

3.3 Representaciones Estructuradas

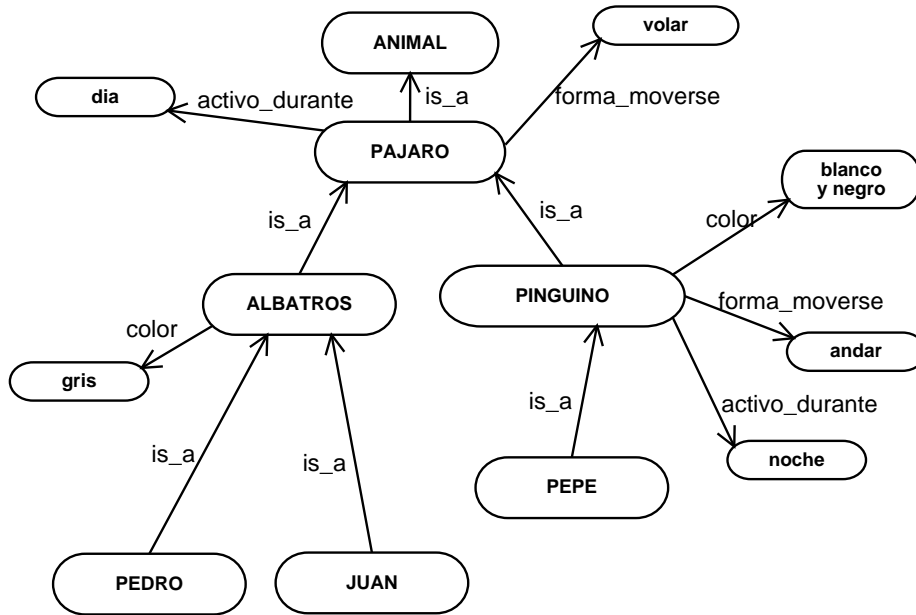
3.3.1 INTRODUCCIÓN

- OBJETIVO: Representar grandes conjuntos de hechos de forma estructurada y comprimida.
 - Agrupar propiedades + representar objetos complejos.
 - Representar conocimiento taxonómico.
 - Relaciones: IS_A, PART_OF
 - Posibilidad de *herencia de propiedades*
 - Representar escenarios y secuencias “típicas” de acontecimientos.
- Difícil e “incómodo” de representar en lógica formal.
- Técnicas de representación:
 - Redes semánticas.
 - Modelos de dependencia conceptual
 - Frames y guiones
 - Reglas de producción

3.3.2 REDES SEMÁNTICAS

- Método declarativo de representación del conocimiento.
- Estructuras gráficas (GRAFOS): codifican propiedades + conocimiento taxonómico sobre objetos
 - NODOS: Entidades del dominio (categorías u objetos)
 - ARCOS ETIQUETADOS: Relaciones entre entidades
- Enlaces (arcos etiquetados):
 - Un enlace UNIDIRECCIONAL por cq. relación/propiedad que podamos definir
 - Definen una relación binaria entre dos nodos
 - Tipos relaciones:
 - OCURRENCIA: vínculo entre un objeto particular y la clase de la que pertenece.
Etiqueta: \in , PERTENECE
 - GENERALIZACIÓN: un objeto es un caso particular de otro objeto de naturaleza más general. Etiqueta: IS_A
 - AGREGACIÓN: vínculo entre un objeto y los objetos que son parte de él.
Etiqueta: PART_OF
 - PROPIEDADES: vínculos entre objetos y características de dichos objetos
 - ACCIONES: vínculos de carácter dinámico
 - OTRAS RELACIONES ESPECÍFICAS
- Relación con Lógica Formal
 - Correspondencia directa Lógica de Predicado (L.P.) y Redes Semánticas (R.S.)
 - Toda R.S. puede representarse mediante fórmulas lógicas
 - Cq. red semántica tendrá asociada un conjunto de tuplas OBJETO-RELACION-VALOR
 - $\text{Nodo1} + \text{Arco_Etiquetado} + \text{Nodo2} \rightarrow \text{EtiquetaArco}(\text{Nodo1}, \text{Nodo2})$
 - VENTAJAS R.S. respecto L.P.
 - Notación gráfica facilita comprensión
 - Fácil especificar y manejar excepciones.
 - Modelo de ejecución más sencillo y eficiente, pero limitado
 - ◇ Inferencias y consultas en base a los enlaces

EJEMPLOS



TUPLAS OBJETO-ATRIBUTO-VALOR

LÓGICA PREDICADOS

OBJETO	ATRIBUTO	VALOR
pájaro	is_a	animal
pájaro	forma_moverse	volar
pájaro	activo_durante	día
pingüino	is_a	pajaro
pingüino	color	blanco_negro
pingüino	forma_moverse	andar
pingüino	activo_durante	día
...
pepe	pertenece	pingüino
...

OPCIÓN 1

$is_a(Pájaro, Animal)$
 $is_a(Pingüino, Pájaro)$
 $pertenece(Pepe, Pingüino)$
 $forma_mover(Pájaro, Volar)$
 $forma_mover(Pingüino, Andar)$

$\forall x, s, f [is_a(x, s) \wedge forma_mover(s, f) \rightarrow forma_mover(x, f)]$
 ...

OPCIÓN 2

$pingüino(Pepe)$
 $albatros(Juan)$

$\forall x pájaro(x) \rightarrow animal(x)$
 $\forall x pingüino(x) \rightarrow pájaro(x)$
 $\forall x albatros(x) \rightarrow pájaro(x)$

$\forall x pajaro(x) \rightarrow forma_mover(x, Volar)$
 $\forall x pajaro(x) \rightarrow activo_durante(x, Día)$

$\forall x pingüino(x) \rightarrow forma_mover(x, Andar)$
 $\forall x pingüino(x) \rightarrow color(x, Blanco_Negro)$

...

-
- Mecanismos de Inferencia y Razonamiento
 - HERENCIA: Mecanismo más importante
 - Toda propiedad de una clase es cierta para cq. ejemplo de la clase
 - Establece jerarquía taxonómica
 - Inferencia no monotónica
 - ◇ Herencia con excepciones (= cancelación de la herencia)
 - ◇ Conocimiento por defecto
 - ◇ Ejemplo: *Pingüino es un Ave que no vuela*
 - PROBLEMAS:
 - ◇ Manejo herencia múltiple (por varias rutas)
 - ◇ Posibilidad de inferir conocimiento incorrecto
 - RASTREO
 - Uso de la propiedad transitiva de algunas relaciones (PART_OF, mayor_que,...)
 - Se infiere a partir de 2 arcos un "tercero"
 - PROBLEMAS: Posibilidad de inferir conocimiento incorrecto
 - EMPAREJAMIENTO
 - Construir, para un problema, un fragmento de red semántica.
 - "Compararlo" con una red semántica completa
 - Se deriva conocimiento implícito en la red global a partir de partes de una red, rellenando las partes "en blanco"
 - VENTAJAS. REDES SEMÁNTICAS:
 - Sencillas y fácil comprensión
 - Representan relaciones jerárquicas de forma modular
 - Eficiencia
 - INCOV. REDES SEMÁNTICAS:
 - Semántica poco clara (no "normalizada")
 - Problemas para representar conocimiento no taxonómico
 - Problemas para representar disyunciones, implicaciones, negaciones
 - Ejemplo: *Una gallina no nada, Una gallina anda o salta ...*
 - Ejemplo: *Si las aves que no vuelan, tienen patas fuertes*
 - IDEM con cuantificación
 - Manejo rutas conflictivas en herencia múltiple

RAZONAMIENTO NO MONOTÓNICO

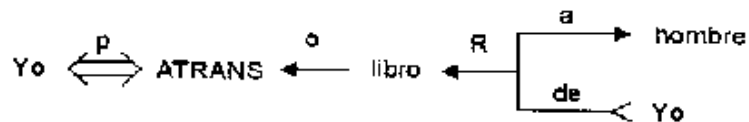
■ RAZONAMIENTO NO MONOTÓNICO

- IDEA BÁSICA: La adición de conocimiento nuevo puede dar lugar a que parte del conocimiento derivado que se había obtenido anteriormente sea ahora incorrecto.
- En Lógica de Predicados:
 - El razonamiento es siempre monotónico
 - Añadir nuevos axiomas a la base de conocimiento no reduce el conjunto de f.b.d que pueden ser demostradas.
 - Formalmente:
Si $\Phi \vdash \Psi$, entonces, siendo Δ un conj. de f.b.d, se verifica $(\Phi \cup \Delta) \vdash \Psi$
- En ocasiones, el razonamiento es no monotónico por naturaleza
 - Uso de *inferencias por defecto*
 - Se asume que algo es cierto, salvo que tengamos conocimiento contrario
 - Si aparece contradicción, debemos retractarnos de resultados obtenidos por defecto
 - En general, problema complejo
 - ◇ Necesidad mecanismos para rastrear conocimiento y razonamientos basados en creencias rebatidas
 - ◇ Util en razonamiento temporal y de sentido común

■ RAZ. NO MONOTÓNICO EN REDES SEMANTICAS

- Cancelación de la herencia
 - Modificación del mecanismo de herencia para incorporar razonamiento no monotónico en R.S.
 - IDEA: Dar preferencia al conocimiento sobre categorías más específicas
 - Categorías generales aportan el conocimiento por defecto
 - Los niveles inferiores pueden cancelar ese conocim. por defecto
 - Análogo al *overriding* de propiedades y métodos en POO.
- Ejemplo (sobre red semántica anterior)
 - Supongamos que no existe el enlace:
$$\text{PINGUINO} \xrightarrow{\text{forma_move}} \text{ANDAR}$$
 - Un *pingüino* es un *pájaro*. Con los que sabemos ahora, por ser un *pájaro* (y mientras no se demuestre lo contrario) asumimos que *vuela* por aplicación de la herencia de propiedades.
 - Si se añade el enlace anterior
Aplicando cancelación de la herencia, el razonamiento por defecto "*los pingüinos vuelan*" es falso.
 - Debemos retractarnos al saber que los pingüinos "*andan*"

"Yo le dí el libro al hombre"



donde los símbolos tienen los siguientes significados:

- Las flechas indican direcciones de la dependencia.
- La flecha doble indica los tipos de enlaces entre el actor y la acción.
- p indica tiempo pasado.
- ATRANS es una de las acciones primitivas utilizadas por la teoría. Indica una transferencia de posesión.
- o indica la relación OBJECT CASE.
- R indica la relación RECIPIENT CASE.

■ ACCIONES PRIMITIVAS (Schank)

- ATRANS transferencia de una relación abstracta (dar,...)
- PTRANS transferencia de una localización física de un objeto (ir,...)
- PROPEL aplicación de fuerza física a un objeto (empujar,...)
- MOVE movimiento de una parte del cuerpo por su propietario (dar patadas,..)
- GRASP asimiento de un objeto por un actor (coger,...)
- INGEST ingestión de un objeto por un ser animado (comer,...)
- EXPEL expulsión de algo del cuerpo de un ser animado (llorar,...)
- MTRANS transferencia de info. metal (decir,...)
- MBUILD construcción de info. nueva a partir de otra vieja (decidir,...)
- SPEAK producir sonidos (hablar,...)
- ATTEND concentración de un órgano sensorial hacia un estímulo (escuchar, mirar,...)

■ CATEGORÍAS CONCEPTUALES PRIMITIVAS

- ACTs: acciones
- PPs: objetos
- AAa: modificadores de acciones
- PAs: modificadores de objetos

■ REGLAS CONSTRUCCIÓN Figura siguiente

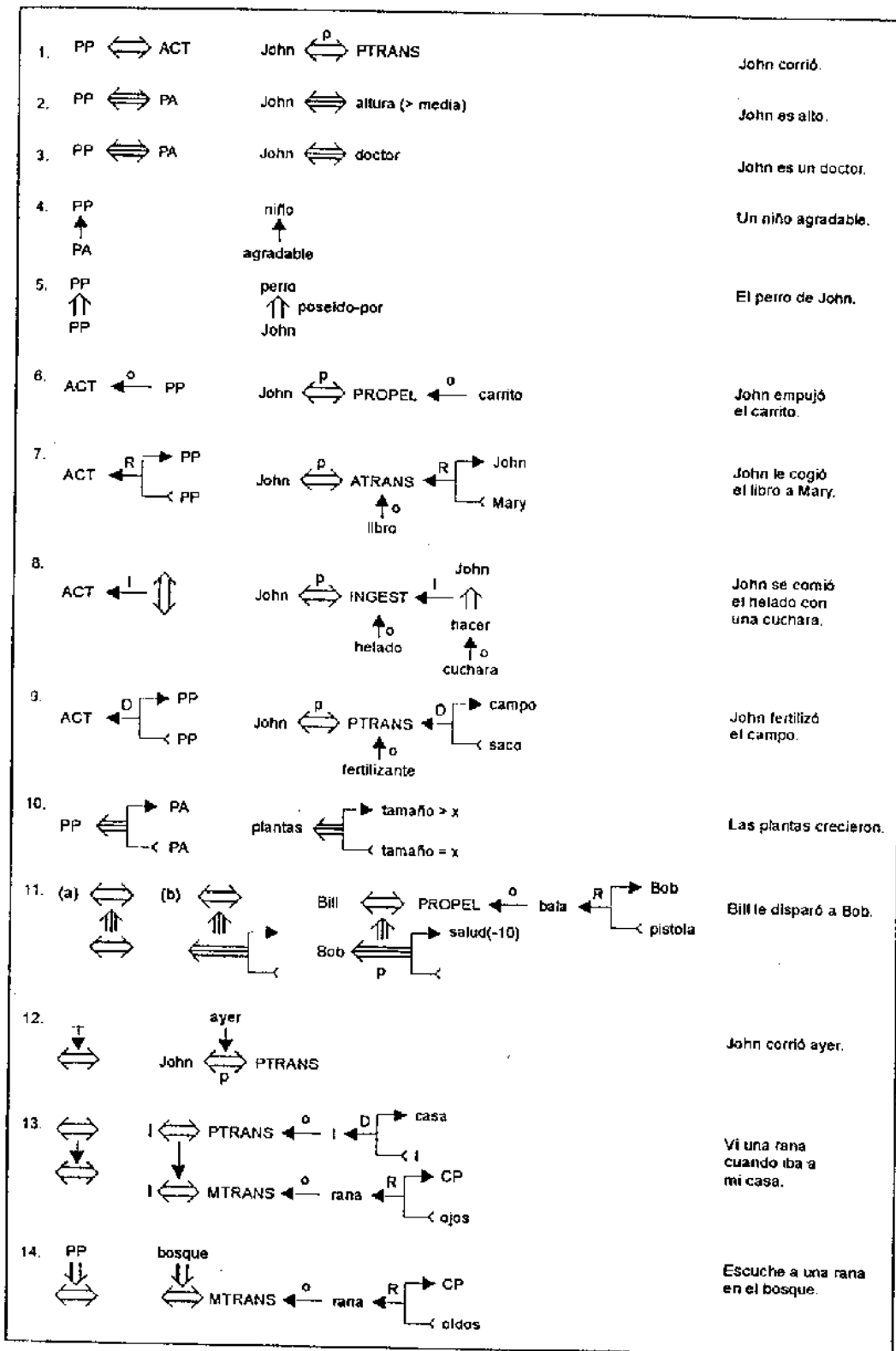


Figura 10.2. Las dependencias de la Dependencia conceptual (CD).

3.3.4 FRAMES (Marcos)

- Método declarativo de representación del conocimiento.
- OBJETIVO: Suministrar mecanismos de razonamiento por semejanza
- Representación estructurada de conocimiento estereotipado
- Un FRAME representa una entidad del mundo real
 - Los hechos se agrupan en objetos
 - FRAME = red semántica compleja
- ESTRUCTURA
 - CABECERA: Etiqueta lingüística que da nombre al frame
 - Es representativa de la clase de objetos que describen
 - SLOTS (ranuras): Contienen la info. relativa a la cabecera del FRAME
 - Conjunto de pares atributo-valor
 - Representan: propiedades de la clase, objetos, propiedades de los objetos, subobjetos, relaciones entre frames, etc...
 - Slots estructurados en niveles: cada indentación especializa nivel superior
 - Pueden ser propios o heredados
- Pueden incorporar:
 - Metaconocimiento: info. sobre el frame, ayuda a manipular el conocimiento
 - Información procedimental (DEMONS)
 - se activan cuando se accede al slot
 - permiten: uniones procedimentales entre frames, interacción mundo exterior
 - aportan carácter dinámico
 - ej: IF_NEEDED, IF_ADDED, IF_REMOVED, etc
- RAZONAMIENTO e INFERENCIA:
 - HERENCIA: Mediante inclusión de slots IS_A
 - EMPAREJAMIENTO: Relleno de slots vacíos.
- VENTAJAS
 - Permiten trabajar con info. incompleta
 - Fácil de implementar y ampliar
 - Herencia de propiedades (“infieren” conocimiento no representado explícitamente)
 - Interacción mundo exterior y cooperación esquemas de representación procedim
- INCONV.
 - Problemas para representar conocimiento no taxonómico

- IDEM con disyunciones e implicaciones

■ EJEMPLOS

RED SEMÁNTICA ANTERIOR

```

FRAME: pájaro
  is_a: animal
  forma_moverse: volar
  activo_durante: día

FRAME: pingüino
  is_a: pájaro
  color: blanco_y_negro
  forma_moverse: andar
  activo_durante: noche
  tamaño: mediano

FRAME: pepe
  is_a: pingüino
  ...

```

INDENTACIÓN DE SLOTS

```

FRAME: casa
  elementos_básicos:
    cimientos:
      composición: hormigón
      localización: bajo_tierra
    paredes:
      composición: ladrillo
    elementos_complementarios:
      puertas:
        tipo: convencional
        material: madera
      ventanas:
        ....
  tejado:
    material: pizarra
    color: negro
    localización: arriba

```

METACONOCIMIENTO y DEMONS

```

FRAME: sintomas_065
  info:
    formato: reducido
    uso: informativo
    autor: Dr. Pepe
    version: 0.99
  paciente: Sr. Pérez
  temperatura: alta
  sudoración: presente
  dolor_muscular: ausente
  descompos_organos_internos: ausente

FRAME: juan
  is_a: empleado
  IF_NEEDED: cargar_base_datos
  IF_REMOVED: print_borrado_base_datos

```

3.3.4 GUIONES

- Especialización de los FRAMES para representación de entornos dinámicos
- Frames donde el tiempo está implícito.
- Representan secuencias típicas de eventos
- ELEMENTOS
 - Nombre del guión
 - Condiciones de entrada (requisitos exigidos para utilizar el guión)
 - Resultados (condiciones de terminación)
 - Herramientas (objetos que aparecen en el guión)
 - Papeles o roles (actores del guión)
 - Escenas (secuencias típicas de acontecimientos)
- Razonamiento por emparejamiento.
- Activación del guión: verificar si el problema planteado satisface las condiciones de entrada.

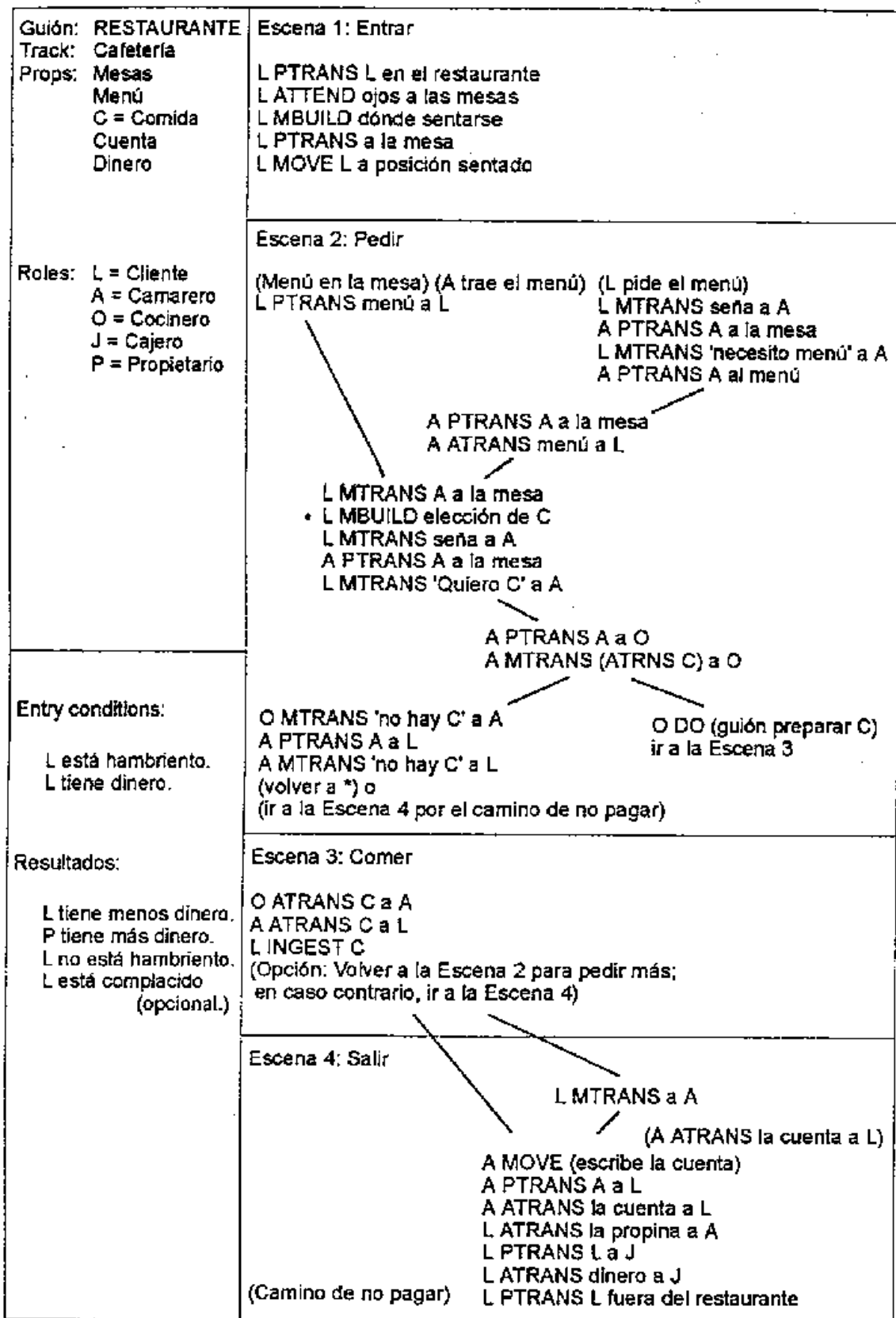


Figura 10.5. El guión de un restaurante.

3.4 Reglas de Producción

3.4.1 REGLAS DE PRODUCCIÓN

- Método procedimental de representación del conocimiento
- Estructura
 - SI** <condiciones>
 - ENTONCES** <conclusiones, acciones, hipótesis>
- Cada regla **SI-ENTONCES** establece un gránulo completo de conocimiento
- Regla \approx Operador válido en un espacio de estados
- CONDICIONES (tb. premisas, precondiciones, antecedentes, ...)
 - Formadas por cláusulas y conectivas (AND, OR, NOT)
 - Representación clausal debe corresponderse con conocimiento del dominio
 - Formato típico: <parámetro/relación/valor>
 - PARÁMETRO: característica relevante del dominio
 - RELACIÓN: entre parámetro y valor
 - VALOR: numérico, simbólico o literal
 - También en forma de predicados lógicos
- CONCLUSIONES, ACCIONES, HIPÓTESIS (tb. consecuentes, ...)
 - Conclusiones, Hipótesis: conocimiento declarativo
 - Acciones: cq. acción procedimental (actualiz. conocimiento, interacción con exterior, etc..)
- REGLAS ESPECIALES
 - Reglas IF ALL: equivalen a reglas con las cláusulas de las condiciones conectadas con AND
 - Reglas IF ANY/ IF SOME: equivalen a reglas con las cláusulas de las condiciones conectadas con OR
- EJEMPLO

<pre>IF: temperatura = alta AND sudoración = presente AND dolor_muscular = presente THEN: diagnóstico_preliminar = gripe</pre>	<pre>IF: diagnóstico_preliminar = gripe AND descompos_organos_internos = presente THEN: diagnóstico_preliminar = ébola</pre>
--	---

3.4.2 SISTEMAS BASADOS EN REGLAS DE PRODUCCIÓN

- Reglas \approx Operadores en búsquedas en espacio de estados
- Inferencia similar al *MODUS PONENS* (con restricciones)
 - Sintaxis relajada
 - Se permiten acciones en los consecuentes
 - Mecanismo de control determina que inferencias se pueden realizar
- TIPOS de SISTEMAS
 - En función de sintaxis de reglas y de mecanismos de control (\approx búsqueda)
 - SIST. ENCADENAM. HACIA ADELANTE (dirigidos por los datos)
 - Regla ACTIVADA si antecedentes emparejan con algunos hechos del sistema
 - En IF ALL, todos. En IF ANY, al menos uno.
 - Se parte de los hechos ya confirmados en el sistema
 - Se razona hacia adelante buscando antecedentes que emparejen
 - SIST. ENCADENAM. HACIA ATRÁS (dirigido por los objetivos)
 - Regla ACTIVADA si consecuentes emparejan con algunos hechos del sistema
 - Se comienza con una hipótesis
 - Se razona hacia atrás buscando consecuentes que emparejen
 - MOTOR DE INFERENCIAS elige que reglas ACTIVADAS ejecutar (resolución de conflictos)
 - Consecuentes y antecedentes pueden verse como submetas a verificar a partir de los hechos o hipótesis, respectivamente.
- CARACTERÍSTICAS
 - Modularidad: reglas = pequeñas cantidades de conocimiento (relativamente) independiente
 - Incrementalidad/Modificabilidad: posible añadir/cambiar reglas con relativa independencia
 - Naturalidad y Transparencia: representación del conocimiento próxima y comprensible por personas
 - Capacidad de generar explicaciones
- GENERACIÓN de EXPLICACIONES
 - Posibilidad de “explicar” el porque de un resultado
 - Devolver a usuario la cadena de reglas empleadas
 - Combinar reglas y hechos del árbol de búsqueda según las conectivas
 - Incrementan la “aceptación” del resultado ofrecido (dominios críticos)

3.4.3 ARQUITECTURA DE SIST. BASADOS EN REGLAS

■ COMPONENTES

1. BASE DE CONOCIMIENTOS (BC)

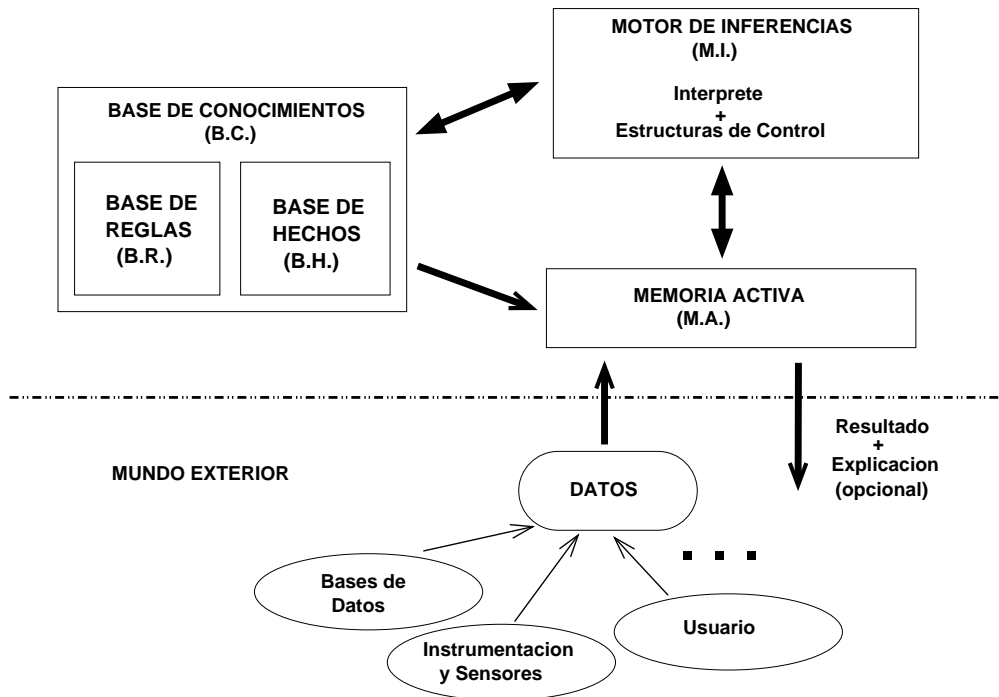
- Reúne todo el conocimiento del sistema
- Formada por base de reglas(**BR**) + base de hechos(**BH**)

2. MEMORIA ACTIVA (MA)

- Colección de hechos, representando el estado actual del problema
- Actúa como “disparador” del motor de inferencias
- Refleja los cambios en el conocimiento del sistema
- Interactúa con el mundo exterior (usuario, bases de datos, etc...)
- Contiene:
 - Datos iniciales del problema + datos incorporados con posterioridad
 - Hechos establecidos durante procesos de inferencia
 - Hipótesis de trabajo, metas y submetas aún no confirmadas
- REGLAS ACTIVADAS
 - Están en condiciones de ser ejecutadas
 - Encadenam. hacia adelante: antecedentes representados en la **MA**
 - Encadenam. hacia atrás: consecuentes representados en la **MA**
 - El **MI** decide cual o cuales de las reglas ACTIVAS se ejecutará

3. MOTOR DE INFERENCIAS (MI)

- Controla el funcionamiento del sistema
 - procesos de emparejamiento
 - selección de reglas
 - ejecución de reglas
 - rutinas externas
- Compuesto por : intérprete + estructuras de control
- Separado e independiente de la **BC**
- Funcionamiento análogo a búsqueda en espacio de estados
 - ESTADO: Representado por conj. hechos de **MA**
 - OPERADORES: Reglas de la **BR**
 - Con encadenam. hacia atrás : búsqueda en grafos AND-OR (búsqueda por subobjetivos)



■ TAREAS MI:

● Ciclo básico:

1. Examen de la **MA** y selección de reglas activas (*emparejamiento*)
 - depende del tipo de encadenamiento
2. Selección reglas a ejecutar (*resolución conflictos*), en función de:
 - estrategia de exploración
 - modelos de resolución de conflictos
3. Ejecución reglas y actualización de la **MA**
4. Mantenimiento del autoconocimiento del sistema
 - control de reglas activadas y ejecutadas
 - control del orden de activación y del orden de ejecución
 - mantener orden de los hechos en la **MA**

● El ciclo anterior depende de la dirección del proceso inferencial

- Encadenamiento hacia adelante (progresivo): emparejamiento de hechos en **MA** con antecedentes
- Encadenamiento hacia atrás (regresivo): emparejamiento de hipótesis en **MA** con consecuentes

■ INTERPRETACIÓN COGNITIVA

- Correspondencia componentes sistemas de reglas con elementos del pensamiento humano
- MEMORIA ACTIVA: Memoria corto plazo.
 - Conocimiento intermedio que se maneja durante el razonamiento
 - Es temporal (se olvida) y de pequeña capacidad
- BASE DE CONOCIMIENTO: Memoria a largo plazo
 - Conocimiento permanente
 - Puede ser innato o adquirido (aprendizaje)
- MOTOR DE INFERENCIAS: Equivale a mecanismos de razonamiento humanos.
 - Es una aproximación limitada

■ MEJORA EFICIENCIA

- Emparejamiento + Resolución conflictos → Determinan eficiencia sist. reglas
- Proceso de Emparejamiento
 - Emparejamiento: proceso muy costoso ⇒ Punto crítico
 - Unificación clásica (o sus variantes) demasiado costosa (sobre todo encad. adelante)
 - Algoritmo RETE: Mejora eficacia emparejamiento
 - ◇ Preprocesa las reglas, construyendo una red
 - ◇ Red RETE se modifica a medida que se incorporar/eliminan hechos en **MA**
 - ◇ IDEA BASE: Modificaciones en **MA** sólo afectan a una porción de la red
- Estrategias Resolución Conflictos
 - Decidir que regla ACTIVADA ejecutar
 - Ejecutar TODAS o usar estrategias más refinadas
 - ◇ No aplicar 2 veces la misma regla
 - ◇ Preferencia por reglas que usen hechos de incorporación reciente a **MA**
 - ◇ Preferencia por reglas más específicas
 - ◇ Asignación de prioridades (sist. de control)

▪ EJEMPLO

- Base de Reglas: $BR = \{R1, R2, R3, R4, R5, R6, R7, R8\}$

- Reglas:

R1: if B and C then Z
R2: if C(x) and D(y) then H(x)
R3: if F or A then I
R4: if I and J then K
R5: if A and H(x) then Y
R6: if H(x) and E then J
R7: if K and E then X
R8: if H and J then X

- Memoria Activa Inicial: $MA_0 = \{A, C(7), D(8), E, F, (X)\}$
(X) es una hipótesis, C, D, E, y F son los hechos de partida

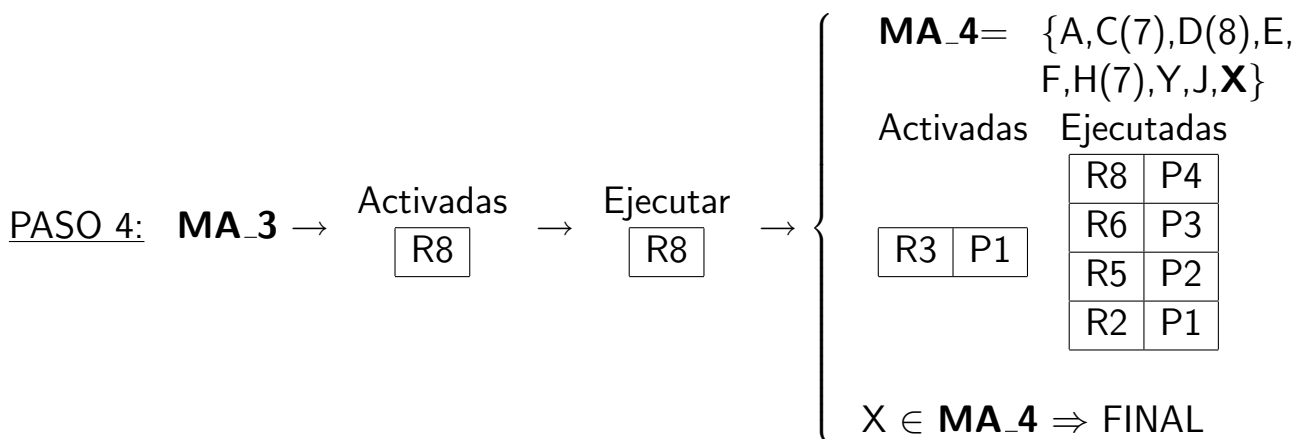
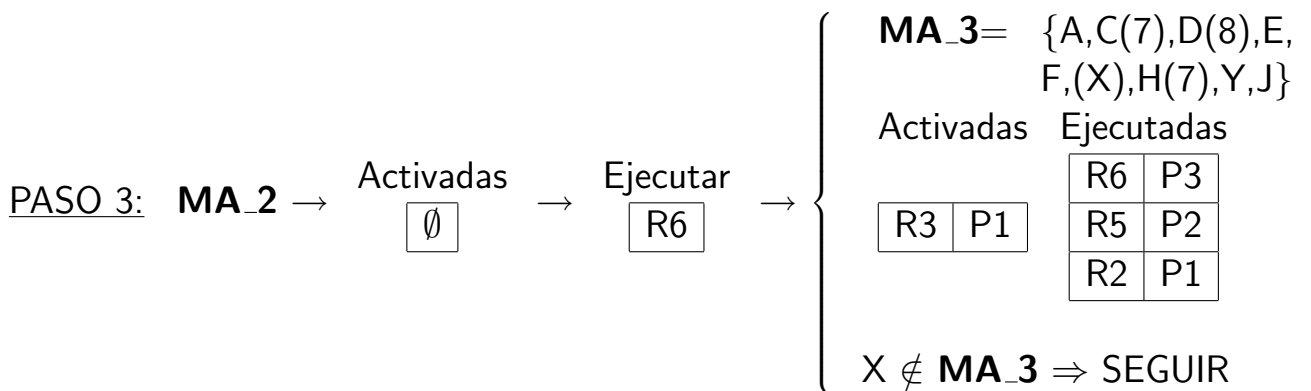
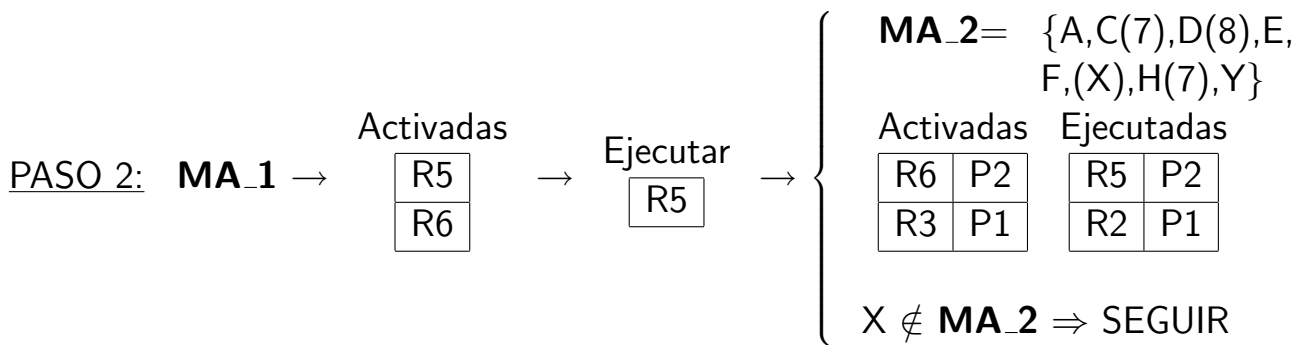
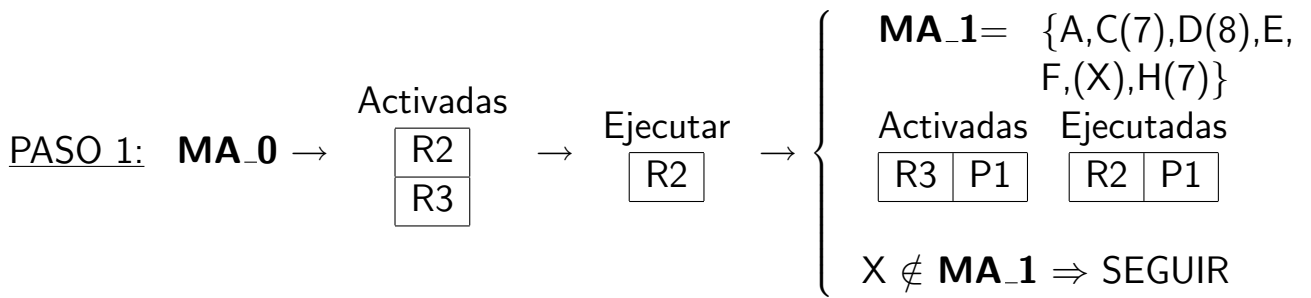
- Motor Inferencias 1:

- Encadenamiento hacia adelante
- Activación de toda regla cuyo antecedente empareje con hechos de MA
- Búsqueda en profundidad
 - ◇ Ejecución de la primera regla de las recientemente activadas
 - ◇ Dar preferencia a reglas nunca ejecutadas
 - Evitar (si es posible) repetir ejecución de la misma regla
 - ◇ Termina cuando la hipótesis X esté en MA

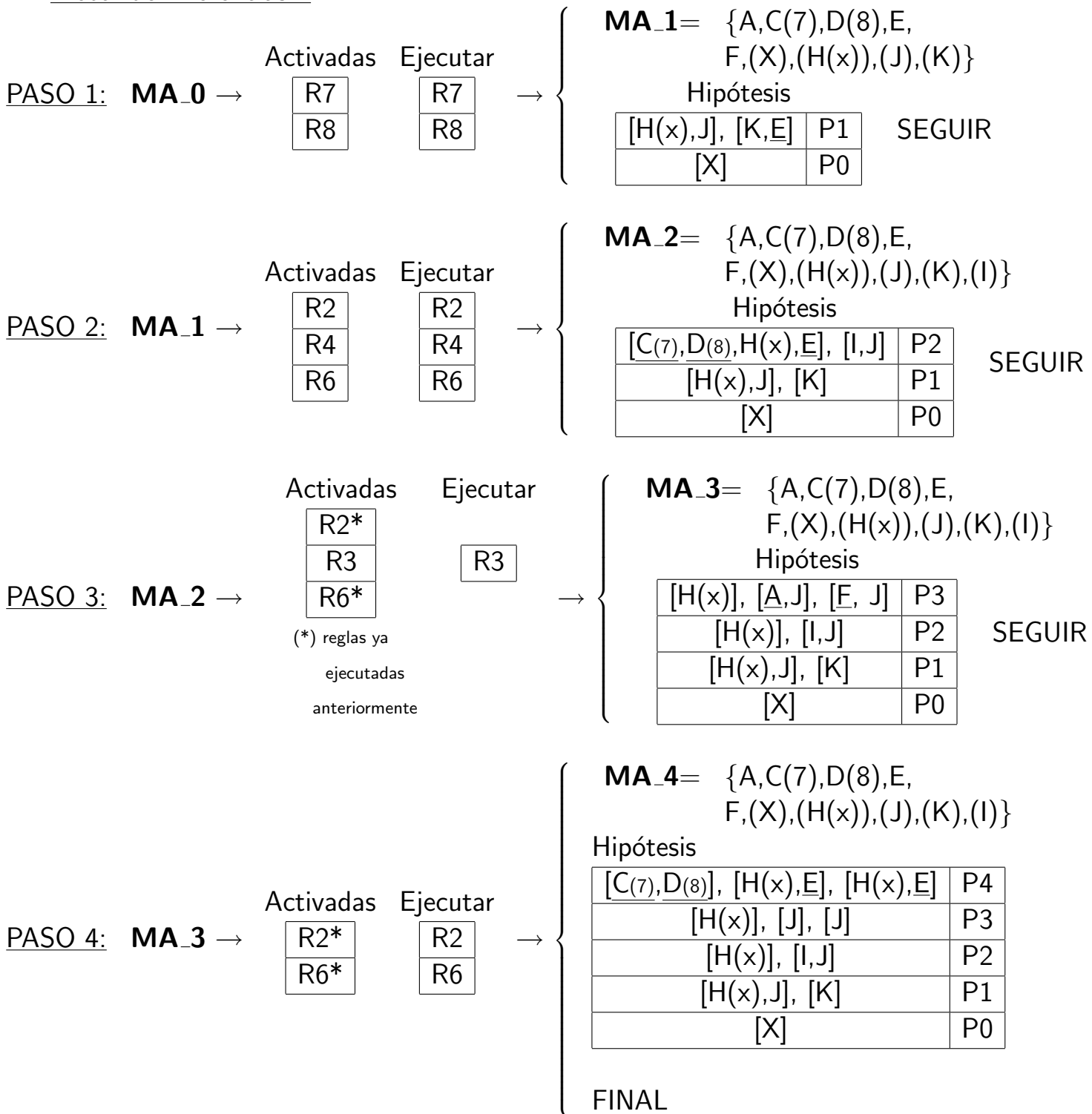
- Motor Inferencias 2:

- Encadenamiento hacia atrás
- Activación de toda regla cuyo consecuente empareje con hipótesis de MA
- Búsqueda exhaustiva en anchura
 - ◇ Ejecutar todas las reglas activadas
 - ◇ Dar preferencia a reglas nunca ejecutadas
 - Evitar (si es posible) repetir ejecución de la misma regla
 - ◇ Termina cuando los hechos de MA agoten una rama de búsqueda

■ Motor de Inferencias 1



▪ Motor de Inferencias 2



NOTA: Las hipótesis subrayadas ya están confirmadas, son hechos de la **MA**

En este punto, se podría:

1. Dejarlo aquí, habiendo “demostrado” **X**
2. Iniciar encadenamiento adelante que añada a **MA** los hechos intermedios utilizados:
 - Paso 5: con R2, añadir H(7) - Paso 6: con R6, añadir J
 - Paso 7: con R8, añadir X

BIBLIOGRAFIA

- LÓGICA
 - RUSSELL & NORVIG: Caps. 6.3, 6.4, 7.1-7.3, 8.1, 8.2, 9.1, 9.2, 9.6, 9.7
 - NILSSON: Caps. 13.1-13.6, 14, 15, 16.1-16.6
- REPRESENTACIONES ESTRUCTURADAS
 - RICH: Caps. 9, 10
 - RUSSELL & NORVIG: Caps. 10.6
 - NILSSON: Caps. 18.3
- SIST. BASADOS EN REGLAS
 - RICH: Caps. 6
 - RUSSELL & NORVIG: Caps. 10.5
 - NILSSON: Caps. 17.4

ENLACES

- **Planning Applet:** University of British Columbia
<http://www.cs.ubc.ca/labs/lci/CIspace/Version4/planning/index.html>
 - Simulación del sistema de planificación STRIPS en el mundo de los bloques
 - STRIPS combina búsquedas en espacios de estados y representación en lógica de predicados restringida
 - Estados representados como conjunciones de átomos (*encima_de(A,B)*, *libre(A)*, *sobre_mesa(A),...*)
 - Operadores representas operaciones sobre esas listas de átomos
 - Precondiciones: describen mediante predicados la situación del mundo antes de aplicar el operador
 - Lista borrar: predicados a eliminar de la nueva representación del mundo
 - Lista añadir: predicados a añadir en la nueva representación del mundo
 - Búsqueda AND-OR (por subobjetivos): devuelve lista de operadores aplicar