Artificial Intelligence

Unit 1

Brandon Syiem

Introduction to AI

What is it?

- “Artificial Intelligence is the study of how to make computers do things which, at the moment, people do better” - Rich & Knight
- Different from ***natural intelligence*** (exhibited by humans).
- *AI* is Artificial Intelligence till it is achieved, after which it becomes *Already Implemented*

The AI Problems

- **Formal Tasks**
  - Game Playing - chess, checkers,etc.
  - Theorem Proofs - Logic Theorists (1955 - 1956, Allen Newell, Herbert A. Simon and Cliff Shaw).
  - Medical Diagnosis

Note: Read about expert systems

- **Mundane Tasks**
  - Commonsense reasoning - daily tasks (how to get to work, relation between physical objects, effects of gravity on a suspended object when released, etc).
  - NLP (Natural Language Processing)
  - Perception

Underlying Assumption

- Physical Symbol System:
  - Symbols (physical patterns).
  - Symbol Structure (Expression) - collection of symbols related in some physical context.
  - Collection of *Expressions*.
  - Processes (operations on expressions to produce other operations)
  - Produces more *Expressions* in time by applying processes to *Expressions*.
- Physical Symbol System Definition by Newell and Simon (1976) - Rich and Knight (1.2 Underlying Assumption)

Cont ..

- Physical Symbol System Hypothesis:
  - "A physical symbol system has the necessary and sufficient means for general intelligent action." - Newell & Simon (1976).

AI Technique

- AI problems encompass a wide range of expertise and as such it is difficult to come up with general techniques to develop AI solutions for all such problems.
- One common aspect for all AI Solutions, however, is their need for **Knowledge.**
    - *Intelligence needs knowledge*

Properties of Knowledge

- Voluminous.
- Difficult to characterize.
- Ever changing.
- Context dependent.

Back to AI Technique

- AI technique is a method that exploits knowledge.
- Knowledge has to be represented in a way such that:
  - It is general.
  - Understood by people that provide it
  - Easily modified (to compensate for its ever changing nature)

The level of the model (*)

- Two types of programs modelled based on how people perform tasks:
  - Simple tasks that are trivial for computers due to exploiting a resource unavailable to humans.
  - Tasks that are more appropriately termed as ***AI Problems.***
- Reasons for developing the latter:
  - To let the computer understand human reasoning.
  - To let people understand computer reasoning.
  - To exploit knowledge obtained from people. (Ex: Chess).
- The goal is to come up with a good model of the processes involved in intelligent reasoning when thinking about an AI solution.

Criteria for success

- **Turing Test**: machine (A) and a person (B) try to convince an interrogator that they are the person.
    - Fails when people do.
- Compare against other humans using standard performance measures (ex: chess ratings).
- Time taken for task as compared to people.

Problem solving, Search and Control Strategies

Defining the Problem as a State Space Search

Four steps when trying to solve an AI problem (in general):

- Define the problem clearly.
- Analyze the problem.
- Pinpoint and represent the knowledge required for the problem.
- Apply appropriate techniques to develop the solution.

State Space Search

- Representing the problem as a state space.
  - S - set of legal states of the problem
  - A - set of all possible actions of the problem
  - Ac(s) - function that returns possible $a \in A$ on state $s \in S$)
  - Result(s,a) - returns the state produced after performing action *a* on state *s*
  - Cost(s,a) - cost to perform action *a* on state *s*

Cont ..

- Search - refers to the *searching* of a path from the ***current state*** to one state belonging to a set of ***final states/goal states***.
    - Usually begins from a state belonging to a set defined as the ***initial states***
- Creating the Action set (A) or the set of rules:
    - Think about assumptions
    - Think about how general the rules should be.
    - Think about how much work needs to be pre-computed and reflected in the rules.

Production System

Since Search forms the core of many AI solutions, it is necessary to describe a structure that helps us define the search process. Consists of:

- Rules
  - Left side - defines the current state that a corresponding action (*a*) can be performed
    - IF statement - called precondition
  - Right side - describes the state after the action (*a*) is performed.
    - THEN - called action
- Knowledge/databases that holds information appropriate for solving the problem (current state, best path so far, heuristics, etc.)
- A Control Strategy to determine the order in which the rules are compared in the database and which rule is applied in case of conflicts.

Cont ..

- Rule Applier

Control Strategies

Search for rules in database.

Deals with conflicting matching of rules.

Requirement for good control strategies:

- **Causes motion** - pointless if rules are applied to states that make them loop on themselves.
- **Is systematic -** Randomly selecting applicable rules will cause motion but can be inefficient. Two methods:
    - Breadth-First Search
    - Depth-First Search

Breadth-First Search

- Initialize a queue (Q)
- Enqueue the initial state to Q
- While Q is not empty:
  - E <- dequeue(Q)
  - If E is a goal state:
    - Stop
  - Generate all offspring states of E by applying all applicable rules
  - Store all offsprings in Q.

Preorder Depth-First Search

- Set E to initial State
- DFS(E)
    - If E is goal, quit
    - Else
        - For each child (C) of E
            - DFS(C)

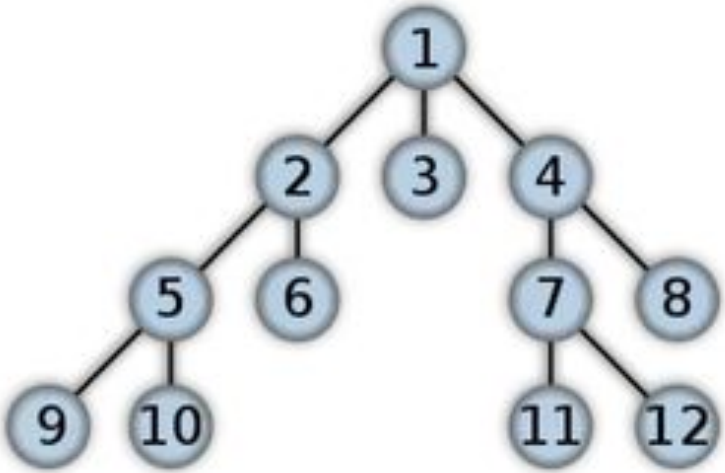
Note:

Three variations of DFS:

- Preorder
- Inorder
- Postorder

*Advantage and Disadvantages of BFS and DFS - book pg. 32 and 33

BFS



DFS

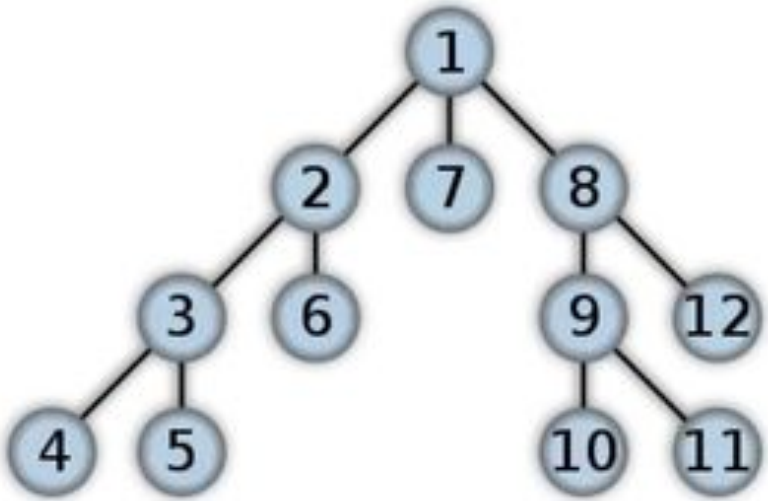


Image URL = https://ucbcsnotes.wordpress.com/2013/05/08/breadth-first-search-vs-depth-first-search/

Depth First Search for graphs

- Proceeds as DFS for tree with slight modifications
- Maintain list of visited nodes
- If going to a new node,
  - Check if new node is visited
    - If not, expand as usual
    - If yes,
      - Set edge from current node to the existing node (with the same node value / state as the new node)
      - If keeping track of best path, update the path if necessary.

Water Jug Problem

Say we have 2 water jugs, a 4 litre jug and a 3 litre jug. We wish to fill the 4 litre jug with 2 litres of water and have 0 litres in the 3 litre jug. How can we do this without having any measuring tools?

Ans.) Let us represent the problem in state space, here let us denote the amount of water in the 4 litre jug by ***x*** and the water in the 3 litre jug by ***y.***

(x,y)

Let us know define a set of rules and actions to work on states to create new states.

Cont..

1. (0,y) -> (4,y)
2. (x,0) -> (x,3)
3. (x,2) -> (0,2) [if x != 0]
4. (0,2) -> (2,0)
5. (4,0) -> (0,3)
6. (0,3) -> (3,0)
7. (3,0) -> (3,3)
8. (3,3) -> (4,2) [pour from 3 litre jug to 4 litre jug until 4 litre jug is filled]
9. (x,y) -> (0,0) [if x!=0 | y != 0]

Sequence of steps to goal.

1-5-6-7-8-3-4

2-6-2-8-3-4

Travelling Salesman problem

Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city?

1. For **N** cities, there are (N-1).(N-2).(N-3)....2.1 or (N-1)! Routes or possibilities.

Heuristic Search

- **Heuristic** - enabling someone to discover or learn something on their own.
- Similar to applying an assumption to our search problem.
    - Applying ***Nearest Neighbour Heuristic*** to the travelling salesman problem will reduce the complexity to N^2 from N!.
    - This may not find the best (shortest) route but will find a good route.
- “A heuristic function is a function that maps from problem state descriptions to measures of desirability”
- Leads to another definition of AI: “The study of techniques for solving exponentially hard problems in polynomial time by exploiting knowledge in that domain.”

Problem Characteristics

- Is the problem decomposable of independent smaller and easier subproblems?
- Can solution steps be ignored or undone? (Ignorable, Recoverable, Irrecoverable)
- Is the problem's universe predictable? (Certain-outcome, Uncertain-Outcome)
- Is a good solution obvious without comparison to other solutions? (Any-path solution, Best-path Solution)

Cont..

- Is the solution a state of the world or a path to that state?
- Is a large amount of knowledge required to solve the problem or simply to reduce the search space?
- Can a computer simply be given the problem and produce a solution or will interaction with a person be necessary? (Solitary, Conversational)

Rich & Knight - Section 2.3 pg. 36

Problem Classification

- Problems can be classified into different types.
- Medical diagnosis can be a problem of ***classification***.
- Predicting the sales of a commodity based on marketing techniques can be a problem of ***regression***.
- Etc.
- No single way to solve problems.
- Learn from past solutions to similar problems.

Production System Characteristics

- Can production systems , like problems, be described by a set of characteristics? (yes)
- Is there a relationship between the problem type and the production system type used to solve the problem?

Types of Production Systems

- Monotonic Production System
  - Is a system where the application of a rule (***r1***) **does not** restrict the application of another rule (***r2***) that could have also been applied when ***r1*** was selected**.**
- Nonmonotonic Production System
  - Is a system where the application of a rule (***r1***) **does** restrict the application of another rule (***r2***) that could have also been applied when ***r1*** was selected**.**
- Partially Commutative Production System
  - Is a system where if a sequence of rules transforms state *x* to state *y*, then any permutation of those rules (if applicable) will transform state *x* to state *y*.
  - Useful for solutions that change things but changes are reversible.
- Commutative Production System
  - Is a system that is both Monotonic and Partially Commutative.
  - Useful for solutions that create new things and not change things.

	Monotonic	Nonmonotonic
Partially commutative	Theorem proving	Robot navigation
Not partially commutative	Chemical synthesis	Bridge

Fig. 2.17 *The Four Categories of Production Systems*

	Monotonic	Nonmonotonic
Partially Commutative	Ignorable	Recoverable
Not Partially Commutative	Irrecoverable	Irrecoverable

- **Note:** Not Partially Commutative systems are important in cases where the order of application of actions is important.

Issues in Design

- Search process can be thought of as a traversal of a tree structure where each node represents a state and each edge an operation/action.
- There may be too many branches when trying to explore a problem
- Implicitly describes the tree using rules, explicitly only generates those paths or partial trees if decided to be explored.

Cont ..

Consider:

- Forward or backward traversal
- Matching. How to select applicable rules for a given state.
- How to represent each node of a problem (knowledge representation and the frame problem).
- Search tree or search graph.
    - Graph : better if a node is likely to appear many times. (ex. Partially commutative production systems)
    - Tree : better if nodes are highly unlikely to appear again.

Additional Problems

- Towers of hanoi problem
- Missionaries and Cannibals problem
- Monkey and bananas problem
- Cryptoarithmetic