HEADING: :	**Dark Purple 1, Light Gold 2**	**20**
TYPES:	**Green**	**18**
SUBTYPES:	**Yellow**	**16**
Important Points:	Orange	16
Important Points:	Gold	16
Important Points:	Lime	16

UNIT 1

Introduction: Intelligent agents, searching:
Basics of AI,
Intelligent Agents:
Agents and environment;
Rationality;
the nature of environments;
the structure of agents.

Problem-solving:
Problem-solving agents;
Searching for solution;
Uninformed search strategies;
Informed search strategies,
Heuristic Functions

AGENT

An **agent** is anything that **perceiving** its environment through **sensors** and **acting** upon that environment through **actuators**

Example:

- Human is an agent
- A robot is also an agent with cameras and motors
- A thermostat detecting room temperature.

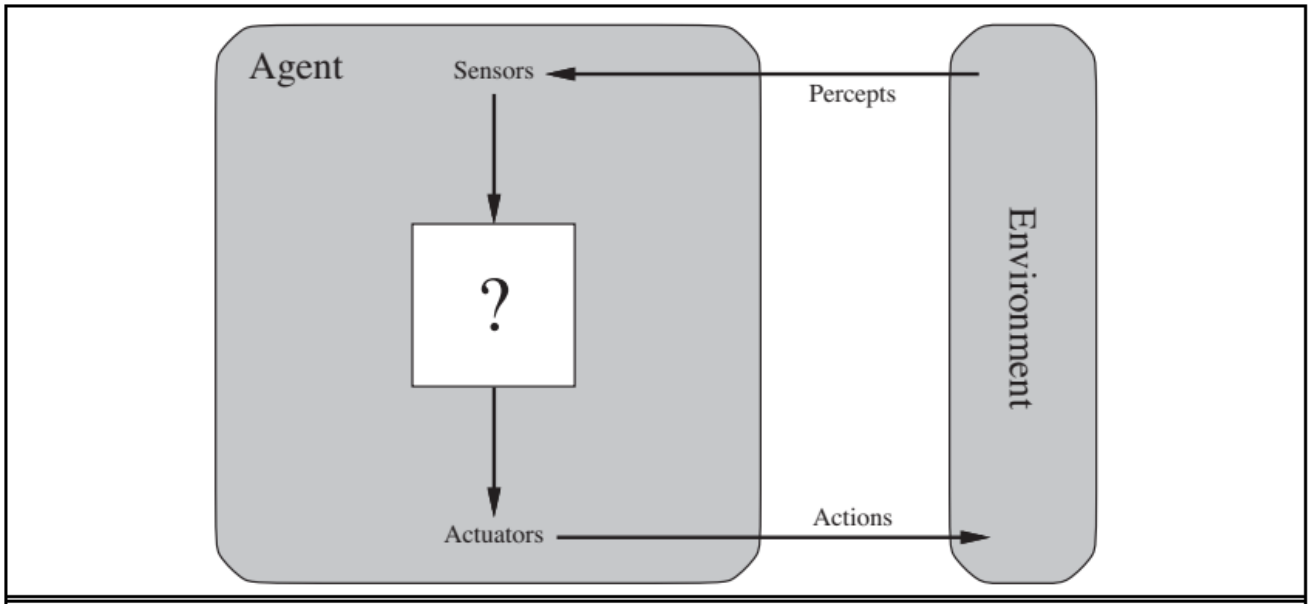


Figure 2.1 Agents interact with environments through sensors and actuators.

PEAS

Agent Type	Performance Measure	Environment	Actuators	Sensors
Taxi driver	Safe, fast, legal, comfortable trip, maximize profits	Roads, other traffic, pedestrians, customers	Steering, accelerator, brake, signal, horn, display	Cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensors, keyboard

Figure 2.4 PEAS description of the task environment for an automated taxi.

Interactive English tutor	Student's score on test	Set of students, testing agency	Display of exercises, suggestions, corrections	Keyboard entry

Rational Agent

A rational agent acts so as to maximize the expected value of the performance measure, given the percept sequence it has seen so far.

List the Different Types of Agent

Based on their characteristics

- reactive or proactive
- fixed or dynamic environment
- single or multi-agent systems

Based on intelligence and capability

- Simple Reflex Agents
- Model-Based Reflex Agents
- Goal-Based Agents
- Utility-Based Agents
- Learning Agent
-- Multi-agent systems
-- Hierarchical agents

(Sketches for 5 of THEM)

SIMPLE REFLEX Agent

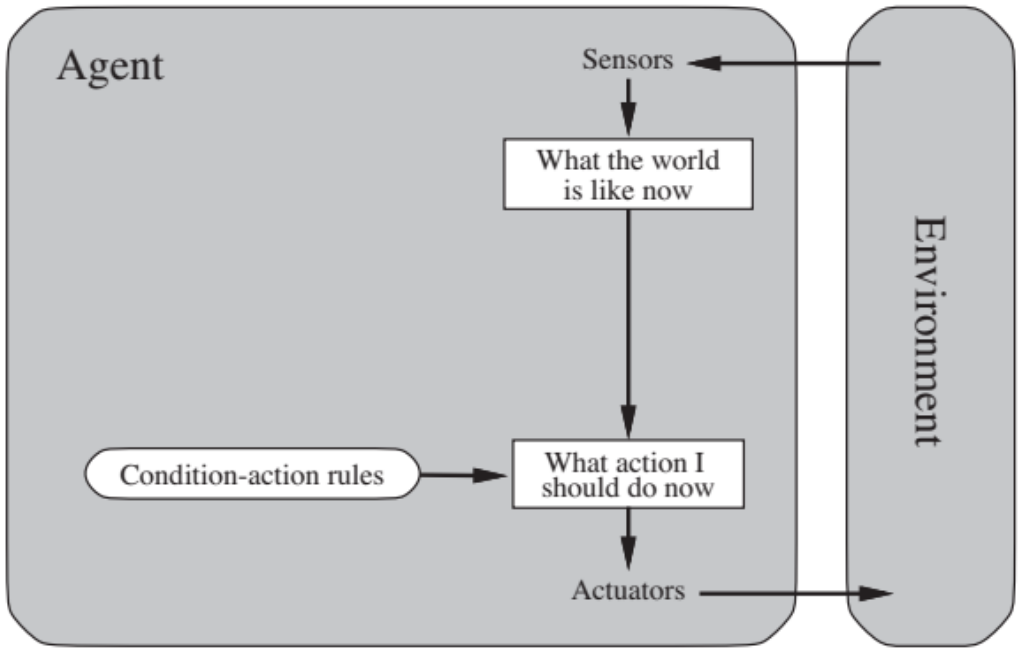


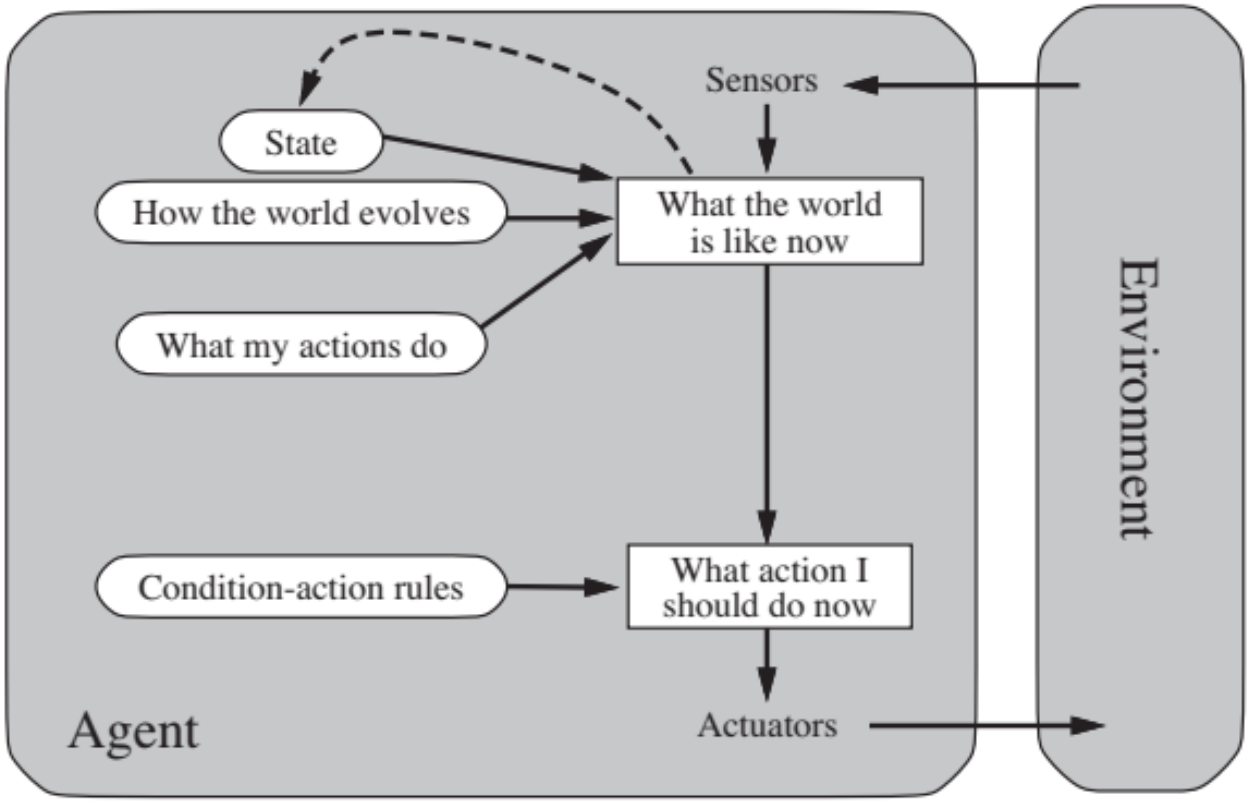
Figure 2.9 Schematic diagram of a simple reflex agent.

These agents select actions on the basis of the current percept, ignoring the rest of the percept history.

Condition – Action Rule

if *car-in-front-is-braking* **then** *initiate-braking*.

MODEL-based REFLEX Agent



The most effective way to handle partial observability is for the agent to keep track of the part of the world it can't see now

That is, the agent should maintain some sort of internal state

This knowledge about "how the world works" is called a model

GOAL-based AGENT

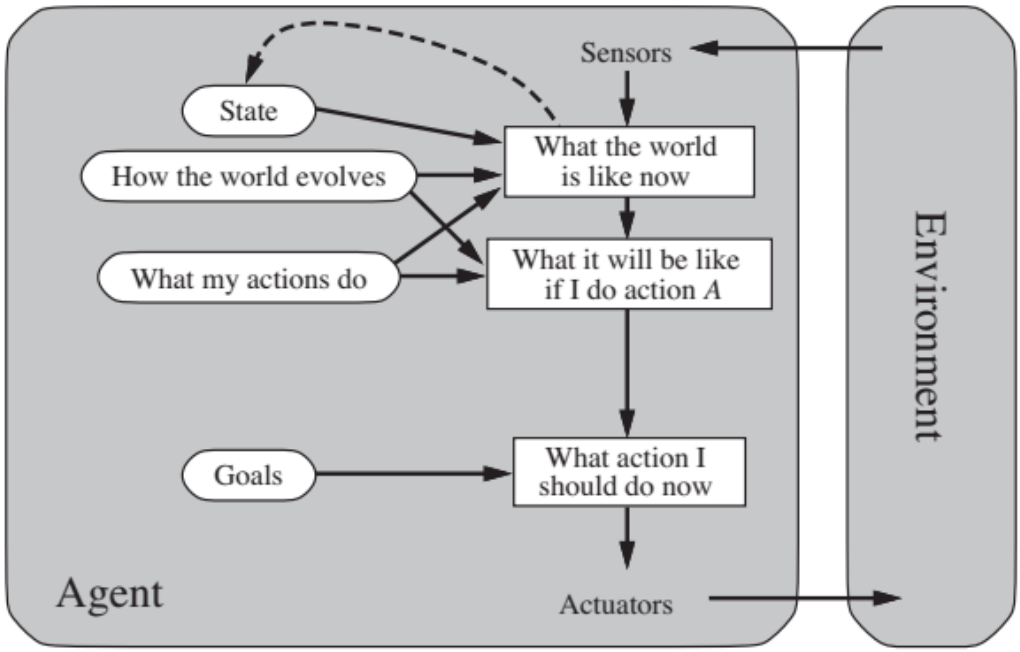


Figure 2.13 A model-based, goal-based agent. It keeps track of the world state as well as a set of goals it is trying to achieve, and chooses an action that will (eventually) lead to the achievement of its goals.

Knowing something about the current state of the environment is not always enough to decide what to do

Agent needs some sort of goal information that describes situations that are desirable - for example, being at the passenger's destination

Search and Planning

Although the goal-based agent appears less efficient, it is more flexible because the knowledge that supports its decisions is represented explicitly and can be modified

UTILITY based AGENT

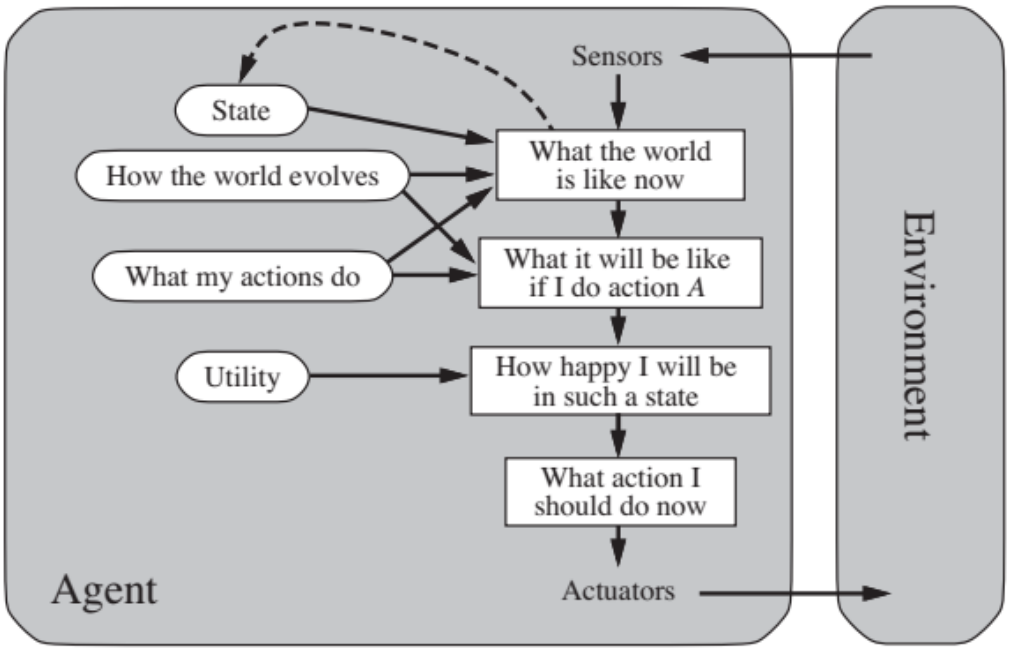


Figure 2.14 A model-based, utility-based agent. It uses a model of the world, along with a utility function that measures its preferences among states of the world. Then it chooses the action that leads to the best expected utility, where expected utility is computed by averaging over all possible outcome states, weighted by the probability of the outcome.

Goals alone are not enough to generate high-quality behaviour in most environments

Many action sequences will get the taxi to its destination (thereby achieving the goal) but some are quicker, safer, more reliable, or cheaper than others

An agent's utility function is essentially an internalization of the performance measure

Functions of Agent

- Agent Behavior is described by the Agent Function
- Agent Function maps the Percept sequence to an Action

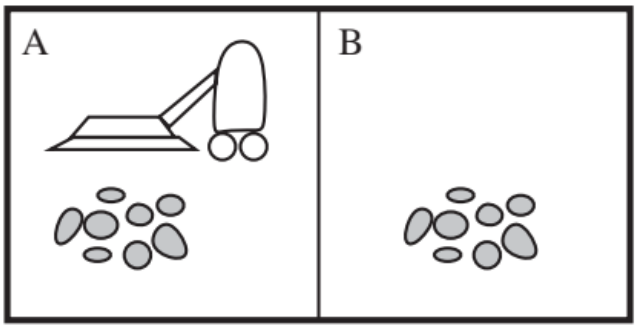


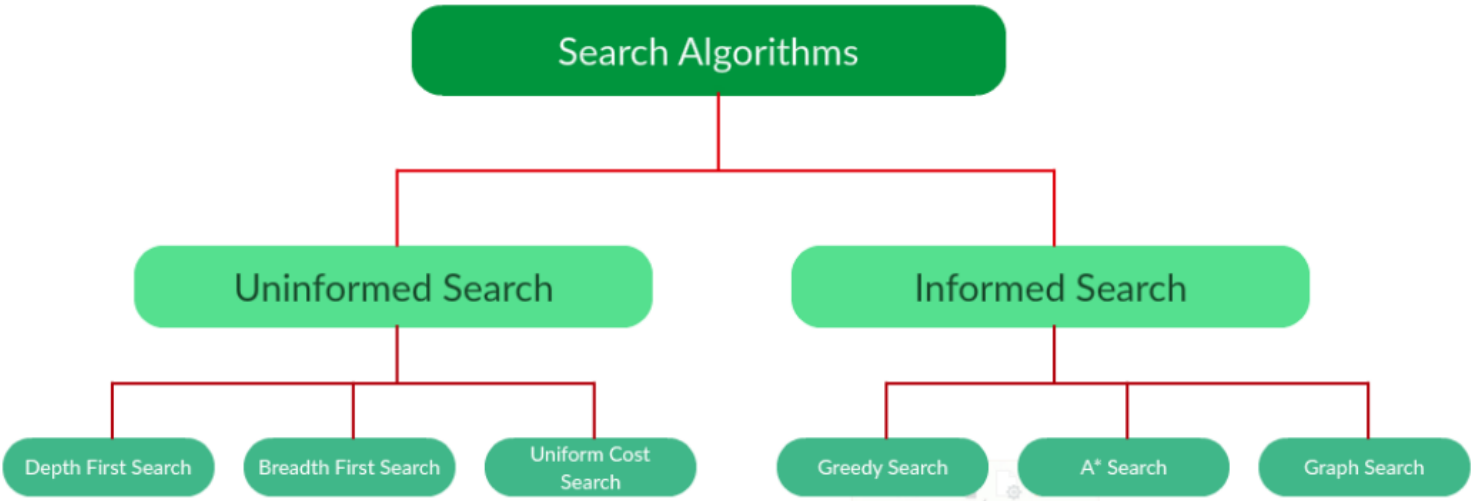
Figure 2.2 A vacuum-cleaner world with just two locations.

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.

An Intelligent Agent must sense, must act, must be autonomous (to some extent). It also must be rational.

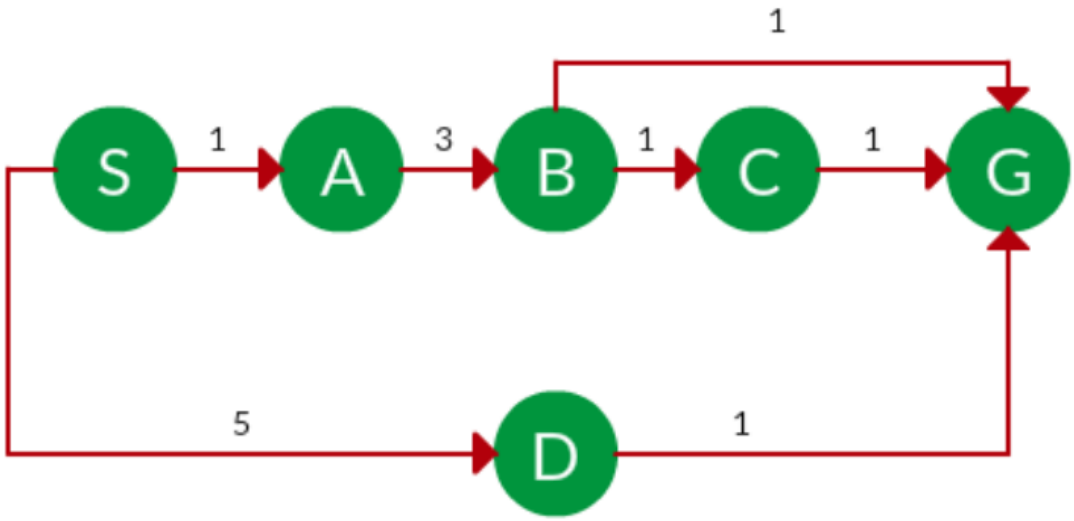
SEARCH ALGORITHMS



Uninformed Search (or) Blind Searching

BFS:
DFS:
UCS: There is COST for the edge

Question. Which solution would UCS find to move from node S to node G if run on the graph below?



Goal is to find a path where the cumulative sum of costs is the least

Informed Search

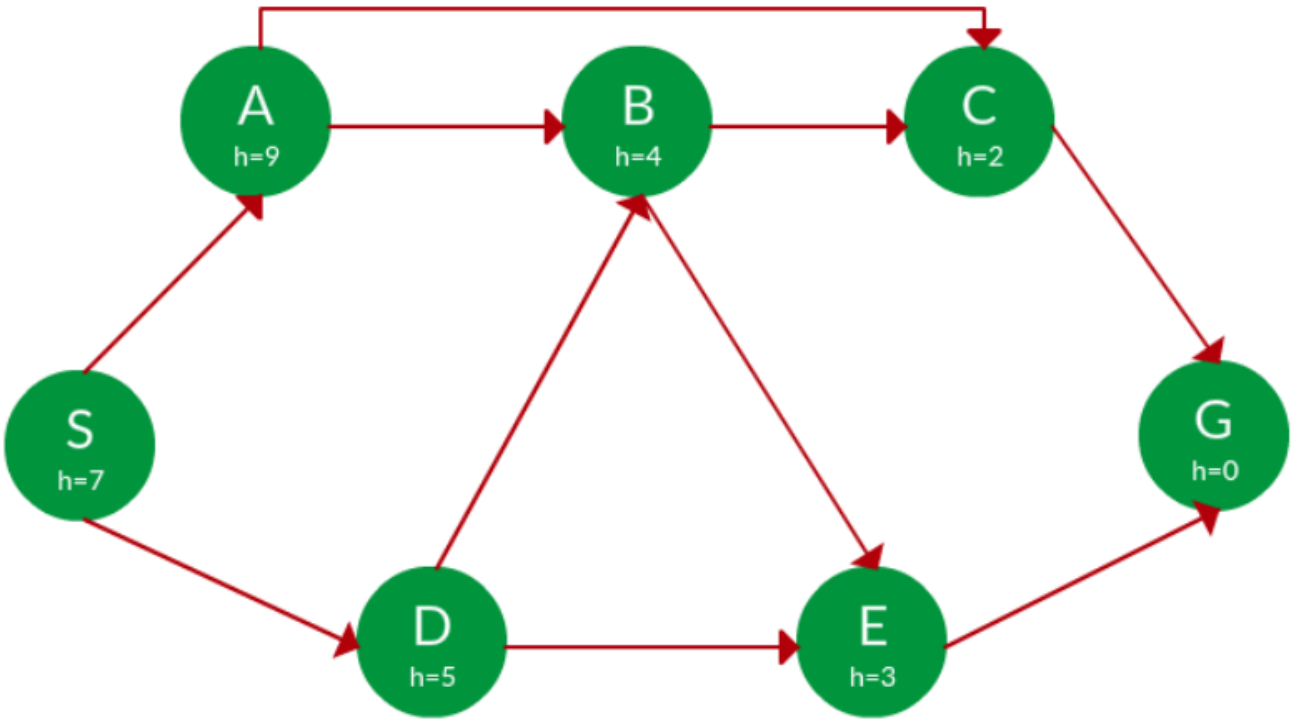
Greedy Search

Expand the node with the smallest estimated cost to reach the goal.

Closeness is estimated by heuristics

h(x) = Estimate of distance of node x from the goal node

Lower the value of h(x), closer is the node from the goal



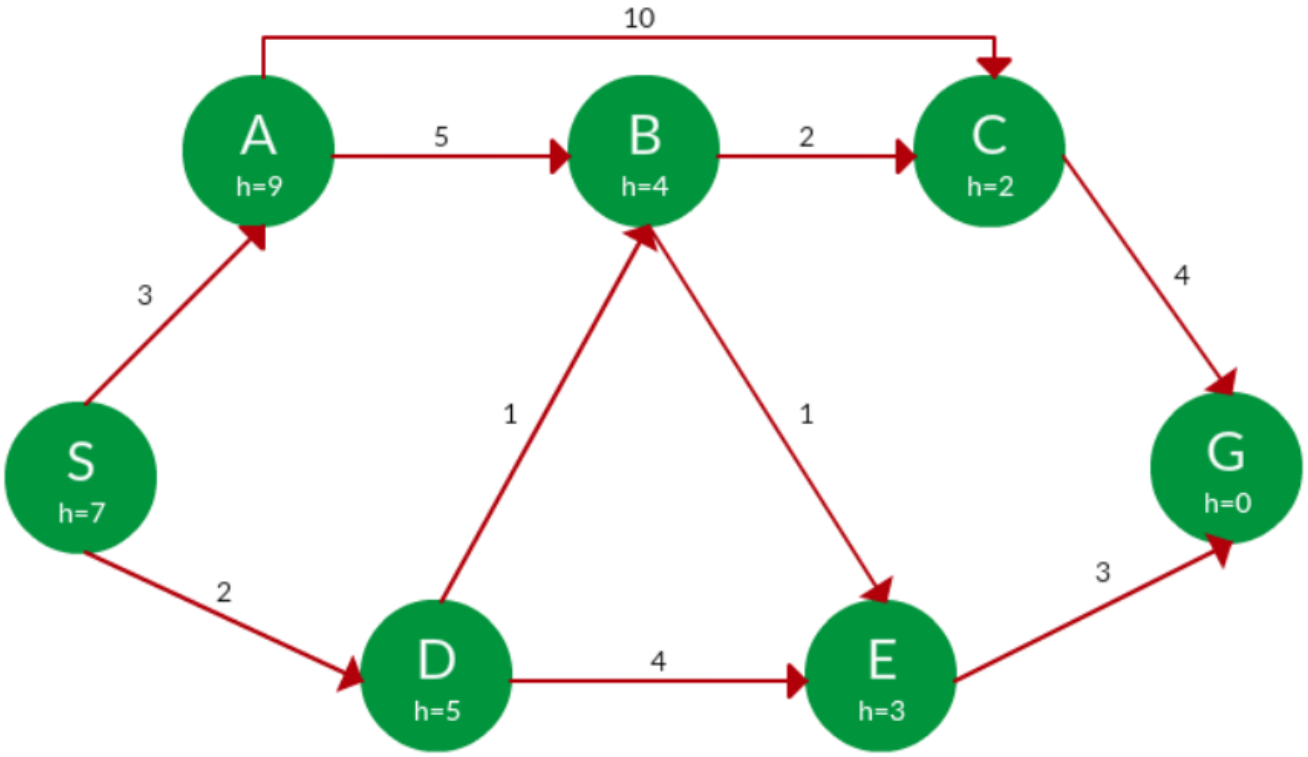
S -> D -> E -> G

A* Search

- Combines UCS with Greedy search
- Best First Search

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

g(n) = sum of edge costs from start to n
h(n) = estimate of lowest cost path from n to goal
f(n) = actual distance so far + estimated distance remaining

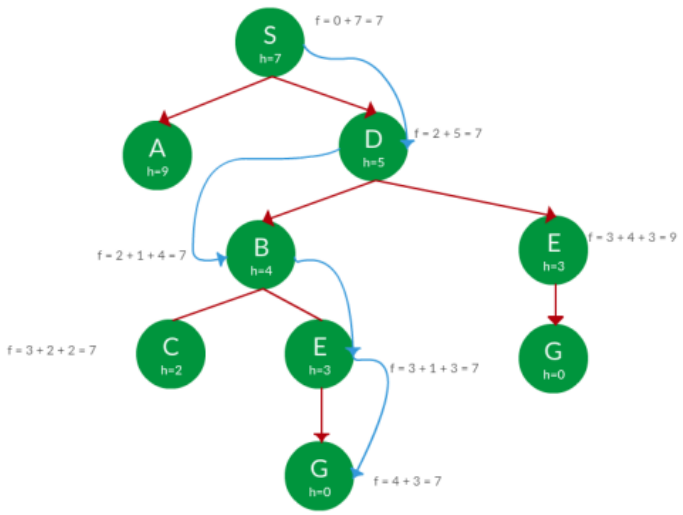


Path	h(x)	g(x)	f(x)
S	7	0	7
S -> A	9	3	12
S -> D	5	2	7
S -> D -> B	4	2 + 1 = 3	7
S -> D -> E	3	2 + 4 = 6	9

Path: S -> D -> B -> E -> G
Cost: 7

A* Graph Search

the **Solution.** We solve this question pretty much the same way we solved last question, but in this case, we keep a track of nodes explored so that we don't re-explore them.



Explored Nodes
S
D
B
E
G

Path: S -> D -> B -> E -> G
Cost: 7

UNIT 2

Adversarial search, constraint satisfaction problems, logical agents:

Games,
Optimal decision in games,
Alpha-Beta Pruning,
Defining Constraint satisfaction problems;
Backtracking search for CSPs;
Knowledge-based agents

Probabilistic reasoning:

Representing knowledge in an uncertain domain;
Semantics of Bayesian Networks;
Efficient representation of conditional distributions;
Exact inference in Bayesian Networks;
Approximate inference in Bayesian Networks

BACKTRACKING

- Algorithmic technique
- Solving problems recursively
- By building a solution incrementally (one piece at a time)
- Removing those solutions that fail to satisfy the constraints of the problem

TYPES

1) Decision: Search for feasible solution
2) Optimization: Search for the best solution
3) Enumeration: Find all feasible the solution

EXAMPLE: Sudoku Solving

- We fill the digits one by one
- Whenever the current digit doesn’t lead to solution, we remove it
- And backtrack it till we find solution
- Better than naive approach (Permutations of all digits)

UNIT 3

Introduction,
Concept Learning and Decision Trees Learning Problems –
Designing Learning systems,
Perspectives and Issues –
Concept Learning –
Version Spaces and Candidate Elimination Algorithm –
Inductive bias –

Decision Tree learning–
Representation –
Algorithm –
Heuristic Space Search

UNIT 4

Bayesian And Computational Learning
Bayes Theorem – Concept Learning – Maximum Likelihood – Minimum Description Length Principle – Bayes Optimal Classifier – Gibbs Algorithm – Naïve Bayes Classifier – Bayesian Belief Network – EM Algorithm – Probably Learning – Sample Complexity for Finite and Infinite Hypothesis Spaces – Mistake Bound Model

UNIT 5

Instant Based Learning K- Nearest Neighbor Learning, Locally Weighted Regression, Radial Basis Functions, Case-Based Reasoning Reinforcement Learning: The Learning Task, Q-Learning, Temporal Difference Learning