Knowledge Representation

(In general we are least aware of what our minds do best)

Representation and Mapping

The **knowledge representation problem** concerns the mismatch between human and computer 'memory’.

i.e. how to encode knowledge so that it is a faithful reflection of the expert's knowledge and can be manipulated by computer.

Representation and Mapping

- In the representation there are two different entities that must be considered:-
  - Facts: truths in some relevant world. These are things that we want to represent
  - Representation of facts in some chosen formalism. These are things that can actually be manipulated.

Representation and Mapping

- Structuring of these entities can be done in two levels:
  - The *knowledge level* at which facts are described
  - The *symbol level* at which representation of some objects at the knowledge-level are defined in terms of symbols that can be manipulated by programs.

Representation and Mapping

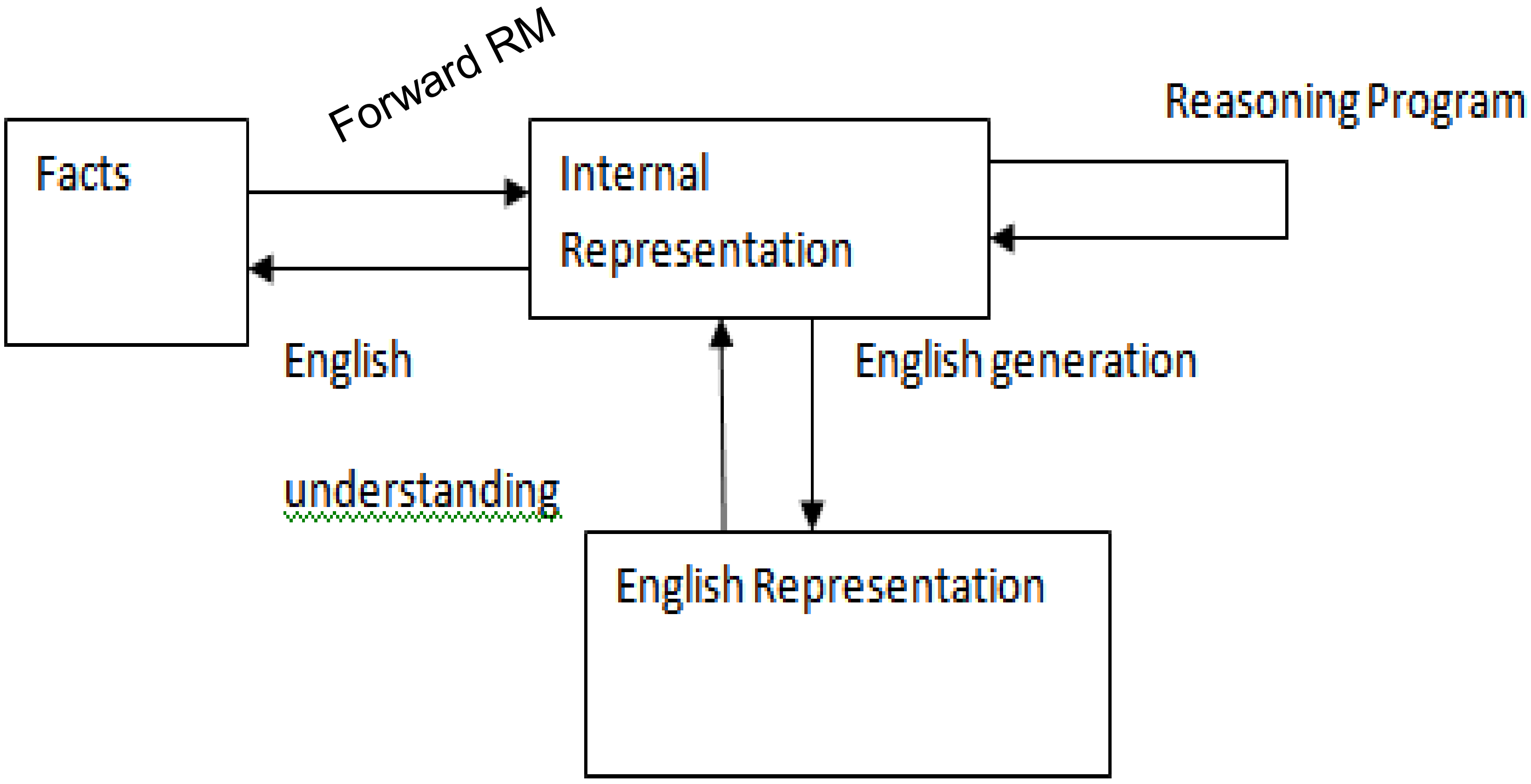


Figure 1: Mappings between Facts and Representation

Representation and Mapping

- Our main goal is to focus on:
  - Facts
  - Representation

- As well as the two-way mappings that must exist between the two as shown in the figure 1 above.
- The links in the figure are called *representation mappings*.

Representation and Mapping

- Representation mappings, there are:-
  - *Forward representation* which maps from facts to representation.
  - *Backward representation* which maps the other way.
- One representation of facts concerns with natural language (particularly English) sentences.

Representation and Mapping

- Regardless of the representation for facts that we use in a program, we may also need to be concerned with an English representation of those facts that in order to facilitate getting information into and out of the system.
- We must also have mapping functions from English sentences to the representation we are actually going to use and from it back to sentences as shown in the figure 1.

Representation and Mapping

- Example we can use mathematical logic as the representation formalism. Consider the English sentences below

  Spot is a dog

- This fact can also be represented in logic as follows:-

  dog(Spot)

Representation and Mapping

- Suppose also we have a logical representation of the fact: *all dogs have tails* as shown below:

$$\forall x : dog(x) \rightarrow hastail(x)$$

- Using the deductive mechanisms of the logic, we may generate the new representation object

$$hastail(Spot)$$

Representation and Mapping

- Using an appropriate backward mapping function we could then generate the English sentence:

  Spot has a tail

- Or we could make use this representation of new fact to cause us to take some appropriate action or to derive representation of additional facts.

Approach to Knowledge Representation

- The following properties should be possessed by a knowledge representation system.
- **Representational Adequacy** -- the ability to represent the required knowledge;
- **Inferential Adequacy** - the ability to manipulate the knowledge represented to produce new knowledge corresponding to that inferred from the original;
- **Inferential Efficiency** - the ability to direct the inferential mechanisms into the most productive directions by storing appropriate guides;
- **Acquisitional Efficiency** - the ability to acquire new knowledge using automatic methods wherever possible rather than reliance on human intervention.

Knowledge Representation Schemes

There are four types of Knowledge representation :
Relational, Inheritable, Inferential, and Declarative/Procedural.

◊ **Relational Knowledge :**

− provides a framework to compare two objects based on equivalent attributes.
− any instance in which two different objects are compared is a relational type of knowledge.

◊ **Inheritable Knowledge**

− is obtained from associated objects.
− it prescribes a structure in which new objects are created which may inherit all or a subset of attributes from existing objects.

Knowledge Representation Schemes

◊ **Inferential Knowledge**

− is inferred from objects through relations among objects.

− e.g., a word alone is a simple syntax, but with the help of other words in phrase the reader may infer more from a word; this inference within linguistic is called semantics.

◊ **Declarative Knowledge**

− a statement in which knowledge is specified, but the use to which that knowledge is to be put is not given.

− e.g. laws, people's name; these are facts which can stand alone, not dependent on other knowledge;

Procedural Knowledge

− a representation in which the control information, to use the knowledge, is embedded in the knowledge itself.

− e.g. computer programs, directions, and recipes; these indicate specific use or implementation

Simple Relational Knowledge

- Simplest way to represent knowledge is set of tables used in database

Database Systems

- They are used in representing Simple Relation Knowledge which is in declarative facts and can be said as a set of relations of the same sort within database systems.
- Figure 2 shows an example of such systems

Player	Height	Weight	Bats-Thrown
Hank Aaaron	6-0	180	Right-Right
Willie Mays	5-10	170	Right-Right
Babe Ruth	6-2	215	Left-Left
Ted William	6-3	205	Left-Right

Simple Relation Knowledge

- Other example representation in DB Systems

Example:

```
person record = {     name : max 20 characters
                       age   : 3 digits in range 000-120
                       sex   : male or female
            marital status : married, bachelor, spinster,
                               divorced, widowed, or engaged
    first names of children : up to 10 names each max
                               15 characters }
```

Simple Relation Knowledge

- Knowledge represented in this form may serve as the input to more powerful inference engines.

Inheritable Knowledge : Semantic Nets

- Semantic nets are useful for representing inheritable knowledge.
- Inheritable knowledge is the most useful for *property inheritance,* in which elements of specific classes inherits attributes and values from more general classes in which they are included.
- Frames also do play a big role in representing this knowledge

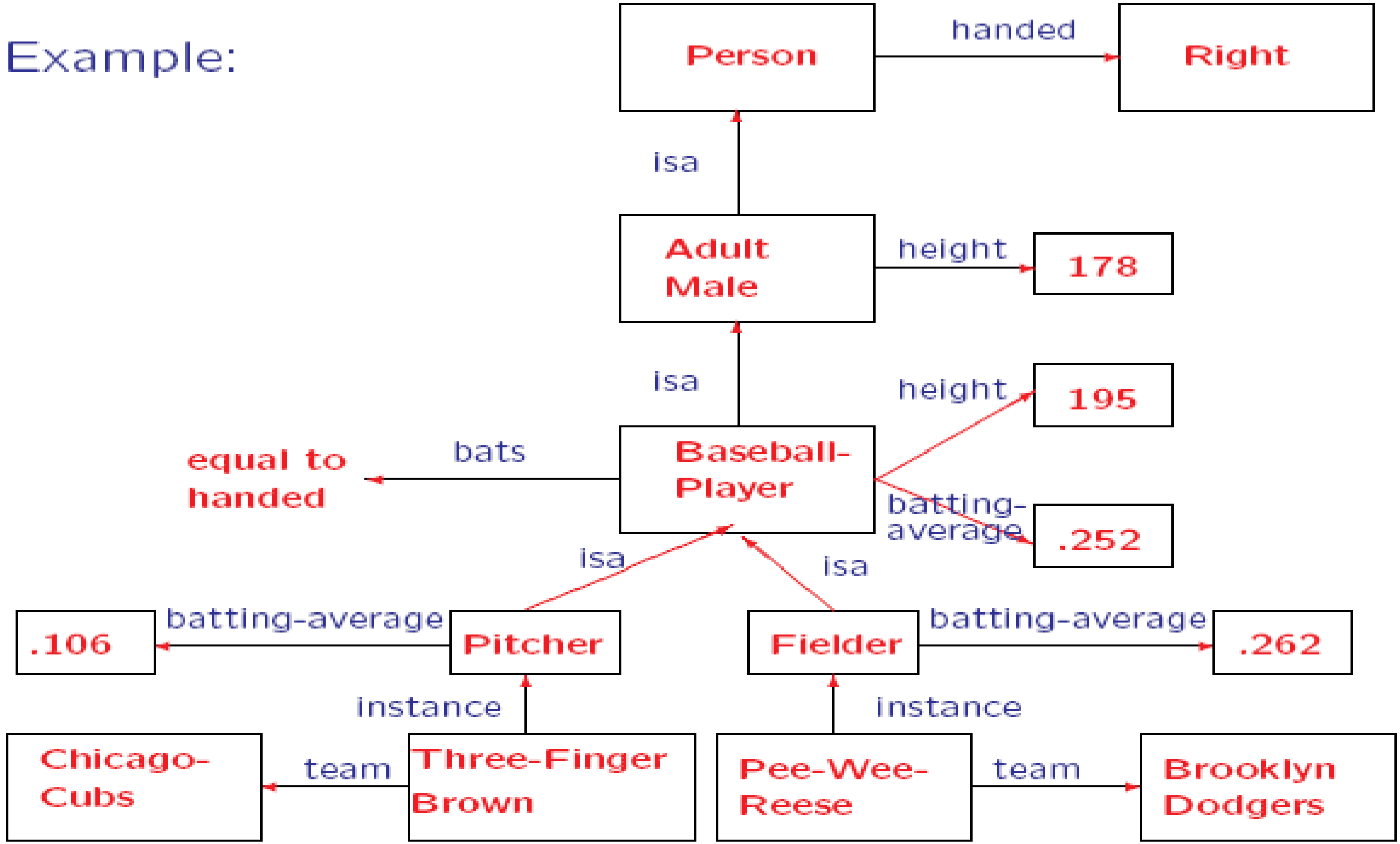
Semantic Nets

- In order to support property inheritance, objects must be organised into classes and classes must be arranged in a generalisation hierarchy.

- Figure 2 below shows some additional baseball knowledge inserted into a structure that is so arranged.

Semantic Nets

Example:



Semantic Nets

- *Isa* hierarchy is normally used in semantic nets/frames.
- Lines represent attributes. Boxed nodes represent objects and values of attributes of objects.
- Correct deduction from Figure 2could be:

  height of Three-Finger Brown is 195cm.
- An incorrect deduction could be:

  height of Three-Finger Brown is 178cm

Semantic Nets

- The structure shown in the figure 2 is a *slot-and-filler* structure.
- It may be also called a *semantic network* or a collection of frames.
- It will be discussed more in the next session

Inferential Knowledge

- This knowledge generates new information from the given information.
- This new information does not require further data gathering form source, but does require analysis of the given information to generate new knowledge.

Example :

- given a set of relations and values, one may infer other values or relations.
- a predicate logic (a mathematical deduction) is used to infer from a set of attributes.
- inference through predicate logic uses a set of logical operations to relate individual data.
- the symbols used for the logic operations are :

  " $\rightarrow$ " (implication), " $\neg$ " (not), " $\vee$ " (or), " $\wedge$ " (and),
  " $\forall$ " (for all), " $\exists$ " (there exists).

Examples of predicate logic statements :

1. *"Wonder"* is a name of a dog :

   dog (wonder)

2. All dogs belong to the class of animals : ∀ **x :**

   dog (x) → animal(x)

3. All animals either live on land or in water:

   ∀ x : animal(x) → live (x,land) V live (x, water)

From these three statements we can infer that :

" Wonder **lives either on land or on water** ***."***

Declarative/Procedural Knowledge

- Differences between Declarative/Procedural knowledge is not very clear.
- **Declarative knowledge :**

  Here, the knowledge is based on declarative facts about *axioms* and *domains* .

  – axioms are assumed to be true unless a counter example is found to invalidate them.

  – domains represent the physical world and the perceived functionality.

  – axiom and domains thus simply exists and serve as declarative statements that can stand alone.

- **Procedural knowledge**:

  Here, the knowledge is a mapping process between domains that specify **“what to do when”** and the representation is of **“how to make it”** rather than *“what it is”* .

The procedural knowledge :

– may have inferential efficiency, but no inferential adequacy and acquisitional efficiency.

– are represented as small programs that know how to do specific things, how to proceed.

Example : A parser in a natural language has the knowledge that a noun phrase may contain articles, adjectives and nouns. It thus accordingly call routines that know how to process articles, adjectives and nouns.

Propositional Logic

Propositional Logic is a formal system in which knowledge is represented as propositions.

Further, these propositions can be joined in various ways using logical operators.

These expressions can then be interpreted as truth-preserving inference rules that can be used to derive new knowledge from the old, or test the existing knowledge.

Propositional Logic

- Propositional logic is fundamental to all logic.
- Propositional logic is also called Propositional calculus, Sentential calculus, or Boolean algebra.
- Propositional logic tells the ways of joining and/or modifying entire propositions, statements or sentences to form more complicated propositions, statements or sentences, as well as the logical relationships and properties that are derived from the methods of combining or altering statements.

First, let’s introduce the proposition.

A *proposition* is a statement, or a simple declarative sentence.

For example, “the book is expensive” is a proposition.

Note that a definition of truth is not assigned to this proposition; it can be either true or false.

In terms of binary logic, this proposition could be false in Cairo, but true in England. But a proposition always has a truth value.

So, for any proposition, we can define the true-value based on a truth table (see the following figure).

This simply says that for any given proposition, it can be either true or false.

P
T
F

We can also negate our proposition to transform it into the opposite truth value.

For example, if P (our proposition) is “the book is expensive,” then ~P is “the book is not expensive.”

Propositions can also be combined to create compound propositions.

The first, called a *conjunction*, is true only if both of the conjuncts are true (P and Q).

The second called a *disjunction*, is true if at least one of the disjuncts are true (P or Q).

The truth tables for these are shown in the following figure. These are obviously the AND and OR truth tables from Boolean logic.

P	Q	P • Q
F	F	F
F	T	F
T	F	F
T	T	T

P	Q	P v Q
F	F	F
F	T	T
T	F	T
T	T	F

Limitations of Propositional Logic

While propositional logic is useful, it cannot represent general-purpose logic in a compact and summary way.

For example, a formula with N variables has 2^N different interpretations. It also doesn’t support changes in the knowledge base easily.

Truth values of propositions can also be problematic, for example; consider the compound proposition below.

This is considered true where P -> Q is true when P is false and Q is false, see the truth table of Modus Ponens).

If dogs can fly, then cats can fly.

Both statements are obviously false, and further, there’s no connection between the two. But from the standpoint of propositional logic, they are syntactically correct. A major problem with propositional logic is that entire propositions are represented as a single symbol.

Issues in Knowledge Representation

- There are several issues that must be considered when representing various kinds of real-world knowledge.
  - Important Attributes
  - Relationship among Attributes
  - Inverses
  - Existence in an Isa hierarchy
  - Technique for reasoning about values
  - Single-valued attributes