Lógica Proposicional

Inteligencia Artificial e Ingeniería del Conocimiento

Constantino Antonio García Martínez

Universidad San Pablo Ceu

Contenidos

Introducción a Agentes Basados en Conocimiento

Introducción a Agentes Basados en Conocimiento

Historia

Lógica en IA: Perspectiva Histórica

La Era de la Lógica (pre-1990s)
La lógica fue el **paradigma dominante** en inteligencia artificial

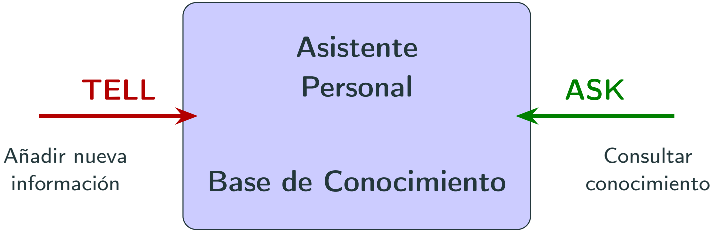
Limitaciones que surgieron:

Fortalezas perdurables:

Introducción a Agentes Basados en Conocimiento

Motivación

Motivando la Lógica: Un Asistente Personal



Este es un ejemplo de un **Sistema Basado en Conocimiento**

Ejemplo de Interacción: Razonamiento Clásico

`> Todos los hombres son mortales.`

`< ¡Eso es interesante!`

`> Sócrates es un hombre.`

`< ¡Eso es interesante!`

`> ¿Es Sócrates mortal?`

$<$ Sí

`> ¿Es Platón mortal?`

$<$ No lo sé.

Aplicaciones Reales de la Lógica Hoy

Verificación de Software y Hardware

- Validación de diseño de hardware (Intel, AMD)
- Verificación formal de software crítico

SAT/SMT Solvers

- Satisfacción de restricciones
- Problemas de planificación
- Asignación de recursos

Sistemas Expertos

- Diagnóstico médico (MYCIN, sistemas actuales)
- Asesoramiento financiero y legal

IA Simbólica

- Integración neuro-simbólica: AlphaGeometry
- Razonamiento sobre conocimiento estructurado
- IA explicable (XAI)

Web Semántica y Grafos

- Grafos de conocimiento
- Ontologías y razonamiento (OWL, RDF)

Demostración de Teoremas

- Verificación matemática formal
- *Herramientas:* Lean, Coq, Isabelle

Fundamentos de Lógica

Ingredientes Clave de Cualquier Lógica

Toda lógica consta de tres componentes fundamentales:

1. Sintaxis: ¿Qué podemos decir?
Define el conjunto de **fórmulas válidas** que se pueden expresar

- Especifica la estructura gramatical
- Reglas para construir fórmulas bien formadas

Ejemplo: $\mathit{Lluvia} \land \mathit{Mojado}$ (conjunción de dos proposiciones)

2. Semántica: ¿Qué significa?
Para cada fórmula, especifica el conjunto de **modelos** (configuraciones del mundo) donde es verdadera **Ejemplo:** $\mathit{Lluvia} \land \mathit{Mojado}$ es verdadera en mundos donde tanto $\mathit{Lluvia} = \mathit{verdadero}$ como $\mathit{Mojado} = \mathit{verdadero}$

Ingredientes Clave de Cualquier Lógica (continuación)

3. Reglas de Inferencia: ¿Qué se deduce?
Dada una fórmula f, ¿qué nuevas fórmulas g se pueden derivar que sean ciertas?

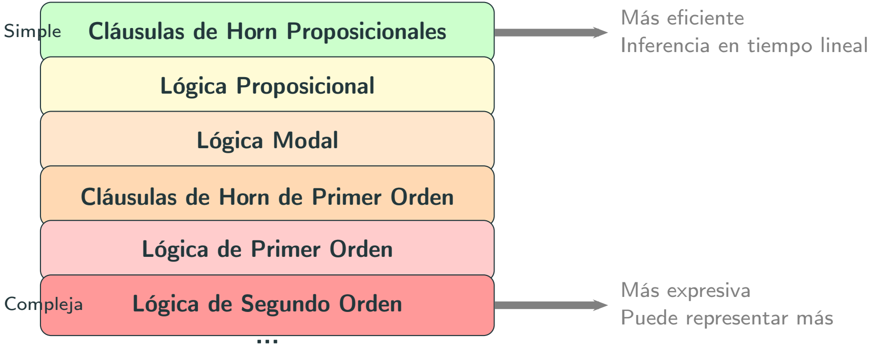
- Notación: $\frac{f}{g}$ significa "de f, derivar g"
- La inferencia debe ser sólida (preservar la verdad)

Ejemplo: $\frac{Lluvia \wedge Mojado}{Lluvia}$ (regla de Eliminación-Y)

Sintaxis + Semántica = Modelado

- **Modelar un problema:** Elegir sintaxis y semántica para representar el conocimiento del dominio
- **Resolver un problema:** Aplicar reglas de inferencia para derivar nuevo conocimiento

El Panorama de las Lógicas



Compromiso Fundamental
Expresividad vs. Eficiencia Computacional

- Las lógicas más expresivas pueden representar conocimiento más complejo
- Pero la inferencia se vuelve más costosa computacionalmente (o indecidible)

Lógica Proposicional (LP)

Lógica Proposicional: Sintaxis

Gramática de la lógica proposicional:

$$
\begin{array}{rcl}
Sentence & \rightarrow & AtomicSentence \mid ComplexSentence \\
AtomicSentence & \rightarrow & True \mid False \mid P \mid Q \mid R \mid \ldots \\
ComplexSentence & \rightarrow & (\, Sentence \,) \\
& \mid & \neg\, Sentence \\
& \mid & Sentence \land Sentence \\
& \mid & Sentence \lor Sentence \\
& \mid & Sentence \;\Rightarrow\; Sentence \\
& \mid & Sentence \;\Leftrightarrow\; Sentence \\
\\
\textsc{Operator Precedence} & : & \neg, \land, \lor, \Rightarrow, \Leftrightarrow
\end{array}
$$

¶

Nótese que las fórmulas atómicas consisten en un único **símbolo proposicional**

La semántica define la verdad de cada fórmula con respecto a cada "**mundo posible**". Usamos **modelo** en lugar de "mundo posible"

Modelo
Un **modelo** w en lógica proposicional es una asignación de valores de verdad a símbolos proposicionales.

Example: 2 símbolos proposicionales: Lluvia y Mojado...
... 4 modelos posibles

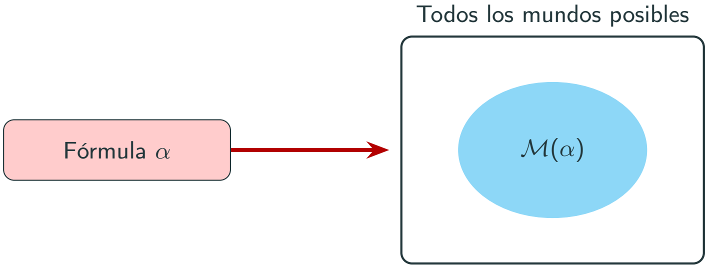
Lógica Proposicional: Semántica

La semántica para la lógica proposicional debe especificar cómo calcular el valor de verdad de cualquier fórmula, dado un modelo. Las **tablas de verdad** pueden representar estas reglas:

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

¶

- Si una fórmula α es verdadera en el modelo *m*, decimos que *m* **satisface** α
- Notación: $\mathcal{M}(\alpha)$: todos los modelos que satisfacen α.



Ejemplo: Modelos de una Fórmula Simple

Example: Modelos de una Fórmula

Fórmula: $\alpha = \textit{Lluvia} \vee \textit{Mojado}$

Pregunta: ¿Qué configuraciones del mundo satisfacen esta fórmula?

Idea Clave: Representación Compacta

Una **fórmula** representa compactamente un conjunto de **modelos**.

- Fórmula: $\textit{Lluvia} \vee \textit{Mojado}$ (muy concisa)
- Modelos: 3 de 4 configuraciones de mundo posibles

Base de Conocimiento: Intersección de Restricciones

Definición: Base de Conocimiento (KB)
Una **base de conocimiento** KB es un conjunto de fórmulas que representan su conjunción/intersección:

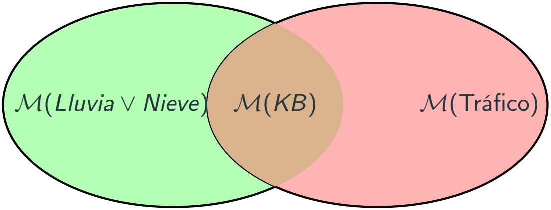
$$\mathcal{M}(KB) = \bigcap_{\alpha \in KB} \mathcal{M}(\alpha)$$

Intuición: KB especifica restricciones sobre el mundo. $\mathcal{M}(KB)$ es el conjunto de todos los mundos que satisfacen esas restricciones.

Lo que KB Representa para el Usuario
La base de conocimiento captura las **creencias del usuario sobre el mundo**:

- Cada fórmula representa un hecho o regla que el usuario conoce
- $\mathcal{M}(KB)$ = los mundos posibles consistentes con el conocimiento del usuario
- A medida que KB crece, $\mathcal{M}(KB)$ se reduce (más restricciones $\rightarrow$ menos posibilidades)
- $\mathcal{M}(KB)$ pequeño = el usuario sabe mucho (mundo bien determinado)
- $\mathcal{M}(KB)$ grande = el usuario sabe poco (quedan muchas posibilidades)

Ejemplo: Sea $KB = \{Lluvia \lor Nieve,\ \text{Tráfico}\}$



Conocimiento Creciente: Añadiendo Fórmulas

¿Qué sucede cuando añadimos más conocimiento?

$$\text{KB} \xrightarrow{\text{añadir } \alpha} KB \cup \{\alpha\}$$

Efecto en los modelos: Añadir fórmulas **reduce** el conjunto de modelos

$$\mathcal{M}(KB) \quad \longrightarrow \quad \mathcal{M}(KB) \cap \mathcal{M}(\alpha)$$

Pregunta Crítica

¿Cuánto se reduce $\mathcal{M}(KB)$? ¡Esta pregunta es fundamental para el razonamiento! Determina qué podemos inferir. Distinguiremos tres casos, y definiremos un concepto clave para entenderlos: **entailment**

Caso 1) Entailment: Información Ya Conocida

Intuición: α no añadió información/restricciones (ya era conocida).

Definición: Entailment
KB **implica lógicamente (entails)** α (escrito $KB \models \alpha$) si y solo si

$$\mathcal{M}(KB) \subseteq \mathcal{M}(\alpha)$$

(También se puede leer como α es **consecuencia lógica** de KB)

Equivalentemente: En todo mundo donde KB es verdadera, α también es verdadera.

Caso 2) Contradicción: Información Inconsistente

Intuición: α contradice lo que sabemos (capturado en KB).

Definición: Contradicción
KB **contradice** α si y solo si $\mathcal{M}(KB) \cap \mathcal{M}(\alpha) = \emptyset$

Ejemplo: $Lluvia \wedge Nieve$ contradice $\neg Nieve$

Observación Importante
KB contradice f si y solo si KB $\models \neg\alpha$ (más sobre esto en la siguiente diapositiva)

La Conexión Entre Contradicción y Entailment

Dos visiones del mismo fenómeno:

Visión de contradicción:

Visión de entailment:

Proposición: Contradicción y Entailment

$$\text{KB } \textbf{contradice } \alpha \quad \iff \quad \text{KB} \models \neg\alpha$$

Por qué esto importa: Muchos algoritmos de inferencia funcionan mediante *demostración por contradicción*

- $KB \models \alpha \iff$ KB $\wedge \neg\alpha$ es **insatisfacible**
- Esto se llama *reductio ad absurdum* o *refutación*

Caso 3) Contingencia: Nueva Información

Intuición: α añade información no trivial a KB (solapamiento parcial)

$$\emptyset \subsetneq \mathcal{M}(KB) \cap \mathcal{M}(\alpha) \subsetneq \mathcal{M}(KB)$$

Características:

- No está implicada: $KB \not\models \alpha$ (KB no garantiza α)
- No está contradicha: $KB \not\models \neg\alpha$ (KB no descarta α)
- Añadir α proporciona nueva información (reduce el conjunto de modelos)

Ejemplo: Dado $KB = \{Lluvia \vee Nieve\}$, la fórmula *Lluvia* es contingente

Operación TELL: Construyendo Conocimiento con los Tres Casos

$$\text{Tell}[\alpha] \longrightarrow \boxed{\text{KB}} \longrightarrow ?$$

Tell: *"Está lloviendo."* $\rightarrow$ Tell[Lluvia]

Tres posibles respuestas:

1. **Ya lo sabía:** entailment ($KB \models \alpha$)
2. **No creo que sea cierto** contradicción ($KB \models \neg\alpha$)
3. **¡Eso es interesante! (actualizar KB):** contingente

Operación ASK: Consultando Conocimiento con los Tres Casos

$$\text{Ask}[f] \longrightarrow \boxed{\text{KB}} \longrightarrow \ ?$$

Ask: *"¿Está lloviendo?"* $\rightarrow$ Ask[Lluvia]

Tres posibles respuestas:

1. **Sí:** entailment ($KB \models \alpha$)
2. **No:** contradicción ($KB \models \neg\alpha$)
3. **No lo sé:** contingente (ni implicada ni contradicha)

El Problema Central de la Inferencia

Dados KB y consulta α, determinar: ¿$KB \models \alpha$? ¿$KB \models \neg\alpha$? ¿o ninguno?

¡Esto es lo que resuelven los algoritmos de inferencia!

Por Qué el Entailment es Central en IA

El entailment es el fundamento del razonamiento lógico porque:

1. **Define la corrección:** ¿Qué se deduce de lo que sabemos?
2. **Permite responder consultas:** La operación ASK se basa en entailment
3. **Guía la adquisición de conocimiento:** La operación TELL usa entailment
4. **Conecta sintaxis y semántica:** Puente entre fórmulas y significado

El resto de este capítulo: algoritmos eficientes para verificar entailment

Ejercicio: Modelando un Sistema de Casa Inteligente

Exercise: Construyendo una KB

Escenario: Sistema de automatización del hogar

Reglas del sistema:

1. La calefacción se enciende si hace frío y hay alguien en casa
2. Las luces solo pueden encenderse si hay movimiento o es de noche (aunque pueden permanecer apagadas)
3. El sistema de seguridad se activa si no hay nadie en casa y es de noche
4. La calefacción no puede estar encendida si las ventanas están abiertas

Tareas:

a) Define las variables proposicionales

b) Traduce cada regla a lógica proposicional

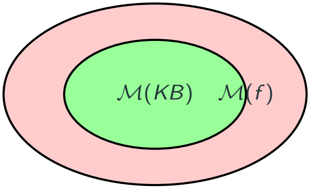
c) ¿Se pueden encender las luces a mediodía sin movimiento?

d) Si hace frío, hay alguien en casa pero las ventanas están abiertas, ¿se enciende la calefacción?

Inferencia

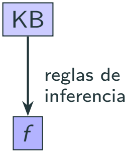
"El entailment es como la aguja que está en el pajar;
la inferencia es como encontrarla."

Entailment: Relación semántica



$KB \models \alpha$: verdad en modelos

Inferencia: Proceso algorítmico (usualmente basado en sintaxis)



$KB \vdash_i \alpha$: i deriva α de KB

Propiedades deseables del algoritmo de inferencia i:

- **Sólido (sound; preserva la verdad)**
- **Completo**

Ideal: $KB \vdash \alpha \Longleftrightarrow KB \models \alpha$

Inferencia

Model Checking

Un algoritmo naive: Model Checking

Example: Model Checking
KB: $\{Lluvia \rightarrow Mojado, Lluvia\}$ **Consulta:** ¿$Mojado$?

Método: Verificar todos los modelos posibles (asignaciones de valores de verdad):

De la Semántica a la Sintaxis: Reglas de Inferencia

Dos enfoques para el razonamiento:

Model Checking

- Trabaja con semántica (modelos)
- Enumera asignaciones de verdad
- Verifica si la consulta es verdadera en todos los modelos
- Siempre sólido y completo
- Pero: exponencial en nº de variables

Reglas de Inferencia

- Trabaja con sintaxis (fórmulas)
- Aplica reglas de transformación
- Deriva nuevas fórmulas de las antiguas
- Puede ignorar hechos irrelevantes
- La eficiencia depende de las reglas

Idea Clave: Reglas de Inferencia

Las reglas operan directamente sobre la **sintaxis**, no sobre la semántica.

Pero deben diseñarse para preservar la verdad (solidez).

Inferencia

Modus Ponens

Modus Ponens: La Regla Fundamental

Ejemplo en lenguaje natural:

Si el cielo está nublado, entonces lloverá.

El cielo está nublado.

Por lo tanto, lloverá.

Definición: Modus Ponens
Para cualesquiera símbolos proposicionales p y q:

$$\frac{p, \quad p \to q}{q}$$

Se lee: "Si tenemos p y tenemos $p \to q$, entonces podemos derivar q"

Ejemplo formal:

$$\frac{Nublado, \quad Nublado \to Lluvia}{Lluvia}$$

Definición: Regla de Inferencia
Si $f_1, \ldots, f_k, g$ son fórmulas, entonces lo siguiente es una regla de inferencia:

$$\frac{f_1, \quad \ldots \quad, f_k}{g}$$

Las premisas $f_1, \ldots, f_k$ *nos permiten concluir* g

Otras reglas de inferencia comunes:

- **Introducción-Y:** $\dfrac{p, \quad q}{p \wedge q}$
- **Eliminación-Y:** $\dfrac{p \wedge q}{p}$ (también se puede derivar q)
- **Introducción-O:** $\dfrac{p}{p \vee q}$ (para cualquier q)

Equivalencias Lógicas como Reglas de Inferencia

$$
\begin{array}{rcll}
(\alpha \wedge \beta) & \equiv & (\beta \wedge \alpha) & \text{commutativity of } \wedge \\
(\alpha \vee \beta) & \equiv & (\beta \vee \alpha) & \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) & \equiv & (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) & \equiv & (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\
\neg(\neg \alpha) & \equiv & \alpha & \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg \beta \Rightarrow \neg \alpha) & \text{contraposition} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg \alpha \vee \beta) & \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) & \equiv & ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) & \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) & \equiv & (\neg \alpha \vee \neg \beta) & \text{De Morgan} \\
\neg(\alpha \vee \beta) & \equiv & (\neg \alpha \wedge \neg \beta) & \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) & \equiv & ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) & \equiv & ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge
\end{array}
$$

[1]

Algoritmo: Forward-Inference
Entrada: Conjunto de reglas de inferencia

Repetir hasta que no haya cambios en KB:

1. Elegir fórmulas $f_1, \ldots, f_k \in KB$
2. Si existe regla coincidente $\frac{f_1, \ldots, f_k}{g}$:
3. Añadir g a KB

Definición: Derivación
KB **deriva/demuestra** f (escrito $KB \vdash f$) si y solo si f eventualmente se añade a KB

Características:

- Guiado por datos: comienza con hechos, deriva consecuencias
- Monotónico: KB solo crece
- Termina cuando no se pueden derivar nuevos hechos

Forward-Inference con Modus Ponens

Example: Forward-Inference solo con Modus Ponens

KB inicial:

$$KB = \{Nublado,\ Nublado \rightarrow Lluvia,\ Lluvia \rightarrow Inundacion\}$$

Ejercicio: ¿Es Modus Ponens Completo?

Exercise: Completitud de Modus Ponens
Dada la siguiente base de conocimiento:

$$\begin{aligned} KB = \{&A \vee B, \\ &A \rightarrow C, \\ &B \rightarrow C\} \end{aligned}$$

Tareas:

1. Aplicar modus ponens exhaustivamente. ¿Qué puedes derivar?
2. ¿Está C implicada por la KB? Es decir, ¿se cumple $KB \models C$?
3. ¿Puedes demostrar C usando solo modus ponens?
4. ¿Qué nos dice esto sobre la completitud de modus ponens?

Conclusión

Idea Clave
¡Modus ponens **no es completo** para la lógica proposicional!

Existen enunciados que son:

- Lógicamente implicados por una base de conocimiento (semánticamente verdaderos)
- Pero no pueden derivarse usando solo modus ponens (sintácticamente indemostrables)

Dos Soluciones:

- Restringir el conjunto de fórmulas permitidas: lógica proposicional $\rightarrow$ lógica proposicional con cláusulas de Horn
- Reglas de inferencia más potentes: Modus Ponens $\rightarrow$ resolución

Inferencia

Solución 1: Cláusulas de Horn, Forward y Backward-Chaining

Definición: Cláusula Definida (definite clauses)
Una **cláusula definida** es una disyunción de literales con **exactamente un literal positivo**:

$$\neg p_1 \vee \neg p_2 \vee \cdots \vee \neg p_k \vee q$$

donde $p_1, \ldots, p_k, q$ son símbolos proposicionales.

Vista alternativa (usando implicación):

$$\neg p_1 \vee \neg p_2 \vee \cdots \vee \neg p_k \vee q \quad \equiv \quad (p_1 \wedge p_2 \wedge \cdots \wedge p_k) \rightarrow q$$

Intuición: Si $p_1, \ldots, p_k$ se cumplen todos, entonces q se cumple

Cláusulas Definidas: Ejemplos

Example: Cláusulas Definidas

Example: Cláusulas No Definidas

Definición: Cláusula de Horn
Una **cláusula de Horn** es una disyunción de literales con **a lo sumo un literal positivo**:

$$\neg p_1 \vee \neg p_2 \vee \cdots \vee \neg p_k \vee q$$

donde q puede estar ausente (es decir, todos los literales son negativos).

Una cláusula de Horn es:

1. Una **cláusula definida**: exactamente un literal positivo
2. Una **cláusula objetivo**: cero literales positivos

Nombrada en honor a Alfred Horn (1951)

Cláusulas de Horn: Ejemplos

Example: Cláusulas objetivo (cero positivos):
¿Cómo indicar que no es posible un día soleado y lluvioso?

Intuición para cláusulas objetivo: Restricciones que prohíben ciertas combinaciones

Example: Cláusulas definidas (exactamente un positivo):
Si está nublado y húmedo, entonces llueve.

Por Qué Importan las Cláusulas Horn

Las bases de conocimiento con solo cláusulas Horn son interesantes por tres razones:

1. **Representación natural como implicaciones**
   - $(\neg Nublado \lor \neg Humedo \lor Lluvia) \equiv (Nublado \land Humedo) \rightarrow Lluvia$
   - Terminología: premisa = **cuerpo**, conclusión = **cabeza**
   - **Hecho**: literal positivo único (ej., *Nublado*)
2. **Algoritmos de inferencia eficientes**
   - Forward-chaining y backward-chaining son naturales y fáciles de seguir
   - Fundamento de la **programación lógica** (ej., Prolog)
3. **Verificación de entailment en tiempo lineal**
   - Decidir entailment: $O(n)$ en el tamaño de la KB
   - Hace que las cláusulas definidas sean prácticas para grandes bases de conocimiento

Forward-Chaining en Cláusulas de Horn

Forward chaining: Comenzar desde hechos conocidos para derivar nuevos

Exercise: Árboles AND-OR

KB inicial: Lunes , Lunes $\rightarrow$ Ocupado , Ocupado $\wedge$ SinTiempo $\rightarrow$ Estresado, Ocupado $\rightarrow$ SinTiempo, Estresado $\rightarrow$ Café

Derivar todos los hechos posibles usando forward chaining

Propiedades:

- Guiado por datos: comienza desde hechos, deriva consecuencias
- Cada regla se dispara a lo sumo una vez: tiempo $O(n)$
- Deriva todo lo implicado (completo)

Backward chaining: Comenzar desde la consulta, trabajar hacia atrás para encontrar hechos de soporte

Exercise: Árboles AND-OR

KB inicial: Lunes , Lunes $\rightarrow$ Ocupado , Ocupado $\wedge$ SinTiempo $\rightarrow$ Estresado, Ocupado $\rightarrow$ SinTiempo, Estresado $\rightarrow$ Café

Consulta: ¿Café?

Exercise: Árbol AND-OR

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

Forward vs. Backward Chaining

Forward Chaining

Características:

- Guiado por datos
- De abajo hacia arriba
- Deriva todos los hechos

Backward Chaining

Características:

- Dirigido por objetivo
- De arriba hacia abajo
- Demuestra consultas específicas

¡Ambos son sólidos y completos para cláusulas de Horn!

Inferencia

Solución 2: Resolución

Resolución: Inferencia Completa para Lógica Proposicional

Idea clave: "Cancelar" literales complementarios

Definición: Regla de Resolución

$$\frac{f_1 \vee \cdots \vee f_n \vee p, \quad \neg p \vee g_1 \vee \cdots \vee g_m}{f_1 \vee \cdots \vee f_n \vee g_1 \vee \cdots \vee g_m}$$

Si p aparece en una cláusula y $\neg p$ en otra, eliminar ambos

Ejemplo simple:

$$\frac{A \vee B, \quad \neg B \vee C}{A \vee C}$$

Intuición:

- O bien B es verdadero o B es falso
- Si B verdadero: $\neg B \vee C$ fuerza C verdadero
- Si B falso: $A \vee B$ fuerza A verdadero
- En cualquier caso: $A \vee C$ debe ser verdadero

Ejemplo de Resolución con Clima

Ejemplo: KB:

$$Soleado \lor Nublado, \quad \neg Nublado \lor Lluvia$$

Otro ejemplo: KB:

$$Calor \lor Humedo \lor Lluvia, \quad \neg Lluvia$$

Forma Normal Conjuntiva (CNF)

Problema: La resolución solo funciona en cláusulas (disyunciones)

Solución: ¡Convertir todas las fórmulas a CNF primero!

Definición: Forma Normal Conjuntiva (CNF)
Una fórmula CNF es una **conjunción de cláusulas**

Cláusula: Disyunción de literales

Literal: Símbolo proposicional (p) o su negación ($\neg p$)

Ejemplo: $(A \vee \neg B \vee C) \wedge (\neg A \vee D) \wedge (B \vee \neg D \vee \neg E)$

Proposición Clave
Toda fórmula en lógica proposicional se puede convertir en una fórmula CNF **equivalente**

$$\mathcal{M}(\alpha) = \mathcal{M}(\alpha')$$

donde α' es la versión CNF de α

Conversión CNF: Reglas Generales

Reglas de conversión (aplicar en orden):

1. **Eliminar bicondicional** ($\leftrightarrow$):
$$f \leftrightarrow g \quad \Rightarrow \quad (f \rightarrow g) \wedge (g \rightarrow f)$$
2. **Eliminar implicación** ($\rightarrow$):
$$f \rightarrow g \quad \Rightarrow \quad \neg f \vee g$$
3. **Mover negación hacia dentro** (De Morgan):
$$\neg(f \wedge g) \quad \Rightarrow \quad \neg f \vee \neg g$$
$$\neg(f \vee g) \quad \Rightarrow \quad \neg f \wedge \neg g$$
4. **Eliminar doble negación:**
$$\neg\neg f \quad \Rightarrow \quad f$$
5. **Distribuir $\vee$ sobre $\wedge$:**
$$f \vee (g \wedge h) \quad \Rightarrow \quad (f \vee g) \wedge (f \vee h)$$

Conversión CNF: Ejemplo

Example: Conversión CNF

Fórmula inicial: (Calor $\rightarrow$ Frío) $\rightarrow$ Extraño

Objetivo: Determinar si $KB \models f$ usando resolución

Algoritmo: Inferencia Basada en Resolución

1. Añadir $\neg f$ a KB (demostración por contradicción)
2. Convertir todas las fórmulas a CNF
3. Aplicar repetidamente la regla de resolución a pares de cláusulas
4. Si se deriva **falso** (cláusula vacía): devolver entailment
5. Si no se pueden derivar nuevas cláusulas: devolver no entailment

Idea clave: $KB \models f$ si y solo si $KB \cup \{\neg f\}$ es insatisfacible

Cláusula vacía: Notación: $\perp$ (o $\square$ o "falso")

Derivada cuando p y $\neg p$ se resuelven sin otros literales:

$$\frac{p, \quad \neg p}{\perp}$$

¡Derivar $\perp$ demuestra que el conjunto de cláusulas es insatisfacible!

Algoritmo de Resolución

Resolución
La resolución es tanto sólida como completa

Resolución: Ejemplo Completo

Example: Resolución

KB: $\{A \to (B \vee C),\ A,\ \neg B\}$

Consulta: ¿$KB \models C$?

Inferencia

Enfoques Modernos: SAT Solvers

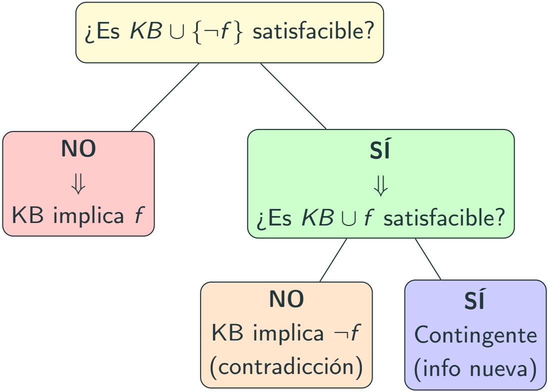
Definición: Satisfacibilidad
Una base de conocimiento KB es **satisfacible** si $\mathcal{M}(KB) \neq \emptyset$

En otras palabras: al menos una configuración del mundo hace verdaderas todas las fórmulas en KB

Por qué importa la satisfacibilidad:

- Verificar satisfacibilidad es el **problema computacional fundamental**
- Se pueden reducir tanto las operaciones TELL como ASK a satisfacibilidad
- Primer problema demostrado NP-completo (Cook, 1971)
- SAT solvers prácticos: millones de variables, impacto en el mundo real

Idea clave: Tanto TELL como ASK se pueden responder verificando satisfacibilidad



Example: Verificación de Consulta SAT
KB: $\{Soleado \vee Nublado,\ \neg Soleado \vee \text{Cálido}\}$

Consulta: ¿$KB \models$ Cálido?

Definición: SAT
Entrada: Base de conocimiento KB

Salida: ¿Existe un modelo que satisface? (¿$\mathcal{M}(KB) \neq \emptyset$?)

Algoritmos SAT populares:

1. **DPLL** (Davis-Putnam-Logemann-Loveland)
   - Búsqueda sistemática con backtracking con poda inteligente
   - Propagación unitaria, eliminación de literales puros
   - Fundamento de los SAT solvers modernos

2. **WalkSAT**
   - Búsqueda local aleatorizada
   - Voltea asignaciones de variables para satisfacer más cláusulas
   - Rápido pero incompleto (puede perder soluciones)

Resumen: Compromisos en la Inferencia

Algoritmo	Fórmulas	¿Completo?	Complejidad
Model checking	Cualquiera	Sí	$O(2^n)$
Modus ponens	LP general	No	—
Forward chaining	Cláusulas de Horn	Sí	$O(n)$
Backward chaining	Cláusulas de Horn	Sí	$O(n)$
Resolución	LP general (CNF)	Sí	$O(2^n)$ peor caso
DPLL	LP general (CNF)	Sí	$O(2^n)$ peor caso
SAT moderno (CDCL)	LP general (CNF)	Sí	Eficiente en práctica
WalkSAT	LP general (CNF)	No	Rápido (incompleto)

Resumen

- Cláusulas de Horn: punto óptimo (expresivo + eficiente, $O(n)$)
- Resolución/DPLL: completos pero exponenciales en el peor caso
- SAT solvers modernos (CDCL): peor caso exponencial, pero **notablemente eficientes en la práctica** en problemas del mundo real
- WalkSAT: sacrifica completitud por velocidad (bueno para instancias grandes satisfacibles)