Knowledge Representation

What is Knowledge?

The Chambers 20th Century Dictionary provides as good a definition as any:

knowledge, assured belief; that which is known; information; …

In order to solve the complex problems encountered in AI, one generally needs a large amount of knowledge, and suitable mechanisms for representing and manipulating all that knowledge.
Knowledge can take many forms. Some simple examples are**:**

John has an umbrella
It is raining
An umbrella stops you getting wet when it's raining
An umbrella will only stop you getting wet if it is used properly
Umbrellas are not so useful when it is very windy

So, how should an AI agent store and manipulate knowledge like this?

What is a Knowledge Representation?

The object of a ***knowledge representation*** is to express knowledge in a computer tractable form, so that it can be used to enable our AI agents to perform well.

A ***knowledge representation language*** is defined by two aspects:

1. **Syntax** The syntax of a language defines which configurations of the components of the language constitute valid sentences.
2. **Semantics** The semantics defines which facts in the world the sentences refer to, and hence the statement about the world that each sentence makes.

This is a very general idea, and not restricted to natural language.

Suppose the language is arithmetic, then

‘ x’, ‘ $\geq$’ and ‘ y’ are *components* (or symbols or words) of the language

the *syntax* says that ‘ $x \geq y$’ is a valid sentence in the language, but ‘ $\geq \geq$ x y’ is not the *semantics* say that ‘ $x \geq y$’ is false if y is bigger than x, and true otherwise

Requirements of a Knowledge Representation

A good knowledge representation system for any particular domain should possess the following properties:

1. **Representational Adequacy** – the ability to represent all the different kinds of knowledge that might be needed in that domain.
2. **Inferential Adequacy** –the ability to manipulate the representational structures to derive new structures (corresponding to new knowledge) from existing structures.
3. **Inferential Efficiency** – the ability to incorporate additional information into the knowledge structure which can be used to focus the attention of the inference mechanisms in the most promising directions.
4. **Acquisitional Efficiency** – the ability to acquire new information easily. Ideally the agent should be able to control its own knowledge acquisition, but direct insertion of information by a 'knowledge engineer' would be acceptable.

Finding a system that optimises these for all possible domains is not going to be feasible.

Practical Aspects of Good Representations

In practice, the theoretical requirements for good knowledge representations can usually be achieved by dealing appropriately with a number of practical requirements:

1. The representations need to be ***complete*** – so that everything that could possibly need to be represented, can easily be represented.
2. They must be ***computable*** – implementable with standard computing procedures.
3. They should make the important ***objects*** and ***relations*** explicit and accessible – so that it is easy to see what is going on, and how the various components interact.
4. They should ***suppress irrelevant detail*** – so that rarely used details don't introduce unnecessary complications, but are still available when needed.
5. They should expose any natural ***constraints*** – so that it is easy to express how one object or relation influences another.
6. They should be ***transparent*** – so you can easily understand what is being said.
7. The implementation needs to be ***concise*** and ***fast*** – so that information can be stored, retrieved and manipulated rapidly.

Components of a Good Representation

For analysis purposes it is useful to be able to break any knowledge representation down into their four fundamental components:

1. The ***lexical*** part – that determines which symbols or words are used in the representation's ***vocabulary***.

2. The ***structural*** or ***syntactic*** part – that describes the***constraints*** on how the symbols can be arranged, i.e. a grammar.

3. The ***semantic*** part – that establishes a way of associating***real world meanings*** with the representations.

4. The ***procedural*** part – that specifies the access procedures that enables ways of ***creating*** and ***modifying*** representations and ***answering questions*** using them, i.e. how we generate and compute things with the representation.

In the following we shall look at these in more detail for some specific examples.

Knowledge Representation in Natural Language

Humans usually use ***natural language*** (English, Spanish, Chinese, etc.) to represent knowledge, so why not use that to represent knowledge in our AI systems?

Advantages of Natural Language

1. It is extremely expressive – we can express virtually everything in natural language (real world situations, pictures, symbols, ideas, emotions, reasoning, …).
2. Most humans use it most of the time as their knowledge representation of choice (how many text books are not written in natural language?).

Disadvantages

1. Both the syntax and semantics are very complex and not fully understood.
2. There is little uniformity in the structure of sentences.
3. It is often ambiguous – in fact, it is*usually* ambiguous.

Database Systems

Simple ***databases*** are commonly used to good effect in Computer Science. They can be use to store and manipulate virtually any kind of information.

For example, the database may consist of a number of simple records stored in ASCII format:

```
Person record = {  name : max 32 characters
                age : 3 digits in range 000-
                127 sex : male or female
                marital status : single, engaged, married, divorced,
                widowed employer : company code of 3 characters
                children's names : up to 8 names each with max 16 characters
               }
```

Generally, the records can have any number of fields, containing whatever information we need, in any format, together with any appropriate links between them

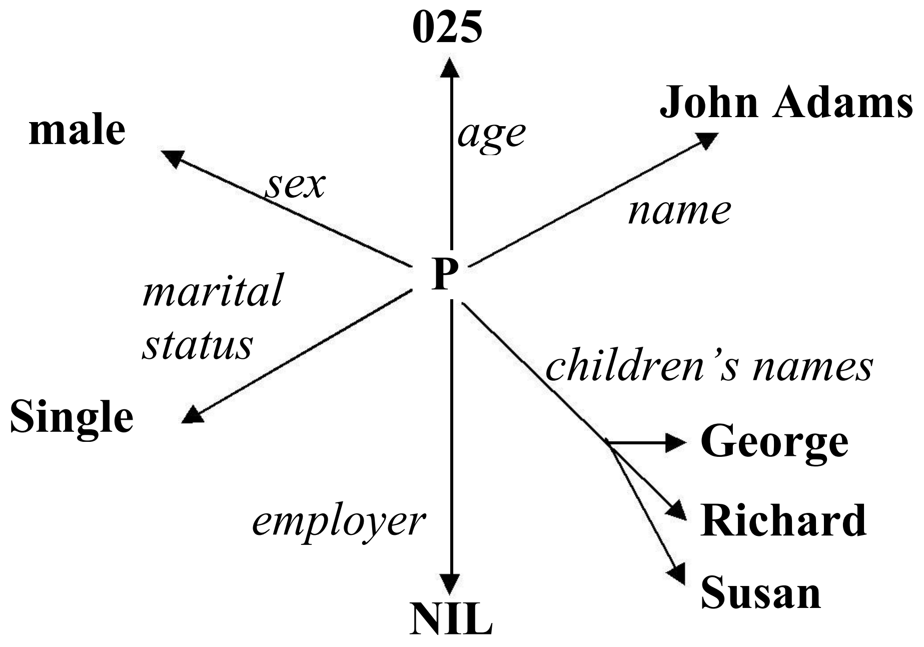
Instances in a Database System

Information in a database can be displayed in a variety of ways, for example:

A Record Structure

John Adams
025
male
single
NIL
George
Richard
Susan

A Directed Graph



But storage and display are not enough – we also need to manipulate the knowledge.

Manipulations of a Database System

We can construct sentences in an appropriate language, for example:

“marital_status(John Adams) is single”	CORRECT
“marital_status(John Adams) is divorced”	INCORRECT SEMANTICS
“marital_status(025) is male”	INCORRECT SYNTAX

We can also generate relations, for example:

R1: Employment

.	.
NIL	John Adams
NIL	Fred Smith
NIL	Sam Patel
NTL	Jo McNeal
.	.

R2: Parent/Child

.	.
John Adams	George
John Adams	Richard
John Adams	Susan
Karen Adams	Richard
.	.

Databases as a Knowledge Representation

Traditional database systems are clearly very powerful, but for AI systems they are rather limited. The important issues are:

Advantages

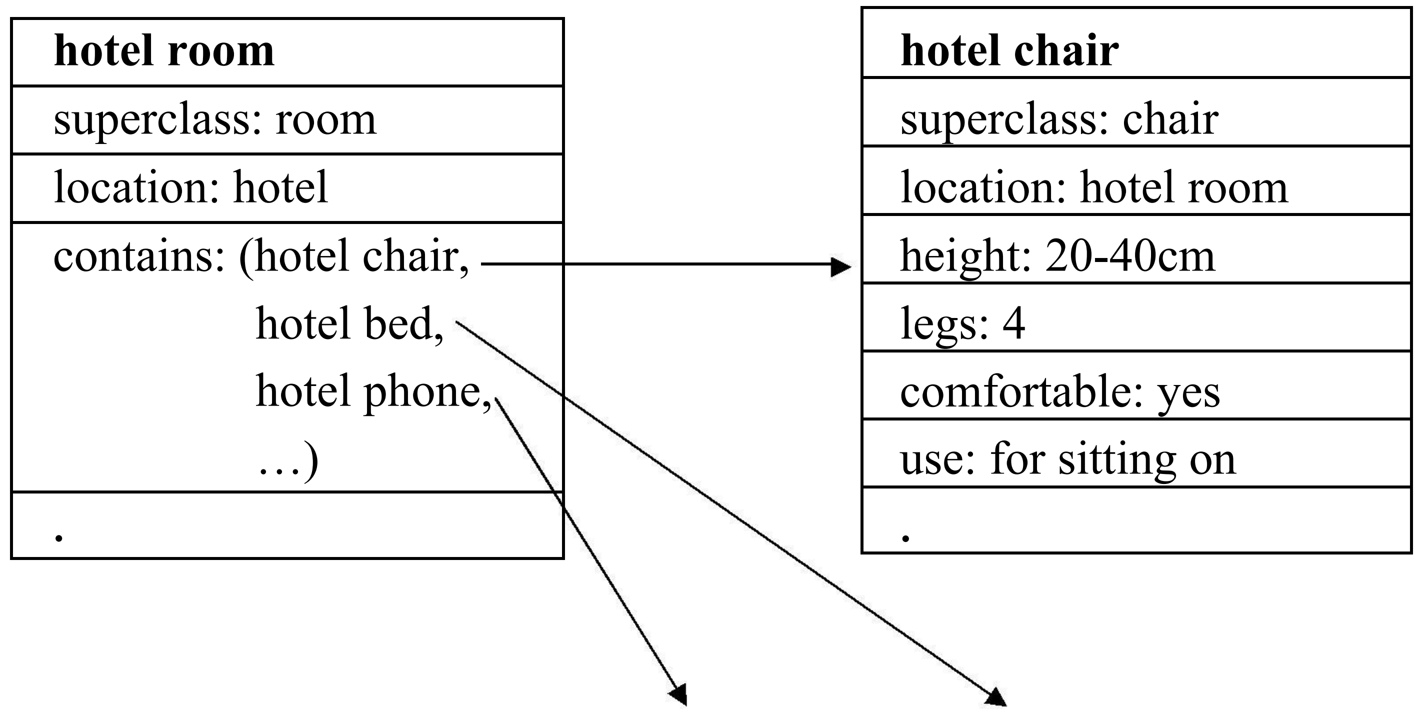
1. Databases are well suited to efficiently representing and processing large amounts of data (and derivation from a database is virtually independent of its size).
2. We can build on traditional database systems to process more complex and more powerful representational devices (e.g. frames).

Disadvantages

1. Only simple aspects of the problem domain can be accommodated.
2. We can represent *entities*, and *relationships* between entities, but not much more.
3. Reasoning is very simple – basically the only reasoning possible is simple lookup, and we usually need more sophisticated processing than that.

Frame Based Systems

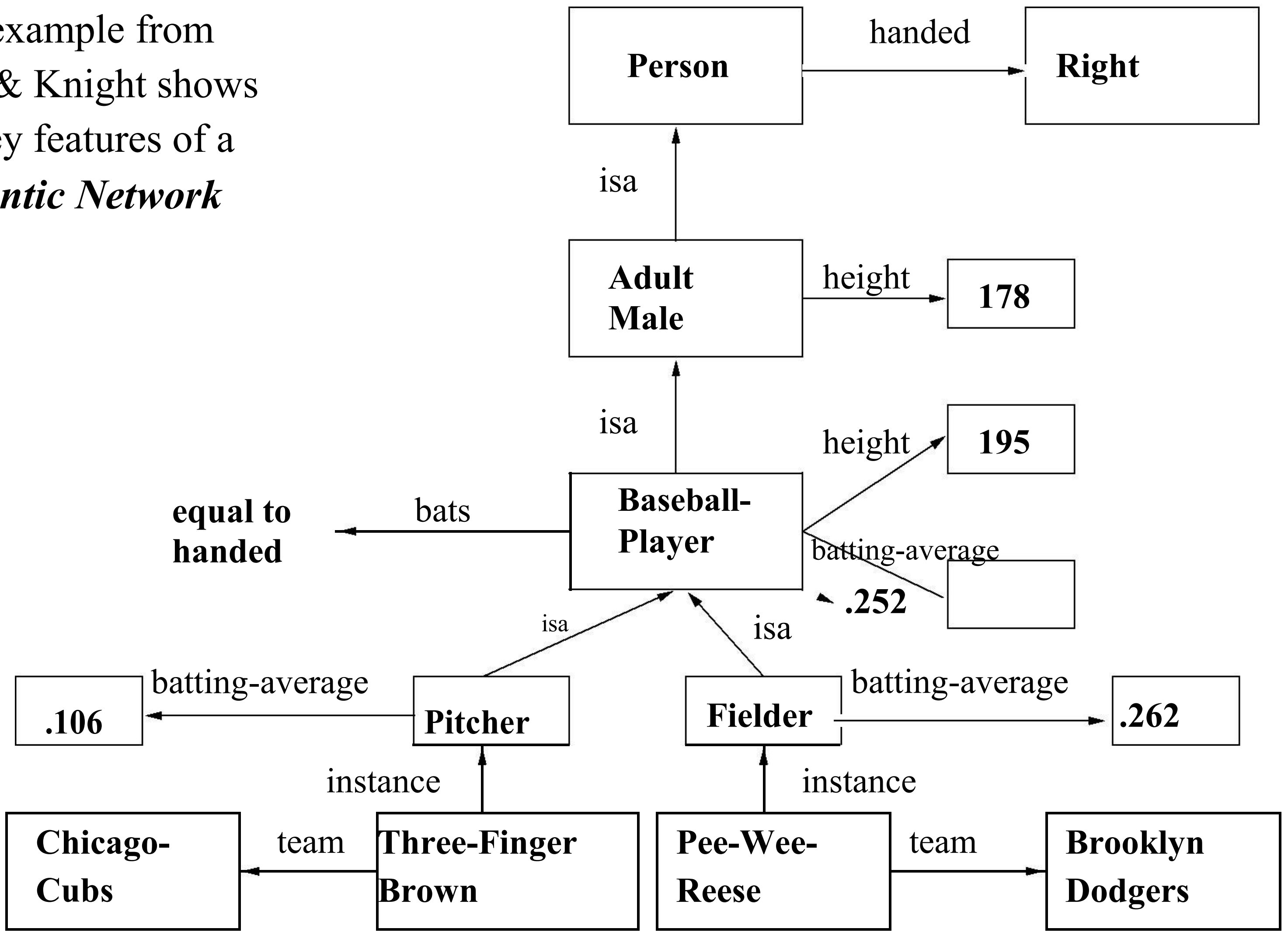
We can extend database records to ***Frames*** consisting of slots which can be filled by values, or procedures for calculating values, or pointers to other frames. For example:



Generally a whole hierarchy of frames is used to represent the required domain. It is often helpful to represent the structure of that hierarchy as a Semantic Network.

Semantic Networks

This example from Rich & Knight shows the key features of a ***Semantic Network***



First Order Logic

The syntax and semantics of **first order logic** will be covered in detail elsewhere.

Some typical sentences in first order logic are:

1. $\text{man(William)} \vee \text{woman(Susan)}$
2. married(William, Susan)
3. $\forall x \exists y[\text{person}(x) \Rightarrow \text{has\_mother}(x,y)]$
4. $\forall x \forall y[[\text{parents}(x,y) \wedge \text{man}(x)] \Rightarrow \neg \text{man}(y)$

The language consists of constants {William, Susan, etc.}, variables {x, y, etc.}, functions/predicates {Married(x,y), person(x), etc.}, and the logic symbols:

Logic	$\vee$	$\wedge$	$\Rightarrow$	$\neg$	$\forall$	$\exists$
Nat. Lang.	or	and	implies	not	for all	there exists

We can also manipulate the logic representations to generate new knowledge.

First Order Logic as a Knowledge Representation

We can combine sentences by the ‘rules of logic’ to produce new sentences, e.g.

$$\frac{\begin{array}{l}\neg\text{man}(\text{Chris}) \\ \neg\text{man}(x) \Rightarrow \text{woman}(x)\end{array}}{\text{woman}(\text{Chris})}$$

As a knowledge representation, first order logic has pros and cons:

Advantages

1. It is very expressive.
2. It has unambiguous syntax and semantics.

Disadvantage

1. There is no generally efficient procedure for processing knowledge

Rule Based Systems

A **rule based system** consists of:

1. A **database management system** for handling the domain specific facts.
2. A **rule set** for representing the knowledge structure/relations.
3. A **rule interpreter** to carry out the problem solving.

A typical rule set might be:

R1. IF Raining $\wedge$ Outside(x) $\wedge$ HasUmbrella(x) THEN UseUmbrella(x)

R2. IF Raining $\wedge$ Outside(x) $\wedge$ $\neg$HasUmbrella(x) THEN GetWet(x)

R3. IF GetsWet(x) THEN CatchCold(x)

R4. IF Sunny $\wedge$ Outside(x) THEN GetSunBurnt(x)

It should be easy enough to set up an appropriate database management system.

Rule based Inference

If we have a knowledge base consisting of **fact*s*** and **rules**, and a rule interpreter to match the rule conditions against the facts, and a means for extracting the rules, then we can derive new knowledge. For example, using the above rule set:

Suppose we have three initial facts: Raining, Outside(John), ¬HasUmbrella(John).

Then only the rule R2 with ‘x = John’ matches the facts, so we are able to infer GetsWet(John). This means we now have four facts in our knowledge base: Raining, Outside(John), ¬HasUmbrella(John), GetsWet(John).

Then R3 with ‘x = John’ matches the facts, so we can also infer CatchesCold(John), and end up with five facts: the initial three, GetsWet(John), CatchesCold(John).

Note that there is no way we can end up with GetsSunTan(John).

The process of deriving new facts from given facts is called ***inference.***

Rule Based Systems as a Knowledge Representation

We can see that rule based systems have many of the properties required of a knowledge representation. However, as always, there are pros and cons:

Advantages

1. These systems are very expressive.
2. The rules lead to a degree of modularity.
3. We can easily introduce procedures for handling certainty factors, and this leads to the possibility of probabilistic reasoning.

Disadvantage

1. There is a lack of precise semantics for the rules.
2. The systems are not always efficient.

We shall study rule based systems in detail in a series of lectures later in this module.

Which Knowledge Representation is Best?

We have now seen what is required of a knowledge representation and taken a brief tour through a number of the most obviously plausible styles of knowledge representation.

There are clearly many more representational formalisms that might be useful. For a start, we have only really considered ***symbolic*** representations. There also exist ***non-symbolic*** (e.g. pictorial) representations. So-called ***sub-symbolic*** representations are also possible (e.g. as one finds in the activation patterns of neural network systems).