

Tema 3: Sistemas Basados en Conocimiento



Material desarrollado por distintos autores del departamento de ISIA



Breve Historia de la IA

- Desde los 50 hasta los 60 la IA se centró en sistemas que fuesen capaces de resolver ***problemas genéricos***
 - ***Test de Turing***
- Desde mediados de los 60 hasta mediados de los 70 se centra en modelar el ***comportamiento humano experto*** al resolver problemas
 - Adquirir el **conocimiento de un experto humano** y representarlo para emplearse en un ordenador
 - Reglas, redes semánticas, objetos/frames, fórmulas lógicas.
- Desde entonces
 - Ingeniería del conocimiento
 - Importancia de **representar explícitamente el conocimiento**
 - Diversas fuentes de conocimiento: bbdd, datos, textos, web, expertos,...



- DENDRAL fue el primer Sistema Experto Universidad de Stanford (mediados de la década de 1960)
- MYCIN (Univ Standford, 1976)
 - Diagnóstico y tratamiento de desordenes de la sangre
 - Alrededor de 400 reglas que relacionen condiciones a posibles interpretaciones (o diagnósticos)
 - QUE SINTOMA PRINCIPAL
 - TIENE EL PACIENTE?
 - > Dolor de cabeza.

 - SUFRE DE ASMA(s/n)?
 - > n.

 - SUFRE DE ULCERA
 - PEPTICA(s/n)?
 - > n.
 - Razonamiento basado en incertidumbre
 - Separa base de conocimiento del motor de inferencia.
 - EMYCIN: Empty MYCIN

- RECOMENDACIÓN:
- TOMAR ASPIRINA

- EXPLICACION:
- 1. EL PACIENTE TIENE DOLOR DE CABEZA.
- 2. LA ASPIRINA ALIVIA EL DOLOR DE CABEZA.
- 3. EL PACIENTE NO TIENE NINGUNA DE LAS CONTRAINDICACIONES DE LA ASPIRINA

IF the infection is meningitis
AND patient has evidence of serious skin or soft tissue
infection
AND organisms were not seen on the stain of the culture
AND type of infection is bacterial
THEN There is evidence that the organism (other than
those seen on cultures or smears) causing the infection
is *Staphylococcus coagulans*.

MYCIN Sample Rule

Human-Readable Format

IF the stain of the organism is gram negative
AND the morphology of the organism is rod
AND the aerobocity of the organism is gram anaerobic
THEN there is strong evidence (0.8)
that the class of the organism is enterobacteriaceae

MYCIN Format

IF (AND (SAME CTEXT GRAM GRAMNEG)
(SAME CTEXT MORPH ROD)
(SAME CTEXT AIR AEROBIC)
THEN (CONCLUDE CTEXT CLASS ENTEROBACTERIACEAE
TALLY .8)

Factor de certidumbre

¿Qué es un Sistema Basado en Conocimiento?

- ❑ ¿Qué los caracteriza?
- ❑ Diferencias con los sistemas tradicionales que conocéis
- ❑ ¿Son lo mismo los SBC que los Sistemas Expertos?



Sistemas Basados en Conocimiento

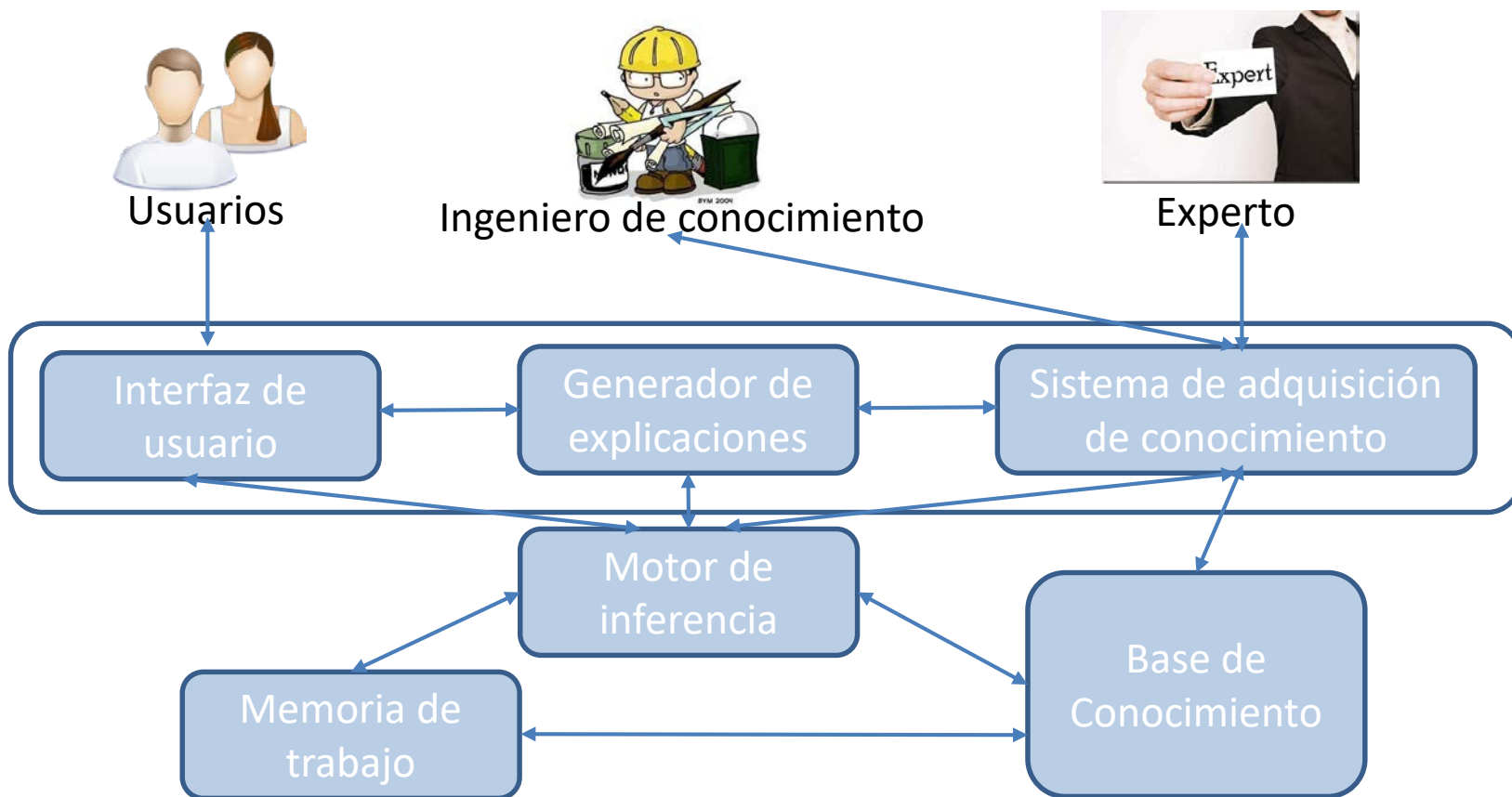
- **Paradigmas básicos de razonamiento e inferencia en IA (no excluyentes)**
 - Lógica
 - Reglas
 - Búsqueda heurística
 - Conexionista
 - Probabilístico
 - Basado en la experiencia
- **¿Qué representación de conocimiento es necesario para cada uno? ¿cómo se adquiere? ¿se puede reutilizar el conocimiento entre distintos paradigmas de razonamiento?**

Sistemas Basados en Conocimiento

- La característica principal es la separación entre:
 - Conocimiento específico del problema (Base de conocimiento)
 - Metodología para solucionar el problema (Motor de inferencia)
- Otras características:
 - Representación explícita del conocimiento
 - Capacidad de razonamiento independiente del dominio
 - Capacidad para explicar sus conclusiones y el proceso de razonamiento

SBC. Arquitectura general

- **Base de conocimiento (BC):** Conocimiento estático sobre el dominio
- **Memoria de trabajo (MT):** Conocimiento sobre la situación actual
- **Motor de inferencia**
- *Generador de explicaciones*



Información vs conocimiento

- Información: Conjunto de datos sin interpretar que sirven de entrada al sistema
 - Datos numéricos de una analítica de sangre
 - Resultado deportivo: 30-15
- Conocimiento: Conjunto de datos que modelan la experiencia que se tiene sobre un dominio o que surgen de interpretar la información de entrada al sistema
 - La interpretación de los valores de la analítica de sangre o los métodos para diagnosticar enfermedades a partir de los valores de la analítica de sangre
 - Saber como se puntúa en un deporte

Propiedades del conocimiento

- Voluminoso
- Impreciso
- Incierto
- **Heurístico**
- Incompleto
- Evoluciona (puede cambiar)
- Hay que adquirirlo y representarlo

¿Qué es el conocimiento heurístico?

- A partir de los 70 → **Uso intensivo del conocimiento explícito**
 - Una parte del conocimiento de un experto la constituyen relaciones de causa y efecto
 - Estas relaciones surgen a partir de la **experiencia** del experto y se llaman **heurísticas**
- Sistemas Expertos \subseteq Sistemas basados en el conocimiento
 - Los SBC pueden usar otras fuentes de conocimiento (no experto)
 - SBC es “Un sistema computerizado que *utiliza conocimiento de un dominio* para resolver un problema específico de ese dominio”
 - Un sistema experto
 - “La solución será esencialmente la misma que la proporcionada por un experto en ese dominio”
 - “Programa que se comporta como un experto en un dominio de aplicación”
 - Entre otras cosas un experto reduce rápidamente la búsqueda al reconocer situaciones (patrones) y utilizar los métodos y reglas adecuados para descubrir la solución (“sabiduría heurística”)

Conocimiento heurístico

- ¿Qué caracteriza el conocimiento experto? La experiencia, por ejemplo, un médico o un técnico
- La parte del conocimiento más difícil es el **conocimiento informal o atajos** que permiten a un experto alcanzar rápidamente la solución de un problema sin tener que llevar a cabo un análisis detallado del problema
- Parte del conocimiento en los sistemas expertos
 - Es **conocimiento heurístico**, es decir, no representa un análisis en profundidad del problema, sino una solución “aceptable”
 - Es el conocimiento específico aplicable a cada situación específica del dominio → “sabiduría heurística”
 - Relaciones de causa y efecto que surgen a partir de la experiencia del experto
- Pueden ser reglas de actuación, el orden de aplicación de inferencias, reglas causa-efecto, “recetas”, casos prototípicos,....

Propiedades del conocimiento

- Voluminoso
- **Impreciso**
- **Incierto**
- Heurístico
- **Incompleto**
- Evoluciona (puede cambiar)
- Hay que adquirirlo y representarlo

Conocimiento incompleto (I)

- En general, suele ser imposible representar todo el conocimiento
- Causas:
 - Falta de conocimiento en la BC
 - Mecanismos de inferencia insuficientes
- ¿Cómo actúa el sistema frente a este hecho?:
 - **Hipótesis de mundo cerrado** → Todo lo que no se puede obtener de la BC es falso
 - **Hipótesis de mundo abierto** → Todo lo que no se puede obtener de la BC no se sabe

Hipótesis de mundo cerrado

- Los asertos verdaderos están incluidos en la BC o pueden ser derivados de ella
- Todo lo que no se puede obtener de la BC es falso
- La respuesta para preguntas de las que no tengo información es siempre FALSO
- Se niega la incompletitud → Esto puede ser engañoso porque el usuario puede pensar que el sistema tiene evidencias de que realmente la respuesta es FALSO
- Fácil de implementar y se utiliza mucho pero hay que ser consciente de que puede no ser válido para algunas situaciones

JUAN
24 años
Soltero
Médico

¿Juan es moreno? NO

JUAN
24 años
Soltero
Rubio

¿Juan es moreno? NO

Hipótesis de mundo abierto

- Todo lo que no se puede obtener de la BC no se sabe
- La respuesta para preguntas de las que no tengo información es siempre **NO SÉ**
- Distingue entre lo que es **falso** basado en evidencias de la BC y lo que **no se sabe**

JUAN
24 años
Soltero
Médico

¿Juan es moreno? NO SÉ

JUAN
24 años
Soltero
Médico
Rubio

¿Juan es moreno? NO

Conocimiento por omisión (I)

- Conocimiento que se asume implícitamente mientras no se niegue explícitamente
- Las excepciones se establecen a posteriori
- Ejemplo: mecanismos de herencia con excepciones
 - Por ejemplo, la propiedad poder volar es habitual en las aves, es una propiedad habitual pero no esencial.
 - Si un individuo x es un ave puedo inferir que vuela, pero si me dicen que el individuo x es un pingüino → diré que no vuela y hay que borrar todas las inferencias que se hayan hecho teniendo en cuenta que x volaba

Conocimiento por omisión (II)

- Sistemas **monótonos**: Lo verdadero no puede dejar de serlo, no puedo retractarme de algo ya inferido
 - Son más fáciles de implementar pero más limitados
- Sistemas **no monótonos**:
 - Permiten usar conocimiento por defecto que puede modificarse después (los pajaros vuelan)
 - Las conclusiones establecidas en un cierto momento pueden dejar de ser ciertas si llega nueva información
 - Requieren garantizar la consistencia → TMS (True Maintenance Systems)
 - Permite saber qué inferencias se han hecho a partir de otras y cuáles hay que deshacer si algo deja de ser verdad.
 - Ejemplo del asesino ABC

Conocimiento inseguro

- El conocimiento añadido al sistema se acompaña de un factor de certeza que se usa en las inferencias
 - Por ejemplo: “Si el paciente presenta la piel amarilla entonces el diagnostico en el 70% de los casos es hepatitis y en el 30% de los casos ictericia”

Conocimiento impreciso

- El conocimiento cuenta con predicados o cuantificadores vagos (no precisos)
 - “Si la humedad es alta, la presión es baja y está muy nublado, entonces lloverá”
 - “Si eres alto puedes jugar en el equipo”
- El factor de certeza es 1 pero es conocimiento impreciso
 - ¿cuánto es alta?, ¿cuánto es baja?, ¿cuánto es muy nublado?
- **Lógica difusa**, cuantificadores especiales...
 - una persona es alta si mide mas de 1.80mts, pero de igual forma se considera a una persona como alta si mide 1.7999mts
 - En lógica tradicional hay demarcaciones estrictas
 $A = \{ x \mid x > 1.8 \}$ Una persona que mide 1.799999mts es baja!

Propiedades del conocimiento

- Voluminoso
- **Impreciso**
- **Incierto**
- **Heurístico**
- **Incompleto**
- Evoluciona (puede cambiar)
- **Hay que adquirirlo y representarlo**

Representación del conocimiento. Componentes

- Representar el conocimiento consiste en escribir con un lenguaje determinado la información de un dominio
 - Sintaxis: símbolos y conjunto de reglas para combinarlos
 - Semántica: significado de las expresiones construidas
- **Representación simbólica:** las etiquetas simbólicas nos dan “poder” sobre los objetos que representan.
 - Si tiene nombre podemos asociar con esta etiqueta todo lo que sabemos sobre este símbolo.
- La aproximación subsimbólica (o conexionista) no representa explícitamente el conocimiento

Conceptualización, Formalización e Implementación

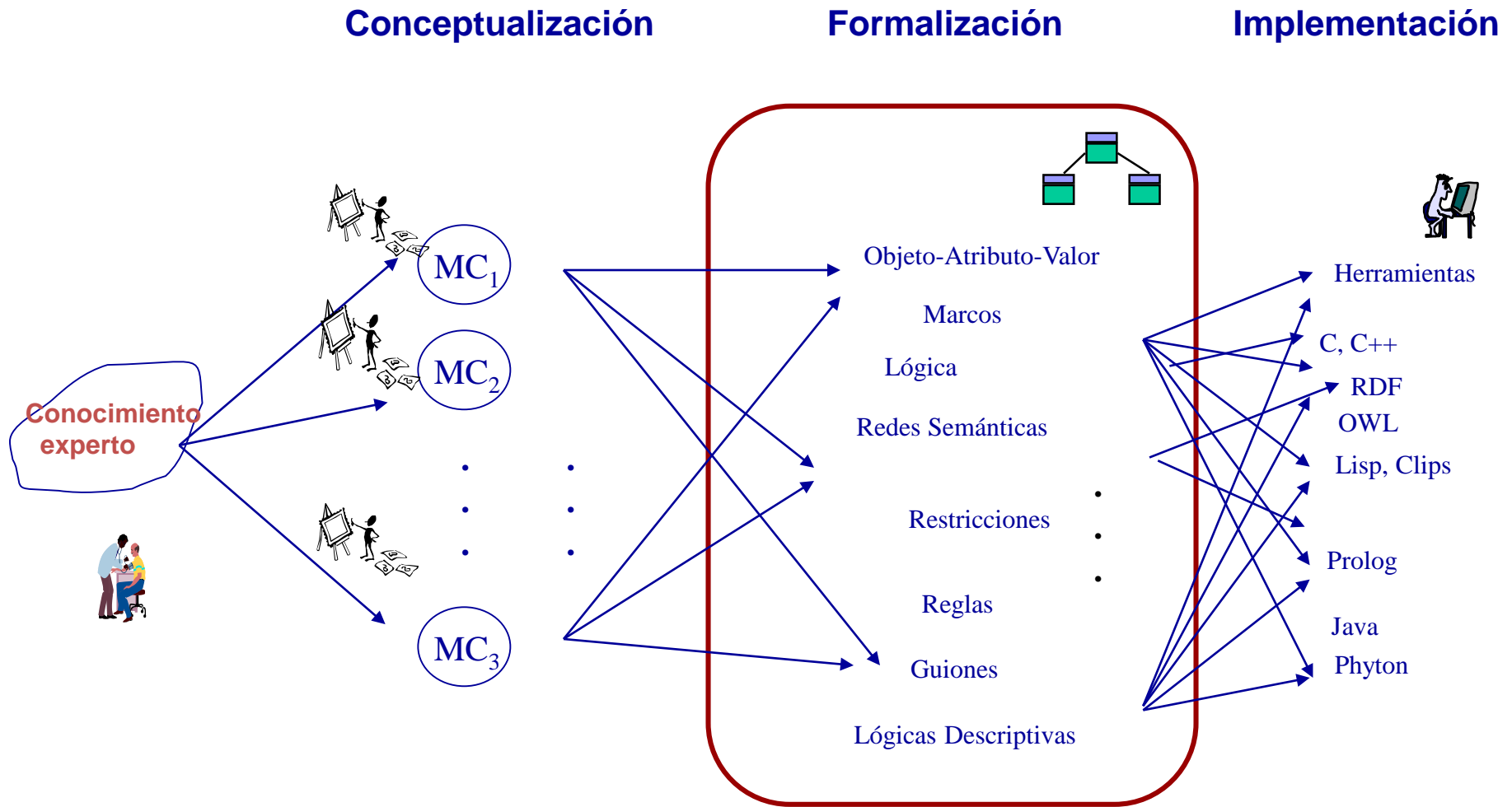


Figura: SBC (UPM) Asunción Gomez Perez

Ingeniería del conocimiento

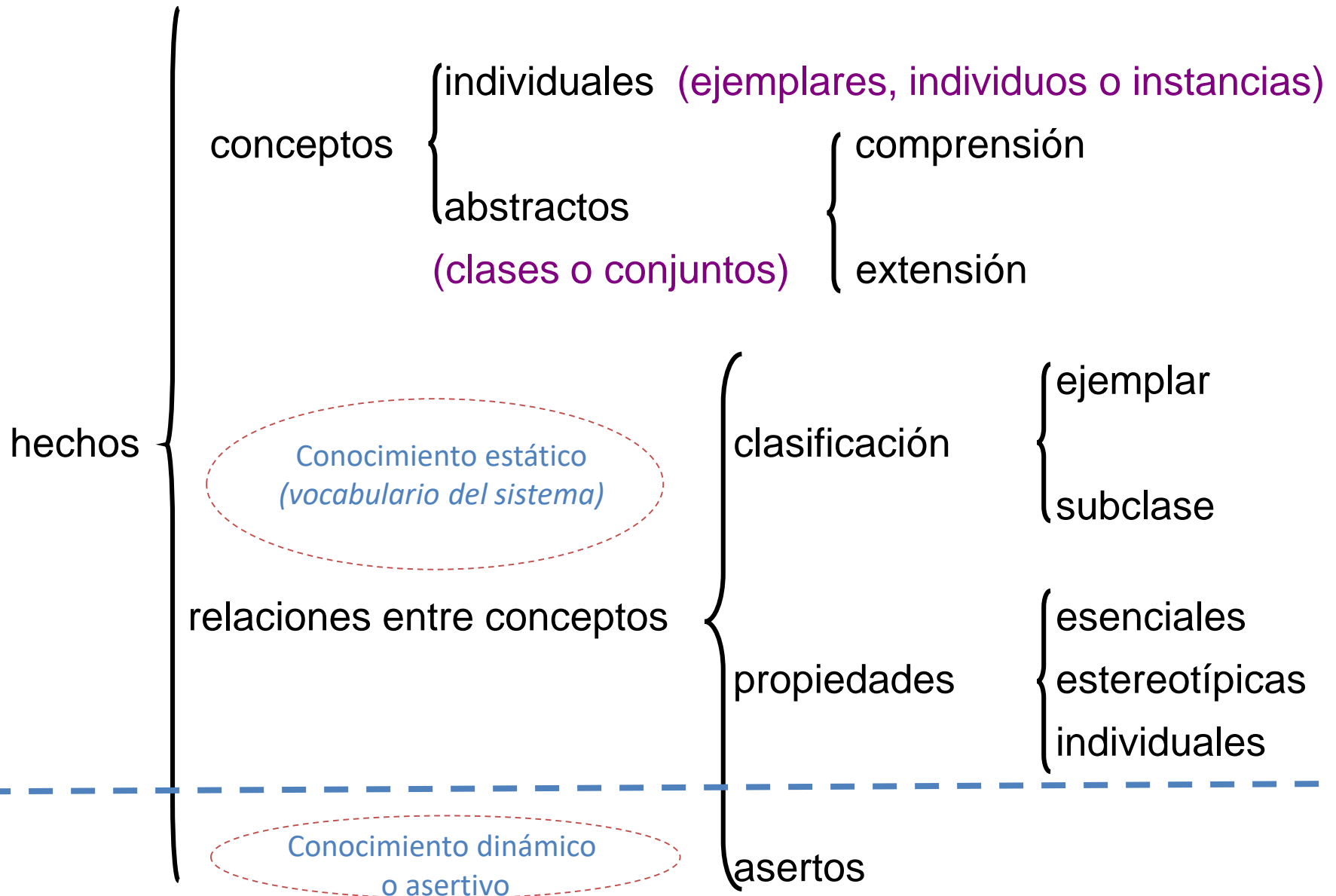
- El proceso de construir una BC inicial junto con un motor de inferencia se llama **ingeniería del conocimiento**
- El ingeniero del conocimiento se encarga de:
 - Construir **la base de conocimiento**:
 - Investiga el dominio concreto y dialoga con el experto humano
 - Conceptualiza el dominio
 - Crea una representación formal de los objetos y relaciones del dominio
 - Implementación
 - Motor de inferencia → Cómo se puede razonar con el conocimiento adquirido

Reglas, casos, taxonomías conceptuales, atributos, rangos, terminología, restricciones, dependencias, procedimientos,...

Tipos de conocimiento

- **Factual o declarativo** (representación de hechos)
 - *Explícito*: Se introduce directamente (basado en observaciones)
 - Conceptos individuales (ejemplares) o abstractos (clases)
 - Relaciones entre conceptos
 - *Implícito*: Se infiere a partir del conocimiento explícito
- **Procedimental**: Indica cómo actuar en diversas situaciones
- **Meta-conocimiento o conocimiento de control**: Conocimiento sobre el propio conocimiento, para facilitar su gestión

Conocimiento factual



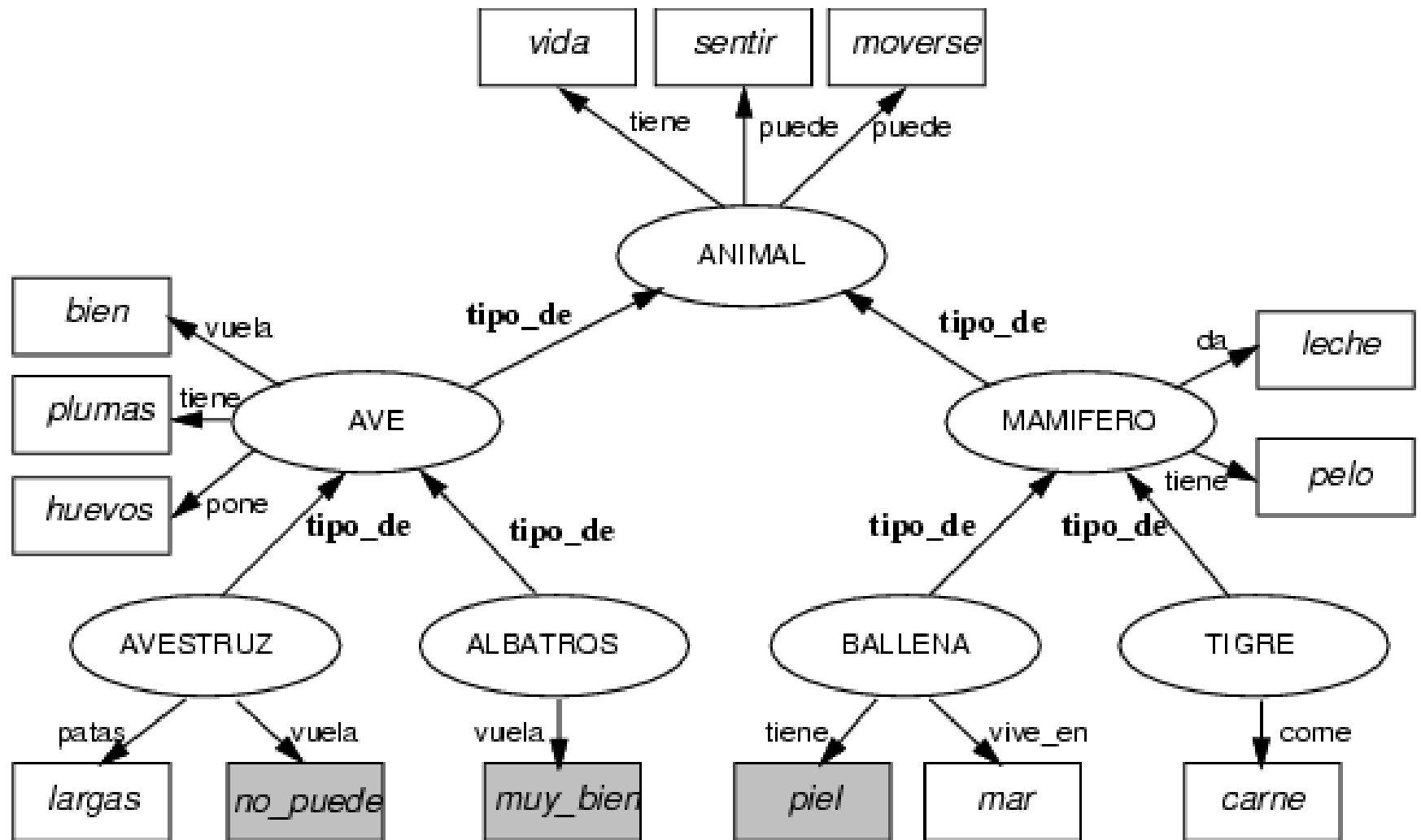
Conocimiento factual o declarativo

- Conceptos:
 - **Individuales** (ejemplares o individuos o **instancias**)
 - Por ejemplo, el borrador1 o el alumno32
 - **Abstractos** (**clases** o conjuntos o conceptos):
Conjunto de individuos con propiedades comunes que permiten agruparlos
 - Por ejemplo: Los borradores o los alumnos
- Relaciones entre conceptos:
 - Relaciones de clasificación: ejemplar o subclase
 - Propiedades
 - Asertos

Conocimiento factual. Relación de clasificación

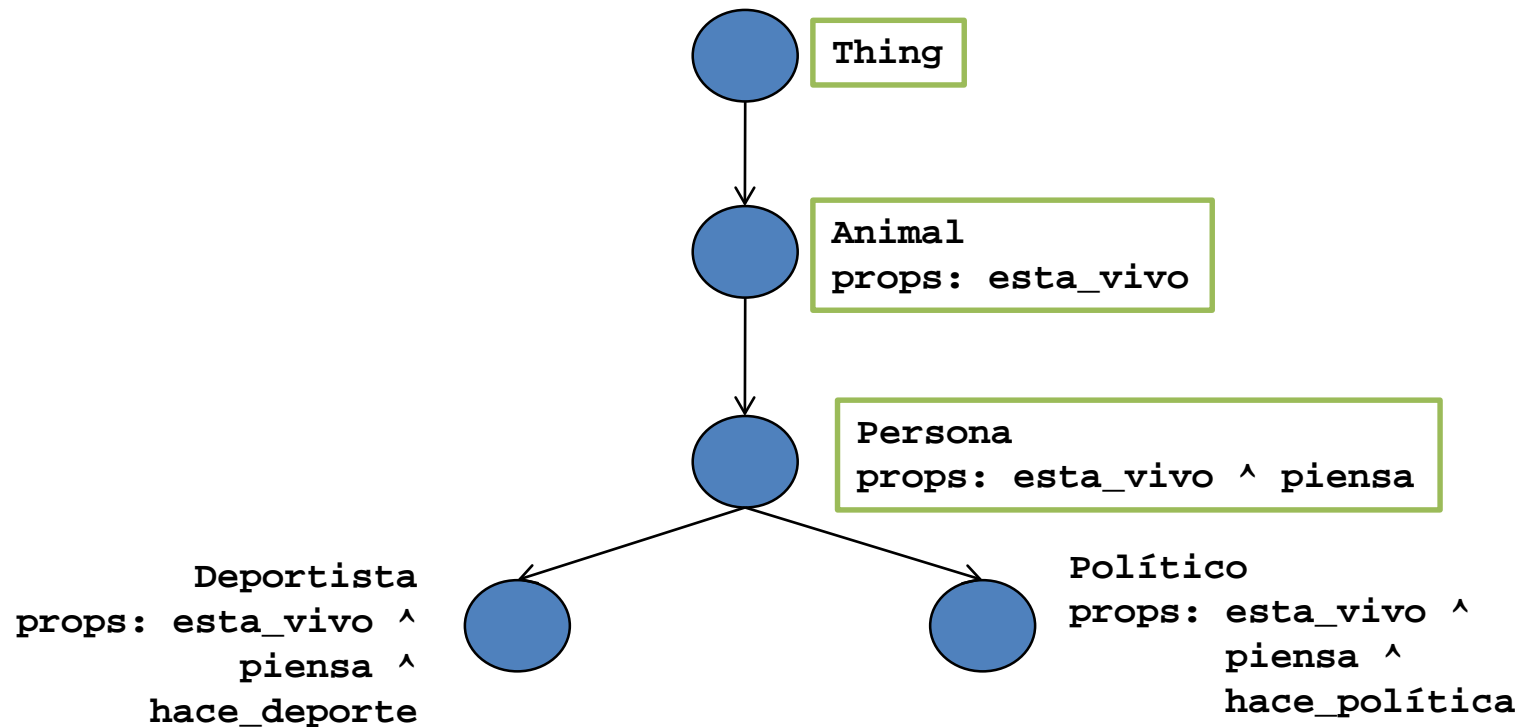
- Relación de clasificación entre:
 - Los ejemplares y las clases a las que pertenecen
 - Las subclases y las superclases
- La relación de clasificación permite usar la **herencia** como mecanismo de inferencia.
 - Con especificar que $A \in \text{ClaseA}$, ya sabemos sus propiedades (las mismas que tenga ClaseA)
 - Se reduce el tamaño de la BC y ayuda a prevenir inconsistencias al añadir nuevas clases e instancias
- El conocimiento obtenido mediante herencia es conocimiento implícito

Ejemplo



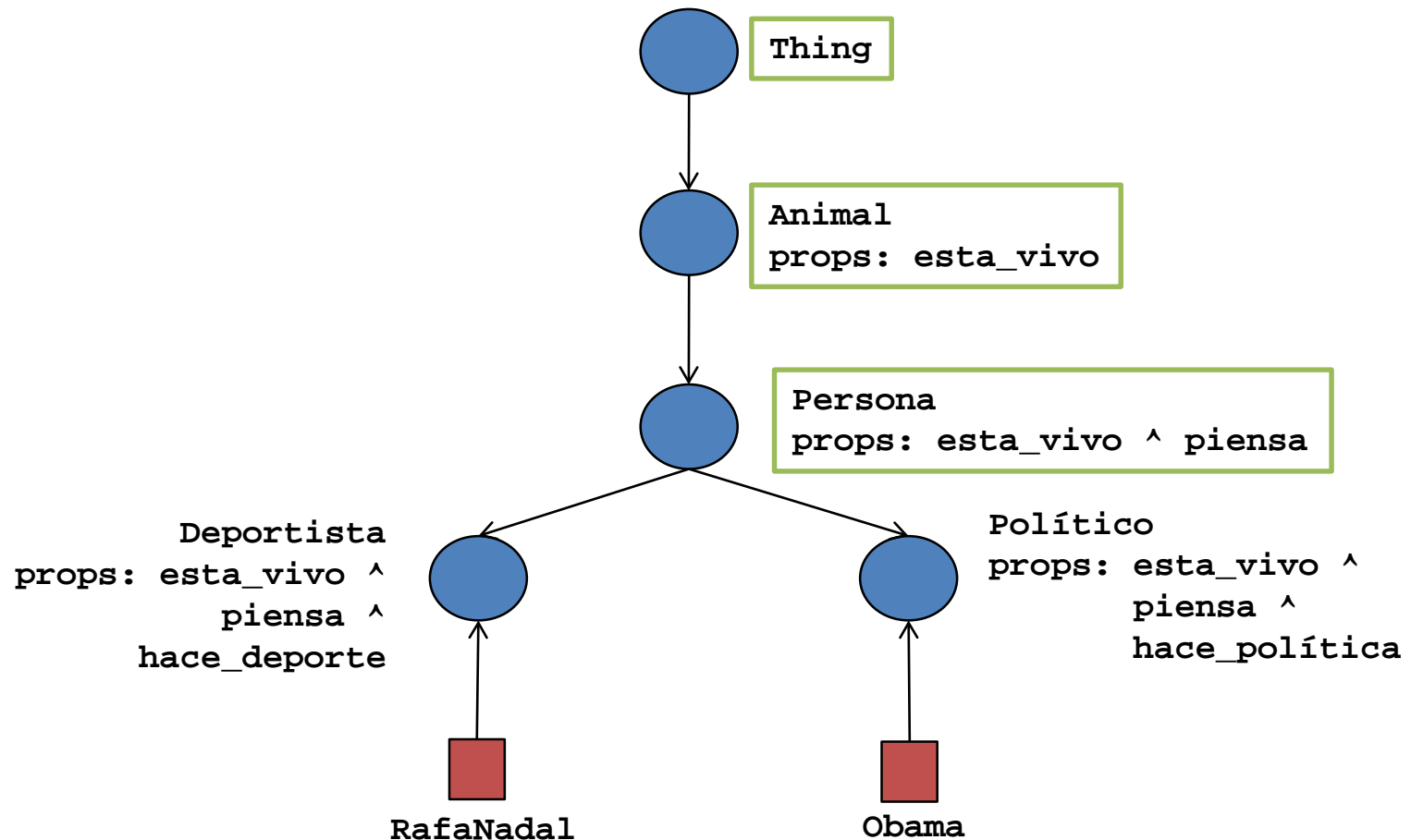
Ejemplo de relación de clasificación en una red semántica

Clases \approx Conceptos \approx Categorías



- ❑ Los conceptos se organizan mediante herencia \rightarrow es_un
- ❑ Los conceptos se definen mediante las propiedades
- ❑ Los subconceptos heredan las propiedades de los conceptos padres

Individuos \approx Instancias



- Los individuos son los distintos objetos del dominio.
- El razonamiento debe hacerse a nivel de las clases

Conocimiento procedimental

- Conocimiento que indica cómo actuar en ciertas situaciones
- Se puede expresar mediante
 - Reglas
 - Fórmulas o algoritmos
 - Ej: calcula la edad a partir de la fecha de nacimiento y la fecha actual

Meta-conocimiento

- Conocimiento que facilita la gestión del conocimiento en la BC → Permite garantizar la **consistencia de la BC**
 - Por ejemplo, fecha de nacimiento < fecha actual
- Permite mejorar la eficacia y eficiencia del sistema → no todo el conocimiento se tiene que manejar igual
- Ejemplo:
 - Prioridad de reglas → mirar el corazón y después fiebre
 - Indicar qué estados son mejores que otros
 - Que reglas son preferibles a otras en una cierta situación
 - **Orden** de consecución de subobjetivos
 - Secuencias útiles de reglas para aplicar en una cierta situación
 - ...

Técnicas de representación

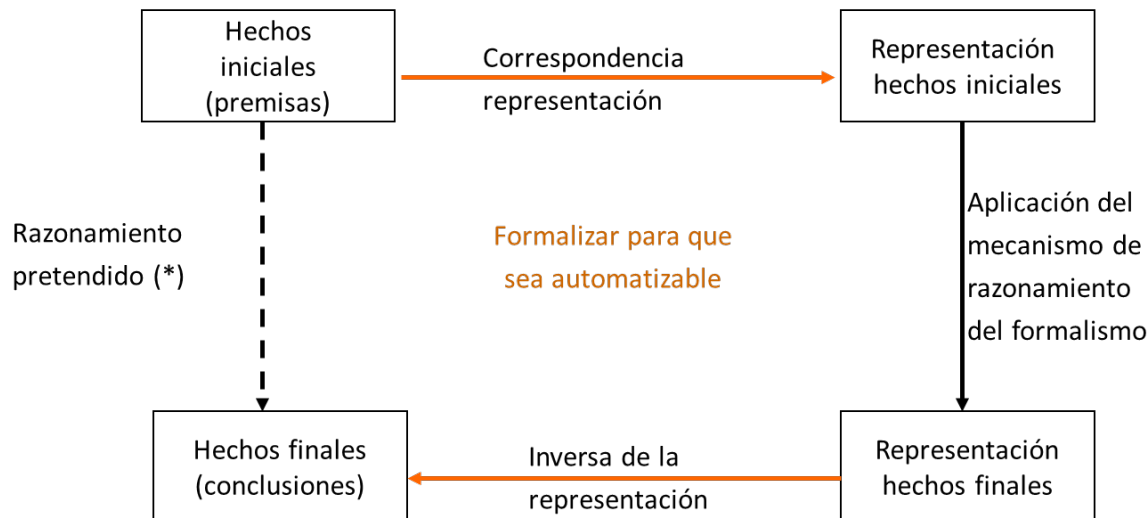
- Representaciones basadas en relaciones
 - Lógica
 - Redes semánticas
- Representaciones estructurada (basadas en objetos)
 - Marcos
 - Objeto-Atributo-Valor
- Representaciones basadas en acciones
 - **Sistemas de reglas (o de producción)**
 - Guiones
- Todos los sistemas anteriores se pueden combinar o variar dando lugar a nuevas técnicas

Técnicas de representación adecuadas

- Una técnica de representación es adecuada para nuestro problema si:
 - Puede representar todo el conocimiento necesario → Potencia expresiva
 - Mediante los mecanismos de inferencia de dicha técnica podemos obtener todo lo que necesitamos
- La técnica de representación seleccionada añadirá dificultad o facilidad a la resolución del problema → depende de su adecuación

Razonamiento

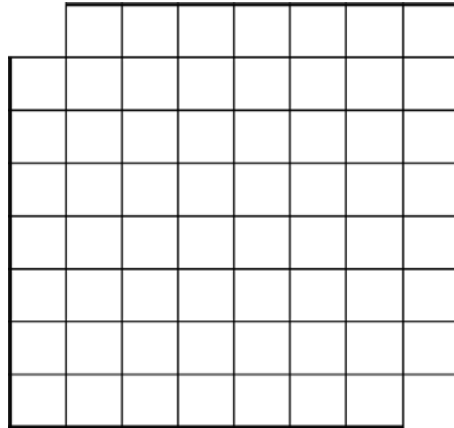
- Partimos de un conjunto de hechos iniciales o premisas que se cumplen en nuestro mundo
- Queremos un mecanismo de razonamiento que nos permita llegar a hechos finales (“conclusiones”)
- Para que las máquinas sean capaces de “razonar” necesitamos un método formal para representar los hechos → La representación que elijamos influye en que el proceso sea más fácil o más complicado



Influencia de la representación. Ejemplo (I)

Problema del tablero de damas mutilado

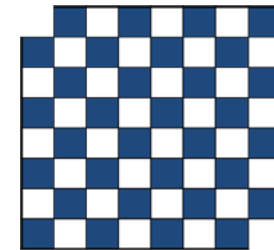
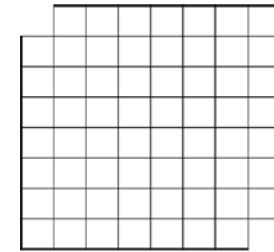
- Tenemos un tablero de 8×8 recortado en dos esquinas opuestas



- ¿Es posible recubrir todo el tablero con todas las fichas de domino sin que las fichas se solapen entre sí y queden trozos fuera?

Influencia de la representación. Ejemplo (II)

- Representación 1 → Tablero en blanco
 - Vamos probando todas las posibilidades de recubrirlo a ver si damos con la buena
- Representación 2 → Tablero pintando las casillas de dos colores: blancas y negras
 - Vamos probando todas las posibilidades de recubrirlo a ver si damos con la buena
- Representación 3 → Número de cuadros negros y número de cuadros blancos
 - Con esta representación vemos rápidamente que el problema no tiene solución porque cada ficha estará a la vez sobre una casilla blanca y otra negra, por lo que necesitamos el mismo número de casillas blancas y negras y tenemos 30 blancas y 32 negras
- No existe una representación óptima para todos los problemas



Mecanismos de inferencia

- Los mecanismos de inferencia permiten obtener nuevo conocimiento a partir del conocimiento actual
 - Tipos de inferencia:
 - **Deducción:** A partir de leyes generales obtenemos conocimiento particular
 - Si las premisas son ciertas, la conclusión es verdadera
 - **Inducción:** Es la generalización de la información extraída de casos particulares
 - No se puede garantizar la validez de la información extraída
- Por ejemplo, si vemos un montón de hojas y todas ellas son verdes inferimos que todas las hojas son verdes

Mecanismos de inferencia

- Tipos de inferencia:

- **Abducción:** Es la capacidad de generar explicaciones plausibles para un cierto hecho que ha ocurrido

Por ejemplo, un paciente tiene la piel amarilla. Sabemos que algunas enfermedades del hígado provocan el color amarillo en la piel => probablemente el paciente tenga una enfermedad del hígado. Es posible que existan multitud de explicaciones (o justificaciones) posibles pero el objetivo es obtener las más plausibles (el paciente podría haberse pintado de amarillo...)

- La comprensión de texto es en muchos casos abducción porque en la historia sólo figuran pistas a partir de las cuales nosotros “entendemos” lo que ha pasado (construimos en nuestra mente una explicación razonable)

Propiedades de una buena representación

- Idoneidad representativa: Todo el conocimiento necesario del dominio se puede representar
- Idoneidad inferencial: Es posible manipular los símbolos del formalismo de representación e inferir nuevo conocimiento
- Eficiencia inferencial: Es posible incorporar meta-conocimiento que permita mejorar los procesos de razonamiento
- Eficiencia adquisitiva: Es posible adquirir fácilmente nuevo conocimiento del exterior manteniendo la consistencia con el conocimiento existente
- La propiedad más importante es la idoneidad representativa
- Dependiendo del problema concreto estaremos interesados en alguna/s propiedad/es concreta/s

SBC. Aplicaciones

- Diagnóstico y reparación de sistemas
- Monitorización y control de procesos para la detección de anomalías, desviaciones o tendencias
- Sistemas expertos
- Planificación
- Configuración
- Gestión de conocimiento (sistemas de recomendación y ayuda)
- ...

Sistemas basados en reglas



Sistemas basados en reglas (SBR) o de producción

- Arquitectura y funcionamiento de los SBR
 - Memoria de trabajo. Representación de hechos
 - Base de reglas. Representación de reglas
 - Motor de inferencia. Ciclo de funcionamiento
- El proceso de razonamiento
 - Encadenamiento progresivo y encadenamiento regresivo
- Implementación
 - Fase de reconocimiento. Algoritmo RETE
 - Fase de selección. Estrategias de resolución de conflictos
 - Sistemas de producción en CLIPS
- Ventajas e inconvenientes de los SBR

Tipos de programas

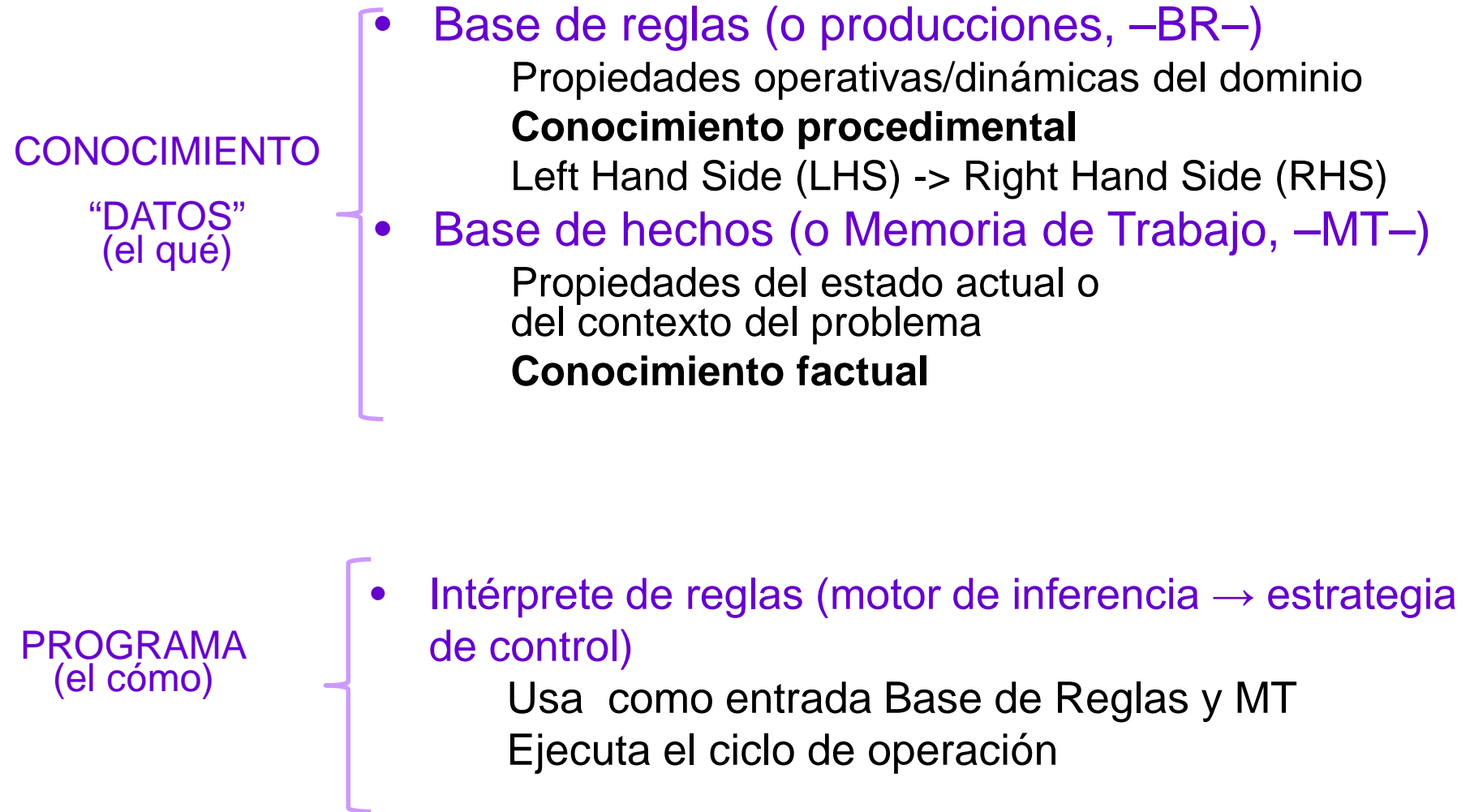
- Programas secuenciales

- El flujo de control y la utilización de los datos están especificados de manera rígida por el programa

- Programas basados en eventos

- El programa responde directamente a un amplio rango de sucesos, algunos imprevistos, en lugar de hacerlo a datos esperados
- Reconocen patrones en los datos, y seleccionan trozos de código en el sistema para que se activen

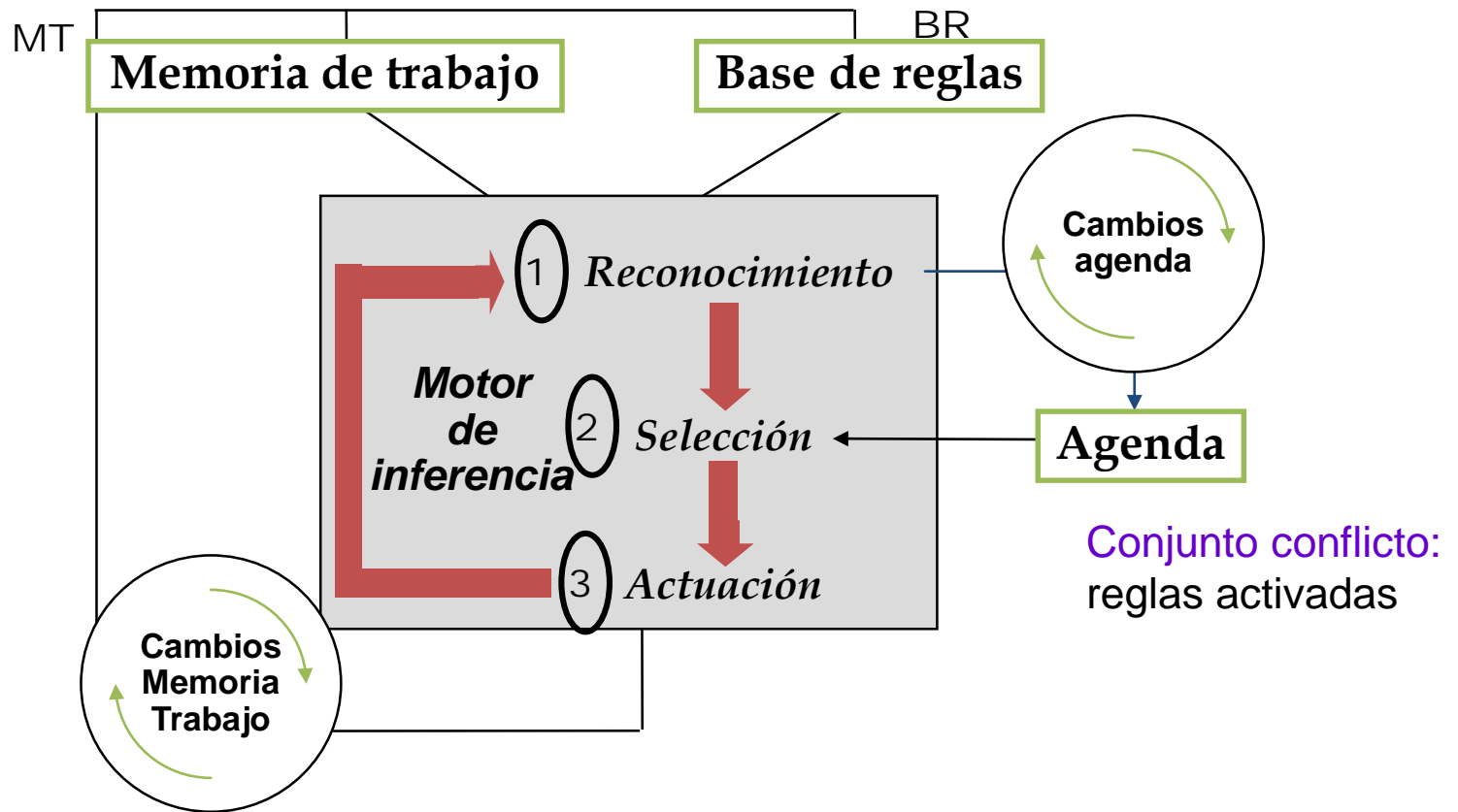
Arquitectura de un sistema de Reglas



Motor de Inferencia: Ciclo de Operación

1. **Reconocimiento o equiparación** (*matching*): LHS
 - Qué reglas son aplicables: crea el **conjunto conflicto** o **agenda**
 2. **Selección o resolución de conflictos**: ¿qué regla se ejecuta?
 - Lo decide de acuerdo a la **estrategia de control**
 3. **Actuación**: se ejecuta la regla (ejecutar las acciones del RHS)
 - Añadir/quitar hechos de MT y operaciones externas al sistema de reglas
 4. **Repite el ciclo**: Inicializa la agenda y vuelve al punto 1.
 - Hasta que se alcanza el objetivo
- Se puede cambiar la **estrategia de control** que resuelve conflictos cuando hay varias reglas en la agenda
- ¿Qué regla debo ejecutar de todas las de la agenda?
 - La estrategia se define al diseñar el sistema
 - Ej: **(set-strategy depth)** : selecciona la más recientemente activada

Diagrama de Funcionamiento



Base de reglas (BR)

- Conocimiento **procedimental** sobre cómo solucionar problemas
- Formato uniforme: **SI condiciones-LHS ENTONCES acciones-RHS**
 - Condiciones (**antecedente** de la regla): verificar en la MT (patrones)
 - Acciones (**consecuente** de la regla):
 - Cambios a realizar sobre la MT
 - Actividades externas al sistema (imprimir, órdenes a dispositivos)
 - **No** son reglas del tipo *if-then-else*
 - No son implicaciones lógicas sino recomendaciones imperativas
 - Los sistemas de producción permiten backtracking
- Las reglas son independientes unas de otras
 - Una regla **no** puede hacer referencia a otra (no están acopladas)
 - La única comunicación entre reglas es a través de la MT
 - Flexibilidad/modularidad versus ineficiencia
 - Una regla resuelve una parte del problema: un paso

Reglas

- Las reglas de producción tienen dos partes:
 - Izquierda (LHS): Patrón o condición que determina la aplicabilidad de la regla
 - Derecha (RHS): Descripción de la operación o acción a llevar a cabo al aplicar el operador (o disparar la regla)

- Ejemplo:

```
(defrule fallo2 "Fallo temperatura"
  (temp ?x)
  (temp-ext ?y)
  (test (>= (abs (- ?y ?x)) 10))
=>
  (printout t "El frigo ha perdido gas" crlf) )
```

} LHS (left-hand side)

} RHS (right-hand side)

Ejemplo

- Si la edad del paciente es inferior a 10 años, y tiene manchas rojas y fiebre entonces tiene varicela

SI (Paciente ?p ?edad) and (?edad < 10) and
(Síntomas ?p fiebre) and (Síntomas ?p manchas-rojas)

ENTONCES Añadir (Enfermedad ?p varicela)

Variables que tomaran valor para un paciente concreto

?p ?edad

El resto son símbolos constantes

Ejemplo

- R1: SI Forma = alargada Y Color = verde o amarillo ENTONCES Fruta = banana
- R2: SI Forma = redonda u ovalada Y Diámetro > 1.6 cm ENTONCES claseFruta = planta
- R6: SI claseFruta = planta Y Color = verde ENTONCES Fruta = sandía
- R7: SI claseFruta = planta Y Color = amarillo ENTONCES Fruta = melón

Ejemplo: ordenación de cadenas

- Sistema que ordena las letras de una cadena de $\{a, b, c\}^*$
 - Estrategia de resolución de conflicto: primera regla según el orden de definición
 - Conjunto de reglas
 - 1) $ba \rightarrow ab$
 - 2) $ca \rightarrow ac$
 - 3) $cb \rightarrow bc$

Iteración	MT	Agenda	R. disparada
0	cbaca	1(cb aca), 2(cb aca), 3(cb aca)	1(cb aca)
1	cabca	2(cab ca), 2(cab ca)	2(cab ca)
2	acbca	2(cab ca), 3(ac bca)	2(cab ca)
3	acbac	1(ac ba c), 3(ac ba c)	1(ac ba c)
4	acabc	2(ac abc)	2(ac abc)
5	aacbc	3(a ac bc)	3(a ac bc)
6	aabcc	-----	Halt

Ejemplo: palíndromos (I)

- Sistema crea cadenas de $\{a, b, c\}^*$ palíndromos (es igual si se lee de izquierda a derecha que de derecha a izquierda)
 - Conjunto de reglas
 - R1: $\$ \rightarrow a\a
 - R2: $\$ \rightarrow b\b
 - R3: $\$ \rightarrow c\c
 - Habrá que usar razonamiento progresivo (encadenamiento hacia adelante)
 - Ejemplo: Dado el símbolo inicial c podríamos generar los siguientes palíndromos:
 - Aplicamos R1 y obtenemos aca
 - Aplicamos R2 a aca y obtenemos $bacab$
 - Aplicamos R3 a $bacab$ y obtenemos $cbacabc$
 - ..

“\$” encaja con cualquier cadena

Ejemplo: palíndromos (II)

- Sistema que determina si una cadena de $\{a, b, c\}^*$ es palíndroma
 - Conjunto de reglas
 - R1: $\$ \rightarrow a\a
 - R2: $\$ \rightarrow b\b
 - R3: $\$ \rightarrow c\c
 - R4: $\text{tam}(\$) \leq 1 \rightarrow \text{Palíndromo} = \text{sí}$
 - Habrá que usar razonamiento regresivo (encadenamiento hacia atrás)
 - Ejemplo: Dada la cadena bacab se aplicarían las siguientes reglas:
 - bacab satisface la parte derecha de R2 \rightarrow En la parte izquierda nos quedaríamos con aca
 - aca satisface la parte derecha de R1 \rightarrow En la parte izquierda nos quedaríamos con c
 - c satisface la parte derecha de R4 \rightarrow es un palíndromo
 - Conclusión \rightarrow bacab sí es un palíndromo

“\$” encaja con cualquier cadena

Ejercicio 1

- Dado un sistema basado en reglas con la siguiente base de conocimiento

- R1: Si h2 y h5 entonces h1
- R2: Si h4 y h3 entonces h2
- R3: Si h6 entonces h3

donde cada h_i representa una situación o concepto y los números al lado de las reglas marcan la prioridad de ejecución de las mismas en caso de conflicto (máxima prioridad 1)

- La base de hechos inicial contiene los siguientes datos:
h6, h7, h9, h8, h4 y h5
 - a. Aplica encadenamiento hacia delante, mostrando cómo evoluciona el sistema en cada ciclo del proceso
 - b. Aplicando encadenamiento hacia atrás, determina si es posible establecer la existencia de la situación h1 a partir de la base de hechos inicial

Ejercicio 1. Solución a

Iteración	MT	Agenda	R. disparada
0	h6, h7, h9, h8, h4, h5	R3	R3
1	h6, h7, h9, h8, h4, h5, h3	R2, R3	R2
2	h6, h7, h9, h8, h4, h5, h3, h2	R1, R2, R3	R1
3	h6, h7, h9, h8, h4, h5, h3, h2, h1	R1, R2, R3	

Ejercicio 1. Solución b

- Objetivo global: $h1 \rightarrow$ No figura en la BH inicial ($h6, h7, h9, h8, h4$ y $h5$) así que hay que recurrir a las reglas en la BR en cuyo consecuente figure el concepto objetivo
- La regla R1 (Si $h2$ y $h5$ entonces $h1$) tiene $h1$ en su consecuente \rightarrow Fijamos $h2$ y $h5$ como subobjetivos
- El hecho $h5$ está en nuestra BH inicial ($h6, h7, h9, h8, h4$ y **$h5$**) así que se demuestra su validez
- La regla R2 (Si $h4$ y $h3$ entonces $h2$) tiene $h2$ en su consecuente \rightarrow Fijamos $h4$ y $h3$ como subobjetivos
- El hecho $h4$ está en nuestra BH inicial ($h6, h7, h9, h8, \mathbf{h4}$ y $h5$) así que se demuestra su validez
- La regla R3 (Si $h6$ entonces $h3$) tiene $h3$ en su consecuente \rightarrow Fijamos $h6$ como subobjetivo
- El hecho $h6$ está en nuestra BH inicial (**$h6$** , $h7, h9, h8, h4$ y $h5$) así que se demuestra su validez
- Se ha demostrado la validez de todos los subobjetivos \rightarrow Puede inferirse el concepto objetivo global
- Conclusión: $h1$ es válido aplicando las reglas R3, R2 y R1 a la BH inicial

Tipos de Razonamiento : Encadenamientos

- Encadenamiento progresivo (hacia adelante, forward chaining)
 - Dirigido por datos (antecedentes-LHS)
 - Desde los datos (hechos) conocidos busca la conclusión
 - Si los datos verifican las condiciones (LHS) de una regla,
 - Entonces la regla se puede aplicar/activar/disparar/ejecutar
- Encadenamiento regresivo (hacia atrás, backward chaining)
 - Dirigido por objetivos (consecuentes-conclusiones-RHS):
 1. Selecciona conclusiones (RHS) posibles:
De reglas cuyos RHS equiparen a la consulta
 2. Intenta probar su validez buscando evidencias (LHS) que la soporten:
Un antecedente-LHS es cierto si:
 - a) Sus hechos están en la MT del sistema
 - b) Si no, se busca si es consecuente (RHS) de alguna regla R:
Se prueban recursivamente los antecedentes de dicha regla R
 - c) Si no son aplicables a) y b), se asume la hipótesis del mundo cerrado
(o se pregunta al usuario)

Otro ejemplo: Palíndromos. Razonamiento progresivo y regresivo

- Las reglas pueden utilizarse de dos formas:

$(R1) \$ \rightarrow a\a

$(R2) \$ \rightarrow b\b

$(R3) \$ \rightarrow c\c

“\$” encaja con cualquier cadena

– Encadenamiento hacia adelante (razonamiento progresivo):

- Utilizar las reglas para generar palíndromos
- Dado un símbolo inicial como c , la secuencia de reglas $(R1)$, $(R2)$, $(R3)$, $(R2)$, $(R3)$ generará la siguiente secuencia de cadenas:

$aca \quad bacab \quad cbacabc \quad bcbacabcb \quad cbcbacabcbc$

– Encadenamiento hacia atrás (razonamiento regresivo):

- Utilizar las reglas para reconocer palíndromos
- Dado un palíndromo como $bacab$, podemos trazar la secuencia de reglas que llevan a su construcción
- $bacab$ satisface la parte derecha de $(R2)$, cuya parte izquierda da aca
- aca satisface la parte derecha de $(R1)$, cuya parte izquierda da c
 - Respuesta : Sí, es un palíndromo

Ejercicio de encadenamiento de reglas

- Dado un sistema basado en reglas con la siguiente base de conocimiento
 - R1: **Si** h_2 y h_5 **entonces** h_1
 - R2: **Si** h_4 y h_3 **entonces** h_2
 - R3: **Si** h_6 **entonces** h_3

donde cada h_i representa una situación o concepto y los números al lado de las reglas marcan la prioridad de ejecución de las mismas en caso de conflicto (máxima prioridad 1)
- La base de hechos inicial contiene los siguientes datos:
 h_6, h_7, h_9, h_8, h_4 y h_5
 - a) Aplica encadenamiento hacia delante, mostrando cómo evoluciona el sistema en cada ciclo del proceso
 - b) Aplicando encadenamiento hacia atrás, determina si es posible establecer la existencia de la situación h_1 a partir de la base de hechos inicial

a) Solución ejercicio: encadenamiento hacia adelante

- Seleccionar aquellas reglas cuyo antecedente se cumple a partir de la base de hechos actual. Si hay más de una regla seleccionada, emplear una estrategia de resolución de conflictos para seleccionar una.
- Ejecutar la regla resultante del paso anterior

Explicación

Inicialmente se tendría BH0, de las tres reglas existentes en la base de conocimientos sólo la 3 se selecciona.

Tras su ejecución, se obtiene BH1. Ahora existirá un conflicto entre las reglas 2 y 3.

2 es más prioritaria y 3 ya se ha aplicado. Se ejecuta 2 obteniendo BH2.

En esta situación todas las reglas forman parte de la agenda. Finalmente, es la regla 1 la que se ejecuta, quedando BH3.

BH ₀	BH ₁	BH ₂	BH ₃
h_6	h_6	h_6	h_6
h_7	h_7	h_7	h_7
h_9	h_9	h_9	h_9
h_8	h_8	h_8	h_8
h_4	h_4	h_4	h_4
h_5	h_5	h_5	h_5
	h_3	h_3	h_3
		h_2	h_2
			h_1

b) Solución ejercicio: encadenamiento hacia atrás

- Se parte de un concepto objetivo a verificar a partir de la BR y la BH.
- Comprobar si dicho concepto pertenecía ya o no a la BH. En caso negativo recurrir a las reglas en la BR en cuyo consecuente figure el concepto objetivo. Así los conceptos del antecedente de dichas reglas pasan a ser subobjetivos. Si se demuestra la validez de estos subobjetivos, podrá inferirse el concepto objetivo global.
- En el ejemplo el objetivo global es $h1$, que no figura en la BH inicial;
- Ir a la BC. La regla 1 tiene $h1$ en su consecuente. Ello nos permite fijar $h2$ y $h5$ como subobjetivos. Al encontrarse $h5$ en la BH inicial, sólo quedaría por demostrar $h2$.
- La regla 2, con $h2$ en su consecuente, establece $h4$ y $h3$ como nuevos subobjetivos. El hecho $h4$ está en la BH inicial. Habrá que verificar, la existencia de $h3$.
- La regla 3 fija como subobjetivo $h6$, que está en la BH inicial, por tanto **final**.
- Entonces $h1$ es válido aplicando las reglas 3, 2 y 1 a la BH inicial.

Criterios de elección de dirección del encadenamiento

- Dado el problema, ir de menos a más:
 - Complejidad del LHS y del RHS de las reglas
 - Empezar por el menor número de hechos (inicial u objetivo)
 - Aspecto del espacio de estados
 - Naturaleza y disponibilidad de los datos del problema
 - Factor de ramificación en ambas direcciones
 - Atrás: Si hay muchos datos iniciales (hechos) y/o pocas posibles conclusiones
 - Adelante: Si hay pocos datos iniciales y/o muchas posibles conclusiones
- Si el proceso de resolución es interactivo
 - Utilizar el mismo que el experto usa de forma natural
- Sistemas de contestación de preguntas
 - Utilizar hacia atrás (*p.ej. sistemas de diagnóstico*)
- Sistemas que reaccionan a datos que llegan periódicamente
 - Utilizar hacia delante
 - Ej: datos de sensores recibidos cada minuto en una central térmica

Ejemplo: Identificación de frutas (I)

- R1: SI Forma = alargada Y Color = verde o amarillo ENTONCES Fruta = banana
- R2: SI Forma = redonda u ovalada Y Diámetro > 1.6 cm ENTONCES claseFruta = planta
- R3: SI Forma = redonda Y Diámetro < 1.6 cm ENTONCES claseFruta = árbol
- R4: SI numSemillas = 1 ENTONCES claseSemilla = hueso
- R5: SI numSemillas > 1 ENTONCES claseSemilla = múltiple
- R6: SI claseFruta = planta Y Color = verde ENTONCES Fruta = sandía
- R7: SI claseFruta = planta Y Color = amarillo ENTONCES Fruta = melón
- R8: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = albaricoque
- R9: SI claseFruta = árbol Y Color = naranja Y claseSemilla = múltiple ENTONCES Fruta = naranja
- R10: SI claseFruta = árbol Y Color = rojo Y claseSemilla = hueso ENTONCES Fruta = cereza
- R11: SI claseFruta=árbol Y Color=naranja Y claseSemilla=hueso ENTONCES Fruta=melocotón
- R12: SI claseFruta=árbol Y Color=rojo o amarillo o verde Y claseSemilla=múltiple ENTONCES
Fruta=manzana
- R13: SI claseFruta = árbol Y Color = morado Y claseSemilla = hueso ENTONCES Fruta = ciruela

Resolución de conflictos: Se desactivan las reglas que no añadan hechos nuevos y se selecciona la primera regla aplicable (la de menor número)

Ejemplo: Identificación de frutas (II)

- Hechos iniciales en la memoria de trabajo:
 - Diametro = 0.4 cm
 - Forma = redonda
 - NumSemillas = 1
 - Color = rojo
- Pasos encadenamiento hacia delante:
 - 1) **Reconocimiento, equiparación o encaje**: encuentra reglas aplicables y las activa
 - 2) **Resolución de conflictos**: desactiva reglas que no añadan hechos nuevos, y selecciona la **primera regla aplicable** (*la de menor número en este ej.*)
 - 3) **Actuación**: ejecuta la acción de la regla (la dispara). Si no hay, se detiene
 - 4) **Reset**: vacía la agenda (desactiva reglas aplicables) y vuelve al paso **1)**

Ejemplo: Identificación de frutas (II)

Ciclo	Reglas aplicables	Regla seleccionada	Hecho derivado
1	3, 4	3	claseFruta = árbol

Hechos en la MT

Diametro = 0.4 cm
Forma = redonda
NumSemillas = 1
Color = rojo



Diámetro = 0.4 cm,
Forma = redonda,
Numsemillas = 1,
Color = rojo,
claseFruta = árbol

Ejemplo: Identificación de frutas (II)

Ciclo	Reglas aplicables	Regla seleccionada	Hecho derivado
1	3, 4	3	claseFruta = árbol
2	3 , 4	4	claseSemilla = hueso

Desactivada por el principio de refracción



Ejemplo: Identificación de frutas (II)

Ciclo	Reglas aplicables	Regla seleccionada	Hecho derivado
1	3, 4	3	claseFruta = árbol
2	3, 4	4	claseSemilla = hueso
3	3, 4, 10	10	Fruta = cereza

Ejemplo: Identificación de frutas (II)

Ciclo	Reglas aplicables	Regla seleccionada	Hecho derivado
1	3, 4	3	claseFruta = árbol
2	3 , 4	4	claseSemilla = hueso
3	3 , 4, 10	10	Fruta = cereza
4	3 , 4, 10		

Ya no hay más reglas en la agenda → FIN
Conclusión → La fruta es una cereza

Reglas de identificación de frutas

- R1: SI Forma = alargada Y Color = verde o amarillo ENTONCES Fruta = banana
- R2: SI Forma = redonda u ovalada Y Diámetro > 1.6 cm ENTONCES claseFruta = planta
- R3: SI Forma = redonda Y Diámetro < 1.6 cm ENTONCES claseFruta = árbol
- R4: SI numSemillas = 1 ENTONCES claseSemilla = hueso
- R5: SI numSemillas > 1 ENTONCES claseSemilla = múltiple
- R6: SI claseFruta = planta Y Color = verde ENTONCES Fruta = sandía
- R7: SI claseFruta = planta Y Color = amarillo ENTONCES Fruta = melón
- R8: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = albaricoque
- R9: SI claseFruta = árbol Y Color = naranja Y claseSemilla = múltiple ENTONCES Fruta = naranja
- R10: SI claseFruta = árbol Y Color = rojo Y claseSemilla = hueso ENTONCES Fruta = cereza
- R11: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = melocotón
- R12: SI claseFruta = árbol Y Color = rojo o amarillo o verde Y claseSemilla = múltiple ENTONCES Fruta = manzana
- R13: SI claseFruta = árbol Y Color = morado Y claseSemilla = hueso ENTONCES Fruta = ciruela



Pasos de un sistema con encadenamiento regresivo (hacia atrás)

- 1) Dado el objetivo a demostrar, se comprueba si forma parte de la BH. En ese caso finaliza su demostración.
- 2) En caso contrario, se buscan todas las reglas que tienen el objetivo como consecuente. Cada una de ellas constituye una alternativa para demostrar el objetivo (nodo O).
- 3) Seleccionar una de las reglas obtenidas en 2) utilizando una estrategia de resolución de conflictos.
- 4) Obtener las premisas que constituyen el antecedente de la regla seleccionada en 3). Todas esas premisas deben ser demostradas y se convierten en nuevos subobjetivos (nodo Y).
- 5) Para cada subobjetivo a demostrar se aplica recursivamente el mismo procedimiento, comenzando por el paso 1.
- 6) Cuando falla la demostración de un descendiente de un nodo Y, se puede abandonar la demostración del nodo Y. Cuando falla la demostración de un descendiente de un nodo O, se prueba con la siguiente rama alternativa.

➔ Búsqueda en profundidad en un árbol Y/O

- Si no se utiliza la hipótesis de mundo cerrado, cuando no es posible demostrar un subobjetivo, se puede preguntar al usuario el valor del correspondiente parámetro.

Ejemplo de encadenamiento regresivo

- Traza de ejecución para derivar cereza como valor de fruta
 - Objetivos: (Fruta) Base de hechos inicial: vacía
 - Reglas que pueden satisfacer el objetivo Fruta: 1, 6 - 13
 - Examinamos premisas:
 - Regla 1: La 1ª premisa (Forma= alargada) no se encuentra en la memoria de trabajo. No hay reglas que deriven este valor → se pregunta al usuario
 - ¿Cuál es el valor de Forma? redonda
- Memoria de trabajo: ((Forma redonda))- La regla 1 falla
- Regla 6: La 1ª premisa (claseFruta = planta) no se encuentra en la MT. Las reglas 2 y 3 pueden derivar el valor claseFruta
- Objetivos: (claseFruta, Fruta)
- Reglas que pueden satisfacer el objetivo claseFruta: 2, 3
- Examinamos premisas

Ejemplo: reglas de identificación de frutas

Objetivo = Fruta

✗ R1: SI Forma = alargada Y Color = verde o amarillo ENTONCES Fruta = banana

R2: SI Forma = redonda u ovalada Y Diámetro > 1.6 cm ENTONCES claseFruta = planta

R3: SI Forma = redonda Y Diámetro < 1.6 cm ENTONCES claseFruta = árbol

R4: SI numSemillas = 1 ENTONCES claseSemilla = hueso

R5: SI numSemillas > 1 ENTONCES claseSemilla = múltiple

R6: SI claseFruta = planta Y Color = verde ENTONCES Fruta = sandía

R7: SI claseFruta = planta Y Color = amarillo ENTONCES Fruta = melón

R8: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = albaricoque

R9: SI claseFruta = árbol Y Color = naranja Y claseSemilla = múltiple ENTONCES Fruta = naranja

R10: SI claseFruta = árbol Y Color = rojo Y claseSemilla = hueso ENTONCES Fruta = cereza

R11: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta =
melocotón

R12: SI claseFruta = árbol Y Color = rojo o amarillo o verde Y claseSemilla = múltiple
ENTONCES Fruta = manzana

R13: SI claseFruta = árbol Y Color = morado Y claseSemilla = hueso ENTONCES Fruta = ciruela

Objetivos: (Fruta)
Hechos: Forma = redonda



Ejemplo: reglas de identificación de frutas

Objetivo = claseFruta

- ✗ R1: SI Forma = alargada Y Color = verde o amarillo ENTONCES Fruta = banana
- R2: SI Forma = redonda u ovalada Y Diámetro > 1.6 cm ENTONCES claseFruta = planta
- R3: SI Forma = redonda Y Diámetro < 1.6 cm ENTONCES claseFruta = árbol
- R4: SI numSemillas = 1 ENTONCES claseSemilla = hueso
- R5: SI numSemillas > 1 ENTONCES claseSemilla = múltiple
- R6: SI claseFruta = planta Y Color = verde ENTONCES Fruta = sandía
- R7: SI claseFruta = planta Y Color = amarillo ENTONCES Fruta = melón
- R8: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = albaricoque
- R9: SI claseFruta = árbol Y Color = naranja Y claseSemilla = múltiple ENTONCES Fruta = naranja
- R10: SI claseFruta = árbol Y Color = rojo Y claseSemilla = hueso ENTONCES Fruta = cereza
- R11: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = melocotón
- R12: SI claseFruta = árbol Y Color = rojo o amarillo o verde Y claseSemilla = múltiple ENTONCES Fruta = manzana
- R13: SI claseFruta = árbol Y Color = morado Y claseSemilla = hueso ENTONCES Fruta = ciruela

Objetivos: (claseFruta, Fruta)
Hechos: Forma = redonda

Ejemplo de encadenamiento regresivo

- Regla 2: La 1ª premisa (Forma = redonda u ovalada) es satisfecha puesto que el valor de Forma es redonda. Se continúa con la siguiente premisa (Diámetro > 1.6 cm). Puesto que no existe un valor de Diámetro ni se puede derivar de otras reglas → preguntar al usuario

– ¿Cuál es el valor de Diámetro? 0.4

Memoria de trabajo: ((Forma redonda) (Diámetro 0.4))

La regla 2 falla

- Regla 3: Las premisas se cumplen: (Forma = redonda), (Diámetro < 1.6 cm)
→ Se deriva (claseFruta = árbol)

Memoria de trabajo: ((Forma redonda) (Diámetro 0.4) (claseFruta árbol))

Objetivos: (Fruta)

– Volvemos a la regla 6

- Regla 6: La 1ª premisa (claseFruta = planta) no se cumple. Falla
- Regla 7: Falla por el mismo motivo

Ejemplo: reglas de identificación de frutas

Objetivo = Fruta

- ✗ R1: SI Forma = alargada Y Color = verde o amarillo ENTONCES Fruta = banana
- ✗ R2: SI Forma = redonda u ovalada Y Diámetro > 1.6 cm ENTONCES claseFruta = planta
- R3: SI Forma = redonda Y Diámetro < 1.6 cm ENTONCES claseFruta = árbol
- R4: SI numSemillas = 1 ENTONCES claseSemilla = hueso
- R5: SI numSemillas > 1 ENTONCES claseSemilla = múltiple
- ✗ R6: SI claseFruta = planta Y Color = verde ENTONCES Fruta = sandía
- ✗ R7: SI claseFruta = planta Y Color = amarillo ENTONCES Fruta = melón
- R8: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = albaricoque
- R9: SI claseFruta = árbol Y Color = naranja Y claseSemilla = múltiple ENTONCES Fruta = naranja
- R10: SI claseFruta = árbol Y Color = rojo Y claseSemilla = hueso ENTONCES Fruta = cereza
- R11: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = melocotón
- R12: SI claseFruta = árbol Y Color = rojo o amarillo o verde Y claseSemilla = múltiple ENTONCES Fruta = manzana
- R13: SI claseFruta = árbol Y Color = morado Y claseSemilla = hueso ENTONCES Fruta = ciruela

Objetivos: (Fruta)
Hechos: Forma = redonda,
Diámetro = 0.4
claseFruta = árbol

Ejemplo de encadenamiento regresivo

- Regla 8: La 1ª premisa (claseFruta = árbol) es satisfecha. Se continúa con la siguiente premisa (Color = naranja). Puesto que no existe un valor de color ni se puede derivar de otras reglas → preguntar al usuario
 - ¿Cuál es el valor de Color? rojo

Memoria de trabajo:

((Forma redonda) (Diámetro 0.4) (claseFruta árbol) (Color rojo))

Falla la regla 8

- Regla 9: Falla por el mismo motivo
 - Regla 10: Se cumplen las dos primeras premisas (claseFruta = árbol), (Color = rojo). Se continúa con la siguiente premisa (claseSemilla = hueso). No existe un valor para claseSemilla pero se puede derivar de las reglas 4 y 5
- Objetivos: (claseSemilla, Fruta)
- Reglas que pueden satisfacer el objetivo claseSemilla: 4, 5
 - Examinamos premisas

Objetivo: claseSemilla

- ✗ R1: *SI Forma = alargada Y Color = verde o amarillo ENTONCES Fruta = banana*
- ✗ R2: *SI Forma = redonda u ovalada Y Diámetro > 1.6 cm ENTONCES claseFruta = planta*
- R3: *SI Forma = redonda Y Diámetro < 1.6 cm ENTONCES claseFruta = árbol*
- R4: *SI numSemillas = 1 ENTONCES claseSemilla = hueso*
- R5: *SI numSemillas > 1 ENTONCES claseSemilla = múltiple*
- ✗ R6: *SI claseFruta = planta Y Color = verde ENTONCES Fruta = sandía*
- ✗ R7: *SI claseFruta = planta Y Color = amarillo ENTONCES Fruta = melón*
- ✗ R8: *SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = albaricoque*
- ✗ R9: *SI claseFruta = árbol Y Color = naranja Y claseSemilla = múltiple ENTONCES Fruta = naranja*
- R10: *SI claseFruta = árbol Y Color = rojo Y claseSemilla = hueso ENTONCES Fruta = cereza*
- R11: *SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = melocotón*
- R12: *SI claseFruta = árbol Y Color = rojo o amarillo o verde Y claseSemilla = múltiple ENTONCES Fruta = manzana*
- R13: *SI claseFruta = árbol Y Color = morado Y claseSemilla = hueso ENTONCES Fruta = ciruela*

Objetivos: (claseSemilla, Fruta)
Hechos: Forma = redonda,
Diámetro = 0.4
claseFruta = árbol
Color = rojo

Ejemplo de encadenamiento regresivo

- Regla 4: La premisa (numSemillas = 1) no se encuentra en la memoria de trabajo. No hay reglas que deriven este valor → preguntar al usuario
 - ¿Cuál es el valor de NumSemilla? 1

Memoria de trabajo:

((Forma redonda) (Diámetro 0.4) (claseFruta árbol) (Color rojo) (numSemilla 1))

La premisa se cumple → Se deriva (claseSemilla = hueso)

Memoria de trabajo:

((Forma redonda) (Diámetro 0.4) (claseFruta árbol) (Color rojo) (numSemilla 1)
(claseSemilla = hueso))

Objetivos: (Fruta)



– Volvemos a la regla 10

- Regla 10: La premisa se cumple → Se deriva (Fruta = cereza)

Memoria de trabajo:

((Forma redonda) (Diámetro 0.4) (claseFruta árbol) (Color rojo) (numSemilla 1)
(claseSemilla hueso) (Fruta cereza))

Objetivos: ()



– El sistema se detiene

Ejemplo: reglas de identificación de frutas

Objetivo = Fruta

- ✗ R1: SI Forma = alargada Y Color = verde o amarillo ENTONCES Fruta = banana
- ✗ R2: SI Forma = redonda u ovalada Y Diámetro > 1.6 cm ENTONCES claseFruta = planta
- R3: SI Forma = redonda Y Diámetro < 1.6 cm ENTONCES claseFruta = árbol
- R4: SI numSemillas = 1 ENTONCES claseSemilla = hueso
- R5: SI numSemillas > 1 ENTONCES claseSemilla = múltiple
- ✗ R6: SI claseFruta = planta Y Color = verde ENTONCES Fruta = sandía
- ✗ R7: SI claseFruta = planta Y Color = amarillo ENTONCES Fruta = melón
- ✗ R8: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = albaricoque
- ✗ R9: SI claseFruta = árbol Y Color = naranja Y claseSemilla = múltiple ENTONCES Fruta = naranja
- R10: SI claseFruta = árbol Y Color = rojo Y claseSemilla = hueso ENTONCES Fruta = cereza
- R11: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = melocotón
- R12: SI claseFruta = árbol Y Color = rojo o amarillo o verde Y claseSemilla = múltiple ENTONCES Fruta = manzana
- R13: SI claseFruta = árbol Y Color = morado Y claseSemilla = hueso ENTONCES Fruta = ciruela

Objetivos: (Fruta)

Hechos:

numSemilla = 1

claseSemilla = hueso

Ejemplo: reglas de identificación de frutas

No quedan objetivos: fin

- ✗ R1: SI Forma = alargada Y Color = verde o amarillo ENTONCES Fruta = banana
- ✗ R2: SI Forma = redonda u ovalada Y Diámetro > 1.6 cm ENTONCES claseFruta = planta
- R3: SI Forma = redonda Y Diámetro < 1.6 cm ENTONCES claseFruta = árbol
- R4: SI numSemillas = 1 ENTONCES claseSemilla = hueso
- R5: SI numSemillas > 1 ENTONCES claseSemilla = múltiple
- ✗ R6: SI claseFruta = planta Y Color = verde ENTONCES Fruta = sandía
- ✗ R7: SI claseFruta = planta Y Color = amarillo ENTONCES Fruta = melón
- ✗ R8: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = albaricoque
- ✗ R9: SI claseFruta = árbol Y Color = naranja Y claseSemilla = múltiple ENTONCES Fruta = naranja
- R10: SI claseFruta = árbol Y Color = rojo Y claseSemilla = hueso ENTONCES Fruta = cereza
- R11: SI claseFruta = árbol Y Color = naranja Y claseSemilla = hueso ENTONCES Fruta = melocotón
- R12: SI claseFruta = árbol Y Color = rojo o amarillo o verde Y claseSemilla = múltiple ENTONCES Fruta = manzana
- R13: SI claseFruta = árbol Y Color = morado Y claseSemilla = hueso ENTONCES Fruta = ciruela

Objetivos: ()

Hechos:

claseFruta = árbol

Color = rojo

claseSemilla = hueso

Fruta = cereza

Implementación: fase de reconocimiento

- Encaje, ajuste o equiparación (*matching*) de patrones
 - Qué reglas son aplicables en un cierto estado
- La situación es distinta dependiendo de si el enfoque es con encadenamiento hacia delante o hacia atrás
 - Encadenamiento hacia atrás → parte derecha de las reglas
 - Encadenamiento hacia adelante → parte izquierda de las reglas
- Existencia de variables → encaje con ligadura de variables
- Los mecanismos de encadenamiento utilizan diversas estrategias para agilizar el reconocimiento de reglas
 - Se pueden indexar las reglas para mejorar la eficiencia
 - Con encadenamiento hacia delante se indexan por el antecedente
 - Con encadenamiento hacia atrás se indexan por el consecuente
 - *Tablas hash*
 - Algoritmo de encaje más eficiente que el encaje uno a uno : RETE (REdundancia TEmporal)

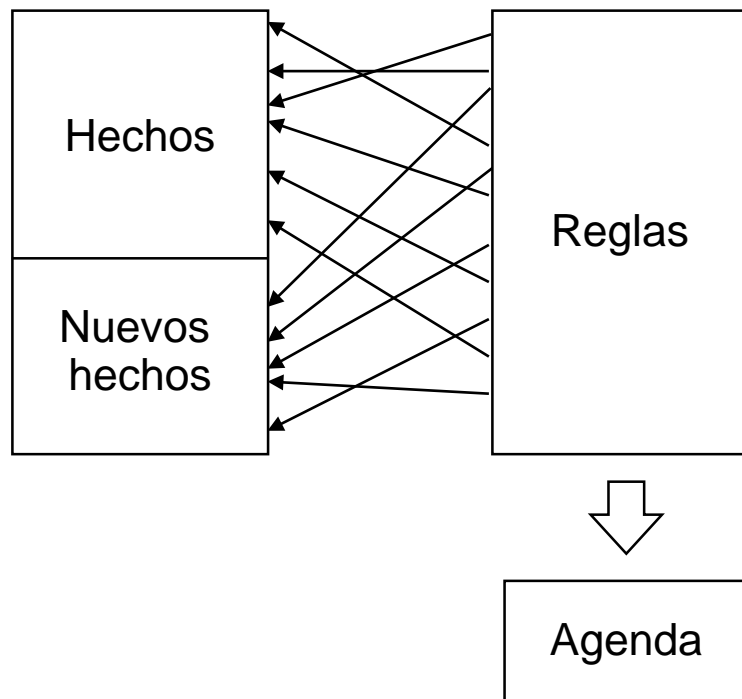
Implementación: fase de reconocimiento

- El encaje uno a uno es muy ineficiente
- En cada ciclo los cambios producidos en la base de hechos son los únicos que influyen en la modificación del conjunto conflicto (*redundancia temporal*)
 - Los nuevos hechos que son introducidos en la memoria de trabajo son los que determinan las nuevas reglas aplicables
 - Idea del algoritmo RETE: en lugar de buscar qué reglas satisfacen los hechos existentes en cada momento, son los nuevos hechos generados en cada ciclo los que buscan o determinan qué nuevas reglas se seleccionan

Implementación: fase de reconocimiento

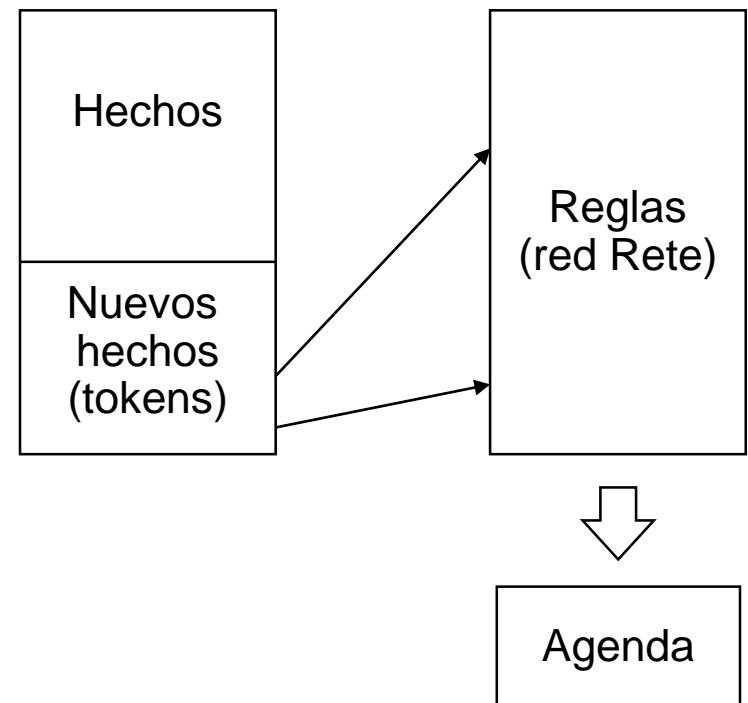
● Método tradicional

- Cada regla verifica si se cumplen sus condiciones a partir de la base de hechos total



● Algoritmo RETE

- Los nuevos hechos buscan reglas



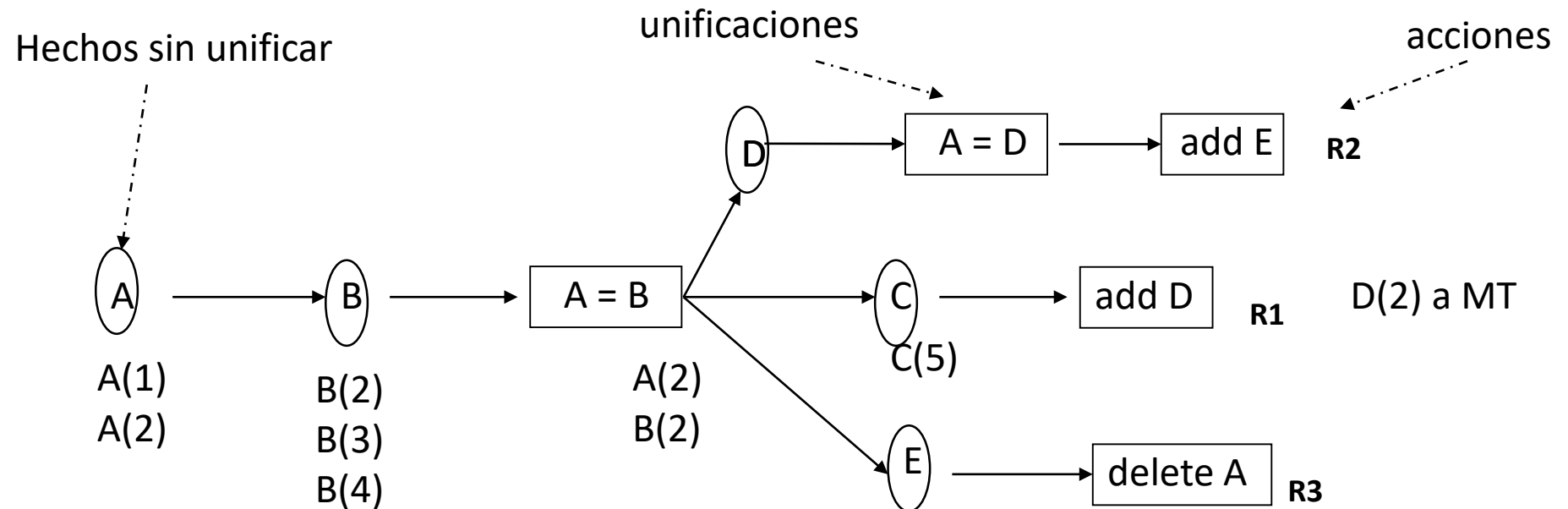
Algoritmo RETE (Forgy, 1982)

- Realiza un encaje de patrones eficiente (muchos a muchos)
 - entre la base de reglas y la base de hechos
- Evita reevaluar condiciones ya evaluadas
- Aprovecha la **redundancia temporal**:
 - Los posibles cambios en la agenda entre ciclo son pocos
- Aprovecha la **similitud estructural**:
 - Muchas reglas comparten condiciones
- Construye un grafo de dependencias estableciendo las reglas que comparten condiciones
 - En un ciclo de operación, las nuevas reglas que se añaden a la agenda son aquéllas que dependían de condiciones que acababan de hacerse ciertas al aplicar la regla anterior

Ejemplo de RETE

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5)\}$

--- Añadir regla a Red --
Si existe nodo de premisa:
conectar
Si no: crearlo y conectar



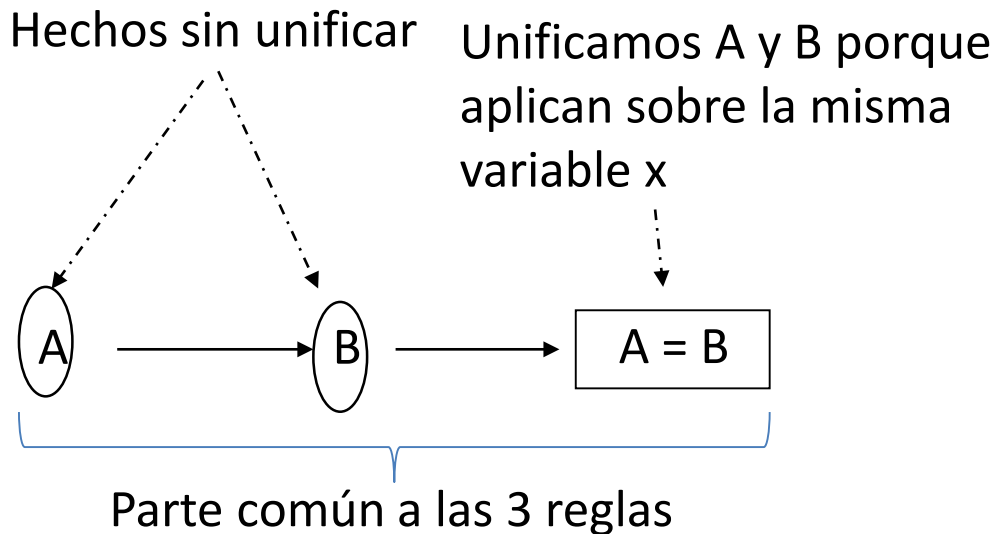
Qué pasa si añado R4 - $A(x), D(x) \Rightarrow \text{add } M(x)$

Algoritmo RETE. Ejemplo

CICLO 1

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5)\}$

Creamos la red con las reglas:

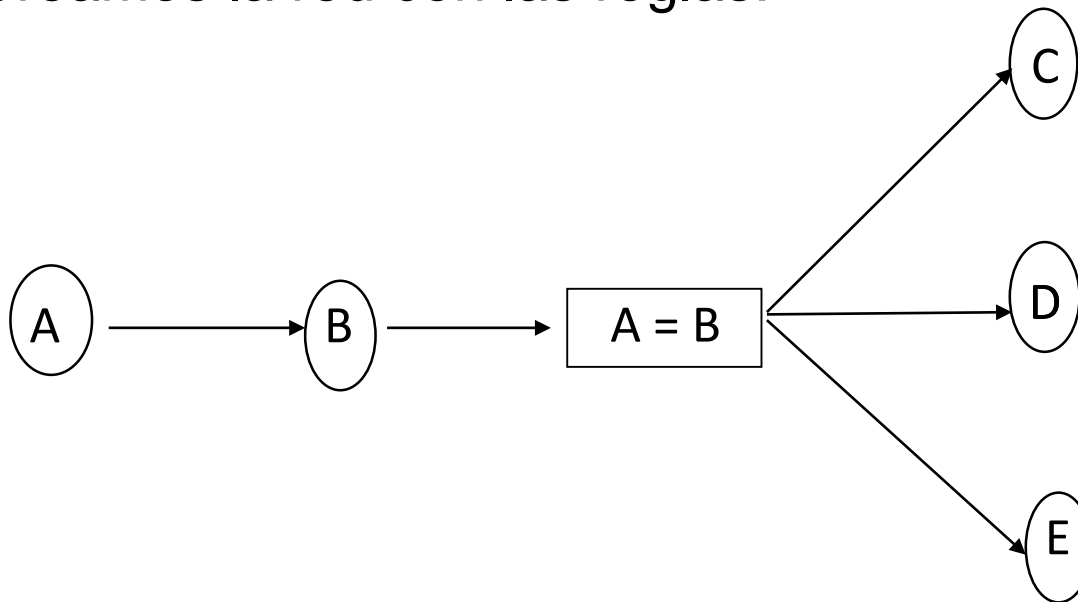


Algoritmo RETE. Ejemplo

CICLO 1

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5)\}$

Creamos la red con las reglas:

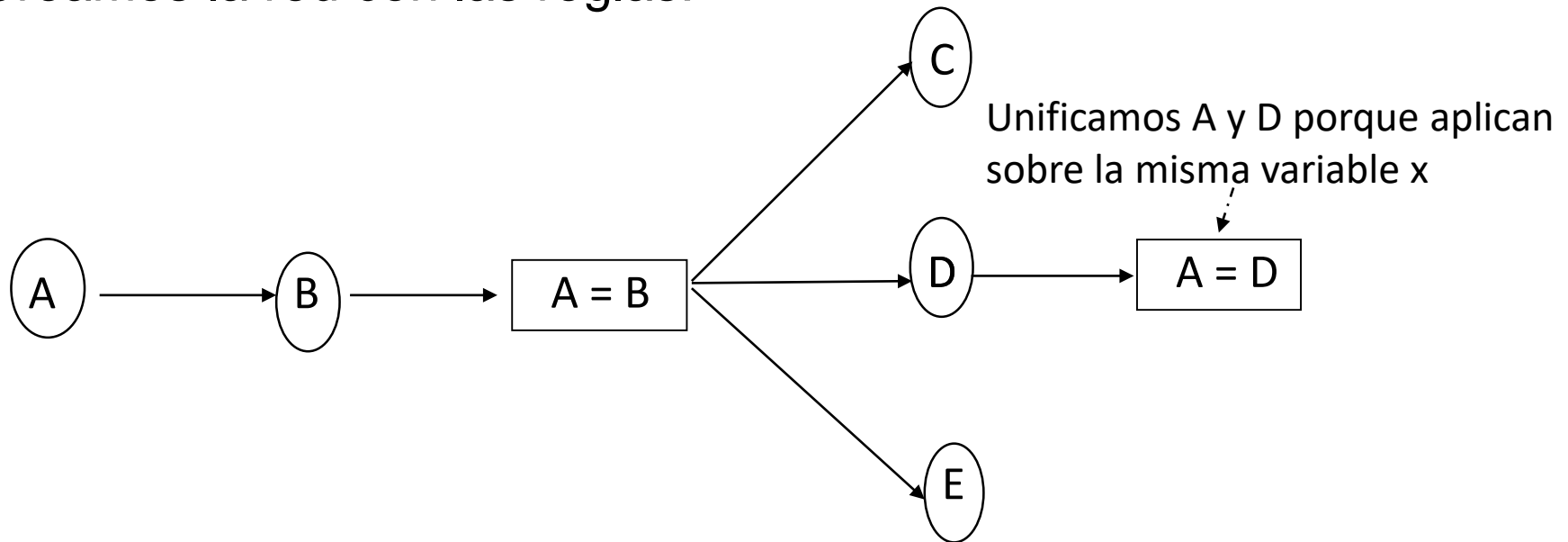


Algoritmo RETE. Ejemplo

CICLO 1

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5)\}$

Creamos la red con las reglas:

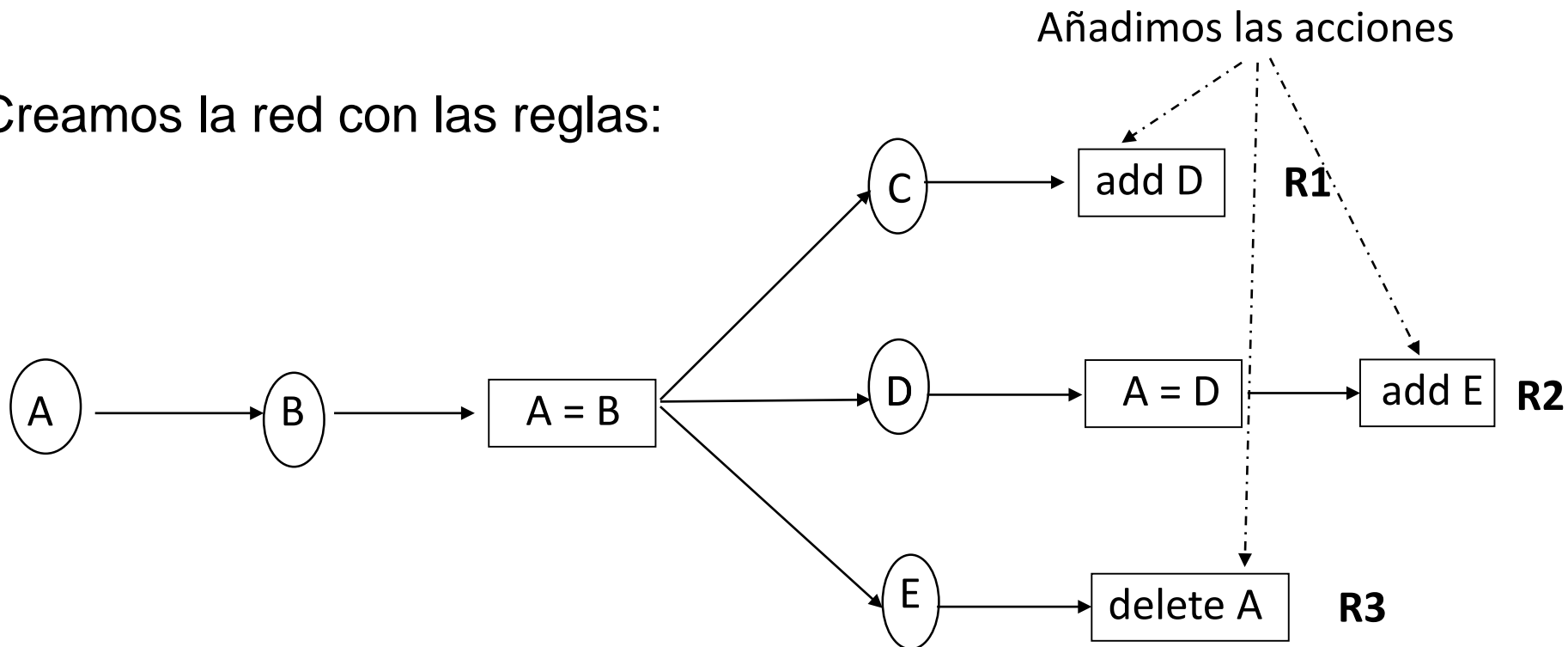


Algoritmo RETE. Ejemplo

CICLO 1

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5)\}$

Creamos la red con las reglas:

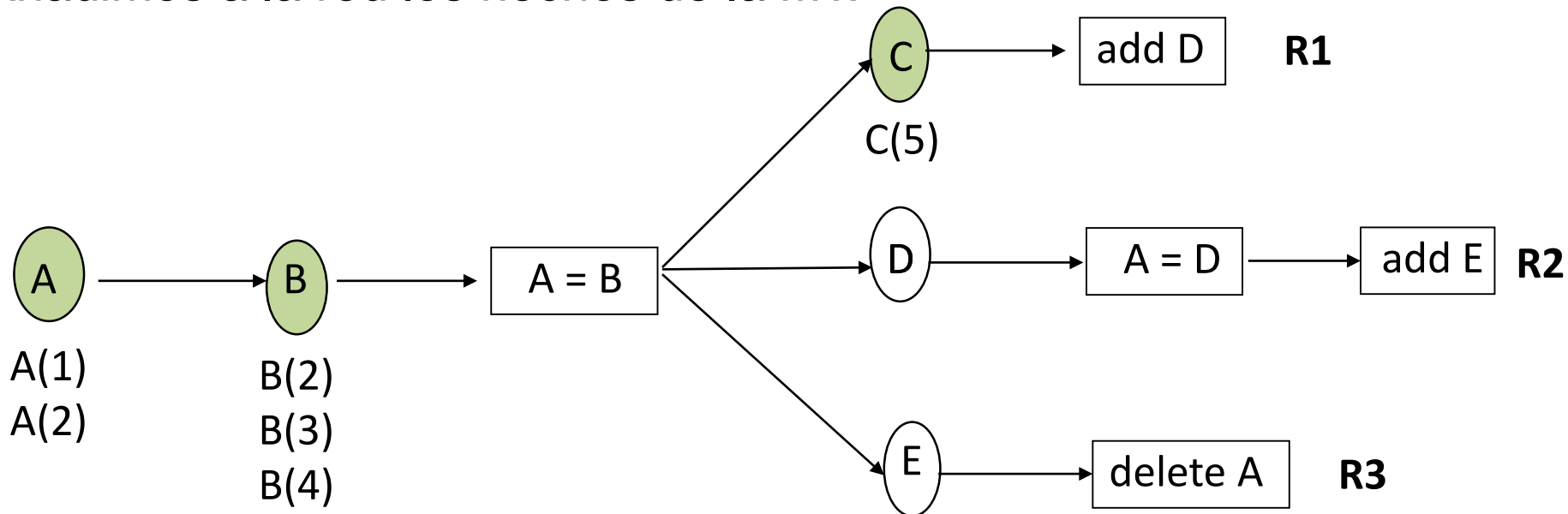


Algoritmo RETE. Ejemplo

CICLO 1

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5)\}$

Añadimos a la red los hechos de la MT:

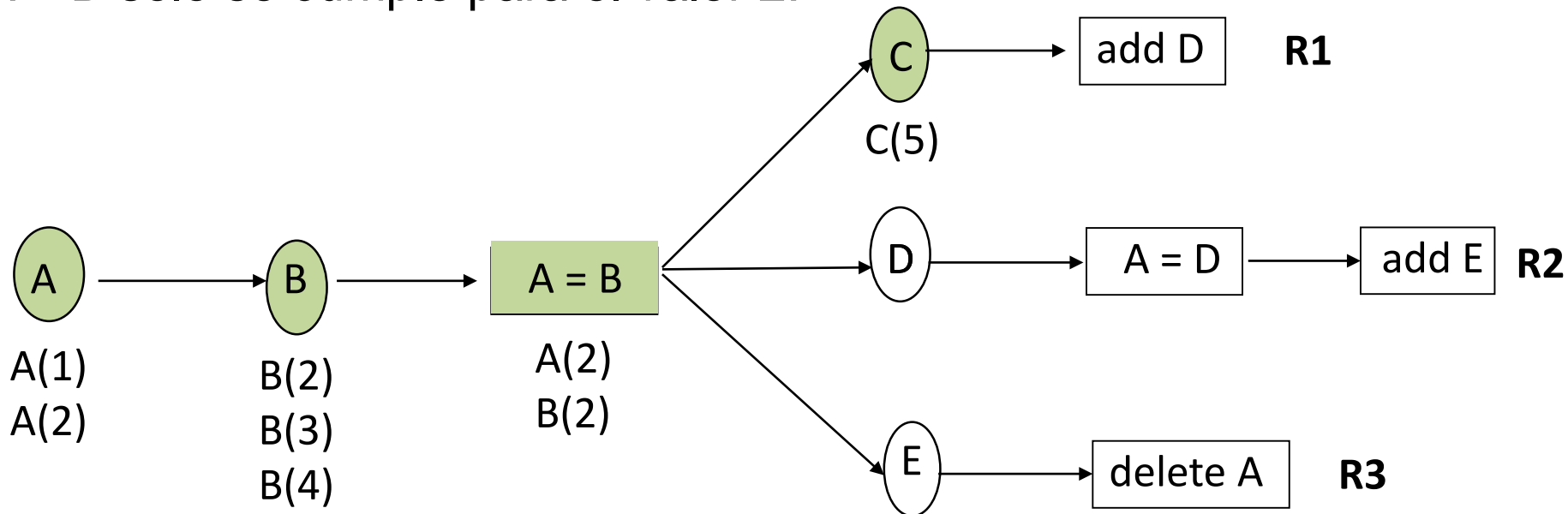


Algoritmo RETE. Ejemplo

CICLO 1

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5)\}$

A = B solo se cumple para el valor 2:

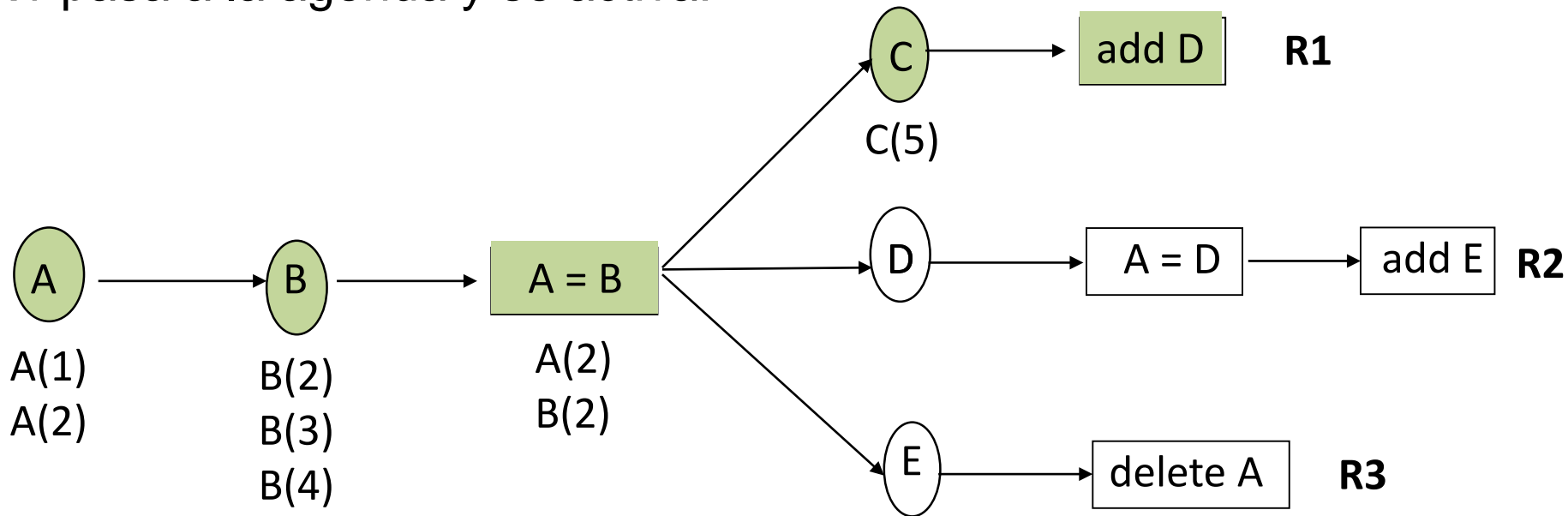


Algoritmo RETE. Ejemplo

CICLO 1

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5)\}$

R1 pasa a la agenda y se activa:

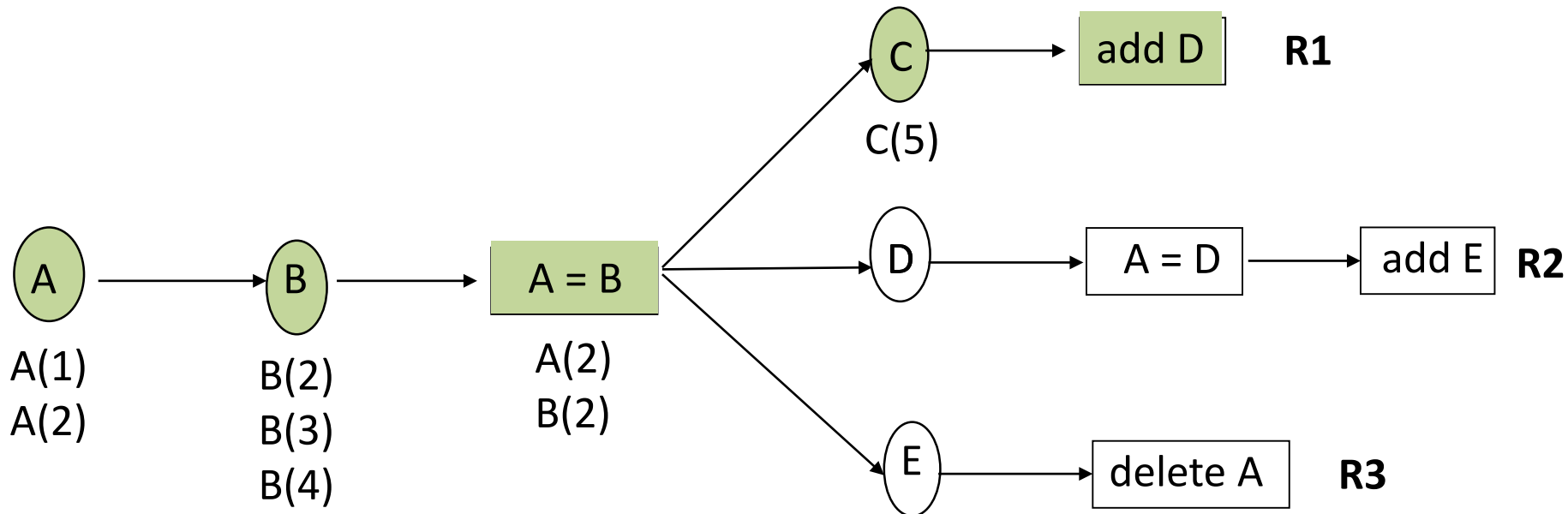


Algoritmo RETE. Ejemplo

CICLO 1

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5), \mathbf{D(2)}\}$

D(2) pasa a la MT como resultado de la ejecución de R1

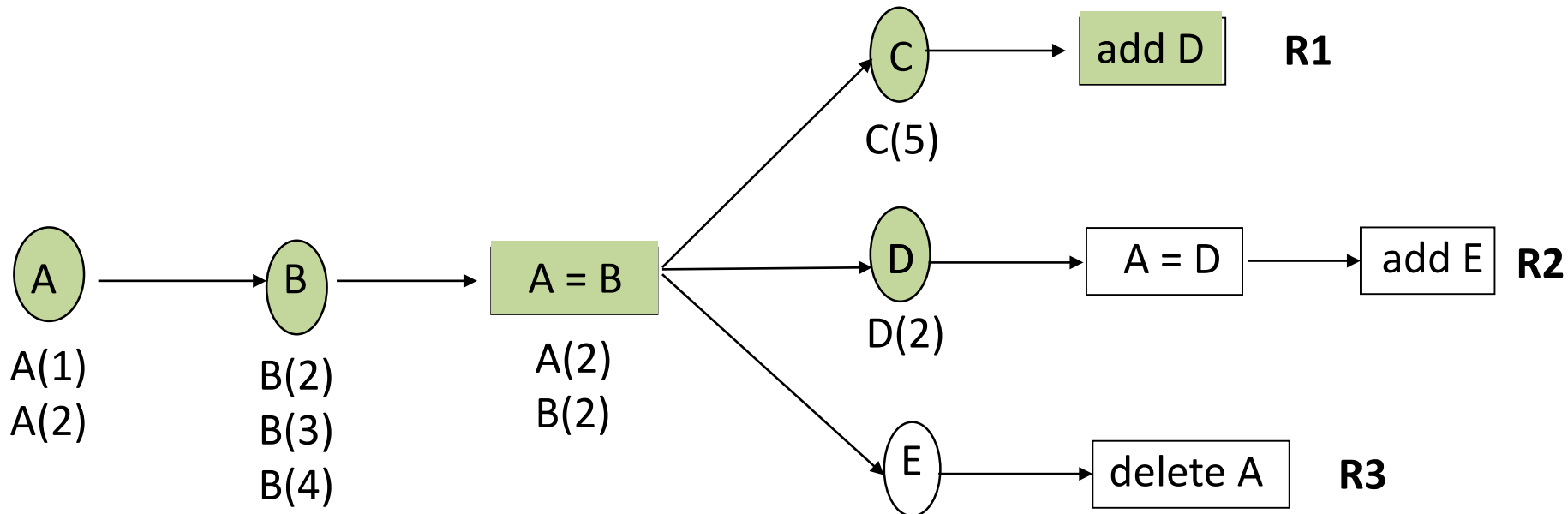


Algoritmo RETE. Ejemplo

CICLO 2

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5), \mathbf{D(2)}\}$

Se añaden los nuevos hechos de la MT ($D(2)$) a la red:

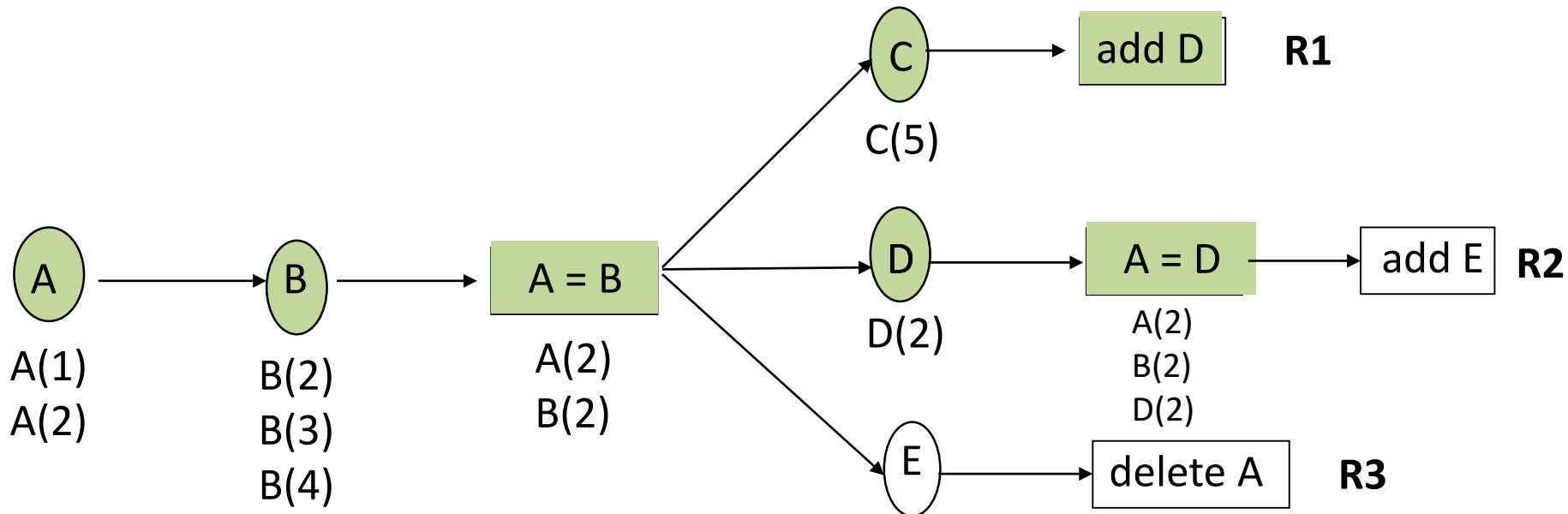


Algoritmo RETE. Ejemplo

CICLO 2

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5), D(2)\}$

Se activa el nodo $A=D$ porque tenemos el hecho $A(2)$ y $D(2)$ en la MT:

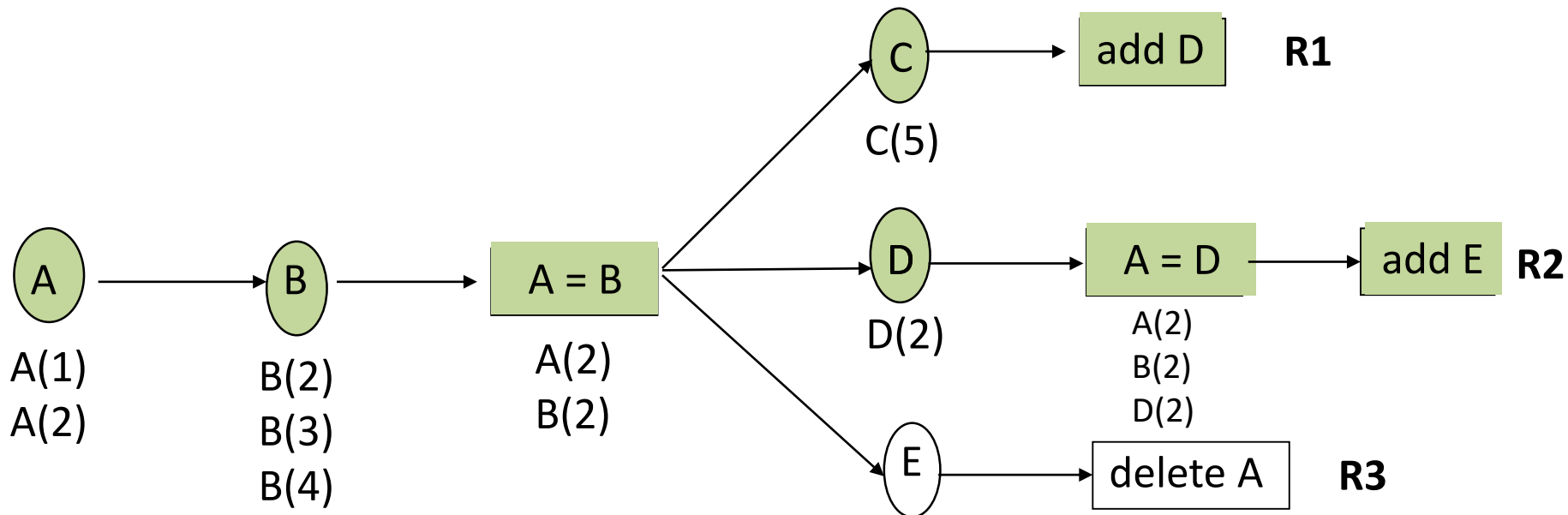


Algoritmo RETE. Ejemplo

CICLO 2

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5), D(2)\}$

R2 pasa a la agenda y se activa:

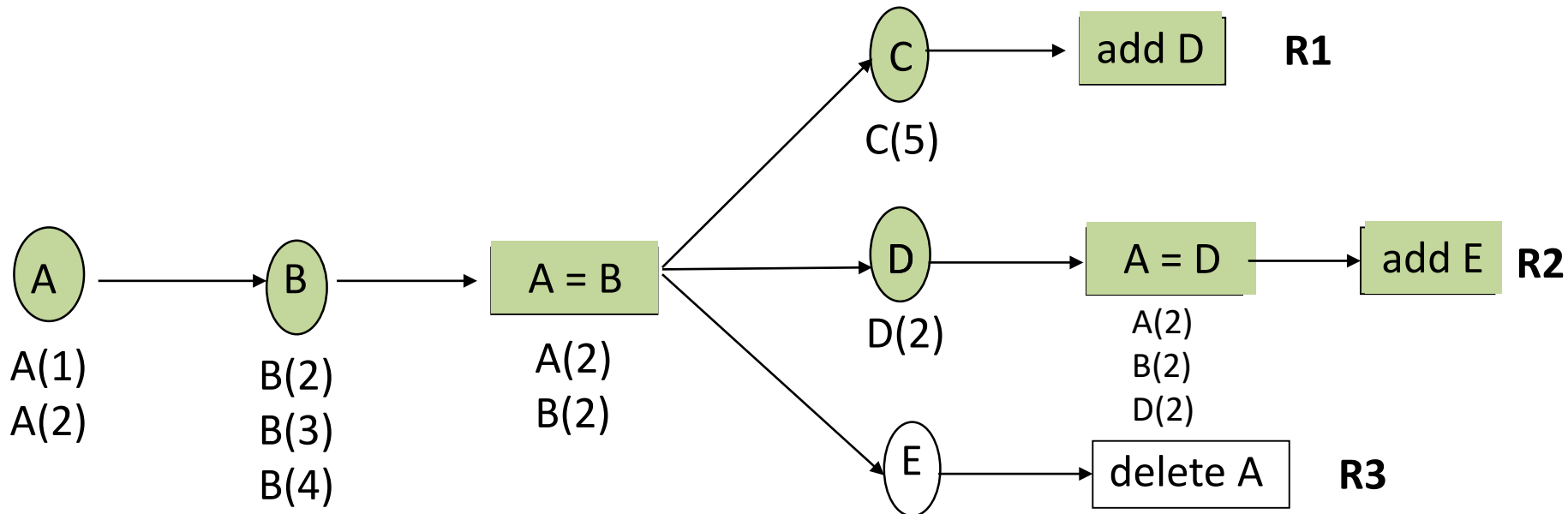


Algoritmo RETE. Ejemplo

CICLO 2

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5), D(2), \mathbf{E(2)}\}$

E(2) pasa a la MT como resultado de la ejecución de R2:

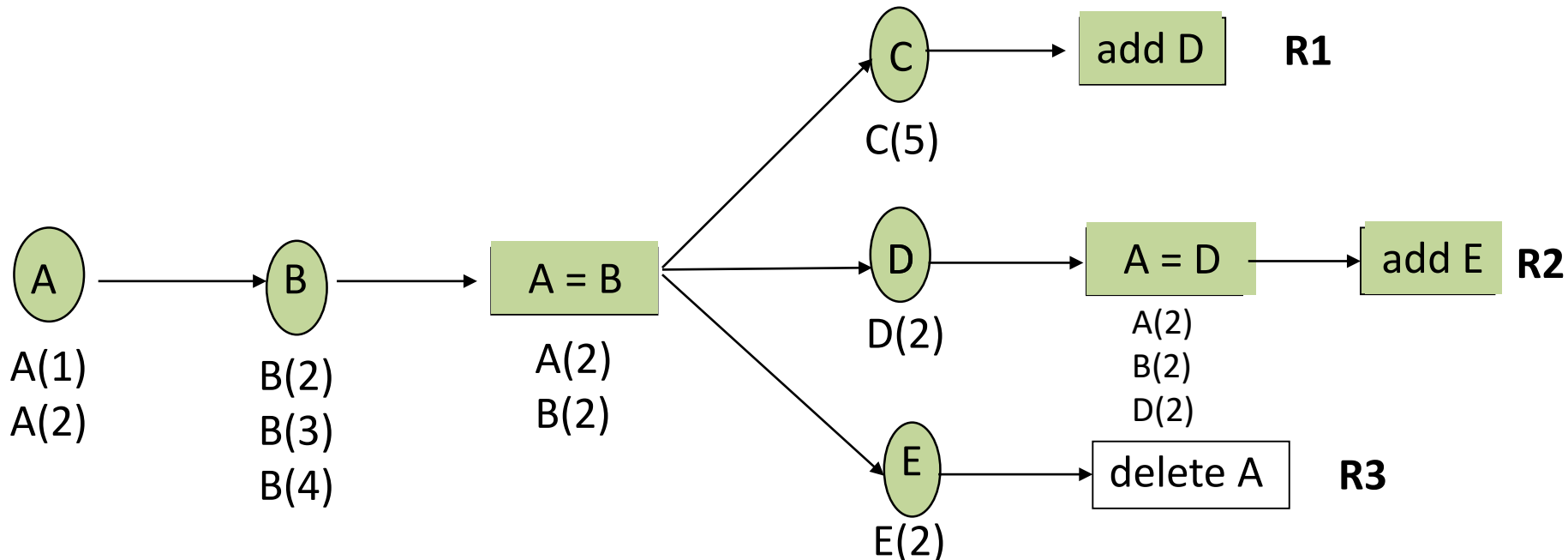


Algoritmo RETE. Ejemplo

CICLO 3

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5), D(2), \mathbf{E(2)}\}$

Se añaden los nuevos hechos de la MT ($E(2)$) a la red:

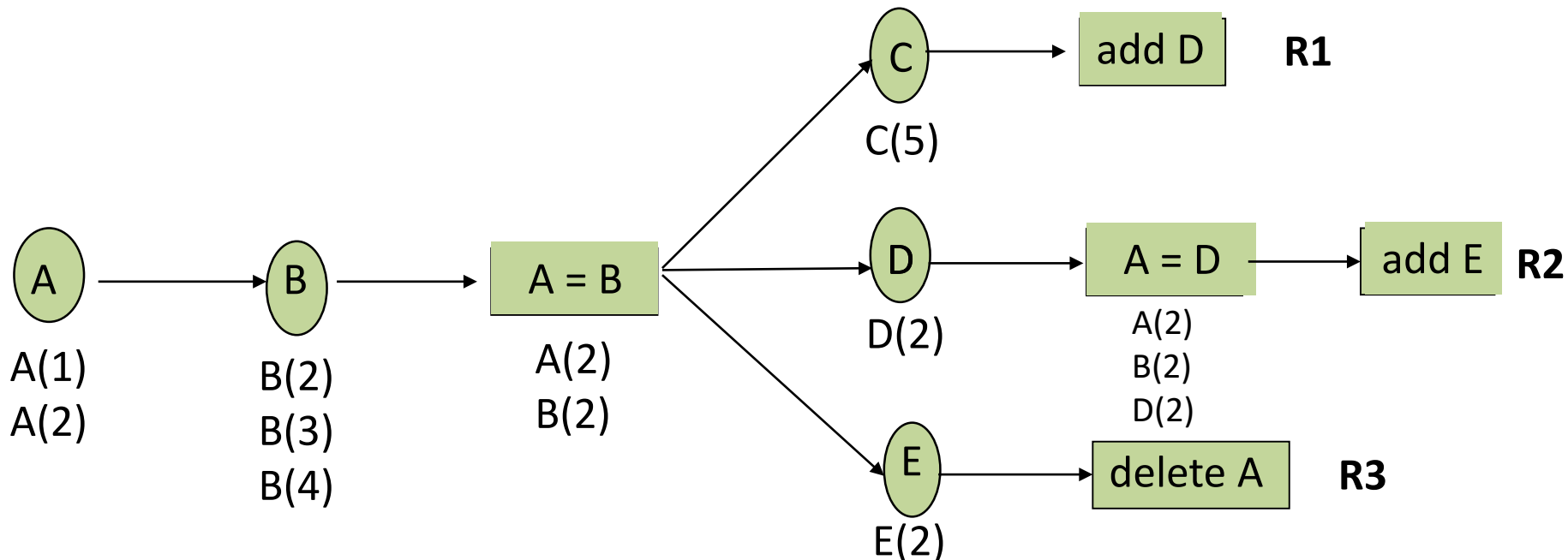


Algoritmo RETE. Ejemplo

CICLO 3

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), A(2), B(2), B(3), B(4), C(5), D(2), E(2)\}$

R3 pasa a la agenda y se activa:

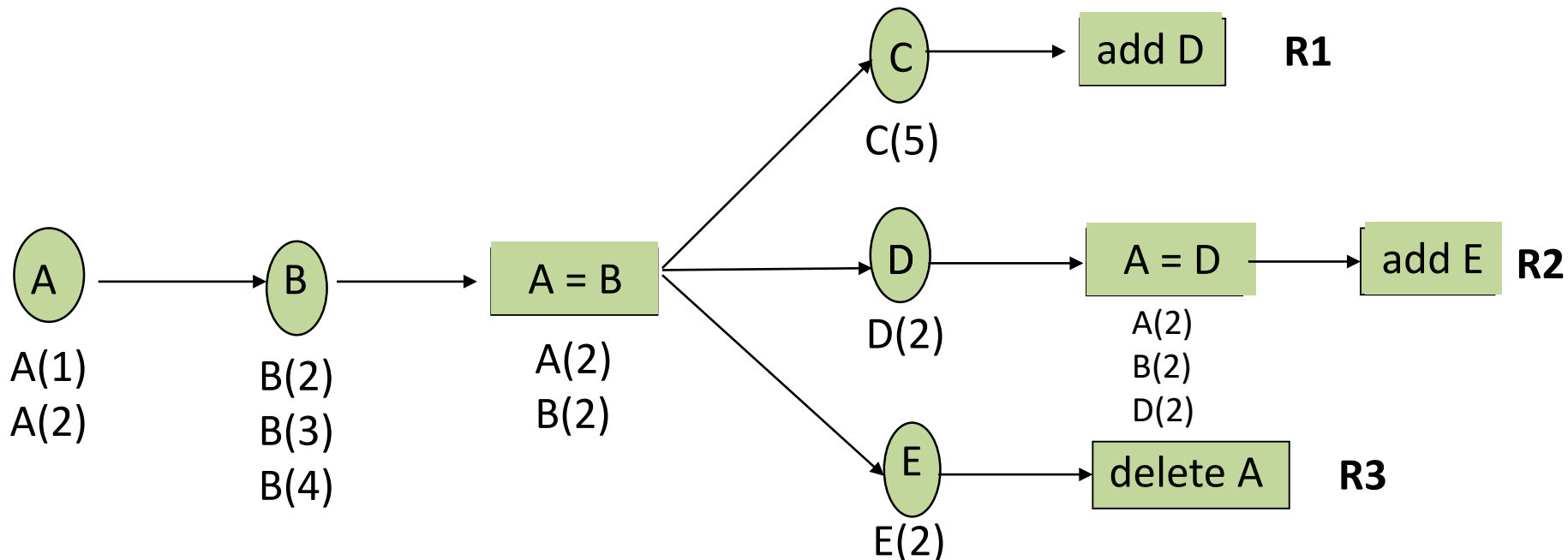


Algoritmo RETE. Ejemplo

CICLO 3

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), B(2), B(3), B(4), C(5), D(2), E(2)\}$

A(2) se elimina de la MT:

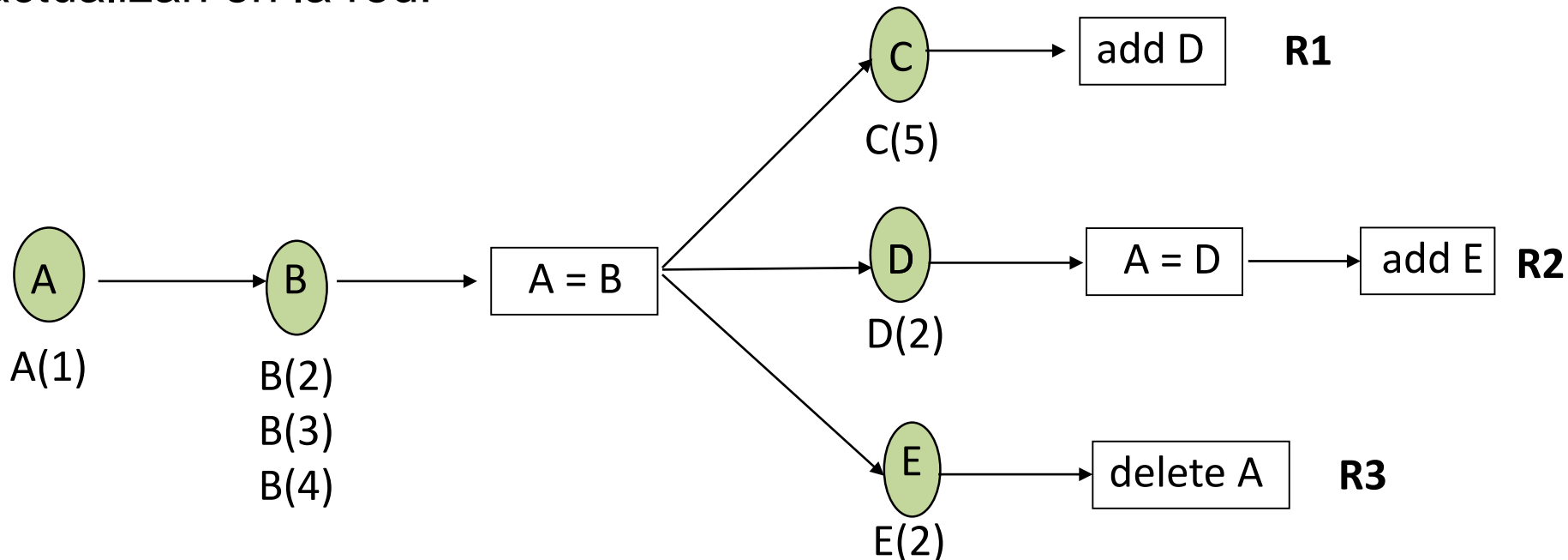


Algoritmo RETE. Ejemplo

CICLO 4

- R1 - $A(x), B(x), C(y) \Rightarrow \text{add } D(x)$
- R2 - $A(x), B(x), D(x) \Rightarrow \text{add } E(x)$
- R3 - $A(x), B(x), E(y) \Rightarrow \text{delete } A(x)$
- MT - $\{A(1), B(2), B(3), B(4), C(5), D(2), E(2)\}$

Los cambios en la memoria de trabajo (eliminación de $A(2)$) se actualizan en la red:



Estrategias de control o resolución de conflictos - I

- Estrategias de resolución del conjunto conflicto (agenda)
 - Qué regla del conjunto conflicto será disparada en cada ciclo
- Dos enfoques de estrategias
 - **Control global:** control independiente del dominio de aplicación
 - Estrategias implementadas en el intérprete (optimización)
 - No son modificables por el programador
 - **Control local:** control dependiente del dominio de aplicación
 - El programador da instrucciones explícitas: establecer prioridades
- **Objetivos, que el sistema se comporte de forma**
 - Sensible: que reaccione a cambios producidos en la memoria de trabajo
 - Estable: que haya continuidad en la línea de razonamiento
- **Principio de refracción:** una regla sólo se dispara una vez con los mismos hechos
- La estrategia ha de tener un equilibrio entre sistemática y eficiente

Estrategias de control o resolución de conflictos – I I

1. **Orden** en la base de reglas
 - Se selecciona la primera regla aplicable,
 - Asume orden lineal explícito en la BR (lo cual puede ser peligroso)
2. **Prioridades** asociadas a las reglas (*conocimiento heurístico*)
 - Se selecciona la regla de **mayor prioridad**
 - La prioridad la establece el experto del dominio
3. **Especificidad**: más prioridad a las reglas con más condiciones
4. **Novedad**: prioridad de las reglas según
 - a) Antigüedad (en MT) de sus hechos
 - b) Momento de activación de la regla
 - (set-strategy depth)** : dispara la más recientemente activada
 - (set-strategy breadth)** : dispara la que lleva más tiempo activada
5. **Arbitrariedad**
 - Cuando no existen criterios adicionales disponibles (*no sistemática*)

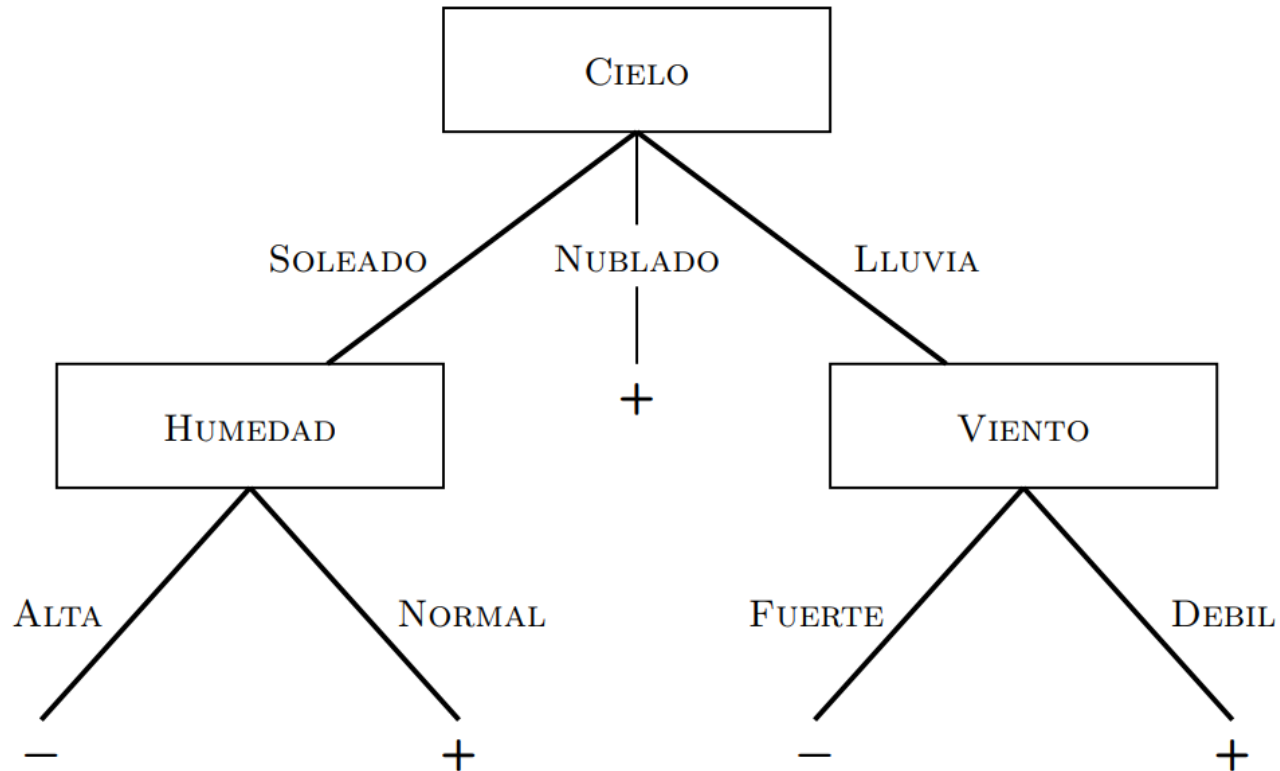
Aprendizaje inductivo y reglas

- Las reglas no sólo conocimiento experto de humanos
- Se pueden aprender las reglas de decisión a partir de ejemplos (casos de experiencias)

EJ.	CIELO	TEMPERATURA	HUMEDAD	VIENTO	JUGAR TENIS
D ₁	SOLEADO	ALTA	ALTA	DÉBIL	-
D ₂	SOLEADO	ALTA	ALTA	FUERTE	-
D ₃	NUBLADO	ALTA	ALTA	DÉBIL	+
D ₄	LLUVIA	SUAVE	ALTA	DÉBIL	+
...					

- Algoritmos de **aprendizaje de reglas** por inducción
 - Construir árbol de decisión ID3
 - FOIL: programación lógica inductiva (ILP)
 - 1R
 - PRISM
 - AQ
 - CN2: maneja valores desconocidos
 - algoritmo RIPPER: poda incremental reducida

Algoritmo de aprendizaje inductivo ID3



Leemos el árbol de decisión como reglas: se aprende una regla (disyunción de conjunciones) para cada clase (+ y -)

Si (cielo =nublado) OR (cielo =lluvia AND viento = débil) OR (cielo = soleado AND humedad= normal) ENTONCES JUGAR_TENIS= SI

Si (cielo =lluvia AND viento = fuerte) OR (cielo = soleado AND humedad= alta) ENTONCES JUGAR_TENIS= NO

Ventajas de los Sistemas de Reglas - I

- Separación de conocimiento y control
 - Simplifica el desarrollo de sistemas expertos
- Carácter totalmente modular de la base de conocimiento
- Uniformidad
 - Conocimiento representado con una sintaxis simple y homogénea
- Naturalidad para representar cierto tipo de conocimiento
- Control dirigido por patrones
 - Los programas en IA necesitan flexibilidad
 - Las reglas pueden “dispararse” en cualquier secuencia
 - Estado → reglas aplicables → camino a la solución
- Independencia del lenguaje
- El modelo es independiente de la representación elegida para reglas y hechos, siempre que el lenguaje soporte encaje de patrones

Ventajas de los Sistemas de Reglas - II

- Modelo plausible del mecanismo humano de resolución de problemas
 - Razonamiento basado en reglas: decidir qué hacer o qué se puede deducir cuando se dan una serie de condiciones
- Trazabilidad del proceso de razonamiento y explicaciones
 - Se puede comunicar qué condiciones se conocían en un momento determinado, por qué se ha escogido una regla y qué se ha deducido tras aplicarla

Inconvenientes de los Sistemas de Reglas

- Ineficiencia (se mejora con algoritmos como RETE)
 - El proceso de reconocimiento de patrones puede ser ineficiente
 - El precio a pagar por la modularidad, flexibilidad y uniformidad
- Dificultad de verificación de la consistencia de la BC
- Opacidad y dificultad de depuración
 - Es difícil examinar una BC y determinar qué acciones van a ocurrir (depende del motor de inferencia)
 - No hay un flujo de control claro
 - La división del conocimiento en reglas hace que cada regla individual sea fácilmente tratable, pero se pierde la visión global
 - Las reglas representan pasos muy pequeños en la resolución del problema sin que haya una jerarquía de reglas
- Incapacidad de aprender
- Dificultad en cubrir todo el conocimiento
 - Cuello de botella de la adquisición de conocimiento
- Herramientas de desarrollo intentan suplir las ineficiencias con utilidades

Razonamiento basado en casos

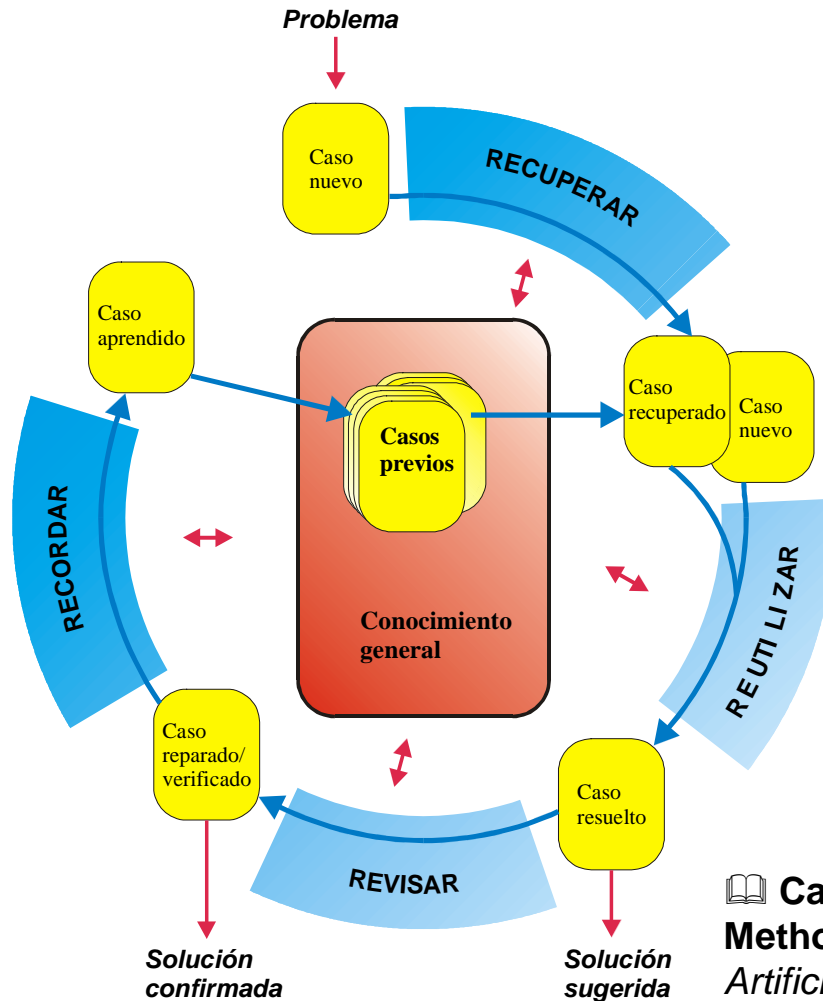
- ¿Cómo conseguir representar mediante hechos y reglas todo lo que un experto –o una comunidad de expertos– sabe?
 - **Cuello de botella de la adquisición de conocimiento**
 - ¿Hay algún experto dispuesto a dedicar el tiempo necesario?
 - ¿El experto y el ingeniero del conocimiento hablan el mismo “idioma”?
 - ¿El ingeniero del conocimiento es capaz de captar todas las sutilezas del dominio para poder así representarlas?
 - ¿Es posible **formalizar** el conocimiento obtenido?
- Adquisición y reutilización de casos de resolución de problemas: Razonamiento por analogía: sistemas basados en casos (CBR, *Case Based Reasoning*)

Razonamiento basado en casos

- Sabiduría heurística en forma de ejemplos
- Los seres humanos resolvemos problemas en base a nuestras experiencias pasadas y no a partir de un conocimiento detallado
 - Médicos, ingenieros, programadores expertos, arquitectos, abogados, jugadores de ajedrez, cocineros, ...
- El CBR facilita la adquisición de conocimiento
 - A los expertos les resulta más sencillo “contar batallitas” que proporcionar reglas de aplicación general.
 - Facilita el aprendizaje - *Lazy Learning*
- Un sistema CBR sólo necesita un conjunto de problemas resueltos (Base de Casos) + conocimiento de similitud + conocimiento de adaptación
 - Los problemas tienden a ser recurrentes
 - Podemos aprender de los errores

Ciclo CBR

□ CBR como alternativa a los sistemas basados en reglas



📖 **Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches.**
Artificial Intelligence Communications, 7 (1),
Aamodt A., & Plaza E., 1994

Bibliografía

- **Akerkar, R. A., Sajja, P. S.** Knowledge-based systems
Jones and Bartlett, 2010
<http://www.ucm.es/BUCM/checkip.php?http://proquest.safaribooksonline.com/?uiCode=complutense&xmlId=9780763776473>
- **Giarratano, J., Riley, G.** Sistemas Expertos: Principios y Programación. International Thomson Editores, 2001
- **Gonzalez, A. J., Dankel, D. D.** The Engineering of Knowledge Based Systems: Theory and Practice. Prentice Hall, 1993
- **Kendal, S. L., Creen, M.** An Introduction to Knowledge Engineering Springer-Verlag London Limited, 2007
<http://www.ucm.es/BUCM/checkip.php?http://dx.doi.org/10.1007/978-1-84628-667-4>