Matemáticas Discretas

Lógica proposicional

Nicolás Alvarado
nfalvarado@mat.uc.cl

Bernardo Barías
bjbarias@uc.cl

Sebastián Bugedo
bugedo@uc.cl

Gabriel Diéguez
gsdieguez@ing.puc.cl

Departamento de Ciencia de la Computación
Escuela de Ingeniería
Pontificia Universidad Católica de Chile

28 de agosto de 2023

Objetivos

1. Formular enunciados formales en notación matemática usando lógica, conjuntos, relaciones, funciones, cardinalidad, y otras herramientas, desarrollando definiciones y teoremas al respecto, así como demostrar o refutar estos enunciados, usando variadas técnicas.
2. Aplicar inducción como técnica para demostración de propiedades en conjuntos discretos y como técnica de definición formal de objetos discretos.
3. Modelar formalmente un problema usando lógica, conjuntos, relaciones, y las propiedades necesarias, y demostrar propiedades al respecto de su modelo.

Contenidos

¿Qué es lógica?

Definición

Lógica es el uso y estudio del **razonamiento válido**.

Wikipedia

Introducción

- La lógica (en su sentido más amplio) estudia la **inferencia**.
  - Cómo obtener conclusiones a partir de premisas.
- Originalmente, hacía esto en el lenguaje natural.

Ejemplo

¿Es el siguiente razonamiento válido?

$$\frac{\begin{array}{l}\text{Todos los hombres son mortales.}\\ \text{Sócrates es hombre.}\end{array}}{\text{Por lo tanto, Sócrates es mortal.}}$$

La lógica debería poder usarse para demostrar que sí.

Introducción

¿Qué pasa si usamos el lenguaje natural para definir cosas? Por ejemplo:

- Podemos representar los números naturales usando oraciones del lenguaje natural: "dos mil dieciocho", "el segundo número"...
- El diccionario de la RAE tiene una cantidad finita de palabras.
- El número de oraciones con a lo más 50 palabras también es finito.
- Por lo tanto, deben existir números naturales que no pueden ser representados con oraciones de a lo más 50 palabras.
- Por PBO, debe existir un menor número que cumpla lo anterior.

Introducción

Sea n el siguiente número natural:

El primer número natural que no puede ser definido por una oración con a lo más cincuenta palabras del diccionario de la Real Academia Española.

La oración anterior define a n con 25 palabras. **¡Contradicción!**

¿Qué pasó?

¿Por qué necesitamos la lógica?

- Necesitamos un lenguaje con una **sintaxis** precisa y una **semántica** bien definida.
  - **Qué** objetos pertenecen al lenguaje y qué **significan**.
- Queremos usar este lenguaje en matemáticas para hacer cosas que hasta ahora hemos hecho de manera intuitiva.
  - Definición de objetos matemáticos (conjuntos, números naturales, números reales. . .).
  - Definición de teorías matemáticas (teoría de conjuntos, teoría de los naturales. . .).
  - Definición del concepto de demostración.
- Pero también queremos usar este lenguaje en computación.

¿Por qué necesitamos la lógica en computación?

- Bases de datos: lenguajes de consulta, lenguajes para restricciones.
- Inteligencia artificial: representación de conocimiento, razonamiento.
- Ingeniería de software: especificación de sistemas, verificación de propiedades.
- Teoría de la computación: complejidad descriptiva, algoritmos de aproximación.
- Criptografía: verificación de protocolos.
- Procesamiento de lenguaje natural.
- ...

¿Cuántas lógicas existen?

- Lógica proposicional
- Lógica de predicados
- Lógica de primer orden
- Lógica de segundo orden
- Lógicas descriptivas
- Lógicas temporales
- Lógica intuicionista
- Lógica trivalente
- etc ...

¿Qué lógicas veremos en el curso?

- **Lógica proposicional**
- **Lógica de predicados**
- Lógica de primer orden
- Lógica de segundo orden
- Lógica de segundo orden monádica
- Lógica temporal lineal
- Lógica intuicionista
- Lógica trivalente
- etc ...

Más en..

- IIC2213 - Lógica para ciencias de la computación
- IIC3263 - Teoría de modelos finitos

Usaremos los siguientes elementos:

- Variables proposicionales: p, q, r, ...
- Conectivos lógicos: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
- Símbolos de puntuación: (,)

Las variables proposicionales representan proposiciones **completas** e **indivisibles**, que pueden ser **verdaderas** o **falsas**. En general llamaremos P al conjunto de ellas.

Ejemplo

$P = \{\text{Juan\_cursa\_IIC1253}, \text{Juan\_aprobo\_IIC1103}\}$

Sintaxis de la Lógica proposicional

Los conectivos lógicos se usan para construir expresiones más complejas que también pueden ser verdaderas o falsas.

Ejemplo

$$\text{Juan\_cursa\_IIC1253} \rightarrow \text{Juan\_aprobo\_IIC1103}$$
$$(\neg\text{Juan\_cursa\_IIC1253}) \rightarrow \text{Juan\_aprobo\_IIC1103}$$

Los símbolos de puntuación se usan para evitar ambigüedades.

Sintaxis de la Lógica proposicional

Definición

Sea P un conjunto de variables proposicionales. El conjunto de todas las **fórmulas** de lógica proposicional sobre P, denotado por $L(P)$, es el menor conjunto que cumple las siguientes reglas:

1. Si $p \in P$, entonces $p \in L(P)$.
2. Si $\varphi \in L(P)$, entonces $(\neg\varphi) \in L(P)$.
3. Si $\varphi, \psi \in L(P)$, entonces $(\varphi \wedge \psi) \in L(P)$, $(\varphi \vee \psi) \in L(P)$, $(\varphi \rightarrow \psi) \in L(P)$ y $(\varphi \leftrightarrow \psi) \in L(P)$.

¡Podemos usar Inducción estructural!

Sintaxis de la Lógica proposicional

Ejercicios

1. Defina la función $largo(\varphi)$ como el largo de una fórmula en lógica proposicional (variables, conectivos y paréntesis).
2. Defina la función $var(\varphi)$ como el número de ocurrencias de variables proposicionales en φ.
3. Demuestre que si φ no contiene el símbolo $\neg$, se cumple que $largo(\varphi) \leq 4 \cdot var(\varphi)^2$.
    - ¿Qué pasa si contiene el símbolo $\neg$?

Ejercicio

Defina la función $largo(\varphi)$ como el largo de una fórmula en lógica proposicional (variables, conectivos y paréntesis).

1. $largo(p) = 1$, con $p \in P$.
2. $largo((\neg\varphi)) = 3 + largo(\varphi)$, con $\varphi \in L(P)$.
3. $largo((\varphi \star \psi)) = 3 + largo(\varphi) + largo(\psi)$, con $\varphi, \psi \in L(P)$ y $\star \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$.

Ejercicio

Defina la función $var(\varphi)$ como el número de ocurrencias de variables proposicionales en φ.

1. $var(p) = 1$, con $p \in P$.
2. $var((\neg\varphi)) = var(\varphi)$, con $\varphi \in L(P)$.
3. $var((\varphi \star \psi)) = var(\varphi) + var(\psi)$, con $\varphi, \psi \in L(P)$ y $\star \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$.

Ejercicio

Demuestre que si φ no contiene el símbolo $\neg$, se cumple que $largo(\varphi) \leq 4 \cdot var(\varphi)^2$.

Demostración: usaremos inducción estructural.
BI: $largo(p) = 1 \leq 4 = 4 \cdot var(p)^2$.
HI: Supongamos que la propiedad se cumple para $\varphi, \psi \in L(P)$ tales que no tienen el símbolo $\neg$; es decir, $largo(\varphi) \leq 4 \cdot var(\varphi)^2$ y $largo(\psi) \leq 4 \cdot var(\psi)^2$.
TI: Necesitamos demostrar que $largo((\varphi \star \psi)) \leq 4 \cdot var((\varphi \star \psi))^2$.

Desarrollando: $largo((\varphi \star \psi)) = 3 + largo(\varphi) + largo(\psi)$

$$\overset{HI}{\leq} 3 + 4 \cdot var(\varphi)^2 + 4 \cdot var(\psi)^2 = 3 + 4(var(\varphi)^2 + var(\psi)^2)$$

$$= 3 + 4((var(\varphi) + var(\psi))^2 - 2 \cdot var(\varphi) \cdot var(\psi))$$

$$= 3 - 8 \cdot var(\varphi) \cdot var(\psi) + 4 \cdot (var(\varphi) + var(\psi))^2$$

Aplicando la definición de var obtenemos que $largo((\varphi \star \psi)) \leq 3 - 8 \cdot var(\varphi) \cdot var(\psi) + 4 \cdot var((\varphi \star \psi))^2$. Ahora, como $var(\theta) \geq 1$ para cualquier $\theta \in L(P)$, tenemos que $8 \cdot var(\varphi) \cdot var(\psi) \geq 8$ y por lo tanto $3 - 8 \cdot var(\varphi) \cdot var(\psi) \leq 0$. Finalmente, aplicando esta última desigualdad obtenemos que $largo((\varphi \star \psi)) \leq 4 \cdot var((\varphi \star \psi))^2$ como queríamos demostrar.

Por inducción estructural, se sigue que la propiedad es cierta para toda fórmula que no contiene $\neg$. $\square$

Ejercicio

¿Qué pasa si contiene el símbolo $\neg$?

Como la regla que permite ocupar el conectivo $\neg$ no agrega variables pero sí aumenta el largo, podríamos ocuparla y aumentar arbitrariamente el largo de una fórmula sin aumentar el número de variables, con lo que la propiedad no se cumpliría luego de algún número de $\neg$ agregadas. Una idea para limitar esto es no permitir el uso de más de una negación; i.e., no permitir fórmulas del tipo $(\neg(\neg\varphi))$ y sucesivas. Esto se fundamenta más adelante.

Semántica de la Lógica proposicional

¿Cuándo una fórmula es verdadera?

- Depende del "mundo" en que la estemos interpretando.
- Un "mundo" particular le dará una interpretación a cada variable proposicional.
  - Un valor verdadero o falso.
- Con estos valores y los conectivos utilizados, podremos determinar el valor de verdad de la fórmula.

Definición

Una **valuación** o **asignación de verdad** para las variables proposicionales en un conjunto P es una función $\sigma : P \to \{0, 1\}$.

Convención: $0 =$ falso, $1 =$ verdadero.

Semántica de la Lógica proposicional

Dados un conjunto P y una asignación de verdad σ para P, necesitamos extender la noción de valuación a las fórmulas en $L(P)$. ¿Cómo lo hacemos?

Definición

La función $\hat{\sigma} : L(P) \to \{0, 1\}$ se define como:

1. Si $p \in P$, entonces $\hat{\sigma}(p) = \sigma(p)$.
2. $\hat{\sigma}((\neg\varphi)) = \begin{cases} 1 & \text{si } \hat{\sigma}(\varphi) = 0 \\ 0 & \text{si } \hat{\sigma}(\varphi) = 1 \end{cases}$
3. $\hat{\sigma}((\varphi \wedge \psi)) = \begin{cases} 1 & \text{si } \hat{\sigma}(\varphi) = 1 \text{ y } \hat{\sigma}(\psi) = 1 \\ 0 & \text{si } \hat{\sigma}(\varphi) = 0 \text{ o } \hat{\sigma}(\psi) = 0 \end{cases}$

   $\hat{\sigma}((\varphi \vee \psi)) = \begin{cases} 1 & \text{si } \hat{\sigma}(\varphi) = 1 \text{ o } \hat{\sigma}(\psi) = 1 \\ 0 & \text{si } \hat{\sigma}(\varphi) = 0 \text{ y } \hat{\sigma}(\psi) = 0 \end{cases}$

Semántica de la Lógica proposicional

Definición

La función $\hat{\sigma} : L(P) \to \{0, 1\}$ se define como:

1. Si $p \in P$, entonces $\hat{\sigma}(p) = \sigma(p)$.
2. $\hat{\sigma}((\neg\varphi)) = \begin{cases} 1 & \text{si } \hat{\sigma}(\varphi) = 0 \\ 0 & \text{si } \hat{\sigma}(\varphi) = 1 \end{cases}$
3. $\hat{\sigma}((\varphi \wedge \psi)) = \begin{cases} 1 & \text{si } \hat{\sigma}(\varphi) = 1 \text{ y } \hat{\sigma}(\psi) = 1 \\ 0 & \text{si } \hat{\sigma}(\varphi) = 0 \text{ o } \hat{\sigma}(\psi) = 0 \end{cases}$

   $\hat{\sigma}((\varphi \vee \psi)) = \begin{cases} 1 & \text{si } \hat{\sigma}(\varphi) = 1 \text{ o } \hat{\sigma}(\psi) = 1 \\ 0 & \text{si } \hat{\sigma}(\varphi) = 0 \text{ y } \hat{\sigma}(\psi) = 0 \end{cases}$

   $\hat{\sigma}((\varphi \to \psi)) = \begin{cases} 1 & \text{si } \hat{\sigma}(\varphi) = 0 \text{ o } \hat{\sigma}(\psi) = 1 \\ 0 & \text{si } \hat{\sigma}(\varphi) = 1 \text{ y } \hat{\sigma}(\psi) = 0 \end{cases}$

   $\hat{\sigma}((\varphi \leftrightarrow \psi)) = \begin{cases} 1 & \text{si } \hat{\sigma}(\varphi) = \hat{\sigma}(\psi) \\ 0 & \text{si } \hat{\sigma}(\varphi) \neq \hat{\sigma}(\psi) \end{cases}$

- $\hat{\sigma}(\varphi)$ es la **evaluación** de φ dada la asignación σ y representa su valor de verdad.
- Por simplicidad, usaremos σ en lugar de $\hat{\sigma}$.

Ejemplo

Si tenemos una valuación σ tal que

$$\sigma(\text{Juan\_cursa\_IIC1253}) = 1$$
$$\sigma(\text{Juan\_aprobo\_IIC1103}) = 0$$

entonces

$$\sigma((\text{Juan\_cursa\_IIC1253} \rightarrow \text{Juan\_aprobo\_IIC1103})) = 0$$

Definición

Dos fórmulas $\varphi, \psi \in L(P)$ son **lógicamente equivalentes** (denotado como $\varphi \equiv \psi$) si para toda valuación σ se tiene que $\sigma(\varphi) = \sigma(\psi)$.

Ejercicio

Demuestre que $(p \to q) \equiv ((\neg p) \vee q)$.

Ejercicio

Demuestre que $(p \to q) \equiv ((\neg p) \lor q)$.

Demostración: Como tenemos 2 variables proposicionales, hay 4 valuaciones posibles. Podemos probarlas todas y verificar que siempre son iguales para ambas fórmulas. Una manera de hacer esto es ponerlas en una tabla como la siguiente:

	p	q	$(p \to q)$	$((\neg p) \lor q)$
σ_1 :	0	0	1	1
σ_2 :	0	1	1	1
σ_3 :	1	0	0	0
σ_4 :	1	1	1	1

Verificamos que para cada valuación posible el valor de verdad de las fórmulas es el mismo, con lo que demostramos que son equivalentes (por definición). $\square$

Tablas de verdad

Las fórmulas se pueden representar y analizar en una tabla de verdad.

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \leftrightarrow q$
0	0	1	0	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	0	0
1	1	0	1	1	1	1

Ejercicio

Si P tiene n variables proposicionales, ¿cuántas tablas de verdad distintas hay para $L(P)$?

Respuesta: Hay 2^n valuaciones posibles para n variables, por lo que cada tabla tiene 2^n filas. Cada una de estas filas puede tomar valor 0 o 1, por lo que podemos tener 2^{2^n} tablas distintas.

Note que $\varphi \equiv \psi$ si y sólo si sus respectivas columnas en una tabla de verdad son iguales.

- Podemos usar tablas de verdad para comparar fórmulas, como en el ejercicio que ya hicimos. Otro ejemplo:

p	q	r	$(p \wedge q) \wedge r$	$p \wedge (q \wedge r)$
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Tablas de verdad

p	q	r	$(p \wedge q) \wedge r$	$p \wedge (q \wedge r)$
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Concluimos que $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$.

- Es decir, el conectivo $\wedge$ es asociativo.
- Podemos omitir paréntesis cuando tengamos una cadena de ellos.
  - Escribimos simplemente $p \wedge q \wedge r$.

Leyes de equivalencia

Doble negación

$\neg(\neg\varphi) \equiv \varphi$

Leyes de De Morgan

$\neg(\varphi \wedge \psi) \equiv (\neg\varphi) \vee (\neg\psi)$
$\neg(\varphi \vee \psi) \equiv (\neg\varphi) \wedge (\neg\psi)$

Conmutatividad

$\varphi \wedge \psi \equiv \psi \wedge \varphi$
$\varphi \vee \psi \equiv \psi \vee \varphi$

Leyes de equivalencia

Asociatividad

$\varphi \wedge (\psi \wedge \theta) \equiv (\varphi \wedge \psi) \wedge \theta$

$\varphi \vee (\psi \vee \theta) \equiv (\varphi \vee \psi) \vee \theta$

Distributividad

$\varphi \wedge (\psi \vee \theta) \equiv (\varphi \wedge \psi) \vee (\varphi \wedge \theta)$

$\varphi \vee (\psi \wedge \theta) \equiv (\varphi \vee \psi) \wedge (\varphi \vee \theta)$

Idempotencia

$\varphi \wedge \varphi \equiv \varphi$

$\varphi \vee \varphi \equiv \varphi$

Leyes de equivalencia

Absorción

$\varphi \wedge (\varphi \vee \psi) \equiv \varphi$
$\varphi \vee (\varphi \wedge \psi) \equiv \varphi$

Implicancia

$\varphi \to \psi \equiv (\neg\varphi) \vee \psi$

Doble implicancia

$\varphi \leftrightarrow \psi \equiv (\varphi \to \psi) \wedge (\psi \to \varphi)$

Ejercicios

1. Demuestre las leyes enunciadas.
2. ¿Es $\to$ asociativo?

Ejercicio

1. Demuestre las leyes enunciadas.

Ya demostramos la ley asociativa para $\wedge$ y la ley de implicancia. El resto se dejan como ejercicios.

Ejercicios

2. ¿Es $\rightarrow$ asociativo?

Esto es lo mismo que preguntarnos si $\varphi \rightarrow (\psi \rightarrow \theta) \equiv (\varphi \rightarrow \psi) \rightarrow \theta$, lo cual no es cierto. Por ejemplo, si tomamos una valuación σ tal que $\sigma(\varphi) = 0$, en la primera fórmula el valor de verdad de θ depende del valor de verdad de ψ, mientras que en la segunda fórmula forzamos que sea 1, al hacer la primera parte de la implicación cierta. Otra forma de comprobarlo es haciendo la tabla de verdad para ambas fórmulas.

Notación

- Omitiremos paréntesis externos y otros que no produzcan ambigüedad.
  - Ejemplo: escribimos $p \wedge q \wedge r$.
- Usaremos los siguientes operadores:

$$\bigvee_{i=1}^{n} \varphi_i = \varphi_1 \vee \varphi_2 \vee \ldots \vee \varphi_n$$

$$\bigwedge_{i=1}^{n} \varphi_i = \varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$$

Conectivos y fórmulas

Las tablas de verdad también nos sirven para definir conectivos.

Ejemplo

El conectivo XOR se define con la siguiente tabla de verdad:

p	q	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Ejercicio

- ¿Cuántos conectivos binarios posibles tenemos?

Conectivos y fórmulas

Tenemos una fórmula $\varphi \in L(P)$, con $P = \{p, q, r\}$, y lo único que conocemos de ella es que cumple la siguiente tabla de verdad:

p	q	r	φ
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

- ¿Podemos construir una fórmula que sea lógicamente equivalente a φ?
- Dada una tabla con n variables proposicionales, ¿podemos generalizar el argumento anterior?

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

Conectivos y fórmulas

Podemos generalizar el argumento anterior. Sea φ una fórmula con n variables proposicionales $p_1, \ldots, p_n$. Tenemos 2^n posibles valuaciones $\sigma_1, \ldots, \sigma_{2^n}$. La tabla de verdad de φ se vería como:

	p_1	p_2	$\ldots$	p_{n-1}	p_n	φ
σ_1 :	0	0	$\ldots$	0	0	$\sigma_1(\varphi)$
σ_2 :	0	0	$\ldots$	0	1	$\sigma_2(\varphi)$
$\vdots$	$\vdots$	$\vdots$	$\ldots$	$\vdots$	$\vdots$	$\vdots$
σ_{2^n} :	1	1	$\ldots$	1	1	$\sigma_{2^n}(\varphi)$

Para cada σ_j con $j \in \{1, \ldots, 2^n\}$ consideremos la siguiente fórmula:

$$\varphi_j = \left(\bigwedge_{\substack{i=1\ldots n \\ \sigma_j(p_i)=1}} p_i \right) \wedge \left(\bigwedge_{\substack{i=1\ldots n \\ \sigma_j(p_i)=0}} (\neg p_i) \right)$$

Conectivos y fórmulas

$$\varphi_j = \left(\bigwedge_{\substack{i=1\ldots n \\ \sigma_j(p_i)=1}} p_i \right) \wedge \left(\bigwedge_{\substack{i=1\ldots n \\ \sigma_j(p_i)=0}} (\neg p_i) \right)$$

Notemos que φ_j representa a la fila j de la tabla de verdad de φ. Ahora hacemos la disyunción entre las valuaciones que hacen verdad a φ:

$$\bigvee_{\substack{j=1\ldots 2^n \\ \sigma_j(\varphi)=1}} \varphi_j = \bigvee_{\substack{j=1\ldots 2^n \\ \sigma_j(\varphi)=1}} \left(\left(\bigwedge_{\substack{i=1\ldots n \\ \sigma_j(p_i)=1}} p_i \right) \wedge \left(\bigwedge_{\substack{i=1\ldots n \\ \sigma_j(p_i)=0}} (\neg p_i) \right) \right)$$

Conectivos y fórmulas

Teorema

Podemos representar cualquier tabla de verdad con una fórmula.

Corolario

Podemos representar cualquier tabla de verdad con una fórmula que sólo usa $\neg$, $\wedge$ y $\vee$.

Conectivos funcionalmente completos

Definición

Un conjunto de conectivos lógicos se dice **funcionalmente completo** si toda fórmula en $L(P)$ es lógicamente equivalente a una fórmula que sólo usa esos conectivos.

- Ya demostramos que $\{\neg, \wedge, \vee\}$ es funcionalmente completo.

Ejercicios

1. Demuestre que $\{\neg, \wedge\}$, $\{\neg, \vee\}$ y $\{\neg, \rightarrow\}$ son funcionalmente completos.
2. Demuestre que $\{\neg\}$ no es funcionalmente completo.
3. ¿Es $\{\wedge, \vee, \rightarrow, \leftrightarrow\}$ funcionalmente completo?

Conectivos funcionalmente completos

Ejercicio

Demuestre que $\{\neg, \rightarrow\}$ es funcionalmente completo.

Demostración: Como sabemos que $C = \{\neg, \wedge, \vee\}$ es funcionalmente completo, demostraremos por inducción que toda fórmula construida usando sólo los conectivos anteriores es lógicamente equivalente a otra fórmula que solo usa $\neg$ y $\rightarrow$. Con esto, queda demostrado que $C' = \{\neg, \rightarrow\}$ es funcionalmente completo.

BI: Si $\varphi = p$, con $p \in P$, la propiedad se cumple trivialmente.

HI: Supongamos que $\varphi, \psi \in L(P)$, que sólo usan conectivos en C, son tales que $\varphi \equiv \varphi'$ y $\psi \equiv \psi'$, donde φ', ψ' sólo usan conectivos en C'.

TI: Consideraremos una fórmula θ construida con los pasos inductivos para los operadores en C:

$i)$ $\theta = (\neg\varphi) \stackrel{HI}{\equiv} (\neg\varphi')$, y como φ' sólo usa conectivos en C', θ es equivalente a una fórmula que sólo usa conectivos en C'.

Conectivos funcionalmente completos

Ejercicio

Demuestre que $\{\neg, \rightarrow\}$ es funcionalmente completo.

TI:

$ii)$ $\theta = \varphi \wedge \psi \overset{HI}{\equiv} \varphi' \wedge \psi'$

Usando las leyes de doble negación, De Morgan y de implicancia:

$$\theta \equiv \neg(\neg(\varphi' \wedge \psi')) \equiv \neg((\neg\varphi') \vee (\neg\psi')) \equiv \neg(\varphi' \rightarrow (\neg\psi'))$$

Y como φ', ψ' sólo usan conectivos en C', θ es equivalente a una fórmula que sólo usa conectivos en C'.

Ejercicio

Demuestre que $\{\neg, \rightarrow\}$ es funcionalmente completo.

TI:

$iii)$ $\theta = \varphi \vee \psi \stackrel{HI}{\equiv} \varphi' \vee \psi'$

Usando la ley de implicancia:

$\theta \equiv (\neg\varphi') \rightarrow \psi'$

Y como φ', ψ' sólo usan conectivos en C', θ es equivalente a una fórmula que sólo usa conectivos en C'. $\square$

Ejercicio

Demuestre que $\{\neg\}$ no es funcionalmente completo.

Demostración: Dado $P = \{p\}$, demostraremos por inducción que toda fórmula en $L(P)$ construida usando sólo p y $\neg$ es lógicamente equivalente a p o a $\neg p$. Como ninguna de estas fórmulas es equivalente a $p \wedge \neg p$, se concluye que $\{\neg\}$ no puede ser funcionalmente completo.

BI: Si $\varphi = p$, con $p \in P$, la propiedad se cumple trivialmente.

HI: Supongamos que $\varphi \in L(P)$ construida usando sólo p y $\neg$ es equivalente a p o a $\neg p$.

Conectivos funcionalmente completos

TI: El único caso inductivo que tenemos que demostrar es $\psi = \neg\varphi$, pues sólo podemos usar el conectivo $\neg$. Por HI, sabemos que para toda valuación σ se cumple que $\sigma(\varphi) = \sigma(p)$ o $\sigma(\varphi) = \sigma(\neg p)$. Podemos hacer una tabla de verdad:

φ	$\psi = \neg\varphi$
p	$\neg p$
$\neg p$	p

Concluimos que ψ es equivalente a p o a $\neg p$. Por lo tanto, por el argumento dado al principio, tenemos que $\{\neg\}$ no es funcionalmente completo. $\square$

Notación

- Omitiremos paréntesis externos y otros que no produzcan ambigüedad.
  - Ejemplo: escribimos $p \wedge q \wedge r$.
- Usaremos los siguientes operadores:

$$\bigvee_{i=1}^{n} \varphi_i = \varphi_1 \vee \varphi_2 \vee \ldots \vee \varphi_n$$

$$\bigwedge_{i=1}^{n} \varphi_i = \varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$$

- De ahora en adelante, el conectivo $\neg$ tendrá **precedencia** sobre los conectivos binarios.
  - Podemos eliminar paréntesis cuando hay términos negados.
  - Ejemplo: escribimos $((\neg p) \vee q) \wedge (\neg r)$ como $(\neg p \vee q) \wedge \neg r$.

Satisfacibilidad

Definición

Una fórmula φ es **satisfacible** si existe una valuación σ tal que $\sigma(\varphi) = 1$.

Ejemplo

Las siguientes fórmulas son satisfacibles:

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

Las siguientes fórmulas no son satisfacibles:

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

Satisfacibilidad

Dada una fórmula φ, queremos verificar si es satisfacible.

- Este es un problema fundamental tanto en computación como en ingeniería.
- Muchos problemas pueden ser resueltos usando este problema, lo que nos muestra el **poder expresivo** de la lógica proposicional.
  - Problemas en grafos (coloración, clique).
  - Problemas de optimización (vendedor viajero, mochila, programación entera).
- ¿Es difícil este problema? ¿Cómo lo resolvemos?

Una aplicación muy importante es que podemos **modelar** otros problemas mediante satisfacción lógica.

- Veamos un par de ejemplos.

Ejercicio

Sea M un mapa conformado por n países. Decimos que M es 3-coloreable si se pueden pintar todos los países con 3 colores sin que ningún par de países adyacente tenga el mismo color. En otras palabras, los países vecinos deben tener colores distintos.

Dado un mapa M, construya una fórmula $\varphi \in L(P)$ tal que M es 3-coloreable si y sólo si φ es satisfacible.

M consiste en una lista de países $\{1, 2, \ldots, n\}$, y una lista de pares de países adyacentes $\{(i, j), (k, m), \ldots\}$.

Primero, debemos definir las variables proposicionales que usaremos:

Para cada $(i, j) \in \{1, \ldots, n\}^2$ tendremos una variable proposicional p_{ij} que indica si el país i es adyacente con el país j. Por ejemplo, en el caso del mapa de Sudamérica, si Chile estuviese numerado por el 1 y Argentina por el 2, tendríamos que $p_{12} = 1$.

Además, para $i \in \{1, \ldots, n\}$ tendremos variables r_i, b_i y g_i, que representan si el país i está pintado de rojo, azul o verde respectivamente. Por ejemplo, si en el mapa ya coloreado Chile tiene el color rojo, tendremos que $\sigma(r_1) = 1$ (bajo el supuesto de que Chile está representado por el número 1).

Satisfacibilidad

Segundo, establecemos las restricciones usando fórmulas:

Para representar el problema vamos a definir φ como la conjunción de las siguientes fórmulas.

Necesitamos una fórmula que diga que a cada país se le asigna un único color:

$$\varphi_C = \bigwedge_{i=1}^{n} \Bigg(\Big(r_i \vee b_i \vee g_i \Big) \wedge \Big(r_i \to (\neg b_i \wedge \neg g_i) \Big) \wedge \Big(b_i \to (\neg r_i \wedge \neg g_i) \Big) \wedge \Big(g_i \to (\neg r_i \wedge \neg b_i) \Big) \Bigg)$$

Necesitamos una fórmula que diga que dos países adyacentes deben tener colores distintos:

$$\varphi_D = \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{n} \Big(p_{ij} \to \big((r_i \to \neg r_j) \wedge (b_i \to \neg b_j) \wedge (g_i \to \neg g_j)\big)\Big)$$

Finalmente incluimos una fórmula que inicialice el mapa M, es decir, que indique qué países son adyacentes y qué países no:

$$\varphi_M = \bigwedge_{\substack{i \text{ y } j \\ \text{son adyacentes en } M}} p_{ij} \quad \wedge \bigwedge_{\substack{i \text{ y } j \\ \text{no son adyacentes en } M}} \neg p_{ij}$$

Entonces, nuestra fórmula será la conjunción de las fórmulas anteriores:

$$\varphi = \varphi_C \wedge \varphi_D \wedge \varphi_M$$

Ahora demostraremos que M es 3-coloreable si y sólo si φ es satisfacible.

Satisfacibilidad

($\Rightarrow$) PD: si M es 3-coloreable, entonces φ es satisfacible.

Supongamos que M es 3-coloreable. Luego, existe una asignación de un único color entre r, g y b para cada país, tal que países adyacentes no están pintados del mismo color. Construimos una valuación σ para las variables proposicionales que definimos en el primer paso como:

$$\sigma(p_{ij}) = \begin{cases} 1 & \text{si } i, j \text{ son adyacentes en } M \\ 0 & \text{si } i, j \text{ no son adyacentes en } M \end{cases}$$

$$\sigma(r_i) = \begin{cases} 1 & \text{si el país } i \text{ es rojo en la 3-coloración de } M \\ 0 & \text{en otro caso} \end{cases}$$

$$\sigma(g_i) = \begin{cases} 1 & \text{si el país } i \text{ es verde en la 3-coloración de } M \\ 0 & \text{en otro caso} \end{cases}$$

$$\sigma(b_i) = \begin{cases} 1 & \text{si el país } i \text{ es azul en la 3-coloración de } M \\ 0 & \text{en otro caso} \end{cases}$$

Satisfacibilidad

($\Rightarrow$) (continuación)

Ahora simplemente hay que verificar que $\sigma(\varphi) = 1$:

- $\sigma(\varphi_C)$: para cada país i, se debe cumplir que $\sigma(r_i) = 1$, o que $\sigma(g_i) = 1$, o que $\sigma(b_i) = 1$, y solo una de estas, por cómo construimos σ. Luego, es claro que $\sigma(\varphi_C) = 1$.
- $\sigma(\varphi_D)$: para cada combinación de países i, j, sabemos que $\sigma(p_{ij}) = 1$ solo cuando los países son adyacentes. Entonces, como en φ_D tenemos una implicancia, solo nos preocuparemos de los pares de países adyacentes. En la parte de la derecha de la implicancia, sabemos que si por ejemplo $\sigma(r_i) = 1$, se debe cumplir que $\sigma(r_j) = 0$, dado que construimos σ a partir de una 3-coloración. Para las otras dos implicancias, sabemos que el lado izquierdo va a ser falso (por la fórmula φ_C), y por lo tanto todo el lado derecho se hace verdadero, y entonces $\sigma(\varphi_D) = 1$. El análisis para cuando i es de otro color es análogo.

Satisfacibilidad

($\Rightarrow$) (continuación)

- $\sigma(\varphi_M)$: por construcción de σ es claro que $\sigma(\varphi_M) = 1$, dado que la construimos precisamente como esta fórmula, asignando 1 a pares de países adyacentes y 0 a los que no.

Finalmente, como φ es la conjunción de las fórmulas anteriores, concluimos que $\sigma(\varphi) = 1$, y entonces φ es satisfacible.

($\Leftarrow$) <u>PD:</u> si φ es satisfacible, entonces M es 3-coloreable.
Supongamos que φ es satisfacible. Luego, existe una valuación σ tal que $\sigma(\varphi) = 1$, y por construcción $\sigma(\varphi_C) = \sigma(\varphi_D) = \sigma(\varphi_M) = 1$. Usaremos esta valuación para colorear el mapa. En primer lugar, como $\sigma(\varphi_C) = 1$, sabemos que $\forall i, \sigma(r_i \vee g_i \vee b_i) = 1$, y por lo tanto cada país tiene asignado al menos un color. Sin pérdida de generalidad, supongamos que $\sigma(r_k) = 1$, es decir, pintamos el país k rojo. Como también se cumple que $\sigma(r_k \rightarrow (\neg g_k \wedge \neg b_k)) = 1$, necesariamente $\sigma(g_k) = 0$ y $\sigma(b_k) = 0$, y por lo tanto cada país tiene un único color.

$(\Leftarrow)$ (continuación)

En segundo lugar, como $\sigma(\varphi_M) = 1$, sabemos que si i, j son adyacentes en M, $\sigma(p_{ij}) = 1$, y si no lo son, $\sigma(p_{ij}) = 0$. Ahora, en $\sigma(\varphi_D) = 1$, solo nos interesa el primer caso (dado que en el segundo no podemos concluir nada de la implicancia). Tomemos entonces i, j adyacentes, y sin pérdida de generalidad supongamos que $\sigma(r_i) = 1$. Como $\sigma(r_i \rightarrow \neg r_j) = 1\ \forall j$ adyacente a i, necesariamente $\sigma(r_j) = 0$, y entonces los países adyacentes no pueden estar pintados del mismo color.

Concluimos que usando los colores asignados por φ a través de r_i, g_i, b_i, podemos 3-colorear M.

Ejercicio

Un **cuadrado mágico** es una matriz de $n \times n$ que sólo contiene números entre 1 y n en que la suma de cada fila y columna es la misma. Por ejemplo, la siguiente matriz de 3×3 es un cuadrado mágico:

$$\begin{bmatrix} 2 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 1 & 3 \end{bmatrix}$$

Dada una matriz C de $n \times n$ con algunas posiciones en 0, diremos que es **completable** si existe una manera de reemplazar los 0 por números entre 1 y n de manera que la matriz resultante sea un cuadrado mágico.

Satisfacibilidad

Ejercicio

Por ejemplo, la siguiente matriz es completable, pues pueden llenarse las casillas y obtener la matriz anterior:

$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix} \rightarrow \begin{bmatrix} 2 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 1 & 3 \end{bmatrix}$$

En cambio, la siguiente matriz no es completable:

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 3 & 0 & 0 \end{bmatrix}$$

Satisfacibilidad

Ejercicio

Dada una matriz C de 3×3, construya una fórmula $\varphi \in L(P)$ tal que C es completable si y sólo si φ es satisfacible.

Observación

Cada valuación σ que satisface a φ representa una forma de completar C.

Satisfacibilidad

Para cada $i, j, k \in \{1, 2, 3\}$, tenemos una variable proposicional $p_{i,j,k}$ que indica que el valor contenido en la posición (i, j) de la matriz C es k. Por ejemplo, en la última matriz tenemos que $p_{1,1,1}$ y $p_{3,1,3}$ son válidas.

Para representar el problema vamos a definir φ como la conjunción de las siguientes fórmulas.

Primero, necesitamos una fórmula que diga que a cada posición de la matriz le asignamos un único valor entre 1 y 3:

$$\bigwedge_{i=1}^{3} \bigwedge_{j=1}^{3} \Bigg(\Big(p_{i,j,1} \vee p_{i,j,2} \vee p_{i,j,3} \Big) \wedge \Big(p_{i,j,1} \rightarrow \neg \left(p_{i,j,2} \wedge \neg p_{i,j,3} \right) \Big) \wedge$$

$$\Big(p_{i,j,2} \rightarrow \left(\neg p_{i,j,1} \wedge \neg p_{i,j,3} \right) \Big) \wedge \Big(p_{i,j,3} \rightarrow \left(\neg p_{i,j,1} \wedge \neg p_{i,j,2} \right) \Big) \Bigg)$$

Segundo, necesitamos fórmulas que digan que si la suma de la primera fila es l, donde l es un número entre 3 y 9, entonces la suma de las restantes filas y columnas también es l. Para hacer esto introducimos notación adicional. Decimos que $suma(l)$ son todos los triples (l_1, l_2, l_3) tal que $l_1 + l_2 + l_3 = l$ y $l_1, l_2, l_3 \in \{1, 2, 3\}$. Por ejemplo, $suma(4) = \{(1, 1, 2), (1, 2, 1), (2, 1, 1)\}$. Entonces, para cada l entre 3 y 9, incluimos una fórmula que indica que si la suma de la primera fila es l, entonces la suma de la segunda fila también es l:

$$\left(\bigvee_{(l_1,l_2,l_3) \in suma(l)} (p_{1,1,l_1} \wedge p_{1,2,l_2} \wedge p_{1,3,l_3}) \right) \rightarrow$$

$$\left(\bigvee_{(m_1,m_2,m_3) \in suma(l)} (p_{2,1,m_1} \wedge p_{2,2,m_2} \wedge p_{2,3,m_3}) \right)$$

Satisfacibilidad

También incluimos el mismo de tipo de fórmulas para la tercera fila y las tres columnas. Por ejemplo, para la tercera columna incluimos la fórmula:

$$\left(\bigvee_{(l_1,l_2,l_3)\in suma(l)} (p_{1,1,l_1} \wedge p_{1,2,l_2} \wedge p_{1,3,l_3}) \right) \rightarrow \left(\bigvee_{(m_1,m_2,m_3)\in suma(l)} (p_{1,3,m_1} \wedge p_{2,3,m_2} \wedge p_{3,3,m_3}) \right)$$

Finalmente, incluimos un conjunto de fórmulas que indican que valores son dados. Por ejemplo, para la última matriz de arriba, incluimos fórmulas: $p_{1,1,1}$, $p_{1,2,1}$, $p_{1,3,1}$ y $p_{3,1,3}$.

No es difícil demostrar (hágalo como ejercicio) que si φ es definida como la conjunción de todas las fórmulas anteriores, entonces C es completable si y sólo si φ es satisfacible. En particular, cada valuación σ que satisface a φ representa una forma de completar C.

Tautologías y contradicciones

Definición

Una fórmula φ es una **contradicción** si no es satisfacible; es decir, para toda valuación σ se tiene que $\sigma(\varphi) = 0$.

Ejemplo

$p \wedge \neg p$

Definición

Una fórmula φ es una **tautología** si para toda valuación σ se tiene que $\sigma(\varphi) = 1$.

Ejemplo

$p \vee \neg p$

$p \leftrightarrow p$

Tautologías y contradicciones

Definición

Una fórmula φ es una **tautología** si para toda valuación σ se tiene que $\sigma(\varphi) = 1$.

Podemos definir la equivalencia lógica de una manera alternativa:

Teorema

Dos fórmulas $\varphi, \psi \in L(P)$ son lógicamente equivalentes si $\varphi \leftrightarrow \psi$ es una tautología.

Formas normales

Definición

Un literal es una variable proposicional o la negación de una variable proposicional. Por ejemplo, p y $\neg r$ son literales.

Formas normales

Definición

Decimos que una fórmula φ está en **forma normal disyuntiva (DNF)** si es una disyunción de conjunciones de literales; o sea, si es de la forma

$$B_1 \vee B_2 \vee \ldots \vee B_k$$

donde cada B_i es una conjunción de literales, $B_i = (l_{i1} \wedge \ldots \wedge l_{ik_i})$

Ejemplo

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

Teorema

Toda fórmula es equivalente a una fórmula en DNF.

Ya demostramos este teorema, ¿cierto?

Formas normales

Definición

Decimos que una fórmula φ está en **forma normal conjuntiva (CNF)** si es una conjunción de disyunciones de literales; o sea, si es de la forma

$$C_1 \wedge C_2 \wedge \ldots \wedge C_k$$

donde cada C_i es una disyunción de literales, $C_i = (l_{i1} \vee \ldots \vee l_{ik_i})$

- Una disyunción de literales se llama una **cláusula**.
  - Los C_i anteriores son cláusulas.
- Una fórmula en CNF es una conjunción de cláusulas.

Ejemplo

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

Formas normales

Teorema

Toda fórmula es equivalente a una fórmula en CNF.

Ejercicio

Demuestre el teorema.

Formas normales

Teorema

Toda fórmula es equivalente a una fórmula en CNF.

Demostración: Como sabemos que toda fórmula es equivalente a una fórmula en DNF, demostraremos que toda fórmula en DNF es equivalente a una fórmula en CNF por inducción en la cantidad de disyunciones de la primera fórmula.

Dado un conjunto de variables proposicionales P, tenemos la siguiente propiedad sobre los naturales:

$Prop(n)$: toda fórmula $\varphi \in L(P)$ en DNF con a lo más n disyunciones es lógicamente equivalente a una fórmula ψ en CNF ($\varphi \equiv \psi$).

Demostraremos esta propiedad por inducción.

Propiedad

$Prop(n)$: toda fórmula $\varphi \in L(P)$ en DNF con a lo más n disyunciones es lógicamente equivalente a una fórmula ψ en CNF ($\varphi \equiv \psi$).

BI: $Prop(0)$: una fórmula en DNF con 0 disyunciones es de la forma

$$l_1 \wedge l_2 \wedge \ldots \wedge l_k$$

por lo que está trivialmente en CNF.

HI: Suponemos que $Prop(n-1)$ es cierta; es decir, toda fórmula φ en DNF con a lo más $n-1$ disyunciones es lógicamente equivalente a una fórmula ψ en CNF.

TI: Debemos demostrar que toda fórmula φ' en DNF con n disyunciones es lógicamente equivalente a una fórmula ψ' en CNF. Cualquier φ' será de la forma

$$\varphi' = B_0 \vee B_1 \vee \ldots \vee B_{n-1} \vee B_n$$

donde los B_i son conjunciones de literales. Por asociatividad:

$$\varphi' \equiv (B_0 \vee B_1 \vee \ldots \vee B_{n-1}) \vee B_n$$

y por HI $\varphi' \overset{HI}{\equiv} \psi \vee B_n$, con ψ una fórmula en CNF. Expandiendo la conjunción B_n:

$$\varphi' \equiv \psi \vee (l_{n,1} \wedge \ldots \wedge l_{n,k_n})$$

Formas normales

TI: Por distributividad:

$$\varphi' \equiv (\psi \vee l_{n,1}) \wedge \ldots \wedge (\psi \vee l_{n,k_n})$$

Como ψ está en CNF, podemos expandirla:

$$\varphi' \equiv ((C_1 \wedge \ldots \wedge C_m) \vee l_{n,1}) \wedge \ldots \wedge ((C_1 \wedge \ldots \wedge C_m) \vee l_{n,k_n})$$

con C_i cláusulas. Por distributividad:

$$\varphi' \equiv ((C_1 \vee l_{n,1}) \wedge \ldots \wedge (C_m \vee l_{n,1})) \wedge \ldots \wedge ((C_1 \vee l_{n,k_n}) \wedge \ldots \wedge (C_m \vee l_{n,k_n}))$$

Y por asociatividad:

$$\varphi' \equiv (C_1 \vee l_{n,1}) \wedge \ldots \wedge (C_m \vee l_{n,1}) \wedge \ldots \wedge (C_1 \vee l_{n,k_n}) \wedge \ldots \wedge (C_m \vee l_{n,k_n})$$

TI: Como los C_i son cláusulas, es claro que $(C_i \vee l_{n,j})$ es una cláusula. Luego, tenemos que

$$\psi' = (C_1 \vee l_{n,1}) \wedge \ldots \wedge (C_m \vee l_{n,1}) \wedge \ldots \wedge (C_1 \vee l_{n,k_n}) \wedge \ldots \wedge (C_m \vee l_{n,k_n})$$

está en CNF y es tal que $\varphi' \equiv \psi'$. $\square$

Consecuencia lógica

Dado un conjunto de fórmulas Σ en $L(P)$, diremos que una valuación σ satisface Σ si para toda fórmula $\varphi \in \Sigma$ se tiene que $\sigma(\varphi) = 1$.

Notación: $\sigma(\Sigma) = 1$.

¿Cuándo decimos que una fórmula ψ se deduce desde Σ?

Definición

ψ es **consecuencia lógica** de Σ si para cada valuación σ tal que $\sigma(\Sigma) = 1$, se tiene que $\sigma(\psi) = 1$.

Notación: $\Sigma \models \psi$.

Ejemplos

- $\{p, p \to q\} \models q$ (Modus ponens)
- $\{\neg q, p \to q\} \models \neg p$ (Modus tollens)
- $\{p \vee q \vee r, p \to s, q \to s, r \to s\} \models s$ (Demostración por partes)

Consecuencia lógica

Definición

Un conjunto de fórmulas Σ es **satisfacible** si existe una valuación σ tal que $\sigma(\Sigma) = 1$. En caso contrario, Σ es **inconsistente**.

Teorema

$\Sigma \models \varphi$ si y sólo si $\Sigma \cup \{\neg\varphi\}$ es inconsistente.

Ejercicio

Demuestre el teorema.

Consecuencia lógica

Teorema

$\Sigma \models \varphi$ si y sólo si $\Sigma \cup \{\neg\varphi\}$ es inconsistente.

Demostración: ($\Rightarrow$) Dado que $\Sigma \models \varphi$, demostraremos que $\Sigma \cup \{\neg\varphi\}$ es inconsistente. Por contradicción, supongamos que $\Sigma \cup \{\neg\varphi\}$ es satisfacible, y luego existe una valuación σ tal que $\sigma(\Sigma \cup \{\neg\varphi\}) = 1$. Esto implica que $\sigma(\Sigma) = 1$ y que $\sigma(\neg\varphi) = 1$, y por lo tanto $\sigma(\Sigma) = 1$ y $\sigma(\varphi) = 0$, lo que contradice que $\Sigma \models \varphi$.

($\Leftarrow$) Dado que $\Sigma \cup \{\neg\varphi\}$ es inconsistente, demostraremos que $\Sigma \models \varphi$. Debemos demostrar que dada una valuación σ tal que $\sigma(\Sigma) = 1$, se tiene que $\sigma(\varphi) = 1$. Como $\Sigma \cup \{\neg\varphi\}$ es inconsistente y $\sigma(\Sigma) = 1$, necesariamente $\sigma(\neg\varphi) = 0$, y luego $\sigma(\varphi) = 1$. Hemos demostrado que si σ es tal que $\sigma(\Sigma) = 1$, entonces $\sigma(\varphi) = 1$, por lo que concluimos que $\Sigma \models \varphi$. $\square$

Matemáticas Discretas

Lógica proposicional

Nicolás Alvarado
nfalvarado@mat.uc.cl

Bernardo Barías
bjbarias@uc.cl

Sebastián Bugedo
bugedo@uc.cl

Gabriel Diéguez
gsdieguez@ing.puc.cl

Departamento de Ciencia de la Computación
Escuela de Ingeniería
Pontificia Universidad Católica de Chile

28 de agosto de 2023