ACDS Lecture Series

Lecture - 6

Artificial Intelligence

G. N. Sastry and Team

ADVANCED COMPUTATION AND DATA SCIENCES (ACDS) DIVISION

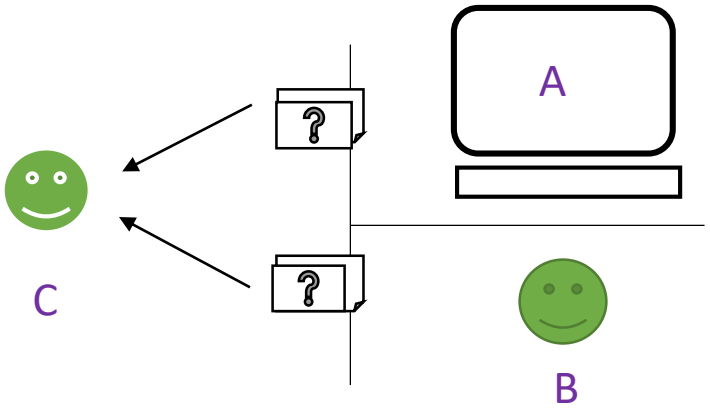
CSIR-North East Institute of Science and Technology, Jorhat, Assam, India

Rationalist Approach to Define AI

The Turing Test approach: Acting humanly

- A hypothetical theory proposed by Alan Turing (1950).
- A machine can be considered intelligent if it has the ability to achieve human-level performance in cognitive tasks.
- "The Imitation Game".

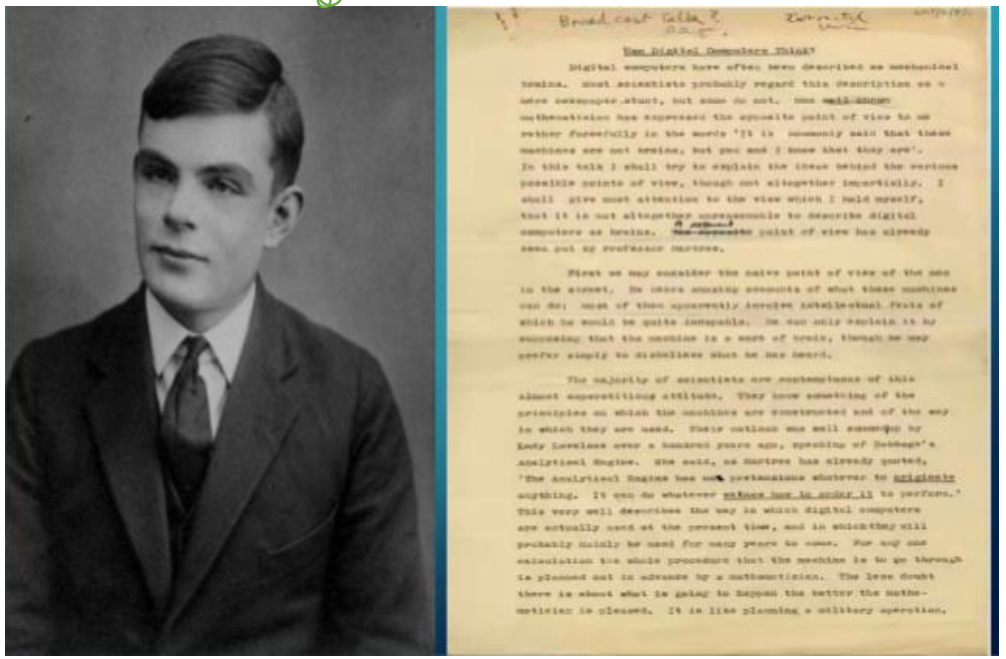
Interrogator ⟷ Machine



- Natural Language Processing
- Computer Vision
- Automated reasoning
- Machine learning
- Robotics

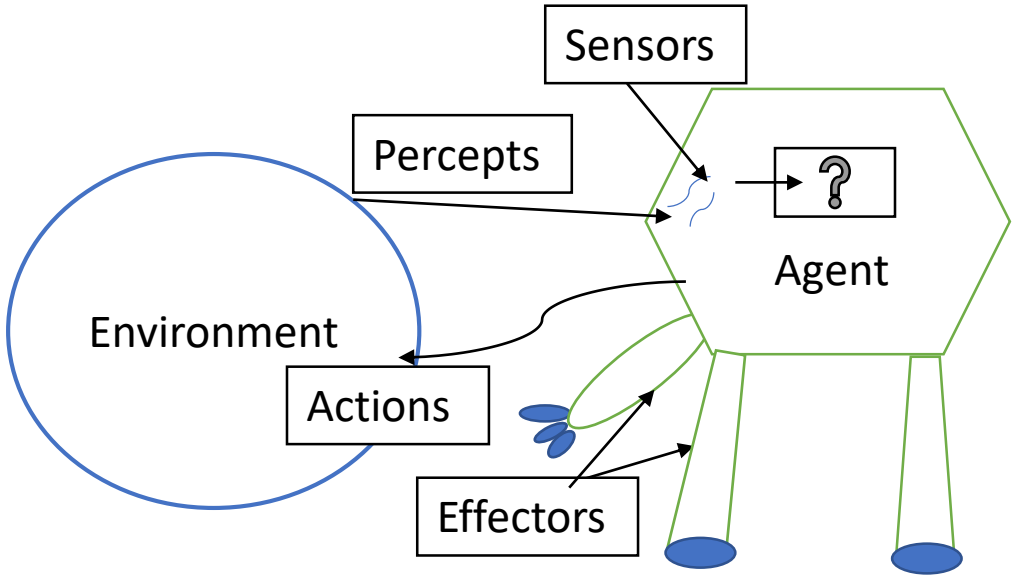
Thinking involves nerve cells to detect information from outside world and transmit to other nerve cells

Can machine think like human?



Agent and Environment

- An agent perceives its environment through sensors and acting through effectors.
- How agents act?
- Rational agent does the right thing!
- Performance measure defines degree of success.
- Sequence perception.
- What the agent knows about the environment.
- The actions that the agent can perform.



AI Agents

- An agent is **autonomous** that its action choices depend on its own experience, rather than on knowledge of the environment that has been built-in by the designer.
- Agent mapping: Function from percept sequences to actions.

Mapping Percepts to Actions

Ideal mapping for Square root problem

Specifying which action an agent ought to take in response to any given percept sequence provides a design for an ideal agent

```
function SQRT(x)
z ← 1 .0            / * initial guess * /

repeat until [z² - x] < 10⁻¹⁵

            z ←  z - (z² - x)/(2z)
end
return z
```

Percept (x)	Action (z)
1.0	1.000000000000000
1.1	1.048808848170152
1.2	1.095445115010332
1.3	1.140175425099138
1.4	1.183215956619923
1.5	1.224744871391589
1.6	1.264911064067352

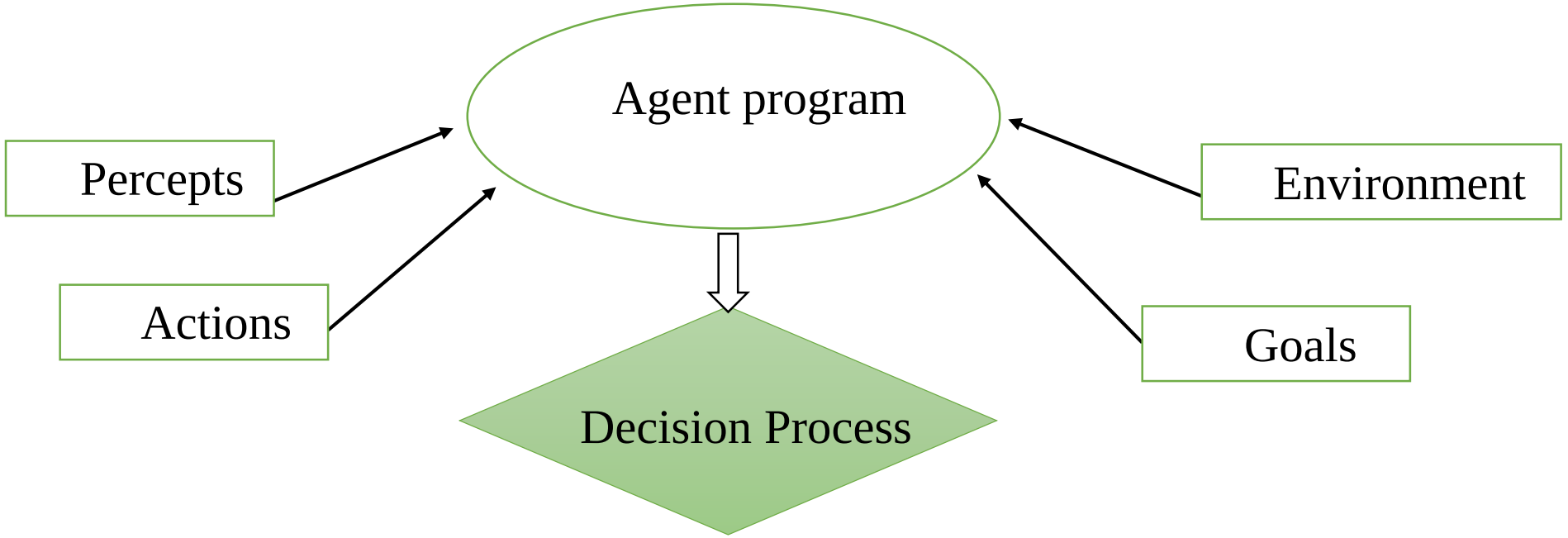
Examples of Agents, Percepts and Environments

Agents	Precepts	Actions	Goals	Environment
Researcher	New Idea	Perform experiments	Technology release, publish papers	Research institute, supervisor, lab mates
Student	Education	Give exams, extra-curriculum activities	High score on exams, high thinking, well social being	School, teachers, class mates
Medical Diagnosis System	Symptoms, findings, patient's answers	Questions, tests, treatments	Healthy patients, minimize costs	Patient, hospital
Satellite Imaging System	Pixels varying intensity, colour	Print a categorization of scene	Correct categorization	Images from orbiting satellite
Farmer	Soil, water, seeds	Crop cultivation	High	Agriculture land

Structure of Intelligent Agents

Agent program: a function that implements the agent mapping from percepts to action

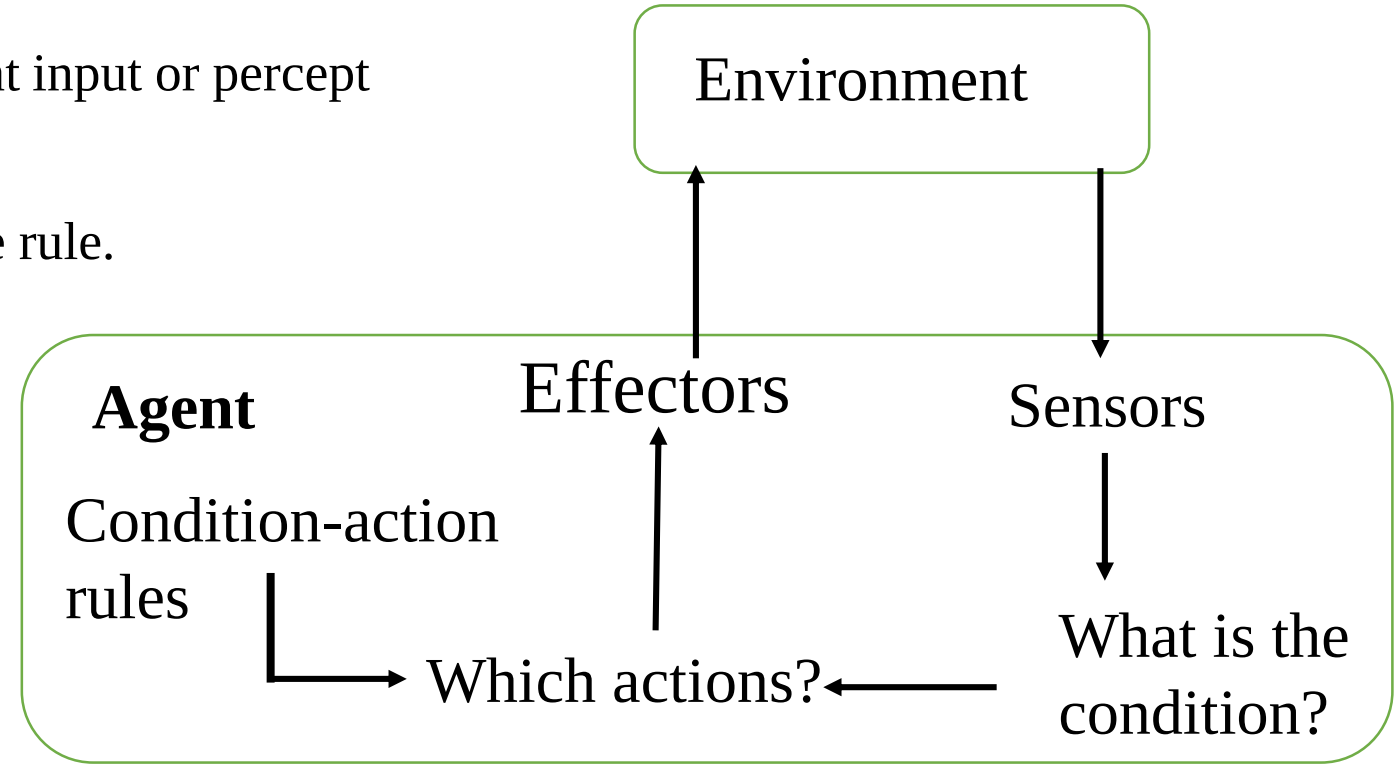
- Agent = architecture (computing device) + program
- An agent program maps from a percept to an action, while updating an internal state.



Types of Agent Programs

1. Simple-Reflex Agents

- Action depends only on the current input or percept
- Unable to plan ahead
- Perform action associated with the rule.



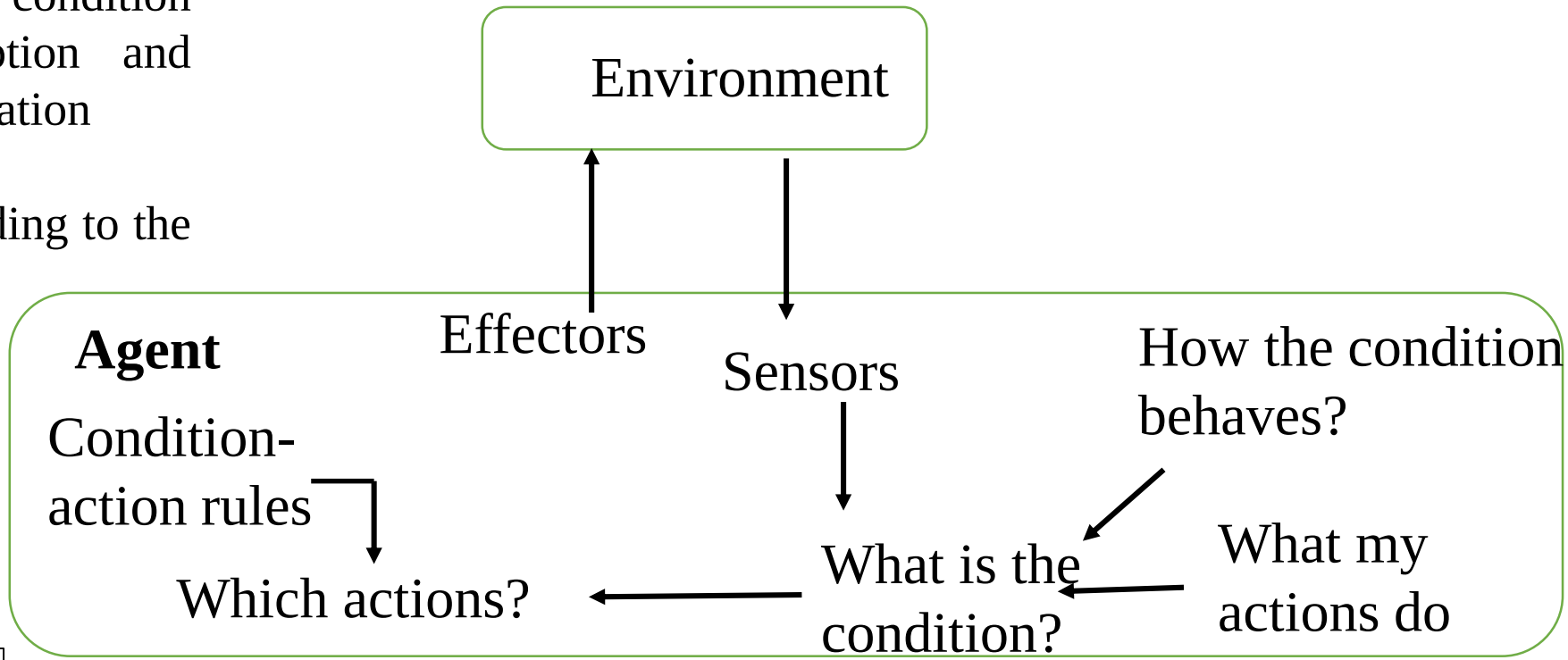
condition- action rule

If (Condition) Then (Action)

Eg. Cleaning agent will work if there is a dirt

2. Reflex Agent with Internal State (Model-based reflex)

- Find a rule whose condition matches the perception and internal stored information
- Perform action according to the associated rule.



maintains an internal state

Eg. Self steering mobile vision

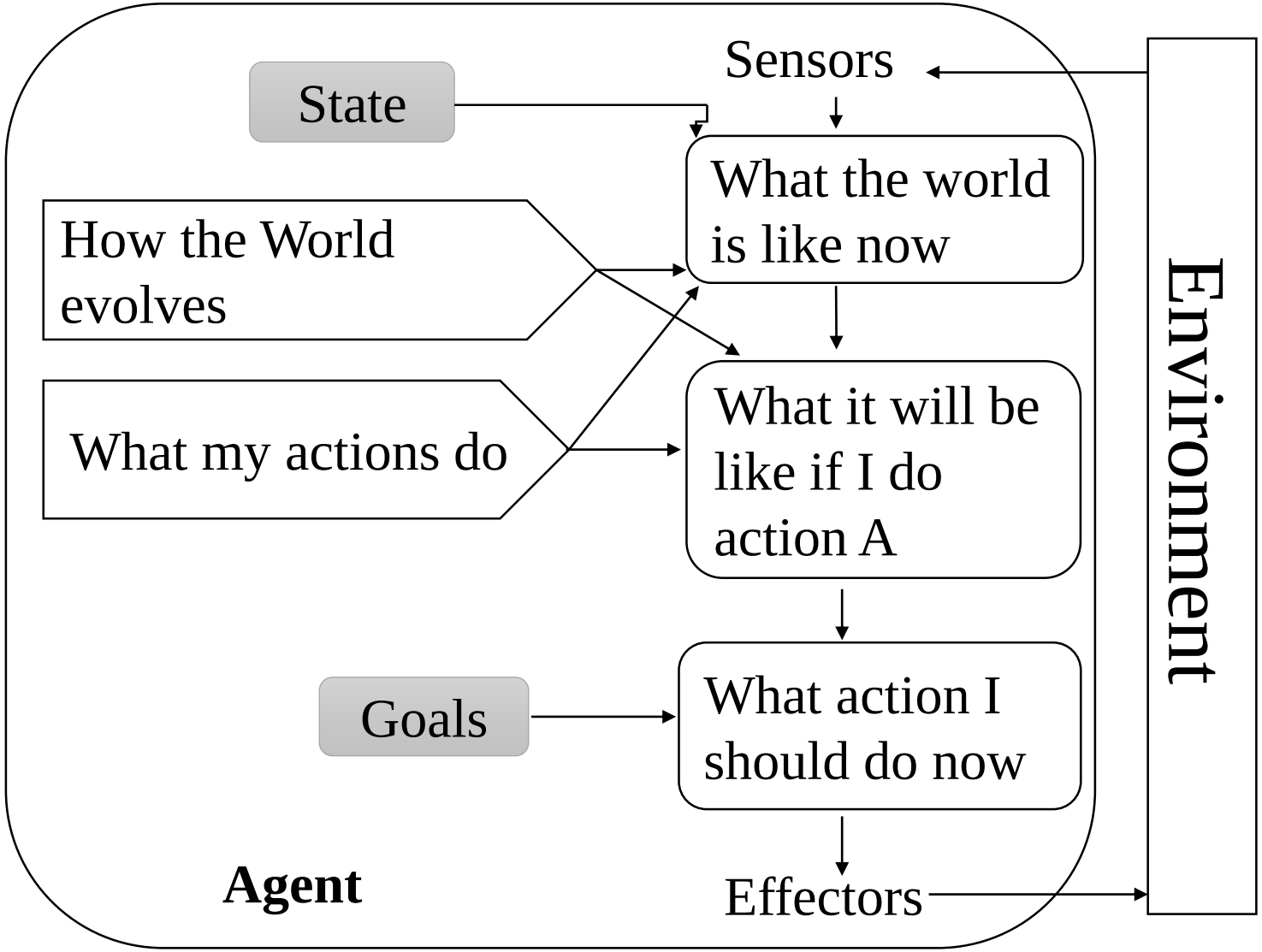
Types of Agent Programs

3. Goal-Based Agents

- Agents attempt to achieve a goal through a series of action.
- Need goal information to describe desirable situation
- Approach: Search through the solution space

Set of possible actions, choose the best

Eg. Searching robot with initial and final state



Types of Agent Programs

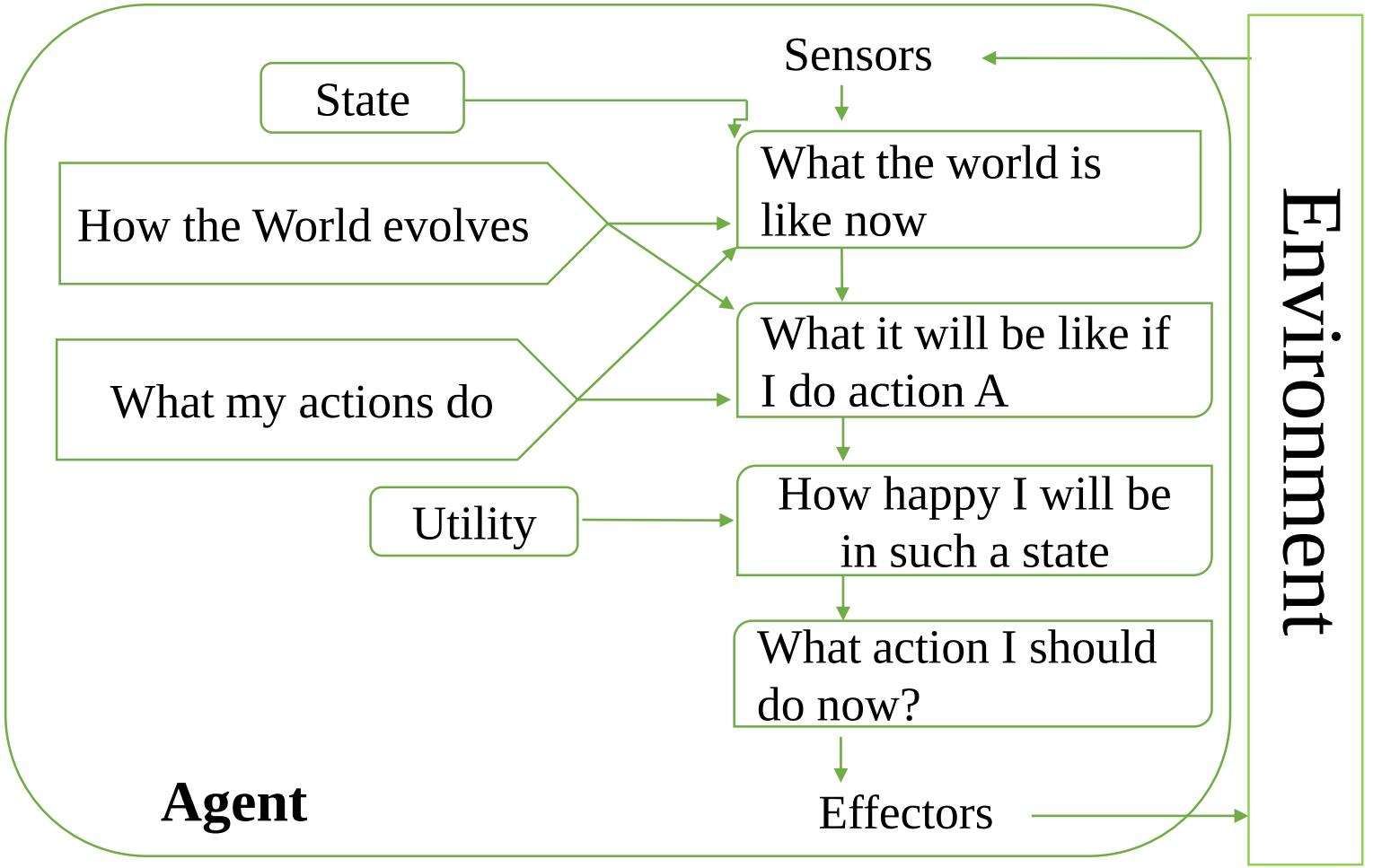
4. Utility-Based Agents

- For faster, safer, more reliable action sequences to achieve goal

- Higher utility for the agent, maximizing their own happiness

Measures to decide the best action

Eg. A route recommendation system to find the best route



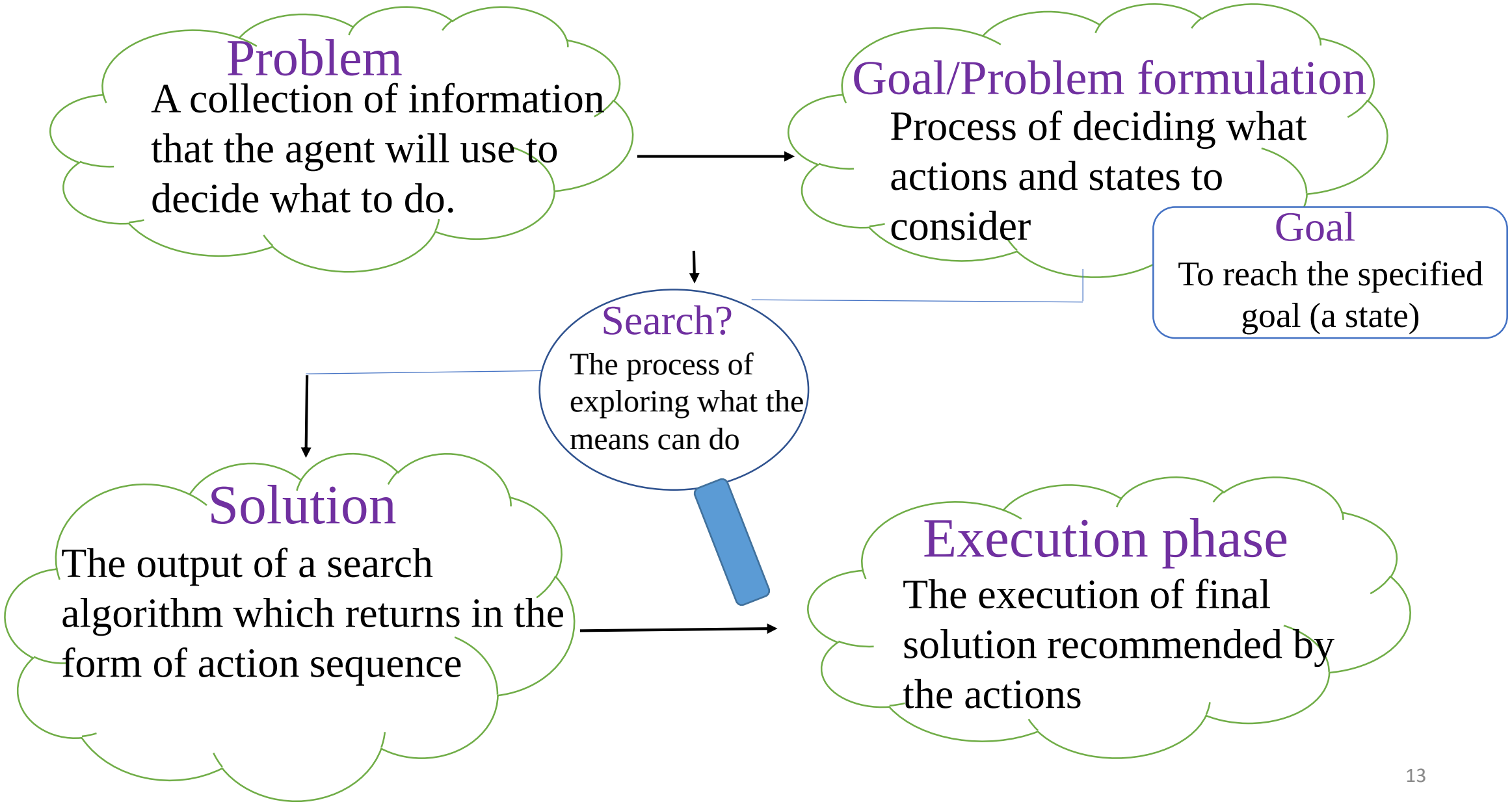
The Nature and Properties of Environments

Accessible vs. Inaccessible	Sensor can access to the complete state of the environment relevant to the choice of the actions
Deterministic vs. Non-deterministic	Deterministic (Certain) environment is completely determined by the current state and the actions selected by the agents whereas Non-deterministic is uncertain
Episodic vs. Sequential	Agent's current action does not affect a future action in episodic, in sequential, previous action is important for future action
Static vs. Dynamic	Static environment do not change while dynamic can change while an agent is deliberating. Static is easy to deal.
Discrete vs. Continuous	Discrete environment has limited distinct, clearly defined percepts and actions. Continuous has changing percepts and actions.

The Nature and Properties of Environments

Environment	Accessible	Deterministic	Episodic	Static	Discrete
Chess with a clock	Yes	Yes	No	Semi	Yes
Chess without a clock	Yes	Yes	No	Yes	Yes
Poker	No	No	No	Yes	Yes
Backgammon	Yes	No	No	Yes	Yes
Taxi driving	No	No	No	No	No
Medical diagnosis system	No	No	No	No	No
Image-analysis system	Yes	Yes	Yes	Semi	No

Solving Problems by Searching



Solving Problems by Searching

- The first step of an agent before searching a solution is to formulate a goal and then use that goal to formulate a problem.

- Components of a well defined

  Problem: state space + initial situation + actions/operators + goal test + path costs.

  Solution = A path from an initial state to a goal state.

- A general search algorithm can be used to solve any problem. Specific variants of the algorithm can use different search strategies.

Terms and concepts of well defined problems and solutions

- State space : Set of all possible states
- Initial state : The state from which the agent infers that it is at the beginning.
- Actions/operators : Description of possible actions and their outcome.
- Goal test : Tests whether the state description matches a goal state.
- Path : A sequence of actions leading from one state to another.
- Path costs : Cost function over paths. Usually the sum of the costs of the actions along the path.
- Solution : Path from an initial to a goal state
- Search costs : Time and storage requirements to find a solution
- Total costs : Search costs + path costs

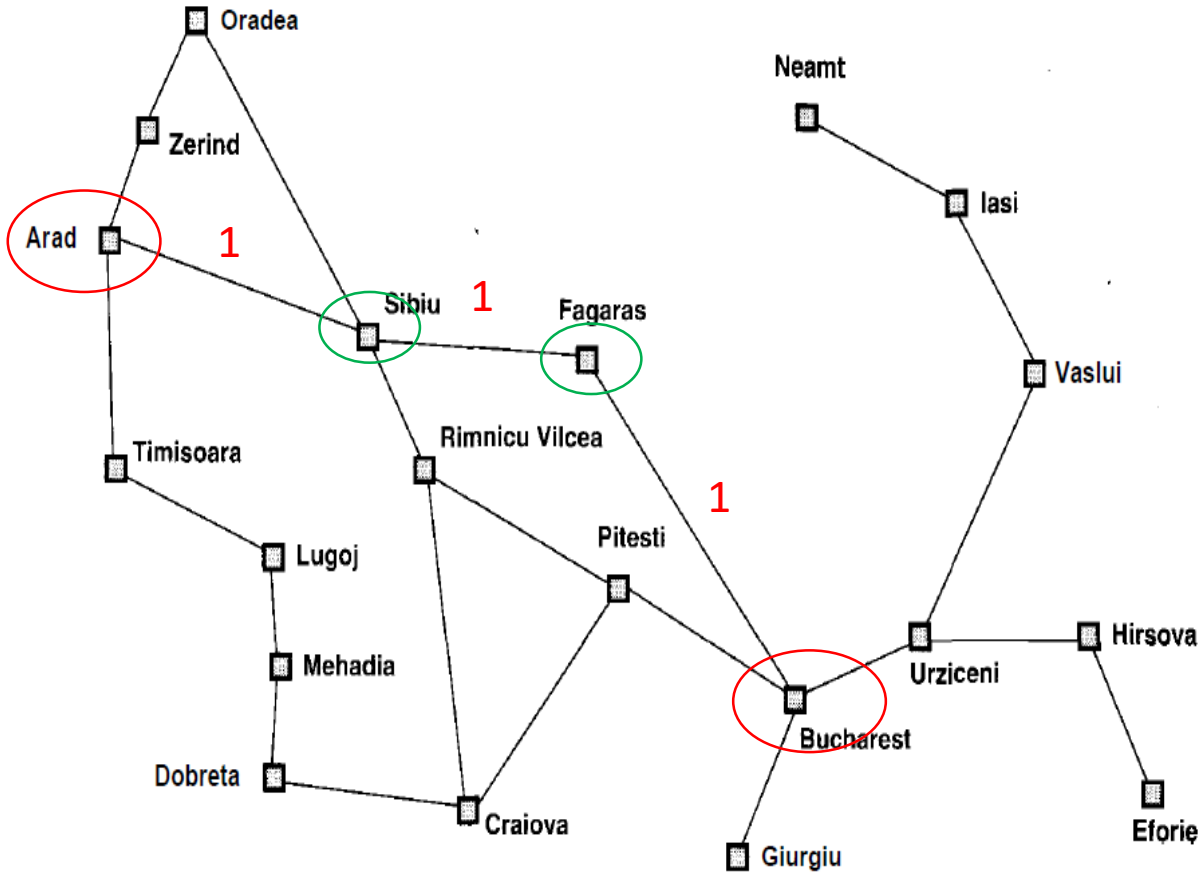
Deciding states and actions

- State space = 20 cities
- Initial state = Arad
- Goal test = "Is this Bucharest?"
- Solution: so many
- Path cost function = total mileage/expected time travel/number of steps
- Best possible solution = least path cost

i.e path through Sibiu and Fragaras

Cost Function = (Arad -> Sibiu = 1) + (Sibiu -> Fagaras = 1) + (Fagaras -> Bucharest = 1)

Total = 3 (least)



Search Criteria

- Decides which state to expand for generating successive states.

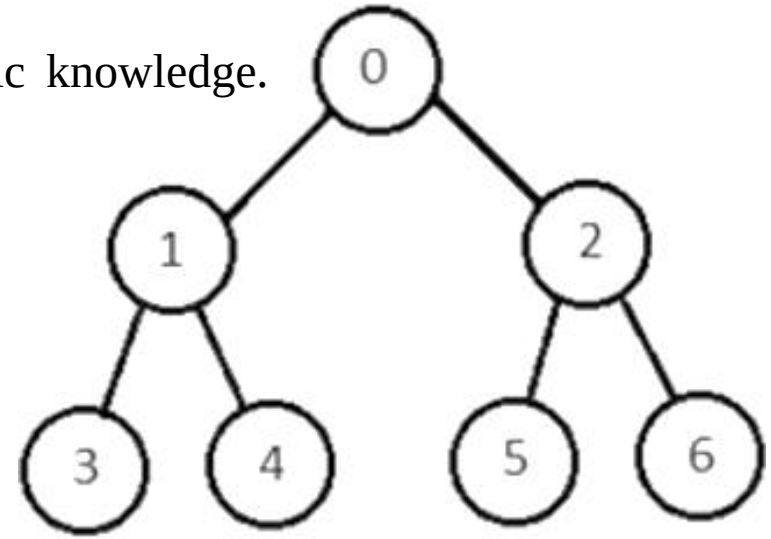
Criteria

1. Completeness: is the strategy guaranteed to find a solution when there is one?
2. Time complexity: how long does it take to find a solution?
3. Space complexity: how much memory does it need to perform the search?
4. Optimality: does the strategy find the highest-quality solution when there are several different solutions (with the lowest path cost)?

- ## Uninformed (Brute–Force) Search Strategies

They are most simple, as they do not need any domain-specific knowledge. They work fine with small number of possible states.

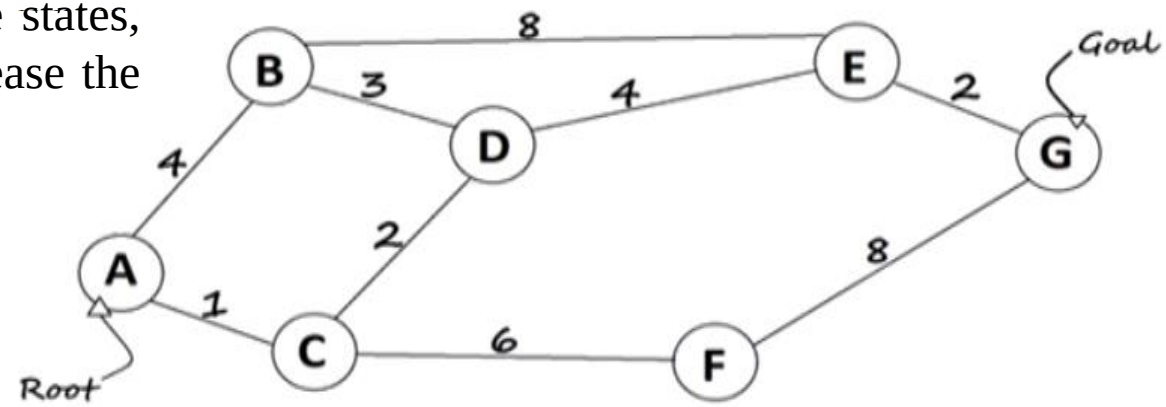
1. Breadth-first search
2. Uniform cost search
3. Depth-first search
4. Depth-limited search
5. Iterative deepening search
6. Bi-directional search.



- ## Informed search/Heuristic Search Strategies

To solve large problems with large number of possible states, problem-specific knowledge needs to be added to increase the efficiency of search algorithms.

1. Greedy Best First Search
2. A* Search

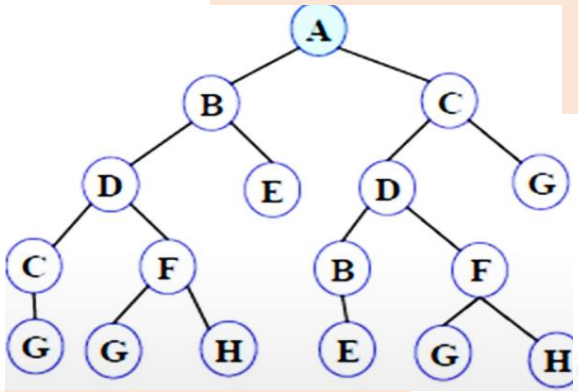


Breadth-First Search

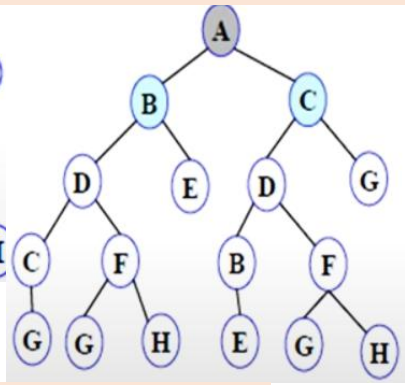
- It starts from the root node, explores the neighboring nodes first and moves towards the next level neighbors. It generates one tree at a time until the solution is found. It can be implemented using FIFO queue data structure. This method provides shortest path to the solution.

Example:-

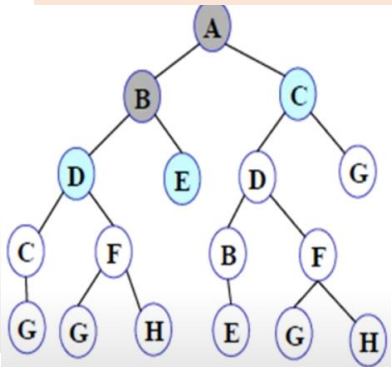
Step-1 Initially the fringe contains only one node
Fringe: A



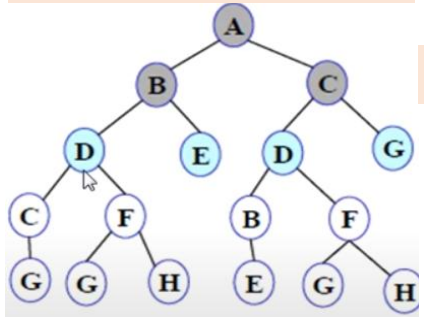
Step-2 A is removed from the fringe. The node is expanded and it's children B and C are generated



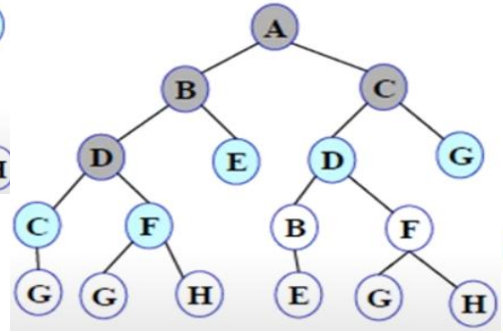
Step-3 Fringe: C D E



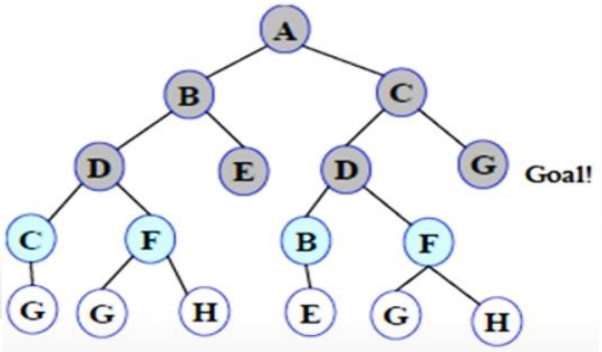
Step-4 Fringe:D E D G

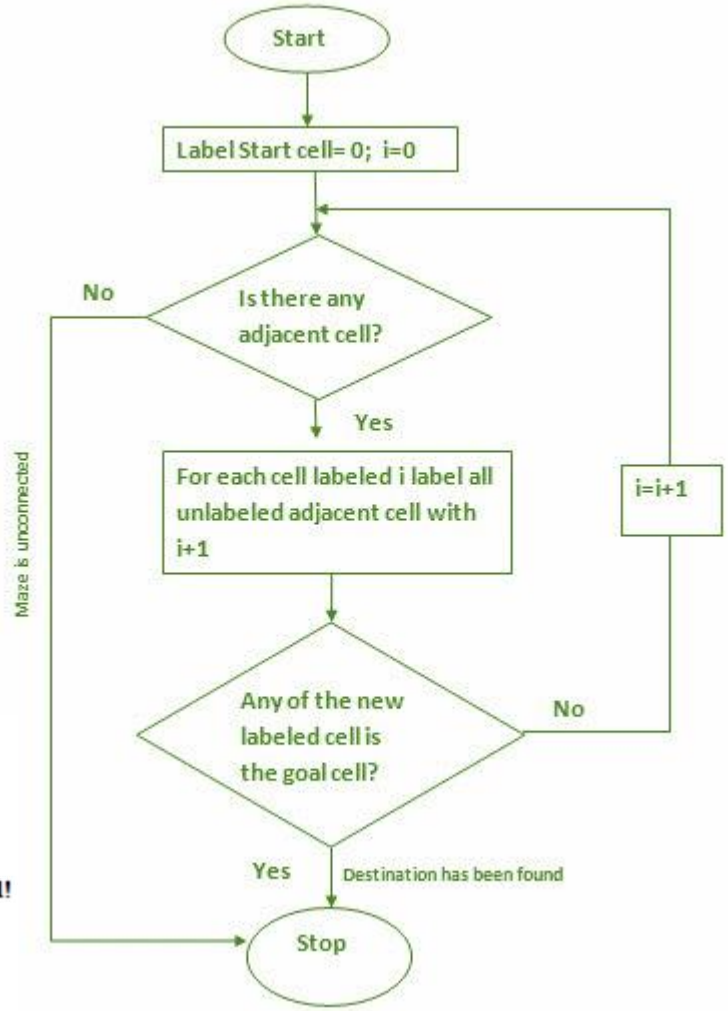


Step-5 Fringe: E D G C F



Step-6 Fringe: G C F B F





Breadth-First Search

Maximum number of nodes expanded before getting a solution is

$b + b^2 + b^3 + b^4 + \ldots\ldots + b^d$

Where, b = branching factor, d = depth of a solution path

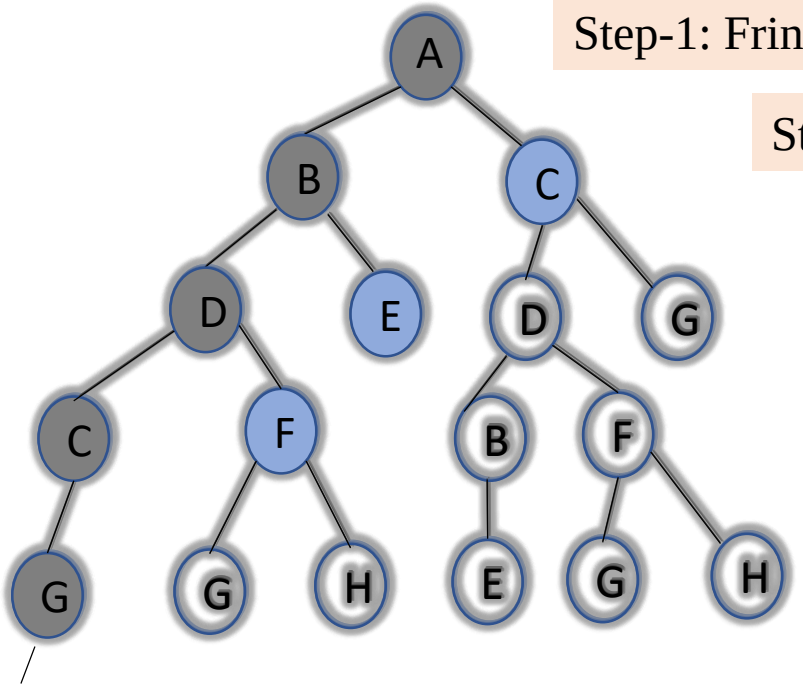
Depth	Nodes	Time	Memory
0	1	1 millisecond	100 bytes
2	111	.1 seconds	11 kilobytes
4	11,111	11 seconds	1 megabyte
6	10^6	18 minutes	111 megabytes
8	10^8	31 hours	11 gigabytes
10	10^{10}	128 days	1 terabyte
12	10^{12}	35 years	111 terabytes
14	10^{14}	3500 years	11,111 terabytes

Limitations:

1. Memory requirements
2. Time requirements

Depth–First Search

- The depth first search algorithm puts newly generated nodes in the front of OPEN.
- This results in expanding the deepest node first.
- Thus the nodes in OPEN follows a LIFO order(Last In First Out).
- OPEN is thus implemented using a stack data structure.



Step-1: Fringe: A

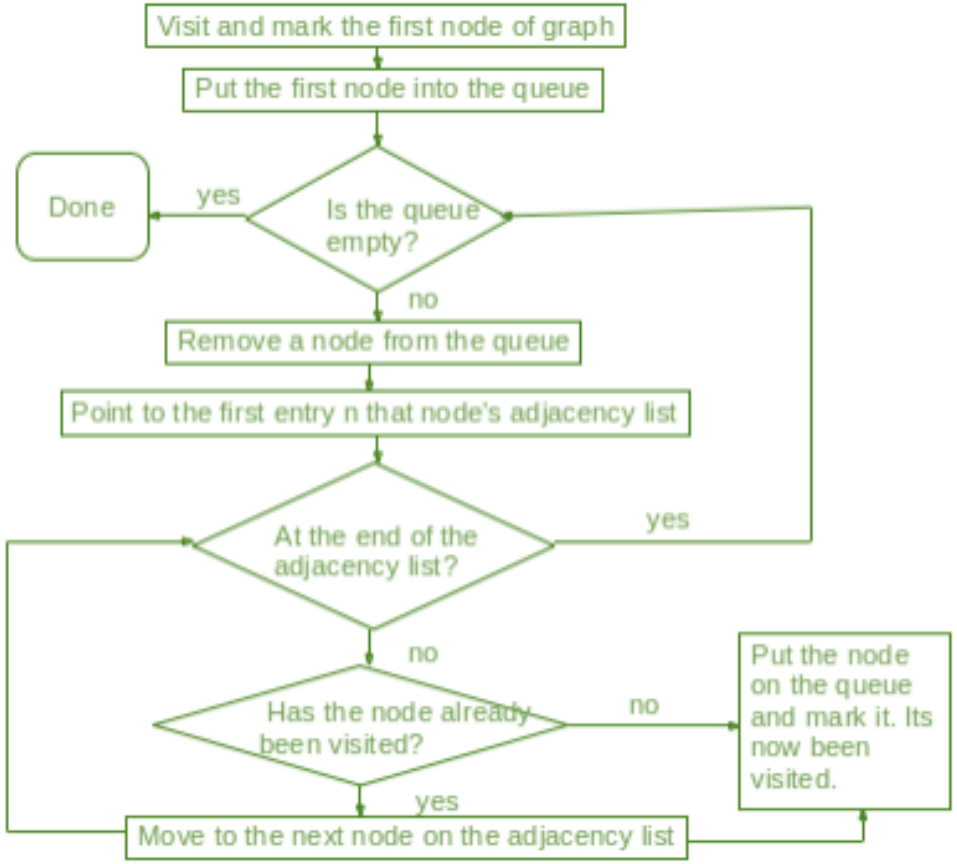
Step-2: Fringe: B C

Step-3: Fringe: D E C

Step-4: Fringe: C F E C

Step-5: Fringe: G F E C

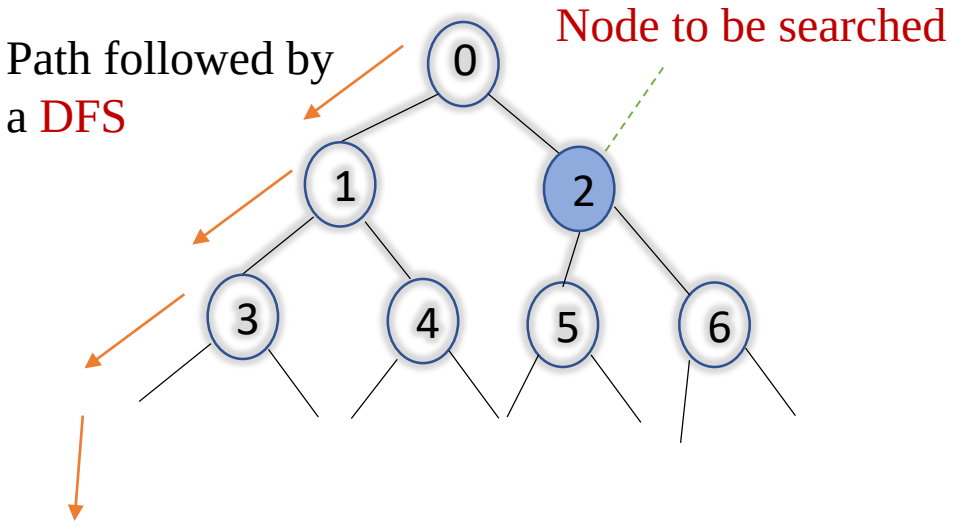
Step-6: Fringe: G F E C



Iterative Deepening Search

There are two common ways to traverse a graph, BFS and DFS. Considering a Tree of huge height and width, both BFS and DFS are not very efficient due to following reasons.

1. DFS first traverses nodes going through one adjacent of root, then next adjacent. The problem with this approach is, if there is a node close to root, but not in first few subtrees explored by DFS, then DFS reaches that node very late. Also, DFS may not find shortest path to a node.



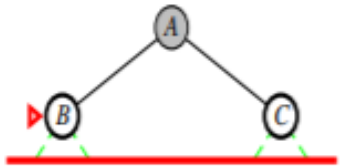
- Suppose, we want to find node – '2' of the given infinite undirected graph/tree.
- A DFS starting from node- 0 will dive left, towards node 1 and so on. Whereas the node 2 is just adjacent to node 1.
- Hence a DFS wastes a lot of time in coming back to node 2.
- An Iterative Deepening Depth first search overcomes this and quickly find the required node.

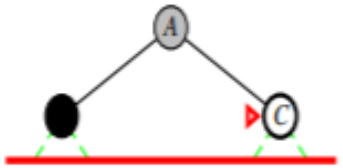
2. BFS goes level by level, but requires more space. The space required by DFS is O(d) where d is depth of tree, but space required by BFS is O(n) where n is number of nodes in tree.

Example

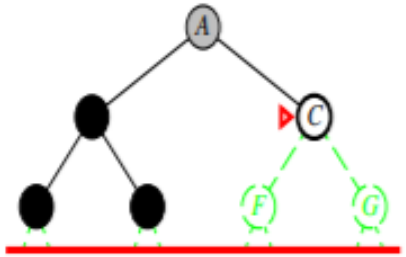
Limit = 0

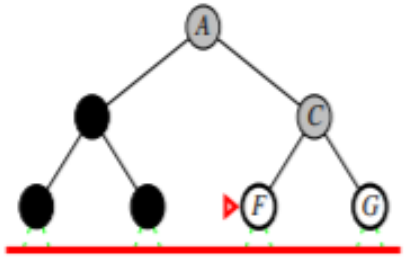
Limit = 1

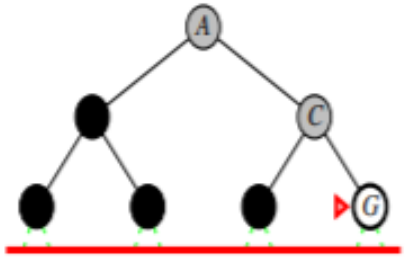


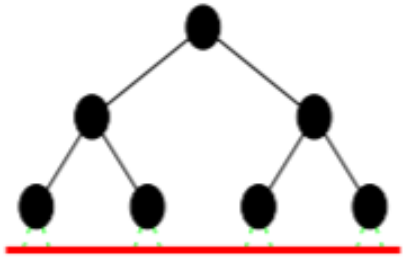


Limit = 2

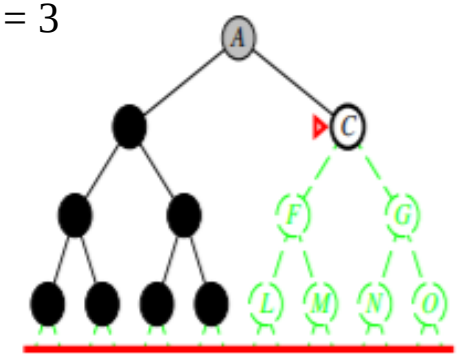


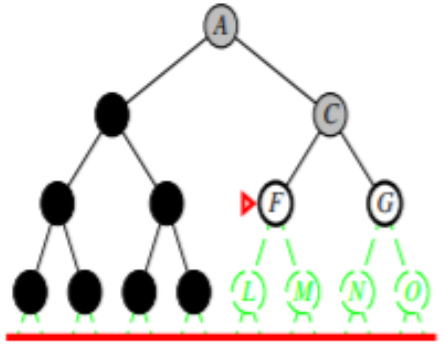


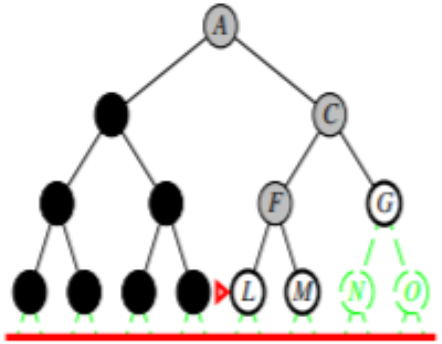


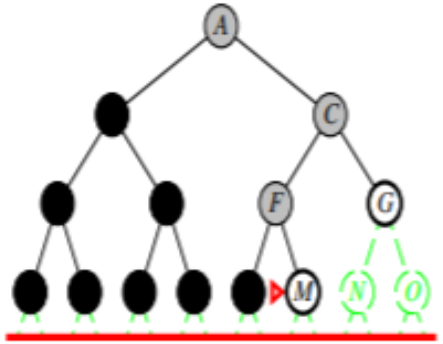


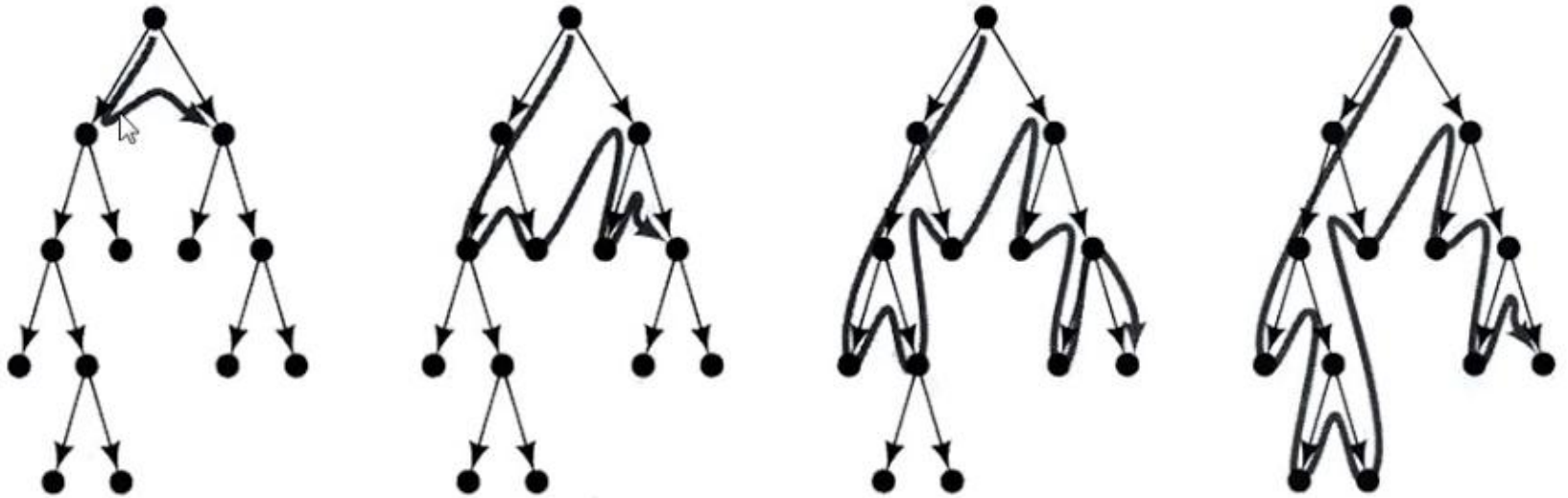
Limit = 3











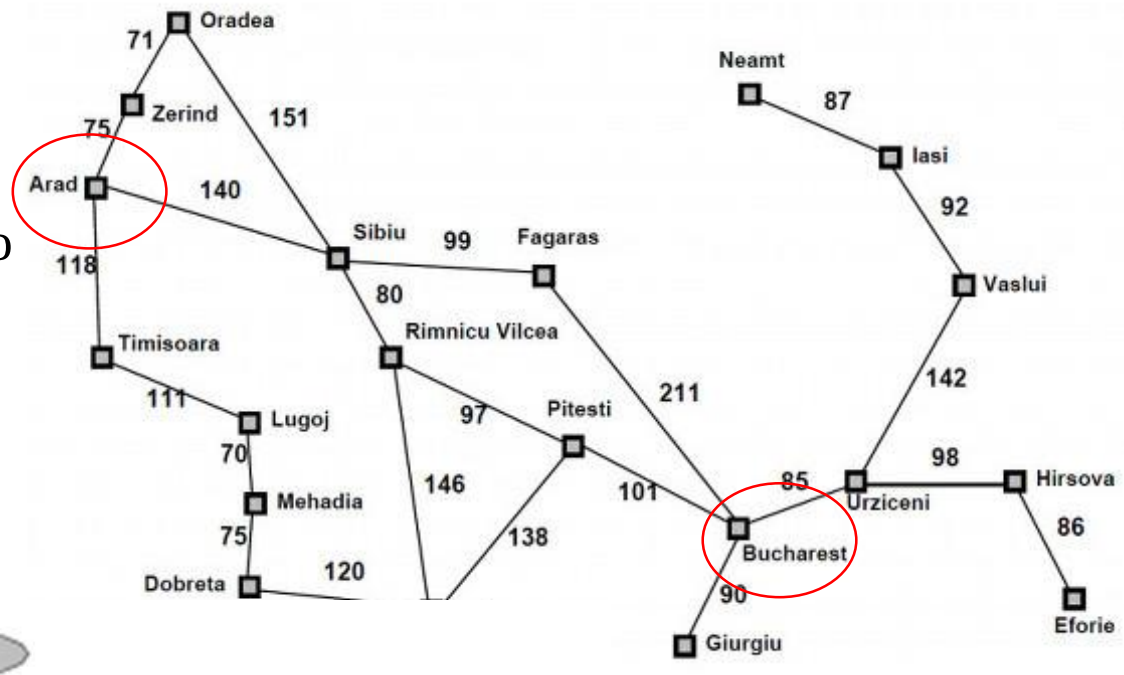
Stages in Iterative-Deepening Search

Comparison table between DFS, BFS and IDDFS

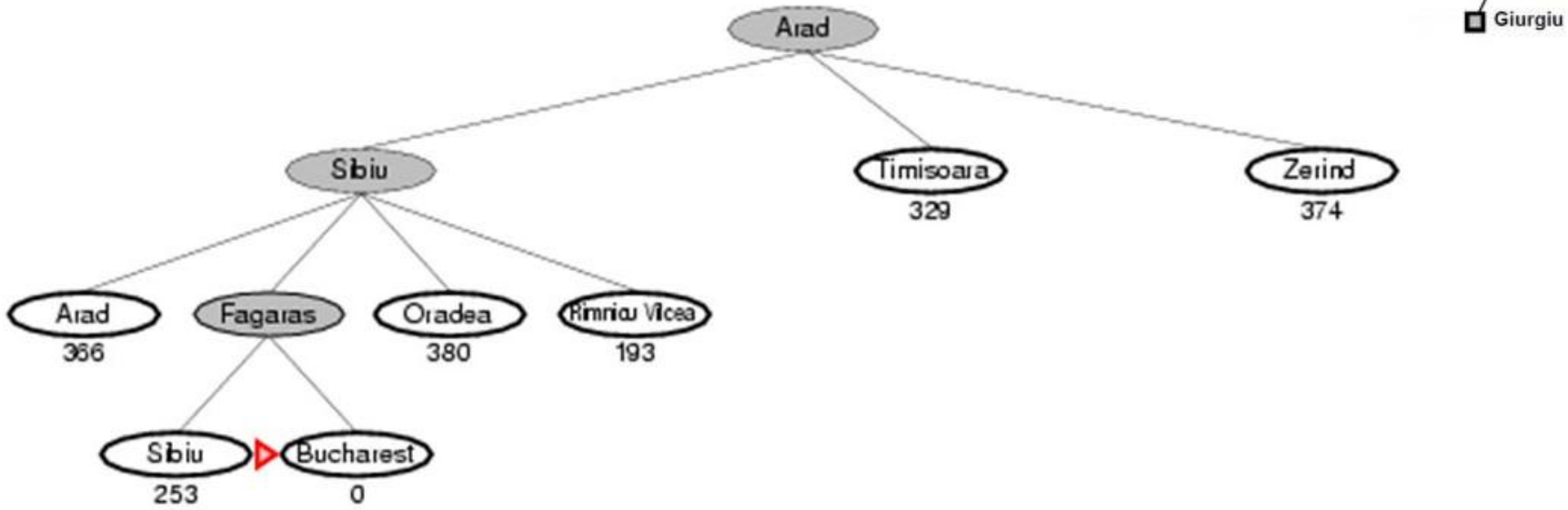
	Time Complexity	Space Complexity	When to Use
DFS	$O(b^d)$	O(d)	• Don't care if the answer is closest to the starting root. • When graph/tree is not very big/infinite
BFS	$O(b^d)$	$O(b^d)$	• When space is not an issue • When we do care/want the closest answer to the root
IDDFS	$O(b^d)$	O(bd)	• BFS + DFS

Greedy Best-First search

- F(n) = estimate of cost from n to goal
- Greedy best-first search expands the no that appears to be closest to goal.
- It expands nodes based on f(n) = h(n)



Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374



Informed Search (A* Search)

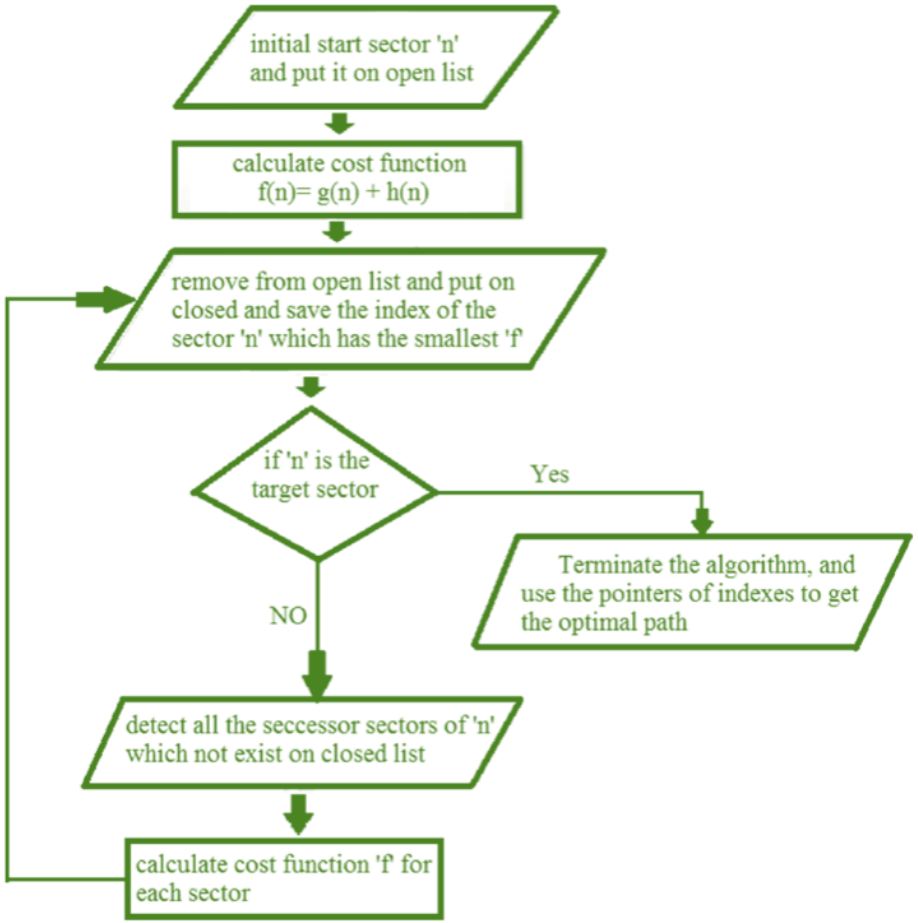
- Idea: avoid expanding paths that are already expensive.

  Evaluation function:

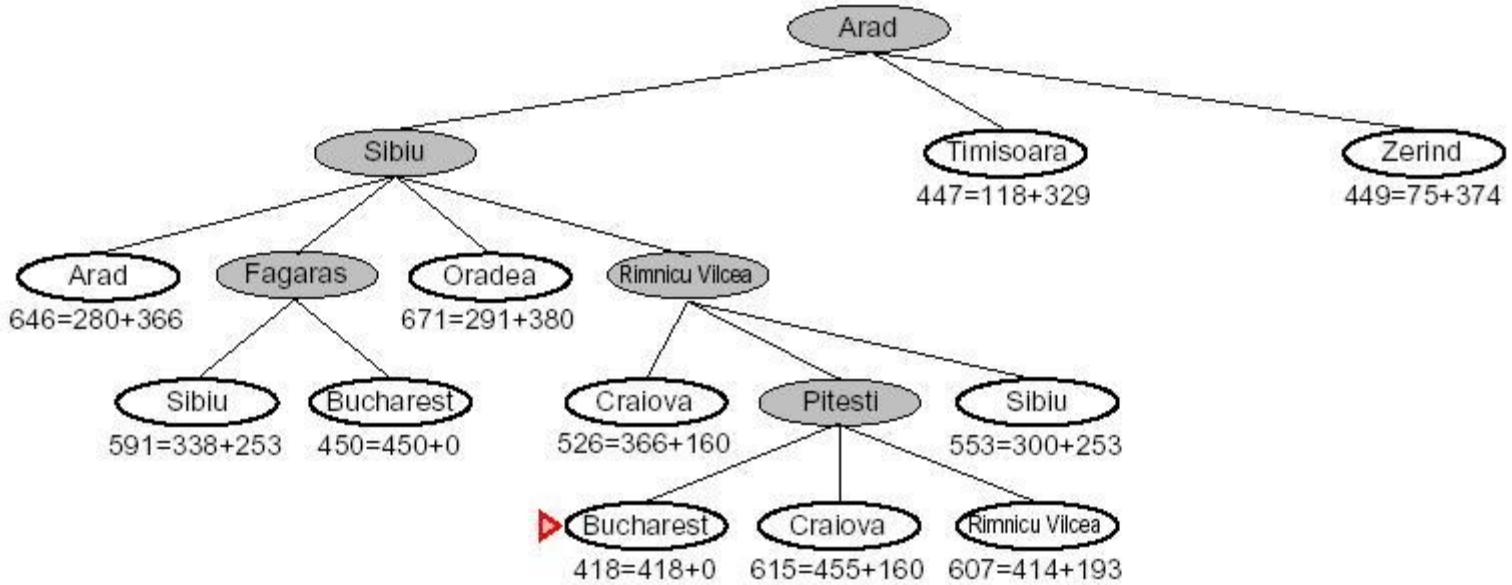
$$f(n) = g(n) + h(n)$$

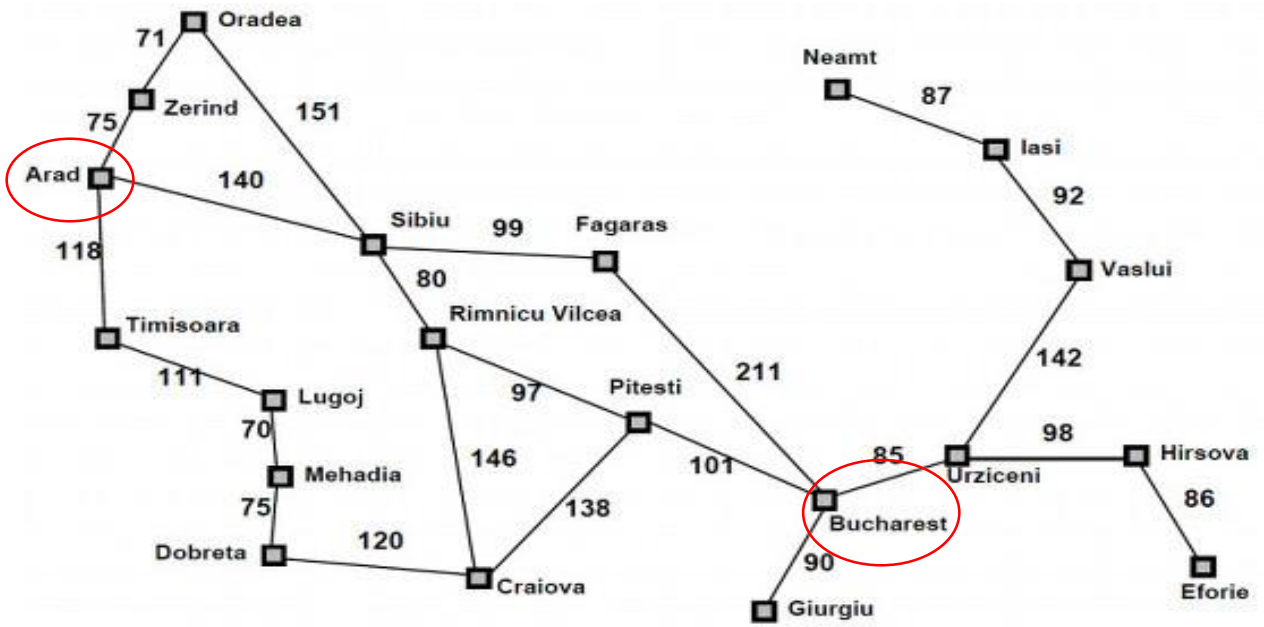
  - g(n) = cost so far to reach n
  - h(n) = estimated cost from n to goal
  - f(n) = estimated total cost of path through n to goal
  - Best First search has f(n) = h(n)

- Choose an admissible heuristic which never overestimates the cost to reach the goal i.e optimistic.
- Completeness
- Optimality
- Monotonicity



Example





Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Game Playing

- Games are well-defined problems that are generally interpreted as requiring intelligence to play well
- A game consists of a set of two or more players, a set of moves for the players, and a specification of payoffs (outcomes) for each combination of strategies
- Examine the problem which arises when we try to plan ahead of the world and other agents are against us.

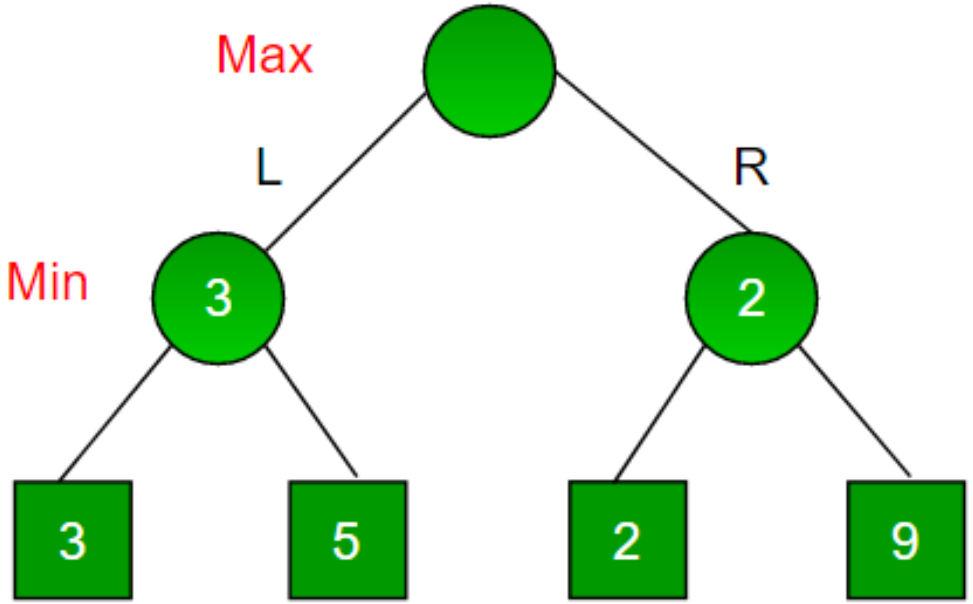
.

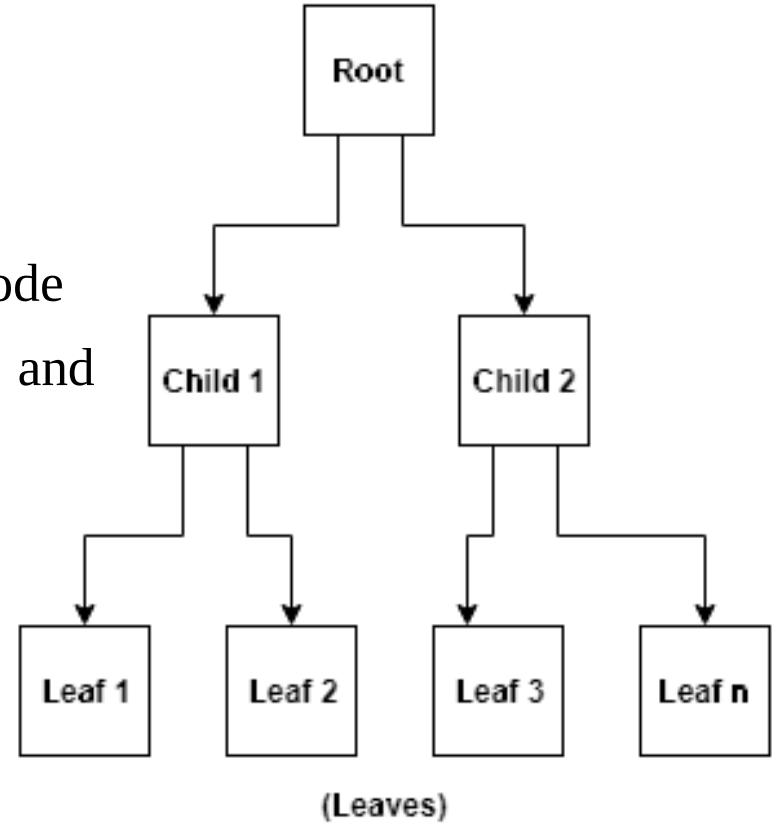
Types of Games in AI

Accessibility	Deterministic	Chance Moves
Perfect information	Chess, Checkers, go, Othello	Backgammon, monopoly
Imperfect information	Battleships, blind, tic-tac-toe	Bridge, poker, scrabble, nuclear war

Game Tree

- A game tree represents a game in the form of a layered tree, nodes of the tree are the game states and edges of the tree are the moves by players.
- At each alternating level, one of the player makes the choices
- The layers/levels in a game are called MAX layers and MIN layers.
- A game starts at the root with MAX playing first and ends at the leaf node
- Leaves of the game tree are labelled with the outcome of the game and game ends there.





Tic-Tac-Toe Game Tree Example

- Involves initial state, actions function, and result Function.
- Key points of the game:

  - 2 players: MAX and MIN.

  -The objective is to write a computer program in such a way that computer wins most of the time

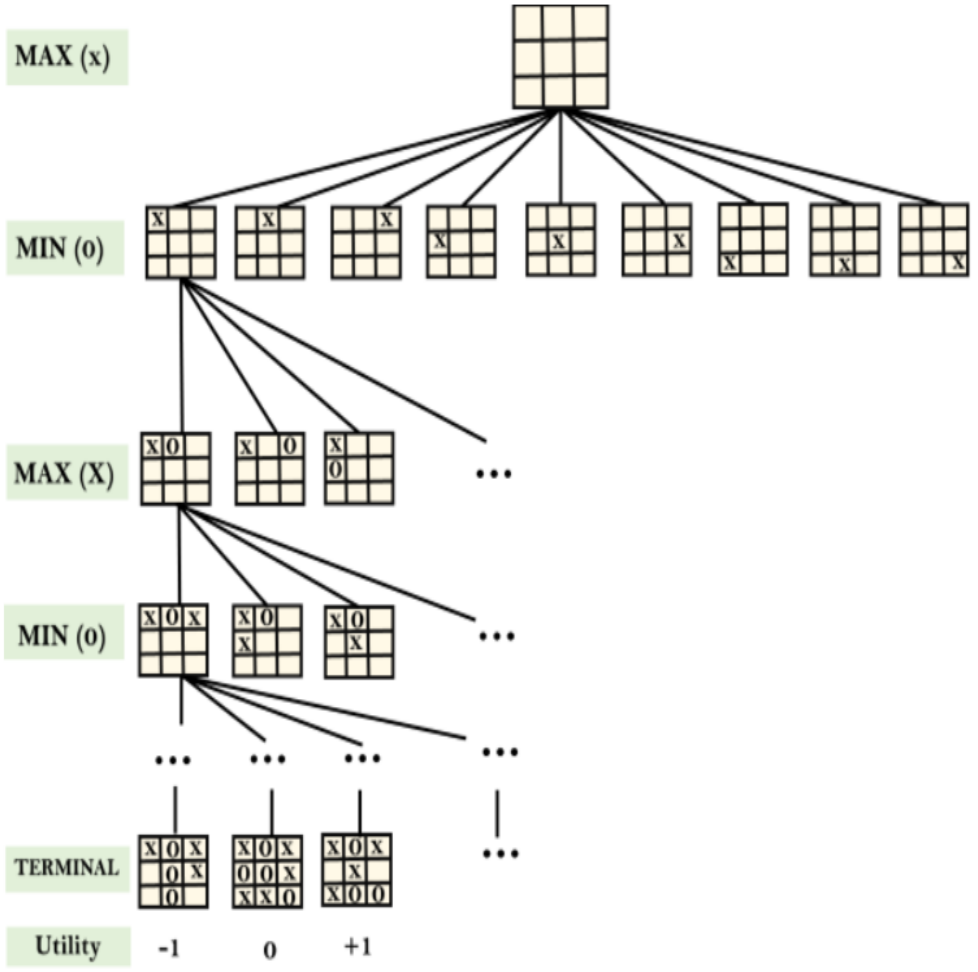
  -Three approaches are presented to play the game which increases in:

    - Complexity
    - Use of generalization
    - Clarity of their knowledge
    - Extensibility of their approach

  - Players have alternate turn and MAX starts.

  - MAX maximizes the result of the game tree

  - MIN minimizes the result.



Adversarial Search

- A search algorithm is an algorithm for finding an item with specified properties among a collection of items.

The items may be stored individually as records in a database; or may be elements of a search space defined by a mathematical formula or procedure.

- Adversarial search in Game playing :

It relates to competitive environment in which the agent goals are in conflict giving rise to adversarial search

Assumption & Aim

- 2 agents whose actions alternate.
- Utility values for each agent are the opposite of the other which creates adversarial situations.
- Fully observable environments.

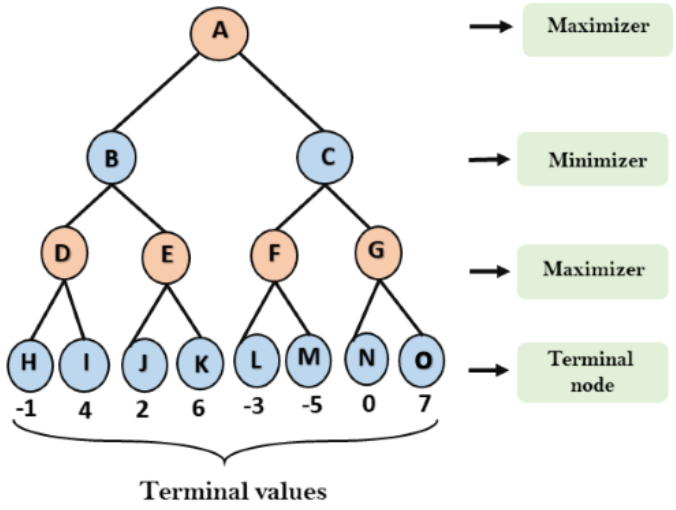
Applications in Game Playing

- Firstly, it should be a 2-player game
- Players alternate the moves
- Both should have perfect information
- Using dice is not involved
- Clear rules for legal moves
- Well defined outcomes.

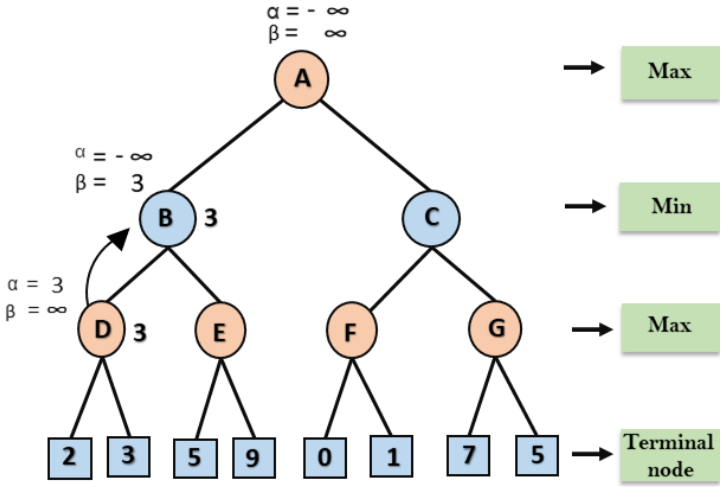
Adversarial Elements

- Initial state(SO): how the game is set up at the start.
- Operators/Players(s): Define which player has the turn
- Action(s): Returns the set of legal moves in a state
- Result(s,a): the result of moves in the state space
- Terminal test(s): determines when the game is over. The state where the game ends is called terminal states.
- Utility function(s,p): gives a numeric value for the outcome of a game.

e.g Chess: Win, loss or draw (+1, -1 or 0)

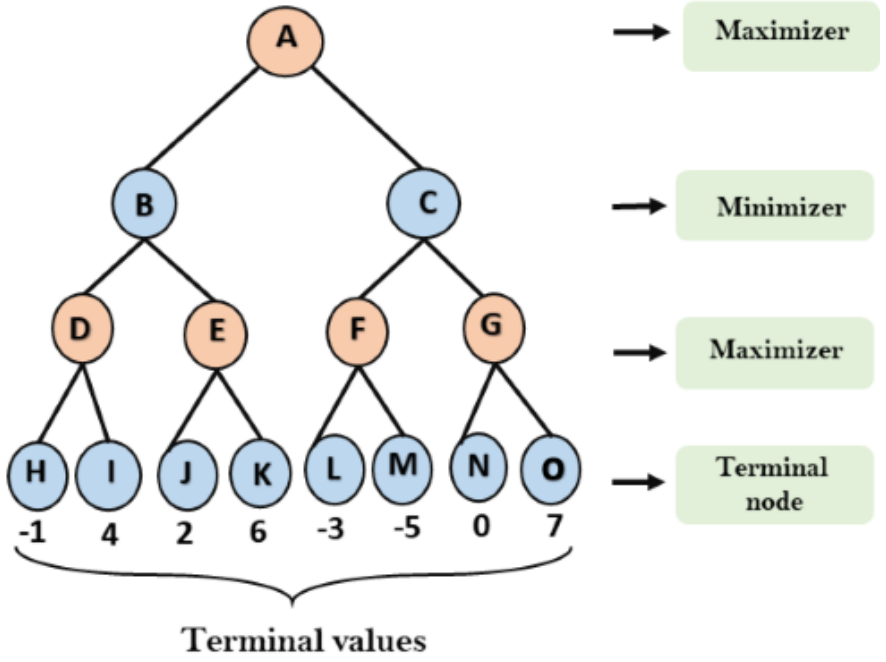


Popular adversarial search algorithms
- ❑ Min-Max algorithm
- ❑ α-β Pruning



Mn–Max Algorithm

- It is widely used in two player based games .
- It is an algorithm that is used to find optimal moves for the player assuming that the opponent plays optimally.
- Two players: MAX and MIN
- The maximizer tries to get the highest score possible while the minimizer tries to get the lowest score possible .

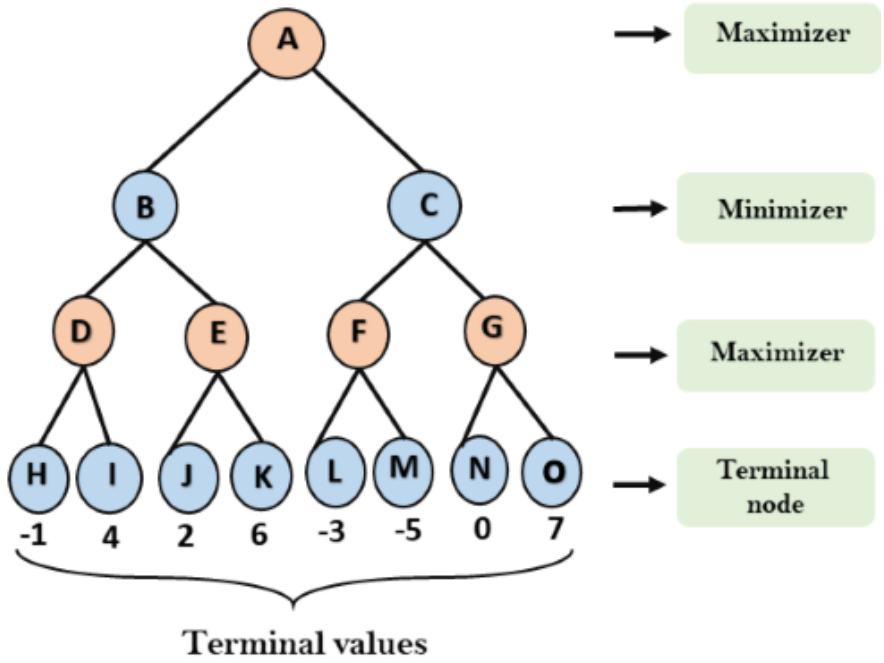


Min–Max Algorithm

- General method for determining optimal move.

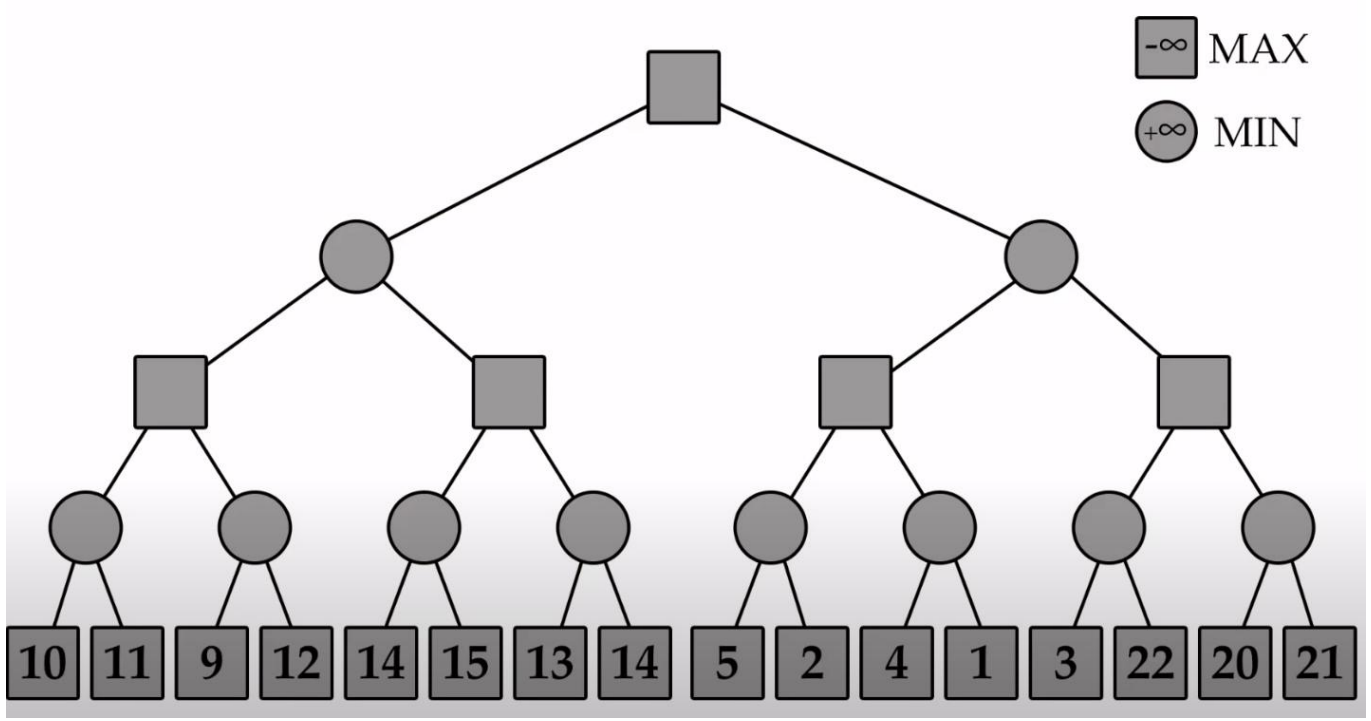
Steps:

- Generate complete game tree down to terminal states.
- Compute utility of each node bottom up from leaves toward root.
- At each MAX node, select the move with maximum utility.
- When reach the root, optimal move is determined



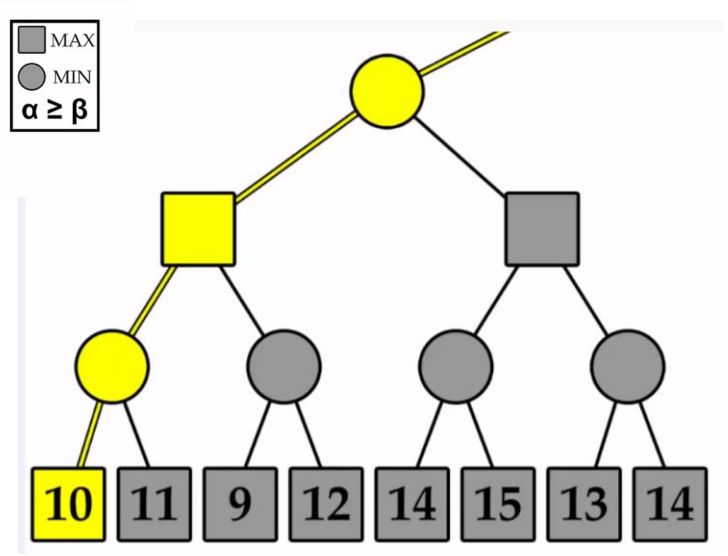
Min-Max Algorithm

- Consider a game tree as follows:
- The maximizer get the chance for first move, i.e. at root, and the minimizer at next level
- Which move would the maximizer move as a maximizing player considering that the opponent also plays optimally?

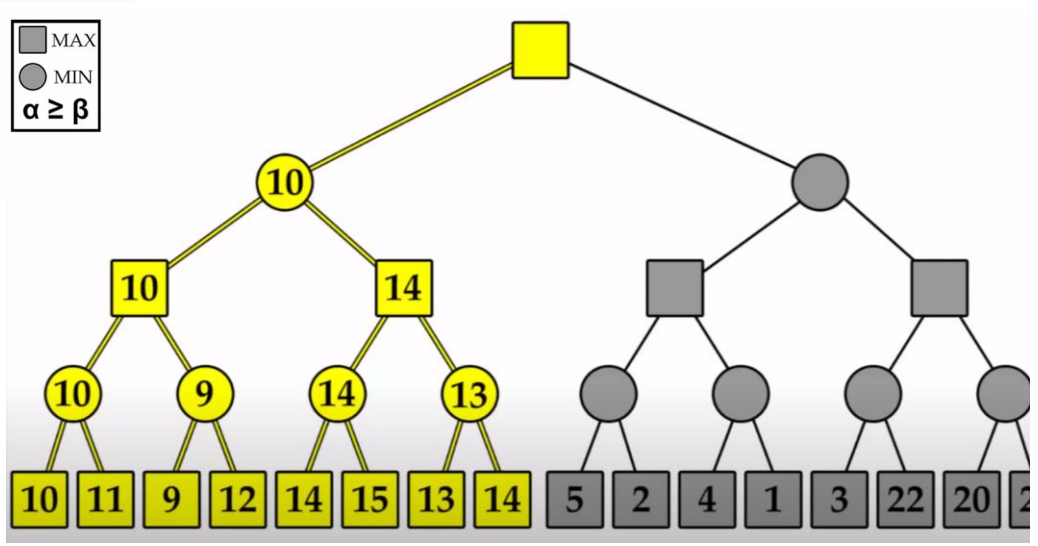


Min-Max Algorithm

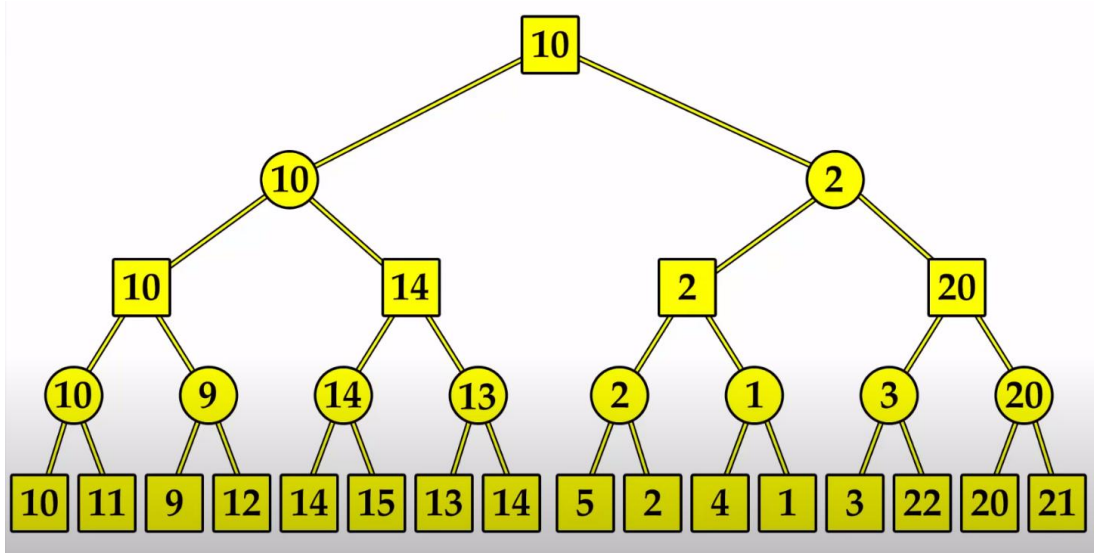
1



2



3

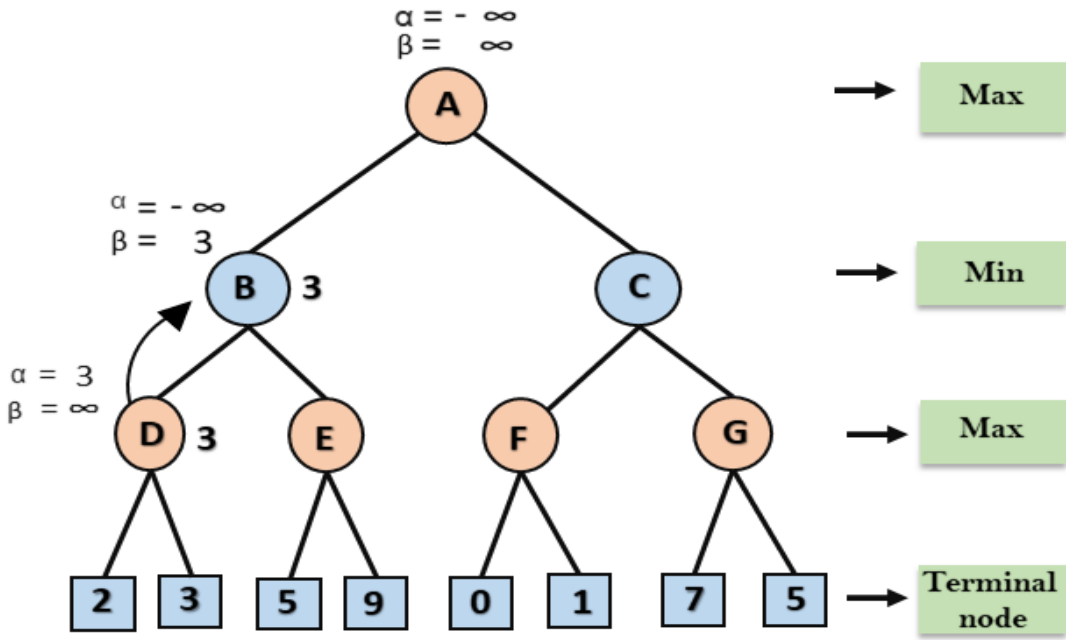


Alpha-Beta Pruning

- It is way of finding the optimal minimax solution by reducing the number of nodes that need to be evaluated by only considering nodes that may be reached in game play.
- The alpha-beta pruning gets its name from two parameters:

  (Alpha)α= Best already explored option along path to the root for maximizer

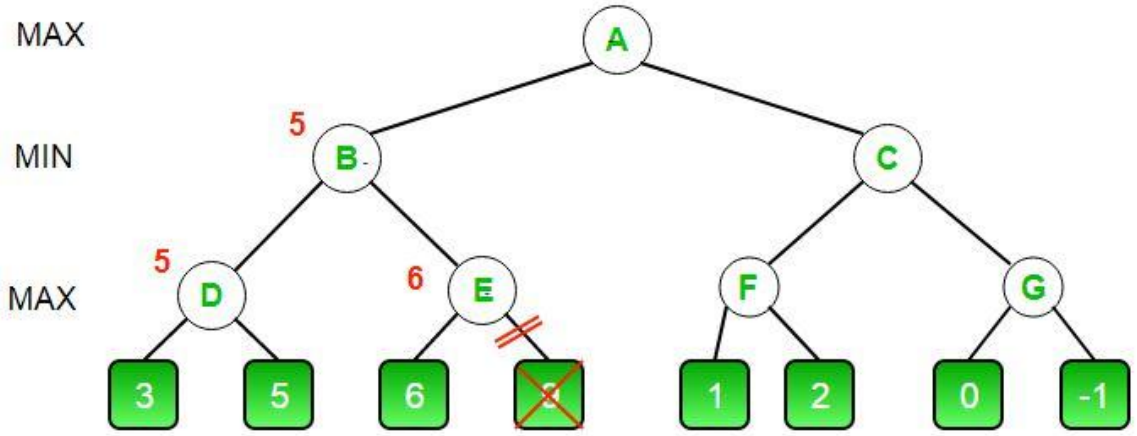
  (Beta)β=best already explored option along path to the root for minimizer
- No need to explore options that are already definitely worse than the current best option.



Alpha-Beta Pruning

General method for determining optimal move.

- Define the alpha and beta value
- Compute utility of each node bottom up from leaves toward root using alpha, beta values.
- At each MAX node, select the move with maximum utility and reduce the number of nodes which are not required
- When reach the root, optimal move is determined



Alpha-Beta Pruning

Rules:

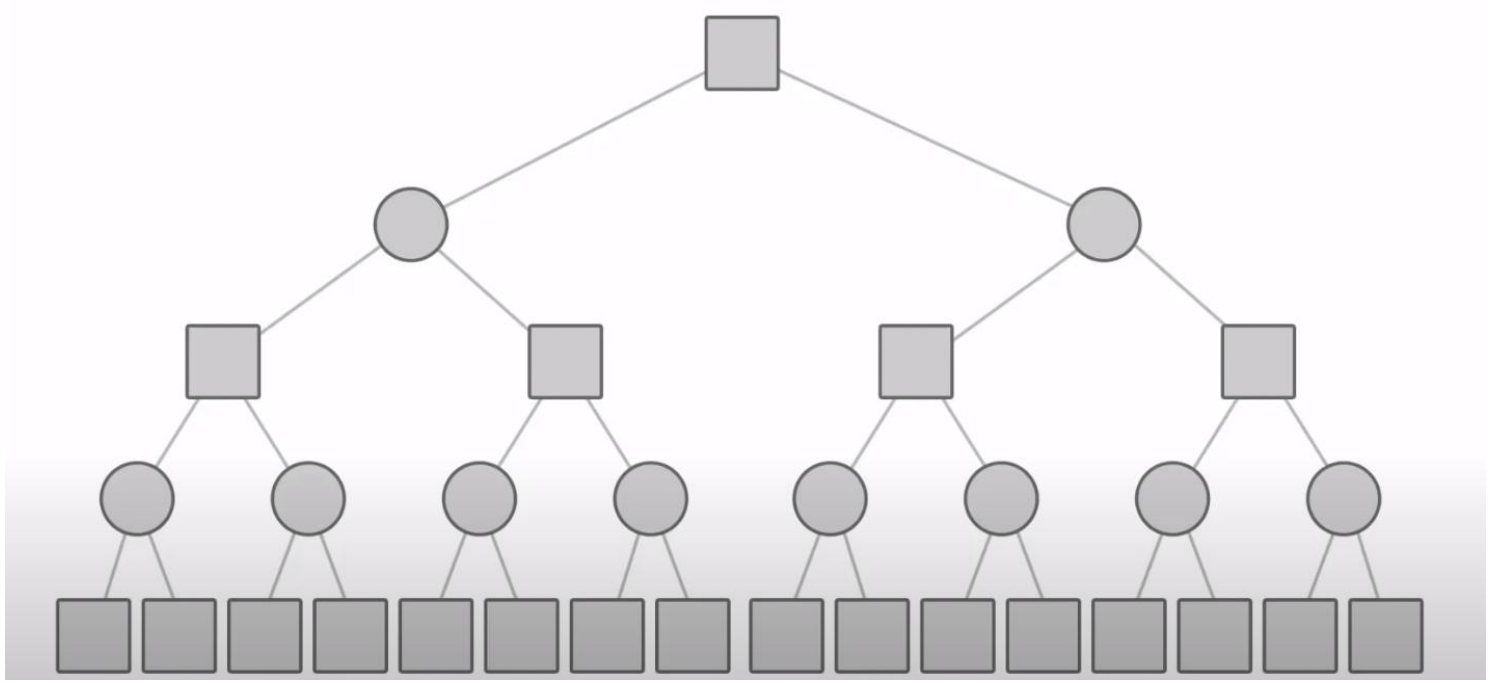
When alpha is greater than beta = pruning

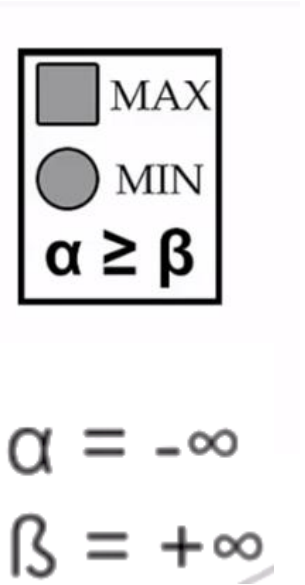
When the value travel downstream, the value of alpha and beta remains the same

When the value travel upward the value is forwarded according to the node i.e. whether the node is MAX or MIN

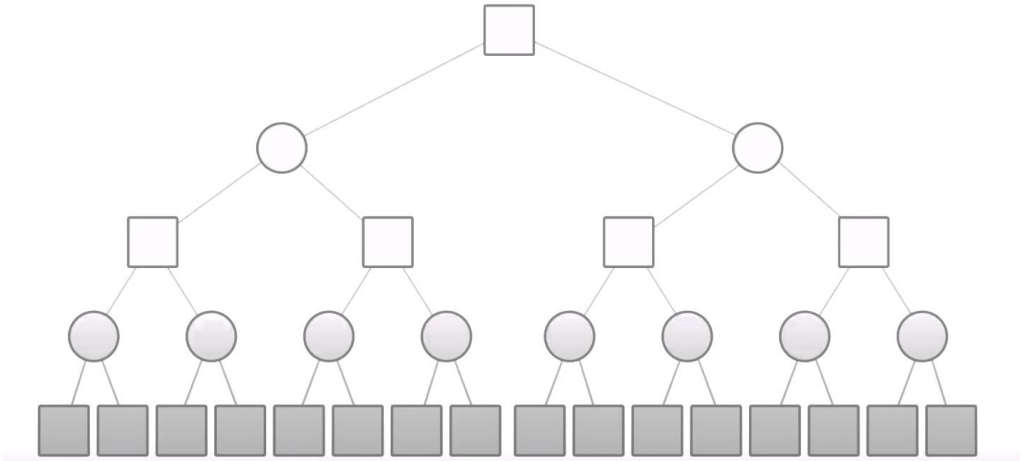
If it is a MAX node= Alpha value updated

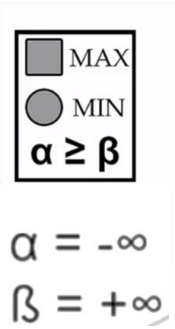
If MIN node= Beta value updated



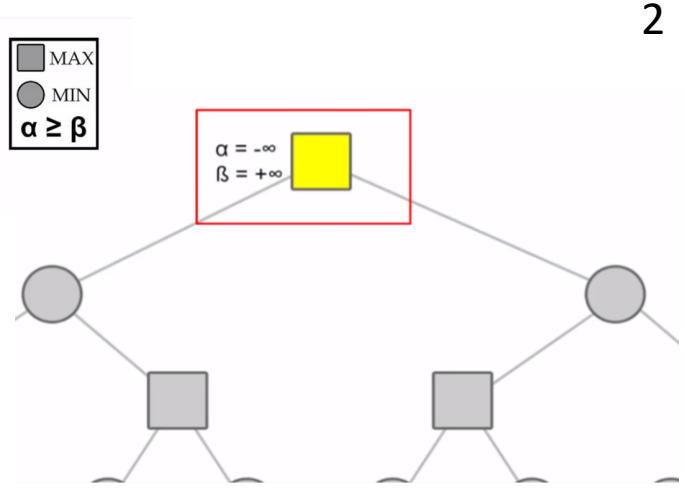


Alpha-Beta Pruning

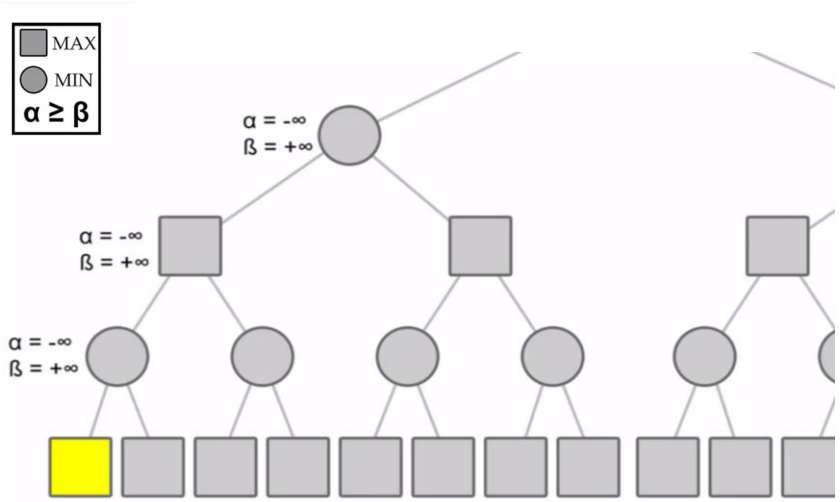




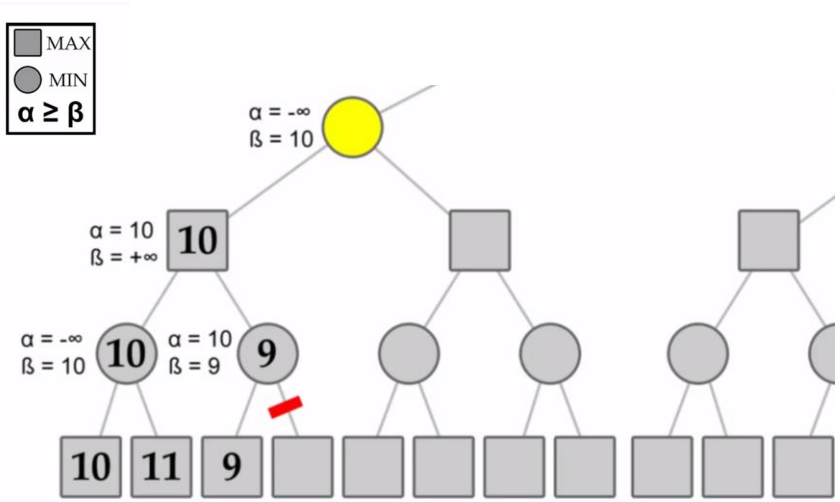
1



2

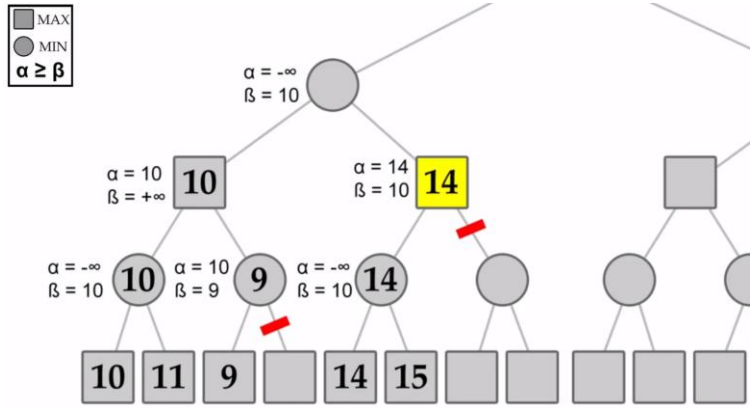


3

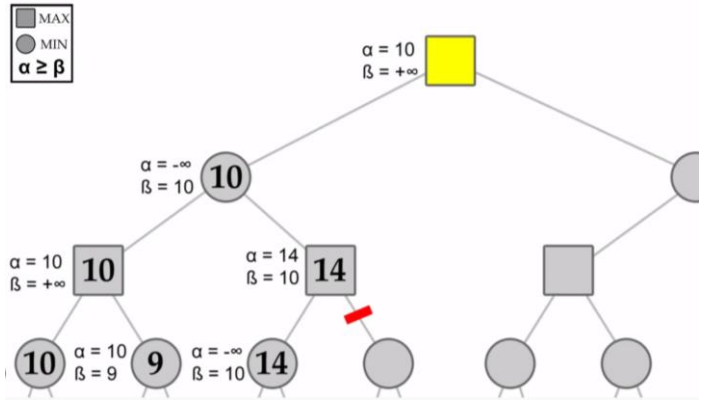


Alpha-Beta Pruning

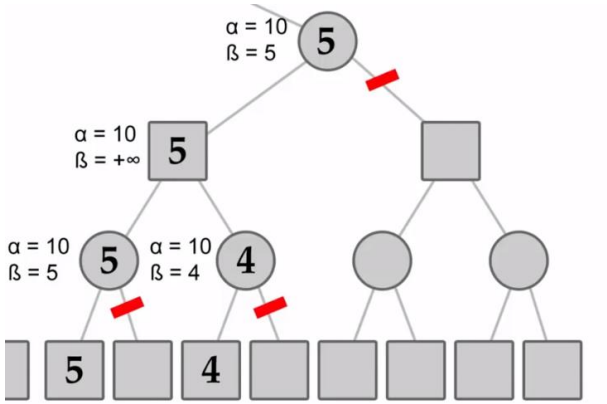
5

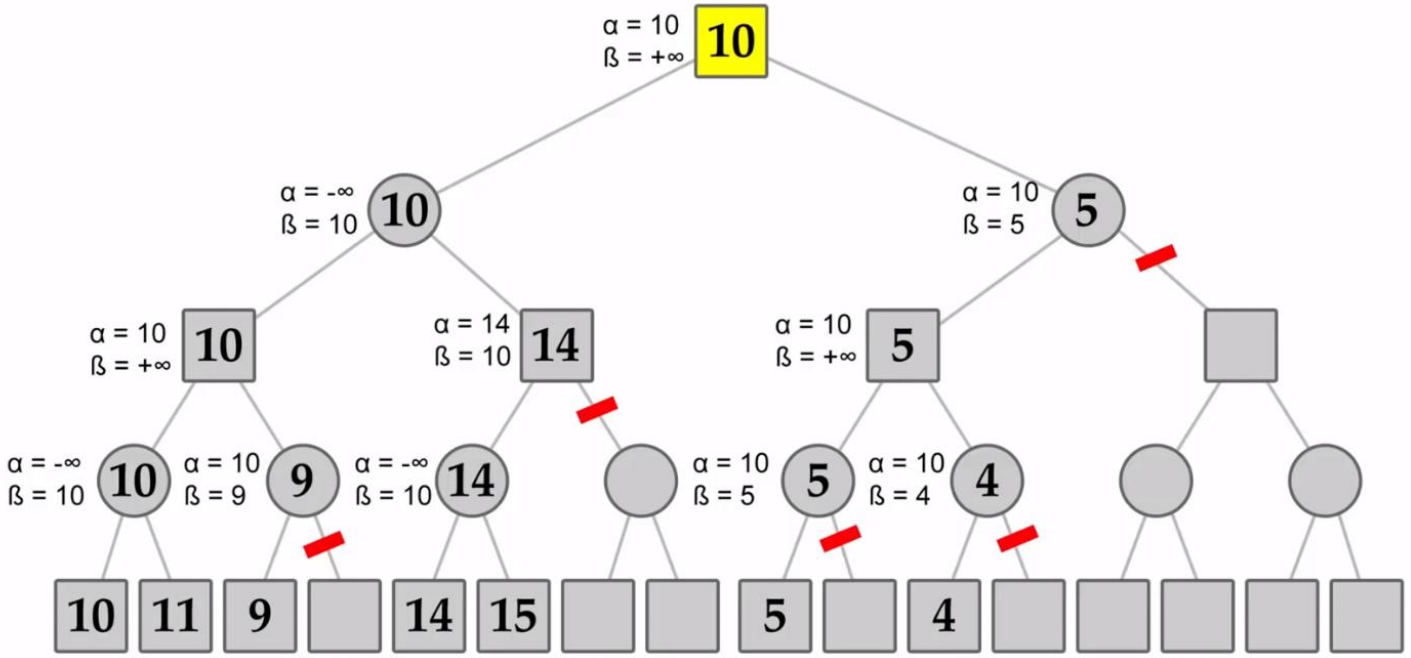


6



7

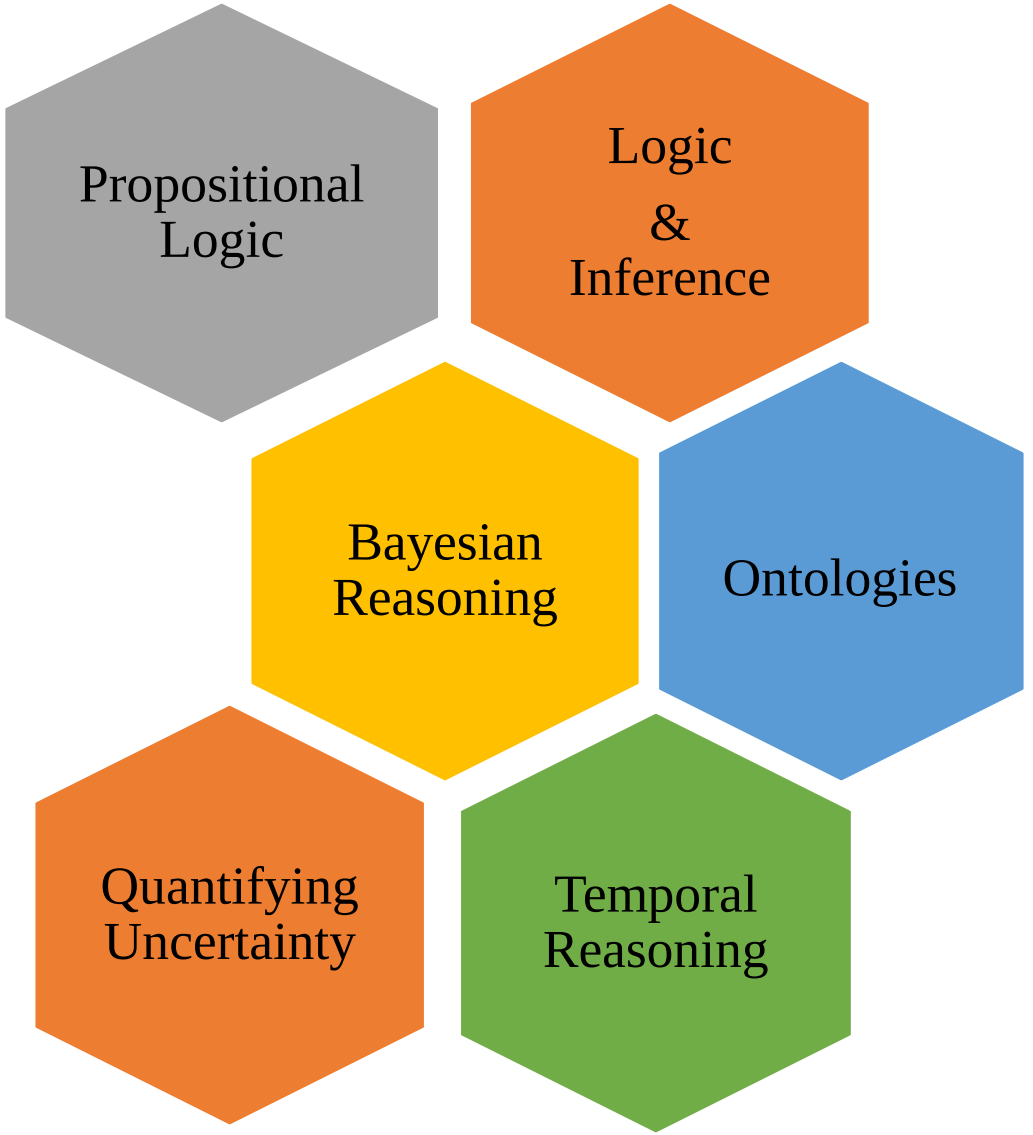




Move Ordering in Pruning

- The effectiveness of alpha – beta pruning is based on the order in which node is examined. Move ordering plays an important role in alpha beta pruning.
- There are two types of move ordering in Alpha beta pruning:
- **Worst Ordering**: In some cases of alpha beta pruning none of the node pruned by the algorithm and works like standard minimax algorithm. This consumes a lot of time as because of alpha and beta factors and also not gives any effective results
- **Ideal Ordering**: In some cases of alpha beta pruning lot of the nodes pruned by the algorithm. In this case, the best move occurs on the left side of the tree. Its first search of the tree go deep twice as minimax algorithm in the same amount of time

Knowledge Representation and Reasoning



Knowledge Representation

Knowledge is the understanding of a subject area

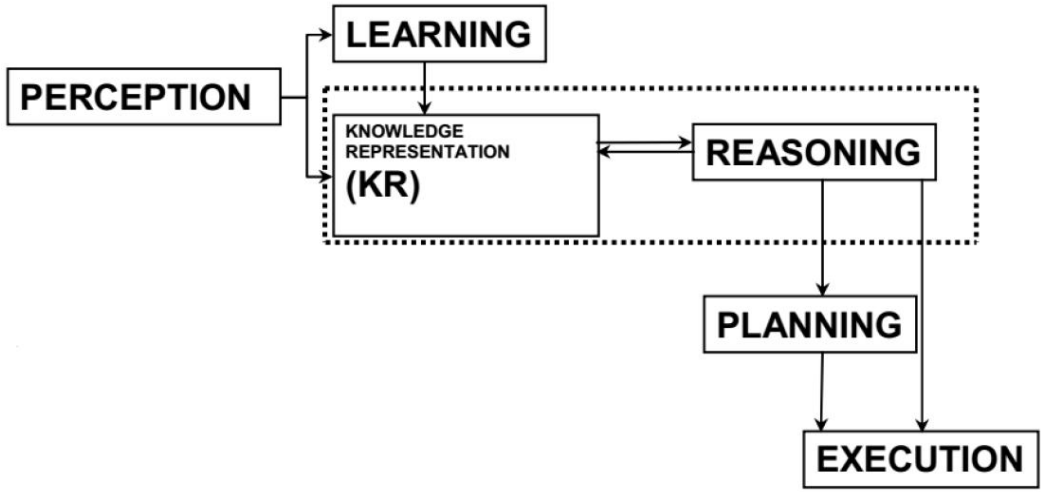
knowledge domain: a well-focused subject area like medical domain, engineering domain, business domain, etc.

Intuitive ways of Knowledge representation:

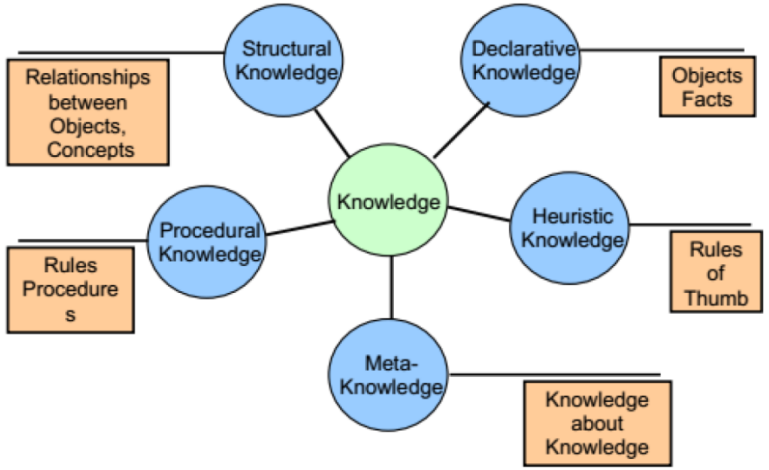
- *Pictures:* high level view of the concept behind
- *Graphs & networks*: relationships among objects
- *Numbers*: knowledge representation by humans

Formal ways of knowledge representation:

- *Facts:* In form of propositions
- *Rules:* Relates known information to others
- *Semantic networks:* Graphs showing relationships
- *Frames:* data structures for stereotypical knowledge
- *Logic:* Formal logics to deal with propositions



The AI Cycle



Categories of Knowledge

Properties of Knowledge Representation

- *Representational adequacy:* Ability to represent the required knowledge
- *Inferential adequacy:* Ability to manipulate the represented knowledge to produce new knowledge
- *Inferential efficiency:* Ability to direct the inferential mechanism into the most productive directions
- *Acquisitional efficiency:* Ability to acquire new knowledge using automatic methods whenever possible

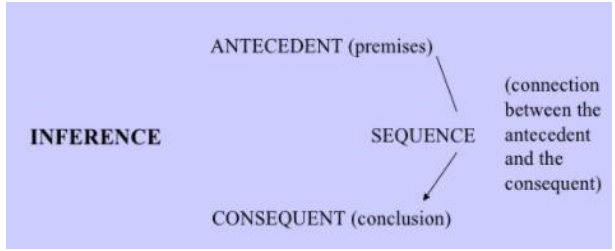
Approaches to Knowledge Representation

- *Simple relational knowledge:* Provides weak inferential capabilities but can be input to large inferential engines
- *Inheritable knowledge:* Objects are organized into classes and classes are organized in a generalization hierarchy. Inheritance is a powerful form of inference, but not adequate
- *Inferential knowledge:* Facts represented in a logical form, which facilitates reasoning. An inference engine is required.
- *Procedural knowledge:* Representation of "how to make it" rather than "what it is". May have inferential efficiency, but no inferential adequacy and acquisitional efficiency.

Logic and Inference

- Logic is used to express knowledge in a particular mathematical notation
  All birds have wings --> ¥*x. Bird(x) -> HasWings(x)*

- Inference is applied to a series of propositions arranged in such a way that the ***consequent*** flows with logical necessity of one or more ***antecedent*** where,
  - ❑ *Antecedent (antecedo):* which goes before
  - ❑ *Consequent (consequeor):* which follows after or inferred by antecedent
- Antecedent and consequent are related in such a way that, truth of antecedent entails truth of consequent and falsity of consequent entails falsity of antecedent



Immediate Inference

Needs no intermediator or advanced knowledge

Cats are animals
Therefore, some animals are cats

Mediate Inference

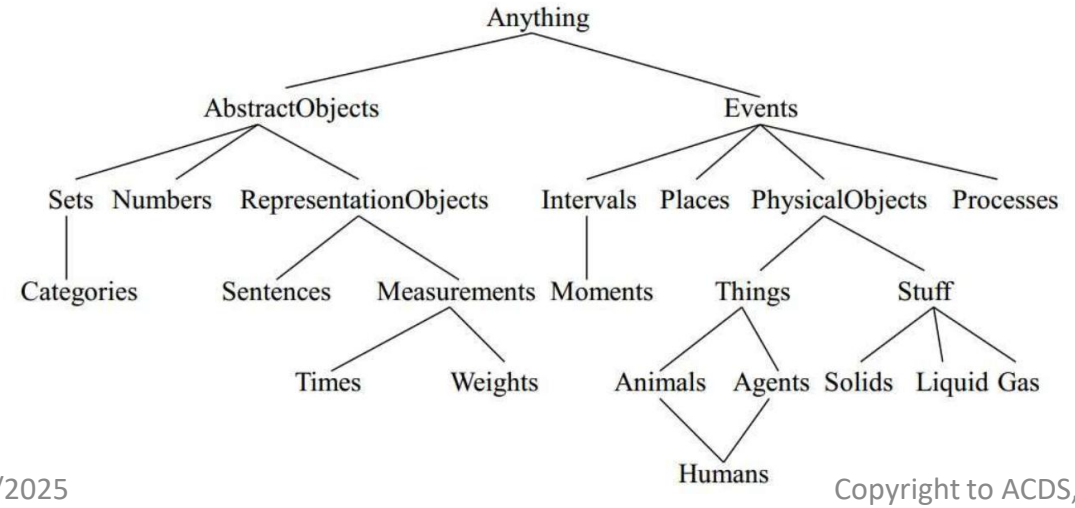
Involves intermediator or needs advanced knowledge

Every animal is mortal
But, every cat is animal
Therefore, cat is mortal

Ontologies

- Ontologies are constructed using knowledge representation languages and logics
- An ontology consists of a set of concepts, axioms, and relationships that describe a domain of interest
  - Create more general and flexible representations.
  - Concepts like actions, time, physical object and beliefs
  - Define general framework of concepts
  - Upper ontology
  - Limitations of logic representation
    - Red, green and yellow tomatoes: exceptions and uncertainty

- Representing a general-purpose ontology is a difficult task called ontology engineering which has only been partially successful on few large AI systems



Each link indicates that the lower concept is a specialization of the upper one.

Specializations are not necessarily disjoint; a human is both an animal and an agent

Propositional Logic

- A proposition is the statement of a fact.
- We usually assign a symbolic variable to represent a proposition, e.g.
  G = "The apartment has a large carpet area"
  D = "The apartment has an area of 1250 sq. ft."
- A proposition is a sentence whose truth values may be determined.
- So, each proposition has a truth value, e.g.
  - The proposition 'A rectangle has four sides' is true
  - The proposition 'The world is a cube' is false.

Logical Connectives

$\wedge$	AND (Conjunction)
$\vee$	OR (Disjunction)
$\neg$	NOT (Negation)
$\rightarrow$	If … then (Conditional)
$\Leftrightarrow$	If and only if (bi-conditional)

p	q	$p \wedge q$	$p \vee q$	$p \Rightarrow q$	$p \Leftrightarrow q$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	F	T	T	F
F	F	F	F	T	T

Limitations of Propositional Logic:

- Propositions can only represent knowledge as complete sentences, e.g. a = the ball's color is blue.
- Cannot analyze the internal structure of the sentence.
- No quantifiers are available, e.g. for-all, there-exists

Propositional logic provides no framework for proving statements such as:

- ❑ All humans are mortal
- ❑ All women are humans
- ❑ Therefore, all women are mortals
- ❑ This is a limitation in its representational power.

Laws of propositional logic

- **Idempotent:**

  $p \vee p \equiv p$ $\quad$ $p \wedge p \equiv p$

- **Commutative:**

  $p \vee q \equiv q \vee p$ $\quad$ $p \wedge q \equiv q \wedge p$

- **Complement:**

  $p \vee \sim p \equiv T$ $\quad$ $p \wedge \sim p \equiv F$

- **Double Negation:**

  $\sim(\sim p) \equiv p$

- **Associative:**

  $p \vee (q \vee r) \equiv (p \vee q) \vee r$

  $p \wedge (q \wedge r) \equiv (p \wedge q) \wedge r$

- **Distributive:**

  $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$

  $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$

- **Absorbtion:**

  $p \vee (p \wedge q) \equiv p$

  $p \wedge (p \vee q) \equiv p$

- **Identity:**

  $p \vee T \equiv T$ $\quad$ $p \wedge T \equiv p$

  $p \vee F \equiv p$ $\quad$ $p \wedge F \equiv F$

- **De Morgan's**

  $\sim(p \vee q) \equiv \sim p \wedge \sim q$

  $\sim(p \wedge q) \equiv \sim p \vee \sim q$

- **Equivalence of Contrapositive:**

  $p \rightarrow q \equiv \sim q \rightarrow \sim p$

- **Others:**

  $p \rightarrow q \equiv \sim p \vee q$

  $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

Propositional Logic

Table of Logical equivalences

- Identity laws
  Like adding 0
- Domination laws
  Like multiplying by 0
- Idempotent laws
  Delete redundancies
- Double negation
  "I don't like you, not"
- Commutativity
  Like "$x+y = y+x$"
- Associativity
  Like "$(x+y)+z = y+(x+z)$"
- Distributivity
  Like "$(x+y)z = xz+yz$"
- De Morgan

TABLE 5 Logical Equivalences.

Equivalence	*Name*
$p \wedge \mathbf{T} \iff p$ $p \vee \mathbf{F} \iff p$	Identity laws
$p \vee \mathbf{T} \iff \mathbf{T}$ $p \wedge \mathbf{F} \iff \mathbf{F}$	Domination laws
$p \vee p \iff p$ $p \wedge p \iff p$	Idempotent laws
$\neg(\neg p) \iff p$	Double negation law
$p \vee q \iff q \vee p$ $p \wedge q \iff q \wedge p$	Commutative laws
$(p \vee q) \vee r \iff p \vee (q \vee r)$ $(p \wedge q) \wedge r \iff p \wedge (q \wedge r)$	Associative laws
$p \vee (q \wedge r) \iff (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \iff (p \wedge q) \vee (p \wedge r)$	Distributive laws
$\neg(p \wedge q) \iff \neg p \vee \neg q$ $\neg(p \vee q) \iff \neg p \wedge \neg q$	De Morgan's laws

Source: Rosen, K.H. and Krithivasan, K., 2012. *Discrete mathematics and its applications: with combinatorics and graph theory*. Tata McGraw-Hill Education.

First Order Logic

First order logic (FOL), the atomic formulas are interpreted as statements about relationship between objects
In FOL, atomic statements use predicates with constants as arguments, eg.

- Male (John)
- Female (Mary)
- Sibling (Peter, Harry)

First-order logic (FOL) models the world in terms of

- *Objects*, which are things with individual identities
- *Properties* of objects that distinguish them from other objects
- *Relations* that hold among sets of objects
- *Functions,* which are a subset of relations where there is only one "value" for any given "input"

Examples:

- *Objects: Students, lectures, companies, cars ...*
- *Relations: Brother-of, bigger-than, outside, part-of, has-color, occurs-after, owns, visits, precedes, ...*
- *Properties: blue, oval, even, large, ...*
- *Functions: father-of, best-friend, second-half, one-more-than ...*

Variables

- **Variable symbols**
  - E.g., x, y, foo
- **Connectives**
  - Same as in PL: not (¬), and (∧), or (∨), implies (→), if and only if (biconditional ↔)
- **Quantifiers**
  - Universal ∀**x** or **(Ax)**
  - Existential ∃**x** or **(Ex)**

- **Universal quantification**
  - (∀x)P(x) means that P holds for **all** values of x in the domain associated with that variable
  - E.g., (∀x) dolphin(x) → mammal(x)
- **Existential quantification**
  - (∃ x)P(x) means that P holds for **some** value of x in the domain associated with that variable
  - E.g., (∃ x) mammal(x) ∧ lays-eggs(x)
  - Permits one to make a statement about some object without naming it

Connections between "universal" & "essential" quantifiers

De Morgans law can be used to connect both the quantifiers,

$$(\forall x)\ \neg P(x) \leftrightarrow \neg(\exists x)\ P(x)$$

$$\neg(\forall x)\ P \leftrightarrow (\exists x)\ \neg P(x)$$

$$(\forall x)\ P(x) \leftrightarrow \neg\ (\exists x)\ \neg P(x)$$

$$(\exists x)\ P(x) \leftrightarrow \neg(\forall x)\ \neg P(x)$$

Translating English into FOL

Every gardener likes the sun.

$\forall x\ \text{gardener}(x) \rightarrow \text{likes}(x,\text{Sun})$

You can fool some of the people all of the time.

$\exists x\ \forall t\ \ \text{person}(x) \wedge \text{time}(t) \rightarrow \text{can-fool}(x,t)$

You can fool all of the people some of the time.

$\forall x\ \exists t\ (\text{person}(x) \rightarrow \text{time}(t) \wedge \text{can-fool}(x,t))$

$\forall x\ (\text{person}(x) \rightarrow \exists t\ (\text{time}(t) \wedge \text{can-fool}(x,t))$

Equivalent

All purple mushrooms are poisonous.

$\forall x\ (\text{mushroom}(x) \wedge \text{purple}(x)) \rightarrow \text{poisonous}(x)$

No purple mushroom is poisonous.

$\neg\exists x\ \text{purple}(x) \wedge \text{mushroom}(x) \wedge \text{poisonous}(x)$

$\forall x\ \ (\text{mushroom}(x) \wedge \text{purple}(x)) \rightarrow \neg\text{poisonous}(x)$

Equivalent

Clinton is not tall.

$\neg\text{tall}(\text{Clinton})$

Probabilistic Reasoning

- Using logic and probability to handle uncertain situation
- Probability based reasoning is same as understanding directly from the knowledge that a given probability rating based on uncertainty present

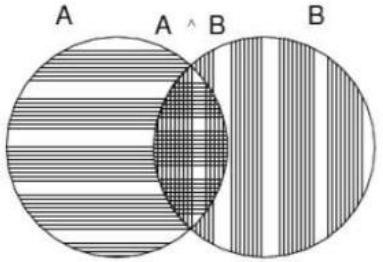
Why use probability?

Definition of probability implies that, some logically related events must have related probability, for eg.

$$p(a \vee b) = p(a) + p(b) - p(a \wedge b)$$

Bayes Theorem

Bayes theorem says how to do inference about hypotheses (uncertain) from data (measured)



$$P(hypothesis \mid data) = \frac{P(hypothesis)P(hypothesis \mid data)}{\sum_h P(h)P(data \mid h)}$$

Why Probabilistic reasoning essential?

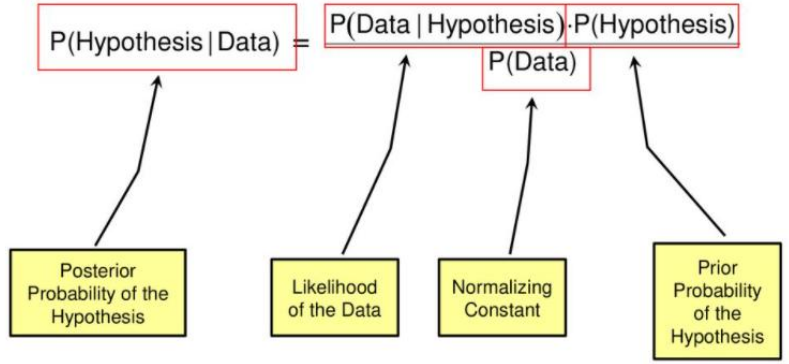
Many aspects of learning and intelligence depend crucially on probabilistic representation of uncertainty like forecasting, decision making, data compression, automatic modelling etc.

Uncertainty in probabilistic reasoning arises due to inability to predict outcomes due to unreliable , incomplete or in consistence knowledge

Bayesian Reasoning

- Bayesian reasoning provides a probabilistic approach to interference
- Based on two assumptions
  - Quantities of interest are governed by probability distributions
  - Optimal decisions can be made by reasoning about these probabilities with the observed data
- Bayes Theorem is the corner stone of Bayesian learning methods

$$P(h \mid D) = \frac{P(D|h)P(h)}{P(D)}$$



- Bayes rule for hypothesis (h) given data (d): $P(h \mid D) = \frac{P(D|h)P(h)}{P(D)}$
- Bayes rule for not-hypothesis (h) given data (d): $P(\sim h \mid D) = \frac{P(D|\sim h)P(\sim h)}{P(D)}$
- Odds form of Bayes rule: $\frac{P(h \mid D)}{P(\sim h \mid D)} = \frac{P(D|h)}{P(D|\bar{h})} \cdot \frac{P(h)}{P(\bar{h})}$

- Odds form of Bayes rule:

$$\underbrace{\frac{P(h \mid D)}{P(\sim h \mid D)}}_{\text{Posterior Odds}} = \underbrace{\frac{P(D|h)}{P(D|\bar{h})}}_{\text{Likelihood ratio}} \cdot \underbrace{\frac{P(h)}{P(\bar{h})}}_{\text{Prior Odds}}$$

- h = a hypothesis
- ~h = the negation of the hypothesis
- D = the data

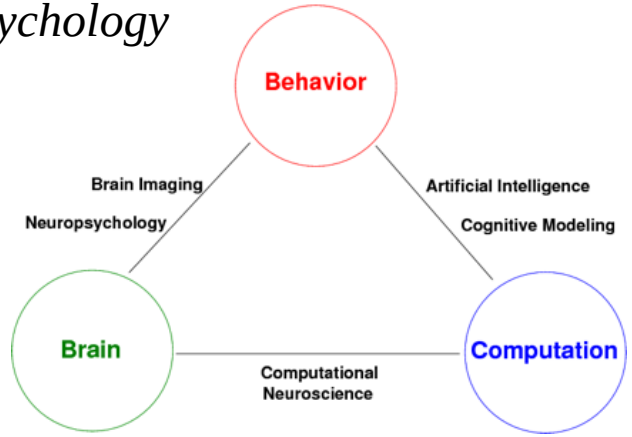
Posterior Odds = Likelihood Ratio x Prior Odds

Violation of Bayes rule in Intuitive thinking

❑ Intuitive violation of Bayes rule show that the rational agent model is not descriptively correct, it can cause serious errors in real-world decision making

Temporal Reasoning

Temporal reasoning is related to two large areas – *cognitive science* and *cognitive psychology*



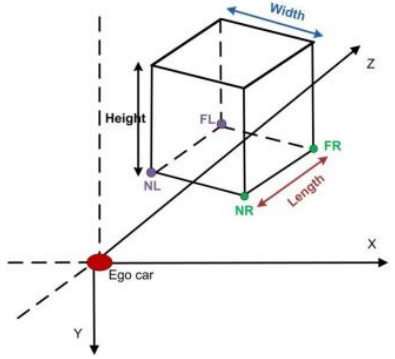
Two prime applications of temporal reasoning:

- *object detection*
- *artificial intelligent characters*

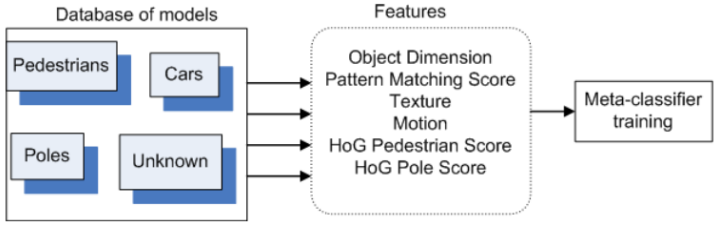
Object detection

Mainly involves three steps – object detection, object recognition and modelling

Detecting the object: Describing object as a cuboid with x, z be the base corner and y top bottom bases for mass, centre and height



Recognizing the object: Machine learning for recognition with labelled datasets for training.



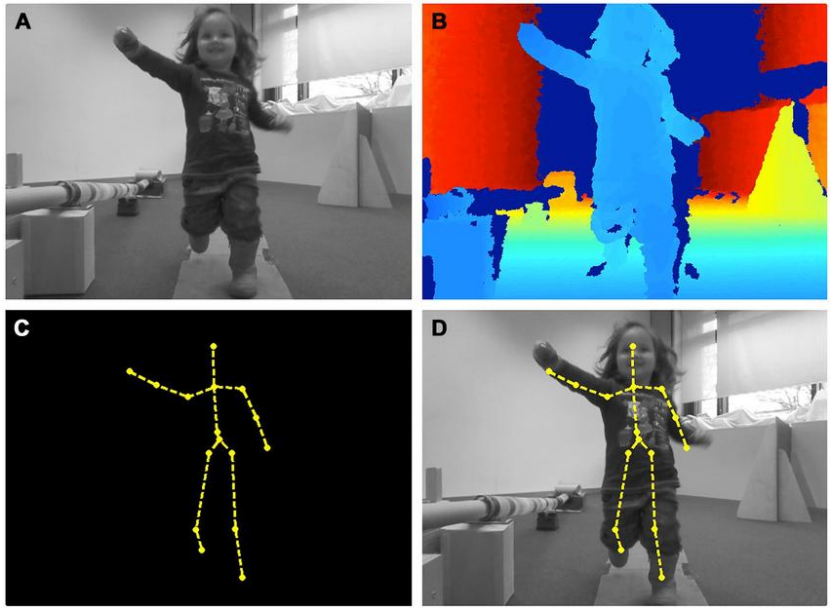
Feature extraction for Traffic scene detection

Ontological modelling: For each object, the object recognition module will provide list of objects as described by the attributes

Temporal Reasoning

Artificial Intelligent Characters

Ada and Grace in Virtual Museum



Kinect Files

MOCAP files

Quantifying Uncertainty

Let the action A_t = leave for office at t minutes before opening time.
With A_t, can one get office on time? **Uncertain!!**

Possible problems:

- ❑ Partial observatory (road state)
- ❑ Noisy sensors (traffic reports forecast)
- ❑ Uncertainty in outcomes (flat tire)
- ❑ Immense complexity of modelling and prediction

Handling uncertainty:

- ❑ Default logic (prior assumptions)
- ❑ Rules of fudge factors
- ❑ **Probability**
- ❑ Fuzzy logic

Sources of uncertainty

- Human Error
- Experimental Error
- Random Error
- Equipment Error
- Environment Variation

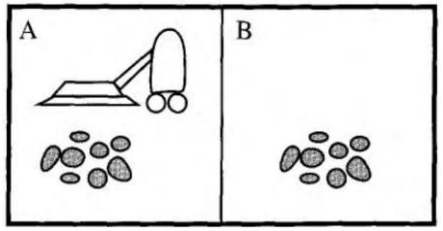
- ❖ Probabilistic reasoning are used when output or outcomes are unpredictable
- ❖ One way to express confidence about such event is probability
- ❖ Probability of an uncertain event is measure of degree of likelihood of occurrence of that event

	Ontological commitments	Epistemological commitments
Logic	world is composed of facts that do or do not hold	each sentence is true or false or unknown
Probability Theory	world is composed of facts that do or do not hold	numerical degree of belief between 0(sentence for certainly false) and 1(sentences are certainly true)

THANK YOU

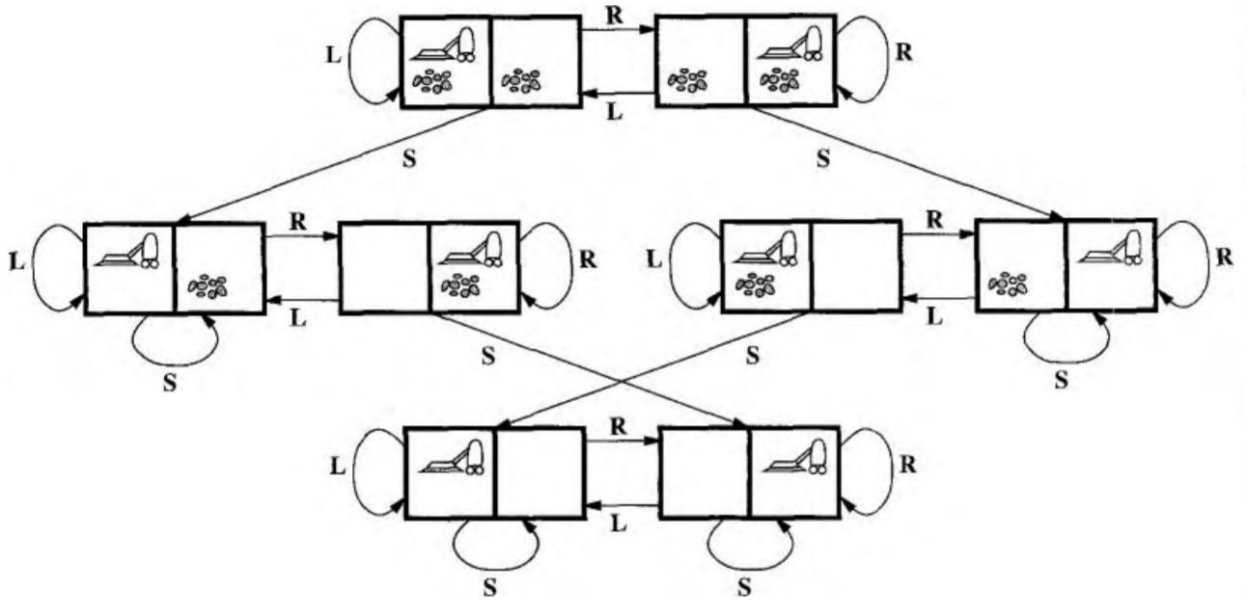
Worked examples & exercises will be discussed at 10:30 AM

Question 1: Consider a vacuum-cleaner world with two locations problem.



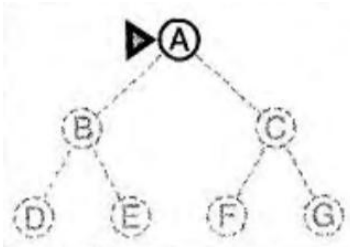
Formulate the state space of the problem?

Answer:



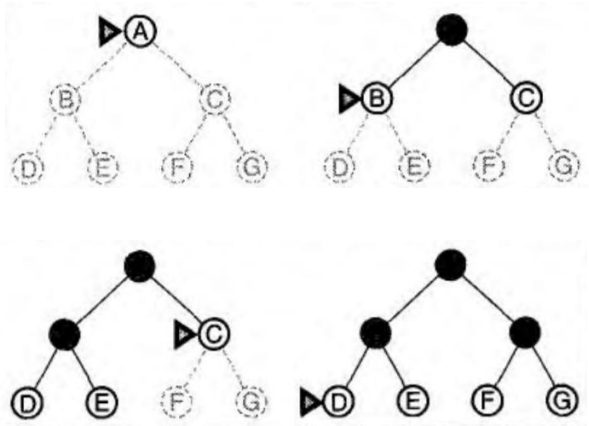
Note: Follow Russell & Norvig book [Chapter 3]

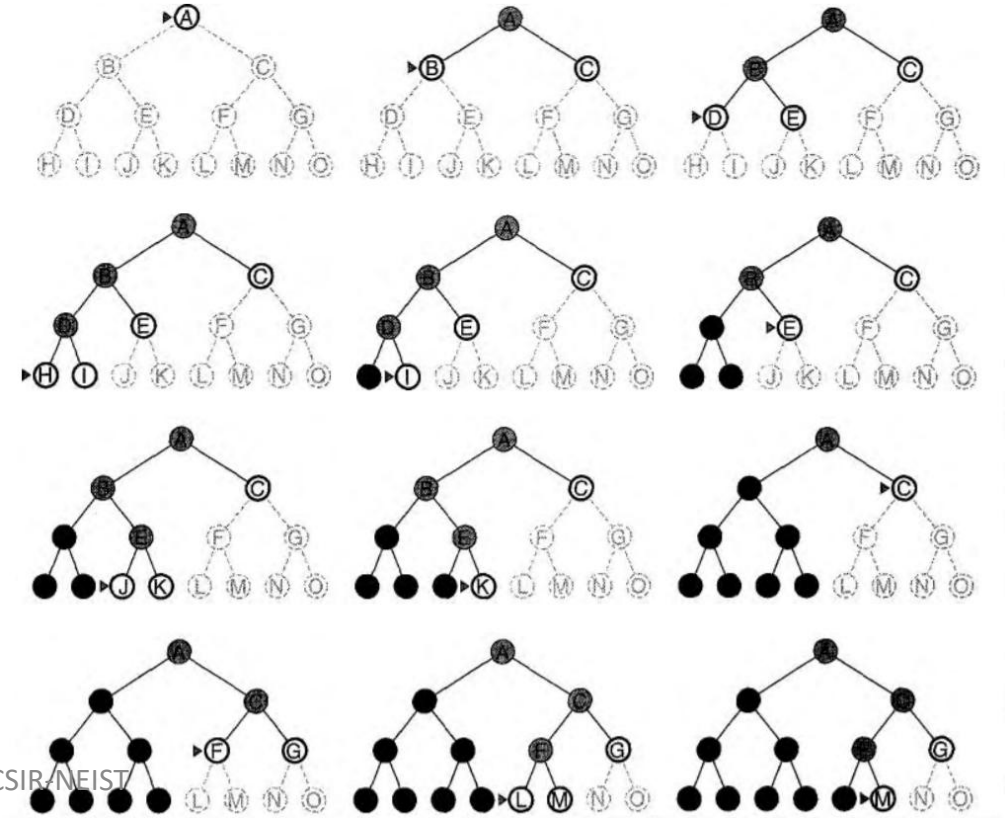
Question 2: Consider the following path search problem.



Perform uninformed (blind) search - Breadth First Search and Depth First Search algorithm to above.

Answer:





Note: Follow Russell & Norvig book [Chapter 3]

Question 3: Explain the concept of offline and online search algorithm?

Answer: Offline search algorithms compute a complete solution before setting foot in the real world and then execute the ONLINE SEARCH solution without recourse to their percepts. In contrast, an online search agent operates by interleaving computation and action: first it takes an action, then it observes the environment and computes the next action.

In general, an offline search would have to come up with an exponentially large contingency plan that considers all possible happenings, while an online search need only consider what actually does happen.

Online search is a good idea in dynamic or semi-dynamic domains-domains where there is a penalty for sitting around and computing too long. Online search is an even better idea for stochastic domains.

Contd…

Russell & Norvig book [chapter 4]

Worked Examples

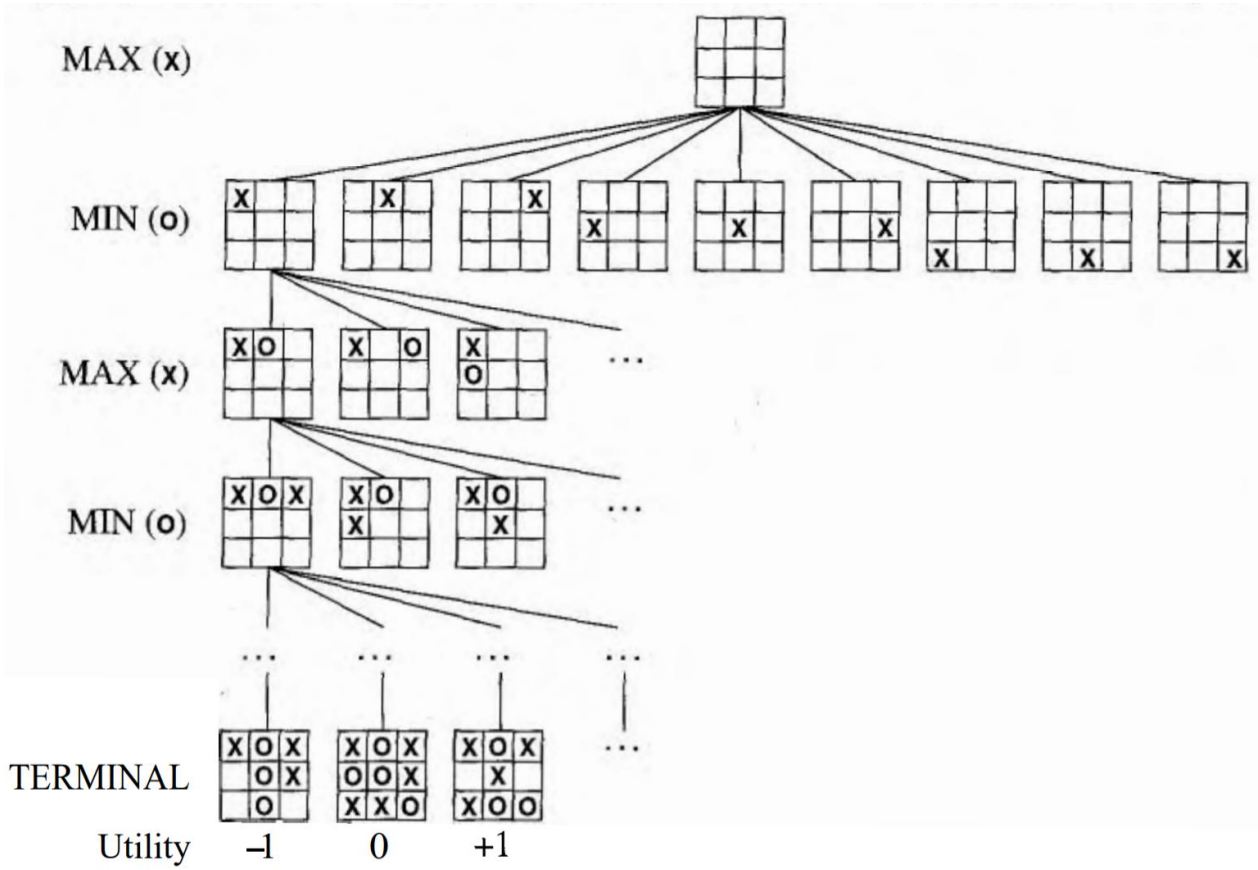
For example, a chess playing agent is well-advised to make its first move long before it has figured out the complete course of the game. Online search is a necessary idea for an exploration problem, where the states and actions are unknown to the agent. An agent in this state of Ignorance must use its actions as experiments to determine what to do next, and hence must interleave computation and action.

The canonical example of online search is a robot that is placed in a new building and must explore it to build a map that it can use for getting from A to B.

Consider a newborn baby: it has many possible actions, but knows the outcomes of none of them, and it has experienced only a few of the possible states that it can reach. The baby's gradual discovery of how the world works is, in part, am online search process.

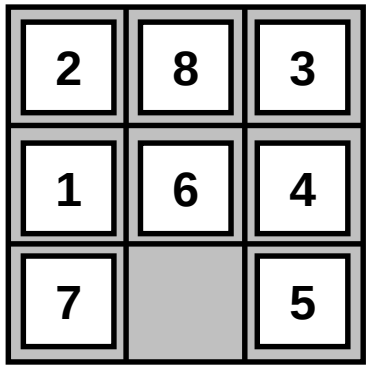
Question 4: Draw the game tree for the game of tic-tac-toe.

Answer:

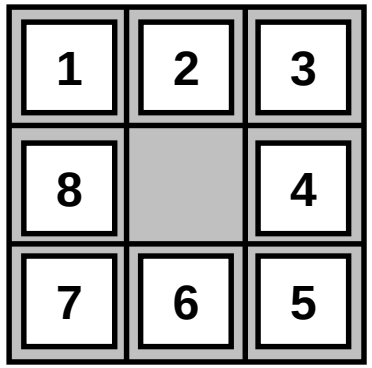


Follow: Russell & Norvig book [Chapter 5]

Question 5: Consider the 8-puzzle problem. Perform breadth-first search algorithm to reach the goal state from the given start state and show the state space.



Start State

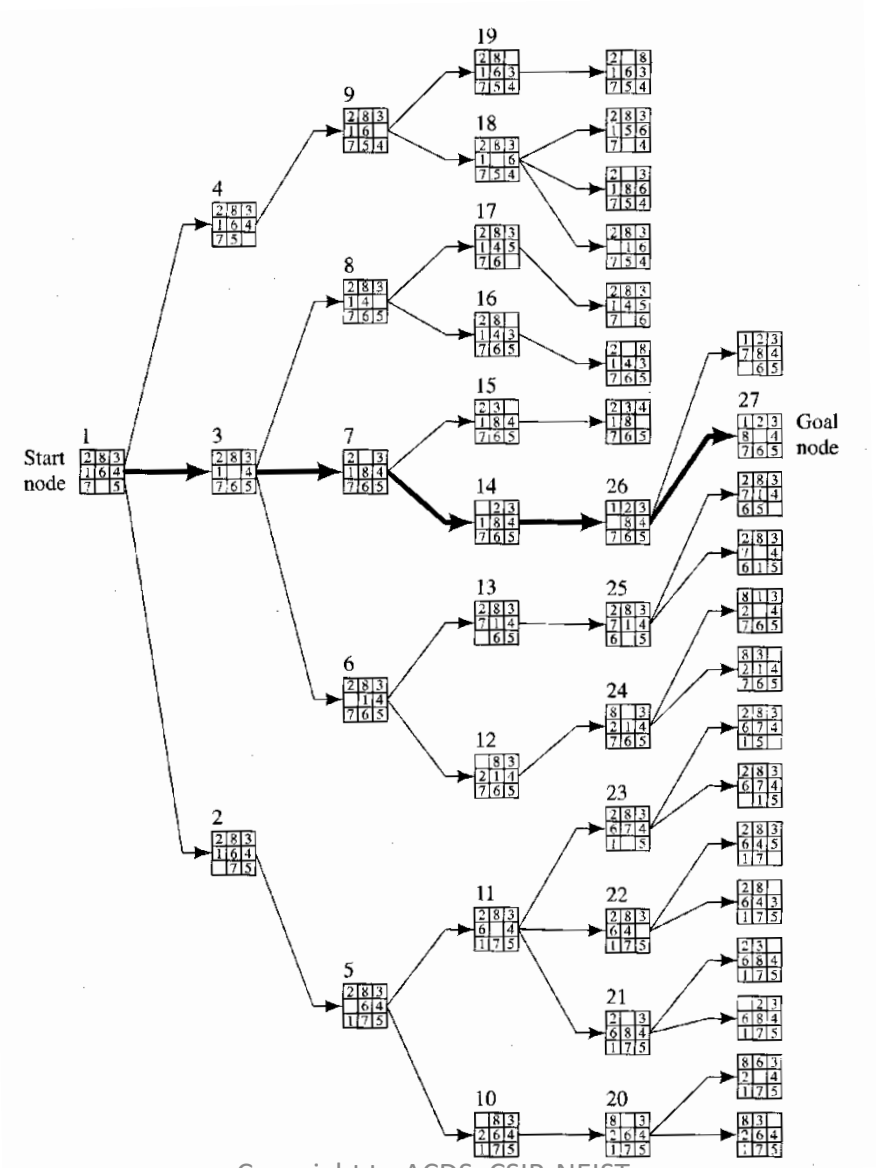


Goal State

Successor function (actions): Blank moves *Left*, *Right*, *Up*, or *Down*.
Path cost: Each move costs 1. The path cost = the number of moves.

Answer:

Answer:

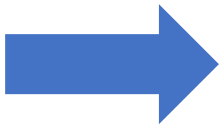


Worked Examples:

Question 6: Given a 3×3 board with 8 tiles (every tile has one number from 1 to 8) and one empty space. The objective is to place the numbers on tiles to match final configuration using the empty space. We can slide four adjacent (left, right, above and below) tiles into the empty space.

1	**2**	**3**
4	**8**	**-**
7	**6**	**5**

Start State



1	**2**	**3**
4	**5**	**6**
7	**8**	**-**

Final State

Solution:

{(1, 2, 3), (4, 8, -), (7, 6, 5)} [Initial State]

Step 1: {(1, 2, 3), (4, 8, 5), (7, 6, -)} [Move Down]

Step 2: {(1, 2, 3), (4, 8, 5), (7, -, 6)} [Move Left]

Step 3: {(1, 2, 3), (4, -, 5), (7, 8, 6)} [Move Up]

Step 4: {(1, 2, 3), (4, 5, -), (7, 8, 6)} [Move Right]

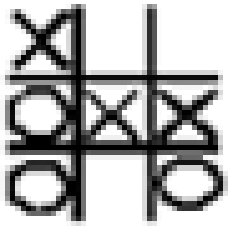
Step 5: {(1, 2, 3), (4, 5, 6), (7, 8, -)} [Move Down]

{(1, 2, 3), (4, 5, 6), (7, 8, -)} [Final State]

Total path = 5 (Cost of solving the puzzle)

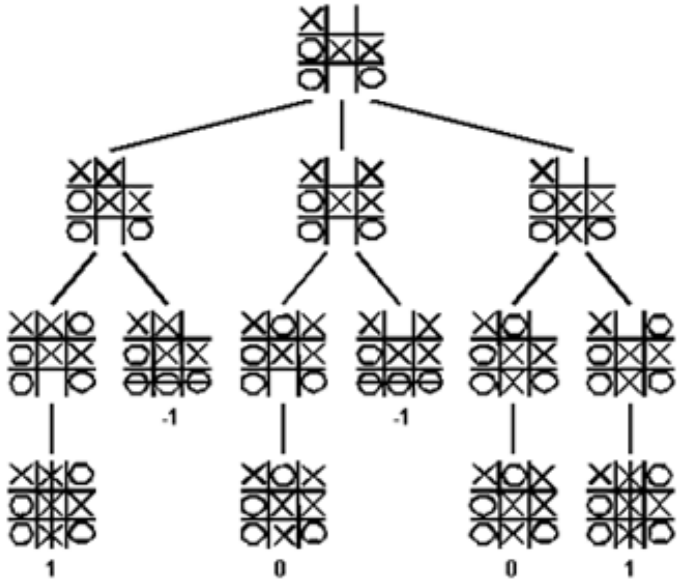
Worked Examples:

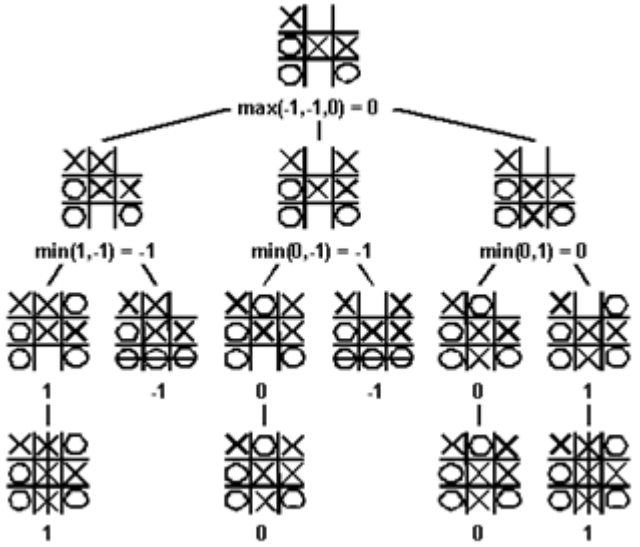
Question 7: Consider the current state given below for the Tic-Tac-Toe game. Design the state space and identify the optimal win route.



Current State

Answer:



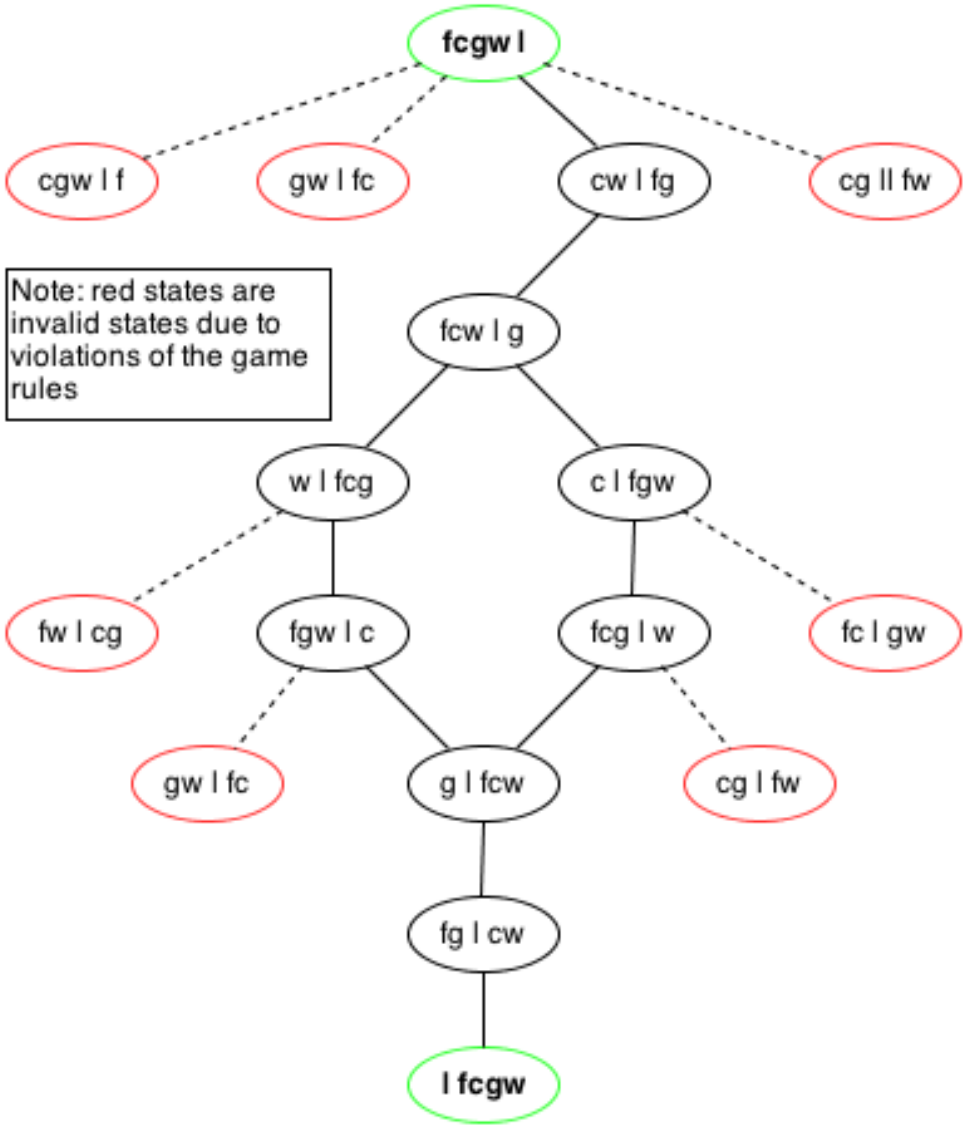


Question 8: Consider the popular **Wolf, a Goat, and a Cabbage** problem which is stated as – "A farmer went to a market and purchased a wolf, a goat, and a cabbage. On his way home, the farmer came to the bank of a river and rented a boat. But crossing the river by boat, the farmer could carry only himself and a single one of his purchases: the wolf, the goat, or the cabbage. If left unattended together, the wolf would eat the goat, or the goat would eat the cabbage. The farmer's challenge was to carry himself and his purchases to the far bank of the river, leaving each purchase intact."

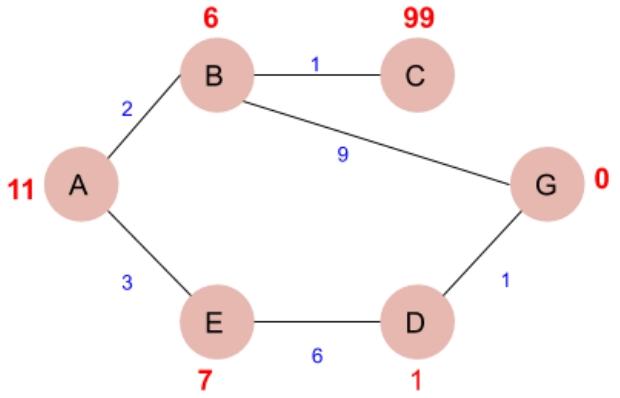
How did he do it? Draw the state space?

Worked Examples

Answer:



Question 9: Perform inform search - A* algorithm on the following graph given actual cost and heuristic cost.



Answer:

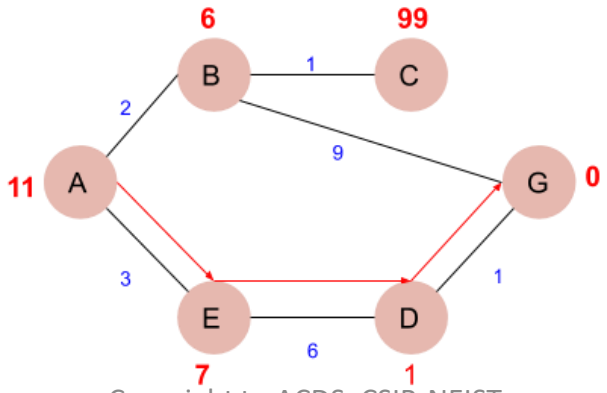
A=11

A → B = 2 + 6 = 8

A → E = 3 + 6 = 10

A → B → C = (2 + 1) + 99= 102

A → B → G = (2 + 9) + 0 = 11

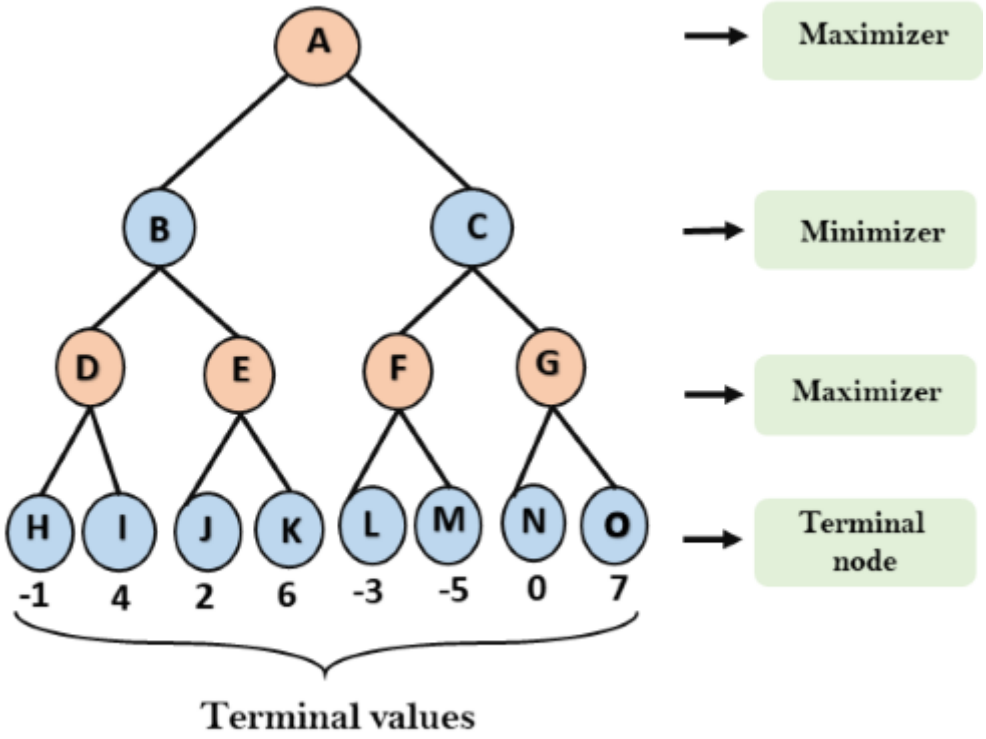


A → E → D = (3 + 6) + 1 = 10

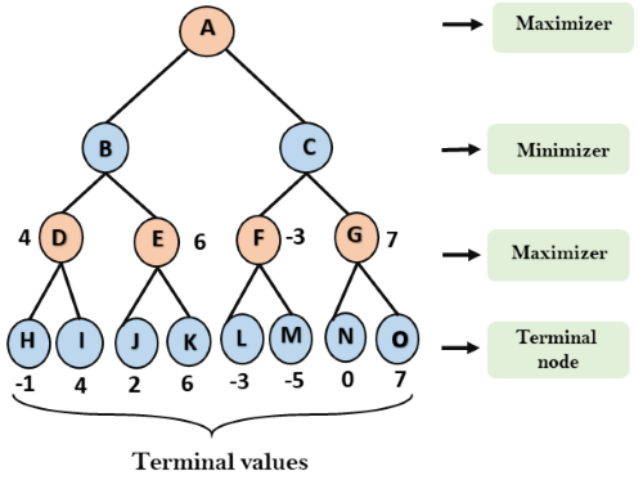
A → E → D → G = (3 + 6 + 1) +0 =**10**

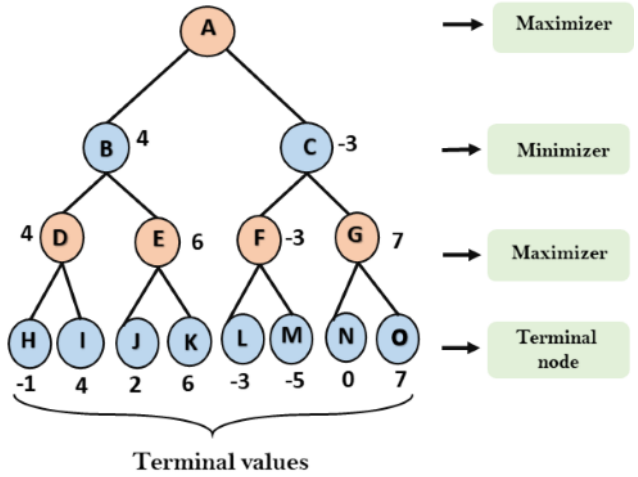
Question 10: Perform min-max algorithm on the tree below.

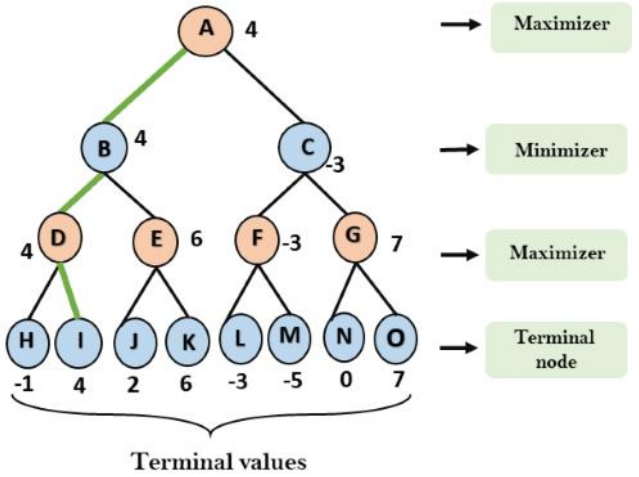
a) Without αβ pruning
b) With αβ pruning



Answer:

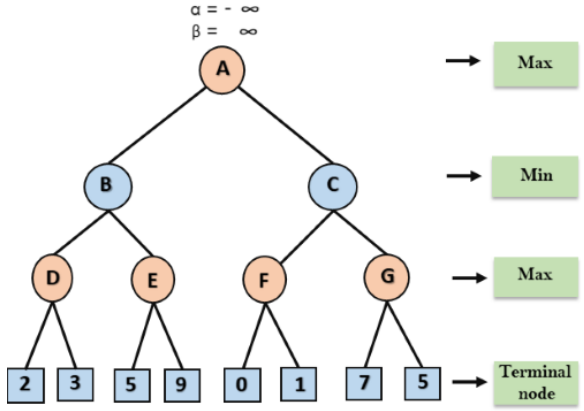


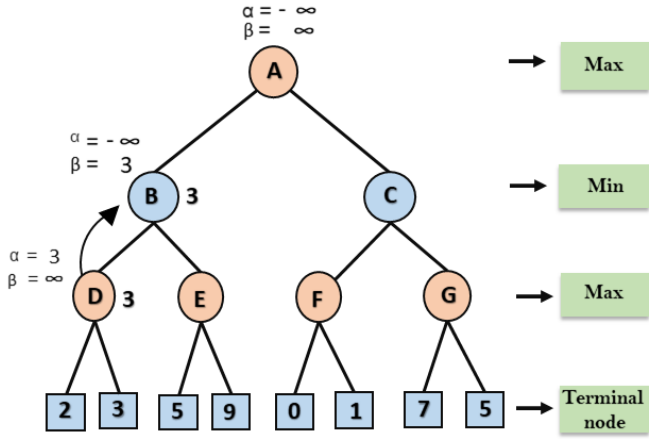


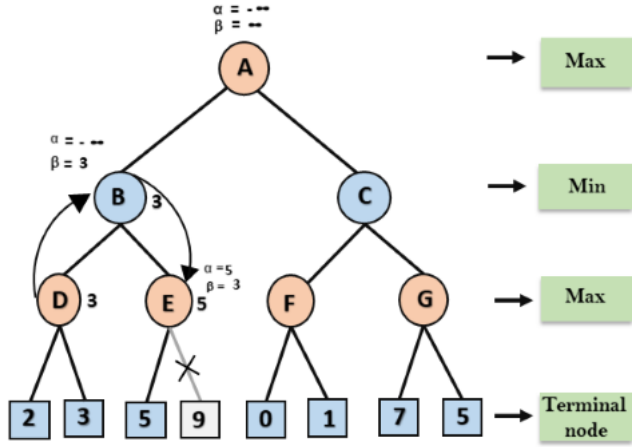


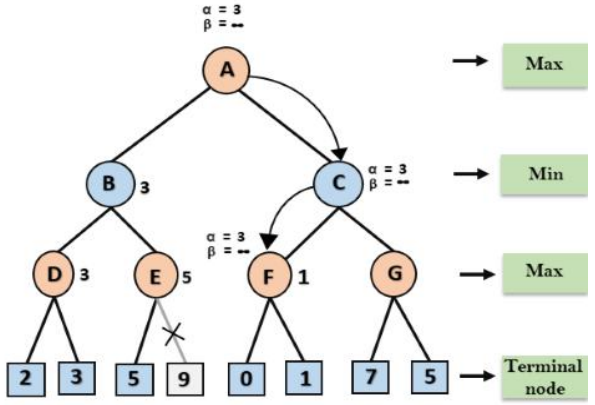
Worked Examples

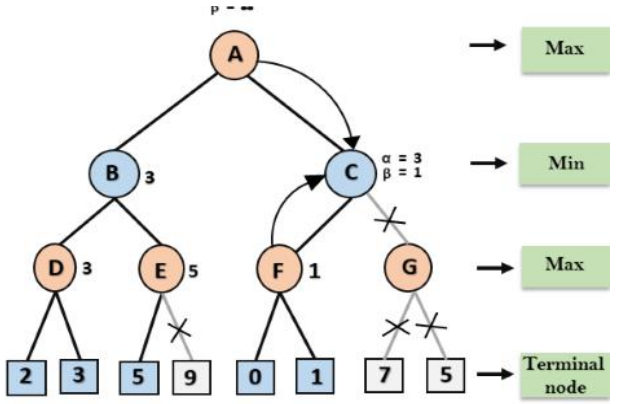
Answer:

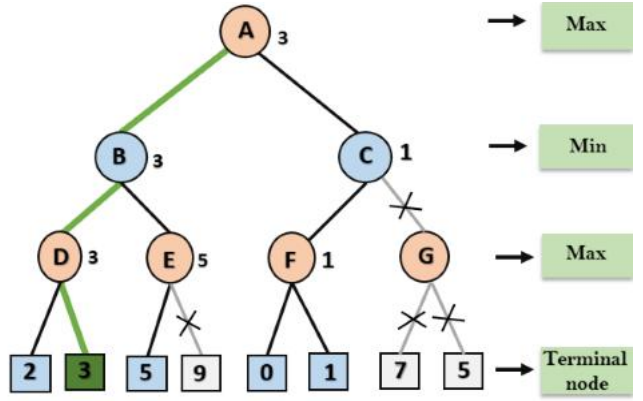












No. of α cut-off = 3
No. of β cut-off = 1

Question 11: Given two propositions $P \rightarrow Q$ and $\neg P \vee Q$, show that both are equivalent using truth table.

Answer:

P	Q	$P \rightarrow Q$	$\neg P$	$\neg P \vee Q$
T	T	T	F	T
T	F	F	F	F
F	T	T	T	T
F	F	T	T	T

Question 12: Construct the truth table for the given proposition $(P \to Q) \wedge (Q \to R)$

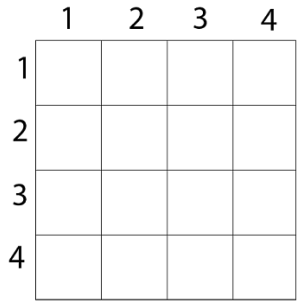
Answer:

P	Q	R	$P \to Q$	$Q \to R$	$(P \to Q) \wedge (Q \to R)$
T	T	T	T	T	T
T	T	F	T	F	F
T	F	T	F	T	F
T	F	F	F	T	F
F	T	T	T	T	T
F	T	F	T	F	F
F	F	T	T	T	T
F	F	F	T	T	T

Use $P \to Q = \neg P \vee Q$

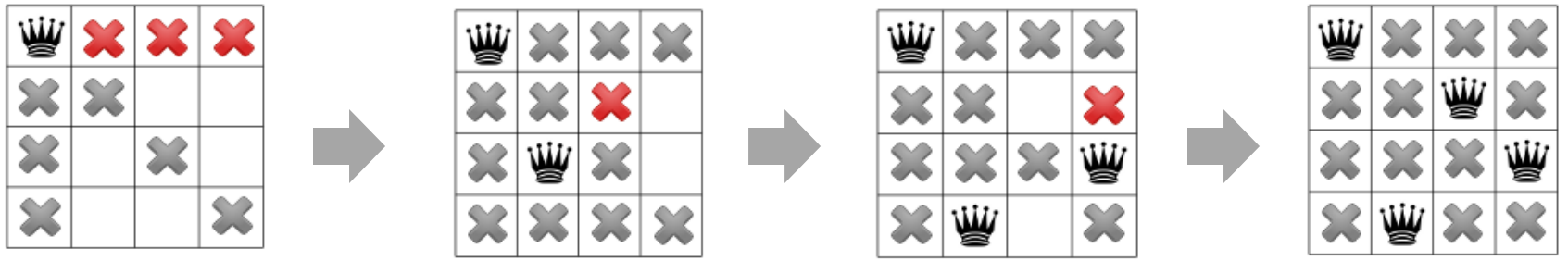
Worked Examples

Question 13: Solve the 4-Queen problem using backtracking (Rules: No two queens can attack each other in same row, same column and diagonal)



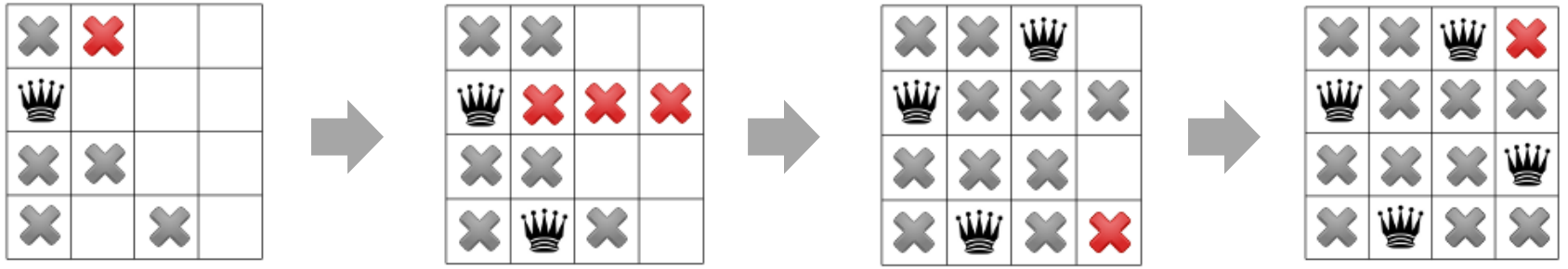
Answer:

Iteration 1



FAIL

Iteration 2



SUCCESS

Question 14: Translating the following knowledge into Logical Expressions

1. Some student in the class has visited Mexico
2. Every student in the class has visited either Canada or Mexico.
3. Every picture larger than one megabyte will be deleted

Answer:

1. $\exists x\,(S(x) \wedge M(x))$
2. $\forall x\,(S(x) \rightarrow C(x) \vee M(x))$
3. $\forall x\,(S(x,1) \rightarrow D(x))$

Question 15: Consider the conditional proposition $p \rightarrow q$. Which among the converse, inverse and contrapositive are logically equivalent to the conditional statement?

Answer: Contrapositive

p	q	$\neg p$	$\neg q$	$p \rightarrow q$	$q \rightarrow p$	$\neg p \rightarrow \neg q$	$\neg q \rightarrow \neg p$
T	T	F	F	T	T	T	T
T	F	F	T	F	T	T	F
F	T	T	F	T	F	F	T
F	F	T	T	T	T	T	T

Exercises

Q. 1 A 3×3 board with 8 tiles, with every tile a number and one empty space. The objective is to place the numbers on tiles to match final configuration using the empty space. Slide four adjacent (left, right, above and below) tiles into the empty space. Find the cost of solving the problem.

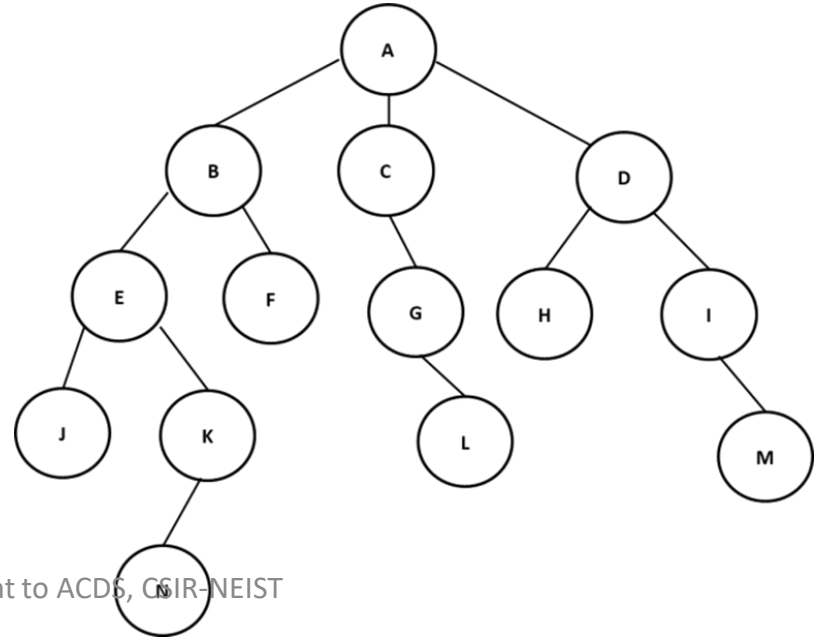
1	*2*	*3*
5	*8*	*6*
-	*7*	*4*

Start State

1	*2*	*3*
5	*6*	-
7	*8*	*4*

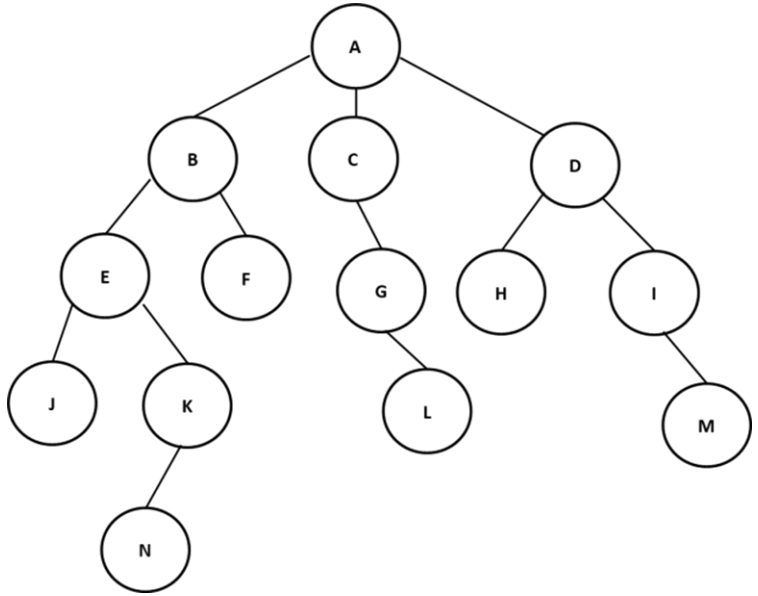
Final State

Q. 2 Using uninformed search technique Breadth First Search (BFS) to reach the destination node N from A.

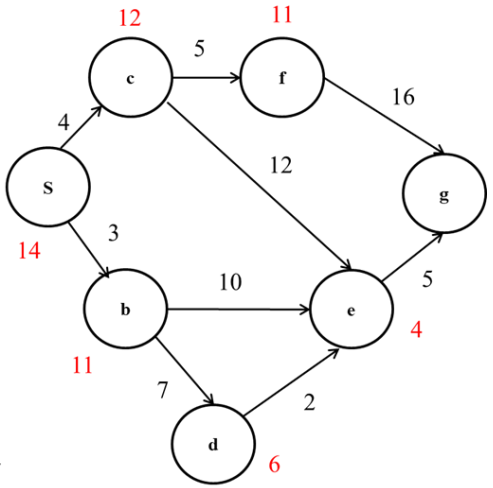


Exercises

Q. 3 Using uninformed search technique Depth First Search (BFS) to reach the destination node N from A.



Q. 4 For heuristic searching, use A* algorithm to find an optimized path from Source (S) to destination (D) using the given route graph.



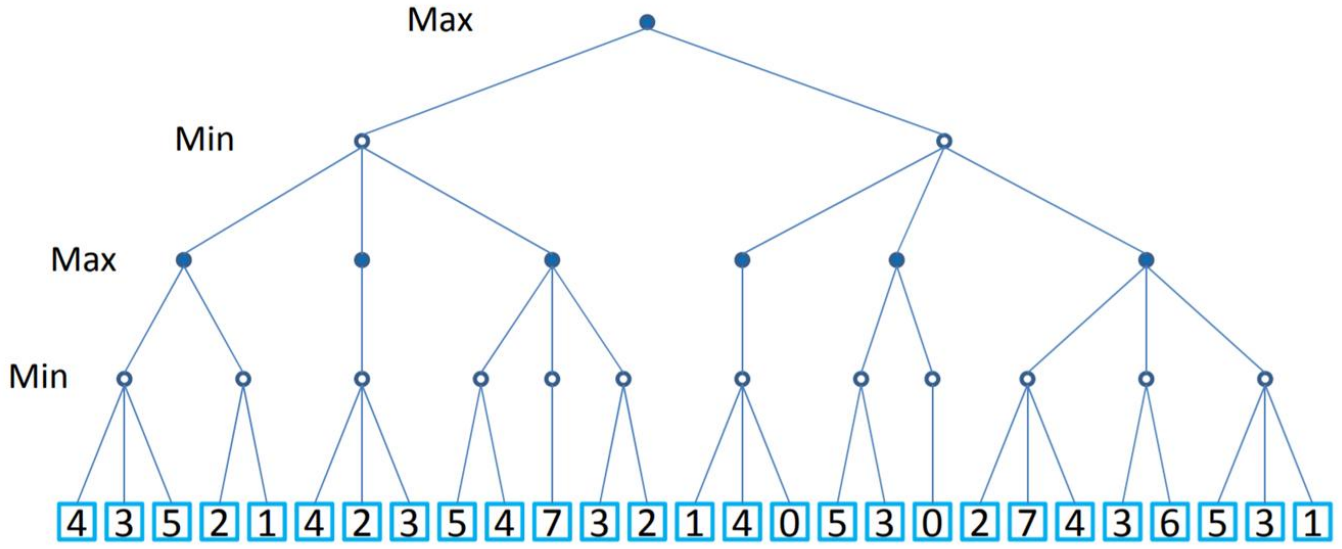
Exercises

Q. 5 KIAA is a two-player game. The rules are as follows:
"The game starts with a single stack of 7 tokens. At each move a player selects one stack and divides it into two non-empty, non-equal stacks. A player who is unable to move loses the game."

Draw the complete search tree for KIAA.

Q. 6 Perform min-max algorithm on the tree below. Also find the α and β cut-off.

a) Without αβ pruning
b) With αβ pruning (show the pruned nodes and branches).



Exercises

Q. 7 Let p and q be the propositions

p: You drive over 65 miles per hour.
q: You get a speeding ticket.

Write these propositions using p and q and logical connectives (including negations).

a) You do not drive over 65 miles per hour.
b) You drive over 65 miles per hour, but you do not get a speeding ticket.
c) You will get a speeding ticket if you drive over 65 miles per hour.
d) If you do not drive over 65 miles per hour, then you will not get a speeding ticket.
e) Driving over 65 miles per hour is sufficient for getting a speeding ticket.
f) You get a speeding ticket, but you do not drive over 65 miles per hour.
g) Whenever you get a speeding ticket, you are driving over 65 miles per hour.

Q. 8 Construct the truth tables for the following compound propositions:

a) $p \rightarrow (\neg q \vee r)$
b) $\neg p \rightarrow (q \rightarrow r)$
c) $(p \rightarrow q) \vee (\neg p \rightarrow r)$

Exercises

Q. 9 Show that the following propositions are equivalent using truth table:

a) $(p \wedge q) \rightarrow (p \vee q) \equiv \neg (p \wedge q) \vee (p \vee q)$

b) $\neg (p \vee (\neg p \wedge q)) \equiv \neg p \wedge \neg (\neg p \wedge q)$

Q. 10 Consider three genes, Gene A, Gene B and Gene C. Construct a Boolean logic to identify all the possible combinations of the three genes using a truth table. If a new gene Gene D is formed using the $\neg (A \wedge (\neg B \vee C))$, where A, B, C represent the three genes, then derive the Boolean logic for Gene D?

Q. 11. Consider the truth table for 3 genes given below at time t_0. If at time t_1 the gene structures changes by following rules, then derive the new truth table for the Gene1, Gene 2 and Gene 3.

Rule 1: Gene 1' = Gene 2;

Rule 2: Gene 2' = Gene 1 OR Gene 3

Rule 3: Gene 3' = (Gene 1 AND Gene 2) OR (Gene 2 AND Gene 3) OR (Gene 1 AND Gene 3)

Exercises

Q. 12 Find a solution for 8 Queen Problem, if the current positions of Q1 = (0, 1), Q2 = (2, 6) and Q8 = (7, 7).

	0	1	2	3	4	5	6	7
0								
1								
2								
3								
4								
5								
6								
7								

Q. 13 Use state space search to achieve the Goal (G) from the start (S) matrix. Slide four adjacent (left, right, above and below) tiles into the empty space.

3	1	2
4		5
6	7	8

Start (S)

	1	2
3	4	5
6	7	8

Goal (G)

Q. 14. Consider the famous **Missionaries and Cannibals** problem in AI which is usually stated as follows – "Three missionaries and three cannibals are on one side of a river, along with a boat that can hold one or two people. Find a way to get everyone to the other side, without ever leaving a group of missionaries in one place outnumbered by the cannibals in that place."

a) Formulate the problem precisely, making only those distinctions necessary to ensure a valid solution.
b) Draw a diagram of the complete state space.
c) Implement and solve the problem optimally using an appropriate search algorithm. Is it a good idea to check for repeated states?
d) Why do you think people have a hard time solving this puzzle, given that the state space is so simple?

Q. 15 Consider the following **Water Jug Problem** which is stated as – "You are given two jugs, a 4-gallon one and a 3-gallon one, a pump which has unlimited water which you can use to fill the jug, and the ground on which water may be poured. Neither jug has any measuring markings on it. How can you get exactly 2 gallons of water in the 4-gallon jug? " Come up with a solution by drawing the state space?

References

- Russell, S.J., & Norvig, P. (2015). *Artificial Intelligence: A Modern Approach.* Englewood Cliffs, NJ: Pearson Education Limited.

➢ This book provides a details on AI with problem solving. It also covers topics like knowledge, reasoning and planning. Communicating approaches like computer vision, robotics are also part of this book.

- Jackson, P.C., (1985). *Introduction to Artificial Intelligence*. New York: Dover Publications, Inc.

➢ This book presents an introduction to the science of reasoning processes in computers, and the research approaches. An easy-to-read coverage of problem-solving methods, representation and models, game playing, automated understanding of natural languages, heuristic search theory, robot systems, heuristic scene analysis etc. are also covered.

- Rothman, D., (2018). *Artificial Intelligence By Example*. Livery Place, Birmingham: Packt Publishing.

➢ This book is a starting point to understand how AI is built, with the help of intriguing examples and case studies. Some of the most advanced machine learning models, how to apply AI to blockchain and IoT, and develop emotional quotient in chatbots using neural networks have also been covered.

- Pattanayak, S., (2019). *Intelligent Projects using Python*. Livery Place, Birmingham: Packt Publishing.

➢ The book covers detailed implementation of projects from all the core disciplines of AI. It starts by covering the basics of how to create smart systems using machine learning and deep learning techniques. Various neural network architectures such as CNN, RNN, LSTM, to solve critical new world challenges have also been assimilated.

References

- Yannakakis, G.N & Togelius, J., (2018). *Artificial Intelligence and Games*. Gewerbestrasse, Cham: Springer International Publishing.

➢ This book covers a lot of contents in game development and academic game research. It also focuses in AI designs, human-computer interactions and computational intelligence.

- Deshpande, A. & Kumar, M., (2018). *Artificial Intelligence for Big Data*. Liverly Place, Birmingham: Packt Publishing.

➢ This book helps to learn to use Machine Learning algorithms such as k-means, SVM, RBF, and regression to perform advanced data analysis. It also covers the current status of Machine and Deep Learning techniques to work on Genetic and Neuro-Fuzzy algorithms. In addition, it also explores how to develop Artificial Intelligence algorithms to learn from data, why they are necessary, and how they can help solve real-world problems.

- Domingos, P., (2015). *The Master Algorithm: How the quest for the ultimate learning machine will remake our world*. London: Penguin Groups.

➢ This book reveals how machine learning is remaking business, politics, science and war.

THANK YOU