Representación del Conocimiento

Frames

Javier Béjar

Inteligencia Artificial - 2023/2024 1Q

CS - GEI - FIB

Representacion del conocimiento

- La lógica como lenguaje de representación tiene dificultades prácticas
- Son necesarios mecanismos mas intuitivos y fáciles de usar
- La psicología cognitiva afirma:
  - La representación y recuperación del conocimiento se realiza a partir de sus relaciones
- Las redes semánticas intentan trasladar esa afirmación a un formalismo
- Una red semántica será un grafo donde:
  - Los nodos representarán conceptos
  - Los arcos (dirigidos) representarán relaciones entre conceptos

- Representan un conjunto restringido de la lógica de predicados
- Permiten representar de manera declarativa los elementos de un dominio
- Se pueden establecer mecanismos de razonamiento específicos que permiten responder a preguntas sobre la representación
  - ¿Están relacionados dos conceptos?
  - ¿Qué relaciona dos conceptos?
  - ¿Cual es el concepto mas cercano que relaciona dos conceptos?
- Si definimos una semántica más rica sobre las relaciones se pueden responder preguntas mas complejas
  - Taxonomías entre conceptos (clase/subclase/instancia)
  - Generalizaciones/Especializaciones
- Ejemplos que conocéis: Diagramas ER, UML

- Se han desarrollado muchos formalismos distintos con diferentes capacidades, no siempre hay un modelo de razonamiento completo tras ellos
- Se mezclan elementos que tienen diferentes niveles de abstracción
  - Conceptos/instancias/valores
  - Relaciones/propiedades
- Es necesaria una estructuración mas adecuada de la información
- Es necesario dotar de una base teórica al modelo de razonamiento

- Son redes semánticas estructuradas
- Un Frame es una colección de atributos y la descripción de sus características
- Las relaciones conectan los frames entre si
- Hay una división explícita entre relación y atributo
- Relaciones y atributos tienen a su vez una estructura que permite describir su semántica
- Son lenguajes de Frames por ejemplo: Diagramas ER, UML

- El mecanismo de razonamiento sobre la parte declarativa esta basado en la lógica de descripción (*Description Logic*) (también es el fundamento de la orientación a objetos)
  - Inclusión entre conceptos (relaciones de especialización/Generalización)
  - Herencia de valores y atributos
  - Relaciones conjuntistas (unión, intersección, pertenencia, transitividad)
- Los Frames pueden añadir a la parte declarativa una procedimental
  - Funciones y métodos que reducen el coste de la inferencia

- Un frame representa un concepto
- Esta dividido en una parte declarativa (atributos) y una procedimental (métodos)
- La parte declarativa permite describir la semántica del concepto (características)
- La parte procedimental permite definir como obtener información o hacer cálculos sobre sus características o las relaciones que pueda tener con otros frames
- Un frame se describirá mediante su nombre y la lista de atributos y métodos que posee

Descripción de un frame:

Frame `<nombre>`
- slot `<nombre-slot>`[1]
- slot `<nombre-slot>`[1]
- ...
- slot `<nombre-slot>`[1]
- métodos [2]
  - **acción** `<nombre-método> (parámetros) [H/noH]`
  - ...
  - **función** `<nombre-método> (parámetros) devuelve <tipo> [H/noH]`

[1] Cada slot puede ir acompañado de modificadores respecto a la definición global del slot

[2] Las acciones/funciones que describen los métodos usan la variable `F` como referencia implícita al frame o instancia de frame en el que se activa el método, y por ello no se ha de pasar como parámetro

- Las relaciones permiten conectar conceptos (frames)
- Una relación poseerá una descripción que establecerá su semántica, sus características y su funcionamiento
- Las relaciones serán la base del mecanismo de inferencia: la herencia de propiedades
- Dividiremos las relaciones en dos clases:
  - Taxonómicas: es-un (clase/subclase), instancia-de (instancia/clase)
  - De usuario: El resto de relaciones

- Los slots describen las características del frame
- Poseen un conjunto de características (facets) que permiten establecer su semántica
  - Dominio, rango, cardinalidad, valor por omisión, ...
- Permiten definir procedimientos de manera que se realicen cálculos bajo ciertos eventos (demons)
- Los demons pueden ser:
  - `If-needed` (al consultar el slot)
  - `if-added` (al asignar valor al slot),
  - `if-removed` (al borrar el valor)
  - `if-modified` (al modificar el valor)
- Los demons no poseen parámetros
- Podemos declarar como afecta a los slots el mecanismo de herencia

Descripción de un slot (atributo):

Slot `<nombre>`
++ dominio (lista de frames)
++ rango `<tipo-simple>`
++ cardinalidad (1 o N)
valor (valor o lista de valores)
demons `<tipo-demon>`
accion `<nombre-accion>` / **función**`<nombre-funcion> devuelve <tipo>*`
herencia (por rels. taxonómicas: SI/NO; por rels. usuario: SI/NO)

- Los facets (propiedades) marcados con ++ son obligatorios en toda descripción de slot
- Las acciones/funciones asociadas a los demons de los slots no tienen parámetros. Usan la variable `F` como referencia implícita al frame al cual pertenece el slot que activa el demon
- Los demons de tipo `if-needed` solo pueden estar asociados a funciones
- Por defecto la herencia por rels. taxonómicas = SI, y por rels. usuario = NO
- Para acceder al valor de un slot usaremos la sintaxis `<nombre-frame>.<nombre-slot>`. Esta expresión será un valor o una lista dependiendo de la cardinalidad del slot.

Métodos: son acciones o funciones que permiten obtener información sobre el frame

- Estos métodos pueden invocarse desde frames abstractos (clases) o frames concretos (instancias)
- Pueden ser
  - heredables (permitimos invocarlos en los descendientes)
  - no heredables (exclusivos del frame)
- Pueden ser invocados con parámetros

Relaciones: permiten conectar los frames entre si para expresar su relación

- Se define su semántica mediante un conjunto de propiedades: Dominio, rango, cardinalidad, inversa, transitividad, composición, ...
- Se pueden establecer mecanismos procedimentales (demons) que tienen efecto ante ciertos eventos:
  - `If-added`: Si establecemos la relación entre instancias
  - `If-removed`: Si eliminamos la relación entre instancias
- Establecemos el comportamiento de la relación respecto al mecanismo de herencia (que slots permite heredar). Dado que las relaciones se definen bidireccionales, los slots se heredan en el sentido que corresponda (del frame en el que esta definido al que lo debe heredar)

Relación `<nombre>`

++ dominio (lista de frames)
++ `rango` (lista de frames)
++ `cardinalidad` (1 o N)
++ inversa `<nombre>` (cardinalidad: 1 o N)
transitiva SI/NO [por defecto es NO]
compuesta NO/`<descripción de la composición>` [por defecto es NO]
demons (`<tipo-demon> acción <nombre-acción>`
herencia (lista de slots) [por defecto es lista vacía]

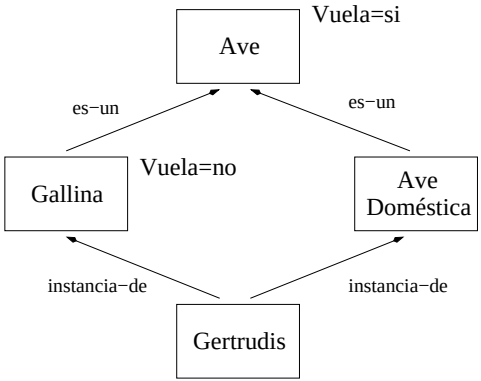
- Los descriptores marcados con ++ son obligatorios en toda descripción de relación
- Las acciones asociadas a los demons de las relaciones no tienen parámetros. Usan las variables `D` y `R` como referencia implícita al frame origen y destino, respectivamente, de la conexión que se está intentando añadir o eliminar entre los dos frames.

- La expresión `<nombre-frame>.<nombre-relación>` nos dará el frame (si la cardinalidad es 1) o la lista de frames (si la cardinalidad es N) con los cuales esta conectado a través de la relación `<nombre-relación>`
- Para consultar la cardinalidad se pude usar una funcion predefinida `card(<nombre-frame>.<nombre-relación>)`
- Relaciones predefinidas:
  - relación `es-un` (inversa: `tiene-por-subclase`) transitiva SI
  - relación `instancia-de` (inversa: `tiene-por-instancia`) composición: `instancia-de` $\otimes$ `es-un`

- Funciones booleanas predefinidas
  - `slot?(<frame>)` Nos dice si `<frame>` posee este slot o no (activando la herencia si hace falta)
  - `relación?(<frame>)`
    Nos dice si `<frame>` esta conectado con algún otro frame a través de la relación indicada por la función
  - `relación?(<frame-o>,<frame-d>)`
    Nos dice si existe una conexión entre `<frame-o>` y `<frame-d>` etiquetada con la relación indicada por la función

- La herencia es el mecanismo básico de deducción en los frames
- Permite obtener en un frame el valor o valores de un atributo o su definición a través de otro frame con el que está relacionado
- En el caso de las relaciones taxonómicas la herencia se da por omisión (se hereda la definición de los slots)
- En el resto de las relaciones se ha de establecer de manera explícita (se hereda el valor de los slots)
- Dado un frame es posible que la representación permita heredar un valor o definicion a través de múltiples relaciones o frames (Herencia múltiple)

- La herencia múltiple tiene sentido dependiendo de la semántica del slot heredado



- El **algoritmo de distancia inferencial** permite establecer cual es el frame del que se ha de heredar

1. Buscar el conjunto de frames que permiten heredar el valor del slot $\longrightarrow$ **Candidatos**
2. Eliminar de Candidatos todo frame que sea padre de otro de la lista
3. Si el número de candidatos es:
   - 3.1 $0 \longrightarrow$ No se puede heredar el slot
   - 3.2 $1 \longrightarrow$ Ese es el valor que buscamos
   - 3.3 $> 1 \longrightarrow$ Problema de herencia múltiple si la cardinalidad del slot no es N
4. En ocasiones un problema de herencia múltiple podría solucionarse por métodos procedimentales (demons)