

## Problem Solving: Agent Architectures

- 2 extremes of agent architecture:
  1. Purely reactive
    - Stimulus-response
    - No thought given to resulting action
    - Such approaches can be implemented via
      - (a) Rules
      - (b) Look-up tables
      - (c) Physically hard-wired
  2. Rational agent
    - Agent is able to reason about world, possible actions and their results,  
...
    - Requires internal representation of world, actions, ...
    - As discussed earlier, this approach most flexible
    - Rational agents are problem solvers
    - Generally goal-driven (but may have reactive components)

## Problem Solving: Requirements

- AI concerned with solving problems
- Representing and reasoning about problems requires
  - Knowledge representation (KR)
    - \* This is the problem of representing information
    - \* In AI, this knowledge is generally
      - Voluminous
      - Hard to characterize accurately
      - Dynamic
    - \* KR schemes should
      - Capture generalizations
      - Be understandable by people
      - Be easily modified
      - Apply to many situations
      - Make access to knowledge efficient
    - \* The nth extreme is to represent all possible situations and associated actions explicitly
      - Very inflexible - designer must account for *every* eventuality
      - Cannot deal with unexpected situations
      - Storage requirements - except for smallest domains - not practical
      - This is *NOT* a reactive agent
  - Abstraction
    - \* Must be able to identify aspects of problem crucial to its solution, while ignoring those that are irrelevant
    - \* This tactic reduces amount of storage and processing required
  - Search
    - \* Problem solving can be thought of as a search
    - \* Given an agent with a set of sensors and effectors in an environment
      - Sensors inform the agent of its environment
      - Effectors change the state of the world
      - Problem solving can be described as a search through a set of such states
      - Solving the problem consists of finding a set of actions that will produce a state of the environment in which the solution is achieved

## Problem Solving: Requirements (2)

- Planning
  - \* For intelligent agents, problem solving is not a random task
  - \* The agent usually wants to be moderately sure of achieving a goal with a reasonable amount of effort and success
  - \* These issues can be addressed by planning a sequence of actions before carrying them out
- The catch-22
  - AI systems need lots of knowledge
  - As knowledge grows, becomes harder to manage
  - To facilitate efficient knowledge management, additional knowledge required

## Problem Solving: Classic AI Problems

### 1. Water jug problem

Given a 3 gallon jug, a 4 gallon jug, and an infinite water source, Fill the 4 gallon jug with 2 gallons

### 2. 8 puzzle

Given an 8 puzzle, arrange the tiles in some specified order

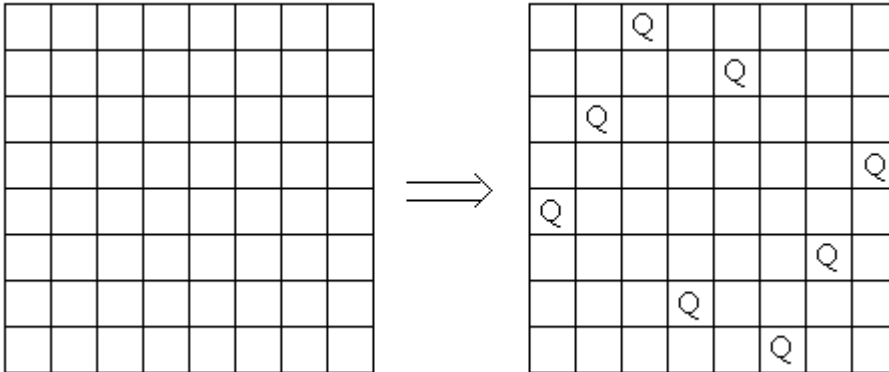
4	7	
5	1	3
6	2	8

 $\implies$ 

1	2	3
8		4
7	6	5

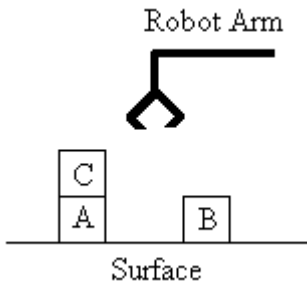
### 3. 8 queens

Given a chess board and 8 queens, arrange the queens - one per row - so that none is under attack



### 4. Blocks world

Given a set of blocks, a surface, and a robot arm, arrange the blocks in a specified configuration



### 5. Missionaries and cannibals (Hens and foxes)

3 missionaries and 3 cannibals are on one side of a river. There is a single canoe that can hold at most 2 people. All want to get to the other side. The missionaries can never be outnumbered because the cannibals will be able to overpower them and eat them. Get all 6 safely across the river.

## Problem Solving: Classic AI Problems (2)

### 6. The monkey and the bananas.

A bunch of bananas hang out of reach of the monkey, who is hungry. With the monkey are tools (depending on the version, one or more boxes, a stick, ...) What steps can the monkey take to get the bananas?

### 7. Cryptarithmic

Given an encoding of digits as characters, decode the problem

$$\begin{array}{r} \text{f o r t y} \\ + \text{t e n} \\ + \text{t e n} \\ \hline = \text{s i x t y} \end{array} \implies \begin{array}{r} 29786 \\ + 850 \\ + 850 \\ \hline = 31486 \end{array}$$

## Problem Specification: Problem Components

- States
  - Descriptions of the world at specific times
  - State changes as a result of agent intervention or environmental events
- Initial state (or states)
  - State at beginning of a problem
  - Single state problem has 1 initial state
  - Multiple state problem has several initial states
- Operators (actions): Functions mapping state  $\rightarrow$  state
  - For multiple state problems, maps sets of states  $\rightarrow$  sets of states
  - State space: Set of all states reachable from initial state(s) given a set of operators
  - Path: Sequence of actions that move from one state to another
  - (Nilsson calls generic representation of action a *schema* and instance of schema an operator)
- Goal test: Determines if goal reached
  - Can be explicit set of states
  - Or an abstract condition to be tested
- Path cost function (g): Measures cost of going from one state to another
  - Determined by:
    1. Fuel or energy expended going from state to state
    2. Danger ”
    3. Distance ”
    4. Etc.
  - Is additive
  - If no info provided, cost is 1 between states

## Problem Specification: Problem Components (2)

- Search cost
  - Measured wrt time and memory
  - Can tradeoff
    - \* Time for memory
    - \* Time for non-optimal path
- Problem defined by 4-tuple  
< initial state, operator set, goal test, cost function >

## Problem Specification: Problem Solving Overview

General steps taken in problem solving:

### 1. Goal formulation

- Id goal
- Goal = set of states, possibly a singleton

### 2. Problem formulation

- Id of means of solving problem
- Primary concerns are
  - Determining what states to consider
  - What actions are applicable
- Process can be thought of as identifying rules for solving the problem

### 3. Search

- Find sequence of actions that will transform start state to goal state
- 2 general approaches:
  - (a) No specific knowledge about the problem
    - Choose an action at random and pick up in new state
    - Reflexive agent
    - This is *uninformed* search
  - (b) Have additional knowledge
    - Can use this knowledge to make a plan of action
    - Goal-based/intelligent agent
    - This is *informed*, or *heuristic* search

### 4. Execute actions

- Solution is sequence of actions that transform initial state to goal state
- Note: Even with planning, planned sequence may not be actual sequence
  - Unforeseen events may necessitate alteration of plan

### 5. Execute actions

## Problem Specification: Problem Solving Overview (2)

General Algorithm for problem-solving agent:

```
function simpleProblemSolvingAgent (p) returns action
  input:  a percept p
  static: action sequence s, initially empty
         state
         goal g (initially null)
         problem
{
  if (empty (s)) {
    g <- formulateGoal (state)
    problem <- formulateProblem (state, g)
    s <- search (problem)
  }
  action <- recommendNextAction (s, state)
  return action
}
```

## Problem Specification: States

- Must determine how to represent states
- *Iconic* representation is a symbolic representation
- *Feature-based* representation is descriptive
- Choice of state influences effort in finding solution

## Problem Specification: Rules

- Several approaches can be taken to determine when an action is applicable
  - Based purely on sensory input
    - \* Inputs represented as binary string
    - \* String represents AND of specific stimuli
    - \* Action triggered by a specific string
    - \* Corresponds to a purely reactive agent
  - If-then rules
    - \* While bit-string approach can be considered if-then rule, rules usually more abstract
    - \* Rules represented in terms of *features* of the environment and internal states of the agent
    - \* Rules can be represented by
      1. Pattern  $\rightarrow$  pattern
      2. Pattern  $\rightarrow$  code
      3. Block of code
    - \* Left-hand side represents a state of the world
    - \* Right-hand side represents action to transform world
    - \* Corresponds to intelligent agent
- Factors to consider in writing rules
  - Specific v general left-hand sides
  - Does rule produce motion toward goal?
  - Special case rules: problem specific

## Problem Specification: Environments

- Environment is context in which agent functions
- Affects problem formulation
- Types of environments:
  1. Accessible v inaccessible:  
Agent can perceive complete state
  2. Deterministic v non-deterministic:  
Next state based solely on agent's actions
  3. Episodic v non-episodic:  
Episodes independent
  4. Static v dynamic:  
Environmental changes effected only by agent
  5. Discrete v continuous:  
Actions and percepts are distinct
- Some examples:

Environment	Accessible	Deterministic	Episodic	Static	Discrete
8 puzzle	Y	Y	N	Y	Y
Chess	Y	Y	N	N	Y
Poker	N	Y	N	N	Y
Driving	N	N	N	N	N
Part-picking robot	N	N	Y	N	N

- Real-world problems tend to be inaccessible, non-deterministic, non-episodic, dynamic, continuous

## Problem Specification: Problem Types

### 1. Single state

- Complete knowledge about current state and actions
- Allows planning of actions before execution

### 2. Multiple state

- Limited knowledge about current state or actions

If incomplete state knowledge:

- Do not know current state, so move from sets of states to sets of states
- Success will be some goal state, but which one is unknown to agent

If incomplete action knowledge:

- Result of action unknown
- From known state, can find path to goal state

### 3. Contingency problem

- Incomplete knowledge about current state and actions
- Cannot guarantee plan will succeed
- Generally act, monitor results, and proceed from there

### 4. Exploration problem

- No knowledge about current state and actions

## Problem Specification: Problem Characteristics

1. Is problem decomposable?
2. Are actions recoverable?
  - (a) Ignorable: Wasted actions ignored.
  - (b) Recoverable: Wasted actions can be undone.
  - (c) Irrecoverable: Once executed, must live with results.
3. Is world predictable?
4. Is solution absolute or relative?
5. Is solution a path or a state?