Artificial Intelligence
人工智慧

Lecture 2
February 27, 2013
洪國寶
Outline

• Review
• The Turing test
• Foundations of AI
• Chapter 3
  – Knowledge Representation
Course information

• **Textbook**
  - *Artificial Intelligence Illuminated*
    Ben Coppin,
    (開發圖書公司)
  - Try Google search (keywords: “artificial intelligence illuminated”)

• **Course web page:**
The **objective** of this course is to learn fundamental techniques of artificial intelligence, including

- symbolic programming,
- knowledge representation,
- search,
- inference,
- machine learning,
- planning, and
- computer understanding of language and images.
Strong AI and Weak AI

• Artificial Intelligence involves using methods based on the intelligent behavior of humans and other animals to solve complex problems.

• **Strong AI:**
  – This is the view that a sufficiently programmed computer would actually *be* intelligent and would think in the same way that a human does.

• **Weak AI:**
  – This is the use of methods modeled on intelligent behavior to make computers more efficient at solving problems.

• **This course is concerned with Weak AI.**
Strong Methods and Weak Methods

• Not to be confused with Strong AI and Weak AI.
• Strong methods use knowledge about the world to solve problems.
• Weak methods use logic and other symbolic systems.
• Strong method systems rely on weak methods, as knowledge is useless without a way to handle that knowledge.
AI history – trends in popular hype


- Early enthusiasm, great expectations
- Big promises, great expectations, all fell short
- Booming industry
- Crude reality, again
Outlook for AI

• **AI has a long way to go.**
  – It may well be 100 years before any computer is as intelligent as a human.

• **No single discovery will suffice.**
  – Real AI will require hundreds of research breakthrough.

• **AI will continue to advance.**
  – Progress will continue but no sudden and final success

• **Useful applications of AI are feasible now.**

• **AI can enhance human performance.**
  – The best overall system will often be a combination of AI programs and humans.
Outline

- Review
- The Turing test
- Foundations of AI
- Chapter 3
  - Knowledge Representation
A paper by Turing

• Computing Machinery and Intelligence
  – “I propose to consider the question, ‘Can machines think?’”
    • And if so, how?
    • And if not, why not?
  – Describes the imitation game, now called the Turing Test.
  – Possibly one of most important and disputed topics in AI, philosophy of mind, cognitive science.
Turing Test

- Alan Turing invented the Turing Test, designed to determine if a computer system can be called an artificial intelligence or not, based on whether it can fool a human into thinking it is human too.
- **No system has yet passed the Turing Test.**
Turing Test

- Three rooms contain a person, a computer, and an interrogator.
- The interrogator can communicate with the other two by teleprinter.
- The interrogator tries to determine which is the person and which is the machine.
- The machine tries to fool the interrogator into believing that it is the person.
- If the machine succeeds, then we conclude that the machine can think.
Turing Test

The interrogator asks a set of questions that must be answered by a machine and by a human being.

If the interrogator cannot differentiate the human from the machine, then, it is said that the machine is intelligent.

The Turing test provides a basis for many of the schemes used to evaluate AI programs. For example, we can use a variation of the Turing test to evaluate the performance of a KBS compared with the performance of a human expert (given a set of problems).
A sample imitation game

- Turing suggests some specimen Q & A’s:

  Q: Please write me a sonnet on the subject of the Forth Bridge
  A: Count me out on this one, I never could write poetry

  Q: Add 34957 to 70764.
     (pause about 30 seconds)
  A: 105621

  Q: Do you play chess?
  A: Yes

  Q: I have K at my K1, and no other pieces. You have only K at K6 and R at R1. It is your move. What do you play?
     (pause about 15s)
  A: R-R8 mate
Some Famous Imitation Games

• 1960s  ELIZA
  – Written by Joseph Weizenbaum
  – Used simple pattern matching
    • “Well, *my* boyfriend made *me* come here”
    • “*Your* boyfriend made *you* come here?”

• 1970s  SHRDLU
  – Written by Terry Winograd
  – Had a very limited domain

• 1990s  Loebner prize
  – win $100,000 if you pass the test
The Loebner Contest

- A modern version of the Turing Test, held annually, with a $100,000 cash prize.
- [http://www.loebner.net/Prizef/loebner-prize.html](http://www.loebner.net/Prizef/loebner-prize.html)
- Restricted topic (removed in 1995) and limited time.
- Participants include a set of *humans* and a set of *computers* and a set of *judges*.
- Scoring
  - Rank from least human to most human.

16
The Loebner Contest


• The Silver Medal Prize of $25,000 + Silver Medal will be awarded if any program fools two or more judges.

• At that point the contest will progress to the MultiModal stage in which entries in subsequent years will necessitate processing of MultiModal input (e.g. music, speech, pictures, videos).

• During the MultiModal stage, if any entry fools half the judges compared to half of the humans, the program's creator(s) will receive the Grand Prize of $100,000 + 18kt Gold Medal, and the competition will be discontinued.
The Loebner Contest

Set 1 - Questions relating to time:

Background facts:
a. The system clock will be accurate to within a minute or two.
b. The competition is scheduled to start at 10:00 AM Sunday, 6 Sept 2009
c. There will be 7 rounds of 30 minutes each.

Sample Questions
• What time is it?
• What round is this?
• Is it morning, noon, or night?
Etc.

http://loebner.net/Prizef/2009_Contest/LP_2009.html
The Loebner Contest

Set 2 - General questions about things.
Sample Questions:
• What is a hammer?
• What would I use a hammer for?
• Of what use is a taxi?
Etc.

Set 3 Questions relating to comparisons
Sample Questions
• Which is larger, a grape or a grapefruit?
• Which is faster, a train or a plane?
• John is older than Mary, and Mary is older than Sarah. Which of them is the oldest?
Etc:
The Loebner Contest

Set 4 - Questions demonstrating "memory" or persistence.

Sample Questions

• I have a friend named Harry who likes to play tennis. <One or more intervening questions or statements>
• What is the name of the friend I just told you about?
• Do you know what game Harry likes to play?
Etc.
Searle’s “Chinese Room”

- A thought experiment used to argue against strong AI.
- A non-Chinese speaker is in a room with a set of cards with Chinese characters, and a set of instructions in English.
- Questions in Chinese are fed into the room, and by following the instructions, the human is able to produce answers.
- The room appears to understand Chinese – it can answer questions in the language – but the human inside cannot.
- But does this mean the room knows Chinese??
- Conclusion: **TT only tests for “weak” AI, not “strong” AI.**
Summary: The Turing Test

• The Turing test turns a philosophical question ...
  • Can Machines think?
• … into an operational one
  • Can machines play the imitation game?
• We are not near writing programs to pass the test
• The Turing test does NOT drive much AI research
• Improving the capabilities of computers DOES ■
Outline

- Review
- The Turing test
- Foundations of AI
- Chapter 3
  - Knowledge Representation
Foundations of AI

- Mathematics
- Computer Science & Engineering
- Philosophy
- Economics
- Biology
- Psychology
- Cognitive Science
- Linguistics
Philosophy (sec 1.7)

• Philosophy provides an important background to a study of AI.
• The philosophy of great thinkers, from Plato to Descartes and to Daniel Dennett, has had a great deal of influence on the modern study of Artificial Intelligence.
• Descartes’ dualism described a universe consisting of two separate things: mind and matter.
• Descartes believed that humans possessed minds, but that animals were simply biological machines.
• The work of Aristotle, Descartes and more recently Daniel Dennett (born March 28, 1942 in Boston, Massachusetts) are worth studying.
Linguistics (sec 1.8)

• The study of human language has a vital role to play in Artificial Intelligence.
• For computers to interact with humans properly, they need to understand human language.
• Noam Chomsky’s work on grammars has informed the study of natural language processing (NLP).
• Knowledge representation which is fundamental to AI is essential to understanding language.
Human Psychology and Biology (sec 1.9)

- While most AI techniques do not map neatly onto real biological systems, some, such as neural networks, do.
- Cognitive Psychology has many links with AI: It involves the idea that the human brain uses processing methods on knowledge to solve problems, make decisions, draw conclusions, and carry out other intelligent acts.
- This contrasts with behaviorism, which is the view that behavior is linked directly to stimuli.
Outline

• Review
• The Turing test
• Foundations of AI
• Chapter 3
  – Knowledge Representation
Chapter 3
Knowledge Representation

• Introduction: The need for a good representation
• Semantic nets, Inheritance, Frames
• Search trees
• Problem reduction
Introduction

• This chapter focuses on the important topic of knowledge representation.
• Common representational methods used in Artificial Intelligence – frames, semantic nets, and search trees – are discussed.
• Also, the chapter provides a number of example problems and explains how the representational methods introduced can be used to solve the problems.

Continued in the next slide.
Introduction

• The Need for a Good Representation
  – A computer needs a representation of a problem in order to solve it.
  – In AI, both problem representation and knowledge representation are needed.

• A representation must be:
  – Efficient – not wasteful in time or resources.
  – Useful – allows the computer to solve the problem.
  – Meaningful – really relates to the problem.
Introduction

• The topic of representation is immensely important in Artificial Intelligence because to solve a problem that relates to the real world, a computer must have some way of internally representing the real world.

• Assumption of (traditional) AI work is that:
  – Knowledge may be represented as “symbol structures” (essentially, complex data structures) representing bits of knowledge (objects, concepts, facts, rules, strategies..).
  – Intelligent behavior can be achieved through manipulation of symbol structures.

Continued in the next slide.
Introduction

• Intelligent systems require that we have
  – Knowledge formally represented
  – New inferences/conclusions possible.
Inferential Adequacy

• Representing knowledge not very interesting unless you can use it to make inferences:
  – **Draw new conclusions** from existing facts.
    • “If it’s raining John never goes out” + “It’s raining today” so..
  – **Come up with solutions** to complex problems, using the represented knowledge.

• **Inferential adequacy** refers to how easy it is to draw inferences using represented knowledge.

• Representing everything as **natural language strings** has good representational adequacy and naturalness, but **very poor** inferential adequacy.
Inferential Efficiency

• You may be able, in principle, to make complex deductions given knowledge represented in a sophisticated way.

• But it may be just too inefficient.

• Generally the more complex the possible deductions, the less efficient will be the reasoner.

• Need representation and inference system sufficient for the task, without being hopelessly inefficient.
Main KR Approaches

• Logic
  – Will be covered in Lecture 7

• Frames/Semantic Networks

• Rule-based systems
  – Will be covered in Lecture 8

• Advanced Knowledge Representation
  – The blackboard architecture, scripts, and the Copycat architecture
Chapter 3

Knowledge Representation

• Introduction: The need for a good representation
• Semantic nets, Inheritance, Frames
• Search trees
• Problem reduction
Semantic networks

• **Semantic network**
  – Proposed by psychologists to represent associative abilities

• **Knowledge represented in a graph**
  – Objects and concepts are represented by *nodes*
  – The relations between objects and concepts by directed *links*
A Simple Semantic Net
Semantic networks

• “A sparrow is a bird”
  – Two concepts: “sparrow” and “bird”

  Spar \[ \text{IS-A} \] Bird

  – A sparrow is a kind of bird, so connect the two concepts with a \textit{IS-A relation}
    • This is an higher-lower relation or abstract-concrete relation
Semantic networks

- “A bird has feathers”
  - This is a different relation: the part-whole relation
  - Represented by a \textit{HAS-A} link or PART-OF link
  - The link is from whole to part, so the direction is the opposite of the IS-A link

\begin{tikzpicture}
  \node[draw, circle] (spar) at (0,0) {Spar};
  \node[draw, circle] (bird) at (1,0) {Bird};
  \node[draw, circle] (feat) at (2,0) {Feat};
  \draw[->] (spar) -- (bird) node[midway, above] {IS-A};
  \draw[->] (bird) -- (feat) node[midway, above] {HAS-A};
\end{tikzpicture}
Adding:
- Tweety and Sweety are birds
- Tweety has a red beak
- Sweety is Tweety’s child
- A crow is a bird
- Birds can fly
Semantic networks

• Semantic networks can answer queries
  – **Query**: “Which birds have red beaks?”
  – **Answer**: Tweety
  – **Method**: Direct match of subgraph

  – **Query**: “Can Tweety fly?”
  – **Answer**: Yes
  – **Method**: Following the IS-A link from “Tweety” to “bird” and the property link of “bird” to “fly”
  – This process is called **inheritance**
Inheritance

• **Inheritance** is the process by which a subclass inherits properties from a superclass.

• Example:
  – Mammals give birth to live young.
  – Fido is a mammal.
  – Therefore fido gives birth to live young.

• In some cases, as in the example above, inherited values may need to be overridden. (Fido may be a mammal, but if he’s male then he probably won’t give birth).
Semantic networks

• **Advantages of semantic networks**
  – Simple representation, easy to read
  – Associations possible
  – Inheritance possible

• **Disadvantages of semantic networks**
  – A separate inference procedure (interpreter) must be build
  – The validity of the inferences is not guaranteed
  – For large networks the processing is inefficient
Frame systems

• **Frame theory**
  – When humans encounter something new, a basic structure called a *frame* is selected from memory
  – A frame is a fixed framework in which all kinds of information is stored
  – For more details about the information in a frame, a different frame is selected
  – A frame is connected to other frames, so this is a network of frames
Frame systems

- **Frame**
  - *Frame name*: represents an object or a concept, so similar to node in the semantic network
  - *Frame type*: shows if this a concept (class) or an object (instance)
- **Slot**
  - Consists of slot name and facets
  - *Slot name*: property or relation name
- **Facet**
  - A facet gives information about the slot, i.e. the value and name
  - *Value*: the value of the property
  - *Default*: connecting frames can have a different value for this property
- **Demon**
  - Perform a certain action if a condition is satisfied

<table>
<thead>
<tr>
<th>Slot</th>
<th>Value</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>IS-A</td>
<td>animal</td>
<td></td>
</tr>
<tr>
<td>HAS-A</td>
<td>feather</td>
<td>leg</td>
</tr>
<tr>
<td>#Leg</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Weight</td>
<td>If-Needed</td>
<td>calc-weight</td>
</tr>
</tbody>
</table>

Frame

Frame type
# Frame systems

<table>
<thead>
<tr>
<th>Class</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>bird</td>
<td>IS-A: animal, HAS-A: feather, leg, #Leg: 2, Weight: If-Needed calc-weight</td>
</tr>
<tr>
<td>crow</td>
<td>IS-A: bird, Color: default black</td>
</tr>
<tr>
<td>beak</td>
<td>IS-A: bird, Beakcol: default yellow</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Instance</th>
<th>Properties</th>
</tr>
</thead>
</table>

The frame system for birds includes the following properties:
- IS-A: animal
- HAS-A: feather, leg, #Leg: 2, Weight: If-Needed calc-weight

For the instance Tweety:
- IS-A: bird
- HAS-A: beak
- Child: Sweety
- Birthday: 1990.1.1
- If-Added calc-age
Frame systems

• **Inference in frame systems**
  
  – *Query:* “How many legs has a crow?”
  
  – *Answer:* 2
  
  – *Inference*
    
    • No information about this in the “crow” frame
    
    • Try to find it in the “bird” frame
    
    • Default value is 2
  
  – Also called *inheritance*

  – As soon as the birthday of Tweety is added, the “calc-age” procedure is invoked

  – *Query:* “What is the weight of Tweety?”
  
  – The answer is obtained by the procedure “calc-weight” in bird
Why frames are useful?

• **The main advantage of using frame-based systems for expert systems** over the rule-based approach **is that all the information about a particular object is stored in one place.**
  
  – In a rule-based system, information about Fido might be stored in a number of otherwise unrelated rules, and so if Fido changes, or a deduction needs to be made about Fido, time may be wasted examining irrelevant rules and facts in the system, whereas with the frame system, the Fido frame could be quickly examined.
  
  – This difference becomes particularly clear when we consider frames that have a very large number of slots and where a large number of relationships exist between frames (i.e., a situation in which objects have a lot of properties, and a lot of objects are related to each other).
## Inheritance

<table>
<thead>
<tr>
<th>Frame Name</th>
<th>Slot</th>
<th>Slot Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mammal</td>
<td>*number of legs</td>
<td>four</td>
</tr>
<tr>
<td>Dog</td>
<td>subclass</td>
<td>Mammal</td>
</tr>
<tr>
<td>Cat</td>
<td>subclass</td>
<td>Mammal</td>
</tr>
<tr>
<td>Fido</td>
<td>is a</td>
<td>Dog</td>
</tr>
<tr>
<td></td>
<td>number of legs</td>
<td>three</td>
</tr>
<tr>
<td>Fang</td>
<td>is a</td>
<td>Cat</td>
</tr>
</tbody>
</table>

Here we have used an asterisk (*) to indicate that the value for the "number of legs" slot for the Mammal class is a default value and can be overridden, as has been done for Fido.
Slots as frames

- It is possible to express a range of values that a slot can take.
- One way to express this kind of restriction is by allowing slots to be frames.

<table>
<thead>
<tr>
<th>Frame Name</th>
<th>Slot</th>
<th>Slot Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of legs</td>
<td>minimum value</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>maximum value</td>
<td>4</td>
</tr>
</tbody>
</table>
Multiple inheritance

• It is possible for a frame to inherit properties from more than one other frame. In other words, a class can be a subclass of two superclasses, and an object can be an instance of more than one class. This is known as multiple inheritance.

<table>
<thead>
<tr>
<th>Frame Name</th>
<th>Slot</th>
<th>Slot Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human</td>
<td>Subclass</td>
<td>Mammal</td>
</tr>
<tr>
<td></td>
<td>Number of legs</td>
<td>two</td>
</tr>
<tr>
<td>Builder</td>
<td>Builds</td>
<td>houses</td>
</tr>
<tr>
<td>Bob</td>
<td>is a</td>
<td>Human</td>
</tr>
</tbody>
</table>
Multiple inheritance

• In some cases, we will encounter conflicts, where multiple inheritance leads us to conclude contradictory information about a frame.

• In this case, we need a mechanism to decide which features to inherit from which superclasses.
  – One simple method is to simply say that conflicts are resolved by the order in which they appear. So if a fact is established by inheritance, and then that fact is contradicted by inheritance, the first fact is kept because it appeared first, and the contradiction is discarded.

• Multiple inheritance is a key feature of most object-oriented programming languages.
Procedures and Demons

- A **procedure** is a set of instructions associated with a frame (or a slot).
  - For example, a **slot reader** procedure might return the value of a particular slot within the frame.
  - Another procedure might insert a value into a slot (a **slot writer**).
  - Another important procedure is the **instance constructor**, which creates an instance of a class.
  - Such procedures are called when needed and so are called **WHEN NEEDED procedures**.

- The procedure can be run upon request.

- A **demon** is a procedure that is run automatically, usually triggered by an event such as when a value is:
  - Read: when a particular value is read.
  - Written: when the value of a slot is changed.
  - Created
  - Changed ✓
Inference in Frames

When using frames we need a procedure for retrieving facts about slot values. There are 3 inference techniques that can be used when using frames:

1. **Explicit information**: When a Frame contains a slot with where the required value is explicitly stated, then, that is the value, otherwise the value is obtained through inference (e.g., inheritance).

2. **Inheritance**: To find a value by inheritance, we need to move from the current frame to a more general frame according to the relations between frames (e.g., instance_of, a_kind_of). Such a move leads to a “parent frame” & a value may be found in this frame explicitly, or through further inheritance.

3. **Using Procedural slots**: When a procedure is given in a slot instead of the value itself, simply execute the procedure. This type of inference may require the use of types 1 and 2, described above.
Object Oriented Programming

• Object oriented programming languages such as Java, C++.

• Use ideas such as:
  – inheritance
  – multiple inheritance
  – overriding default values
  – procedures and demons

• Languages such as IBM’s APL2 use a frame based data structure.
Chapter 3

Knowledge Representation

- Introduction: The need for a good representation
- Semantic nets, Inheritance, Frames
- Search trees
- Problem reduction
Many problems in Artificial Intelligence can be represented as search spaces.

In simple terms, a search space is a representation of the set of possible choices in a given problem, one or more of which are the solution to the problem.

Figure 3.3 shows a very simple state-space diagram for a robot that lives in an environment with three rooms (room A, room B, and room C) and with a block that he can move from room to room.

– Each state consists of a possible arrangement of the robot and the block.
A simple state-space diagram
Search Trees

- **Used to represent** search spaces.
- Root node **has no predecessor.**
- Leaf nodes **have no successors.**
- **Goal nodes (of which there may be more than one)** represent solutions to a problem.
Search Trees: An Example

- A is the root node.
- L is the goal node.
- H, I, J, K, M, N and O are leaf nodes.
- There is only one complete path:
  - A, C, F, L
Example: Missionaries and Cannibals

- Three missionaries and three cannibals
- Want to cross a river using one canoe.
- Canoe can hold up to two people.
- Can never be more cannibals than missionaries on either side of the river.
- Aim: To get all safely across the river without any missionaries being eaten.
A Representation

• The first step in solving the problem is to choose a suitable representation.

• We will show number of cannibals, missionaries and canoes on each side of the river.

• Start state is therefore:

  - 3,3,1  0,0,0
  - 0,0,1  3,3,1
A Simpler Representation

• In fact, since the system is closed, we only need to represent one side of the river, as we can deduce the other side.

• We will represent the finishing side of the river, and omit the starting side.

• So start state is:
  – 0,0,0

65
Operators

- **Now we have to choose suitable operators that can be applied:**
  1. Move one cannibal across the river.
  2. Move two cannibals across the river.
  3. Move one missionary across the river.
  4. Move two missionaries across the river.
  5. Move one missionary and one cannibal.
A partial search tree for the missionaries and cannibals problem

1. Move one cannibal across the river.
2. Move two cannibals across the river.
3. Move one missionary across the river.
4. Move two missionaries across the river.
5. Move one missionary and one cannibal.
The Search Tree

- Cycles have been removed.
- Nodes represent states, edges represent operators.
- There are two shortest paths that lead to the solution.

1. Move one cannibal across the river.
2. Move two cannibals across the river.
3. Move one missionary across the river.
4. Move two missionaries across the river.
5. Move one missionary and one cannibal.
More examples

• Example 2: The Traveling Salesman
• Example 3: The Towers of Hanoi
  – We have three pegs and a number of disks of different sizes. The aim is to move from the starting state where all the disks are on the first peg, in size order (smallest at the top) to the goal state where all the pegs are on the third peg, also in size order.
  – We are allowed to move one disk at a time, as long as there are no disks on top of it, and as long as we do not move it on top of a peg that is smaller than it.
• Example 4: Describe and Match
  – The Describe and Match method for identifying objects is to describe an object and then search for the same description in a database, which will identify the object.
Search tree representation used with Describe and match to identify a penguin
Chapter 3

Knowledge Representation

• Introduction: The need for a good representation
• Semantic nets, Inheritance, Frames
• Search trees
• Problem reduction
Combinatorial Explosion

• Problems that involve assigning values to a set of variables can grow exponentially with the number of variables.

• This is the problem of **combinatorial explosion**.

• Some such problems can be extremely hard to solve (NP-Complete, NP-Hard).

• Selecting the correct representation can help to reduce this, as can using **heuristics** (see chapter 4).
Problem Reduction

- Breaking a problem down into smaller sub-problems (or sub-goals).
- Can be represented using goal trees (or and-or trees).
- Nodes in the tree represent sub-problems.
- The root node represents the overall problem.
- Some nodes are and nodes, meaning all their children must be solved.
Problem Reduction: Example

- E.g. to solve the Towers of Hanoi problem with 4 disks, you can first solve the same problem with 3 disks.
- The solution is thus to get from the first diagram on the left, to the second, and then to apply the solution recursively.
Goal Trees

• Also called **and-or trees**.
• Nodes in the tree represent sub-problems.
• The root node represents the overall problem.
• Some nodes are **and** nodes, meaning all their children must be solved.
• An and-node is shown by drawing an arc across the arcs that join it to its subgoals (children). Or-nodes are not marked in this way.
And/Or Tree

Goal: Acquire TV

Steal TV

Buy TV

and

Get Job

Earn Money

Buy TV

Earn Money
Goal tree for Towers of Hanoi problem with four disks
Goal Trees

• There are two main approaches to breaking down a problem into subgoals—**top down** and **bottom up**.

• A **top-down approach** involves first breaking down the main problem into smaller goals and then recursively breaking down those goals into smaller goals, and so on, until leaf nodes, or success nodes, are reached, which can be solved.

• A **bottom-up approach** involves first determining all of the subgoals that are necessary to solve the entire problem, and then starting solve the success nodes, and working up until the complete solution is found.
Homework # 1

1. 3.1 (p. 65)
2. 3.9 (p. 65)
3. 3.10 (p. 65)
4. 3.13 (p. 66)

**Due** March 6, 2013
Homework # 1

3.1 Why are representations so important in Artificial Intelligence? What risks are inherent in using the wrong representation?

3.9 What is the difference between a top-down approach to solving a problem and a bottom up approach? In what kinds of situations might each be more appropriate?
3.10 Convert the following information into:

a) a semantic net

b) a frame-based representation

A Ford is a type of car. Bob owns two cars. Bob parks his car at home. His house is in California, which is a state. Sacramento is the state capital of California. Cars drive on the freeway, such as Route 101 and Highway 81.
3.13 Design a suitable representation and draw the complete search tree for the following problem:

A farmer is on one side of a river, and wishes to cross the river with a wolf, a chicken and a bag of grain. He can only take one item at a time in his boat with him. He can’t leave the chicken alone with the grain, or it will eat the grain, and he can’t leave the wolf alone with the chicken, or the wolf will eat the chicken. How does he get all three safely across to the other side?
Lisp

- **CLISP – an ANSI Common Lisp**

- **Common Lisp the Language, 2nd Edition**
  by Guy L. Steele Jr.

- **Association of Lisp Users**
  [http://alu.org/alu/home](http://alu.org/alu/home)