Lesson 3: Knowledge Representation

Carlos García Martínez

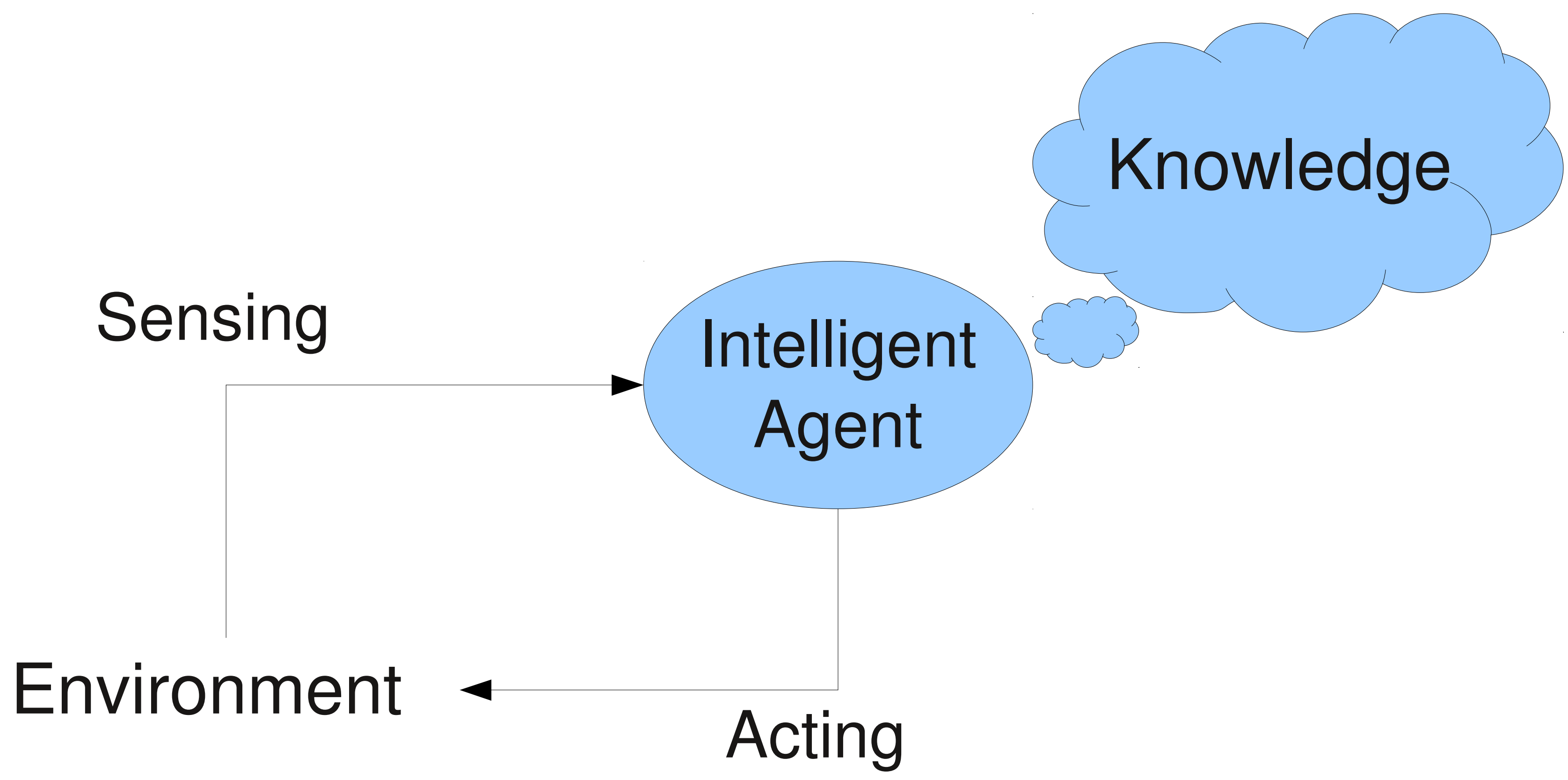
DIAN
DPTO. DE INFORMATICA Y ANALISIS NUMERICO

Outline

1. Introduction
2. Main properties in Knowledge Representation
3. Ontology Languages
4. Issues

1. Introducción

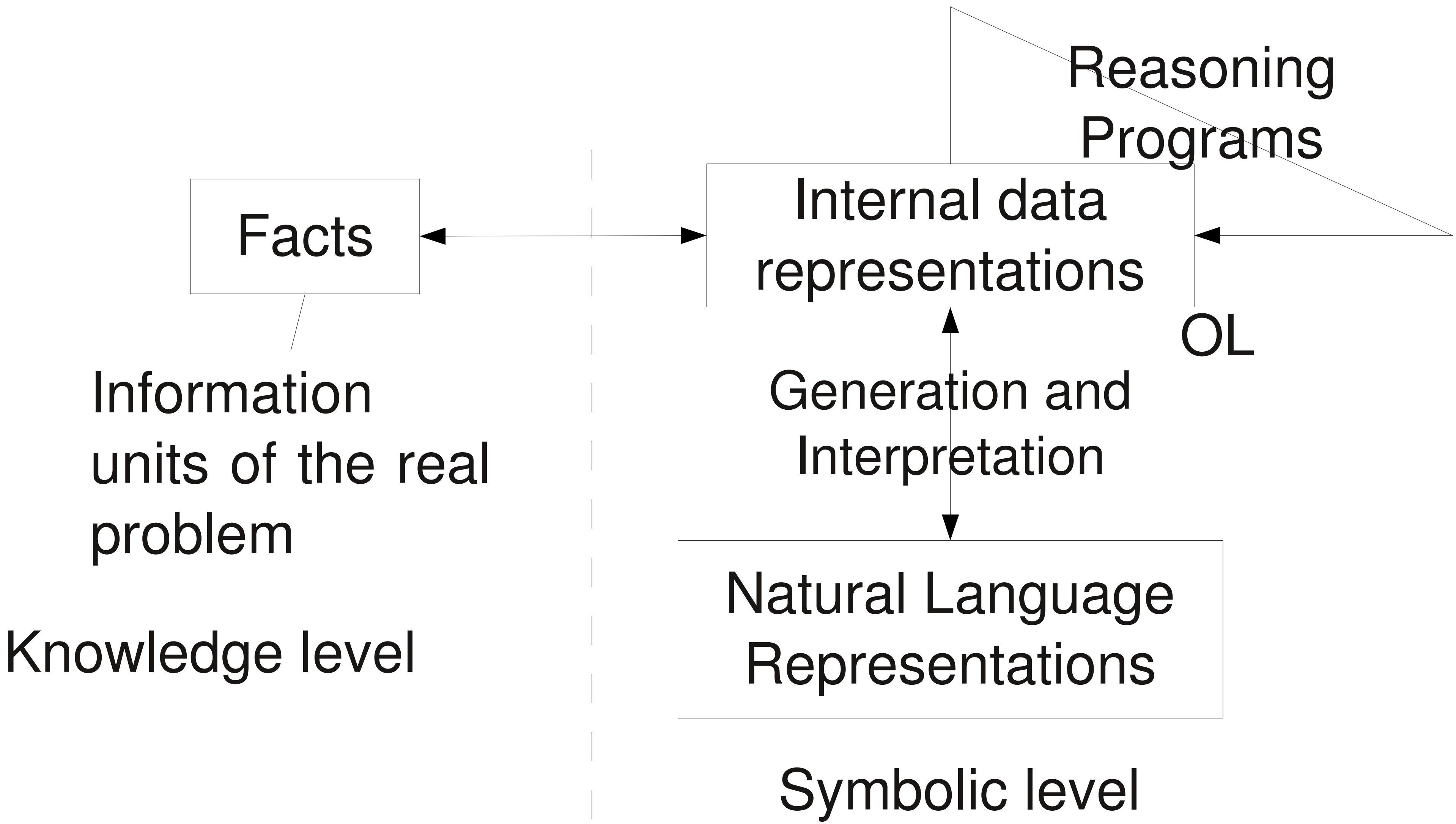
- Knowledge and Intelligence



1. Introduction

- Artificial Intelligence Problems require ***Ontology Languages*** (OLs) able to:
  - ***Represent*** the main knowledge of a problem in a proper way.
  - ***Manipulate*** that knowledge.
  - Make ***inferences*** easily (to obtain new information from the one maintained).

Facts and Representation



Example

?

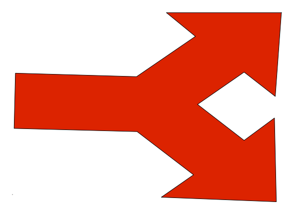
- Every dog has a tail
- All dogs have tail

?

$$\forall x\, dog(x) \rightarrow haveTail(x)$$

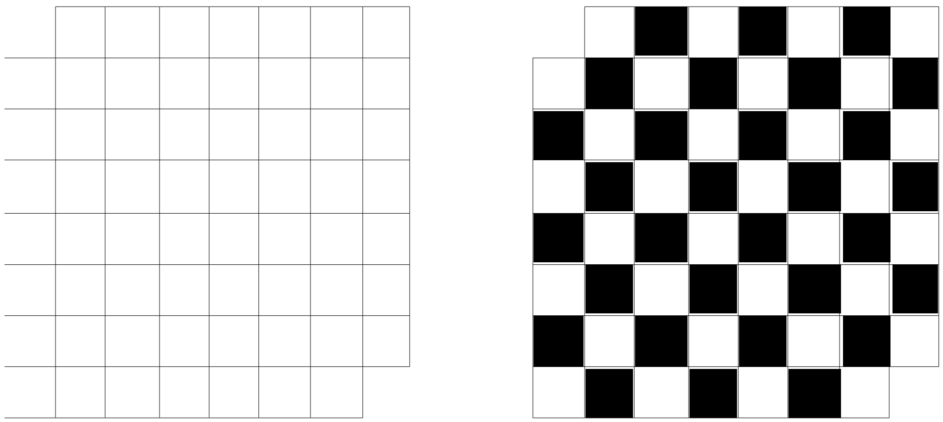
$$\forall x\, dog(x) \rightarrow have(x, tail)$$

$$\forall x\, dog(x) \rightarrow \exists y\, tail(y) \wedge have(x, y) \wedge (\forall z\, have(z, y) \rightarrow z = x) \wedge (\forall u\, tail(u) \wedge have(x, u) \rightarrow u = y)$$



Significance of the OL

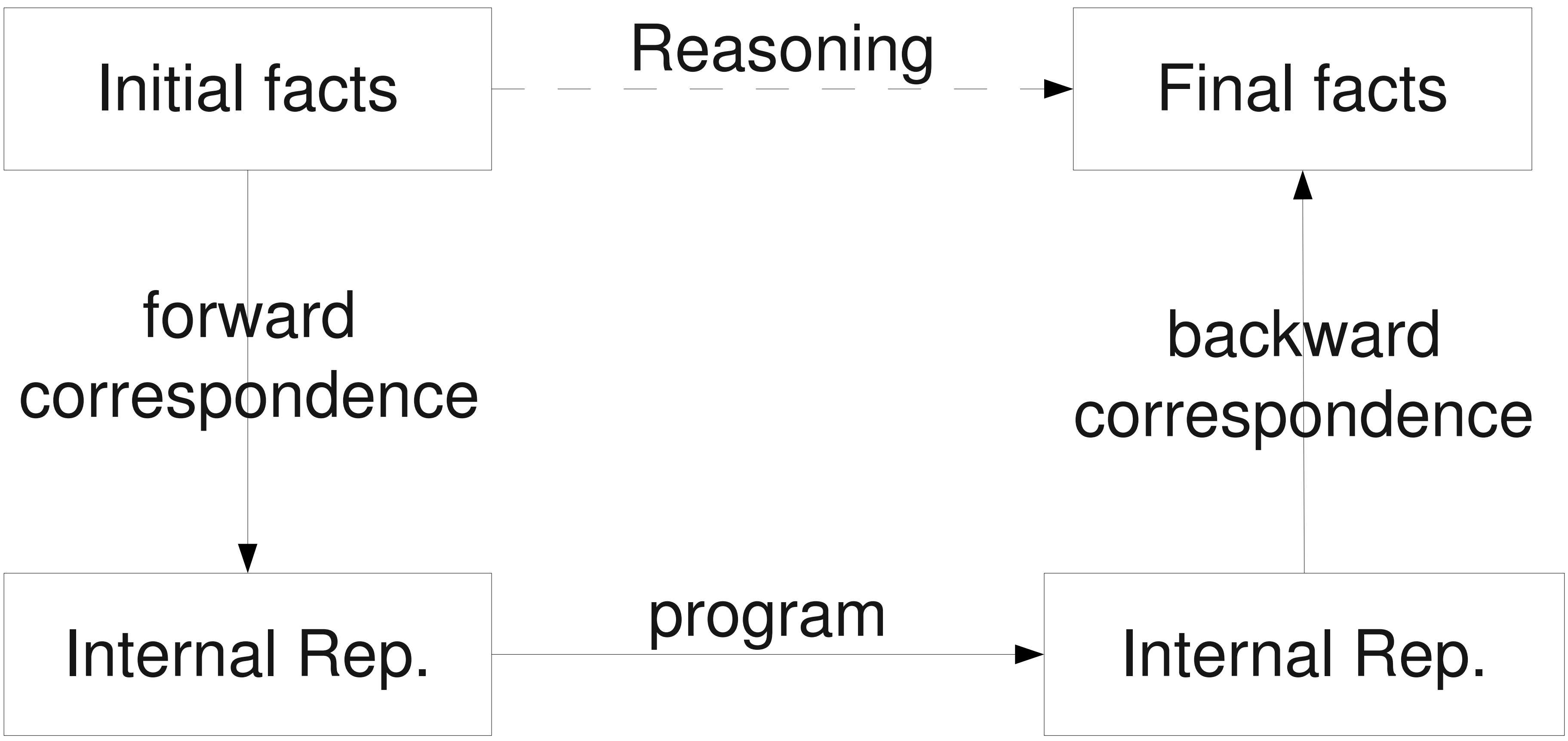
- A bad representation makes the reasoning program incorrect.
- A good representation makes the problem easier.
- An excellent representation may make the problem trivial



Number of black boxes = 30

Number of white boxes = 32

~~Artificial~~ Reasoning



Outline

1. Introduction
2. ***Main properties in Knowledge Representation***
3. Ontology Languages
4. Issues

Main Properties

- **Representational Adequacy**: it should be able to represent all relevant facts of the problem.
- **Inferential Adequacy**: it should be able to infer new facts from the existing ones.
- **Inferential efficiency**: it should be efficient in terms of time and space.
- **Acquisitional efficiency**: it should be able to incorporate new information easily.

There is no single OL, neither having all these properties, nor being suitable to any kind of knowledge.

Outline

1. Introduction
2. Main properties in Knowledge Representation
3. ***Ontology Languages***
4. Issues

Ontology Languages

3.1 Relational knowledge: Relational Data Bases

3.2 Inheritable knowledge: Objects and Taxonomic nets

3.3 Deductive Knowledge: Logic

3.4 Procedural Knowledge: Rule based Systems

3.1 Relational Knowledge

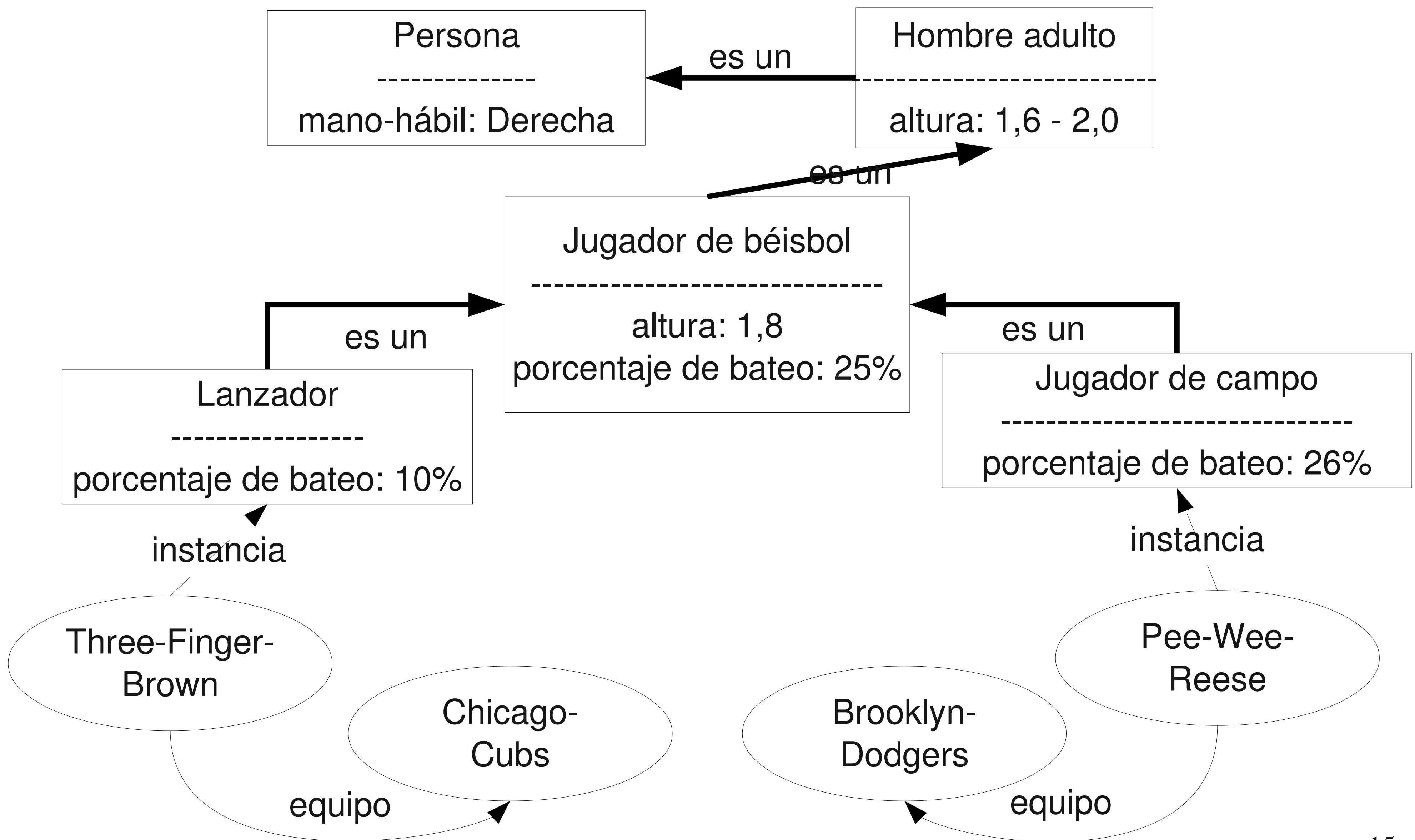
- It is commonly used by other OLs
- ***It lacks deductive competence***.

Nombre	***Capital***	***Moneda***	***Idioma***
Alemania	Berlín	Marco	Alemán
Austria	Viena	Chelín	Alemán
España	Madrid	Peseta	Español
Nicaragua	Managua	Córdoba	Español
Portugal	Lisboa	Escudo	Portugués

3.2 Inheritable Knowledge

- It lies on the ***inheritance*** concept: individuals inherit attributes from their classes and superclasses.
- Classes are organized hierarchically
- Examples: Taxonomic nets and objects (frames)

Objects, Taxonomic net



3.3 Deductive Knowledge

- Knowledge is represented by means of *elements*, *predicates* and ***logic relations***:

$$\forall\ x\, \text{Simple\_Search\_Problem}(x) \rightarrow \exists\, y\, \text{Search\_Technique}(y) \wedge Solve(y, x)$$

$$\text{Simple\_Search\_Problem}(viajante)$$

- Logic makes inferences possible:
  - **Forward search**: it starts from the initial facts and it ends finding the conclusions.
  - **Backward search**: it starts from the conclusions and looks for initial facts that let them be true.
- Problem: Problem representation uses to be complex.

3.4 Procedural Knowledge

- Knowledge is represented by means of ***programs***. It often lacks deductive competence and acquisition efficiency.
- In AI, ***Rule based Systems*** is the technique most commonly applied:
  - IF <condition> THEN <action> (or <consequent>)
- ***Difficulties***:
  - It is not easy to differentiate between declarative and procedural knowledge.
  - To debug these systems is usually and impossible task.

Outline

1. Introduction

2. Main properties in Knowledge Representation

3. Ontology Languages

4. Issues

4. Issues

4.1 Significant Attributes

4.2 Relations between attributes

4.3 Representation *granularity*

4.4 Set of objects

4.5 Forward correspondence: Search for a suitable structure

4.6 Frame problem

4.1 Significant Attributes

- Some attributes are more important than other. This importance should be stated:
  - ***Instance*** attribute: It states that an element belongs to a specific class.
  - ***IS-A*** attribute: It establishes an inheritance relation between classes.
- Logic does not emphasize significant attributes.
- Taxonomic nets (objects) do.

4.2 Relations between attributes

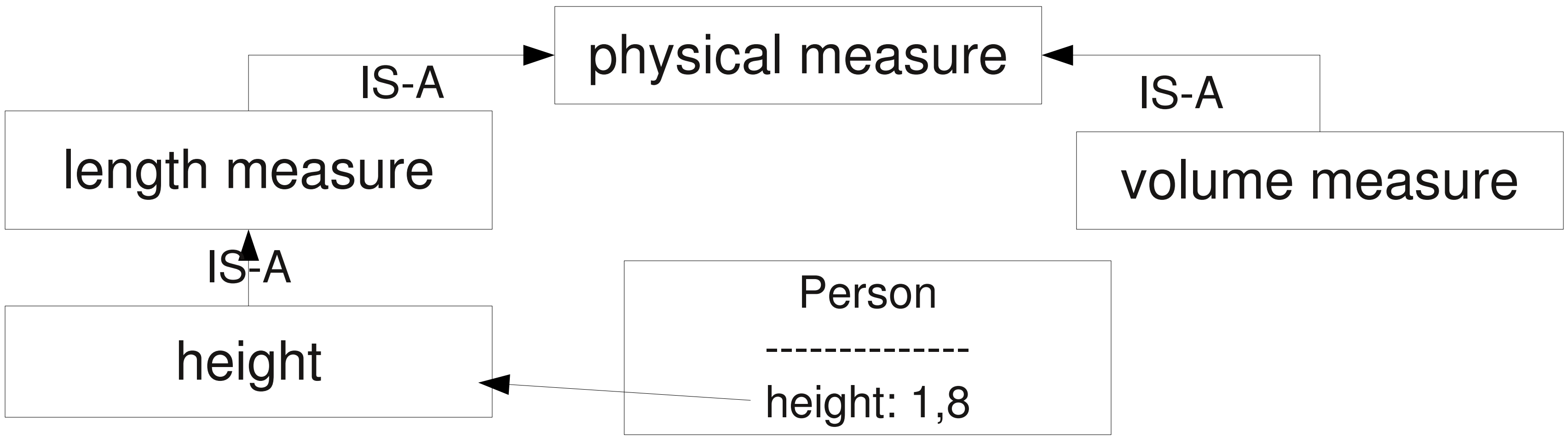
- *Objects' attributes may be as well objects*.
- ***Issues***:
  - Attribute-object relations.
  - Attribute IS-A hierarchy.
  - Reasoning about values.
  - Single value attributes.

Attribute-object relations (Inverses)

- *Objects' attributes may be as well objects*.
- Where should we put the relation?:
  - Relation is implemented in only one object. The relation is not symmetric.
  - Relation is implemented in both objects. Drawback: information duplication (modifications).
  - Relation is implemented independently.

Attribute IS-A hierarchy

- Attributes may be structured in a IS-A hierarchy



- They may inherit restrictions, values computation...
- Storage and search may be optimized.

Reasoning about values

- An Intelligent System may reason about some attribute values, though they are not present.
- It may consider:
  - Attribute type.
  - Restrictions.
  - Interactions with other attributes.
  - Forward and backward rules.

Single Value Attributes

- Examples: height, weight...
- Multivalued examples: language, car...
- Some OLs (Logic) have problems dealing with single value attributes.
- Solutions:
  - To introduce ***temporal intervals***.
  - To substitute the old value.

4.3 Granularity

- **Granularity**: Detail level at which information must be represented.
- To chose the granularity is not easy:
  - ***Coarse grain***:
    - Low detail level
    - Easy to manage
    - Difficult to obtain
  - ***Fine grain***:
    - High detail
    - Difficult to manage
    - Easy to obtain
  - Examples:
    - "Dogs have a tail"
    - "Juan broke the window"

4.4 Set of objects

- Some sets of objects have their own properties. How do we represent their properties?
- How to represent set of objects:
  - First: give it a name.
  - ***Extensional deffinition,*** indicating its members
  - ***Intensional deffinition,*** describing the members' characteristics:
    - It allows us to define infinite sets (prime numbers)
    - It lets us to define sets with unknown members.
    - It lets us to define parametrised set (relatives)
    - There may be more than one definition for the same set (planets of the solar system)

4.5 Forward correspondence: Search for a suitable structure

- Perform an ***Initial Selection***
- ***Fill in appropriate details***.
- ***Find better structure*** if first choice is bad.
- What to do if none of the available structures is appropriate?
- When to create and remember a new structure

Perform an Initial Selection

- Several approaches:
  - Index the structures by significant words (ambiguity).
  - Relate each major concept as a pointer to all of the potential scripts.
  - Locate one major clue in the description

Revising the choice when neccessary

- Fill in appropriated details in the selected structure
- ***If the selected structure does not match***:
  - Select fragments of the current structure to find another structure.
  - Make an excuse for the shortcomings of the first and continue to use it
  - Refer to specific stored links between structures
  - Use IS-A links

4.6 The Frame Problem

- How to represent efficiently sequences of problem states that arise from a search process.
- If there are many facts that do not change they are repeated across every state:
  - A lot of time will be spent copying these nodes
  - A lot of memory is wasted

Frame problem: A Solution

- ***Frame Axiom*** describe all the things that do not change when an operator is applied in state n.
- All things that do change are mentioned as part of the operator itself.
- Two ways:
  - Do not modify the initial state
  - Modify the initial state

Lesson 3:
Knowledge Representation

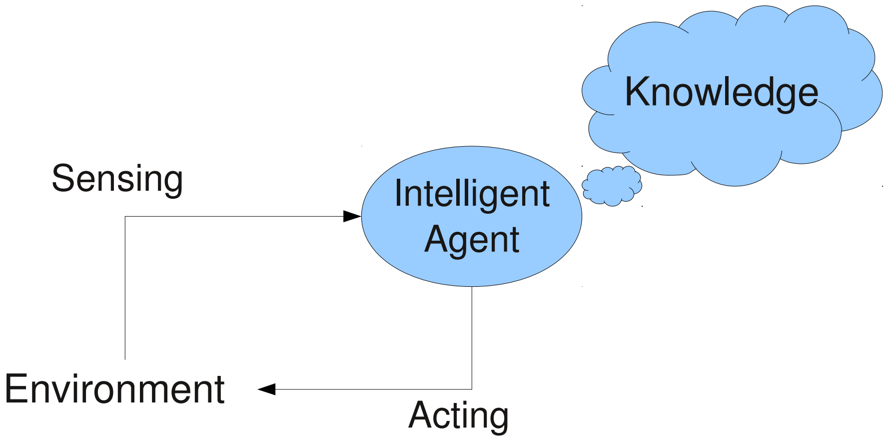
Carlos García Martínez

Outline

1. Introduction
2. Main properties in Knowledge Representation
3. Ontology Languages
4. Issues

1. Introducción

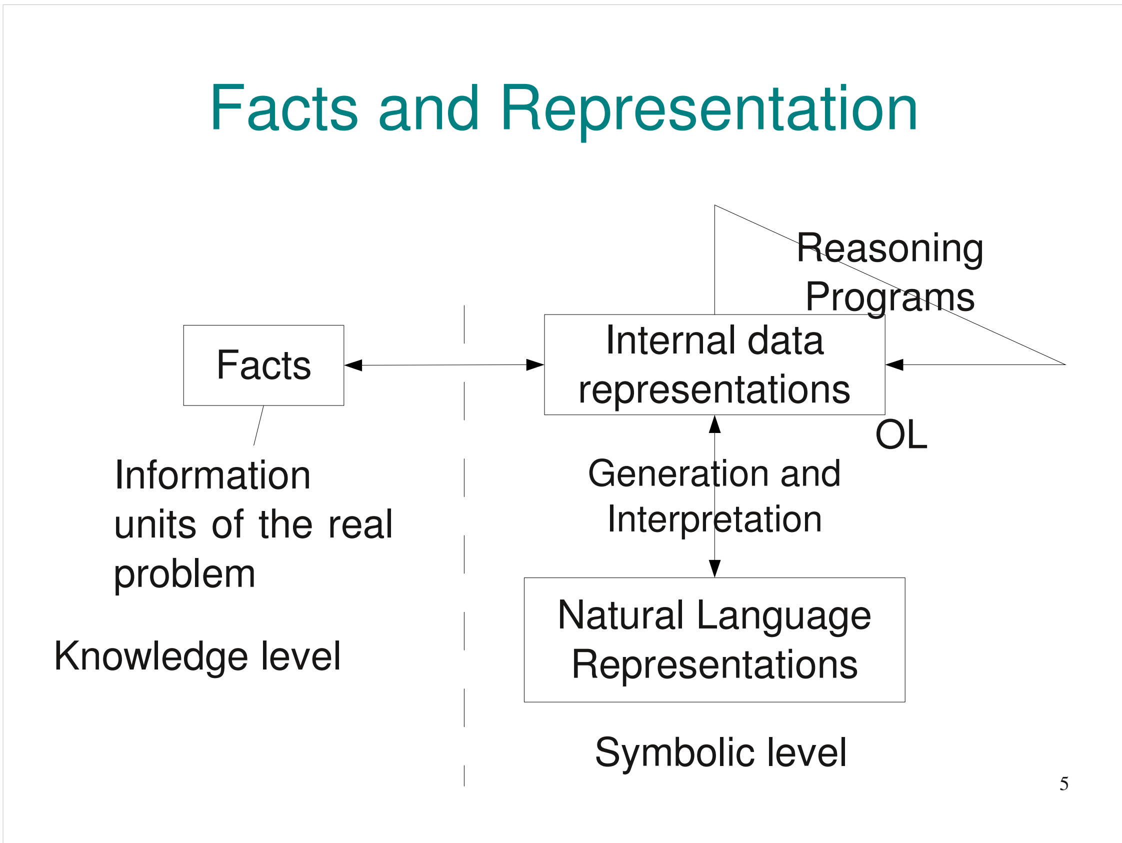
- Knowledge and Intelligence



1. Introduction

- Artificial Intelligence Problems require ***Ontology Languages*** (OLs) able to:
  - ***Represent*** the main knowledge of a problem in a proper way.
  - ***Manipulate*** that knowledge.
  - Make ***inferences*** easily (to obtain new information from the one maintained).

Los problemas de la IA necesitan manejar una cantidad respetable de conocimiento. Para ello necesitan **sistemas de representación del conocimiento**



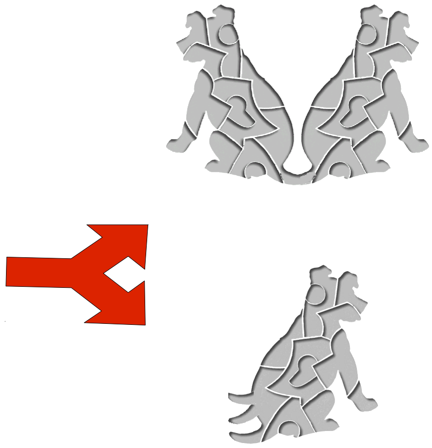
Es importante conocer que el programa no maneja los hechos reales, sino una representación del mismo

Example

?

- Every dog has a tail
- All dogs have tail

?



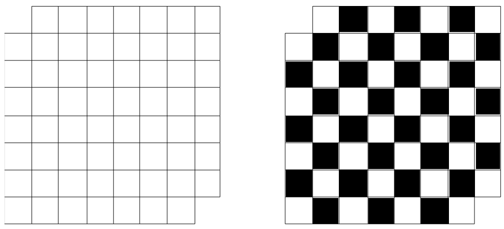
$$\forall\ x\, dog(x) \rightarrow haveTail(x)$$

$$\forall\ x\, dog(x) \rightarrow have(x, tail)$$

$$\forall\ x\, dog(x) \rightarrow \exists y\, tail(y) \wedge have(x, y) \wedge (\forall\ z\, have(z, y) \rightarrow z = x) \wedge (\forall\ u\, tail(u) \wedge have(x, u) \rightarrow u = y)$$

Significance of the OL

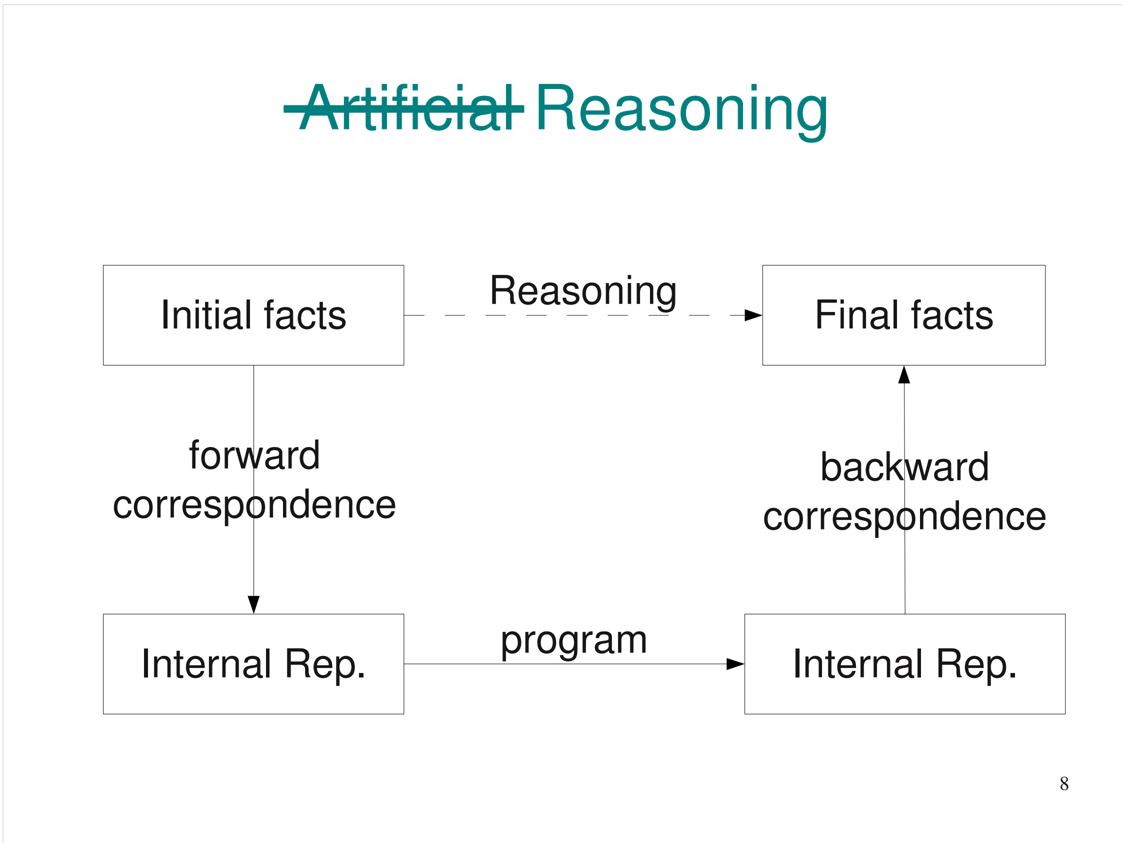
- A bad representation makes the reasoning program incorrect.
- A good representation makes the problem easier.
- An excellent representation may make the problem trivial



Number of black boxes = 30

Number of white boxes = 32

Sea un tablero de 8x8 al que se le han quitado las dos esquinas opuestas ¿se puede cubrir con fichas de dominó?



El razonamiento artificial consiste entonces en: dados unos hechos iniciales....

Los errores en este razonamiento pueden deberse a tres fuentes, la representación de los hechos, el programa o la interpretación de las representaciones finales.

Outline

1. Introduction
2. ***Main properties in Knowledge Representation***
3. Ontology Languages
4. Issues

Main Properties

- **Representational Adequacy**: it should be able to represent all relevant facts of the problem.
- **Inferential Adequacy**: it should be able to infer new facts from the existing ones.
- **Inferential efficiency**: it should be efficient in terms of time and space.
- **Acquisitional efficiency**: it should be able to incorporate new information easily.

There is no single OL, neither having all these properties, nor being suitable to any kind of knowledge.

Al añadir conocimiento podemos estropear la base completa

Outline

1. Introduction
2. Main properties in Knowledge Representation
3. ***Ontology Languages***
4. Issues

Ontology Languages

3.1 Relational knowledge: Relational Data Bases

3.2 Inheritable knowledge: Objects and Taxonomic nets

3.3 Deductive Knowledge: Logic

3.4 Procedural Knowledge: Rule based Systems

3.1 Relational Knowledge

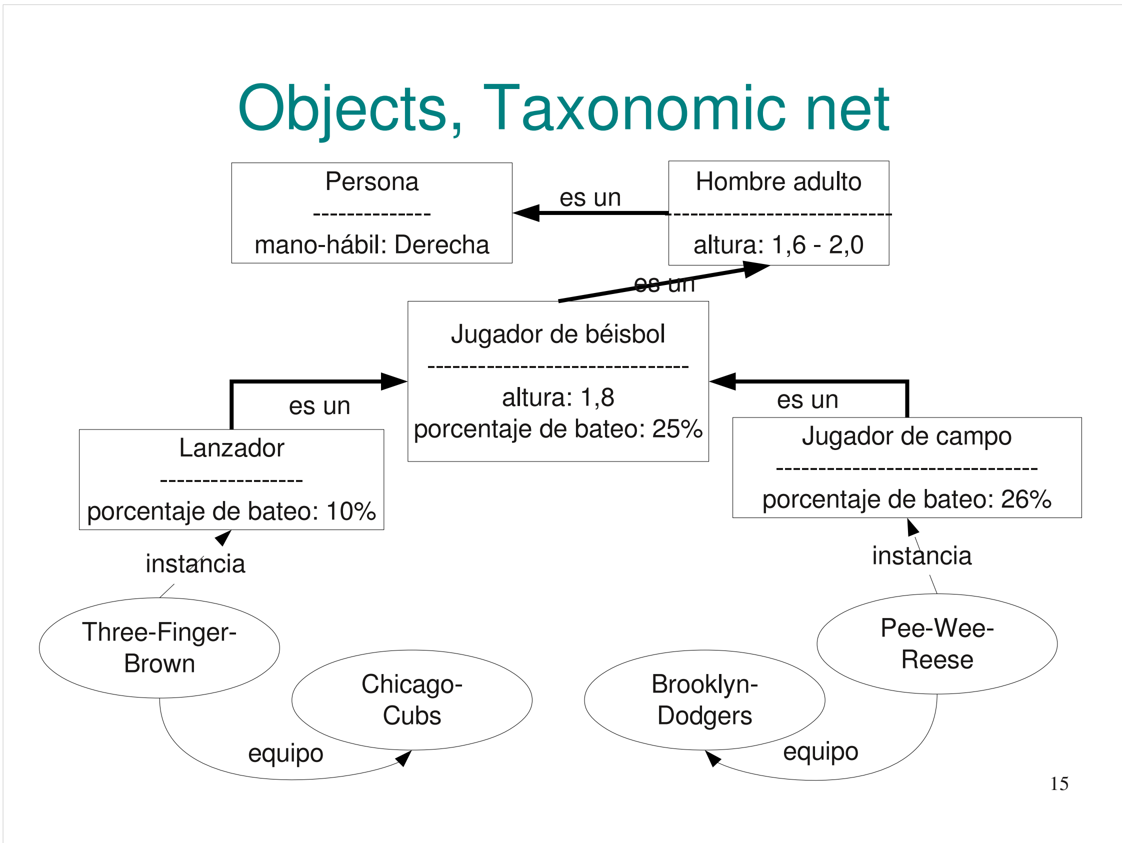
- It is commonly used by other OLs
- ***It lacks deductive competence***.

Nombre	*Capital*	*Moneda*	*Idioma*
Alemania	Berlín	Marco	Alemán
Austria	Viena	Chelín	Alemán
España	Madrid	Peseta	Español
Nicaragua	Managua	Córdoba	Español
Portugal	Lisboa	Escudo	Portugués

Un ejemplo, no puede decir cuantos países hablan portugués

3.2 Inheritable Knowledge

- It lies on the ***inheritance*** concept: individuals inherit attributes from their classes and superclasses.
- Classes are organized hierarchically
- Examples: Taxonomic nets and objects (frames)



Describir el algoritmo de herencia de propiedades

3.3 Deductive Knowledge

- Knowledge is represented by means of *elements*, *predicates* and ***logic relations***:

$$\forall\ x\, \text{Simple\_Search\_Problem}(x) \rightarrow \exists\, y\, \text{Search\_Technique}(y) \wedge Solve(y, x)$$

$$\text{Simple\_Search\_Problem}(viajante)$$

- Logic makes inferences possible:
  - **Forward search**: it starts from the initial facts and it ends finding the conclusions.
  - **Backward search**: it starts from the conclusions and looks for initial facts that let them be true.
- Problem: Problem representation uses to be complex.

Aunque el conocimiento heredable resulta ser muy potente, en algunos casos no es suficiente y se necesita conocimiento deductivo

3.4 Procedural Knowledge

- Knowledge is represented by means of ***programs***. It often lacks deductive competence and acquisition efficiency.
- In AI, ***Rule based Systems*** is the technique most commonly applied:
  - IF <condition> THEN <action> (or <consequent>)
- ***Difficulties***:
  - It is not easy to differentiate between declarative and procedural knowledge.
  - To debug these systems is usually and impossible task.

Código para ordenar, contar o calcular el máximo de un vector

Outline

1. Introduction
2. Main properties in Knowledge Representation
3. Ontology Languages
4. ***Issues***

4. Issues

4.1 Significant Attributes

4.2 Relations between attributes

4.3 Representation *granularity*

4.4 Set of objects

4.5 Forward correspondence: Search for a suitable structure

4.6 Frame problem

4.1 Significant Attributes

- Some attributes are more important than other. This importance should be stated**:**
  - ***Instance*** attribute: It states that an element belongs to a specific class.
  - ***IS-A*** attribute: It establishes an inheritance relation between classes.
- Logic does not emphasize significant attributes.
- Taxonomic nets (objects) do.

Poner un ejemplo, persona, perro, color de los ojos, pelo....

4.2 Relations between attributes

- *Objects' attributes may be as well objects*.
- ***Issues***:
  - Attribute-object relations.
  - Attribute IS-A hierarchy.
  - Reasoning about values.
  - Single value attributes.

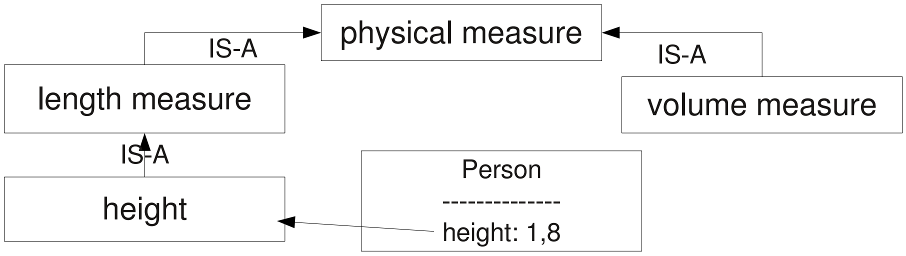
Attribute-object relations (Inverses)

- *Objects' attributes may be as well objects*.
- Where should we put the relation?:
  - Relation is implemented in only one object. The relation is not symmetric.
  - Relation is implemented in both objects. Drawback: information duplication (modifications).
  - Relation is implemented independently.

Poner un ejemplo de coche y propietario

Attribute IS-A hierarchy

- Attributes may be structured in a IS-A hierarchy



- They may inherit restrictions, values computation...
- Storage and search may be optimized.

Reasoning about values

- An Intelligent System may reason about some attribute values, though they are not present.
- It may consider:
  - Attribute type.
  - Restrictions.
  - Interactions with other attributes.
  - Forward and backward rules.

Single Value Attributes

- Examples: height, weight...
- Multivalued examples: language, car...
- Some OLs (Logic) have problems dealing with single value attributes.
- Solutions:
  - To introduce ***temporal intervals***.
  - To substitute the old value.

4.3 Granularity

- **Granularity**: Detail level at which information must be represented.
- To chose the granularity is not easy:
  - ***Coarse grain***:
    - Low detail level
    - Easy to manage
    - Difficult to obtain
  - ***Fine grain***:
    - High detail
    - Difficult to manage
    - Easy to obtain
  - Examples:
    - “Dogs have a tail”
    - “Juan broke the window”

4.4 Set of objects

- Some sets of objects have their own properties. How do we represent their properties?
- How to represent set of objects:
  - First: give it a name.
  - ***Extensional deffinition,*** indicating its members
  - ***Intensional deffinition,*** describing the members' characteristics:
    - It allows us to define infinite sets (prime numbers)
    - It lets us to define sets with unknown members.
    - It lets us to define parametrised set (relatives)
    - There may be more than one definition for the same set (planets of the solar system)

Algunos conjuntos tienen sus própias propiedades: el conjunto de elefantes es grande.

El primer guión: asignar un nombre, se refiere a crear una clase. Entonces todas las instancias heredan las propiedades que se definen, pero en realidad, sólo eso no crea ningún conjunto.

4.5 Forward correspondence: Search for a suitable structure

- Perform an ***Initial Selection***
- ***Fill in appropriate details***.
- ***Find better structure*** if first choice is bad.
- What to do if none of the available structures is appropriate?
- When to create and remember a new structure

Perform an Initial Selection

- Several approaches:
  - Index the structures by significant words (ambiguity).
  - Relate each major concept as a pointer to all of the potential scripts.
  - Locate one major clue in the description

29

Primer guión: volar: 1) volar a la Palma, 2)volar la caja fuerte

Segundo guión: a cada concepto un conjunto de estructuras, y después seleccionar la intersección: Juan viajó con la patente

Tercero: relativizar la importancia de los conceptos

Revising the choice when neccessary

- Fill in appropriated details in the selected structure
- ***If the selected structure does not match***:
  - Select fragments of the current structure to find another structure.
  - Make an excuse for the shortcomings of the first and continue to use it
  - Refer to specific stored links between structures
  - Use IS-A links

Primer guión, seleccionar algunas propiedades de la estructura actual y buscar esas propiedades en otra estructura

Segundo guión: seguir con la misma estructura haciendo anotaciones en los problemas encontrados

Tercero y cuarto: usar los enlaces entre estructuras

4.6 The Frame Problem

- How to represent efficiently sequences of problem states that arise from a search process.
- If there are many facts that do not change they are repeated across every state:
  - A lot of time will be spent copying these nodes
  - A lot of memory is wasted

En los problemas de búsqueda se crean muchos nodos, cada uno de aplicar una regla de transición a un nodo anterior. Si hay muchos hechos que no cambian, éstos se duplican en los nodos

Frame problem: A Solution

- ***Frame Axiom*** describe all the things that do not change when an operator is applied in state n.
- All things that do change are mentioned as part of the operator itself.
- Two ways:
  - Do not modify the initial state
  - Modify the initial state

Si se utiliza el axioma del marco, todo lo que no aparece en el operador se considera que permanece igual.

Entonces, hay dos formas de aplicarlo, no modificando el estuado inicial y en cada nodo apuntar sólo lo que cambia (coste de generar el nodo) (y coste de deshacerlo)

O modificar el estado y apuntar en cada nodo qué es necesario deshacer para volver al estado anterior. (coste de generar y coste de deshacer)