Knowledge Representation: Notes on Chapter 4

S. Sheerazuddin

SSN College of Engineering, Chennai

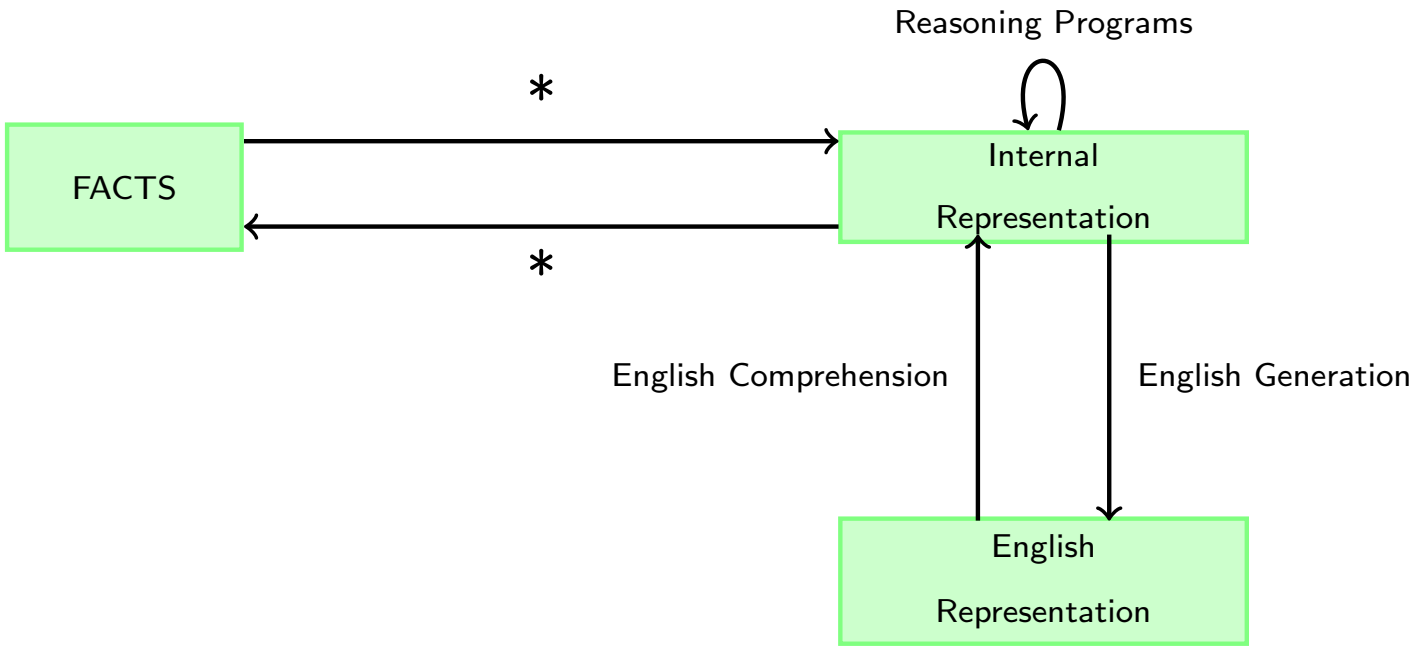
Knowledge Representation

In context of knowledge representation, we have to deal with two kinds of entities:

- **Facts:** Facts are truths in some relevant world. These are the things that we represent. Facts are described at the **knowledge level.**
- **Representation of Facts:** Facts are represented in some chosen formalism (say, logic), using symbols (The **symbol level**). These are the things that we manipulate.

The most common representation of facts is via **natural language**.

Mappings between Facts and Representations



The (forward & backward) mappings are not functions but many-to-one relations.

Approaches to Knowledge Representation

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

- **Representational Adequacy:** The system should be able to represent all kinds of knowledge in the domain.
- **Inferential Adequacy:** The system should be able to manipulate representational structures (symbols) to derive new knowledge from old knowledge.
- **Inferential Efficiency:** The system should be able to direct the inference mechanism to promising areas by adding information to the knowledge structure.
- **Acquisitional Efficiency:** The ability to acquire new information easily. Ideally, the program should control knowledge acquisition.

Different Kinds of Knowledge Representation

- **Relational Knowledge:** Represent declarative facts as a set of relations, as in database systems.
- **Inheritable Knowledge:** Represent declarative facts as a semantic network, essentially a hierarchical description of attributes and associated values of objects in the knowledge base, and extract information using a property inheritance algorithm.
- **Inferential Knowledge:** Represent declarative facts as a set of well-formed formulas in some logic and use rules of inference to derive new facts.
- **Procedural Knowledge:** Inferential mechanism is incorporated within the knowledge representation structure, as in PROLOG and LISP etc.

Issues in Knowledge Representation

- Are there any **attributes** of objects so **basic** that they occur in almost every problem domain ? If yes, what are they ?
- Are there any important **relationships** that exist among attributes of objects ?
- At what **level** should knowledge be represented ? Is there a good **set of primitives** into which all knowledge can be broken down ? Is it helpful to use such primitives ?
- How should **set of objects** be represented ?
- Given a large amount of knowledge stored in a database, how can **relevant** parts be **accessed** when they are needed ?

Issues in Knowledge Representation

- Two attributes of general significance: **instance** (class membership) and **isa** (class inclusion).
- Important relationships among attributes are: **inverse** and **isa hierarchy** (specialization hierarchy).
- Low level of knowledge representation needs less inference to create new knowledge or reasoning about actions consistent with knowledge. But needs large storage and more inference to just create the representation and finding appropriate primitives.
- Three ways to represent sets of objects:
  - By **name**. Does not provide any information about the set it represents.
  - By **extension**. List the elements of the set.
  - By **intension**. Provide a rule that, when a particular object is evaluated, returns true or false on whether the object is in the set or not.

The Frame Problem

The problem of representing facts that change and those that do not is known as **the frame problem**.

- Consider a domain comprising two objects **door** (can be open or close) and **light** (can be switched on or off).
- This domain is represented using predicates, if the conditions can change over time: $open(t)$ & $on(t)$.
- Suppose the knowledge base of the domain comprises the following well formed formulas:
  - Initial condition: $\neg open(0)$ and $\neg on(0)$.
  - Door opened at time 1: $true \Rightarrow open(1)$.

The Frame Problem

- The following two situations are consistent with the knowledge base:

$$S1: \quad \neg open(0), open(1), \neg on(0), \neg on(1)$$

$$S2: \quad \neg open(0), open(1), \neg on(0), on(1)$$

- But $S1$ is the intended situation and not $S2$.
- The representation above is deficient (not tight enough) and there lies the frame problem.
- This problem is addressed by adding **frame axioms**.
  In this case: $on(0) \iff on(1)$.