What is Artificial Intelligence?

- Definitions of AI vary
- Artificial Intelligence is the study of systems that

think like humans	think rationally
act like humans	act rationally

Systems that Act like Humans

- Behavioral approach
- Turing test: test for intelligent behavior
  - Interrogator writes questions and receives answers
  - System providing the answers passes the test if interrogator cannot tell whether the answers come from a person or not
- Necessary components of such a system form major AI sub-disciplines:
  - Natural language, knowledge representation, automated reasoning, machine learning

Systems that Think like Humans

- Not only I/O, but how humans do
- Formulate a theory of mind/brain
- Express the theory in a computer program
- Two Approaches
  - Cognitive Science and Psychology (testing/predicting responses of human subjects)
  - Cognitive Neuroscience (observing neurological data)
  - GPS: General Problem Solver

Systems that Think Rationally

- Laws of thought approach
- "Rational" -> **ideal** intelligence
  (contrast with **human** intelligence)
- Rational thinking governed by precise "laws of thought"
  - Aristotle's syllogisms
  - notation and logic
- Systems (in theory) can solve problems using such laws

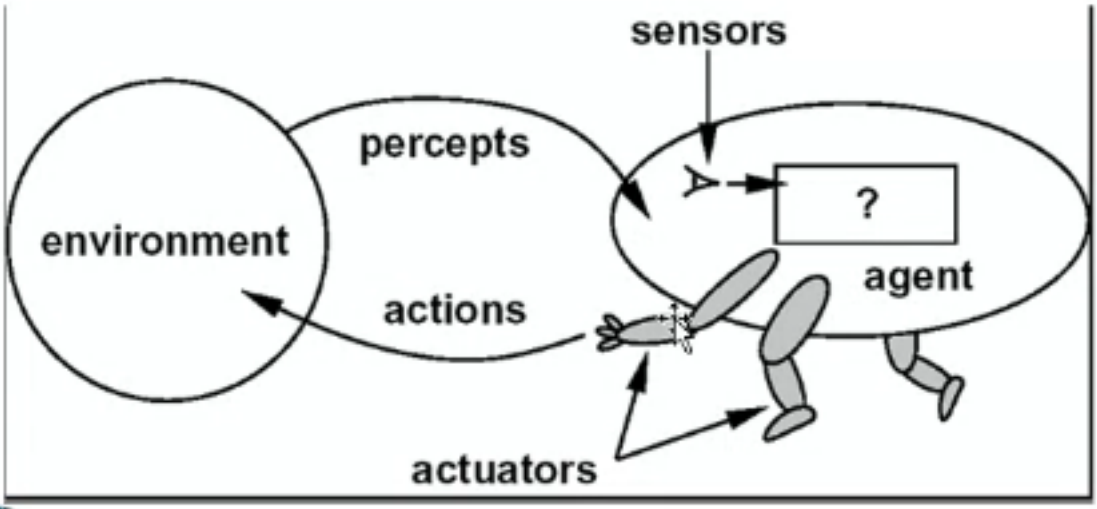
Systems that Act Rationally

- Satisfying approach
- Building systems that carry out actions to achieve the **best outcome**
- Rational **behavior**
- May or may not involve rational thinking
  - i.e., consider reflex actions

Intelligent Agents

- Agent: perceives, reasons and acts
- An agent is equipped with goals
- Modern AI: Engineering of rational agents
- A rational agent is one that acts so as to achieve its goals

Intelligent Agents



Agent Function

- $a = F(p)$

  where p is the current percept, a is the action carried out, and F is the agent function
- F maps percepts to actions

  $F: P \rightarrow A$

  where P is the set of all percepts, and A is the set of all actions
- In general, an action may depend on all percepts observed so far, not just the current percept, so…

Structure of Agents

- Agent = architecture + program
- architecture
  - device with sensors and actuators
  - e.g., A robotic car, a camera, a PC, …
- program
  - implements the agent function on the architecture

Specifying the Task Environment

PEAS

- Performance Measure: captures agent's aspiration
- Environment: context, restrictions
- Actuators: indicates what the agent can carry out
- Sensors: indicates what the agent can perceive

Properties of Environments

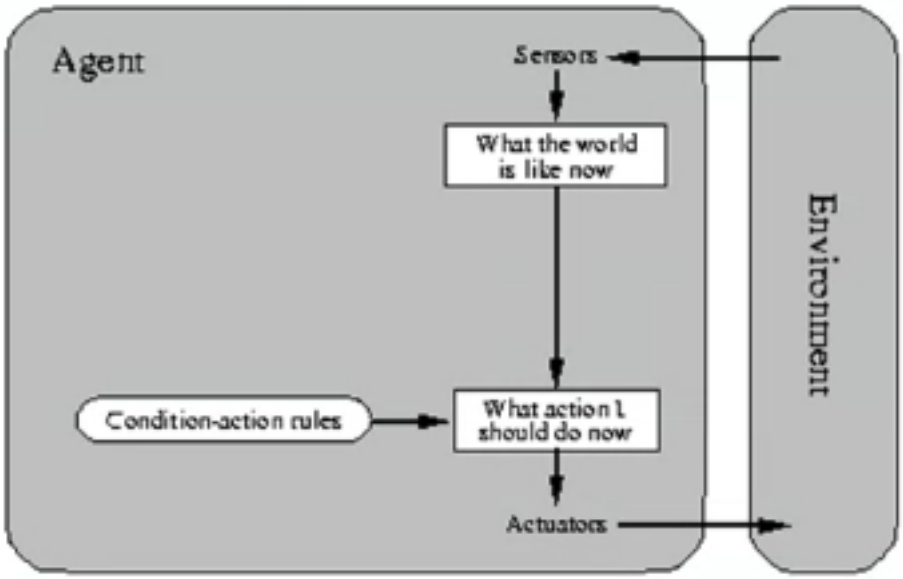
- Fully versus partially observable
- Deterministic versus stochastic
- Episodic versus sequential
- Static versus dynamic
- Discrete versus continuous
- Single agent versus multiagent

Types of Agents

- Table driven Agent -- Table look up
- Reactive/ Rule based Reflex Agent
- Reflex Agent with State (Memory)
- Goal-based Agent
- Utility-Based Agent

- Problem Solving Agent / Learning Agent

Reflex Agent

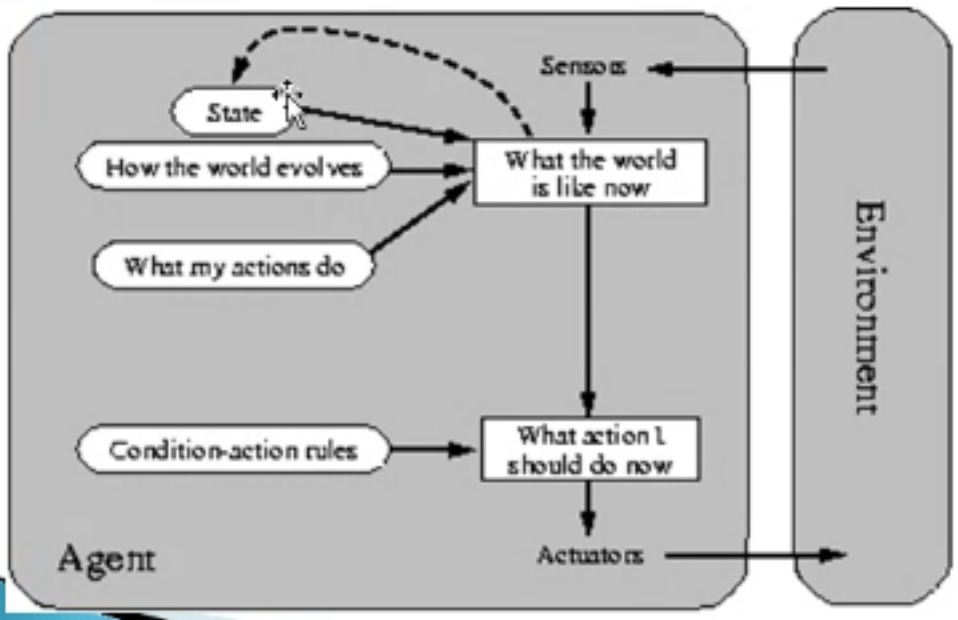


Reflex Agent

- Input: Percept
- Stored: set of condition action RULES
- Abstract: Interpret World's percept
- Match: match interpretation and rules
- Action: Select action

- Can be short sighted

Reflex Agent with State

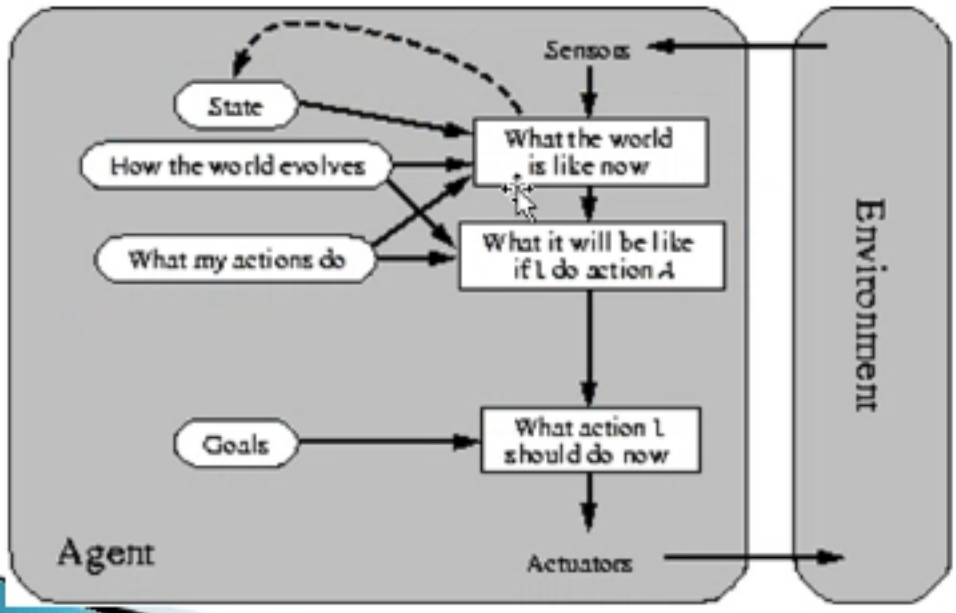


Reflex Agent with State

- Sensors do not give complete information of the world
- Memory remembers past states
- Stores: internal STATE
- Abstract: Interpret world's percept
- Match: match interpretation and state and rules
- Action: select action
- Remembers state

- Stored information may be incorrect

Goal-based Agent



Goal based Agent

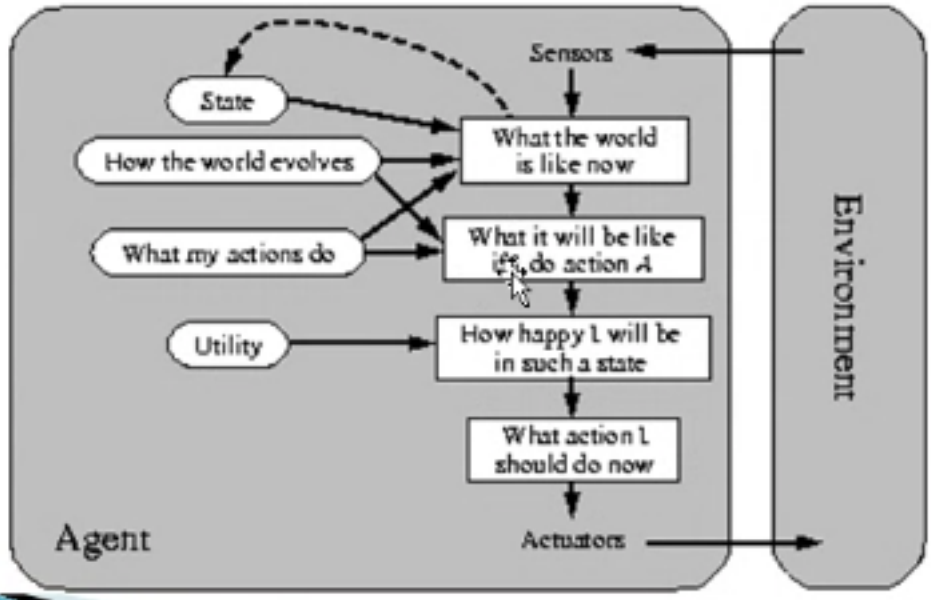
- Goal information: where to go?
- What will happen if this action is performed?

- May involve projection of actions to see if consistent with goals

- Search and planning, very flexible

- Takes time – world may change while reasoning about it

Utility-based Agent



Utility based Agent

- Generates high quality behaviour
- Choices of HOW to achieve goals
- Solution quality
- Agent's happiness

- f(state): real number
- Useful for evaluating complete goal and guiding search

- Game playing program

Problem Solving Agents

Agents as Search

•Goal formulation

•Domain formulation
- Actions
- States

•Problem formulation
- Initial state
- Specific goal

•Search

Look for a sequence of actions that moves from the initial state into a state where the specific goal is satisfied

•Execute

Carry out the solution

Formulate---Search---Execute

Problem Solving Agents

Agents as Search

•Goal formulation

•Domain formulation
- Actions
- States

•Problem formulation
- Initial state
- Specific goal

•Search

Look for a sequence of actions that moves from the initial state into a state where the specific goal is satisfied

•Execute

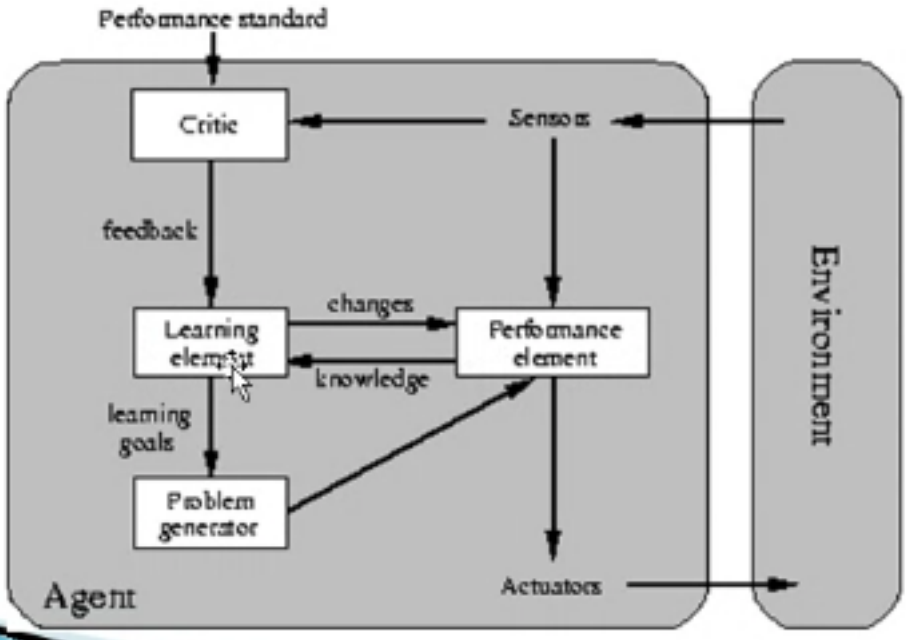
Carry out the solution

Formulate---Search---Execute

Learning Agents

- After an agent is programmed, can it work immediately?
  - No, it still needs teaching
- In AI,
  - Once an agent is done
  - We teach it by giving it a set of examples
  - Test it by using another set of examples
- We then say the agent learns
  - A learning agent

Learning System/ Agent



Components of Learning System

- **Performance Element (PE)** – It is the agent itself that acts in the world. It takes in percepts, decides on external actions, executes domain specific tasks, and provide feedback to the LE.
- **Learning Element (LE)** –Major component; Responsible for making improvement; takes knowledge from PE and some feedback, gathers examples and then draws inferences (which will be subsequently used by the PE)---- determines how to modify PE.
- **Critic** – Tells LE how the agent is doing (success/ failure) by comparing with a fixed standard of performance; it may also answer questions posed by the Learning System.
- **Problem Generator** – Suggests problems or actions that will generate new examples to aid in training the system further.
- **Knowledge Base** – contains knowledge about the specific problem domain D (domain knowledge), and knowledge about learning in domain D; domain knowledge is used by the PE and knowledge about learning by the LE

  Ex: Automated taxi on roads

  PE– consists of knowledge and procedures for driving actions (turning, accelerating)

  LE– formulate goals – learn rules for braking, accelerating,…

  Critic – observes world and passes info to LE (e.g., quick right turn across 3 lanes of traffic, observe reaction of other drivers)

  Problem generator – Try a congested road or …

In short ...

- Can be applied to any of the previous agent types
  - Agent <-> Performance Element
- Learning Element
  - Causes improvements on agent/ **performance element**
  - Uses feedback from **critic**
  - Provides goals to **problem generator**

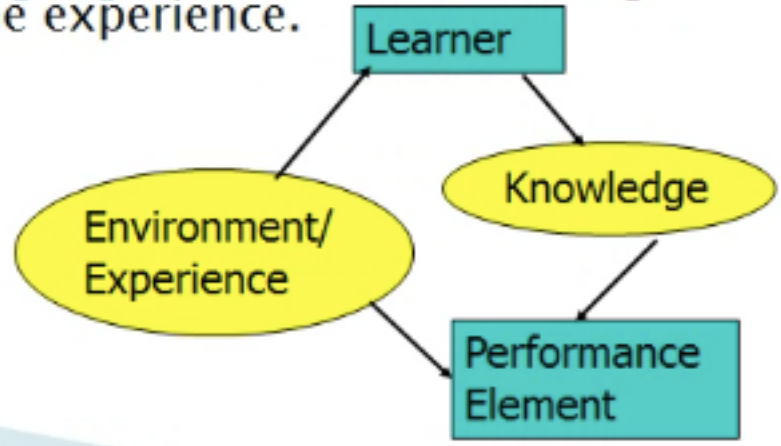
Learning Element (LE)

Design of LE is affected by:------

- *performance element* used
  - e.g., utility-based agent, reactive agent
- *functional component* to be learned
  - e.g., classifier, evaluation function, perception-action function
- *representation* of functional component
  - e.g., weighted linear function, logical theory, HMM
- *feedback* available
  - e.g., correct action, reward, relative preferences

Designing a Learning System -- main design decisions

- Choose the *training experience* – *how will the system access and use data ?*
- Choose exactly what is to be learned, i.e. the ***target function***.
- Hypothesis representation – how to represent the concepts to be learnt (target function).
- Choose a learning algorithm to infer the target function from the experience.



Training Experience: How will the system be exposed to its training experience ?

- Direct experience: Given sample input and output pairs for a useful target function ----supervised
  - Checker boards labeled with the correct move, e.g., extracted from record of experts' play
- Indirect experience: Given feedback which is ***not*** direct I/O pairs for a useful target function
  - Potentially arbitrary sequences of game moves and their final game results ---- reinforcement

Uncertainty Handling

Susmita Ghosh
Department of Computer Science & Engineering
Jadavpur University
Kolkata
INDIA

Reasoning based on PL

- Reasoning based on PL operates under:
  Certain, complete, unchanging, consistent facts
- Given facts, rules -> confident conclusions
- Derived truth -> never contradicts, given that no contradictions exist on axioms
- *Real life -> imprecise, incomplete, uncertain*

Limitations of PL

- limited in expressive power (more or less….)
- Only the True/ False statements
- No way to produce new knowledge about the world; only what is derivable

Uncertain Situation

- Uncertain knowledge – Heuristic knowledge – certain sets of evidences probably imply certain conclusions
- Uncertain Data – domain knowledge is certain, but not data – to infer a specific cause from an observed effect, we may have to rely on questionable test result
- Incomplete info. – decisions are taken in course of processing incrementally acquired info.
- Randomness – Knowledge & Info is complete, data is certain but domain is inherently random – stochastic domain
- *PL is monotonic (#facts known to be T at any time)*

NMR systems

- Human problem solver often augments absolute *truths* with *beliefs* that are subject to change given further info; *Tentative beliefs* – based on *default assumptions* that are made in the light of lack of evidence to the contrary
- An NMRS tracks a set of beliefs & revises those beliefs when new knowledge is observed/ derived
- NMRS – a set of premises (immutably T), tentative beliefs (potentially incorrect), dependency record for each tentative belief (tracks beliefs vs. justifications)

NMR systems

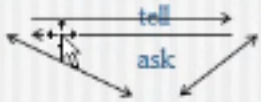
- Human problem solver often augments absolute *truths* with *beliefs* that are subject to change given further info; *Tentative beliefs* – based on *default assumptions* that are made in the light of lack of evidence to the contrary
- An NMRS tracks a set of beliefs & revises those beliefs when new knowledge is observed/ derived
- NMRS – a set of premises (immutably T), tentative beliefs (potentially incorrect), dependency record for each tentative belief (tracks beliefs vs. justifications)

NMR systems

- In NMR, the new facts became known which contradicted and invalidated old knowledge -> old knowledge was retracted causing other dependent knowledge to become invalid -> thereby requiring further retractions → shrinkage on non monotonic growth at times
- Methods to accurately represent and deal with different forms of uncertainty →

1. TMS – permits addition of changing (even contradictory) statements to KB
2. Modal & temporal logic – permit representations & reasoning about necessary & possible situations, temporal & other related situations
3. Fuzzy logic – deals with vague & imprecise info
4. Reasoning based on Probability

TMS

- Doyle, 1979
- Main job -> maintain consistency in KB being used by any problem solver and not to perform any inference fns.; As such, it frees the problem solver from any concerns of consistency
- IE 

TMS

KB

TMS maintains currently active belief set. Updating is incremental; after each inference info. Is exchanged between IE & TMS; IE tells TMS what deductions it has made. TMS asks questions about current beliefs and reasons for failure. It maintains a consistent set of beliefs for IE.

TMS operates as KBMS and is called every time the reasoning systems generates a new Truth value. TMS, using belief revision, takes any action required to modify dependent beliefs to maintain consistency

TMS

- In TMS,

Node -> one unit of knowledge: fact/rule/assertion; at any point of time, every node is in one of two conditions:

IN → currently believed to be T

OUT → currently believed to be F (there is no possible condition that would make it T/ the conditions required to make it T are not currently IN

SL justifications

- Associated with each node -> justifications for node's Truth value; assume, 1 justification for each node
- For each node, that is IN, TMS records a well founded support -> proof of the validity of the node
- [SL (in-nodes) (out-nodes)]
- In-nodes -> list of all nodes that must be IN for this node to be T
- Out-nodes -> OUT ..... T

1. It is sunny 1 SL(2)(3)
2. It is daytime 2 SL() ()
3. It is raining 3 SL () (1)
4. It is warm 4 SL (1)(3)

SL justifications

- Premise : always valid; in-list, out-list always empty
- Normal deduction: it is formed in normal sense of a monotonic system ; out-list always empty
- Assumption: a belief supported by lack of contrary info. Out-list never empty

CP Justification

Used to support hypothetical reasoning

[CP <consequent> (in-hypothesis)]

CP also includes out-hypothesis, always empty

A CP justification is valid iff the consequent node is IN whenever all of the nodes on in-hypothesis are IN