Tema 2. Lógica Proposicional

Asignatura: *Lógica y Métodos Discretos*
Wuolah: Bl4ckTyson

Voy a intentar hacer un documento resumido de manera enriquecida. El resumen contendrá información de diapositivas, ejercicios, vídeos...

Bl4ckTyson

2024

Contents

1 Sintaxis del lenguaje proposicional

Sea X un conjunto distinto de vacío finito, definimos un conjunto de fórmulas Form(X) de la siguiente forma:

- Todo elemento de X es Form(X)
- Si α, β son fórmulas sobre X, entonces también lo son:

$$\alpha \vee \beta, \quad \alpha \wedge \beta, \quad \alpha \to \beta, \quad \alpha \leftrightarrow \beta, \quad \neg\alpha$$

- Cualquier elemento que no se obtenga siguiendo estas reglas no es Form(X).

1.0.1 Vocabulario

- Los elementos de Form(X) se consideran fórmulas
- Los elementos del conjunto $C = \{\vee, \wedge, \to, \leftrightarrow, \neg\}$ se llaman conectivas.
- Los elementos de $X \cup C$ son los símbolos del lenguaje. Una fórmula es una sucesión de símbolos del lenguaje que siguen las reglas descritas anteriormente

1.0.2 Tabla de conectivas

Conectiva	Nombre	Fórmula	Lectura
$\vee$	Disyunción	$\alpha \vee \beta$	α o β
$\wedge$	Conjunción	$\alpha \wedge \beta$	α y β
$\to$	Implicación	$\alpha \to \beta$	α implica β
$\leftrightarrow$	Equivalencia	$\alpha \leftrightarrow \beta$	α equivale a β
$\neg$	Negación	$\neg\alpha$	No α

1.1 Reglas de prioridad

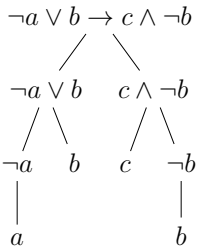
Para deshacernos de la ambigüedad en las fórmulas, utilizaremos los paréntesis solo en los casos necesarios siguiendo las siguientes reglas de prioridad:

1. La negación tiene más prioridad que el resto de conectivas
   Si $\beta_1 = a \to b \implies \neg\beta_1 = \neg(a \to b)$ y no $\neg a \to b$
2. Las conectivas $\vee$ y $\wedge$ tienen prioridad sobre $\to$ y $\leftrightarrow$
   Si $\beta_1 = a$ y $\beta_2 = b \to c$ entonces $\beta_1 \vee \beta_2 = a \vee (b \to c)$ y no $a \vee b \to c$
3. Las conectivas $\wedge$ y $\vee$ tienen la misma prioridad.
4. Las conectivas $\to$ y $\leftrightarrow$ tienen la misma prioridad
5. Cuando tengamos las conectivas $\{\vee, \wedge, \leftrightarrow\}$ dos o más veces seguidas, se asocia por la izquierda.
6. Cuando tengamos la conectiva $\to$ dos o más veces seguidas asociamos por la derecha.

1.2 Estructuras de árbol, subfórmulas y notación polaca

1.2.1 Estructuras de arbol

Sea $\alpha = \neg a \vee b \to c \wedge \neg b$.
Vamos a representar α mediante un diagrama en árbol



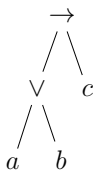
Las hojas son fórmulas atómicas. Si se recorre de abajo arriba se construye la fórmula

1.2.2 Subformulas

Todas las fórmulas que aparecían en el árbol anterior son subfórmulas de α. Serán Sub(α)
Dado que en nuestra fórmula tenemos que un grupo de subfórmulas implica otro, podemos darle nombre:
$\alpha_1 = \neg a \vee b$
$\alpha_2 = c \wedge \neg b$
$\alpha = \alpha_1 \to \alpha_2 \implies \text{Sub}(\alpha) = \text{Sub}(\alpha_1) \to \text{Sub}(\alpha_2) =$
$= \{\neg a \vee b \to c \wedge \neg b\} \cup \{\neg a \vee b, \neg a, b, a\} \cup \{c \wedge \neg b, c, \neg b, b\}$

1.2.3 Notación polaca

Se caracteriza porque las conectivas van al comienzo en lugar de entre dos formulas. El objetivo de usarla es descartar el uso de paréntesis ya que elimina la ambigüedad.
Ejemplo: Si $\alpha = a \vee b \to c$ y $\beta = a \vee (b \to c)$
$\alpha = \to \vee$ a b c
$\beta = \vee$ a $\to$ b c
Para pasar de forma polaca a arbol solo hay que ir colocando en orden de aparición en el arbol. En nuestro caso con alpha:



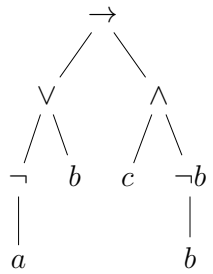
Si quisieramos volver a notación polaca, lo recorremos en preorden

2 Semantica del lenguaje proposicional

2.1 Ejemplo evaluación de interpretacion

$\alpha = \neg a \vee b \rightarrow c \wedge \neg b \; I(a) = 0, I(b) = 1, I(c) = 0$

1. Ponemos α como arbol



2. Recorremos el arbol en postorden
   α = a ¬ b ∨ c b ¬ ∧ →

3. Evaluamos la interpretacion con un sistema de pila. Iremos metiendo los valores en una pila imaginaria para ir comparándolos con las diferentes conectivas.

   (a) I(A) = 0. Dado que no tenemos nada en la pila, metemos el 0 directamente.

0

   (b) ¬. Lo aplicamos a lo que existe en la pila. $\neg 0 = 1$

1

   (c) I(B) = 1. Dado que existe un uno, lo añadimos a la pila

1
1

   (d) ∨. Aplicamos la conectiva en lo existente de la pila $(1 \vee 1 = 1)$

1

   (e) I(C) = 0. Añadimos a la pila el 0.

0
1

   (f) I(B) = 1. Añadimos a la pila el 1

1
0
1

   (g) ¬. Se aplica la conectiva al top de la pila $(\neg 1 = 0)$

0
0
1

(h) $\wedge$. Se aplica la conectiva al top (los dos) de la pila. $(0 \wedge 0 = 0)$

0
1

(i) $\rightarrow$. Se aplica la conectiva al top de la pila. $(1 \rightarrow 0 = 0)$

0

(j) $I(\alpha) = 0$

2.2 Polinomio de Gegalkine

Para calcular el polinomio de Gegalkine hay que pasar α como suma y productos de variables, 1 y 0. Ejemplo de resolución.
$\alpha = \neg a \vee b \rightarrow c \wedge \neg b$

$\neg a \vee b = \neg a + b + (\neg a) \cdot b = 1 + a + b + (1 + a) \cdot b = 1 + a + b + b + a \cdot b = 1 + a + a \cdot b$
$c \wedge \neg b = c \cdot (\neg b) = c \cdot (1 + b) = c + c \cdot b$
$\alpha = 1 + (\neg a \vee b) + (\neg a \vee b) \cdot (c \wedge \neg b) =$
$= 1 + (1 + a + a \cdot b) + (1 + a + a \cdot b) \cdot (c + b \cdot c) =$
$= 1 + 1 + a + a \cdot b + c + b \cdot c + a \cdot c + a \cdot b \cdot c + a \cdot b \cdot c + a \cdot b \cdot b \cdot c =$
$= a + c + a \cdot b + a \cdot c + b \cdot c + a \cdot b \cdot c$ Para dejarlo correcto y completo. Hay que pasarlo a interpretación añadiendo un 1 por delante.
$I(\neg a \vee b \rightarrow c \wedge \neg b) = 1 + I(a) + I(c) + I(a) \cdot I(b) + I(a) \cdot I(c) + I(b) \cdot I(c) + I(a) \cdot I(b) \cdot I(c)$

2.3 Ejemplo de Tablas de verdad

Sea $\alpha = a \wedge \neg b \rightarrow \neg c$

a	b	c	$\neg b$	$\neg c$	$a \wedge \neg b$	$a \wedge \neg b \rightarrow \neg c$
0	0	0	1	1	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	1
0	1	1	0	0	0	1
1	0	0	1	1	0	1
1	0	1	1	0	0	0
1	1	0	0	1	1	1
1	1	1	0	0	1	1

2.4 Clasificación semántica de las fórmulas

- Tautología: Para cualquier interpretación se tiene que $I(\alpha) = 1$
- Satisfacible: Existe una Interpretación para la cual $I(\alpha) = 1$
- Refutable: Existe una Interpretación para la cual $I(\alpha) = 0$
- Contradicción: Para cualquier interpretación se tiene que $I(\alpha) = 0$
- Contigente: Cuando una expresión es satisfacible y refutable

Toda tautología es satisfacible y toda contradicción es refutable.

3 Equivalencia lógica ($\equiv$)

Dos fórmulas son equivalentes lógicamente si para cualquier interpretación: $I(\alpha) = I(\beta)$

3.1 Algunas equivalencias lógicas importantes

1.	$\neg\neg\alpha \equiv \alpha$	2.	$\alpha \leftrightarrow \beta \equiv (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$
3.	$\neg(\alpha \leftrightarrow \beta) \equiv \neg\alpha \leftrightarrow \beta$	4.	$\alpha \rightarrow \beta \equiv \neg\alpha \vee \beta$
5.	$\neg(\alpha \vee \beta) \equiv \neg\alpha \wedge \neg\beta$	6.	$\neg(\alpha \wedge \beta) \equiv \neg\alpha \vee \neg\beta$
7.	$\alpha \vee (\beta \wedge \gamma) \equiv (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$	8.	$\alpha \wedge (\beta \vee \gamma) \equiv (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$

4 Implicación Semántica

4.1 Conjuntos satisfacibles e insatisfacibles

Teorema 1 *Sea Γ un conjunto de fórmulas en lenguaje proposicional, decimos que es satisfacible si existe alguna interpretación para la que son ciertas todas las fórmulas de Γ*
El conjunto Γ es insatisfacible si para cualquier interpretación I existe un $\gamma \in \Gamma$ tal que $I(\gamma) = 0$

Por ahora, podemos resolver si un conjunto Γ es insatisfacible usando Tablas de verdad o el polinomio de Gegalkine.

4.2 Implicación semántica

Teorema 2 *Sea Γ un conjunto de fórmulas construido sobre un conjunto y α una fórmula del mismo lenguaje, Γ implica semánticamente a α o que α es consecuencia lógica de Γ si para cualquier interpretación se tiene que $I(\alpha) = 1$*
Se escribirá como $\Gamma \models \alpha$

4.2.1 Ejemplo resolución Implicación Semántica

$\Gamma = \{b \to c \vee a, a \leftrightarrow \neg(b \wedge d), d \to a \wedge b\}$
$\alpha = b \leftrightarrow c \vee d$
Queremos ver si $\Gamma \models \alpha$
Llamaremos $\gamma_1 = \{b \to c \vee a\}$
Llamaremos $\gamma_2 = \{a \leftrightarrow \neg(b \wedge d)\}$
Llamaremos $\gamma_3 = \{d \to a \wedge b\}$
Lo resolveremos con tabla de verdad

a	b	c	d	γ_1	γ_2	γ_3	α
0	0	0	0	1	0	1	
0	0	0	1	1	0	0	
0	0	1	0	1	0	1	
0	0	1	1	1	0	0	
0	1	0	0	0	0	1	
0	1	0	1	1	1	0	
0	1	1	0	0	0	1	
0	1	1	1	1	1	0	
1	0	0	0	**1**	**1**	**1**	1
1	0	0	1	1	1	0	
1	0	1	0	**1**	**1**	**1**	**0**
1	0	1	1	1	1	0	
1	1	0	0	**1**	**1**	**1**	**0**
1	1	0	1	1	0	1	
1	1	1	0	**1**	**1**	**1**	
1	1	1	1	1	0	1	1

Como vemos, hay 4 interpretaciones que cumplen γ_1, γ_2, γ_3, de las cuales dos son falsas, lo que nos indica que γ no es consecuencia lógica de α

4.3 Teorema de Refutacion y deduccion

4.3.1 Teorema de refutación

Teorema 3 *Γ es consecuencia lógica de α si $\Gamma \cup \{\neg\alpha\}$ es insatisfacible*

4.3.2 Teorema de deduccion

Teorema 4 *Sea Γ un conjunto de fórmulas en lenguaje proposicional y α y β dos fórmulas de ese lenguaje, entonces:*
$\Gamma \models \alpha \to \beta$ *si* $\Gamma \cup \{\alpha\} \models \beta$

5 Resolución implicación semántica

5.1 Forma clausulada

5.1.1 Vocabulario

- Los literales son una forma atómica o su negada: $a, b, c, \neg a, \neg b, \neg c$
- Las cláusulas son disyunciones de literales: $a \vee b, \neg a \vee c, a \vee b \vee \neg c, a, \neg c, \square$

Teorema 5 *Sea α una fórmula de un lenguaje proposicional, existe una fórmula equivalente a α que está en forma clausulada.*

5.1.2 Cálculo de forma clausulada

- Eliminar equivalencia: $\beta \leftrightarrow \gamma \equiv (\beta \rightarrow \gamma) \wedge (\gamma \rightarrow \beta)$
- Eliminar implicción: $\beta \rightarrow \gamma \equiv \neg\beta \vee \gamma$

Cálculo de ejemplo: $\alpha = ((a \vee \neg b) \rightarrow (\neg c \rightarrow b)) \wedge ((a \rightarrow b) \leftrightarrow c)$
Calcularemos α_1 y α_2 por separado debido a que puede escribirse como $\alpha_1 \wedge \alpha_2$
$\alpha_1 = (a \vee \neg b) \rightarrow (\neg c \rightarrow b) \equiv$
$\equiv \neg(a \vee \neg b) \vee (c \vee b) \equiv$
$\equiv (\neg a \wedge b) \vee c \vee b \equiv$
$\equiv (\neg a \vee b \vee c) \wedge (b \vee c)$

$\alpha_2 = ((a \rightarrow b) \leftrightarrow c) \equiv$
$\equiv ((a \rightarrow b) \rightarrow c) \wedge (c \rightarrow (a \rightarrow b)) \equiv$
$\equiv ((\neg a \vee b) \rightarrow c) \wedge (c \rightarrow (\neg a \vee b)) \equiv$
$\equiv (\neg(\neg a \vee b) \vee c) \wedge (\neg c \vee (\neg a \vee b)) \equiv$
$\equiv ((a \wedge \neg b) \vee c) \wedge (\neg c \vee (\neg a \vee b)) \equiv$
$\equiv (a \vee c) \wedge (\neg b \vee c) \wedge (\neg a \vee b \vee \neg c))$

$\alpha = \alpha_1 \wedge \alpha_2 \equiv (\neg a \vee b \vee c) \wedge (b \vee c) \wedge (a \vee c) \wedge (\neg b \vee c) \wedge (\neg a \vee b \vee \neg c) \equiv$
$\equiv (\neg a \vee b) \wedge c \wedge (a \vee c) \equiv$
$\equiv (\neg a \vee b) \wedge c$

5.2 Algoritmo de Davis-Putnam

Para poder resolver una consecuencia lógica por Davis-Putnam tiene que estar en forma clausulada.

5.2.1 Ejemplo de aplicacion

$$\{b \vee c, \neg a \vee b \vee c \vee d, a \vee \neg c \vee d, \neg a \vee d, \neg b \vee \neg e, c \vee \neg d, a \vee d, \neg c \vee d\}$$

1. Empezamos con $\lambda = \neg e$ (por ejemplo). Tenemos que comparar con el conjunto si las clausulas tienen algun literal $= \lambda$, en cuyo caso se elimina la cláusula o igual al complementario de lambda, en cuyo caso solo se elimina el literal.

$$\{b \vee c, \neg a \vee b \vee c \vee d, a \vee \neg c \vee d, \neg a \vee d, c \vee \neg d, a \vee d, \neg c \vee d\}$$

2. Sigamos con $\lambda = b$

$$\{a \vee \neg c \vee d, \neg a \vee d, c \vee \neg d, a \vee d, \neg c \vee d\}$$

3. Sigamos con $\lambda = a$ y $\lambda = \neg a$

   - $\lambda = a$
   $$\{d, c \vee \neg d, \neg c \vee d\}$$
   - $\lambda = \neg a$
   $$\{\neg c \vee d, c \vee \neg d, d, \neg c \vee d\}$$

4. Sigamos con $\lambda = d$ sobre el resultado de $\lambda = a$

$$\{c\}$$

5. Finalizamos con $\lambda = c$

$$\{\varnothing\}$$

6. Como no hemos llegado a $\square$ el conjunto es satisfacible. Además hemos encontrado una interpretacion para que todas las clausulas son ciertas:

   - $I(a) = 1$
   - $I(b) = 1$
   - $I(c) = 1$
   - $I(d) = 1$
   - $I(e) = 0$

5.3 Algoritmo de resolucion

5.3.1 Ejemplo de aplicacion

Vamos a partir de este α

$$\alpha = [[(\varphi \to \psi) \to (\neg\chi \to \neg\theta)] \to \chi) \to \tau] \to [(\tau \to \varphi) \to (\theta \to \varphi)]\,.$$

1. Extraemos los diferentes α_n
   - $\alpha_1 = ([(\varphi \to \psi) \to (\neg\chi \to \neg\theta)] \to \chi) \to \tau]$
   - $\alpha_2 = (\tau \to \varphi)$
   - $\alpha_3 = (\theta)$
   - $\alpha_4 = (\neg\varphi)$
2. Pasamos a forma clausulada cada fórmula
   - $\alpha_1 = ([(\varphi \to \psi) \to (\neg\chi \to \neg\theta)] \to \chi) \to \tau \equiv$
     $\equiv \neg([(\varphi \to \psi) \to (\neg\chi \to \neg\theta)] \to \chi) \vee \tau \equiv$
     $\equiv \neg(\neg[(\varphi \to \psi) \to (\chi \to \neg\theta)] \vee \chi) \vee \tau \equiv$
     $\equiv \neg(\neg[\neg(\varphi \to \psi) \vee (\chi \to \neg\theta)] \vee \chi) \vee \tau \equiv$
     $\equiv \neg(\neg[\neg(\neg\varphi \vee \psi) \vee (\neg\chi \vee \neg\theta)] \vee \chi) \vee \tau \equiv$
     $\equiv ([\neg(\neg\varphi \vee \psi) \vee (\neg\chi \vee \neg\theta) \wedge \neg\chi) \vee \tau \equiv$
     $\equiv ([\varphi \wedge \neg\psi) \vee (\neg\chi \vee \neg\theta) \wedge \neg\chi) \vee \tau \equiv$
     $\equiv ([(\varphi \vee \chi \vee \neg\theta) \wedge (\neg\psi \vee \chi \vee \neg\theta)] \wedge \neg\chi) \vee \tau \equiv$
     $\equiv (\varphi \vee \chi \vee \neg\theta \vee \tau) \wedge (\neg\psi \vee \chi \vee \neg\theta \vee \tau) \wedge (\neg\chi \vee \tau)$
   - $\alpha_2 = (\tau \to \varphi) \equiv$
     $\equiv (\neg\tau \vee \varphi)$
   - $\alpha_3 = (\theta)$
   - $\alpha_4 = (\neg\varphi)$
3. Vamos a poner en el conjunto α' todas las clausulas
   $\alpha' = \{(\varphi \vee \chi \vee \neg\theta \vee \tau), (\neg\psi \vee \chi \vee \neg\theta \vee \tau), (\neg\chi \vee \tau), (\neg\tau \vee \varphi), (\theta), (\neg\varphi)\}$
4. Empezamos a resolver si α' es insatisfacible, ya que si lo es, α también lo será
   (a) Empezamos con la cláusula $(\varphi \vee \chi \vee \neg\theta \vee \tau)$
   Utilizamos como resolvente $(\neg\chi \vee \tau)$ y se nos queda $(\varphi \vee \neg\theta \vee \tau)$
   (b) Integramos esto con la clausula (θ) y se nos queda $(\varphi \vee \tau)$
   (c) Integramos esto con la clausula $(\neg\tau \vee \varphi)$ y se nos queda (φ)
   (d) Integramos finalmente con la clausula $(\neg\varphi)$ y se nos queda $\square$
5. Como hemos llegado a $\square$, el conjunto α' es insatisfacible, por lo tanto, α es insatisfacible, lo que demuestra que la expresión es una tautologia

5.4 Deducciones lineales

Teorema 6 *Una deducción es lineal si en todos los cálculos de resolventes se ha usado la resolvente obtenida en el paso anterior*

5.5 Deducciones input

Teorema 7 *Una deducción es input si en el cálculo de cada una de los resolventes se ha usado al menos una cláusula del conjunto*

5.6 Deducciones lineales-input Cláusulas de Horn

.

Teorema 8 *Un conjunto es conjunto de Hornn si tiene una cláusula negativa (todos los literales son negativos) y el resto son cláusulas de horn (tienen exactamente un literal positivo)*

Teorema 9 *Si tenemos un conjunto de Hornn, es insatisfacible si y solo si existe una deducción lineal-input de la cláusula vacía que empiece en la cláusula negativa.*

5.6.1 Procedimiento para pasar un conjunto a conjunto de hornn

Partamos del conjunto $\Gamma = \{\neg a \vee b, a \vee c, \neg c \vee \neg d, \neg b \vee \neg c, \veebar \neg c, d\}$
Vamos a pasarlo a una tabla donde estudiaremos cada una de las cláusulas, poniendo, correspondiendole un 0 si el literal es positivo y un 1 si el literal es negativo. Nuestro objetivo es encontrar un cambio para el cual se genere el conjunto de horn (una clausula completamente negativa y el resto con solo 1 positivo).
Es decir, buscamos una fila la cual tenga una casilla con todo 1 y el resto con un 0 y el resto 1.
Este es el procedimiento:

1. Se selecciona una clausula de ejemplo.
2. Se realizan los cambios para que esa clausula aparezca negada.
3. Si hemos realizado algún cambio, hay que cambiar los valores correspondientes al resto de tablas. Por ejemplo, si hemos cambiado b por $\neg b$ para una clausula, hay que cambiar los valores de b para el resto.
4. Se realizan las modificaciones pertinentes para que el resto de clausulas tengan como máximo un 0 y el resto 1s. Si en algun momento nos topamos con una clausula con más de un 0 o una cláusula con solo 1s, descartamos el estudio de esa cláusula y estudiamos la siguiente.
5. Si conseguimos que alguna cláusula genere el conjunto de Hornn tras realizar cambios, nos quedamos con esos cambios y resolvemos el sistema con la deducción lineal-input