

# Reglas y Sistemas Basados en Reglas

Juan Luis Castro

# Preguntas a responder del Tema

- ¿Qué es una regla? ¿y un Sistema Basado en Reglas?
- ¿Cuáles son los componentes de una regla?
- ¿Cómo se interpreta una regla?
- ¿Cuál es el mecanismo de inferencia asociado a un sistema basado en reglas?

# Hechos

Son afirmaciones que en un caso concreto serán verdad o mentira (lógica clásica)

Ejemplos:

- “el valor del oro es mayor que el de la plata” (verdadero en todos los casos)
- “el precio del oro es 5400” (concreto, verdadero para un caso concreto y falso para otros)
- “la plata no ha subido” (la negación de un hecho es otro hecho)
- “vende la plata” (consiste en realizar una acción)

# Reglas

## Formato:

**Si** antecedente1 y ..... y antecedenteN

**Entonces** consecuente1 y ... y consecuenteM

(donde antecedentes y consecuentes son hechos)

## Ejemplos:

Si “la plata ha subido mas de un 10%” y “(la plata) no está en periodo alcista”, Entonces “vender plata”

Si “el precio de la plata es mayor que su precio medio”  
Entonces “la plata esta cara”

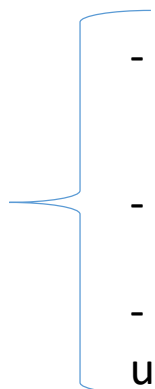
Si “comprar plata” y “la plata esta cara” entonces elimina  
“comprar plata”

# Usos de reglas en IA

Newel y Simon proponen las Reglas como modelo para describir comportamiento inteligente:

El comportamiento inteligente puede describirse mediante un conjunto de reglas que indiquen en cada momento cómo actuar en función de la información disponible

cómo actuar →

- 
- deducir nuevos “hechos”
  - eliminar “hechos”
  - Realizar acciones (obtener datos, interactuar con usuario, añadir tareas a realizar, etc...)

# Dedución con reglas (Modus Ponens o Razonamiento hacia adelante)

Dada una base de conocimiento formada por un conjunto de reglas:

1. Se mantiene una base de hechos que son ciertos en este momento
2. Si todos los hechos del antecedente de una regla están o casan con los hechos de la base de hechos, se realiza todo lo indicado en el consecuente: añadir hechos, eliminar hechos, realizar acciones,...
3. Se reitera hasta que no haya ninguna regla que se pueda disparar

# Sistemas de reglas vs lógica clásica

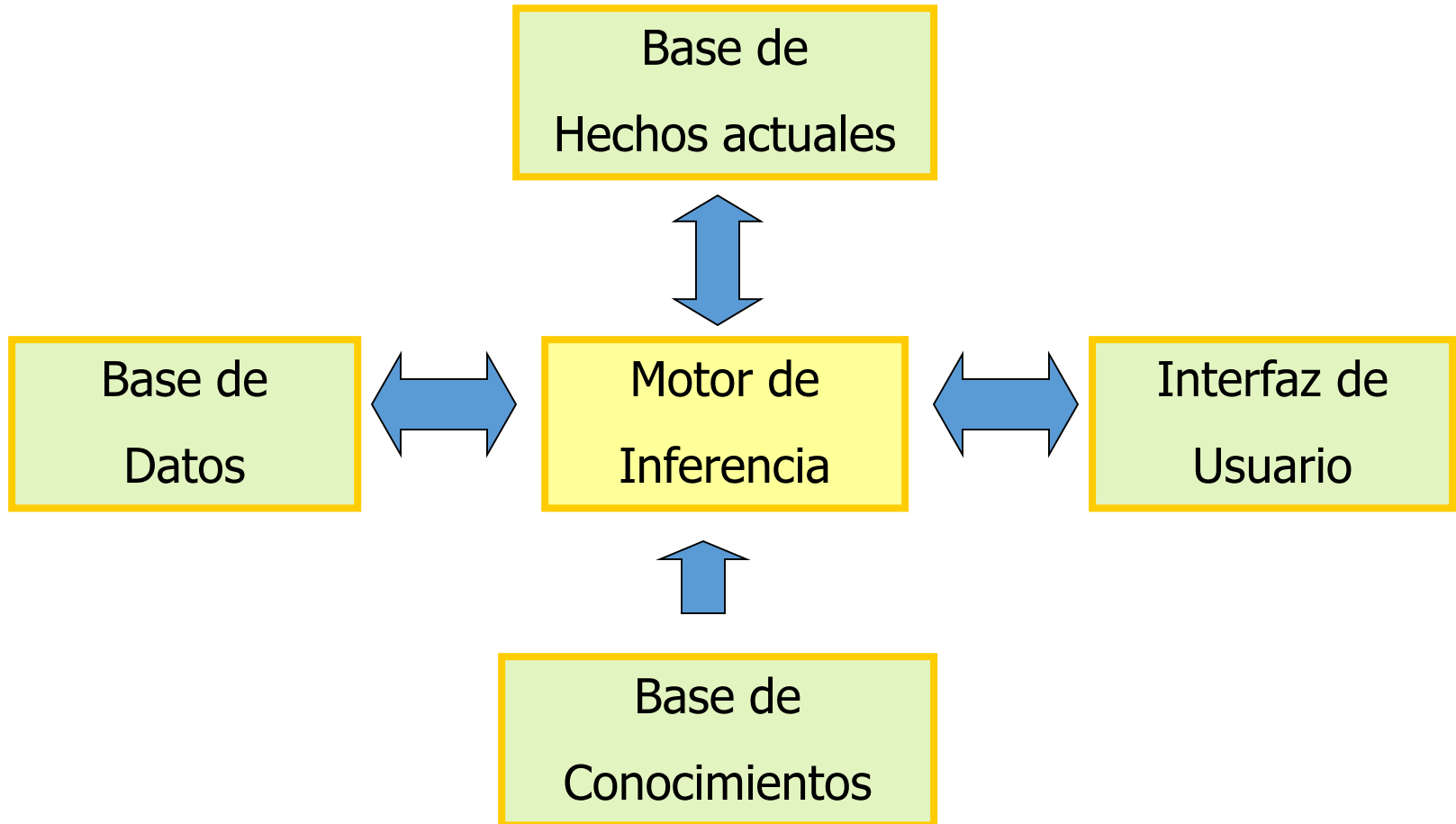
- Los sistemas basados en reglas incluyen a la lógica clásica

Proposiciones  $\rightarrow$  forma normal clausal  $\rightarrow$  cada clausula equivalente a una regla  $\rightarrow$  cada proposición equivalente a un conjunto de reglas

$$“l_1 \vee \dots \vee l_n \vee c” \longleftrightarrow “\neg l_1 \wedge \dots \wedge \neg l_n \rightarrow c”$$

- Los sistemas basados en reglas incluyen con la retractación una lógica no monótona con mas posibilidades de representación (puede deducir hechos como ciertos que después elimina)

# Componentes de un Sistema basado en reglas





# Descripción de los componentes: conocimiento

- **Base de conocimientos:** contiene las reglas y, a veces, también algunas afirmaciones iniciales (los hechos que son validos para cualquier caso). La Base de Conocimiento es específica del dominio en que el sistema puede considerarse experto.
- **Base de hechos actuales:** También conocida como memoria de trabajo. Contiene las afirmaciones que se consideran ciertas para el caso actual en el momento actual
- **Base de Datos:** contiene información acerca a variables que se utilizan en la resolución del problema. Pueden contener información relativa al histórico de casos

# Ejemplo

- Base de conocimiento:

- Hechos iniciales  $\rightarrow$  (producto Oro); (producto Plata); (evaluar);

- Reglas

(precio ?X barato)  $\Rightarrow$  indicar al usuario que compre X porque esta barato

(precio ?X caro)  $\Rightarrow$  indicar al usuario que venda X porque esta caro

(precio\_actual ?X) > (precio\_medio ?X)  $\Rightarrow$  (precio ?X caro)

(precio\_actual ?X) < (precio\_medio ?X)  $\Rightarrow$  (precio ?X barato)

(evaluar) y (producto ?X) y Not (precio\_medio ?X)  $\Rightarrow$  leer de BD y añadir (precio\_medio ?X)

(producto ?X) y Not (precio\_actual ?X)  $\Rightarrow$  preguntar al usuario, añadir (precio\_actual ?X) y actualizar BD

- Base de hechos: (producto Oro); (producto Plata); (evaluar);

(precio\_actual Oro) = 255;

- Base de datos:  $\rightarrow$

	Precio_medio	Precio-1	Precio-2	Precio-3
Oro	250	252	250	248
Plata	128	129	127	128

# Motor de inferencia

- **Motor de inferencia:** Decide que regla disparar y ejecuta la acción de dicha regla (añadir nuevos hechos a la base de hechos, eliminar hechos de la base de hechos, interaccionar con el usuario, o obtener nuevos hechos a partir de cálculos con la base de datos)
  - Se pueden disparar las reglas para las cuales haya hechos en la base de hechos que hagan cierto cada uno de los antecedentes
  - Una regla se puede disparar muchas veces con hechos distintos, no se dispara 2 veces con exactamente los mismos hechos

# Interacción con usuario y base de datos

- **Interfaz del Usuario:** se encarga de pedirle información al usuario y de mostrarle la información que va deduciendo.
- **Comunicación con BD:** es aconsejable realizar y almacenar todos los cálculos del sistema en BD (cuya implementación realiza el programador), y reducir la comunicación al proceso de leer y añadir datos. De esta forma la parte procedimental se separa del conocimiento, y así se puede modificar, depurar o adaptar el conocimiento sin necesidad de reprogramar.

# Estructura de una regla

Una regla consta de dos partes:

- **Antecedente:** o parte izquierda. Contiene las cláusulas que deben cumplirse para que la regla pueda evaluarse o ejecutarse (dispararse)
- **Consecuente:** o parte derecha. Indica las conclusiones que se deducen de las premisas (interpretación declarativa) o las acciones que el sistema debe realizar cuando ejecuta la regla (interpretación imperativa)
  - Si bombilla\_encendida entonces habitación\_iluminada
  - Si temperatura\_interior > temperatura\_deseada entonces encender el ventilado

# Parte antecedente de una regla

- El **antecedente** de una regla es la condición para que la regla pueda dispararse. Es la parte del “Si” o la parte izquierda de la regla.
- Esta formado usualmente por una conjunción de literales (las disyunciones se pueden representar como varias reglas)
- Un **literal** es una afirmación simple o la negación de una afirmación simple
- Cada **afirmación simple** puede ser una hipótesis o una relación de comparación:
  - **Hipótesis**: por ejemplo, hielo\_en\_la\_carretera, averia\_electrica, etc. En un momento dado, cada hipótesis tiene asociado un valor de verdad
  - **Dato**: por ejemplo, nivel\_de\_gasolina, temperatura\_interior, etc. Cada dato puede tomar cierto tipo de valores
  - **Relación de comparación**: se establece entre dos datos o entre un dato y un valor. Por ejemplo: nivel\_de\_gasolina = 8; temperatura\_interior < 0

# Tipos de acciones de las reglas

- Los tipos de acciones que pueden aparecer en el consecuente de una regla son:
  - **Afirmar**: se establece algún tipo de afirmación
  - **Retractar**: se modifica una afirmación anterior
  - **Actuar**: se envía una orden a los actuadores con los que está conectado el sistema
- Si `temperatura_interior > temperatura_deseada` entonces
  - `actuar encender_el_ventilador`
  - `afirmar ventilador = encendido`

# Uso de variables

- Hasta ahora se han usado cláusulas en que los datos y las instancias aparecían explícitamente. De este modo, las reglas constituían una lógica de primer orden y su capacidad expresiva era mínima. Como consecuencia, habría que definir una regla particular para cada individuo o elemento,, lo cual haría impracticable este método de representación. Por tanto, nos interesa poder representar mediante reglas afirmaciones equivalentes a las que aparecen en la lógica de predicados, y para ello se hace necesario introducir variables en las reglas

Si  $\exists x$  es catedrático entonces  $\exists x$  es doctor



# ¿Cómo indicar la falsedad de una afirmación?

- Algunos sistemas indican explícitamente si una determinada hipótesis es cierta, falsa, no investigada o desconocida (cuando el sistema ha tratado de averiguar su valor de verdad, pero no ha llegado a ninguna conclusión).
- Otros sistemas sólo almacenan las hipótesis confirmadas. El considerar falsa toda proposición que no se encuentre en la Base de Afirmaciones ni pueda deducirse de la información disponible, se conoce como Axioma del Mundo Cerrado.
- Otros sistemas (como Goldworks) son menos explícitos a la hora de establecer una semántica para las proposiciones no deducibles, y el programador tiene una cierta libertad para determinar si las va a considerar como desconocidas o falsas

# Datos univaluados o multivaluados

- Algunos sistemas admiten la posibilidad de que algunos datos sean univaluados, mientras que otros pueden ser multivaluados.
- Cuando un dato es univaluado, la asignación de un valor elimina el que pudiera estar asignado anteriormente, y de este modo podemos evitar que en la Base de Conocimientos se introduzca información contradictoria. En el caso de datos multivaluados, la asignación de un valor no elimina asignaciones anteriores

# Retractabilidad

- Una de las ventajas de los Sistemas Basados en Reglas frente a la Lógica Clásica es la capacidad de retractar afirmaciones anteriores como consecuencia de nuevas inferencias.
- Ahora bien, ¿qué ocurre si la información que ahora se retracta ha sido utilizada para obtener otras conclusiones?
  - Las herramientas más avanzadas permiten que sea el diseñador del sistema quien tome esta decisión. Para ello, al definir una regla hay que indicar el tipo de dependencia
- Una dependencia es **reversible** si cuando se retracta el antecedente debe retractarse en consecuente. Es **irreversible** si al retractar el antecedente no puede retractarse el consecuente.
  - *Dependencia reversible*  
Si bombilla\_encendida entonces habitación\_iluminada
  - *Dependencia irreversible*  
Si bombilla\_encendida entonces película\_velada

# Inferencia

- Mientras exista alguna regla que se pueda lanzar con algunos hechos de la base de hechos y que todavía no se haya lanzado con esos hechos:
  - 1: se seleccionan todas las reglas que se pueden lanzar
  - 2: se ordenan de acuerdo a algún criterio
  - 3: se ejecuta la primera de esas reglas

# Tipos de encadenamiento

- **Encadenamiento hacia adelante o basado en datos:** se da cuando la información introducida en el sistema hace que se ejecute una regla, y la conclusión obtenida permite que se ejecuten otras reglas.
- **Encadenamiento hacia atrás o basado en objetivos:** consiste en buscar una regla que permita establecer cierta conclusión. Si en la base de afirmaciones no se encuentra la afirmación que permita lo anterior, se puede establecer ésta como objetivo intermedio y buscar una regla que nos lleve a esta conclusión. Y así sucesivamente.
- En general, el encadenamiento hacia delante utiliza sólo los datos disponibles, mientras que el encadenamiento hacia atrás suele solicitar al usuario la afirmación que no ha podido deducir.
- El encadenamiento hacia adelante es menos específico, pues en principio ejecutará todas las reglas posibles en función de la información introducida. En cambio, el razonamiento hacia atrás lleva implícito un proceso de búsqueda, por lo que es más específico y, en consecuencia, más eficaz

# Control del razonamiento

- Se llama **control del razonamiento** al mecanismo que permite seleccionar qué regla ejecutar en primer lugar, cuando hay varias disponibles. El control del razonamiento es importante por tres motivos:
  - Por el **contenido** de la inferencia: en algunos casos, las conclusiones pueden depender del orden en que se apliquen las reglas. Por eso, las reglas más específicas, y en particular las que tratan las excepciones, deben activarse primero para que no se apliquen otras reglas más generales.
  - Por **eficiencia**: utilizar la regla adecuada llevará rápidamente a una conclusión, mientras que una elección equivocada puede hacer que se pierda mucho tiempo en un camino que no conduce a ninguna parte.
  - Por el **diálogo** que genera: es importante que el sistema no pregunte al usuario la información que pueda deducir por sí mismo y, además, que el orden en que se solicita la información sea razonable

# Mecanismos simples de control

- Algunos mecanismos sencillos para controlar el razonamiento (o resolver conflictos) son:
  - Ordenar las reglas en la base de conocimientos, colocando en primer lugar las que deseamos que se examinen antes
  - Ordenar los literales dentro de cada regla, para que unos sean examinados antes que otros. Cuando fracasa uno de ellos, los que aparecen después dentro de la misma regla no necesitan ser investigados.
  - Añadir nuevos literales relacionados con el punto de inferencia en que nos encontramos, con el fin de controlar en cada momento qué reglas deben aplicarse y cuáles no

# Metarreglas

- Las metarreglas son reglas destinadas a razonar acerca de otras reglas.
- Ejemplos de metarreglas son

- *Dependiente del dominio:*

Si objetivo = determinar\_cualificacion

?r1 es regla

r1.antecedente menciona titulación

Entonces r1.prioridad = 100

- *Independiente del dominio*

Si objetivo = ?x

?r1.consecuente no menciona ?x

Entonces r1.prioridad = -500