Overview

• Last time
  – Game playing
    • Minimax decisions
    • Alpha-beta pruning

• Today
  – Introduce the need for explicit knowledge representation
  – Describe means of knowledge representation
  – Consider rules as one particular means of knowledge representation

• Learning outcome covered today:
  Distinguish the characteristics, and advantages and disadvantages, of the major knowledge representation paradigms that have been used in AI, such as production rules, semantic networks, propositional logic and first-order logic;

Knowledge in AI

• Search is a “universal method” for problem solving
• But real problems require methods with more power, which comes from tailoring to the specific problem
  – Heuristic searches
  – Evaluation functions for game playing
  – Solution templates
• 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

The Knowledge Principle

Ed Feigenbaum:

• “. . . power exhibited . . . is primarily a consequence of the specialist knowledge employed by the agent and only very secondarily related to . . . the power of the [computer]”

• “Our agents must be knowledge rich, even if they are methods poor”
The Role of Knowledge

• Knowledge about a domain allows problem solving to be **focused** - it is not necessary to search exhaustively: useless branches need not be explored

• Explicit representations of knowledge allow a **domain expert** to understand the knowledge a system has, add to it, edit it, and so on
  – Knowledge engineering

• Comparatively **simple** algorithms can be used to **reason** with the knowledge and derive **new** knowledge

What is Knowledge?

• Knowledge is information about some **domain** or subject area, or about **how to do** something

• Knowledge can take many forms. Some simple examples are:
  – Eve is a female, Adam is a male
  – Females with children are mothers
  – Mothers are females, fathers are males
  – cf. Prolog facts and rules

How to Represent Knowledge?

• Why don’t we use **natural languages** (e.g. English) to represent knowledge?
  – Natural language is certainly expressive enough!
  – But it is also **too ambiguous** for automated reasoning
  – No clear semantics

• Syntactic ambiguities
  – “Time flies like an arrow; Fruit flies like a banana”

• Semantic ambiguities
  – “bank” can be “river bank” or “financial bank”

Dedicated KR Language
Dedicated KR Language

- **requirements**
  - store facts
  - reason ('inference')
  - map from/to real world

Databases

- Simple databases are commonly used to good effect in computer science
- They can be use to store and manipulate virtually any kind of information

### A Record Structure

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>John Adams</td>
<td>025</td>
<td>male</td>
</tr>
<tr>
<td>Sally</td>
<td>single</td>
<td></td>
</tr>
<tr>
<td>Richard</td>
<td></td>
<td></td>
</tr>
<tr>
<td>George</td>
<td>NIL</td>
<td></td>
</tr>
</tbody>
</table>

### A Directed Graph

- But storage and display are not enough - we also need to **manipulate** the knowledge

Databases as a KR

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

Databases as a KR

- **Disadvantages**
  - Only simple aspects of the problem domain can be accommodated
  - We can represent entities, and relationships between entities, but not much more
    - Prolog facts
  - Reasoning is very simple. Basically, the only reasoning possible is simple lookup, and we usually need more sophisticated processing than that
Knowledge Representation

• So, how can we represent knowledge in a form amenable to computer manipulation?

• Desirable features of a KR scheme
  – representational adequacy
  – inferential adequacy
  – inferential efficiency
  – well-defined syntax and semantics
  – naturalness

Representational Adequacy

• A KR scheme must be able to represent the knowledge appropriate to our problem
  – e.g. Chess: must represent type of piece, colour of piece, position
  – Cannot permit two pieces on same square

• Some KR schemes are better for particular sorts of knowledge than others

• There is no one ideal KR scheme

Inferential Adequacy

• A KR scheme must allow us to make new inferences from old knowledge

• It must make inferences that are
  – sound - the new knowledge really does follow from the old knowledge
  – complete - it should make all the right inferences

• Soundness is usually easy, completeness is often very hard

Exercise

• Is there a problem with the following inference?

Knowledge: John is a man
All men are mortal

Inference: Harry is mortal
Inferential Efficiency

• A KR scheme should be tractable - make inferences in reasonable (polynomial) time
• Unfortunately, any KR scheme with significant expressive power is not going to be efficient
• Often, the more general a KR scheme is, the less efficient it is
• Use KR schemes tailored to problem domain - less general, but more efficient
  – KR scheme with expressive power: first-order logic, is undecidable
  – Prolog uses Horn Clauses – a tractable subset of first order logic

Syntax and Semantics

• It should be possible to tell
  – whether any construction is “grammatically correct”
  – how to read any particular construction - no ambiguity
  – Thus a KR scheme should have a well-defined syntax
• It should be possible to precisely determine, for any given construction, exactly what its meaning is (the circumstances under which it is true)
  – Thus a KR scheme should have well-defined semantics
• Syntax is easy, semantics is hard!

Example

Arithmetics

• Syntax
  – The expression $A + B > 3$ is correct while $A + B >$ is not
• Semantics
  – $A + B > 3$ evaluates to either “true” or “false” depending on the values of $A$ and $B
Other examples: SQL, programming languages, etc.

Naturalness

• Ideally, KR scheme should closely correspond to our way of thinking, reading, and writing
• Allow knowledge engineer to read and check knowledge base
• Again, the more general a KR scheme is, the less likely it is to be readable and understandable
  – People may have preferences: logic is natural to some; some people like diagrams or graphs while others do not
Basic Approaches

• Neither natural languages nor traditional computer formalisms are good enough for KR
• Some alternative basic approaches are
  – Rule-based systems (a.k.a. production systems)
    • Expert systems
  – Semantic networks
    • Graphical representation convenient for knowledge engineers
    • Later developed into 'ontologies'
  – Logic
    • Formal semantics
    • ...

Rule-Based Systems

• Knowledge is specified as a collection of rules
• Each rule has the form
  condition -> action
  which may be read if condition then action
• condition (antecedent) is a pattern
• action (consequent) is an operation to be performed if the rule fires
• Rules are applied to facts - unconditional statements that are assumed to be correct (at the time they are used)
  – A rule can fire if the condition matches the facts

Example Rule Base

Rules:

R1: IF animal has feathers
   THEN animal is a bird
R2: IF animal is a bird
   THEN animal can fly
R3: IF animal can fly
   THEN animal is not scared of heights

Action is ADD this fact

this and next lecture
Further lectures
Example Rule Base

Rules:

R1: IF animal has feathers
    THEN animal is a bird
R2: IF animal is a bird
    THEN animal can fly
R3: IF animal can fly
    THEN animal is not scared of heights

Suppose F1: kiwi has feathers

R1 fires so add F2: kiwi is a bird
R2 fires so add F3: kiwi can fly
R3 fires so add F4: kiwi is not scared of heights
Rule-Based System Architecture

- A collection of rules
- A collection of facts
- A rule **fires** if a fact matches the **condition** of the rule
  - Mechanism that fires rules is *inference engine*

What can we do with rules?

- See what new facts can be **derived**, e.g.
  - F3: kiwi is not scared of heights
- Ask whether a fact is **implied** by the knowledge base and already known facts, e.g.
  - Can a giraffe fly?

Rule-Based Systems as KR

- **Advantages**
  - These systems are very expressive
  - The rules lead to a degree of modularity
- **Disadvantages**
  - There is a lack of precise semantics for the rules
  - The systems are not always efficient
  - What if several rules match the facts?

Relation to Search

- Using rules can be thought of as just another form of search
- The sets of facts are states
- Rules are the actions performed in states
- This suggests that there are schemes for applying rules that are similar to search techniques
- We will look at these in the next lecture
Summary

• Discussed the need for explicit knowledge representation
• Considered properties of KR schemes
• Looked at rules as one such scheme

• Next time
  – Algorithms for reasoning with rules