx + xy + yz + zw + \bar{w}\,\bar{x}y\bar{z} + \bar{w}\,\bar{x}\,\bar{y}z$

d) $wx\bar{y}\,\bar{z} + wx\bar{y}z + wxyz + w\bar{x}yz + \bar{w}\,\bar{x}yz + \bar{w}\,\bar{x}y\bar{z} + \bar{w}\,\bar{x}\,\bar{y}\,\bar{z} + \bar{w}xy\bar{z} + \bar{w}x\bar{y}\,\bar{z} + \bar{w}x\bar{y}z$

e) $vw(x + y + x\bar{z}) + \bar{v}\,\bar{x}z(w\bar{y} + \bar{x}(\bar{z} + \bar{v}y))$

[4] Demostrar las siguientes propiedades de la función lógica O-exclusiva:

a) Asociativa

b) Conmutativa

c) Existencia de elemento neutro *e* tal que $x \oplus e = x$

d) Existencia de Inverso (A todo elemento *x* se le puede hacer corresponder un elemento $\hat{x}$ tal que $x \oplus \hat{x} = e$

e) Distributiva del Producto respecto a la O-exclusiva: $x(y \oplus z) = xy \oplus xz$

f) que mediante la O-exclusiva y la función AND se pueden realizar las otras dos operaciones fundamentales del álgebra de Boole: negación y suma (OR).

Nota: Calcular el valor de $1 \oplus x$ y de $1 \oplus \big((1 \oplus x)(1 \oplus y)\big)$

g) que $x \oplus y = \bar{x} \oplus \bar{y}$

[5] Una función de tres variables *f(a,b,c)* debe tomar el valor cero cuando la variable *b* esté a uno y la variable *a* no está en estado uno. En los demás casos posibles debe estar en estado uno.

a) Realizar la tabla de verdad de la función.

b) Obtener las formas canónicas en forma de suma de productos y producto de sumas.

c) Minimizar dichas expresiones.

[6] Obtener la expresión algebraica mínima de una función lógica de cuatro variables que toma el valor uno cuando el número de variables que están en estado uno es superior al de las que se encuentran en estado cero. Nunca pueden estar más de tres variables en estado uno. Realizar la expresión obtenida con puertas NOR y NAND.

[7] Formalizar utilizando las conectivas $\{\neg,\vee,\wedge\}$ el siguiente enunciado:

" Si p entonces q y, en caso contrario, si p1 entonces q1 y, en caso contrario r1."

[8] Formalizar el siguiente enunciado donde hemos notado "si p entonces q y en caso contrario r" de la siguiente forma:

$$p \to q; r$$

- Normalizar dicho enunciado.
- Estudiar la validez de la siguiente fórmula en la que se emplea la notación anterior:

$$\Big(\big(p \to (p \vee r);(r \vee s)\big) \wedge \Big(\big(r \to \neg w;(s \to p)\big) \wedge \big(\neg(q \wedge t) \wedge (q \to t)\big)\Big)\Big) \to \neg(r \to w)$$

[9] Determinar la validez de las siguientes fórmulas:

a) $\big((\neg p \wedge q) \vee (p \vee \neg q)\big) \to \big(\big(p \to (q \vee r)\big) \to (p \to r)\big)$

b) $\big(p \wedge (q \to p)\big) \to p$

c) $(\neg p \to q) \to \big((p \wedge \neg q) \vee r\big)$

[10] Obtener la FND de:

a) $\big(p \to (p \to q)\big) \vee p \vee q$

b) $\neg(p \vee \neg q) \wedge (\neg r \vee s)$

[11] Obtener la FNC de:

a) $\neg(p \to q) \vee p \vee q$

b) $\big((p \to q) \to r\big) \to s$

[12] Probar, usando el algoritmo de resolución, la validez de los siguientes razonamientos:

a) $\{p \to q, q \to r, p\} \Rightarrow r$

b) $\{p \to (\neg q \vee (r \wedge s)), p, \neg s\} \Rightarrow \neg q$

[13] Compruébese si los siguientes razonamientos son correctos o no:

a.-*" Si Antonio ganó la carrera, entonces Baltasar o Carlos fueron los segundos. Si Baltasar fue segundo, entonces no ganó Antonio. Si Demetrio fue segundo, no lo fue Carlos. Antonio ganó la carrera. Por tanto, Demetrio no fue segundo".*

b.-*" No llora, ríe. Si no llora, ríe sólo si tiene un juguete. Nunca tiene un juguete cuando se está riendo si no come un caramelo. Luego come un caramelo."*

c.-*" Juan quiere a María si y sólo si María quiere a Juan y promete casarse con él. María no quiere a Juan si Juan no quiere a María. María promete casarse con Juan si y sólo si Juan promete casarse con María. Por tanto, Juan quiere a María y María no quiere a Juan".*

d.-*" Si ha nevado será difícil conducir. Si no es fácil conducir llegaré tarde si no salgo temprano. Ha nevado. Luego saldré temprano. "*

e.-*" Si no llueve salgo al campo. Si salgo al campo respiro. Por tanto, respiro si y sólo si no llueve."*

f.-*" Si un monte se quema algo tuyo se quema. Algo tuyo se quema si y sólo si eres descuidado. Si eres descuidado no mereces que te feliciten. Por tanto si no mereces que te feliciten entonces es que un monte se quema."*

g.-*" El Ministro de Economía y Hacienda ha hecho las siguientes declaraciones:*

A la prensa: " Si los impuestos suben, la inflacción bajará si y sólo si no se devalúa la peseta."

A la radio: " Si la inflacción baja o si la peseta no se devalúa, los impuestos no subirán."

A la tele: " O bien baja la inflacción y se devalúa la peseta, o bien los impuestos deben subir."

Como consecuencia, publica un informe en el que asegura: "Los impuestos deben subir, pero la inflación bajará y la peseta no se devaluará."

¿ Fue consecuente con sus declaraciones a los medios de comunicación?."

h.-*" Es suficiente whisky para que chocolate. Chocolate si y solo si jamón. No ginebra a menos que chocolate. Whisky*. ¿ Es posible afirmar: (1) que bebió ginebra? (2) que no tomó chocolate?"

i.-*" Si no especifico las condiciones iniciales mi programa no comenzará. habré programado un ciclo infinito solo si mi programa no termina. Basta que el programa no comience o no finalice para que falle. De ahí que sea necesario no solamente especificar las condiciones iniciales sino también no programar un ciclo infinito para que el programa no falle."*

j.-*"Si 25 divisiones son suficientes, el general ganará la batalla; por otra parte, o se suministran 3 alas de apoyo aéreo táctico, o el general no ganará la batalla. Además, no es cierto que sean suficientes 25 divisiones y que se vayan a suministrar 3 alas de apoyo aéreo táctico. Coclusión: no son suficientes 25 divisiones."*

[14].-Le digo a un amigo:

Cuando salgo sin paraguas, llueve.

Cuando está despejado, no llueve.

Según el hombre del tiempo, mañana estará despejado o hará niebla.

De todos modos saldré sin paraguas.

Entonces mi amigo responde: Entonces mañana, además de llover, habrá niebla. ¿Cómo lo supo?

[15] Don Juan Tenorio, hizo las siguientes declaraciones, con respecto a las doncellas Inés, Juana y María, que le costaron la vida. ¿Quién o quiénes son las asesinas?.

" Amo a la última de las tres"

" Si amo a Inés pero no a María, entonces también amo a Juana"

" O amo a María y a Juana o no amo a ninguna"

" Si amo a María, entonces amo a Inés"

(se supone que la asesina era aquélla a la que Don Juan no amaba)

[16].-En un juicio el fiscal argumenta:

" Si el acusado es culpable, entonces tenía un testigo".

A ello, el abogado defensor respondió inmediatamente:

" Eso es falso".

El acusado decidió cambiar de abogado defensor. ¿ Es lógico?.

[17].-Analizar la coherencia lógica - no teológica - del siguiente razonamiento:

- Si Dios existe es todo amor y omnipotencia.

- Si Dios es incapaz de erradicar el sufrimiento del mundo entonces no es omnipotente.

- Dios no es amor o está dispuesto a erradicar el sufrimiento del mundo.

- Dios es capaz de erradicar el sufrimiento del mundo y está dispuesto a ello solo si no existe sufrimiento en el mundo.

- Existe sufrimiento en el mundo.

Por tanto:

- Dios no existe.

[18] *Si la Bella Durmiente despierta, los habitantes del castillo también lo harán. Si el príncipe la besa, despertará. El príncipe la besará si está de buen ver. O la besa o no se despierta nadie. Como esto es un cuento, la princesa, a pesar de llevar 100 años dormida sigue de muy buen ver. Si la princesa se despierta se casarán, vivirán felices y comerán perdices si no estamos en veda. Estamos en veda.*

¿ Se casan? ¿ Son felices? ¿ Comen perdices?.

[19] Los dos carteles siguientes están colgados respectivamente a la puerta de las habitaciones 1 y

2. Uno dice la verdad y otro miente. Sabiendo que en la misma habitación no puede haber una dama y un tigre y que puede haber dos damas y dos tigres, se pide decidir lógicamente qué puerta se debe abrir para liberar a la dama si es que existe. Ambas habitaciones están ocupadas.

CARTEL 1
EN ESTA HABITACION HAY UNA DAMA Y EN LA OTRA UN TIGRE.
CARTEL 2
EN UNA DE ESTAS HABITACIONES HAY UNA DAMA Y EN UNA DE ESTAS HABITACIONES HAY UN TIGRE.

- Suponiendo que los dos carteles siguientes dicen ambos la verdad o mienten ambos. Deducir en qué habitación hay una dama, sabiendo, como antes, que puede no haberla.

CARTEL 1
AL MENOS EN UNA DE ESTAS HABITACIONES HAY UNA DAMA
CARTEL 2
HAY UN TIGRE EN LA OTRA HABITACION

[20] Discurso sobre los estudios de Informática en clase de Lógica:

Señoras, señores, buenas tardes:

Es hora de que recapacitemos sobre los estudios de informática en vísperas del asentamiento de la titulación en nuestra Universidad. Se sabe que si los ordenadores hablasen los informáticos no existirían. Por otra parte, en la última reunión del Consejo de Universidades, éste afirmó que: "...la Universidad titulará informáticos mientras los ordenadores no hablen ..."; afirmación que nos parece muy correcta, si bien lo cierto es que los ordenadores no hablan pero los informáticos existen.

A la vista de todo ello nos preguntamos: ¿Es, por tanto, coherente que la Universidad expida títulos de informática en la actualidad?.

8. Soluciones

[1] a) $p \to q$

b) $p \wedge q$

c) p

d) $p \wedge q$

e) $p \to q$

f) $q \to p$

g) $p \wedge q$

h) $p \to q$

i) $p \to q$

j) $p \wedge q$

k) $p \to q$

l) $q \to p$

m) $(p \vee \neg p) \wedge q \equiv q$

n) $\neg q \to p$

[2] p = Resultado obtenido menor al previsto en 5 unidades.

q = Haber realizado el proceso a la temperatura adecuada.

r = Existencia de errores en los cálculos finales.

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

p = Análisis realizado es necesario.

q = Nos dejamos llevar por la precipitación.

r = Nos paramos a reflexionar sobre el mensaje que se pretende transmitir.

$$(q \to \neg p) \wedge (r \to p)$$

p = El cáncer logrará curarse.

q = Se logra determinar su causa.

r = Se consigue encontrar fármacos adecuados para prevenirlo.

s = Se consigue encontrar fármacos adecuados para curarlo.

$$\neg(q \wedge (r \vee s)) \to \neg p \equiv p \to (q \wedge (r \vee s))$$

[3] a) $x + \bar{y}$

b) $x + y + \bar{z}$

c) $xw + \bar{x}z + \bar{w}y$

d) Habría tres posibles soluciones $\begin{cases} x\bar{y} + \bar{w}\,\bar{z} + \bar{x}yz + wyz \\ x\bar{y} + \bar{w}\,\bar{z} + \bar{x}yz + wxz \\ x\bar{y} + \bar{w}\,\bar{z} + \bar{w}\,\bar{x}z + wyz \end{cases}$

e) $\bar{v}\,\bar{x}yz + \bar{v}w\bar{x}z + vwx + vwy$

[4] Para realizar las demostraciones se parte de que $x \oplus y = x\bar{y} + \bar{x}y$, y se desarrollan las expresiones resultantes hasta llegar a la demostración deseada. A continuación se presentan los primeros pasos de la demostración de asociatividad:

$$(x \oplus y) \oplus z = (x \oplus y)\bar{z} + \overline{(x \oplus y)}z = (x\bar{y} + \bar{x}y)\bar{z} + \overline{(x\bar{y} + \bar{x}y)}z = \cdots = x\bar{y}\,\bar{z} + \bar{x}y\bar{z} + \bar{x}\,\bar{y}z + xyz$$

Si desarrollásemos $x \oplus (y \oplus z)$ obtendríamos la misma expresión, con lo cual habríamos demostrado la propiedad asociativa.

Las demás demostraciones se realizarían de forma similar.

[5] La función, minimizada sería $a + \overline{b}$

[6] La expresión mínima sería: $bcd + acd + abd + abc$.

[7] $(p \to q) \wedge (\neg p \to ((p1 \to q1) \wedge (\neg p1 \to r1)))$

[8] $(p \to q) \wedge (\neg p \to r)$

- Forma normal conjuntiva: $(\neg p \vee q) \wedge (p \vee r)$

- El razonamiento NO es válido, pues podemos encontrar varias interpretaciones que lo hagan Falso, por ejemplo, *r*=***F***, *w* = ***F***, *q* = ***F***, *p* = ***V***, *s* = ***V***, *t*= ***V***.

[9] a) NO es válida, ejemplo *p* = ***V***, *q* = ***V***, *r* = ***F***

b) SI es válida.

c) NO es válida, ejemplo *p* = ***V***, *q* = ***V***, *r* = ***F***

[10] a) Al operar quedaría: $\neg p \vee \neg p \vee q \vee p \equiv \mathbf{V}$

b) $(\neg p \wedge q \wedge \neg r) \vee (\neg p \wedge q \wedge s)$

[11] a) $p \vee q$

b) $(\neg p \vee q \vee s) \wedge (\neg r \vee s)$

[12] En ambos casos se llega a la cláusula vacía $\Rightarrow$ Razonamiento correcto.

[13] a) SI es válido, Antonio ganó la carrera.

b) ES válido, come un caramelo.

c) El razonamiento NO es válido, ya que puede darse el caso de que ninguno de los dos quiera al otro, y las premisas serían ciertas, pero la conclusión Falsa.

d) El razonamiento NO es válido porque puede darse el caso de NO salir temprano y llegar tarde habiendo nevado y siendo difícil conducir. Cumpliéndose todas las premisas.

e) NO es válido, puedo salir al campo, lloviendo y respirar. Luego no se deduce que respire si y solo si no llueve.

f) NO es válido.

g) NO es correcto.

h) NO se puede deducir ninguna de los dos, ni que bebiese Ginebra ni que no tomase chocolate.

i) El razonamiento ES correcto.

j) El razonamiento ES correcto.

[14] Ambas expresiones se deducen lógicamente suponiendo que todas las frases sean Verdaderas, se puede comprobar que sólo existe esa posibilidad.

[15] Ninguna de ellas era la asesina, pues las amaba a las tres.

[16] Si es Falsa la sentencia "Si el acusado es culpable entonces tenía un testigo", por la tabla de verdad de la implicación, el antecedente es Verdadero y el consecuente Falso, luego el antecedente es Verdadero, es decir, el acusado sería culpable.

[17] El razonamiento es correcto en términos lógicos.

[18] Se casan, viven felices, pero no comen perdices porque estamos en veda.

[19] - La única forma de que un cartel diga la verdad y otro mienta es que el Cartel 1 mienta y el Cartel 2 diga la verdad. Con lo cual habría un tigre en la habitación 1 y una dama en la habitación 2.

- La única posibilidad es que los dos carteles digan la verdad y habría una dama en la habitación 2 y un tigre en la habitación 1.

[20] Sí, se sigue que la universidad expenda títulos de Informática a partir de las premisas. El razonamiento es correcto.

Bibliografía

[Abramsky, 92] S. Abramsky, D.M. Gabbay, T.S. Maibaum
Handbook of Logic in Computer Science
Oxford Science Publications (1992)

[Ben-Ari, 93] M. Ben-Ari
Mathematical Logic for Computer Science
Prentice Hall Intl. (1993)

[Birkhoff, 70] G. Birkhoff, T. C. Bartee
Modern Applied Algebra
McGraw-Hill (1970)

[Burke, 96] E. Burke, E. Foxley
Logic and its Applications
Prentice Hall Intl. (1996)

[Dijkstra, 91] E. W. Dijkstra, C. S. Scholten
Predicate Calculus and Program Semantics
Springer-Verlag (1990)

[Fitting, 96] M. Fitting
First-Order Logic and Automated Theorem Proving
Springer-Verlag, 2nd Ed. (1996)

[Genesereth, 87] M. R. Genesereth, N.J. Nilsson
Logical foundations of Artificial Intelligence
Morgan Kaufmann Publishers, Inc. (1987)

[Grassmann, 96] W. K. Grassmann, J. Tremblay
Matemática Discreta y Lógica
Prentice-Hall (1996)

[Gries, 94] D. Gries, F. B. Schneider
A Logical Approach to Discrete Math
Springer-Verlag (1994)

[Kelly, 97] J. Kelly
The Essence of Logic
Prentice Hall (1997)

[Maier, 88] David Maier, David S. Warren
Computing with Logic
The Benjamin/Cummings Publishing Company, Inc. (1988)

[Mandado, 88] E. Mandado
Sistemas Electrónicos Digitales
Ed. Marcombo (1988)

[Sperschneider, 91] V. Sperschneider, G. Antoniou
Logic: A foundation for Computer Science
Addison-Wesley Publishing Company (1991)

[Whitesitt, 61] J. E. Whitesitt
Boolean Algebra and its Applications
Addison-Wesley (1961)

Indice