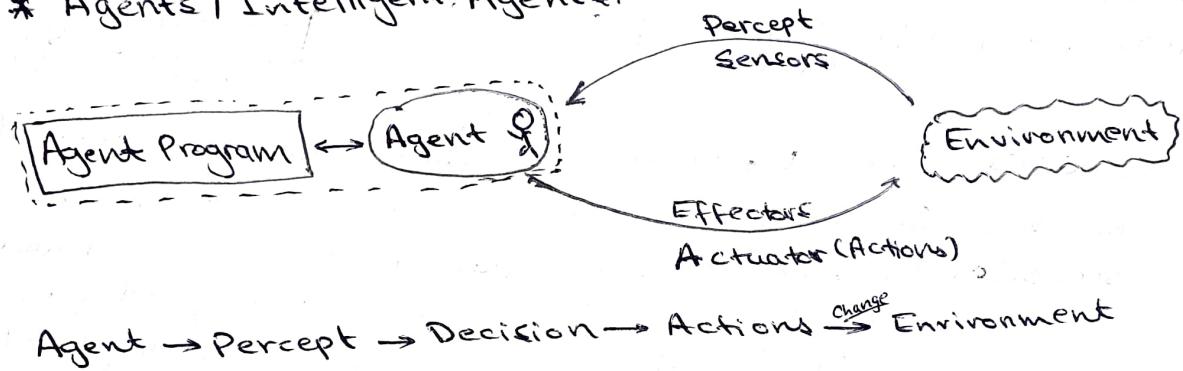


CS401 : Artificial Intelligence:

Syllabus:

- Approach to AI : Heuristic Search, Game Playing
- Knowledge Representation: Approaches, Predicate Logic, Reasoning
- Planning : Overview, Hierarchical, Goal Stack
- NLP : Syntactic, Semantic
- Multiagent System : Types, Properties
- Fuzzy Sets : Crisp, Fuzzy Set, α -cut, Operations.
- ANN and Genetic Algo : Single, Multilayer, Feed forward, Recurrent, ML

* Agents / Intelligent Agents.



"Goals of Agent" : High Performance
Optimised Result
Rational Action

P → Performance
E → Environment
A → Actions
S → Sensors.

four imp.
factors
to model
an agent

Types of Agents:

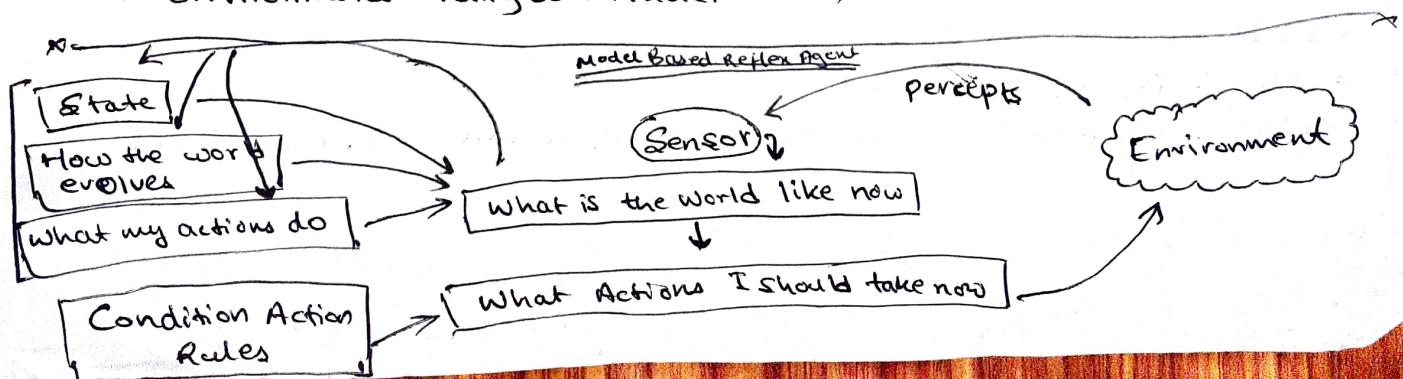
- Simple Reflex
- Model Based
- Goal Based
- Utility based
- Learning

• Simple Reflex Agents:

- Act only on the basis of current perception
- Ignore the rest of percept history
- Based on if-else Rules.
- Environment: fully observable.

• Model - Based Agents:

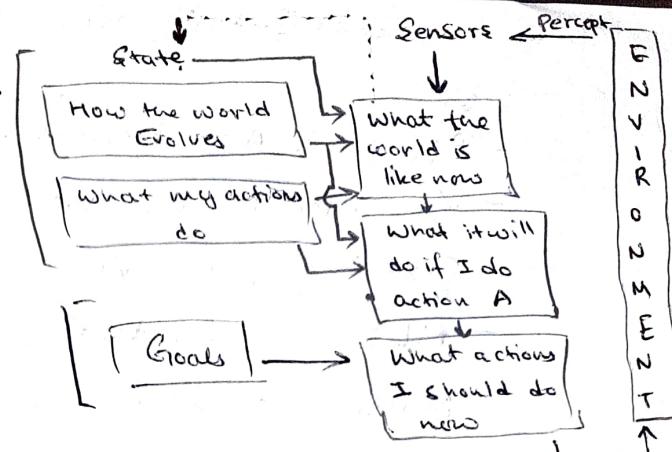
- Environment: Partially Observable
- Store Percept History (Internal Model)



• Goal Based Agents:

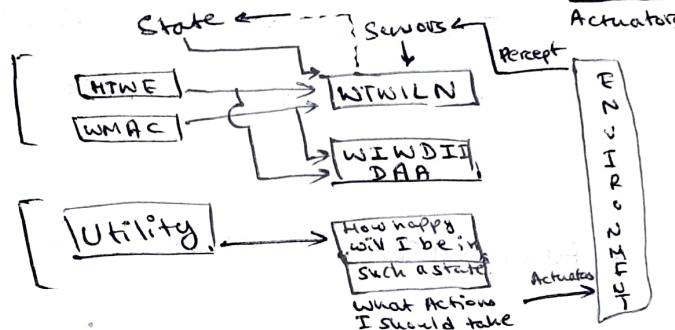
- Expansion of Model Based Reflex Agent.
- Desirable Situation (Goal)
- Searching and Planning.

• Theoretical Model.



• Utility Based Agents:

- focus on utility NOT goal.
- Utility function.
- Deals with Happy and Unhappy state.
- Expansion of Model Based Reflex Agent.



Goal vs Utility: Example: Google Map: final destination is the goal, however,

GPS does not know unexpected accident / traffic scenario. In such cases, new path is determined or should it continue on the same path even if it may delay the travel? Goal based will set new path; but utility based will see if the agent is happy or not; if not happy it will set new path, if happy it will continue on the same set path.

* Artificial Intelligence:

"Can Machines think" :- Alan Turing, 1950

↳ Problem Solving
↳ Reasoning
↳ Learning
↳ Perception.

- 1956: Birth of AI (Dartmouth meeting: 'Artificial Intelligence' name adopted).
- 1950: Turing's 'Computing Machinery and Intelligence'.
- 1964: Computers can understand NLP enough to solve algebra word problems.
- 1986: Rise of ML.
- 1995: AI as Science.
- 1997: Deep Blue chess program beats world chess champion.

* State Space Search : (falls under Problem Solving factors) :

$S: \{S, A, \text{Action}(S), \text{Result}(S, A), \text{cost}(S, A)\}$

- Precise
- Analyse.

where, S is start/goal state.

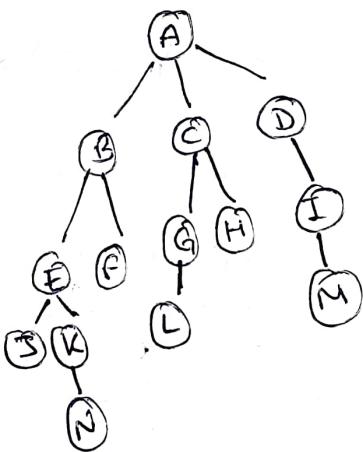
A is set of all possible actions.

→ Uninformed (Blind) Search. (Brute-force) :

→ Informed Search ? Heuristic :

Uninformed Searching	Informed Search
<ul style="list-style-type: none"> → Search without Information → No Knowledge → Time Consuming → More Complexity (Time, Space) → Blind, Brute-force E.g.: DFS, BFS etc. → Guaranteed Optimal Solution 	<ul style="list-style-type: none"> → Search with information → Use knowledge to find steps to solution. → Quick Solution. → Less Complexity (Time, Space). → Heuristic E.g.: A*, Heuristic DFS, Best-first search. → May be optimal, may not be optimal.

• BFS (Breadth First Search) : → uninformed, FIFO(queue), Shallowest Node, Complete, → Optimal, → Time Complexity $O(V+E)$

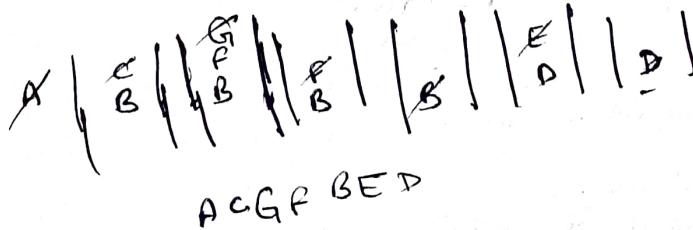


$\rightarrow AISKL$
 $\rightarrow IJKL$
 $\rightarrow JKLM$
 $\rightarrow KLM$
 $\rightarrow KMN$
 $\rightarrow MN$

$\rightarrow N$
 $O(b^d)$
 where,
 b is branch factor
 d is depth.

$N \rightarrow$ Nodes No.
 $E \rightarrow$ Edges No.

• DFS (Depth First Search) : → uninformed, stack(LIFO), Deepest Node, Incomplete, → Non-optimal → Time Complexity $O(V+E)$



ACGF BEP

$O(b^d)$

$b \rightarrow$ branching factor
 $d \rightarrow$ depth
 $V \rightarrow$ Num. of Nodes
 $E \rightarrow$ Num. of Edges

* Bidirectional Search:

- can be both BFS and DFS; better to use BFS.
- Uniformed - Complete in BFS - Incomplete in DFS.
- Time Complexity $\Omega(b^{d/2})$

* Generate and Test (DFS with backtracking)

- Heuristic Technique - Continuous Process - Complete
- Non Redundant - Informed.

* Best first Search

- Informed - Heuristic
- Greedy Method.
- Good Solution (^{may/may not} optimal)
- Time complexity: $O(b^m)$
 $O_b, O(b^d)$
- Priority queue (sort every time A \subseteq C)
- Takes only straight line distance
- Does not consider distance between two travelled nodes; i.e., $f(n) = h(n)$
- Complete.

* Beam First Search

- Takes care of space complexity (constant)
- Priority queue (sorts ~~only~~ A \subseteq C and stores only ~~all~~ best nodes)
- Like Best first search, takes only straight line distance; $f(n) = h(n)$
- Time complexity: $O(b^d)$
- Incomplete
- Good solution, but may/may not provide optimal soln)
- Greedy.

B is beam width
i.e., B can be 1, 2, 3, ... n.

* Hill Climbing Algorithm:

- It is beam first search but $B=1$.
- Greedy Approach. - No backtrack.
- Local Search Algorithm.

- Problems
- Local Maxima
- Plateau/flat Maximum
- Ridge.

* A* Algorithm:

- Informed. - Time complexity: $O(b^d)$
- Priority Queue? - Global consideration.
Sort everytime.
- like Best first search.
- Admissible. (Optimal soln when underestimation)

$$\rightarrow f(n) = g(n) + h(n)$$

$g(n)$: Actual cost from Start Node to n

$h(n)$: Estimation cost from n to Goal Node size, Straight line distance.

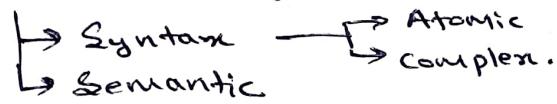
* AO* Algorithm:

- And/or
- Problem decomposition (breakdown into smaller pieces.)
 - Does not explore all the solution paths once it got a soln (incomplete)

* Knowledge Representation and Reasoning

- Logic : Propositional logic and Predicate logic
- Rules : if then
- Semantic Net : Google graph / Meaning graphs
- Frame : Slots and fillers / Object Attribute.
- Script :

* Propositional logic (Either True or False ; not Both).



- Negation: \neg , \sim , \top
- Disjunction/or : \vee
- Conjunction/And : \wedge
- if then : \rightarrow imply, implications.
- iff : \leftrightarrow
- .

Example: You can access internet from campus only if you are CSE student or you are not freshman.

Let, P: ~~can~~ access internet from campus

Q: ~~is~~ CSE Student

R: freshman.

Then, Answer: $P \rightarrow Q \vee \neg R$

Syntax of First order logic:

constants: 1, 2, A, Name, Location ...

variables: x, y, z, a, b

Predicates: Brother, father, greater than ...

functions: sqrt, ...

Connectives: \neg , \Rightarrow , \wedge , \vee , \Leftrightarrow , \rightarrow , \Leftarrow

Equity : =

Quantifiers : \forall , \exists

Artificial Intelligence



**Dr. Partha Pakray
Assistant Professor
Department of CSE
NIT Silchar**

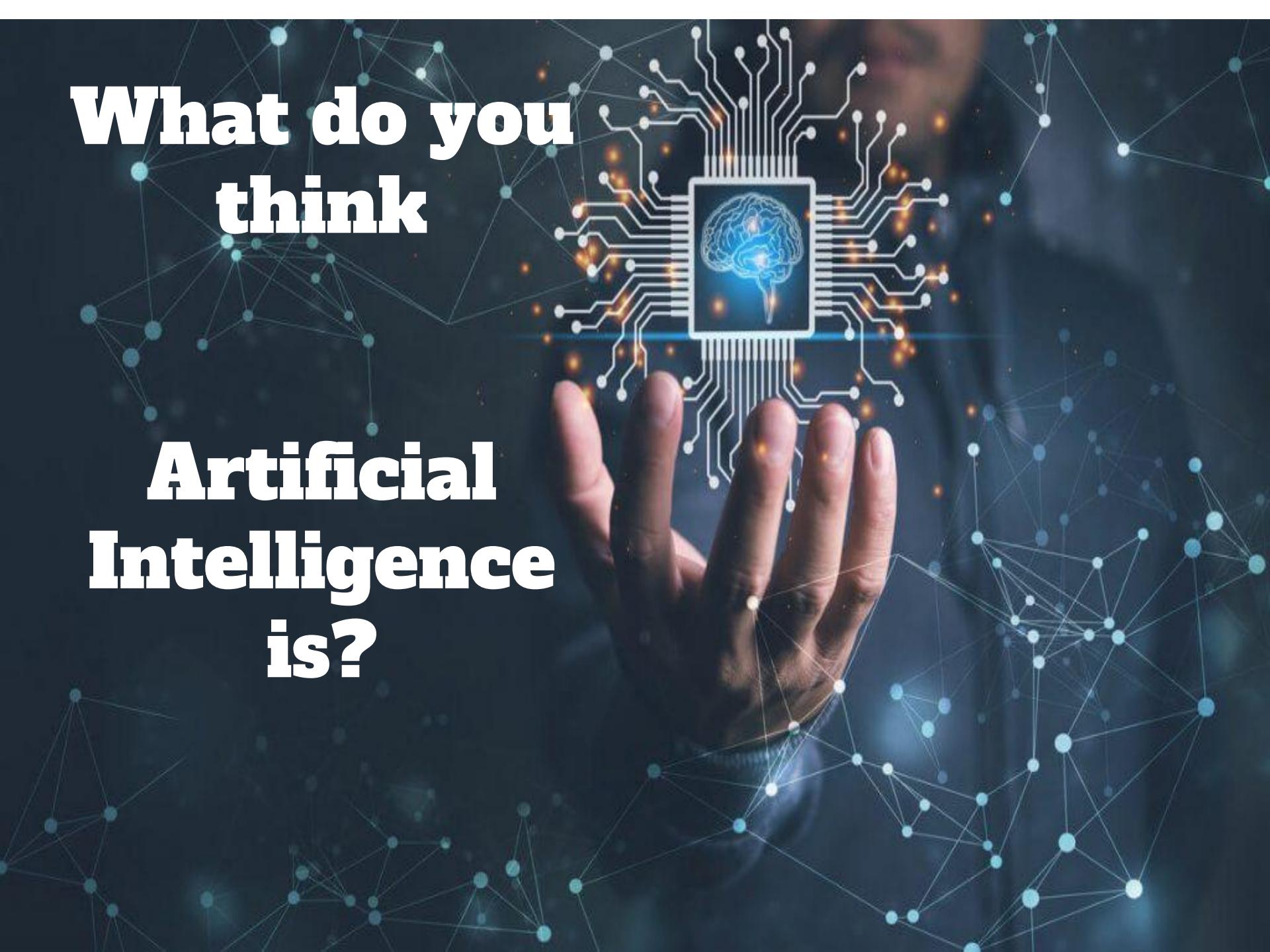
Artificial Intelligence

Lecture 1 Introduction to AI

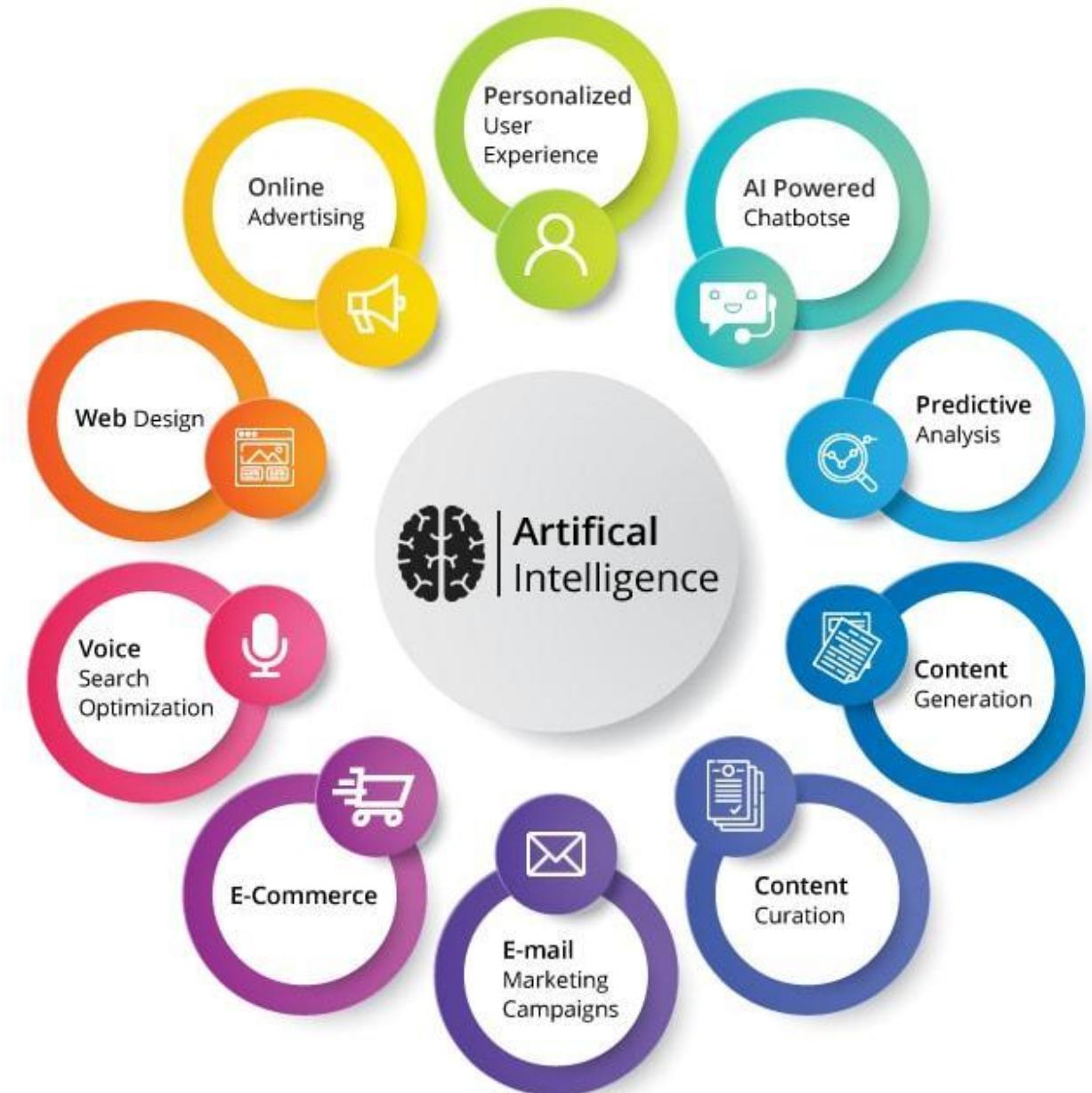
Dr. Partha Pakray

**What do you
think**

**Artificial
Intelligence
is?**

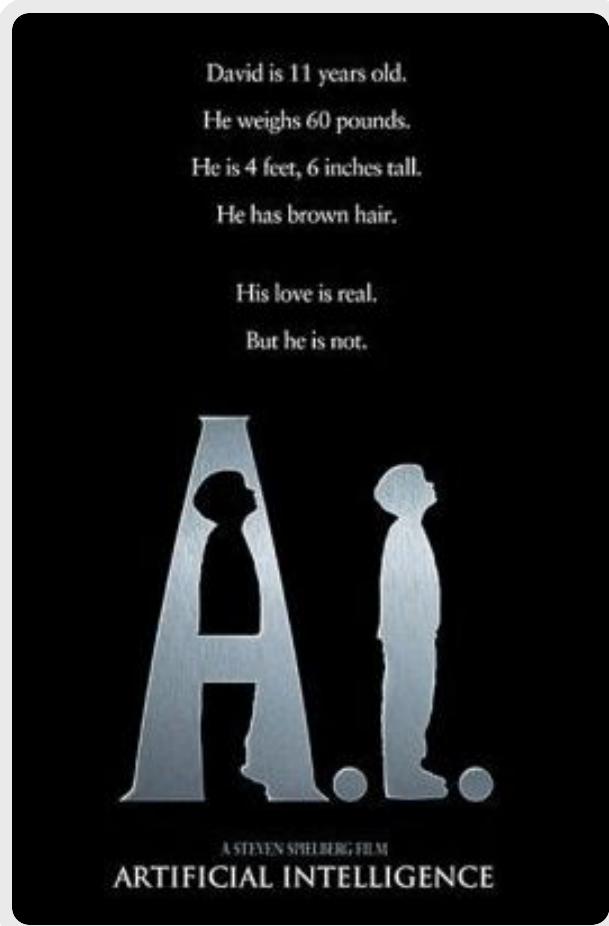


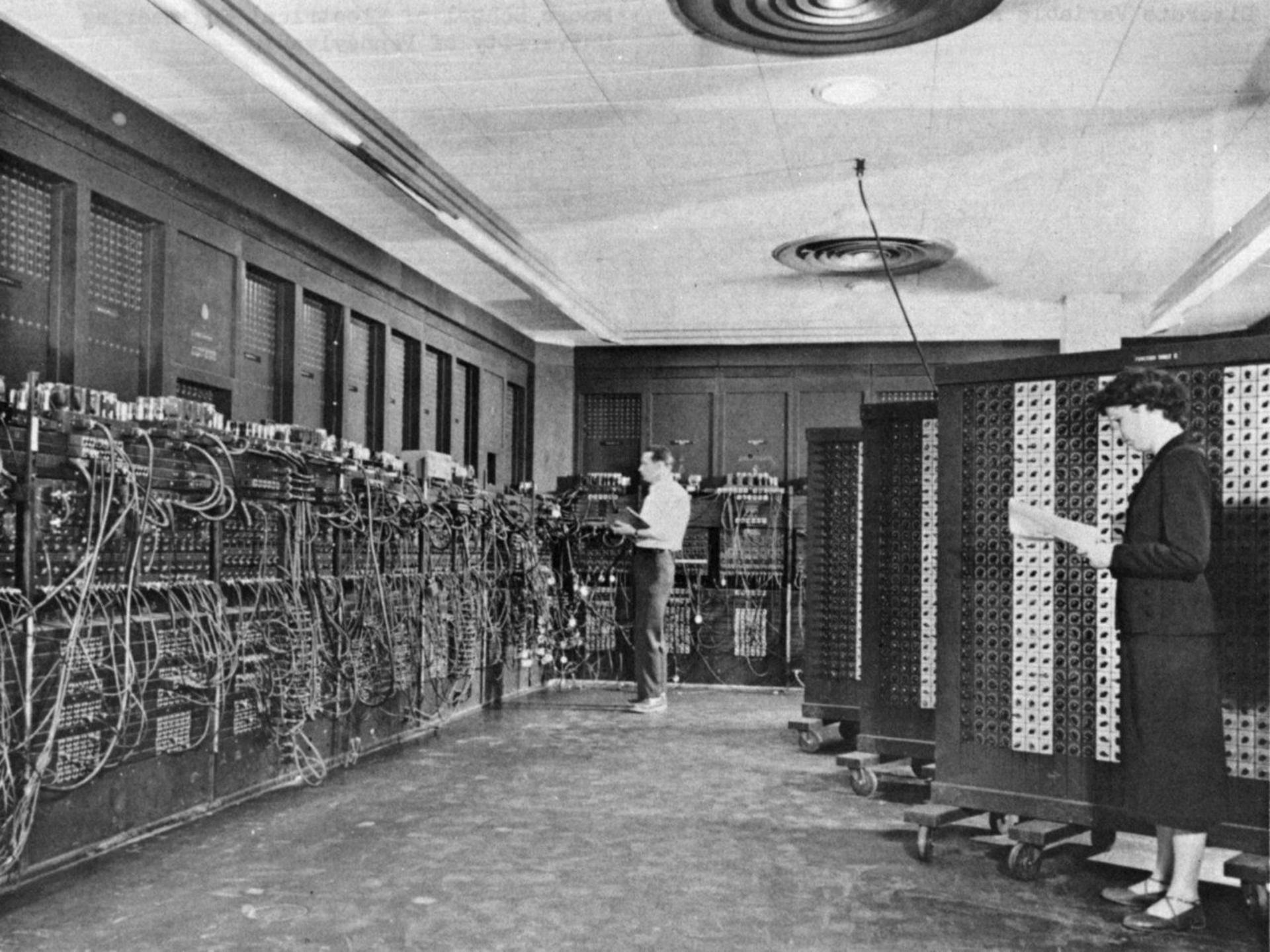
AI in Real Life



Introduction

- Brief History
- Definition of AI
- Example AI System
- AI Approaches







ENIAC (Electronic Numerical Integrator And Computer) in Philadelphia, Pennsylvania. Glen Beck (background) and Betty Snyder (foreground) program the ENIAC in building 328 at the Ballistic Research Laboratory (BRL).

ENIAC was completed in 1945 and first put to work for practical purposes on December 10, 1945.

<https://en.wikipedia.org/wiki/ENIAC>

History of AI

1943: McCulloch & Pitts: Boolean circuit model of brain

Article: “*A logical calculus of the ideas immanent in nervous activity*”

- Explaining for the first time how it is possible for neural networks to computer.

1951: Marvin Minsky and Dean Edmonds built SNARC

SNARC

Stochastic Neural Analog
Reinforcement Computer

First Neural Network Machine
Marvin Minsky, 1951

One of 40 “neurons”

In the real world, A.I.
innovation started in
the 1950's.



Source: <http://cyberneticzoo.com/mazesolvers/1951-maze-solver-minsky-edmonds-american/>

- **1950:** Turing
 - Turing's "Computing Machinery and Intelligence"
- **1956:** birth of AI
 - Dartmouth meeting: “**Artificial Intelligence**” name adopted

Source: https://www.chessprogramming.org/index.php?title=John_McCarthy&mobileaction=toggle_view_mobile



John McCarthy, (September 4, 1927 - October 23, 2011) was an American researcher in computer science and pioneer in the field of artificial intelligence. After short-term appointments at Princeton, Stanford, Dartmouth, and MIT, John McCarthy became a full professor at Stanford in 1962, where he remained until his retirement at the end of 2000. In 1955, McCarthy co-organized the Dartmouth Conference, where he coined the term Artificial intelligence, and introduced the idea of the Alpha-beta algorithm, to become their eponym. Alpha-beta was also approximated by Herbert Simon with Allen Newell and Arthur Samuel and was released to the public by Daniel Edwards and Timothy Hart in 1961 [4] and independently by Alexander Brudno in 1963. In 1958 at MIT, **John McCarthy created the Lisp programming language**.

- **1950:** Turing
 - Turing's "Computing Machinery and Intelligence"
- **1956:** birth of AI
 - Dartmouth meeting: “**Artificial Intelligence**” name adopted
- **1950s:** initial promise
 - Samuel's checkers program
 - Newell & Simon's Logic Theorem Machine
- **1955-65:** “great enthusiasm”
 - Newell and Simon: GPS, general problem solver
 - Gelertner: Geometry Theorem Prover
 - McCarthy: invention of LISP



- **1964:** Danny Bobrow shows that computers can understand *Natural Language Processing* well enough to solve algebra word problems correctly.
- **1965:** J. Allen Robinson invented a mechanical proof procedure, the *Resolution Method*, which allowed programs to work efficiently with formal logic as a representation language.
- **1966-74:** AI discovers Computational Complexity.
- **1969:** SRI robot, demonstrated combining locomotion, perception and problem solving.

- **1969-85:** Adding domain knowledge
 - Development of knowledge-based systems
 - Success of rule-based expert systems,
 - E.g., DENDRAL, MYCIN
- **1986:** Rise of machine learning
 - Neural networks return to popularity
 - Major advances in machine learning algorithms and applications
- **1990:** Role of uncertainty
 - Bayesian networks as a knowledge representation framework

- **1990's:** Major advances in all areas of AI
 - Machine Learning, data mining
 - Intelligent tutoring
 - Case based reasoning
 - Multi-agent planning, scheduling
 - Uncertain reasoning
 - Natural Language Understanding and Translation
 - Vision, Virtual Reality, games, etc.
- **1995:** *AI as Science*
 - Integration of learning, reasoning, knowledge representation
 - AI methods used in vision, language, data mining, etc

Definition of AI

- Artificial Intelligence
 - is concerned with the design of intelligence in an artificial device.
 - Term coined by McCarthy in 1956

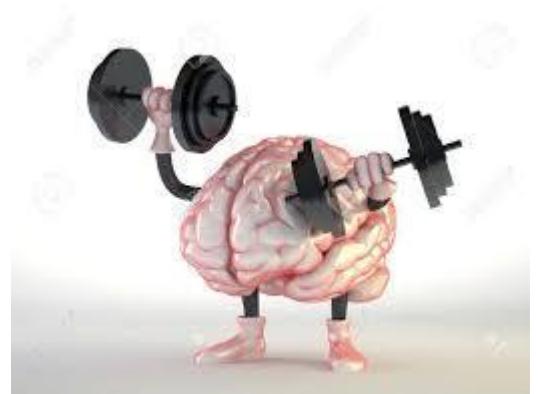
Intelligence:

“the capacity to learn and solve problems”

Artificial Intelligence:

Artificial intelligence (AI) is the intelligence of machines and robots and the branch of computer science that aims to create it

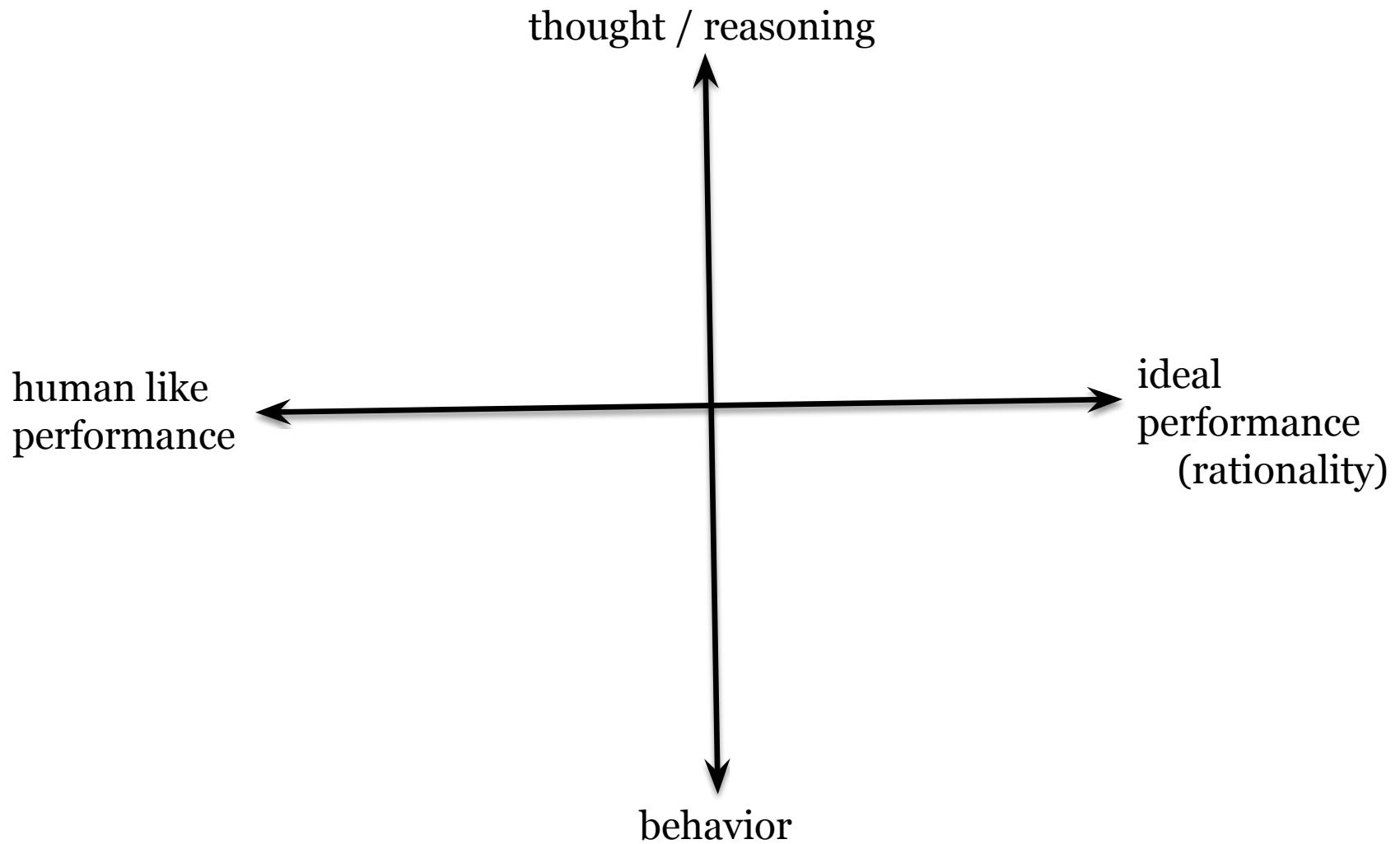
- the ability to solve problems
- the ability to act rationally
- the ability to act like humans

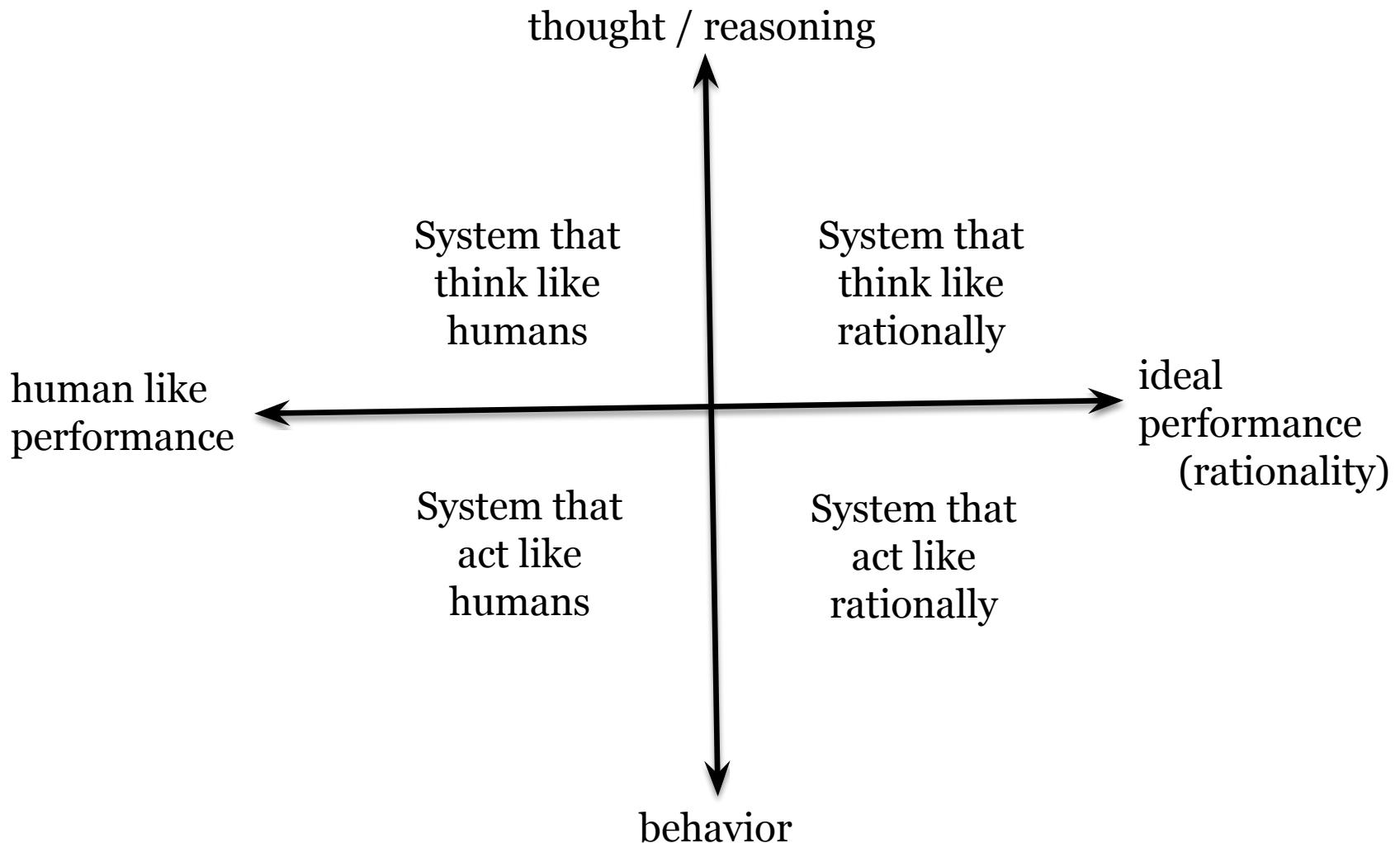


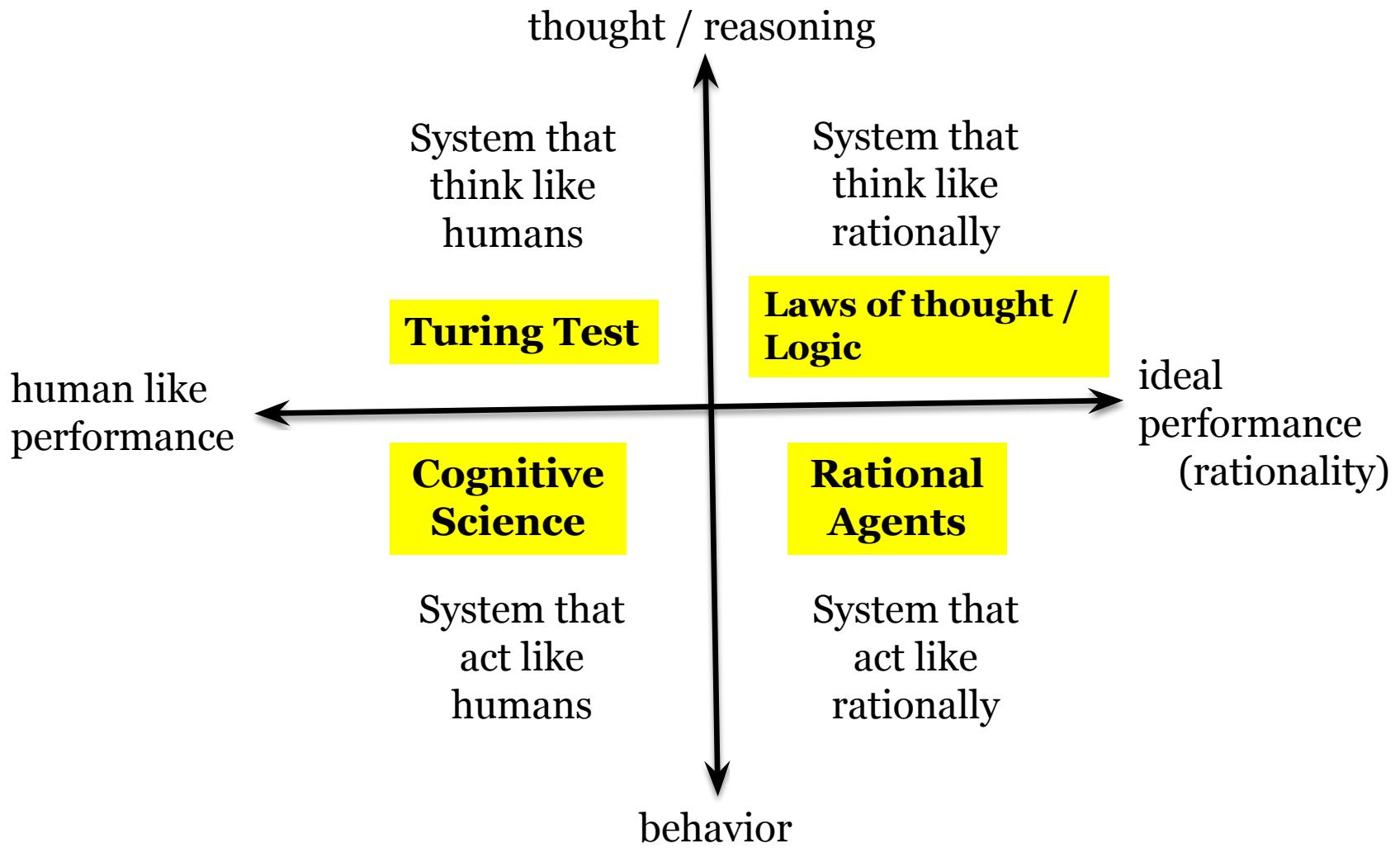
What is intelligence?

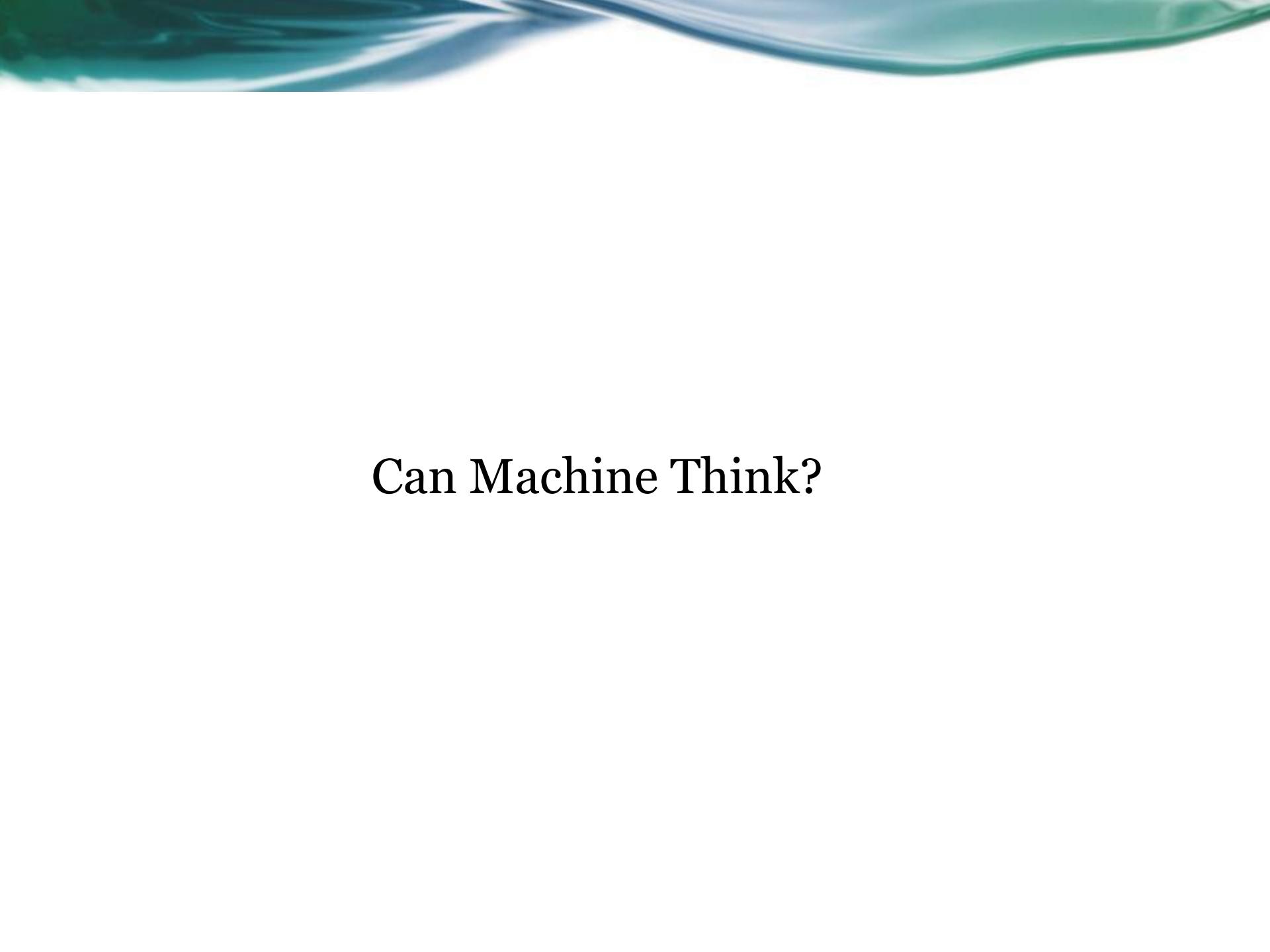
- Behave as intelligently as a human
- Behave in the best possible manner
- Thinking
- Acting

- What to look at
 - *thought processes/reasoning* vs. *behavior*
- How to **measure performance**
 - *Human-like performance* vs. *ideal performance*







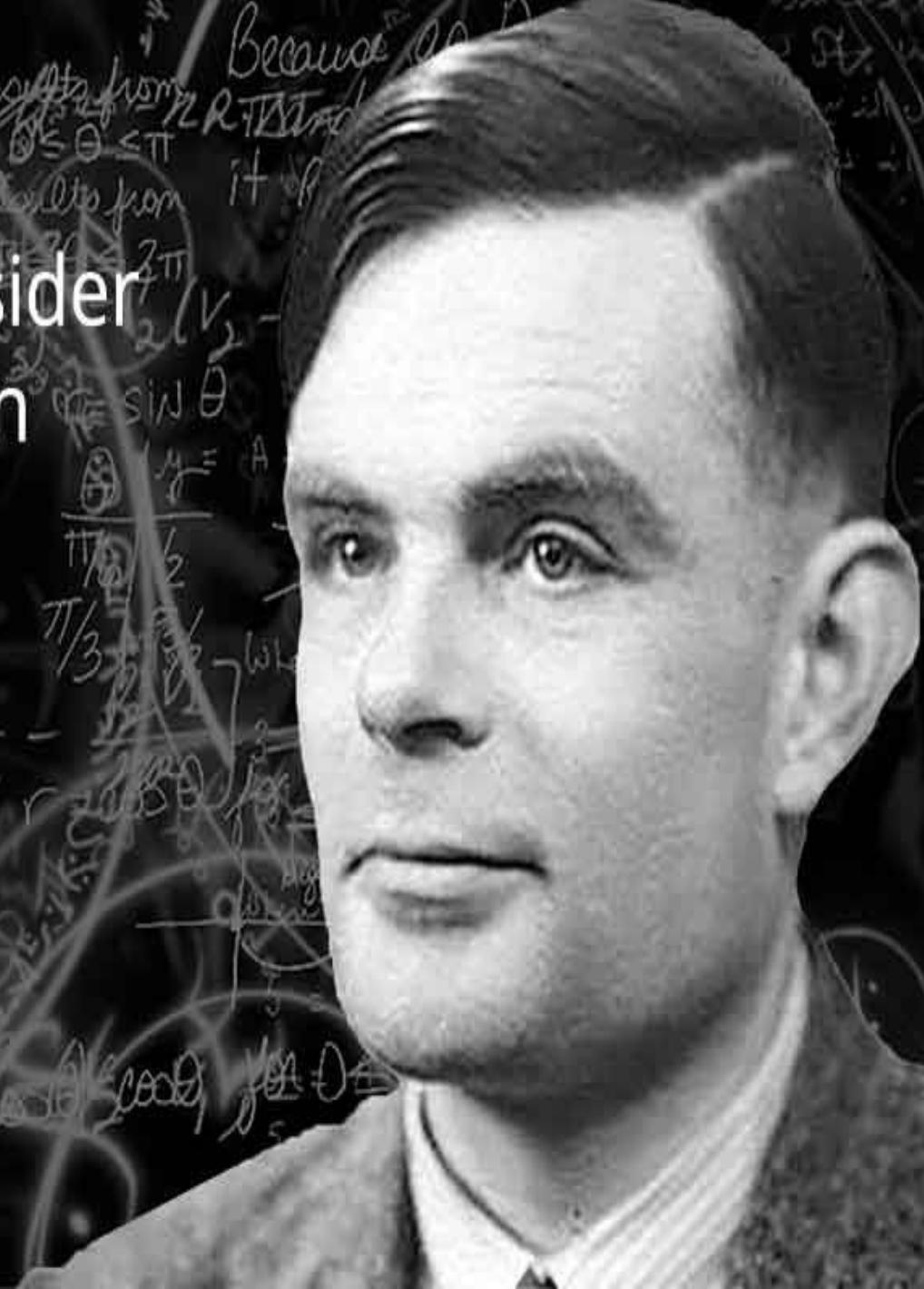


Can Machine Think?

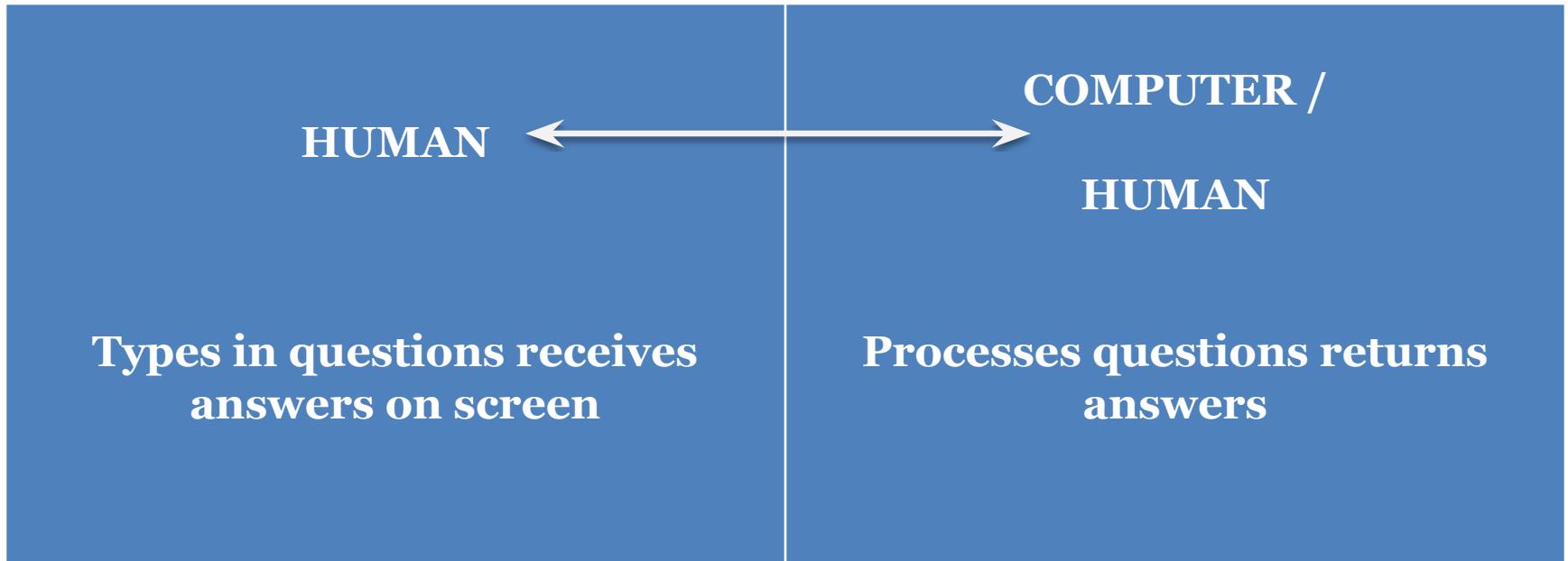
"I propose to consider
the question, 'Can
machines think?'

~ Alan Turing

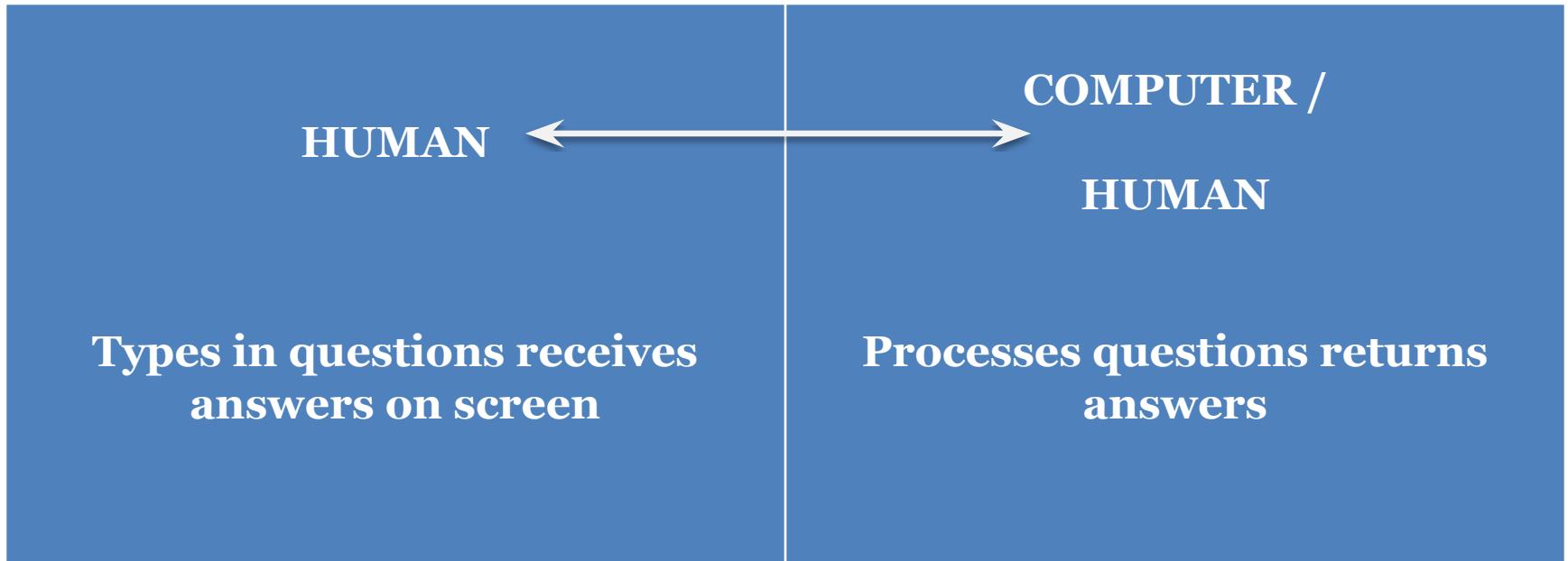
Carnegie Mellon University
Machine Learning



The Turing Test



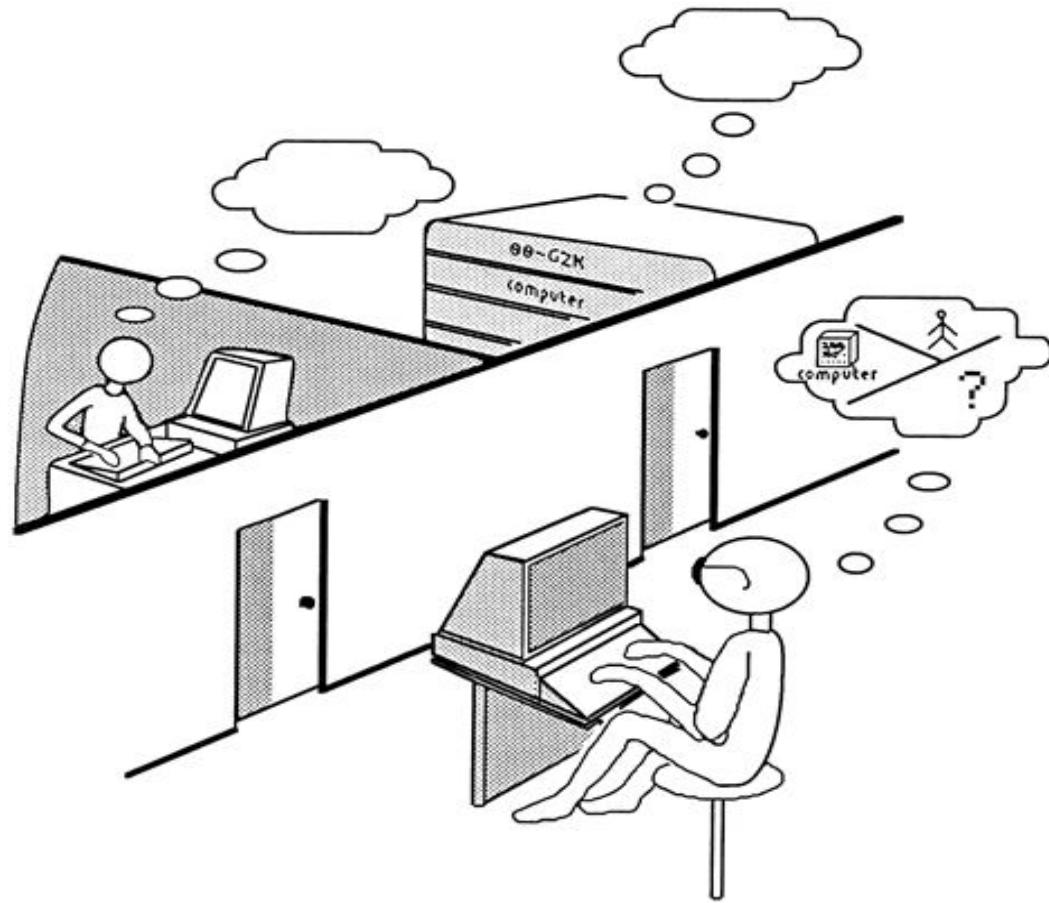
The Turing Test



Both claim to be the (intelligent) human

Interrogator must decide who is human?

The Turing Test



Result: The Turing Test

If the **interrogator** cannot reliably distinguish the human from the computer

Then the computer does possess (artificial) intelligence

Typical AI Problems

- Intelligent entities (or “Agents”) need to be able to do both “mundane” and “expert” tasks:
- “**Mundane**” Tasks:
 - Planning route, activity.
 - Recognizing (through vision) people, objects.
 - Communicating (through Natural Language)
 - Navigating round obstacles on the street
- “**Experts**” Task:
 - Medical Diagnosis
 - Mathematical Problem solving

What's easy and what's hard?

- It has been easier to mechanize many of the high-level tasks we usually associate with “intelligence” in people.
 - Symbolic integration
 - Proving theorems
 - Playing chess
 - Medical diagnosis

- It has been very hard to mechanize tasks that lots of animals can do
 - Walking around without running into things
 - Catching prey and avoiding predators
 - Interpreting complex sensory information
 - Modeling the internal states of other animals from their behavior

Intelligent behavior

- Perception
- Reasoning
- Learning
- Understanding Language
- Solving Problems

Application

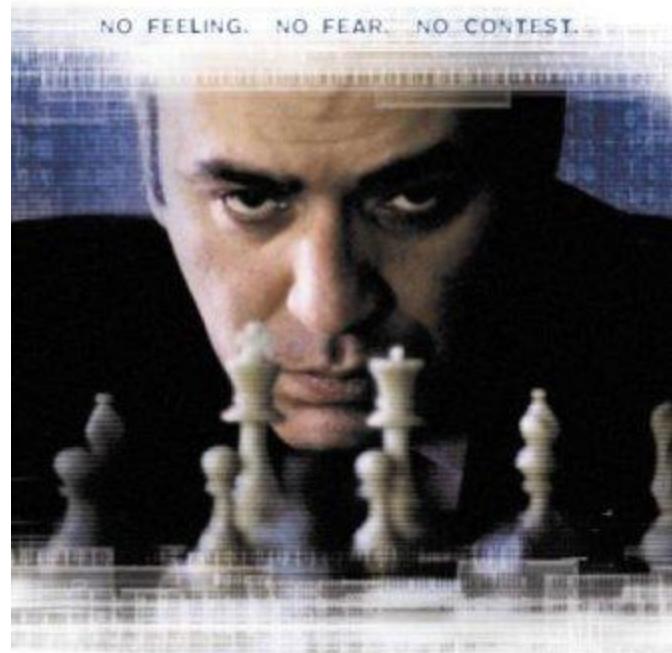
- Computer Vision
- Image Recognition
- Robotics
- Language Processing
- etc.

Practical Impact of AI

- AI Components are embedded in numerous devices e.g. copy machines.
- AI systems are in everyday use
 - Detecting credit card fraud
 - Configuring products
 - Aiding complex planning tasks
 - Advising physicians
- Intelligent tutoring systems provide students with personalized attention

Deep Blue

- 1997: The Deep Blue chess program beats. The current world chess champion, Gary Kasparov, in a widely followed match.



GAME OVER
KASPAROV AND THE MACHINE

"Engrossing, tragic and above all, entertaining. See it" ★★★★ Film Review

"Fascinating. Remarkable. Gripping." ★★★★ Johnny Vaughn, The Sun



AI Topics

Core areas

Knowledge representation
Reasoning
Machine Learning

Perception

Vision
Natural Language
Robotics

Uncertainty

Probabilistic approaches

General Algorithm

Search
Planning
Constraint satisfaction

Applications

Game playing
AI and education
Distributed agents

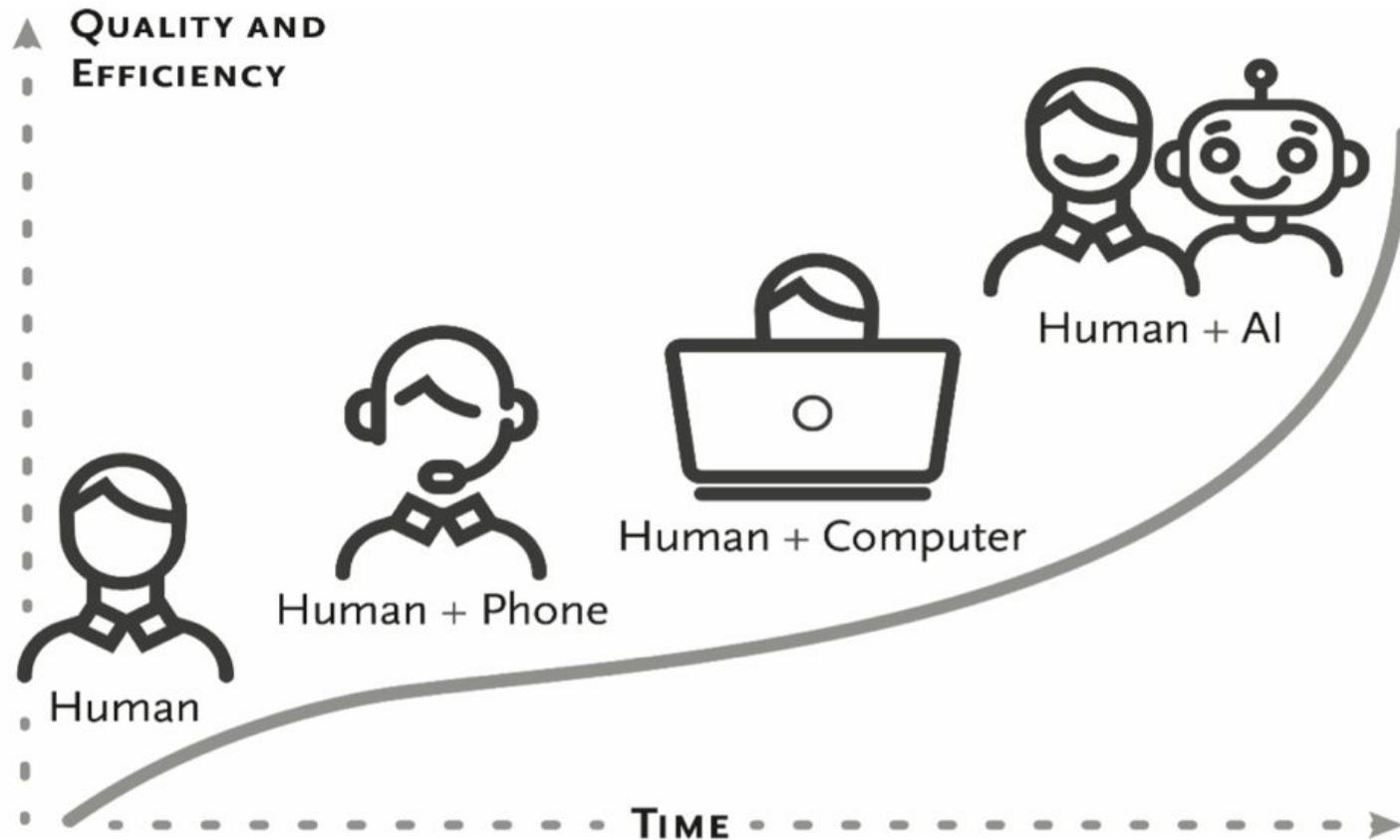
Decision Theory

Reasoning with symbolic data

Artificial Intelligence

Lecture 2: Agent and Environment

1. Define intelligence.
2. What are the different approaches in defining artificial intelligence?
3. What is Turing test?
4. Suppose you design a machine to pass the Turing test. What are the capabilities such a machine must have?



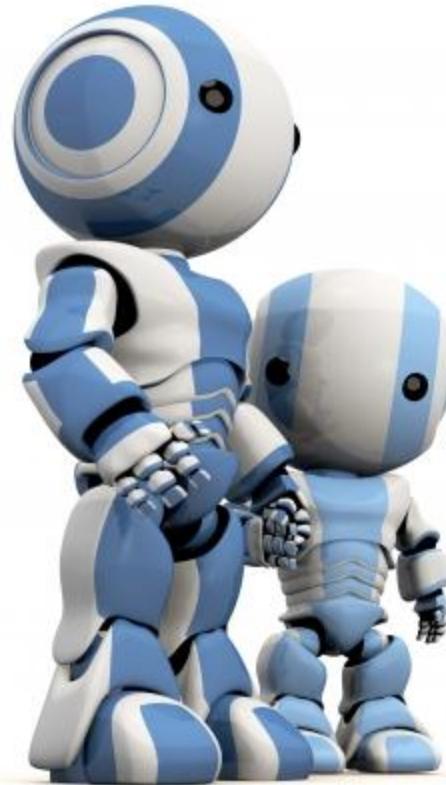
2. What are the different approaches in defining artificial intelligence?

- Thinking rationally
- Acting rationally
- Thinking like a human
- Acting like a human

4. Suppose you design a machine to pass the Turing test. What are the capabilities such a machine must have?

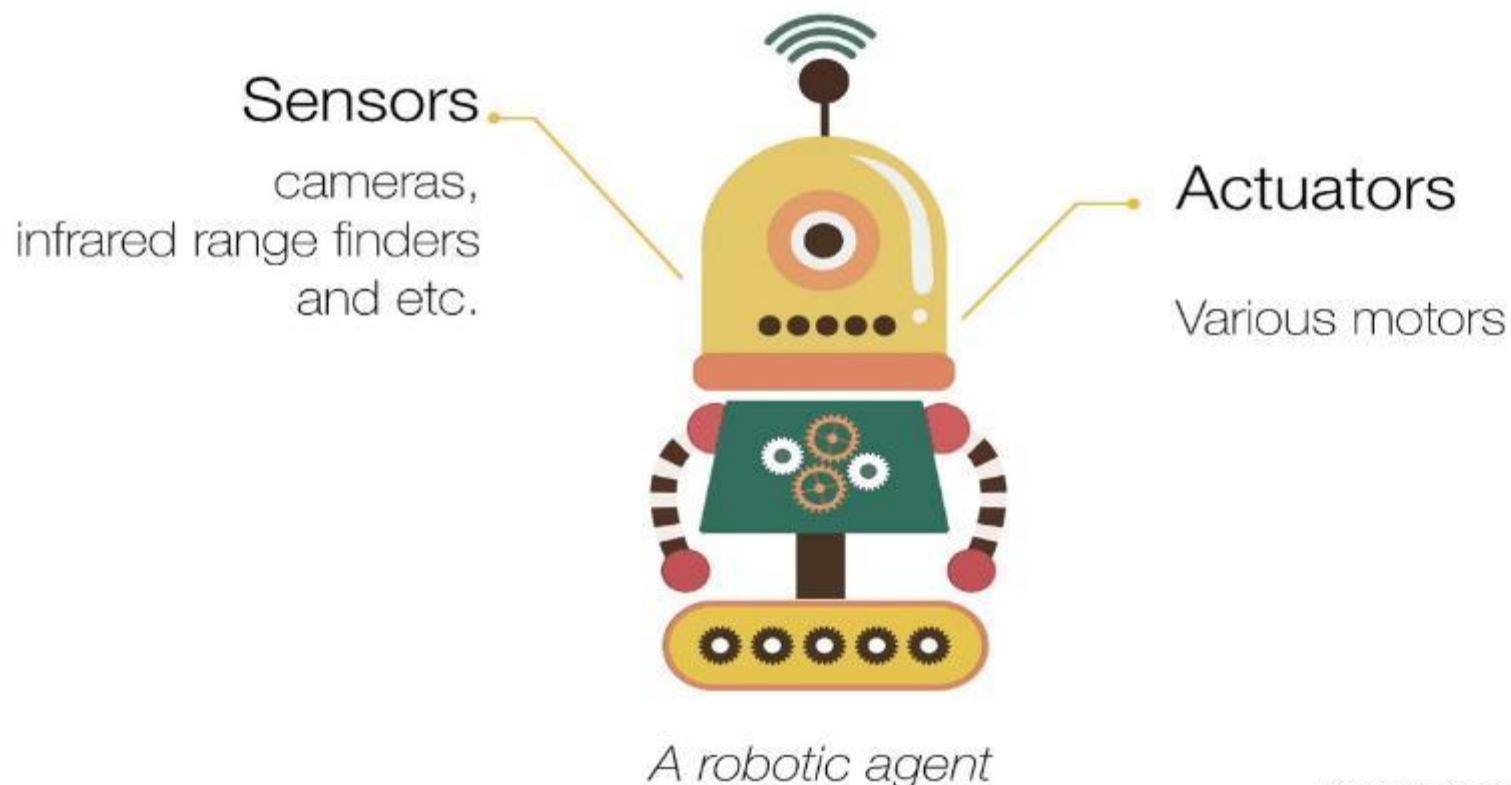
- Natural language processing
- Knowledge representation
- Automated reasoning
- Machine Learning
- Computer vision
- Robotics

- Define an Agent
- Define an Intelligent Agent
- Define an Rational Agent
- Explain Bounded Rationality
- Discuss different types of environments
- Explain different agent architecture



Agents

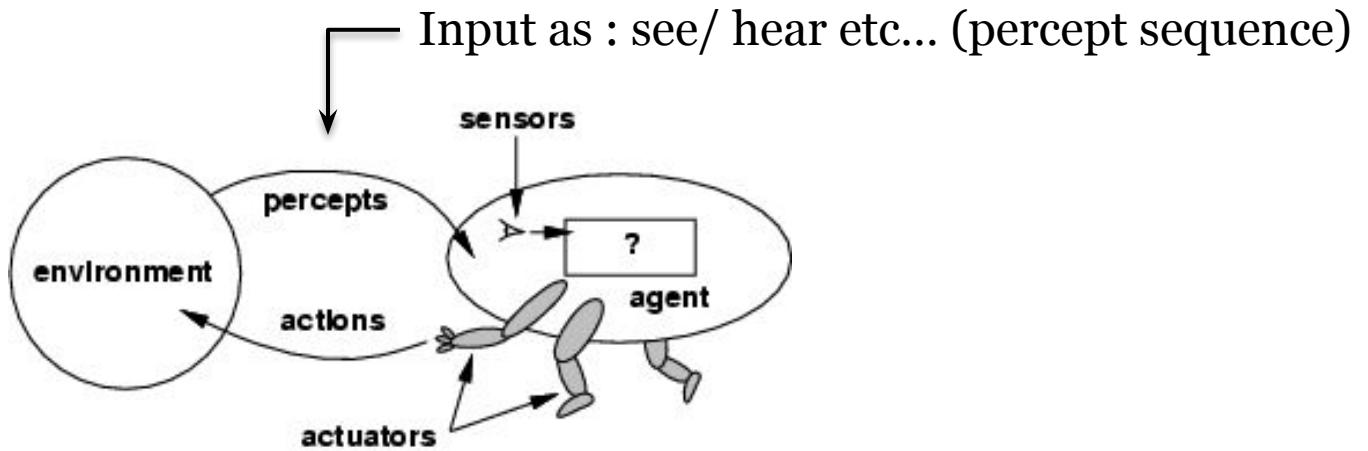
- An **agent** is anything that can be viewed as **perceiving** its **environment** through **sensors** and **acting** upon that environment through **actuators**



Agents

- An **agent** is anything that can be viewed as **perceiving** its **environment** through **sensors** and **acting** upon that environment through **actuators**
- Human agent: eyes, ears, and other organs for **sensors**; hands, legs, mouth, and other body parts for **actuators**
- Robotic agent: cameras and infrared range finders for **sensors**; various motors for **actuators**

Agents & Environments

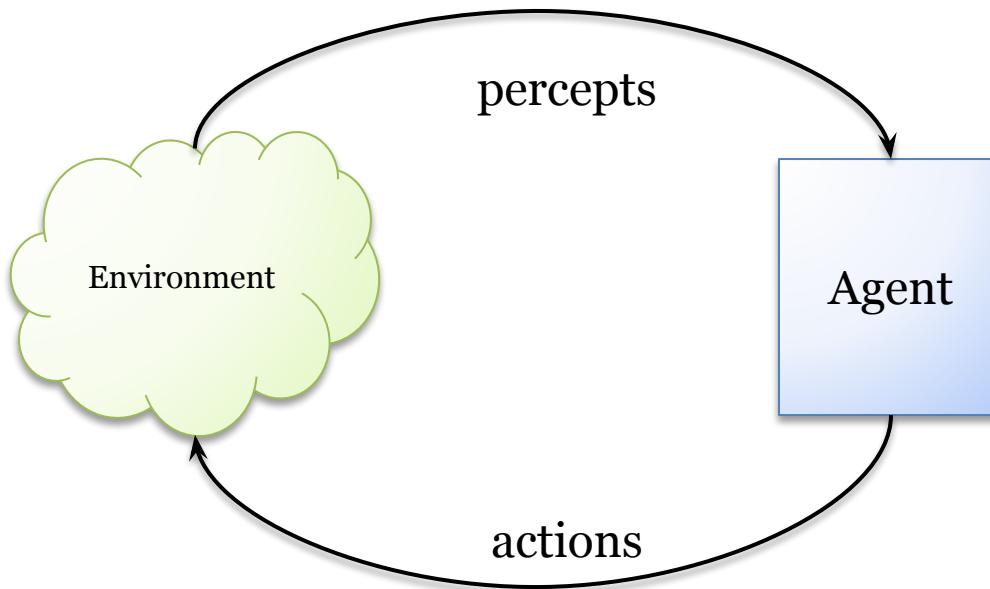


- ❖ Operate in an environment
- ❖ Perceives its environment through sensors
- ❖ Acts upon its environment through actuators / effectors
- ❖ Have Goals

Sensors & Effectors

- An Agent perceives its environment through sensors
 - The complete set of inputs at a given time is called a **percept**
 - The current percept, or a sequence of percepts can influence the actions of an agent
- It can change the environment through actuators / effectors
 - An operation involving an actuator is called an **action**
 - Action can be grouped into action **Sequence**

Agents



- Implement mapping from percept sequence to actions
- Performance measure to evaluate agents
- Autonomous agent: decide autonomously which action to take in the current situation to maximize progress towards its goals.

Performance

- Behavior and performance of IAs in terms of agent function
 - Perception history (sequence) to Action Mapping
 - Ideal mapping: specifies which action an agent should take at any point in time.
- Performance measure: a subjective measure to characterize how successful an agent is (e.g., speed, power usage, accuracy, money, etc..)

Example of Agents

- Humans:
 - Eyes, ears, skin, taste buds, etc. for Sensors
 - Hand, fingers, legs, mouth for Effectors
- Robot:
 - Camera, infrared, etc. for Sensors
 - wheels, lights, speakers etc. for Actuators
- Software Agents (softbot)
 - Functions as sensors
 - Functions as actuators

Types of Agents

- Softbots
 - ask.com
- Expert System
 - Cardiologist
- Autonomous Spacecraft
- Intelligent buildings

Agent Faculties

The fundamental faculties of intelligence are

- Acting
- Sensing
- Understanding, reasoning, learning

- In order to act intelligently, **one must sense**. Blind action is not a characterization of intelligence.
- Understanding is essential to interpret the sensory percepts and decide on an action.
- Many robotic agents stress sensing and acting, and do not have understanding.

Intelligent Agent

- Must Sense
- Must Act
- Must be autonomous (to some extent)
- Must be rational

Rationality

- AI is about building rational agents.
- An agent is something that perceives and acts.
- A rational agent always does the right thing.
 1. What are the functionalities (goals)?
 2. What are the components?
 3. How do we build them?

Perfect Rationality

- Perfect Rationality assumes that the rational agent knows all and will take the action that maximizes his utility.
- Human beings do not satisfy this definition of rationality.

Rational Action

- **Rational Action** is the action that maximizes the expected value of the performance measure given the percept sequence to date
- Is Rational Action means best action? <Rational = Best>?
 - Yes, to the best of its agent knowledge
- Does Rational means Optimal? <Rational = Optimal>?
 - Yes, to the best of its abilities
 - And its constraints
- In 1957, Simon proposed the notion of Bounded Rationality:

Bounded Rationality

“Because of the limitations of the human mind, humans must use approximate methods to handle many tasks.” Herbert Simon, 1972

Agent Environment

- Environments in which agents operate can be defined in different ways. It is helpful to view the following definitions as referring to the way the environment appears from the point of view of the agent itself.

Environment: Observability

- **Fully observable** environment
 - all of the environment relevant to the action being considered is observable.
 - In such environments, the agent does not need to keep track of the changes in the environment.
 - Example: A chess playing system is an example of a system that operates in a fully observable environment.
- **Partially observable** environment
 - the relevant features of the environment are only partially observable.
 - Example: A bridge playing program is an example of a system operating in a partially observable environment.

Environment: Determinism

- **Deterministic:** In ***deterministic environments***, the next state of the environment is **completely described by the current state and the agent's action**. *Image analysis* systems are examples of this kind of situation. The processed image is determined completely by the current image and the processing operations.
- **Stochastic:** If an element of interference or uncertainty occurs then the environment is stochastic. Note that a deterministic yet **partially observable** environment will appear to be stochastic to the agent. Examples of this are the automatic vehicles that navigate a terrain, say, the Mars rovers robot. The new environment in which the vehicle is in is stochastic in nature.

- **Strategic:** If the environment state is wholly determined by the preceding state and the actions of multiple agents, then the environment is said to be strategic. Example: Chess. There are two agents, the players and the next state of the board is strategically determined by the players' actions.

Environment: Episodicity

- An **episodic environment** means that subsequent episodes **do not depend on what actions occurred in previous episodes.**
- In a **sequential environment**, the agent **engages in a series of connected episodes.**

Environment: Dynamism

- **Static Environment:** does not change from one state to the next while the agent is considering its course of action. The only changes to the environment are those caused by the agent itself.
 - A static environment does not change while the agent is thinking.
 - The passage of time as an agent deliberates is irrelevant.
 - The agent doesn't need to observe the world during deliberation.
- **Dynamic Environment:** A Dynamic Environment changes over time independent of the actions of the agent -- and thus if an agent does not respond in a timely manner, this counts as a choice to do nothing.
 - Example: Interactive tutor

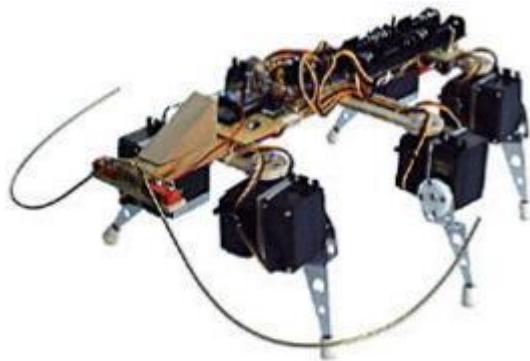
Environment: Continuity

- **Discrete/Continuous**
 - If the number of distinct percepts and actions is limited, the environment is discrete, otherwise it is continuous.

Examples of Agents



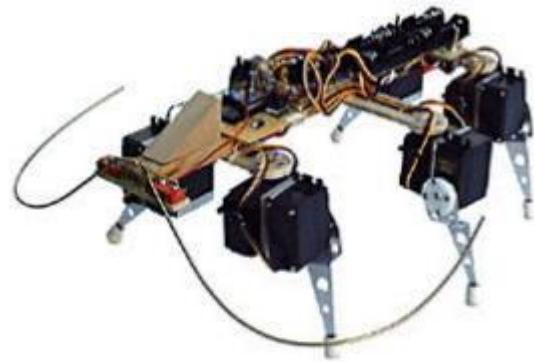
```
Else
    If (Len(Trim$(strI)
        blnFileNext =
    Else
        ' Process data
        intColon = InSt
        If (intColon
            strName =
            strValue =
            strName = S
            Call xnNod
        End If
    End If
End If
```



Examples of Agents



```
Else
    If (Len(Trim$(strI)
        blnFileNext =
    Else
        ' Process data
        intColon = InSt
        If (intColon
            strName =
            strValue =
            strName = S
            Call xnNode
        End If
    End If
End If
```



Humans

Programs

Robots

senses

keyboard, mouse, dataset

cameras, pads

body parts

monitor, speakers, files

motors, limbs

What is Rational Agents?

Rational Agents

A **rational agent** is one that does the right thing

- Need to be able to assess agent's performance
 - Should be independent of internal measures
- Ask yourself: has the agent acted rationally?
 - Not just dependent on how well it does at a task
- First consideration: evaluation of rationality



Artificial Intelligence

Lecture 3: Agent Classification

Book

Artificial Intelligence: A Modern Approach

Stuart Russell
Peter Norvig

PEARSON Publication

1. Define an agent?
2. What is a rational agent ?
3. What is bounded rationality ?
4. What is an autonomous agent ?
5. Describe the salient features of an agent.



"Surely computers cannot be intelligent; they can do only what their programmers tell them". Is the latter statement true, and does it imply the former?

"Surely computers cannot be intelligent; they can do only what their programmers tell them". Is the latter statement true, and does it imply the former?

The statement that the computers can do only what their programmers tell them is ambiguous. It is true that computers cannot be intelligent because computers are machines that do not have knowledge. Thus, they are not intelligent. They act according to the instructions given by the humans (programmers). They cannot act on their own and they cannot make the decisions by themselves. They are dependent on the algorithms. As they completely depend on the knowledge and instructions of programmers, computers cannot be intelligent.

Understand Types of Environments in Artificial Intelligence

Fully Observable vs Partially-Observable

Real-life Example?

Understand Types of Environments in Artificial Intelligence

Fully Observable vs Partially-Observable

In a fully observable environment, The Agent is familiar with the complete state of the environment at a given time. There will be no portion of the environment that is hidden for the agent.

Real-life Example: While running a car on the road (Environment), The driver (Agent) is able to see road conditions, signboard and pedestrians on the road at a given time and drive accordingly. So Road is a fully observable environment for a driver while driving the car.

In a partially observable environment, The agent is not familiar with the complete environment at a given time.

Real-life Example: Playing card games is a perfect example of a partially-observable environment where a player is not aware of the card in the opponent's hand. Why partially-observable? Because the other parts of the environment, e.g opponent, game name, etc are known for the player (Agent).

Deterministic vs Stochastic

Real-life Example?

Deterministic vs Stochastic

Deterministic are the environments where the next state is observable at a given time. So there is no uncertainty in the environment.

Real-life Example: The traffic signal is a deterministic environment where the next signal is known for a pedestrian (Agent)

The Stochastic environment is the opposite of a deterministic environment. The next state is totally unpredictable for the agent. So randomness exists in the environment.

Real-life Example: The radio station is a stochastic environment where the listener is not aware about the next song or playing a soccer is stochastic environment.

Episodic vs Sequential

Real-life Example?

Episodic vs Sequential

Episodic is an environment where each state is independent of each other. The action on a state has nothing to do with the next state.

Real-life Example: A support bot (agent) answer to a question and then answer to another question and so on. So each question-answer is a single episode.

The sequential environment is an environment where the next state is dependent on the current action. So agent current action can change all of the future states of the environment.

Real-life Example: Playing tennis is a perfect example where a player observes the opponent's shot and takes action.

Static vs Dynamic

Real-life Example?

Static vs Dynamic

The Static environment is completely unchanged while an agent is perceiving the environment.

Real-life Example: Cleaning a room (Environment) by a dry-cleaner robot (Agent) is an example of a static environment where the room is static while cleaning.

Dynamic Environment could be changed while an agent is perceiving the environment. So agents keep looking at the environment while taking action.

Real-life Example: Playing soccer is a dynamic environment where players' positions keep changing throughout the game. So a player hits the ball by observing the opposite team.

Discrete vs Continuous

Real-life Example?

Discrete vs Continuous

Discrete Environment consists of a finite number of states and agents have a finite number of actions.

Real-life Example: Choices of a move (action) in a tic-tac game are finite on a finite number of boxes on the board (Environment).

While in a Continuous environment, the environment can have an infinite number of states. So the possibilities of taking an action are also infinite.

Real-life Example: In a basketball game, the position of players (Environment) keeps changing continuously and hitting (Action) the ball towards the basket can have different angles and speed so infinite possibilities.

Single Agent vs Multi-Agent

Real-life Example?

Single Agent vs Multi-Agent

Single agent environment where an environment is explored by a single agent. All actions are performed by a single agent in the environment.

Real-life Example: Playing tennis against the ball is a single agent environment where there is only one player.

If two or more agents are taking actions in the environment, it is known as a multi-agent environment.

Real-life Example: Playing a soccer match is a multi-agent environment.

Characterizing a Task Environment

- Must first specify the setting for intelligent agent design.
- **PEAS: Performance measure, Environment, Actuators, Sensors**
- **Example:** the task of designing a self-driving car



- **Performance measure** Safe, fast, legal, comfortable trip
- **Environment** Roads, other traffic, pedestrians
- **Actuators** Steering wheel, accelerator, brake, signal, horn
- **Sensors** Cameras, LIDAR (light/radar), speedometer, GPS, odometer, engine sensors, keyboard

Assignment-I

**#DateofSubmission 03/10/2021
(through google classroom)**

Assignment 1: For Flipkart shopping agent, what is the Agent Type, Environment, Actuators, Performance Measure, Sensors, Agent Architecture?

Assignment 2: Suppose you want to develop an automated taxi, so what will be the Agent Type, Environment, Actuators, Performance Measure, Sensors?

Assignment 3: Write the names in side the table

Task Environment	Observable	Deterministic	Episodic	Static	discrete	Agents
Crossword puzzle						
Chess with a clock						
Poker						
Taxi driving						
Medical Diagnosis						
Image Analysis						
Interactive Tutoring system (English Tutor)						

Structure of agents

Agent = architecture + program

Architecture = some sort of computing device
(sensors + actuators)

(Agent) Program = some function that implements
the agent mapping = “?”

Agent Program = Job of AI

Agent architectures

- *Table based agent*
- *Percept based agent or reflex agent*
- *State-based Agent or model-based reflex agent*
- *Goal-based Agent*
- *Utility-based Agent*
- *Learning Agent*

Table based agent

- In table based agent the action is looked up from a table based on information about the agent's percepts.
- A table is simple way to specify a mapping from percepts to actions. The mapping is implicitly defined by a program.
- The mapping may be implemented by a rule based system, by a neural network or by a procedure.

```
function TABLE-DRIVEN-AGENT(percept) returns action
  static: percepts, a sequence, initially empty
           table, a table, indexed by percept sequences, initially fully specified
  append percept to the end of percepts
  action  $\leftarrow$  LOOKUP(percepts, table)
  return action
```

Disadvantages:

- The tables may become very large.
- Learning a table may take a very long time, especially if the table is large.

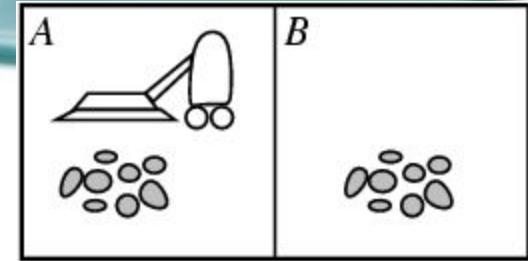
Toy example:
Vacuum world.

Percepts: robot senses its **location** and “cleanliness.”

So, **location** and **contents**, e.g., [A, Dirty], [B, Clean].

With 2 locations, we get **4 different possible sensor inputs**.

Actions: *Left, Right, Suck, NoOp*



Percept sequence	Action
[A, Clean]	Right
[A, Dirty]	Suck
[B, Clean]	Left
[B, Dirty]	Suck
[A, Clean], [A, Clean]	Right
[A, Clean], [A, Dirty]	Suck
:	:
[A, Clean], [A, Clean], [A, Clean]	Right
[A, Clean], [A, Clean], [A, Dirty]	Suck
:	:

Figure 2.3 Partial tabulation of a simple agent function for the vacuum-cleaner world shown in Figure 2.2.

Percept based agent or reflex agent

- In percept based agents,
 - information comes from **sensors - percepts**
 - changes the agents current **state of the world**
 - triggers **actions** through the **effectors**
- Such agents are called **reactive agents** or **stimulus-response agents**.

Reactive agents have no notion of history. The current state is as the sensors see it right now. The action is based on the current percept only.

```
function SIMPLE-REFLEX-AGENT(percept) returns action
  static: rules, a set of condition-action rules
```

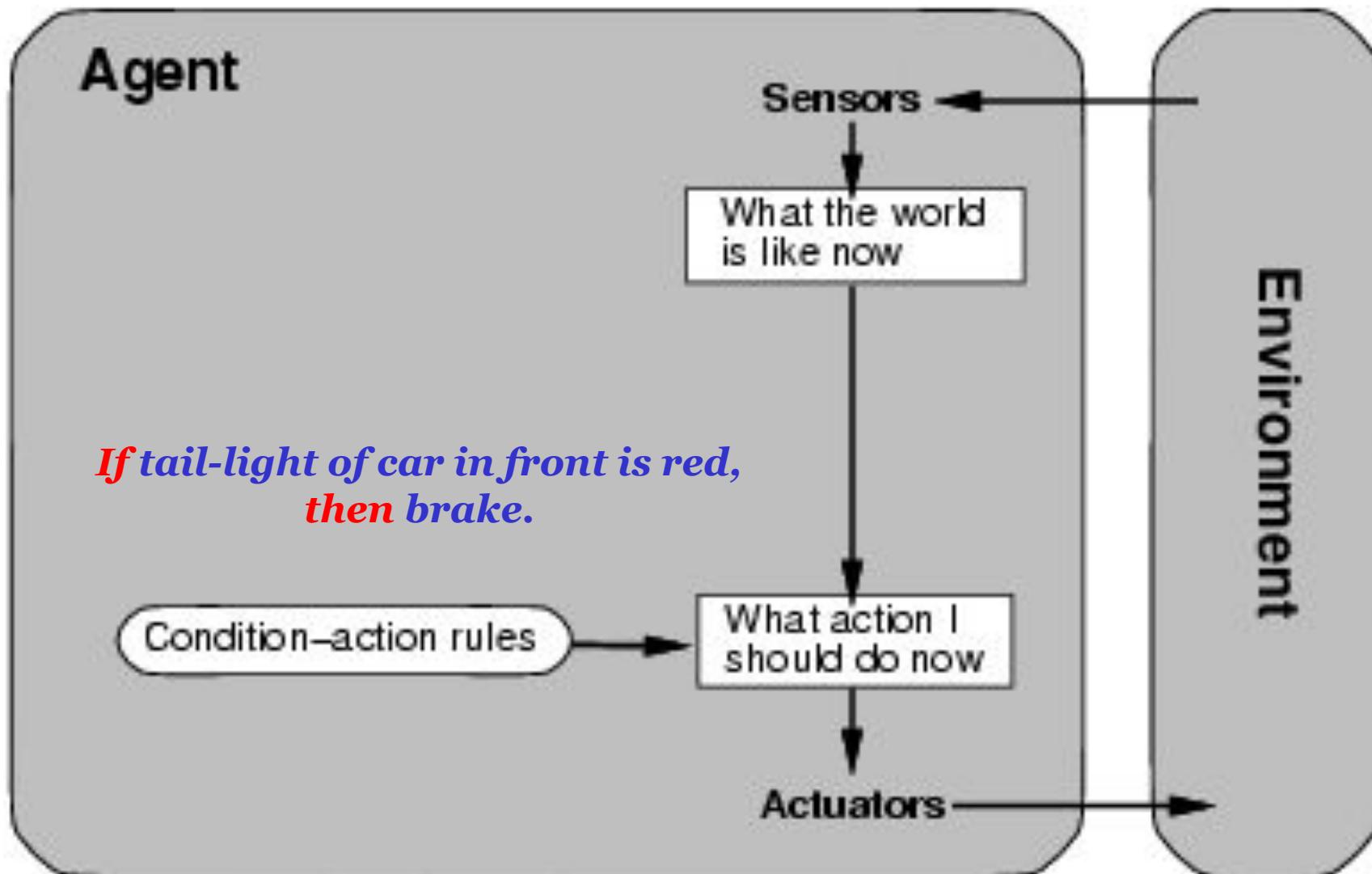
```
    state  $\leftarrow$  INTERPRET-INPUT(percept)
```

```
    rule  $\leftarrow$  RULE-MATCH(state, rules)
```

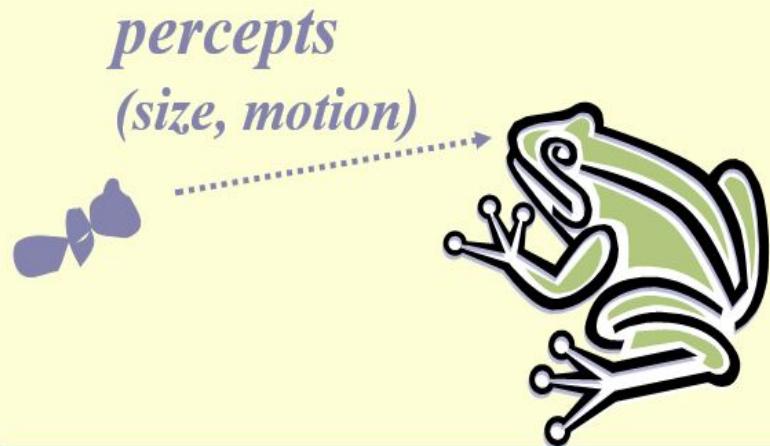
```
    action  $\leftarrow$  RULE-ACTION[rule]
```

```
    return action
```

**Agent selects actions on the basis
of *current* percept only.**



A Simple Reflex Agent in Nature



RULES:

- (1) If small moving object,
then activate SNAP
 - (2) If large moving object,
then activate AVOID and inhibit SNAP
- ELSE (not moving) then NOOP

needed for
completeness

Action: SNAP or AVOID or NOOP

- The following are some of the characteristics of percept-based agents.
 - Efficient
 - No internal representation for reasoning, inference.
 - No strategic planning, learning.
 - Percept-based agents are not good for multiple, opposing, goals.

State-based Agent or model-based reflex agent

- state based agent works as follows:
 - information comes from **sensors – percepts**
 - based on this, the agent changes the current state of the world
 - based on state of the world and knowledge (memory), it triggers actions through the effectors
 - E.g., driving a car and changing lane

Requiring two types of knowledge

- How the world evolves independently of the agent
- How the agent's actions affect the world

```
function REFLEX-AGENT-WITH-STATE(percept) returns action
  static: state, a description of the current world state
           rules, a set of condition-action rules

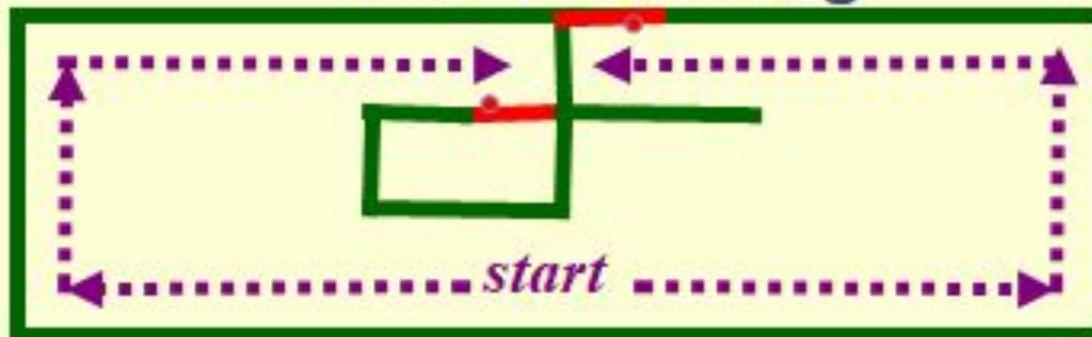
  state  $\leftarrow$  UPDATE-STATE(state, percept)
  rule  $\leftarrow$  RULE-MATCH(state, rules)
  action  $\leftarrow$  RULE-ACTION[rule]
  state  $\leftarrow$  UPDATE-STATE(state, action)
  return action
```

The agent is with memory

Example Table Agent With Internal State

IF	THEN
Saw an object ahead, and turned right, and it's now clear ahead	Go straight
Saw an object Ahead, turned right, and object ahead again	Halt
See no objects ahead	Go straight
See an object ahead	Turn randomly

Example Reflex Agent With Internal State: Wall-Following

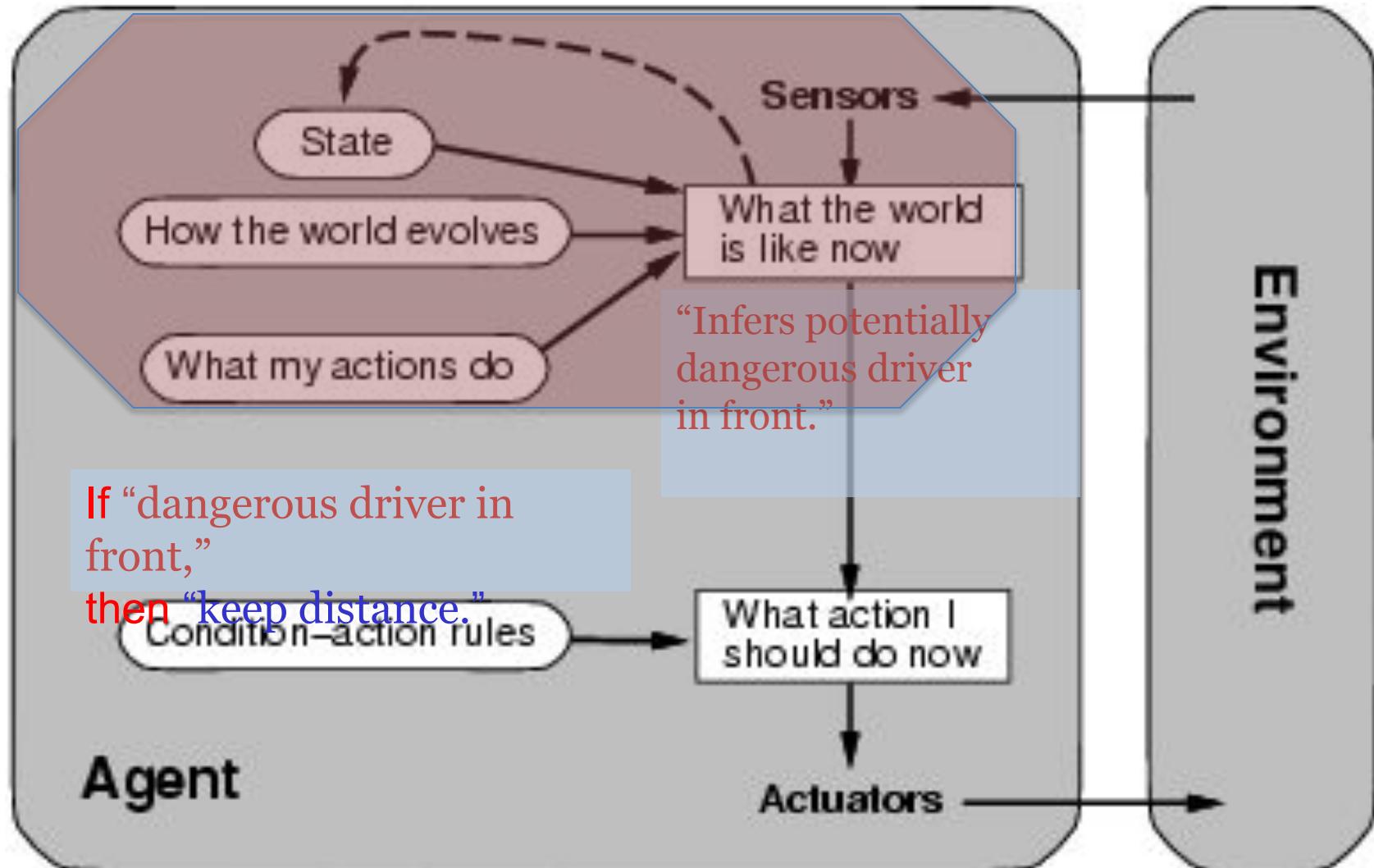


Actions: left, right, straight, open-door

Rules:

1. If open(left) & open(right) and open(straight) then choose randomly between right and left
2. If wall(left) and open(right) and open(straight) then straight
3. If wall(right) and open(left) and open(straight) then straight
4. If wall(right) and open(left) and wall(straight) then left
5. If wall(left) and open(right) and wall(straight) then right
6. If wall(left) and door(right) and wall(straight) then open-door
7. If wall(right) and wall(left) and open(straight) then straight.
8. (Default) Move randomly

[Here]

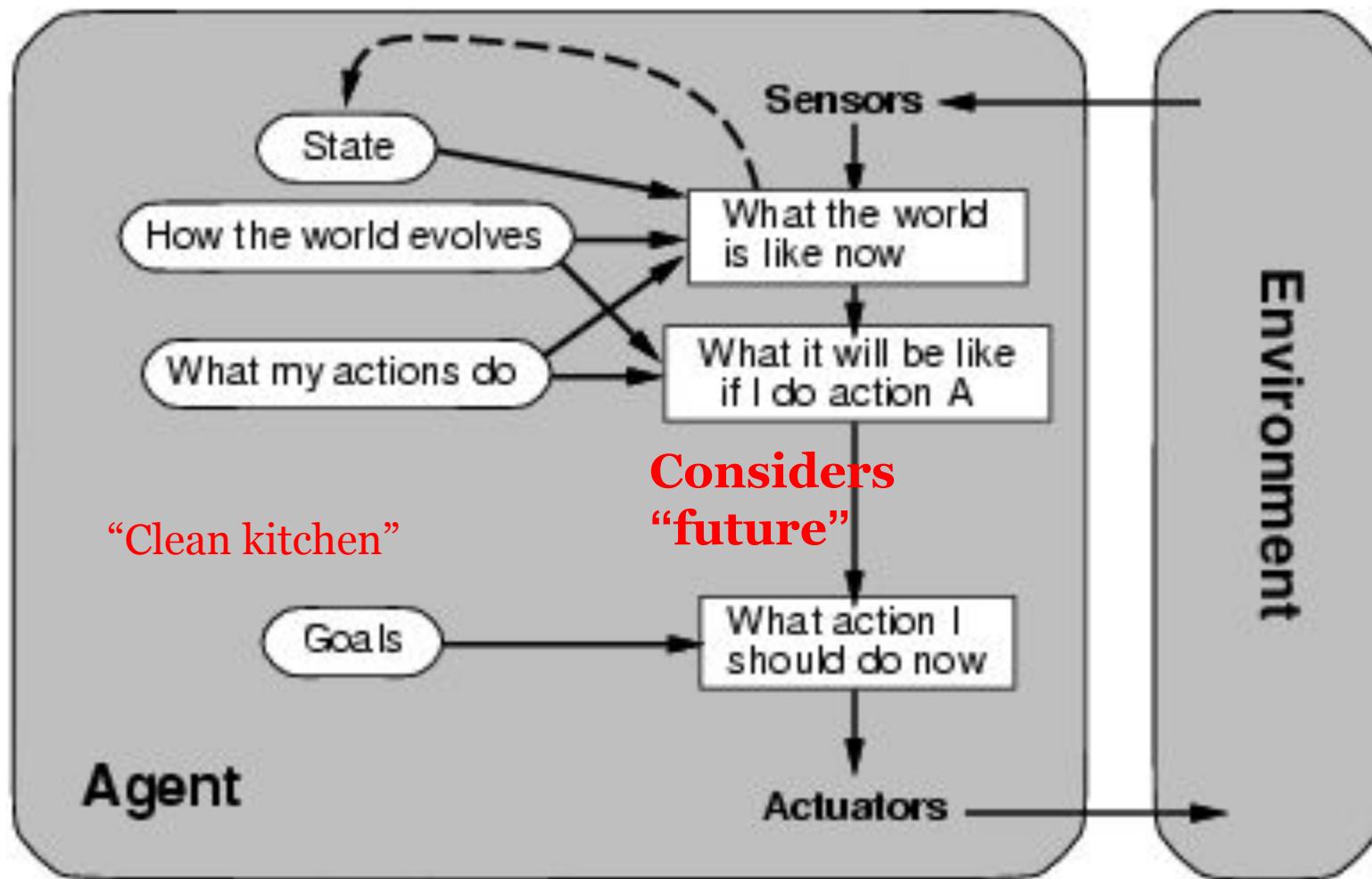
Logical Agents**Representation and Reasoning: Part****III/IV R&N****How detailed?**

Goal-based Agent

- Agent Actions will depend upon its goal.
- Such agents work as follows:
 - information comes from **sensors – percepts**
 - changes the agents current state of the world
 - based on **state of the world** and **knowledge (memory)** and **goals/intentions**, it chooses **actions** and does them through the **effectors**.
- Goal formulation based on the current situation is a way of solving many problems and search is a universal problem solving mechanism in AI.
- The sequence of steps required to solve a problem is not known **a priori** and must be determined by **a systematic exploration** of the alternatives.

Goal-based agents

Module: Problem Solving



Agent keeps track of the world state as well as set of goals it's trying to achieve: chooses actions that will (eventually) lead to the goal(s).

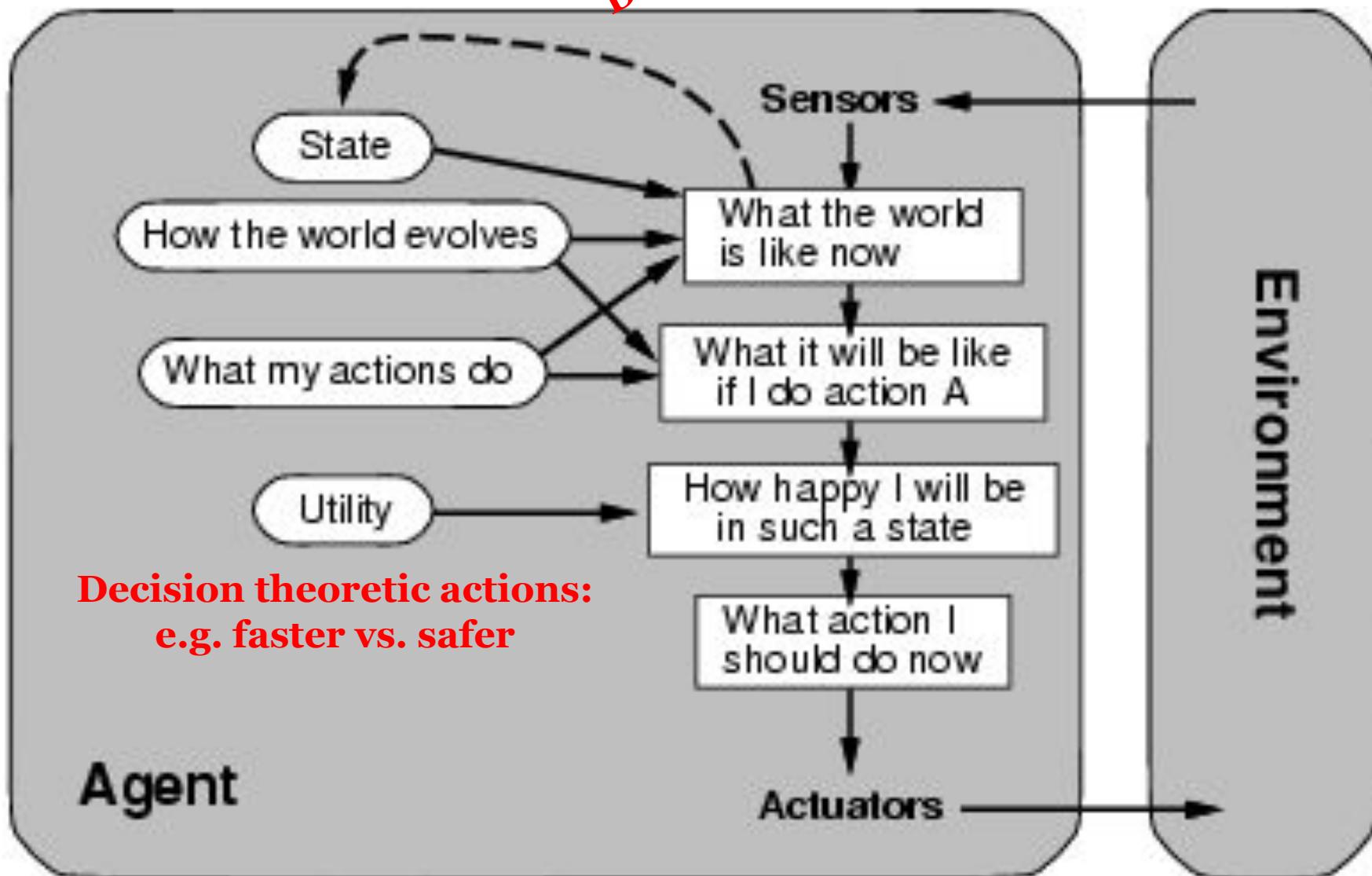
More flexible than reflex agents may involve search and planning

Utility-based Agent

- Utility based agents provides a more general agent framework.
- different preferences for the different goals
- A utility function maps a state or a sequence of states to a real valued utility.
- The agent acts so as to maximize expected utility
- look for a **quicker, safer, cheaper trip** to reach a destination.
Agent happiness should be taken into consideration.
- **Utility describes how “happy” the agent is.**

Module:
Decision Making

Utility-based agents

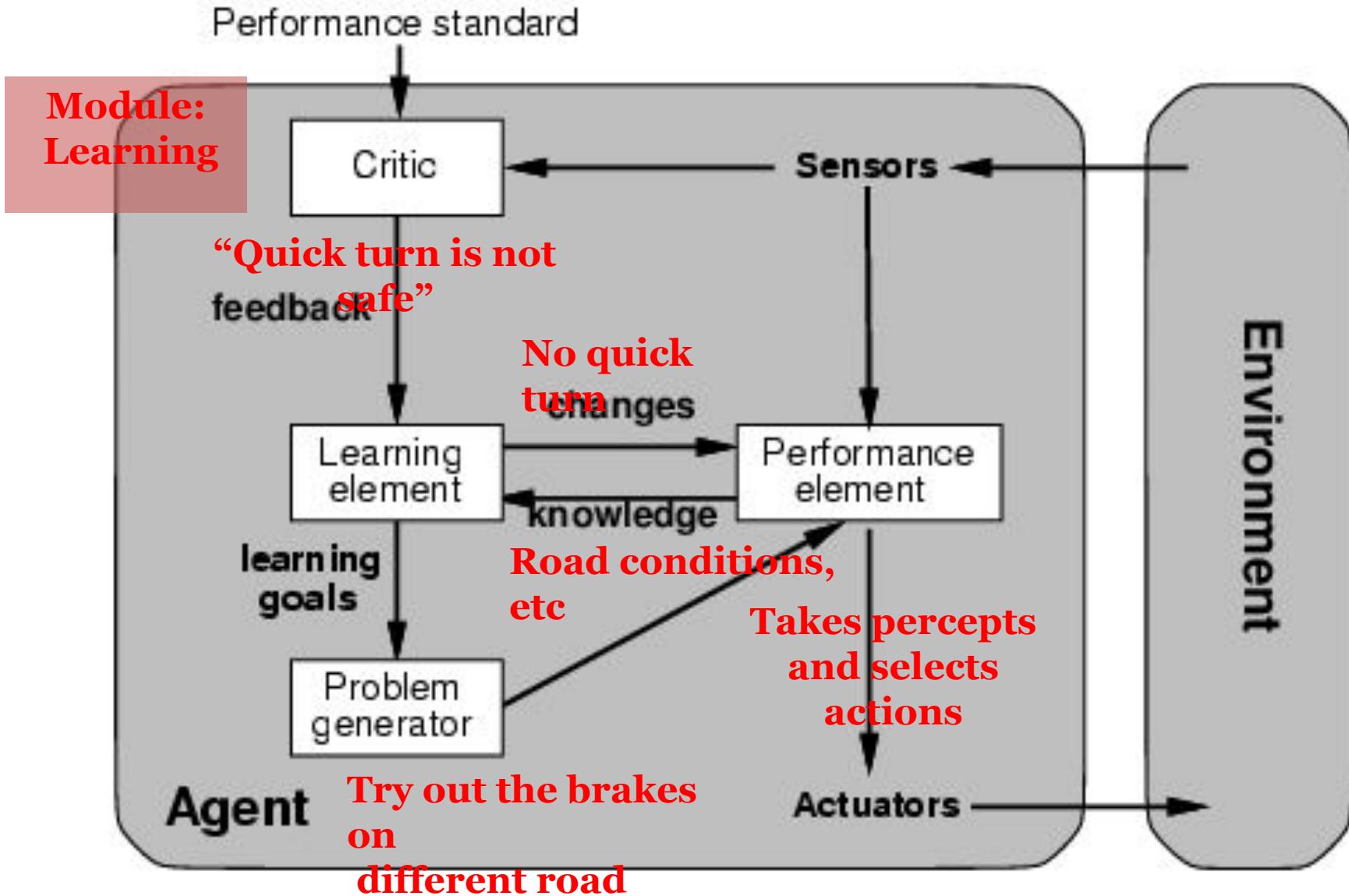


Learning Agent

- Learning allows an agent to operate in initially unknown environments.
- The learning element modifies the performance element.
- Learning is required for true autonomy

More complicated when agent needs to learn utility information:
Reinforcement learning
(based on action payoff)

Learning agents
Adapt and improve over time



Conclusion

- An **agent** perceives and acts in an environment, has an architecture, and is implemented by an agent program.
- An **ideal agent** always chooses the action which maximizes its expected performance, given its percept sequence so far.
- An **autonomous agent** uses its own experience rather than built-in knowledge of the environment by the designer.
- An agent program maps from percept to action and updates its internal state.
 - Reflex agents respond immediately to percepts.
 - Goal-based agents act in order to achieve their goal(s).
 - Utility-based agents maximize their own utility function.
- Representing knowledge is important for successful agent design.
- The most challenging environments are partially observable, stochastic, sequential, dynamic, and continuous, and contain multiple intelligent agents.

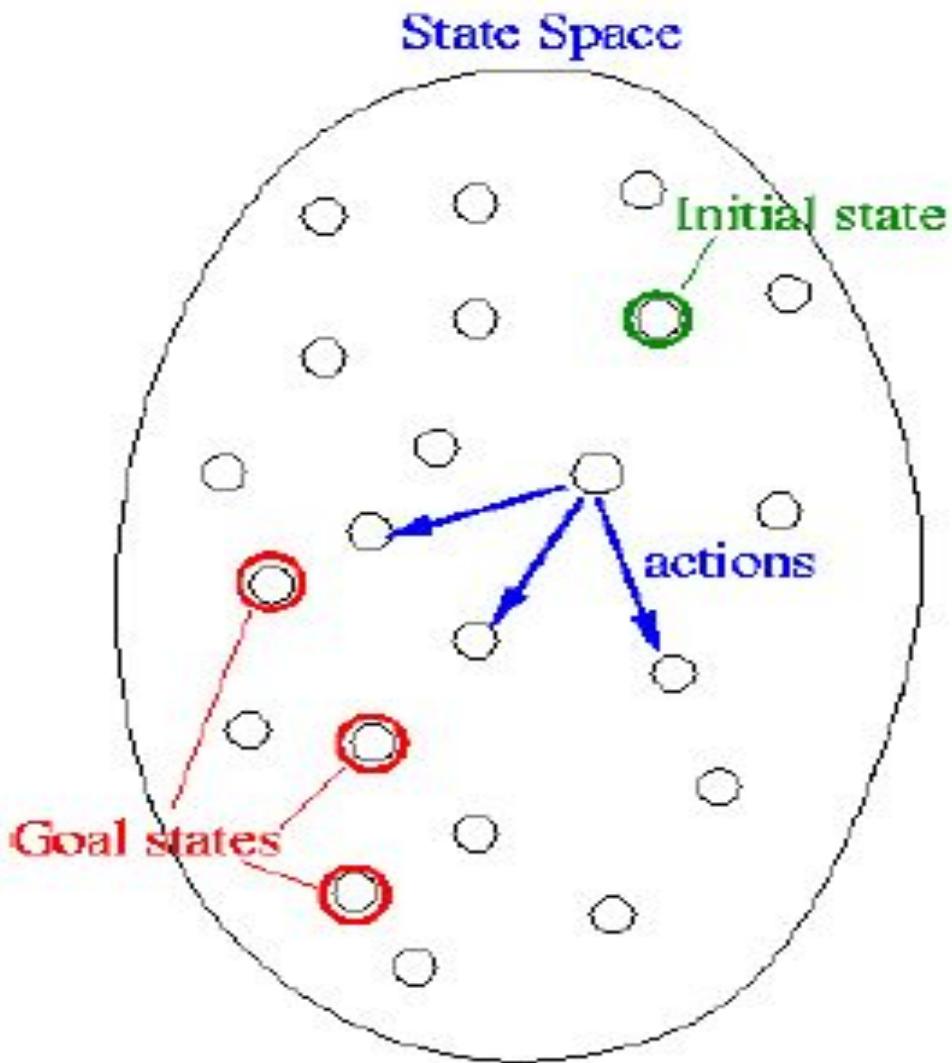
Artificial Intelligence

Lecture 4: Problem Solving using Search (Single agent search)

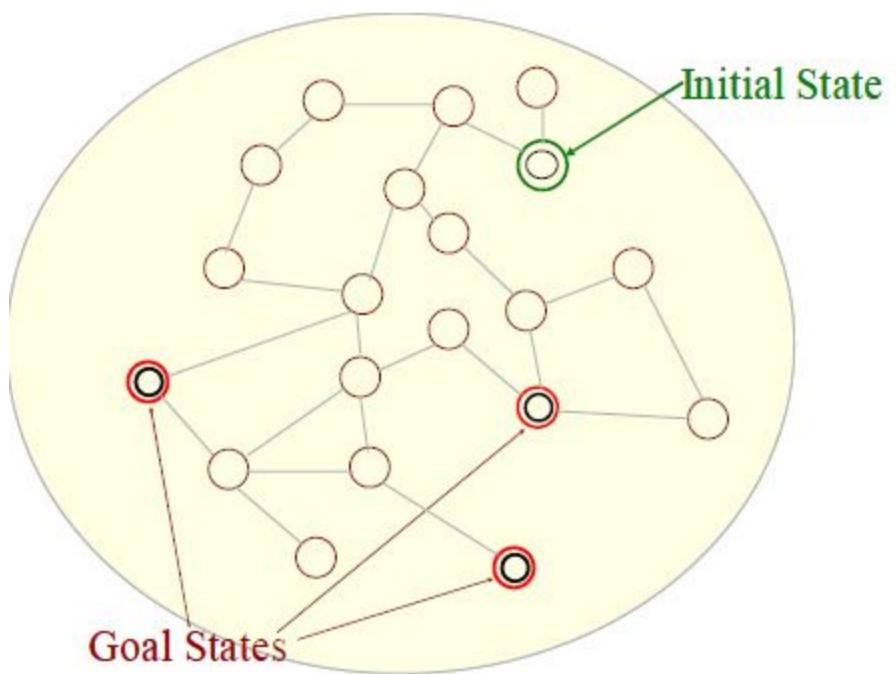
Search Problem

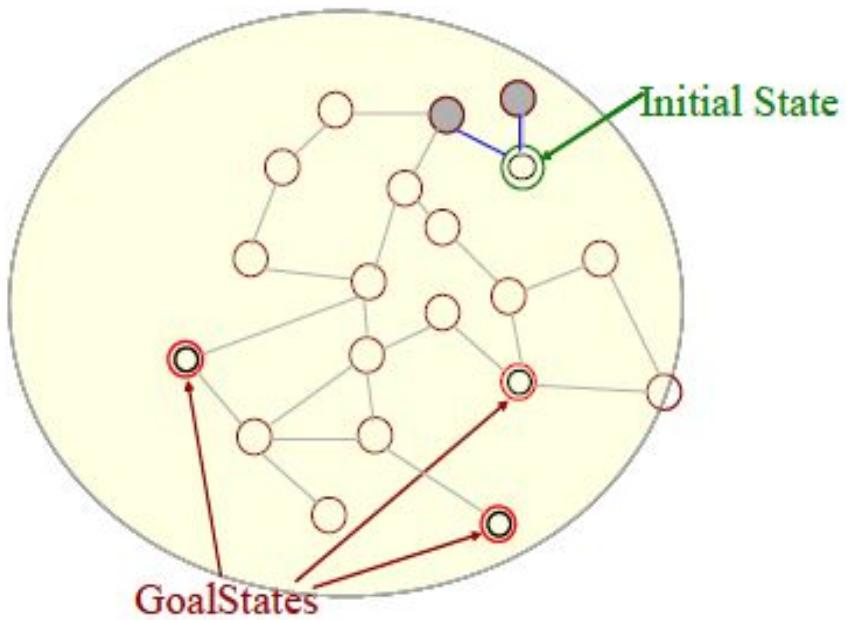
A search problem consists of the following:

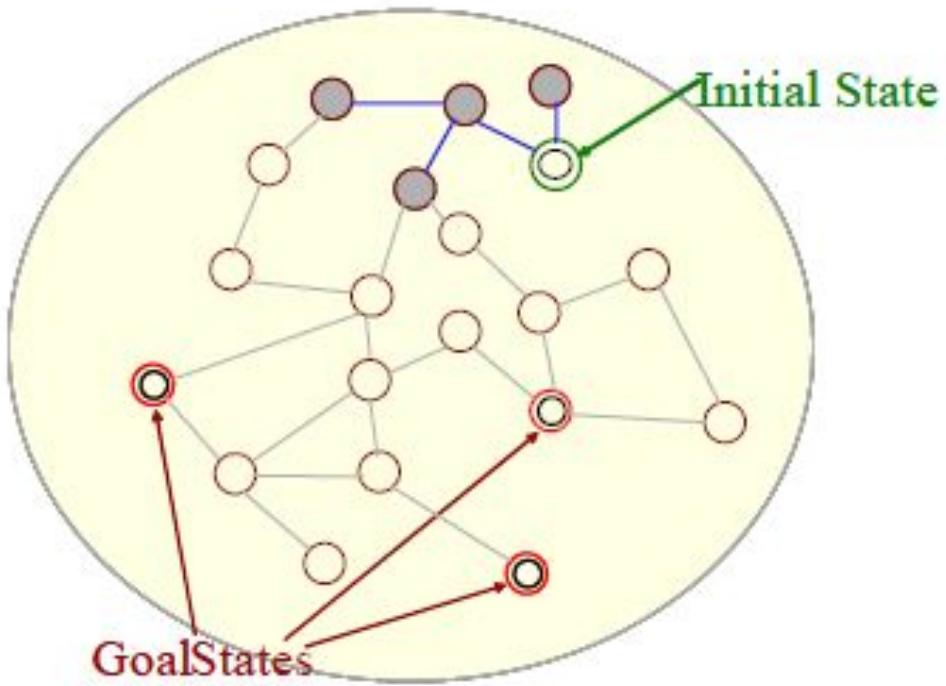
- S : the full set of states
 - s_o : the initial state
 - $A: S \rightarrow S$ is a set of operators
 - G is the set of final states.

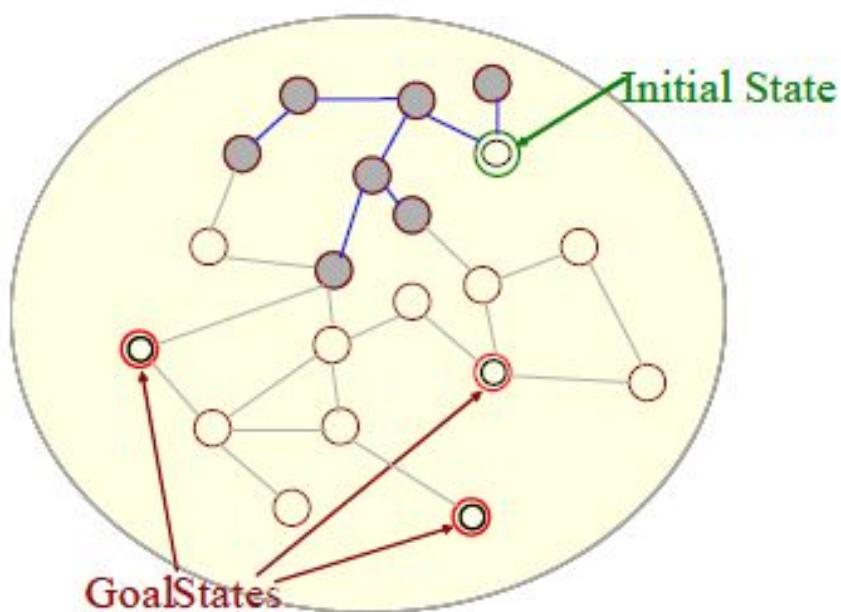


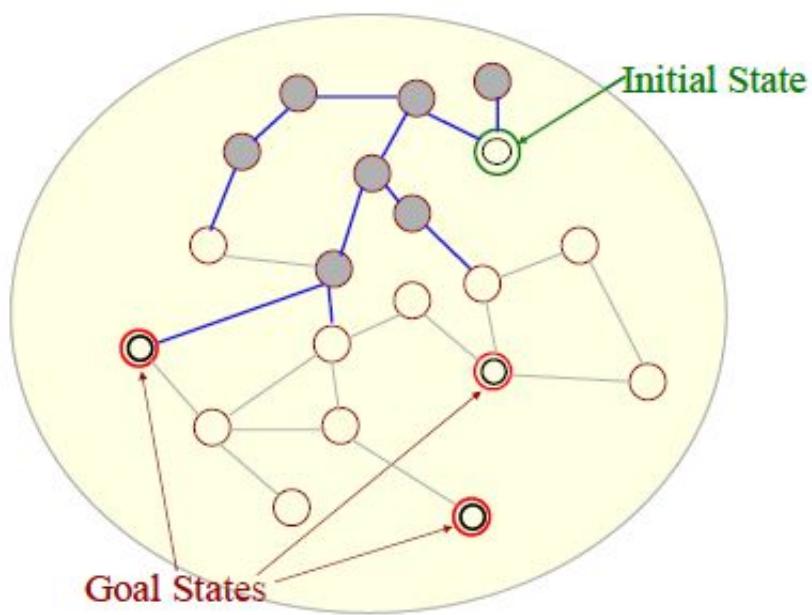
- The **search problem** is to find a sequence of actions which transforms the agent from the initial state to a goal state $g \in G$. A search problem is represented by a 4-tuple $\{S, s_o, A, G\}$.
 - S : set of states
 - $s_o \in S$: initial state
 - A : $S \square S$ operators/ actions that transform one state to another state
 - G : goal, a set of states.

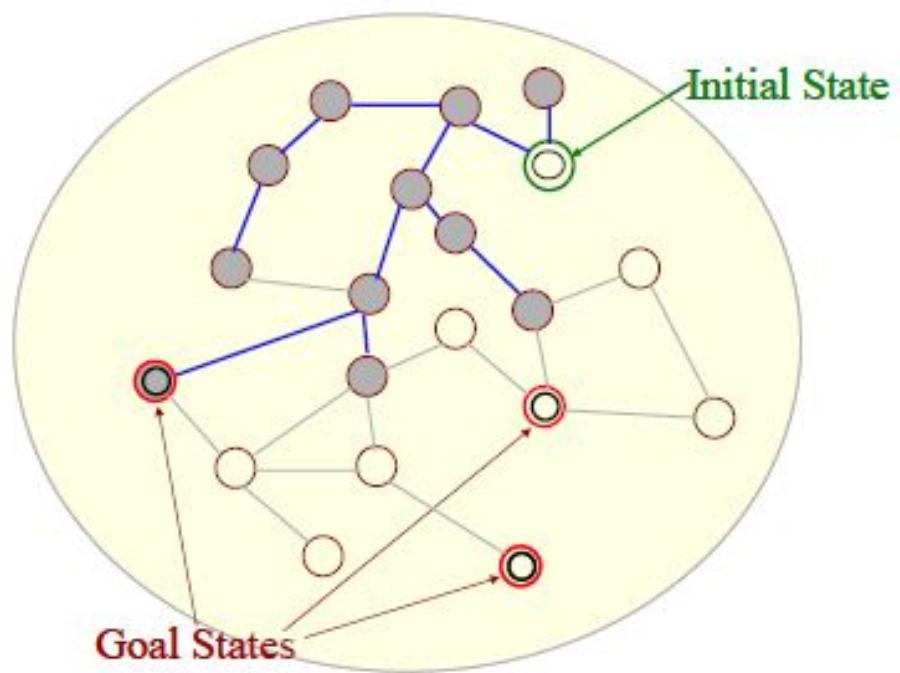






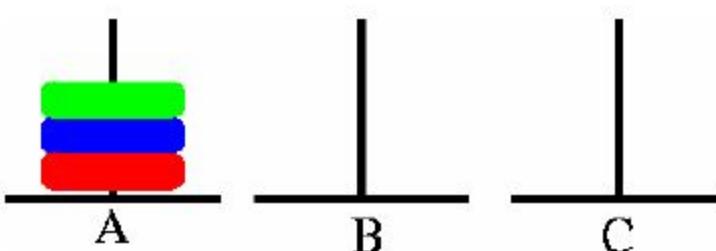




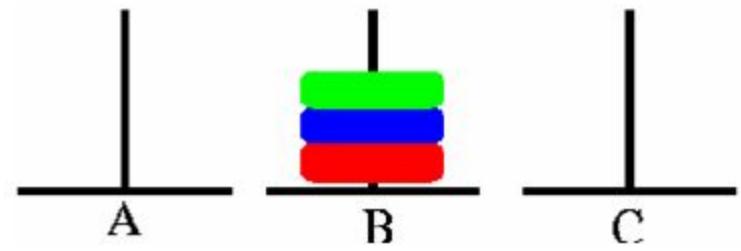


Example problem: Pegs and Disks problem

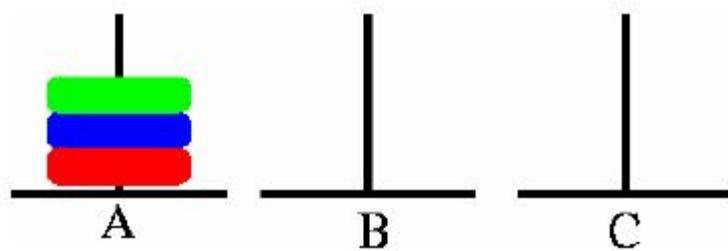
- Consider the following problem. We have 3 pegs and 3 disks.
- **Operators:** one may move the topmost disk on any needle to the topmost position to any other needle
- In the goal state all the pegs are in the needle B as shown in the figure below.



Initial State

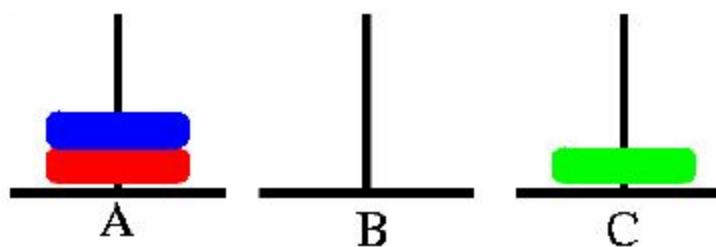


Goal State

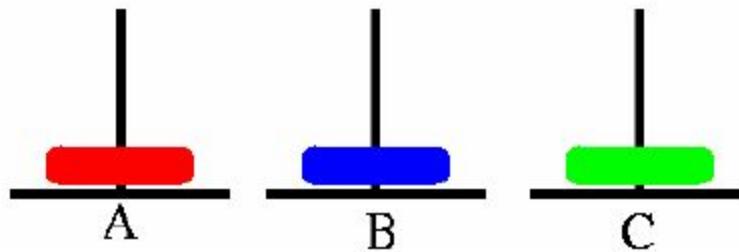


Initial State

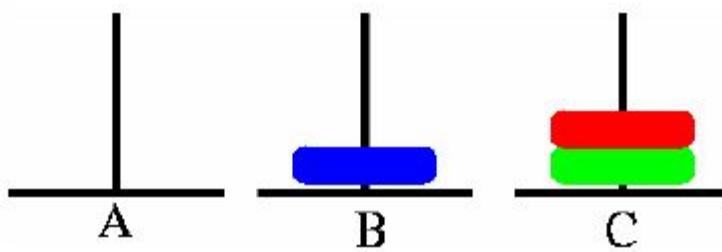
Step 1: Move A → C



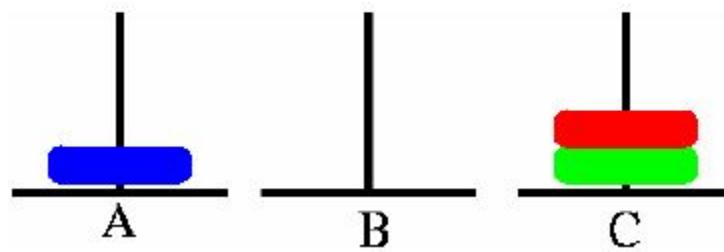
Step 2: Move A → B



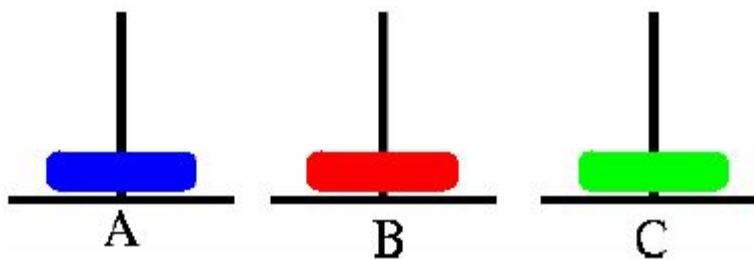
Step 3: Move A → C



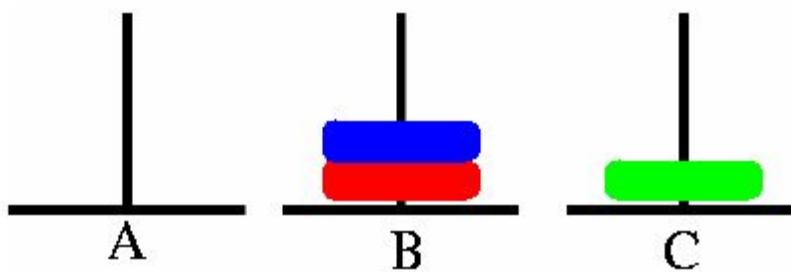
Step 4: Move B → A



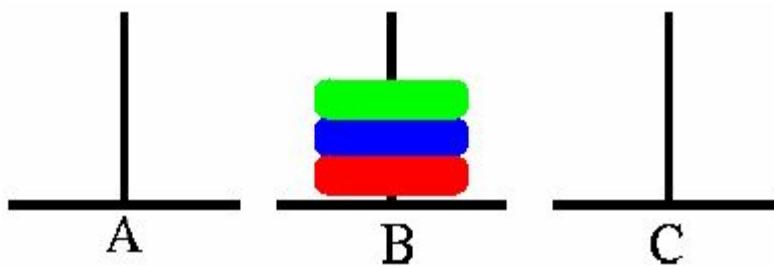
Step 5: Move C → B



Step 6: Move A → B



Step 7: Move C → B



Goal State

Example: the 8-puzzle

- How would you use AI techniques to solve the 8-puzzle problem?

5	4	
6	1	8
7	3	2

Initial State

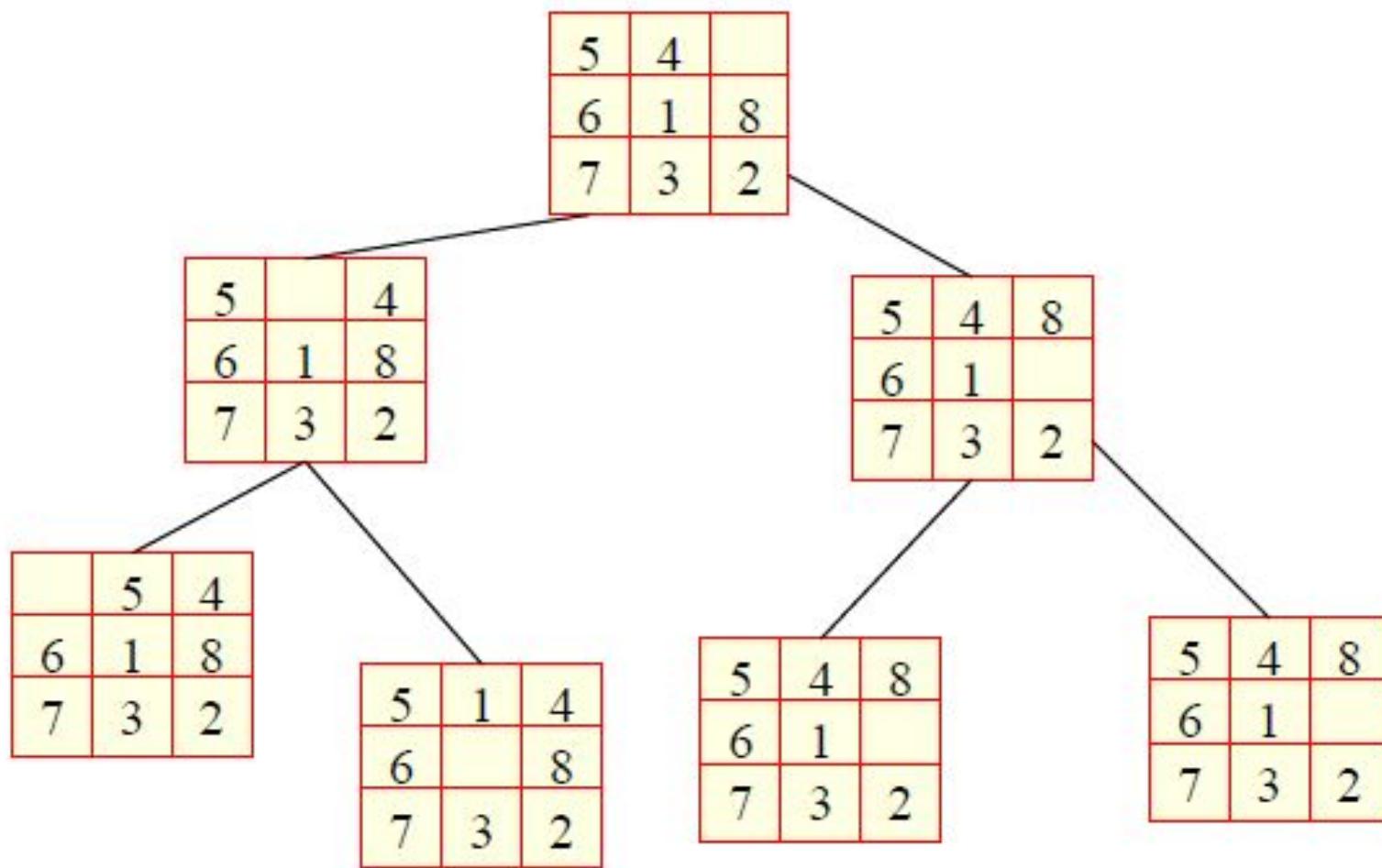
1	4	7
2	5	8
3	6	

Goal State

Problem Definition - Example, 8 puzzle

- The state space representation for this problem is summarized below:
 - **States:** A state is a description of each of the eight tiles in each location that it can occupy.
 - **Operators/Action:** The blank moves left, right, up or down
 - **Goal Test:** The current state matches a certain state (e.g. one of the ones shown on previous slide)
 - **Path Cost:** Each move of the blank costs 1

8-puzzle partial state space

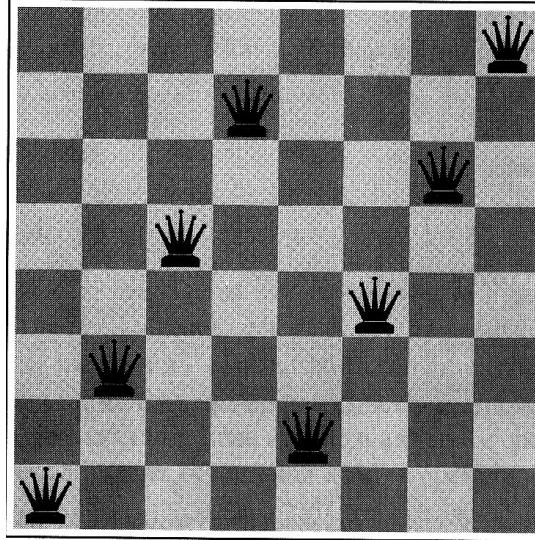


Artificial Intelligence

Lecture 5:

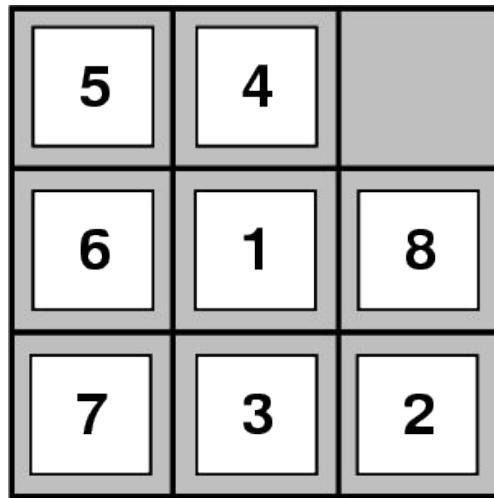
Problem Solving using Search - (Single agent search)
Uninformed Search

Example: 8-queens

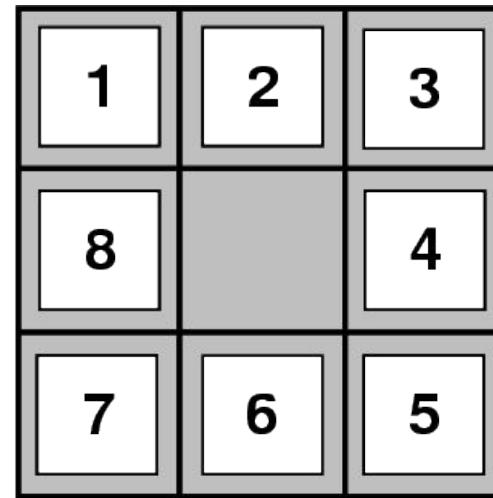


- State: any arrangement of up to 8 queens on the board
- Operation: add a queen (incremental), move a queen (fix-it)
- Initial state: no queens on board
- Goal state: 8 queens, with no queen is attacked
- Solution Path: The set of operations that allowed you to get to the board that you see above at the indicated positions.

Example: 8-puzzle



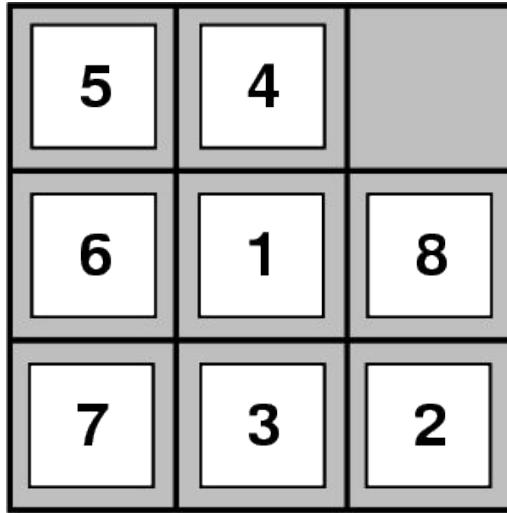
Start State



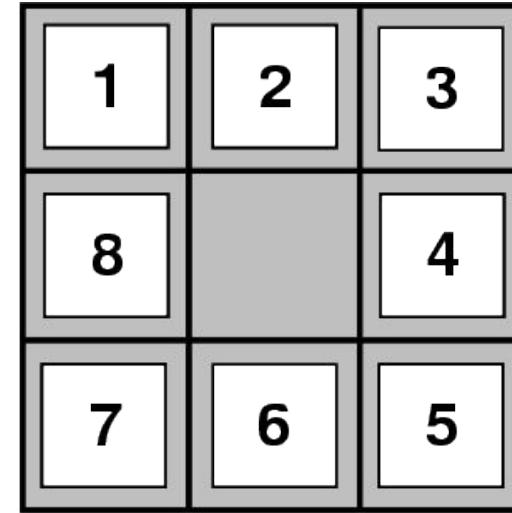
Goal State

- State:
- Operators:
- Goal test:
- Solution path:

Example: 8-puzzle



Start State



Goal State

- Operators: moving blank left, right, up, down (ignore jamming)
- Goal test: goal state
- State: integer location of tiles (ignore intermediate locations)
- Solution: move 4 tile to blank, move 1 tile blank, etc.

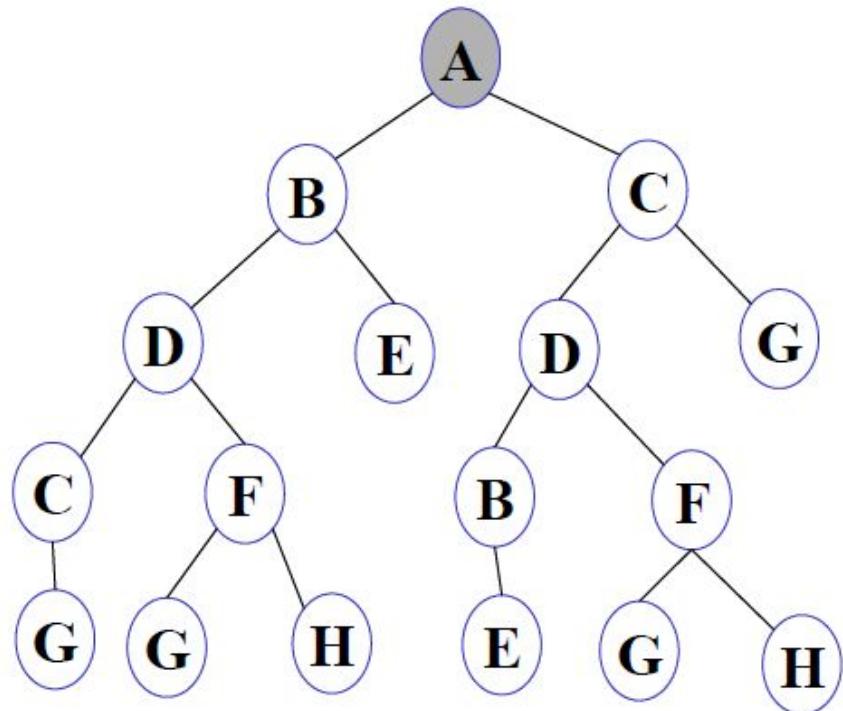
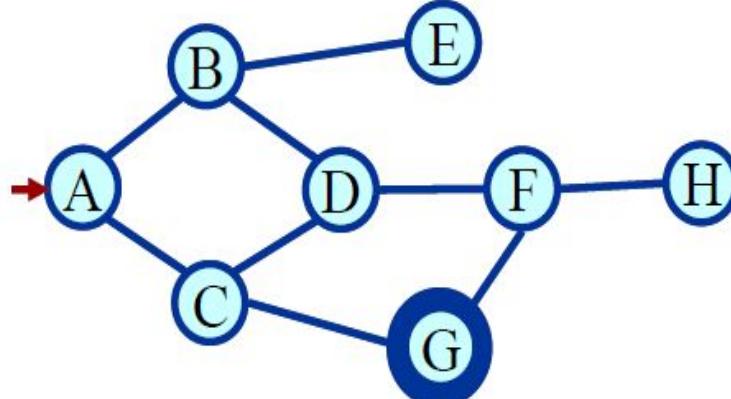
Search

- What is Search?

- Formulate appropriate problems in optimization and planning (sequence of actions to achieve a goal) as search tasks: initial state, operators, goal test, path cost

Search Tree

List all Possible Path
Eliminate Cycles from paths
Result: Search Tree



The basic search algorithm

Let L be a list containing the initial state (L= the fringe)

Loop

 if L is empty return failure

 Node \leftarrow select (L)

 if Node is a goal

 then return Node

 (the path from initial state to Node)

 else generate all successors of Node, and

 merge the newly generated states into L

End Loop

Evaluating Search Strategies

- **Completeness**
 - Guarantees finding a solution whenever one exists
- **Time complexity**
 - How long (worst or average case) does it take to find a solution? Usually measured in terms of the number of nodes expanded
- **Space complexity**
 - How much space is used by the algorithm? Usually measured in terms of the maximum size of the “nodes” list during the search
- **Optimality/Admissibility**
 - If a solution is found, is it guaranteed to be an optimal one? That is, is it the one with minimum cost?

Problem solving

- We want:
 - To automatically solve a problem
- We need:
 - A representation of the problem
 - Algorithms that use some strategy to solve the problem defined in that representation

Problem representation

- General:
 - **State space:** a problem is divided into a set of resolution steps from the initial state to the goal state
 - **Reduction to sub-problems:** a problem is arranged into a hierarchy of sub-problems
- Specific:
 - Game resolution
 - Constraints satisfaction

States

- A problem is defined by its elements and their relations.
- In each instant of the resolution of a problem, those elements have specific descriptors (How to select them?) and relations.
- A **state** is a representation of those elements in a given moment.
- Two special states are defined:
 - **Initial state** (starting point)
 - **Final state** (goal state)

State modification: successor function

- A successor function is needed to move between different states.
- A **successor function** is a description of possible actions, a set of operators. It is a transformation function on a state representation, which convert it into another state.
- The successor function defines a relation of accessibility among states.
- Representation of the successor function:
 - Conditions of applicability
 - Transformation function

State space

- The **state space** is the set of all states reachable from the initial state.
- It forms a graph (or map) in which the nodes are states and the arcs between nodes are actions.
- A **path** in the state space is a sequence of states connected by a sequence of actions.
- The solution of the problem is part of the map formed by the state space.

Problem solution

- A **solution** in the state space is a path from the initial state to a goal state or, sometimes, just a goal state.
- **Path/solution cost:** function that assigns a numeric cost to each path, the cost of applying the operators to the states
- Solution quality is measured by the path cost function, and an **optimal solution** has the lowest path cost among all solutions.
- Solutions: any, an **optimal one**, all. Cost is important depending on the problem and the type of solution sought.

Problem description

- Components:
 - State space (explicitly or implicitly defined)
 - Initial state
 - Goal state (or the conditions it has to fulfill)
 - Available actions (operators to change state)
 - Restrictions (e.g., cost)
 - Elements of the domain which are relevant to the problem (e.g., incomplete knowledge of the starting point)
 - Type of solution:
 - Sequence of operators or goal state
 - Any, an optimal one (cost definition needed), all

Uninformed vs. informed search

- **Uninformed search strategies**
 - Also known as “blind search,” uninformed search strategies use **no information about the likely “direction” of the goal node(s)**
 - Uninformed search methods: Breadth-first, depth-first, depth-limited, uniform-cost, depth-first iterative deepening, bidirectional
- **Informed search strategies**
 - Also known as “heuristic search,” informed search strategies use information about the domain to (try to) (usually) head in the general direction of the goal node(s)
 - Informed search methods: Hill climbing, best-first, greedy search, beam search, A, A*

Uninformed

1. Search without information
2. No Knowledge
3. Time Consuming
4. More complexity - time and space
5. BFS, DFS

Informed

1. Search with information
2. use knowledge to find steps to solution
3. quick solution
4. less complexity
5. A*, Best First, AO*

Breadth-First Search (BFS)

- Uninformed Search technique
- FIFO (Queue)
- Expand shallowest unexpanded node
- Level search
- Fringe: nodes waiting in a queue to be explored, also called **OPEN**
- Optimal - shortest path
- Implementation:
 - For BFS, *fringe* is a first-in-first-out (FIFO) queue
 - new successors go at end of the queue
- Repeated states?
 - Simple strategy: do not add parent of a node as a leaf

BFS Algorithm

Breadth first search

Let *fringe* be a list containing the initial state

Loop

 if *fringe* is empty return failure

 Node \leftarrow remove-first (*fringe*)

 if Node is a goal

 then return the path from initial state to Node

 else generate all successors of Node, and

 (merge the newly generated nodes into *fringe*)

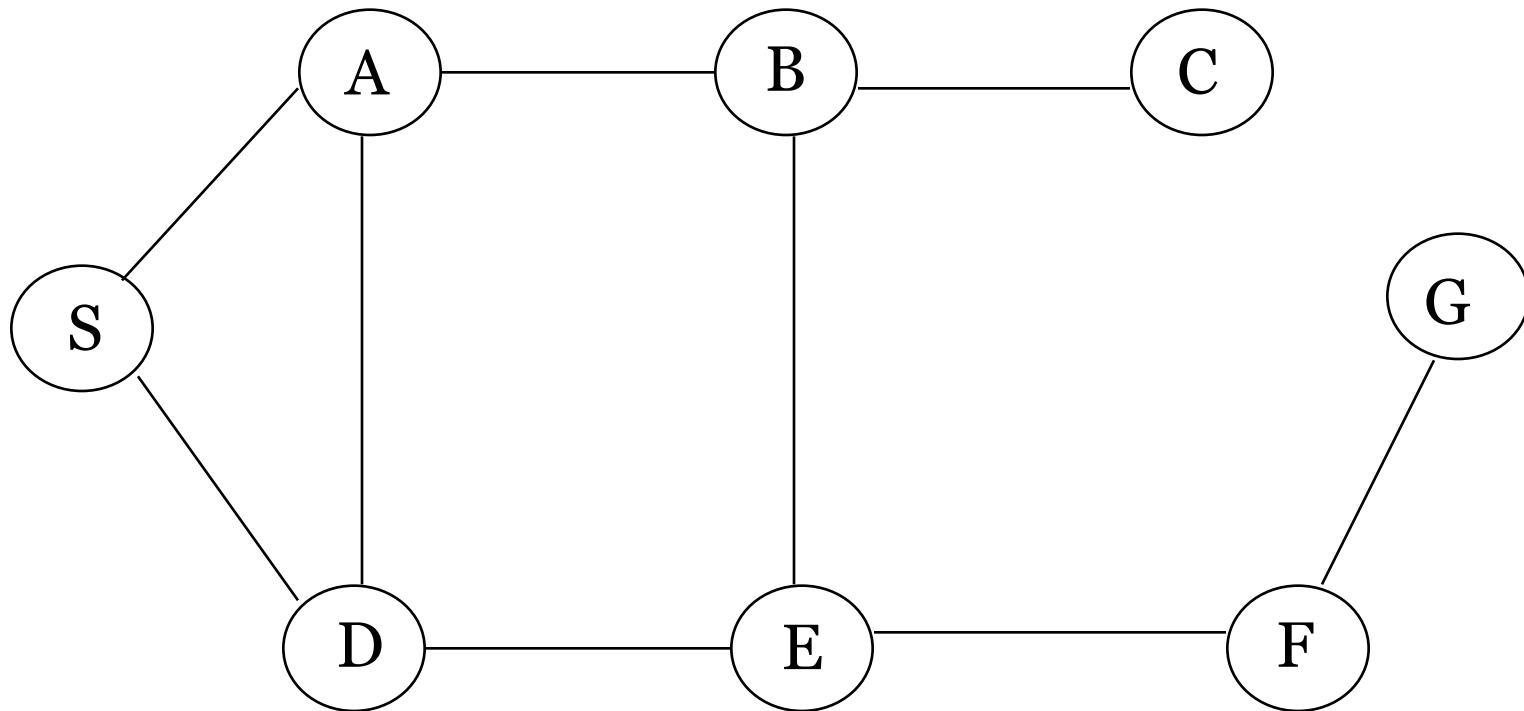
 add generated nodes to the back of *fringe*

End Loop

Example: Map Navigation

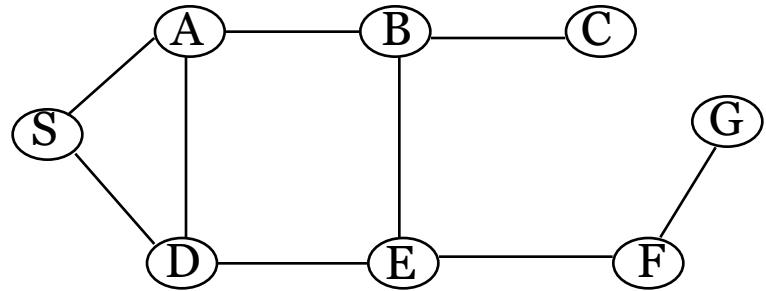
State Space:

S = start, G = goal, other nodes = intermediate states, links = legal transitions



BFS Search Tree

S



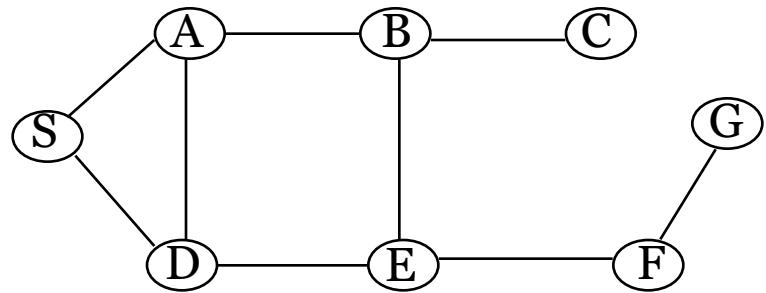
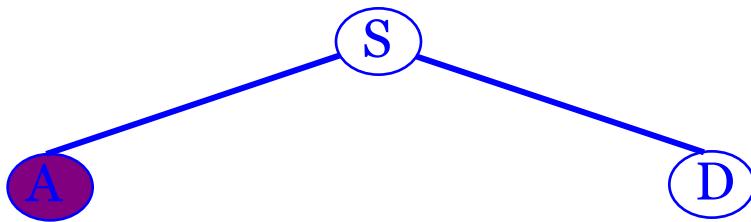
Queue = {S}

Select S

Goal(S) = true?

If not, Expand(S)

BFS Search Tree



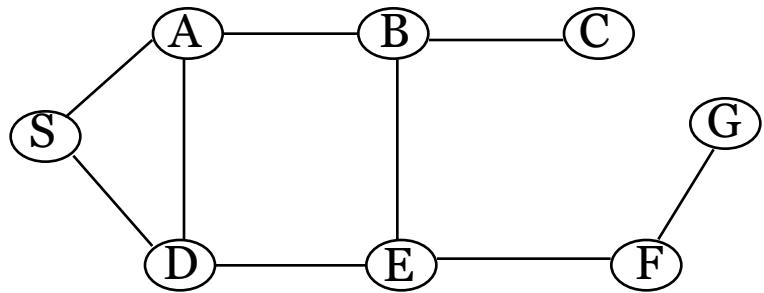
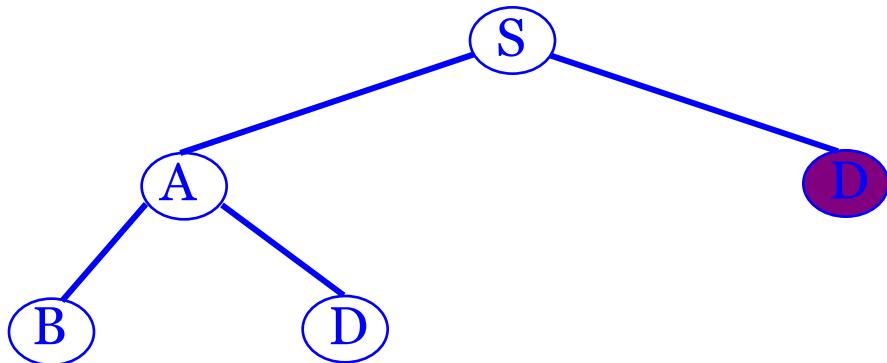
Queue = {A, D}

Select A

Goal(A) = true?

If not, Expand(A)

BFS Search Tree



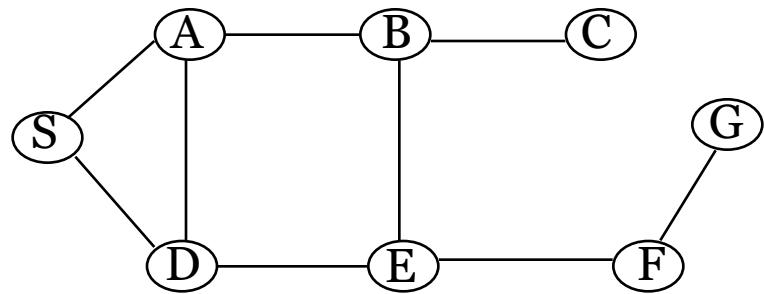
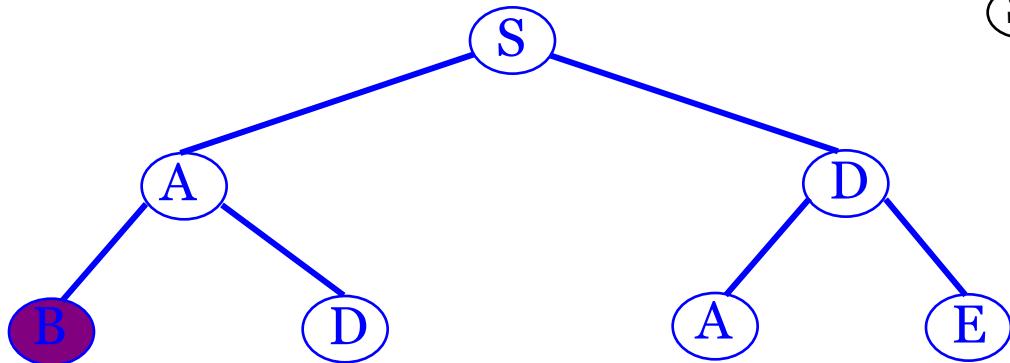
Queue = {D, B, D}

Select D

Goal(D) = true?

If not, expand(D)

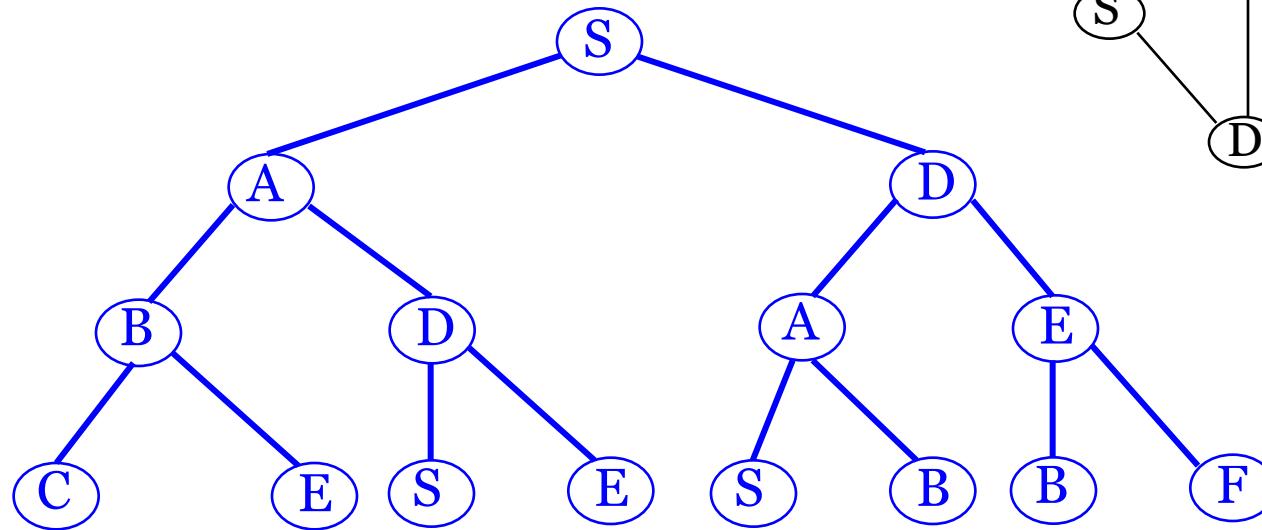
BFS Search Tree



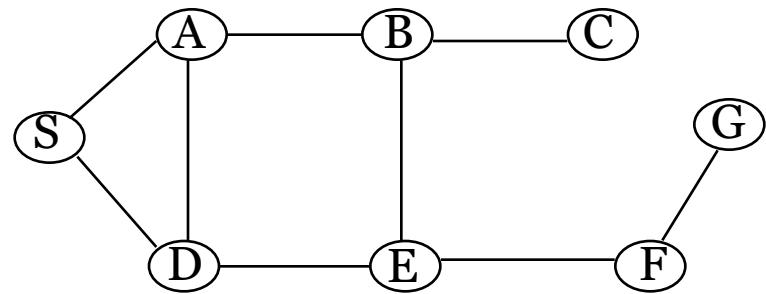
Queue = {B, D, A, E}

Select B
etc.

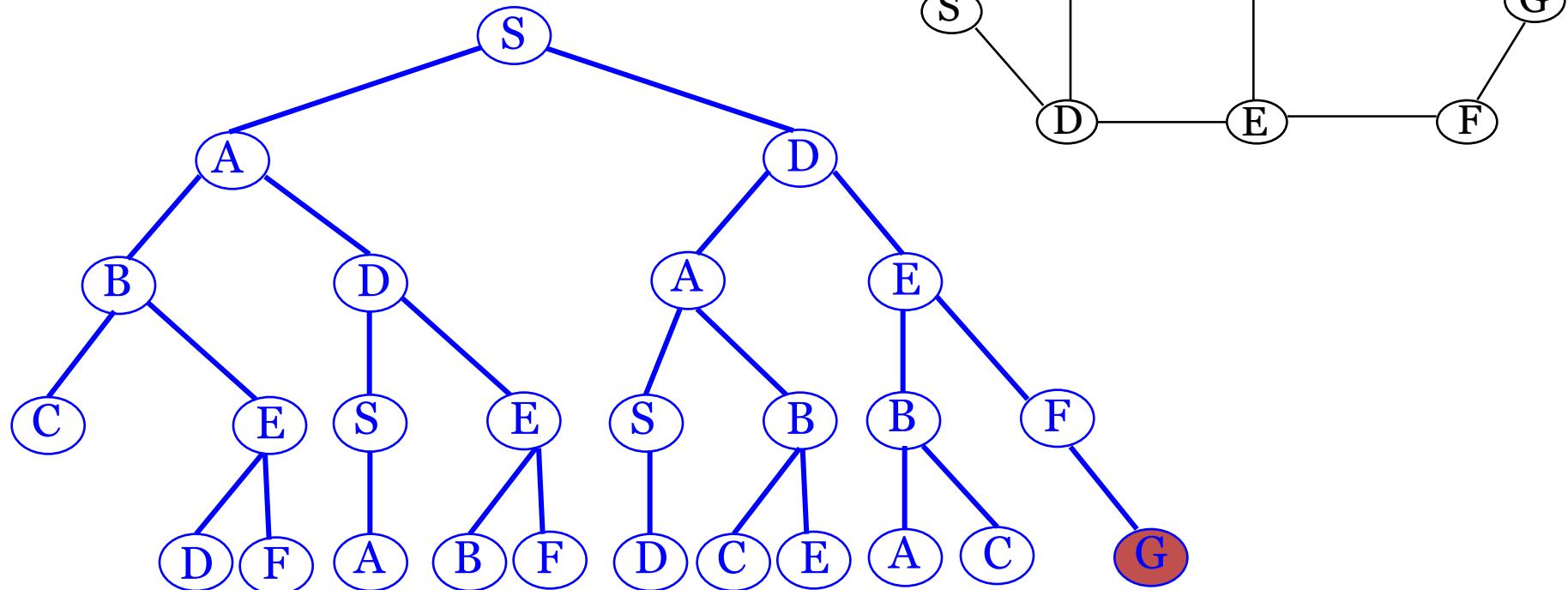
BFS Search Tree



Level 3
Queue = {C, E, S, E, S, B, B, F}

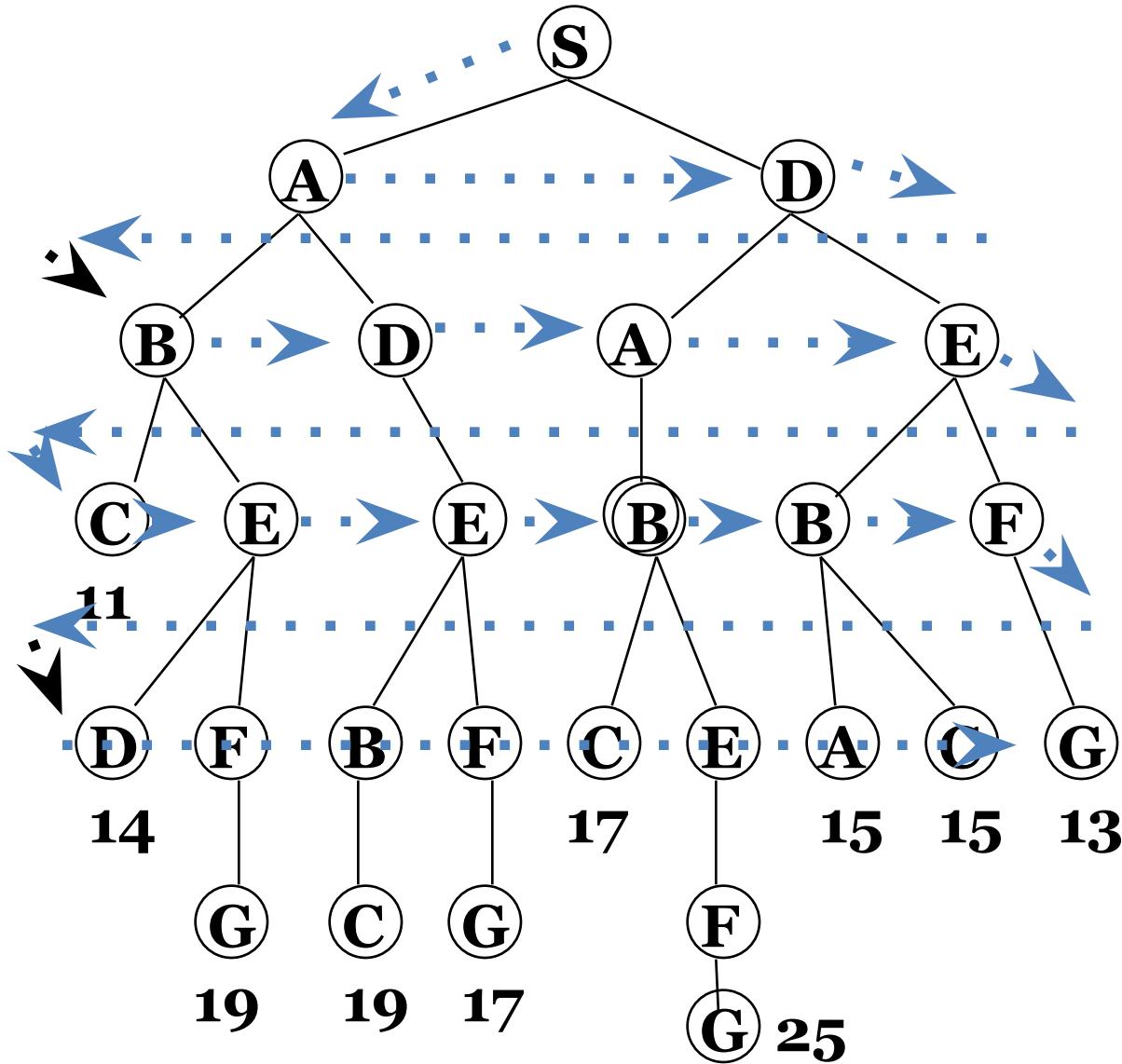


BFS Search Tree



Level 4
Expand queue until G is at front
Select G
Goal(G) = true

Another Breadth-first search



Depth-First Search (DFS)

- Uninformed Search
- Stack (LIFO)
- Expand deepest unexpanded node
- LIFO
- Deepest Node
- Incomplete ---> 1. Loop 2. search space infinite i.e. unlimited depth
- Implementation:
 - For DFS, *fringe* is a LIFO queue
 - new successors go at beginning of the queue
- Repeated nodes?
 - Simple strategy: Do not add a state as a leaf if that state is on the path from the root to the current node

DFS Algorithm

Depth First Search

Let $fringe$ be a list containing the initial state

Loop

 if $fringe$ is empty return failure

 Node \leftarrow remove-first ($fringe$)

 if Node is a goal

 then return the path from initial state to Node

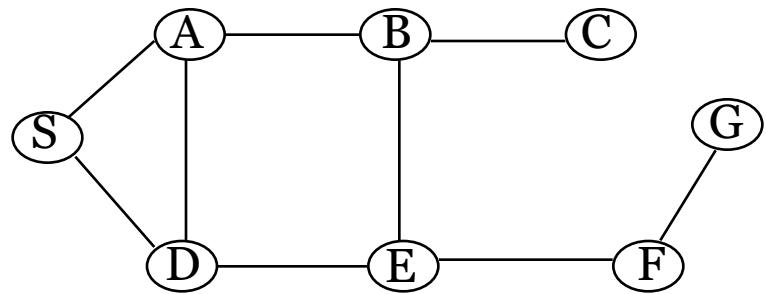
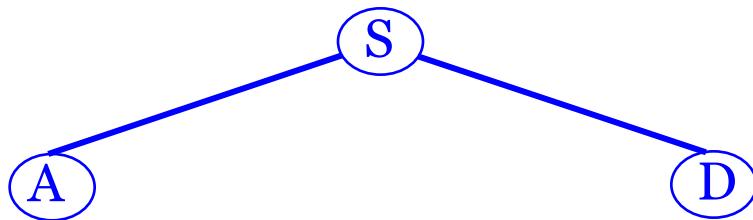
 else generate all successors of Node, and

 merge the newly generated nodes into $fringe$

 add generated nodes to the front of $fringe$

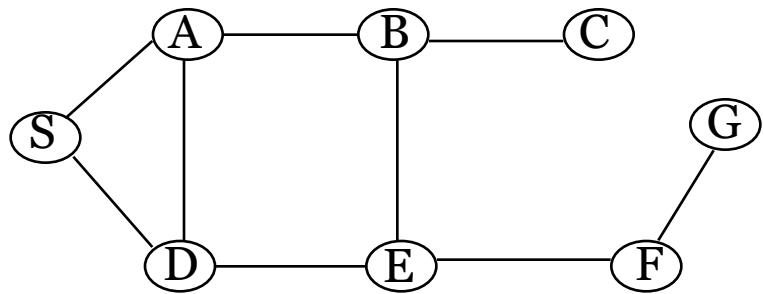
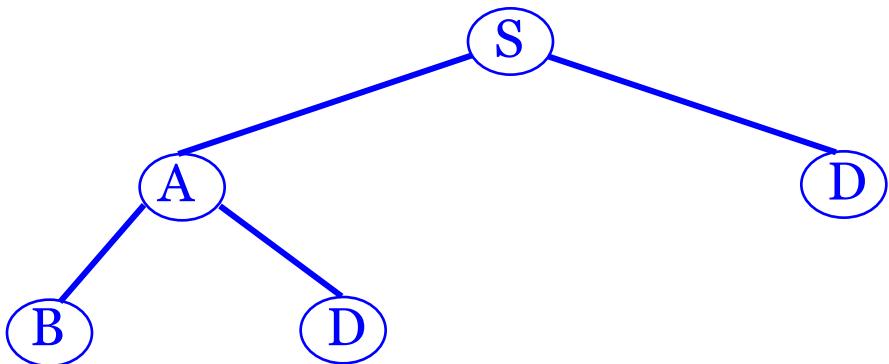
End Loop

DFS Search Tree



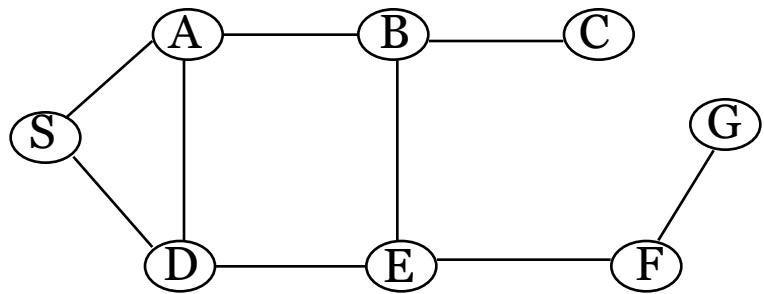
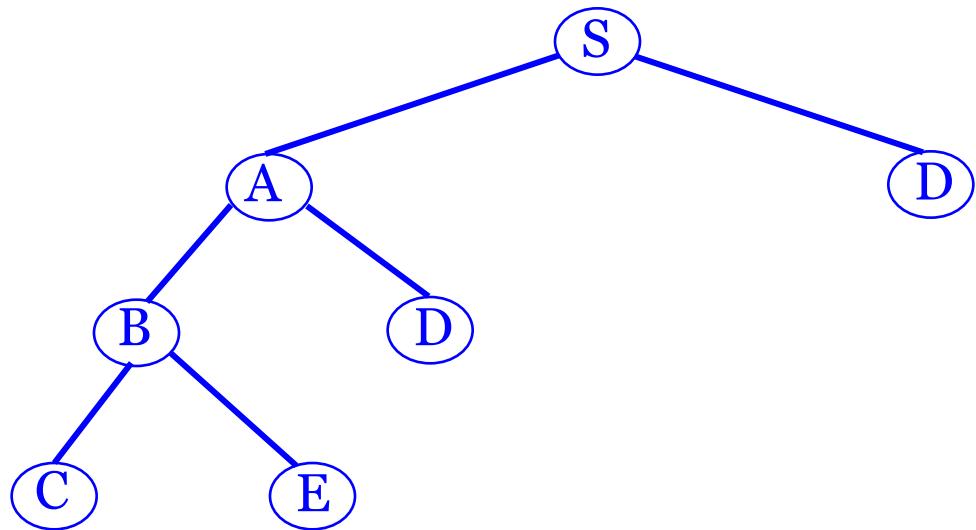
Queue = {A,D}

DFS Search Tree



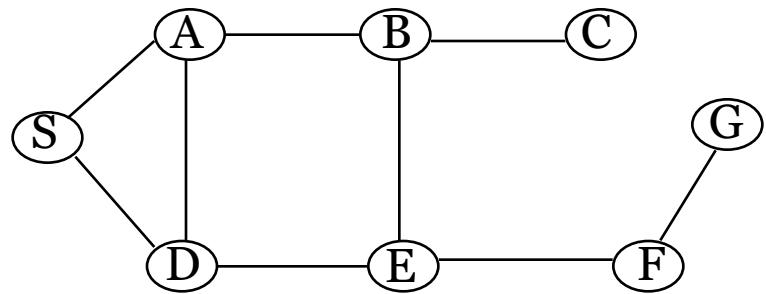
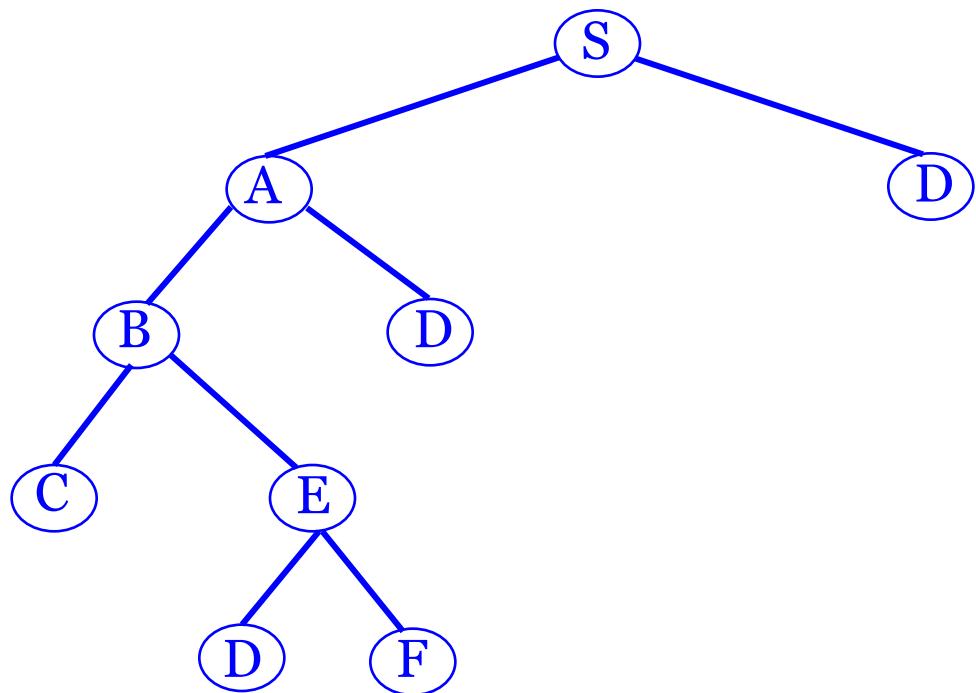
Queue = {B,D,D}

DFS Search Tree



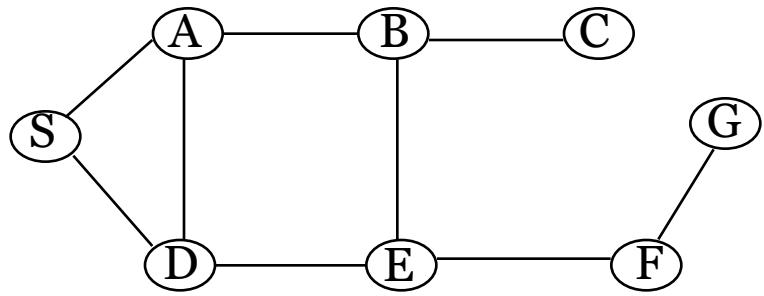
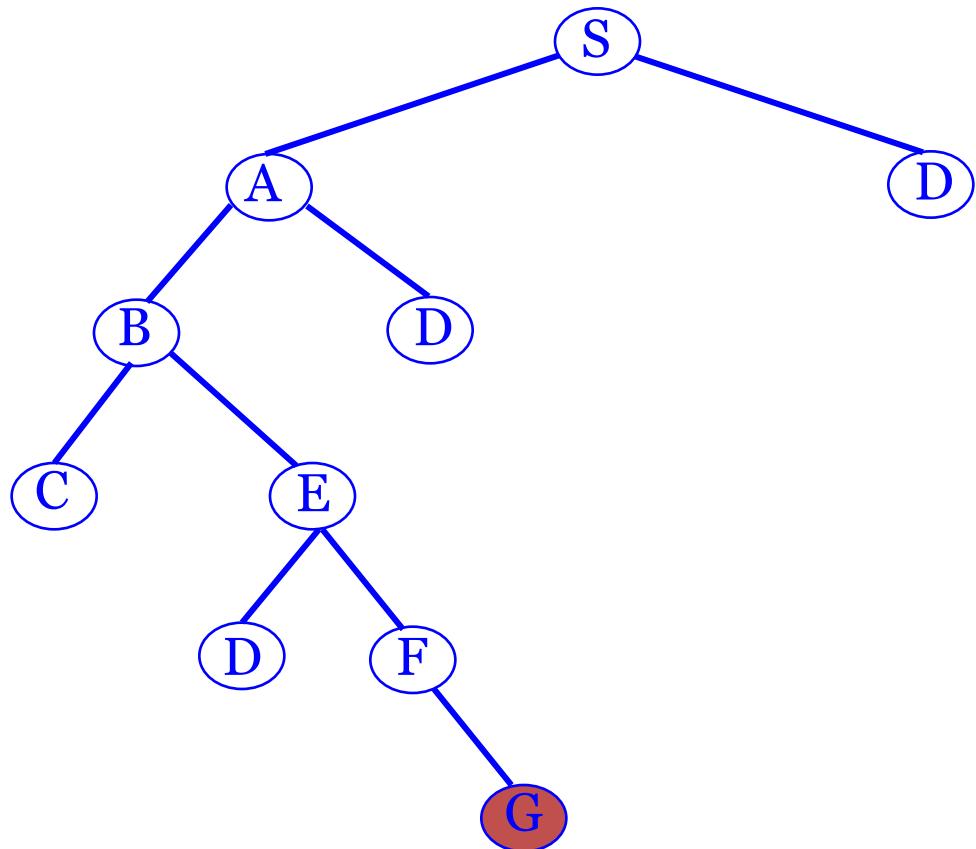
Queue = {C,E,D,D}

DFS Search Tree



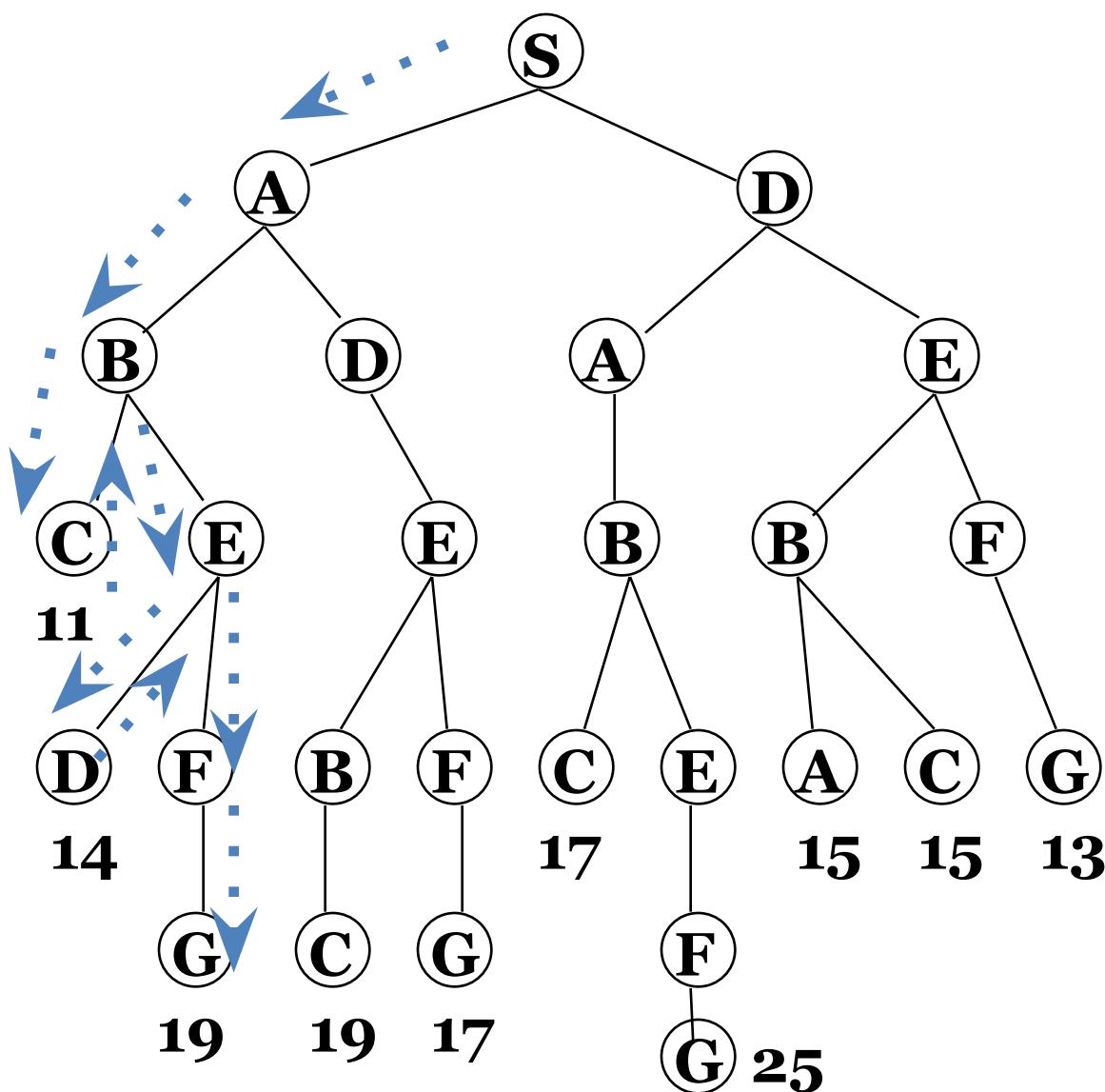
Queue = {D,F,D,D}

DFS Search Tree



Queue = {G,D,D}

Depth-first search



Evaluation of Search Algorithms

- **Completeness**
 - does it always find a solution if one exists?
- **Optimality**
 - does it always find a least-cost (or min depth) solution?
- **Time complexity**
 - number of nodes generated (worst case)
- **Space complexity**
 - number of nodes in memory (worst case)
- **Time and space complexity are measured in terms of**
 - b : maximum branching factor of the search tree
 - d : depth of the least-cost solution
 - m : maximum depth of the state space (may be ∞)

Breadth-First Search (BFS) Properties

- Complete? Yes
- Optimal? Yes, for the shortest path
- Time complexity? $O(b^d)$

$$1 + b + b^2 + \dots + b^d = O(b^d)$$

exponential in the depth of the solution d

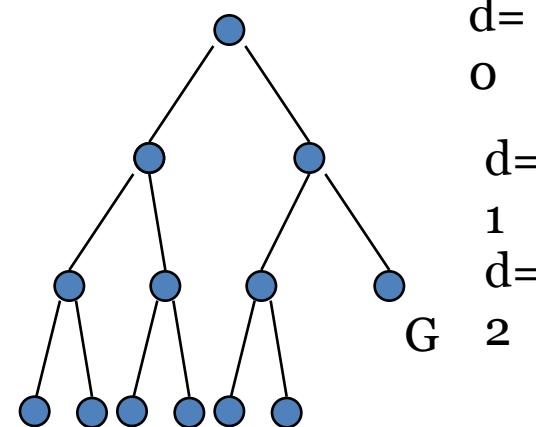
- Space complexity? $O(b^d)$
same as time - every node is kept in the memory
- Main practical drawback? exponential space complexity

Complexity of Breadth-First Search

- **Time Complexity**

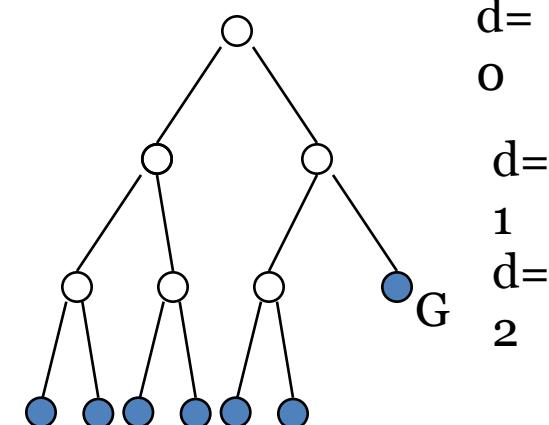
- assume (worst case) that there is 1 goal leaf at the RHS at depth d
- so BFS will generate

$$\begin{aligned} &= b + b^2 + \dots + b^d + b^{d+1} - b \\ &= O(b^{d+1}) \end{aligned}$$



- **Space Complexity**

- how many nodes can be in the queue (worst-case)?
- at depth d there are b^{d+1} unexpanded nodes in the Q = $O(b^{d+1})$



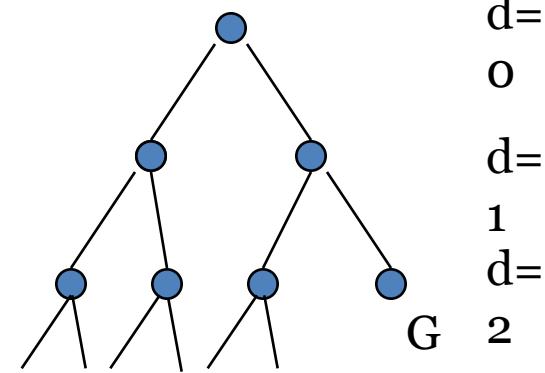
Examples of Time and Memory Requirements for Breadth-First Search

Assuming $b=10$, 10000 nodes/sec,
1kbyte/node

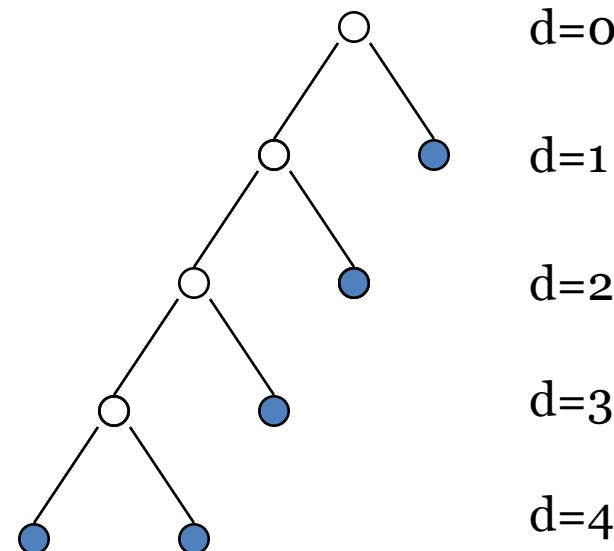
Depth of Solution	Nodes Generated	Time	Memory
2	1100	0.11 seconds	1 MB
4	111,100	11 seconds	106 MB
8	10^9	31 hours	1 TB
12	10^{13}	35 years	10 PB

What is the Complexity of Depth-First Search?

- Time Complexity
 - maximum tree depth = m
 - assume (worst case) that there is 1 goal leaf at the RHS at depth d so DFS will generate $\mathbf{O}(b^m)$



- Space Complexity
 - how many nodes can be in the queue (worst-case)?
 - at depth m we have b nodes
 - and b-1 nodes at earlier depths
 - total = $b + (m-1)*(b-1) = \mathbf{O}(bm)$



Examples of Time and Memory Requirements for Depth-First Search

Assuming $b=10$, $m = 12$, 10000 nodes/sec,
1kbyte/node

Depth of Solution	Nodes Generated	Time	Memory
2	10^{12}	3 years	120kb
4	10^{12}	3 years	120kb
8	10^{12}	3 years	120kb
12	10^{12}	3 years	120kb

Depth-First Search (DFS) Properties

- **Complete?**
 - **No.** Not complete if tree has unbounded depth
- **Optimal?**
 - **No.** Solution found first may not be the shortest possible
- **Time complexity?** $O(b^m)$
 - **Exponential exponential in the maximum depth of the search tree m**
- **Space complexity?** $O(bm)$
 - **Linear linear in the maximum depth of the search tree m**

Comparing DFS and BFS

- **Time complexity:** same, but
 - In the worst-case BFS is always better than DFS
 - Sometime, on the average DFS is better if:
 - many goals, no loops and no infinite paths
- BFS is much worse memory-wise
 - DFS is linear space
 - BFS may store the whole search space.
- In general
 - BFS is better if goal is not deep, if infinite paths, if many loops, if small search space
 - DFS is better if many goals, not many loops,
 - DFS is much better in terms of memory

Artificial Intelligence

Problem Solving using Search (Single agent search) Uninformed Search

Comparing DFS and BFS

- **Time complexity:** same, but
 - In the worst-case BFS is always better than DFS
 - Sometime, on the average DFS is better if:
 - many goals, no loops and no infinite paths
- BFS is much worse memory-wise
 - DFS is linear space
 - BFS may store the whole search space.
- In general
 - BFS is better if goal is not deep, if infinite paths, if many loops, if small search space
 - DFS is better if many goals, not many loops,
 - DFS is much better in terms of memory

Example:

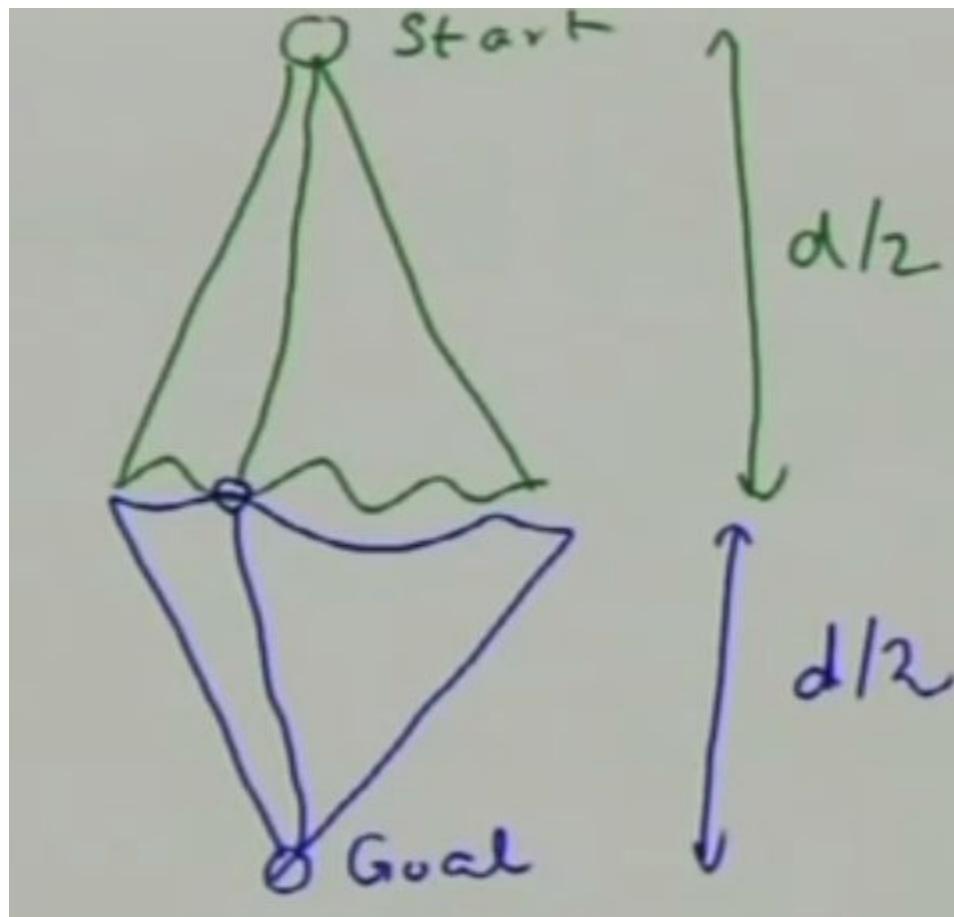
- **For example, given a family tree if one were looking for someone on the tree who's still alive, then it would be safe to assume that person would be on the bottom of the tree.**
 - This means that a BFS would take a very long time to reach that last level. A DFS, however, would find the goal faster.
- **But, if one were looking for a family member who died a very long time ago, then that person would be closer to the top of the tree.**
 - Then, a BFS would usually be faster than a DFS. So, the advantages of either vary depending on the data and what you're looking for.

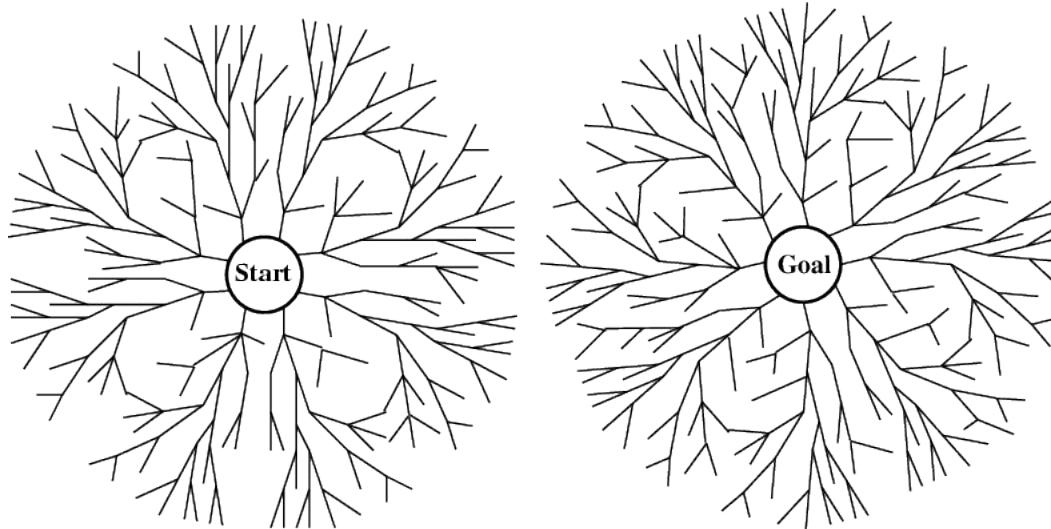
Search Strategies

- Uninformed/Blind Search
 - DFS
 - BFS
 - Bidirectional search
- Informed Search
- Constraint satisfaction

Bidirectional Search

- Idea
 - **simultaneously search forward from S and backwards from G stop when both “meet in the middle”**
 - need to keep track of the intersection of 2 open sets of nodes
- What does searching backwards from G mean
 - need a way to specify the predecessors of G
 - this can be difficult,
 - e.g., predecessors of checkmate in chess?
 - what if there are multiple goal states?
 - what if there is only a goal test, no explicit list?





- Alternate searching from the start state toward the goal and from the goal state toward the start.
- Stop when the frontiers intersect.
- Works well only when there are unique start and goal states.
- Requires the ability to generate “predecessor” states.
- Can (sometimes) lead to finding a solution more quickly.
- Time complexity: $O(b^{d/2})$. Space complexity: $O(b^{d/2})$.

Complexity

time and space complexity are: $O(b^{d/2})$

Comparison

	Breadth first	Depth first	Iterative deepening	Bidirectional (if applicable)
Time	b^d	b^d	b^d	$b^{d/2}$
Space	b^d	bm	bd	$b^{d/2}$
Optimum?	Yes	No	Yes	Yes
Complete?	Yes	No	Yes	Yes

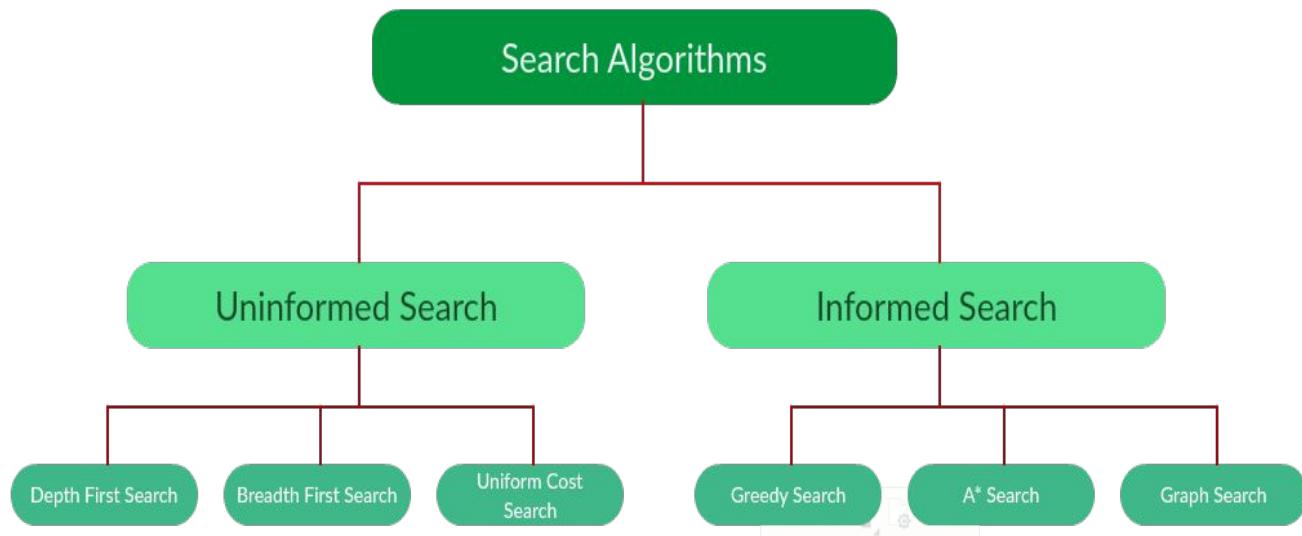
b – branching factor

m – maximum depth

d – depth of optimal solution

l – depth limit

Artificial Intelligence



Lecture 7: Informed Search

Search Strategies

- Search Strategies are classified into
 - **uninformed search** (or) **Blind search**
 - **informed search** (or) **Heuristic search**

Uninformed Vs Informed

- **Uninformed search methods** that systematically explore the state space and find the goal. They are inefficient in most cases.
- **Informed search methods** use problem specific knowledge, and may be more efficient. At the heart of such algorithms there is the concept of a heuristic function.

Heuristics

- Heuristic means “**rule of thumb**”.
- To quote J. Pearl, “*Heuristics are criteria, methods or principles for deciding which among several alternative courses of action promises to be the most effective in order to achieve some goal*”.
- In heuristic search or informed search, heuristics are used to identify the **most promising search path**.

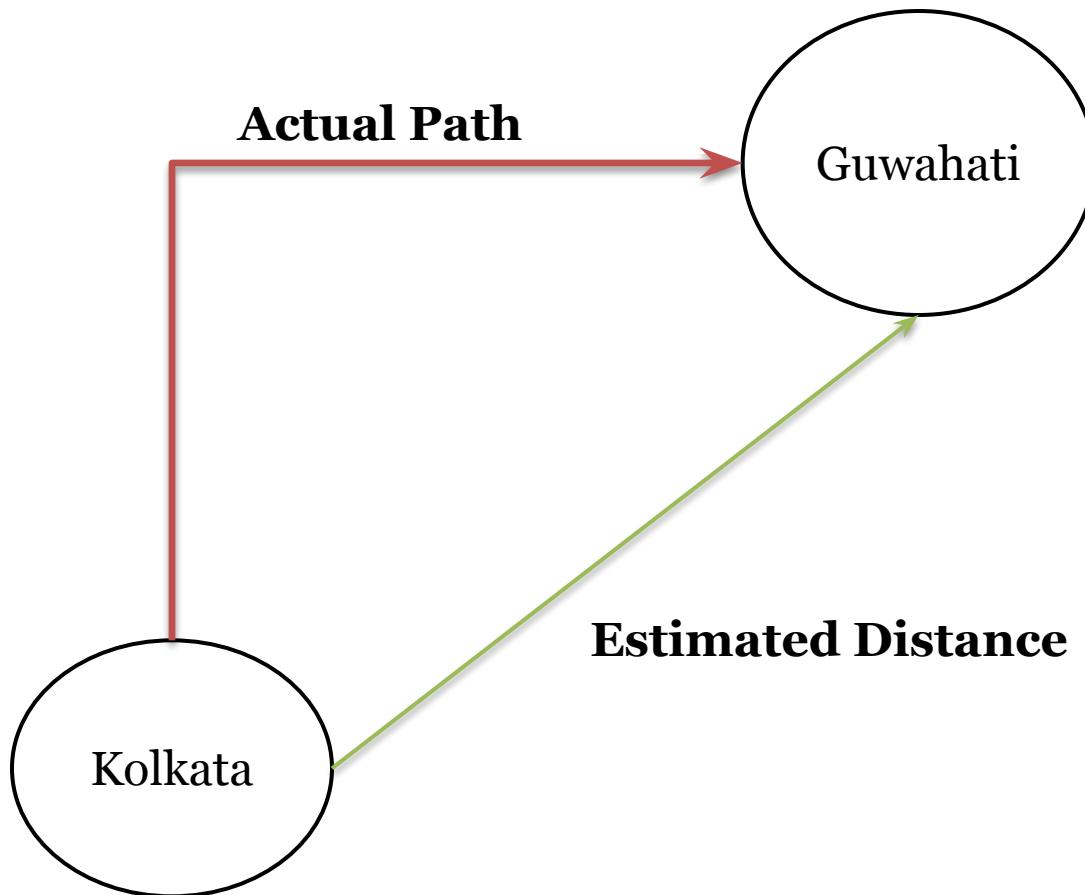
- **Informed (or heuristic) search** uses problem-specific heuristics to improve efficiency
 - **Best-first**
 - **A***
 - **RBFS**
 - **SMA***
 - **Techniques for generating heuristics**

Example of Heuristic Function

- A heuristic function at a **node n** is an estimate of the optimum cost from the current node to **a goal**. It is denoted by $h(n)$.

$h(n)$ = estimated cost of the cheapest path from node n to a goal node

- Example 1: We want a path from Kolkata to Guwahati



Heuristic for Guwahati may be straight-line distance between Kolkata and Guwahati

$$h(\text{Kolkata}) = \text{euclideanDistance}(\text{Kolkata}, \text{Guwahati})$$

Example 2: 8-puzzle: Misplaced Tiles Heuristics
is the number of tiles out of place.

2	8	3
1	6	4
	7	5

Initial State

1	2	3
8		4
7	6	5

Goal state

2	8	3
1	6	4
	7	5

Initial State

1	2	3
8		4
7	6	5

Goal state

Tiles: 2, 8, 1, 6 and 7 \Rightarrow Not in Correct location

The first picture shows the current state n , and the second picture the goal state.

$$h(n) = 5$$

5 tiles are not their correct location

We must make at least 5 moves to move them to their correct location

So, $h(n)$ is the underestimate of actual number of step required to move to their goal state

Heuristic Example

8-Puzzle: Manhattan Distance (distance tile is out of place)

2	8	3
1	6	4
	7	5

Initial State

1	2	3
8		4
7	6	5

Goal state

This heuristic sums the distance that the tiles are out of place. The distance of a tile is measured by the sum of the differences in the x-positions and the y-positions.

- For the above example, using the Manhattan distance heuristic,

$$h(n) = 1 + 1 + 0 + 0 + 0 + 1 + 1 + 2 = 6$$

8 puzzle problem using heuristic

1	2	3
	4	6
7	5	8

Initial state

1	2	3
4	5	6
7	8	

final state

Heuristic value=1+1+1=3

Set of action={up, down, left, right}

1	2	3
4	5	6
7	8	

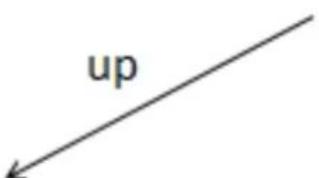
final state

1	2	3
	4	6
7	5	8

H=3

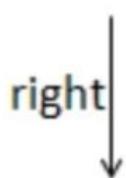
	2	3
1	4	6
7	5	8

H=4



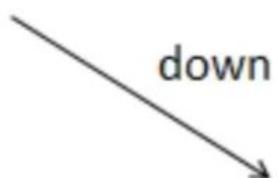
1	2	3
4		6
7	5	8

H=2



1	2	3
7	4	6
	5	8

H=4



1	2	3
4	5	6
7	8	

final state

1	2	3
	4	6
7	5	8

H=3

	2	3
1	4	6
7	5	8

H=4

1	2	3
4		6
7	5	8

H=2

1	2	3
7	4	6
	5	8

H=4

up

right

down

1	2	3
	4	6
7	5	8

H=3

1	2	3
4	5	6
7		8

H=1

1	2	3
4	6	
7	5	8

H=3

1		3
4	2	6
7	5	8

H=3

right

down

left

up

1	2	3
4	5	6
7	8	

1	2	3
	4	6
7	5	8

H=3

final state

	2	3
1	4	6
7	5	8

H=4

1	2	3
4		6
7	5	8

H=2

1	2	3
7	4	6
	5	8

H=4

1	2	3
	4	6
7	5	8

H=3

1	2	3
4	5	6
7		8

H=3

3

8

H=3

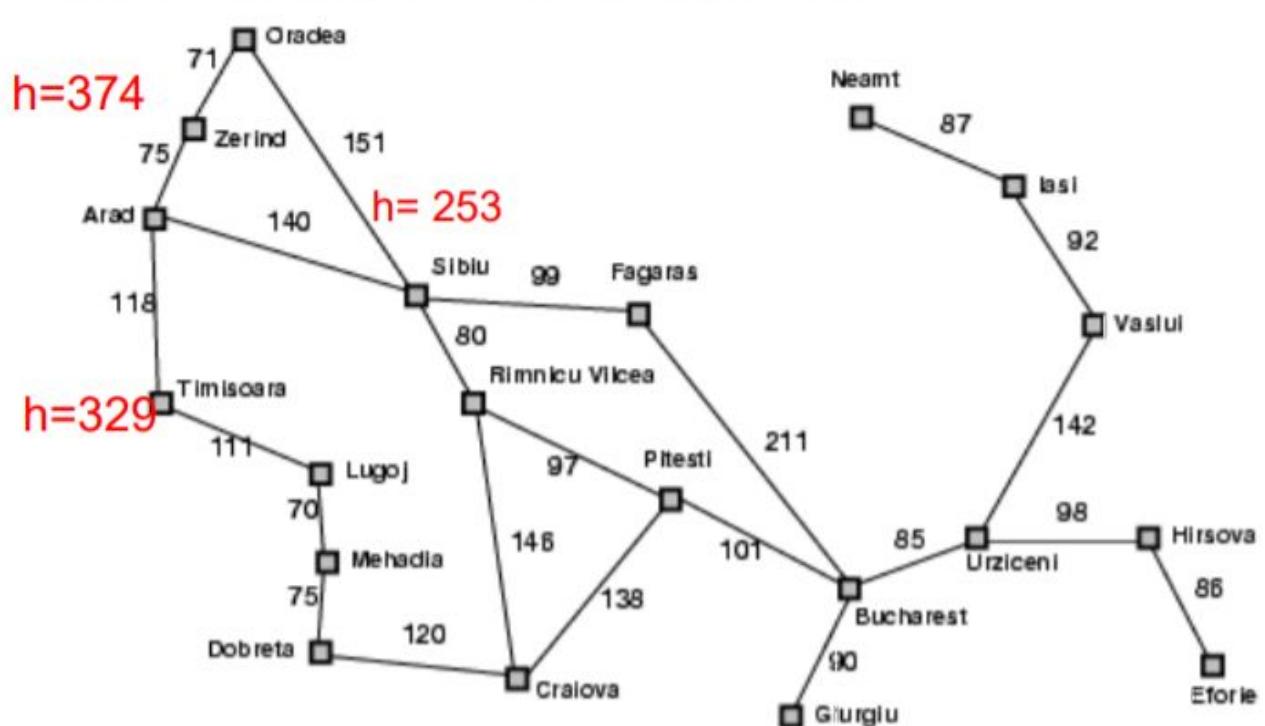
1	2	3
4	5	6
7	8	

$$H=0$$



Heuristic Search

- State-Space Search: every problem is like search of a map
- A problem solving agent finds a path in a **state-space graph** from **start state** to **goal state**, using **heuristics**



	Straight-line distance to Bucharest
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Heuristic = straight-line distance



Searching Strategies

- **Blind search** → traversing the search space until the goal nodes is found (might be doing exhaustive search).
- *Techniques :* **Breadth First Uniform Cost ,Depth first, Interactive Deepening search.**

- **Heuristic search** → search process takes place by traversing search space with applied rules (information).
- *Techniques:* **Greedy Best First Search, A* Algorithm**

Best-First Search

- Idea: use an **evaluation function** $f(n)$ for each node
 - $f(n)$ provides an estimate for the total cost.
 - Expand the node n with smallest $f(n)$.
- Implementation:
Order the nodes in fringe increasing order of cost.
- Special cases:
 - greedy best-first search
 - A* search

Greedy Search

- **Idea:** Expand the node with the **smallest estimated cost** to reach the goal.

- We use a heuristic function

$$f(n) = h(n)$$

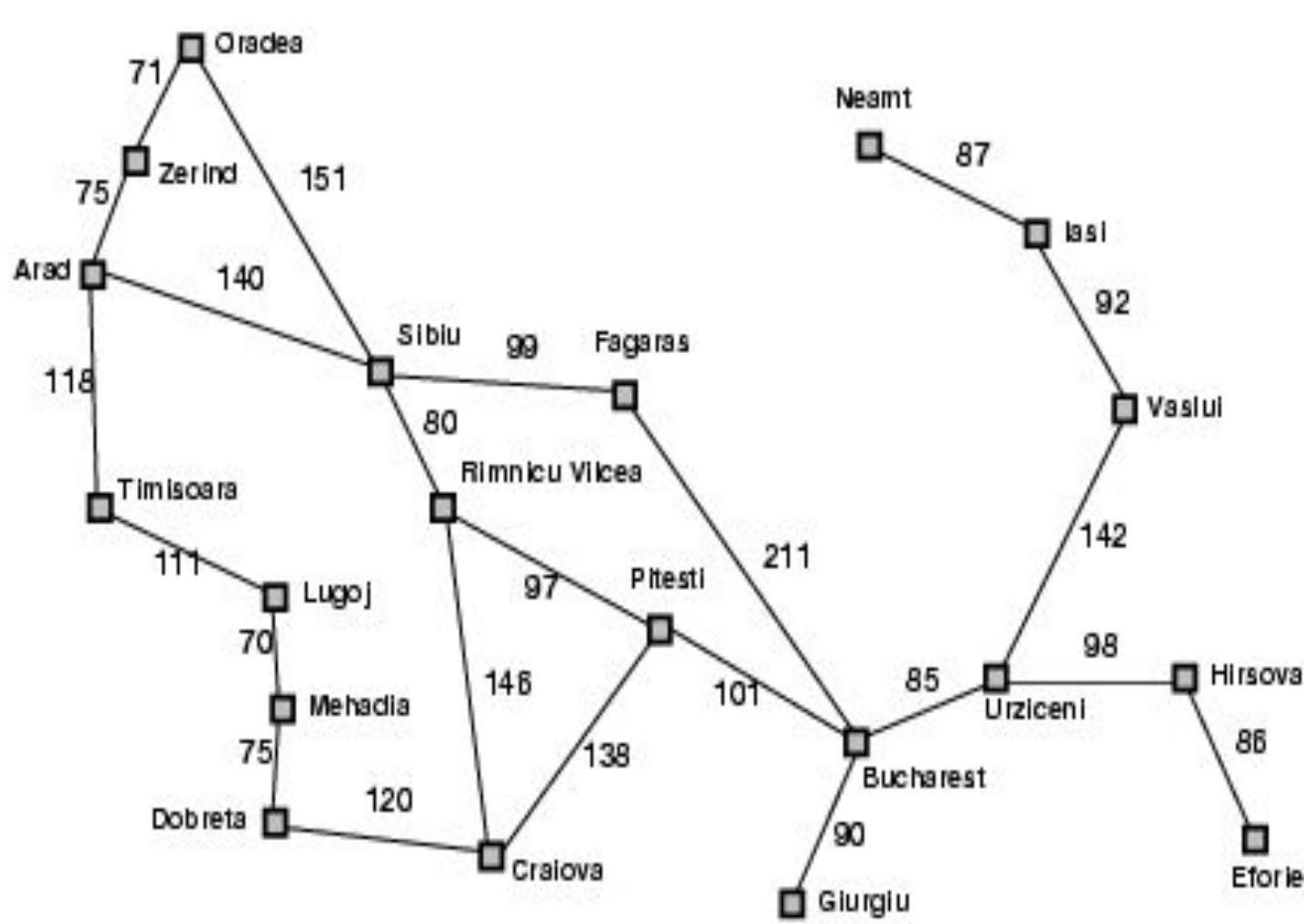
$h(n)$ estimates the distance remaining to a goal.

- Greedy algorithms often perform very well.

- **Disadvantage:**

- They tend to find good solutions quickly, although not always optimal ones.
- The algorithm is also incomplete, and it may fail to find a solution even if one exists.

Romania with straight-line dist.

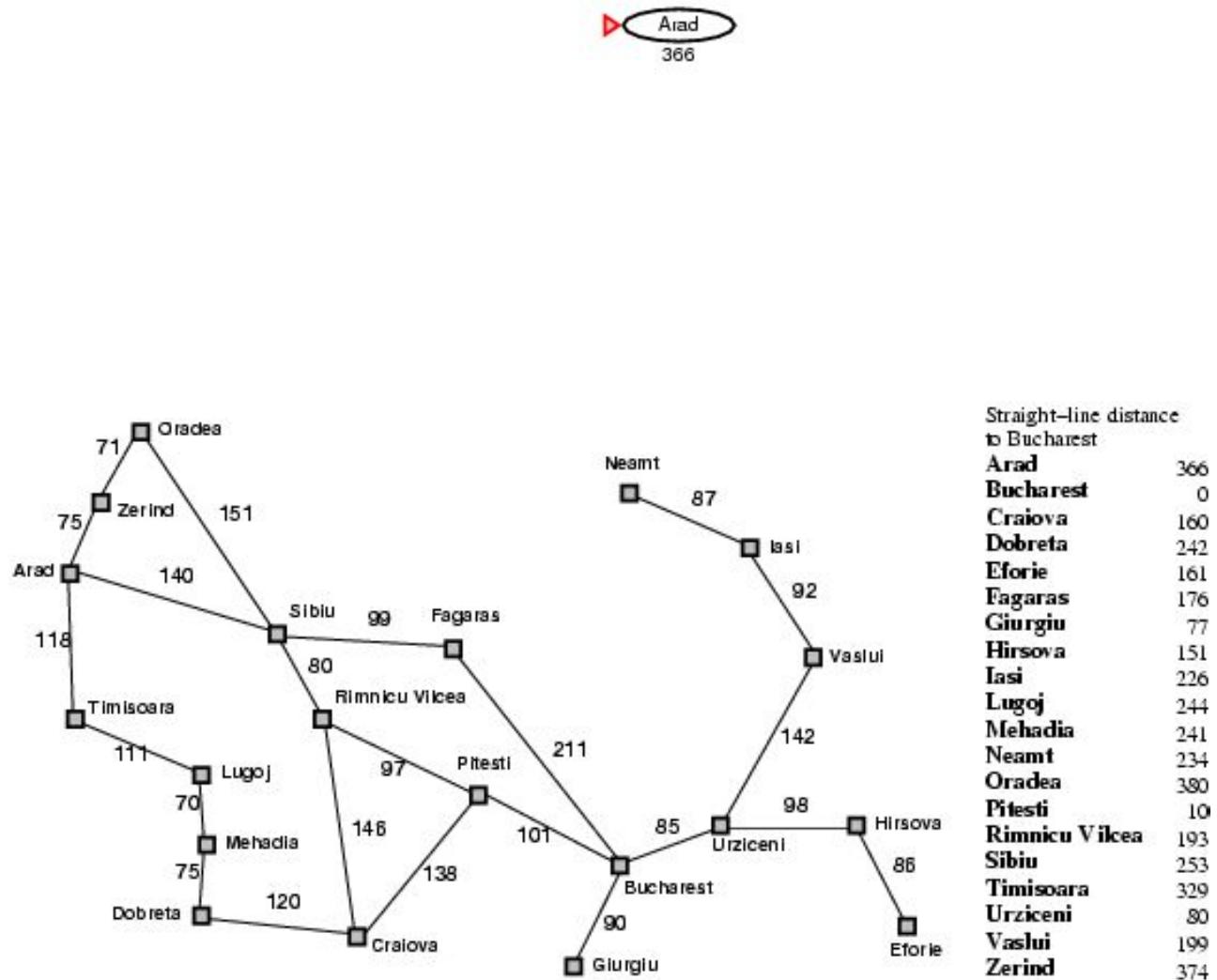


Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

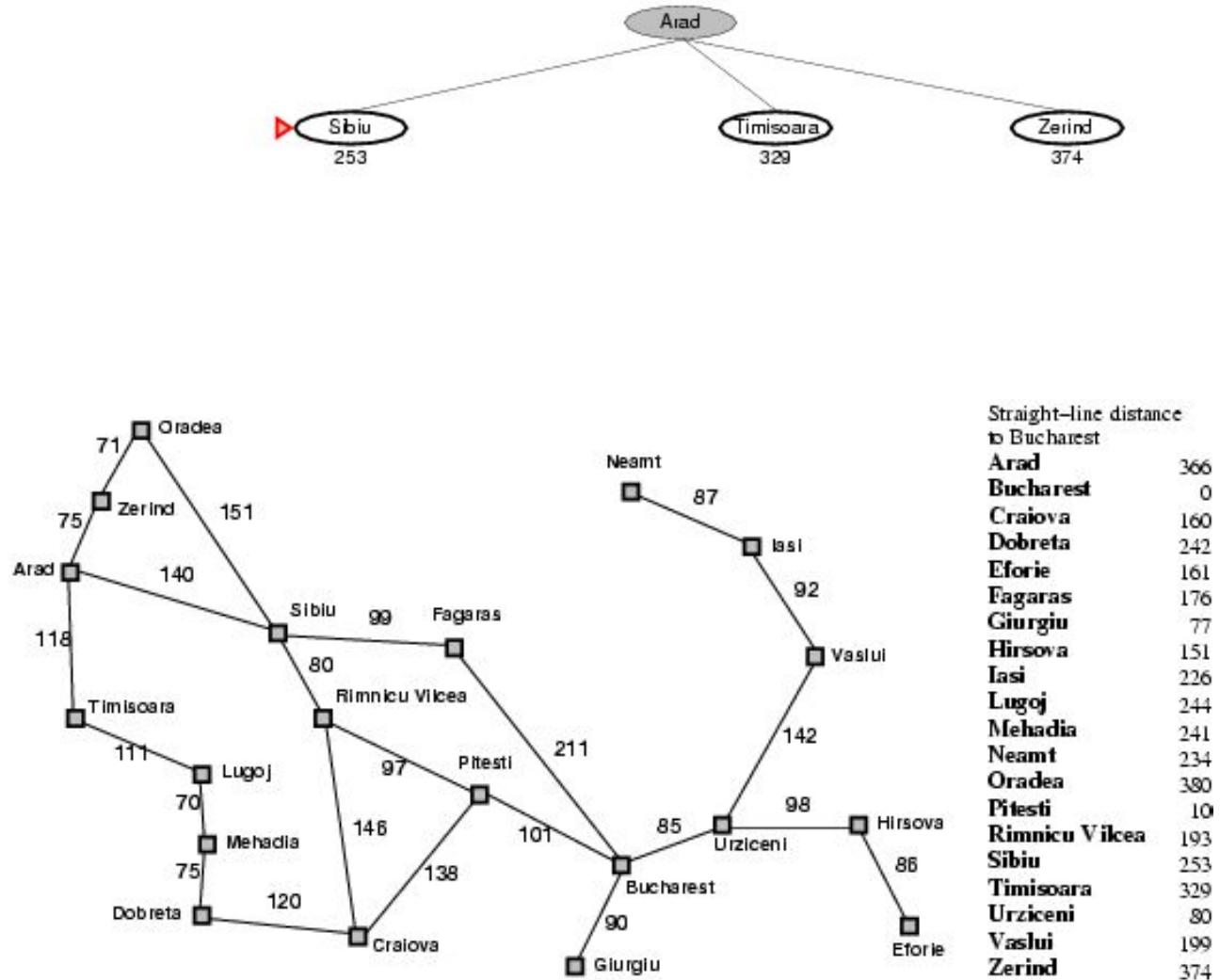
Greedy best-first search

- $f(n)$ = estimate of cost from n to *goal*
- e.g., $f(n)$ = straight-line distance from n to Bucharest
- Greedy best-first search expands the node that **appears** to be closest to goal.

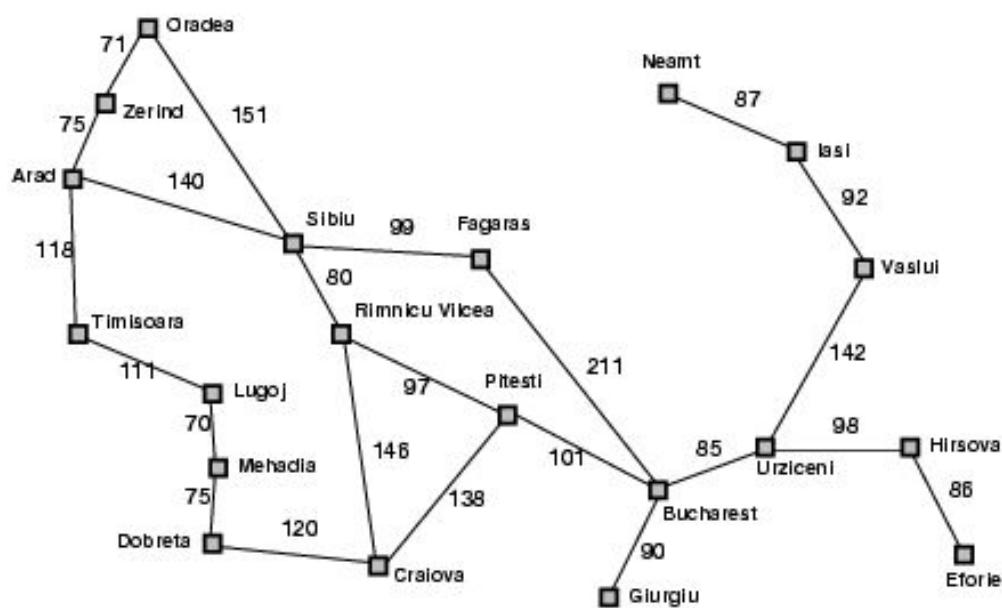
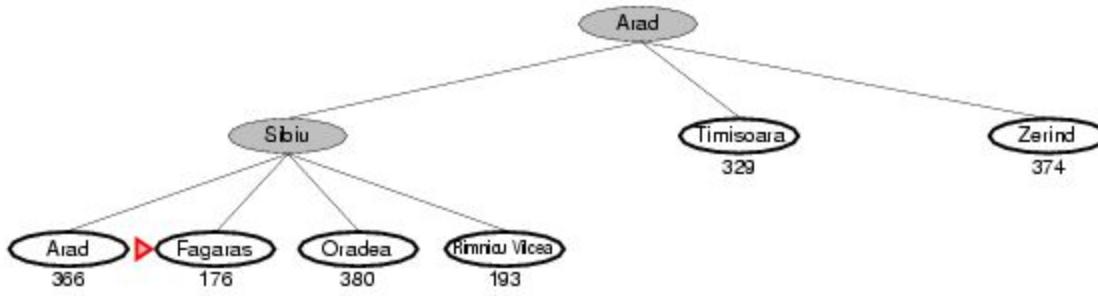
Greedy best-first search example



Greedy best-first search example

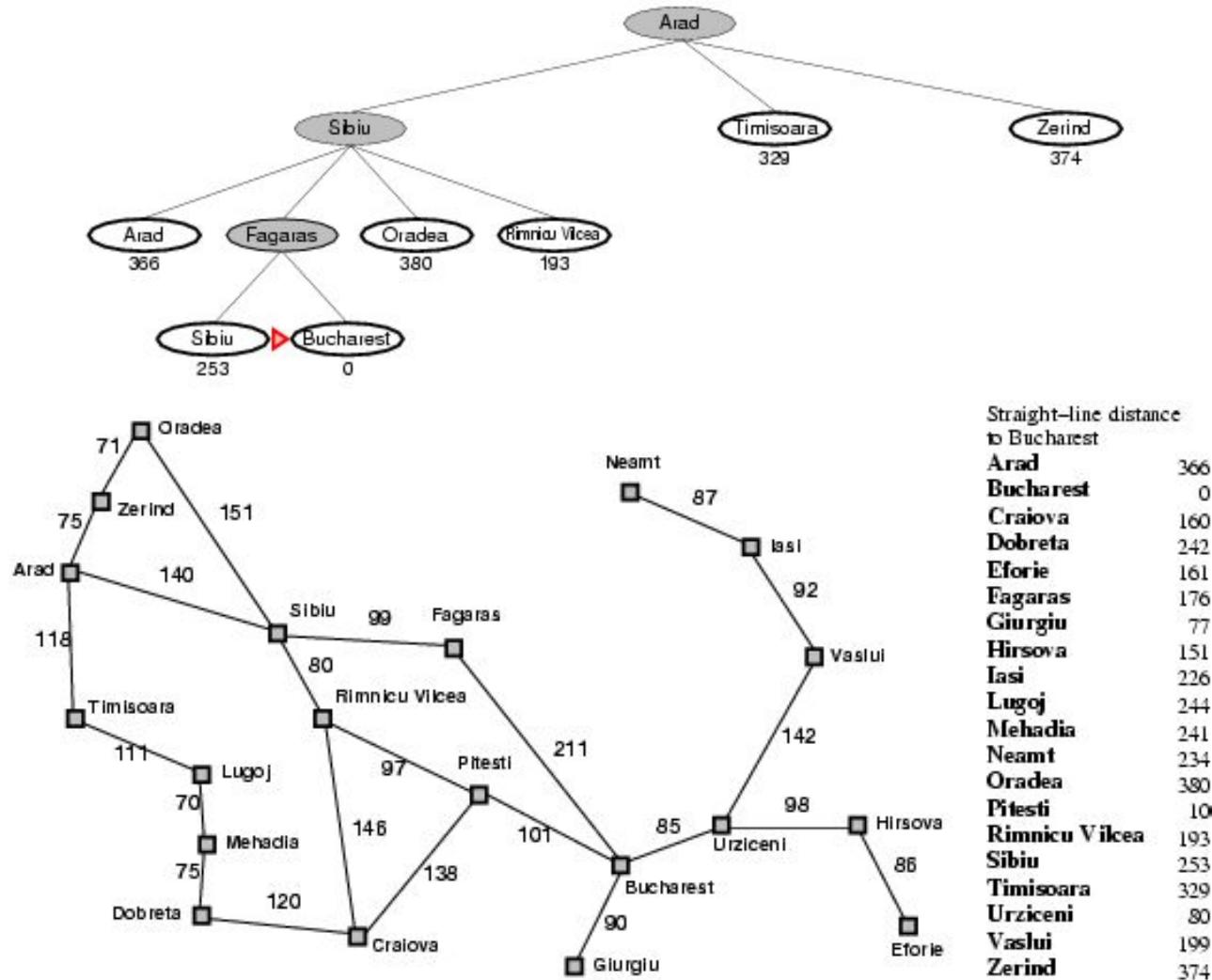


Greedy best-first search example



Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Greedy best-first search example



Properties of greedy best-first search

- Complete? No – can get stuck in loops.
- Time? $O(b^m)$, but a good heuristic can give dramatic improvement
- Space? $O(b^m)$ - keeps all nodes in memory
- Optimal? No

e.g. Arad \square Sibiu \square Rimnicu Virea \square Pitesti \square Bucharest is shorter!

A^{*} search

- Idea: avoid expanding paths that are already expensive

Evaluation function $f(n) = g(n) + h(n)$

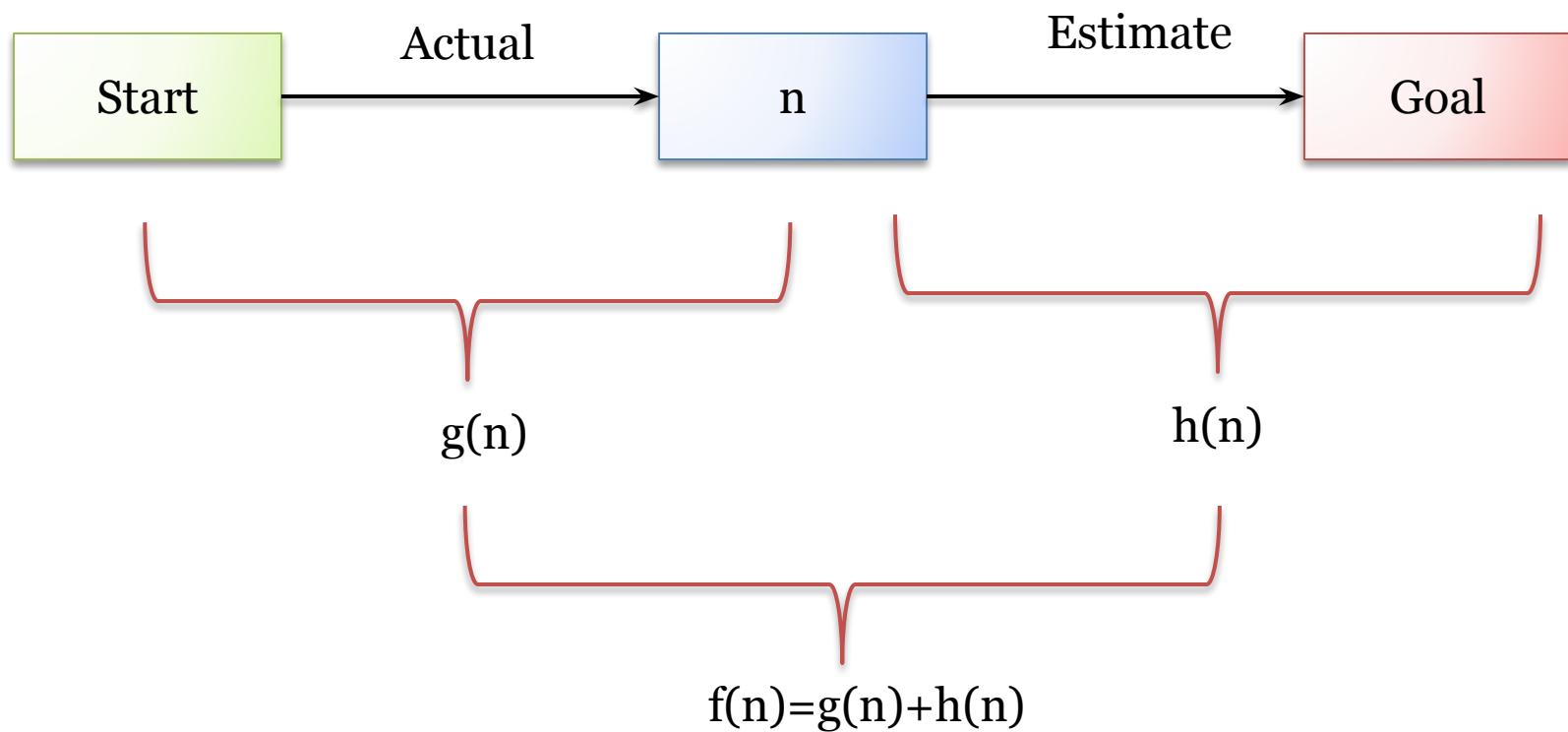
$g(n)$ = cost so far to reach n

$h(n)$ = estimated cost from n to goal

$f(n)$ = estimated total cost of path through n to goal

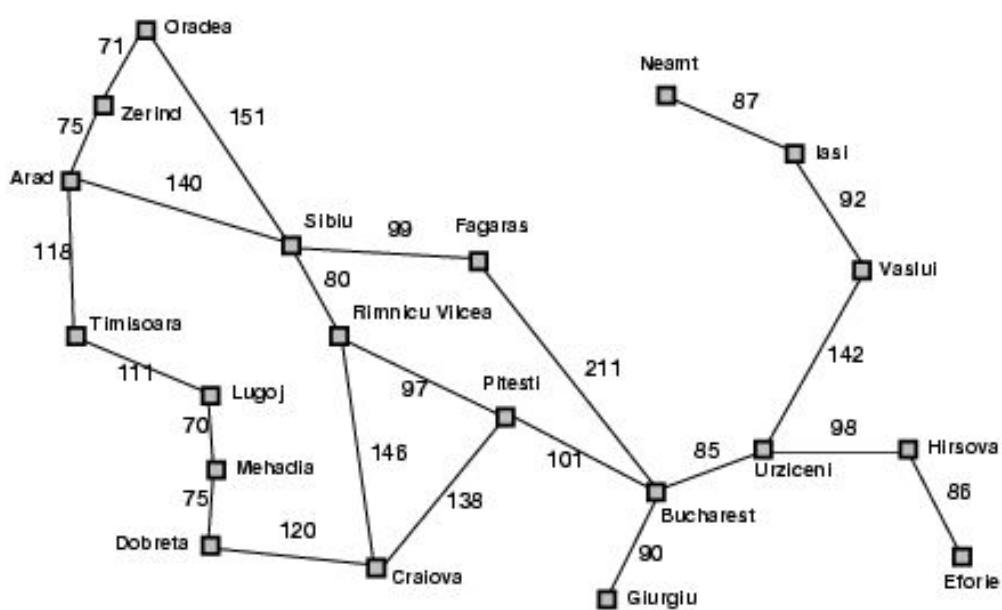
A* Search

$f(n)$ = estimated total cost of path through n to goal



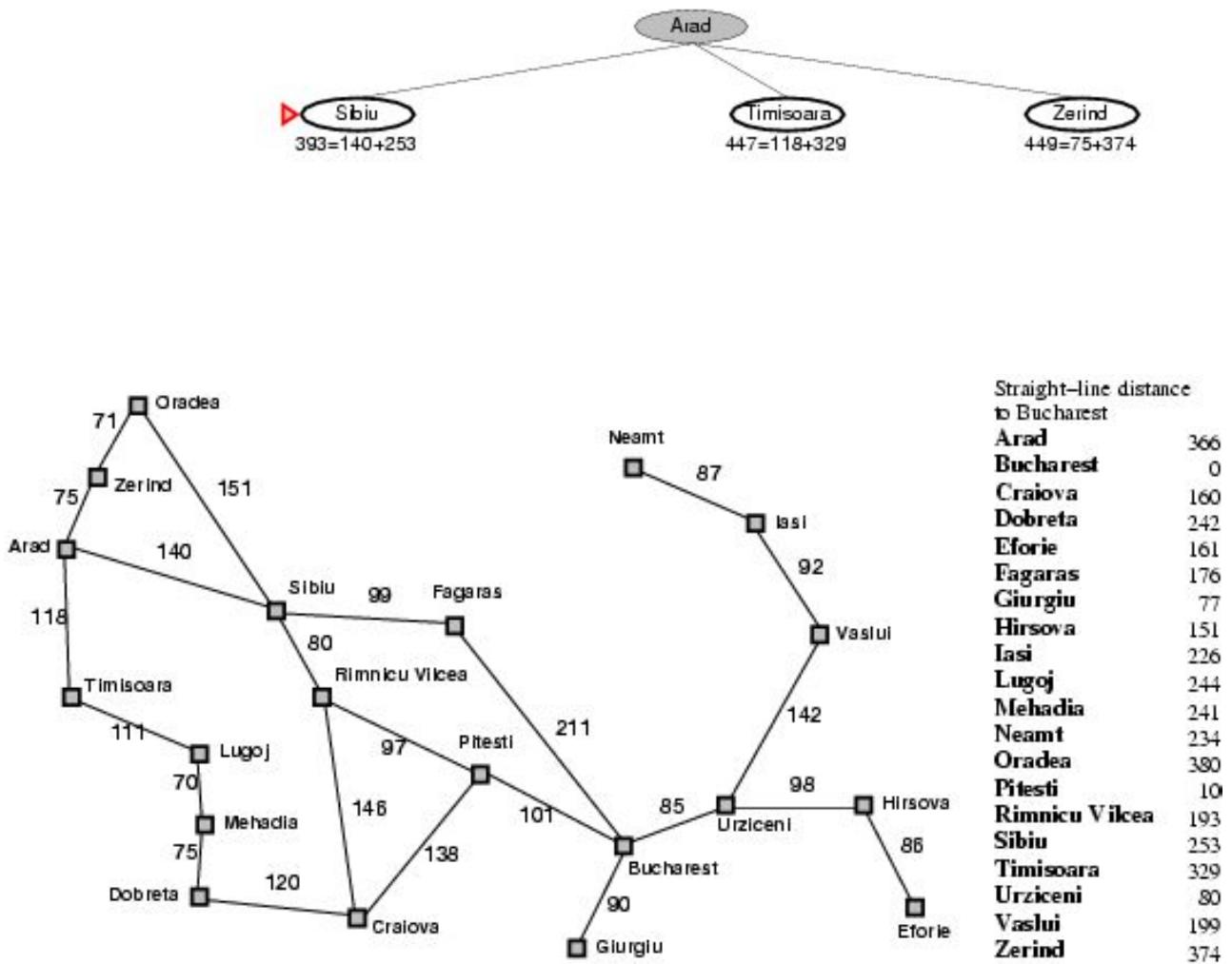
A* search example

► Arad
366=0+366

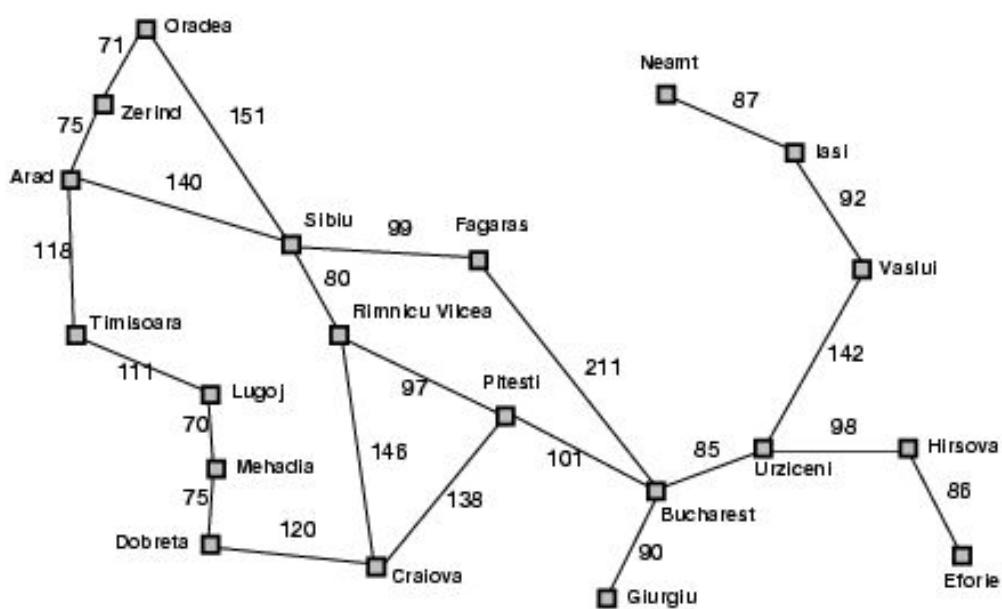
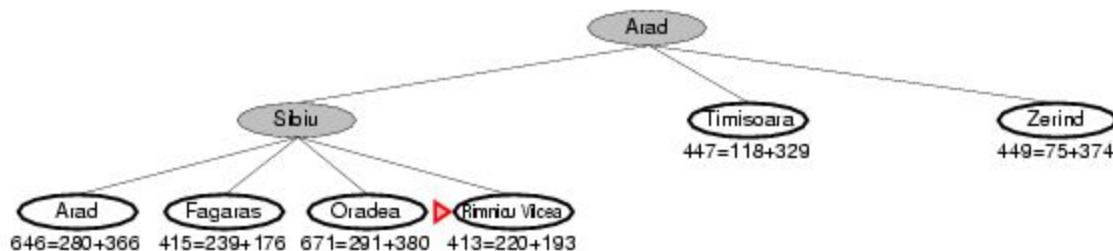


Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

A* search example

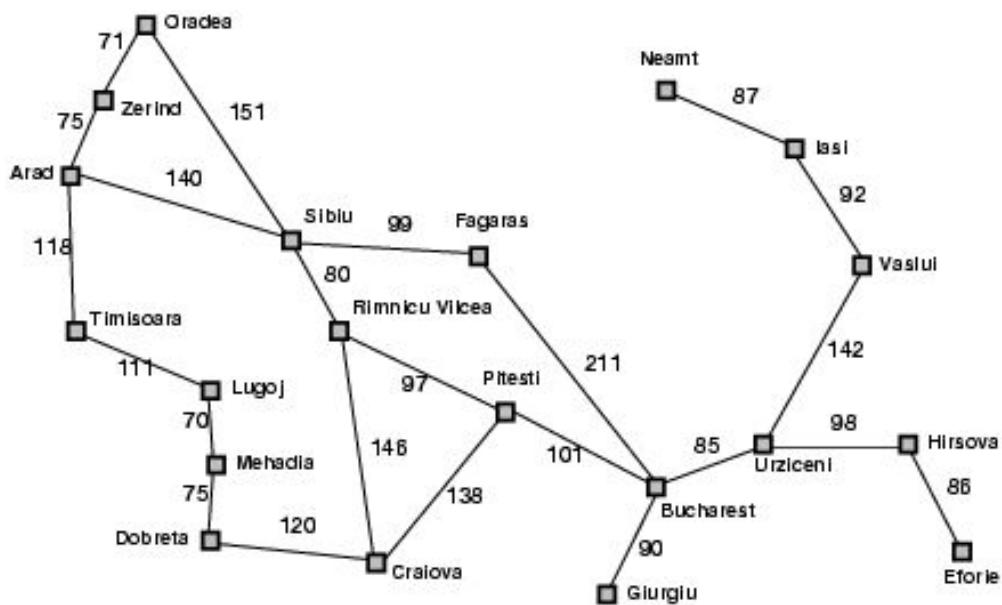
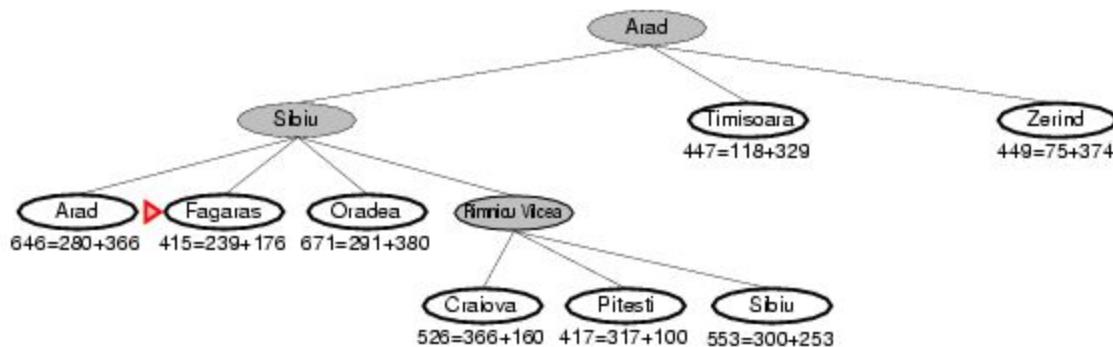


A* search example



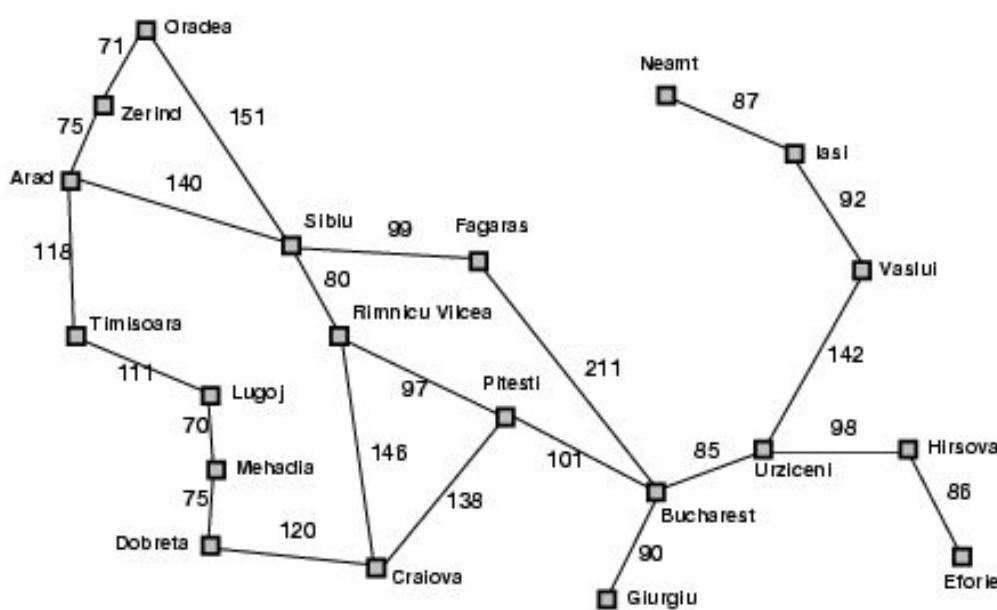
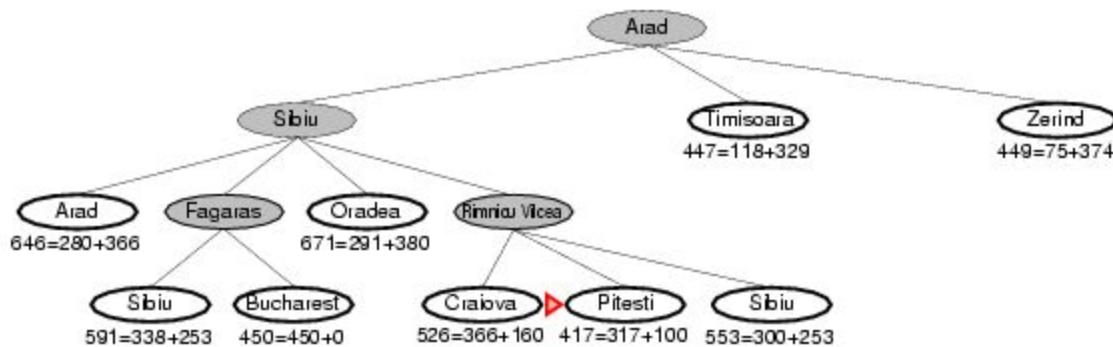
Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

A* search example

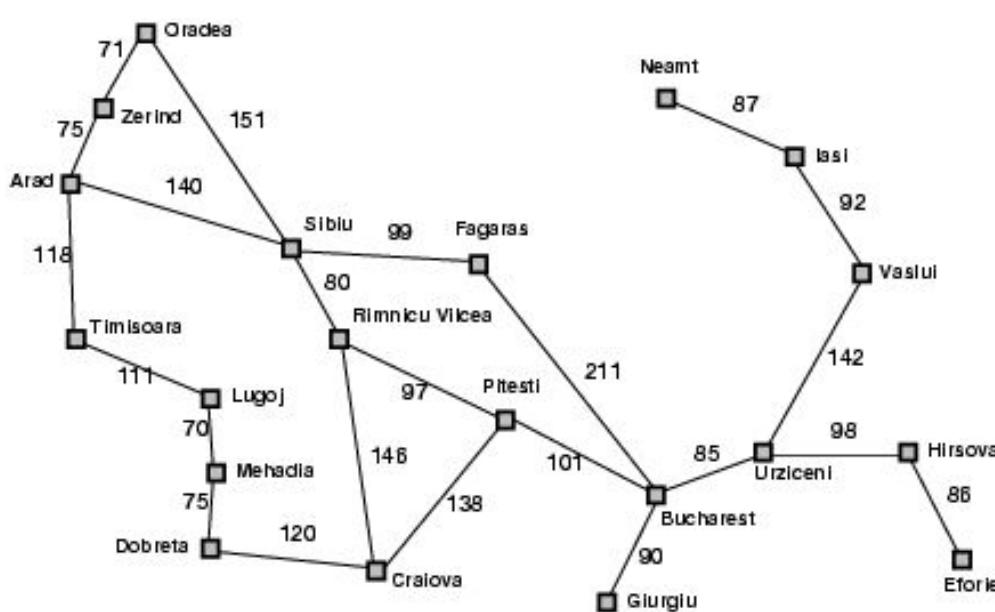
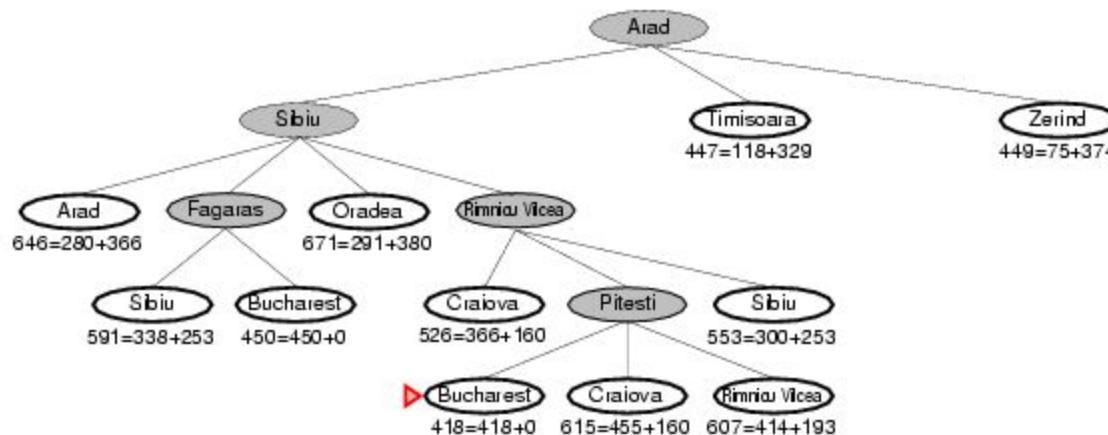


Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

A* search example



A* search example



Admissible heuristics

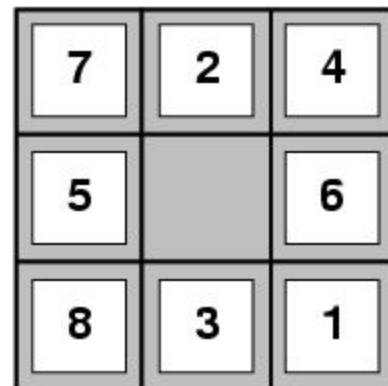
- A heuristic $h(n)$ is **admissible** if for every node n ,
 $h(n) \leq h^*(n)$, where $h^*(n)$ is the **true** cost to reach the goal state from n .
- An admissible heuristic **never overestimates** the cost to reach the goal, i.e., it is **optimistic**

Admissible heuristics

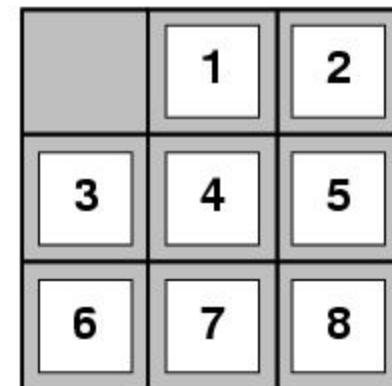
E.g., for the 8-puzzle:

- $h_1(n)$ = number of misplaced tiles
- $h_2(n)$ = total Manhattan distance
(i.e., no. of squares from desired location of each tile)

- $h_1(S) = ?$
- $h_2(S) = ?$



Start State

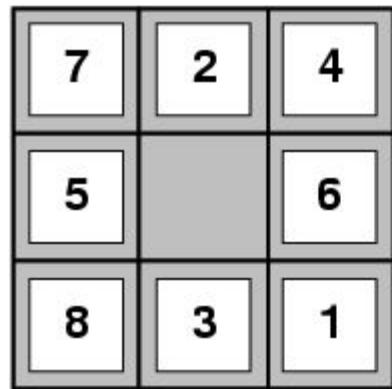


Goal State

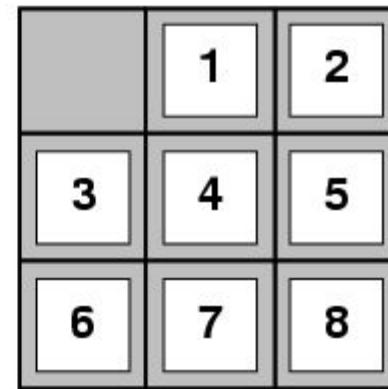
Admissible heuristics

E.g., for the 8-puzzle:

- $h_1(n)$ = number of misplaced tiles
- $h_2(n)$ = total Manhattan distance
(i.e., no. of squares from desired location of each tile)



Start State



Goal State

- $\underline{h_1(S) = ?}$ 8
- $\underline{h_2(S) = ?}$ $3+1+2+2+2+3+3+2 = 18$

Dominance

- If $h_2(n) \geq h_1(n)$ for all n (both admissible)
then h_2 **dominates** h_1
- h_2 is better for search: it is guaranteed to
expand less or equal no. of nodes.

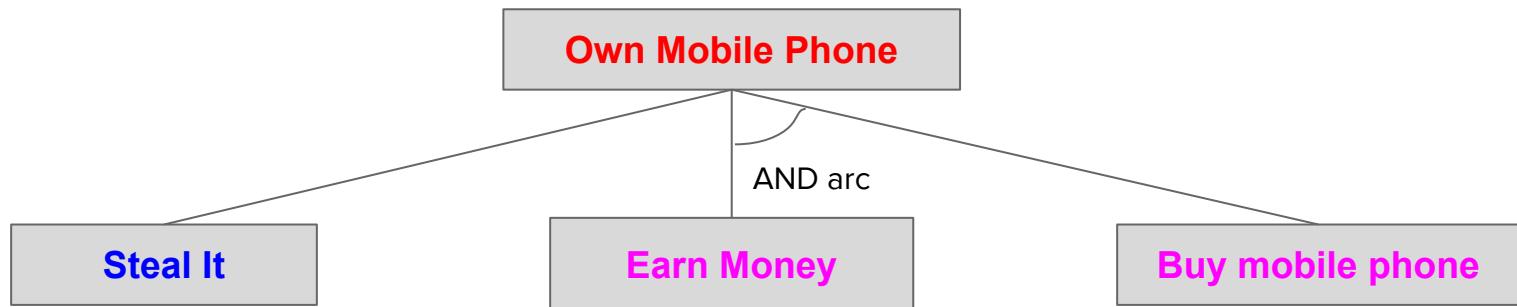
AO* Algorithm

Introduction

- Its an informed search and works as **best first search**.
- AO* Algorithm is based on problem decomposition (Breakdown problem into small pieces).
- Its an efficient method to explore a solution path.
- AO* is often used for the common pathfinding problem in applications such as video games, but was originally designed as a general graph traversal algorithm.
- It finds applications in diverse problems, including the problem of parsing using stochastic grammars in NLP.
- Other cases include an Informational search with online learning.
- It is useful for searching game trees, problem solving etc.

AND-OR Graph

- AND-OR graph is useful for representing the solution of problems that can be solved by decomposing them into a set of smaller problems, all of which must then be solved.



AND-OR Graph

- Node in the graph will point both down to its successors and up to its parent nodes.
- Each Node in the graph will also have a heuristic value associated with it.

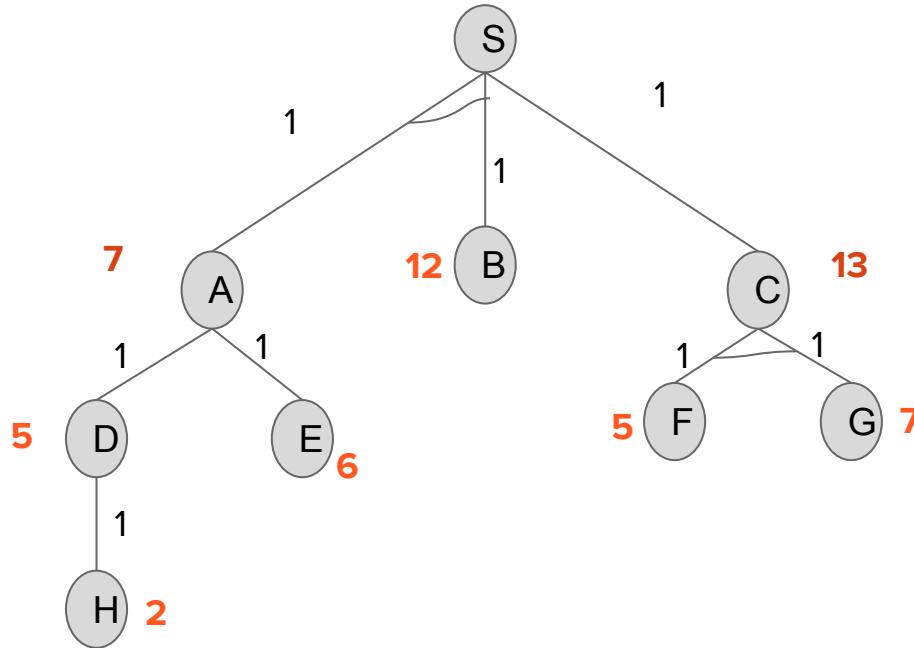
$$f(n) = g(n) + h(n)$$

$f(n)$: Cost function.

$g(n)$: Actual cost or Edge value

$h(n)$: Heuristic/ Estimated value of the nodes

AO* Example 1



AO* Example 1

$$f(n) = g(n) + h(n)$$

Path-1: $f(S-C) = 1+13=14$

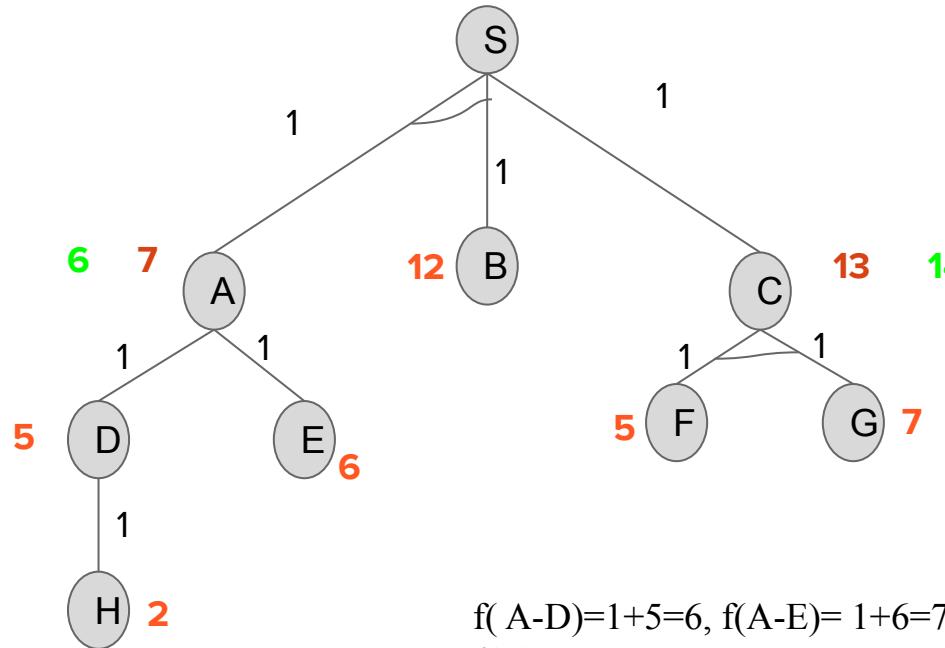
Path-2: $f(S-A-B)=1+1+7+12=21$

$$f(C-F-G)=1+1+5+7=14$$

$$f(S-C)=1+14=15 \text{ (revised)}$$

Revised cost: 15

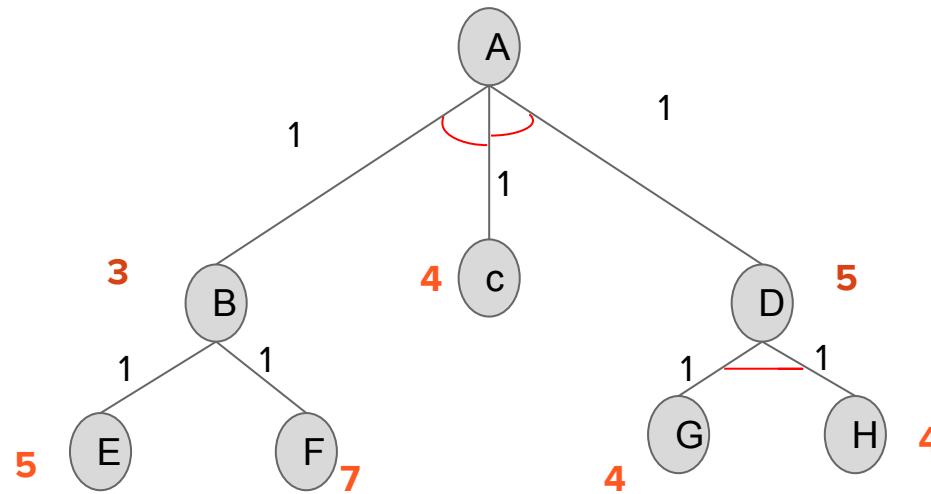
$$\min(14, 21) = 14$$



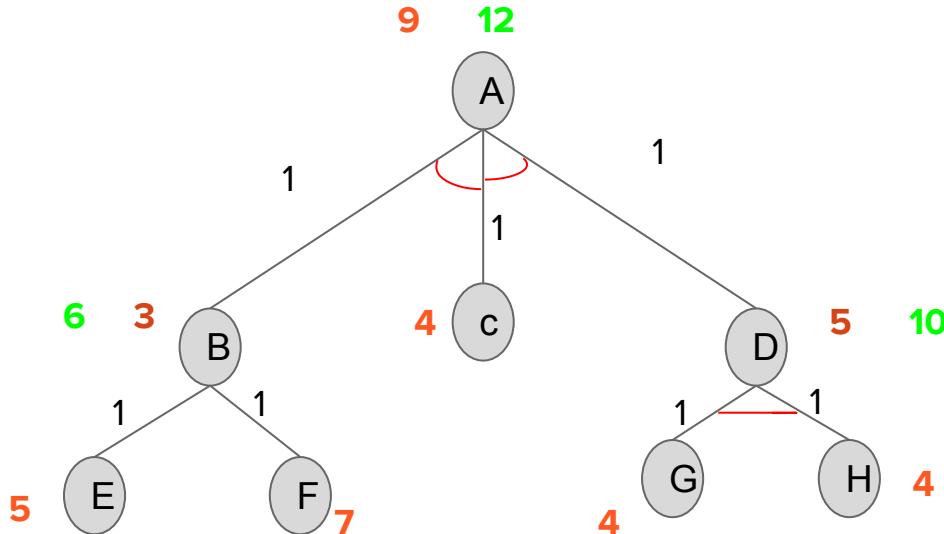
$f(A-D)=1+5=6, f(A-E)=1+6=7$ so revised
 $f(A)=6$

Revised $(S-A-B)=1+6+1+12=20$ X

AO* Example 2



AO* Example 2



$$\text{Path -1: } f(A-B-C) = 1+1+3+4 = 9$$

$$f(B-E) = 1+5 = 6$$

$$f(B-F) = 1+7 = 8$$

$$f(A-B-C) = 1+1+6+4 = 12$$

$$\text{Path-2: } f(A-C-D) = 1+1+4+5 = 11$$

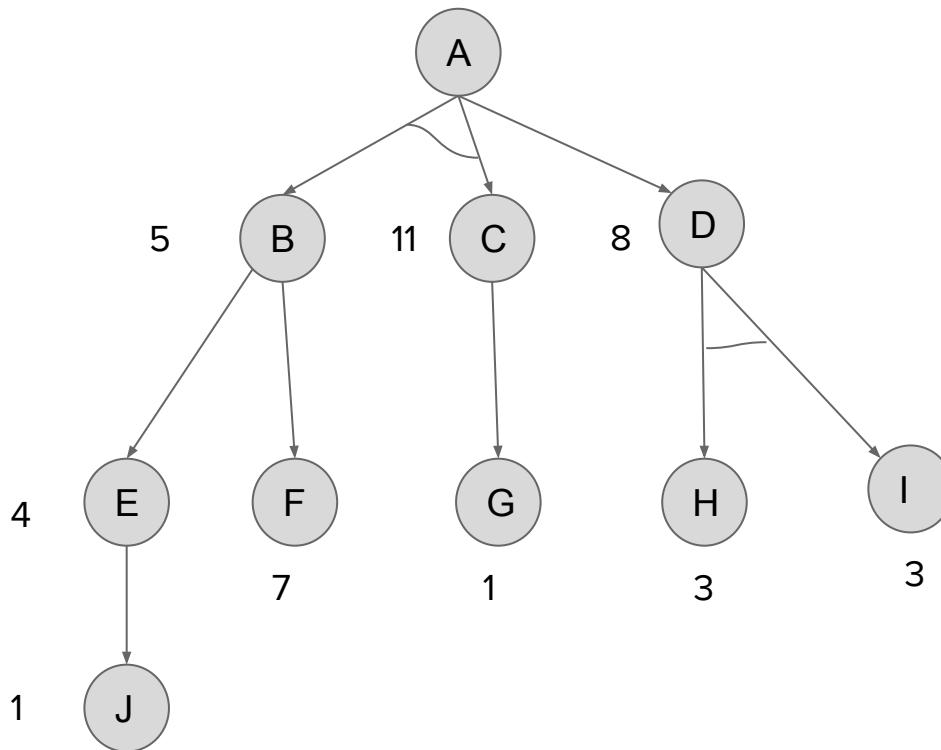
$$f(D-G-H) = 1+1+4+4 = 10$$

$$f(A-C-D) = 1+1+4+10 = 16$$

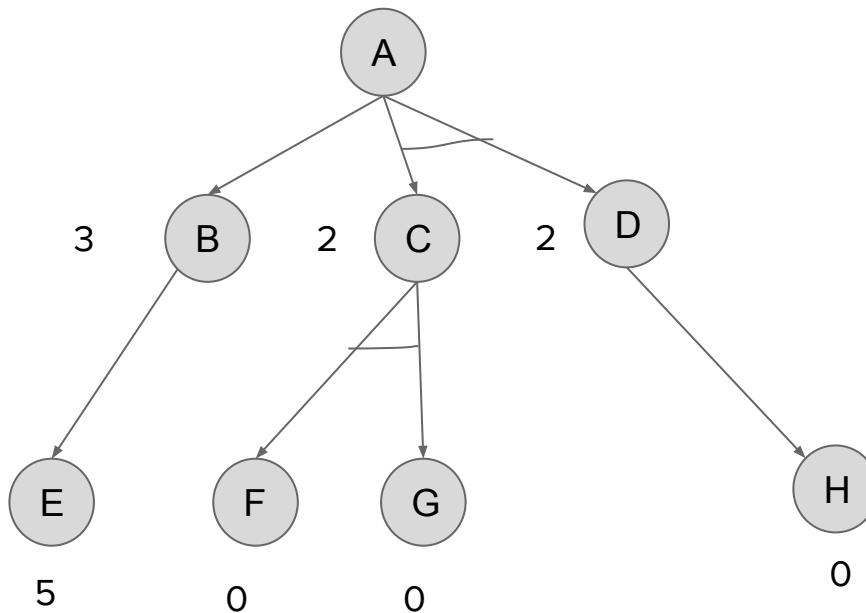
AO* Algorithm:

The algorithm does not explore all the solution path once it find a solution

Example 3



Example 4



AO* Algorithm

1. Initialise the graph to start node
2. Traverse the graph following the current path accumulating nodes that have not yet been expanded or solved
3. Pick any of these nodes and expand it and if it has no successors call this value FUTILITY otherwise calculate only f for each of the successors.
4. If f is 0 then mark the node as SOLVED
5. Change the value of f for the newly created node to reflect its successors by back propagation.
6. Wherever possible use the most promising routes and if a node is marked as SOLVED then mark the parent node as SOLVED.
7. If starting node is SOLVED or value greater than FUTILITY, stop, else repeat from 2.

A*	AO*
<p>It represents an OR graph algorithm that is used to find a single solution (either this or that).</p> <p>It is a computer algorithm which is used in path-finding and graph traversal. It is used in the process of plotting an efficiently directed path between a number of points called nodes.</p> <p>In this algorithm you traverse the tree in depth and keep moving and adding up the total cost of reaching the cost from the current state to the goal state and add it to the cost of reaching the current state.</p>	<p>It represents an AND-OR graph algorithm that is used to find more than one solution by ANDing more than one branch.</p> <p>In this algorithm you follow a similar procedure but there are constraints traversing specific paths.</p> <p>When you traverse those paths, cost of all the paths which originate from the preceding node are added till that level, where you find the goal state regardless of the fact whether they take you to the goal state or not.</p>

Thank You

Lecture:

Hill Climbing

Dr. Partha Pakray

Local Search

Local search methods work on complete state formulations. They keep only a small number of nodes in memory.

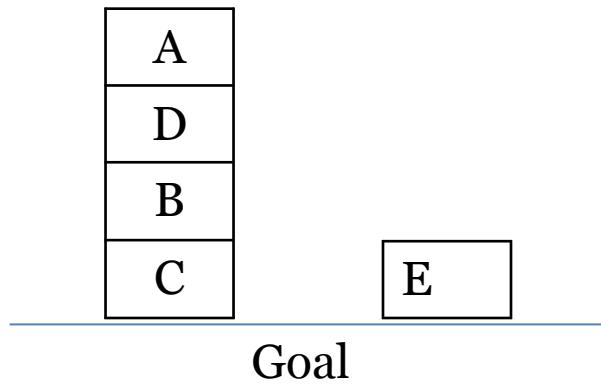
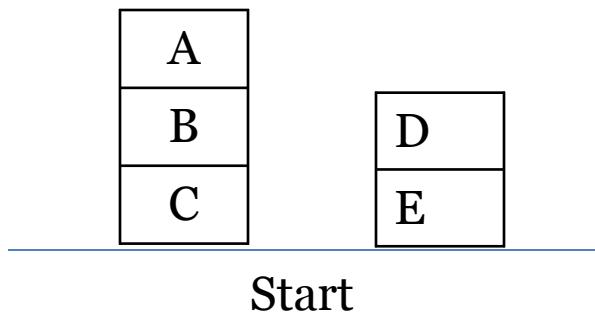
Local search is useful for solving optimization problems:

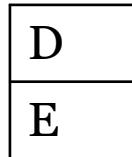
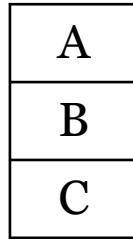
- o Often it is easy to find a solution
- o But hard to find the best solution

Algorithm goal: find optimal configuration (e.g., TSP),

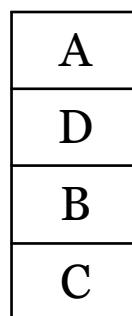
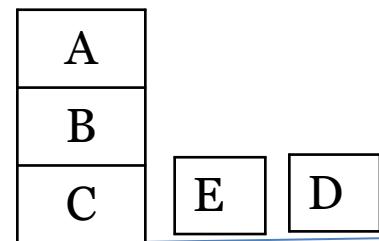
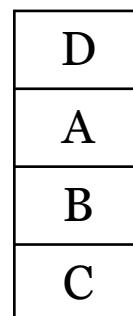
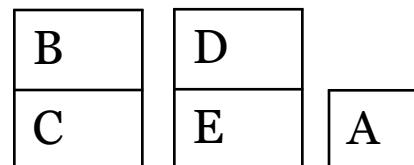
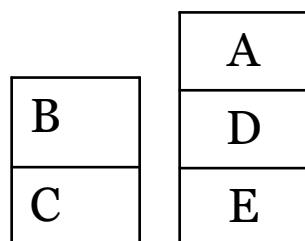
- Hill climbing
 - Gradient descent
 - Simulated annealing
-
- For some problems the state description contains all of the information relevant for a solution. Path to the solution is unimportant.
 - Examples:
 - o map coloring
 - o 8-queens
 - o cryptarithmetic

Block Worlds Domain





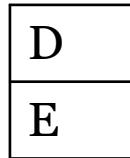
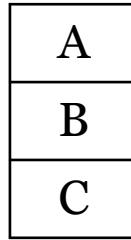
Start



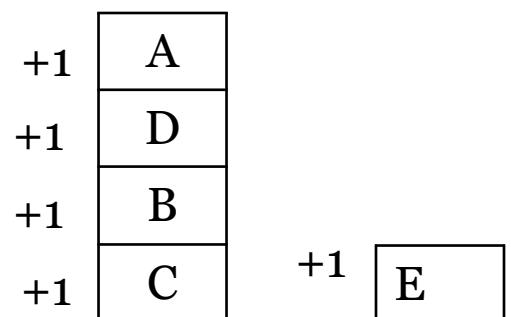
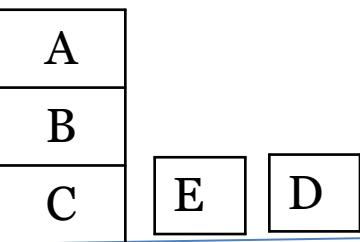
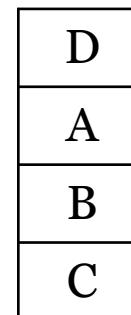
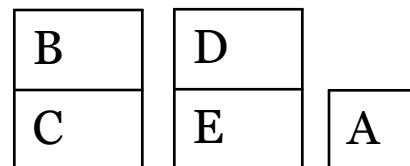
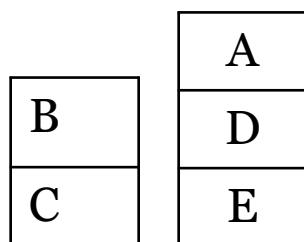
Goal

Rule: Heuristic Function

$h_1(n) =$ add 1 if the block is on the correct location
subtract 1 if the block is on the incorrect location

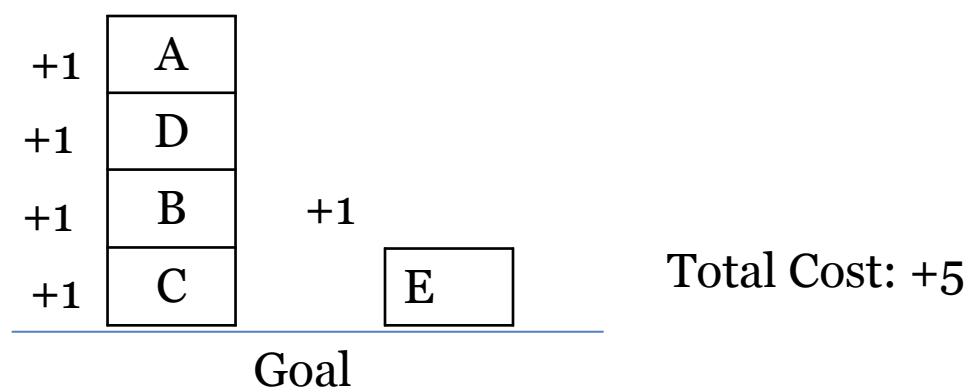
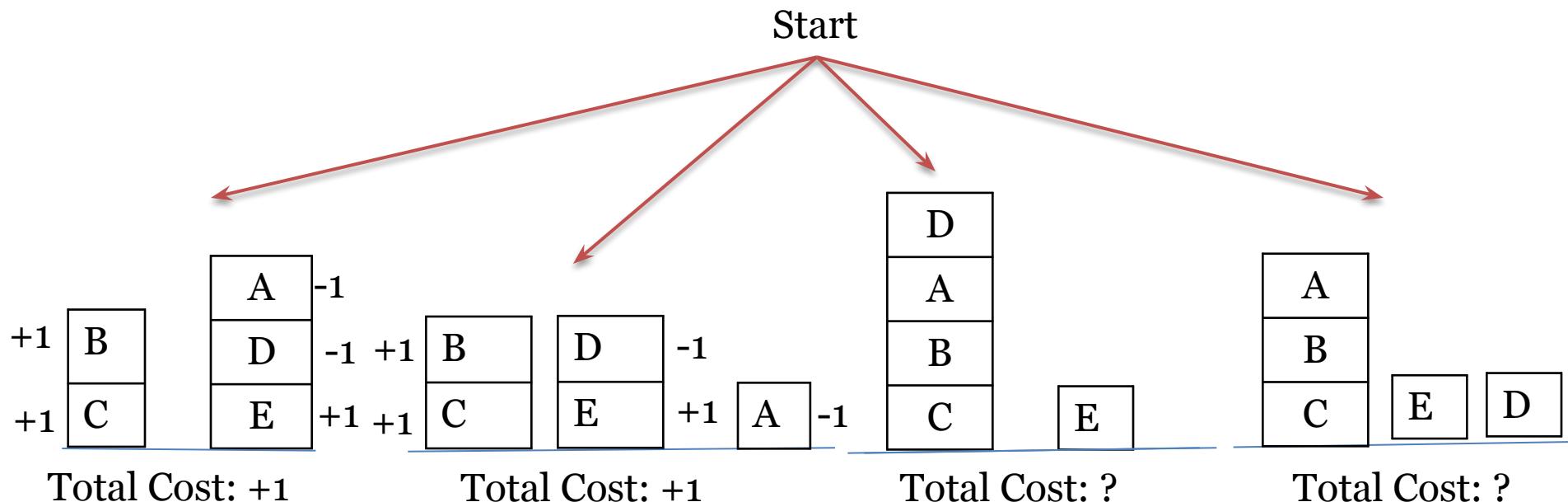
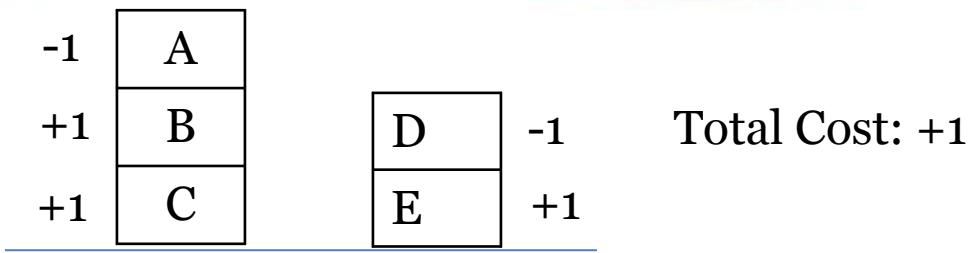


Start

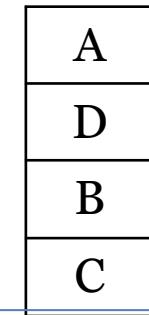
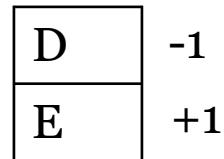
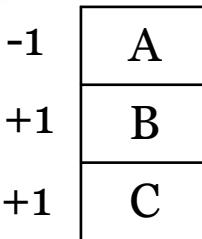


Goal

Total Cost: +5

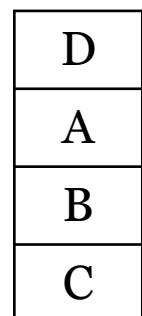
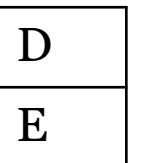
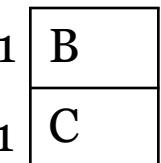
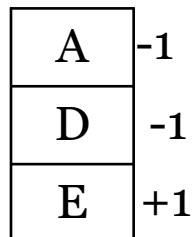
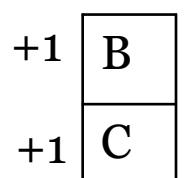
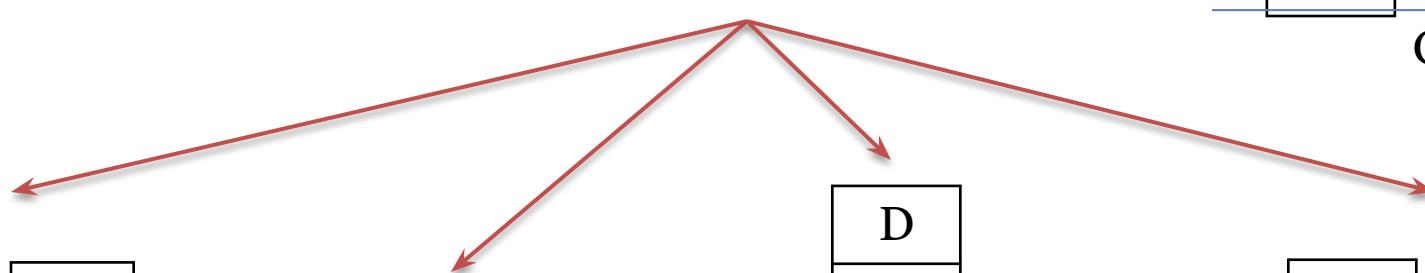


Total Cost: +1

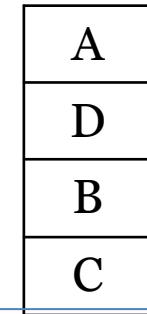
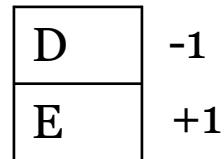
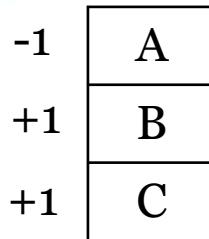


Start

Goal



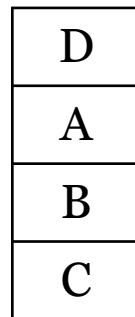
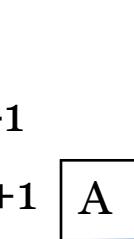
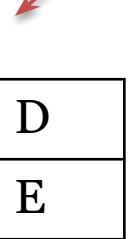
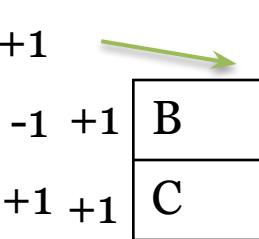
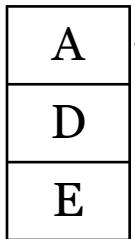
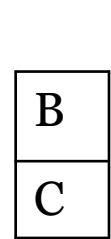
Total Cost: +1



Start

Goal

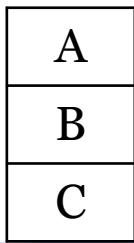
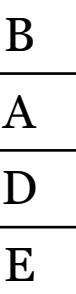
Total Cost: +1



Total Cost: +1

Total Cost: +1

Total Cost: +1

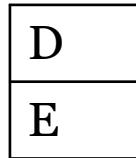
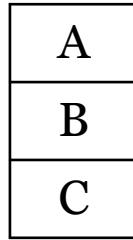


Total Cost: +1

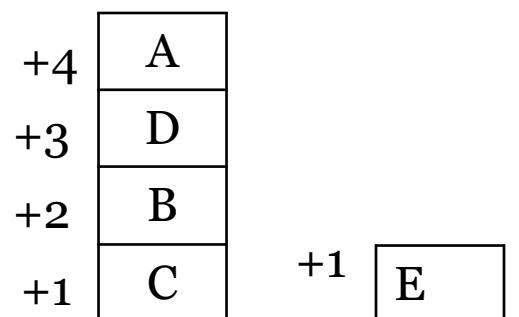
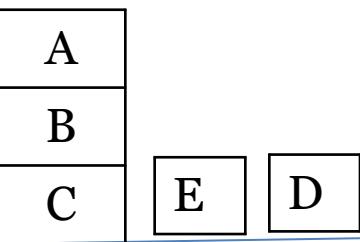
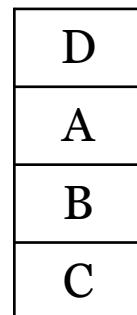
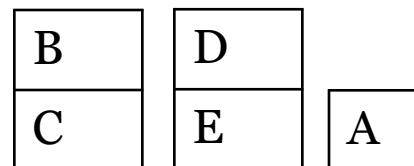
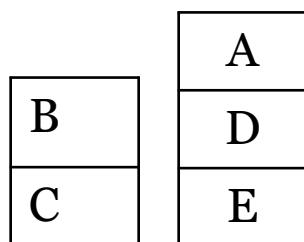
Total Cost: +1

Rule

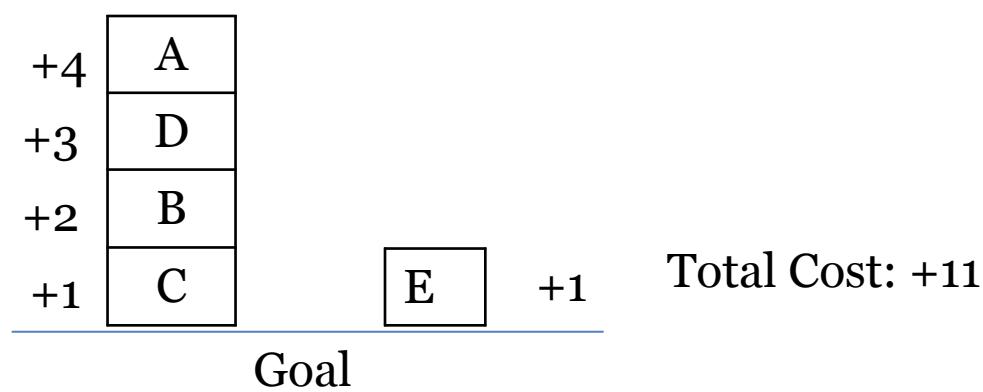
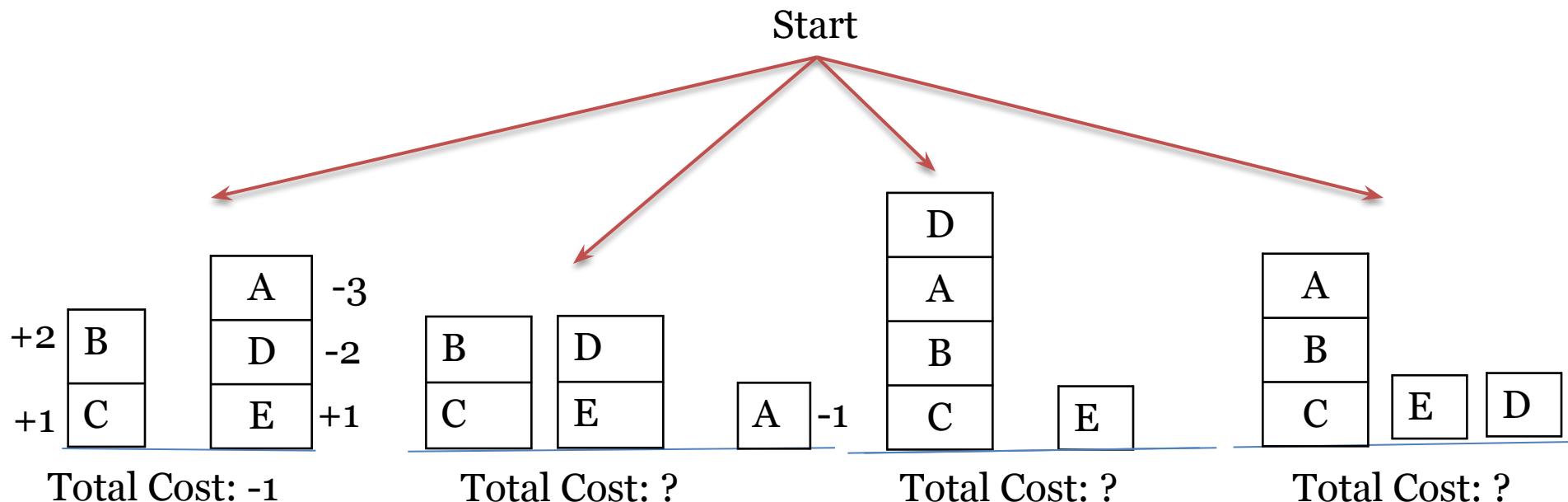
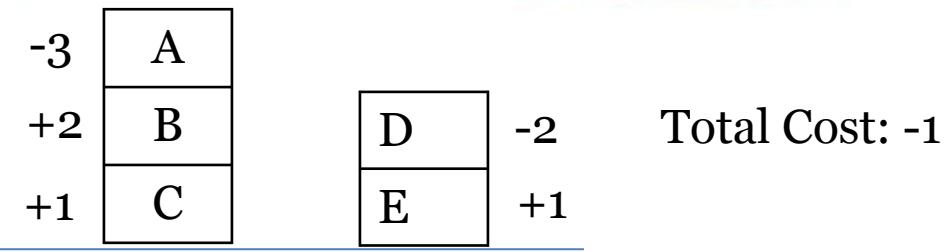
$h_2(n)$ = add 1 for every block in the correct structure that the block is sitting on
subtract 1 for every block in the incorrect structure

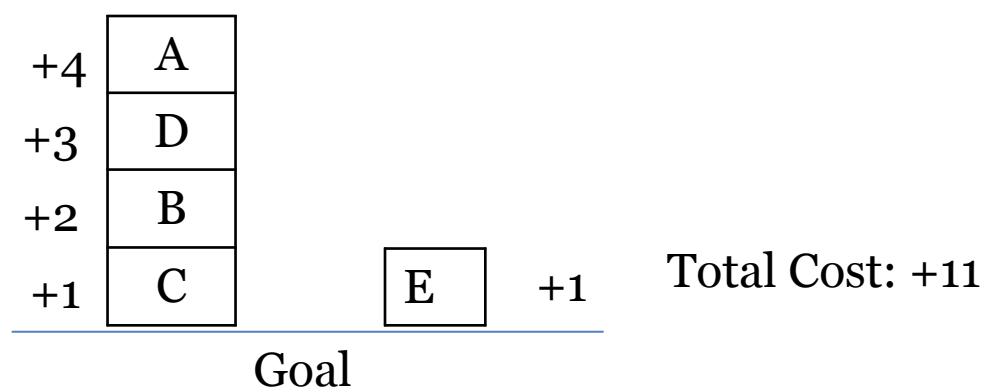
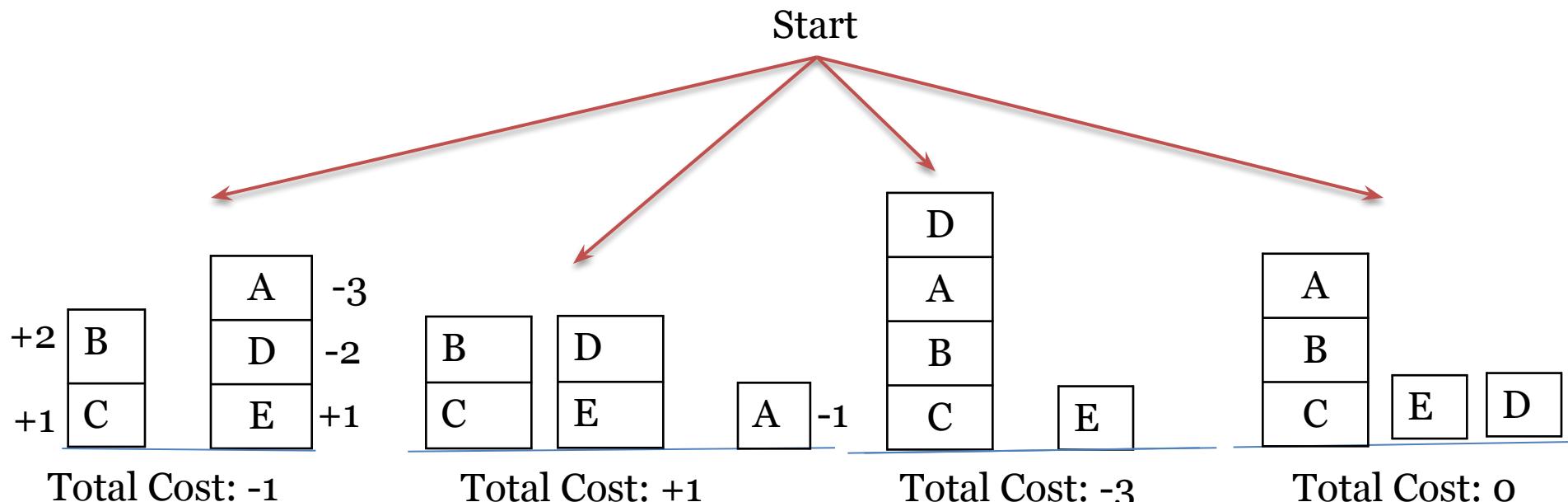
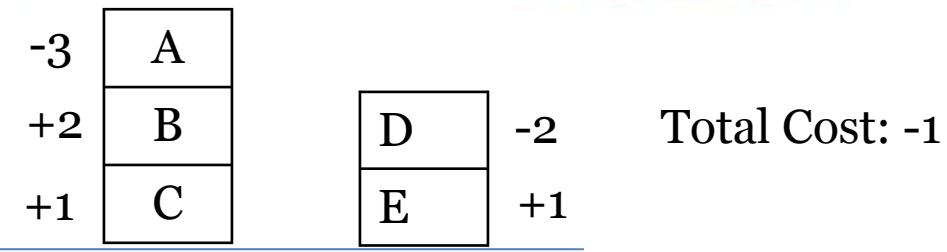


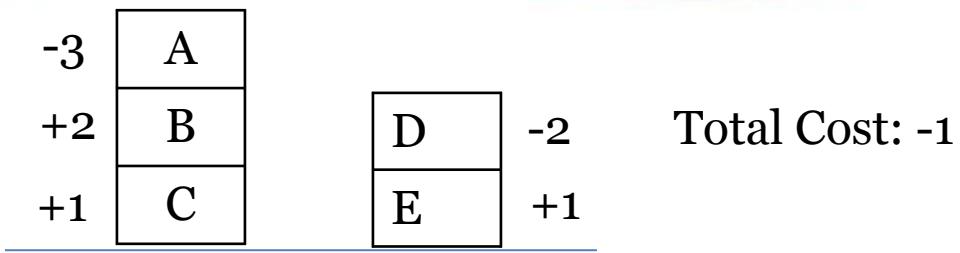
Start



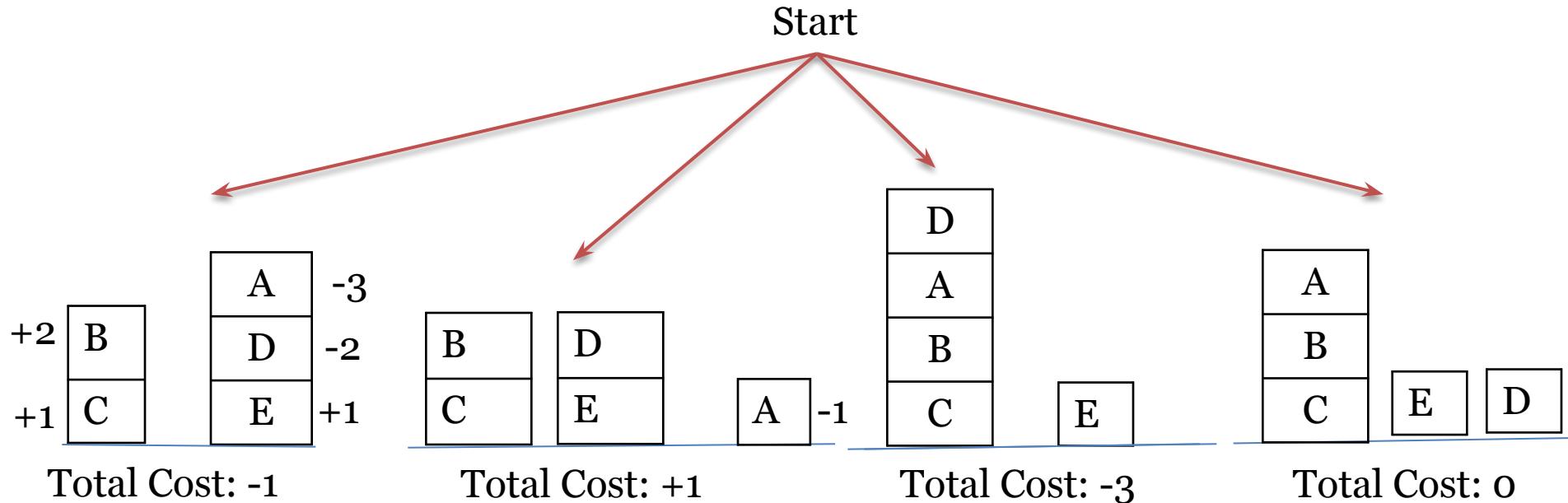
Goal

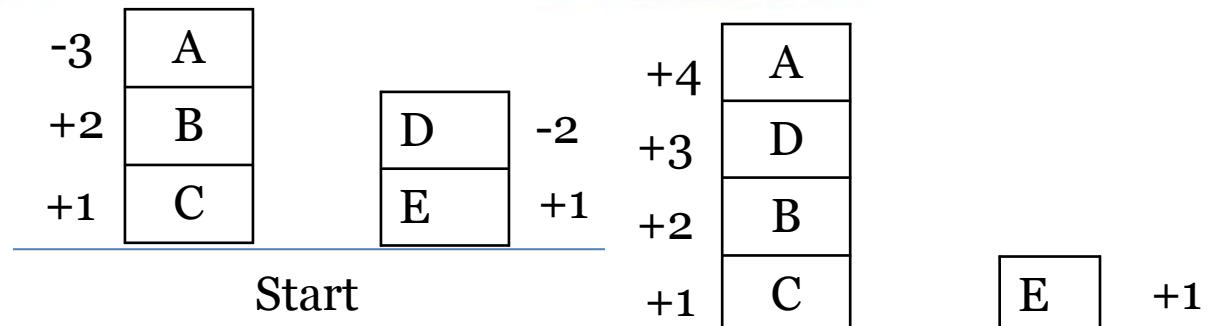




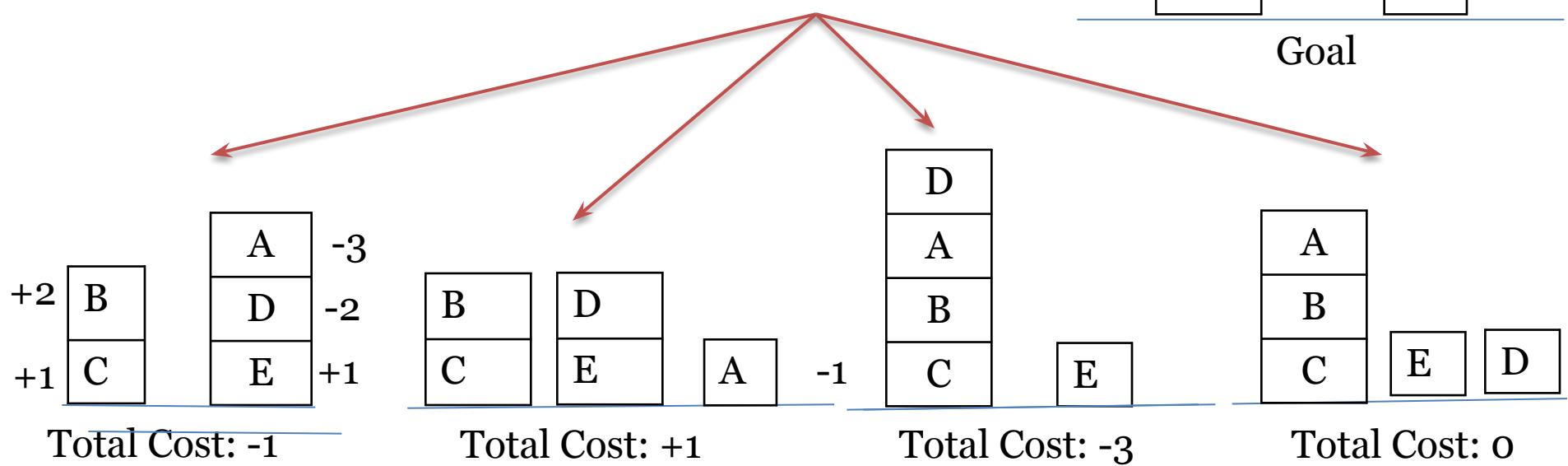
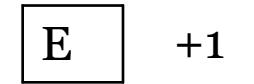


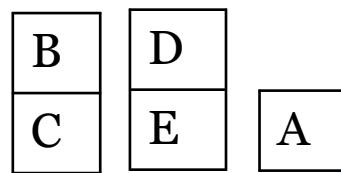
Start



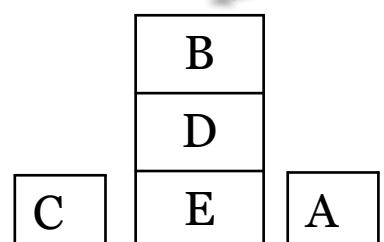


Goal

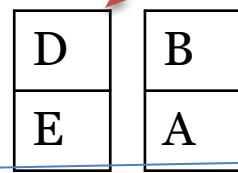




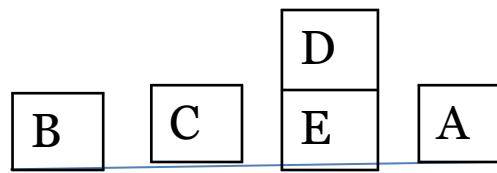
Total Cost: +1



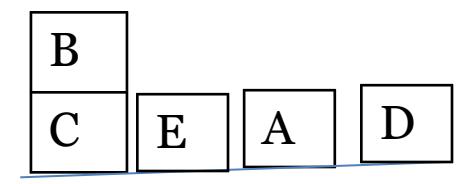
Total Cost: ?



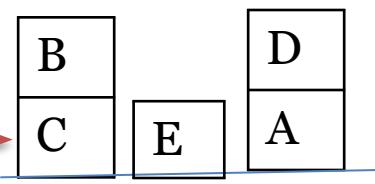
Total Cost: ?



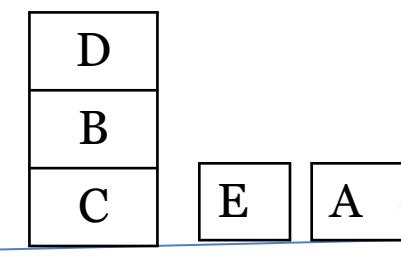
Total Cost: ?



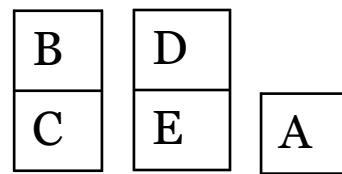
Total Cost: ?



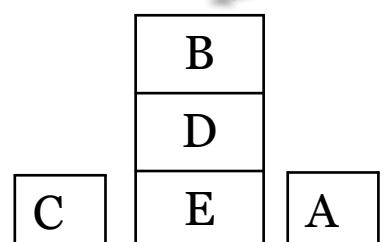
Total Cost: ?



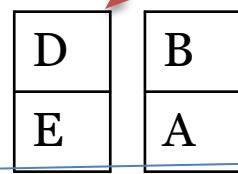
Total Cost: ?



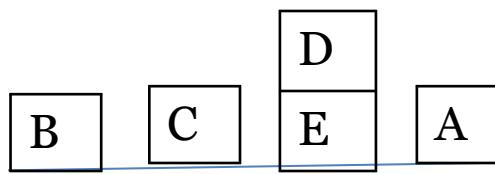
Total Cost: +1



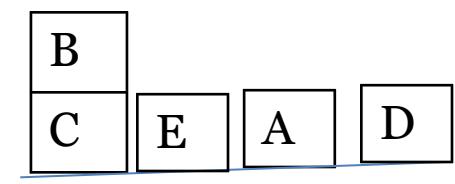
Total Cost: -4



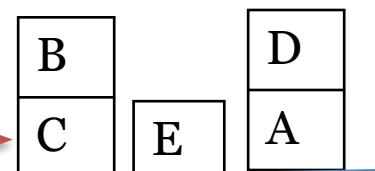
Total Cost: -3



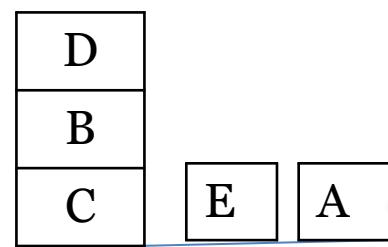
Total Cost: -2



Total Cost: +2

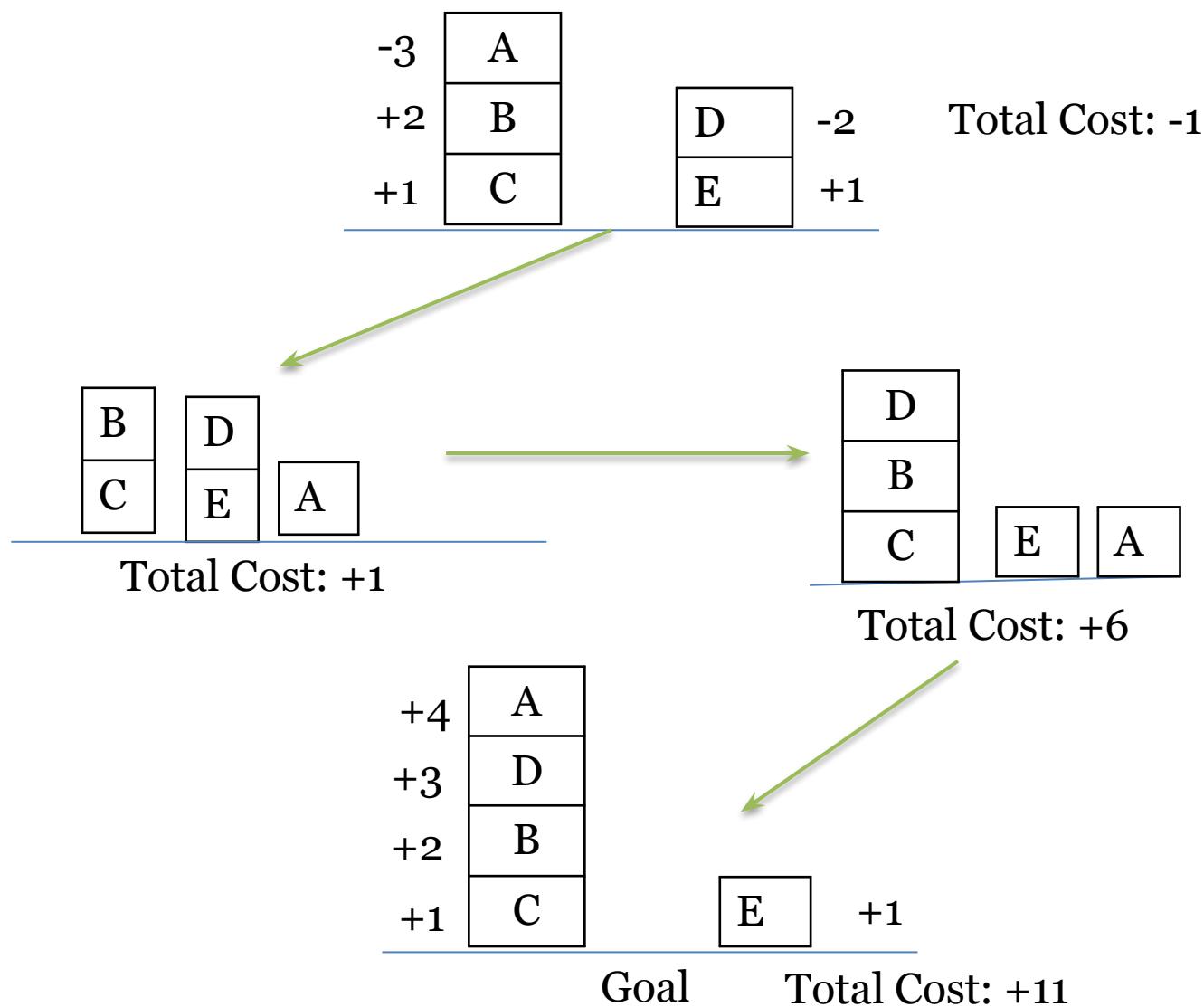


Total Cost: +1

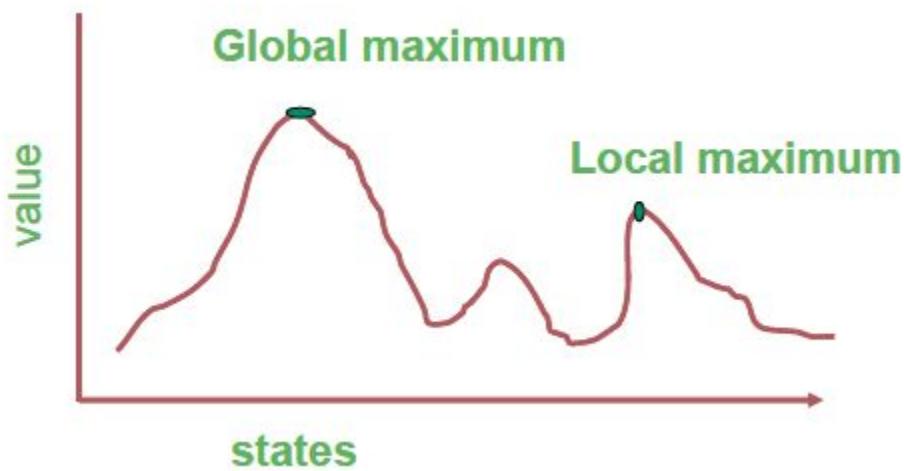


Total Cost: +6

Final Solution



Local Maxima



Hill Climbing (Cont...)

Local maxima

Once the top of a hill is reached the algorithm will halt since every possible step leads down.

Plateaux

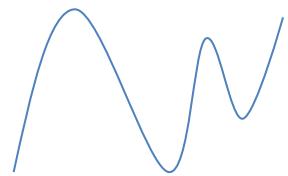
If the landscape is flat, meaning many states have the same goodness, algorithm degenerates to a random walk.

Ridges

If the landscape contains ridges, local improvements may follow a zigzag path up the ridge, slowing down the search.

Surface

Surface
h1: Local Maxima



Surface
h2: Global Maxima



More Answers for Practice in Logic and HW 1

This is an expanded version showing additional right and wrong answers.

I. Practice in 1st-order predicate logic – with answers.

1. Mary loves everyone. [assuming D contains only humans]
 $\forall x \text{ love (Mary, } x)$
Note: No further parentheses are needed here, and according to the syntax on the handout, no further parentheses are possible. But “extra parentheses” are in general considered acceptable, and if you find them helpful, I have no objection. So I would also count as correct any of the following:
 $\forall x (\text{love (Mary, } x)), (\forall x \text{ love (Mary, } x)), (\forall x (\text{love (Mary, } x)))$

2. Mary loves everyone. [assuming D contains both humans and non-humans, so we need to be explicit about ‘everyone’ as ‘every person’]
 $\forall x (\text{person}(x) \rightarrow \text{love (Mary, } x))$
A wrong answer: $\forall x (\text{person}(x) \& \text{ love (Mary, } x))$ This says that everything in the universe is a person and loves Mary.

3. No one talks. [assume D contains only humans unless specified otherwise.]
 $\neg \exists x \text{ talk}(x)$ or equivalently, $\forall x \neg \text{talk}(x)$

4. Everyone loves himself.
 $\forall x \text{ love (} x, x)$

5. Everyone loves everyone.
 $\forall x \forall y \text{ love (} x, y)$

6. Everyone loves everyone except himself. (= Everyone loves everyone else.)
 $\forall x \forall y (\neg x = y \rightarrow \text{love (} x, y))$ or $\forall x \forall y (x \neq y \rightarrow \text{love (} x, y))$
Or maybe it should be this, which is not equivalent to the pair above:
 $\forall x \forall y (\neg x = y \leftrightarrow \text{love (} x, y))$ or $\forall x \forall y (x \neq y \leftrightarrow \text{love (} x, y))$
The first pair allows an individual to also love himself; the second pair doesn’t.

7. Every student smiles.
 $\forall x (\text{student}(x) \rightarrow \text{smile(} x))$

8. Every student except George smiles.
 $\forall x ((\text{student}(x) \& x \neq \text{George}) \rightarrow \text{smile(} x))$
That formula allows the possibility that George smiles too; if we want to exclude it (this depends on what you believe about *except*; there are subtle differences and perhaps some indeterminacy among *except*, *besides*, *other than* and their nearest equivalents in other languages), then it should be the following, or something equivalent to it:
 $\forall x ((\text{student}(x) \rightarrow (x \neq \text{George} \leftrightarrow \text{smile(} x)))$

9. Everyone walks or talks.
 $\forall x (\text{walk (} x) \vee \text{talk (} x))$

10. Every student walks or talks.

- $\forall x (\text{student}(x) \rightarrow (\text{walk}(x) \vee \text{talk}(x)))$
11. Every student who walks talks.
 $\forall x ((\text{student}(x) \& \text{walk}(x)) \rightarrow \text{talk}(x))$ or
 $\forall x (\text{student}(x) \rightarrow (\text{walk}(x) \rightarrow \text{talk}(x)))$
12. Every student who loves Mary is happy.
 $\forall x ((\text{student}(x) \& \text{love}(x, \text{Mary})) \rightarrow \text{happy}(x))$
13. Every boy who loves Mary hates every boy who Mary loves.
 $\forall x((\text{boy}(x) \& \text{love}(x, \text{Mary})) \rightarrow \forall y((\text{boy}(y) \& \text{love}(\text{Mary}, y)) \rightarrow \text{hate}(x, y)))$
14. Every boy who loves Mary hates every other boy who Mary loves.
(So if John loves Mary and Mary loves John, sentence 13 requires that John hates himself, but sentence 14 doesn't require that.)
 $\forall x((\text{boy}(x) \& \text{love}(x, \text{Mary})) \rightarrow \forall y((\text{boy}(y) \& \text{love}(\text{Mary}, y) \& y \neq x) \rightarrow \text{hate}(x, y)))$

II. Homework #1, with answers.

1. Everyone loves Mary.

$$\forall x \text{ love}(x, \text{Mary})$$

2. John does not love anyone. (Not ambiguous, but there are two equivalent and equally good formulas for it, one involving negation and the existential quantifier, the other involving negation and the universal quantifier. Give both.)

$$\neg \exists x \text{ love}(\text{John}, x) \quad \text{or equivalently, } \forall x \neg \text{ love}(\text{John}, x)$$

Wrong: $\exists x \neg \text{ love}(\text{John}, x)$: That says there is someone John doesn't love.

Wrong: $\neg \forall x \text{ love}(\text{John}, x)$: That says John doesn't love everyone; it's equivalent to the preceding formula.

3. Everyone who sees Mary loves Mary.

$$\forall x (\text{see}(x, \text{Mary}) \rightarrow \text{love}(x, \text{Mary}))$$

4. Everyone loves someone. (Ambiguous)

- (i) $\forall x \exists y \text{ love}(x, y)$ (For every person x, there is someone whom x loves.)
- (ii) $\exists y \forall x \text{ love}(x, y)$ (There is some person y whom everyone loves, i.e. everyone loves some one specific person.)

5. Someone loves everyone. (Ambiguous)

- (i) $\exists x \forall y \text{ love}(x, y)$ (There is some person x who loves everyone.)
- (ii) $\forall y \exists x \text{ love}(x, y)$ (For every person y, there is someone who loves them – i.e., no one is totally unloved. This second reading is probably dispreferred for the active sentence. It's the preferred reading for the passive sentence "Everyone is loved by someone" and it's the only reading for the agentless passive "Everyone is loved.")

6. Someone walks and talks.

$\exists x(\text{walk}(x) \& \text{talk}(x))$

7. Someone walks and someone talks.

 $(\exists x \text{ walk}(x) \& \exists x \text{ talk}(x)) \text{ or } (\exists x \text{ walk}(x) \& \exists y \text{ talk}(y))$

Because neither quantifier is inside the scope of the other – i.e. their scopes are independent – it doesn't matter whether we use different variables here or use the same variable twice. But if one quantifier is inside the scope of the other, then it matters a great deal. When one quantifier is inside the scope of another, as in questions 4 and 5 above, always give them different variables!

Also equivalent: $\exists x \exists y(\text{walk}(x) \& \text{talk}(y))$

8. Everyone who walks is calm.

 $\forall x (\text{walk}(x) \rightarrow \text{calm}(x))$

9. No one who runs walks. (Not ambiguous, but same note as for number 2.)

(i) $\neg \exists x (\text{run}(x) \& \text{walk}(x))$ or equivalently,

(ii) $\forall x (\text{run}(x) \rightarrow \neg \text{walk}(x))$ or equivalently,

(iii) $\forall x \neg (\text{run}(x) \& \text{walk}(x))$

A wrong answer: $\forall x (\neg \text{run}(x) \rightarrow \text{walk}(x))$ What does this one say?

Another wrong answer: $\neg \exists x (\text{run}(x) \rightarrow \text{walk}(x))$ This one doesn't correspond to any English sentence; see notes to questions 11 and 6' below.

10. Everyone who Mary loves loves someone who is happy.

 $\forall x (\text{love}(\text{Mary}, x) \rightarrow \exists y (\text{love}(x, y) \& \text{happy}(y)))$

Also correct: $\forall x \exists y (\text{love}(\text{Mary}, x) \rightarrow (\text{love}(x, y) \& \text{happy}(y)))$

But I recommend keeping each quantifier as close as possible to the noun it quantifies, or to its surface position. The more you move quantifiers around, the easier it is to make mistakes.

11. If anyone cheats, he suffers. (English paraphrases: Anyone who cheats suffers. Everyone who cheats suffers. On the subtle difference between these two, see (Kadmon and Landman 1993).)

 $\forall x (\text{cheat}(x) \rightarrow \text{suffer}(x))$

A wrong answer: $\exists x (\text{cheat}(x) \rightarrow \text{suffer}(x))$ A wide scope $\exists x$ like this creates too weak a statement. If $\exists x$ were given scope only over the antecedent, as in: $\exists x \text{cheat}(x) \rightarrow \text{suffer}(x)$, then that error would be corrected but there would be a new problem because the second x would not be bound.

Note on any: Sometimes *anyone* corresponds to \exists and sometimes to \forall ; you have to think about the meaning of the whole sentence. Many papers have been written exploring the issue of how best to account for the distribution of meanings of *any*, and whether it does or doesn't require lexical ambiguity as part of the account. A few classics include (Carlson 1980, Carlson 1981, Haspelmath 1997, Hintikka 1980, Kadmon and Landman 1993, Kratzer and Shimoyama 2002, Ladusaw 1980, Linebarger 1987, Vendler 1962). See also the note about *any* in the next item.

12. If anyone cheats, everyone suffers.

$$\forall x (\text{cheat}(x) \rightarrow \forall y \text{ suffer}(y))$$

Equivalent: $\forall x \forall y (\text{cheat}(x) \rightarrow \text{ suffer}(y))$

Also equivalent: $\forall y \forall x (\text{cheat}(x) \rightarrow \text{ suffer}(y))$

Also equivalent: $\exists x \text{ cheat}(x) \rightarrow \forall y \text{ suffer}(y)$ (Each quantifier has narrow scope here.)

Also equivalent: $\exists x \text{ cheat}(x) \rightarrow \forall x \text{ suffer}(x)$ (If each quantifier has narrow scope, then they don't need to involve different variables. If one is inside the scope of the other, then they do.)

Also equivalent: $\forall y (\exists x \text{ cheat}(x) \rightarrow \text{ suffer}(y))$

A wrong answer: $\forall y \exists x (\text{cheat}(x) \rightarrow \text{ suffer}(y))$ This has no natural English paraphrase.

A different wrong answer: $\forall y (\forall x \text{ cheat}(x) \rightarrow \text{ suffer}(y))$ This is one way of saying "If everyone cheats, then everyone suffers."

Another note about any: As the equivalent answers above illustrate, *any* in this case can be viewed either as a wide-scope universal (with scope over the *if*-clause) or as a narrow-scope existential (with scope inside the *if*-clause). The fact that these are equivalent, at least in this case, is part of the source of debates about *any*. In example 11, we didn't have that choice, because if *any* were treated as a narrow-scope existential in that case, it couldn't bind the second occurrence of the variable *x* corresponding to the pronoun *he*. The same is true for *anyone* in the next example, which has to be treated as a wide-scope universal in order to bind *himself*.

13. Anyone who loves everyone loves himself.

$$\forall x (\forall y \text{ love}(x,y) \rightarrow \text{love}(x,x))$$

note: Not this: $\forall x \forall y (\text{love}(x,y) \rightarrow \text{love}(x,x))$ What this one says is "Anyone who loves anyone loves himself" What the correct one says is IF you love everyone, THEN you love yourself. So the $\forall y$ quantifier has to be inside the scope of the \rightarrow .

Another wrong answer: $\exists x \forall y (\text{love}(x,y) \rightarrow \text{love}(x,x))$ This has no natural English paraphrase. *Any* may sometimes be a wide-scope universal, and sometimes a narrow-scope existential, but it is never a "wide-scope existential."

14. Mary loves everyone except John. (For this one, you need to add the two-place predicate of identity, " $=$ ". Think of "everyone except John" as "everyone who is not identical to John".)

$$\forall x (\neg x = \text{John} \rightarrow \text{love}(\text{Mary}, x)) \text{ or equivalently}$$

$$\forall x (x \neq \text{John} \rightarrow \text{love}(\text{Mary}, x))$$

As in the case of some earlier examples, this is a 'weak' reading of *except*, allowing the possibility of Mary loving John. To get a 'strong' reading of *except*, ruling out that possibility, replace \rightarrow above by \leftrightarrow , or add a conjunct " $\& \neg \text{love}(\text{Mary}, \text{John})$ " at the end.

15. Redo the translations of sentences 1, 4, 6, and 7, making use of the predicate **person**, as we would have to do if the domain D contains not only humans but cats, robots, and other entities.

1'. Everyone loves Mary.

$$\forall x (\text{person}(x) \rightarrow \text{love}(x, \text{Mary}))$$

4'. Everyone loves someone. (Ambiguous)

(i) $\forall x (\text{person}(x) \rightarrow \exists y (\text{person}(y) \& \text{love}(x, y)))$ (For every person x, there is some person y whom x loves.)

(ii) $\exists y (\text{person}(y) \& \forall x (\text{person}(x) \rightarrow \text{love}(x, y)))$ (There is some person y whom every person x loves.)

An equivalent correct answer for (i): $\forall x \exists y (\text{person}(x) \rightarrow (\text{person}(y) \& \text{love}(x, y)))$

But I don't recommend moving the second quantifier, because then it's too easy to come up with the following **wrong** answer for (i): $\forall x \exists y ((\text{person}(x) \& \text{person}(y)) \rightarrow \text{love}(x, y))$. It's always safer to keep a quantifier and its "restrictor" (in this case *person*) as close together as possible, and both of them as close to their surface position as possible.

6'. Someone walks and talks.

$$\exists x (\text{person}(x) \& \text{walk}(x) \& \text{talk}(x))$$

Note: technically, we need more parentheses – either

$$\exists x (\text{person}(x) \& (\text{walk}(x) \& \text{talk}(x))) \text{ or}$$

$$\exists x ((\text{person}(x) \& \text{walk}(x)) \& \text{talk}(x))$$

But since it's provable that $\&$ is associative, i.e. the grouping of a sequence of $\&$'s doesn't make any difference, it is customary to allow expressions like $(p \& q \& r)$. And similarly for big disjunctions, $(p \vee q \vee r)$. But not with \rightarrow !

Wrong: $\exists x (\text{person}(x) \rightarrow (\text{walk}(x) \& \text{talk}(x)))$ This has weird truth-conditions, which you can see if you remember that $p \rightarrow q$ is equivalent to $\neg p \vee q$. You will never really want to combine \exists with \rightarrow -- it always makes a statement that is too weak.

7'. Someone walks and someone talks.

$$(\exists x (\text{person}(x) \& \text{walk}(x)) \& \exists y (\text{person}(y) \& \text{talk}(y))) \text{ or equivalently} \\ (\exists x (\text{person}(x) \& \text{walk}(x)) \& \exists y (\text{person}(y) \& \text{talk}(y)))$$

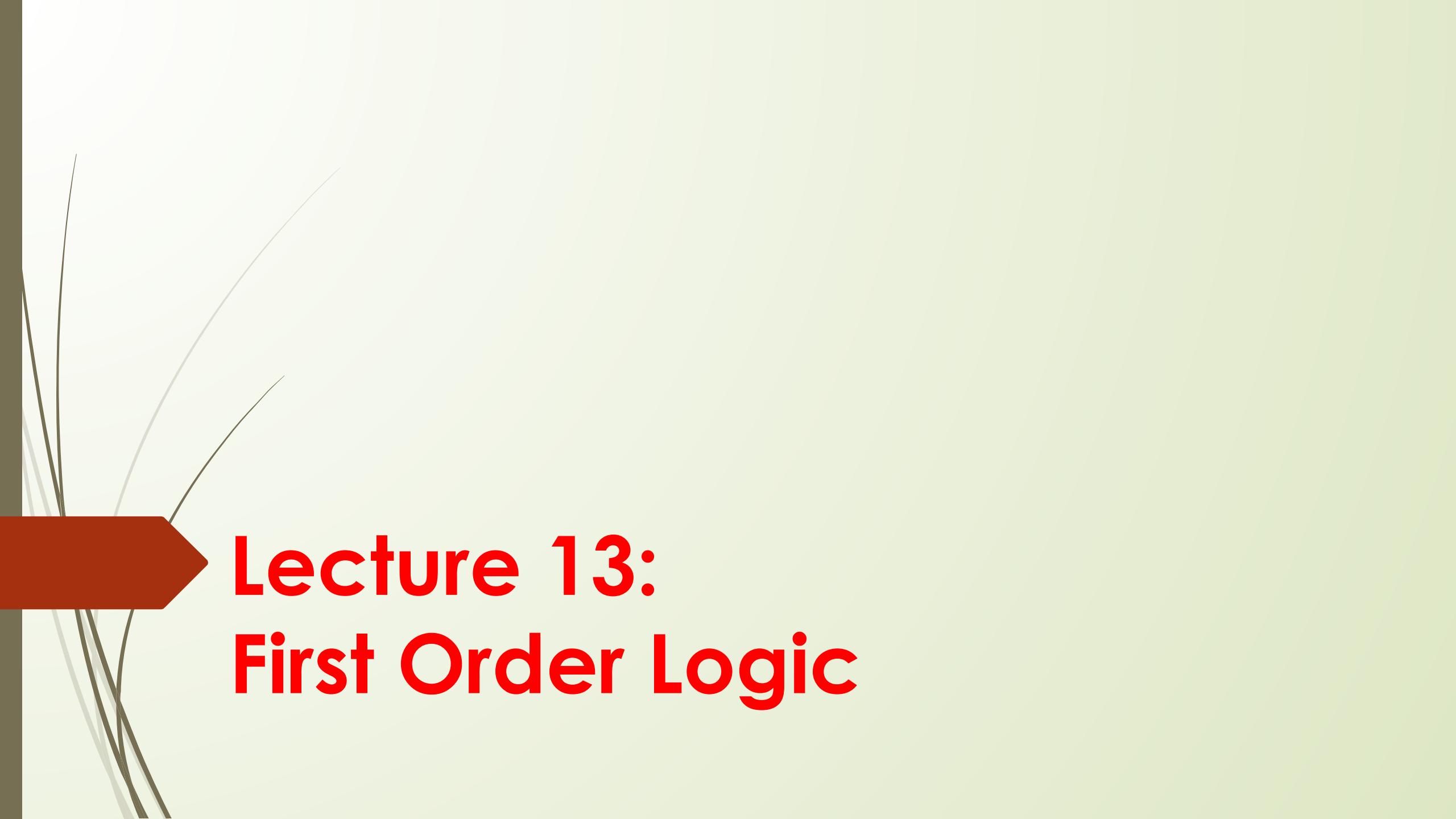
Note: both in the original 7 and in this 7', it would be OK and customary to drop outermost parentheses, i.e. the very first left parenthesis and the very last right parenthesis may be dropped. (But no parentheses can be dropped in 6; they are not really "outermost". Only when a pair of parentheses contains the entire formula can it be dropped under the "drop outermost parentheses" convention.)

Also correct: $\exists x \exists y (\text{person}(x) \& \text{walk}(x) \& \text{person}(y) \& \text{talk}(y))$

References

Carlson, Greg. 1980. Polarity *Any* is Existential. *Linguistic Inquiry* 11:799-804.

- Carlson, Greg. 1981. Distribution of free-choice 'any'. In *Chicago Linguistic Society 17*, 8-23. Chicago.
- Haspelmath, Martin. 1997. *Indefinite Pronouns*. Oxford: Oxford University Press.
- Hintikka, Jaakko. 1980. On the "Any"-Thesis and the Methodology of Linguistics. *Linguistics and Philosophy* 4:101-122.
- Kadmon, Nirit, and Landman, Fred. 1993. *Any*. *Linguistics & Philosophy* 16:353-422.
- Kratzer, Angelika, and Shimoyama, Junko. 2002. Indeterminate pronouns: the view from Japanese. In *The Proceedings of the Third Tokyo Conference on Psycholinguistics*, ed. Yukio Otsu, 1-25. Tokyo: Hituzi Syobo.
- Ladusaw, William. 1980. On the notion "affective" in the analysis of negative polarity items. *Journal of Linguistic Research* 1:1-16. Reprinted in Portner and Partee (2002), pp. 457-470.
- Linebarger, Marcia. 1987. Negative polarity and grammatical representation. *Linguistics and Philosophy* 10:325-387.
- Vendler, Zeno. 1962. *Each and Every, Any and All*. *Mind* 71:145-160.



Lecture 13: **First Order Logic**

Pros and cons of propositional logic

- ☺ Propositional logic is **declarative**
- ☺ Propositional logic allows partial/disjunctive/negated information
 - (unlike most data structures and databases)
- ☺ Propositional logic is **compositional**:
 - meaning of $B_{1,1} \wedge P_{1,2}$ is derived from meaning of $B_{1,1}$ and of $P_{1,2}$
- ☺ Meaning in propositional logic is **context-independent**
 - (unlike natural language, where meaning depends on context)
- :(Propositional logic has very limited expressive power
 - (unlike natural language)
 - E.g., cannot say "pits cause breezes in adjacent squares"
 - except by writing one sentence for each square

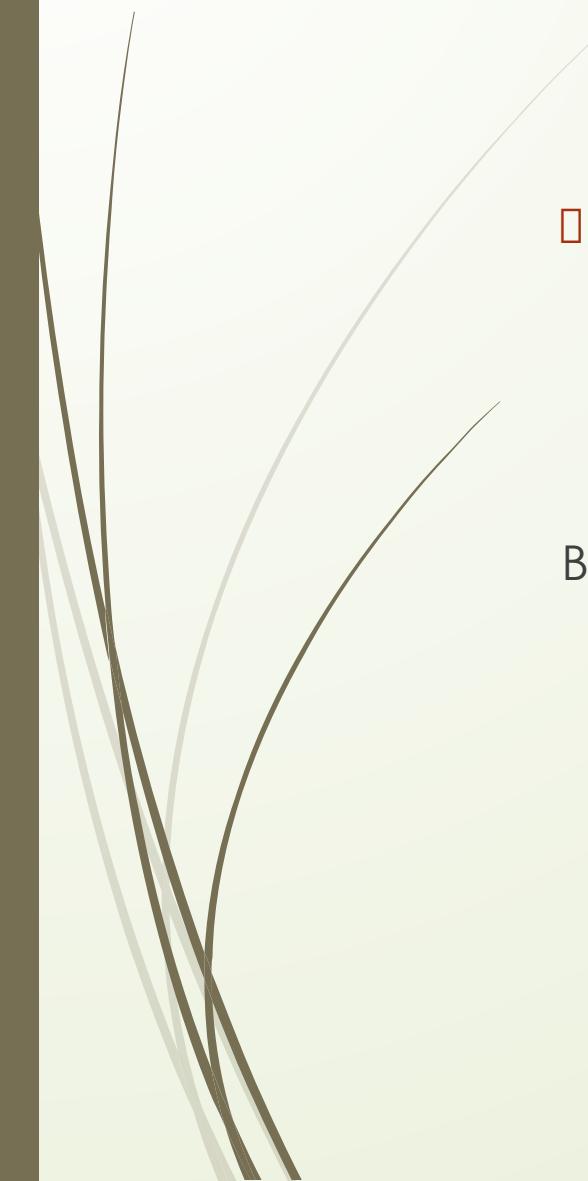
First-order logic

- Whereas propositional logic assumes the world contains **facts**,
- first-order logic (like natural language) assumes the world contains
 - **Objects**: people, houses, numbers, colors, baseball games, wars, ...
 - **Relations**: red, round, prime, brother of, bigger than, part of, comes between, ...
 - **Functions**: father of, best friend, one more than, plus, ...

Limitations of propositional logic



- Propositional logic has limited expressive power
 - unlike natural language
 - E.g., cannot say "pits cause breezes in adjacent squares"
 - except by writing one sentence for each square



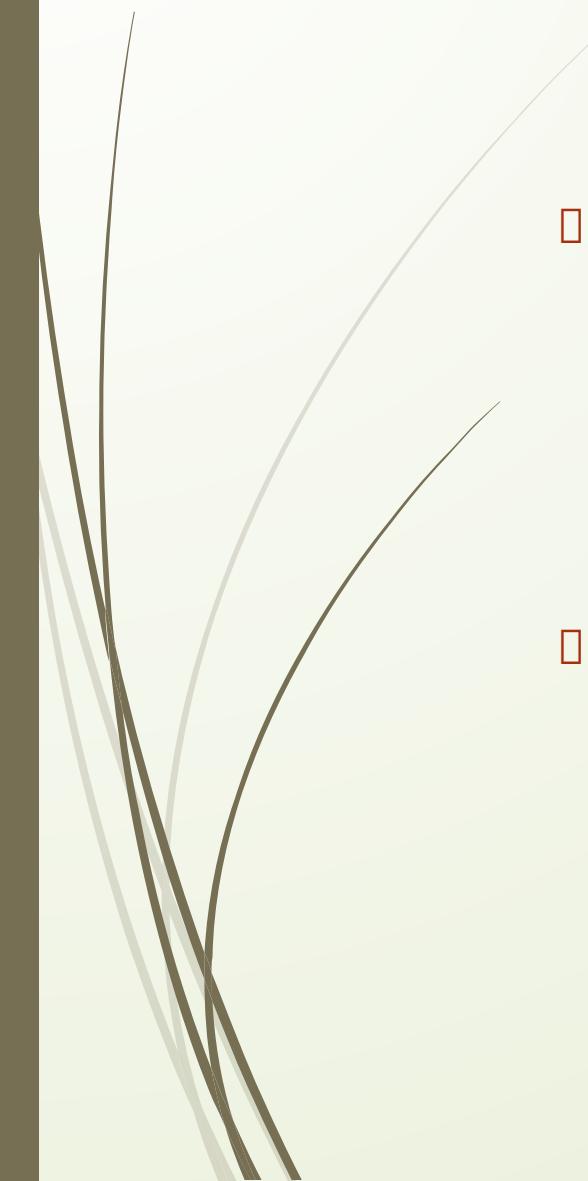
□ For Example

Every Lion drinks water

Bunny is a lion

Brain can concludes:

Bunny drinks water



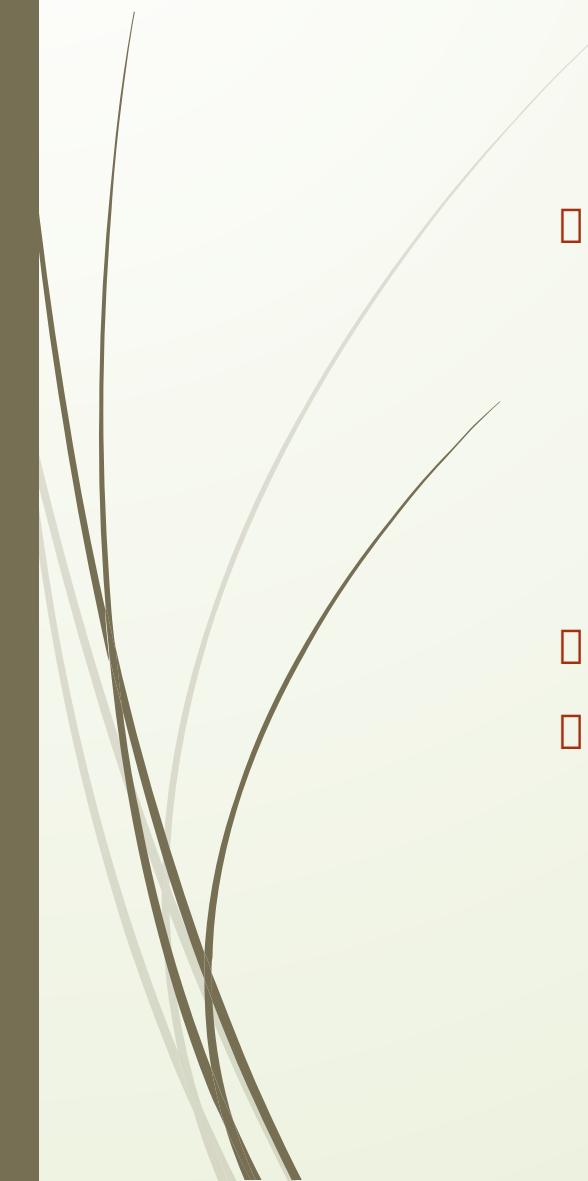
□ For Propositional Logic

P Every Lion drinks water

Q Bunny is a lion

R Bunny drinks water

□ But you can't go inside P & Q statement so by PL you can't conclude.



□ For Propositional Logic

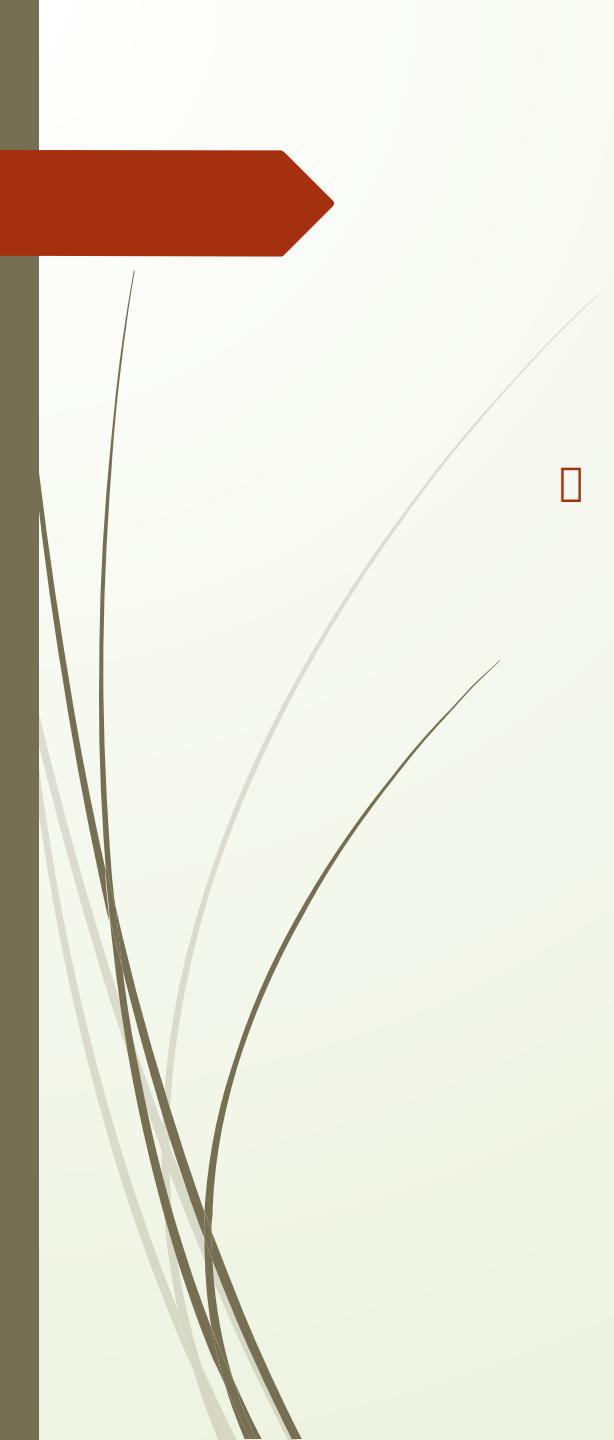
P Every Lion drinks water

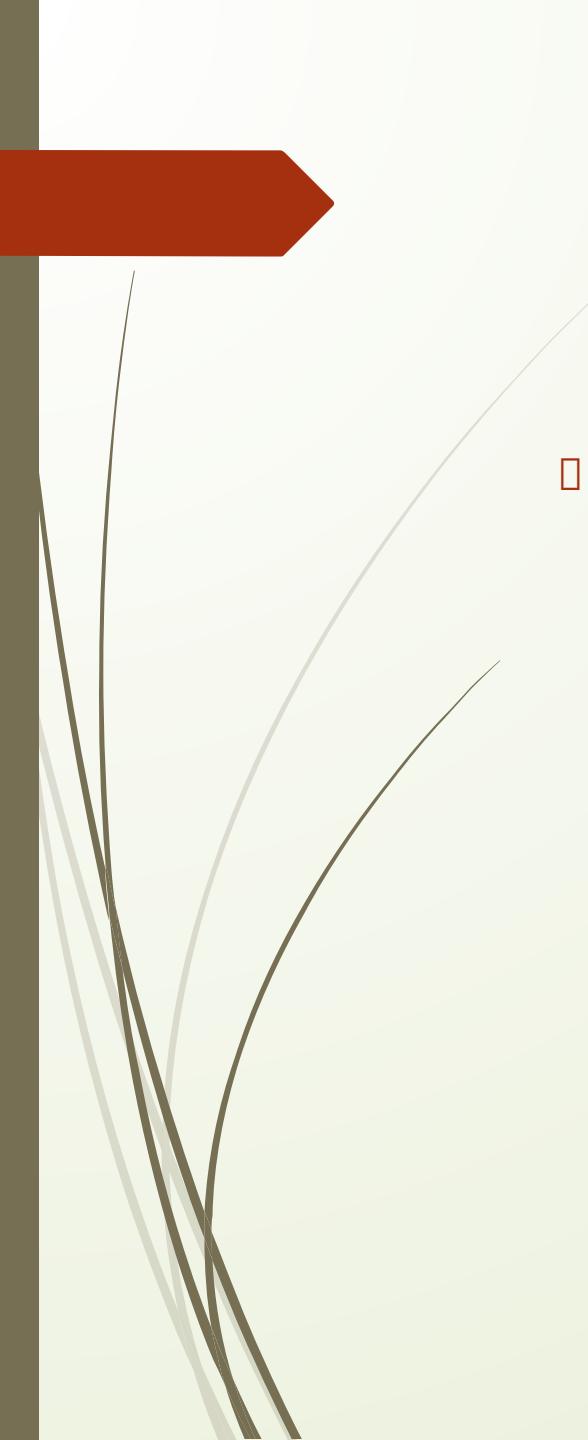
Q Bunny is a lion

R Bunny drinks water

□ you can't go inside P & Q statement so by PL you can't conclude.

□ **But You can solve by First Order Logic**

- 
- Every FOL is divided by two parts
 - Subject
 - Predicate

- 
- Every FOL is divided by two parts

- Subject

- Predicate

X is an integer.

Subject: x

Predicate: is an integer

Pinky is a cat.

Subject: Pinky

Predicate: is a cat.

Short hand notation

Pinky is a cat.

Subject: Pinky

Predicate: is a cat.

$\text{cat}(x) = x \text{ is a cat}$

$\text{cat}(\text{Pinky})$

$\text{Int}(x) = x \text{ is an integer}$

Every man drinks coffee

Universe of Discussion/
Domain of Discussion

man

x_1

x_2

x_3

x

$\forall x \text{ coffee}(x)$

All statement must true

$x_1 \text{ drinks coffee}$

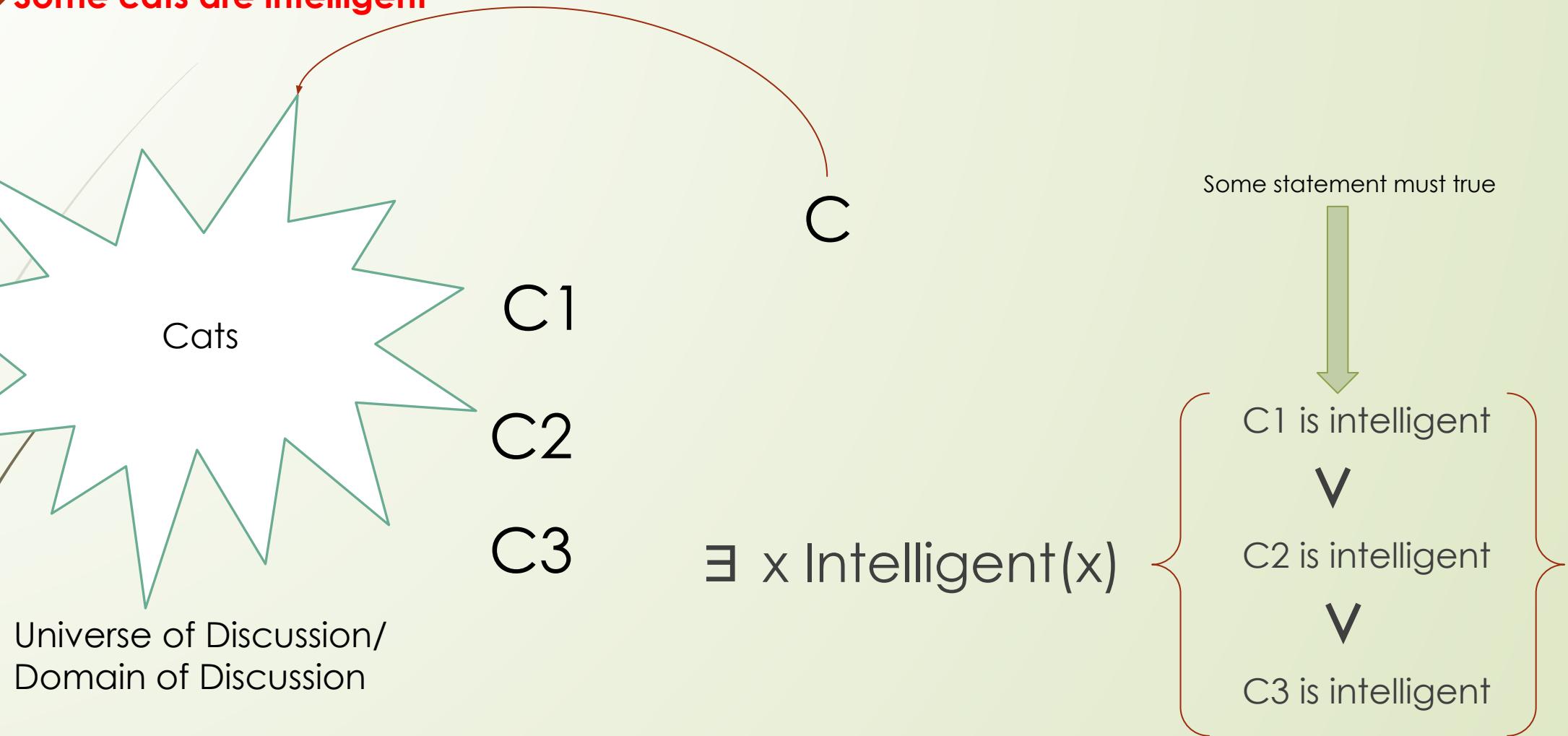
\wedge

$x_2 \text{ drinks coffee}$

\wedge

$x_3 \text{ drinks coffee}$

Some cats are intelligent



First-Order Logic

- Propositional logic assumes that the world contains **facts**.
- First-order logic (like natural language) assumes the world contains
 - **Objects**: people, houses, numbers, colors, baseball games, wars, ...
 - **Relations**: red, round, prime, brother of, bigger than, part of, comes between, ...
 - **Functions**: father of, best friend, one more than, plus, ...

Logics in General

- Ontological Commitment:
 - What exists in the world — TRUTH
 - PL : facts hold or do not hold.
 - FOL : objects with relations between them that hold or do not hold
- Epistemological Commitment:

Language	Ontological Commitment	Epistemological Commitment
Propositional logic	facts	true/false/unknown
First-order logic	facts, objects, relations	true/false/unknown
Temporal logic	facts, objects, relations, times	true/false/unknown
Probability theory	facts	degree of belief $\in [0, 1]$
Fuzzy logic	degree of truth $\in [0, 1]$	known interval value

Syntax of FOL: Basic elements

□ Constant Symbols:

- Stand for objects
- e.g., KingJohn, 2, UCI,...

□ Predicate Symbols:

- Stand for relations
- E.g., Brother(Richard, John), greater_than(3,2)...

□ Function Symbols:

- Stand for functions
- E.g., Sqrt(3), LeftLegOf(John),...

Syntax of FOL: Basic elements

- **Constants** KingJohn, 2, UCI,...
- **Predicates** Brother, >,...
- **Functions** Sqrt, LeftLegOf,...
- **Variables** x, y, a, b,...
- **Connectives** \neg , \Rightarrow , \wedge , \vee , \Leftrightarrow
- **Equality** =
- **Quantifiers** \forall , \exists

Universal Quantification \forall

- \forall means “for all”
- Allows us to make statements about all objects that have certain properties
- Can now state general rules:

$\forall x \text{ King}(x) \rightarrow \text{Person}(x)$ “All kings are person”

$\forall x \text{ Person}(x) \rightarrow \text{HasHead}(x)$ “Every person has a head.”

Note that:

$\forall x \text{King}(x) \wedge \text{Person}(x)$ is not correct!

This would imply that all objects x are Kings and are People/Person

$\forall x \text{King}(x) \rightarrow \text{Person}(x)$ is the correct way to say

Existential Quantification \exists

- $\exists x$ means “there exists an x such that....” (at least one object x)
- Allows us to make statements about some object without naming it
- Examples:

$\exists x \text{ King}(x)$

“Some object is a king.”

$\exists x \text{ Lives_in}(\text{John}, \text{Castle}(x))$

“John lives in somebody's castle.”

$\exists i \text{ Integer}(i) \wedge \text{GreaterThan}(i, 0)$

“Some integer is greater than zero.”

Note that:

\wedge is the natural connective to use with \exists

(And \rightarrow is the natural connective to use with \forall)

Nested Quantifiers

Definition: Two quantifiers are said to be nested if one is within the scope of the other.

For example: $\forall x \exists y Q(x, y)$


 \exists is within the scope of \forall

Note: Anything within a scope of the quantifier can be thought of as a propositional function.

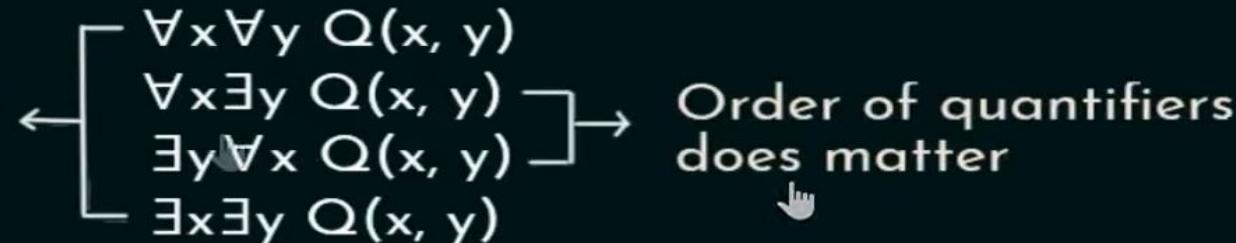
$$\forall x \exists y Q(x, y) \Rightarrow \forall x P(x)$$

\downarrow

$$P(x)$$

Different combinations of Nested Quantifiers

Order of quantifiers
doesn't matter



Order of quantifiers
does matter



Some cats are intelligent

$\exists x[\text{cat}(x) \wedge I(x)]$



Some cats are intelligent

□ Proof that correct or wrong?

$$\exists x[\text{cat}(x) \wedge I(x)]$$

Some cats are intelligent

$$\exists x[\text{cat}(x) \wedge I(x)]$$

(From table: **False**)

Alias	Animal	Intelligent
a1	cat	No
a2	cat	No
a3	dog	Yes

P	Q	$P \rightarrow Q$
T	T	T
T	F	F
F	T	T
F	F	T

$$\exists a1[\text{cat}(a1) \wedge I(a1)] \vee \exists a2[\text{cat}(a2) \wedge I(a2)] \vee \exists a3[\text{cat}(a3) \wedge I(a3)]$$

False

False

False \wedge True

True

True

$\exists x[\text{cat}(x) \wedge I(x)]$
This is true which is
contradict
of the statement



Some cats are intelligent

□ Solution:

$$\exists x[\text{cat}(x) \wedge \text{I}(x)]$$

Can you proof again?

Some cats are intelligent

$$\exists x[\text{cat}(x) \wedge I(x)]$$

(From table: **False**)

Alias	Animal	Intelligent
a1	cat	No
a2	cat	No
a3	dog	Yes

P	Q	$P \rightarrow Q$
T	T	T
T	F	F
F	T	T
F	F	T

$$\exists a[\text{cat}(a) \wedge I(a)] \vee \exists a_2[\text{cat}(a_2) \wedge I(a_2)] \vee \exists a_3[\text{cat}(a_3) \wedge I(a_3)]$$

False

False

False \wedge True

False

False

$\exists x[\text{cat}(x) \wedge I(x)]$
This is false which is
Same as
the statement



Every student in this class has visited Africa or America

- Student(x): x is student in this class
- vaf(x): x has visited Africa
- vam(x): x has visited America

$$\forall x[\text{student}(x) \rightarrow (\text{vaf}(x) \vee \text{vam}(x))]$$

Some prime number is even number

- $\text{prime}(x)$: x is prime no
- $\text{Even}(x) = x$ is even no

$$\exists x [\text{prime}(x) \wedge \text{even}(x)]$$



Raju likes Rani



Likes(Raju, Rani)



“Raju likes Every one”

□ Proof?



“Raju likes Every one”

Raju likes x1

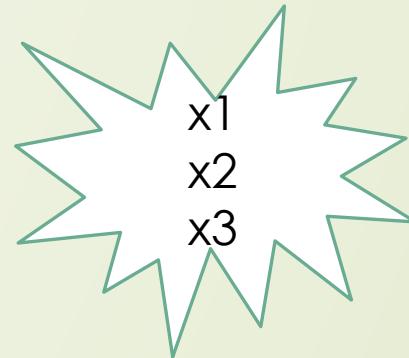


Raju likes x2



Raju likes x3

$\text{Likes}(\text{Raju}, \text{x1}) \wedge \text{Likes}(\text{Raju}, \text{x2}) \wedge \text{Likes}(\text{Raju}, \text{x3})$



$\forall x \text{Likes}(\text{Raju}, x)$



Everyone likes everyone

□ Proof?



Everyone likes everyone

Raju likes everyone

$\forall x \text{Likes}(Raju, x)$

Rani likes everyone

$\forall x \text{Likes}(Rani, x)$

Everyone likes Raju

$\forall y \text{Likes}(y, Raju)$

...

...

...

$\forall y \forall x [\text{Likes}(y, x)]$



Someone likes someone

□ ?



Someone likes someone

Raju likes someone

...

...

$\exists y \text{ likes}(Raju, y)$

$\exists x \exists y \text{ Likes}(x, y)$



Someone likes Everyone

□ ?



Someone likes Everyone

Raju likes Everyone

...

...

$\forall x \text{ likes}(Raju, x)$

$\exists y [\forall x \text{ Likes}(y, x)]$



Everyone likes Someone

□ ?



Everyone likes Someone

Raju likes someone $\exists x \text{ Likes}(Raju, x)$

...

....

$\forall y [\exists x \text{ Likes}(y, x)]$



Everyone is liked by someone

Raju is liked by someone $\exists y \text{ Likes}(y, \text{Raju})$

$\forall x \exists y \text{ Likes}(y, x)$



Someone is liked by everyone

□ ?



Someone is liked by everyone

□ Raju is liked by everyone

$\forall x \text{ Likes}(x, \text{Raju})]$

$\exists y \forall x \text{ Likes}(x, y)]$

Nobody likes everyone

Raju does not like everyone

...

...

...

¬ $\forall x \text{ Likes}(Raju, x)$

....

$\forall y [\neg \forall x \text{ Likes}(y, x)]$

GATE 2009

Which one of the following is the most appropriate logical formula to represent the statement? "Gold and silver ornaments are precious".

The following notations are used:

$G(x)$: x is a gold ornament

$S(x)$: x is a silver ornament

$P(x)$: x is precious

(A) $\forall x(P(x) \rightarrow (G(x) \wedge S(x)))$

(B) $\forall x((G(x) \wedge S(x)) \rightarrow P(x))$

(C) $\exists x((G(x) \wedge S(x)) \rightarrow P(x))$

(D) $\forall x((G(x) \vee S(x)) \rightarrow P(x))$



Thank you!

Lecture: **Semantic Nets, Frames, Conceptual Graphs**

Dr. Partha Pakray

Knowledge Representation as a medium for human expression

- An intelligent system must have KRs that can be interpreted by humans.
 - We need to be able to encode information in the knowledge base without significant effort.
 - **We need to be able to understand what the system knows and how it draws its conclusions.**

Knowledge Representation

- **Logic** (propositional, predicate)
- **Network representation**
 - Semantic nets
- **Structured representation**
 - Frames
- **Issues in KR**
 - Hierarchies, inheritance, exceptions
- **Advantages and disadvantages**

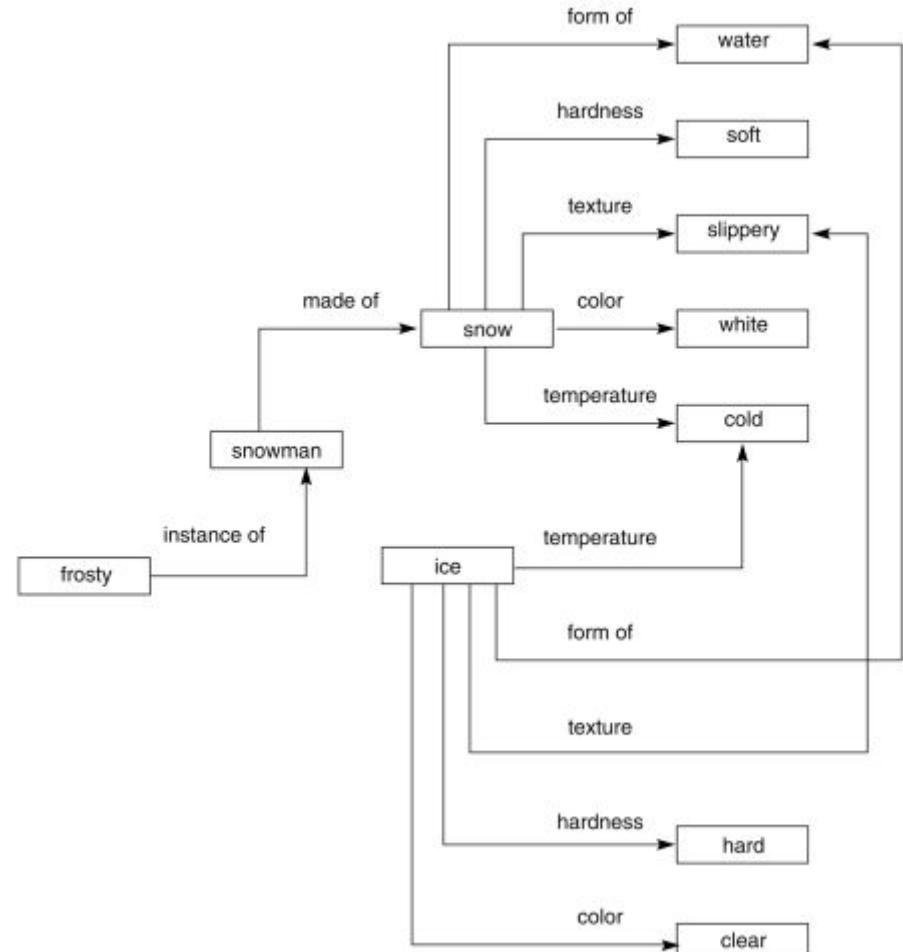
Semantic Networks

- First introduced by Quillian back in the late-60s

M. Ross Quillian. "Semantic Memories", In M. M. Minsky, editor, *Semantic Information Processing*, pages 216-270. Cambridge, MA: MIT Press, 1968
- **Semantic network** is simple representation scheme which uses a graph of labeled nodes and labeled directed arcs to encode knowledge
 - Nodes – objects, concepts, events
 - Arcs – relationships between nodes
- **Graphical depiction** associated with semantic networks is a big reason for their popularity

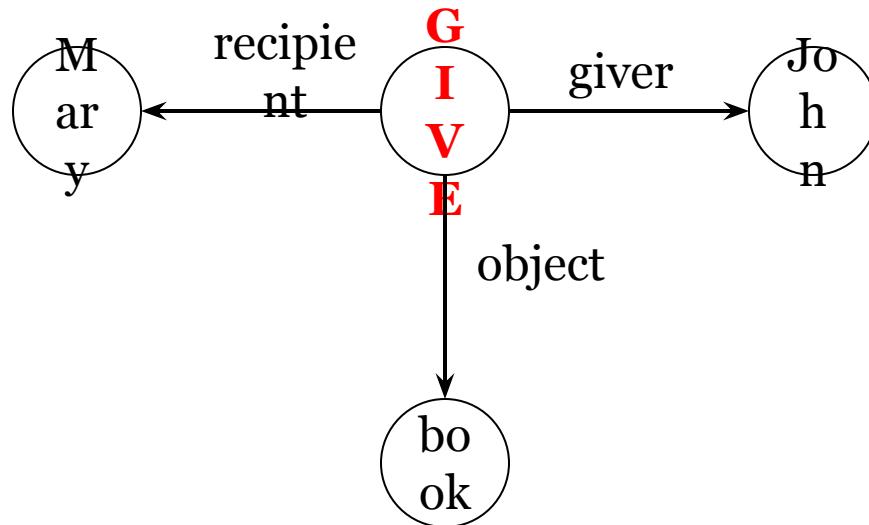
A brief look at semantic networks

- A semantic network is an irregular graph that has concepts in vertices and relations on arcs.
- Relations can be ad-hoc, but they can also be quite general, for example, “is a” (ISA), “a kind of” (AKO), “an instance of”, “part of”.
- Relations often express physical properties of objects (colour, length, and lots of others).
- Most often, relations link two concepts.



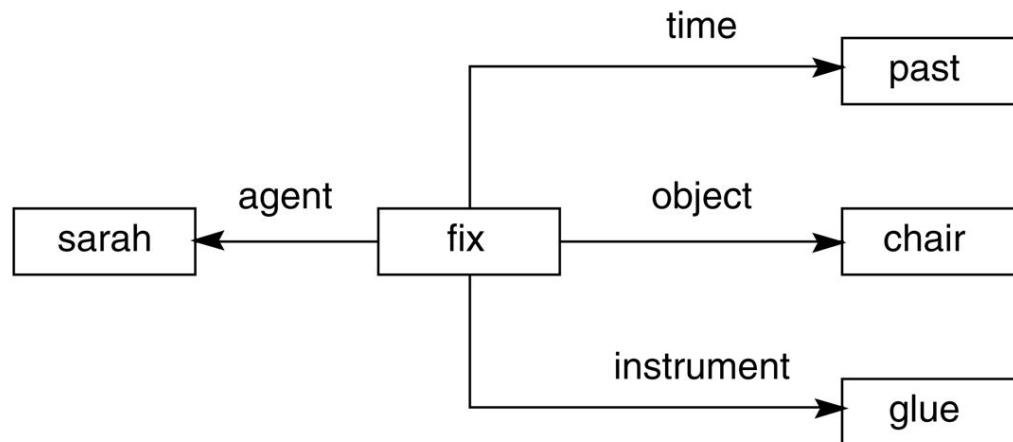
Non-binary relations

- We can represent the generic *give* event as a relation involving three things:
 - A giver
 - A recipient
 - An object



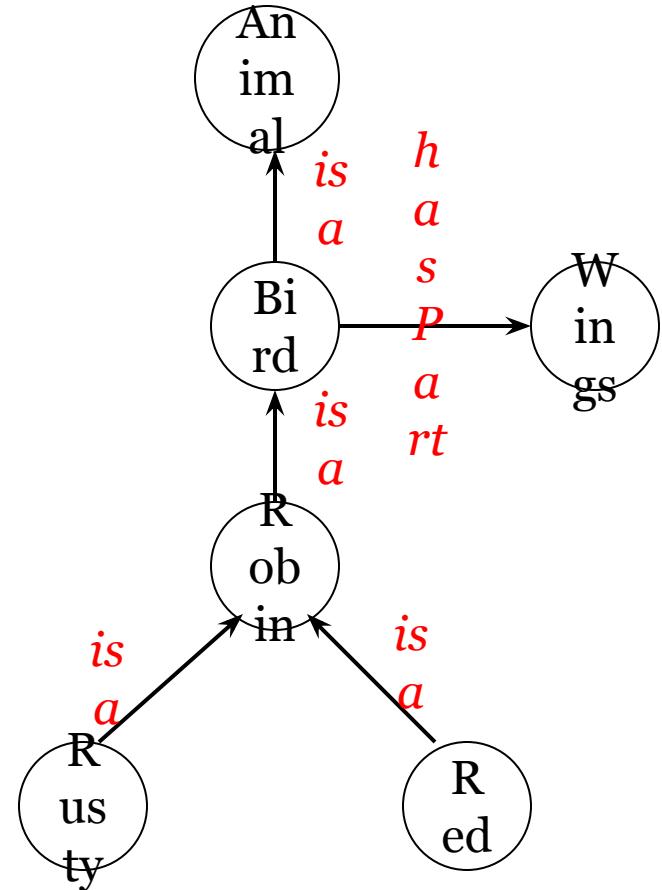
... semantic networks

- General semantic relations help represent the meaning of simple sentences in a systematic way.
- A sentence is centred on a verb that *expects* certain arguments.
- For example, verbs usually denotes actions (with *agents*) or states (with passive *experiencers*, for example, “he dreams” or “he is sick”).



Inheritance

- Inheritance is one of the main kind of reasoning done in semantic nets
- The **ISA** (is a) relation is often used to link a class and its superclass.
- Some links (e.g. **haspart**) are inherited along **ISA** paths
- The semantics of a semantic net can be relatively informal or very formal
 - Often defined at the implementation level



Advantages of Semantic nets

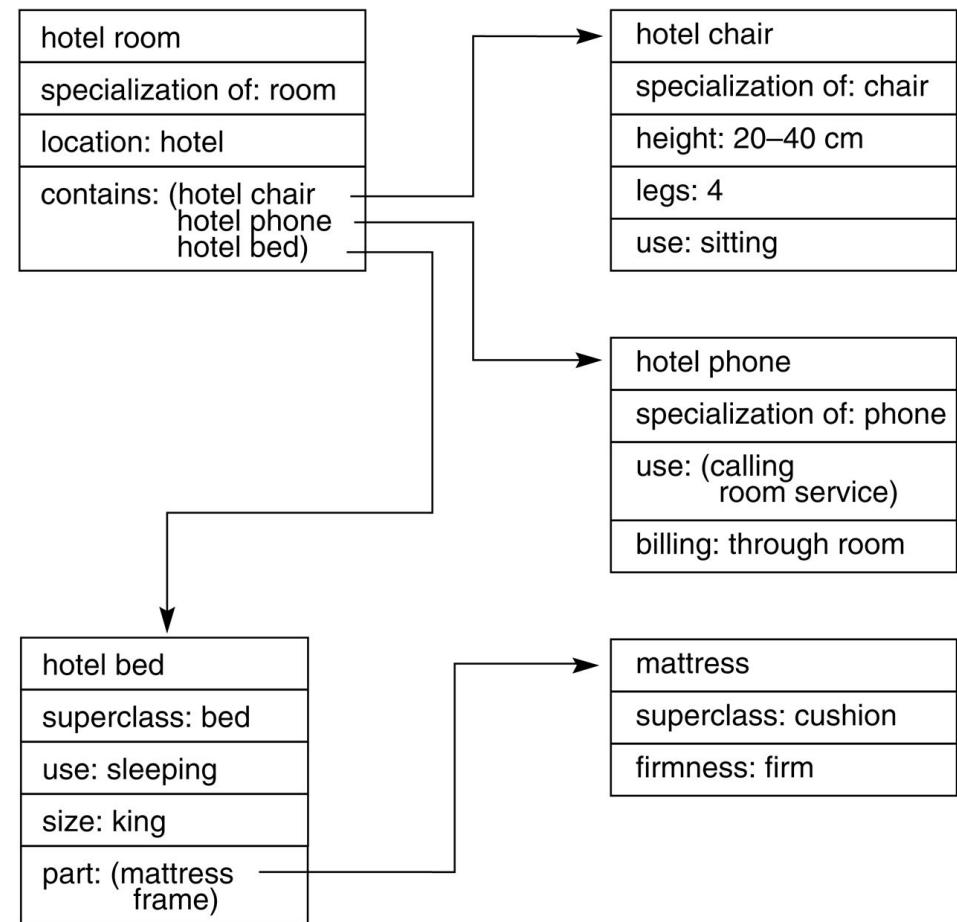
- Easy to visualize
- Formal definitions of semantic networks have been developed.
- Related knowledge is easily clustered.
- Efficient in space requirements
 - Objects represented only once
 - Relationships handled by pointers

Disadvantages of Semantic nets

- Inheritance (particularly from multiple sources and when exceptions in inheritance are wanted) can cause problems.
- Facts placed inappropriately cause problems.
- No standards about node and arc values

Frames and frame systems

- A frame represents a concept;
- a frame system represents an organization of knowledge about a set of related concepts.
- A frame has slots that denote properties of objects. Some slots have *default* fillers, some are empty (may be filled when more becomes known about an object).
- Frames are linked by relations of specialization/generalization and by many ad-hoc relations.



Frames

3 components of a frame

- frame name
- attributes (slots)
- values (fillers: list of values, range, string, etc.)

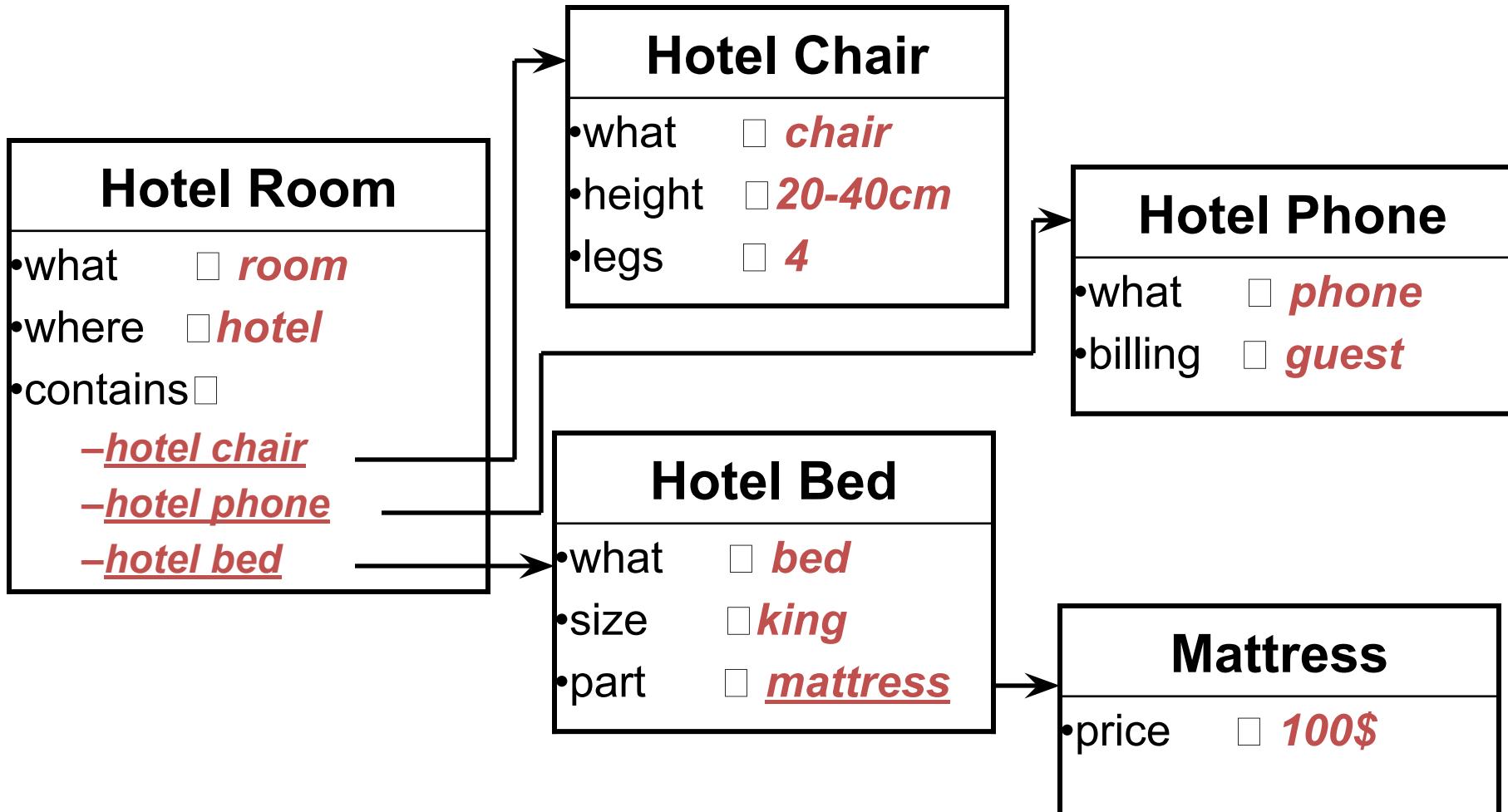
Book Frame	
Slot	Filler
Title	<i>AI. A modern Approach</i>
Author	Russell & Norvig
Year	2003

Features of Frame Representation

- More natural support of values than semantic nets (each slot has constraints describing legal values that a slot can take)
- Can be easily implemented using object-oriented programming techniques
- Inheritance is easily controlled

Inheritance

- Similar to Object-Oriented programming paradigm



Benefits of Frames

- Makes programming easier by grouping related knowledge
- Easily understood by non-developers
- Expressive power
- Easy to set up slots for new properties and relations
- Easy to include default information and detect missing values

Drawbacks of Frames

- No standards (slot-filler values)
- More of a general methodology than a specific representation:
 - Frame for a class-room will be different for a professor and for a maintenance worker
- No associated reasoning/inference mechanisms

Conceptual graphs

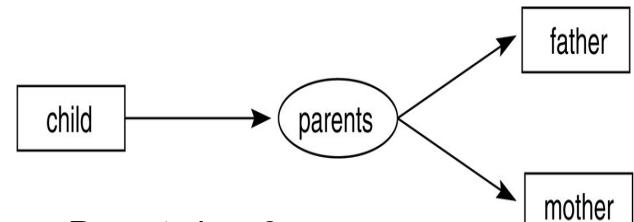
- John Sowa created the conceptual graph notation in 1984. It has substantial philosophical and psychological motivation.
- It is still quite a popular knowledge representation formalism, especially in semantic processing of language, and a topic of interesting research.
- Conceptual graphs can be expressed in first-order logic but due to its graphical form it may be easier to understand than logic.



Flies is a 1-ary relation.



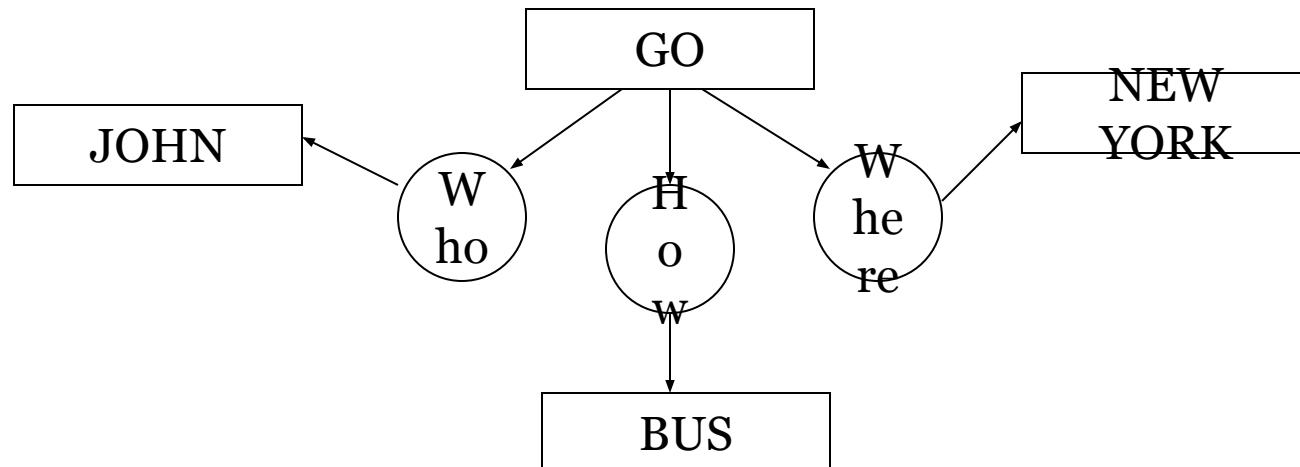
Color is a 2-ary relation.



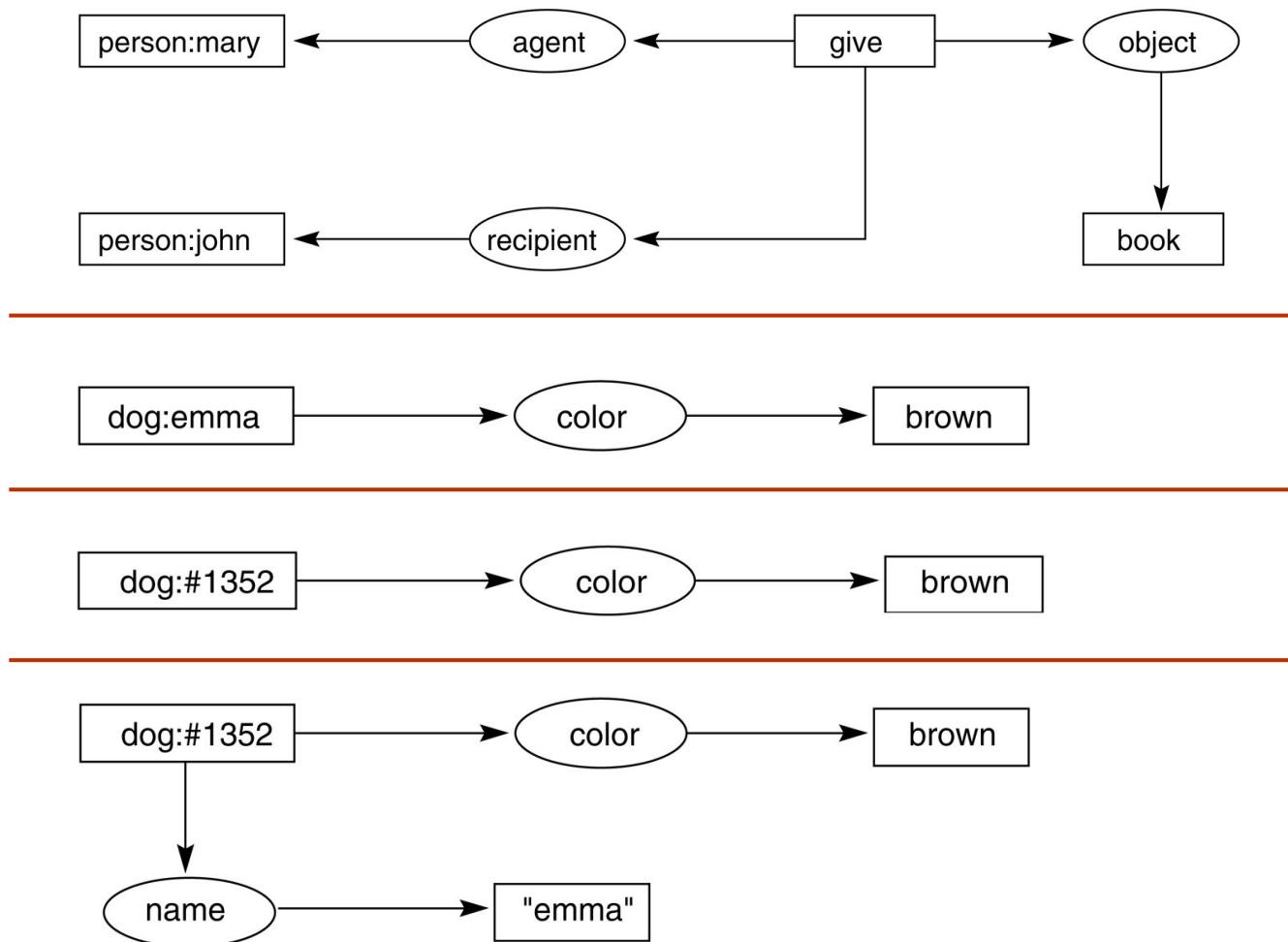
Parents is a 3-ary relation.

Conceptual Graphs

- *Conceptual graphs* are semantic nets representing the meaning of (simple) sentences in natural language
- Two types of nodes:
 - *Concept nodes*; there are two types of concepts, individual concepts and generic concepts
 - *Relation nodes*(binary relations between concepts)

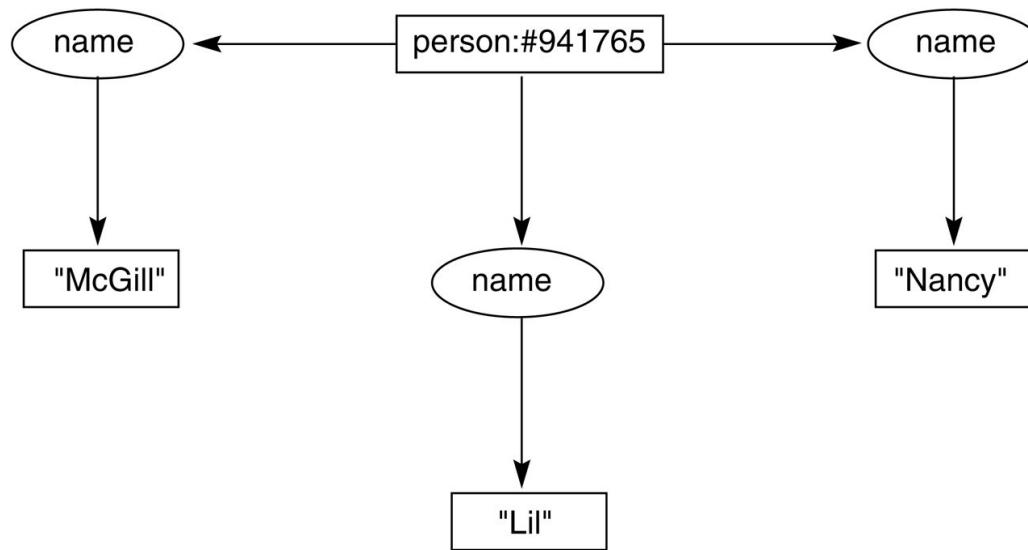


Conceptual graphs (2)

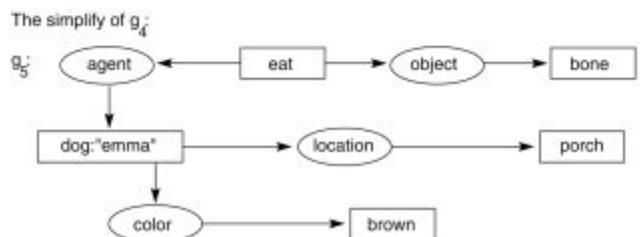
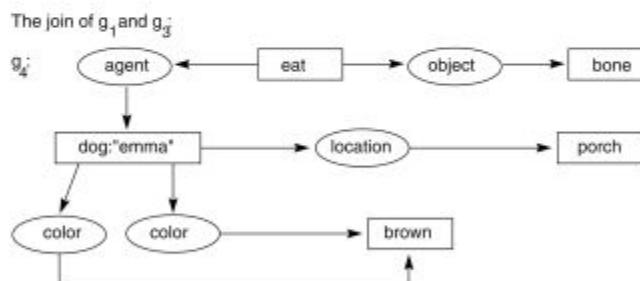
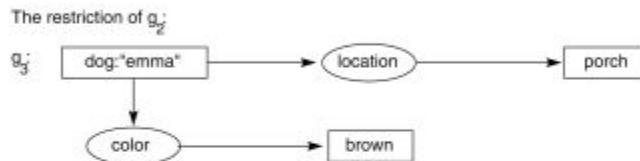
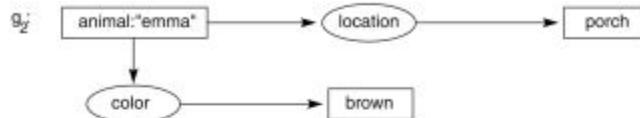
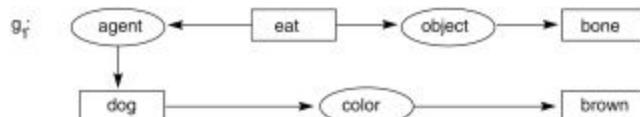


Conceptual graphs (3)

Her name was Magill, and she called herself Lil,
but everyone knew her as Nancy.



Conceptual graphs (5)



Specialization and type hierarchy

dogs are animals

(g_1) A brown dog eats a bone.

(g_2) ... Emma, the brown animal on the porch...

(g_3) ... Emma, the brown dog on the porch...

(g_4) Emma, the brown dog on the porch, eats a bone.

The challenge is to get this from text!

Resolution Refutation

Predicate Calculus

- Introduction through an example (Zohar Manna, 1974):
 - Problem: **A, B and C belong to the Himalayan club.**
Every member in the club is either a mountain climber or a skier or both. A likes whatever B dislikes and dislikes whatever B likes. A likes rain and snow. No mountain climber likes rain. Every skier likes snow. Is there a member who is a mountain climber and not a skier?
- Given knowledge has:
 - Facts
 - Rules

Predicate Calculus: Example contd.

- Let mc denote mountain climber and sk denotes skier. Knowledge representation in the given problem is as follows:
 1. $member(A)$
 2. $member(B)$
 3. $member(C)$
 4. $\forall x[member(x) \rightarrow (mc(x) \vee sk(x))]$ Every member in the club is either a mountain climber or a skier or both
 5. $\forall x[mc(x) \rightarrow \sim like(x, rain)]$ No mountain climber likes rain
 6. $\forall x[sk(x) \rightarrow like(x, snow)]$ Every skier likes snow
 7. $\forall x[like(B, x) \rightarrow \sim like(A, x)]$ dislikes whatever B likes
 8. $\forall x[\sim like(B, x) \rightarrow like(A, x)]$ A likes whatever B dislikes
 9. $like(A, rain)$ A likes rain.
 10. $like(A, snow)$ A likes snow.
 - Question: $\exists x[member(x) \wedge mc(x) \wedge \sim sk(x)]$
Is there a member who is a mountain climber and not a skier?
- 2. We have to infer the 11th expression from the given 10.
- 3. Done through Resolution Refutation.

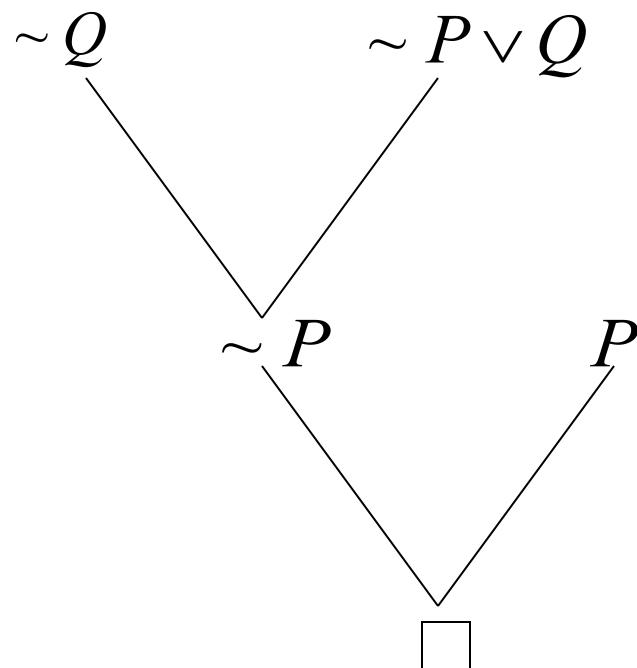
Inferencing in Predicate Calculus

- Forward chaining
 - Given P, $P \rightarrow Q$, to infer Q
 - P, match L.H.S of $P \rightarrow Q$
 - Assert Q from R.H.S
- Backward chaining
 - Q, Match R.H.S of $P \rightarrow Q$
 - assert P
 - Check if P exists
- Resolution – Refutation
 - Negate goal
 - Convert all pieces of knowledge into clausal form (disjunction of literals)
 - See if contradiction indicated by null clause can be derived



1. P
2. $P \rightarrow Q$ converted to $\sim P \vee Q$
3. $\sim Q$

Draw the resolution tree (actually an inverted tree). Every node is a clausal form and branches are intermediate inference steps.



Terminology

- Pair of clauses being resolved is called the Resolvents. The resulting clause is called the Resolute.
- Choosing the correct pair of resolvents is a matter of search.

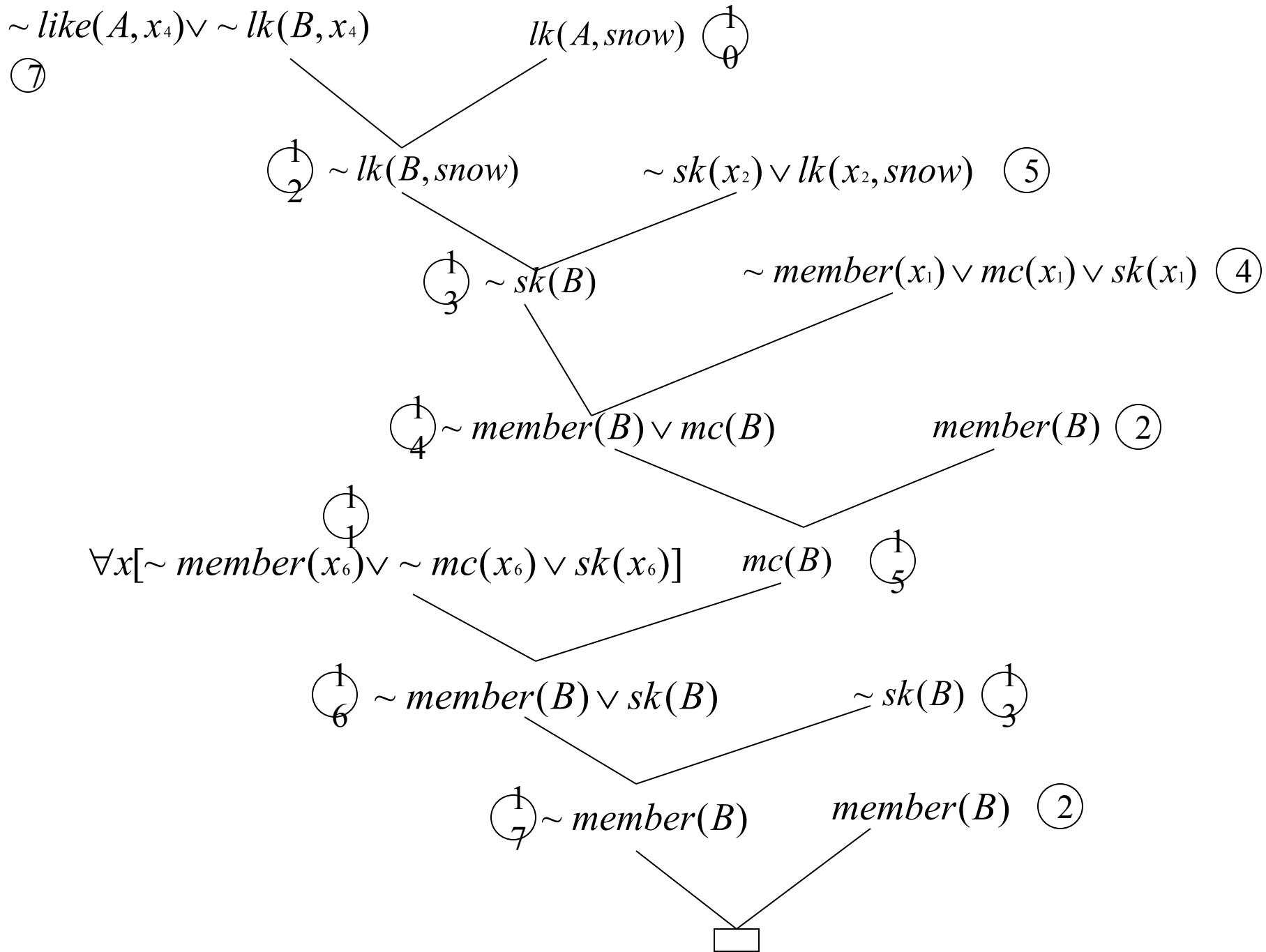
Club example revisited

1. $\text{member}(A)$
2. $\text{member}(B)$
3. $\text{member}(C)$
4. $\forall x[\text{member}(x) \rightarrow (\text{mc}(x) \vee \text{sk}(x))]$
 - Can be written as $[\text{member}(x) \rightarrow (\text{mc}(x) \vee \text{sk}(x))]$
 - $\sim \text{member}(x) \vee \text{mc}(x) \vee \text{sk}(x)$
5. $\forall x[\text{sk}(x) \rightarrow \text{lk}(x, \text{snow})]$
 - $\sim \text{sk}(x) \vee \text{lk}(x, \text{snow})$
6. $\forall x[\text{mc}(x) \rightarrow \sim \text{lk}(x, \text{rain})]$
 - $\sim \text{mc}(x) \vee \sim \text{lk}(x, \text{rain})$
7. $\forall x[\text{like}(A, x) \rightarrow \sim \text{lk}(B, x)]$
 - $\sim \text{like}(A, x) \vee \sim \text{lk}(B, x)$

8. $\forall x[\sim lk(A, x) \rightarrow lk(B, x)]$
 - $lk(A, x) \vee lk(B, x)$
9. $lk(A, rain)$
10. $lk(A, snow)$
11. $\exists x[member(x) \wedge mc(x) \wedge \sim sk(x)]$
 - Negate – $\forall x[\sim member(x) \vee \sim mc(x) \vee sk(x)]$

- Now standardize the variables apart which results in the following

1. $\text{member}(A)$
2. $\text{member}(B)$
3. $\text{member}(C)$
4. $\sim \text{member}(x_1) \vee \text{mc}(x_1) \vee \text{sk}(x_1)$
5. $\sim \text{sk}(x_2) \vee \text{lk}(x_2, \text{snow})$
6. $\sim \text{mc}(x_3) \vee \sim \text{lk}(x_3, \text{rain})$
7. $\sim \text{like}(A, x_4) \vee \sim \text{lk}(B, x_4)$
8. $\text{lk}(A, x_5) \vee \text{lk}(B, x_5)$
9. $\text{lk}(A, \text{rain})$
10. $\text{lk}(A, \text{snow})$
11. $[\sim \text{member}(x_6) \vee \sim \text{mc}(x_6) \vee \text{sk}(x_6)]$



Problem 1

- Ravi likes all kind of food.
- Apple and chicken are food.
- Anything anyone eats and is not killed, it is food.
- Ajay eats peanuts and is still alive.
- Rita eats everything Ajay eats.

Prove

1. Ravi likes peanuts

Robotic Knowledge Representation and inferencing

A *planning* agent

- An agent interacts with the world via perception and actions
- Perception involves sensing the world and assessing the situation
 - creating some internal representation of the world
- Actions are what the agent does in the domain. Planning involves reasoning about actions that the agent intends to carry out
- *Planning* is the reasoning side of acting
- This reasoning involves the representation of the world that the agent has, as also the representation of its actions.
- Hard constraints where the objectives *have to* be achieved completely for success
- The objectives could also be soft constraints, or *preferences*, to be achieved as much as possible

Interaction with static domain

- The agent has complete information of the domain (perception is perfect), actions are instantaneous and their effects are deterministic.
- The agent knows the world completely, and it can take all facts into account while planning.
- The fact that actions are instantaneous implies that there is no notion of time, but only of sequencing of actions.
- The effects of actions are deterministic, and therefore the agent knows what the world will be like after each action.

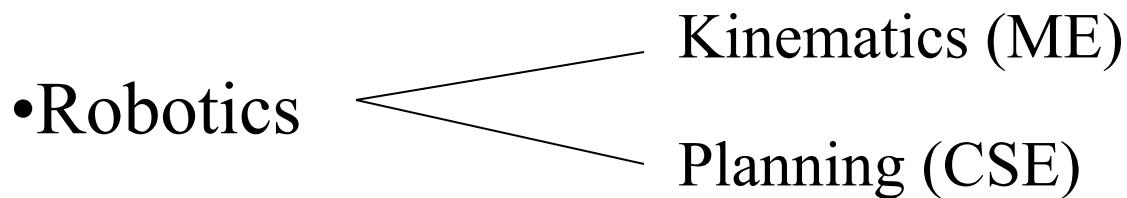
Two kinds of planning

- *Projection* into the future
 - The planner searches through the possible combination of actions to find the *plan* that will work
- *Memory based planning*
 - looking into the past
 - The agent can retrieve a plan from its memory

Planning

- *Definition : Planning is arranging a sequence of actions to achieve a goal.*

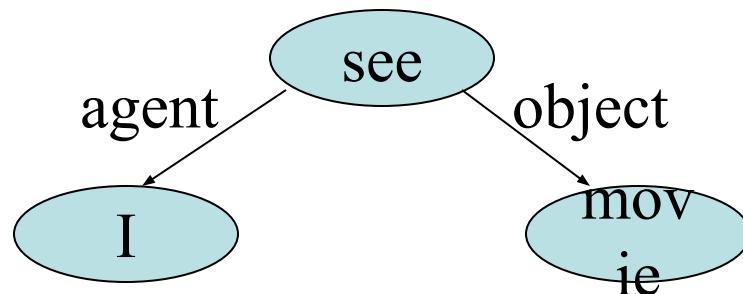
- Uses core areas of AI like searching and reasoning &
- Is the core for areas like NLP, Computer Vision.



- Examples : Navigation , Manoeuvring, Language Processing (Generation)

Language & Planning

- Non-linguistic representation for sentences.



- Sentence generation
 - Word order determination (Syntax planning)
E.g. I see movie (English)
I movie see (Intermediate Language)

STRIPS

- Stanford Research Institute Problem Solver (1970s)
 - Planning system for a robotics project : SHAKEY (by Nilsson et.al.)
- Knowledge Representation : First Order Logic.
- Algorithm : Forward chaining on rules.
- Any search procedure : Finds a path from *start* to *goal*.
 - Forward Chaining : Data-driven inferencing
 - Backward Chaining : Goal-driven

Forward & Backward Chaining

- Rule : $\text{man}(x) \square \text{mortal}(x)$
- Data : $\text{man}(\text{Shakespeare})$
To prove : $\text{mortal}(\text{Shakespeare})$

- *Forward Chaining:*

$\text{man}(\text{Shakespeare})$ matches LHS of Rule.

$X = \text{Shakespeare}$

$\Rightarrow \text{mortal}(\text{Shakespeare})$ added

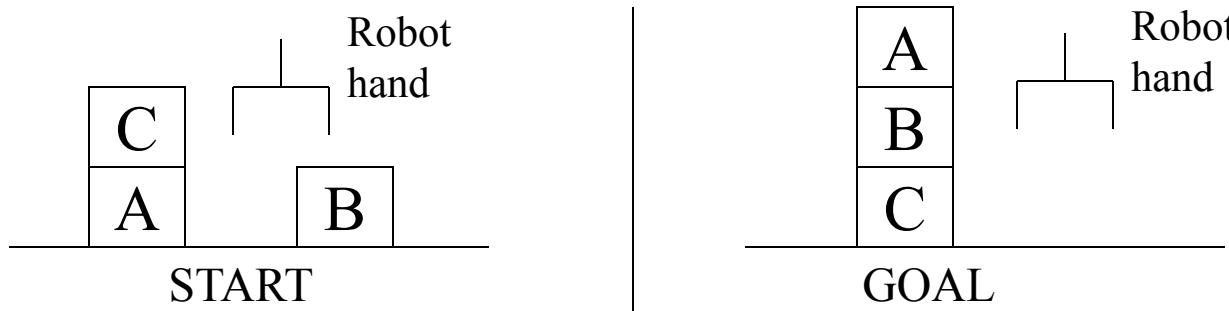
- Forward Chaining used by design expert systems

- *Backward Chaining:* uses RHS matching

- Used by diagnostic expert systems

Example : Blocks World

- STRIPS : A planning system – Has rules with precondition deletion list and addition list



Sequence of actions :

1. Grab C
2. Pickup C
3. Place on table C
4. Grab B
5. Pickup B
6. Stack B on C
7. Grab A
8. Pickup A
9. Stack A on B

Example : Blocks World

- Fundamental Problem :

The *frame problem* in AI is concerned with the question of what piece of knowledge is relevant to the situation.

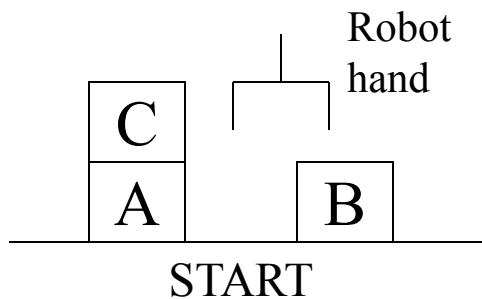
- Fundamental Assumption : Closed world assumption

If something is not asserted in the knowledge base, it is assumed to be false.

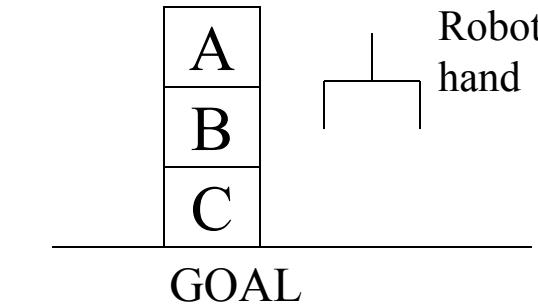
(Also called “Negation by failure”)

Example : Blocks World

- STRIPS : A planning system – Has rules with precondition deletion list and addition list



on(B, table)
on(A, table)
on(C, A)
hand empty
clear(C)
clear(B)



on(C, table)
on(B, C)
on(A, B)
hand empty
clear(A)

Rules

- $R1 : pickup(x)$

Precondition & Deletion List : hand empty,
on(x,table), clear(x)

Add List : holding(x)

- $R2 : putdown(x)$

Precondition & Deletion List : holding(x)

Add List : hand empty, on(x,table), clear(x)

Rules

- $R3 : stack(x,y)$

Precondition & Deletion List : holding(x), clear(y)

Add List : on(x,y), clear(x)

- $R4 : unstack(x,y)$

Precondition & Deletion List : on(x,y), clear(x)

Add List : holding(x), clear(y)

Plan for the block world problem

- For the given problem, Start \square Goal can be achieved by the following sequence :
 1. Unstack(C,A)
 2. Putdown(C)
 3. Pickup(B)
 4. Stack(B,C)
 5. Pickup(A)
 6. Stack(A,B)
- Execution of a plan: achieved through a data structure called Triangular Table.

Triangular Table

1	on(C,A) clear(C) hand empty	unstack(C,A)				
2		holding(C)	putdown(C)			
3	on(B,table)		hand empty	pickup(B)		
4		clear(C)	holding(B)	stack(B,C)		
5	on(A,table)	clear(A)		hand empty	pickup(A)	
6				clear(B)	holding(A)	stack(A,B)
7		on(C,table)		on(B,C)		on(A,B) clear(A)

0

1

2

3

4

5

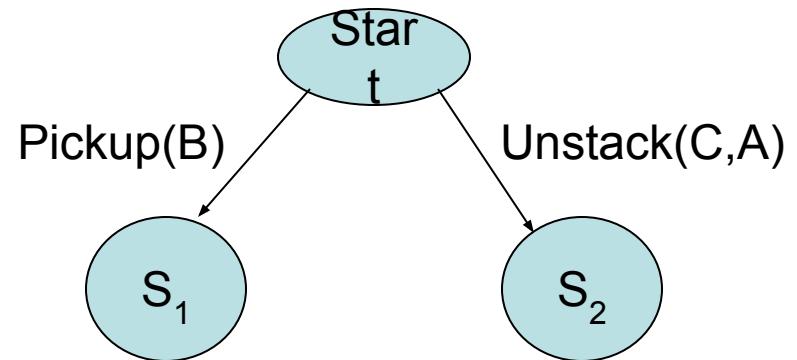
6

Triangular Table

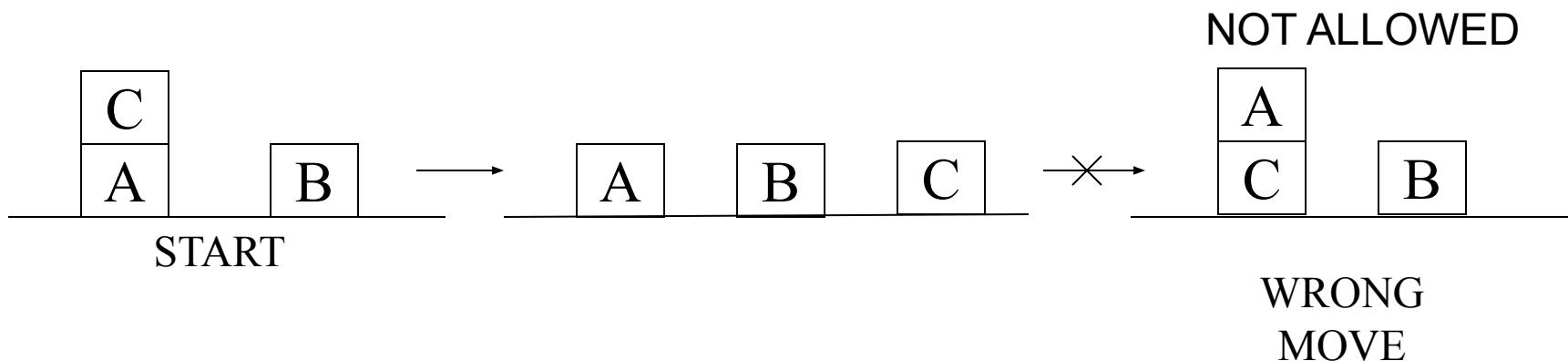
- For n operations in the plan, there are :
 - $(n+1)$ rows : $1 \square n+1$
 - $(n+1)$ columns : $0 \square n$
- At the end of the i^{th} row, place the i^{th} component of the plan.
- The row entries for the i^{th} step contain the pre-conditions for the i^{th} operation.
- The column entries for the j^{th} column contain the add list for the rule on the top.
- The $\langle i, j \rangle^{\text{th}}$ cell (where $1 \leq i \leq n+1$ and $0 \leq j \leq n$) contain the pre-conditions for the i^{th} operation that are added by the j^{th} operation.
- The first column indicates the starting state and the last row indicates the goal state.

Search in case of planning

- Ex: Blocks world



- Triangular table leads
- to some amount of fault-tolerance in the robot



Resilience in Planning

- After a wrong operation, can the robot come back to the right path ?
- *i.e.* after performing a wrong operation, if the system again goes towards the goal, then it has resilience w.r.t. that operation
- Advanced planning strategies
 - Hierarchical planning
 - Probabilistic planning
 - Constraint satisfaction

Predicate Calculus

- Well Known Example:
 - Man is mortal : rule
$$\forall x[\text{man}(x) \rightarrow \text{mortal}(x)]$$
 - shakespeare is a man
 $\text{man}(\text{shakespeare})$
 - To infer shakespeare is mortal
 $\text{mortal}(\text{shakespeare})$

Forward Chaining/ Inferencing

- $man(x) \rightarrow mortal(x)$
 - *Dropping the quantifier, implicitly Universal quantification assumed*
 - $man(shakespeare)$
- Goal $mortal(shakespeare)$
 - Found in one step
 - $x = shakespeare$, unification

Backward Chaining/ Inferencing

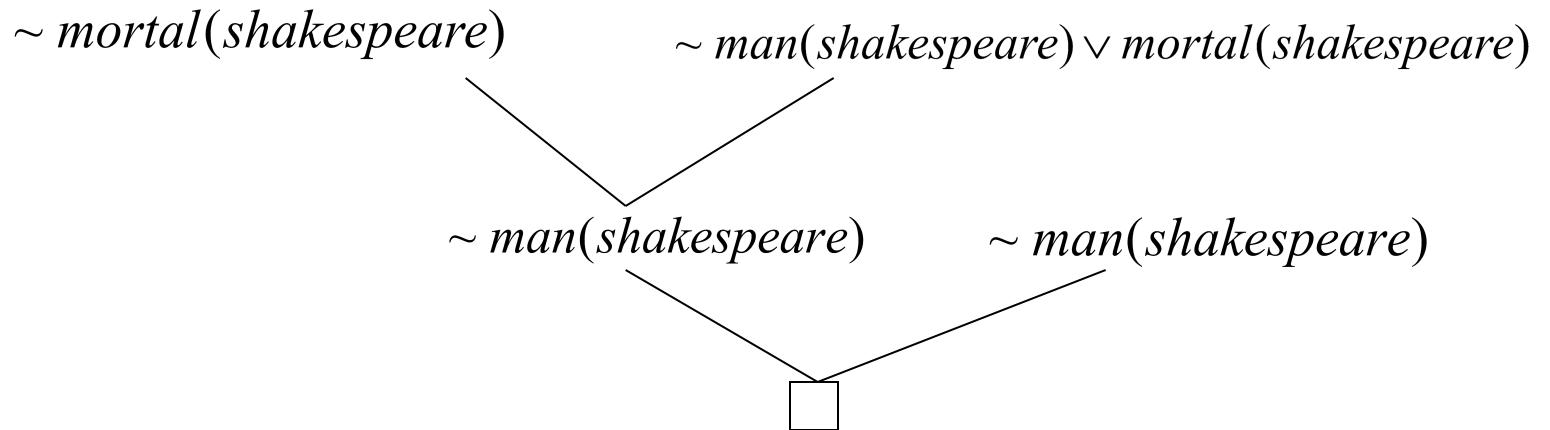
- $man(x) \rightarrow mortal(x)$
- Goal $mortal(\text{shakespeare})$
 - $x = \text{shakespeare}$
 - Travel back over and hit the fact asserted
 - $man(\text{shakespeare})$

Resolution - Refutation

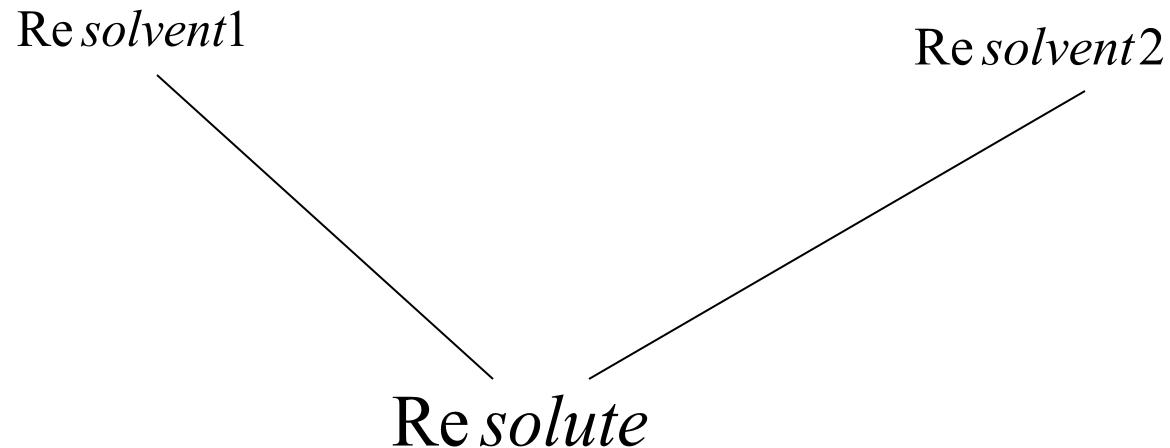
- $man(x) \rightarrow mortal(x)$
 - Convert to clausal form
 - $\neg man(shakespeare) \vee mortal(x)$
- Clauses in the knowledge base
 - $\neg man(shakespeare) \vee mortal(x)$
 - $man(shakespeare)$
 - $mortal(shakespeare)$

Resolution – Refutation contd

- Negate the goal
 - $\sim \text{man}(\text{shakespeare})$
- Get a pair of resolvents



Resolution Tree



Search in resolution

- Heuristics for Resolution Search
 - Goal Supported Strategy
 - Always start with the negated goal
 - Set of support strategy
 - Always one of the resolvents is the most recently produced resolute

Assignment

- Prove the inferencing in the himalayan club example with different starting points, producing different resolution trees.
- Think of a Prolog implementation of the problem
- Prolog Reference (Prolog by Chocksin & Melish)

Role and applications of AI in Cybersecurity

Dr. Partha Pakray

Assistant Professor

Department of Computer Science and Engineering
National Institute of Technology Silchar

<http://cs.nits.ac.in/partha>

8259065018 (M)

Email: partha@cse.nits.ac.in

Outline of this talk

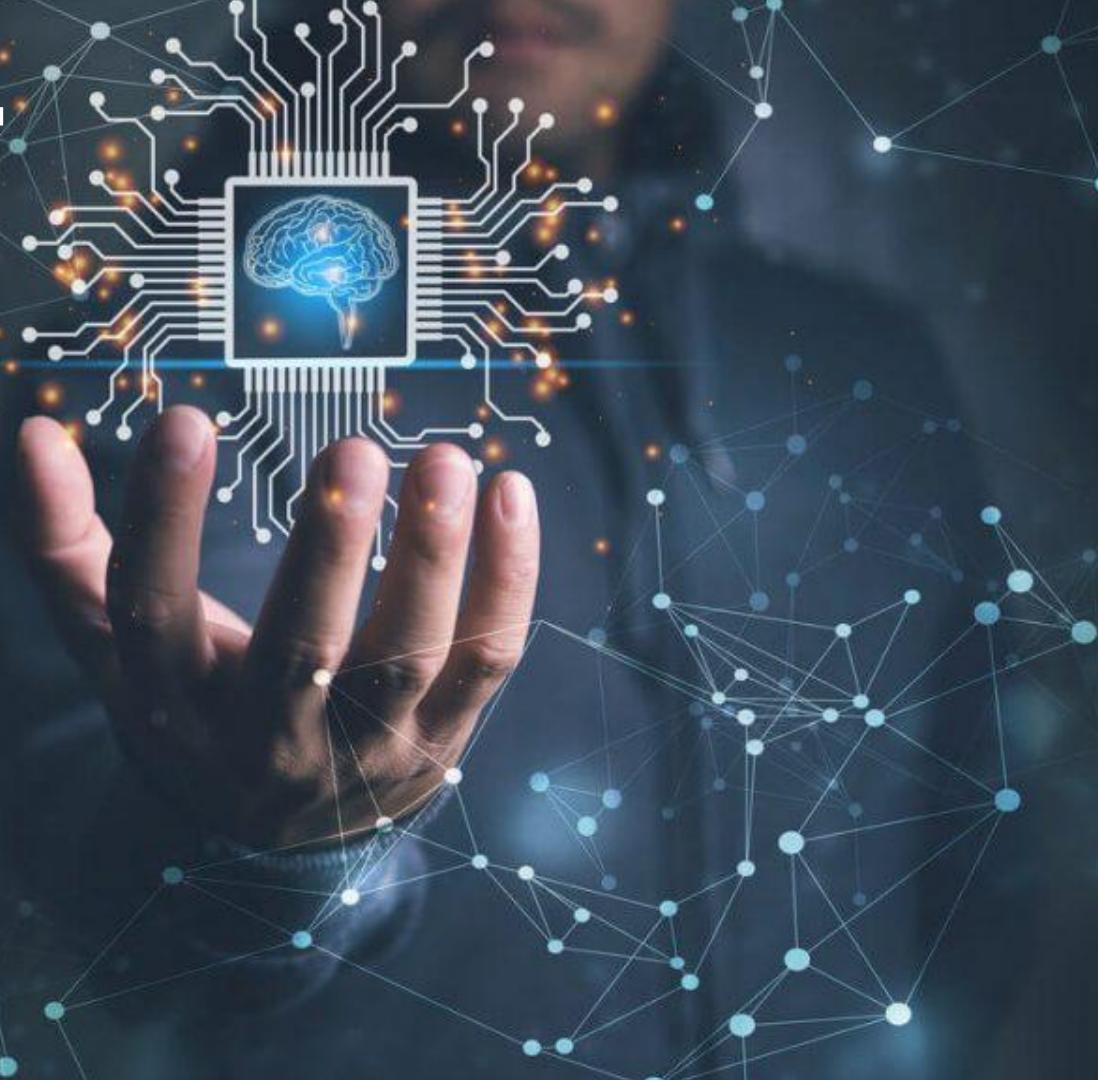
- What is AI?
- What is NLP?
- What is Cyber Security?
- Relation: AI-NLP-Cyber Security?
- Machine Learning
- Social Media - counter terrorism
- Deep Fake
- Cyberbullying
- Text Summarization
- Fake News
- Spam Detection
- Identification of Hate Content

What is Artificial Intelligence

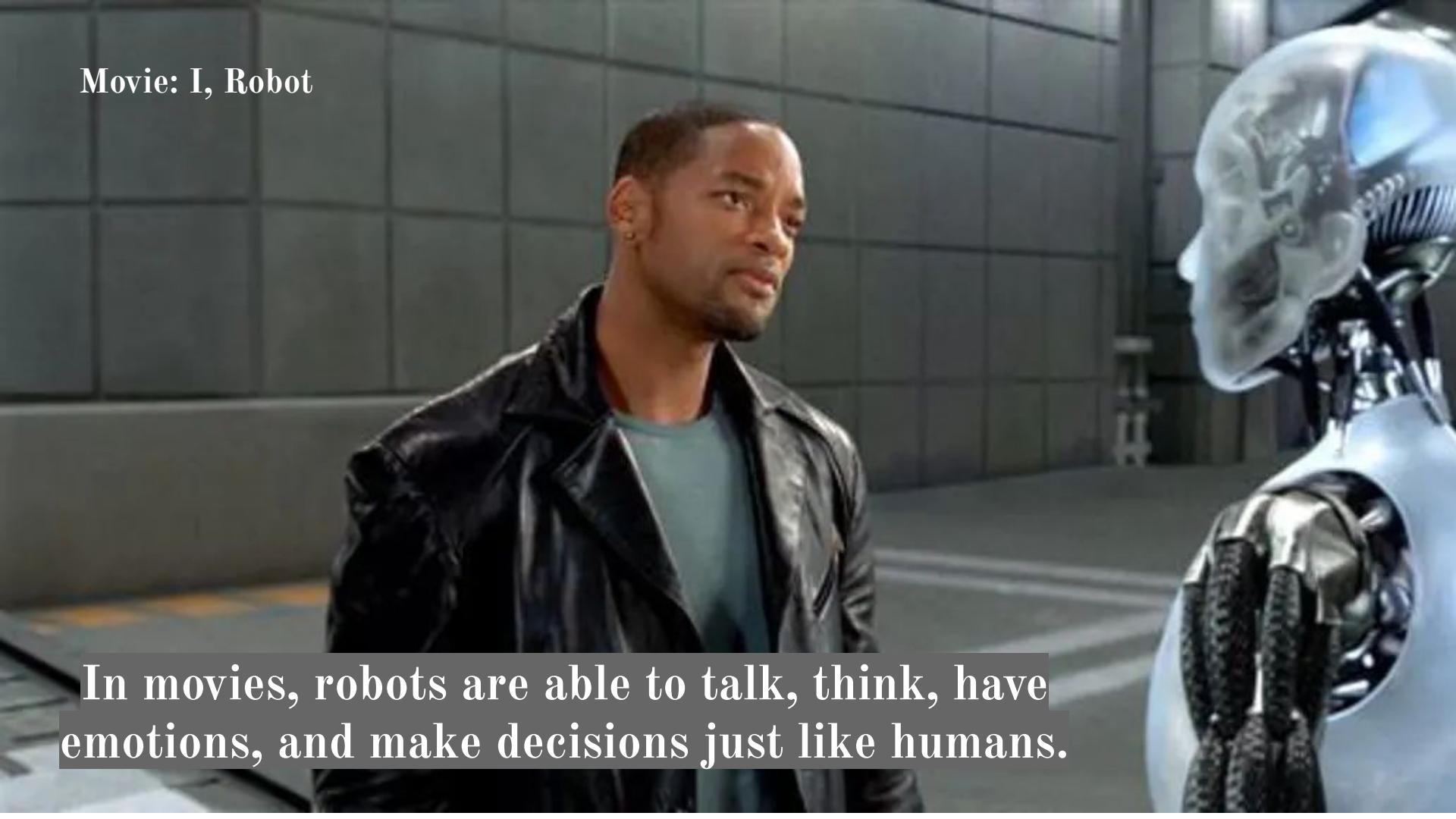


What do you think

Artificial Intelligence is?



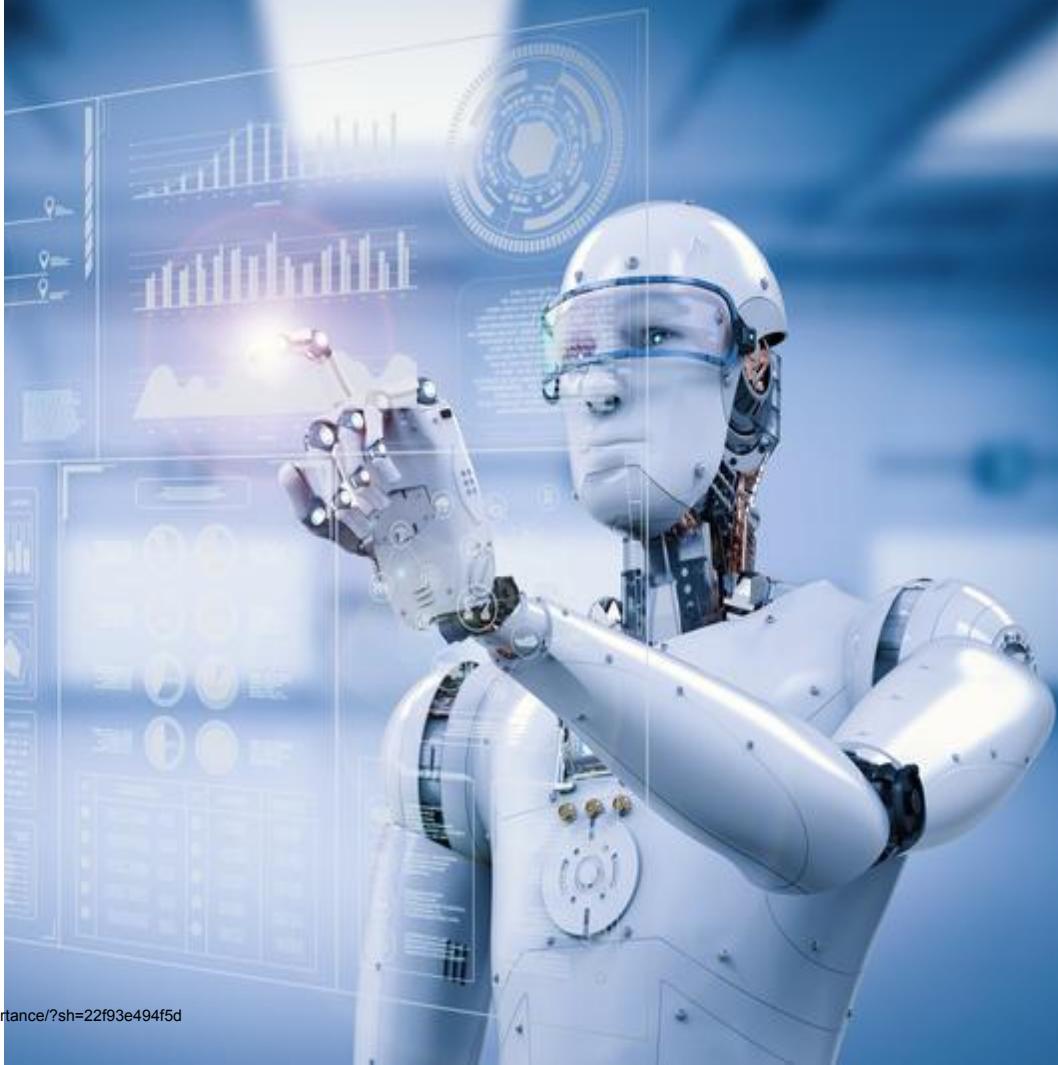
Movie: I, Robot



In movies, robots are able to talk, think, have emotions, and make decisions just like humans.

What is Artificial Intelligence?

- Artificial Intelligence
 - is concerned with the design of **intelligence** in an **artificial device**.
 - Term coined by McCarthy in 1956
- Examples: visual perception, speech recognition, decision-making, translation between languages, robotics.



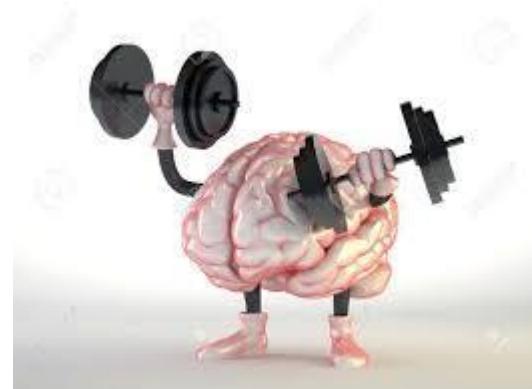
Intelligence:

“the capacity to learn and solve problems”

Artificial Intelligence:

Artificial intelligence (AI) is the intelligence of machines and robots and the branch of computer science that aims to create it

- Behave as intelligently as a human
- Behave in the best possible manner
- Thinking
- Acting



Some Definition

Artificial Intelligence is...

- “**... the science and engineering of making intelligent machines**” ... “[where] intelligence is the computational part of the ability to achieve goals in the world (by John McCarthy in 1955)
- “**... the science of making machines do things that would require intelligence if done by men**” (by Marvin Minsky in 1968)
- “**... the science of making machines smart**” (Demis Hassabis, CEO and founder of DeepMind, now part of Google)
- **anything that makes machines act more intelligently**” (IBM’s definition)
- **intelligence demonstrated by a machine or by software where** intelligence measures an agent’s general ability to achieve goals in a wide range of environments” (by Calum Chase)

Some Fundamental Questions

What is **intelligence**?

What is **thinking**?

What is **machine**?

Is the **computer** a machine?

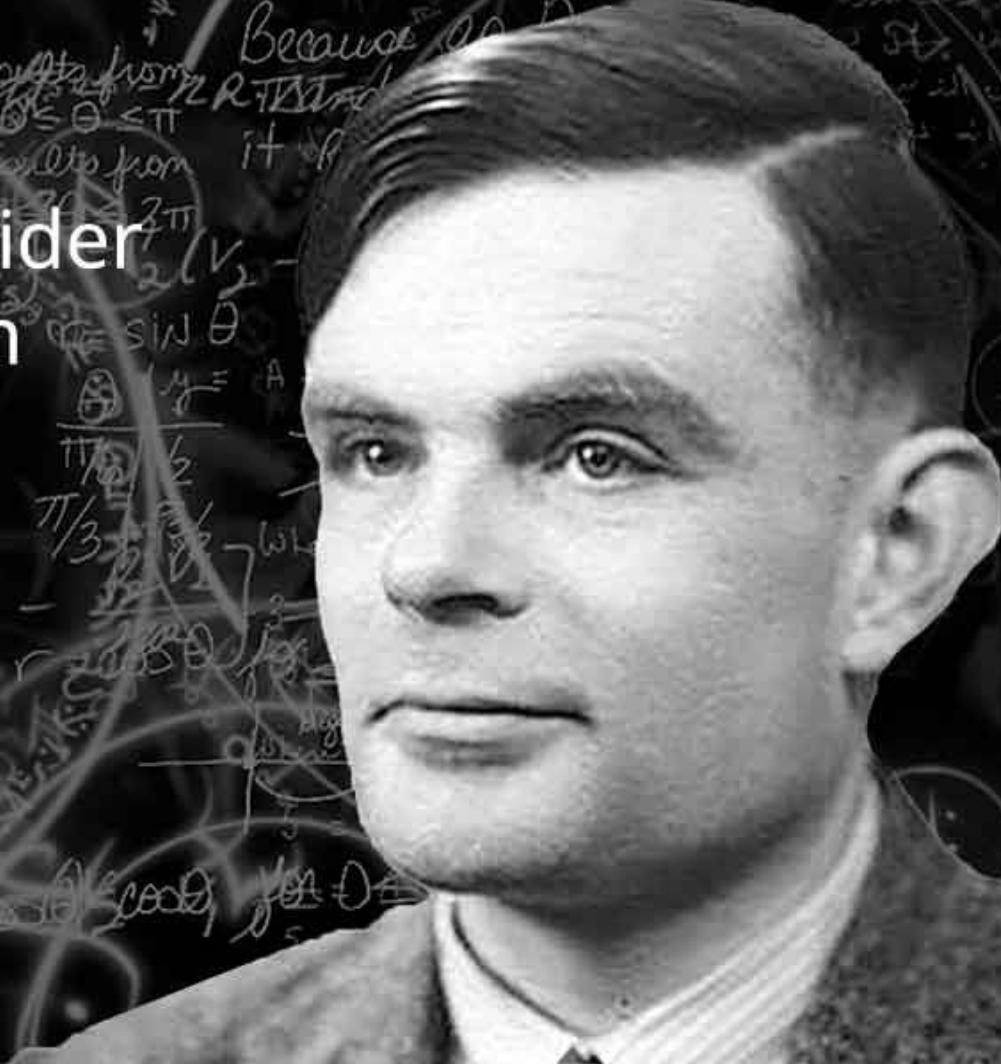


Can a **machine think**?

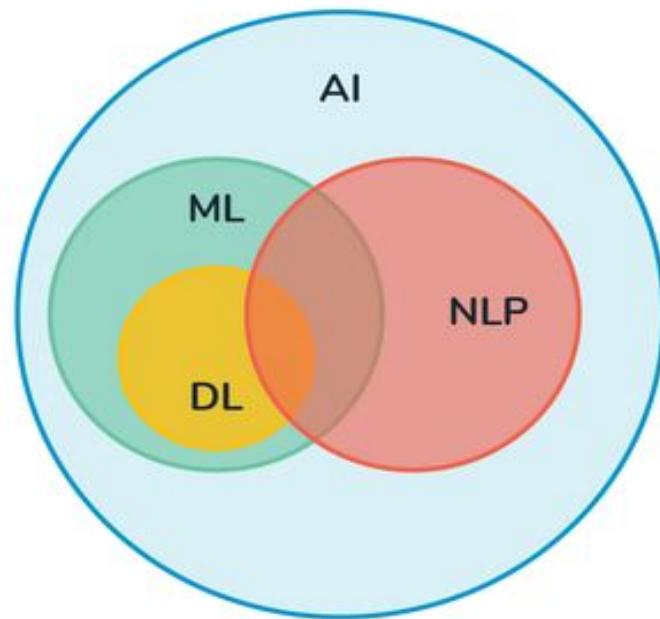
"I propose to consider
the question, 'Can
machines think?'

~ Alan Turing

Carnegie Mellon University
Machine Learning

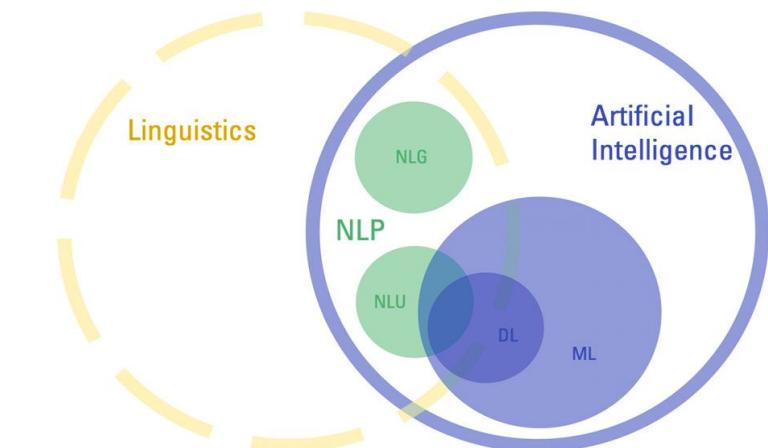


Natural Language Processing



Natural Language Processing

- NLP is the branch of computer science focused on developing systems that allow computers to communicate with people using everyday language.
- Also called Computational Linguistics – Also concerns how computational methods can aid the understanding of human language.





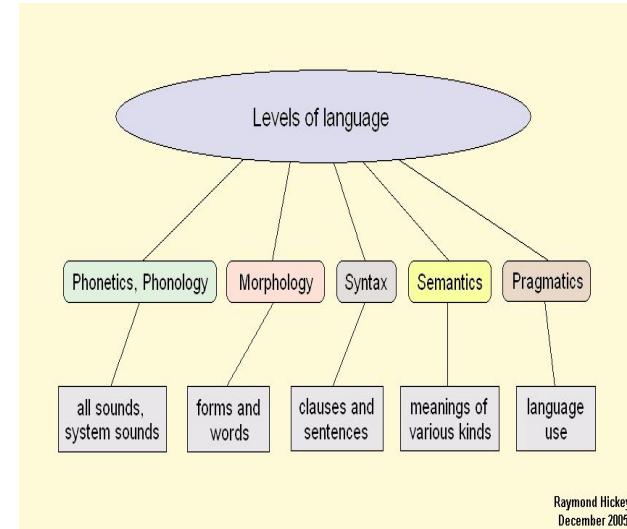
NLP Applications

In the Commercial World



NLP for Machines

- Analyze, understand and generate human languages just like humans do.
- Applying computational techniques to language domain.
- To explain linguistic theories, to use the theories to build systems that can be of social use.
- Started off as a branch of Artificial Intelligence.
- Borrows from Linguistics, Psycholinguistics, Cognitive Science & Statistics.
- Make computers learn our language rather than we learn theirs.



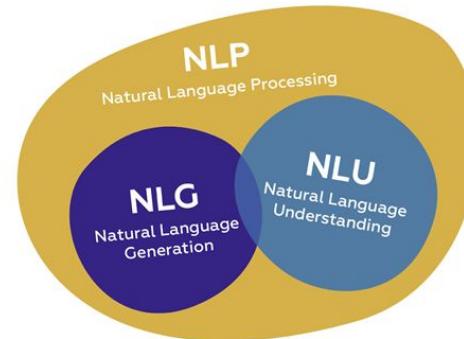
Components of NLP

- **Natural Language Understanding**

Taking some spoken/typed sentence and working out what it means

- **Natural Language Generation**

Taking some formal representation of what you want to say and working out a way to express it in a natural (human) language (e.g., English)



Cyber Security



Cyber Security

- Misinformation is a greater concern than other online security risks like cyber bullying or fraud schemes, according to Lloyd's Register Foundation World Risk Poll.
- The online poll, which conducted more than 150,000 interviews across 142 countries, found that 57% of internet users believe fake news was the biggest threat, followed by online fraud (45%), and cyberbullying (30%).
- concerns about cyberbullying are high in low-income economies, with young internet users more likely to worry about online bullying than older users.

Cyber Security

- Cyber security is the protection of computer systems and networks from information disclosure, theft of or damage to their hardware, software, or **electronic data**, as well as from the disruption or misdirection of the services they provide.
- It's also known as information technology security or electronic information security.



Categories of Cyber security

- Network security.
- Application security
- Information security
- Operational security

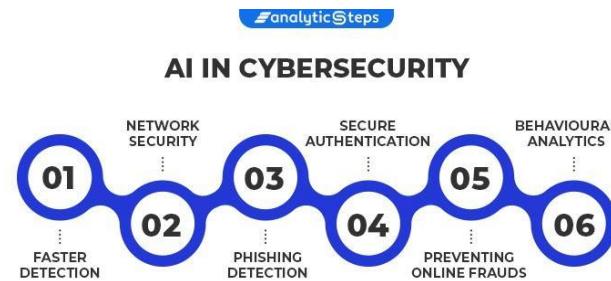


Relation Between AI-NLP-Cyber Security



AI-NLP-Cyber Security

- AI-NLP works collaboratively to handle the cyber related issues like ***hate speech detection, fake news detection, counter terrorism, spam email detection*** etc.
- AI systems can be trained to generate alerts for threats, identify new types of malware and protect sensitive data for organisations.
- AI help to automate the fraud detection system.
- AI provide the safety to social platform like facebook, instagram, twitter etc.
- AI application help to maintain the societal balance by identifying the unethical messages, acts, etc.



Machine Learning

- **Machine learning (ML)** is the study of computer algorithms that can improve automatically through experience and by the use of data.
- Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so.
- Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

Rule based vs Machine Learning

Rule-based systems and machine learning models are widely utilized to ***make conclusions from data.***

- **Rule-based system:** The two major components of rule-based artificial intelligence models are “a set of rules” and “a set of facts”. You can develop a basic artificial intelligence model with the help of these two components.
 - Deterministic models
- **Machine Learning:** The machine learning system defines its own set of rules that are based on data outputs.
 - Probabilistic models

Why Machine Learning?

- Develop systems that can automatically adapt and customize themselves to individual users.
 - Personalized news or mail filter
- Discover new knowledge from large databases (data mining).
 - Market basket analysis (e.g. diapers and beer)
- Ability to mimic human and replace certain monotonous tasks - which require some intelligence.
 - like recognizing handwritten characters
- Develop systems that are too difficult/expensive to construct manually because they require specific detailed skills or knowledge tuned to a specific task (knowledge engineering bottleneck)

Terminology

- **Features:**
The number of features or distinct traits that can be used to describe each item in a quantitative manner.
- **Samples:**
A sample is an item to process (e.g. classify). It can be a document, a picture, a sound, a video, a row in database or CSV file, or whatever you can describe with a fixed set of quantitative traits.
- **Feature vector:**
is an n-dimensional vector of numerical features that represent some object.
- **Feature extraction:**
Preparation of feature vector.
Transforms the data in the high-dimensional space to a space of fewer dimensions.
- **Training/Evolution set:**
Set of data to discover potentially predictive relationships.

Learning (Training)



Features:

1. Color: Radish/Red
2. Type : Fruit
3. Shape etc...



Features:

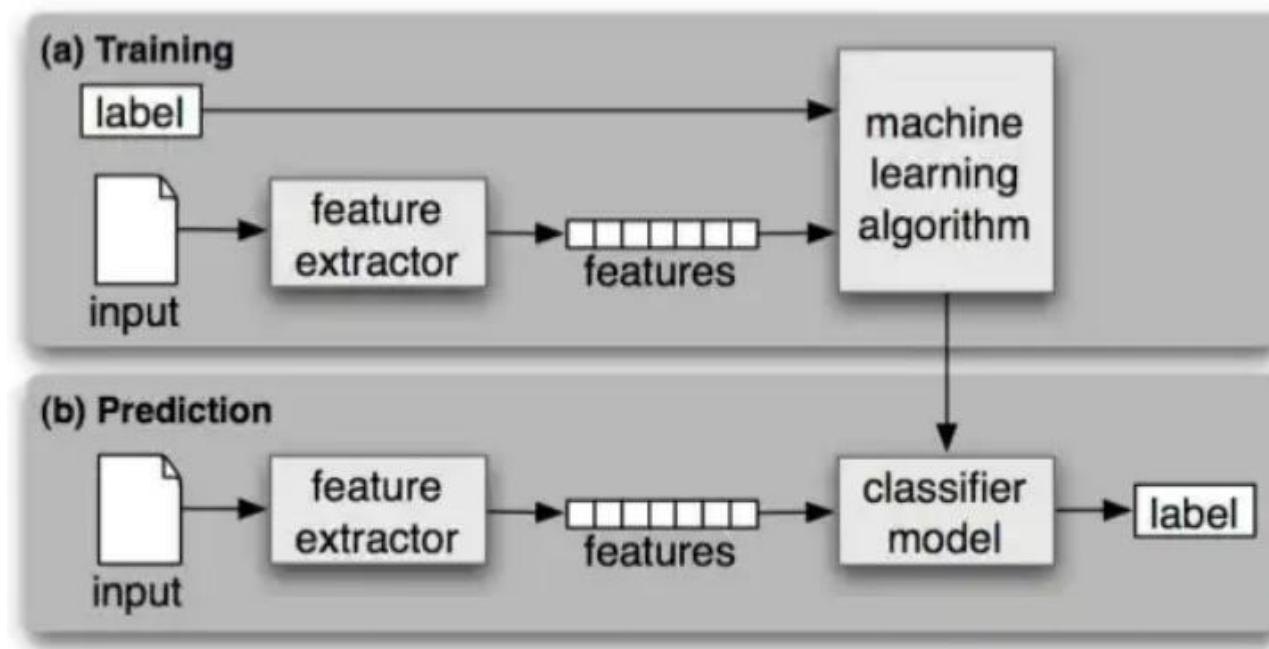
1. Black
2. Logo
3. Shape etc...



Features:

1. Yellow
2. Fruit
3. Shape etc...

Workflow

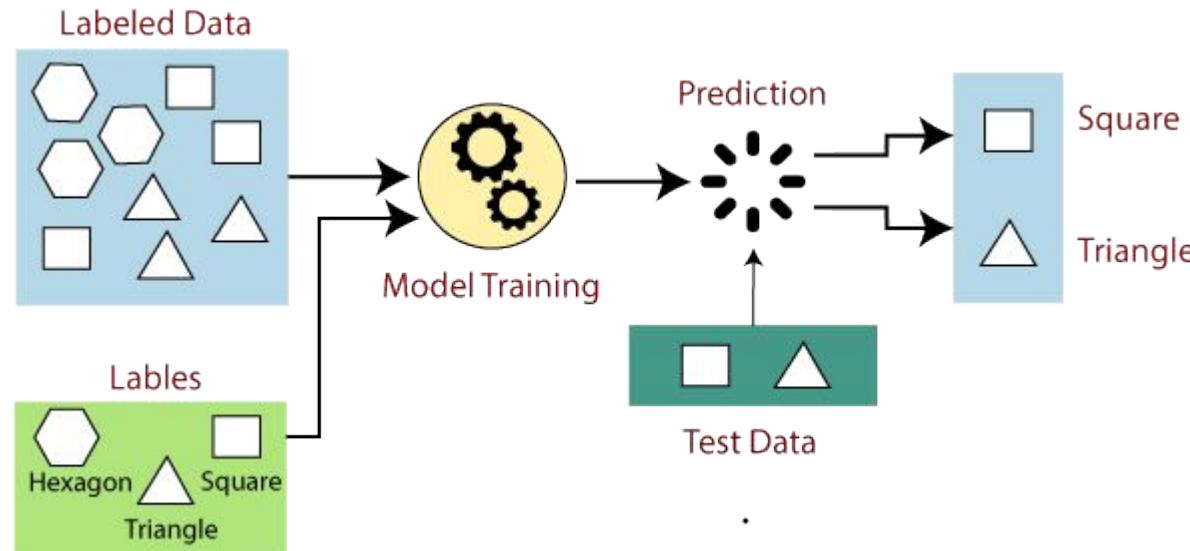


Types of machine learning

- Supervised Learning
- Unsupervised Learning
- Semi-Supervised Learning
- Reinforcement Learning

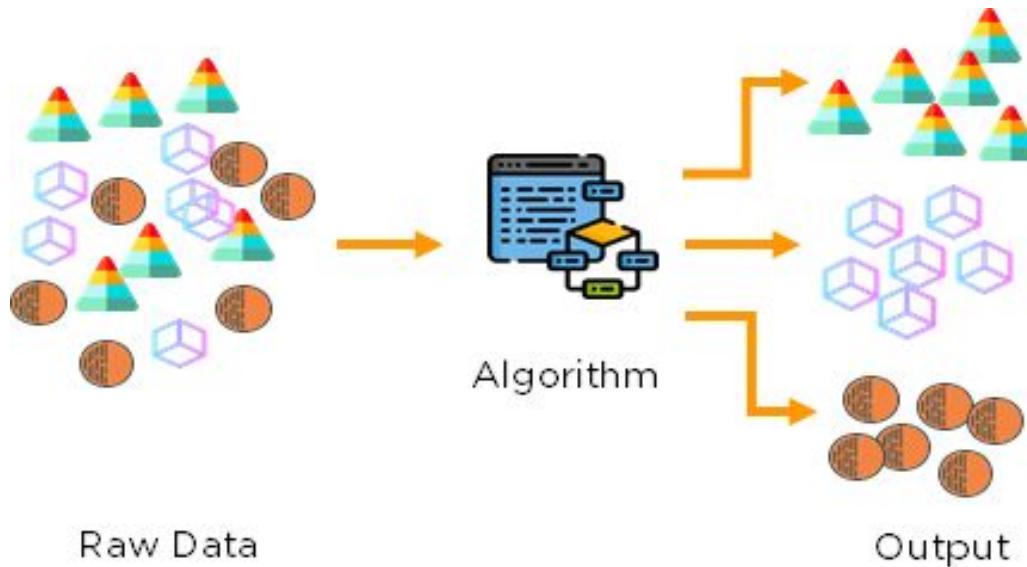
Supervised Learning

- The correct classes of the training data are known (labels are available).

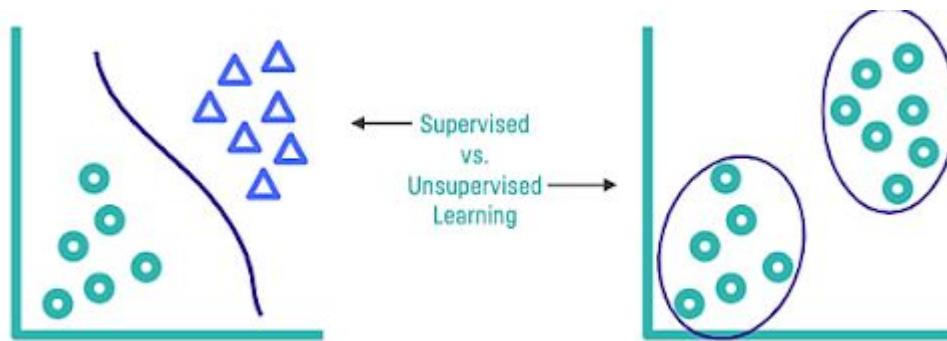


Unsupervised Learning

- The correct classes of the training data are not known (labels are not available).

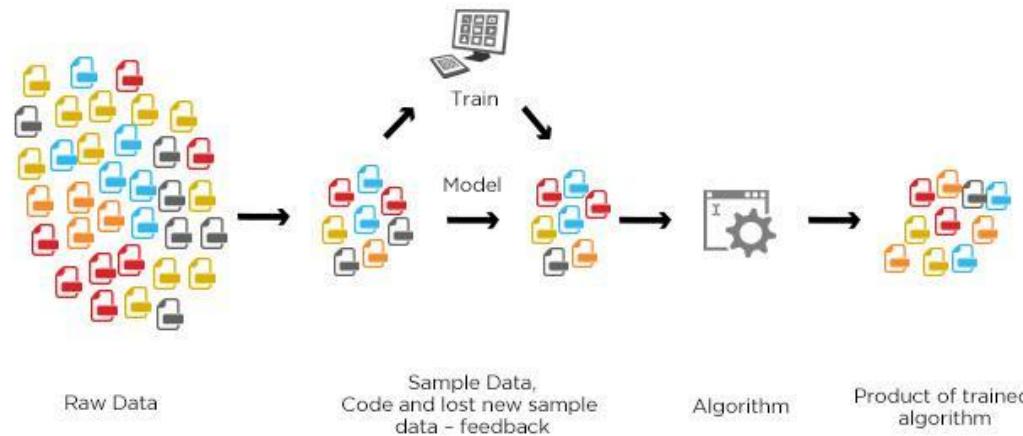


Supervised Vs Unsupervised Learning



Semi-Supervised Learning

- A Mix of Supervised and Unsupervised learning

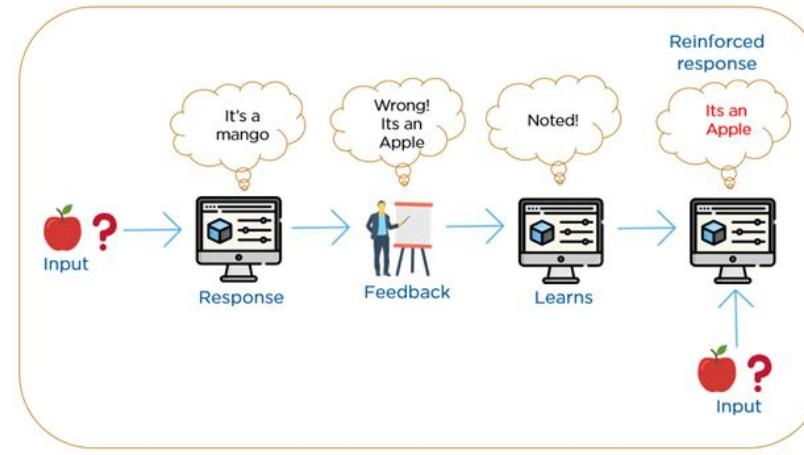


Example:

Internet Content Classification (Labeling each webpage is an impractical and unfeasible process and thus uses Semi-Supervised learning algorithms.)

Reinforcement Learning

- Allows the machine or software agent to learn its behavior based on feedback from the environment.



Reinforcement learning is all about making decisions sequentially. In simple words, we can say that the output depends on the state of the current input and the next input depends on the output of the previous input.

Example: Chess Game

Machine Learning Techniques

classification: predict class from observations

clustering: group observations into “meaningful” groups

regression (prediction): predict value from observations

ML in Societal Issues

- **Fact-Checking:** News article assessment, Fake news detection
- **Enabling Better Education:** Mathematical Information Retrieval, Math Plagiarism Detection
- **Mental Illness:** Chatbot-Based Counselling
- **Epidemics And Outbreaks:** Healthcare Chatbot
- **Agriculture:** Weather Prediction, Fertilizers recommendation
- **Language Translation**
- **Personalized digital media:** Siri, Cortana, and Google Now
- **Smart home and home security:** Digital assistants like Amazon Echo and Alexa allow for voice-activated control of our smart home(dimming light, locking the door, etc at our command)
- **Healthcare:** ML programs can predict health problems based on age, socioeconomic status, and genetic history which helps prevent illness.

Social Media



Social Media

- **Social media** enables people to communicate, share, and seek information at an accelerated rate.
- In recent years, **social media** became the pinnacle of news consumption through its rapid dissemination, low costs, and its accessibility to consumers worldwide.

Often breaking and sensitive news is first made available on social media.

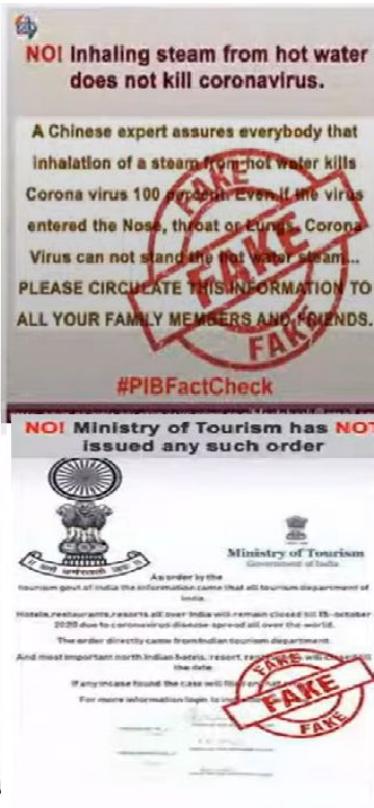
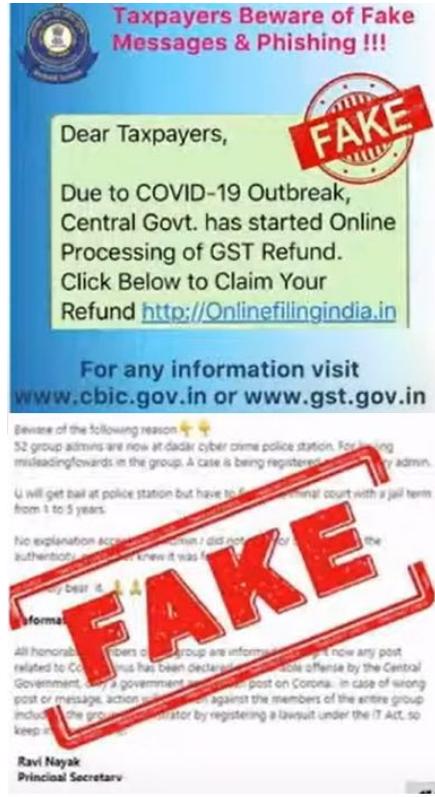
Social media provides users the ability to exchange thoughts and ideas with people from corners of the worlds they might not have visited, enables strangers to collaborate and positively impact our collective society, and increase awareness to help grow businesses, communities etc.

Social Media Threats Social Media Misinformation Campaigns and Measures to Fact-Check

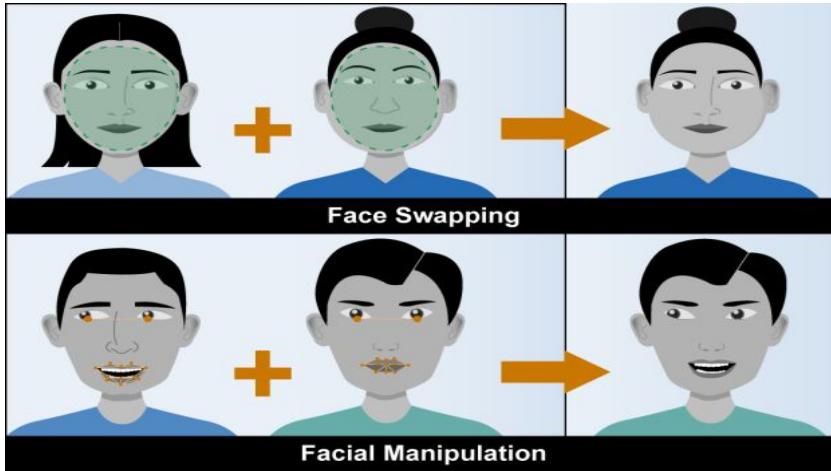
- Quick dissemination and viral posts allow adversaries to spread misinformation and “fake news” through fake accounts, bots, big data, and trolls to create circumstances to benefit their agendas.
- In 2016, “**fake news**” emanated on social media as the deliberate presentation of typically misleading or false news claims.

Pennycook, G., & Rand, D. (2019, February 12). Fighting misinformation on social media using crowdsourced judgments of news source quality. Retrieved September 30, 2020, from <https://www.pnas.org/content/116/7/2521>

Fake News on the name of Govt officials



Deepfake



DeepFake Cyber Security Threats And Opportunity - Matt Lewis

Deepfakes evolved the past couple of years as a subset of Artificial Intelligence (AI) that leverages neural networks to manipulate videos and photos while maintaining an authentic presence.

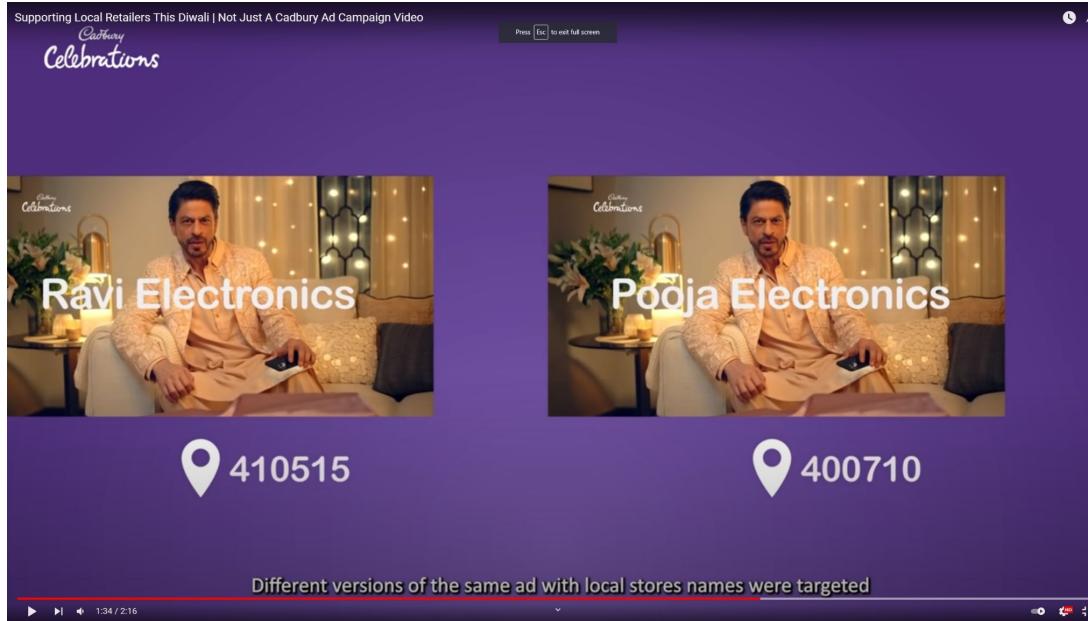
The screenshot shows a BBC News article titled "'Dangerous' AI offers to write fake news'. The article is by Jane Wakefield, Technology reporter, and was published on 27 August 2019. It has 1 comment. The main image shows a close-up of a person's hands typing on a laptop keyboard. To the right of the article, there is a sidebar titled 'Top Stories' with three news items: 'Pfizer says booster shot promising against Omicron' (published 9 hours ago), 'Ex-boyfriend of Maxwell accuser backs up statement' (published 3 hours ago), and 'How Myanmar military uses torture to suppress women' (published 1 hour ago). At the bottom right, there is an advertisement for SABIC with the text 'HOW DO WE BUILD A SUSTAINABLE ENERGY FUTURE?'.

Taulli, T. (2019, August 12). Deepfake: What You Need to Know. Retrieved September 30, 2020, from <https://www.forbes.com/sites/tomtaulli/2019/06/15/deepfake-what-you-need-to-know/>

Deepfake - As per wiki

- Deepfakes (stemming from "deep learning" and "fake") are synthetic media in which a person in an existing image or video is replaced with someone else's likeness.
- While the act of faking content is not new, deepfakes leverage powerful techniques from machine learning and artificial intelligence to manipulate or generate visual and audio content with a high potential to deceive.
- The main machine learning methods used to create deepfakes are based on deep learning and involve training generative neural network architectures, such as autoencoder or generative adversarial networks (GANs).
- Lip-sync deepfakes refer to videos that are modified to make the mouth movements consistent with an audio recording. Puppet-master deepfakes include videos of a target person (puppet) who is animated following the facial expressions, eye and head movements of another person (master) sitting in front of a camera

Using deep learning Techniques



Link Here: <https://www.youtube.com/watch?v=5WECsbqAQSk>

Fake Dubbing

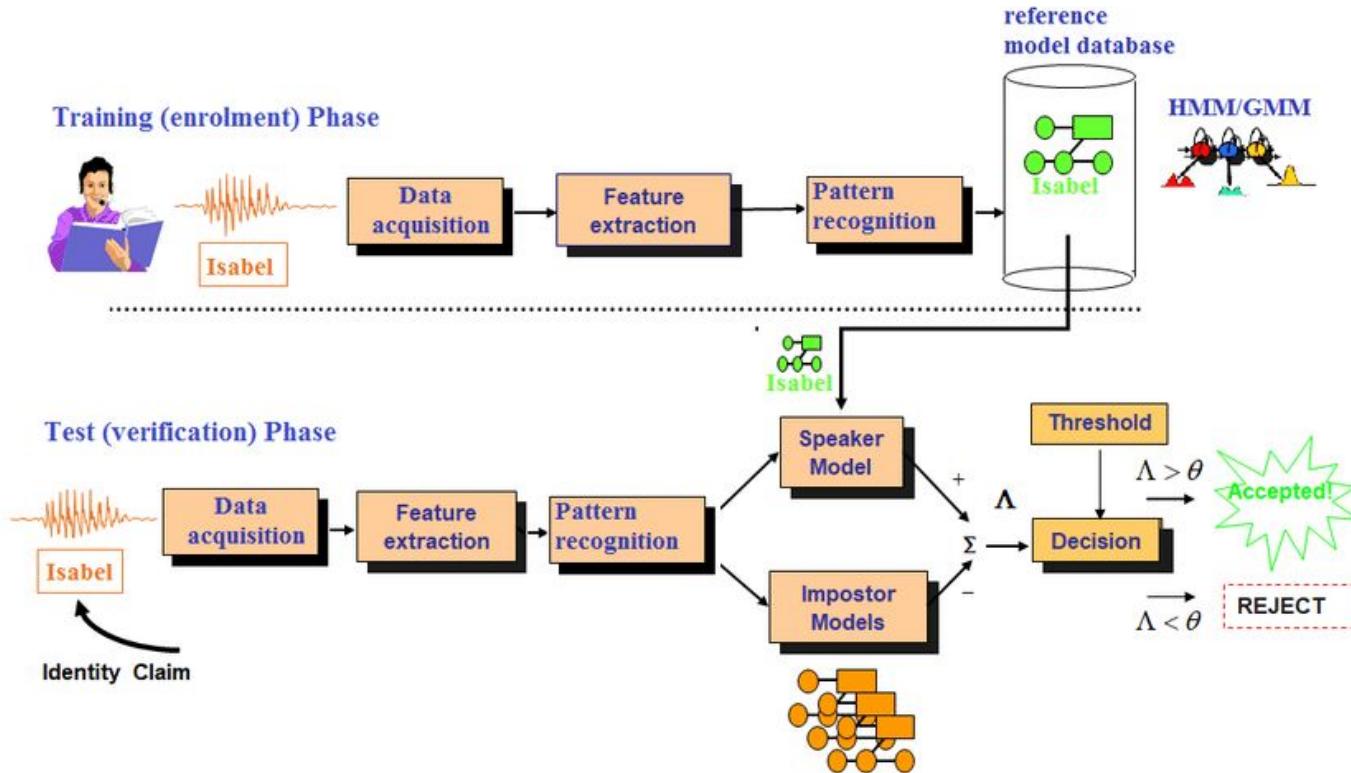
- **Dubbing**, in filmmaking, the process of adding new dialogue or other sounds to the soundtrack of a motion picture that has already been shot.
- Dubbing is much more active. Dubbing, also known as Language Replacement, substitutes a target language with each person appearing on screen.
- Deepfakes are a recent advancement in artificial synthetic media technology. Through deepfakes, one can easily create/synthesize a more natural fake voice of a person identical to the original voice.



Tools for voice generation

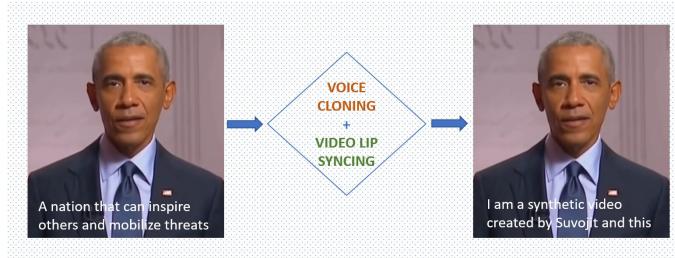
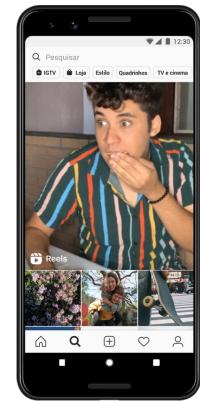
- Murf
- iSpring Suite
- Speechelo
- Notevibes
- Natural Reader
- Linguatec Voice Reader
- Capti Voice
- Voicedream
- Wideo

Basic Architecture



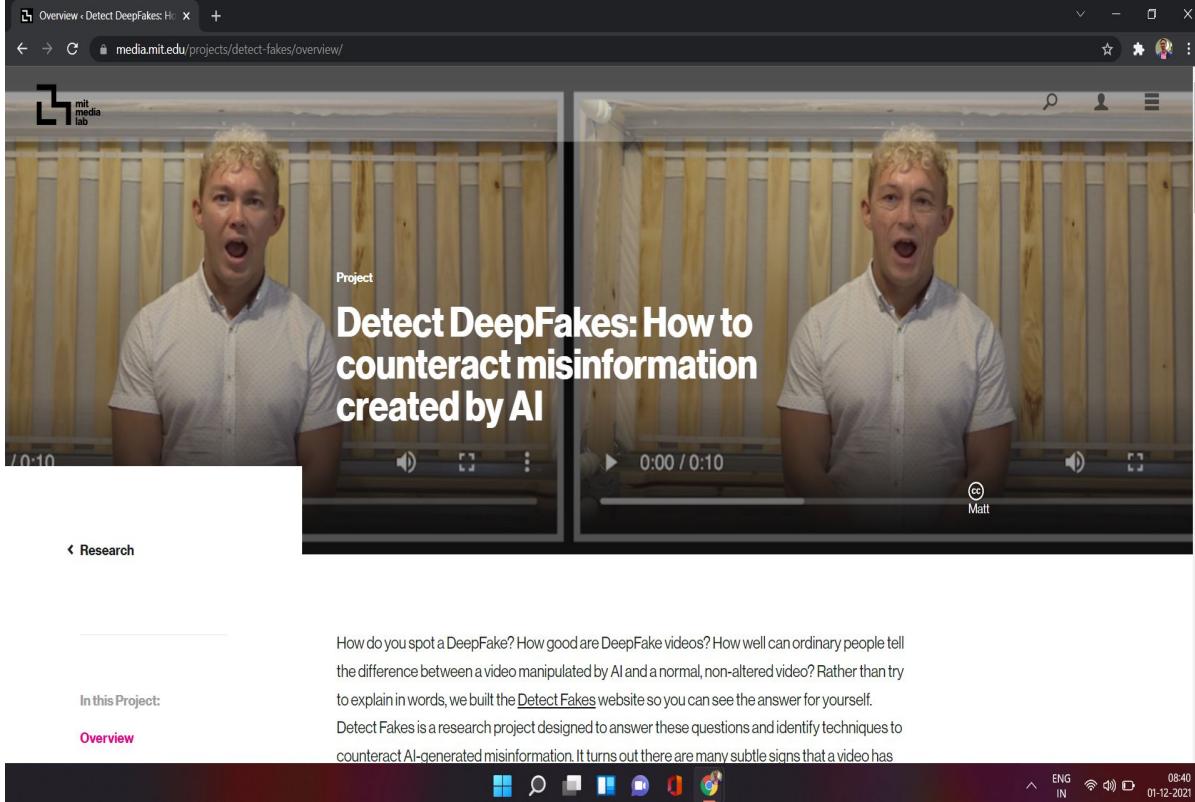
Fields of Dubbing/Fake budding

- Fake Audio-based news generation
- Fake lip-sync
- Filmmaking
- Reels generation like video on instagram, etc.



Identification of artificially generated deepfake content

How do you spot a DeepFake?



How good are DeepFake videos?

How well can ordinary people tell the difference between a video manipulated by AI and a normal, non-altered video?

Ref: <https://www.media.mit.edu/projects/detect-fakes/overview/>

Deepfake detection tool unveiled by Microsoft

By Leo Kelion
Technology desk editor

1 September 2020



Microsoft has developed a tool to spot deepfakes - computer-manipulated images in which one person's likeness has been used to replace that of another.

The software analyses photos and videos to give a confidence score about whether the material is likely to have been artificially created.

The firm says it hopes the tech will help "combat disinformation".

One expert has said it risks becoming quickly outdated because of the pace at which deepfake tech is advancing.

To address this, Microsoft has also announced a separate system to help content producers add hidden code to their footage so any subsequent changes can be easily flagged.

Top Stories

Multi-billion EU bid to challenge China
3 hours ago

Survivor haunted by deadliest Channel crossing
2 hours ago

CNN star anchor suspended over help to his brother
2 hours ago



Features



Kashmiri skier taking India to Beijing Olympics





manipulations in the future. As such, we hosted a website called [Detect Fakes](#) to display thousands of these curated, high-quality DeepFake and real videos publicly.

The Detect Fakes experiment offers the opportunity to learn more about DeepFakes and see how well you can discern real from fake. When it comes to AI-manipulated media, there's no single tell-tale sign of how to spot a fake. Nonetheless, there are several DeepFake artifacts that you can be on the look out for.

1. Pay attention to the face. High-end DeepFake manipulations are almost always facial transformations.
2. Pay attention to the cheeks and forehead. Does the skin appear too smooth or too wrinkly? Is the agedness of the skin similar to the agedness of the hair and eyes? DeepFakes are often incongruent on some dimensions.
3. Pay attention to the eyes and eyebrows. Do shadows appear in places that you would expect? DeepFakes often fail to fully represent the natural physics of a scene.
4. Pay attention to the glasses. Is there any glare? Is there too much glare? Does the angle of the glare change when the person moves? Once again, DeepFakes often fail to fully represent the natural physics of lighting.
5. Pay attention to the facial hair or lack thereof. Does this facial hair look real? DeepFakes might add or remove a mustache, sideburns, or beard. But, DeepFakes often fail to make facial hair transformations fully natural.
6. Pay attention to facial moles. Does the mole look real?
7. Pay attention to blinking. Does the person blink enough or too much?
8. Pay attention to the size and color of the lips. Does the size and color match the rest of the person's face?

These eight questions are intended to help guide people looking through DeepFakes. High-quality DeepFakes are not easy to discern, but with practice, people can build intuition for identifying what is fake and what is real. You can practice trying to detect DeepFakes at [Detect Fakes](#).



DeepFake Detection

[Edit](#)

32 papers with code • 3 benchmarks • 9 datasets

DeepFakes involves videos, often obscene, in which a face can be swapped with someone else's using neural networks.

DeepFakes are a general public concern, thus it's important to develop methods to detect them.

Description source: [DeepFakes: a New Threat to Face Recognition? Assessment and Detection](#)

Image source: [DeepFakes: a New Threat to Face Recognition? Assessment and Detection](#)



Benchmarks

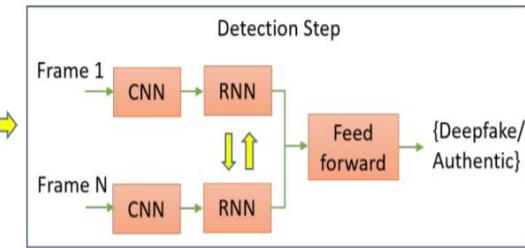
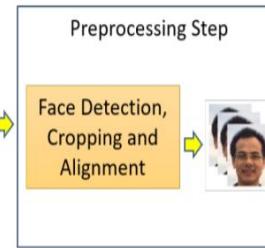
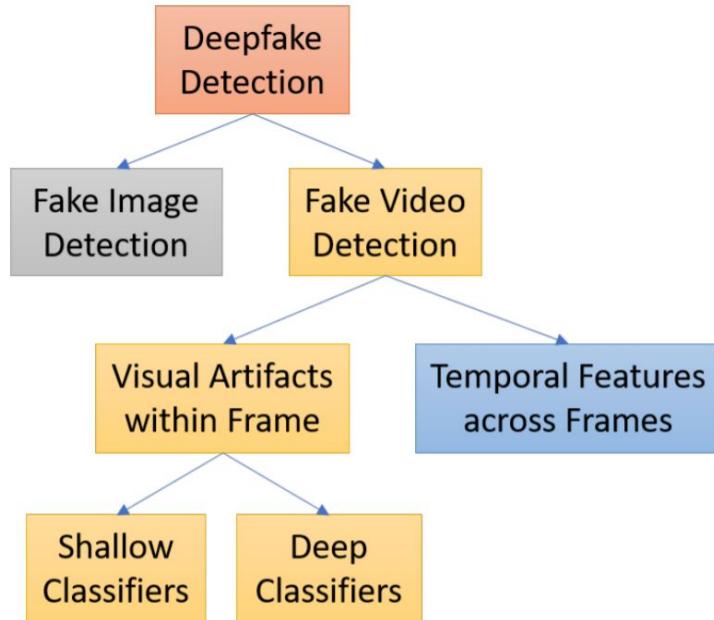
[Add a Result](#)

Trend	Dataset	Best Model	Paper Title	Paper	Code	Compare
	DFDC	Cross Efficient Vision Transformer	Combining EfficientNet and Vision Transformers for Video Deepfake Detection			See all
	FaceForensics++	EfficientNetB4 + EfficientNetB4ST + B4Att + B4AttST	Video Face Manipulation Detection Through Ensemble of CNNs			See all
	FaceForensics	XceptionNet	FaceForensics++: Learning to Detect Manipulated Facial Images			See all

Datasets



Nguyen et al (2021)



Social Media

Identification of terrorist profiles on various social media platforms

Analysing social media + interpol.int/en/Crimes/Terrorism/Analysing-social-media Incognito EN Search

INTERPOL Who we are Crimes How we work Our partners What you can do News Wanted persons

Identifying terrorist suspects
Preventing terrorist travel
Tracing terrorist finances
Analysing social media
+ Chemical and Explosives terrorism
+ Bioterrorism
+ Radiological and Nuclear terrorism
Partnerships against terrorism
+ Counter-terrorism projects

Terrorists use social media for radicalization, recruitment, funding, planning and execution of terror activities.

— Identifying suspects and witnesses

We analyse the terrorist use of social media platforms in order to enhance identification and detection efforts in national counter-terrorism investigations.

For example, we search social media platforms to identify potential witnesses, as was the case following the London Bridge attack in the UK in 2017, and the attack at a hotel complex in Nairobi, Kenya, in January 2019.

We are also exploring ways to use facial recognition technology to support member countries in their investigations. This technology offers new opportunities to share and compare data in order to identify terrorists, unknown persons of interest, and subjects who may appear in posts on social media channels.

— Investigative training and resources

As part of our first joint project with the United Nations Counter-Terrorism Centre (UNCCT), we ran workshops for investigators covering four main areas:

- detecting terrorist-related activities online;
- collecting electronic records;
- requesting e-evidence across borders through police-to-police cooperation;
- engaging with the private sector to advance investigations by law enforcement agencies.

The workshops (conducted in 2018 and 2019) also served to enhance participants' operational picture on the phenomenon of Foreign Terrorist Fighters (FTFs) more generally.

Complementing the workshops is a Handbook, jointly published by INTERPOL and UNCCT, entitled "Using the Internet and Social Media for Counter-Terrorism Investigations". It offers Investigators practical guidance on how best to generate online investigative leads and collect and preserve electronic records often across international borders to contribute to successful investigations and prosecutions.

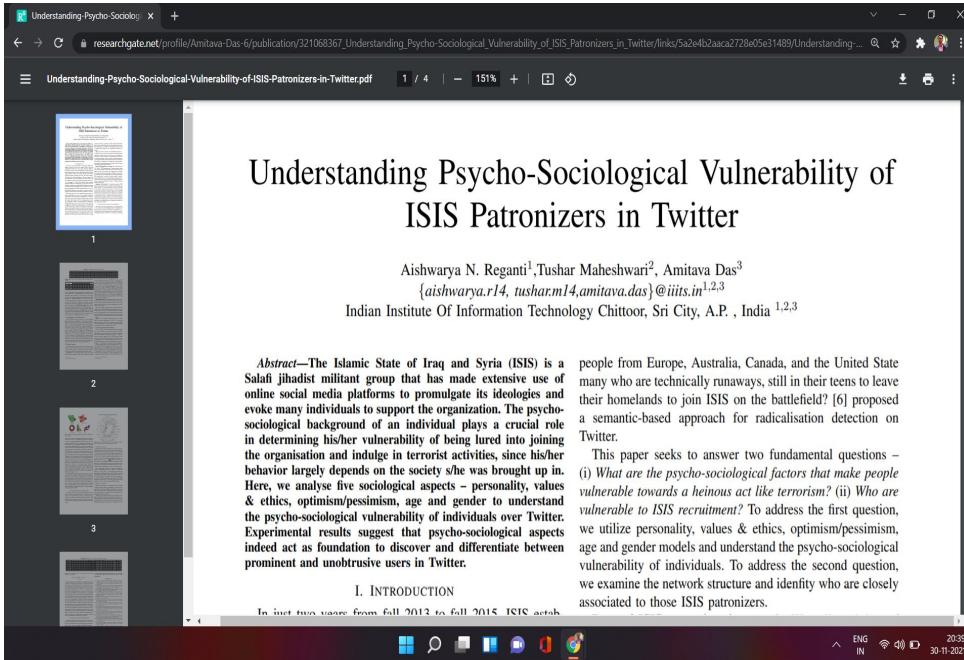
This project was funded by the Governments of Japan, Saudi Arabia, and the United Arab Emirates.

Our site uses cookies to ensure technical functionality, gather statistics and enable sharing on social media platforms. OK Tell me more X

9:02 ENG IN 01-12-2021

Identification of terrorist profiles on various social media platforms

Reganti et al (2017)



In this paper seeks to answer two fundamental questions:

- ***What are the psycho-sociological factors that make people vulnerable towards a heinous act like terrorism?***
- ***Who are vulnerable to ISIS recruitment?***

To address the first question, they utilized personality, values & ethics, optimism/pessimism, age and gender models and understand the psycho-sociological vulnerability of individuals.

To address the second question, they examined the network structure and identify who are closely associated to those ISIS patronizers.

Ahmad et al (2019)

Based on user-generated social media posts on Twitter, they have developed a tweet classification system using deep learning-based sentiment analysis techniques to classify the tweets as extremist or non-extremist.

Research questions:

RQ#1: *How to recognize and classify tweets as extremist vs non-extremist, by applying deep learning-based sentiment analysis techniques?*

RQ#2: *What is the performance of classical feature sets like n-grams, bag-of-words, TF-IDF, bag-of-words (BoW) over word embedding learned using CNN, LSTM, FastText, and GRU?*

RQ#3: *What is the performance of proposed technique for extremist affiliation classification with respect to the state-of-the-art methods?*

RQ#4: *How to perform the sentiment classification of user reviews w.r.t emotional affiliations of Extremists on Twitter and Deep Web?*

Abstraction-based text summarization - for investigating officers

Text summarization

- Text summarization refers to the technique of shortening long pieces of text.
- The intention is to create a coherent and fluent summary having only the main points outlined in the document.
- Summaries reduce reading time.
- When researching documents, summaries make the selection process easier.
- Automatic summarization improves the effectiveness of indexing.
- Automatic summarization algorithms are less biased than human summarizers.
- Personalized summaries are useful in question-answering systems as they provide personalized information.

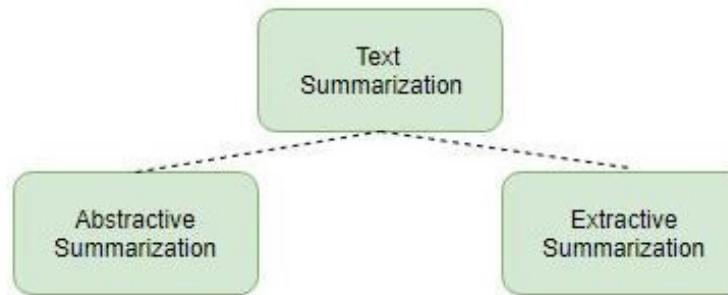
Summarization in Defence

- Help the officer to understand the long-range of information using summarizer.
- Provide the huge information in a summary format, so that officer can take the action in very less time instead of study of longe report.



Types of Text Summarization

1. **Extractive:** - In Extractive text summarization , summary is generated by selecting a set of words, phrases, paragraph or sentences from the original document.
2. **Abstractive:** - Abstractive methods are based on semantic representation and then use natural language processing techniques to generate a summary that is nearer to summary generated manually.

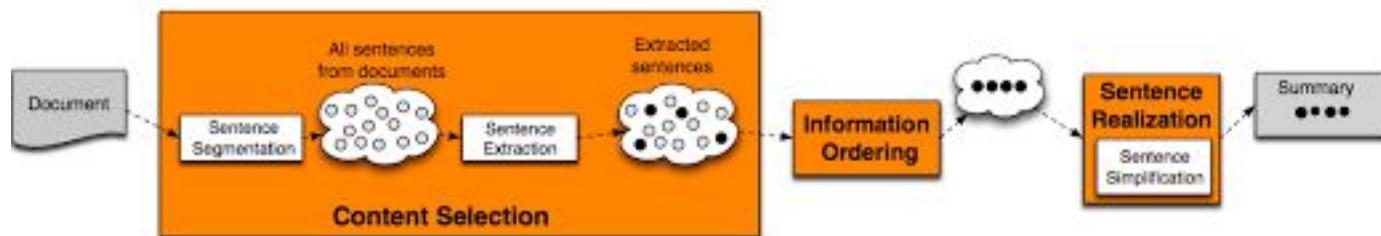


Summarization Stages

- **Content Selection:** Choose sentences to extract from the document Text

Summarization.

- Information Ordering: Choose an order to place them in the summary.
- Sentence Realization: Clean up the sentence.



Examples:

- **Resoomer**
- **Scholarcy**
- **QuillBot**
- **Text Compactor**
- **SummarizeBot**

RESOOMER

Servicio Extensiones Cómo funciona PREMIUM Contacto Inicio de sesión Español ▾

Extraiga lo esencial de su texto, resuma « [de forma pertinente](#) » en un clic.

Ejemplo de texto Borrar el texto Resoomer

Sólo textos argumentativos

Resumen: Automático Manual Optimizado Análisis Ayuda

Texto reducido al 33% (118 palabras / 402)

Una gatita lanzada desde un coche en la ronda de circunvalación de TOULOUSE

La asociación protectora de animales que acogió al pequeño felino herido ha dado la voz de alarma ante el aumento del número de mascotas abandonadas durante el verano.

Un acto incomprensible y cruel. Una gatita de tres meses de edad, llamada Nounette, fue lanzada voluntariamente desde un coche en marcha a finales de julio en la ronda de circunvalación de Toulouse, informa la emisora France Bleu. Una asociación protectora de animales locales, Cha'Mania, ha difundido la historia, para alertar del creciente número de abandonos de animales durante el verano.

Los hechos ocurrieron en la ronda de circunvalación de Toulouse, el 27 de Julio. Después de deshacerse violentamente del animal mientras conducía, el conductor dio media vuelta y desapareció. Por su parte, Nounette estuvo a punto de morir. Fue atropellada por otro vehículo y encontrada gravemente herida en un arbusto por un testigo de Los hechos, que la llevó a las instalaciones de Cha'Mania para que recibiera tratamiento.

«Trasladada a un veterinario, se descubrió que la gatita tenía tres fracturas en la pelvis. Se le prescribieron antibióticos y se le aplicó una técnica de jaula-terapia, con la esperanza de que se recuperase», añadió la emisora de radio.

«Un animal no es un mueble»

Enfurecidos, los miembros de la asociación quieren utilizar este suceso para concienciar al público en general sobre el abandono y el abuso de las mascotas, un fenómeno que les parece demasiado común. «Es fácil llamar a una asociación, a un refugio y decir que ya no quieres a tu mascota por las razones que sean, pero de ahí a tirarla a la carretera desde una ventanilla de un coche en marcha... La falta de respeto por el animal es evidente», declaró a France Bleu Brigitte Marechaux, una voluntaria del centro. Esta mujer fue quien acogió a la gatita hasta que se recuperó. Ahora será propuesta para adopción.

Parafrasear Volver a escribir Traducir PDF Doc Copiar

↑

Cyberbullying

Cyberbullying - as per wikipedia

- Cyberbullying or cyberharassment is a form of bullying or harassment using electronic means. Cyberbullying and cyberharassment are also known as online bullying.
- It has become increasingly common, especially among teenagers, as the digital sphere has expanded and technology has advanced.
- Cyberbullying is when someone, typically a teenager, bullies or harasses others on the internet and other digital spaces, particularly on social media sites. Harmful bullying behavior can include posting rumors, threats, sexual remarks, a victims' personal information, or hate speech.
- Bullying or harassment can be identified by repeated behavior and an intent to harm.
- Victims of cyberbullying may experience lower self-esteem, increased suicidal ideation, and various negative emotional responses, including being scared, frustrated, angry, or depressed

Cyberbullying can include:

- **Sending mean texts or IMs** to someone
- **Pranking** someone's cell phone
- **Hacking** into someone's gaming or social networking profile
- **Being rude or mean** to someone in an online game
- **Spreading secrets or rumours** about people online
- **Pretending to be someone else** to spread hurtful messages online

Dataset

The screenshot shows a web browser window with the title 'Fighting bullying with machine learning' and the URL 'research.cs.wisc.edu/bullying/data.html'. The page content is as follows:

Data and code for the study of bullying

This page contains our data sets and code release for the scientific research of bullying.

Bullying Traces Data Set

- **Version 3.0:** [bullyingV3.o.zip](#) (size 534950, released in June 2015). 7321 tweets with tweet ID, bullying, author role, teasing, type, form, and emotion labels.
 - This version was described in: Junming Sui. *Understanding and Fighting Bullying with Machine Learning*. PhD thesis, Department of Computer Sciences, University of Wisconsin-Madison. 2015.
- (Archived version) [bullyingV2.o.zip](#) (size 217680, released in September 2014). 1762 tweets with tweet ID, bullying, author role, and teasing labels.
- (Archived version) [bullyingV1.zip](#) (size 19141, released in April 2012). Same tweets as in V2.0 but without tweet IDs.
 - Versions 1.0 and 2.0 of this data set were introduced in the paper: Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. *Learning from bullying traces in social media*. Proceedings of NAACL HLT 2012.

Code to Recognize Bullying Traces

- [bullyingtraceV2.zip](#) (size 425383, released in April 2016)
 - Java source and ready-to-use jar files for performing the following classification tasks on input tweets:
 - classify input text as bullying trace or not; and if yes:
 - classify the tweet author's role in a bullying event
 - classify the tweet as teasing or not
 - classify the type, form and sentiment of the tweet.
 - The classifiers in this version are SVMs with linear kernel, and are trained on Bullying Traces Data V3.0 (see above). For more information about its usage, please refer to the README file inside the zip file.
- (Archived version) [bullyingtraceV1.zip](#) (size 156181, released in June 2012)
 - Java source code and ready-to-use jar files to classify text input as bullying traces or not.
 - This classification task was introduced as "Task A" in the paper: Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. *Learning from bullying traces in social media*. Proceedings of NAACL HLT 2012.

Use Agreement

Access to the data and code files below is conditioned upon your agreement to this [agreement](#).

Windows taskbar icons include: File Explorer, Search, Task View, Start, Microsoft Edge, Google Chrome, File History, Task Scheduler, and File Explorer (second instance).

System tray icons: Battery (21:47), Network (ENG IN), Wi-Fi, Volume, and Date/Time (02-12-2021).

CodeMixin Data

- “**Hinglish**” - Hindi sentence could be written using English alphabets.
- For example, “main aap ke sath kal milenge”, as well as “कल I will meet you” or “Kal I will meet you” wherein both Hindi and English have been used.
- Trace and track combined lingo statements using NLP/ NLU programmed from scratch or auto-populated library of words using own datasets.

Cyberbullying detection

Shrikant et al. (2019)

- Created the code-mixed dataset (Hinglish Cyberbullying Comments (HCC) labeled) consisting of comments from social media networks such as Instagram and YouTube.

Model	Accuracy	Precision	Recall	F1-score
Logistic Regression	76.39%	70.66%	95.16%	81.10%
Linear SVC	80.26%	77.46%	88.71%	82.71%
Multinomial NB	79.83%	74.84%	93.55%	83.15%
Passive Aggressive Classifier	78.97%	76.98%	86.29%	81.37%
Bernoulli NB	56.22%	54.91%	99.19%	70.69%
Perceptron	76.82%	78.23%	78.23%	78.23%
Adaboost Classifier	74.25%	68.60%	95.16%	79.73%
Decision Tree Classifier	78.11%	73.86%	91.13%	81.59%
Hybrid Model	80.26%	76.71%	90.32%	82.96%



[International Conference on Advances in Computing and Data Sciences](#)
[ICACDS 2019: Advances in Computing and Data Sciences](#) pp 543-551 | [Cite as](#)

Cyberbullying Detection in Hindi-English Code-Mixed Language Using Sentiment Classification

Authors

Shrikant Tarwani, Manan Jethanandani, Vibhor Kant

Conference paper

First Online: 19 July 2019

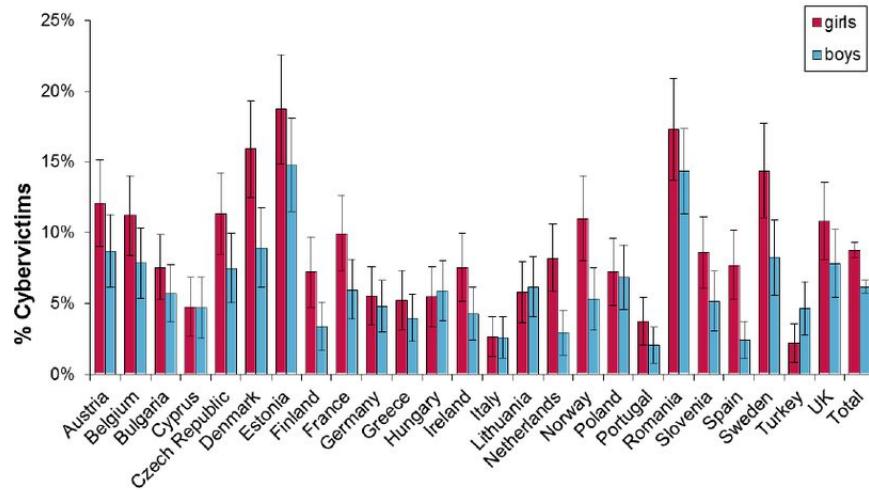
5
Citations
973
Downloads

Part of the [Communications in Computer and Information Science](#) book series (CCIS, volume 1046)

Abstract

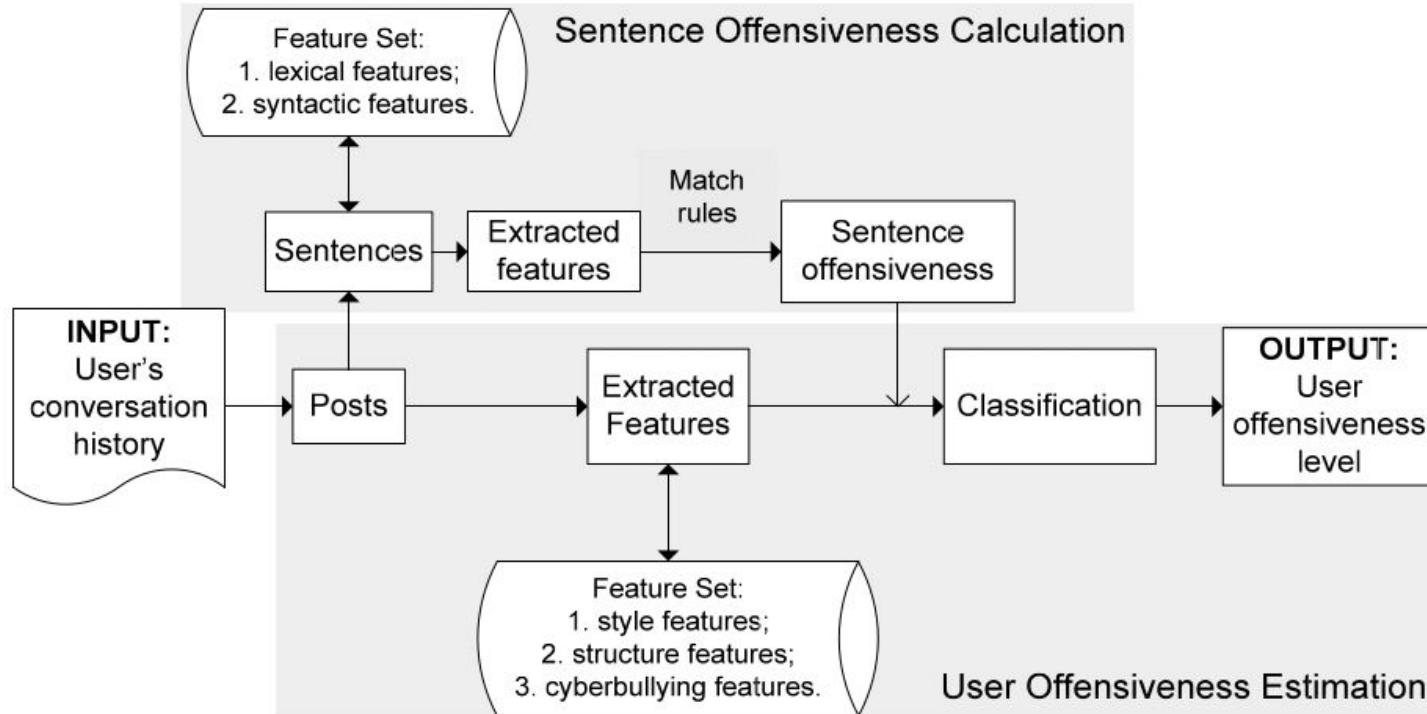
Cyberbullying is one of the radical emerging problems with the advancements in the Internet, connecting people around the globe by social media networks. Existing studies mostly focus only on cyberbullying detection in the English language, thus the main objective of this paper is to develop an approach to detect cyberbullying in Hindi-English code-mixed language (Hinglish) which is exorbitantly used by Indian users. Due to the unavailability of Hinglish dataset, we created the Hinglish Cyberbullying Comments (HCC) labeled dataset consisting of comments from social media networks such as Instagram and YouTube. We also developed eight different machine learning models for sentiment classification in order to automatically detect incidents of cyberbullying. Performance measures namely accuracy, precision, recall and f1 score are used to evaluate these models. Eventually, a hybrid model is developed based on top performers of these eight baseline classifiers which perform better with an accuracy of 80.26% and f1-score of 82.96%.

Percentage of teens around the world reporting being bullied / according to countries



Görzig, Anke, and Hana Machackova. "Cyberbullying from a socio-ecological perspective: a contemporary synthesis of findings from EU Kids Online." (2015): 37-37.

Basic Architecture

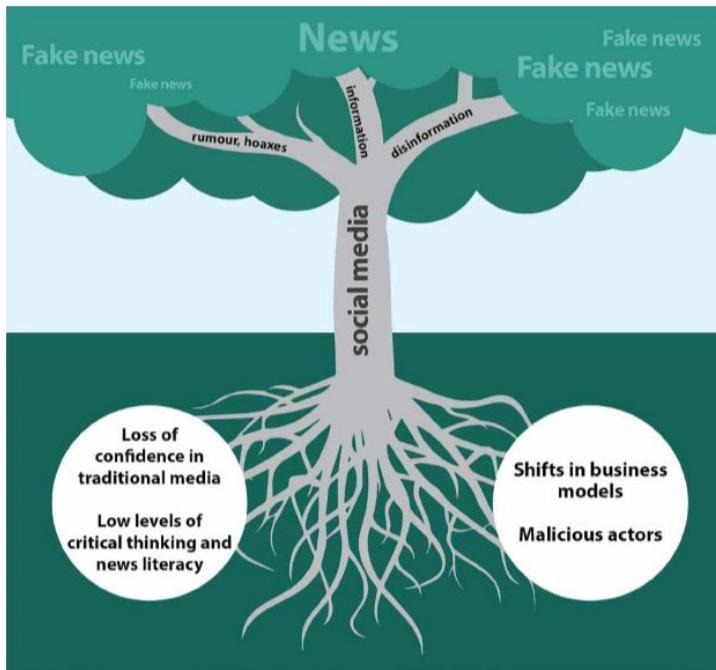


How to help prevent cyberbullying?

- Block all communication with cyberbullies
- Do not forward any messages, comments, etc. that involve cyberbullying
- Always report any cyberbullying taking place to an adult
- Educate your students about the damages of cyberbullying

Fake News / Rumor Detection

roots of fake news by UNESCO



Fake News / Rumor Detection

- Determining if a story or online post is a rumor or non-rumor (i.e. a real story, a news article)
- Task of determining the veracity of a rumor (true, false or unverified) is defined as rumor verification

Fake News Detection

- Participated in the 2020 Fake News Detection Challenge in the Urdu Language organized by FIRE 2020 & Center for Computing Research (CIC), Instituto Politécnico Nacional (IPN), Mexico and have stood second.
- Used XLNet based on generalized autoregressive pre-training for language understanding, to train model for classification of news: fake or real.
- Achieved an overall accuracy of 0.8400 and F1 macro score of 0.8370.

<http://fire.irsi.res.in/fire/2020/home>

Urdu Fake News Detection using Generalized Autoregressors

Abdullah Faiz Ur Rahman Khilji^a, Sahinur Rahman Laskar^a, Partha Pakray^a and Sivaji Bandyopadhyay^a

^aDepartment of Computer Science and Engineering, National Institute of Technology Silchar, Assam, India

Abstract

Automatic fake news detection has become vital in today's digital age. Differentiating legit news from fake ones has become an important classification challenge in natural language processing (NLP). Various transformer-based deep learning approaches have taken widespread adoption from the research community due to its outstanding performance. We have participated in the 2020 Fake News Detection Challenge in the Urdu Language organized by Center for Computing Research (CIC), Instituto Politécnico Nacional (IPN), Mexico and have stood second. In this work, we have implemented a generalized autoregressor based model to classify news into fake or real. We have achieved an overall accuracy of 0.8400 and F1 macro score of 0.8370.

Keywords

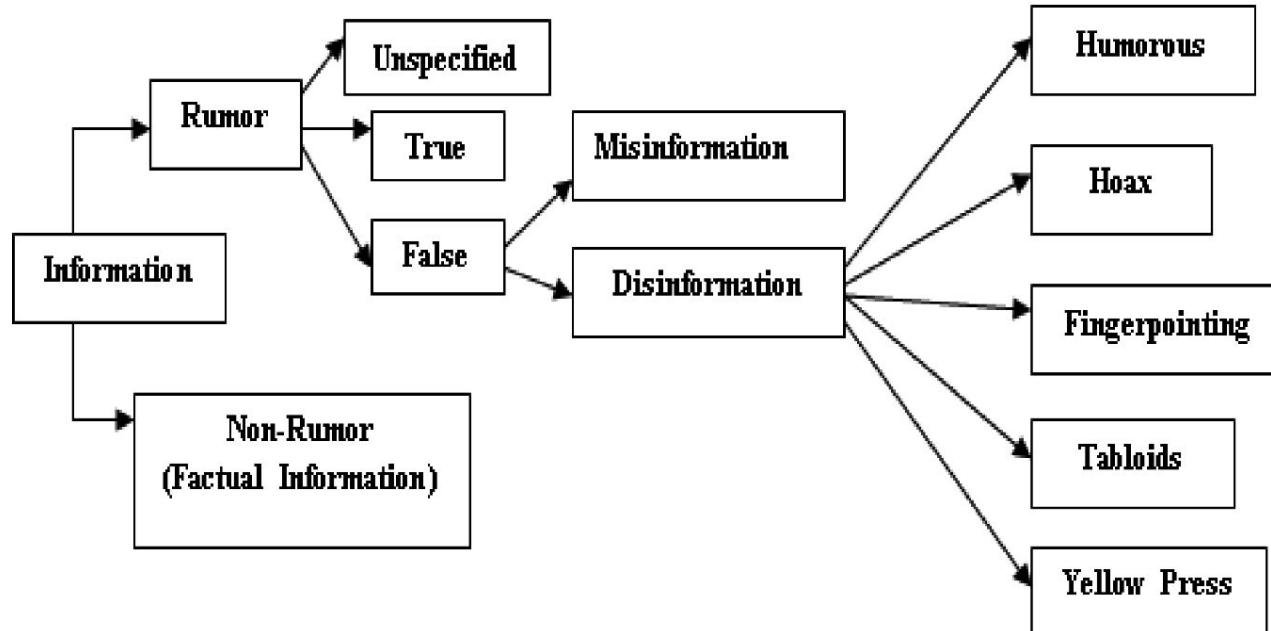
Fake News Detection, Classification, Autoregressors, XLNet

1. Introduction

The age of the internet has led to a massive free flow of information. This information flow has an added advantage of being easily accessible through widespread virtualization, leading to cost benefits to humankind. This increased accessibility has led to widespread usage and adaptability of the services it has to offer, to the point that one's principal source of information becomes the internet. Due to its ubiquitous and global nature, it is quite difficult to have a central authority to have control over it, without defeating the true purpose it has to offer. As we are moving more and more of our life to the virtual world, the importance of detecting fake news has increased manifold as it has proved to have a negative impact on our society [1]. There are various instances where fake news has been used intentionally [2] to spread misinformation. This deliberate attempt has amplified in the era of social media where any individual is ready to express their views without considering any facts, one such example is given by [3]. It is also shown that the spread of fake news is exponential [3] and any initial attempts could greatly help in curtailing the issue. Hence, there is a need to automate fake

Forum for Information Retrieval Evaluation 2020, December 16-20, 2020, Hyderabad, India
EMAIL: abdullah_ur_rahman_khilji@csse.nits.ac.in (A. F. U. R. Khilji); sahinur_ug@csse.nits.ac.in (S. R. Laskar); partha@cse.nits.ac.in (P. Pakray); sivaji.csse19@gmail.com (S. Bandyopadhyay)
URL: <https://abdullahkhilji.github.io/> (A. F. U. R. Khilji); <https://sahnurlaskar.github.io/> (S. R. Laskar); <http://cs.nits.ac.in/partha/> (P. Pakray)
ORCID: 0000-0001-6621-1810 (A. F. U. R. Khilji); 0000-0002-8413-2718 (S. R. Laskar); 0000-0003-3834-5154 (P. Pakray)
© 2020 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution +0 International (CC BY 4.0).
CEUR - Wschr. 2666, pp. 1–10, 2020. CEUR - Wschr. 2666, December 16-20, 2020, Hyderabad, India

Rumor Detection



Rumor detection on Social Network Platform

Aoshuang et al. (2021)

- Investigated the propagation pattern of social events that do not depend on feature engineering and domain knowledge, which overcome the limitation that the propagation pattern features are difficult to be structured as input for general machine learning models.
- Shows propagation pattern features can effectively detect rumors by using convolutional neural networks (CNNs) and recurrent neural networks (RNNs).
- Designed a feature aggregation model based on DNN to exploit the aggregated feature combined by propagation pattern feature and text content feature

Hindawi
Complexity
Volume 2021, Article ID 6659430, 16 pages
<https://doi.org/10.1155/2021/6659430>



Research Article

An End-to-End Rumor Detection Model Based on Feature Aggregation

Aoshuang Ye ^{1,2} Lina Wang ^{1,2} Run Wang ^{1,2} Wenqi Wang,^{1,2} Jianpeng Ke,^{1,2} and Danlei Wang^{1,2}

¹*Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, Wuhan, Hubei, China*

²*School of Cyber Science and Engineering, Wuhan University, Wuhan, Hubei, China*

Correspondence should be addressed to Lina Wang; lnwang@whu.edu.cn and Run Wang; wangrun@whu.edu.cn

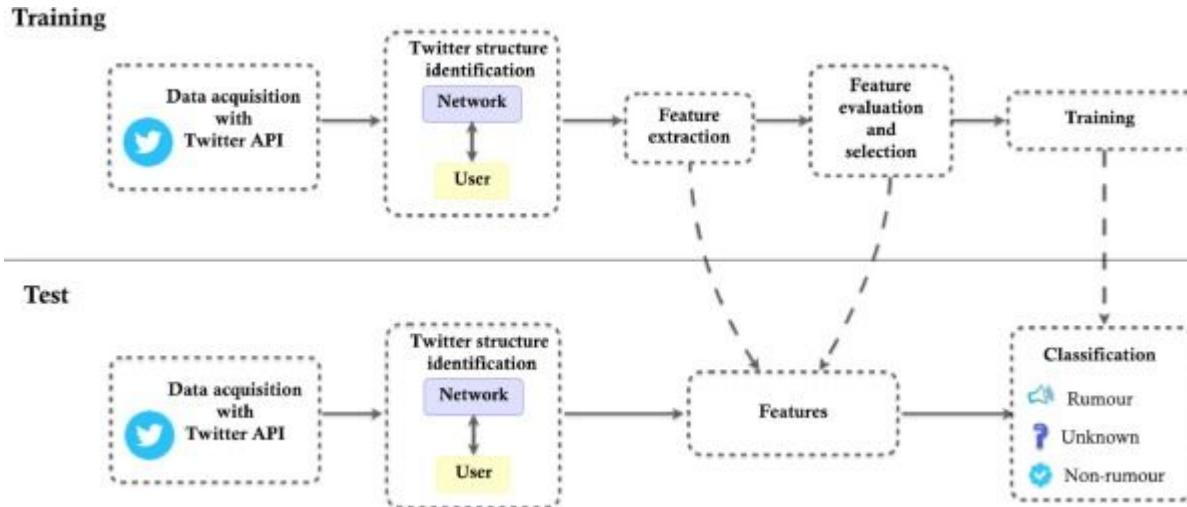
Received 27 December 2020; Revised 10 April 2021; Accepted 23 April 2021; Published 12 May 2021

Academic Editor: Hocine Cherif

Copyright © 2021 Aoshuang Ye et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The social network has become the primary medium of rumor propagation. Moreover, manual identification of rumors is extremely time-consuming and laborious. It is crucial to identify rumors automatically. Machine learning technology is widely implemented in the identification and detection of misinformation on social networks. However, the traditional machine learning methods profoundly rely on feature engineering and domain knowledge, and the learning ability of temporal features is insufficient. Furthermore, the features used by the deep learning method based on natural language processing are heavily limited. Therefore, it is of great significance and practical value to study the rumor detection method independent of feature engineering and effectively aggregate heterogeneous features to adapt to the complex and variable social network. In this paper, a deep neural network- (DNN-) based feature aggregation modeling method is proposed, which makes full use of the knowledge of propagation pattern feature and text content feature of social network event without feature engineering and domain knowledge. The experimental results show that the feature aggregation model has achieved 94.4% of accuracy as the best performance in recent works.

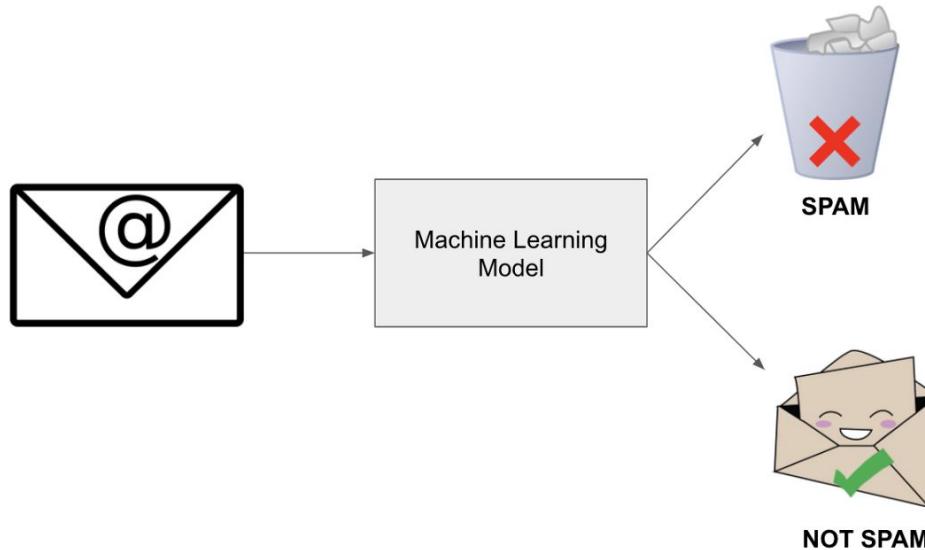
Workflow of Rumor detection



Importance of Rumor detection

- Openness and unrestricted way to share the information on social media platforms fosters information spread across the network regardless of its credibility.
- Such kind of spreading the misinformation usually happens in the context of breaking news.
- Due to unverified information, such misinformation, also known as rumors may cause severe damages.
- Despite overwhelming use, uncontrolled nature of social media platforms usually results in generation and unfold of rumors.
- Automatically detecting the rumors from social media platforms is one of the highly sought-after research area in the domain of social media analytics or Cybersecurity

Spam Email Detection

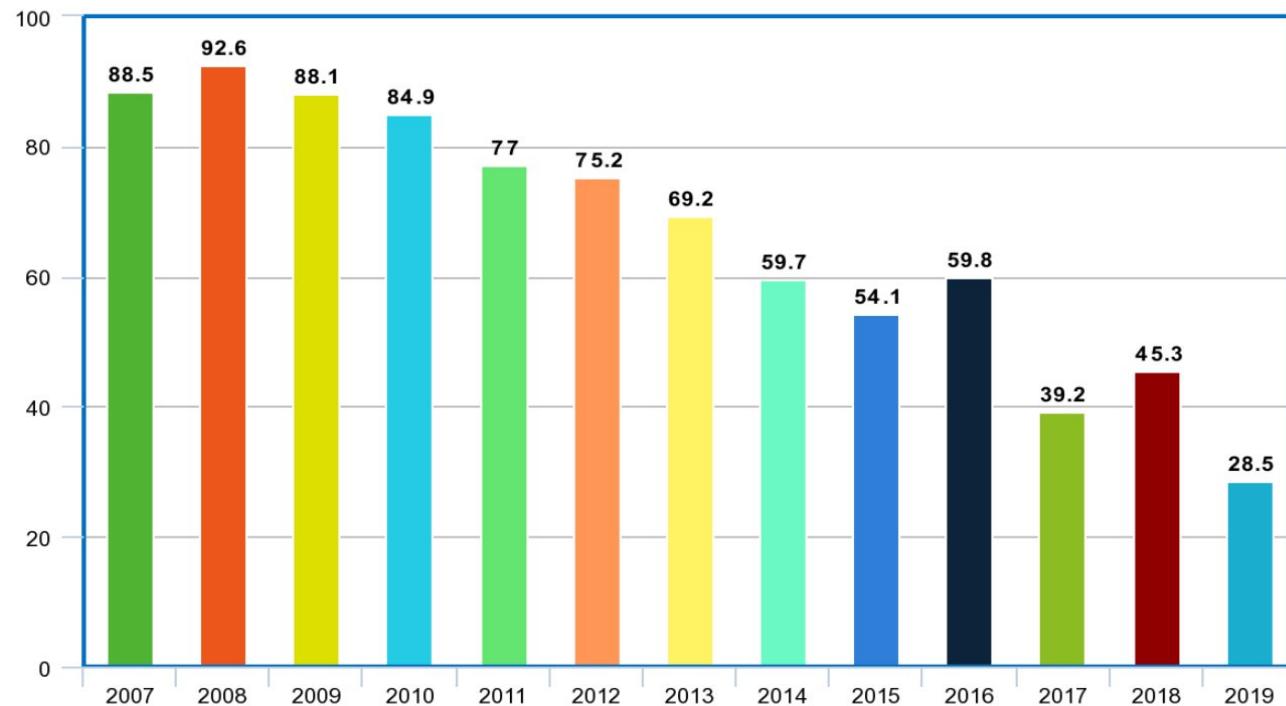


Spam Email Detection

- Spam e-mail are message randomly sent to multiple addressees by all sorts of groups, but mostly lazy advertisers and criminals who wish to lead you to phishing sites.
- Spam e-mails can be not only annoying but also dangerous to consumers.
- Spam e-mails can be defined as :
 1. Anonymity
 2. Mass Mailings



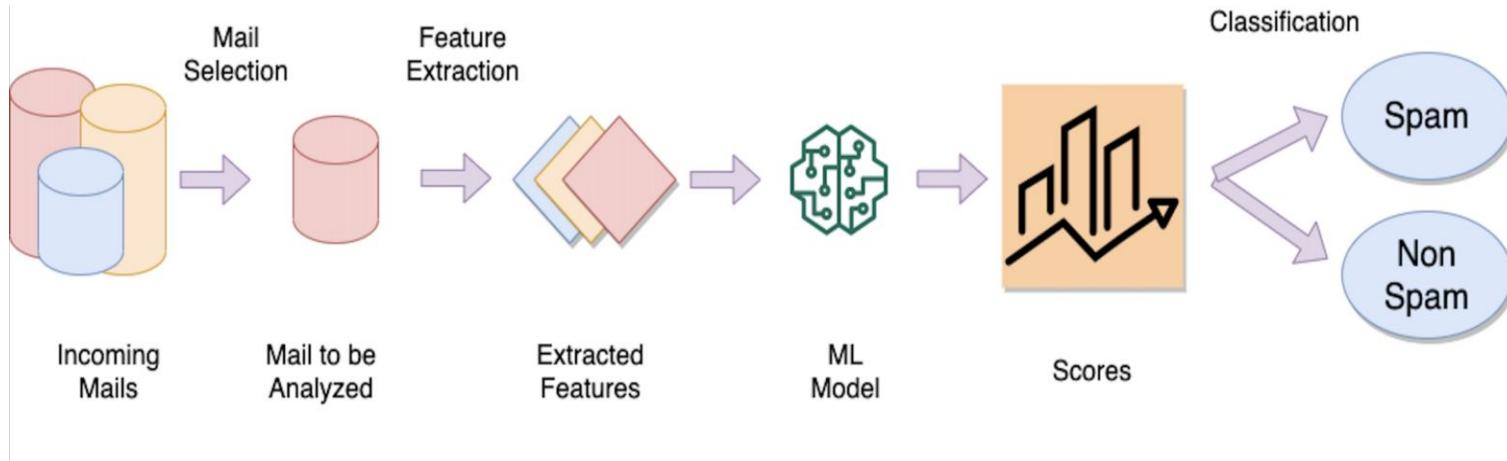
Statistics: Global spam volume as percentage of total e-mail traffic from 2007 to 2019



Problem Statement

- Unwanted e-mails irritating internet connection
- Critical e-mail message are missed and / or delayed
- Millions of compromised computers Billions of dollars lost worldwide
- Identity theft
- Spam can crash mail servers and fill up hard drive.

Basic Architecture



Benefits of spam filtering

- Protection against Viruses. Spam emails are not just innocent marketing tools they can be carriers of dangerous computer viruses.
- Keeping Hackers at Bay.
- Saving Time.
- Keeping your Reputation Intact.
- Customized Services.

Email threats

- Email messages are the dominant way of communication for online users
- Among the massive daily email traffic, unsolicited and unwanted messages have become a growing nuisance
- Increasingly posing serious threats to users' privacy and security by distributing false information

Email threats

Esteban et al. (2020)

- To create reliable content-based models that can classify email
- Evaluated of multiple neural network architectures applied to pre-trained word embedding representation to automatically acquire accurate indicators of malicious emails

*Proceedings of the Workshop on Social Threats in Online Conversations: Understanding and Management (STOC-2020), pages 48–55
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020
© European Language Resources Association (ELRA), licensed under CC-BY-NC*

Email Threat Detection Using Distinct Neural Network Approaches

Esteban Castillo¹, Sreekar Dhaduval², Peng Liu², Kartik-Singh Thakur²,
Adam Dalton³ and Tomek Strzalkowski¹

¹Rensselaer Polytechnic Institute, Troy, NY, USA, {castie2, tomek}@rpi.edu

²State University of New York at Albany, Albany, NY, USA, {sdhaduval, pliu3, kthakur}@albany.edu

³IHMC, Ocala, FL, USA, adalton@ihmc.us

Abstract

This paper describes different approaches to detect malicious content in email interactions through a combination of machine learning and natural language processing tools. Specifically, several neural network designs are tested on word embedding representations to detect suspicious messages and separate them from non-suspicious, benign email. The proposed approaches are trained and tested on distinct email collections, including datasets constructed from publicly available corpora (such as Enron, APWG, etc.) as well as several smaller, non-public datasets used in recent government evaluations. Experimental results show that back-propagation both with and without recurrent neural layers outperforms current state of the art techniques that include supervised learning algorithms with stylometric elements of texts as features. Our results also demonstrate that word embedding vectors are effective means for capturing certain aspects of text meaning that can be teased out through machine learning in non-linear/complex neural networks, in order to obtain highly accurate detection of malicious emails based on email text alone.

Email threats

Esteban et al. (2020)

- Dataset: Enron and APWG

Dataset type	Approach	Features (Matrix columns)	Email length (Matrix rows)	LSTM Neurons	Convolutional Neurons	Back-Propagation Neurons	Batches	Epochs	Accuracy
Dry-Run 1	BP	60	40	—	—	8-8	70	6	0.9568*
	LSTM	50	40	50	—	32-16-8-3	90	7	0.9317
	BP	50	50	—	—	4-4	100	6	0.9127
	CN	20	40	—	40	8-8	50	4	0.9175
	LSTM	30	50	60	—	4-4	110	6	0.9114
	BP	40	40	—	—	8-8	90	5	0.9031
	BP	30	60	—	—	8-8	50	4	0.9012
	LSTM	40	40	50	—	32-16-8-3	60	6	0.8855
	LSTM	40	50	60	—	50-50	90	7	0.8821
	CN	20	30	—	40	3-3-3	60	4	0.8793
	SVM	—	—	—	—	—	—	—	0.8137
	NB	—	—	—	—	—	—	—	0.7915
	LR	—	—	—	—	—	—	—	0.7824
Dry-Run 2	LSTM	30	40	60	—	3-3-3	60	5	0.9185*
	BP	40	60	—	—	8-8	70	7	0.9136
	BP	30	60	—	—	3-3-3	80	5	0.9023
	BP	40	40	—	—	4-4	70	6	0.9012
	CN	20	30	—	40	50-50	50	3	0.8983
	BP	30	40	—	—	32-16-8-3	70	4	0.8839
	CN	30	30	—	40	32-16-8-3	50	3	0.8748
	LSTM	40	50	60	—	8-8	100	6	0.8612
	SVM	—	—	—	—	—	—	—	0.8529
	BP	40	40	—	—	50-50	110	6	0.8512
	BP	30	30	—	—	3-3-3	80	5	0.8507
	LR	—	—	—	—	—	—	—	0.8045
	NB	—	—	—	—	—	—	—	0.7749

BP: Back-propagation

CN: Convolutional Network

LSTM: Long Short-Term Memory Network (recurrent network)

SVM: Support Vector Machine

NB: Naïve Bayes

LR: Logistic Regression

Email threats

Emmanuel et al. (2019)

- Lack of effective strategy to handle the threats to the security of the spam filters.
- The inability of the current spam filtering techniques to effectively deal with the concept drift phenomenon.
- Majority of the existing email spam filters does not possess the capacity to incrementally learn in real-time.
- Failure of many spam filters to reduce their false positive rate.
- Development of more efficient image spam filters.
- The need to develop adapted, scalable, and integrated filters by applying ontology and semantic web to spam email filtering.
- Lack of filters that have the capacity to dynamically update the feature space.
- The need to apply deep learning to spam filtering in order to exploit its numerous processing layers and many levels of abstraction to learn representations of data.

<https://www.sciencedirect.com/science/article/pii/S2405844018353404>

Heliyon

Volume 5, Issue 6, June 2019, e01802



Machine learning for email spam filtering: review, approaches and open research problems

Emmanuel Gbenga Dada^a, Joseph Stephen Bassi^a, Haruna Chiroma^b, Shaf'i Muhammad Abdulhamid^c, Adebayo Olusola Adetunmbi^d, Opeyemi Emmanuel Ajibawa^e

Show more ▾

+ Add to Mendeley Share Cite

<https://doi.org/10.1016/j.heliyon.2019.e01802>

Under a Creative Commons license

Get rights and content

open access

Abstract

The upsurge in the volume of unwanted emails called spam has created an intense need for the development of more dependable and robust antispy filters. Machine learning methods of recent are being used to successfully detect and filter spam emails. We present a systematic review of some of the popular machine learning based email spam filtering approaches. Our review covers survey of the important concepts, attempts, efficiency, and the research trend in spam filtering. The preliminary discussion in the study background examines the applications of machine learning techniques to the email spam filtering process of the leading internet service providers (ISPs) like Gmail, Yahoo and Outlook emails spam filters. Discussion on general email spam filtering process, and the various efforts by different researchers in combating spam through the use machine learning techniques was done. Our review compares the strengths and drawbacks of existing machine learning approaches and the open research problems in spam filtering. We recommended deep leaning and deep adversarial learning as the future techniques that can effectively handle the menace of spam emails.

Identification of Hate Content



Hate Speech as per wiki

Hate speech is defined by the Cambridge Dictionary as "*public speech that expresses hate or encourages violence towards a person or group based on something such as race, religion, sex, or sexual orientation*".

Hate speech is "*usually thought to include communications of animosity or disparagement of an individual or a group on account of a group characteristic such as race, colour, national origin, sex, disability, religion, or sexual orientation*".

A legal definition of hate speech varies from country to country.

Hate mail as per wiki

- Hate mail (as electronic, posted, or otherwise) is a form of harassment, usually consisting of invective and potentially intimidating or threatening comments towards the recipient. Hate mail often contains exceptionally abusive, foul or otherwise hurtful language.
- The recipient may receive disparaging remarks concerning their ethnicity, sexuality, gender, religion, intelligence, political ideology, sense of ethics, or sense of aesthetics. The text of hate mail often contains profanity, or it may simply contain a negative, misappropriating message.
- Senders of hate mail normally send anonymous letters or pose as someone else (either a different or fictitious individual) in order to avoid being identified and tracked down, as the nature of some hate mail would inevitably result in criminal charges if the sender was identified.
- For Example: Hate mail has frequently been issued to footballers and managers by fans of rival football teams, and also by their own fans who are dissatisfied with the performance of an individual player, manager or the team.
- McGivern, Mark (12 March 2015). "Rangers fan who sent letter bomb to Neil Lennon is pictured back at Ibrox". Daily Record. Retrieved 26 October 2020 --- *Neil Lennon, the former Celtic F.C. manager, received hate mail including a package containing nail bomb from Rangers fans. Two men were jailed for five years in April 2012 for sending a nail bomb to Lennon*

Hate Content

- Online hate, described as abusive language, aggression, cyberbullying, hatefulness, insults, personal attacks, provocation, racism, sexism, threats, or toxicity, has been identified as a major threat on online social media platforms
- Proliferation of social media enables people to express their opinions widely online
- Resulted in the emergence of conflict and hate, making online environments uninviting for users

Identification of Hate Content

- Aim is recognizing posts on social media platforms that spread hatred of any kind.
- Posts could be in the form of text, image and video.

Identification of Hate Content

Joni et al. (2020)

Springer Open

Search

Research | Open Access | Published: 02 January 2020

Developing an online hate classifier for multiple social media platforms

Joni Salminen , Maximilian Hopf, Shammur A. Chowdhury, Soon-gyo Jung, Hind Almerekhi & Bernard J. Jansen

Human-centric Computing and Information Sciences 10, Article number: 1 (2020) | Cite this article

23k Accesses | 44 Citations | 27 Altmetric | Metrics

Abstract

The proliferation of social media enables people to express their opinions widely online. However, at the same time, this has resulted in the emergence of conflict and hate, making online environments uninviting for users. Although researchers have found that hate is a problem across multiple platforms, there is a lack of models for online hate detection using multi-platform data. To address this research gap, we collect a total of 197,566 comments from four platforms: YouTube, Reddit, Wikipedia, and Twitter, with 80% of the comments labeled as non-hateful and the remaining 20% labeled as hateful. We then experiment with several classification algorithms (Logistic Regression, Naive Bayes, Support Vector Machines, XGBoost, and Neural Networks) and feature representations (Bag-of-Words, TF-IDF, Word2Vec, BERT, and their combination). While all the models significantly outperform the keyword-based baseline classifier, XGBoost using all features performs the best ($F1 = 0.92$). Feature importance analysis indicates that BERT features are the most impactful for the predictions. Findings support the generalizability of the best model, as the platform-specific results from Twitter and Wikipedia are comparable to their respective source papers. We make our code publicly available for application in real software systems as well as for further development by online hate researchers.

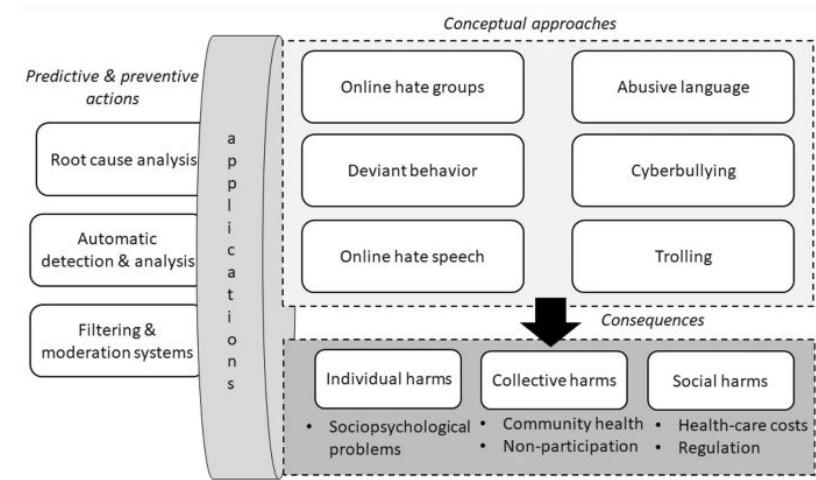


Fig. Conceptual framework of the focus areas: online hate groups, abusive language, deviant behavior, cyberbulling Trolling, and its consequences of online hate (effects on individuals and groups)

Identification of Hate Content: Challenges

Research Question (Joni et al., 2020)

- How well do different algorithms and feature representations perform for hate detection in multiple social media platforms?
- What are the most impactful features when predicting hate in multiple social media platforms?
- How well does a machine learning model learn linguistic characteristics in the hateful and non-hateful language in a cross-platform environment?
- How to develop an effective classifier that could identify the hate contents which possess probability score beyond a certain threshold

Identification of Hate Content: Challenges

Challenges (Sean et al., 2019)

- Challenge faced by automatic hate speech detection systems is the changing of attitudes towards topics over time and historical context.

“ . . .The merciless Indian Savages, whose known rule of warfare, is an undistinguished destruction of all ages, sexes and conditions. . .”

- Intuition suggests that this is hate speech; it refers to Native Americans as “merciless Indian savages”,
- However, this text is actually a quote from the Declaration of Independence.

OPEN ACCESS PEER-REVIEWED
RESEARCH ARTICLE

Hate speech detection: Challenges and solutions

Sean MacAvaney, Hao-Ren Yao, Eugene Yang, Katina Russell, Nazli Goharian, Ophir Frieder

Published: August 20, 2019 • <https://doi.org/10.1371/journal.pone.0221152>

Article	Authors	Metrics	Comments	Media
▼				

Abstract

Introduction

Defining hate speech

Datasets

Automatic approaches for
hate speech detection

Shortcomings and future
work

Conclusion

Abstract

As online content continues to grow, so does the spread of hate speech. We examine challenges faced by online automatic approaches for hate speech detection. Among these difficulties are subtleties in language, differing definitions on what constitutes hate speech, and limitations of data availability for training and testing of these systems. Furthermore, many recent approaches suffer from an interpretability problem, making it difficult to understand why the systems make the decisions that they do. We view SVM approach that achieves near state-of-the-art performance, while being able to produce more easily interpretable decisions than neural methods. We also discuss technical and practical challenges that remain for this task.

Identification of Hate Content: Challenges

György et al. (2021)

- Difficulty of automatic detection, as social media posts include paralinguistic signals (e.g. emoticons, and hashtags)
- Linguistic content contains plenty of poorly written text.
- Difficulty is presented by the context-dependent nature of the task, and the lack of consensus on what constitutes as hate speech
- Makes the task of creating large labelled corpora difficult, and resource consuming

The screenshot shows a web browser displaying an academic article from Springer. The title of the article is "Challenges of Hate Speech Detection in Social Media". It is categorized as "Original Research | Open Access" and was published on "13 February 2021". The article is authored by "György Kovács", "Pedro Alonso", and "Rajkumar Saini". It is part of the journal "SN Computer Science" (Volume 2, Article number: 95, 2021). The abstract discusses the challenges of detecting hate speech in social media, mentioning the uncontrolled spread of hate speech, its potential damage to society, and the difficulty of automatic detection due to paralinguistic signals like emoticons and hashtags, as well as the poor quality of written text. The abstract also notes the context-dependent nature of the task and the lack of consensus on what constitutes hate speech. The page includes standard Springer navigation links and a sidebar with various research-related sections.

<https://link.springer.com/article/10.1007/s42979-021-00457-3>

Identification of Hate Content: Challenges

(György et al., 2021)

Technique	Dataset	Result (Hate speech detection)
Deep Neural Network (CNN-LSTM) (Combining convolutional and recurrent layers)	Hate Speech and Offensive Content Identification in Indo-European Languages (HASOC2019)	Macro F1 score of 0.63

Identification of Hate Content: Challenges

Paul et al. (2021)

- Struggle with correctly **classifying negated phrases** such as “I don’t hate trans people”
- Need to include **functionalities for negation** in hateful and non-hateful content
- Contributed a dataset “**HATECHECK**” for Functional Tests of Hate Speech Detection
- Transformer model architecture is used for the development of Hate Speech Detection model

<https://aclanthology.org/2021.acl-long.4.pdf>

The screenshot shows a PDF document titled "HATECHECK: Functional Tests for Hate Speech Detection Models". The title is at the top, followed by the authors' names: Paul Röttger^{1,2}, Bertram Vidgen², Dong Nguyen³, Zeerak Waseem⁴, Helen Margetts^{1,2}, and Janet B. Pierrehumbert¹. Below the authors are four superscripted footnotes: ¹University of Oxford, ²The Alan Turing Institute, ³Utrecht University, and ⁴University of Sheffield. The abstract section begins with a paragraph about the difficulty of detecting online hate and the limitations of current evaluation metrics like accuracy and F1 score. It highlights the need for more generalizable models and the risks of overestimating performance.

HATECHECK: Functional Tests for Hate Speech Detection Models

Paul Röttger^{1,2}, Bertram Vidgen², Dong Nguyen³, Zeerak Waseem⁴,
Helen Margetts^{1,2}, and Janet B. Pierrehumbert¹

¹University of Oxford
²The Alan Turing Institute
³Utrecht University
⁴University of Sheffield

Abstract

Detecting online hate is a difficult task that even state-of-the-art models struggle with. Typically, hate speech detection models are evaluated by measuring their performance on held-out test data using metrics such as accuracy and F1 score. However, this approach makes it difficult to identify specific model weak points. It also risks overestimating generalisable model performance due to in-

Identification of Hate Content: Challenges

Aditya et al. (2021)

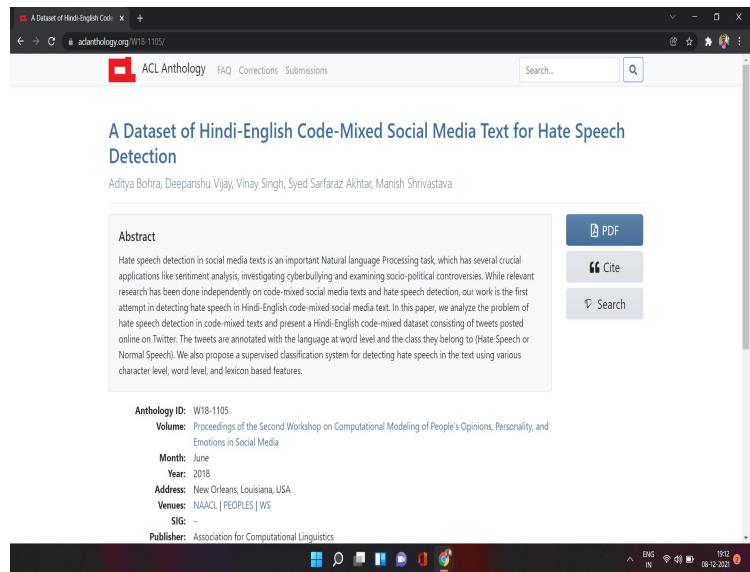
- Hate Speech Detection is a challenging task for Code-Mixed Social Media Text in multilingual societies like India, usage of code-mixed languages (among which Hindi-English is most prominent)

T1 : “Mujhe apne manager se nafrat hai, I want to kill that guy.”

Translation : “I hate my manager, I want to kill that guy.”

- Requirement of suitable dataset for Code-Mixed Social Media Text to detect hate speech

<https://aclanthology.org/W18-1105/>



Identification of Hate Content: Challenges

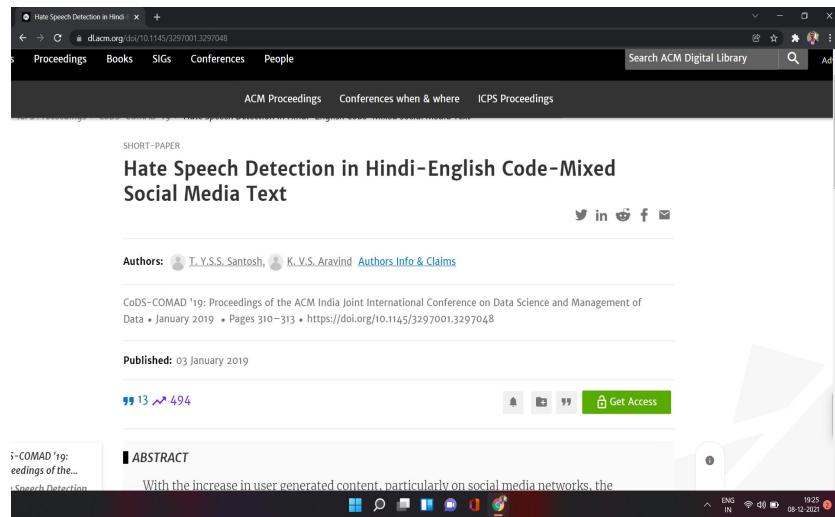
Aditya et al. (2021)

Technique	Dataset (Developed)	Result (Accuracy on hate speech detection)
SVM (Considering different features: Character N-Grams, Word N-Grams, Punctuations, Negation Words and Lexicon)	Hindi-English Code-Mixed Social Media Text	71.7

Identification of Hate Content: Challenges

Santosh et al. (2019)

- Identification of hate speech from code-mixed social media text
- Comparative study on evaluating various models (SVM, Random Forest, Sub-word level, LSTM)



<https://dl.acm.org/doi/10.1145/3297001.3297048>

Identification of Hate Content: Challenges

Santosh et al. (2019)

Technique	Dataset (Tweets)	Result (Hate speech detection)
Hierarchical LSTM model with attention based on phonemic sub-words	Hindi-English Code-Mixed Social Media Text (3800 Tweets: 2300 tweets as hate and 1500 tweets as non-hate tweets)	Accuracy: 66.6 Recall: 45.1 F1-Score: 48.7

References

- <https://www.cio.gov/2020-10-13-Social-Media-and-Threats/>
- Pennycook, G., & Rand, D. (2019, February 12). Fighting misinformation on social media using crowdsourced judgments of news source quality. Retrieved September 30, 2020, from <https://www.pnas.org/content/116/7/2521>
- Taulli, T. (2019, August 12). Deepfake: What You Need to Know. Retrieved September 30, 2020, from <https://www.forbes.com/sites/tomtaulli/2019/06/15/deepfake-what-you-need-to-know/>
- Salminen, Joni, et al. "Developing an online hate classifier for multiple social media platforms." *Human-centric Computing and Information Sciences* 10.1 (2020): 1-34.
- MacAvaney, Sean, et al. "Hate speech detection: Challenges and solutions." *PLoS one* 14.8 (2019): e0221152.
- Thanh Thi Nguyen, Quoc Viet Hung Nguyen, Cuong M. Nguyen, Dung Nguyen, Duc Thanh Nguyen, Saeid Nahavand, "Deep Learning for Deepfakes Creation and Detection: A Survey", aRxiv, <https://arxiv.org/pdf/1909.11573.pdf>
- Bohra, Aditya, et al. "A dataset of hindi-english code-mixed social media text for hate speech detection." *Proceedings of the second workshop on computational modeling of people's opinions, personality, and emotions in social media*. 2018.
- Santosh, T. Y. S. S., and K. V. S. Aravind. "Hate speech detection in hindi-english code-mixed social media text." Proceedings of the ACM India Joint International Conference on Data Science and Management of Data. 2019.
- Kovács, György, Pedro Alonso, and Rajkumar Saini. "Challenges of Hate Speech Detection in Social Media." *SN Computer Science* 2.2 (2021): 1-15.
- Castillo, Esteban, et al. "Email Threat Detection Using Distinct Neural Network Approaches." *Proceedings for the First International Workshop on Social Threats in Online Conversations: Understanding and Management*. 2020.
- Dada, Emmanuel Gbenga, et al. "Machine learning for email spam filtering: review, approaches and open research problems." *Heliyon* 5.6 (2019): e01802.
- Ye, Aoshuang, et al. "An End-to-End Rumor Detection Model Based on Feature Aggregation." *Complexity* 2021 (2021).

*Thank
you!*

CS401: Artificial Intelligence.

Book :- Peter Norvin

03/08/2022

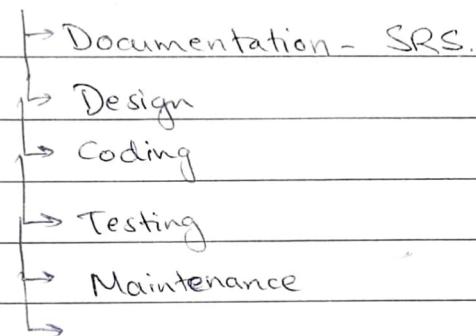
- Automation.
- Goal ; End Result ; Purpose

04/08/2022

- Software Engineering 1960s. started.

Current Model → Agile model.

Waterfall
Lifecycle
Spiral
Agile

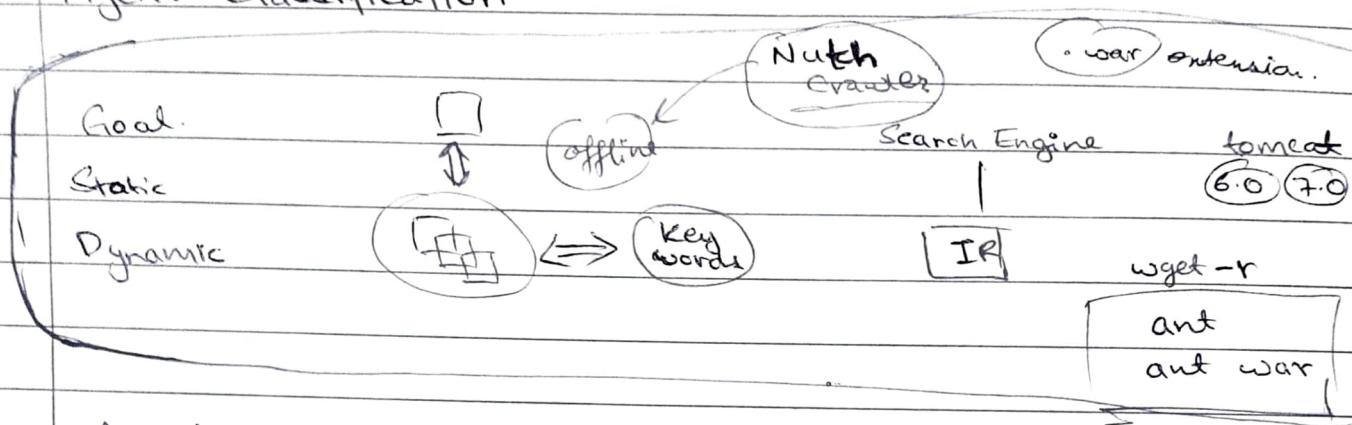


- Why GPUs instead of CPUs?
- Knowledge-based vs. Rule-based systems. (Domain knowledge 1969-85)
- AI in real life: Predictive Analysis, Content Generation, E-commerce, E-mail marketing, Web Design, Online Advertising, Personalised User Experience
- Introduction → History, definition.
 - 1950 : Turing's 'Computing Machinery and Intelligence'
 - 1956 : Dartmouth meeting 'AI' name adopted. Birth of AI
 - 1964 : Showing computers understand NLP to solve algebraic word problems correctly.
 - 1986 : Rise of ML.
 - 1990s : Major advances in all areas of AI
 - 1995 : AI as Science.
- Supervised vs. unsupervised learning.
- AI: concerned with design of intelligence in an artificial device.
 - McCarthy, 1956.

- The Turing Test:

- 1997: Deep Blue chess program beats chess champion.

* Agent Classification:



Agent and Environment Relation:

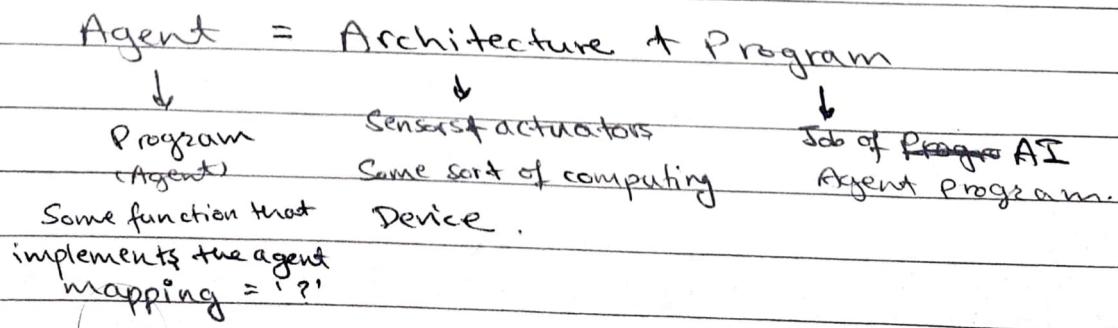
- Static & Dynamic
- Fully and Partially Observable
- Deterministic & Stochastic
- Episodic vs Sequential
- Discrete vs Continuous

Characterizing a Task Environment:

→ PEAS: performance measure, Environment, Actuators, Sensors.

Examples for automated task	→ Safe, fast, legal	→ Roads, other	→ Steering wheel	→ Camera
	→ Comfortable.	→ traffic, pedestrian	→ Accelerator	→ LIDAR
			→ Brake	→ GPS
			→ Signal	→ Speedometers
			→ Horn	→ Odometers
				→ Engine sensor
				→ Keyboard

* Structure of agents:



* Types of Environment in AI:

- Fully Observable : Chess
- Partially Observable : ~~Maze~~ Maze, Driving Cars, Card game
- Deterministic : Chess, Traffic Signal
- Stochastic : Driving, Football, Random Song Playlist
- Competitive : Chess
- Collaborative : Self Driving Cars
- Single Agent : Maze, Brushing Teeth, Restaurant Order
- Multi Agent : Football, Chess, Cards,
- Dynamic : Roller Coaster Ride
- Static : Empty House
- Discrete : Chess
- Continuous : Self driving Cars.
- Episodic : Restaurant Order, Support Bot
- Sequential : Brushing teeth, Chess, Cards, Tennis,

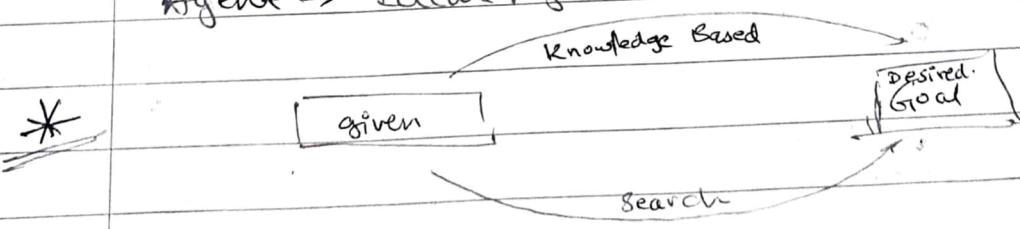
↓ Takes input from the environment.

19/08/2022

* Agent Architectures:

- Table Based Agent
 - Percept-based / Reflex Agent
 - State-based / Model-based Agent
- Goal-based Agent
→ Utility-based Agent
→ Learning Agent

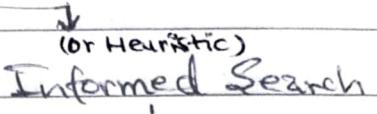
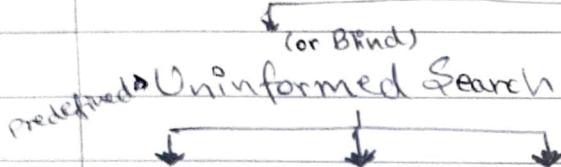
Agent → Ideal Agent → Autonomous Agent.



* State Space Search:

* Informed Search:

Search Algorithms



* Inefficien Uninformed

- Systematically explore the state space search and find the goal.
- Inefficient in most cases

Informed

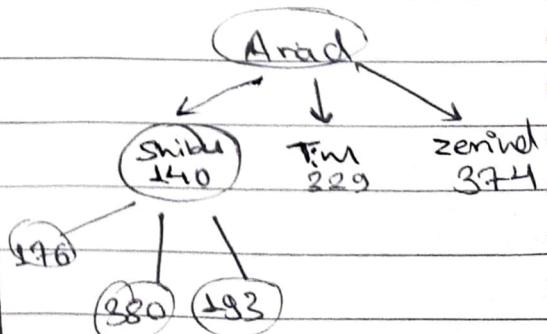
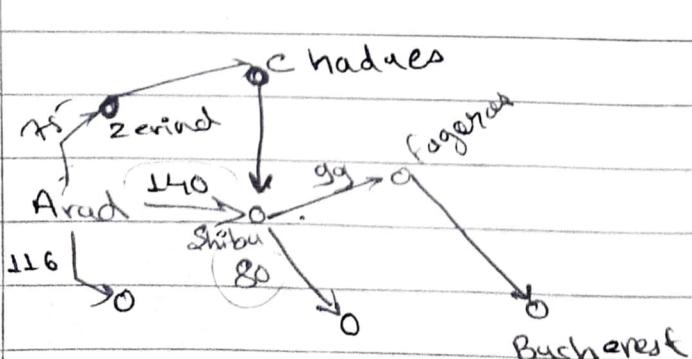
- Problem specific knowledge.
- May be more efficient.
- Concept of heuristic function.

- Heuristic means 'rule of thumb': most promising search path

- Informed/ Heuristic Search:
 - Best first
 - ~~A^{*}~~ Search
 - A^{*} Search

- 8-Puzzle : Manhattan Distance.

- Greedy Best-first Search.



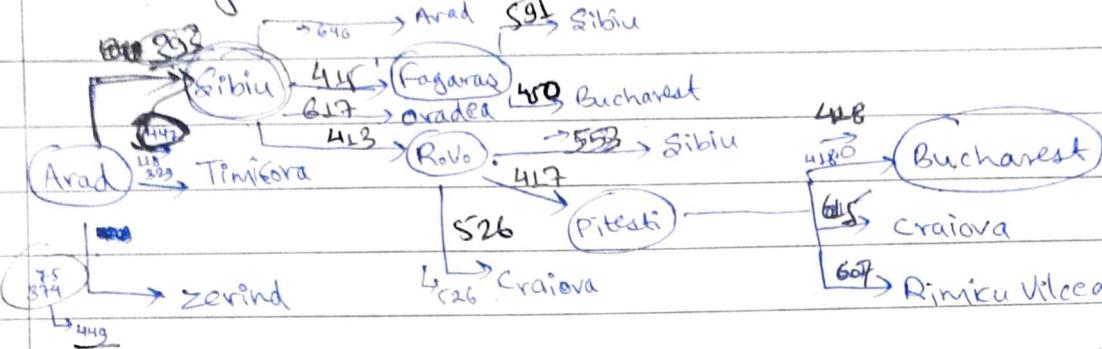
329 + 118

140
253

393

— / / —

A* Search: Starting Node: Arad → Destination: Bucharest



Arad → Sibiu → RV → Pitesti → Bucharest
 \downarrow
 Fagaras ~~X~~

418 ~~X~~

8-puzzle Misplaced Tiles Heuristic the number of tiles out of place.

2	8	3
1	6	4
7	5	

Initial state

1	2	3
8		4
7	6	5

Goal state

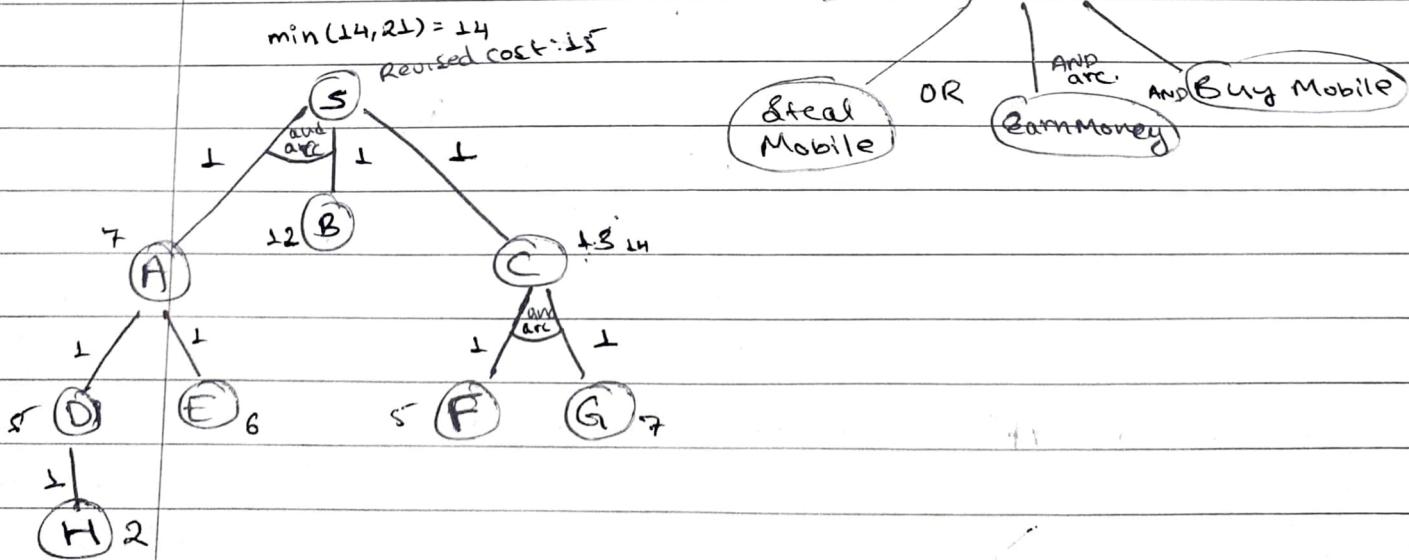
* A* Algorithm: Informed search & works as Best first Search
AND Arcs: Based on problem decomposition (Breakdown problem into small pieces)
 Efficient method to explore a solution path

AND OR Graph:

e.g.: [own Mobile phone]

$$\min(14, 21) = 14$$

revised cost: 15

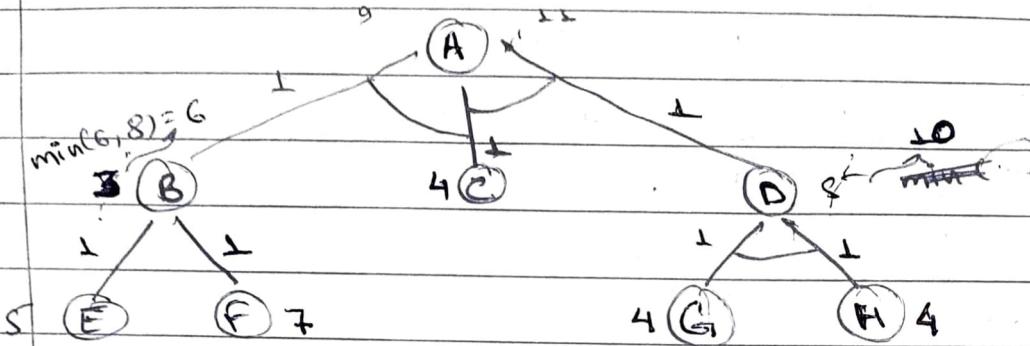


Revised cost $\min(12, 6) = 6$ final, so we take left path.

$$\min(12, 11) = 11$$

$$\min(9, 11) = 9$$

ABE

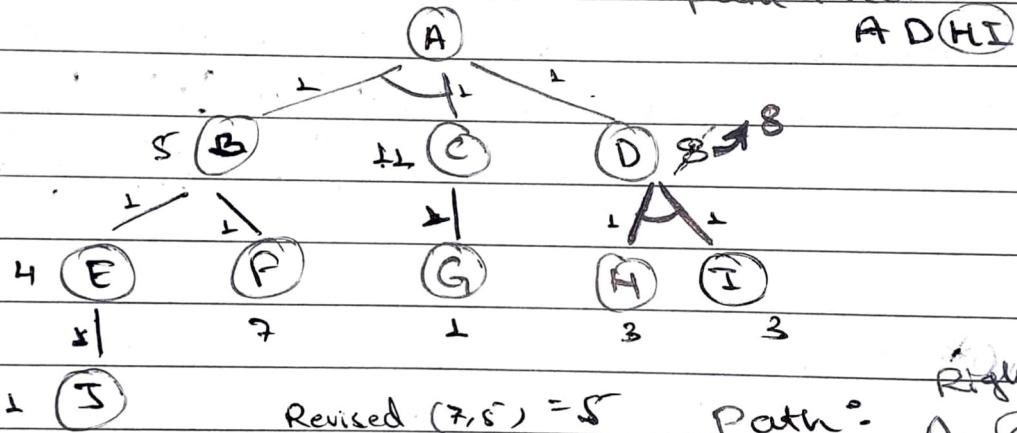


~~A* Algorithm: The algorithm does not explore all the solution paths once it finds a solution.~~

Revised cost $\min(18, 9) = 9$ final.

$$\min(18, 9) = 9$$

path taken — Right Direction



Revised $(7, 8) = 5$

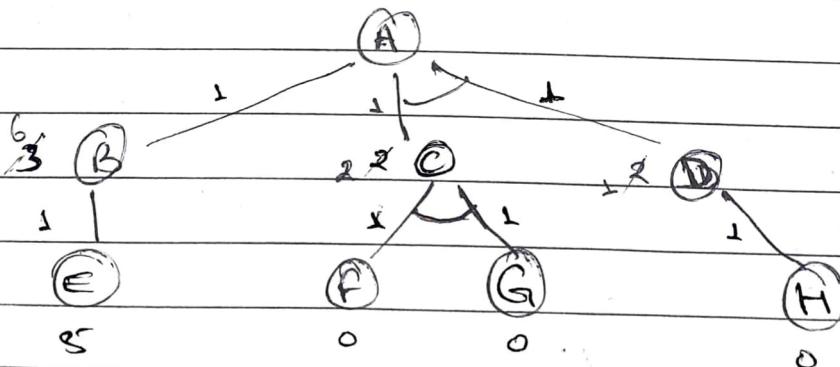
cost $(7, 6) = 6$

$(4, 6) = 4$

Path:

A C D F G H

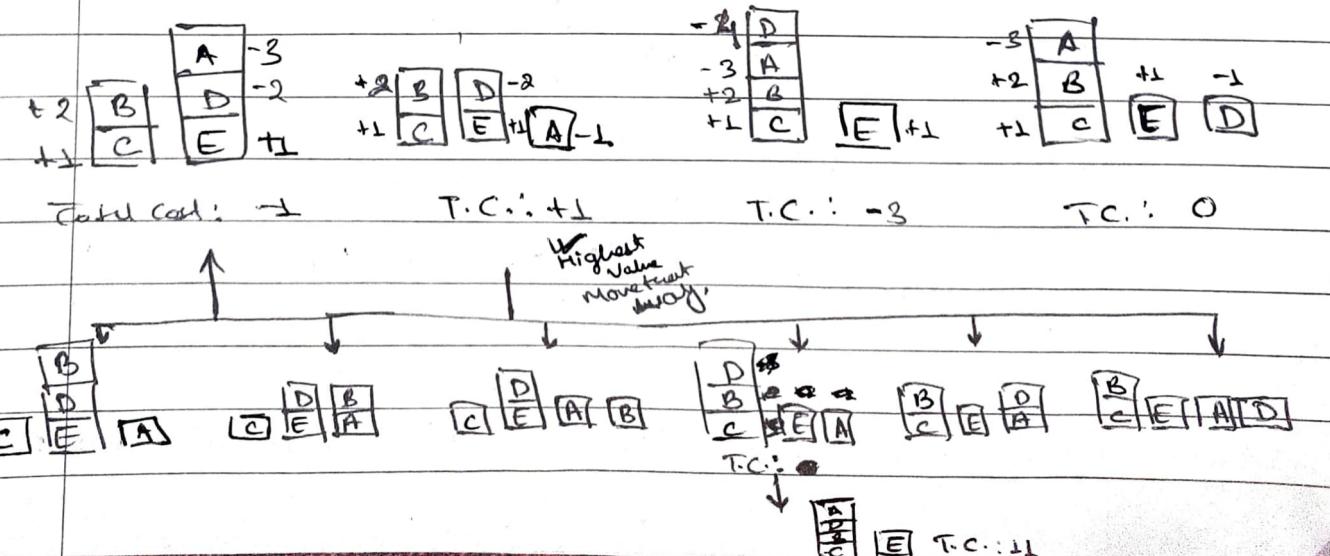
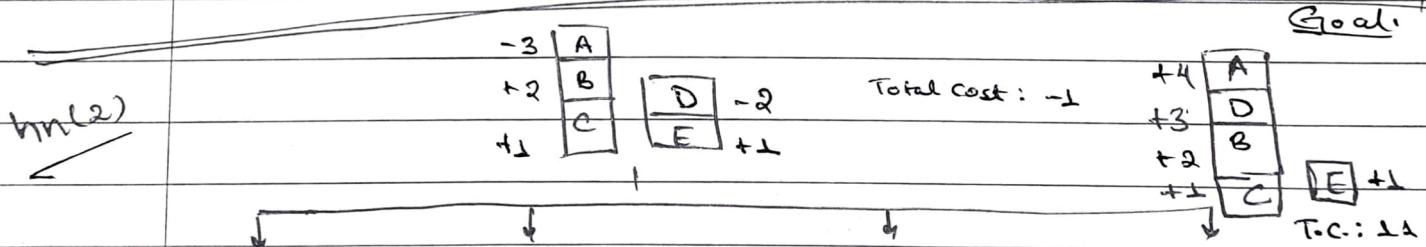
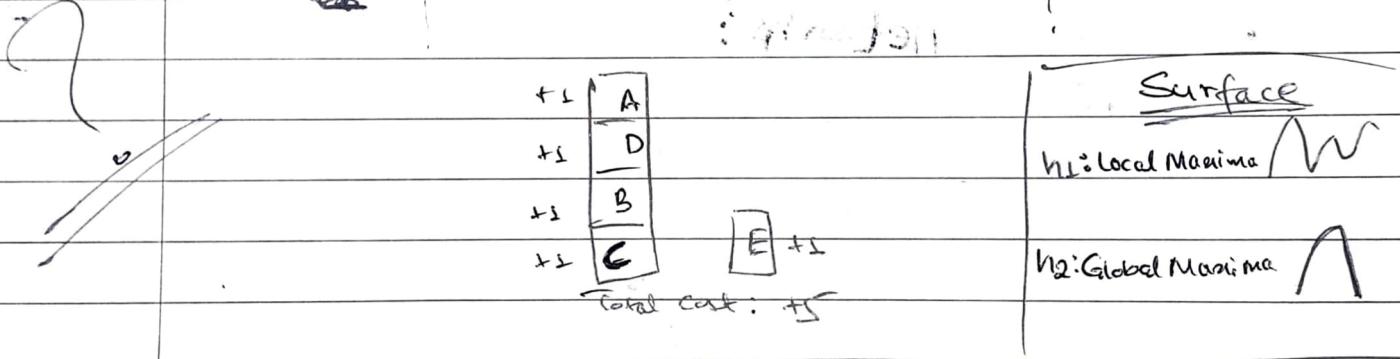
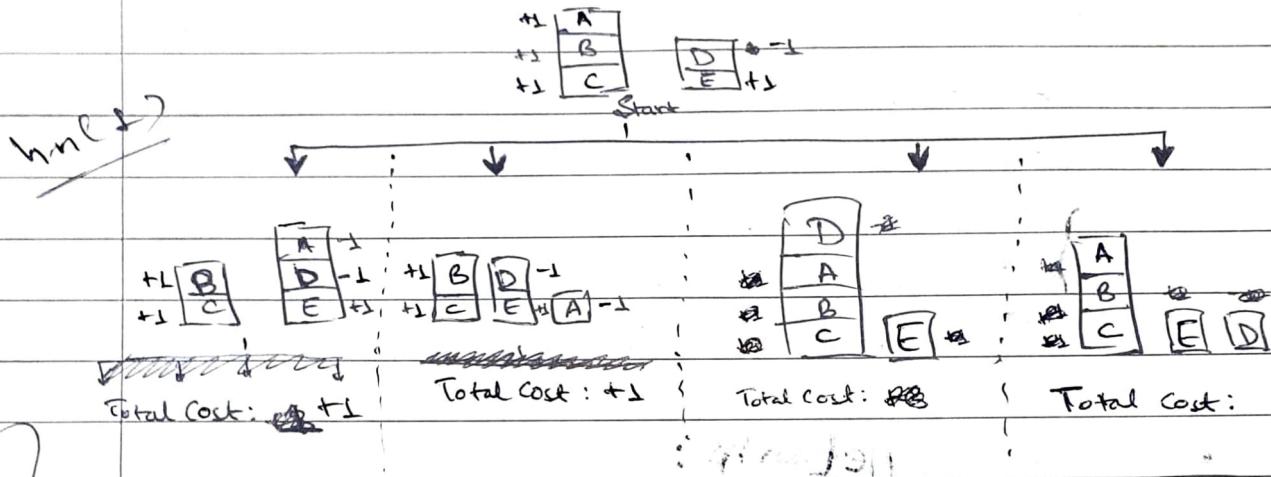
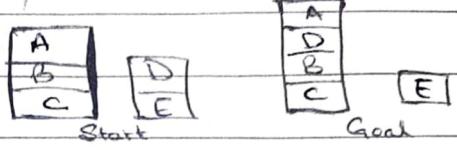
Right Direction.



Q. When A^* is optimal vs. Admissible?

* Hill Climbing:

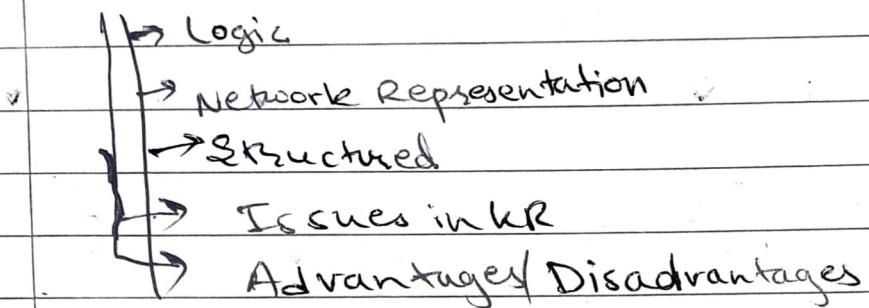
- Local Search Method
- Keep small ^{number of} nodes in memory.



* Semantic Nets, Frames, Conceptual Graphs :

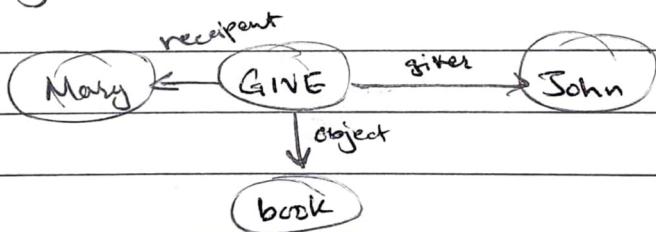
- Knowledge Representation as a medium for human expression.
- We need to be able to understand what the system knows and how it draws its conclusion.

Knowledge Representation

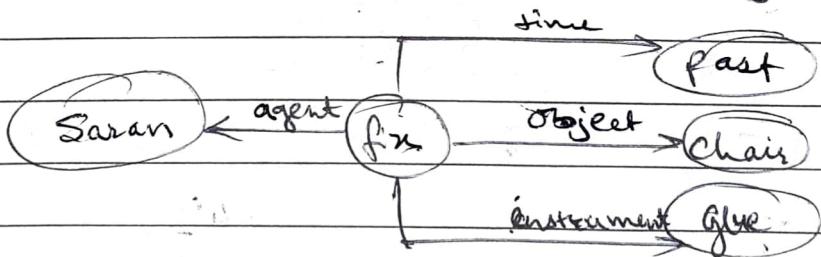


• Semantic Networks :

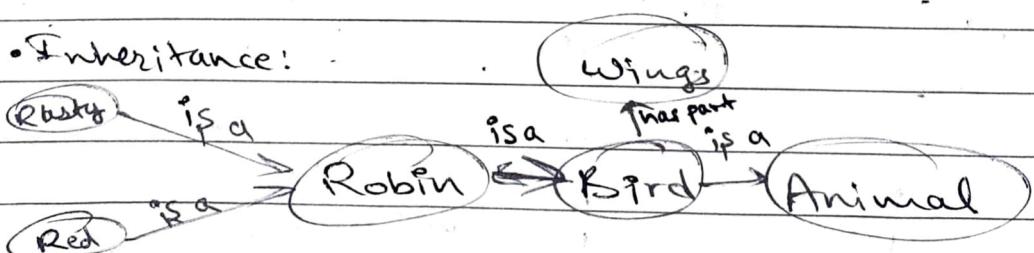
Non-binary Relations:



John gives book to Mary.



• Inheritance:



Relation: ISA (is a) Link: haspart (has part)

- First Order Logic :

 - Pros cons of Propositional logic

- Universal Quantification \forall
- Existential Quantification \exists
- Nested Quantifiers:

12/09/2022



$$\sim(\exists x) P(x) \equiv (\forall x)(\sim P(x))$$

$$\sim(\forall x) P(x) \equiv (\exists x)(\sim P(x))$$

$$\forall x [P(x) \wedge Q(x)] \equiv \forall x P(x) \wedge \forall y Q(y)$$

$$\exists x [P(x) \vee Q(x)] \equiv \exists x P(x) \vee \exists y Q(y)$$

- de-Morgan's Law:

$$\sim(x_1 \wedge x_2) \equiv (\sim x_1 \vee \sim x_2)$$

$$\sim(x_1 \vee x_2) \equiv (\sim x_1 \wedge \sim x_2)$$

- Distributive Law:

$$P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$$

$$P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$$

- Commutative Laws:

$$P \wedge Q \equiv Q \wedge P$$

$$P \vee Q \equiv Q \vee P$$

- Associative Law:

$$(P \wedge Q) \wedge R \equiv P \wedge (Q \wedge R)$$

$$(P \vee Q) \vee R \equiv P \vee (Q \vee R)$$

- Contrapositive Law:

$$P \rightarrow Q \equiv \sim Q \rightarrow \sim P$$

- AND of ORs

- $A \vee B \vee C \underbrace{\quad}_{\text{OR}} \wedge \underbrace{(\neg A \wedge \neg B \wedge \neg C)}_{\text{AND}} \underbrace{\quad}_{\text{OR}}$

- $(A \vee B) \wedge C$

- Conversion from FOL (first order logic) to CNF
(Conjunctive NF)

Step I: Eliminate Bi-directional / Implication

$$P \Leftrightarrow Q \equiv (\neg P \vee Q) \wedge (\neg Q \vee P)$$

$$\begin{aligned} P \Leftrightarrow Q &\equiv (\neg P \vee Q) \wedge (\neg Q \vee P) \\ &\equiv (\neg(\neg P \vee Q)) \wedge (\neg(\neg Q \vee P)) \end{aligned}$$

II: Move all Negations inwards \neg (\neg, \vee)

$$\neg(\forall x P(x)) \equiv \forall x \neg P(x)$$

$$\neg(\neg P(x)) \equiv P(x)$$

$$\neg(P \vee Q) \equiv \neg P \wedge \neg Q$$

III: Standardise Variables apart by renaming them, each Quantitative quantifier should use a different variable.

$$(\forall x P(x)) \vee (\exists y Q(y))$$

IV: Skolemise: Each existential variable is replaced by Skolem constants or Skolem function.

$$\exists x \text{Rich}(x) \equiv \text{Rich}(\text{G}_x)$$

$$\forall x \text{Person}(x) \rightarrow \exists y \text{Accent}(y) \wedge \text{Has}(x, y)$$

V: Drop Universal Quantifier.

VI: Distribute \wedge over \vee

$$P(\wedge Q) \vee S \equiv (P \vee S) \wedge (Q \vee S)$$

- $A \leftrightarrow (B \vee C)$: Converts into CNF

13/09/2022

$$1. (A \rightarrow B) \rightarrow C$$

$$2. A \rightarrow (B \rightarrow C)$$

$$3. (A \rightarrow B) \vee (B \rightarrow A)$$

$$4. (\neg P \rightarrow (P \rightarrow Q))$$

$$5. (P \rightarrow (Q \rightarrow R)) \rightarrow (P \rightarrow (R \rightarrow Q))$$

$$6. (P \rightarrow Q) \rightarrow ((Q \rightarrow R) \rightarrow (P \rightarrow R))$$

* Horn Clause: A disjunction with at most one positive literal.

$\sim P \vee \overset{*}{Q} \vee \overset{*}{S}$] Not Horn's clause? * literal.

$\sim P \vee \sim Q \vee \overset{*}{S}$] Horn's clause?

$$\bullet (n_1 \wedge n_2 \wedge n_3) \rightarrow Y$$

$$\equiv \sim (n_1 \wedge n_2 \wedge n_3) \vee Y$$

$$\equiv (\sim n_1 \vee \sim n_2 \vee \sim n_3) \vee Y \quad] \text{Horn's clause.}$$

14/09/2022

* Lexical Analysis

↳ Syntactic Analysis

↳ Semantic Analysis

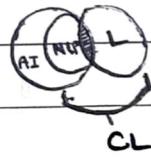
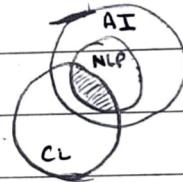
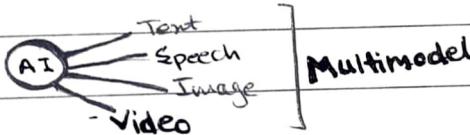
↳ Pragmatics and Discussion?

* NLP

* AI

* Linguistics

* Computational Linguistics



Challenges of Text detection:

- Language dependent
- Data dependent
- AI → Rule-based / ~~Learning-based~~

• IS ML / DL itself Rule-based?

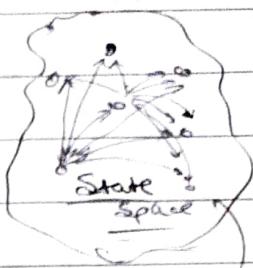
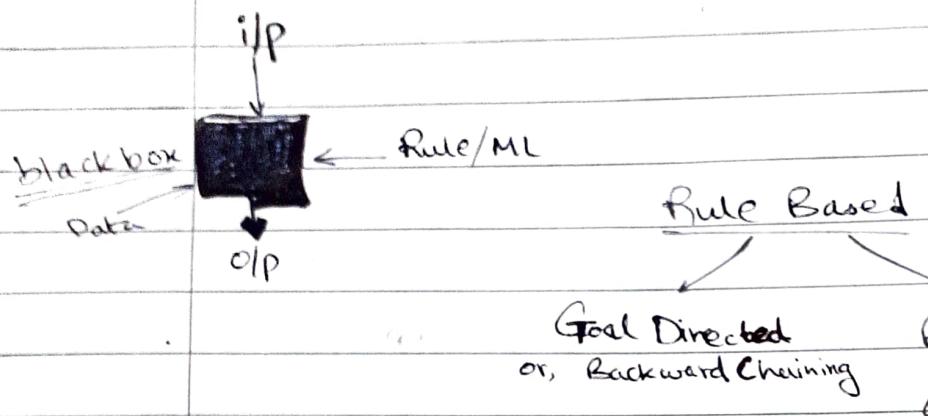
15/09/2022

* What is Rational, Ideal Agent.

* Challenge / Drawback of A* Algorithm.

* Environment Types.

- Revision: AI \rightarrow Rule Based State Space.

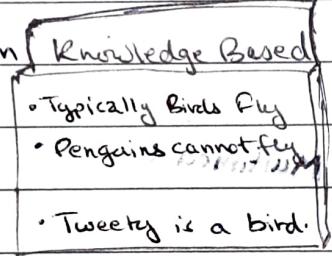


Monotonen

Who controls?
move from
One state to
another state.

Monotonic Reasoning

In monotonic, conclusion does not change.
(Universally True).



Non-monotonic Reasoning

In non-monotonic, conclusion may change depending on addition or subtraction of facts/statements.

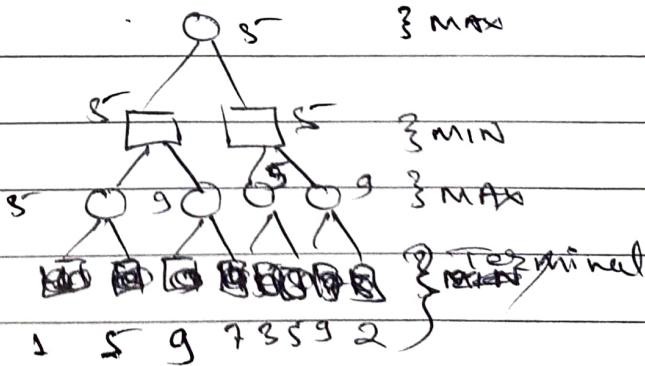
It is non-monotonic.

27/10/2022

- Minmax Algo. & Minimax Disadvantage

- Alpha-beta pruning.

Time complexity very high.



Alpha-Beta Pruning

First node

$$\alpha = -\infty$$

$$\beta = +\infty$$

MAX

$$\alpha = -\infty$$

$$\beta = +\infty$$

MIN

$$\alpha = -\infty$$

$$\beta = +\infty$$

MAX

$$\alpha = -\infty$$

$$\beta = +\infty$$

MIN

10 5 7 11 12 8 9 8 5 12 11 12 9 8 7 10

Terminal

$$\alpha = \text{MAX}$$

$$\beta = \text{MIN}$$

$\alpha \geq \beta$ at beginning, first node

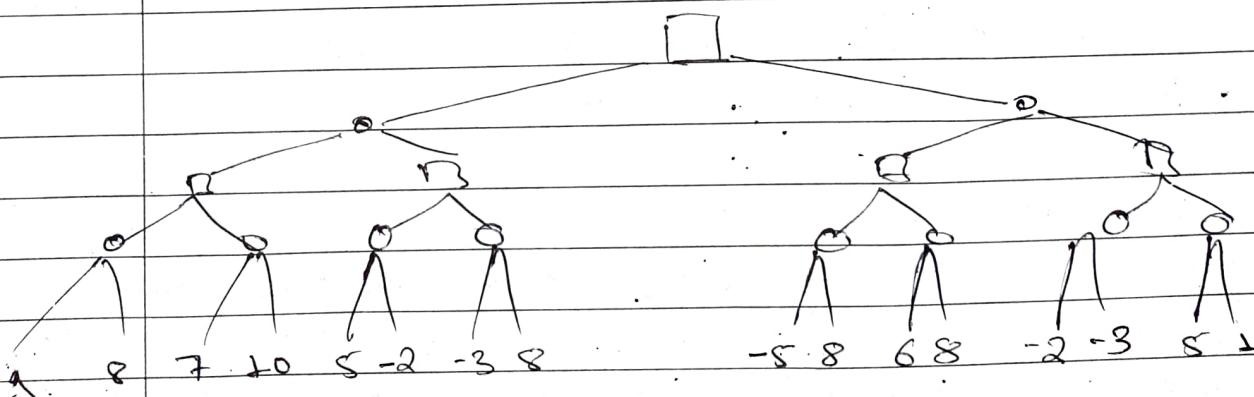
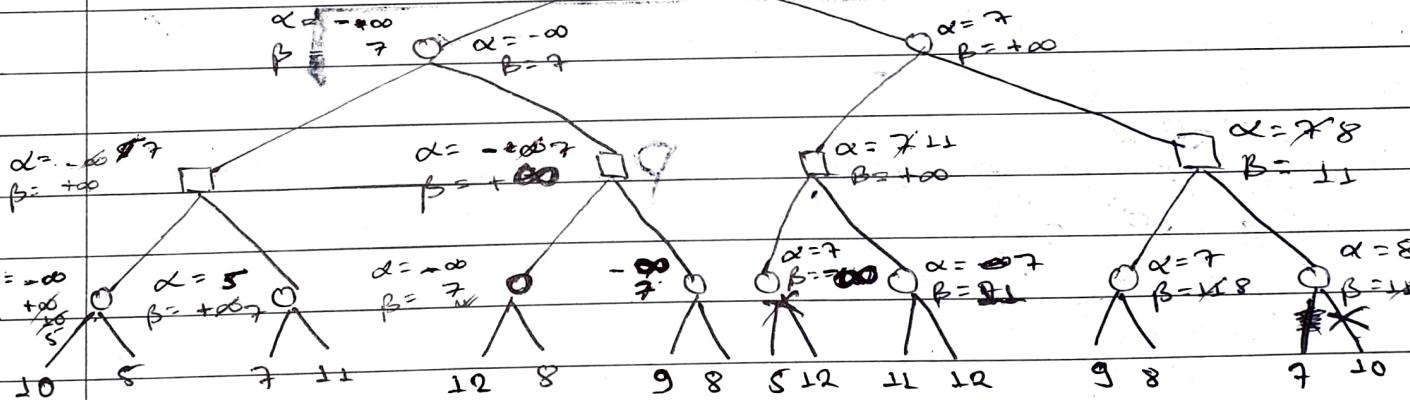
$$\alpha = -\infty$$

$$\beta = +\infty$$

[infinity]

$$\alpha = -\infty$$

$$\beta = +\infty$$



Knowledge Base : Resolution

07/11/2022

- * Unit Resolution:
- * Input Resolution:
- * Linear Resolution:

Derived Clause (dc)

Clause? Clock? Class?

Knowledge Base (B) or (KB) First order logic.

Input Resolution: One of the two clauses resolved upon must be from the original set of clauses (KB)

Linear Resolution: After the first resolution step, one of the two clauses resolved upon must be the last clause generated.

Completeness and Incompleteness?

Predicate Calculus.

09/11/2022

* Constraint Satisfaction Problems.

- Standard Search Problem? / State?

* CRYPT-ARITHMETIC

2	1	TO
+ G	1	+ GO
1	U	OUT
2	T	

- No two letters have same value.
- Sum of digits must be shown in problem.
- There should be only one carry forward.

carry is always 1. So, 0 → 1. $2+2=2 \rightarrow T \Rightarrow 2+G=U?$ U has carry. Possible values of G → 8, 9.

If $G=8$; $8+2=10$; $\boxed{1} \boxed{0}$ (Confirmed).

If $G=9$; $9+2=11$; $\boxed{1} \boxed{0}$ ($0 \neq 0$ two letters should not have same digit not possible)

$\therefore G=8$. $U=0$.

So, $T \rightarrow 2$. $O \rightarrow 1$; $G \rightarrow 8$; $U \rightarrow 0$

16/11/2022

* AI Applications:



chatbot, Deepfake, cyberbullying deEPFAKE



Automation deEPFAKE CYBERBULLYING

Crawling means crawling data from the websites, be it online or offline.

* Clap *

OPEN

Chatbot Challenges :-
- multiple questions at once
- Descriptive type questions

CHATBOTS

Lots of problems!
- Short form usage.

CLOSED DOMAIN!

Lots of problems!

MONOLOGUE!

OPEN DOMAIN!

REMEMBER
ENGINEER

MONOLOGUE! MONOLOGUE! RASA - 10 min KILL! KILL!

Dialogue MANAGEMENT System.



RULES

INTROVERTISM

17/11/2022

• Deepfake: evolved the past couple of years as a subset of AI that leverages neural networks to manipulate videos and photos while maintaining an authentic presence.

- fake Dubbing

Tools for voice generation →

• Fields of dubbing / fake dubbing?

- Identification of Artificially generated deepfake content

THAT

Lots of AI APPLICATIONS BASED ON SOCIAL MEDIA!

* CYBERBULLYING:

- CODENAME IN DATA
- FAKE NEWS DETECTION AS PER WIKIPEDIA
- IDENTIFICATION OF HATE CONTENT

- Fuzzy Reasoning X ~~Robot Solving Systems (very old systems)~~

- Bayes Theorem ✓ imp for sem.

Unit 4 - monotonic, nonmonotonic, Bayes theorem

Other than these, skip.

Stack Search

A* AO*

may or may
not come
but important

~~AO*~~ 15% from mid sem

Rest 85% from ~~after mid sem~~

all imp

Both Theory

and Numerical type combined

for theory
write examples
also.

MCQ Questions - Minor

Crypt Arithmetic (only for addition). { one question guaranteed for End Sem.

CRYPTARITHMETIC

If example is correct,
then, full marks!

Subhojit Ghimire, 1912LGO
CS-401.

09/11/2022

CRYPT-ARITHMETIC.

Q. If, SEND + MORE = MONEY, find the respective values.

$$\begin{array}{r} S \quad E \quad N \quad D \\ + \quad M \quad O \quad R \quad E \\ \hline M \quad O \quad N \quad E \quad Y \end{array}$$

9567
+ 1085
10652
Answer
cue

* Carry $\rightarrow M = 1$. (Always)

$$\begin{array}{r} S \quad E \quad N \quad D \\ + \quad 1 \quad O \quad R \quad E \\ \hline T \quad O \quad N \quad E \quad Y \end{array}$$

$$S+1 = 1'0'; S = \cancel{8} \cancel{9}$$

~~if S = 9; O = 0~~

~~if S = 8; O = 9~~

$$\therefore O = 0; S = \cancel{9}$$

$$\begin{array}{r} \cancel{9} \cdot E \quad N \quad D \\ + \quad 1 \quad O \quad R \quad E \\ \hline T \quad O \quad N \quad E \quad Y \end{array}$$

~~Carry 1 from T to O, 1 from O to N, 1 from N to E, 1 from E to Y~~

$$\text{Again, } 9'E' + 10 = 10'N'$$

Possible values of E = 2, 3, 4, 5, 6, 7, 8

Whatever be the value, E = N, but E \neq N cannot be.

$$\text{So, } 1+E+O = N. \quad (\because \text{it will have carry})$$

$$\text{So, Either, } N+R = E+10.$$

$$\text{OR, } N+R+1 = E+10$$

$$\text{So, } R = 8$$

$$\text{So, } N+8 = E; N, E > 1 \text{ and } N, E < 8$$

$$\text{Now, } D+E = Y \quad (\text{must have carry}).$$

$$S \rightarrow 9$$

$$D \rightarrow 7$$

$$R \rightarrow 8$$

$$E \rightarrow \cancel{8} 5$$

$$M \rightarrow 1$$

$$Y \rightarrow 2$$

$$N \rightarrow \cancel{8} 6$$

$$O \rightarrow 0$$