

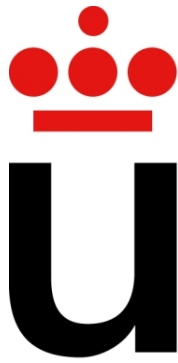
Ciencias Experimentales

 Universidad
Rey Juan Carlos
| Servicio de Publicaciones

Joaquín Arias

Apuntes de Lógica: desde Aristóteles hasta Prolog

ISBN: 978-84-09-52634-5



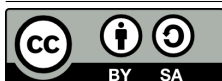
Universidad
Rey Juan Carlos


ESCUELA TÉCNICA SUPERIOR
DE INGENIERÍA INFORMÁTICA

Apuntes de Lógica

DESDE ARISTÓTELES HASTA PROLOG

Autor: Joaquín Arias



Copyright ©2023 Joaquín Arias . Este obra está bajo la licencia CC BY-SA 4.0, [Creative Commons Atribuciónn-CompartirIgual 4.0 Internacional](https://creativecommons.org/licenses/by-sa/4.0/). Como citar esta obra: Arias, Joaquín (2023). Apuntes de Lógica: desde Aristóteles hasta Prolog. Madrid, Servicio Publicaciones de la URJC. ISBN: 978-84-09-52634-5_____

Índice general

Índice general	i
1 Introducción	1
1.1 Por qué es necesaria, tipos e historia.	2
1.1.1 Introducción	2
1.1.2 Lógica	3
1.1.3 Tipos de lógicas	4
1.1.4 Algo de historia	4
1.2 Teoría de conjuntos, relaciones, funciones y álgebra de Boole.	6
1.2.1 Teoría de Conjuntos	6
1.2.2 Relaciones Binarias	10
1.2.3 Relaciones n-arias	12
1.2.4 Funciones n-arias	12
1.2.5 Algebra de Boole	12
2 Lógica Proposicional	15
2.1 Sintaxis, semántica y teoría interpretativa.	16
2.1.1 Lenguaje proposicional	16
2.1.2 Semántica de la lógica proposicional	19
2.1.3 Satisfacibilidad	20
2.1.4 Teoría interpretativa	22
2.2 Sistemas formales y deducción natural.	26
2.2.1 Motivación	26
2.2.2 Sistemas formales	27
2.2.3 Deducción Natural	28
3 Lógica Primer Orden	37
3.1 Sintaxis, semántica y teoría interpretativa.	39
3.1.1 Sintaxis de la lógica de primer orden.	39
3.1.2 Semántica de la lógica de primer orden	47
3.1.3 Satisfacibilidad	51
3.1.4 Teoría interpretativa de la lógica de primer orden.	52
3.2 Teorema de la demostración y deducción natural.	56
3.2.1 Decibilidad de la lógica de Primer Orden.	56

3.2.2	Deducción Natural	57
3.3	Tema 3.3:	59
3.3.1	Objetivo.	59
3.3.2	Forma Normal de Skolem	60
3.3.3	Forma Clausular	62
3.4	Método de resolución de Robinson.	62
3.4.1	Introducción	62
3.4.2	Interpretaciones de Herbrand	63
3.4.3	Teorema de Herbrand	65
3.4.4	Método de Resolución de Robinson	66
3.4.5	Resolución con Unificador de Máxima Generalidad	68
3.4.6	Estrategias de resolución	71
4	Introducción a la Programación Lógica	73
4.1	Tema 4.1:	74
4.1.1	Estrategia de resolución SLD	74
4.1.2	Árbol de derivación	74
4.2	Tema 4.2:	75
4.2.1	Programación Declarativa	75
4.2.2	Programación Funcional	76
4.2.3	Programación Lógica	79
4.2.4	UTD HackReason	82

Agradecimientos

Esta obra, contiene en formato de apuntes las transparencias de Arias, Joaquín (2022). [Lógica: desde Aristóteles hasta Prolog](#). Madrid: Servicio de Publicaciones de la Universidad Rey Juan Carlos. ISBN:978-84-09-38265-1, las cuales están basada en transparencias de Antonio González Pardo (URJC'20) & Pepa Hernández (UPM'11),

C pítulo 1

Introducci n

1.1	Por qu� es necesaria, tipos e historia.	2
1.1.1	Introducci�n	2
	Lenguaje natural	2
	Lenguaje formal	3
1.1.2	L�gica	3
	Sistemas l�gicos	4
1.1.3	Tipos de l�gicas	4
1.1.4	Algo de historia	4
	L�gica y filosof�a	4
	L�gica y matem�ticas	5
	L�gica e inform�tica	6
1.2	Teor�a de conjuntos, relaciones, funciones y �lgebra de Boole.	6
1.2.1	Teor�a de Conjuntos	6
	Representaci�n	6
	Inclusi�n e igualdad	7
	Operaciones	7
	Propiedades de las operaciones	9
1.2.2	Relaciones Binarias	10
	Definiciones	11
	Homog�neas	11
1.2.3	Relaciones n-arias	12
1.2.4	Funciones n-arias	12
1.2.5	Algebra de Boole	12
	Definici�n	12
	Interpretaci�n	13

Propiedades y teoremas	13
Simplificación de ecuaciones	14

1.1. Por qué es necesaria, tipos e historia.

1.1.1. Introducción

- La lógica formal es la ciencia que estudia las leyes de inferencia en los razonamientos.
- Trata de resolver diversos problemas basándose en la formación del lenguaje y sus reglas básicas.
- Se aplica en multitud de áreas:
 - En matemáticas para demostrar teoremas
 - En ciencias de la computación para verificar si son o no correctos los programas
 - En las ciencias física y naturales, para sacar conclusiones de experimentos
 - En las ciencias sociales y en la vida cotidiana, para resolver una multitud de problemas.

Lenguaje natural

- Lenguaje natural: lenguaje usado en la comunicación humana.
- Dentro del lenguaje encontramos sintaxis y semántica.
 - Sintaxis: las reglas de formación de frases correctas.
 - Semántica: las frases deben tener significado completo e independiente.

Análisis sintáctico



- No es válida sintácticamente.

Análisis semántico

- Una vez que una frase es válida sintácticamente, se puede analizar su semántica.
 - Si tiene (o no) sentido desde el punto de vista semántico.



Limitaciones

- Es complicado (o casi imposible) dar una representación completa de las reglas sintácticas de los lenguajes hablados.
- No podemos formular afirmaciones que se definan como correctas (sintácticamente) o verdaderas (semánticamente) sin ambigüedad.

Tenemos que definir un lenguaje más preciso: **Un lenguaje formal.**

Lenguaje formal

- El lenguaje formal se construye a partir de proposiciones
 - Elementos básicos (atómicos)
 - Simples.
 - Se les puede asociar valores de verdad sin ninguna ambigüedad.

1.1.2. Lógica

- El objetivo de la lógica es estudiar la validez formal de un razonamiento.
- **Razonamiento:** la obtención de un nuevo conocimiento (conclusión) a partir de una serie de conocimientos (premisas).

- **Validez formal de un razonamiento:** si la conclusión es verdadera, siendo las premisas verdaderas.

Sistemas lógicos

- La estructura de todo sistema lógico viene definido por:
 - **Sintaxis:** definición axiomática de los elementos básicos del lenguaje y de las reglas que permiten obtener nuevas expresiones correctas.
 - **Semántica:** definición de un conjunto de significados (verdadero o falso) que permiten definir la validez de un razonamiento.
 - **Sistemas de demostración:** sistemas formales que permiten averiguar cuándo un razonamiento es (o no es) válido.

1.1.3. Tipos de lógicas

- **Lógica proposicional:** se estudian las fórmulas proposicionales construidas a partir de fórmulas atómicas y conectivos lógicos.
- **Lógica de predicados de primer orden:** es una generalización de la lógica de proposiciones. Distingue entre los objetos del discurso y sus relaciones entre ellos.

Lógica proposicional

- “Tiene un perro” p .
- “Tiene una mascota” m .
- “Es adorable” a .

- “Si tiene un perro y tiene una mascota, entonces es adorable”

$$p \wedge m \rightarrow a$$

Lógica de primer orden

- “ x es una mascota” $M(x)$.
- “ x es un perro” $P(x)$.
- “ x es adorable” $A(x)$.

- “Todas las mascotas que sean perros, son adorables”

$$\forall x.M(x) \wedge P(x) \rightarrow A(x)$$

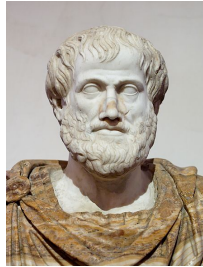
1.1.4. Algo de historia

Lógica y filosofía

- **S. IV a.C.:** Aristóteles formaliza el razonamiento humano. Fundó la lógica clásica o lógica aristotélica. Por Ejemplo:

Todos los hombres son mortales. Sócrates es un hombre. Luego Sócrates es mortal.

- **En el s. XIII:** Santo Tomás de Aquino empleó la lógica en el contexto de discusiones teológicas.
- **s. XVIII:** Leibniz fue el primero en formular la lógica como base del razonamiento matemático.



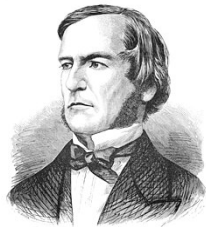
Aristóteles



Leibniz

Lógica y matemáticas

- **En 1854:** George Boole definió la lógica como sistemas formal. Esto proporcionó un modelo algebraico de la lógica de proposiciones.
- **En 1879:** Gottlob Frege formalizó la lógica de predicados.

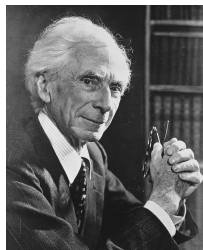


Boole



Frege

- **1910-1913:** Alfred North Whitehead y Bertrand Russell publican “Principia Mathematica”.
- **1920:** Hilbert propone el problema de la axiomatización de las matemáticas...
- **...en 1930:** Gödel rebate esta posibilidad al demostrar los teoremas de incompletitud.



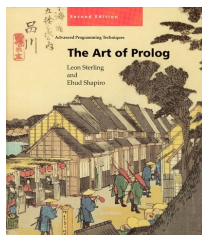
Russel



Gödel

Lógica e informática

- **1965:** Alan Robinson publica un método de resolución para lógica de primer orden. Sienta las bases de la deducción automática:
 - Verificación automática de programas: a partir de su especificación formal y utilizando demostradores automáticos de teoremas.
- **1972:** Alain Colmerauer crea Prolog, el primer lenguaje de programación lógica. Sienta las bases de la inteligencia artificial:
 - Permite inferir/deducir conocimiento a partir de una base de conocimientos y una serie de reglas (de inferencia).



Prolog



Colmerauer

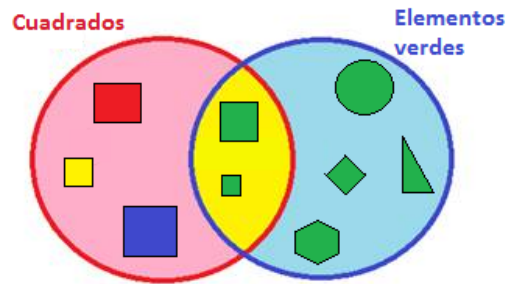
1.2. Teoría de conjuntos, relaciones, funciones y álgebra de Boole.

1.2.1. Teoría de Conjuntos

- Definición de conjunto y pertenencia.
- Un conjunto A es una colección, finita o infinita, de objetos de un universo U tal que para todo objeto x se puede determinar si x pertenece a A ($x \in A$).
- Si no perteneciera se indicaría como: ($x \notin A$).
- Los objetos que pertenecen a un conjunto se conocen como **elementos** del conjunto.

Representación

- Una manera muy común de representar relaciones y operaciones entre conjuntos son los diagramas de Venn.



Inclusión e igualdad

- Si todo elemento x de un conjunto A es también elemento de un conjunto B , diremos que A está contenido en B , o que A es un subconjunto de B :

$$A \subseteq B$$

- Si A es un subconjunto de B y existe un elemento de B que no pertenece a A , entonces A es un subconjunto propio de B :

$$A \subset B$$

- Dos conjuntos serán iguales ($A \equiv B$), si contienen los mismo elementos:

$$A \subseteq B \text{ y } B \subseteq A$$

Conjunto vacío

- El conjunto vacío \emptyset es el conjunto que no tiene elementos:

$$\emptyset = \{ \}$$

- Dado el conjunto $A = \{\emptyset\}$. ¿Cual es la respuesta correcta?:

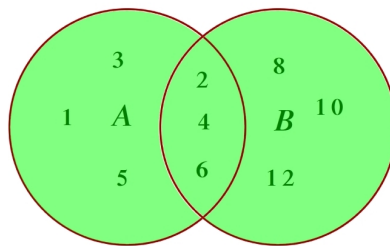
$$A = \emptyset \text{ o } A \neq \emptyset$$

Operaciones

Unión

- La unión de dos conjuntos A y B , es el conjunto $A \cup B$ de todos los elementos de A o de B , es decir:

$$A \cup B = \{x : x \in A \text{ o } x \in B\}$$

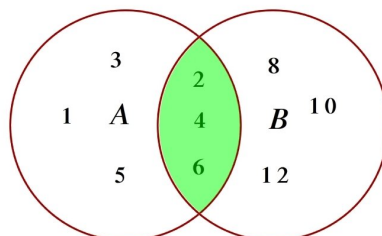
Union

$$A \cup B = \{1, 2, 3, 4, 5, 6, 8, 10, 12\}$$

Intersección

- La intersección de dos conjuntos A y B , es el conjunto $A \cap B$ de todos los elementos que pertenecen a A y a B , es decir:

$$A \cap B = \{x : x \in A \text{ y } x \in B\}$$

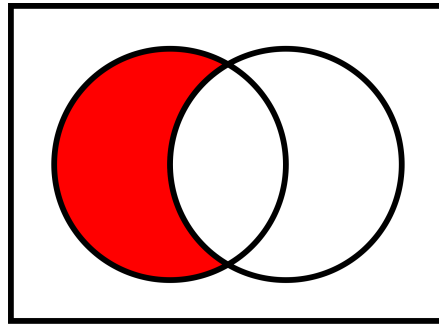
Intersection

$$A \cap B = \{2, 4, 6\}$$

Complemento relativo

- El complemento relativo de B respecto de A (o diferencia $A - B$), es el conjunto $A \setminus B$ de todos los elementos de A que no pertenecen a B , es decir:

$$A \setminus B = \{x : x \in A \text{ y } x \notin B\}$$



Producto cartesiano

- El producto cartesiano de dos conjuntos no vacíos A y B , es el conjunto de todos los pares ordenados (a, b) con $a \in A$ y $b \in B$, es decir:

$$A \times B = \{(a, b) \mid a \in A \text{ y } b \in B\}$$

- Dado $A = \{1, 2\}$ y $B = \{a, b, c\}$

$$A \times B = \{(1, a), (1, b), (1, c), (2, a), (2, b), (2, c)\}$$

Partes de un conjunto

- Sea A un conjunto, se llama conjunto de las partes de A , $P(A)$, al conjunto cuyos elementos son exactamente los subconjuntos de A .
- Dado $A = \{1, 2, 3\}$, el conjunto de las partes de A es:

$$P(A) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$$

Propiedades de las operaciones

- Sean A , B y C tres conjuntos. Las principales propiedades de las operaciones con conjuntos son las siguientes:

- Idempotencia** de la unión y de la intersección:

$$A \cup A = A$$

$$A \cap A = A$$

- Conmutatividad** de la unión y de la intersección:

$$A \cup B = B \cup A$$

$$A \cap B = B \cap A$$

3. **Asociatividad** de la unión y de la intersección:

$$A \cup (B \cap C) = (A \cup B) \cap C$$

$$A \cap (B \cup C) = (A \cap B) \cup C$$

4. **Distributiva** de la unión respecto de la intersección y de la intersección respecto a la unión:

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

5. **Leyes de Morgan:**

$$C \setminus (A \cup B) = (C \setminus A) \cap (C \setminus B) \quad \overline{A \cup B} = \bar{A} \cap \bar{B}$$

$$C \setminus (A \cap B) = (C \setminus A) \cup (C \setminus B) \quad \overline{A \cap B} = \bar{A} \cup \bar{B}$$

6. **Otras propiedades interesantes:**

$$(A \setminus B) \cup B = A \cup B$$

$$(A \setminus B) \cap B = \emptyset$$

$$A \setminus (A \setminus B) = A \cap B$$

Cardinal

- El cardinal de un conjunto, $Card(A)$ o $|A|$, es el número de elementos de dicho conjunto.
- Sean A y B dos conjuntos finitos cualesquiera:

$$Card(A \cup B) = Card(A) + Card(B) - Card(A \cap B)$$

$$|A \cup B| = |A| + |B| - |A \cap B|$$

1.2.2. Relaciones Binarias

- Una relación binaria entre A y B es un subconjunto R del producto cartesiano $A \times B$. Si $(a, b) \in R$ se dirá que a y b están relacionados:

$$aRb$$

- Dado $A = \{1, 2\}$ y $B = \{a, b, c\}$, cuyo producto cartesiano es:

$$A \times B = \{(1, a), (1, b), (1, c), (2, a), (2, b), (2, c)\}$$

- Si consideramos $R = \{(1, c), (2, b), (1, a)\}$ entonces tenemos que:

$$1Rc, 2Rb, 1Ra$$

Definiciones

- Dada una relación binaria $R \subseteq A \times B$, se denomina:

- Dominio de R :

$$\text{dom}(R) = \{x \in A : \exists y \in B \mid (x, y) \in R\} \quad \text{dom}(R) \subseteq A$$

- Imagen directa (o rango) de R :

$$\text{Im}(R) = \{y \in B : \exists x \in A \mid (x, y) \in R\} \quad \text{Im}(R) \subseteq B$$

- Imagen inversa (o recíproca) de un subconjunto C de B :

$$R^{-1}(C) = \{x \in A : \exists y \in C \mid (x, y) \in R\} \quad R^{-1}(C) \subseteq A$$

- Codominio de R al conjunto B

Homogéneas

- Sea R una relación binaria homogénea en un conjunto A , no vacío ($R \subseteq A \times A$). La relación binaria R puede ser:

1. **Reflexiva:** $\forall x \in A \mid (x, x) \in R$

2. **Simétrica:** $\forall x, y \in A \mid (x, y) \in R \rightarrow (y, x) \in R$

3. **Antisimétrica:** $\forall x, y \in A \mid (x, y) \in R \wedge (y, x) \in R \rightarrow x = y$

4. **Transitiva:** $\forall x, y, z \in A \mid (x, y) \in R \wedge (y, z) \in R \rightarrow (x, z) \in R$

de Equivalencia

- Una relación binaria homogénea R , en un conjunto no vacío A , se denomina relación de equivalencia si es *reflexiva*, *simétrica* y *transitiva*.
- Si R es una relación de equivalencia en lugar de R se escribe \sim .
- Si $a \in A$ y \sim es una relación de equivalencia en A , se define la clase de equivalencia como:

$$C(a) = \{x \in A : x \sim a\}$$

1.2.3. Relaciones n-arias

- Una relación n-aria entre n conjuntos A_1, A_2, \dots, A_n , es un subconjunto R del producto cartesiano $A_1 \times A_2 \times \dots \times A_n$.
- Es una generalización de la relación binaria donde R está formada por una tupla de n términos:

$$R = \{(x_1, x_2, \dots, x_n) : x_1 \in X_1 \wedge x_2 \in X_2 \wedge \dots \\ \dots \wedge x_n \in X_n \wedge R(x_1, x_2, \dots, x_n) = \text{verdadero}\}$$

- El predicado n-ario: $R(x_1, x_2, \dots, x_n)$ es una función que asigna el valor de verdad verdadero si y solo si $(x_1, x_2, \dots, x_n) \in R$.

1.2.4. Funciones n-arias

- Una función (o aplicación) n-aria, $f : A_1 \times A_2 \times \dots \times A_n \rightarrow B$, de un conjunto no vacío $A = A_1 \times A_2 \times \dots \times A_n$ a un conjunto no vacío B se puede definir como una “regla de correspondencia” que asigna a cada elemento elemento $(a_1, a_2, \dots, a_n) \in A$ un único elemento $b \in B$:

$$f(a_1, a_2, \dots, a_n) = b$$

- Una función $f : A_1 \times A_2 \times \dots \times A_n \rightarrow B$ es a su vez una relación binaria entre el conjunto $A = A_1 \times A_2 \times \dots \times A_n$ y el conjunto B , tal que a cada elemento $(a_1, a_2, \dots, a_n) \in A$ le corresponde un único elemento $f(a_1, a_2, \dots, a_n)$ del conjunto B .

1.2.5. Algebra de Boole

Definición

- Definida en 1847 por el matemático inglés George Boole.
- Es un álgebra con sólo dos valores, p.ej.: **Falso** o **Verdadero**.
...una variable *lógica* tiene dos valores posibles, uno excluye al otro.
- Sobre estos valores definió tres operaciones básicas, p.ej.:
 - NOT (\neg): negación lógica o función complemento.

- OR (\vee): suma lógica o función unión.
- AND (\wedge): producto lógico o función intersección.

\neg	
V	F
F	V

\vee	V	F
V	V	V
F	V	F

\wedge	V	F
V	V	F
F	F	F

- Son Álgebras de Boole (entre otras):
 $(\{\emptyset, U\}, \sim, \oplus, \odot)$, $(\{F, V\}, \neg, \vee, \wedge)$, $(\{0, 1\}, -, +, \cdot)$, y $(P(U), -, \cup, \cap)$.

Interpretación

- Los elementos del álgebra de Boole representa estados diferentes de un dispositivo, p.ej., abierto/cerrado, encendido/apagado, etc.
- Matemáticamente se representan por 1 y 0, y responde a la pregunta:
 - ¿Cómo expresar la clase *not-X* (clase de individuos que no son Xs)?
 - La clase X y la *not-X* juntas hacen el Universo, representado por 1.
 - Por lo tanto, si la clase X es x , la clase *not-X* es $1 - x$.
- ... y muchas más. p.ej.:
 - Todos Xs son Ys y todos Ys son Xs. $x = y$
 - Todos Xs son Ys. $x(1 - y) = 0$
 - No Xs son Ys. $xy = 0$
 - Todos Ys son Xs y algunos Xs son Ys. ...
 - etc...

Propiedades y teoremas

- Usando la notación matemática del álgebra de Boole:

Propiedades	Suma	Producto
Conmutativa	$x + y = y + x$	$x \cdot y = y \cdot x$
Elemento neutro	$0 + x = x$	$1 \cdot x = x$
Distributiva	$x \cdot (y + z) = (x \cdot y) + (x \cdot z)$	$x + (y \cdot z) = (x + y) \cdot (x + z)$
Asociativa	$x + (y + z) = (x + y) + z$	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$
Complementario	$x + \bar{x} = 1$	$x \cdot \bar{x} = 0$
Teoremas		
Idempotencia	$x + x = x$	$x \cdot x = x$
Identidad	$x + 1 = 1$	$x \cdot 0 = 0$
Absorción	$x + x \cdot y = x$	$x \cdot (x + y) = x$
DeMorgan	$\overline{x + y} = \bar{x} \cdot \bar{y}$	$\overline{x \cdot y} = \bar{x} + \bar{y}$

Simplificación de ecuaciones

- Aplicando propiedades y teoremas podemos simplificar ecuaciones:

$$x_2 \cdot x_1 + x_2 \cdot \bar{x}_1 + x_2 \cdot x_0 \stackrel{?}{=} x_2$$

$x_2 \cdot x_1 + x_2 \cdot \bar{x}_1 + x_2 \cdot x_0 =$	Distributiva
$= x_2 \cdot (x_1 + \bar{x}_1) + x_2 \cdot x_0 =$	Complemento
$= x_2 \cdot 1 + x_2 \cdot x_0 =$	Distributiva
$= x_2 \cdot (1 + x_0) =$	Identidad
$= x_2 \cdot 1 =$	Elemento neutro
$= x_2$	

...para ver si son equivalentes y/o para construir circuitos mas baratos.

- Las ecuaciones simplificadas se llaman formas canónicas. Hay dos:
 - Suma de productos.
 - Producto de sumas.



C pítulo 2

L gica Proposicional

2.1	Sintaxis, sem�ntica y teor�a interpretativa.	16
2.1.1	Lenguaje proposicional	16
	Alfabeto	16
	F�rmula Bien Formada	17
	Precedencia	17
	Par�ntesis	17
	Ejercicios: FBF y par�ntesis	18
	Ejercicios: Formalizaci�n	18
2.1.2	Sem�ntica de la l�gica proposicional	19
	Funciones de verdad	19
	Funciones de interpretaci�n	20
	Ejercicios	20
2.1.3	Satisfacibilidad	20
	Validez	21
	Conclusi�n	21
	Ejercicios: Satisfacibilidad	21
2.1.4	Teor�a interpretativa	22
	Consecuencia L�gica	22
	Ejercicios: Consecuencia L�gica	24
	Equivalencia l�gica	25
	Ejercicios: Equivalencia l�gica	26
2.2	Sistemas formales y deducci�n natural.	26
2.2.1	Motivaci�n	26
2.2.2	Sistemas formales	27
	Propiedades	27

2.2.3	Deducción Natural	28
	Supuestos y Premisas	28
	Teorema de la Deducción	29
	Reglas de inferencia I	29
	Deducción Natural: Ejemplo	30
	Ejercicios: Deducción Natural	32
	Reglas de inferencia Derivadas	32
	Teorema de intercambio	34
	Ejercicios: Deducción Natural	34

2.1. Sintaxis, semántica y teoría interpretativa.

2.1.1. Lenguaje proposicional

- En el contexto de la lógica formal se requiere un lenguaje:
 - Sintácticamente preciso y no ambiguo.
 - Con un significado (semántica) unívoco, y no que una palabra pueda significar cosas distintas según algún tipo de contexto.
 - Cuya definición sea muy compacta.
- Queremos símbolos para representar hechos lógicos, es decir, enunciados por los que tiene sentido preguntarse si son verdaderos o falsos.
- Los enunciados más sencillos son los que no dependen de otros:
 - Llueve.
 - Nadal ganó el US Open 2019.

Alfabeto

- Los símbolos de proposición representan este tipo de hechos:
 - p puede representar “llueve”.
 - q puede representar “Nadal ganó el US Open 2019”.
- Para enunciados más complejos necesitamos símbolos que representen la relación entre los más sencillos que los componen:
 - llueve o nieva.
 - si p representa “llueve” y r representa “nieva”, $p \vee r$ representaría “llueve o nieva”.
- **Alfabeto** de un lenguaje proposicional:

- Símbolos de proposición: $p, q, r, \dots, p_1, p_2, \dots$
- Conectivas lógicas: $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$.

Fórmula Bien Formada

- Del conjunto de todas las posibles secuencias (finitas) de símbolos de un alfabeto, ¿cuáles se consideran expresiones bien formadas?
- Son **fórmula bien formada (FBF)**:
 - F si F es un símbolo de proposición.
 - $\neg F$ si F es un símbolo de proposición.
 - y si F y G son FBFs, también son FBFs:
 - $F \wedge G$ (conjunción).
 - $F \vee G$ (disyunción).
 - $F \rightarrow G$ (implicación).
 - $F \leftrightarrow G$ (doble implicación).

Definición recursiva de fórmulas bien formadas.

- Dado un alfabeto A , el conjunto (infinito) de FBFs que se pueden definir sobre A se denomina FBF_A .

Precedencia

- ¿Que dice la siguiente fórmula?

$$p \rightarrow q \wedge r \begin{cases} \mathbf{p} \rightarrow \mathbf{q} \wedge r & (a) \\ p \rightarrow \mathbf{q} \wedge r & (b) \end{cases}$$

- (a) Si p entonces q , y además r .
- (b) Si p entonces suceden dos cosas: q y r .
- Se define la siguiente precedencia entre las conectivas:
 - \neg mayor precedencia que \wedge y \vee .
 - \wedge y \vee mayor precedencia que \rightarrow y \leftrightarrow .
 - En caso de igual precedencia se agrupan de derecha a izquierda.

Paréntesis

- Si los paréntesis siguen las reglas de precedencia no son necesarios:

- $p \wedge q \wedge r$ es igual a $p \wedge (q \wedge r)$
 - $p \wedge q \vee r$ es igual a $p \wedge (q \vee r)$
 - $q \rightarrow r1 \wedge \neg r2$ es igual a $q \rightarrow (r1 \wedge (\neg r2))$
 - $cgm \wedge p \vee \neg \neg q \leftrightarrow ll$ es igual a $(cgm \wedge (p \vee (\neg(\neg q)))) \leftrightarrow ll$
 - $p \rightarrow q \rightarrow r \rightarrow s$ es igual a $p \rightarrow (q \rightarrow (r \rightarrow s))$
 - $\neg \neg \neg \neg \neg \neg p$ es igual a $\neg(\neg(\neg(\neg(\neg(\neg p))))))$
- Los paréntesis son necesarios para dar otro “significado”:
- $p \wedge q \wedge r$ sería $(p \wedge q) \wedge r$
 - $cgm \wedge p \vee \neg \neg q \leftrightarrow ll$ sería $(cgm \wedge p) \vee (\neg \neg q \leftrightarrow ll)$
 - $p \rightarrow q \rightarrow r \rightarrow s$ sería $(p \rightarrow q) \rightarrow (r \rightarrow s)$

Ejercicios: FBF y paréntesis

- Determinar si son FBFs las siguientes fórmulas y eliminar paréntesis superfluos usando las convenciones de precedencia:
1. $(q \wedge \neg(p)) \rightarrow (\neg(q) \vee r)$.
 2. $(p \wedge \neg(q \vee r)) \vee (\neg(p) \vee q)$.
 3. $((p \vee q) \vee \neg(r)) \leftrightarrow (r \wedge q)$.
 4. $(p \wedge \neg(q \vee r)) \vee (\neg(\rightarrow p))$.
 5. $(p) \leftrightarrow ((p \wedge q) \vee (p \wedge (\neg(q))))$.

Ejercicios: Formalización

- Formalizar en lógica proposicional:
1. Es necesario abrir la botella para disfrutar su contenido.
 2. Basta con romper el sello para perder la garantía.
 3. Llama al 112 cuando tengas una emergencia.
 4. Perderé el tren a menos que coja un taxi.
 5. Cambia la bombilla sólo y siempre que esté fundida.
 6. No iré al cine a no ser que me inviten.
 7. No pulse la alarma de incendios excepto cuando detecte humo en la escalera.
 8. Pulse la alarma sólo y únicamente si detecta humo en la escalera.
- Formalizar en lógica proposicional (Manzano y Huertas, 2004):
9. Sólo si Pedro juega(p) jugará también Alex(q)
 10. Pedro irá al dentista(p), tanto si quiere(q) como si no quiere($\neg q$)

11. La magia se revela(p) sólo si Pinocho miente(q) o Blancanieves muerde la manzana(r)
 12. El certificado tiene validez(p) si está firmado por el director(q) o el tutor del proyecto(r)
 13. La inflación aumentará(p) a menos que baje la emisión de moneda(q) u ocurra un milagro(r)
- Formalizar en lógica proposicional (Manzano y Huertas, 2004):
 14. Leeré a Proust(p) si me voy de vacaciones(q) y encuentro sus libros en oferta(r)
 15. Si el mal existe en el mundo(p) y no se origina por las acciones humanas(q), entonces Dios no quiere(r) o no puede(s) impedirlo
 16. Te regalaré el cuadro que te gusta(p) y viajaremos juntos a Italia(q) cuando me toque la lotería(r), o dejo de llamarme Ernesto(s)
 17. Es necesario que llueva(p) o que haga viento(q) para que disminuya la contaminación(r)
 18. Si llueve(p) y hace viento(q), disminuye la contaminación(r)

2.1.2. Semántica de la lógica proposicional

- Proposición: Una condición/afirmación posible del mundo sobre el que queremos decir algo.
- Una proposición puede ser *Verdadera* (V) o *Falsa* (F).
- Proposiciones simples:
 - Su valor de verdad no depende de otra proposición.
- Proposiciones compuestas (FBF):
 - Su valor de verdad depende del que tengan las proposiciones simples que la definan y
 - del significado de las conectivas (definido por las funciones de verdad).

Funciones de verdad

- Primero definimos una función de verdad (s) para cada conectiva:
 - $s_{\neg}(V) = F$ $s_{\neg}(F) = V$
 - $s_{\wedge}(V, V) = V$ $s_{\wedge}(V, F) = s_{\wedge}(F, V) = s_{\wedge}(F, F) = F$
 - $s_{\vee}(F, F) = F$ $s_{\vee}(V, F) = s_{\vee}(F, V) = s_{\vee}(V, V) = V$
 - $s_{\rightarrow}(V, F) = F$ $s_{\rightarrow}(F, F) = s_{\rightarrow}(F, V) = s_{\rightarrow}(V, V) = V$
 - $s_{\leftrightarrow}(V, V) = s_{\leftrightarrow}(F, F) = V$ $s_{\leftrightarrow}(V, F) = s_{\leftrightarrow}(F, V) = F$

Funciones de interpretación

- Una función de interpretación, i , asignar un significado a todas y cada una de las FBFs de un alfabeto A .

$$i : FBF_A \Rightarrow \{V, F\}$$

- La interpretación de una FBF se define como:

- $i(p) = V/F$ si p es una proposición p .
- $i(\neg G) = s_{\neg}(i(G))$ si G es una FBF.
- y si G y H son FBFs:
 - $i(G \wedge H) = s_{\wedge}(i(G), i(H))$.
 - $i(G \vee H) = s_{\vee}(i(G), i(H))$.
 - $i(G \rightarrow H) = s_{\rightarrow}(i(G), i(H))$.
 - $i(G \leftrightarrow H) = s_{\leftrightarrow}(i(G), i(H))$.

Ejercicios

- Asignar significado a las siguientes fórmulas cuando:

$$i(p) = i(q) = V \text{ y } i(r) = F.$$

1. $(p \rightarrow q \vee r) \rightarrow (p \wedge q \rightarrow \neg r)$
2. $(p \wedge (\neg q \vee \neg r)) \leftrightarrow (\neg(p \vee q) \rightarrow r)$

- Asignar significado a las siguientes fórmulas para toda posible interpretación:

1. $\neg(p \vee q) \leftrightarrow \neg p \wedge \neg q$
2. $(\neg p \rightarrow q) \wedge (\neg q \wedge p)$
3. $(p \vee q) \rightarrow (p \wedge q)$

2.1.3. Satisfacibilidad

Definición Se dice que una interpretación i satisface una fórmula proposicional $G \in FBF_A$, si y sólo si (sii) es verdadera bajo esa interpretación:

$$i(G) = V$$

- Para conjuntos de fórmulas $\{G_1, \dots, G_n\}$, $G_i \in FBF_A$ para todo $i : 1 \leq i \leq n$: Una interpretación i **satisface** $\{G_1, \dots, G_n\}$ sii

$$i(G_i) = V \text{ para todo } i : 1 \leq i \leq n$$

Definición

- Una fórmula $G \in FBF_A$ es **satisfacible** sii existe (al menos) una interpretación i tal que $i(G) = V$.
- Una fórmula $G \in FBF_A$ es **insatisfacible** sii no existe ninguna interpretación i tal que $i(A) = V$
- **Modelo** de una fórmula: Una interpretación que la satisface.
- **Contramodelo** de una fórmula: Una interpretación que la hace falsa.

Validez

Atendiendo a su semántica, una fórmula G es:

- **Contradicción** sii no existe una interpretación i tal que $i(G) = V$.
- **Válida** (o tautología) sii no existe una interpretación i tal que $i(G) = F$ (se representa $\vdash G$).
- **Contingente** sii existe alguna interpretación i tal que $i(G) = V$ y existe alguna interpretación i' tal que $i'(G) = F$.
- Una fórmula G es válida sii $\neg G$ es una contradicción.
- Una fórmula G es contingente sii $\neg A$ es contingente.

Conclusión

- Una fórmula es válida sii
 - no tiene contramodelos sii
 - todas sus interpretaciones son modelos sii
 - todas sus interpretaciones la satisfacen.
- Una fórmula es una contradicción sii
 - no tiene modelos sii
 - todas sus interpretaciones son contramodelos sii
 - es insatisfacible.
- Una fórmula es contingente sii
 - tiene modelos y contramodelos.

Ejercicios: Satisfacibilidad

- Determinar para cada una de las siguientes fórmulas si es válida, contradictoria o contingente, indicando la(s) interpretación(es) que lo demuestran:

1. $p \wedge q \rightarrow p$.
2. $p \vee q \rightarrow p$.
3. $p \rightarrow \neg p$.
4. $p \vee q \rightarrow (r \vee s \rightarrow p)$.
5. $(p \rightarrow q) \wedge (p \wedge \neg q)$.
6. $(p \rightarrow q) \wedge (q \rightarrow r) \rightarrow (p \rightarrow r)$.
7. $\neg(p \vee q) \leftrightarrow \neg p \vee \neg q$.
8. $\neg(p \vee q) \leftrightarrow \neg p \wedge \neg q$.

- Determinar para cada una de las siguientes fórmulas si es válida, contradictoria o contingente, indicando la(s) interpretación(es) que lo demuestran:

9. $(p \rightarrow \neg q) \wedge \neg(r \wedge \neg p) \rightarrow (q \rightarrow \neg r)$.
10. $(p \wedge q \rightarrow r) \rightarrow (p \rightarrow (q \rightarrow r))$.
11. $\neg(p \rightarrow q) \leftrightarrow p \wedge \neg q$.
12. $p \rightarrow (q \rightarrow r)$.
13. $p \rightarrow (q \wedge \neg q \rightarrow \neg p)$.
14. $(p \rightarrow q) \wedge (q \rightarrow p)$.
15. $(p \rightarrow q) \rightarrow ((p \rightarrow r) \rightarrow (p \rightarrow q \wedge r))$.

2.1.4. Teoría interpretativa

Consecuencia Lógica

- Dado un lenguaje proposicional LP, un conjunto de fórmulas $\{A_1, \dots, A_n\}$, $A_i \in FBF_{LP}$ para todo $i: 1 \leq i \leq n$, y una fórmula $B \in FBF_{LP}$:

Consecuencia lógica

- B es consecuencia lógica de $\{A_1, \dots, A_n\}$.

$$[A_1, \dots, A_n] \models B$$

- sii toda interpretación que satisface $\{A_1, \dots, A_n\}$ también satisface B.
- sii no existe ninguna interpretación que satisfaga $\{A_1, \dots, A_n\}$ y no satisfaga a B.

Argumento correcto

- Un argumento con premisas $\{A_1, \dots, A_n\}$ y consecuente B es correcto sii $[A_1, \dots, A_n] \models B$

- Para decidirlo se pueden hacer dos análisis:

- 1 Ver si todas las interpretaciones que satisfacen $\{A_1, \dots, A_n\}$ también satisfacen B .
- 2 o bien ver que no existe una sola interpretación que satisfaga $\{A_1, \dots, A_n\}$ y no satisfaga B .
 - El caso 1 requiere examinar todas las interpretaciones posibles y ver si se cumple la condición.
 - El caso 2 podemos centrarnos en definir una interpretación i tal que $i(\{A_1, \dots, A_n\}) = V$ y $i(B) = F$.

Teoría interpretativa: Consecuencia Lógica: Ejemplo I

- Analiza si se cumple la relación de consecuencia lógica:

$$\{p \rightarrow (q \rightarrow r), p \wedge q\} \models r$$

p	q	r	$q \rightarrow r$	$p \rightarrow (q \rightarrow r)$	$p \wedge q$	$[A_1, \dots, A_n]$ $\{p \rightarrow (q \rightarrow r), p \wedge q\}$	B r
V	V	V	V	V	V	V	V
V	V	F	F	F	V	F	F
V	F	V	V	V	F	F	V
V	F	F	V	V	F	F	F
F	V	V	V	V	F	F	V
F	V	F	F	V	F	F	F
F	F	V	V	V	F	F	V
F	F	F	V	V	F	F	F

- De todas las interpretaciones posibles, sólo una hace verdad a las dos premisas (A_1 y A_2), y esa interpretación también hace verdad al consecuente (B). Por tanto, sí hay relación de consecuencia lógica.

Teoría interpretativa: Consecuencia Lógica: Ejemplo II

- Analiza si se cumple la relación de consecuencia lógica:

$$\{p \wedge \neg\neg q, r\} \models q \vee s$$

- tratamos de definir un contramodelo del argumento:

1. $i(p \wedge \neg\neg q) = V$ sii

- a) $i(p) = V$ b) y $i(\neg\neg q) = V$ sii $i(\neg q) = F$ sii $i(q) = V$

2. $i(r) = V$

3. $i(q \vee s) = F$ sii

- a) $i(q) = F$ (en contradicción con 1.b)

- b) y $i(s) = F$

- Puesto que **no** es posible definir un contramodelo:

El argumento es correcto, **hay relación de consecuencia lógica.**

Teoría interpretativa: Consecuencia Lógica: Ejemplo III

- Analiza si se cumple la relación de consecuencia lógica:

$$\{p \wedge q, \neg(p \rightarrow r)\} \models q \wedge (p \rightarrow r)$$

- tratamos de definir un contramodelo del argumento:

1. $i(p \wedge q) = V$ sii

a) $i(p) = V$

b) y $i(q) = V$

2. $i(\neg(p \rightarrow r)) = V$ sii $i(p \rightarrow r) = F$ sii

a) $i(p) = V$

b) y $i(r) = F$

3. $i(q \wedge (p \rightarrow r)) = F$ sii

a) $i(q) = F$

(en contradicción con 1.b)

b) o $i(p \rightarrow r) = F$

(OK, compatible con 2).

- Existe un contramodelo al argumento:

$$i(p) = i(q) = V, i(r) = F.$$

El argumento no es correcto. **NO** hay relación de consecuencia lógica.

Ejercicios: Consecuencia Lógica

- Determinar si las siguientes argumentaciones son correctas. Si no lo son, indicar la interpretación que lo demuestra (contramodelo).

1. $\{p, p \rightarrow q\} \models q$

2. $\{\neg p, p \vee q\} \models q$

3. $\{p \rightarrow q, \neg p\} \models \neg q$

4. $\{p \rightarrow q, \neg q\} \models \neg p$

5. $\{p \leftrightarrow q, \neg p\} \models q$

6. $\{p \wedge q\} \models p$

7. $\{\neg(p \wedge q)\} \models \neg p \wedge \neg q$

8. $\{\neg(p \vee q)\} \models \neg p \wedge \neg q$

- Determinar si las siguientes argumentaciones son correctas. Si no lo son, indicar la interpretación que lo demuestra (contramodelo).

9. $\{p \rightarrow q, p\} \models q$

10. $\{p \vee q\} \models q \vee p$

11. $\{p \wedge (q \vee r)\} \models p$

12. $\{p \vee q \rightarrow r\} \models q \rightarrow r$

13. $\{\neg\neg r \wedge \neg q\} \models \neg r$

14. $\{p \rightarrow (q \rightarrow r)\} \models q \rightarrow (p \rightarrow r)$

15. $\{\neg q \rightarrow r, t \rightarrow \neg q, \neg s \rightarrow \neg q\} \models t \vee \neg s \rightarrow r$

16. $\{p \vee (q \rightarrow r) \rightarrow q, p\} \models q$
 17. $\{\neg p \rightarrow \neg s, \neg p \vee r, r \rightarrow \neg t\} \models \neg s \vee \neg t$
 18. $\{(p \rightarrow q) \wedge t, (r \vee p) \wedge \neg q, \neg t \leftrightarrow \neg s\} \models r \wedge s$

Equivalencia lógica

- Dos fórmulas A y B son (lógicamente) equivalentes ($A \leftrightarrow B$) sii para toda interpretación i se cumple que $i(A) = i(B)$.
- Esta definición implica que:
 - A y B son consecuencia lógica una de la otra ($A \models B$ y $B \models A$).
 - La fórmula $A \leftrightarrow B$ es válida (es una tautología).
- Por ejemplo: $p \rightarrow q \leftrightarrow \neg p \vee q$

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

- La equivalencia entre fórmulas proporciona numerosas ventajas prácticas, entre ellas:
 - permite utilizar indistintamente las fórmulas equivalentes en una demostración (lo utilizaremos más adelante).
 - permite reducir el tamaño de un lenguaje proposicional (disminuir el n^o de conectivas que emplea).
- Por ejemplo, cualquier lenguaje proposicional puede reducirse a otro que sólo utiliza $\{\neg, \vee\}$
 - Esta reducción simplifica tareas como:
 - construcción de sistemas sintácticos de demostración.
 - demostración de las propiedades metalógicas del sistema formal.

$\neg(\neg p) \leftrightarrow p$	doble negación
$p \wedge p \leftrightarrow p$	ley de idempotencia
$p \wedge q \leftrightarrow q \wedge p$	conmutatividad de la conjunción
$p \vee q \leftrightarrow q \vee p$	conmutatividad de la disyunción
$p \wedge (q \wedge r) \leftrightarrow (p \wedge q) \wedge r$	asociatividad de la conjunción
$p \vee (q \vee r) \leftrightarrow (p \vee q) \vee r$	asociatividad de la disyunción
$p \wedge (q \vee r) \leftrightarrow (p \wedge q) \vee (p \wedge r)$	distributividad de la conjunción respecto a la disyunción
$p \vee (q \wedge r) \leftrightarrow (p \vee q) \wedge (p \vee r)$	distributividad de la disyunción respecto a la conjunción
$p \leftrightarrow q \leftrightarrow (p \rightarrow q) \wedge (q \rightarrow p)$	definición de la doble implicación en función de la implicación
$p \rightarrow q \leftrightarrow \neg p \vee q$	implicación vs. disyunción
$\neg(p \vee q) \leftrightarrow \neg p \wedge \neg q$	ley de De Morgan
$\neg(p \wedge q) \leftrightarrow \neg p \vee \neg q$	ley de De Morgan

Ejercicios: Equivalencia lógica

- De los pares de fórmulas siguientes, ¿en cuáles se cumple $A \models B$?, ¿en cuáles se cumple $B \models A$?, ¿en cuáles A y B son equivalentes?
 1. A: $p \vee q$, B: $q \vee p$
 2. A: $p \rightarrow q$, B: $q \rightarrow p$
 3. A: $p \vee q \rightarrow r$, B: $p \rightarrow r$
 4. A: $(p \rightarrow q) \rightarrow r$, B: $p \vee q \rightarrow r$
 5. A: $p \rightarrow q$, B: $p \leftrightarrow q$

2.2. Sistemas formales y deducción natural.

2.2.1. Motivación

- ¿Por qué queremos construir un cálculo deductivo?
 - Es difícil determinar $\{A_1, \dots, A_n\} \models B$ por medios **semánticos**.
 - Confirmar la corrección de un argumento puede ser muy costoso.
 - Alternativa: determinar que B se deduce de $\{A_1, \dots, A_n\}$ por medios **sintácticos**: $\{A_1, \dots, A_n\} \vdash B$
 - Encontrar un procedimiento que nos permita construir una argumenta-

ción paso a paso, manipulando los símbolos de las fórmulas, sabiendo que cada paso es válido.

- El análisis de la corrección de un argumento por medios **sintácticos** se hace siempre en un contexto o marco formal, denominado **sistema formal**.
- En un sistema formal **los símbolos carecen de significado**, y al manipularlos hemos de ser cuidadosos y no presuponer nada sobre sus propiedades, salvo lo que se especifique en el sistema.

2.2.2. Sistemas formales

- Un sistema formal de demostración consiste en:
 - Un **lenguaje** formal (alfabeto y reglas sintácticas de formación de fórmulas).
 - Un conjunto de axiomas lógicos o **axiomas** (fórmulas válidas sin prueba, podría ser vacío).
 - Un conjunto de **reglas de inferencia** para demostrar fórmulas.
 - Una definición de prueba o **demostración**.
- Una teoría T es un sistema formal ampliado con un conjunto Γ de axiomas no lógicos o premisas (es decir, que se consideran como verdad): $T[\Gamma]$.
 - Si $\Gamma = \emptyset$ entonces T es la teoría básica del sistema formal.
- Una **demostración** o prueba de una fórmula B en una teoría $T[\Gamma]$ (escrito $T[\Gamma] \vdash B$) es una secuencia finita de fórmulas tal que:
 - Toda fórmula de la secuencia es:
 - Un axioma o premisa de la teoría.
 - o, el resultado de aplicar una regla de inferencia a fórmulas anteriores en la secuencia.
 - B es la última fórmula de la secuencia
- Un **teorema** de una teoría $T[\Gamma]$ es una fórmula para la que existe al menos una demostración en $T[\Gamma]$
- $T[\Gamma] \vdash B$ indica que B se deduce de $T[\Gamma]$ (B es teorema de $T[\Gamma]$)

Propiedades

Corrección: Teorema de validez. Todos los teoremas de $T[\Gamma]$ son consecuencias lógicas de Γ :

$$\text{si } T[\Gamma] \vdash B \text{ entonces } \Gamma \models B.$$

Completitud: Teorema de completitud.

Dada una teoría $T[\Gamma]$, todas las consecuencias lógicas de Γ son teoremas de $T[\Gamma]$:

$$\text{si } \Gamma \models B \text{ entonces } T[\Gamma] \vdash B.$$

Si el cálculo es **correcto** y **completo** entonces \vdash y \models son equivalentes.

2.2.3. Deducción Natural

- Es un sistema formal para la lógica proposicional.
 - Lenguaje: Lenguaje proposicional.
 - Axiomas: No tiene.
 - Reglas de inferencia: dos por cada conectiva:
 - Introducción y eliminación.
 - Demostración: es una secuencia finita de fórmulas en la que cada elemento es:
 - Un *supuesto* o *premisa* de la teoría.
 - o, el resultado de la aplicación de una regla de inferencia a fórmulas anteriores en la secuencia.
- y la última fórmula de la secuencia es la fórmula probada.

Supuestos y Premisas

- Son fórmulas añadidas a una teoría básica T y pueden aparecer en una prueba sin requerir demostración. Sin embargo:
 - Las **premisas**: son añadidas permanentemente
 - Los **supuestos**: son incorporados temporalmente a T :
 - es **introducido** en un determinado punto de la demostración
 - luego, es **cancelado** (descargado) en otro punto posterior,
 - y como resultado una **nueva fórmula** queda demostrada.
- Lo que significa usar un supuesto es lo siguiente:
 - “supongamos que A ”
 - “entonces demuestro (usando A) que B ”
 - “en realidad acabo de mostrar que si tuviera A como premisa, entonces podría demostrar B ”
 - “eso equivale a decir que he demostrado la implicación $A \rightarrow B$ ”

Teorema de la Deducción

- Siendo $T[A_1, A_2, \dots, A_n]$ una teoría básica ampliada con un conjunto de n **premisas**, A_i , si la incorporación como **supuesto** de un fórmula A permite deducir otra fórmula B : entonces

$$T[A_1, A_2, \dots, A_n] \cup \{A\} \vdash B$$

entonces:

$$T[A_1, A_2, \dots, A_n] \vdash A \rightarrow B$$

Teorema de la deducción: En general, tanto para premisas como para supuestos:

$$T[A] \vdash B \text{ si y sólo si } T \vdash A \rightarrow B$$

Reglas de inferencia I

- En la definición de las reglas de inferencia vamos a usar A y B , que no son símbolos de proposición: son variables sobre fórmulas del lenguaje (**metavARIABLES**)
 - Mediante metavariables podemos razonar sobre conjuntos (infinitos) de fórmulas que comparten una misma forma lógica.
 - Por ejemplo: $A \wedge \neg A$ agrupa $p \wedge \neg p, q \wedge \neg q, \dots$
 - Cada regla de inferencia es una **metaregla** con infinitas instancias.

Deducción Natural: Reglas de inferencia: Conjunción

Introducción de \wedge (I_{\wedge})

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

$T[p, q] \vdash p \wedge q$	
1. p	premisa
2. q	premisa
3. $p \wedge q$	$I_{\wedge} (1,2)$

Eliminación de \wedge (E_{\wedge})

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

$T[p \wedge q] \vdash p$	
1. $p \wedge q$	premisa
2. p	$E_{\wedge} (1)$

$T[p \wedge q] \vdash q$	
1. $p \wedge q$	premisa
2. q	$E_{\wedge} (1)$

Deducción Natural: Reglas de inferencia: Disyunción

Eliminación de \vee (E_{\vee})

$$\frac{\begin{array}{l} A \vee B \\ A \rightarrow C \\ B \rightarrow C \end{array}}{C}$$

$\top[p \vee q, p \rightarrow \neg r, q \rightarrow \neg r] \vdash \neg r$	
1. $p \vee q$	premisa
2. $p \rightarrow \neg r$	premisa
3. $q \rightarrow \neg r$	premisa
4. $\neg r$	E_{\vee} (1,2,3)

Introducción de \vee (I_{\vee})

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

$\top[p] \vdash p \vee r$	
1. p	premisa
2. $p \vee r$	I_{\vee} (1)

$\top[p] \vdash r \vee p$	
1. p	premisa
2. $r \vee p$	I_{\vee} (1)

Deducción Natural: Ejemplo

- Prueba el ejemplo online



$$\top[s \wedge (p \vee q), p \rightarrow \neg r, q \rightarrow \neg r] \vdash s \wedge \neg r$$

- $s \wedge (p \vee q)$ premisa
- $p \vee q$ $E_{\wedge}(1)$
- $p \rightarrow \neg r$ premisa
- $q \rightarrow \neg r$ premisa
- $\neg r$ $E_{\vee}(2,3,4)$
- s $E_{\wedge}(1)$
- $s \wedge \neg r$ $I_{\wedge}(5,6)$

```

Prolog playground
1 %~~~~~ use_module('/draft.pl').
2 % Copyright (C)2022
3 % Name: deduccion.pl
4 % Author: Joaquin Arias
5 % Date: 15 August 2022
6 % Purpose: Execute Natural Deduction Proofs
7 %~~~~~
8
9 :- module(_,_)
10
11 :- op(200, fy, !).
12 :- op(400, xfy, [and, or, not, !]).
13 :- op(600, xfy, [=>, <=>]).
14
15 % Ejemplos
16 ejemplo1 :-
17     main(!s and p or q, p => !r, q => !r)
18
19 ejemplo2 :-
20     main(!p => q and !q, p, ['Premisa' (1),
21
22 ejemplo3 :-
23     main(!p => !r, !r => q, p, q, ['Premisa' (1)
24
25 ejemplo4 :-
26     main(!p => q, q => r, p => r, ['Premisa' (1)
27
28 ejemploMT :-
29     main(!r => (q and s), !(q and s)), !r,
30
31 ejemploSupuesto :- %% Falla porque no esta c
32     main(!s and p or q, p => !r, q => !r
33
34 :- data counter/1, formula/2, tabular/1, cer
35 main(Premisas, Deduccion, ReglasInferencia)
36 retractall(counter(_)), retractall(formu
    
```

Deducción Natural: Reglas de inferencia: Negación

Introducción \neg (I_¬)

$$\frac{A \rightarrow B \wedge \neg B}{\neg A}$$

Eliminación \neg (E_¬)

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

$\top[\neg p \rightarrow q \wedge \neg q] \vdash p$	
1. $\neg p \rightarrow q \wedge \neg q$	premisa
2. $\neg p$	I_¬ (1)
3. p	E_¬ (2)

Deducción Natural: Reglas de inferencia: Implicación

Eliminación \rightarrow (E_→)

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

$\top[p \rightarrow \neg r, \neg r \rightarrow q, p] \vdash q$	
1. $p \rightarrow \neg r$	premisa
2. p	premisa
3. $\neg r$	E_→ (1,2)
4. $\neg r \rightarrow q$	premisa
5. q	E_→ (3,4)

Introducción \rightarrow (I_→)

$$\frac{\begin{array}{l} A \text{ (supuesto)} \\ B \end{array}}{A \rightarrow B}$$

$\top[p \rightarrow q, q \rightarrow r] \vdash p \rightarrow r$	
1. $p \rightarrow q$	premisa
2. $q \rightarrow r$	premisa
3. p	supuesto
4. q	E_→ (1,3)
5. r	E_→ (2,4)
6. $p \rightarrow r$	I_→ (3,5)

Deducción Natural: Reglas de inferencia: Doble Implicación

Introducción \leftrightarrow (I_{\leftrightarrow})

$$\frac{A \rightarrow B \quad B \rightarrow A}{A \leftrightarrow B}$$

$\mathcal{T}[p \rightarrow \neg r, \neg r \rightarrow p] \vdash p \leftrightarrow \neg r$	
1. $p \rightarrow \neg r$	premisa
2. $\neg r \rightarrow p$	premisa
3. $p \leftrightarrow \neg r$	$I_{\leftrightarrow} (1,2)$

Eliminación \leftrightarrow (E_{\leftrightarrow})

$$\frac{A \leftrightarrow B}{A \rightarrow B} \quad \frac{A \leftrightarrow B}{B \rightarrow A}$$

$\mathcal{T}[p \leftrightarrow q \wedge r, p] \vdash r$	
1. $p \leftrightarrow q \wedge r$	premisa
2. $p \rightarrow q \wedge r$	$E_{\leftrightarrow} (1)$
3. p	premisa
4. $q \wedge r$	$E_{\rightarrow} (2,3)$
5. r	$E_{\wedge} (4)$

Ejercicios: Deducción Natural

- Demostrar los siguientes esquemas argumentales:

1. $\mathcal{T}[p] \vdash q \rightarrow p$
2. $\mathcal{T}[p \rightarrow (q \rightarrow r)] \vdash (p \rightarrow q) \rightarrow (p \rightarrow r)$
3. $\mathcal{T}[\neg p \rightarrow \neg q] \vdash (\neg p \rightarrow q) \rightarrow p$
4. $\mathcal{T}[\neg p \rightarrow \neg q] \vdash q \rightarrow p$
5. $\mathcal{T}[p \rightarrow q, q \rightarrow r] \vdash p \rightarrow r$
6. $\mathcal{T}[p \rightarrow (q \rightarrow r), q] \vdash p \rightarrow r$
7. $\mathcal{T}[p \wedge q \rightarrow r, r \wedge s \rightarrow t] \vdash p \wedge q \wedge s \rightarrow t$
8. $\mathcal{T}[p \wedge q \rightarrow r] \vdash p \wedge \neg r \rightarrow \neg q$
9. $\mathcal{T}[p \vee q \rightarrow r, s \rightarrow p] \vdash s \rightarrow r$
10. $\mathcal{T}[p \wedge q \rightarrow r, \neg(p \vee r) \rightarrow s, p \rightarrow q] \vdash \neg s \rightarrow r$
11. $\mathcal{T}[p \vee p] \vdash p$
12. $\mathcal{T}[p] \vdash \neg\neg p$
13. $\mathcal{T}[p \vee q \rightarrow r] \vdash q \rightarrow r$

Reglas de inferencia Derivadas

- En distintas demostraciones se repiten con frecuencia ciertos pasos:

$T[r \rightarrow (q \wedge s), \neg(q \wedge s)] \vdash \neg r$	
1. $r \rightarrow (q \wedge s)$	premisa
2. $\neg(q \wedge s)$	premisa
3. r	supuesto
4. $q \wedge s$	$E \rightarrow (1,3)$
5. $(q \wedge s) \wedge \neg(q \wedge s)$	$I \wedge (2,4)$
6. $r \rightarrow (q \wedge s) \wedge \neg(q \wedge s)$	$I \rightarrow (3,5)$
7. $\neg r$	$I \neg (6)$

$T[p \rightarrow q, r \wedge \neg q] \vdash r \wedge \neg p$	
1. $p \rightarrow q$	premisa
2. $r \wedge \neg q$	premisa
3. $\neg q$	$E \wedge (2)$
4. p	supuesto
5. q	$E \rightarrow (1,4)$
6. $q \wedge \neg q$	$I \wedge (3,5)$
7. $p \rightarrow q \wedge \neg q$	$I \rightarrow (4,6)$
8. $\neg p$	$I \neg (7)$
9. r	$E \wedge (2)$
10. $r \wedge \neg p$	$I \wedge (8,9)$

- Aunque son distintas las fórmulas que aparecen en estos dos ejemplos, las líneas destacadas tienen una estructura común
- Podríamos acortar las dos demostraciones si previamente demostramos con carácter general que $T[A \rightarrow B, \neg B] \vdash \neg A$, para cualesquiera fórmulas A y B .

- Las reglas derivadas se representan como las reglas básicas:

$$\frac{A \rightarrow B \quad \neg B}{\neg A} \quad \text{Modus Tollens (MT)}$$

- Las demostraciones anteriores quedarían ahora:

$T[r \rightarrow (q \wedge s), \neg(q \wedge s)] \vdash \neg r$	
1. $r \rightarrow (q \wedge s)$	premisa
2. $\neg(q \wedge s)$	premisa
3. $\neg r$	$MT (1,2)$

$T[p \rightarrow q, r \wedge \neg q] \vdash r \wedge \neg p$	
1. $p \rightarrow q$	premisa
2. $r \wedge \neg q$	premisa
3. $\neg q$	$E \wedge (2)$
4. $\neg p$	$MT (1,3)$
5. r	$E \wedge (2)$
6. $r \wedge \neg p$	$I \wedge (8,9)$

Reglas para la implicación

$T[A \rightarrow B, B \rightarrow C] \vdash A \rightarrow C$ Transitividad

$T[A \rightarrow B, \neg B] \vdash \neg A$ Modus Tollens

Reglas para la disyunción

$T[(A \vee B) \vee C] \vdash A \vee (B \vee C)$ Asociatividad

$T[A \vee B] \vdash B \vee A$ Conmutatividad

Reglas de Morgan

$T[\neg(A \wedge B)] \vdash \neg A \vee \neg B$ De Morgan

$T[\neg(A \vee B)] \vdash \neg A \wedge \neg B$ De Morgan

Reglas de corte

$$T[A \vee B, \neg A] \vdash B \quad \text{Corte}$$

$$T[A \vee B, \neg B] \vdash A \quad \text{Corte}$$

$$T[A \vee B, \neg A \vee C] \vdash B \vee C \quad \text{Corte}$$

Teorema de intercambio

- Sea:
 - A una fórmula.
 - B1 una sub-fórmula de A.

Si tenemos:

- $\vdash A$.
- $\vdash B1 \leftrightarrow B2$.

Entonces tenemos:

- $\vdash A'$

Donde A' resulta de sustituir en A todas o alguna aparición de $B1$ por $B2$.

- Ejemplo:

$$T[p \leftrightarrow r, q \rightarrow s, s \rightarrow t \wedge r] \vdash q \rightarrow p \wedge t$$

1. $q \rightarrow s$ premisa
2. $s \rightarrow t \wedge r$ premisa
3. q supuesto
4. s E_{\rightarrow} (1,3)
5. $t \wedge r$ E_{\rightarrow} (2,4)
6. $p \leftrightarrow r$ premisa
7. $t \wedge p$ **Intercambio** (5,6)
8. $p \wedge t$ conmutatividad (7)
9. $q \rightarrow p \wedge t$ I_{\rightarrow} (3,8)

Ejercicios: Deducción Natural

- Demostrar los siguientes formulas y/o esquemas argumentales:
 1. $T \vdash p \wedge q \rightarrow p$
 2. $T \vdash p \rightarrow (q \rightarrow p)$
 3. $T \vdash (p \vee q) \leftrightarrow (q \vee p)$
 4. $T \vdash (p \rightarrow q) \wedge (q \rightarrow r) \rightarrow (p \rightarrow r)$

5. $T \vdash \neg(p \vee q) \leftrightarrow \neg p \wedge \neg q$
6. $T \vdash (p \rightarrow \neg q) \wedge \neg(r \wedge \neg p) \rightarrow (q \rightarrow \neg r)$
7. $T \vdash (p \wedge q \rightarrow r) \rightarrow (p \rightarrow (q \rightarrow r))$
8. $T \vdash \neg(p \rightarrow q) \leftrightarrow p \wedge \neg q$
9. $T \vdash p \rightarrow (q \wedge \neg q \rightarrow \neg p)$
10. $T \vdash (p \rightarrow q) \rightarrow ((p \rightarrow r) \rightarrow (p \rightarrow q \wedge r))$

■ Demostrar los siguientes formulas y/o esquemas argumentales:

11. $T[p \rightarrow q, p] \vdash q$
12. $T[p] \vdash p \vee q$
13. $T[p \vee p] \vdash p$
14. $T[p \wedge (q \vee r)] \vdash p$
15. $T[p] \vdash \neg\neg p$
16. $T[p \vee q \rightarrow r] \vdash q \rightarrow r$
17. $T[p \rightarrow (q \rightarrow r)] \vdash q \rightarrow (p \rightarrow r)$
18. $T[p \rightarrow q] \vdash p \vee r \rightarrow q \vee r$
19. $T[\neg q \rightarrow r, t \rightarrow \neg q, \neg s \rightarrow \neg q] \vdash t \vee \neg s \rightarrow r$
20. $T[p \vee (q \rightarrow r) \rightarrow q, p] \vdash q$
21. $T[\neg p \rightarrow \neg s, \neg p \vee r, r \rightarrow \neg t] \vdash \neg s \vee \neg t$
22. $T[(p \rightarrow q) \wedge t, (r \vee p) \wedge \neg q, \neg t \leftrightarrow \neg s] \vdash r \wedge s$

Cápítulo 3

Lógica Primer Orden

3.1	Sintaxis, semántica y teoría interpretativa.	39
3.1.1	Sintaxis de la lógica de primer orden.	39
	Alfabeto	40
	Expresiones	41
	Ejercicios I: Formalización	42
	Variables	43
	Ejercicios II: Ligadura de variables	43
	Sustituciones	43
	Ejercicios III: Sustituciones	44
	Composición de Sustituciones	45
	Ejercicios IV: Composiciones	45
	Ejercicios V: Formalización de argumentos	46
3.1.2	Semántica de la lógica de primer orden	47
	Dominios interpretación	48
	Interpretación de Fórmulas	48
	Ejemplo Interpretación I	49
	Ejemplo Interpretación II	49
	Ejercicio	50
3.1.3	Satisfabilidad	51
	Formulas Abiertas	51
	Fórmulas Cerradas	51
	Modelos vs. Contramodelos	52
3.1.4	Teoría interpretativa de la lógica de primer orden.	52
	Validez y Consecuencia Lógica	52
	Ejemplo Validez I	53

	Ejemplo Validez II	53
	Ejemplo Consecuencia I y II	54
	Ejercicios Validez Argumentos	55
3.2	Teorema de la demostración y deducción natural.	56
3.2.1	Decibilidad de la lógica de Primer Orden.	56
3.2.2	Deducción Natural	57
	Deducción Natural: Reglas de inferencia	57
	Ejemplos: Deducción Natural	58
3.3	Tema 3.3:	59
3.3.1	Objetivo.	59
3.3.2	Forma Normal de Skolem	60
	1. Forma Prenex	60
	2. Cierre existencial	61
	3. Forma normal conjuntiva	61
	4. Eliminación de cuantificadores existenciales	61
3.3.3	Forma Clausular	62
	Forma Clausular: De una deducción	62
3.4	Método de resolución de Robinson.	62
3.4.1	Introducción	62
3.4.2	Interpretaciones de Herbrand	63
	Universo de Herbrand	63
	Base de Herbrand	64
	Definición	64
	Ejemplos: Interpretaciones de Herbrand	65
	Satisfabilidad	65
3.4.3	Teorema de Herbrand	65
3.4.4	Método de Resolución de Robinson	66
	Algoritmo	66
	Ejemplo: Método de Resolución de Robinson	67
3.4.5	Resolución con Unificador de Máxima Generalidad	68
	Sustitución	68
	Composición sustituciones	68
	Unificadores y UMG	69
	Algoritmo de Unificación	69
	Definición	70
	Ejemplo: Resolución con UMG	71

3.4.6	Estrategias de resolución	71
	Motivación	71
	Ejemplos y propiedades	72
	Resolución lineal	72

3.1. Sintaxis, semántica y teoría interpretativa.

3.1.1. Sintaxis de la lógica de primer orden.

- Si este fin de semana nieva y Juan ha terminado el trabajo, iré a esquiar.

Nieva
Juan ha terminado el trabajo
Juan irá a esquiar

- Si este fin de semana nieva y Juan ha terminado el trabajo, iré a esquiar.
Si este fin de semana nieva y Paco ha terminado el trabajo, iré a esquiar.

Nieva
Juan ha terminado el trabajo
Paco ha terminado el trabajo
Juan y Paco irán a esquiar

- Si este fin de semana nieva, los que terminan su trabajo van a esquiar

Nieva
Juan ha terminado el trabajo
Paco ha terminado el trabajo
Nacho ha terminado el trabajo
Juan, Paco y Nacho irán a esquiar

- Cada vez que nieva, los que terminan su trabajo van a esquiar

Nieva este fin de semana
Va a nevar el próximo fin de semana
Juan ha terminado el trabajo esta semana
Paco ha terminado el trabajo esta semana
Juan y Paco irán a esquiar este fin de semana
y (si terminan su trabajo) también el siguiente

- La deducción clásica por excelencia:

Si Sócrates es un hombre entonces Sócrates es mortal
Sócrates es un hombre
 Sócrates es mortal

Todos los hombres son mortales
Sócrates es un hombre
 Sócrates es mortal

- Las proposiciones se refieren a individuos:
 - Un empleado de los de la empresa.
 - Un fin de semana del año.
- Si tenemos que hablar demás de un individuo (para expresar un hecho más general), las proposiciones se nos quedan cortas.
- Podríamos multiplicar las proposiciones, una para cada individuo y hecho lógico que se refiere a él, pero:
 - Esto lo haría todo mucho mas largo y pesado...
 - ... o imposible. En campos como las matemáticas se podría requerir hablar de conjuntos infinitos.
- Necesitamos ampliar la capacidad expresiva de los lenguajes proposicionales con nuevos símbolos.

Alfabeto

- Un alfabeto de un LPO se define con los siguientes tipos de símbolos:
 - Símbolos de constante: $a, b, c, \dots, a_1, a_2, \dots$
 - Símbolos de variable: $x, y, z, \dots, x_1, x_2, \dots$
 - Símbolos de función: $f(_), g(_, _), \dots$ $o f^1, g^2, o f/1, g/2$
 - Símbolos de predicado: $P(_), Q(_), \dots$ $o P^1, Q^2, o P/1, Q/2$
 - Conectivas lógicas: $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$
 - Cuantificadores: \exists, \forall
- Un símbolo de predicado 0-ario es una proposición
- En un LPO no puede haber símbolos comunes entre los conjuntos de variables y constantes, ni entre los símbolos de función y de predicado
- Símbolos auxiliares de puntuación: paréntesis y comas
 - Innecesarios si la aridad es parte del nombre del símbolo de predicado/función
 - Mejoran la legibilidad: $P(a, f(x, g(b)))$ en lugar de $P^2 a f^2 x g^1 b$
- El uso de los paréntesis se puede reducir al mínimo mediante las convenciones de precedencia:

- $\{\forall, \exists, \neg\}$ tienen mayor precedencia que $\{\wedge, \vee\}$
- $\{\wedge, \vee\}$ tienen mayor precedencia que $\{\rightarrow, \leftrightarrow\}$

Expresiones

- Una expresión de un LPO es cualquier concatenación finita de símbolos de su alfabeto. Las expresiones relevantes son:
 - Términos:
 - Un símbolo de constante es un término.
 - Un símbolo de variable es un término.
 - Si f es un símbolo de función n -aria y t_1, \dots, t_n son términos entonces $f(t_1, \dots, t_n)$ es un término.
 - Fórmulas (Bien Formadas, FBF):
 - Si P es un símbolo de predicado n -ario y t_1, \dots, t_n son términos entonces $P(t_1, \dots, t_n)$ es una fórmula (átomo o fórmula atómica).
 - Si F y G son fórmulas entonces también lo son $\neg F, F \wedge G, F \vee G, F \rightarrow G$ y $F \leftrightarrow G$.
 - Si $F(x)$ es una fórmula en la que x es una variable libre entonces $\forall x F(x)$ y $\exists x F(x)$ son fórmulas.
 - Ejemplos de términos:
 - Constantes: $a, b, c, 1, spot, john$
 - Funciones: $f/1, g/3, h/2, +/3$
 - Variables: x, y, m

Correctas: $spot, f(john), f(x), +(1, 2, 3), +(x, y, m), h(f(h(1, 2))), m$

Incorrectas: $spot(x), +(1, 2), g, f(f(h))$

- Ejemplos de fórmulas:
 - Términos correctos...
 - Predicados: $dog/1, p/2, q/0, r/0, barks/1$

Correctas: $q, q \rightarrow r, barks(x) \rightarrow dog(x), p(x, y), \exists x(dog(x) \wedge barks(x) \wedge \neg q), \exists y(dog(y) \rightarrow barks(y)), dog(x),$

Incorrectas: $q \vee, \exists p$

Ejercicios I: Formalización

- Seleccionar los predicados, constantes y funciones necesarios para definir un LPO en el que formalizar las siguientes oraciones:
 1. Vicente es mejicano.
 2. Mi casa es roja.
 3. Luisa y María son brasileñas pero Vicente es mejicano.
 4. Jorge adora a Juan.
 5. Jorge adora a su hermano Juan.
 6. Juan ama a Rosa pero ella no le corresponde.
 7. Pedro sujetó a Juan y María le atizó.
 8. Homero escribió la Ilíada y la Odisea.
 9. Nieves se peina a sí misma y también peina a Juan.
 10. Titán es satélite de Saturno pero Europa no lo es.

- Seleccionar los predicados, constantes y funciones necesarios para definir un LPO en el que formalizar las siguientes oraciones:
 11. O Pedro o María (pero no ambos) son hermanos míos.
 12. Si Colón descubrió América, merece un lugar en la Historia.
 13. El asesino de mi padre es Juan o Pedro, pero no Alberto.
 14. María ama a mi padre mientras que Julia me ama a mí.
 15. Cela leía a Borges aunque éste lo detestaba públicamente.
 16. María está enamorada de alguien.
 17. Hay al menos un número primo.
 18. Algunas cantantes de ópera no están gordas.
 19. Cualquier crimen será castigado.
 20. No todos los crímenes merecen la pena capital.

- Seleccionar los predicados, constantes y funciones necesarios para definir un LPO en el que formalizar las siguientes oraciones:
 21. Las novelas de Cela me fascinan.
 22. Hay profesores que no saben explicar.
 23. Sólo los suecos entienden a Bergman.
 24. Todo ciudadano tiene derecho a una vivienda.
 25. Hay genios, pero no todos los poetas lo son.
 26. No todos los satélites de Júpiter tienen atmósfera.
 27. Todos los estudiantes de tercer curso ayudan a al menos uno de primero.
 28. Los caballeros las prefieren rubias pero se casan con las morenas.
 29. Nadie respeta a quien no se respeta a sí mismo.

30. Hay un pintor a quien todo el mundo admira.

Variables

- **Ámbito (alcance) de un cuantificador en una fórmula:**
 - La subfórmula inmediatamente a su derecha (puede ser necesario el uso de paréntesis para indicar el alcance del cuantificador):

$$\begin{array}{ll} \exists x P(\mathbf{x}, y) \vee Q(x, y); & \exists x (P(\mathbf{x}, y) \vee Q(\mathbf{x}, y)); \\ \forall y \exists x P(\mathbf{x}, y) \vee Q(x, y); & \forall y \exists x (P(\mathbf{x}, y) \vee Q(\mathbf{x}, y)) \end{array}$$
- **Variables libres y ligadas en fórmulas de un LPO. Una variable x :**
 - Se encuentra ligada en una fórmula A cuando está en el ámbito de un cuantificador $\exists x$ o $\forall x$ en A .
 - Se encuentra libre en una fórmula cuando no se encuentra ligada.

NOTA: Un mismo símbolo de variable puede estar libre y ligado en una misma fórmula.
- **Fórmulas abiertas y cerradas en un LPO:**
 - Es abierta cuando contiene al menos una variable libre.
 - Es cerrada cuando todas sus variables están ligadas.

Ejercicios II: Ligadura de variables

- Señalar las variables libres y ligadas en las siguientes fórmulas:
 1. $\exists x (P(x, f(y)) \rightarrow \exists y Q(x, y))$
 2. $\exists x P(x) \rightarrow \forall y Q(x, f(y))$
 3. $\exists x \exists y (P(x, y) \vee Q(x, y)) \wedge R(a, y)$
 4. $\exists x \exists y ((P(x, y) \vee Q(x, y)) \wedge R(x, y))$
 5. $\forall x (x = y \rightarrow \exists z P(x, z))$
 6. $\exists x \forall y P(x, f(x, y)) \rightarrow \exists y Q(x, y)$
 7. $x = y + z \rightarrow x \leq y + z$
 8. $\forall x (x + 0 = x)$
 9. $\forall x (N(x) \rightarrow N(s(x)))$
 10. $\forall x \exists y (P(g(x, a), y) \vee Q(x) \vee R(z, b)) \wedge \exists z S(x, y, z)$

Sustituciones

- Una **sustitución** es una función finita de un conjunto de variables de un lenguaje en el de términos. Se representa como $\{x_1/t_1, x_2/t_2, \dots, x_n/t_n\}$ donde x_1, \dots, x_n

son variables diferentes y t_1, \dots, t_n son términos.

- Cada par x_i/t_i se denomina **ligadura**.
- Una sustitución que no sustituye ninguna variable se llama **sustitución vacía** (\emptyset)
- Dada una fórmula A y una sustitución $\alpha = \{x_1/t_1, \dots, x_n/t_n\}$, se denomina **aplicación de α a A** ($A\alpha$) a la fórmula obtenida reemplazando simultáneamente cada ocurrencia en A de x_i por t_i , para cada $x_i/t_i \in \alpha$.
 - $\alpha = \{x/f(a), y/x, z/h(b, y), w/a\}$
 - $P(x, y, z)\alpha = P(f(a), f(a), h(b, f(a)))$ incorrecto
 - $P(x, y, z)\alpha = P(f(a), x, h(b, y))$ correcto
- Notación: Siendo A una fórmula y x una variable de un LPO:
 - $A(x)$ indica la aparición de al menos una ocurrencia **libre** de x en A
 - $A\{x/t\}$ representa la fórmula obtenida a partir de A sustituyendo **todas** las apariciones de la **variable libre** x por el término t .
- Ejemplos:

$$A(x) : P(\mathbf{x}, f(y)) \rightarrow \exists y Q(\mathbf{x}, y) \qquad A\{x/a\} : P(\mathbf{a}, f(y)) \rightarrow \exists y Q(\mathbf{a}, y)$$

$$A(y) : \exists x((P(x, \mathbf{y}) \vee Q(x, \mathbf{y})) \wedge R(x, \mathbf{y}))$$

$$A\{y/f(z)\} : \exists x((P(x, \mathbf{f}(\mathbf{z})) \vee Q(x, \mathbf{f}(\mathbf{z}))) \wedge R(x, \mathbf{f}(\mathbf{z})))$$

- Condiciones para la aplicación de una sustitución a una fórmula A :
 - Se aplica **única y exclusivamente** sobre variables **libres** presentes en A . De no haberlas, la sustitución no tiene ningún efecto sobre la expresión inicial.
 - Su aplicación reemplaza **todas y sólo** las ocurrencias de la variable libre en la fórmula por el término.

Errores comunes:

$$(\forall x(P(x, f(y)) \rightarrow \exists y Q(x, y)))\{y/a\} : \qquad \exists x(P(x, f(a)) \rightarrow \exists y Q(x, y))$$

$$(\exists x(P(x, f(y)) \rightarrow \exists y Q(x, y)))\{y/a\} : \qquad \exists x(P(x, f(a)) \rightarrow Q(x, a))$$

$$(\exists x(P(x, f(y)) \wedge Q(x, y))\{y/a\} : \qquad \exists x(P(x, f(a)) \rightarrow Q(x, y))$$
 - Una sustitución α **no** se puede aplicar a A si pasa lo siguiente:
 - (i) α contiene una ligadura x/t y t contiene la variable y .
 - (ii) hay una ocurrencia de x en A que se puede reemplazar por t
 - (iii) dicha ocurrencia está dentro del ámbito de un cuantificador sobre y .

$$\exists y(\forall z P(x, z) \wedge Q(y))\{x/f(y)\} : \qquad \exists y(\forall z P(f(y), z) \wedge Q(y))$$

Ejercicios III: Sustituciones

- Realizar las siguientes sustituciones:
 1. $(\exists x(P(x, f(y)) \rightarrow \exists y Q(x, y)))\{y/g(z)\}$
 2. $(\forall x \forall y(P(x, y) \rightarrow Q(x, y)))\{y/a\}$

3. $(\forall x(\forall y(P(x,y) \rightarrow Q(x,y))))\{y/a\}$
4. $(\exists x(\forall y(P(x,y) \vee Q(x,y)) \wedge R(x,y))\{y/b\}$
5. $(\exists x(\forall y(P(x,y) \vee Q(x,y)) \wedge R(x,y))\{x/b\}$
6. $(\exists x\forall y(P(x,y) \vee Q(x,y)) \wedge R(x,y))\{x/a, y/b\}$
7. $(\forall x(P(x,y) \rightarrow Q(x,y))\{y/f(x,a)\}$
8. $(\forall y(P(x,y) \rightarrow \forall xQ(x,y))\{y/f(x,a)\}$
9. $(x = y + z \rightarrow x \leq y + z)\{x/1, y/2\}$
10. $(x = y + z \rightarrow x \leq y + z)\{x/s(x)\}$
11. $(x = y + z \rightarrow x \leq y + z)\{x/s(y)\}$
12. $(\forall x(x + 0 = x))\{x/1\}$

Composición de Sustituciones

- Dadas dos sustituciones $\alpha = \{x_1/t_1, \dots, x_n/t_n\}$ y $\beta = \{y_1/s_1, \dots, y_m/s_m\}$ su **composición** $\alpha\beta$ se define eliminando del conjunto $\{x_1/t_1\beta, \dots, x_n/t_n\beta, y_1/s_1, \dots, y_m/s_m\}$
 - las ligaduras $x_i/t_i\beta$ tales que $x_i \equiv t_i\beta$,
 - y las ligaduras y_i/s_i tales que $y_i \in \{x_1, \dots, x_n\}$
- Ejemplo:
 - si $\alpha = \{x/3, y/f(x, 1)\}$ y $\beta = \{x/4\}$ entonces $\alpha\beta = \{x/3, y/f(4, 1)\}$ y $\beta\alpha = \{x/4, y/f(x, 1)\}$
- Propiedades de la composición:
 - $(F\alpha)\beta = F(\alpha\beta)$
 - $(\alpha\beta)\gamma = \alpha(\beta\gamma)$
 - $\alpha\lambda = \lambda\alpha = \alpha$
 - $\alpha\beta \neq \beta\alpha$

Ejercicios IV: Composiciones

- Realizar la composición de las siguientes sustituciones:
 1. $\{x/a, y/g(z), z/f(x)\} \{x/b, z/y\}$
 2. $\{x/b, z/y\} \{x/a, y/g(z), z/f(x)\}$
 3. $\{x/f(a, y), y/b\} \{x/a, y/c\}$
 4. $\{x/a, y/c\} \{x/f(a, y), y/b\}$
 5. $\{x/y, y/z, z/w\} \{x/a, y/b, z/c, w/d\}$
 6. $\{x/a, y/b, z/c, w/d\} \{x/y, y/z, z/w\}$
 7. $\{x/f(z), y/w\} \{z/a, w/f(y, z)\}$

$$8. \{z/a, w/f(y, z)\} \{x/f(z), y/w\}$$

Ejercicios V: Formalización de argumentos

- Definir LPOs en los que formalizar los siguientes argumentos:
 1. La Tierra orbita en torno al Sol. La Luna orbita en torno a la Tierra. Todo cuerpo que orbita en torno al Sol es un planeta. Son satélites los cuerpos que orbita en torno a planetas. Luego la Tierra es un planeta y la Luna un satélite.
 2. Cero es un número natural. El sucesor de un número natural es también un número natural. Uno es sucesor de cero. Luego uno es un número natural.
 3. Cero es un número natural. El sucesor de un número natural es también un número natural. La suma de cualquier número natural y cero es igual a ese mismo número. La suma de un número y el sucesor de otro es igual al sucesor de la suma del primero más el antecesor del segundo.
- Definir LPOs en los que formalizar los siguientes argumentos:
 4. Aquel que no existe no puede engañarse. Yo me engaño. Luego yo existo. (A. Deaño)
 5. Solamente las personas bien educadas están suscritas al Times. Ningún puercoespín sabe leer. Las personas bien educadas saben leer. Luego ningún puercoespín está suscrito al Times. (L. Carroll)
 6. Todos los filósofos se han preguntado qué es la Filosofía. Todos los que se han preguntado qué es la Filosofía han dado en la locura. Nietzsche es un filósofo. El Padre Ceballos no acabó loco. Luego Nietzsche y el Padre Ceballos no son la misma persona. (A. Deaño)
- Definir LPOs en los que formalizar los siguientes argumentos:
 7. Todos los libros de texto son tediosos Algunos libros de texto están llenos de ejercicios. Luego algunos libros llenos de ejercicios son tediosos
 8. Todos los políticos tienen algo que ocultar Algunos políticos salen en televisión. Luego hay quienes tienen algo que ocultar y salen en televisión
 9. Todos los chimpancés son primates Sara es un chimpancé. Luego Sara es un primate
 10. Algunos primates no son chimpancés Algunos chimpancés saben hablar. Luego algunos primates no saben hablar
 11. Ningún individuo que no sea chimpancé es un primate Ninguno que no sea primate es inteligente Sara no es un chimpancé. Luego Sara no es inteligente
- Definir LPOs en los que formalizar los siguientes argumentos:
 12. Hay individuos inteligentes o que saben hablar. Juan no sabe hablar. Luego

Juan no es inteligente.

13. Todo elemento químico es oxidante o reductor. El carbono es un elemento químico no oxidante. Luego el carbono es reductor.
 14. No todos los seres humanos saben hablar o son inteligentes. Sara es un ser humano pero no sabe hablar. En consecuencia, Sara es inteligente.
 15. Todos los chimpancés saben hablar. Algunos primates no saben hablar. Algunos primates son humanos. Por tanto, algunos seres humanos son chimpancés y saben hablar.
- Definir LPOs en los que formalizar los siguientes argumentos:
 16. Todos los chimpancés son primates. Algunos seres humanos son inteligentes. Algunos primates son seres humanos. Juan es un chimpancé y Sara es un ser humano que sabe hablar. Así pues, Juan es un primate y Sara es inteligente.
 17. Todos los rinocerontes tienen un cuerno. Todos y sólo los rinocerontes son dignos de ser cazados. Luego todos los animales dignos de ser cazados tienen un cuerno.
 18. Todas las selvas tropicales tienen color verde. Nada que tenga color verde esta seco. Por tanto, ninguna selva tropical esta seca.
 19. Ningún fotógrafo es pintor. Todo aquel que no es fotógrafo es buen dibujante. Así pues, todos los pintores son buenos dibujantes.

3.1.2. Semántica de la lógica de primer orden

- Dado un LPO, definir de modo preciso el significado de sus fórmulas.
- Conceptos semánticos empleados en semántica formal:
 - **Dominio** de interpretación **D**: conjunto no vacío de objetos
 - Relaciones n-arias: subconjuntos de Dominio^n
 - Funciones n-arias: n-tuplas de obj. del dominio \rightarrow obj. del dominio
 - **Función** de interpretación $i()$:
 - Fórmulas $\rightarrow \{V, F\}$
 - Términos \rightarrow objetos del dominio
 - Predicados y funciones \rightarrow relaciones y funciones en obj. del dominio
 - **Interpretación I**: $\langle D, i() \rangle$
 - Un dominio no vacío de individuos, D
 - Una función $i()$ de individuos de D , funciones y relaciones sobre D a todas las constantes, funciones y predicados del LPO.

Dominios interpretación

- Las expresiones de un lenguaje sólo significan algo cuando se refieren o hablan de algo. Esto es el dominio o universo de discurso.
- Punto de partida: elección de un dominio no vacío de interpretación:
 - $D = \{Sol, Tierra, Luna\}$; $D = \{1, 2, 3, 4, 5, \dots\}$; $D = \{\diamond, \circ, \square\}$
- A continuación, se define la función de interpretación para el lenguaje L :
 - Para toda constante $a \in L : i(a) = d, d \in D$
 - Todo individuo de D debe tener un nombre (distinto) en L
 - Si L no tuviera suficientes constantes, se amplía con las constantes necesarias y se denomina $L(D)$
 - Para toda función n -aria $f \in L : i(f) = f_D$
 - $f_D : \langle d_1, \dots, d_n \rangle \rightarrow d$ (aplicación de D^n en D)
 - Para todo predicado n -ario $P \in L : i(P) = P_D$
 - $P_D : \langle d_1, \dots, d_n \rangle \rightarrow \{V, F\}$ (aplicación de D^n en $\{V, F\}$)

Interpretación de Fórmulas

- Dada una interpretación $I = \langle D, i() \rangle$ para un lenguaje de primer orden L :
 - Asignar significado a las fórmulas de L implica:
 - Asignar significado a las fórmulas atómicas
 - Asignar significado al resto de fórmulas mediante inducción
 - Asignación de valor de verdad a las fórmulas atómicas de L :
 - $i(P(t_1, \dots, t_n)) = V/F$ sii $P_D(i(t_1), \dots, i(t_n)) = V/F$
 - Cada interpretación concreta asignará un y sólo un valor de verdad a cada fórmula atómica de L . Dos interpretaciones sobre un mismo dominio y para un mismo lenguaje difieren entre sí en el valor de verdad que asignan.
 - En el caso del predicado $=$, su semántica es fija, sin variación entre interpretaciones:
 - $i(t_1 = t_2) = V/F$ sii $i(t_1)$ es idéntico a / no es idéntico a $i(t_2)$
- Asignación de valor de verdad a fórmulas:

$$\begin{array}{ll}
i(\neg A) = V & \text{sii } i(A) = F \\
i(A \wedge B) = V & \text{sii } i(A) = i(B) = V \\
i(A \vee B) = F & \text{sii } i(A) = i(B) = F \\
i(A \rightarrow B) = F & \text{sii } i(A) = V \text{ y } i(B) = F \\
i(A \leftrightarrow B) = V & \text{sii } i(A) = i(B) \\
i(\exists xA) = V & \text{sii } i(Ax/a) = V \text{ para al menos una constante } a \text{ de } L(D) \\
i(\forall xA) = V & \text{sii } i(Ax/a) = V \text{ para toda constante } a \text{ de } L(D)
\end{array}$$

Ejemplo Interpretación I

Ejemplo I:

$$\forall x(M(a,x) \wedge P(x)) \rightarrow \neg \exists y Q(y)$$

- Construimos una interpretación sobre el dominio de los números naturales:
$$D = \{0, 1, 2, 3, 4, \dots\}$$
- El lenguaje de la fórmula sólo tiene un símbolo de constante (a), por lo que ampliamos el conjunto de constantes y definimos $L(D)$:
$$Ctes. = \{a, a_1, a_2, a_3, a_4, \dots\}$$
- La función de interpretación i podría ser:
 - $i(a) = 0, i(a_1) = 1, i(a_2) = 2, i(a_3) = 3, \dots$
 - $P_D(x)$: x es par; $Q_D(x)$: x es impar; $M_D(x,y)$: $x < y$

Bajo esta interpretación: $i(\forall x(M(a,x) \wedge P(x)) \rightarrow \neg \exists y Q(y)) = V$

- $i(\forall x(M(a,x) \wedge P(x))) = F$ porque:
 - $i((M(a,x) \wedge P(x))\{x/a\}) = i(M(a,a) \wedge P(a)) =$
 $M_D(i(a), i(a)) \wedge P_D(i(a)) = M_D(0,0) \wedge P_D(0) = F$
- $i(\neg \exists y Q(y)) = F$ porque:
 - $i(\exists y Q(y)) = V$ porque:
 - $i(Q(y)\{y/a_1\}) = i(Q(a_1)) = Q_D(i(a_1)) = Q_D(1) = V$

Ejemplo Interpretación II

Ejemplo II:

$$\forall x(M(a,x) \wedge P(x)) \rightarrow \neg \exists y Q(y)$$

- Construimos una interpretación sobre un dominio finito:
$$D = \{\circ, \square, \blacklozenge\}$$
- El lenguaje de la fórmula sólo tiene un símbolo de constante (a), por lo que ampliamos el conjunto de constantes y definimos $L(D)$:
$$Ctes. = \{a, b, c\}$$

- La función de interpretación i podría ser:

- $i(a) = \bigcirc, i(b) = \square, i(c) = \blacklozenge$

P_D	
\bigcirc	V
\square	V
\blacklozenge	V

Q_D	
\bigcirc	V
\square	F
\blacklozenge	V

M_D			
\bigcirc	V	V	V
\square	F	F	V
\blacklozenge	V	F	F

Bajo esta interpretación:

$$i(\forall x(M(a,x) \wedge P(x)) \rightarrow \neg \exists y Q(y)) = F$$

- $i(\forall x(M(a,x) \wedge P(x))) = V$ porque:

- $i((M(a,x) \wedge P(x))\{x/a\}) = M_D(i(a), i(a)) \wedge P_D(i(a)) = M_D(\bigcirc, \bigcirc) \wedge P_D(\bigcirc) = V$
- $i((M(a,x) \wedge P(x))\{x/b\}) = M_D(i(a), i(b)) \wedge P_D(i(b)) = M_D(\bigcirc, \square) \wedge P_D(\square) = V$
- $i((M(a,x) \wedge P(x))\{x/c\}) = M_D(i(a), i(c)) \wedge P_D(i(c)) = M_D(\bigcirc, \blacklozenge) \wedge P_D(\blacklozenge) = V$

- $i(\neg \exists y Q(y)) = F$ porque:

- $i(\exists y Q(y)) = V$ porque:
 - $i(Q(y)\{y/a\}) = i(Q(a)) = Q_D(i(a)) = Q_D(\bigcirc) = V$

Ejercicio

Ejercicio: Aritmética de Peano

- Dadas las fórmulas:

$$\{N(a), \forall x(N(x) \rightarrow N(s(x))), \forall x(N(x) \rightarrow x + a = x),$$

$$\forall x \forall y(N(x) \wedge N(y) \rightarrow x + s(y) = s(x + y))\}$$

interpretarlas en el dominio de los números naturales (con el 0), siendo $N(_)$ la propiedad de ser un número natural, $= (_, _)$ la relación de identidad y $s(_)$ y $+(_, _)$ las funciones sucesor y suma, respectivamente.

3.1.3. Satisfabilidad

Formulas Abiertas

- Hemos visto como establecer claramente el significado de una fórmula cuando ésta es cerrada, pero ¿cómo interpretar una fórmula abierta?
- Ejemplos de fórmulas abiertas:
 1. Alguien es el rey de Canadá: $R(x, a)$
 2. Algo es igual a dos: $x = 2$
 3. Alguien va a morir: $M(x)$
- ¿Que significado tienen estas fórmulas abiertas?
 1. $R(x, a)$ es **falsa** pues ninguna sustitución de x produce una afirmación verdadera: ninguna persona es rey de Canadá.
 2. $x = 2$ es una fórmula que **puede ser verdadera** si sustituimos x por el número 2, mientras que será falsa si elegimos otro número.
 3. $M(x)$ es una fórmula que **siempre es verdadera** para cualquier nombre de persona que sustituya a x .
- Sea A una fórmula abierta de L (LPO):
 - A es **satisfacible** cuando puede ser verdadera:
 - Hay una interpretación $\langle D, i(\cdot) \rangle$ y una sustitución $\theta = \{x_1/c_1, \dots, x_n/c_n\}$ de todas sus variables libres por constantes de $L(D)$ que la hacen verdadera
(existe una $\langle D, i(\cdot) \rangle$ y una θ tal que $i(A\theta) = V$)
 - A es **verdadera** en una interpretación $\langle D, i(\cdot) \rangle$ cuando:
 - Toda sustitución θ de sus variables libres por constantes de $L(D)$ la hacen verdadera
($i(A\theta) = V$ para toda θ)
 - A es **válida** cuando es verdadera en toda interpretación
 - $i(A\theta) = V$ para toda $\langle D, i(\cdot) \rangle$ y toda θ

Fórmulas Cerradas

- Dada una fórmula abierta $A(x_1, \dots, x_n)$, definimos:
 - **Cierre existencial** de $A(x_1, \dots, x_n)$: $\exists x_1, \dots, x_n A$
 - **Cierre universal** de $A(x_1, \dots, x_n)$: $\forall x_1, \dots, x_n A$

Relaciones semánticas entre fórmulas abiertas y cerradas:

- $A(x_1, \dots, x_n)$ es satisfacible sii $\exists x_1, \dots, x_n A$ es satisfacible.

- $A(x_1, \dots, x_n)$ es insatisfacible sii $\exists x_1, \dots, x_n A$ es insatisfacible.
 - $A(x_1, \dots, x_n)$ es verdadera en I sii $\forall x_1, \dots, x_n A$ es verdadera en I .
 - $A(x_1, \dots, x_n)$ es válida sii $\forall x_1, \dots, x_n A$ es válida.
- Satisfabilidad de una fórmula:
 - Una interpretación $\langle D, i() \rangle$ **satisface** una fórmula A sii $i(A) = V$
 - Una fórmula A es **satisfacible** sii existe al menos una interpretación $\langle D, i() \rangle$ tal que $i(A) = V$
 - Una fórmula A es **insatisfacible** sii no existe ninguna interpretación $\langle D, i() \rangle$ tal que $i(A) = V$
 - Extensión a conjuntos de fórmulas $\{A_1, \dots, A_n\}$:
 - Una interpretación $\langle D, i() \rangle$ **satisface** $\{A_1, \dots, A_n\}$ sii $i(A_i) = V$ para todo $i : 1 \leq i \leq n$
 - $\{A_1, \dots, A_n\}$ es **satisfacible** sii existe al menos una interpretación $\langle D, i() \rangle$ tal que $i(A_i) = V$ para todo $i : 1 \leq i \leq n$
 - $\{A_1, \dots, A_n\}$ es **insatisfacible** sii no existe ninguna interpretación $\langle D, i() \rangle$ tal que $i(A_i) = V$ para algún $i : 1 \leq i \leq n$
 - Una fórmula A es **válida** sii $i(A) = V$ para toda interpretación $\langle D, i() \rangle$

Modelos vs. Contramodelos

- **Modelo:** una interpretación I es modelo de una fórmula A sii
 - $i(A) = V$ cuando A es una fórmula cerrada
 - $i(A\theta) = V$ para toda sustitución θ de todas sus variables libres, cuando A es una fórmula abierta
- **Contramodelo:** una interpretación I es contramodelo de A sii
 - $i(A) = F$ cuando A es una fórmula cerrada
 - $i(A\theta) = F$ para alguna sustitución θ de todas sus variables libres, cuando A es una fórmula abierta

3.1.4. Teoría interpretativa de la lógica de primer orden.

Validez y Consecuencia Lógica

- **Validez lógica**
 - Una fórmula $A \in L$ es válida (lógicamente válida) sii
es verdadera en toda interpretación: $\models A$
- **Consecuencia lógica**

- Dado un conjunto de fórmulas $\Gamma = \{A_1, \dots, A_n\}$ ($A_i \in L$) y una fórmula $B \in L$, B es consecuencia lógica de Γ ($\Gamma \models B$) sii:
 - Todo modelo de Γ es también modelo de B

(toda interpretación que haga verdad a Γ también hace verdad a B)
 - No existe ninguna interpretación que haga verdad a Γ y que no haga verdad a B

Ejemplo Validez I

Ejemplo Validez I:

$$\models \exists x \forall y P(x, y) \rightarrow \forall y \exists x P(x, y)$$

Considerando $D = \{1, 2\}$, $L(D) = \{P/2\} \cup \{a, b\}$

La fórmula es **válida**.

- No hay forma de encontrar contramodelos. Justificación:
 1. $i(\exists x \forall y P(x, y) \rightarrow \forall y \exists x P(x, y)) = F$ sii $i(\exists x \forall y P(x, y)) = V$ sii
 - $i(\forall y P(a, y)) = V$ sii $i(P(a, a)) = V$ y $i(P(a, b)) = V$
 - o bien $i(\forall y P(b, y)) = V$ sii $i(P(b, a)) = V$ y $i(P(b, b)) = V$
 2. y $i(\forall y \exists x P(x, y)) = F$ sii
 - $i(\exists x P(x, a)) = F$ sii $i(P(a, a)) = F$ y $i(P(b, a)) = F$
 - o bien $i(\exists x P(x, b)) = F$ sii $i(P(a, b)) = F$ y $i(P(b, b)) = F$
- La imposibilidad de encontrar contramodelos NO es exclusiva de esta interpretación. Verificar el antecedente es incompatible con hacer falso el consecuente.

Ejemplo Validez II

Ejemplo Validez II:

$$\models \forall y \exists x P(x, y) \rightarrow \exists x \forall y P(x, y)$$

Considerando $D = \{1, 2\}$, $L(D) = \{P/2\} \cup \{a, b\}$

La fórmula **NO es válida**.

- Porque la siguiente interpretación es un contramodelo:
 - $i(a) = 1$
 - $i(b) = 2$
 - $P_D(1, 1) = V$, $P_D(1, 2) = F$, $P_D(2, 1) = F$, $P_D(2, 2) = V$
- Justificación:
 1. $i(\forall y \exists x P(x, y)) = V$ porque:
 - a) $i(P(a, a)) = V$ porque $P_D(i(a), i(a)) = P_D(1, 1) = V$

- b) y $i(P(b,b)) = V$ porque $P_D(i(b),i(b)) = P_D(2,2) = V$
2. $i(\exists x\forall yP(x,y)) = F$ porque:
- a) $i(P(a,b)) = F$ porque $P_D(i(a),i(b)) = P_D(1,2) = F$
- b) y $i(P(b,a)) = F$ porque $P_D(i(b),i(a)) = P_D(2,1) = F$

Ejemplo Consecuencia I y II

Ejemplo Consecuencia Lógica I

- Sea el argumento: Hay individuos inteligentes o que saben hablar. Juan no sabe hablar. Luego Juan no es inteligente.
- Y su formalización: $\{\exists x(I(x) \vee H(x)), \neg H(a)\} \models \neg I(a)$
- Para determinar la corrección de este argumento (si hay consecuencia lógica por medios semánticos, intentamos encontrar un contramodelo, es decir una interpretación que:
 1. Verifique las premisas: $i(\exists x(I(x) \vee H(x))) = V$ y $i(\neg H(a)) = V$
 2. Haga falsa la pretendida conclusión: $i(\neg I(a)) = F$
- Elijamos para ello un dominio de interpretación, p. ej., $D = \{Juan, Pedro\}$
- Ampliemos el lenguaje de formalización con una nueva constante: b
- La interpretación:
 1. $i(a) = Juan, i(b) = Pedro$
 2. $I_D(Juan) = V, I_D(Pedro) = V, H_D(Juan) = F, H_D(Pedro) = V$
- Verifica las premisas 1 y 2 y hace falsa la conclusión 3:
 1. $i(\exists x(I(x) \vee H(x))) = V$ sii $i((I(x) \vee H(x))\{x/c\}) = V$
para alguna constante c de L .

Sea a esa constante: $i(I(a) \vee H(a)) = V$ sii

 - 1.a $i(I(a)) = V$ sii: $i(I(a)) = V, i(a) = Juan$ y $I_D(Juan) = V$
 - 1.b o bien $i(H(a)) = V$...
 2. y $i(\neg H(a)) = V$ sii $i(H(a)) = F$ sii: $i(a) = Juan$ y $H_D(Juan) = F$
 3. y $i(\neg I(a)) = F$ sii $i(I(a)) = V$ sii: $i(a) = Juan$ y $I_D(Juan) = V$
- ... es un contramodelo, luego el **argumento NO es correcto**.

Ejemplo Consecuencia Lógica II

- Sea el argumento: Todo elemento químico es oxidante o reductor. El carbono es un elemento químico no oxidante. Luego el carbono es reductor.
- Y su formalización: $\{\forall x(E(x) \rightarrow O(x) \vee R(x)), E(a) \wedge \neg O(a)\} \models R(a)$
- Para determinar la corrección de este argumento por medios semánticos, tratemos de encontrar un contramodelo que:

1. Verifique las premisas:
 $i(\forall x(E(x) \rightarrow O(x) \vee R(x))) = V$ y $i(E(a) \wedge \neg O(a)) = V$
2. Haga falsa la pretendida conclusión: $i(R(a)) = F$
- Elijamos para ello un dominio de interpretación, p. ej., $D = \{\text{carbono}, \text{oxígeno}\}$
- Ampliemos el lenguaje de formalización con una nueva constante: b
- La interpretación:
 1. $i(a) = \text{carbono}, i(b) = \text{oxígeno}$
 2. $E_D(\text{carbono}) = E_D(\text{oxígeno}) = V, O_D(\text{carbono}) = F, O_D(\text{oxígeno}) = V, R_D(\text{carbono}) = R_D(\text{oxígeno}) = F$
- No puede verificar las premisas y hacer falsa la conclusión:
 1. $i(\forall x(E(x) \rightarrow O(x) \vee R(x))) = V$ sii
 - a) $i(E(a) \rightarrow O(a) \vee R(a)) = V$ sii $i(E(a)) = F$ o $i(O(a) \vee R(a)) = V$
 - b) $i(E(b) \rightarrow O(b) \vee R(b)) = V$ sii $i(E(b)) = F$ o $i(O(b) \vee R(b)) = V$
 2. $i(E(a) \wedge \neg O(a)) = V$ sii
 - a) $i(E(a)) = V$ y $i(O(a)) = F$ por 1.1 $R(a) = V$
 3. $i(R(a)) = F$ por 1.1 y 2.1 contradicción.
- ... lo que impide crear el contramodelo no depende de I .
- Como no existe contramodelo, el **argumento es correcto**.

Ejercicios Validez Argumentos

- Determina por medios semánticos la validez de los siguientes argumentos:
 1. $0 > 1. 1 > 2$. La relación ' $>$ ' es transitiva. Por tanto, $0 > 2$.
 2. Robin Hood es generoso. Todos los ladrones son generosos. Por tanto, Robin Hood es un ladrón.
 3. Siempre que Pedro discute con María, ésta se enfada con su padre. Una persona que se enfada con otra no la invita a su boda. Luego, si Pedro discute con María, ésta no invitará a su padre a su boda.
 4. Todos los ladrones son generosos. Sólo los ricos son generosos. Robin Hood es un ladrón. Por tanto, Robin Hood es rico.
 5. Juan Carlos ama a Sofía. Quienes trabajan con Juan Carlos no conocen a fondo a Sofía. Si se ama a alguien, se le conoce a fondo. Sabino trabaja con Juan Carlos. Por tanto, Sabino no ama a Sofía.
- Determina por medios semánticos la validez de los siguientes argumentos:
 6. Hay un ladrón generoso. Por tanto, hay un ladrón y hay alguien generoso.
 7. Hay un ladrón y hay alguien generoso. Por tanto, hay un ladrón generoso.
 8. Todo elemento pesado es metálico. Por tanto, ningún no metal es un ele-

mento pesado.

9. No es cierto que todo sea blanco o negro. Por tanto, hay algo que no es ni blanco ni negro.

3.2. Teorema de la demostración y deducción natural.

3.2.1. Decibilidad de la lógica de Primer Orden.

- Entscheidungsproblem: Denominación original del problema planteado por Hilbert, que consiste en determinar si un argumento dado es correcto o no para la Lógica de Primer Orden.
- Problema de la parada: saber si dada una tarea (descripción de sus instrucciones) y un input para la misma es posible determinar de manera efectiva si dicha tarea finalizaría arrojando un output, no importa cuál.

Teorema (Church y Turing 1936)

La Lógica de Primer Orden con identidad no es decidable.

- Demostración: Determinar el problema de parada se reduce a demostrar en LPO la fórmula que expresa la existencia de un output a partir de la aplicación de una serie de instrucciones previamente fijadas. \square
- Es decir, NO es posible decidir, para un conjunto cualquiera de fórmulas Γ de un LPO, si $\Gamma \vdash A$ o bien $\Gamma \not\vdash A$.
- Ejemplo:
 - Sea $\Gamma = \{P(a), \forall x(P(x) \rightarrow P(f(x)))\}$,
 - $\Gamma \vdash P(a), \Gamma \vdash P(f(a)), \Gamma \vdash P(f(f(a))), \Gamma \vdash P(f(f(f(a))))$, ...
 - $\text{¿}\Gamma \vdash P(f(b))\text{? o ¿}\Gamma \not\vdash P(f(b))\text{?}$
- Estrategias:
 1. Extender la deducción Natural de la lógica proposicional.
 2. Utilizar el método de resolución de Robinson (1965).
 - A partir de fórmulas simplificadas (Forma Normal de Skolem).
 - Resolver usando el unificador de máxima generalidad (UMG).
 - Elegir una estrategia de resolución computacionalmente eficiente.

3.2.2. Deducción Natural

- Al igual que en Lógica Proposicional podemos aplicar Deducción Natural incluyendo 4 nuevas reglas de inferencia.
- Hay fórmulas de LPO que podemos demostrar sin añadir nuevas reglas:

$$T[\exists xP(x) \rightarrow \forall yQ(y), \forall yQ(y) \rightarrow \forall zR(z)] \vdash \exists xP(x) \rightarrow \forall zR(z)$$

- En general las nuevas reglas son necesarias para:
 - “abrir” las fórmulas cuantificadas (eliminar cuantificadores).
 - aplicar las reglas de inferencia proposicionales.
 - “cerrar” las fórmulas resultantes (introducir cuantificadores).

Deducción Natural: Reglas de inferencia

Existencial

Introducción de \exists

$$\frac{F(a)}{\exists xF(x)}$$

$T[P(a) \vee Q(a)] \vdash \exists x(P(x) \vee Q(x))$	
1. $P(a) \vee Q(a)$	premisa
2. $\exists x(P(x) \vee Q(x))$	I_{\exists} 1

Eliminación de \exists

$$\frac{\exists xF(x)}{F(a^*)}$$

CONDICIÓN: 'a' ha de ser un nombre temporal nuevo en la demostración*

$T[\exists x(P(x) \wedge Q(x))] \vdash \exists xP(x)$	
1. $\exists x(P(x) \wedge Q(x))$	premisa
2. $P(a^*) \wedge Q(a^*)$	E_{\exists} 1
3. $P(a^*)$	E_{\wedge} 2
4. $\exists xP(x)$	I_{\exists} 3

**Universal
Eliminación de \forall**

$$\frac{\forall x F(x)}{F(t)}$$

$T[\forall x P(x)] \vdash P(f(b))$	
1. $\forall x P(x)$	Premisa
2. $P(f(b))$	$E_{\forall} 1$

Introducción de \forall

$\frac{F(x)}{\forall x F(x)}$	<i>CONDICIONES:</i>
	<ul style="list-style-type: none"> <input type="checkbox"/> $F(x)$ no es ni una premisa ni un supuesto <input type="checkbox"/> $F(x)$ no incluye nombres temporales <input type="checkbox"/> x no aparece libre en ninguna premisa ni supuesto previo no cancelado

$T[\forall x P(x)] \vdash \forall x(P(x) \vee Q(x))$	
1. $\forall x P(x)$	premisa
2. $P(x)$	$E_{\forall} 1$
3. $P(x) \vee Q(x)$	$I_{\vee} 2$
4. $\forall x(P(x) \vee Q(x))$	$I_{\forall} 3$

Ejemplos: Deducción Natural

- $T[\exists x(P(x) \wedge Q(x))] \vdash \exists x P(x) \wedge \exists x Q(x)$

 1. $\exists x(P(x) \wedge Q(x))$ premisa
 2. $P(a^*) \wedge Q(a^*)$ $E_{\exists}(1)$
 3. $P(a^*)$ $E_{\wedge}(2)$
 4. $\exists x P(x)$ $I_{\exists}(3)$
 5. $Q(a^*)$ $E_{\wedge}(2)$
 6. $\exists x Q(x)$ $I_{\exists}(5)$
 7. $\exists x P(x) \wedge \exists x Q(x)$ $I_{\wedge}(4, 6)$
- Se introduce a^* como una constante nueva, sólo vigente mientras llegamos a deducir la fórmulas que nos interesan, que son $\exists x P(x)$ y $\exists x Q(x)$

Deducción Natural: Ejemplo (Incorrecto)

- $T[\exists x P(x) \wedge \exists x Q(x)] \vdash \exists x(P(x) \wedge Q(x))$

 1. $\exists x(P(x) \wedge \exists x Q(x))$ premisa
 2. $\exists x P(x)$ $E_{\wedge}(1)$
 3. $P(a^*)$ $E_{\exists}(2)$
 4. $\exists x Q(x)$ $E_{\wedge}(1)$
 5. $Q(a^*)$ $E_{\exists}(4)$
 6. $P(a^*) \wedge Q(a^*)$ $I_{\wedge}(3, 5)$
 7. $\exists(x P(x) \wedge x Q(x))$ $I_{\exists}(6)$
- Al usar otra vez a^* damos a entender que el elemento que cumple Q es el mismo que cumple P , y no tiene por qué ser así.

Deducción Natural: Ejemplo

- $T[\forall x(P(x) \rightarrow Q(x)), \forall x(Q(x) \rightarrow R(x))] \vdash \forall x(P(x) \rightarrow R(x))$
 1. $\forall x(P(x) \rightarrow Q(x))$ premisa
 2. $\forall x(Q(x) \rightarrow R(x))$ premisa
 3. $P(x) \rightarrow Q(x)$ $E_{\forall}(1)$
 4. $Q(x) \rightarrow R(x)$ $E_{\forall}(2)$
 5. $P(x)$ supuesto
 6. $Q(x)$ $E_{\rightarrow}(3,5)$
 7. $R(x)$ $E_{\rightarrow}(4,6)$
 8. $P(x) \rightarrow R(x)$ $I_{\rightarrow}(5,7)$
 9. $\forall x(P(x) \rightarrow R(x))$ $I_{\forall}(8)$

Deducción Natural: Ejemplo (Incorrecto)

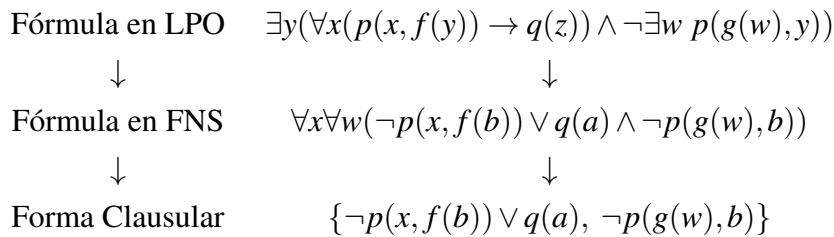
- $T[\exists xP(x)] \vdash \forall xP(x)$
 1. $\exists xP(x)$ premisa
 2. $P(a^*)$ $E_{\exists}(1)$
 3. $\forall xP(x)$ $I_{\forall}(2)$
- $T[\forall x\exists yM(y,x)] \vdash \exists y\forall xM(y,x)$
 1. $\forall x\exists yM(y,x)$ premisa
 2. $\exists yM(y,x)$ $E_{\forall}(1)$
 3. $M(a^*,x)$ $E_{\exists}(2)$
 4. $\forall xM(a^*,x)$ $I_{\forall}(3)$
 5. $\exists y\forall xM(y,x)$ $I_{\exists}(4)$
- No se puede generalizar una fórmula que tiene constantes temporales.

3.3. Simplificación de fórmulas.**3.3.1. Objetivo.**

- Queremos simplificar las fórmulas mediante transformaciones de una fórmula F a otra F' (preservando ciertas propiedades):
 - La satisfacibilidad: Existe un modelo I de F sii existe un modelo I' (probablemente no el mismo) de F'

$$\text{SAT}(\exists x p(x)) \text{ sii } \text{SAT}(p(a))$$
 - La semántica: Para toda interpretación I , I es un modelo de F sii es un modelo de F'

$$\forall x p(x) \text{ es semánticamente equivalente a } \neg\exists x \neg p(x)$$
- Ejemplo: De LPO a Forma Normal de Skolem (FNS) a Forma Clausular:



- Preserva la satisfacibilidad pero NO preserva todos los modelos (la semántica): el resultado no es equivalente a la fórmula original.

3.3.2. Forma Normal de Skolem

- Procedimiento para obtener la Forma Normal de Skolem (FNS) de una fórmula A :
 1. **Obtener la forma Prenex:** Todos los cuantificadores a la cabeza de la fórmula. La forma Prenex(A) siempre existe, aunque puede no ser única.
 2. **Realizar el cierre existencial:** Se quitan las variables libres. Una fórmula $A(x)$ es satisfacible sii $\exists x A(x)$ es satisfacible.
 3. **Obtener la forma normal conjuntiva:** Conjunción de disyunciones. La forma normal conjuntiva de A siempre existe.
 4. **Eliminar los cuantificadores existenciales:** Dejar solo los cuantificadores universales. El resultado es la FNS.

Una fórmula A es satisfacible sii $FNS(A)$ es satisfacible.

1. Forma Prenex

- Poner cuantificadores a la cabeza de la fórmula usando:

- Cambio de nombre de variables ligadas:

$$\forall x A(x) \leftrightarrow \forall y A(x/y) \qquad \exists x A(x) \leftrightarrow \exists y A(x/y)$$

... si y no está libre en A

- Interdefinición de cuantificadores:

$$\neg \forall x A(x) \leftrightarrow \exists x \neg A(x) \qquad \neg \exists x A(x) \leftrightarrow \forall x \neg A(x)$$

- Distribución de conectivas respecto a cuantificadores:

$$\forall x A \wedge C \leftrightarrow \forall x (A \wedge C) \qquad (\forall x A \rightarrow C) \leftrightarrow \exists x (A \rightarrow C)$$

$$\exists x A \wedge C \leftrightarrow \exists x (A \wedge C) \qquad (\exists x A \rightarrow C) \leftrightarrow \forall x (A \rightarrow C)$$

$$\forall x A \vee C \leftrightarrow \forall x (A \vee C) \qquad (A \rightarrow \forall x C) \leftrightarrow \forall x (A \rightarrow C)$$

$$\exists x A \vee C \leftrightarrow \exists x (A \vee C) \qquad (A \rightarrow \exists x C) \leftrightarrow \exists x (A \rightarrow C)$$

... si x no está libre en la otra subfórmula

$$\forall x A \wedge \forall x C \leftrightarrow \forall x (A \wedge C) \qquad (\exists x A \vee \exists x C) \leftrightarrow \exists x (A \vee C)$$

2. Cierre existencial

- Las variables libres de la fórmula se ligan existencialmente poniendo el cuantificador correspondiente en cabeza de la fórmula.
- Ejemplo:

$$\begin{array}{c} \forall y \exists z (P(x) \wedge Q(y) \rightarrow r(f(z), x)) \\ \downarrow \\ \exists x \forall y \exists z (P(x) \wedge Q(y) \rightarrow r(f(z), x)) \end{array}$$

3. Forma normal conjuntiva

- Transformar en conjunción de disyunciones usando:

- Interdefinición:

$$(A \rightarrow B) \leftrightarrow (\neg A \vee B)$$

$$(A \leftrightarrow B) \leftrightarrow (A \rightarrow B) \wedge (B \rightarrow A)$$

- Leyes de De Morgan

$$\neg(A \wedge B) \leftrightarrow \neg A \vee \neg B$$

$$\neg(A \vee B) \leftrightarrow \neg A \wedge \neg B$$

- Distribución de \vee y \wedge

$$A \wedge (B \vee C) \leftrightarrow (A \wedge B) \vee (A \wedge C)$$

$$A \vee (B \wedge C) \leftrightarrow (A \vee B) \wedge (A \vee C)$$

4. Eliminación de cuantificadores existenciales

- Se elimina el cuantificador existencial sustituyendo la variable que ligaba por una función de Skolem o constante de Skolem.
- La función de Skolem será una función nueva en la fórmula, aplicada a todas las variables cuantificadas universalmente que aparecen antes que el cuantificador existencial a eliminar. Si no hay tales variables se utilizará una constante nueva en la fórmula para hacer la sustitución.

- Ejemplos:

- $\forall x \exists y (p(x) \rightarrow \neg q(y))$ se transforma en $\forall x (p(x) \rightarrow \neg q(\mathbf{f}(\mathbf{x})))$

- $\exists x \forall z (q(x, z) \vee r(a, x))$ se transforma en $\forall z (q(\mathbf{b}, z) \vee r(a, \mathbf{b}))$

- $\exists x \forall y \exists z (p(x) \wedge q(y) \rightarrow r(f(z), x))$ se transforma en $\forall y (p(\mathbf{a}) \wedge q(y) \rightarrow r(f(\mathbf{g}(\mathbf{y})), \mathbf{a}))$

3.3.3. Forma Clausular

- Una **cláusula** es una disyunción de literales.
- La **forma clausular** es la conjunción de las cláusulas, cuyas variables están todas ellas ligadas universalmente.
- Ejemplo:

$$A : \forall x \forall w (\neg p(x, f(b)) \vee q(a) \wedge \neg p(g(w), b))$$

$$\downarrow$$

$$FC(A) : \{ \neg p(x, f(b)) \vee q(a), \neg p(g(w), b) \}$$

Teorema Una fórmula F es satisfacible sii $FC(F)$ es satisfacible

Forma Clausular: De una deducción

- Una deducción $T[P_1, P_2, \dots, P_n] \vdash C$ es correcta sii $P_1 \wedge P_2 \wedge \dots \wedge \neg C$ es insatisfacible
- Por lo tanto dada una deducción $T[P_1, P_2, \dots, P_n] \vdash C$:
 - Obtener la forma clausular de cada P_i , $i \leq i \leq n$.
 - Obtener la forma clausular de $\neg C$.
 - Realizar la unión de todos los conjuntos de cláusulas.
 - Comprobar la satisfacibilidad (p.ej. mediante el método de resolución de Robinson).

Objetivo Una deducción $T[P_1, P_2, \dots, P_n] \vdash C$ es correcta sii

$$\{P_1, P_2, \dots, \neg C\} \text{ es insatisfacible}$$

3.4. Método de resolución de Robinson.

3.4.1. Introducción

- Una fórmula A es insatisfacible sii no existe ninguna interpretación I tal que $I(A) = V$ (sii todas las interpretaciones la evalúan como falsa)
 - Si A es una fórmula proposicional el número de interpretaciones a analizar es finito: 2^n , siendo n el número de proposiciones diferentes que la integran.

- Si A es una fórmula de primer orden el número de interpretaciones a analizar sería infinito y no numerable.
- Solución: Las interpretaciones de Herbrand:
 - Cuyo número sea menor (finito o infinito numerable como mucho).
 - Cuyo análisis fuese suficiente para determinar la insatisfacibilidad de A (si A es insatisfacible para esas interpretaciones también lo es para cualquier otra).

3.4.2. Interpretaciones de Herbrand

Universo de Herbrand

- El dominio de la interpretación de un fórmula A en las interpretaciones de Herbrand, es H , el **Universo de Herbrand de A** .
- Se define de manera recursiva a partir de las constantes y funciones de A , aplicadas de todas las formas posibles.
 - Sea C el conjunto de constantes de A .
 - Sea F el conjunto de funciones de A .

$$H_0 = C \text{ si } C \neq \emptyset \quad \text{o} \quad H_0 = \{a\} \text{ si } C = \emptyset$$

$$H_i = \{f(t_1, \dots, t_n) / t_j \in (H_{i-1} \cup \dots \cup H_0), f/n \in F\}$$

$$H = H_0 \cup H_1 \cup \dots \cup H_i \cup \dots$$

Ejemplos: Universos de Herbrand

- $A = \{P(x), Q(y)\}$ $C = \emptyset \quad F = \emptyset$
 - $H_0 = \{a\}$
 - $H_1 = H_2 = \dots = \emptyset$
 - $H = \{a\}$
- $A = \{P(a), Q(y) \vee \neg R(b, f(x))\}$ $C = \{a, b\} \quad F = \{f/1\}$
 - $H_0 = \{a, b\}$
 - $H_1 = \{f(a), f(b)\}$
 - $H_2 = \{f(a), f(f(a)), f(b), f(f(b))\}$
 - ...
 - $H = \{a, b, f(a), f(b), f(f(a)), f(f(b)), \dots\} =$
 $= \{a, b, f^n(a), f^n(b), \forall n \geq 1\}$

Base de Herbrand

- **Átomo básico:** Se obtiene aplicando un símbolo de predicado de la fórmula a términos de su universo de Herbrand.
- **Base de Herbrand (BH)** de una fórmula A : Conjunto de todos los átomos básicos de A .
- Contiene todos los predicados de A aplicados a todos los términos de H de todas las formas posibles.

- Sea P el conjunto de predicado de A

$$BH = \{P(t_1, \dots, t_n) / t_j \in H, P/n \in P\}$$

- $A = \{P(x), Q(y)\}$ $H = \{a\}$
 $BH = \{P(a), Q(a)\}$

- $A = \{P(a), Q(y) \vee \neg R(b, f(x))\}$ $H = \{a, b, f(a), f(b), \dots\}$
 $BH = \{P(a), P(f(a)), \dots, P(f^n(a)), \dots,$
 $P(b), P(f(b)), \dots, P(f^n(b)), \dots,$
 $Q(a), Q(f(a)), \dots, Q(f^n(a)), \dots,$
 $Q(b), Q(f(b)), \dots, Q(f^n(b)), \dots,$
 $R(a, a), R(a, b), \dots, R(a, f^n(a)), \dots,$
 $R(b, a), \dots, R(f^n(b), f^m(a)), \dots\} =$
 $= \{P(t), Q(t), R(t, t'), \forall t, t' \in H\}$

Definición

- Finalmente, una interpretación de Herbrand (IH) de una fórmula A es un interpretación sobre H tal que:¹
 - Cada constante $a \in C$ se asocia consigo misma: $IH(a) = a$
 - Cada símbolo de función $f/n \in F$ se asocia con la función f_H/n
 - $f_H : H^n \Rightarrow H$
 - $IH(f(t_1, t_2, \dots, t_n)) = f_H(IH(t_1), IH(t_2), \dots, IH(t_n)) =$
 $f_H(t_1, t_2, \dots, t_n) = t$
 - Cada símbolo de predicado $P/n \in P$ se asocia con el predicado P_H/n
 - $P_H : H^n \Rightarrow \{V, F\}$
 - $IH(P(t_1, \dots, t_n)) = P_H(IH(t_1), \dots, IH(t_n)) =$
 $P_H(t_1, t_2, \dots, t_n) = V$ (o F)
 - Cada átomo básico de BH tiene un valor de verdad.

¹Notación: C es el conjunto de constantes, F el conjunto de símbolos de función y P el conjunto de símbolos de predicado del LPO de A

Ejemplos: Interpretaciones de Herbrand

- Una interpretación Herbrand se puede representar como el conjunto de los átomos básicos de la Base de Herbrand, afirmados si se interpretan como verdaderos y negados si se interpretan como falsos
- $A = \{P(x), Q(y)\}$

$IH_1 = \{P(a), Q(a)\}$	$BH = \{P(a), Q(a)\}$
$IH_2 = \{P(a), \neg Q(a)\}$	$IH_1(P(a))=V, IH_1(Q(a))=V$
$IH_3 = \{\neg P(a), Q(a)\}$	$IH_2(P(a))=V, IH_2(Q(a))=F$
$IH_4 = \{\neg P(a), \neg Q(a)\}$	$IH_3(P(a))=F, IH_1(Q(a))=F$
	$IH_4(P(a))=F, IH_4(Q(a))=F$

Satisfabilidad

Una fórmula A es insatisfacible sii

A es falsa para todas sus interpretaciones Herbrand (IH_i)

- Por lo tanto, para estudiar la insatisfacibilidad de una fórmula basta con estudiar las interpretaciones Herbrand de su forma clausular:
 - Si alguna IH_i evalúa A como verdadera entonces A es satisfacible.
 - Si ninguna IH_i evalúa A como verdadera entonces es insatisfacible.
- Ejemplo: $A = \{P(x), Q(y)\}$
 - $IH_1 = \{P(a), Q(a)\}$
es un modelo de A (hace verdad todas las instancias).
 - $IH_2 = \{P(a), \neg Q(a)\}, IH_3 = \{\neg P(a), Q(a)\}, IH_4 = \{\neg P(a), \neg Q(a)\}$,
son contramodelos de A (falsifican una o las dos instancias).

3.4.3. Teorema de Herbrand

- Cualquier interpretación de Herbrand que evalúe una fórmula A como verdadera, debe hacer verdad a todas y cada una de las instancias básicas de la fórmula.

Teorema de Herbrand

Un conjunto de cláusulas C es insatisfacible sii existe un conjunto finito de instancias básicas de cláusulas de C que es insatisfacible.

- El teorema sugiere que dado un conjunto de cláusulas C insatisfacible, mediante un procedimiento mecánico generar incrementalmente conjuntos de instancias básicas $S_1, S_2, \dots, S_n, \dots$ de cláusulas de C y analizar de forma sucesiva la insatisfacibilidad de $S_1, S_2, \dots, S_n, \dots$, se podría detectar un k tal que S_k es insatisfacible.

- P.ej., método de Resolución de Robinson (1965).

3.4.4. Método de Resolución de Robinson

- **Idea general:** Plantear un método de obtención de nuevas instancias deducidas del conjunto original, de forma que si llega a deducirse un literal y su negación puede concluirse que el conjunto original es insatisfacible.
- Está basado en la **regla de resolución básica:** De dos instancias básicas $L \vee C1$ y $\neg L \vee C2$ (L es un literal) puede deducirse una nueva instancia básica $C1 \vee C2$, llamada **resolvente**:

$$\begin{array}{ccc} L \vee C1 & & \neg L \vee C2 \\ & \searrow & \swarrow \\ & C1 \vee C2 & \end{array}$$

- La aplicación sucesiva de la regla de resolución permite obtener una **contradicción** cuando el conjunto original es insatisfacible.
- La contradicción se obtiene cuando se deducen dos instancias básicas (literales aislados) L y $\neg L$. La aplicación de la regla sobre L y $\neg L$ genera \square , llamada cláusula vacía.
- Para asegurarnos deducir la cláusula vacía hay que tener en cuenta la idempotencia $L \vee L \Rightarrow L$

$$\begin{array}{ccc} L \vee L & & \neg L \vee \neg L \\ & \searrow & \swarrow \\ & L \vee \neg L & \\ & & \Rightarrow \\ & & \begin{array}{ccc} L & & \neg L \\ & \searrow & \swarrow \\ & \square & \end{array} \end{array}$$

- o aplicar la regla de resolución básica extendida, que de dos instancias $L \vee \dots \vee L \vee C1$ y $\neg L \vee \dots \vee \neg L \vee C2$ deduce $C1 \vee C2$.

Algoritmo

- Dado un conjunto C de instancias básicas:
 1. Generar el conjunto R de todos los resolventes que pueden obtenerse aplicando la regla de resolución entre instancias del conjunto C de todas las formas posibles.
 2. Si \square está incluida en R entonces terminar $\Rightarrow C$ es insatisfacible.
 3. Si $R \subseteq C$ significa que ya se han generado todos los resolventes posibles, entonces terminar $\Rightarrow C$ es satisfacible.
 4. Hacer $C = C \cup R$ y repetir desde paso 1.
- El método es **correcto**: Si deducimos \square entonces C es insatisfacible.

- El método es **completo**: Si C es insatisfacible, entonces con la aplicación de la regla de resolución deduciremos \square .

Ejemplo: Método de Resolución de Robinson

- Resolución aplicando el procedimiento de saturación:

$C = \{I1: \neg p(a, f(b)), I2: \neg p(b, f(b)), I3: p(a, f(b)) \vee q(f(b)), I4: p(b, f(b)) \vee \neg q(f(b))\}$

resuelve I1 con I2: NO resuelve I2 con I3: NO
 resuelve I1 con I3: $q(f(b))$ resuelve I2 con I4: $\neg q(f(b))$
 resuelve I1 con I4: NO resuelve I3 con I4: $p(a, f(b)) \vee p(b, f(b))$

$R = \{I5: q(f(b)), I6: \neg q(f(b)), I7: p(a, f(b)) \vee p(b, f(b))\}$

En R no está \square , por tanto redefinimos $C = C \cup R$ y buscamos **nuevos** resolventes:

resuelve I1 con I5: NO resuelve I2 con I5: NO
 resuelve I1 con I6: NO resuelve I2 con I6: NO
 resuelve I1 con I7: $p(b, f(b))$ resuelve I2 con I7: $p(a, f(b))$

resuelve I3 con I5: NO resuelve I4 con I5: $p(b, f(b))$
 resuelve I3 con I6: $p(a, f(b))$ resuelve I4 con I6: NO
 resuelve I3 con I7: NO resuelve I4 con I7: NO

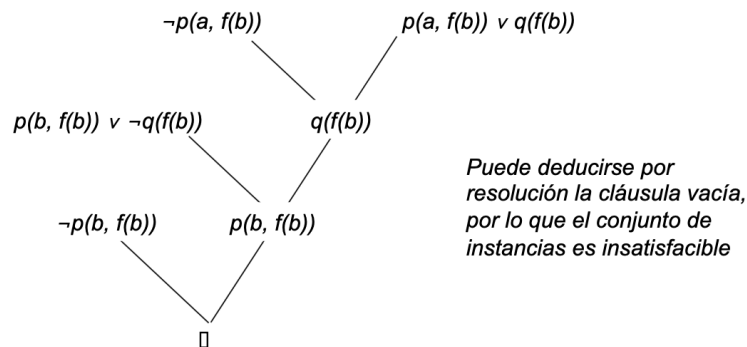
resuelve I5 con I6: \square
 resuelve I5 con I7: NO
 resuelve I6 con I7: NO

$R = \{p(b, f(b)), p(a, f(b)), \square\}$

R incluye a $\square \rightarrow C$ es insatisfacible

- En la práctica, la aplicación de sucesivos pasos de resolución se puede representar en forma de árbol (**árbol de resolución**):
 - Árbol binario invertido (cada dos nodos tienen un “hijo” común)
 - Cada nodo representa una instancia básica.
 - Sólo se representan los pasos relevantes para llegar a \square .

Conjunto de instancias básicas: $\{\neg p(a, f(b)), \neg p(b, f(b)), p(a, f(b)) \vee q(f(b)), p(b, f(b)) \vee \neg q(f(b))\}$



3.4.5. Resolución con Unificador de Máxima Generalidad

Sustitución

- Un **sustitución** $\alpha = \{x_1/t_1, \dots, x_n/t_n\}$ es una función finita de un conjunto de variables de un lenguaje en el de términos. Donde x_i/t_i es una **ligadura**.
- $\text{Dominio}(\alpha) = \{x_i : x_i/t_i \in \alpha\}$
- $\text{Rango}(\alpha) = \{y_i : y_i \text{ aparece en } t_i, x_i/t_i \in \alpha\}$
- Ejemplos

$$\alpha_1 = \{x/f(a), y/x, z/h(b, y), w/a\} \quad \text{Dominio}(\alpha_1) = \{x, y, z, w\} \quad \text{Rango}(\alpha_1) = \{x, y\}$$

$$\alpha_2 = \{x/a, y/a, z/h(b, c), w/f(d)\} \quad \text{Dominio}(\alpha_2) = \{x, y, z, w\} \quad \text{Rango}(\alpha_2) = \emptyset$$

$$\alpha_3 = \{x/y, z/w\}$$

Renombrado

$$\lambda = \{x/x, y/y, z/z\}$$

Sustitución vacía

- Dada una fórmula F y una sustitución $\alpha = \{x_1/t_1, \dots, x_n/t_n\}$, se denomina **aplicación de α a F** ($F\alpha$) a la fórmula obtenida reemplazando simultáneamente cada ocurrencia en F de x_i por t_i , para cada $x_i/t_i \in \alpha$.
- Ejemplo: $\alpha = \{x/f(a), y/x, z/h(b, y), w/a\}$
 - $P(x, y, z)\alpha = P(f(a), x, h(b, y))$ Correcto
 - $P(x, y, z)\alpha = P(f(a), f(a), h(b, f(a)))$ Incorrecto
- Si $\text{Dominio}(\alpha) \cap \text{Rango}(\alpha) = \emptyset$, α es **idempotente** y $(F\alpha)\alpha = F\alpha$:
 - $\alpha_1 = \{x/a, y/f(b), z/v\}$ Es idempotente
 - $P(x, y, w, z)\alpha_1 = P(a, f(b), w, v)$ $P(a, f(b), w, v)\alpha_1 = P(a, f(b), w, v)$
 - $\alpha_2 = \{x/a, y/f(b), z/x\}$ No es idempotente
 - $P(x, y, w, z)\alpha_2 = P(a, f(b), w, x)$ $P(a, f(b), w, x)\alpha_2 = P(a, f(b), w, a)$
- F' es instancia de F si existe una sustitución $\alpha \neq \emptyset$ tal que $F' = F\alpha$

Composición sustituciones

- Dadas dos sustituciones:
 - $\alpha = \{x_1/t_1, \dots, x_n/t_n\}$ y $\beta = \{y_1/s_1, \dots, y_m/s_m\}$
 - su composición $\alpha\beta$ se define como el conjunto $\{x_1/t_1\beta, \dots, x_n/t_n\beta, y_1/s_1, \dots, y_m/s_m\}$ una vez eliminadas:
 - las ligaduras $x_i/t_i\beta$ tales que $x_i \equiv t_i\beta$,
 - y las ligaduras y_i/s_i tales que $y_i \in \{x_1, \dots, x_n\}$
- **Ejemplo:** Si $\alpha = \{x/3, y/f(x, 1)\}$ y $\beta = \{x/4\}$ entonces:
 - $\alpha\beta = \{x/3, y/f(4, 1)\}$

- $\beta\alpha = \{x/4, y/f(x, 1)\}$
- Propiedades de la composición:
 - $(F\alpha)\beta = F(\alpha\beta)$
 - $(\alpha\beta)\gamma = \alpha(\beta\gamma)$
 - $\alpha\lambda = \lambda\alpha = \alpha$
 - $\alpha\beta \neq \beta\alpha$

Unificadores y UMG

- Una sustitución α es un **unificador** de dos fórmulas A y B si $A\alpha = B\alpha$. En este caso se dice que A y B son **unificables**.
- Un unificador α de A y B se denomina **unificador de máxima generalidad** (umg) si para cualquier otro unificador β de A y B existe alguna sustitución γ tal que $\beta = \alpha\gamma$
 - Si dos fórmulas son unificables entonces tienen umg
 - El umg de dos fórmulas es único (salvo renombrado)
- Ejemplo: $A \equiv P(x, f(x, g(y)), z)$ y $B \equiv P(r, f(r, u), a)$

$$\alpha_1 = \{x/r, u/g(y), z/a\} \qquad \alpha_2 = \{x/a, r/a, y/b, u/g(b), z/a\}$$

$$A\alpha_1 = B\alpha_1 = P(r, f(r, g(y)), a) \qquad A\alpha_2 = B\alpha_2 = P(a, f(a, g(b)), a)$$
- α_1 y α_2 son unificadores de A y B .
 - α_1 es el umg de A y B , porque con $\gamma = \{r/a, y/b\}$, $\alpha_2 = \alpha_1\gamma$.

Algoritmo de Unificación

- Sean A y B dos átomos con el mismo símbolo de predicado:
 1. $\alpha = \lambda$
 2. Mientras $A\alpha \neq B\alpha$:
 - a) Encontrar el símbolo más a la izquierda en $A\alpha$ tal que el símbolo correspondiente en $B\alpha$ sea diferente.
 - b) Sean t_A y t_B los términos de $A\alpha$ y $B\alpha$ que empiezan con esos símbolos:
 - Si ni t_A ni t_B son variables o, si uno de ellos es una variable que aparece en el otro \Rightarrow terminar con fallo (A y B no son unificables)
 - En otro caso, sea t_A una variable \Rightarrow el nuevo α es el resultado de $\alpha\{t_A/t_B\}$
 3. Terminar, siendo α el umg de A y B

Ejemplo: Resolución con UMG: Algoritmo de Unificación

- Ejemplo 1: $A = P(x, x)$ y $B = P(f(a), f(b))$

α	$A\alpha$	$B\alpha$	(t_a, t_b)
λ	$P(x, x)$	$P(f(a), f(b))$	$(x, f(a))$
$\{x/f(a)\}$	$P(f(a), f(a))$	$P(f(a), f(b))$	(a, b)
FALLO	A y B NO son unificables		

- Ejemplo 2: $A = P(x, f(y))$ y $B = P(z, x)$

α	$A\alpha$	$B\alpha$	(t_a, t_b)
λ	$P(x, f(y))$	$P(z, x)$	(x, z)
$\{x/z\}$	$P(z, f(y))$	$P(z, z)$	$(f(y), z)$
$\{x/f(y), z/f(y)\}$	$P(f(y), f(y))$	$P(f(y), f(y))$	ÉXITO
	A y B son unificables, su umg es $\{x/f(y), z/f(y)\}$		

Definición

- **Regla de resolución con umg:** Sean $L_1 \vee \dots \vee L_n \vee C_1$ y $\neg L'_1 \vee \dots \vee \neg L'_m \vee C_2$ dos cláusulas, donde todos los L_{ij} son literales con el mismo símbolo de predicado. Puede deducirse una nueva cláusula $(C_1\rho_1 \vee C_2\rho_2)\beta$, llamada resolvente, donde
 - ρ_1 y ρ_2 son renombrados cuyos dominios respectivos son todas las variables de cada cláusula y $Rango(\rho_1) \cap Rango(\rho_2) = \emptyset$
 - β es **umg** de $\{L_1\rho_1, \dots, L_n\rho_1, \neg L'_1\rho_2, \dots, \neg L'_m\rho_2\}$
- La regla de resolución con umg se apoya en una versión de la **regla de factorización** para LPO: Dada una cláusula $L_1 \vee \dots \vee L_n \vee C$, siendo L_1, \dots, L_n literales con el mismo símbolo de predicado, puede deducirse una nueva cláusula $L \vee C\beta$ donde
 - β es unificador de L_1, \dots, L_n
 - $L = L_1\beta = \dots = L_n\beta$ El literal L se denomina
factor de $L_1 \vee \dots \vee L_n \vee C$
- La aplicación de la regla de resolución con UMG es **correcta**.
- La aplicación de la regla de resolución con UMG es **completa**.

Teorema

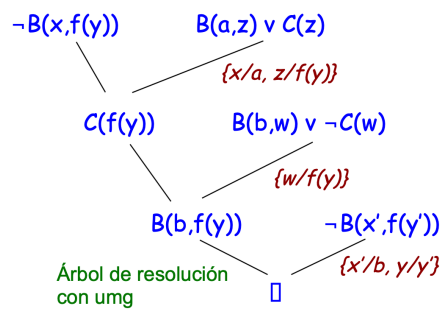
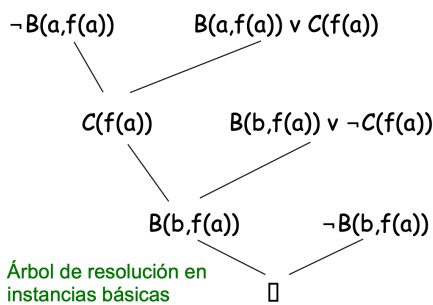
Un conjunto de cláusulas es insatisfacible sii se puede deducir o a partir de él por resolución con umg.

- Por tanto, el método general de insatisfacibilidad se puede reducir a la búsqueda de \square a partir del conjunto de cláusulas, en lugar de tener que generar conjuntos de instancias básicas.

Ejemplo: Resolución con UMG

- Pueden construirse árboles de resolución en los que los resolventes de cada dos cláusulas se obtienen en un paso de resolución con umg.
 - Por cada paso de resolución en instancias básicas puede definirse un paso de resolución con umg.

Ejemplo: $C_1: \neg B(x, f(y))$ $C_2: B(a, z) \vee C(z)$ $C_3: B(b, w) \vee \neg C(w)$
 $I_1: \neg B(a, f(a))$ $I_1': \neg B(b, f(a))$ $I_2: B(a, f(a)) \vee C(f(a))$ $I_3: B(b, f(a)) \vee \neg C(f(a))$



3.4.6. Estrategias de resolución

Motivación

- Distintas estrategias de resolución tienen sus ventajas e inconvenientes.
 - P.ej., la aplicación del procedimiento de saturación, sin limitaciones, genera normalmente muchas cláusulas irrelevantes y redundantes.
- Para hacer el proceso de resolución computacionalmente eficiente es necesario aplicar criterios selectivos de forma sistemática que simplifiquen el proceso.
 - Estrategias de **simplificación**: con el objetivo de reducir el número de cláusulas en el conjunto.
 - Estrategias de **refinamiento**: con el objetivo de limitar la generación de cláusulas.

Ejemplos y propiedades

Estrategia	Correcta	Completa
Saturación	Si	Si
Lineal	Si	Si
Input	Si	No, en el caso general
Dirigida	Si	Si, si el conjunto soporte es satisficible
Ordenada	Si	No

- **Correcta:** \square se deduce sólo si el conjunto de cláusulas S es insatisficible:
 $\square \rightarrow S$ insatisficible
- **Completa:** Si el conjunto de cláusulas S es insatisficible, se deduce \square :
 S insatisficible $\rightarrow \square$

Resolución lineal

- Aplica las siguientes reglas de simplificación:
 - **Eliminación de cláusulas idénticas:** Es posible deducir por resolución \square a partir de un conjunto de cláusulas C sii es posible deducir por resolución \square a partir de C tras eliminar cláusulas idénticas.
 - **Eliminación de cláusulas con literales puros:** Un literal L de un conjunto de cláusulas es puro si y sólo si no existe ningún otro literal complementario $\neg L'$ en el conjunto tal que L y L' son unificables.
 - Una cláusula con un literal puro es inútil de cara a la refutación porque el literal nunca podrá eliminarse en el proceso de resolución.
 - **Eliminación de cláusulas tautológicas:** Una cláusula tautológica es verdad para cualquier interpretación. P.ej., $P(x) \vee \neg P(x) \vee Q(y)$.
- Aplica la siguiente regla de refinamiento:
 - **Derivación lineal:** Una derivación lineal de C_m a partir de $\{C_1, \dots, C_n\}$ es una secuencia $C_1, \dots, C_n, C_{n+1}, \dots, C_m$ tal que:
 - C_{n+1} es el resolvente de dos cláusulas $\in \{C_1, \dots, C_n\}$ (cláusulas de cabecera) y
 - Para todo $i > n + 1$, C_i es el resolvente de C_{i-1} con otra cláusula $C_j, j < i$.
- **Resolución lineal:** Sólo genera derivaciones lineales
- La resolución lineal es **completa:** Un conjunto de cláusulas C es insatisficible sii existe una refutación lineal de C .
 - Podemos restringir las derivaciones posibles a derivaciones lineales.

C apitulo 4

Introducci on a la Programaci on L ogica

4.1	Tema 4.1:	74
4.1.1	Estrategia de resoluci�on SLD	74
	Ejercicio: Estrategia de resoluci�on SLD	74
4.1.2	�Arbol de derivaci�on	74
	Ejemplo: �Arbol de derivaci�on SLD	75
4.2	Tema 4.2:	75
4.2.1	Programaci�on Declarativa	75
4.2.2	Programaci�on Funcional	76
	C�alculo Lambda	76
	Haskell	77
	Booleanos en c�alculo lambda	77
	Caracter�isticas y Ventajas	78
4.2.3	Programaci�on L�ogica	79
	Cl�ausulas Horn	79
	Prolog	80
	Hay algo m�as ...	80
	... mucho m�as (CLP)	81
4.2.4	UTD HackReason	82

4.1. Estrategia de resolución SLD.

4.1.1. Estrategia de resolución SLD

- SLD significa resolución Lineal con función de Selección para cláusulas Definidas.
- Se trata de la estrategia usada por el lenguaje de programación Prolog.
- Es un caso particular de la resolución general para cláusulas de Horn:
 - Las cláusulas objetivo no tiene literal afirmado.
 - Las cláusulas soporte tienen un literal afirmado (el primero).
- Dado un conjunto inicial de cláusulas de Horn $\{C_1, \dots, C_i, \dots, C_n\}$:
 - Existe una secuencia (*derivación*) $\langle C_i, C_{n+1}, \dots, [] \rangle$ tal que:
 - C_{n+1} es el resolvente de la cláusula objetivo C_i y una cláusula soporte.
 - C_k , con $k > n + 1$, es el resolvente de C_{k-1} con una cláusula soporte.
 - Cada paso de resolución es de la forma $L \vee C, \neg L \vee C' \rightarrow C \vee C'$.
- Si y solo si el conjunto inicial es insatisfacible.

Ejercicio: Estrategia de resolución SLD

1. Obtener la forma clausular y resolver usando la estrategia SLD:

$$\frac{\forall ls \quad \text{Concatenar}([], ls, ls) \quad \forall x, xs, ls, ns \quad \text{Concatenar}(xs, ls, ns) \rightarrow \text{Concatenar}([x|xs], ls, [x|ns])}{\exists la, lb \quad \text{Concatenar}(la, lb, [1, 2, 3, 4])}$$

2. Comprobar que es equivalente al programa Prolog:

```

1 concatenar([], Ls, Ls).
2 concatenar([X|Xs], Ls, [X|Ns]) :- concatenar(Xs, Ls, Ns).
3
4 ?- concatenar(La, Lb, [1,2,3,4]).

```

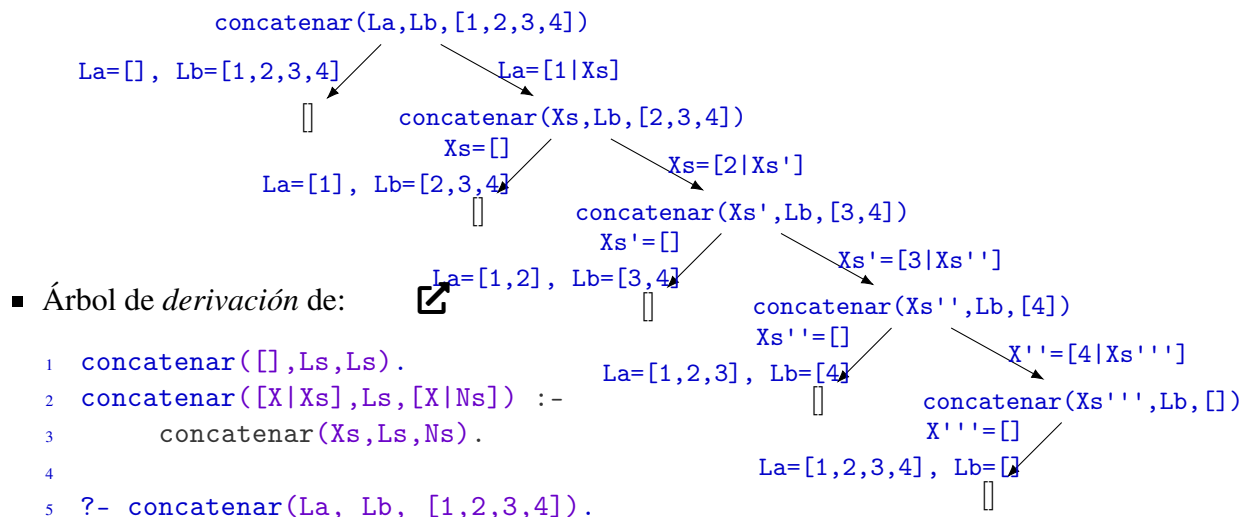


4.1.2. Árbol de derivación

- La estrategia de resolución SLD, desde el punto de vista de la evaluación de un programa Prolog, es un árbol de derivación:
 1. La cabeza es la consulta Q .

2. Los nodos hijos son el cuerpo (resolvente) de las cláusulas cuya cabeza unifica con la consulta.
3. Si el resolvente es [] la consulta tiene éxito:
 - Si la consulta tiene variables la solución es la composición de las sustituciones calculadas (en las aristas).
4. En caso contrario, se “resuelve” un literal del resolvente.
 - Se crea un hijo por cada cláusula que unifica con dicho literal.
 - Se añade el cuerpo de la cláusula al resolvente.
 - Se vuelve al punto 3.

Ejemplo: Árbol de derivación SLD



4.2. Prolog & la Programación Declarativa.

4.2.1. Programación Declarativa

Programación Declarativa

- ¿Que es la programación declarativa?
 - Paradigma de programación diferente a la imperativa (**R**) o la orientada a objetos (**Java**).
 - Los programas especifican las propiedades del problema a resolver.
 - La ejecución del programa consiste en “encontrar” la(s) solución(es).

Asignación Destructiva	Recursión
1 <code>def sumalista(lista):</code>	1 <code>sumaLista :: [Int] -> Int</code>
2 <code> sum = 0</code>	2 <code>sumaLista [] = 0</code>
3 <code> for elem in lista:</code>	3 <code>sumaLista (n:list) = n + (sumaLista list)</code>
4 <code> sum += elem</code>	4
5 <code> return sum</code>	5
6	6
7 <code>print(sumalista([1,2,3,4]))</code>	7 <code>main = print (sumaLista [1,2,3,4])</code>

Ejemplos de programación declarativa:

- Lenguajes funcionales: Haskell, Scala by EPFL.
- Lenguajes lógicos: Prolog, ASP, Logica by Google.
- Lenguajes algebraicos: Maude, SQL.
- etc...

4.2.2. Programación Funcional

- La programación funcional esta basada en funciones matemáticas.
- Función: Una función es una regla de correspondencia entre dos conjuntos de tal manera que a cada elemento del primer conjunto le corresponde uno y sólo un elemento del segundo conjunto.
- Cualquier función computable puede expresarse y evaluarse con el cálculo lambda.
- El cálculo lambda fue usado por Church para resolver el Entscheidungsproblem (1936):
 - No hay un algoritmo que determine si dos expresiones lambda arbitraria son equivalentes.

Cálculo Lambda

- Introducción al cálculo lambda.
- Reglas de formación de las expresiones lambda (λ -expresiones):
 - x es una λ -expresión si x es una variable.
 - $(\lambda x. t)$ es una λ -expresión (función) si t una expresión y x una variable.
 - $(t s)$ es una λ -expresión (aplicación) si t y s son expresiones.

Evaluando λ -expresiones

Función identidad aplicada al 3: $((\lambda x. x) 3) \equiv 3$

Función suma aplicada al 2 y el 3: $((\lambda x. \lambda y. x+y) 2) 3 \equiv ((\lambda y. 2+y) 3) \equiv (2+3)$

Función identidad aplicada a la suma: $((\lambda x. x) (\lambda x. \lambda y. x+y)) \equiv (\lambda x. \lambda y. x+y)$

Haskell

Programación Funcional: Haskell

en honor a Haskell Curry (1900-1982).

- *Currificación*: Transforma $f : (X_1 \times X_2 \times \dots \times X_n) \rightarrow Z$ en una secuencia de funciones con un único argumento:

$$\text{curry}(f) : X_1 \rightarrow X_2 \rightarrow \dots \rightarrow X_n \rightarrow Z.$$



Ejemplos de funciones en Haskell

```

1 suma :: Int -> Int -> Int           % suma a y b
2 suma a b = a + b                   % version con dos argumentos
3 suma' = \a -> \b -> a + b          % version currificada
4 sucesor :: Int -> Int               % sucesor de a
5 sucesor = suma 1
6 aplica :: (Int -> Int) -> [Int] -> [Int]
7 aplica _ [] = []                   % caso base
8 aplica f (n : list) = ((f n) : (aplica f list)) % caso recursivo
9
10 main = print (aplica sucesor [1,3,4]) % imprime [2,4,5]
```

Booleanos en cálculo lambda

- Definición de los booleanos en λ :
 - true: $\lambda x. \lambda y. x$
 - false: $\lambda x. \lambda y. y$
 - If-then-else: $\lambda x. \lambda y. \lambda z. x y z$

Evaluación

If-then-else True P Q $\equiv (\lambda x. \lambda y. \lambda z. x y z) (\lambda x. \lambda y. x) P Q \equiv (\lambda x. \lambda y. x) P Q \equiv P$

- Implementación usando Haskell:

```

1 true = \x -> \y -> x
2 false = \x -> \y -> y
3 if_then_else = \x -> \y -> \z -> x y z
4
5 k = if_then_else true 3 2           % ¿cuánto vale k?
```

- .. ahora, basadas en estas expresiones definimos And, Or y Not:

- And: $\lambda p. \lambda q. p q \text{ false} \equiv \lambda p. \lambda q. p q (\lambda x. \lambda y. y)$
- Or: $\lambda p. \lambda q. p \text{ true } q \equiv \lambda p. \lambda q. p (\lambda x. \lambda y. x) q$
- Not: $\lambda p. p \text{ false } \text{true} \equiv \lambda p. p (\lambda x. \lambda y. y) (\lambda x. \lambda y. x)$

Evaluación

$\text{And True False} \equiv (\lambda p. \lambda q. p \ q \ (\lambda x. \lambda y. y)) (\lambda x_1. \lambda y_1. x_1) (\lambda x_2. \lambda y_2. y_2) \equiv$
 $(\lambda x_1. \lambda y_1. x_1) (\lambda x_2. \lambda y_2. y_2) (\lambda x. \lambda y. y) \equiv (\lambda x_2. \lambda y_2. y_2) \equiv \text{False}$
 $\text{Or True False} \equiv (\lambda p. \lambda q. p \ (\lambda x. \lambda y. x) \ q) (\lambda x_1. \lambda y_1. x_1) (\lambda x_2. \lambda y_2. y_2) \equiv$
 $(\lambda x_1. \lambda y_1. x_1) (\lambda x. \lambda y. x) (\lambda x_2. \lambda y_2. y_2) \equiv (\lambda x. \lambda y. x) \equiv \text{True}$
 $\text{Not True} \equiv (\lambda p. p \ (\lambda x. \lambda y. y) \ (\lambda x. \lambda y. x)) (\lambda x_1. \lambda y_1. x_1) \equiv$
 $\dots \equiv (\lambda x. \lambda y. y) \equiv \text{False}$

- Implementación usando Haskell (cont.):

```

6 my_and = \x -> \y -> x y false
7 my_or  = \x -> \y -> x true y
8 my_not = \x -> x false true
9
10 k = if_then_else (my_and true false) 3 2 % ¿cuánto vale k?

```

- Alternativa para And, Or y Not:

- And: $\lambda p. \lambda q. p \ q \ p$
- Or: $\lambda p. \lambda q. p \ p \ q$
- Not: $\lambda p. \lambda x. \lambda y. p \ y \ x$ ¿Cuántos argumentos tiene?

Evaluación

Sin hacer :-), deberes para casa

- Implementación usando Haskell:

```

6 {-# LANGUAGE Rank2Types #-}
7 type CB = forall a . a -> a -> a % Ojo, requiere tipos
8
9 my_and :: CB -> CB -> CB
10 my_and = \p -> \q -> p q p
11 my_or  :: CB -> CB -> CB
12 my_or  = \p -> \q -> p p q
13 my_not :: CB -> CB % tiene 1 argumento !!!
14 my_not = \p -> \x -> \y -> p y x

```

Características y Ventajas**Características:**

- Evaluación de funciones vs. ejecución de instrucciones (recursión vs. iteración).
- El valor de una función sólo depende de sus argumentos (siempre se obtiene el mismo valor, transparencia referencial).
- Las funciones son “ciudadanos de primera clase” (argumentos y/o valores)

Ventajas:

- Código más limpio, conciso y expresivo.

- Sin efectos secundarios, al ser el estado inmutable.
 - Adecuado para sistemas concurrentes/paralelos.
- Permite verificación formal y demostración automática.

Concatenar listas

```

1 concatenar :: [a] -> [a] -> [a]           % declaración de tipos
2 concatenar [] list = list                 % caso base
3 concatenar (x:xs) list = (x: (concatenar xs list)) % recursión
4
5 k = concatenar [1,2] [3,4]                % ¿cuánto vale k?

```

4.2.3. Programación Lógica

- La programación lógica esta basada en lógica de 1^{er} orden (LPO).
- Predicados: Un predicado es una afirmación sobre propiedades de un objeto y/o una relación entre dos o más objetos.
- Dado un conjunto de fórmulas inferimos nuevo conocimiento. Pej.:

$$T[\forall x (Hombre(x) \rightarrow Mortal(x)), Hombre(socrates)] \vdash Mortal(socrates)$$

1 Se reescribe como el siguientes conjunto de cláusulas:

$$\{ Mortal(x) \vee \neg Hombre(x), Hombre(socrates), \neg Mortal(socrates) \}$$

donde el consecuente, $Mortal(socrates)$, están negado.

2 Si es **insatisfacible**, significa que hay consecuencia lógica.

3 Se resuelve aplicando método de Robinson con estrategia SLD.

Cláusulas Horn

- Introducción a las cláusulas de Horn, definidas por Alfred Horn en 1951.
- Dada una cláusula (disyunción de literales) cualquiera $L_1 \vee L_2 \vee \dots \vee L_n$, es una cláusula de Horn si tiene como máximo un literal positivo y esta reescrita como una implicación. Por ejemplo:

$\neg p \vee \neg q \vee \dots \vee \neg t \vee u$	es una regla y se reescribe como	$p \wedge q \wedge \dots \wedge t \rightarrow u$
u	es un hecho y se reescribe como	u
$\neg p \vee \neg q \vee \dots \vee \neg t$	sin literal positivo, es una consulta	$p \wedge q \wedge \dots \wedge t \rightarrow$

Aristóteles:

$Hombre(x) \rightarrow Mortal(x)$
 $Hombre(socrates)$

$Mortal(socrates)$

Cláusulas:

$Mortal(x) \vee \neg Hombre(x)$
 $Hombre(socrates)$

$\neg Mortal(socrates)$

Prolog:

`mortal(X) :- hombre(X).`
`hombre(socrates).`

`?- mortal(socrates).`

Prolog

- **Predicados:** Transforma $f : (X_1 \times X_2 \times \dots \times X_n) \rightarrow Z$ en una relación (n+1)-aria R y define el predicado r tal que:

$$r(x_1, x_2, \dots, z_n, z) = true \iff (x_1, x_2, \dots, z_n, z) \in R.$$



SWI-Prolog



Ciao Prolog

Aristóteles

```

1 mortal(X) :- hombre(X).           % Todos los hombres son mortales.
2 hombre(socrates).                % Sócrates es un hombre.
3
4 ?- mortal(socrates).              % ¿Sócrates es mortal?

```



Al invocar la consulta `?- mortal(socrates).`, Prolog contesta **yes** si la “pregunta” es consecuencia lógica (**no** en caso contrario).

Concatenar listas

```

1 concatenar([],Lista,Lista).
2 concatenar([X|Xs],Lista,[X|N_Lista]) :- concatenar(Xs,Lista,N_Lista).
3
4 ?- concatenar([1,2],[3,4],Lista).  % ¿Cuánto vale Lista?

```



Al invocar la consulta `?- concatenar([1,2],[3,4],Lista).`, Prolog devuelve la(s) sustitución(es) que la hace(n) consistente: `Lista = [1,2,3,4] ?`

Hay algo más ...

- Mientras las funciones devuelven un único resultado:
`k = concatenar [1,2] [3,4]`
- Los predicados pueden “consultarse” de diferentes formas sin cambiar el programa:
`?- concatenar([1,2], L, [1,2,3,4]).`
 devuelve:
`L = [3,4] ?`
- Pero, hay algo más ...

?- concatenar(LA, LB, [1,2,3,4]).

devuelve 5 respuestas:

1. LA = [], LB = [1,2,3,4] ?;
2. LA = [1], LB = [2,3,4] ?;
3. LA = [1,2], LB = [3,4] ?;
4. LA = [1,2,3], LB = [4] ?;
5. LA = [1,2,3,4], LB = [] ?

Esto permite usar un único predicado para codificar/decodificar mensajes 

```


1 char2morse('A','.-'). char2morse('B','-...'). char2morse('C','-.-.')...
2
3 ?- char2morse('B', Morse).           % devuelve Morse = '-... '
4 ?- char2morse(Char, '-.-.').        % devuelve Char = 'C'

```

... mucho más (CLP)

- Constraint Logic Programming (CLP): Incorpora restricciones que nos permite expresar relaciones entre variables mediante ecuaciones:

$$\begin{cases} 3x + 5y = 2 \\ 5x + 3y = -2 \end{cases} \quad \begin{array}{l} 1 \text{ sol}(X,Y) :- \\ 2 \quad 3 * X + 5 * Y \# = 2, \\ 3 \quad 5 * X + 3 * Y \# = -2. \end{array} \quad \begin{array}{l} ?- \text{ sol}(X,Y). \\ \quad X = -1, \\ \quad Y = 1 ? \end{array}$$

CLP(Q) nos permite definir la relación de una hipoteca como: 

```

1 mg(P,T,_,_,B) :- T \# = 0, B \# = P.
2 mg(P,T,R,I,B) :- T \# >= 1, NP \# = P + P*I - R, NT \# = T - 1, mg(NP,NT,R,I,B).

```

P=principal, T=time periods, R=repayment each period, I=interest rate, B=balance owing.

Podemos consultar de diferentes formas:

... y mucho más.

```

?- mg(1000,10,150,0.10,B). ?- mg(P,10,150,0.10,0). ?- mg(P,10,R,0.10,B).
   B = 203.13 ?           P = 921.68 ?           P = 6.14*R + 0.38*B ?

```

4.2.4. UTD HackReason

“Una cosa más...”*

UTD HackReason 2021, 2022, ... : January 14-15 (World Logic Day)



*“One More Thing...” is a reference to a practice that started in 1999, where Steve Jobs would leave (often quite big) announcements to the end of a presentation.