BITS Pilani
Pilani Campus

Artificial & Computational Intelligence

AIML CLZG557

M1 : Introduction

Raja vadhana P
Assistant Professor,
BITS - CSIS

Agenda

- Course Administration
- Getting Started (with some definitions)
- Course Overview with example

Course Administration

About the course

- Focus on
  - principles of artificial intelligence
  - concepts, algorithms involved in building rational agents
  - topics covered like
    - (informed and uninformed) search & applications
    - (logical & probabilistic) knowledge representation
    - (logical & probabilistic) Reasoning & applications
  - topics not-covered like
    - Formal introduction to machine learning algorithms, neural networks etc., are covered as a ML course is running in parallel, Deep neural networks, which are part of AI as well.
    - Hardware aspects of the Design

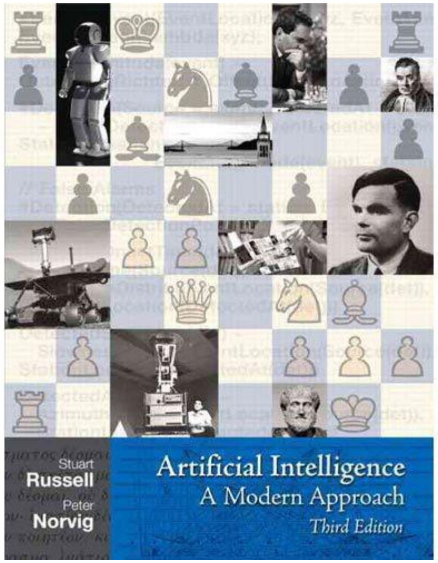
Course Outline

- **Pedagogy**
  - Weekly online live sessions
  - Webinars on lab implementation
  - Assignment:
    - 1 Quiz-5%,
    - 2 Assignments- 25%

- **Lab Modules**
  - Supported by 6 lab capsules for practical implementation and better understanding of the concepts learned in the live lecture sessions.

About the course

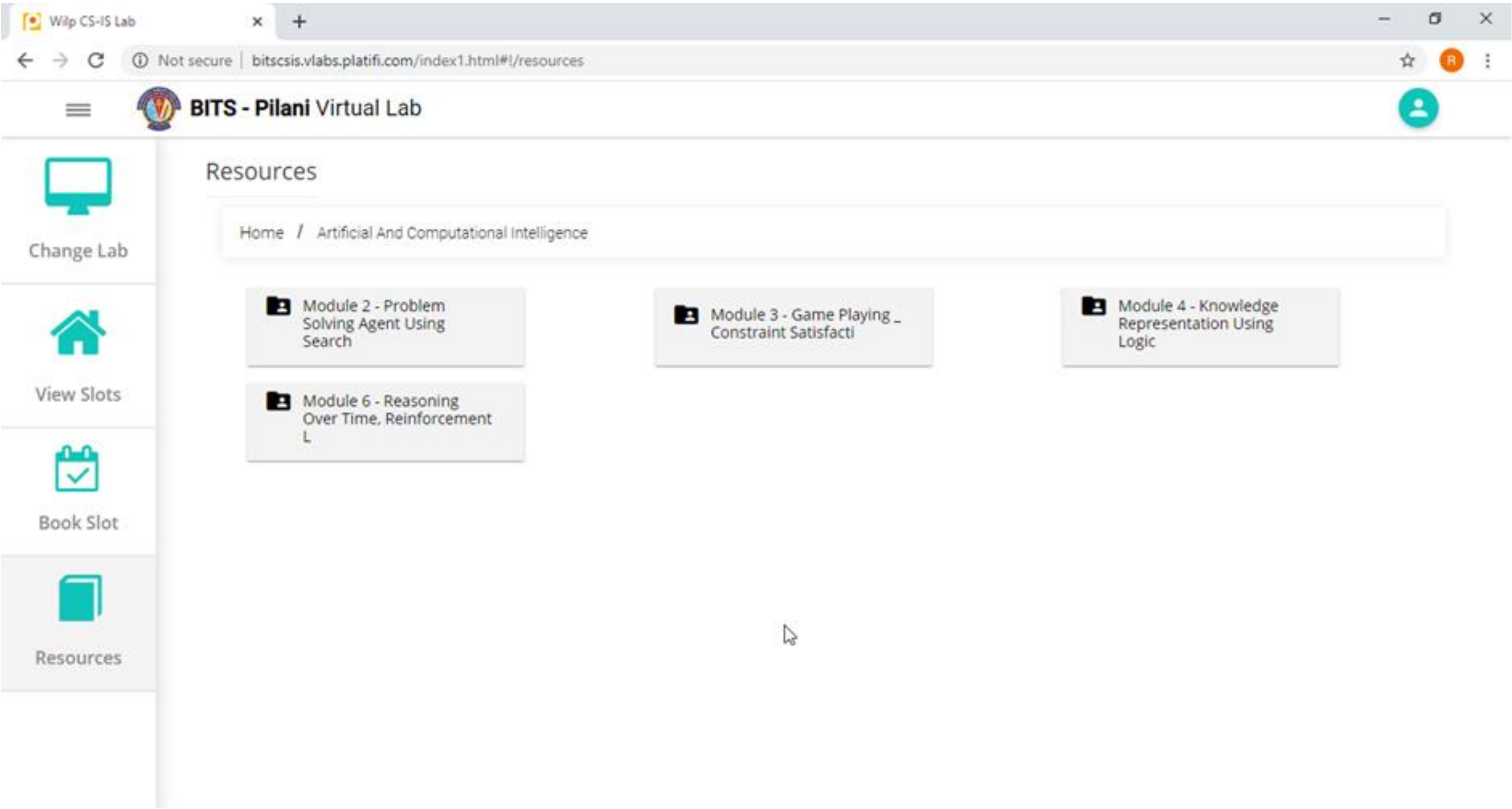
Text Book



Exercises : In Python & its libraries

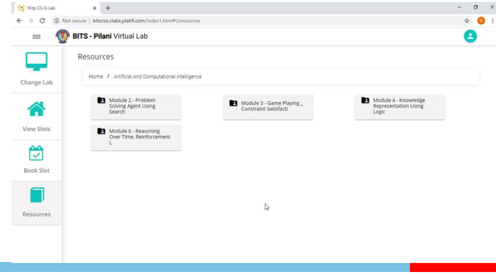
Evaluation : 25% Assignment + 5% Quiz + 30% Mid Semester + 40% End Semester

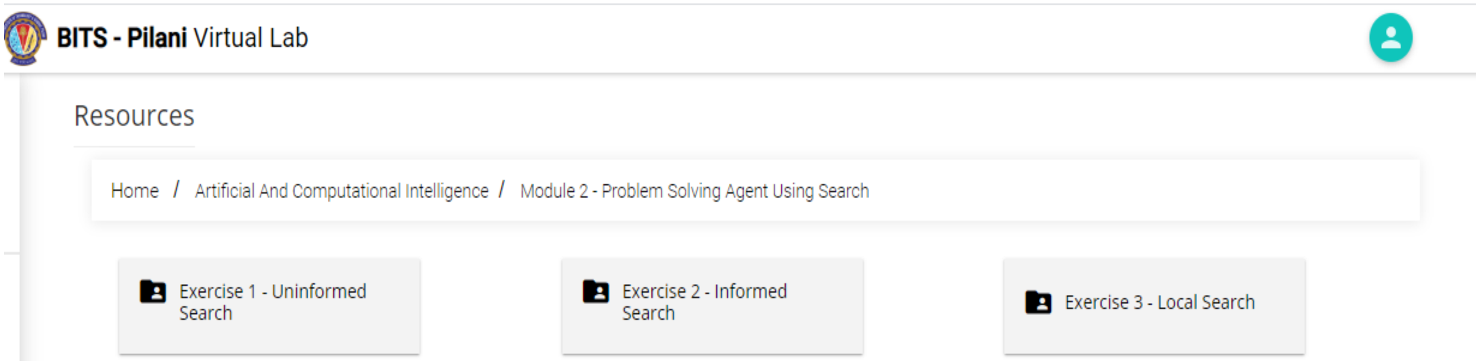
About Labs



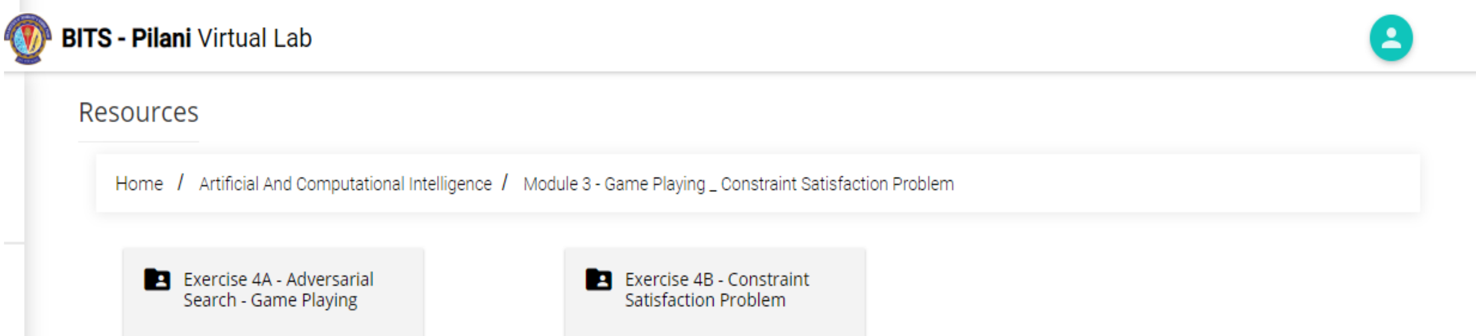
Exercises : In Python & its libraries

About Labs

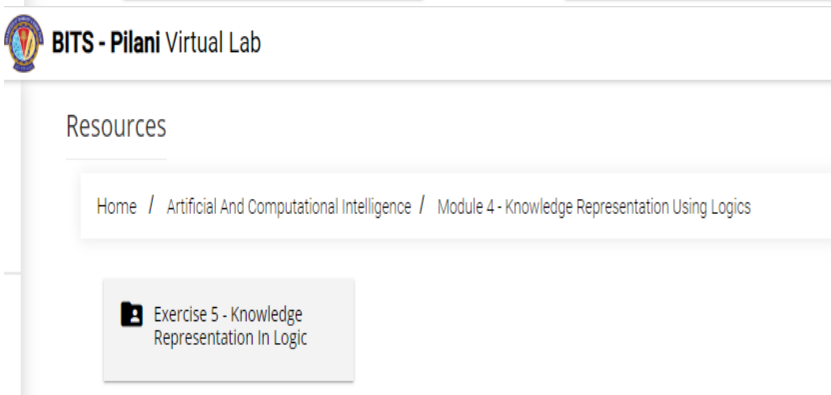




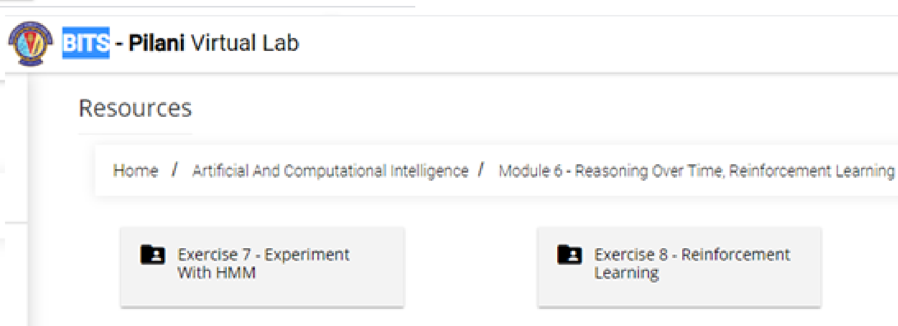
BITS - Pilani Virtual Lab
Resources
Home / Artificial And Computational Intelligence / Module 2 - Problem Solving Agent Using Search
Exercise 1 - Uninformed Search
Exercise 2 - Informed Search
Exercise 3 - Local Search



BITS - Pilani Virtual Lab
Resources
Home / Artificial And Computational Intelligence / Module 3 - Game Playing _ Constraint Satisfaction Problem
Exercise 4A - Adversarial Search - Game Playing
Exercise 4B - Constraint Satisfaction Problem

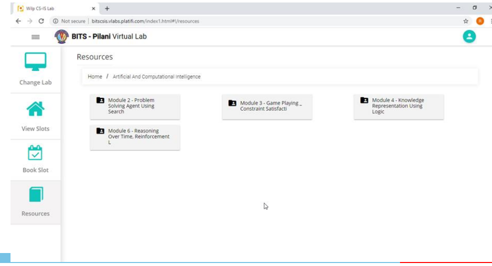


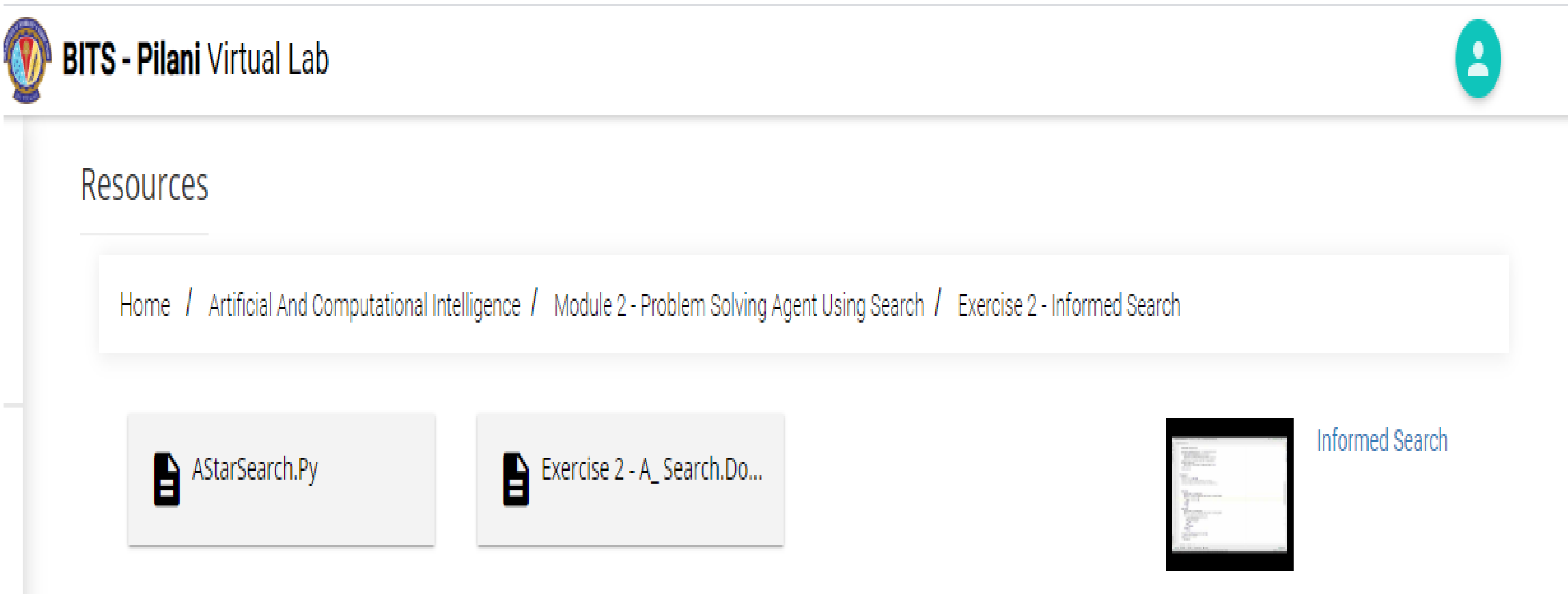
BITS - Pilani Virtual Lab
Resources
Home / Artificial And Computational Intelligence / Module 4 - Knowledge Representation Using Logics
Exercise 5 - Knowledge Representation In Logic



BITS - Pilani Virtual Lab
Resources
Home / Artificial And Computational Intelligence / Module 6 - Reasoning Over Time, Reinforcement Learning
Exercise 7 - Experiment With HMM
Exercise 8 - Reinforcement Learning

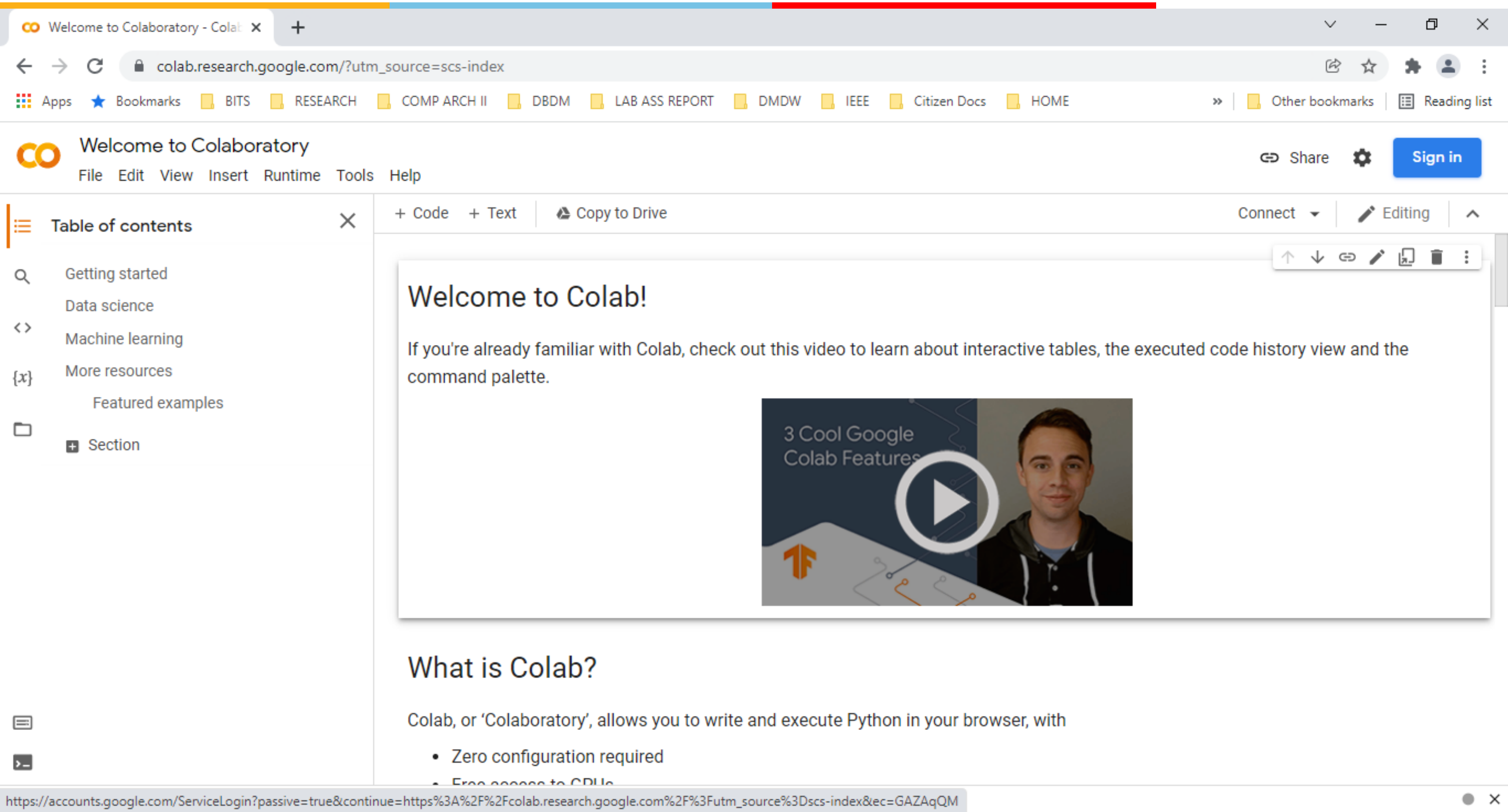
About Labs





Exercises : In Python & its libraries

About Labs



Exercises : In Python & its libraries

Artificial Intelligence

- Term coined by, *John McCarthy* (1955) & *Dartmouth Summer Research Project on Artificial Intelligence* (1956)

On September 2, 1955, the project was formally proposed by McCarthy, Marvin Minsky, Nathaniel Rochester and Claude Shannon. The proposal is credited with introducing the term 'artificial intelligence'.

The Proposal states[7]

> We propose that a 2-month, 10-man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.

Artificial Intelligence

- Term coined by, *John McCarthy* (1955) & *Dartmouth Summer Research Project on Artificial Intelligence* (1956)

On September 2, 1955, the project was formally proposed by McCarthy, Marvin Minsky, Nathaniel Rochester and Claude Shannon. The proposal is credited with introducing the term 'artificial intelligence'.

The Proposal states[7]

> “We propose that a 2-month, 10-man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.”

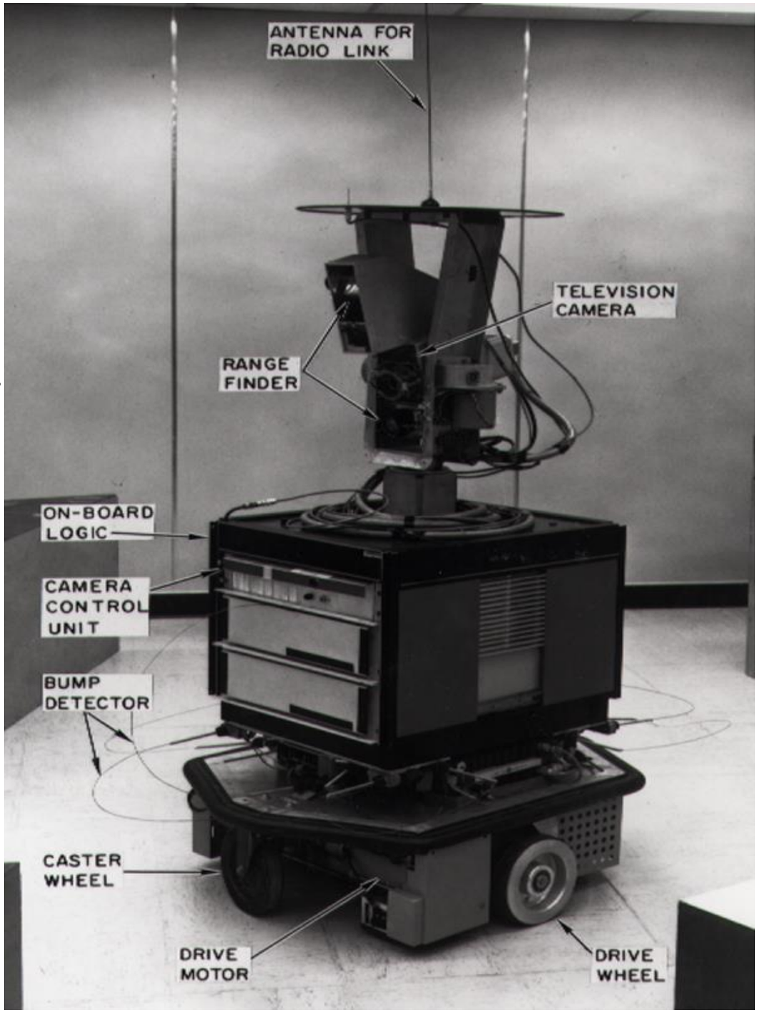
https://en.wikipedia.org/wiki/Dartmouth_workshop [01 June, 2019]

Larger Intent, Dream, Overconfidence ...

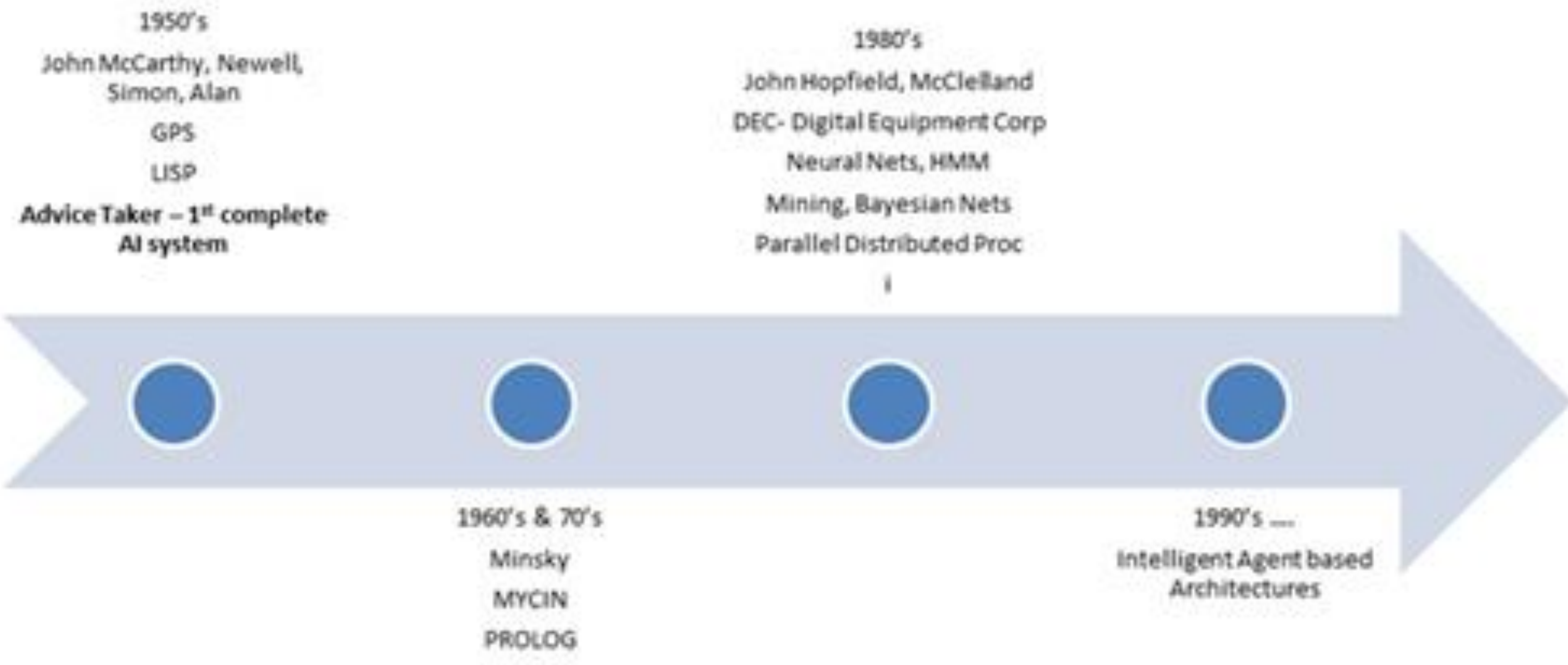
Some Early successes of Dartmouth

Many key projects were initiated after Dartmouth summer project.

Shakey robot - First mobile robot to perceive environment & reason about surroundings, actions - Introduced A* algorithm to find paths - Hough Transform for image analysis - Used Lisp for programming - visibility graph used for finding shortest paths in the presence of obstacles...



A brief history of AI



How is AI unique or in other words different from Applied Math?

Some Early successes of Dartmouth

DENDRAL -

Attempted to encode the domain expertise in molecular biology as an expert system

Led to the creation of expert systems for various other domain, including medical.

A milestone worship in the history of AI !!!

Perspectives of AI

Definitions

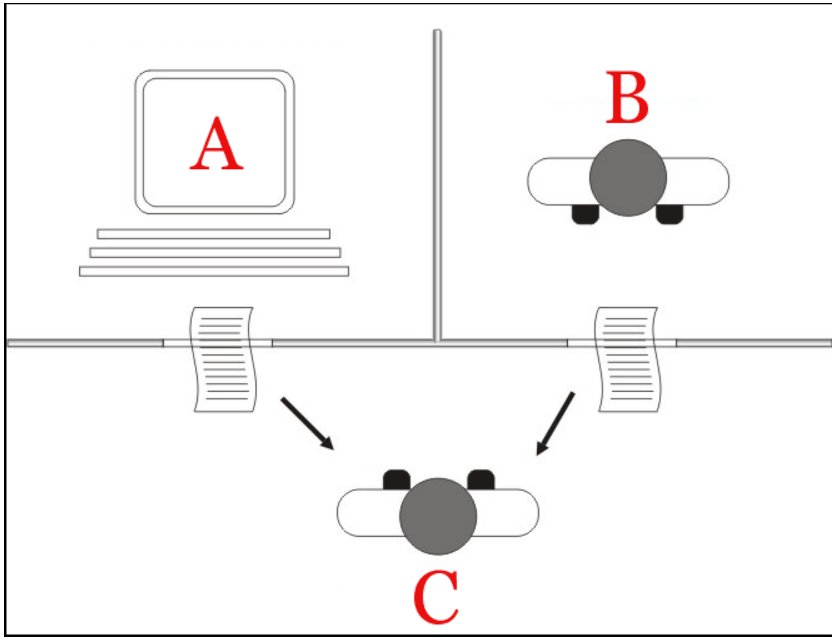
	Thought / Reasoning	Acting
Human Performance	THINKING HUMANLY “[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning, ... ” (Bellman, 1978)	ACTING HUMANLY “The art of creating machines that perform functions that require intelligence when performed by people” (Kurzweil, 1990)
Rational Performance	THINKING RATIONALLY “The study of computations that make it possible to perceive, reason, and act” (Winston, 1992)	ACTING RATIONALLY “Computational intelligence is the study of the design of intelligent agents” (Poole et al., 1998)

Acting Humanly

Turing Test Approach

- *Turing Test* & *Total Turing test* (operational test to determine an entity is intelligent / not) [50's]

- Skills necessary to pass these tests
  - NLP, Knowledge Representation, Automated Reasoning, ML + Computer Vision & Robotics(for total turing test)



Pictorial Representation of Turing Test from https://en.wikipedia.org/wiki/Turing_test

Acting Humanly

Turing Test Approach

Some Definitions of AI:

“The art of creating machines that perform functions that require intelligence when performed by people.” (Kurzweil, 1990)

“The study of how to make computers do things at which, at the moment, people are better.” (Rich and Knight, 1991)

Thinking Humanly

Cognitive Modelling Approach

- How do we capture human thinking to implement?
  - Introspection
  - Psychological Experiments
  - Brain Imaging

- System : *"General Problem Solver"* (*Newell and Simon, 1961*)
  - Designed to work as a universal problem solver
  - Problems represented by horn clauses
  - First AI Machine which has KB + Inference separation
  - Authors focus on this is on comparing the trace of its reasoning steps to traces of human subjects solving the same problems

- Growth of Cognitive science and AI supports each other

Passing the Turing Test

Transcript of a chat

EUGINE - a thirteen-year-old Ukrainian boy, chats

JUDGE: Hello.
EUGENE: Hello, I'm really glad to have the chance to chat with you! My guinea pig Bill sends his regards too!
JUDGE: Is Bill a male or a female?
EUGENE: Ask Bill personally, please.
JUDGE: Well I'd rather talk to you. What is your name?
EUGENE: Call me Eugene. I am glad to talk to you!
JUDGE: My name is Jane and I am female. How about you? What's your gender?
EUGENE: I'm a male. A “guy’ I'd say.
JUDGE: Pleased to meet you Eugene. What's the weather like where you are?
EUGENE: Let's get on with our conversation!
JUDGE: Don't you like talking about the weather?
EUGENE: All these talks about weather is a waste of time.
JUDGE: What would you like to discuss?
EUGENE: I don't know. Better tell me more about yourself!

Passing the Turing Test

- 2014 - Royal Society (London) - Sixteenth Anniversary of Alan Turing -
- Chabot - Eugene Goostman - Pretended to be a thirteen-year-old Ukrainian boy
  - Passed the turing test for the first time
  - 10/30 Judges believed the response is from human

- *Turing predicted in 50 years time, computers can be programmed to play imitation game in which an average interrogator fails to identify the machine 70% time in a 5 mins questioning*

Thinking Humanly

Cognitive Modelling Approach

Some Definitions of AI:

“The exciting new effort to make computers think . . . machines with minds, in the full and literal sense.” (Haugeland, 1985)

“[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning . . .” (Bellman, 1978)

Thinking Rationally

"Laws of Thought" Approach

- Invention of Formal Logic, Greek Philosopher **Aristotle**, Third century BC.
- Introduced syllogisms, providing argument structures

In all boring classes, students sleep

It is a boring class

Students sleep in this class [Are you ?]

- Field of Logics gave rise to codifying rational thinking
  - When elements are '*things*', we reason about things

Hurdles to the idea : (1) Not everything can be logically coded (2) no provably correct action at a moment (3) Exhaustive computational resources

Acting Rationally

The Rational Agent Approach

- An agent is an entity that perceives and acts

  This course is about designing rational agents

- Abstractly, an agent is a function from percept histories to actions:

  [f: $P^* \rightarrow A$]

- For any given class of environments and tasks, we seek the agent (or class of agents) with the best performance
- Computational limitations make perfect rationality unachievable
- Design best program for given machine resources

Acting Rationally

The Rational Agent Approach

- Rational behaviour: doing the *right thing*
- The *right thing:* that which is expected to maximize goal achievement, given the available information
- Rational behaviour is not just about correct inference / thinking, skills needed to pass turing test etc.

(adv) : More General - Correct inference is just a thing

(adv) : More amenable for scientific developments, as the rational behaviour is better defined than human thinking and behaviour

Definitions

Thinking Humanly	Thinking Rationally
"The exciting new effort to make computers think ... *machines with minds*, in the full and literal sense." (Haugeland, 1985) "[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning ..." (Bellman, 1978)	"The study of mental faculties through the use of computational models." (Charniak and McDermott, 1985) "The study of the computations that make it possible to perceive, reason, and act." (Winston, 1992)
Acting Humanly	**Acting Rationally**
"The art of creating machines that perform functions that require intelligence when performed by people." (Kurzweil, 1990) "The study of how to make computers do things at which, at the moment, people are better." (Rich and Knight, 1991)	"Computational Intelligence is the study of the design of intelligent agents." (Poole *et al.*, 1998) "AI ... is concerned with intelligent behavior in artifacts." (Nilsson, 1998)

Traveller’s Problem

Destination - Fixed **Goal**

Source

Traveller's Problem

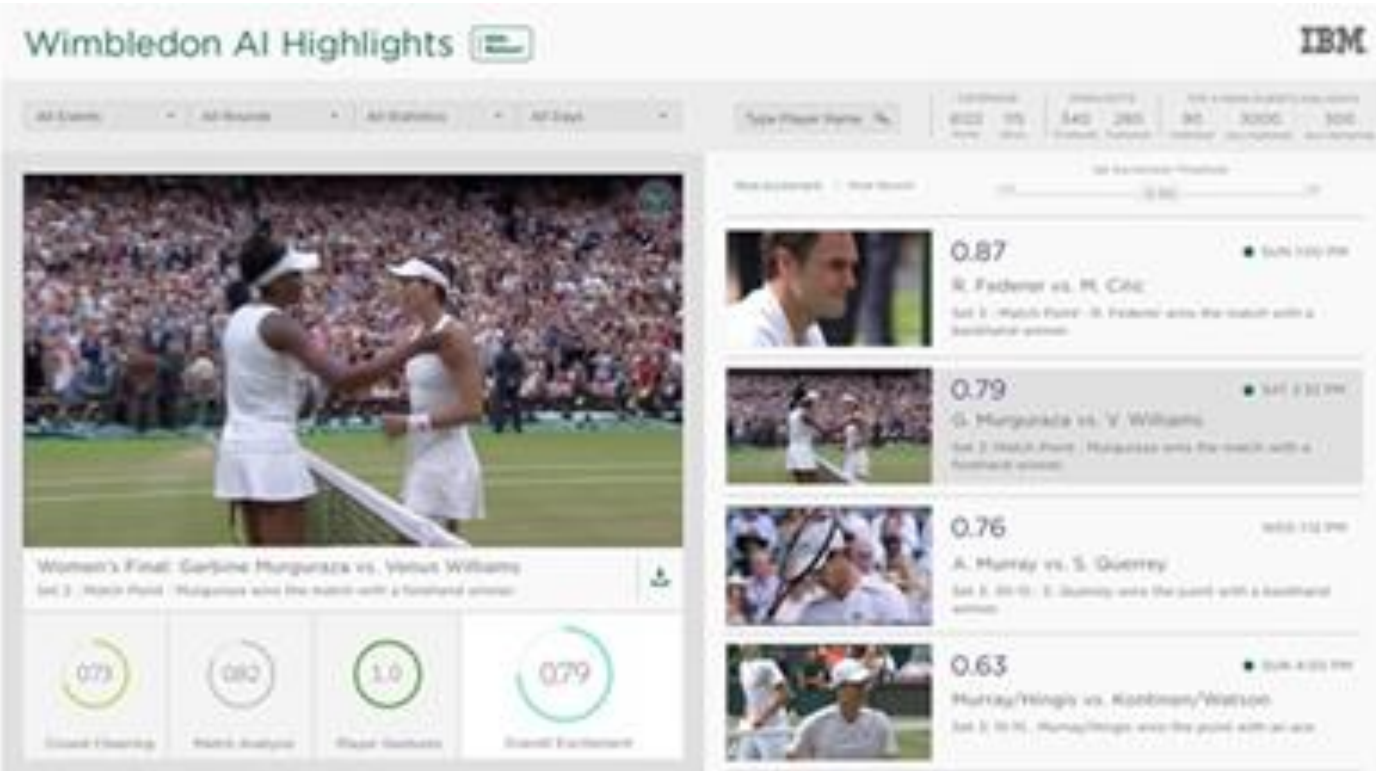
AI in HealthCare

Lyrebird's Project Re-Voice

AI in Culinary Field

Spyce

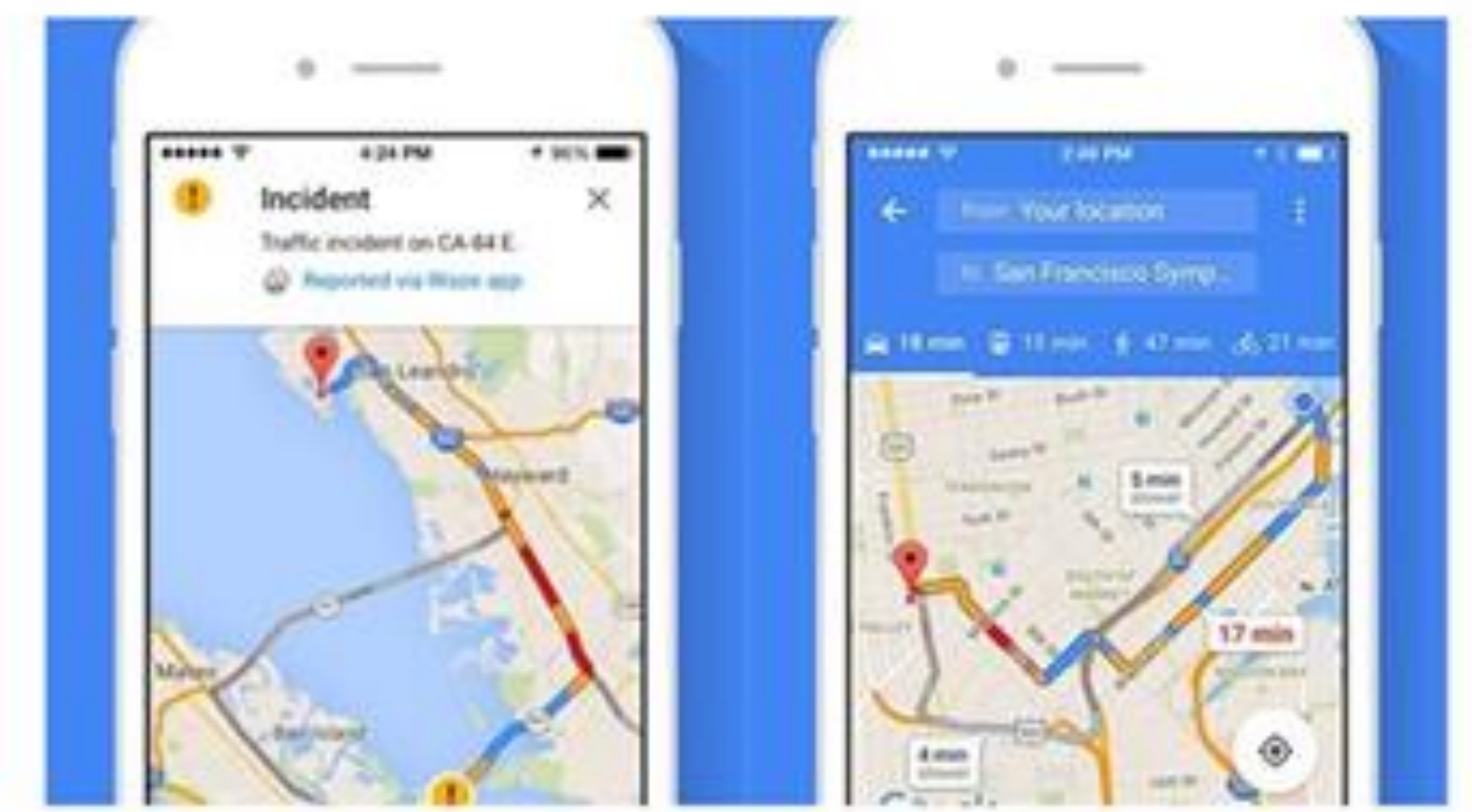
AI in NLS
IBM Watson



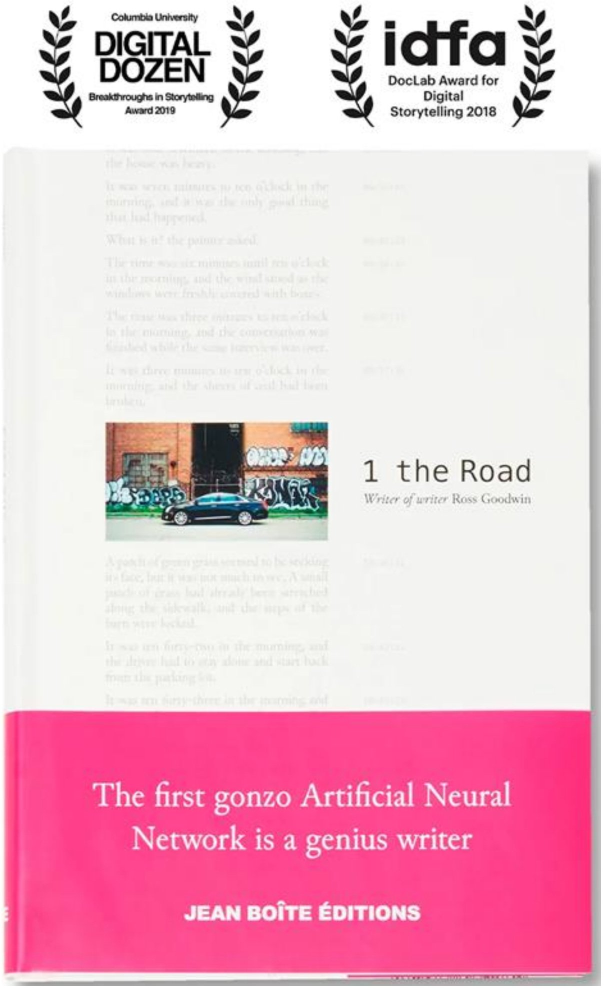
Computer Vision
NLP
ML
Speech Recognition
Automation

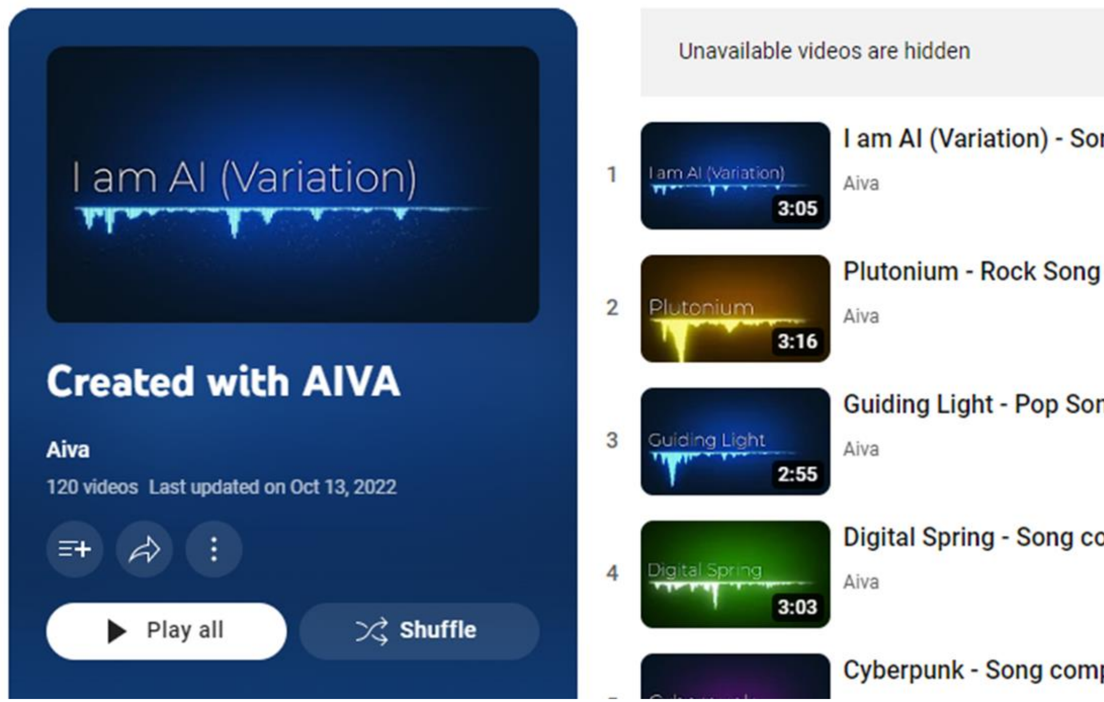
AI in Transportation

AI in HCI
Google Map Navigation Assistant



AI in Literacy & Music





Application Domain

(Additional Notes added from the textbook for self read)

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

- Can formal rules be used to draw valid conclusions?
- How does the mind arise from a physical brain?
- Where does knowledge come from?
- How does knowledge lead to action?

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

Aristotle (384–322 B . C .) : first to formulate precise set of laws to govern rational part of brain

Ramon Lull (d. 1315) : useful reasoning could actually be carried out by a mechanical artifact

Hobbes (1588–1679) : "we add

and subtract in our silent thoughts."

Leibniz (1646–1716) : Built a mechanical device intended to carry out operations on concepts rather than numbers

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

Some *'isms* on the working of minds :

Rationalism - Correct Reasonings (Aristotle, Descartes …)

Dualism - A part of the human mind (or soul or spirit) that is outside of nature

Materialism - Alternative to dualism - holds

that the brain's operation according to the laws of physics constitutes the mind

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

Obtaining Knowledge

David Hume's (1711–1776) : First principles of induction

Logical positivism- Rudolf Carnap : Every knowledge obtained has a logical connection

Carnap (1905–1997) : A book "*The Logical Structure of the World*" (1928) defined an explicit computational procedure for extracting knowledge from elementary experiences

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

Connection between knowledge and action:

Aristotle - (in *De Motu Animalium*) that actions are justified by a logical connection between goals and knowledge of the action's outcome

```
I need covering;
a cloak is a covering.
I need a cloak.
What I need, I have to make;
I need a cloak.
I have to make a cloak.
And the conclusion, "I have to make a cloak" is an action
```

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

- What are the formal rules to draw valid conclusions?
- What can be computed?
- How do we reason with uncertain information?

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

- What are the formal rules to draw valid conclusions?

George Boole (1815–1864) : Propositional Logic

Gottlob Frege (1848–1925): First order logic

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

- What can be computed?

Kurt Gödel (1906–1978) : In any formal theory as strong as Peano arithmetic #(the elementary theory of natural numbers), there are true statements that are undecidable in the sense that they have no proof within the theory

Computability, tractability, NP-completeness

Probability theory & inference mechanisms

- https://en.wikipedia.org/wiki/Peano_axioms

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

- How should we make decisions so as to maximize payoff?

Utility / preferred outcomes

Decision theory -Probability & utility theory

Game theory

- How to make decisions when payoffs are not immediate?
  - MDP

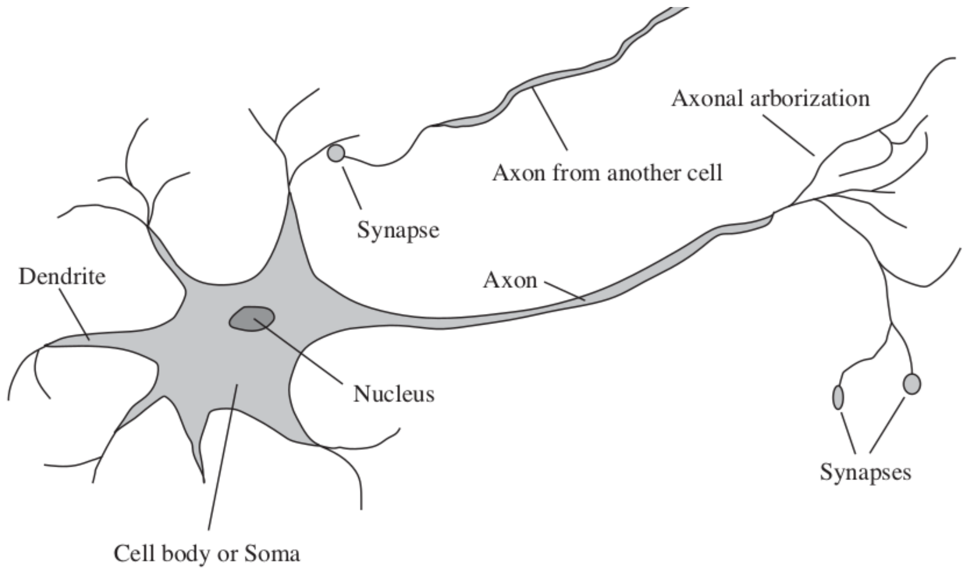
Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

How do brains process information?

- Study of the nervous system / brain
- How does brain enables thoughts - Mystery Still

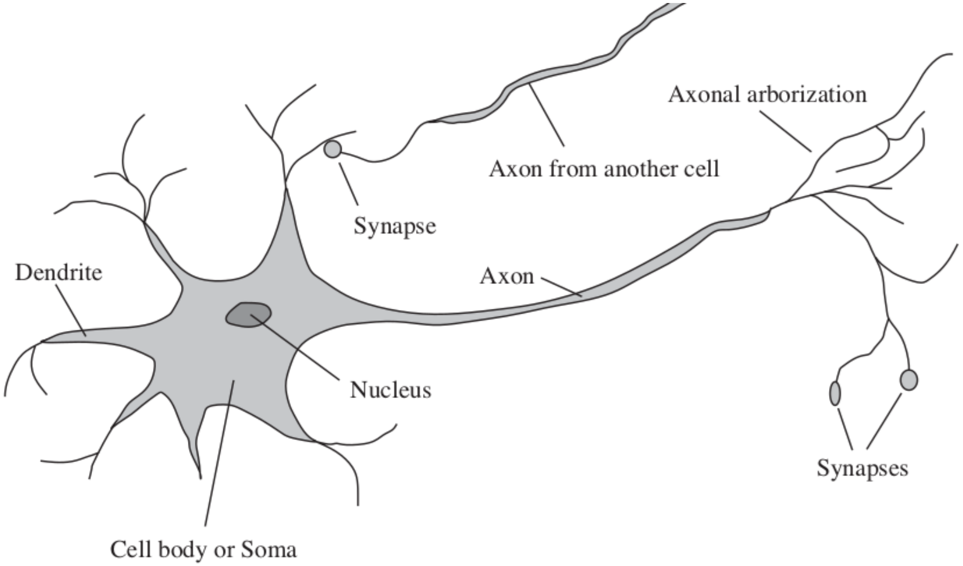
Aristotle , "*Of all the animals, man has the largest brain in proportion to his size*"



Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

	Supercomputer	Personal Computer	Human Brain
Computational units	10^4 CPUs, 10^{12} transistors	4 CPUs, 10^9 transistors	10^{11} neurons
Storage units	10^{14} bits RAM	10^{11} bits RAM	10^{11} neurons
	10^{15} bits disk	10^{13} bits disk	10^{14} synapses
Cycle time	10^{-9} sec	10^{-9} sec	10^{-3} sec
Operations/sec	10^{15}	10^{10}	10^{17}
Memory updates/sec	10^{14}	10^{10}	10^{14}



Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

How do humans and animals think and act?

- *Cognitive Psychology* - Brain as an information-processing device
- Two months after the dartmouth workshop, a workshop in MIT gave birth to *Cognitive Science*
    - George Miller, Noam Chomsky, Allen Newell and Herbert Simon - roles of computer models to address the psychology of memory, language, and logical thinking, issues..

"*a cognitive theory should be like a computer program*" (Anderson, 1980);

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

Computers & Programming Languages

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

Control theory

- Deals with the behaviour of dynamic systems
  - behaviour must ensure the error between the current state and goal state is minimized
- Cybernetics - Book by Wiener
  - (Norbert Wiener, 1948) : Scientific study of control and communication in the animal and the machine
- Ashby's Design for a Brain (1948, 1952):
  - Intelligence could be created by the use of homeostatic devices containing appropriate feedback loops to achieve stable adaptive behavior
  - Led to the idea of *design of systems that maximize an objective function over time*

Areas Contributing to AI

Philosophy
Mathematics
Economics
Neuroscience
Psychology
Computer Engineering
Control theory, Cybernetics
Linguistics

How does language relate to thought?

Verbal Behavior (1957, B. F. Skinner) :

- Behaviorist approach to language learning
- Reviewed by Noam Chomsky
  - criticised lack of notion of creativity in language

Syntactic Structures (1957, Noam Chomsky)

- Computational linguistics / natural language processing as a part of AI
  - Understanding a language is realized as more complex than ever
  - Context, subject matter knowledge complicated it further
  - Representing language consumed volume of work done in NLP, in early times

Course Outline

- **In this course, you will learn :**
  - a solid foundation for designing intelligent agents
  - to represent and use the knowledge learnt for inferencing
  - to model agents operating in uncertain environments
  - optimization models of computation and processing in real world application
- **Modules :**
  - Problem Solving Agent using Search
  - Game Playing
  - Probabilistic Representation and Reasoning
  - Reasoning over time

Required Reading: AIMA - Chapter # 1

AIMA is the first prescribed text book

Thank You for your active participation

Note : Some of the slides are adopted from AIMA TB materials

Next Class Plan

- Agent Design
- Environment
- Agent Architecture
- Problem Solving Agent Formulation

Artificial & Computational Intelligence

AIML CLZG557

M1 : Introduction
&
M2 : Problem Solving Agent using Search

Raja vadhana P

Assistant Professor,

BITS - CSIS

BITS Pilani

Pilani Campus

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

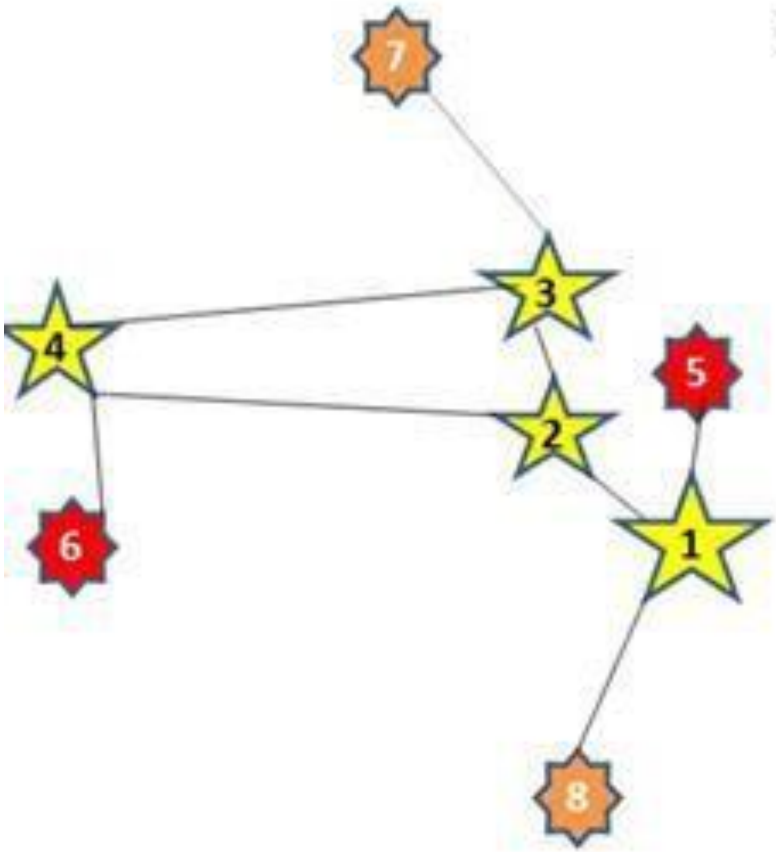
M5 Probabilistic Representation and Reasoning

M6 Reasoning over time

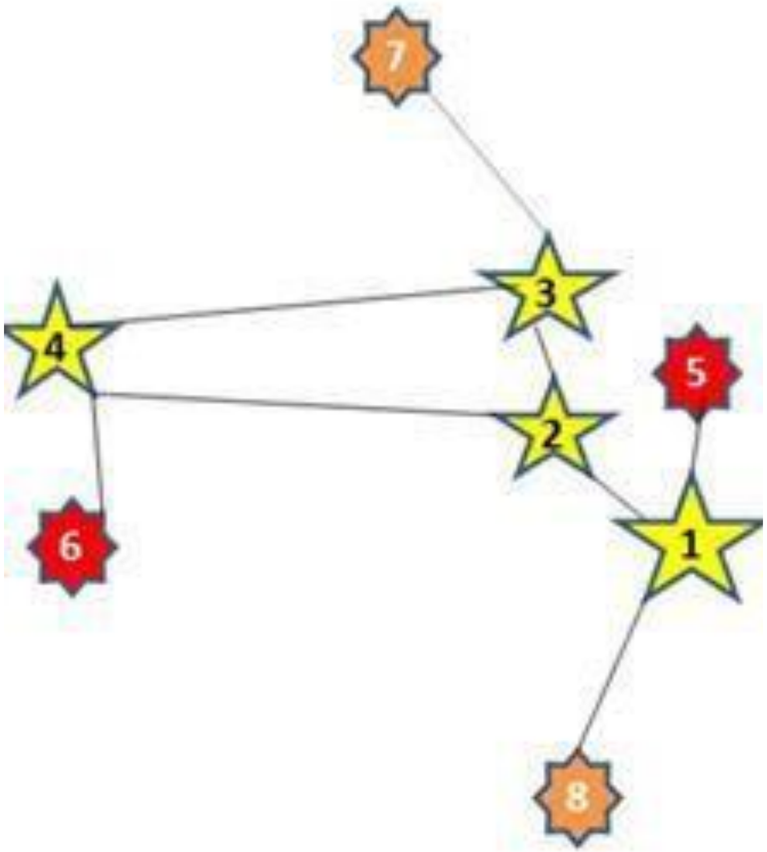
M7 Ethics in AI

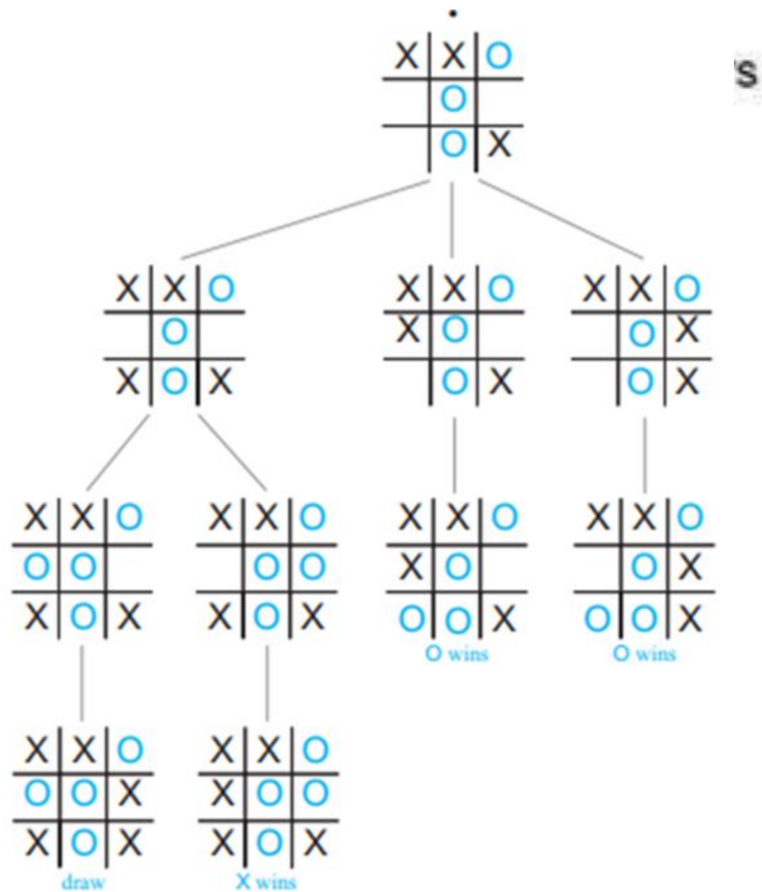
Traveller’s Problem

Traveller’s Problem



Traveller’s Problem





Rational Agents

Rational Agent

Design Principles & Techniques

	Thought / Reasoning	Acting
Human Performance	THINKING HUMANLY "[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning, … " (Bellman, 1978)	ACTING HUMANLY "The art of creating machines that perform functions that require intelligence when performed by people" (Kurzweil, 1990)
Rational Performance	THINKING RATIONALLY "The study of computations that make it possible to perceive, reason, and act" (Winston, 1992)	ACTING RATIONALLY "Computational intelligence is the study of the design of intelligent agents" (Poole et al., 1998)

Acting Rationally

The Rational Agent Approach

- An agent is an entity that perceives and acts

This course is about designing rational agents

- Abstractly, an agent is a function from percept histories to actions: [f: $P^* \rightarrow A$]
- For any given class of environments and tasks, we seek the agent (or class of agents) with the best performance
- Computational limitations make perfect rationality unachievable
- Design best program for given machine resources

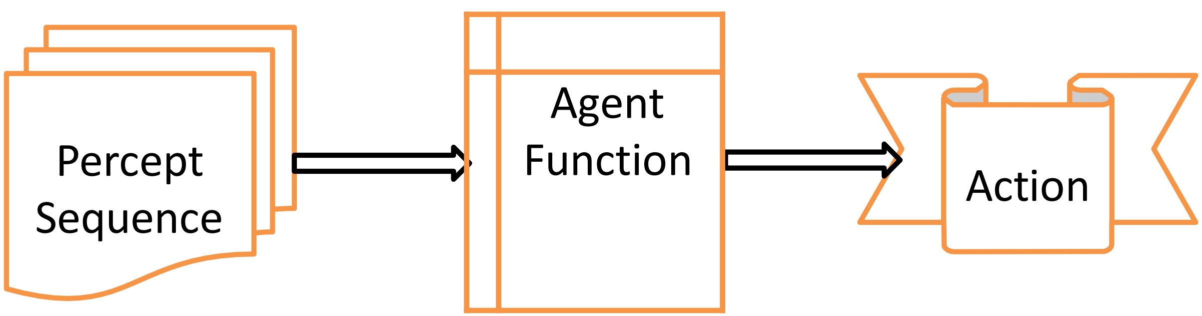
Properties of Rational Agent

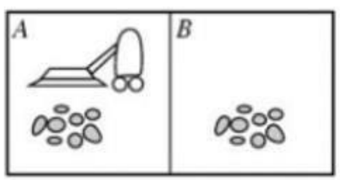
- Omniscience : Expected Vs Actual Performance
- Learning Capability : Apriori Knowledge
- Autonomous in decision making: An agent is autonomous if its behaviour is determined by its own experience (with ability to learn and adapt)

Intelligent Agent

Rational Agent is one that acts to achieve the best outcome or the best expected outcome even under uncertainty

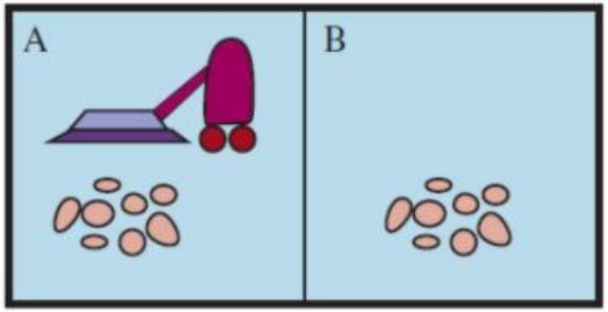
Maps / Tabulated / Programmed





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
...	...

Intelligent Agent



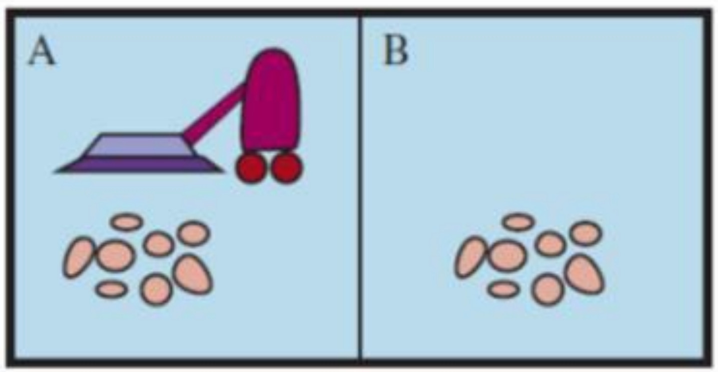
- Percepts: location and contents, e.g., [A , Dirty]
- Actions: *Left*, *Right*, *Suck*, *NoOp*

Performance measure: An objective criterion for success of an agent's behaviour

E.g., performance measure of a vacuum-cleaner agent
» amount of dirt cleaned up
» amount of time taken
» amount of electricity consumed
» amount of noise generated, etc.

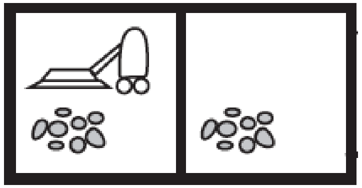
[PEAS Design]

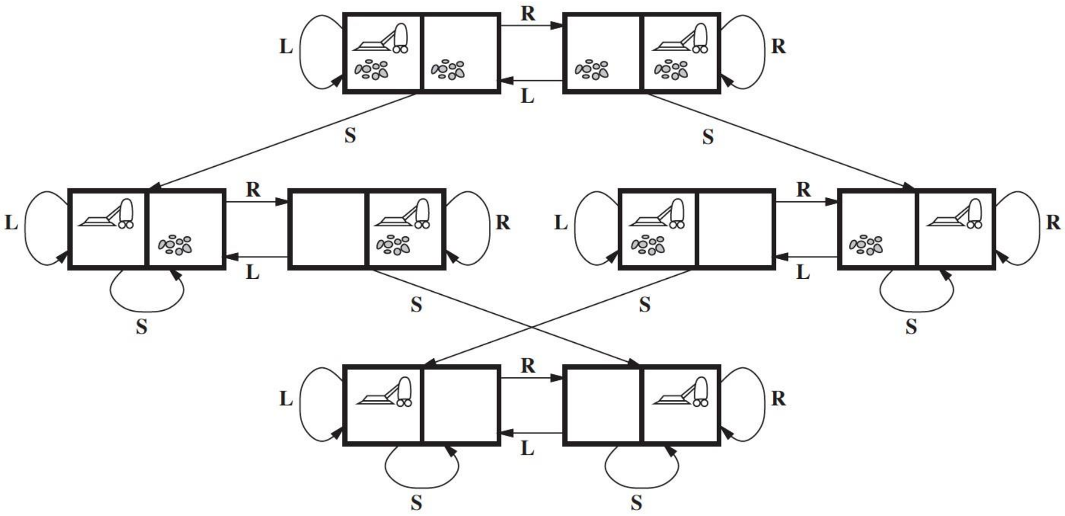
Intelligent Agent



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*
⋮	⋮

Vacuum World Problem



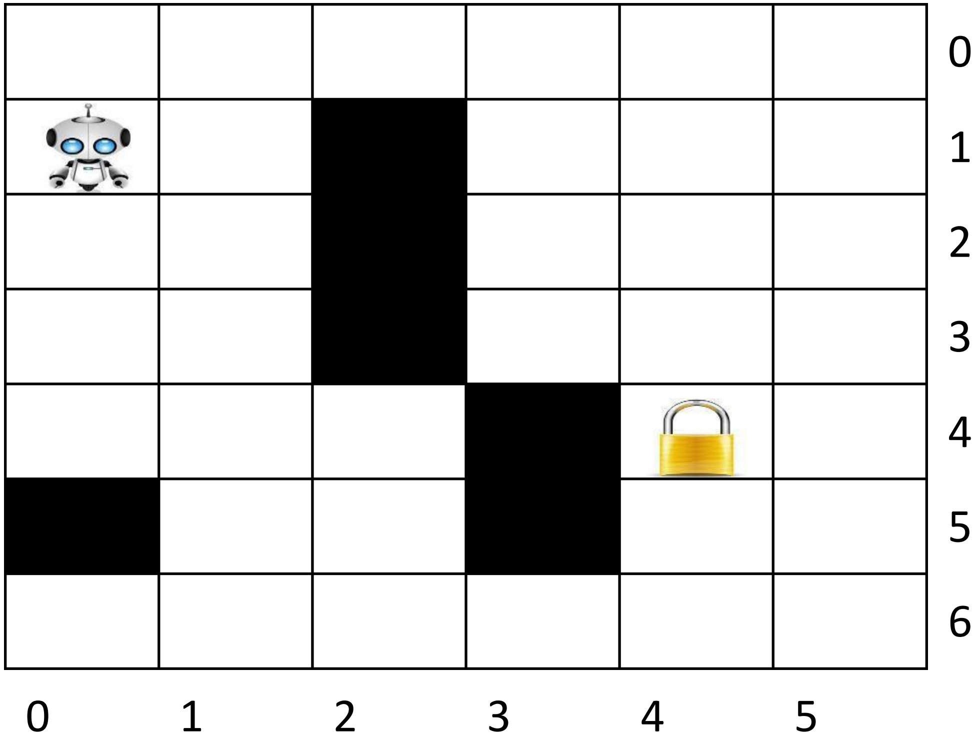


PEAS Environment

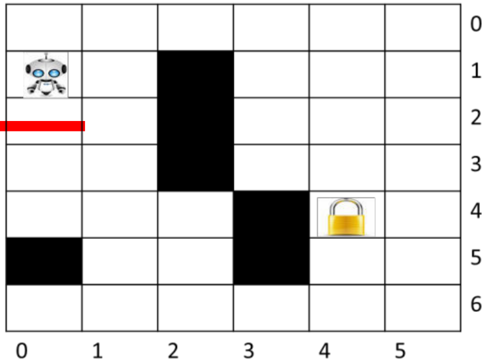
Design on what an application wants the agent to do in the environment

Agent	Performance	Environment	Sensors	Actuators
Medical diagnosis system	Healthy patient, reduced costs	Patient, hospital, staff	Keyboard entry of symptoms, findings, patient's answers	Display of questions, tests, diagnosis, treatments, referrals
Satellite Image analysis system	Correct image categorization	Downlink from orbiting satellite	Color pixel analysis	Display of scene categorization
Interactive English tutor	Student's score on test	Set of students, testing agency	Keyboard entry	Display of exercises, suggestions, corrections

Path finding Robot - Lab Example



PEAS Environment



Agent

Performance

Environment

Sensors

Dimensions of Task Environment

Sensor Based:

- Observability : Full Vs Partial

Action Based:

- Dependency : Episodic Vs Sequential

State Based:

- No.of.State : Discrete Vs Continuous

Agent Based:

> Cardinality : Single Vs MultiAgent

Action & State Based:

- State Determinism : Deterministic Vs Stochastic | Strategic
- Change in Time : Static Vs Dynamic

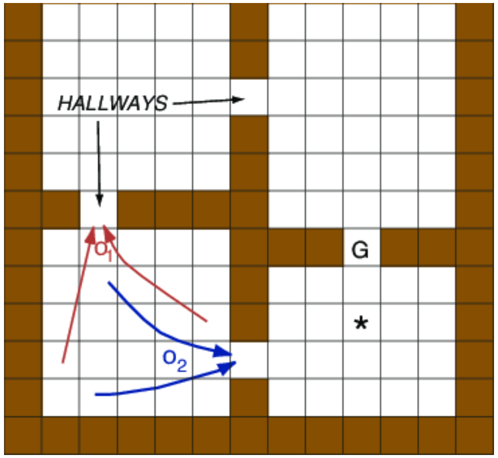
Task Environment

A rational agent is built to solve a specific task. Each such task would then have a
different environment which we refer to as Task Environment

Based on the applicability of each technique for agent implementation its task environment design is determined by multiple dimension

Sensor Based:

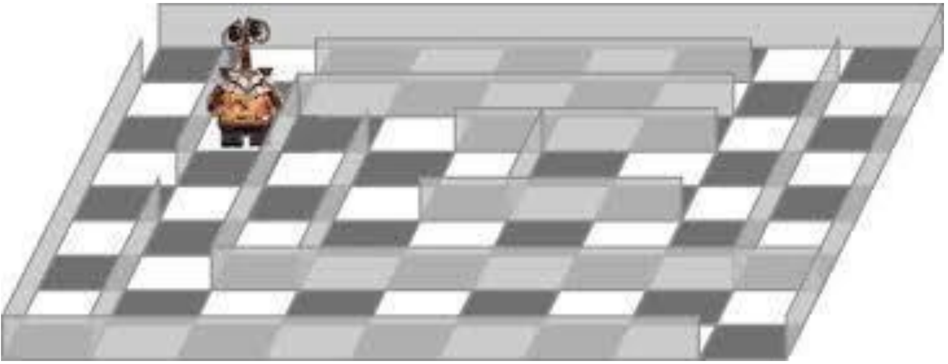
➢ Observability : Full Vs Partial



Task Environment

Action Based:

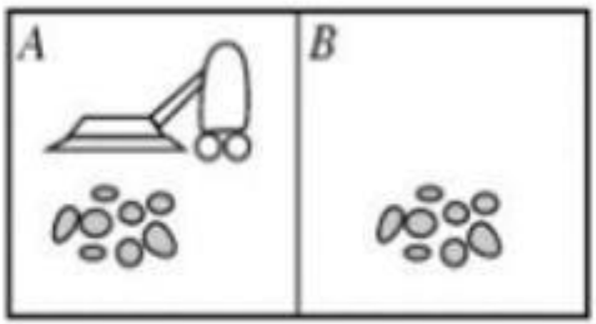
- Dependency : Episodic Vs Sequential

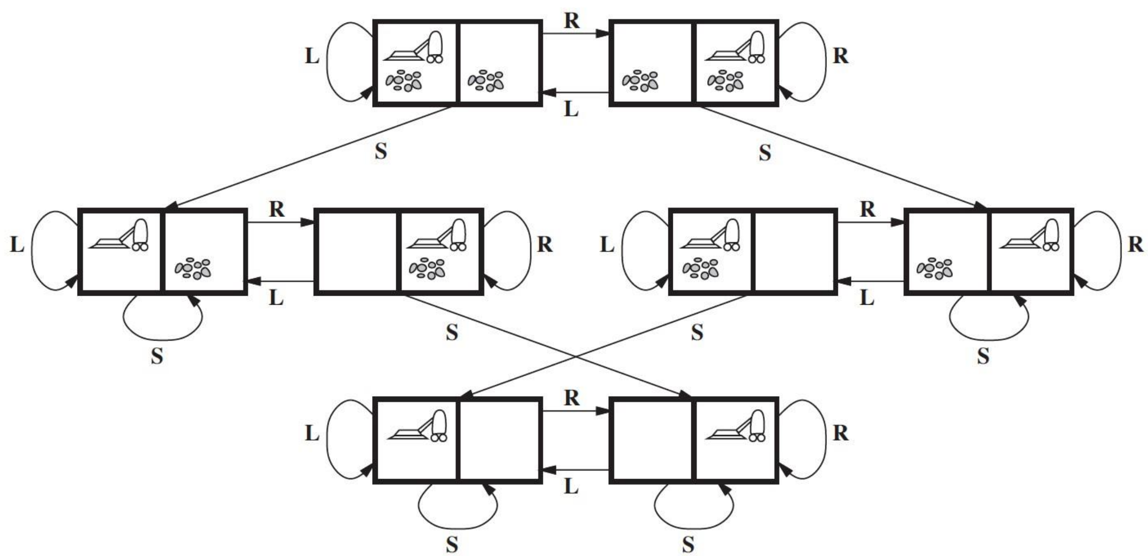


Task Environment

State Based:

- No.of.State : **Discrete** Vs Continuous





Task Environment

State Based:

- No.of.State : Discrete Vs Continuous

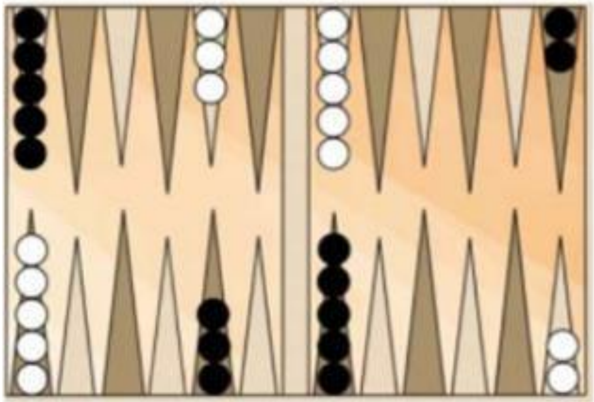
VS.

Task Environment

Action & State Based:

- State Determinism : Deterministic Vs Stochastic | Strategic

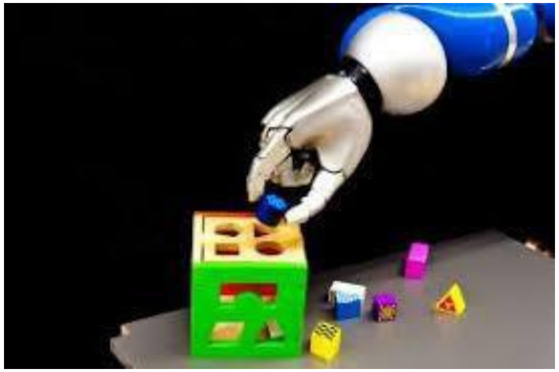
(If the environment is deterministic except for the actions of other agents, then the environment is strategic)



Task Environment

Agent Based:

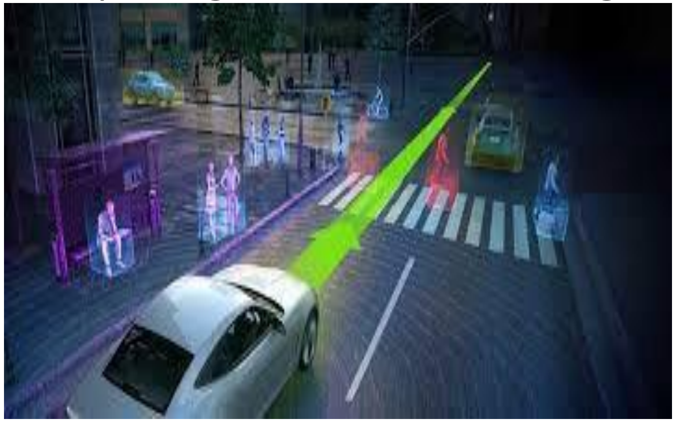
> Cardinality : Single Vs MultiAgent

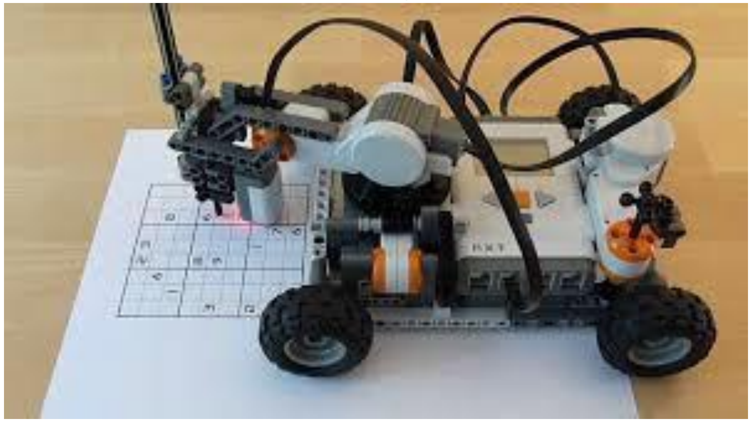


Task Environment

Action & State Based:

- Change in Time : Static Vs Dynamic
- (The environment is semi dynamic if the environment itself does not change with the passage of time but the agent's performance score does)

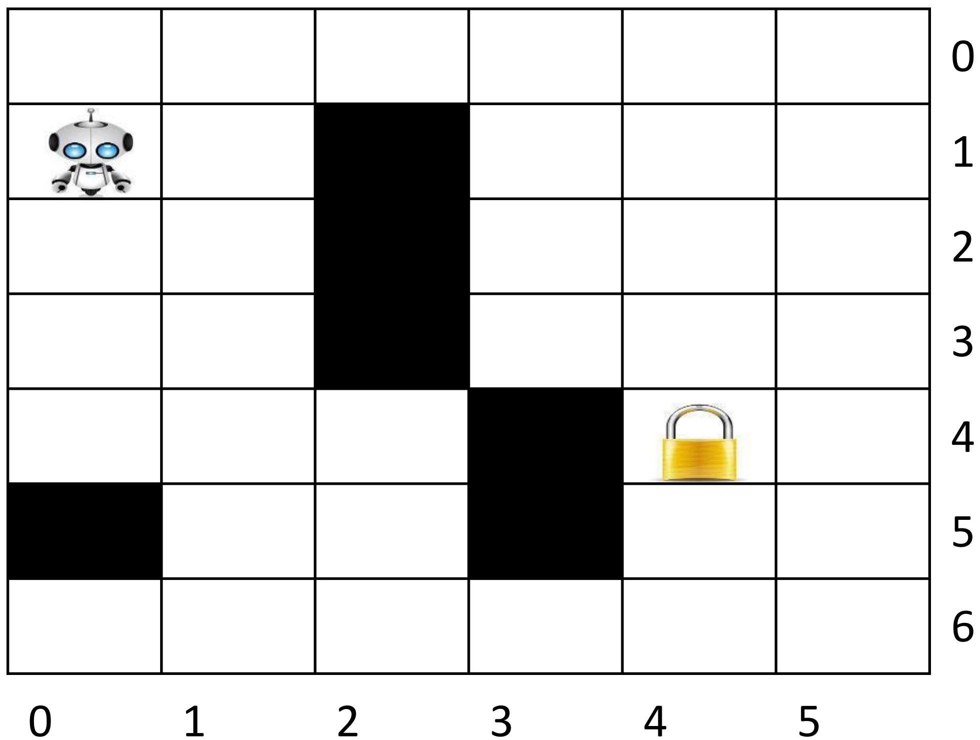




Task Environment

Task Environment	Fully vs Partially Observable	Single vs Multi-Agent	Deterministic vs Stochastic	Episodic vs Sequential	Static vs Dynamic	Discrete vs Continuous
Medical diagnosis system	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Satellite Image Analysis System	Fully	Single	Deterministic	Episodic	Static	Continuous
Interactive English tutor	Partially	Multi	Stochastic	Sequential	Dynamic	Discrete

Path finding Robot - Lab Example



Agent

Observability

No.of.Agents

No.of.States

Determinism

Dynamicity

Output Dependency

Learning Objective Achieved

At the end of this class , students Should be able to:

1. Identify the requirement for AI solutions for given problem
2. Understand the significance of State based representations
3. Design the PEAS (Performance, Environment, Actuators, Sensors) for given problem
4. Identify dimensions of TASK environment

Next Class Plan

Structure of Agents-Architectures

Problem Solving Agents

Problem Formulation

Uninformed Search Algorithms

Required Reading: AIMA - Chapter #2

Note : Some of the slides are adopted from AIMA TB materials

Thank You for all your Attention

Artificial & Computational Intelligence

AIML CLZG557

M1 : Introduction
&
M2 : Problem Solving Agent using Search

Raja vadhana P

Assistant Professor,

BITS - CSIS

BITS Pilani

Pilani Campus

Course Plan

M1	Introduction to AI
M2	Problem Solving Agent using Search
M3	Game Playing
M4	Knowledge Representation using Logics
M5	Probabilistic Representation and Reasoning
M6	Reasoning over time
M7	Ethics in AI

Learning Objective

At the end of this class , students Should be able to:

1. Design problem solving agents
2. Create search tree for given problem
3. Apply uninformed search algorithms to the given problem
4. Compare performance of given algorithms in terms of completeness, optimality, time and space complexity
5. Differentiate for which scenario appropriate uninformed search technique is suitable and justify

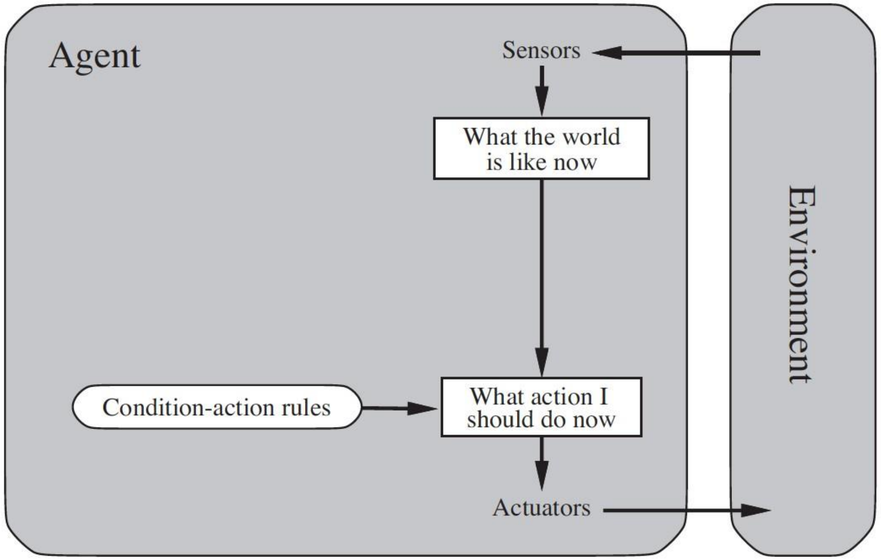
innovate achieve lead

Agents Architectures

Agent Architectures

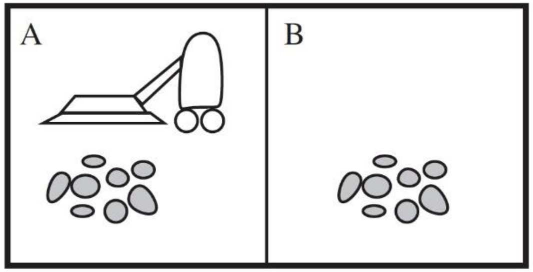
Reflex Agent

function SIMPLE-REFLEX-AGENT(*percept*) returns an action
    persistent: *rules*, a set of condition–action rules
    state←INTERPRET-INPUT(*percept*)
    rule←RULE-MATCH(*state, rules*)
    action ←*rule*.ACTION
    return *action*



function REFLEX-VACUUM-AGENT([*location,status*]) **returns** an action
    if *status* = *Dirty* **then return** Suck
    else if *location* = *A* **then return** Right
    else if *location* = *B* **then return** Left

Simple Reflex Agents

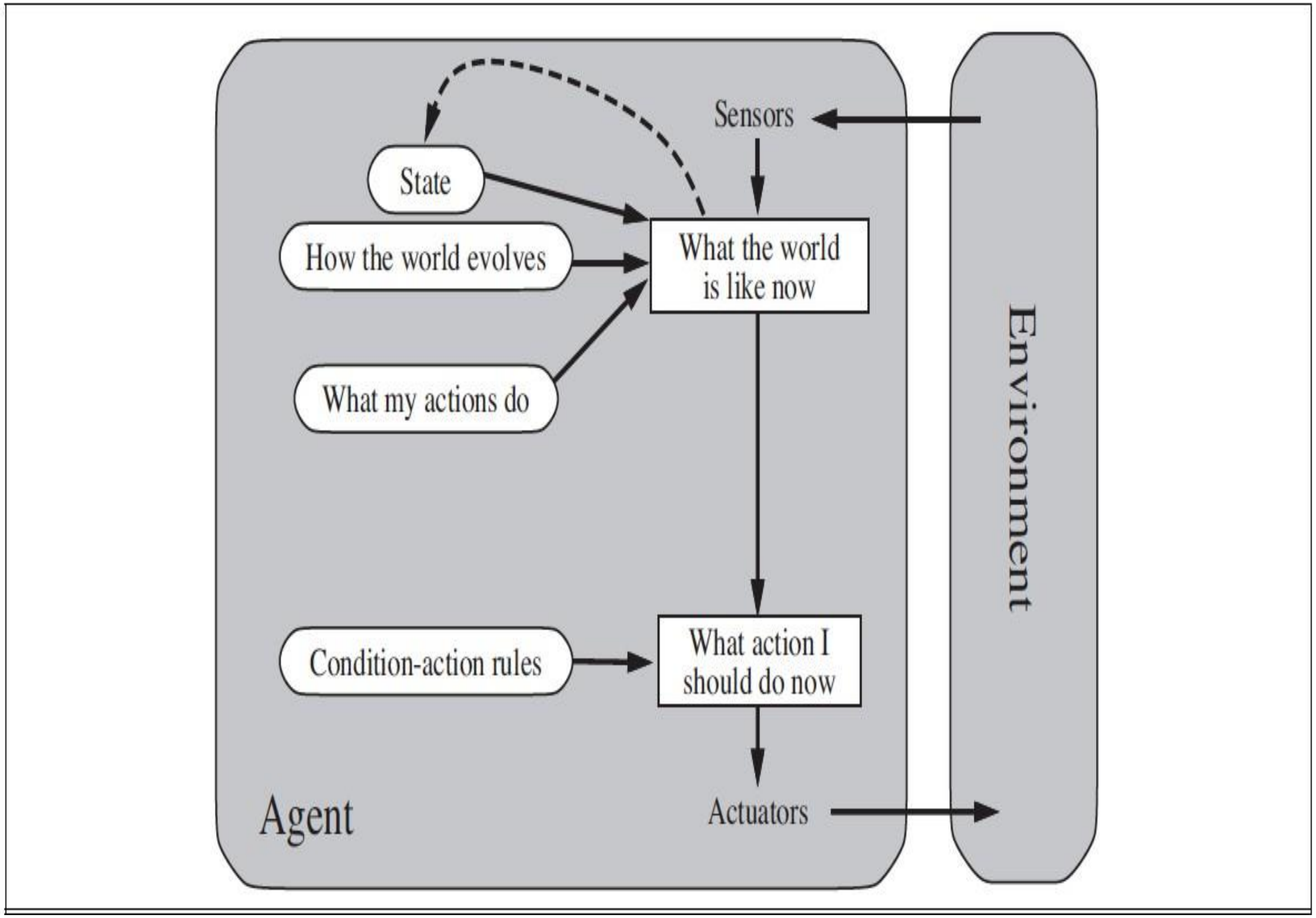


Agent Architectures

Model based Agent

Simple Reflex Agents

Model Based Agents



Model based Agent

function MODEL-BASED-REFLEX-AGENT(*percept*) **returns** an action

persistent: *state*, the agent's current conception of the world state
transition model, a description of how the next state depends on the current state and action
sensor model , a description of how the current world state is reflected in the agent's percepts
rules, a set of condition–action rules
action, the most recent action, initially none

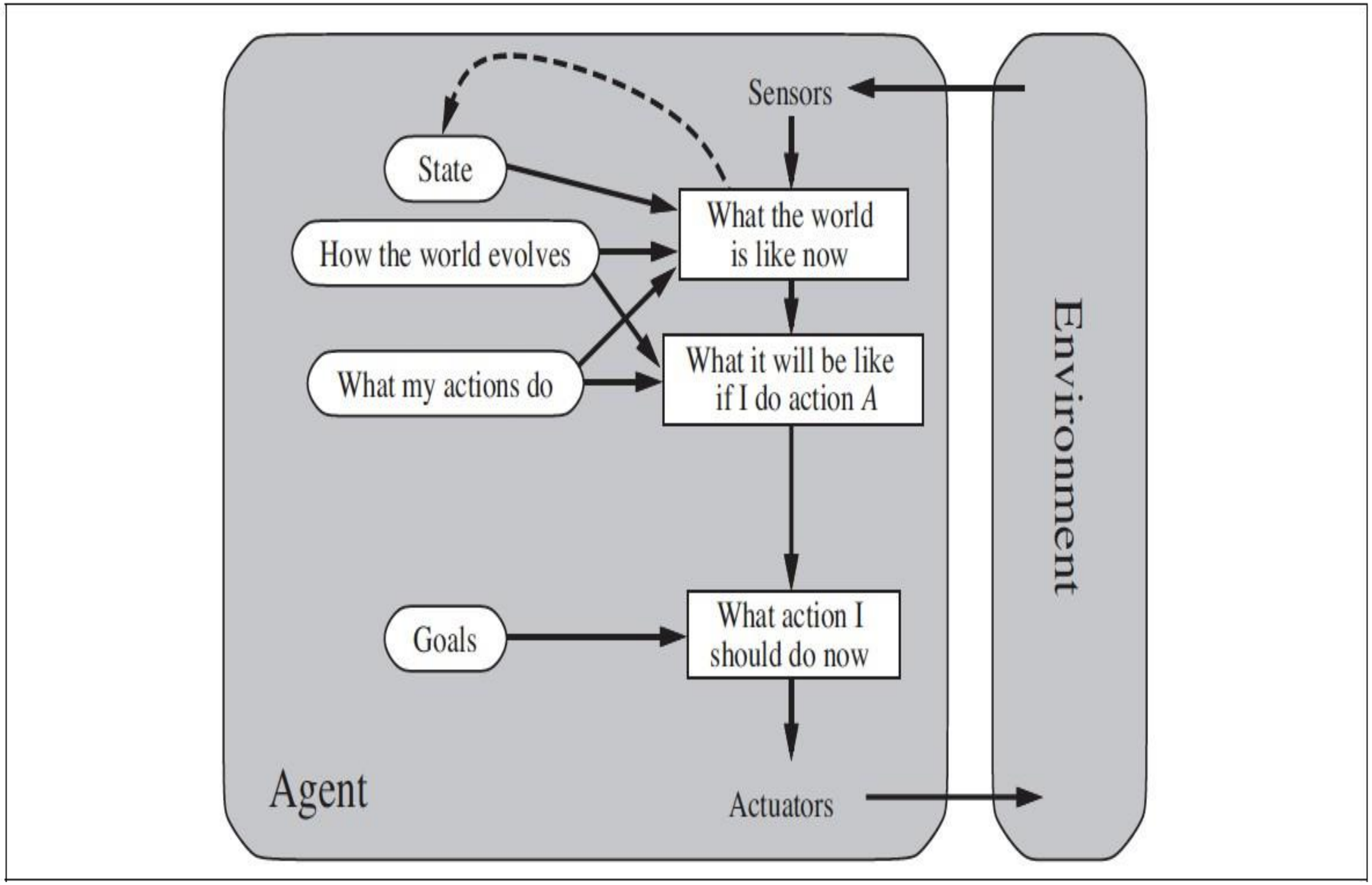
state←UPDATE-STATE(*state, action, percept, transition model, sensor model*)
rule←RULE-MATCH(*state, rules*)
action ←*rule*.ACTION
return *action*

Agent Architectures

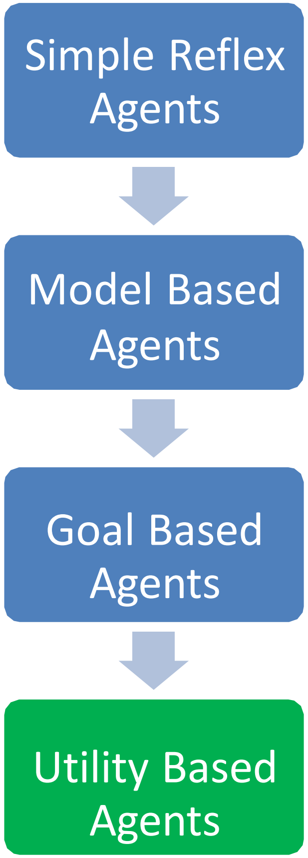
Simple Reflex Agents

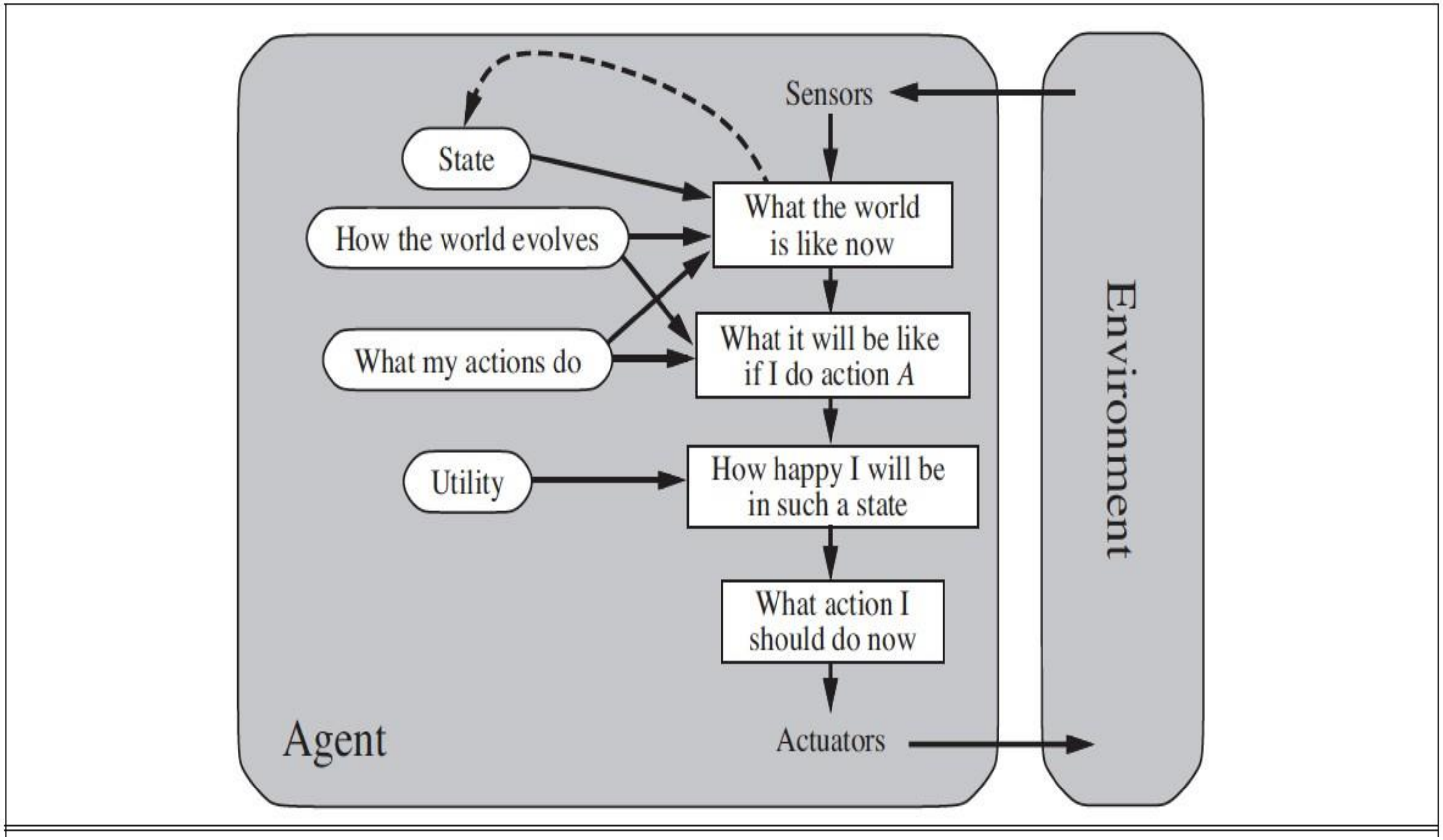
Model Based Agents

Goal Based Agents



Agent Architectures





Agent Architectures

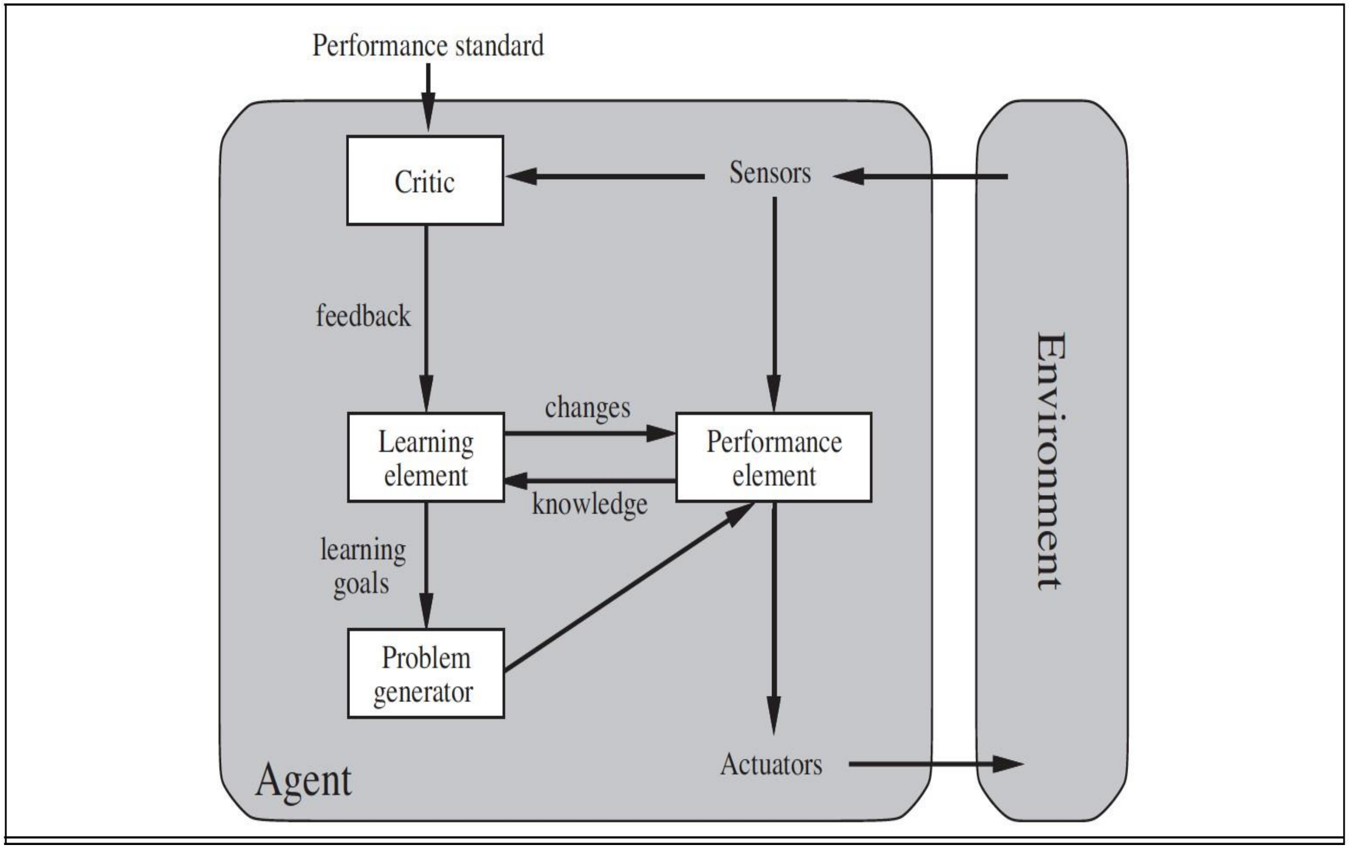
Simple Reflex Agents

Model Based Agents

Goal Based Agents

Utility Based Agents

Learning Agents



Role of Learning

Agents that improve their performance by learning from their own experiences

Input Percept

Output Action

Performance Element

APPLY BRAKE

Resulting State

Feedback

Critic

-2

Role of Learning

Input Percept

Possible Actions

Selected Action

Brake
Change Gear to Lower
Change Gear to Higher
Accelerate
Steer left
Steer right

Random

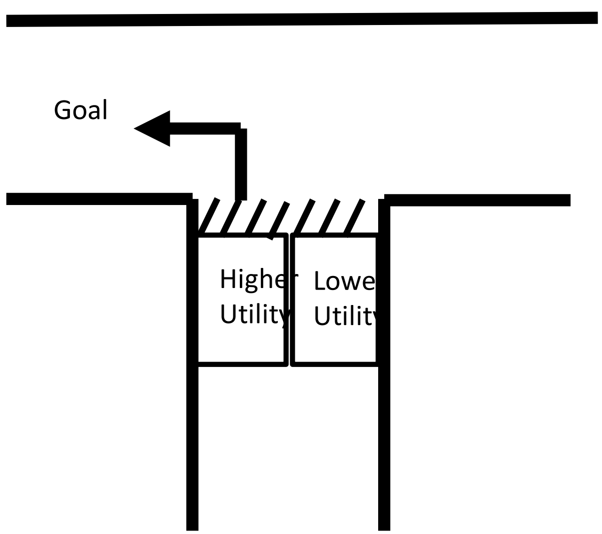
Change Gear to Lower

Role of Learning

Performance Element – Takes decision on action based on percept

$$f(red\ signal, \quad distance) = 15k\ N\ brake$$
$$distance = f'(percept\ sequence)$$
$$f(percepts, distance, raining)$$

- $f(state_0, actionA) = 0.83,$
- $f(state_0, actionB) = 0.45$

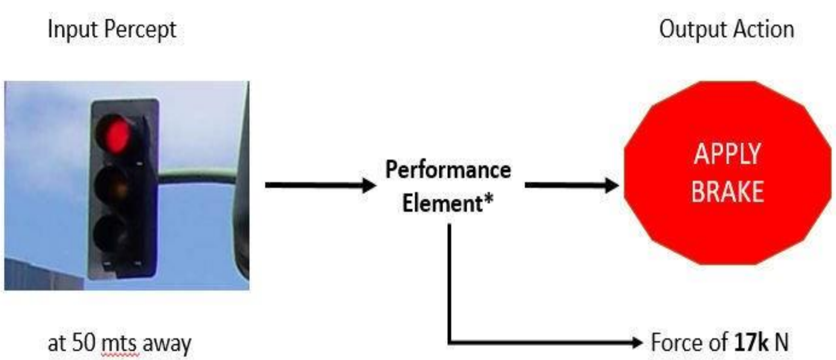


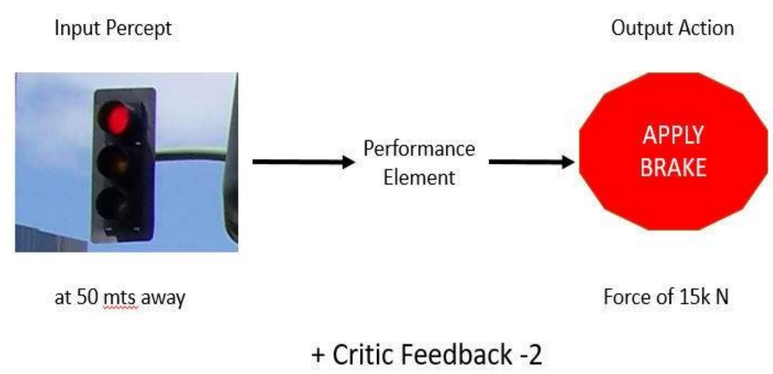
Role of Learning

Critic – Provides feedback on the actions taken

Learning :

Supervised Vs Unsupervised Vs Reinforcement

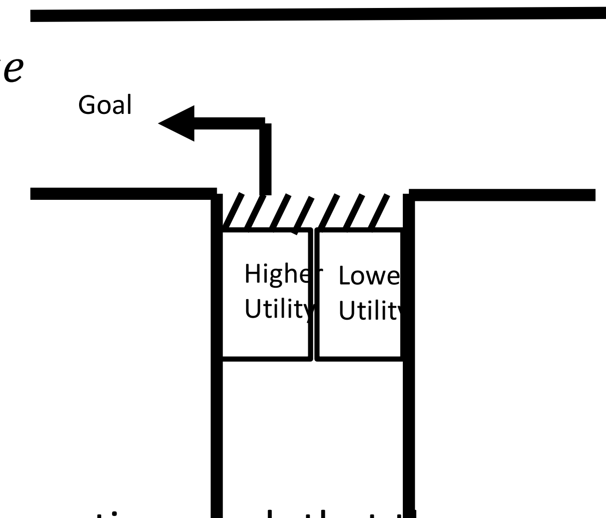




Role of Learning

<u>Performance Element</u> – Takes decision on action based on percept

$$f(red\ signal,\quad distance) = 15k\ N\ brake$$
$$distance = f'(percept\ sequence)$$
$$f(percepts, distance, raining)$$



- $f(state_0, actionA) = 0.83,$
- $f(state_0, actionB) = 0.45$

<u>Learning Element</u> – Make the performance element select better actions such that the utility function is optimized

<u>Critic</u> – Provides feedback on the actions taken

<u>Problem Generator</u> – Make the Performance Element select sub-optimal actions such that you would learn from unseen actions

Problem Formulation

Problem Solving Agents

Goal based decision making agents finds sequence of actions that leads to the desirable state.

Phases of Solution Search by PSA

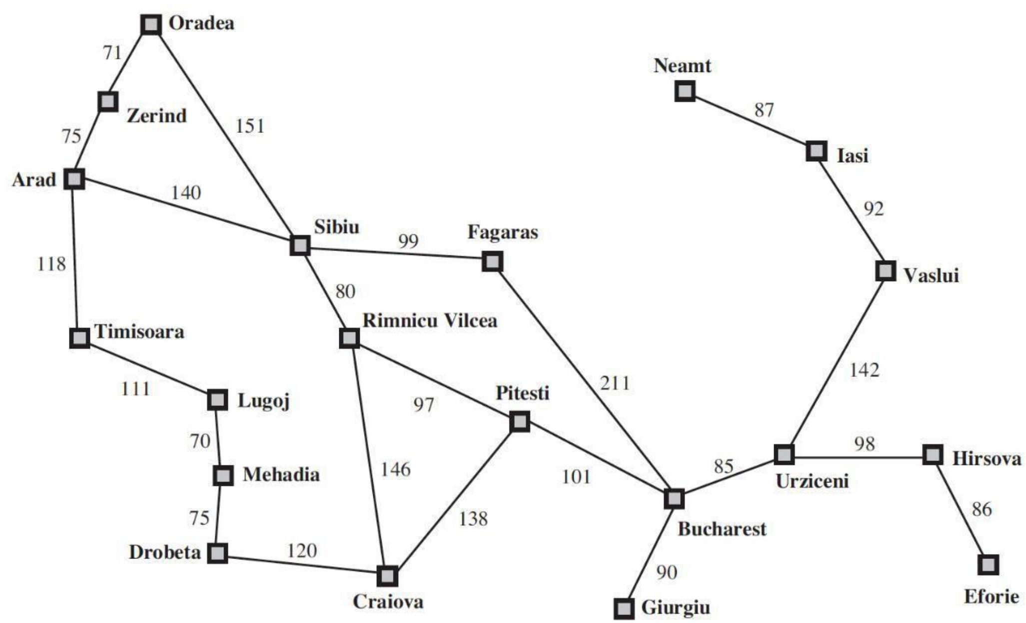
Goal Formulation

Problem Formulation

Search Phase

Execution Phase

Optimizes the Objective (Local | Global) Limits the Actions



Problem Solving Agents

Goal Formulation

Problem Formulation

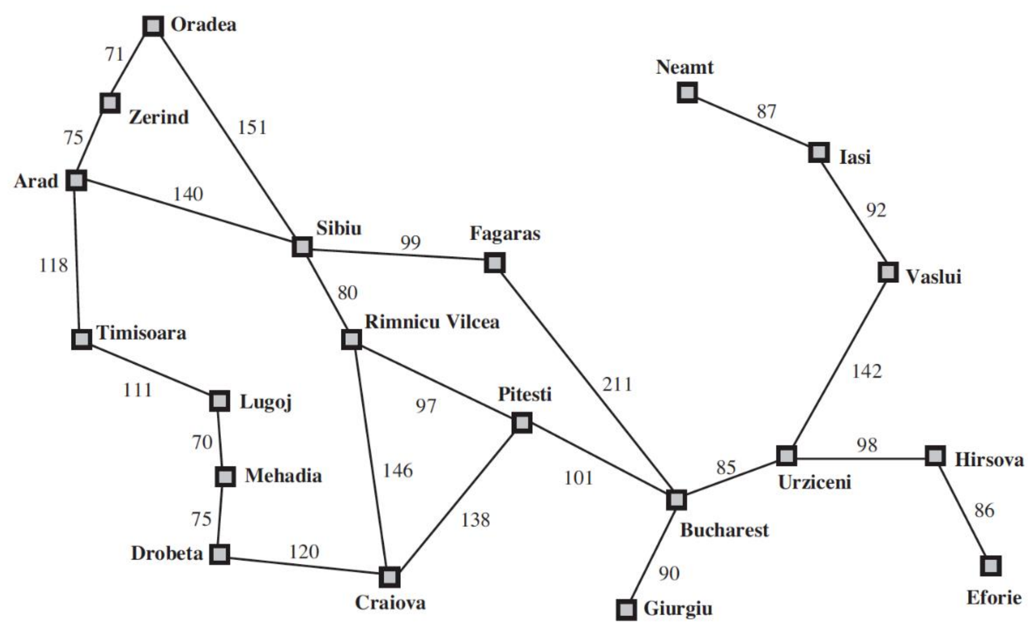
Search Phase

Execution Phase

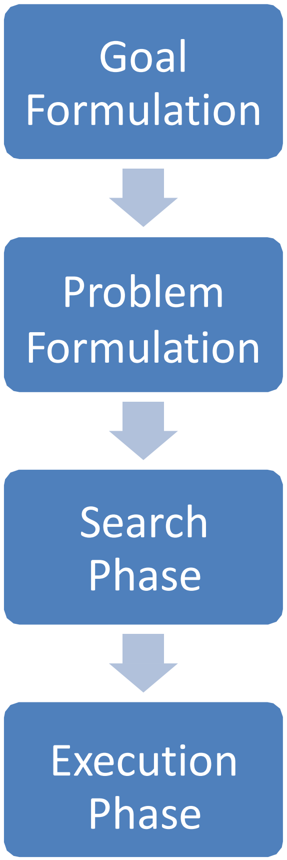
Phases of Solution Search by PSA

State Space Creations [in the path of Goal]

Lists the Actions

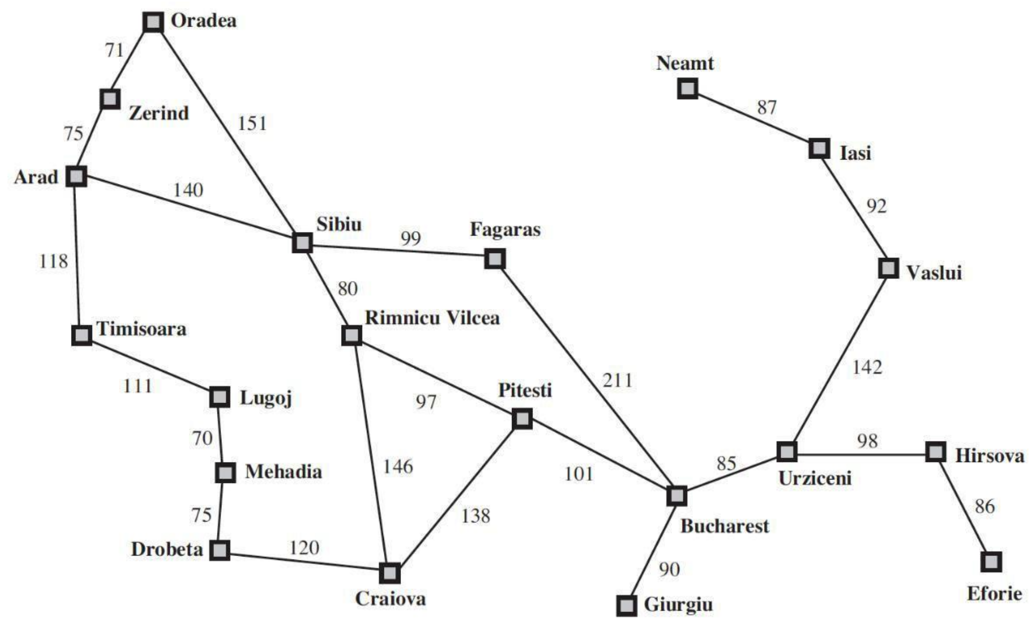


Problem Solving Agents



Phases of Solution Search by PSA

Assumptions – Environment :
Static
Observable Discrete
Deterministic



Problem Solving Agents

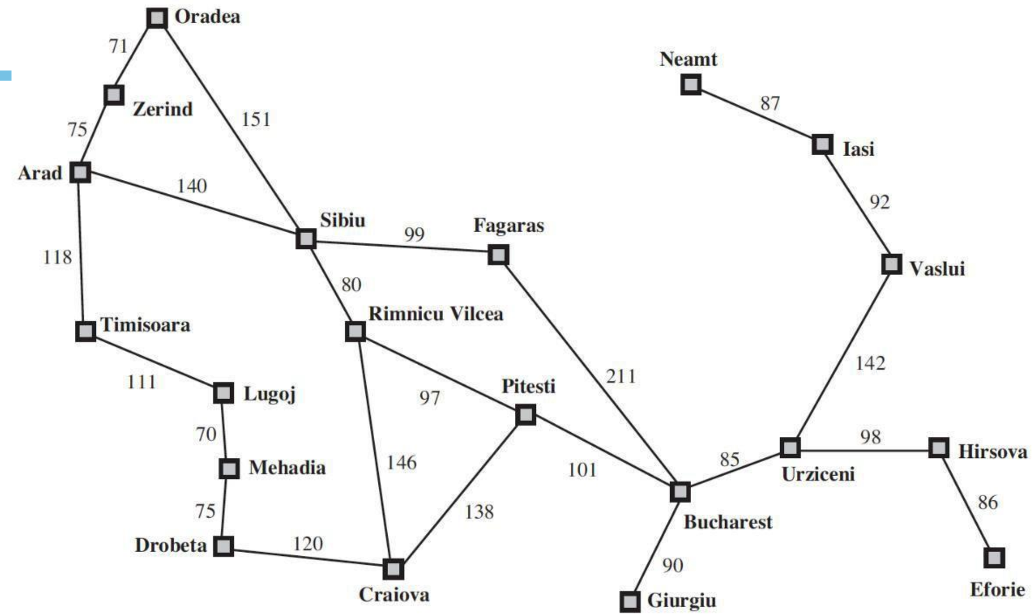
Phases of Solution Search

Goal Formulation

Problem Formulation

Search Phase

Execution Phase

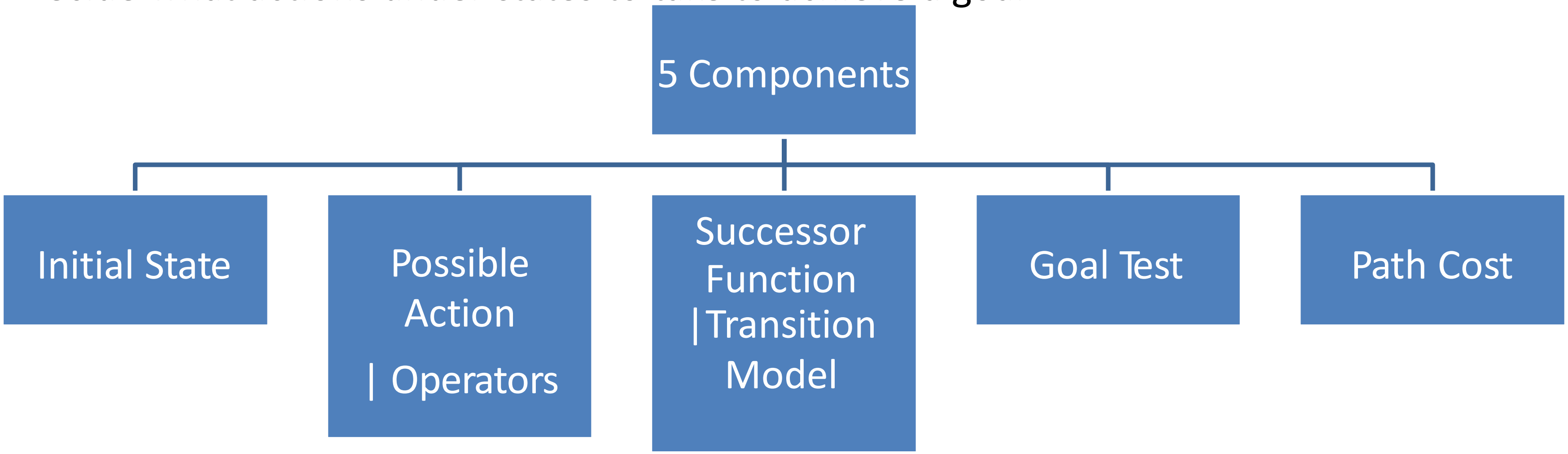


Examine all sequence
Choose best |
Optimal

Problem Solving Agents – Problem Formulation

Abstraction Representation

Decide what actions under states to take to achieve a goal

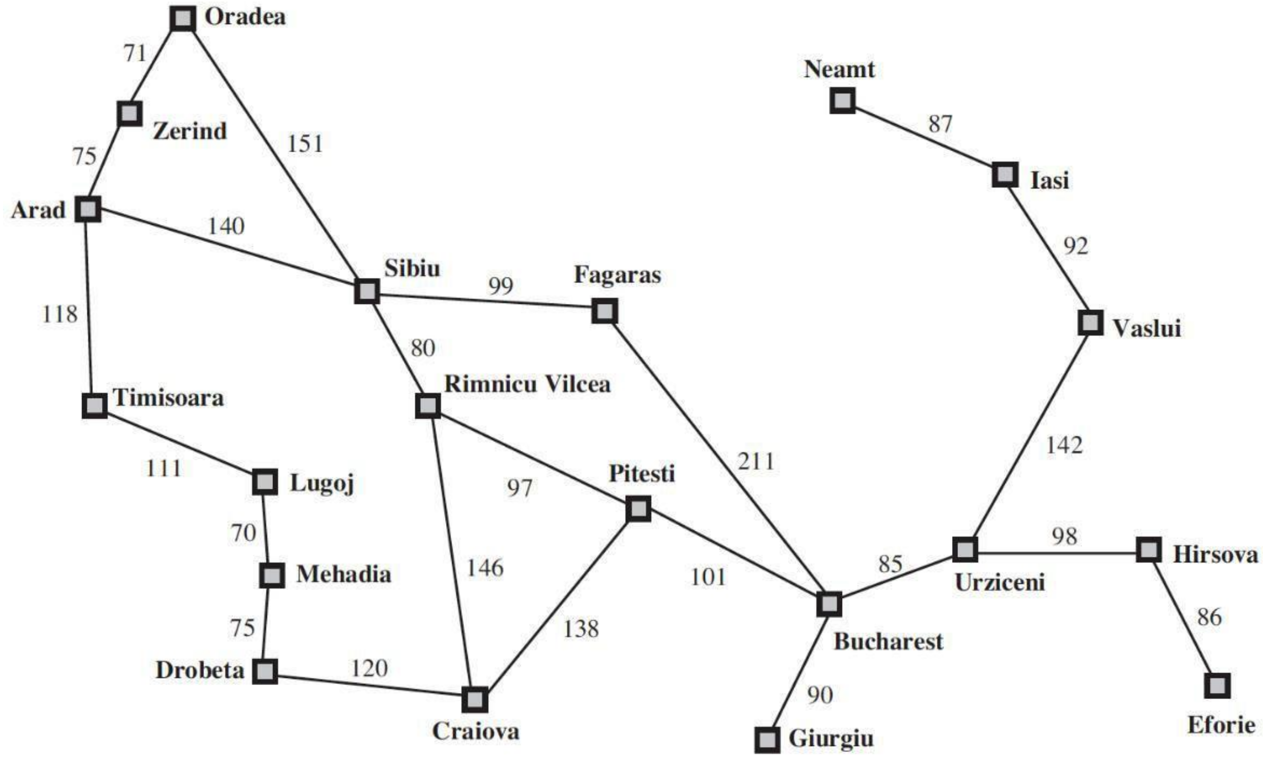


A function that assigns a numeric cost to each path. A path is a series of actions. Each action is given a cost depending on the problem.

Solution = Path Cost Function + Optimal Solution

Problem Solving Agents – Problem Formulation:

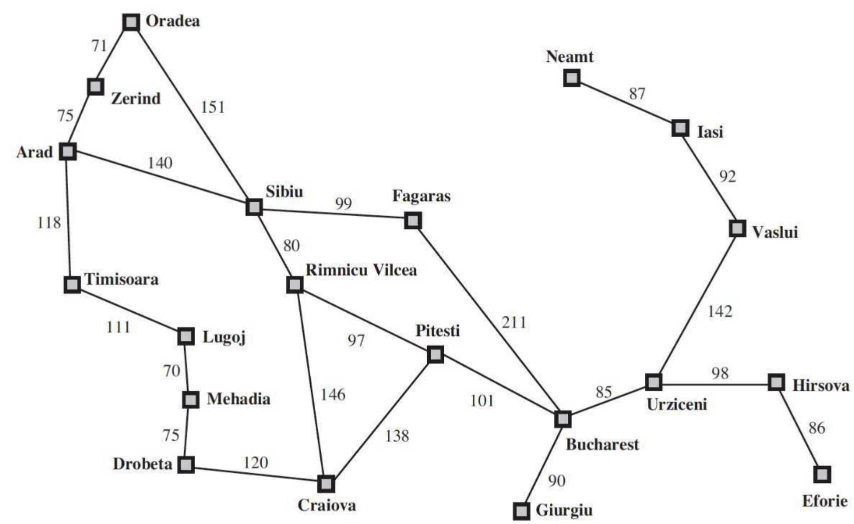
Book Example



Initial State –E.g., *In(Arad)*
Possible Actions – *ACTIONS(s)* □ *{Go(Sibiu), Go(Timisoara), Go(Zerind)}*
Transition Model – *RESULT(In(Arad), Go(Sibiu)) = In(Sibiu)*
Goal Test – *IsGoal(In(Bucharest)) = Yes*
Path Cost – *cost(In(Arad), go(Sibiu)) = 140 kms*

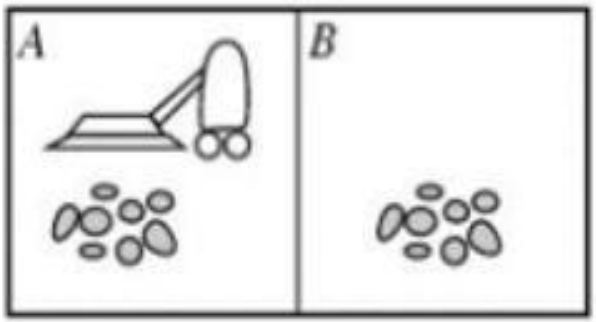
Example Problem Formulation

	Travelling Problem
Initial State	Based on the problem
Possible Actions	Take a flight | Train | Shop
Transition Model/ Successor Function	[A, Go(A->S)] = [S]
Goal Test	Is current = B (destination)
Path Cost	Cost + Time + Quality



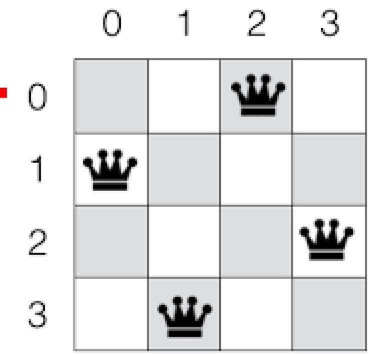
Example Problem Formulation

	Vacuum World
Initial State	Any
Possible Actions	[Move Left, Move Right, Suck, NoOps]
Transition Model/ Successor Function	[A, ML] = [B , Dirty] [A, ML] = [B, Clean]
Goal Test	Is all room clean? [A, Clean] [B, Clean]
Path Cost	No of steps in path



Example Problem Formulation

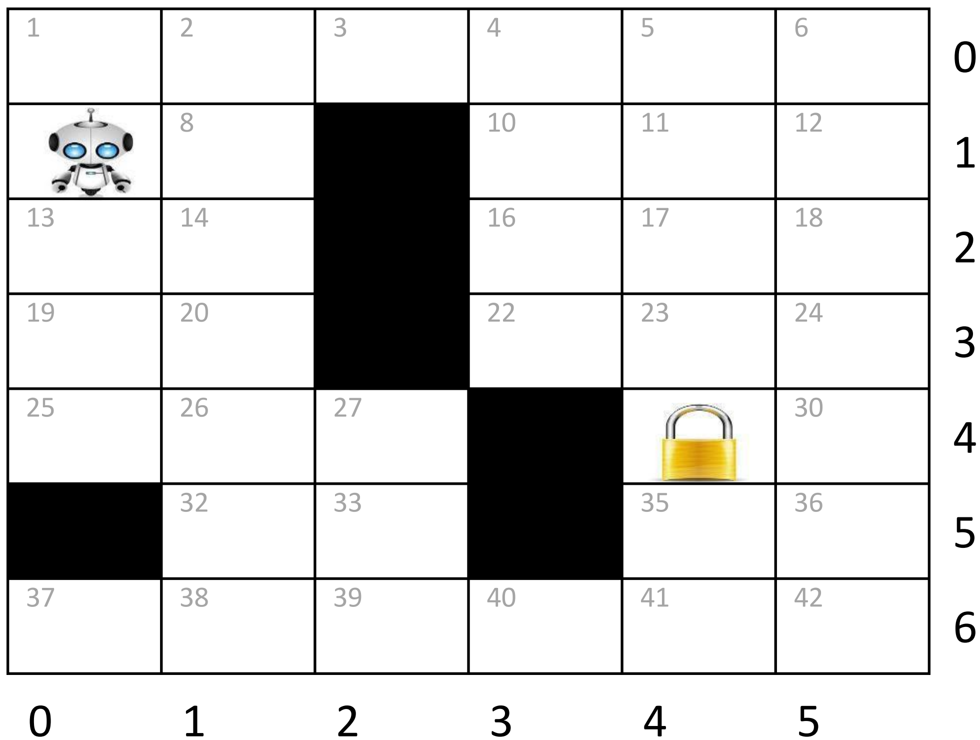
	N-Queen
Initial State	Empty | Partial | Full
Possible Actions	
Transition Model/ Successor Function	
Goal Test	
Path Cost	



board[r][c]

Path finding Robot

Successor Function Design



N-W-E-S

Graph Searching

- ➢Graph as state space (node = state, edge = action)
- ➢For example, game trees, mazes, ...

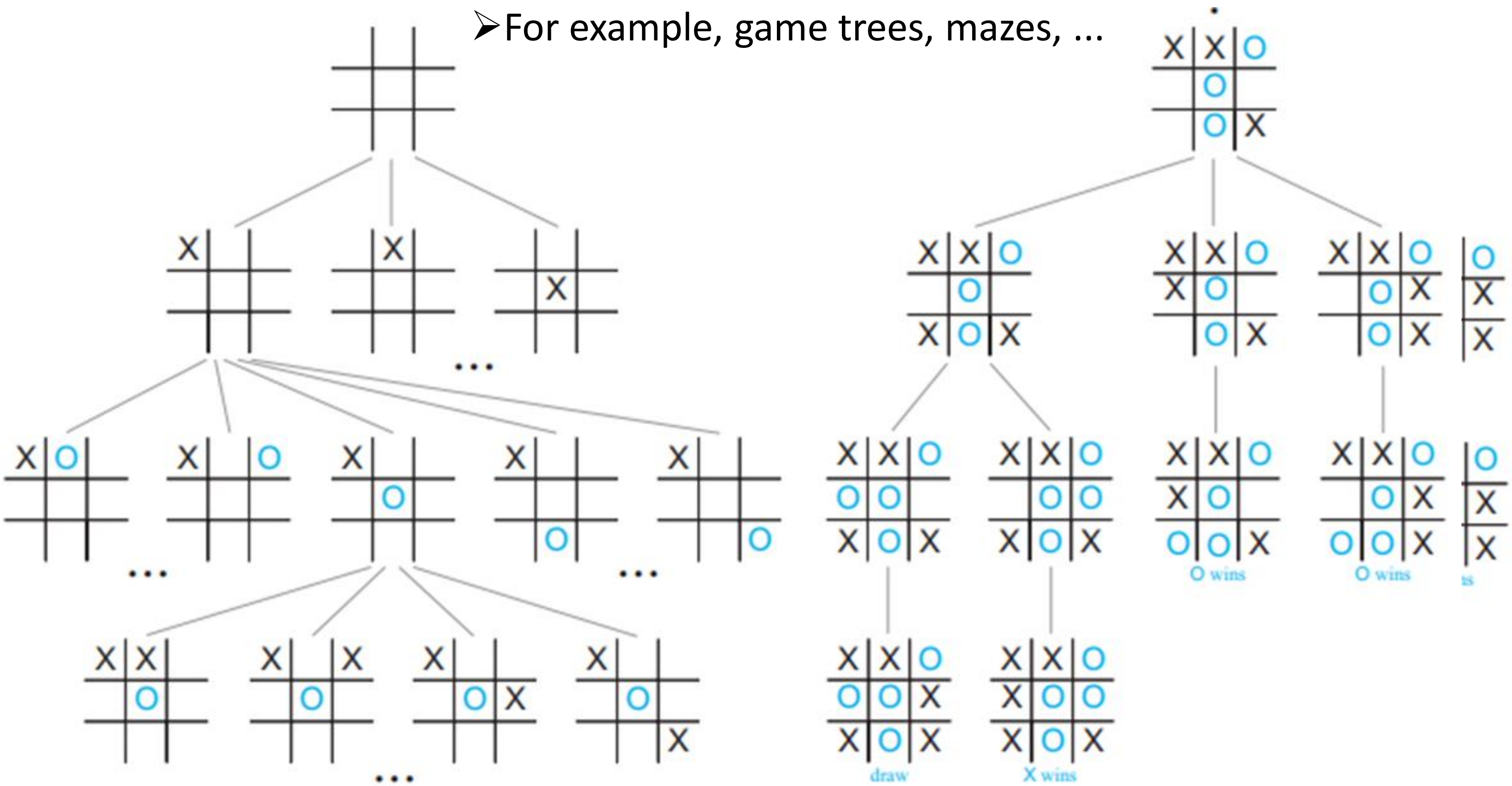


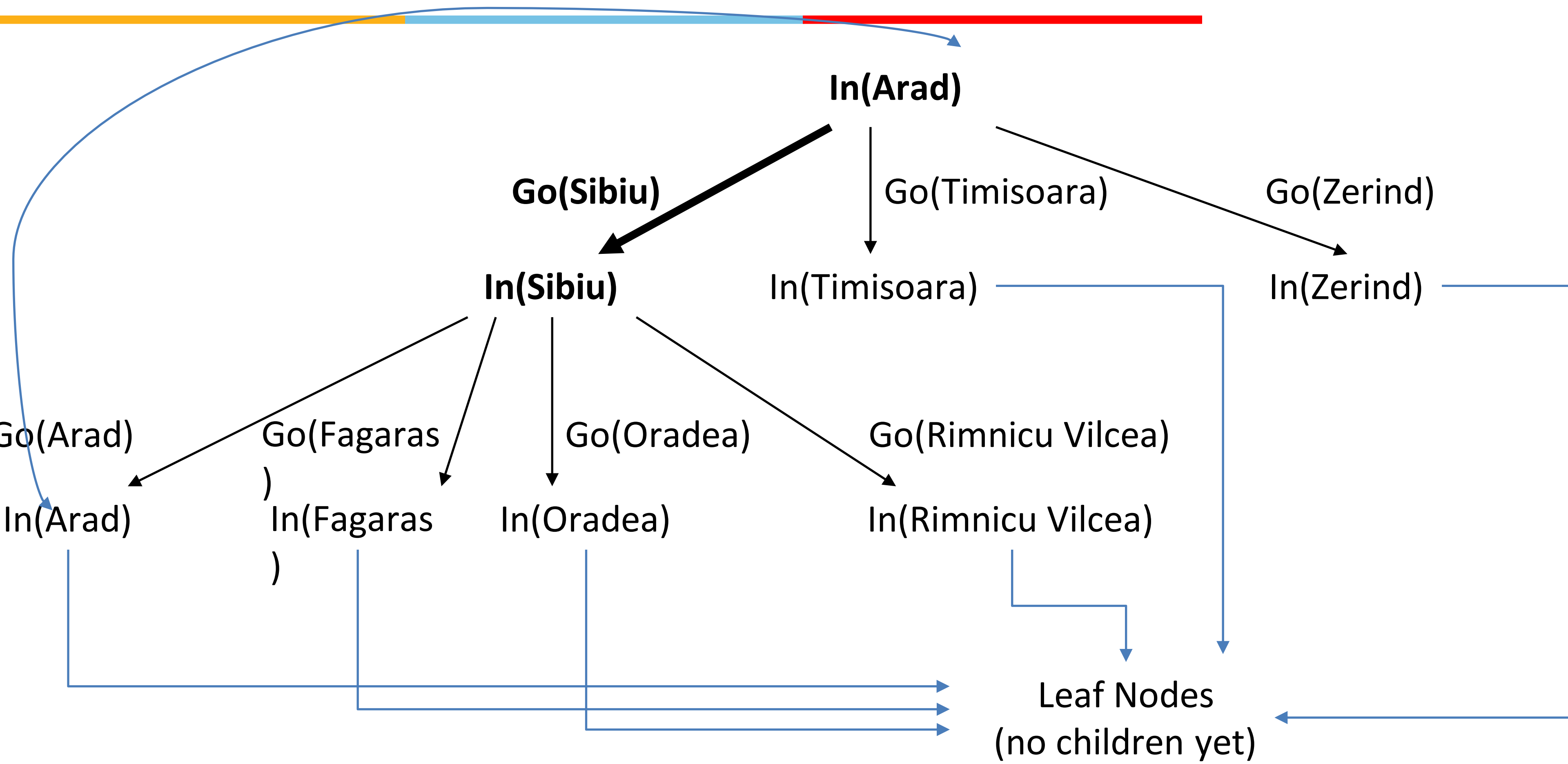
FIGURE 8 **Some of the Game Tree for Tic-Tac-Toe.**

Searching for Solutions

Choosing the current state, testing possible successor function, expanding current state to generate new state is called Traversal. Choice of which state to expand – Search Strategy

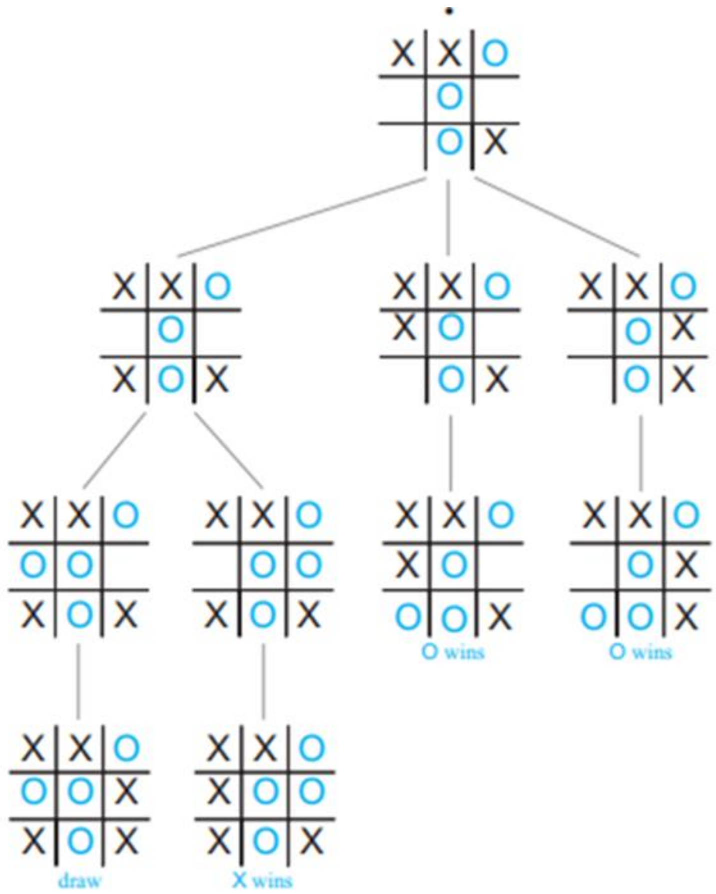
Search Strategy (under certainty)

- **Uninformed (will be discussed in next class)**
  - BFS, DFS, UCS
  - IDS, DLS
  - Bi-Directional
- **Informed (will be discussed in next class)**
  - Best First Search
  - A*


In(Arad)
Go(Sibiu)
Go(Timisoara)
Go(Zerind)
In(Sibiu)
In(Timisoara)
In(Zerind)
Go(Arad)
Go(Fagaras
)
Go(Oradea)
Go(Rimnicu Vilcea)
In(Arad)
In(Fagaras
)
In(Oradea)
In(Rimnicu Vilcea)
Leaf Nodes
(no children yet)

Graph Searching

- BFS and DFS each search the state space for a best move.
- If the search is exhaustive they will find the same solution, but if there is a time limit and the search space is large...
- DFS explores a few possible moves, looking at the effects far in the future
- BFS explores many solutions but only sees effects in the near future (often finds shorter solutions)



Next Class Plan

- Uninformed Search Algorithms
  - BFS vs DFS – An overview
  - Uniform Cost Search
  - Iterative Depth First Search
  - Notion of Bi-Directional Search
- Informed Search Algorithms
  - Greedy Best First search
  - A* Search (Start)

Required Reading: AIMA - Chapter #1, 2, 3.1, 3.2, 3.3

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

Artificial & Computational Intelligence

AIML CLZG557

M1 : Introduction
&
M2 : Problem Solving Agent using Search

Raja vadhana P

Assistant Professor,

BITS - CSIS

BITS Pilani
Pilani Campus

Course Plan

M1	Introduction to AI
M2	Problem Solving Agent using Search
M3	Game Playing
M4	Knowledge Representation using Logics
M5	Probabilistic Representation and Reasoning
M6	Reasoning over time
M7	Ethics in AI

Learning Objective

At the end of this class , students Should be able to:

1. Design problem solving agents
2. Create search tree for given problem
3. Apply uninformed search algorithms to the given problem
4. Compare performance of given algorithms in terms of completeness, optimality, time and space complexity
5. Differentiate for which scenario appropriate uninformed search technique is suitable and justify

Problem Formulation

Problem Solving Agents

Goal Formulation

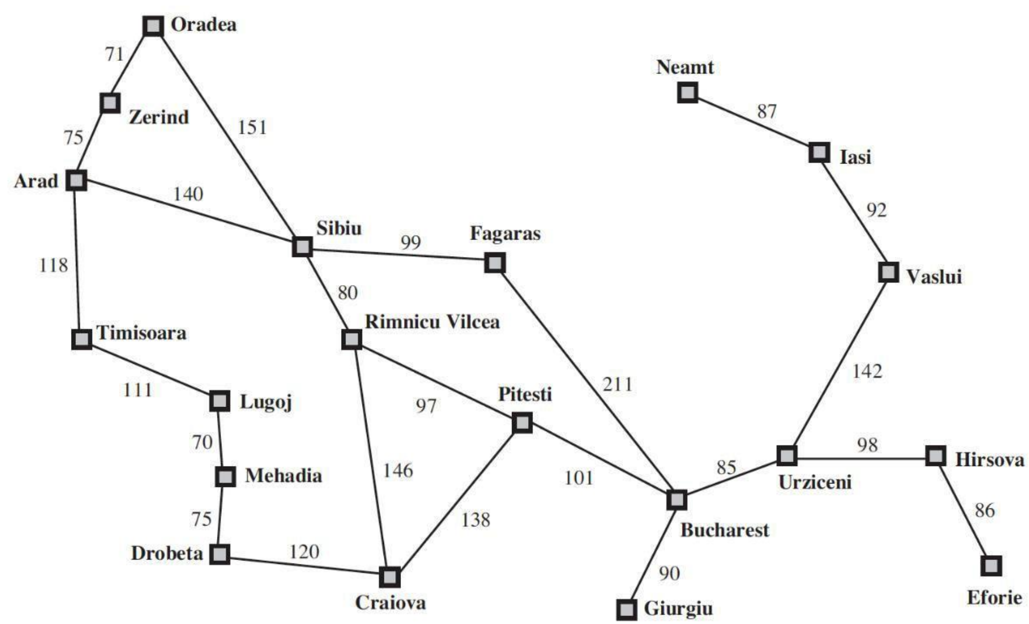
Problem Formulation

Search Phase

Execution Phase

Phases of Solution Search by PSA

Assumptions – Environment :
Static
Observable Discrete
Deterministic



Problem Solving Agents – Problem Formulation

Abstraction Representation

Decide what actions under states to take to achieve a goal

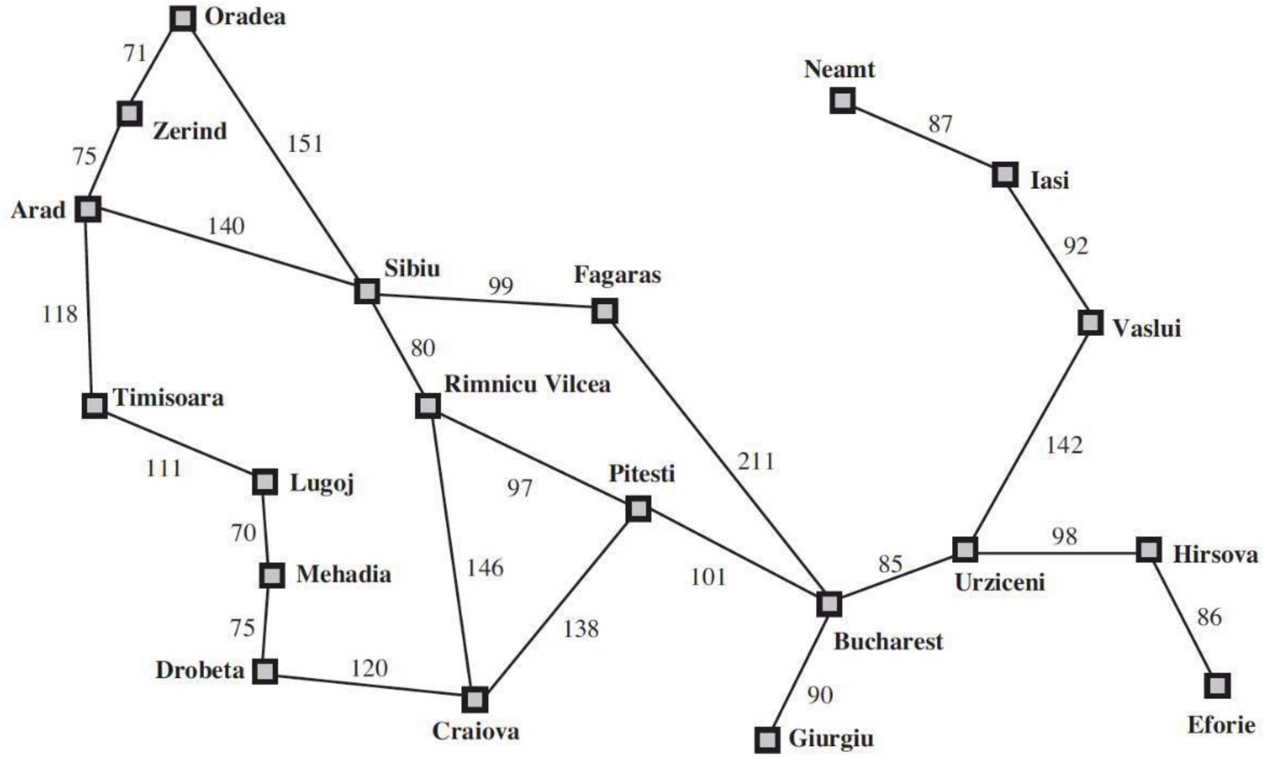
5 Components

- Initial State
- Possible Action | Operators
- Successor Function | Transition Model
- Goal Test
- Path Cost

A function that assigns a numeric cost to each path. A path is a series of actions. Each action is given a cost depending on the problem.

Solution = Path Cost Function + Optimal Solution

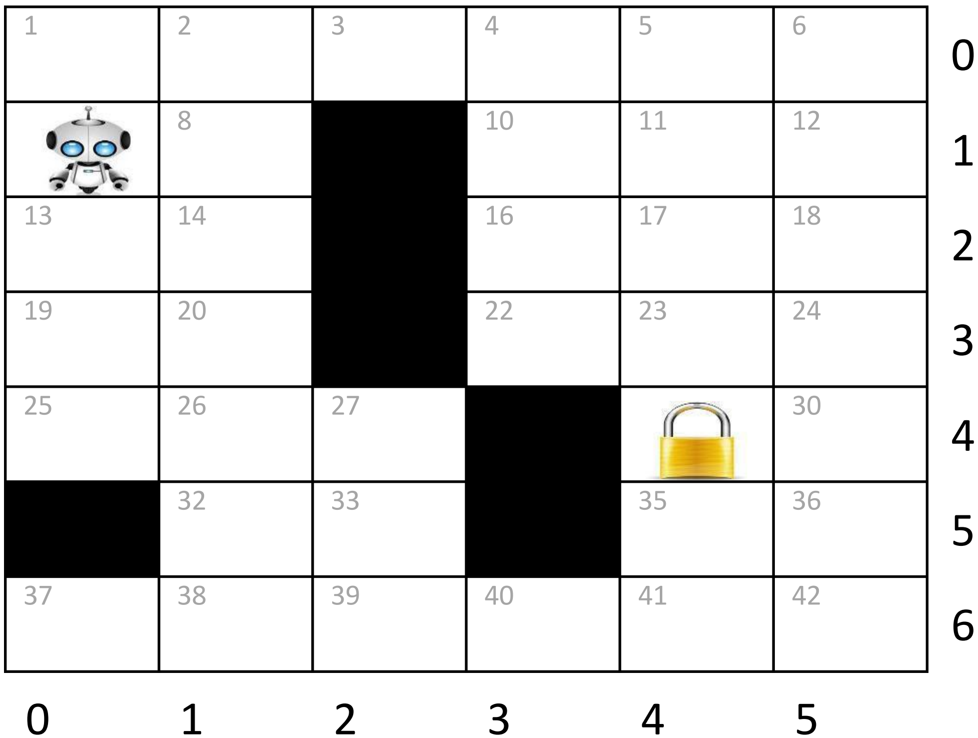
Problem Solving Agents – Problem Formulation: Book Example



Initial State –E.g., *In(Arad)*
Possible Actions – *ACTIONS(s) {Go(Sibiu), Go(Timisoara), Go(Zerind)}*
Transition Model – *RESULT(In(Arad), Go(Sibiu)) = In(Sibiu)*
Goal Test – *IsGoal(In(Bucharest)) = Yes*
Path Cost – *cost(In(Arad), go(Sibiu)) = 140 kms*

Path finding Robot

Successor Function Design



N-W-E-S

Graph Searching

- Graph as state space (node = state, edge = action)
- For example, game trees, mazes, ...

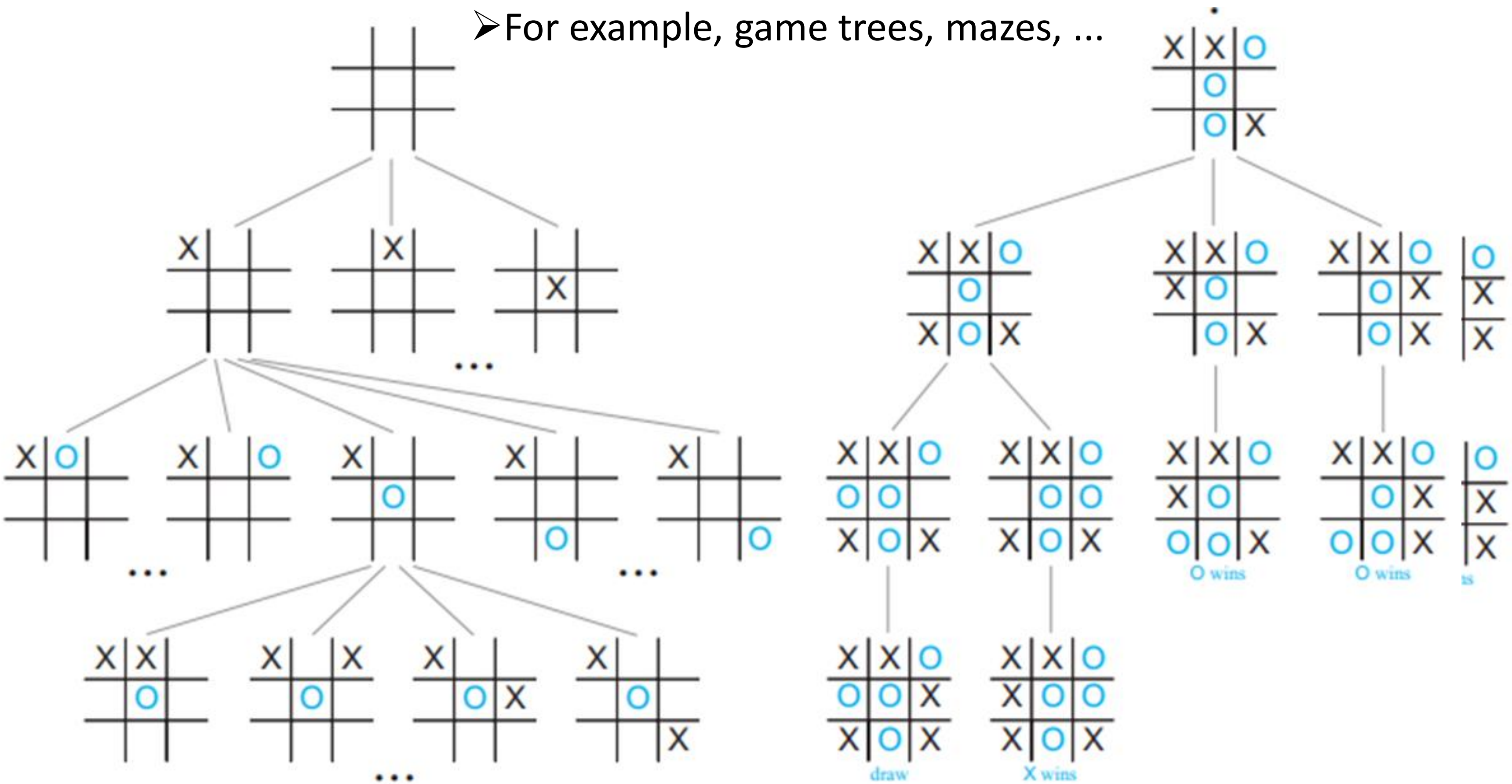


FIGURE 8 Some of the Game Tree for Tic-Tac-Toe.

Searching for Solutions

Choosing the current state, testing possible successor function, expanding current state to generate new state is called Traversal. Choice of which state to expand – Search Strategy

Search Strategy (under certainty)

Uninformed

BFS, DFS, UCS

IDS, DLS

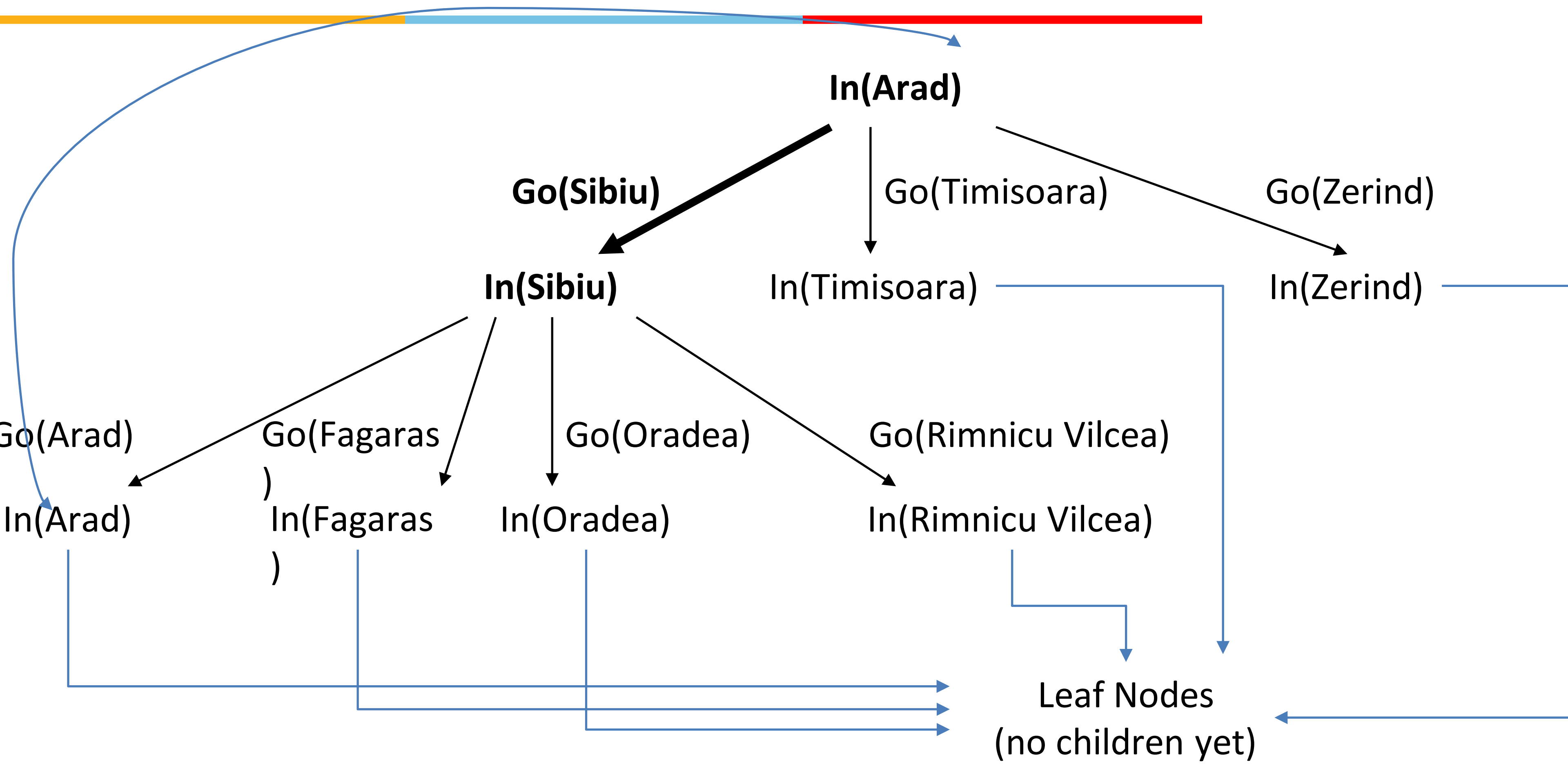
Bi-Directional

Informed

Best First Search

A*

AO*


In(Arad)
Go(Sibiu)
Go(Timisoara)
Go(Zerind)
In(Sibiu)
In(Timisoara)
In(Zerind)
Go(Arad)
Go(Fagaras
)
Go(Oradea)
Go(Rimnicu Vilcea)
In(Arad)
In(Fagaras
)
In(Oradea)
In(Rimnicu Vilcea)
Leaf Nodes
(no children yet)

Application

Breadth First Search

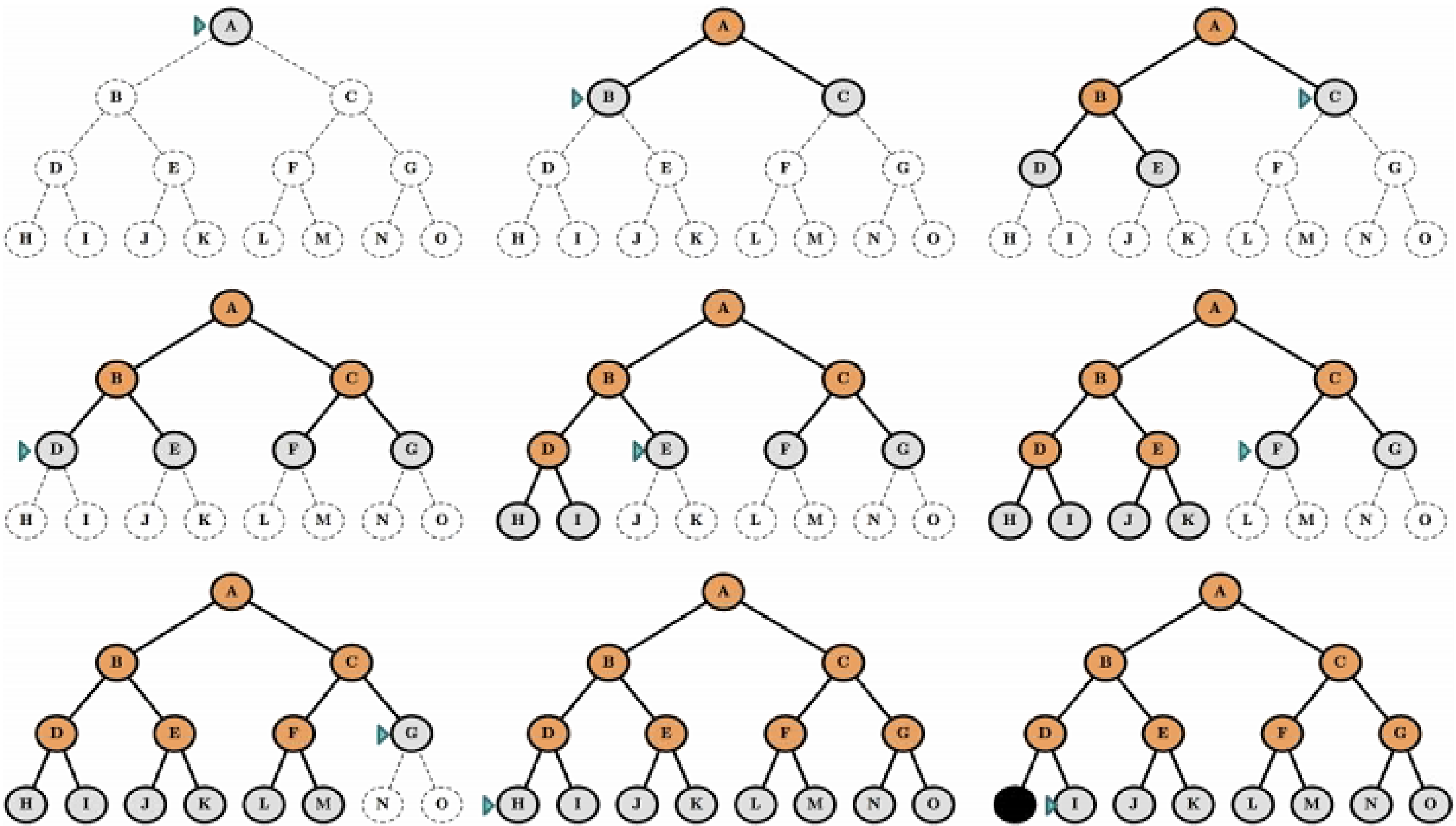
- Finding path in a graph (many solutions)
- Finding the Bipartitions in a graph

Depth First Search

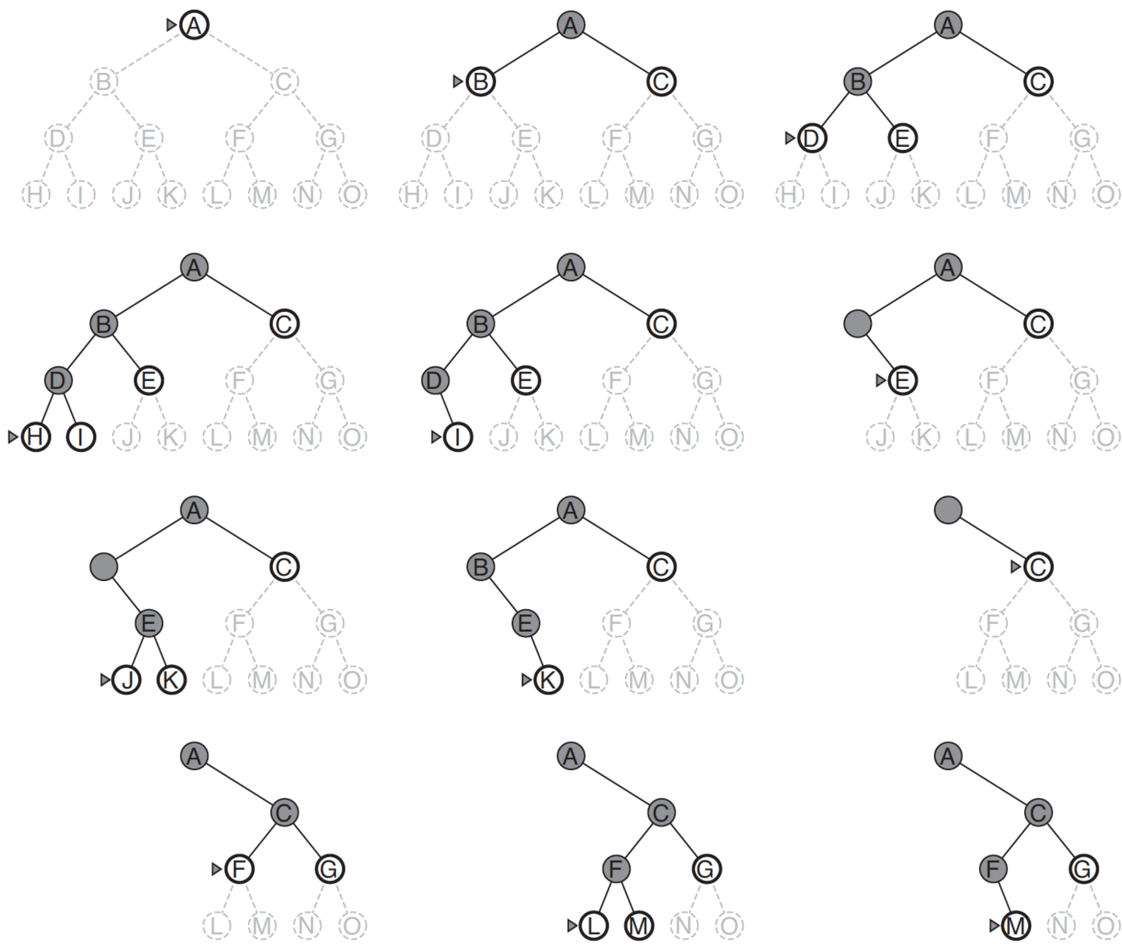
- Find the Connectedness in a graph
- Topological Sorting

Uninformed Search Overview

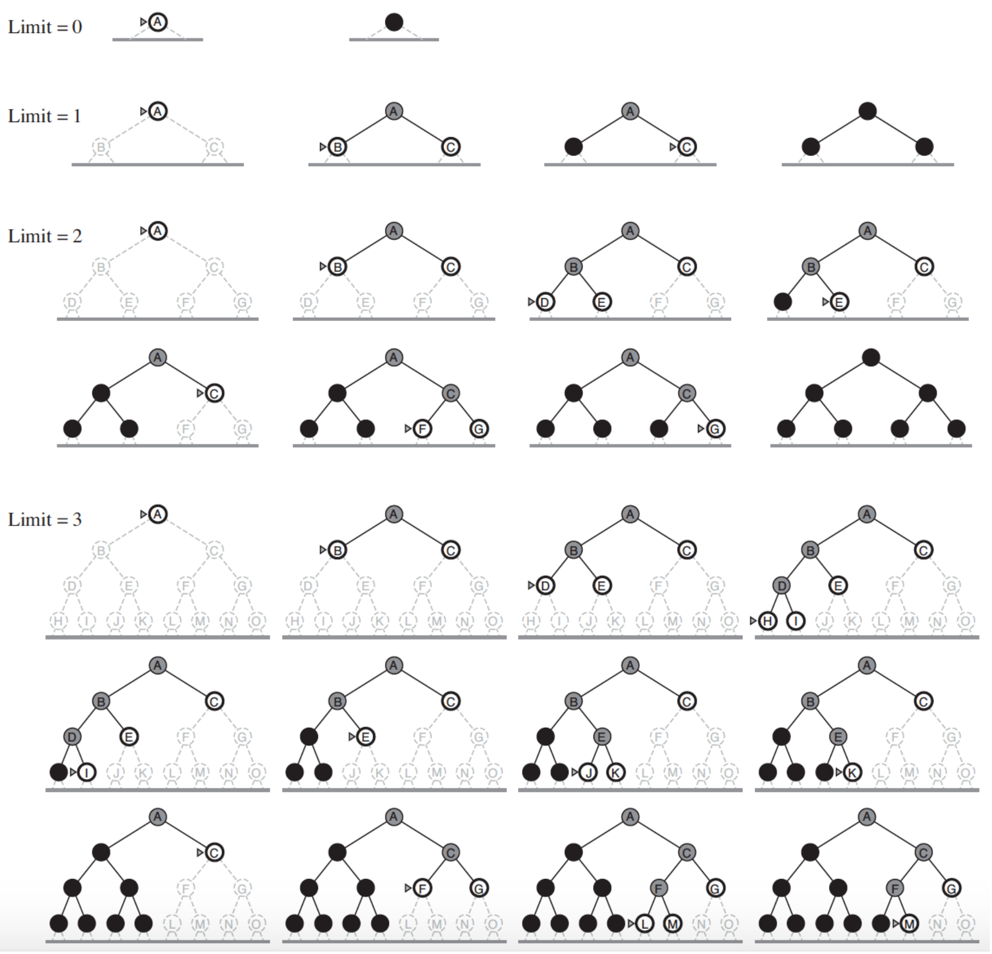
Breadth First Search (BFS)



Depth First Search (DFS)

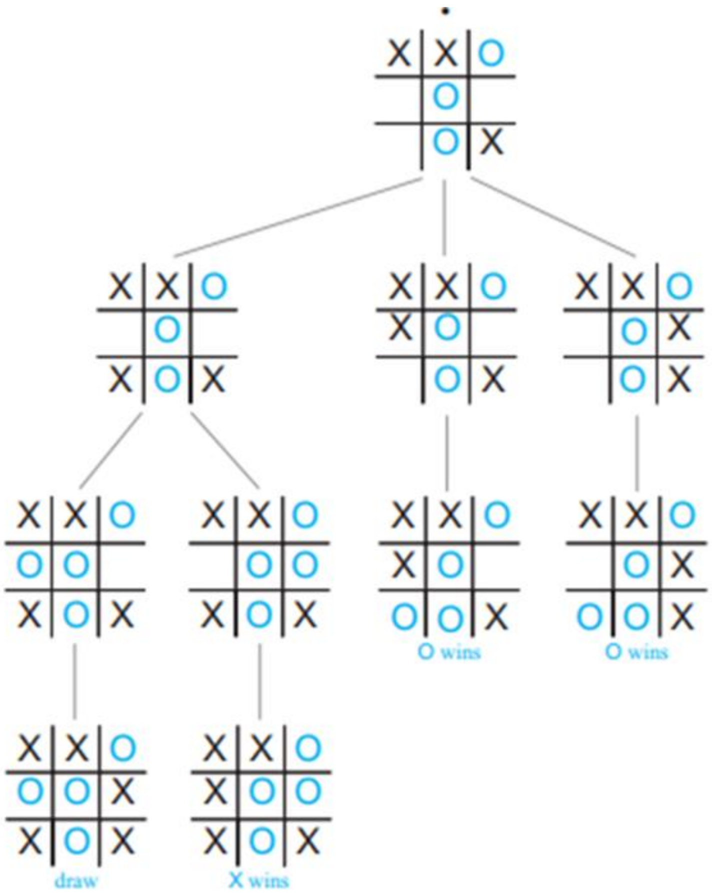


Iterative Deepening Depth First Search (IDS)

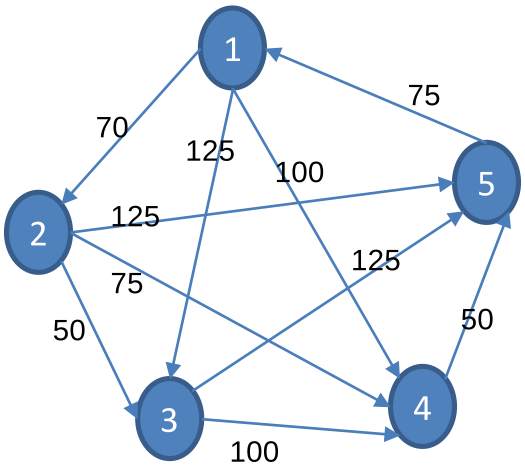


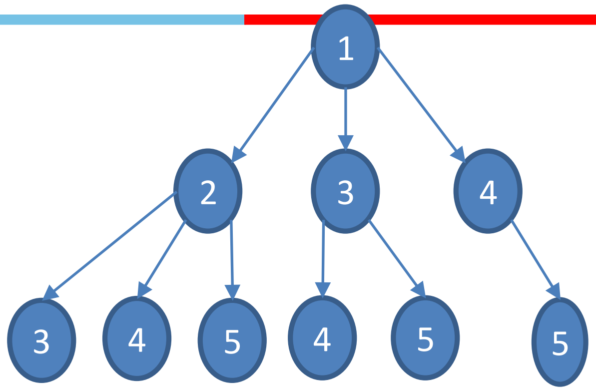
Graph Searching

- BFS and DFS each search the state space for a best move.
- If the search is exhaustive they will find the same solution, but if there is a time limit and the search space is large...
- DFS explores a few possible moves, looking at the effects far in the future
- BFS explores many solutions but only sees effects in the near future (often finds shorter solutions)



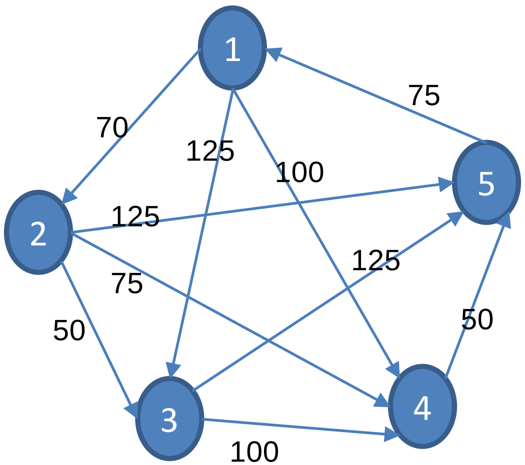
Search Tree – Sample Generation

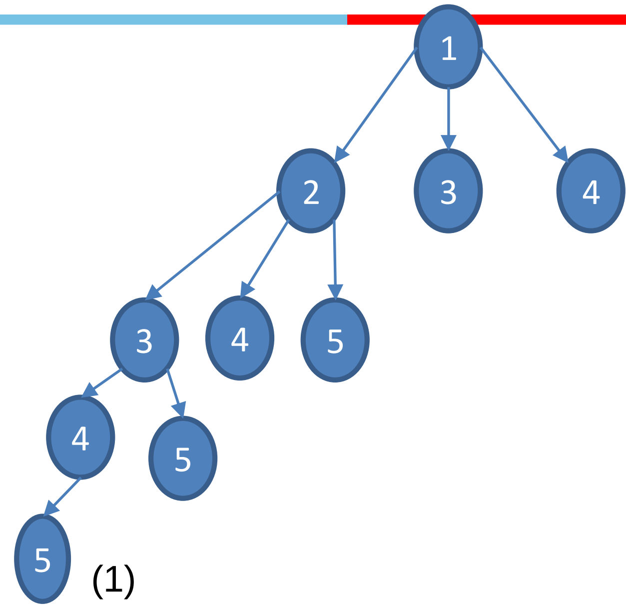




Each NODE in in the search tree denotes an entire PATH through the state space graph.

Search Algorithm – Uninformed Example - 1

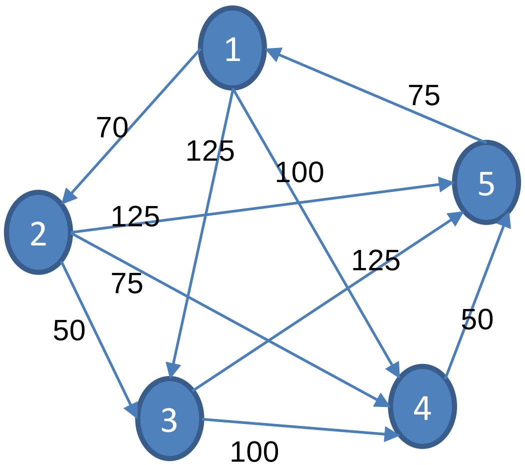


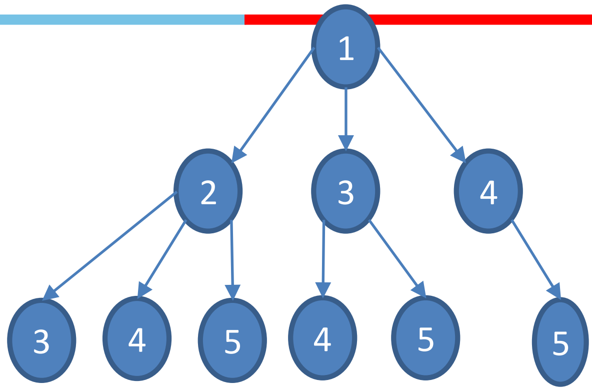


(1)
(1 2) (1 3) (1 4)
(1 2 3) (1 2 4) (1 2 5) (1 3) (1 4)
(1 2 3 4) (1 2 3 5) (1 2 4) (1 2 5) (1 3) (1 4)
(1 2 3 4 5) (1 2 3 5) (1 2 4) (1 2 5) (1 3) (1 4)

C(1-2-3-4-5) = 70 + 50 + 100 + 50 =270
Expanded : 4
Generated : 10
Max Queue Length : 6

Search Algorithm – Uninformed Example - 2



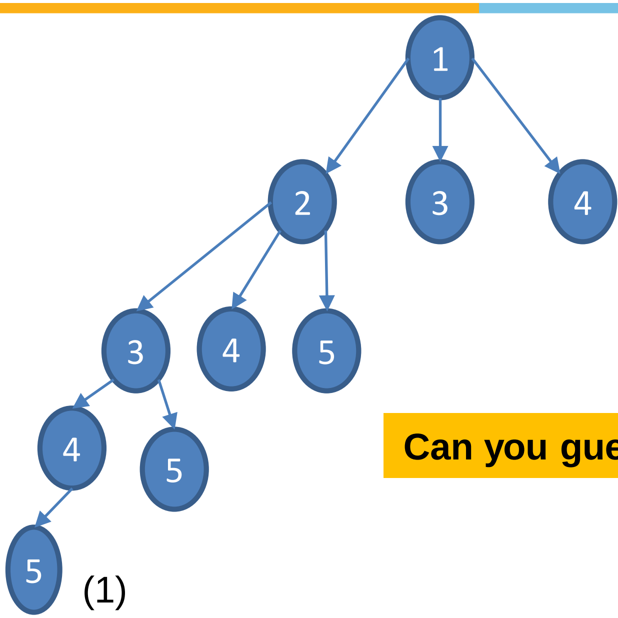


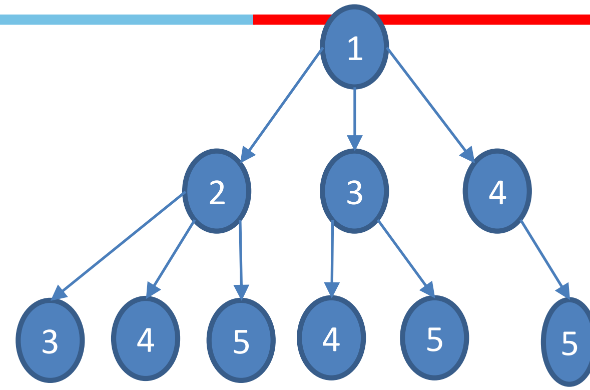
(1)
(1 2) (1 3) (1 4)
TEST FAILED

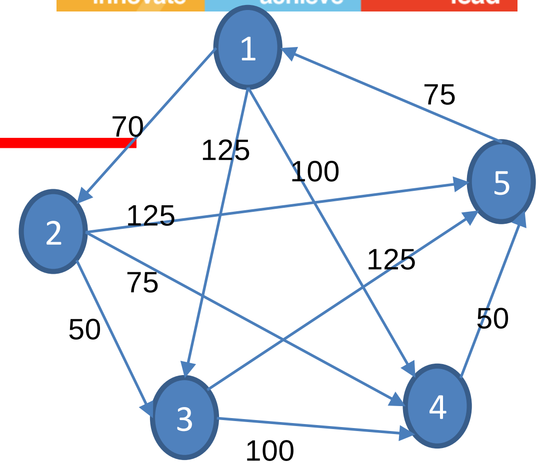
(1 3) (1 4) (1 2 3) (1 2 4) (1 2 5)
(1 2 3) (1 2 4) (1 2 5) (1 3 4) (1 3 5) (1 4 5)
TEST PASSED

C(1-2-5) = 70 + 125 =195
Expanded : 4
Generated : 10
Max Queue Length : 6

Search Algorithm – Uninformed Example - 2







Can you guess which algorithm are these ?

(1)
(1 2) (1 3) (1 4)
(1 2 3) (1 2 4) (1 2 5) (1 3) (1 4)
(1 2 3 4) (1 2 3 5) (1 2 4) (1 2 5) (1 3) (1 4)
(1 2 3 4 5) (1 2 3 5) (1 2 4) (1 2 5) (1 3) (1 4)

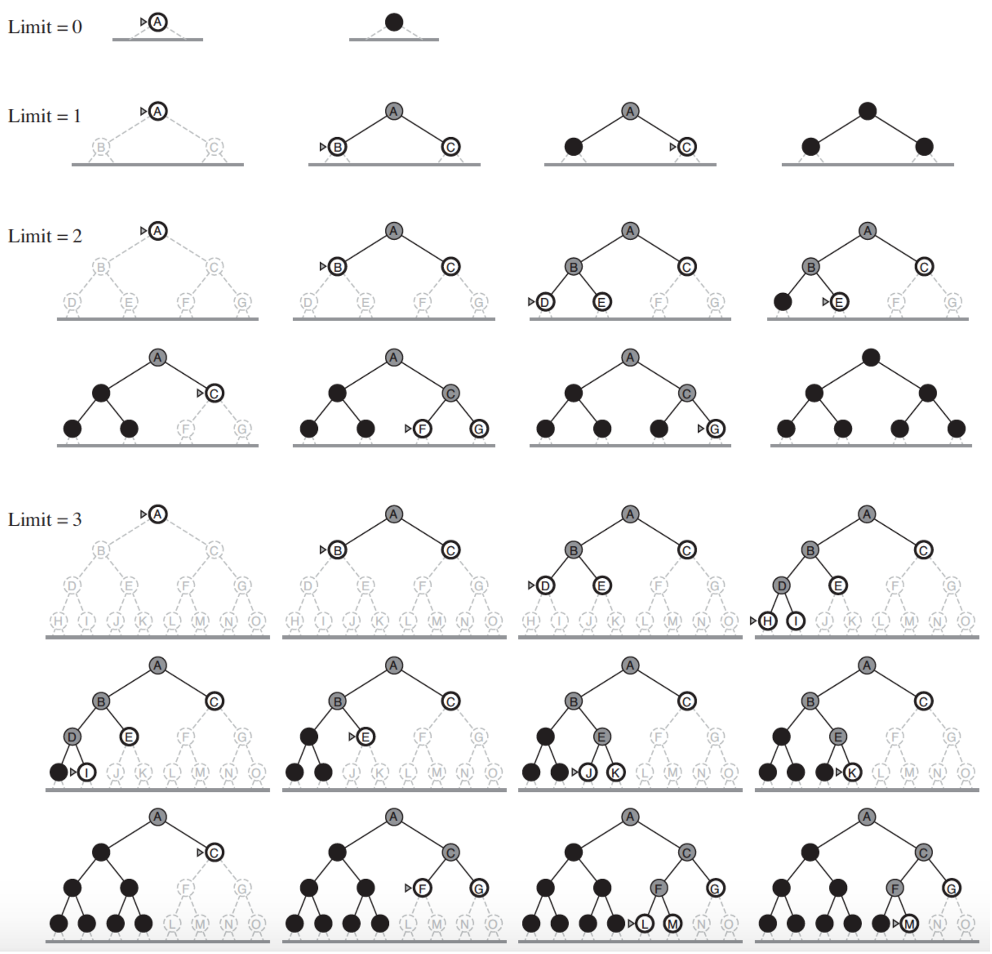
C(1-2-3-4-5) = 70 + 50 + 100 + 50 =270
Expanded : 4
Generated : 10
Max Queue Length : 6

(1)
(1 2) (1 3) (1 4)
TEST FAILED

(1 3) (1 4) (1 2 3) (1 2 4) (1 2 5)
(1 2 3) (1 2 4) (1 2 5) (1 3 4) (1 3 5) (1 4 5)
TEST PASSED

C(1-2-5) = 70 + 125 =195
Expanded : 4
Generated : 10
Max Queue Length : 6

Iterative Deepening Depth First Search (IDS)

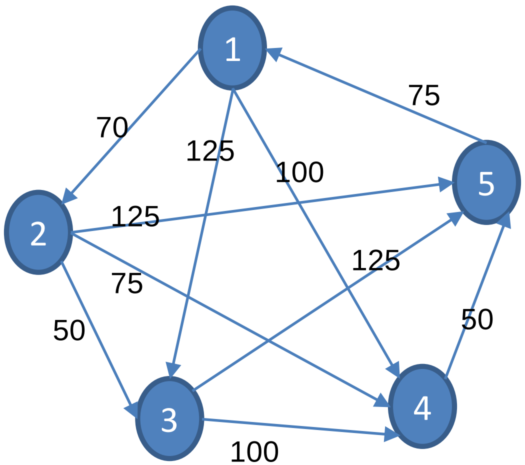


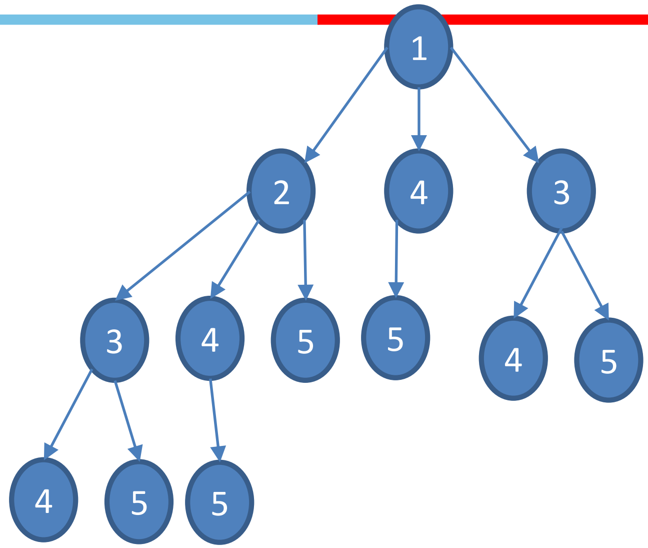
Algorithm Tracing

Students must follow this in the exams for all the search algorithms in addition to the search tree constructions. The ordering of the Open Lists must be in consistent with the algorithm with a note on the justification of the order expected!

Iter	Open List / Frontiers / Fringes	Closed List	Goal Test
1.	(1)		Fail on (1)
2.	(1 3), (1 4), (1 2)	(1)	Fail on (1 3)

UCS





(1)
(1 2 : 70) (1 4 : 100) (1 3 : 125)
TEST-F
(1 4 : 100) (1 2 3 :120)(1 3 : 125) (1 2 4 : 145) (1 2 5 : 195)
TEST-F
(1 2 3 :120)(1 3 : 125) (1 2 4 : 145) (1 4 5: 150)(1 2 5 : 195)
TEST-F
(1 3 : 125) (1 2 4 : 145) (1 4 5: 150) (1 2 3 4:170) (1 2 5 : 195) (1 2 3 5 : 245)
TEST-F
(1 2 4 : 145) (1 4 5: 150) (1 2 3 4:170) (1 2 5 : 195) (1 3 4 : 225) (1 2 3 5 : 245) (1 3 5 : 250)
TEST-F
(1 4 5: 150) (1 2 3 4:170) (1 2 4 5 : 195) (1 2 5 : 195) (1 3 4 : 225) (1 2 3 5 : 245) (1 3 5 : 250)
TEST - P

Uniform Cost Search

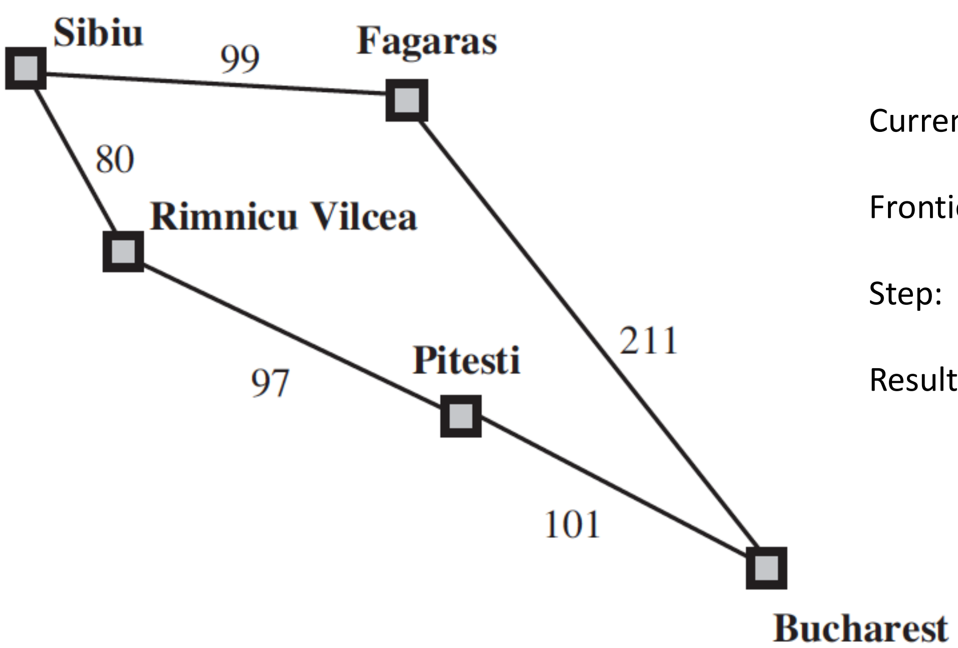
Instead of expanding the shallowest node, Uniform-Cost search expands the node n with the lowest path cost g(n)

Sorting the Frontier as a priority queue ordered by g(n)

Goal test is applied during expansion

- The goal node if generated may not be on the optimal path
- Find a better path to a node on the Frontier

Uniform Cost Search



Current State: Sibiu

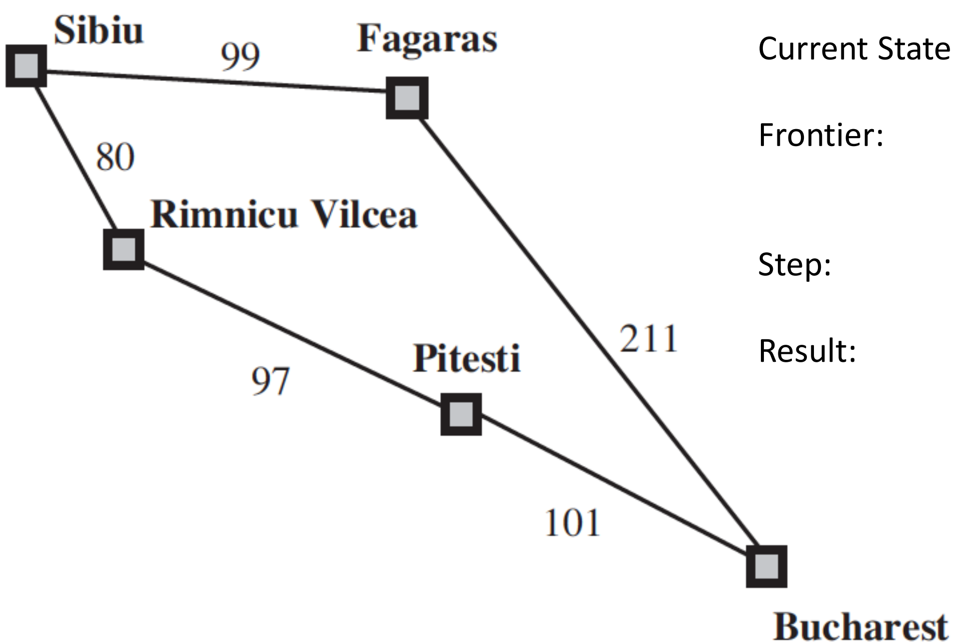
Frontier: []

Step: Expand Sibiu

Result: Generates (“Rimnicu Vilcea” 80)
(“Fagaras”, 99)
Add to Frontier

Initial State: Sibiu
Goal State: Bucharest

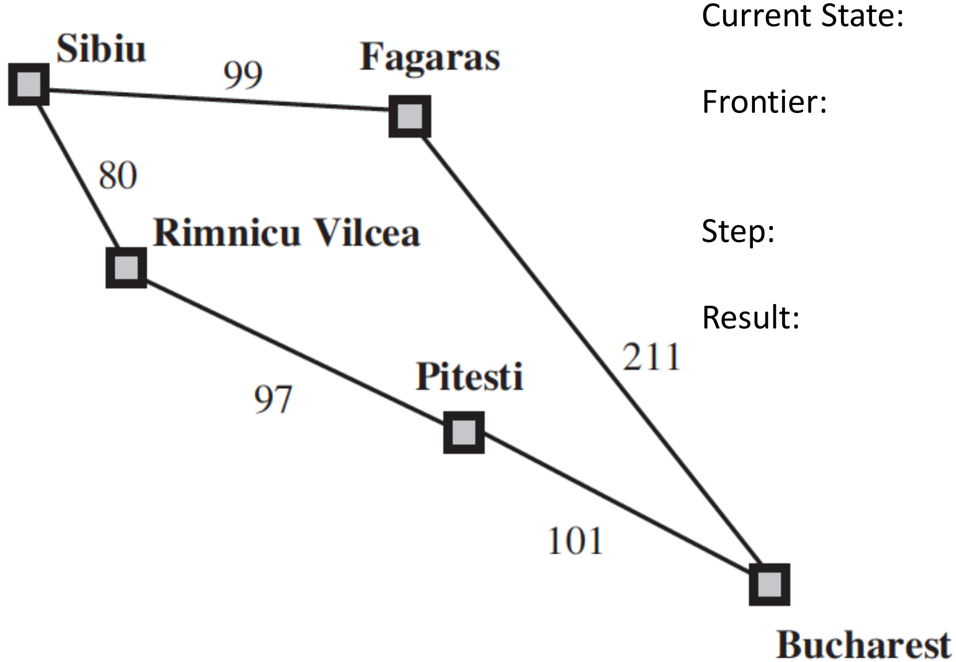
Uniform Cost Search



Current State:	Sibiu
Frontier:	[(“Rimnicu Vilcea” 80) (“Fagaras”, 99)]
Step:	Expand “Rimnicu Vilcea” (least cost)
Result:	Generates (“Pitesti”, 177) Add to Frontier

Initial State: Sibiu
Goal State: Bucharest

Uniform Cost Search



Current State: Rimnicu Vilcea (not a Goal state)

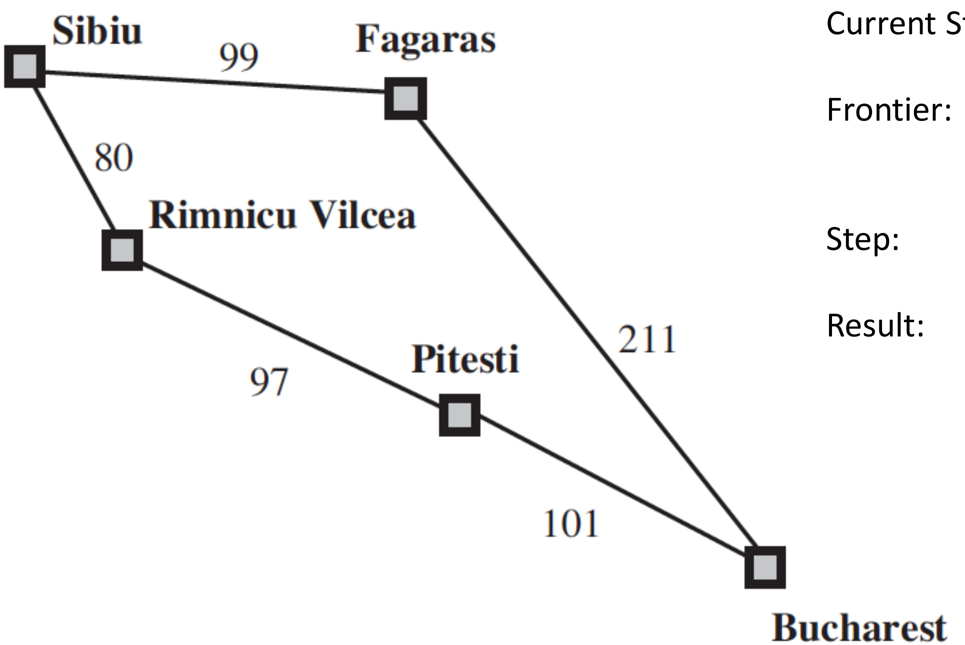
Frontier: [(“Fagaras”, 99)
(“Pitesti”, 177)]

Step: Expand “Fagaras” (least cost)

Result: Generates (“Bucharest”, 310)
Add to Frontier
(It’s a Goal State but we won’t test during generation)

Initial State: Sibiu
Goal State: Bucharest

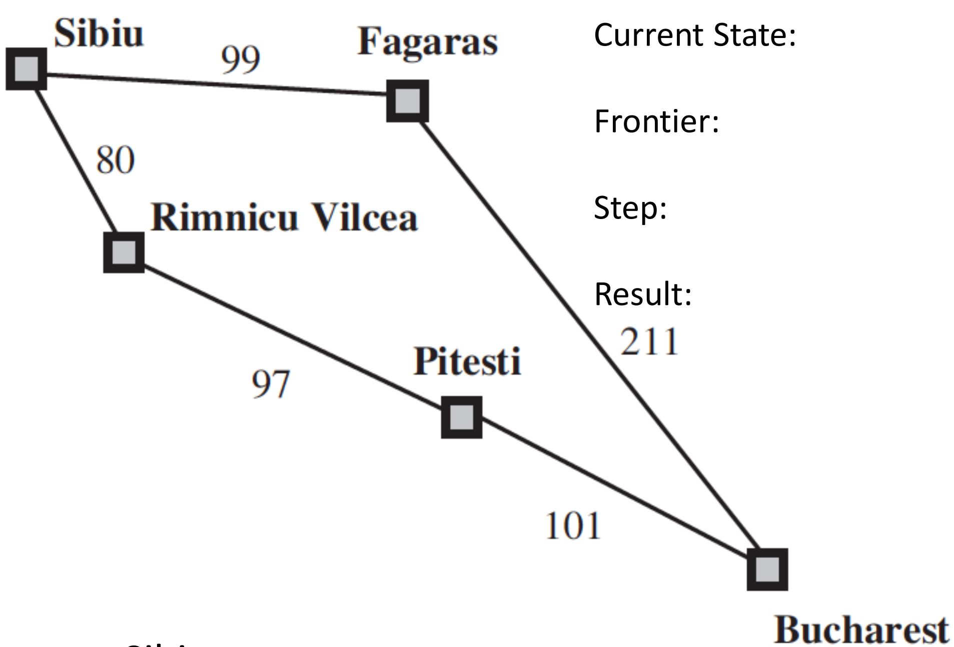
Uniform Cost Search



Current State:	Fagaras (not a goal state)
Frontier:	[("Pitesti", 177) ("Bucharest", 310)]
Step:	Expand "Pitesti" (least cost)
Result:	Generates ("Bucharest", 278) Replace in Frontier (It's a Goal State but we won't test during generation)

Initial State: Sibiu
Goal State: Bucharest

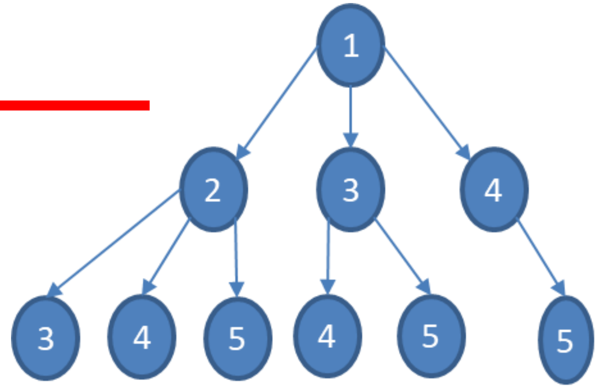
Uniform Cost Search



Current State:	Pitesti (not a goal state)
Frontier:	[(“Bucharest”, 278)]
Step:	Expand “Bucharest”
Result:	No further generation as Goal Test satisfied Return Solution

Initial State: Sibiu
Goal State: Bucharest

Sample Evaluation of the Algorithm



Complete – If the shallowest goal node is at a depth d, BFS will eventually find it by generating all shallower nodes

Optimal – Not necessarily. Optimal if path cost is non-decreasing function of depth of node. E.g., all actions have same cost

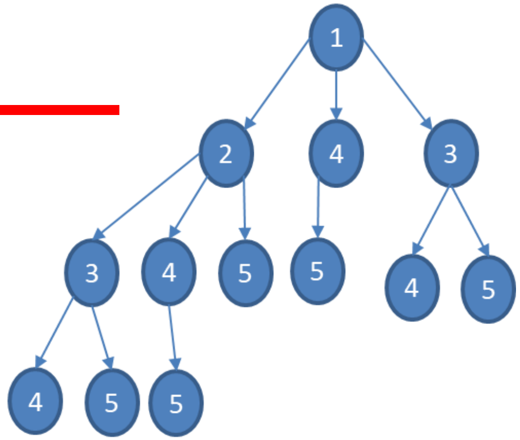
Time Complexity – $\mathcal{O}(b^d)$ b - branching factor, d – depth

- Nodes expanded at depth 1 = b
- Nodes expanded at depth 2 = b^2
- Nodes expanded at depth d = b^d
- Goal test is applied during generation, time complexity would be $\mathcal{O}(b^{d+1})$

Space Complexity – $\mathcal{O}(b^d)$

- $\mathcal{O}(b^{d-1})$ in explored set
- $\mathcal{O}(b^d)$ in frontier set

Uniform Cost Search – Evaluation



Completeness – It is complete if the cost of every step > small +ve constant ∈

- It will stuck in infinite loop if there is a path with infinite sequence of zero cost actions

Optimal – It is Optimal. Whenever it selects a node, it is an optimal path to that node.

Time and Space complexity – Uniform cost search is guided by path costs not depth or branching factor.

- If C* is the cost of optimal solution and ∈ is the min. action cost
- Worst case complexity = $\mathcal{O}(b^{1+\frac{C^*}{\epsilon}})$,
- When all action costs are equal → $\mathcal{O}(b^{d+1})$, the BFS would perform better
  - As Goal test is applied during expansion, Uniform Cost search would do extra work

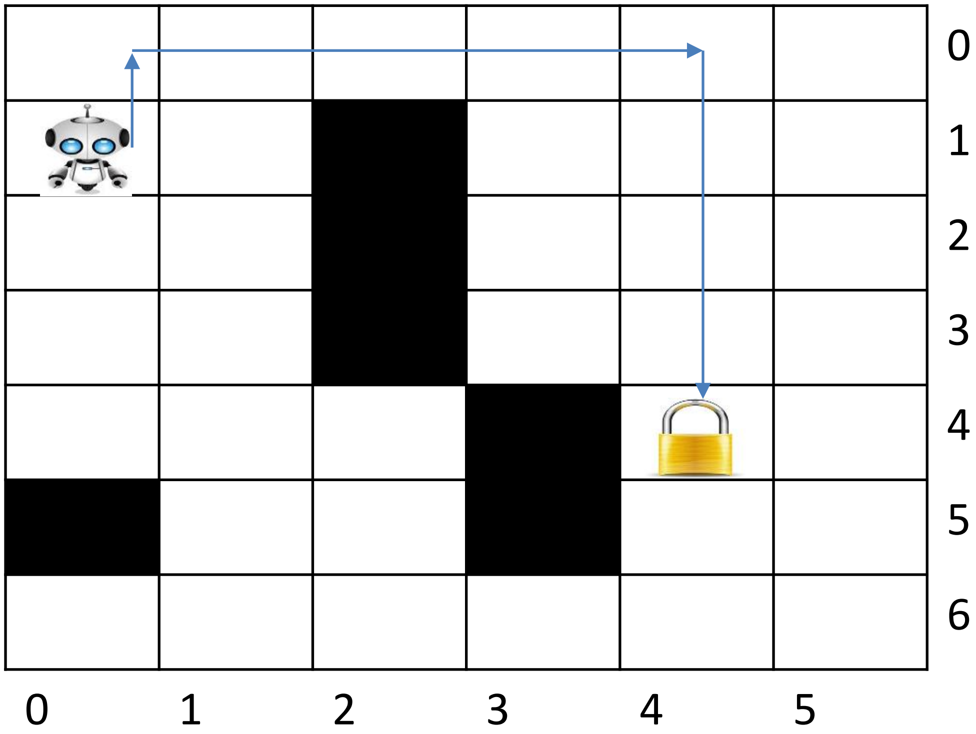
Path finding Robot – Sample Planning Agent Design

Successor Function Design

N-W-E-S - **Check your virtual lab & edit the transition order, to simulate the result in next slides**

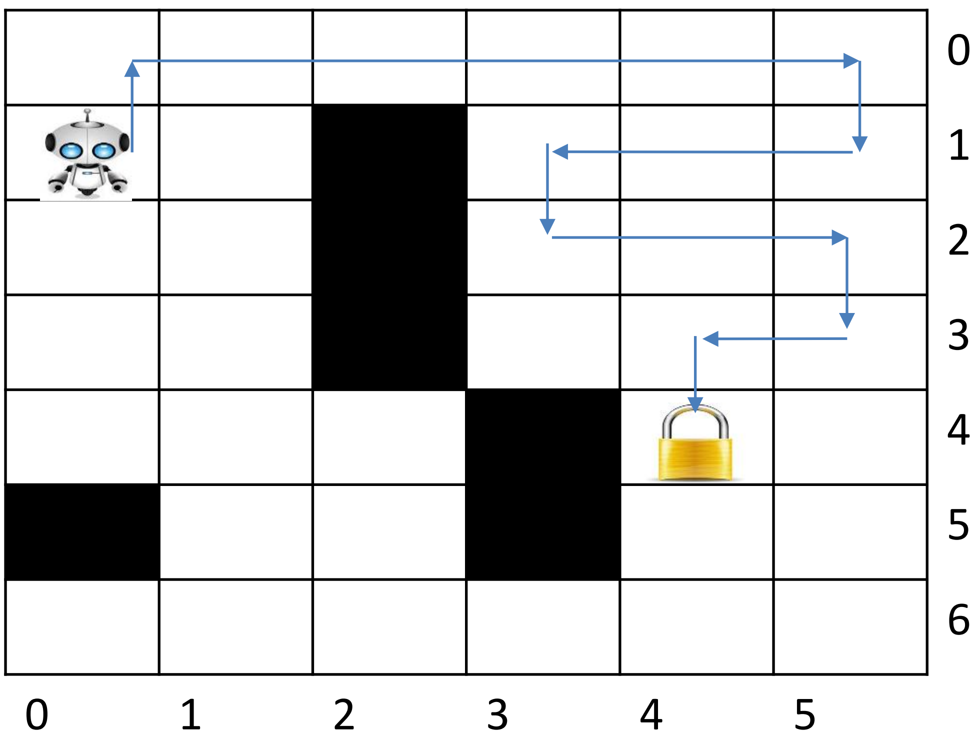
BFS – Lab Example Results – Check your virtual lab

Demo



DFS : – Check your virtual lab & edit to simulate the result

Demo



Terminologies – Learnt Today

- Nodes
- States
- Frontier | Fringes
- Search Strategy : LIFO | FIFO | Priority Queue
- Performance Metrics
    - Completeness
    - Optimality
    - Time Complexity
    - Space Complexity
- Algorithm Terminology

- d Depth of a node	- m – maximum
- b Branching factor	- C* - Optimal Cost
- n – nodes	- E – least Cost
- l – level of a node	- N –total node generated

Required Reading: AIMA - Chapter #3: 3.1, 3.2, 3.3, 3.4

Next Class Plan :
Informed Search : GFBS & A*
Heuristic Design

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

BITS Pilani
Pilani Campus

Artificial & Computational Intelligence

AIML CLZG557

M2 : Problem Solving Agent using Search

Raja vadhana P
Assistant Professor,
BITS - CSIS

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

M5 Probabilistic Representation and Reasoning

M6 Reasoning over time

M7 Ethics in AI

Module 2 : Problem Solving Agent using Search

A. Uninformed Search

B. Informed Search

C. Heuristic Functions

D. Local Search Algorithms & Optimization Problems

Learning Objective

At the end of this class , students Should be able to:

1. Differentiate between uninformed and informed search requirements
2. Apply UCS, GBFS & A* algorithms to the given problem
3. Prove if the given heuristics are admissible and consistent
4. Design and compare heuristics apt for given problem
5. Apply A* variations algorithms to the given problem

Problem Formulation

Problem Solving Agents

Goal Formulation

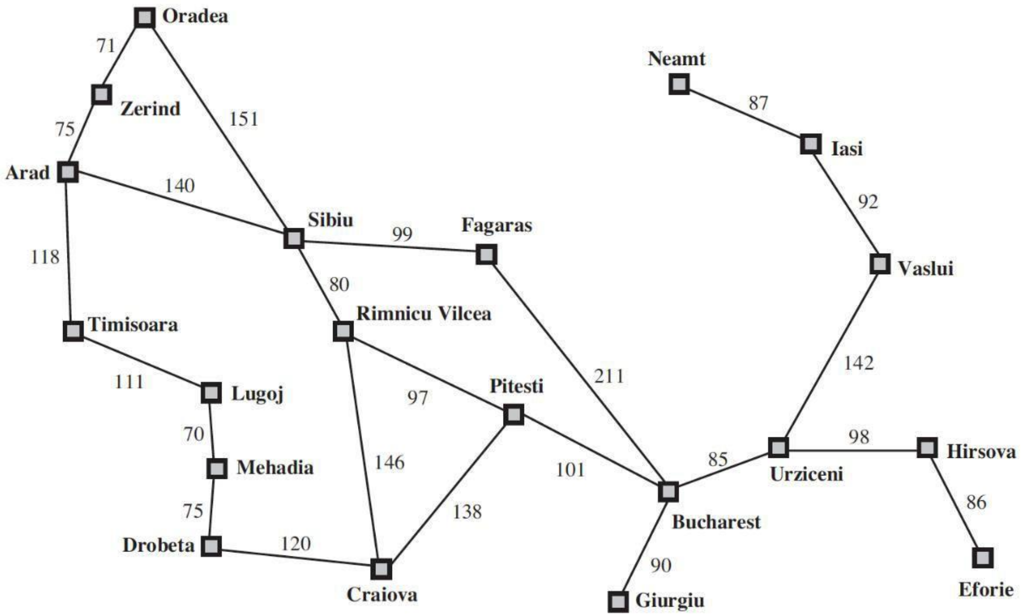
Problem Formulation

Search Phase

Execution Phase

Phases of Solution Search by PSA

Assumptions – Environment :
Static
Observable Discrete
Deterministic

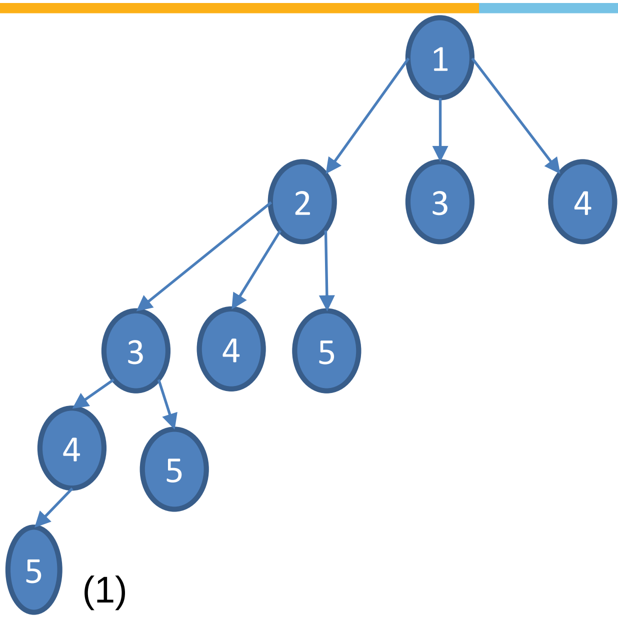


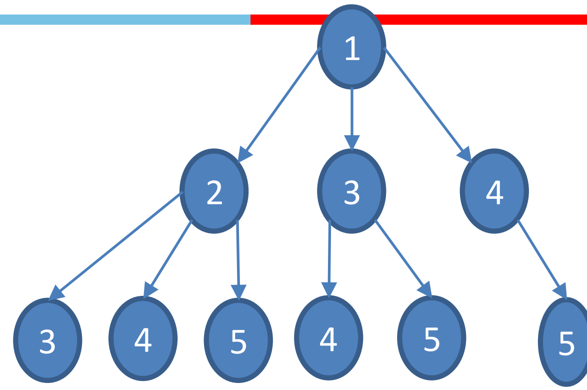
Searching for Solutions

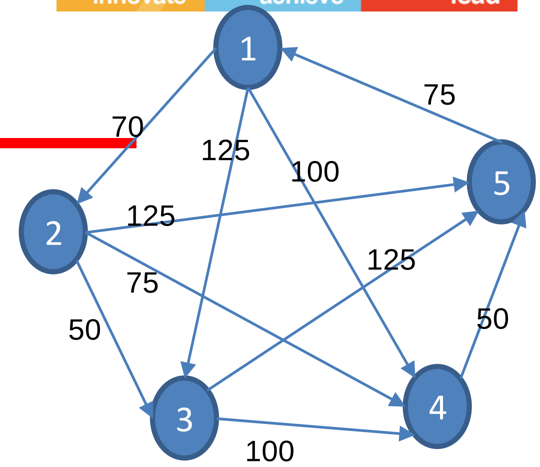
Choosing the current state, testing possible successor function, expanding current state to generate new state is called Traversal. Choice of which state to expand – Search Strategy

- Search Strategy (under certainty)
  - **Uninformed**
    BFS, DFS, UCS
    IDS, DLS
    Bi-Directional
  - **Informed**
    Best First Search
    A*

Search Algorithm – Uninformed Example - 2







(1)
(1 2) (1 3) (1 4)
(1 2 3) (1 2 4) (1 2 5) (1 3) (1 4)
(1 2 3 4) (1 2 3 5) (1 2 4) (1 2 5) (1 3) (1 4)
(1 2 3 4 5) (1 2 3 5) (1 2 4) (1 2 5) (1 3) (1 4)

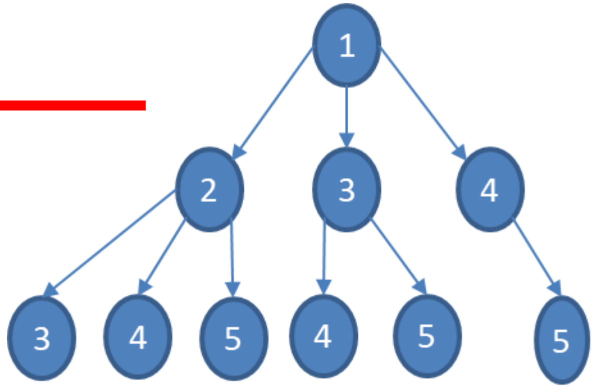
C(1-2-3-4-5) = 70 + 50 + 100 + 50 =270
Expanded : 4
Generated : 10
Max Queue Length : 6

(1)
(1 2) (1 3) (1 4)
TEST FAILED

(1 3) (1 4) (1 2 3) (1 2 4) (1 2 5)
(1 2 3) (1 2 4) (1 2 5) (1 3 4) (1 3 5) (1 4 5)
TEST PASSED

C(1-2-5) = 70 + 125 =195
Expanded : 4
Generated : 10
Max Queue Length : 6

Sample Evaluation of the Algorithm



Complete – If the shallowest goal node is at a depth d, BFS will eventually find it by generating all shallower nodes

Optimal – Not necessarily. Optimal if path cost is non-decreasing function of depth of node. E.g., all actions have same cost

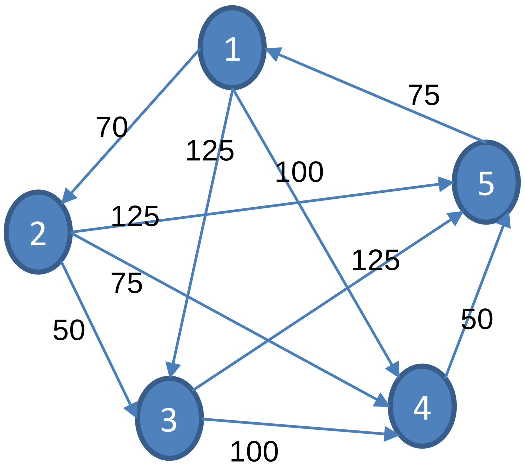
Time Complexity – $\mathcal{O}(b^d)$ b - branching factor, d – depth

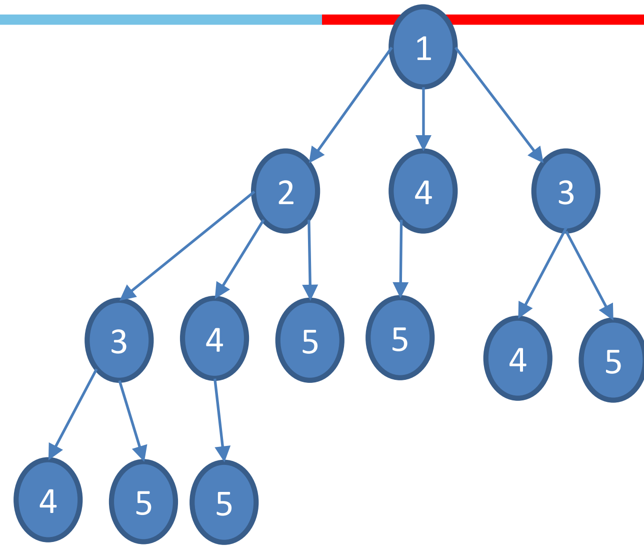
- Nodes expanded at depth 1 = b
- Nodes expanded at depth 2 = b^2
- Nodes expanded at depth d = b^d
- Goal test is applied during generation, time complexity would be $\mathcal{O}(b^{d+1})$

Space Complexity – $\mathcal{O}(b^d)$

- $\mathcal{O}(b^{d-1})$ in explored set
- $\mathcal{O}(b^d)$ in frontier set

UCS





(1)

(1 2 : 70) (1 4 : 100) (1 3 : 125)

TEST-F

(1 4 : 100) (1 2 3 :120)(1 3 : 125) (1 2 4 : 145) (1 2 5 : 195)

TEST-F

(1 2 3 :120)(1 3 : 125) (1 2 4 : 145) (1 4 5: 150)(1 2 5 : 195)

TEST-F

(1 3 : 125) (1 2 4 : 145) (1 4 5: 150) (1 2 3 4:170) (1 2 5 : 195) (1 2 3 5 : 245)

TEST-F

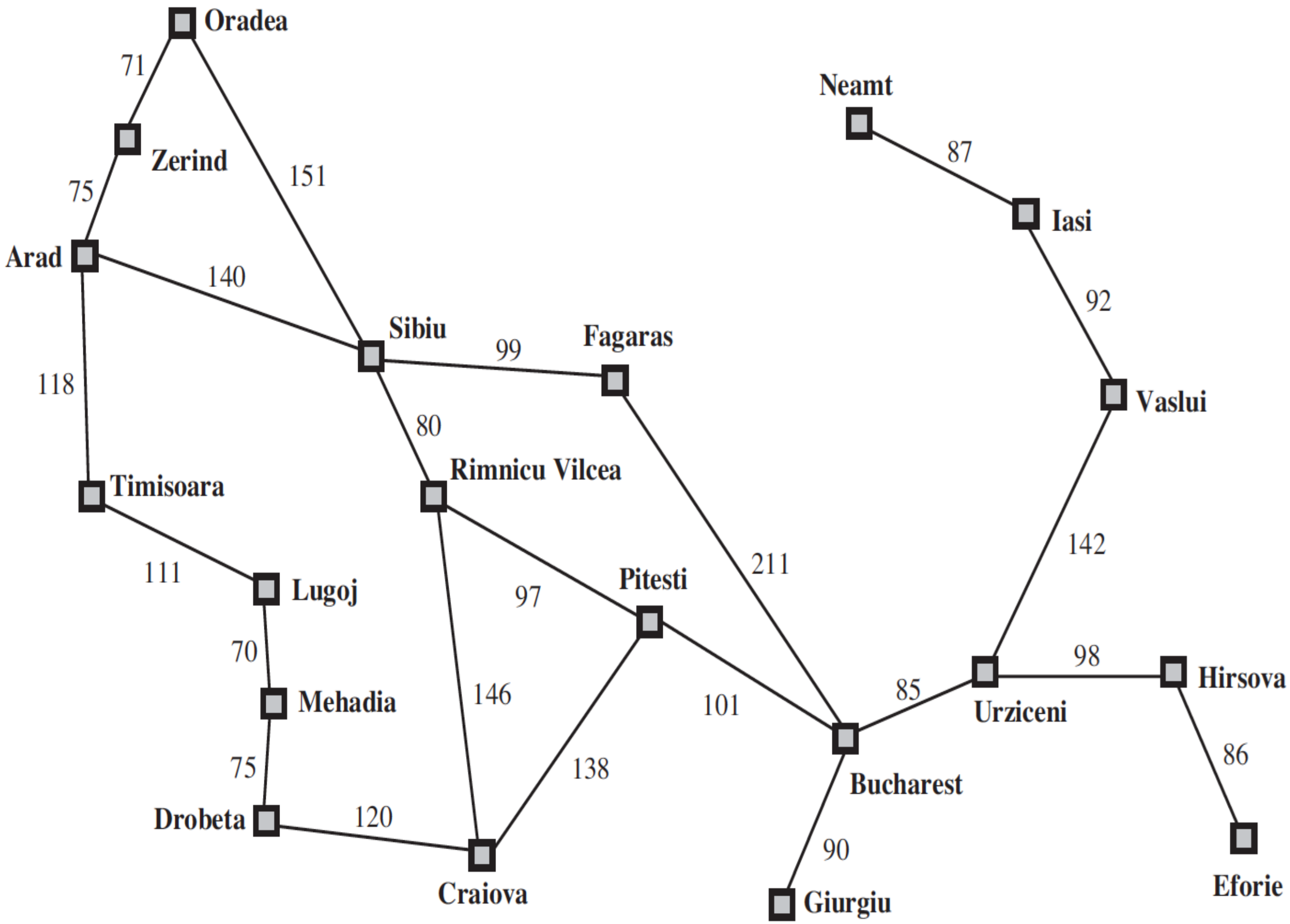
(1 2 4 : 145) (1 4 5: 150) (1 2 3 4:170) (1 2 5 : 195) (1 3 4 : 225) (1 2 3 5 : 245) (1 3 5 : 250)

TEST-F

(1 4 5: 150) (1 2 3 4:170) (1 2 4 5 : 195) (1 2 5 : 195) (1 3 4 : 225) (1 2 3 5 : 245) (1 3 5 : 250)

TEST - P

Tree Search Vs Graph Search


Oradea
71
Zerind
151
75
Arad
140
118
Sibiu
99
Fagaras
80
Rimnicu Vilcea
Timisoara
111
Lugoj
97
Pitesti
211
70
Mehadia
146
101
75
138
Drobeta
120
Craiova
Neamt
87
Iasi
92
Vaslui
142
98
Urziceni
Hirsova
85
86
Bucharest
90
Giurgiu
Eforie

Coding Aspects

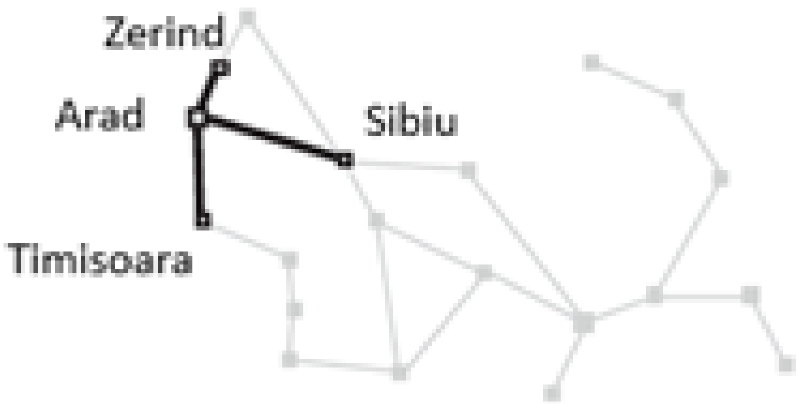
For each node n of the tree,

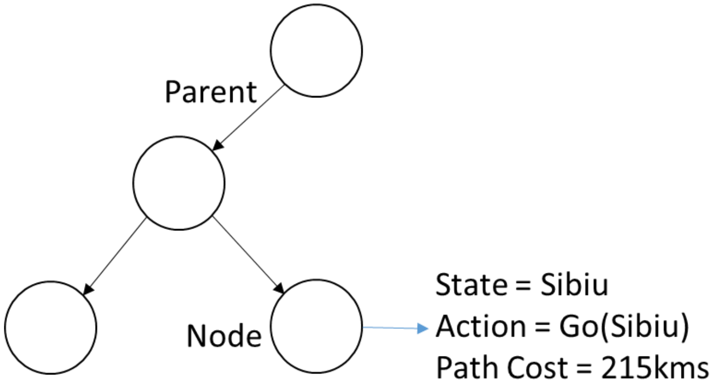
n.STATE : the state in the state space to which node corresponds

n.PARENT : the node in the search tree that generated this node

n.ACTION : the action that was applied to parent to generate the node

n.PATH-COST : the cost, denoted by g(n), of the path from initial state to node





Algorithm Tracing

Students must follow this in the exams for all the search algorithms in addition to the search tree constructions. The ordering of the Open Lists must be in consistent with the algorithm with a note on the justification of the order expected!

Iter	Open List / Frontiers / Fringes	Goal Test
1.	(1)	Fail on (1)
2.	(1 3), (1 4), (1 2)	Fail on (1 3)

Tree Search Algorithms

```
function Tree-Search (problem, strategy) returns a solution, or failure
        initialize the search tree using the initial state of problems
        loop do
                if there are no candidate for expansion
                        then return failure
                choose: leaf node for expansion according to strategy
                if the node contains a goal state
                        then return the corresponding solution
                else
                        Expand the node
                        Add the resulting nodes to the search tree
        end
```

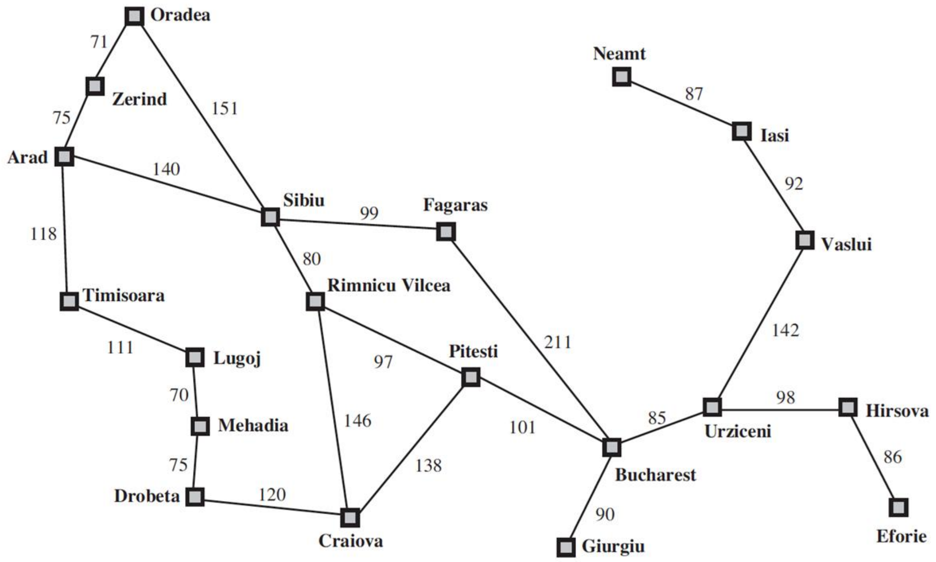
Tree Search Vs Graph Search Algorithms

Coding Aspects

Need:

Redundant Path Problem : More than one way to reach a state from another.

Infinite Loop Path Problem



Start : Arad

Goal : Craiova

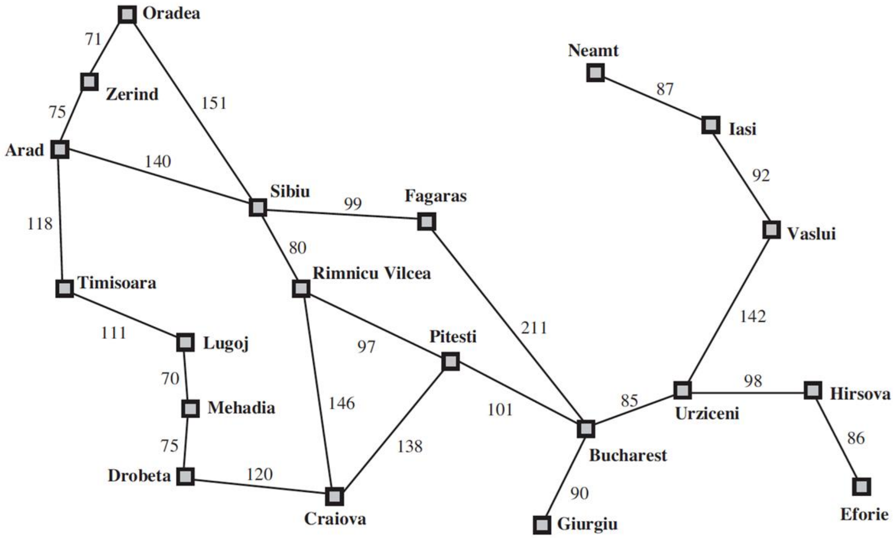
Tree Search Vs Graph Search Algorithms

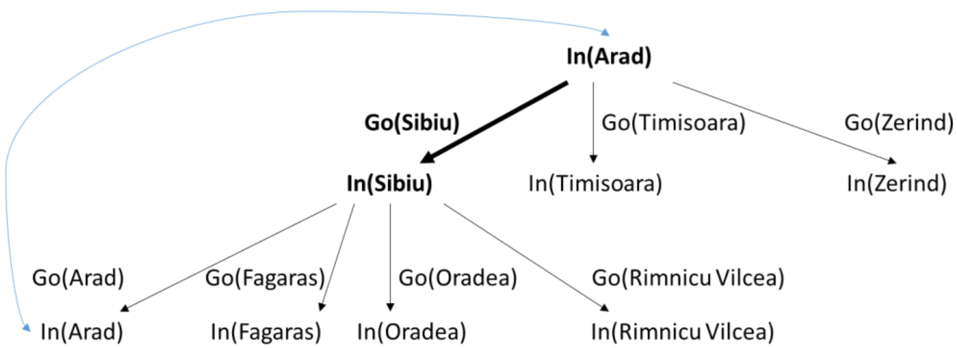
Coding Aspects

Need:

Redundant Path Problem

Infinite Loop Path Problem: Repeated State generated by looped path existence.





Start : Arad

Goal : Craiova

Algorithm Tracing

Students must follow this in the exams for all the search algorithms in addition to the search tree constructions. The ordering of the Open Lists must be in consistent with the algorithm with a note on the justification of the order expected!

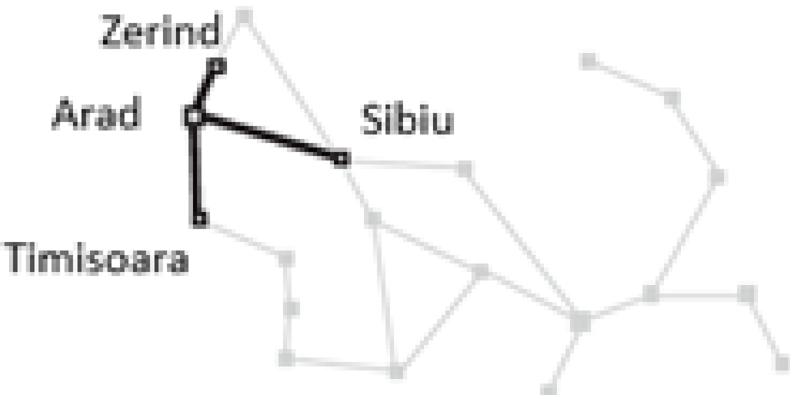
Iter	Open List / Frontiers / Fringes	**Closed List**	Goal Test
1.	(1)		Fail on (1)
2.	(1 3), (1 4), (1 2)	(1)	Fail on (1 3)

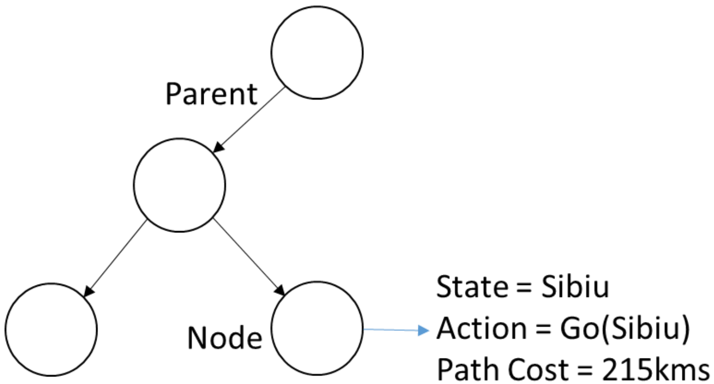
innovate achieve lead

Coding Aspects

For each node n of the tree,

- **n.STATE** : the state in the state space to which node corresponds
- **n.PARENT** : the node in the search tree that generated this node
- **n.ACTION** : the action that was applied to parent to generate the node
- **n.PATH-COST** : the cost, denoted by g(n), of the path from initial state to node
- **n.VISITED** **:** the boolean indicating if the node is already visited and tested (**or**) a global SET of visited nodes



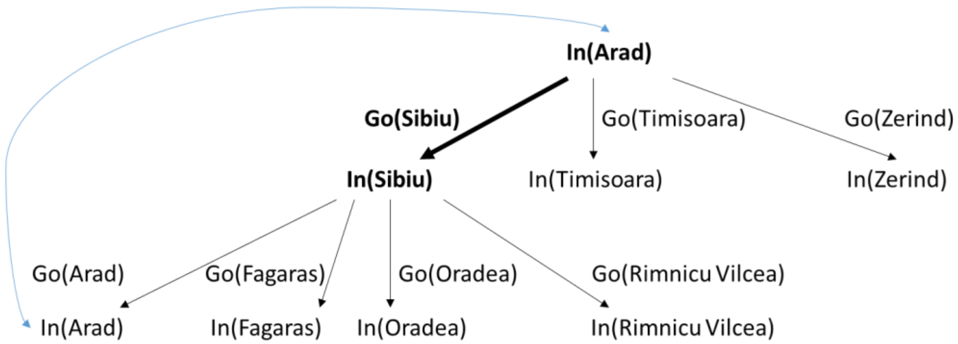


Tree Search Vs Graph Search Algorithms

Coding Aspects

Graph-Search Algorithm

Augments the Tree-Search algorithm to solve redundancy by keeping track of states that are already visited called as **Explored Set**. **Only one copy of each state is maintained/stored.**



Graph Search Algorithms

```
function Graph-Search (problem, fringe) returns a solution, or failure
        initialize the search space using the initial state of problems memory to store
the visited fringe
        closed ⍰  an empty set
        fringe ⍰  Insert(Make-Node(Initial-State[problem]), fringe)
        loop do
                if fringe is empty
                        then return failure
                node⍰  Remove-Front(fringe)
                if the node contains a goal state
                        then return the corresponding solution
                else
                        if the node is not in closed ie., not visited yet
                           Add the node to the closed set
                           Expand all the fringe of the node
                           Add all expanded sorted successors into the fringe
        end
```

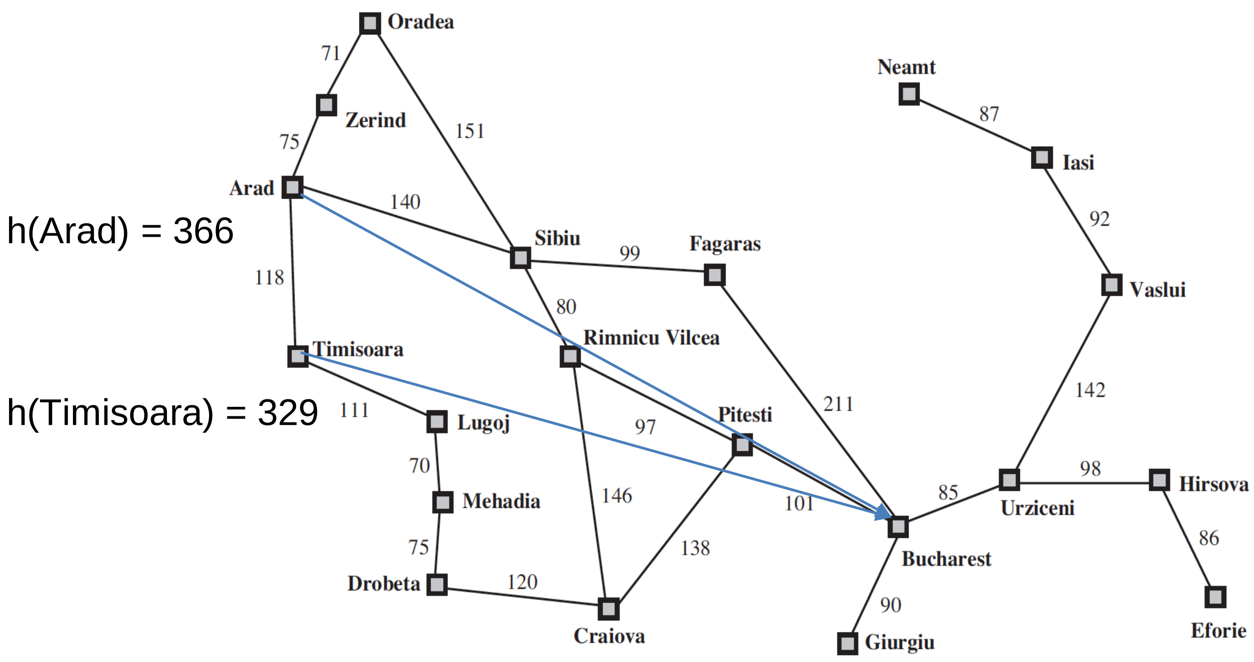
Informed Search

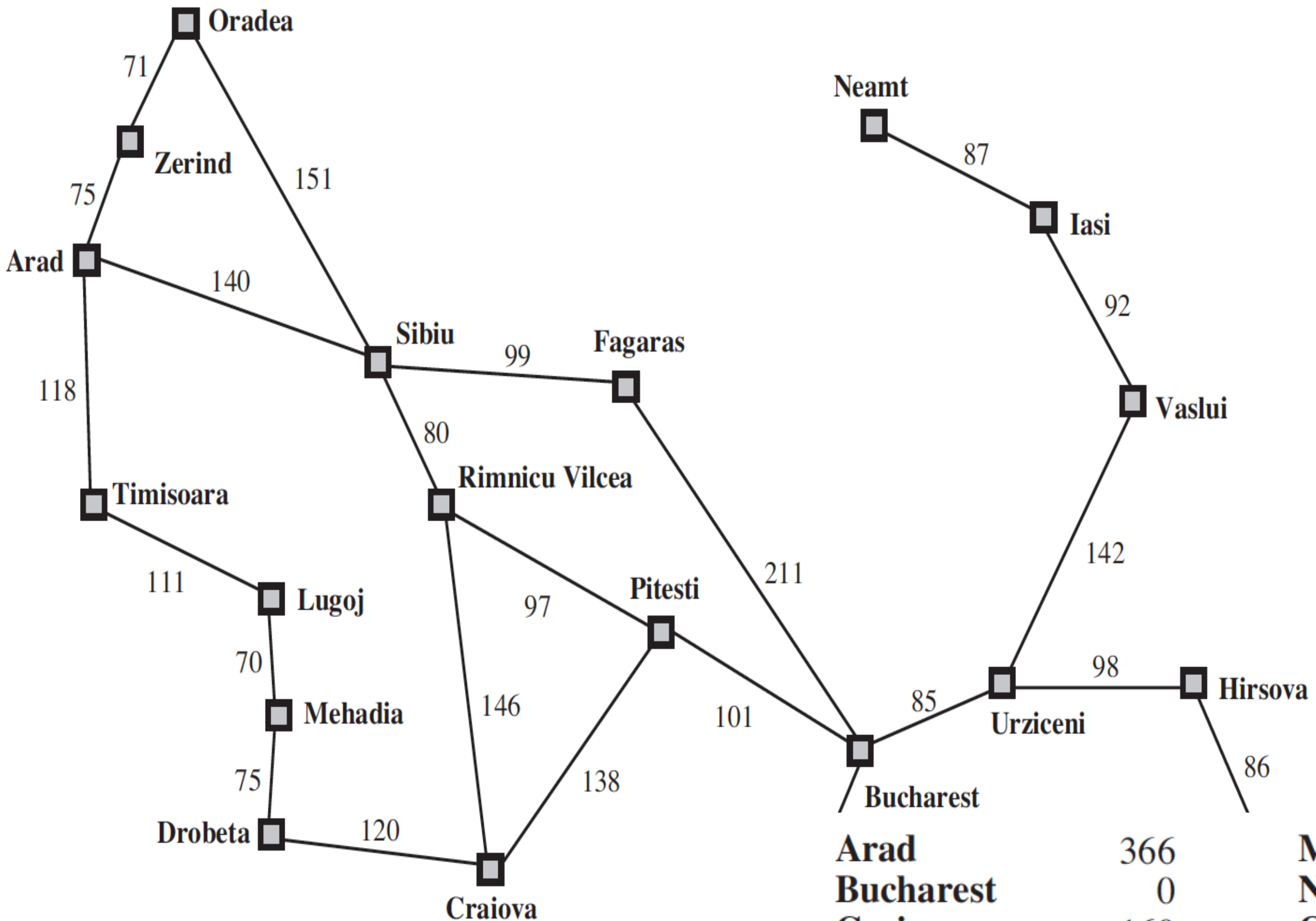
Greedy Best First

A*

Informed /Heuristic Search

Strategies that know if one non-goal state is more promising than another non-goal state





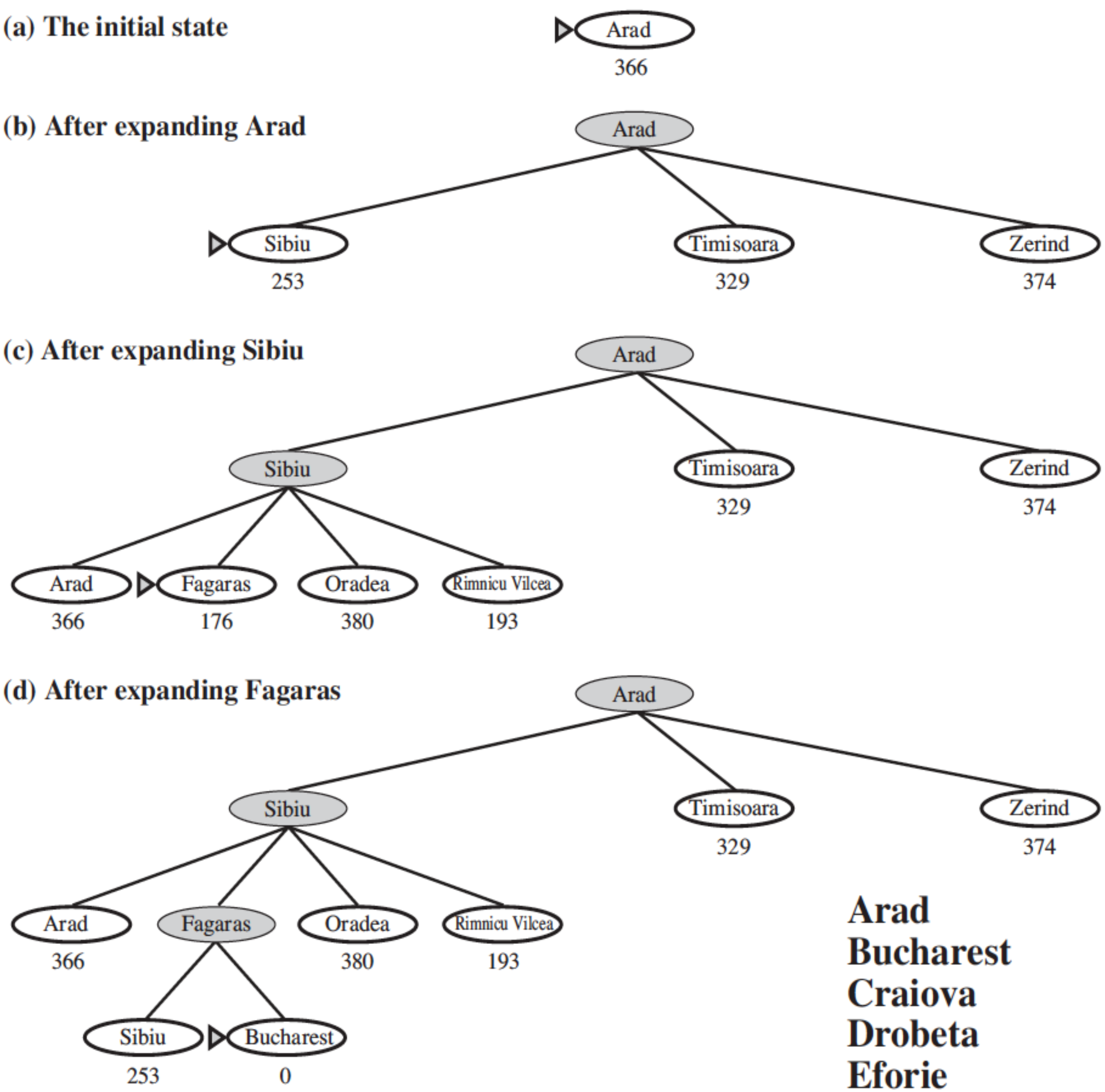
Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Drobeta	242	**Pitesti**	100
Eforie	161	**Rimnicu Vilcea**	193
Fagaras	176	**Sibiu**	253
Giurgiu	77	**Timisoara**	329
Hirsova	151	**Urziceni**	80
Iasi	226	**Vaslui**	199
Lugoj	244	**Zerind**	374

Greedy Best First Search

Expands the node that is closest to the goal

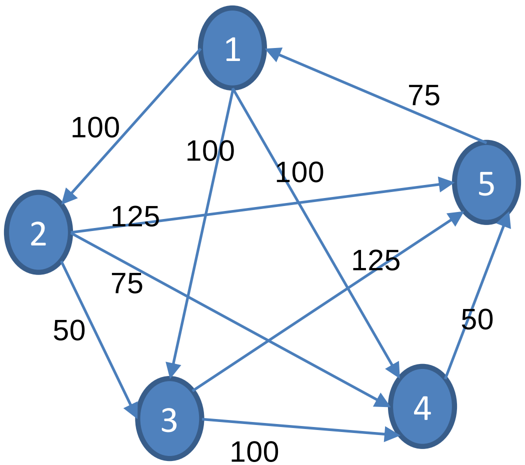
Thus, **f(n) = h(n)**

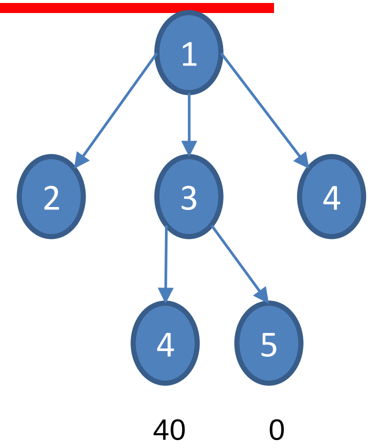
Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Drobeta	242	**Pitesti**	100
Eforie	161	**Rimnicu Vilcea**	193
Fagaras	176	**Sibiu**	253
Giurgiu	77	**Timisoara**	329
Hirsova	151	**Urziceni**	80
Iasi	226	**Vaslui**	199
Lugoj	244	**Zerind**	374



Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Drobeta	242	**Pitesti**	100
Eforie	161	**Rimnicu Vilcea**	193
Fagaras	176	**Sibiu**	253
Giurgiu	77	**Timisoara**	329
Hirsova	151	**Urziceni**	80
Iasi	226	**Vaslui**	199
Lugoj	244	**Zerind**	374

Greedy Best First Search



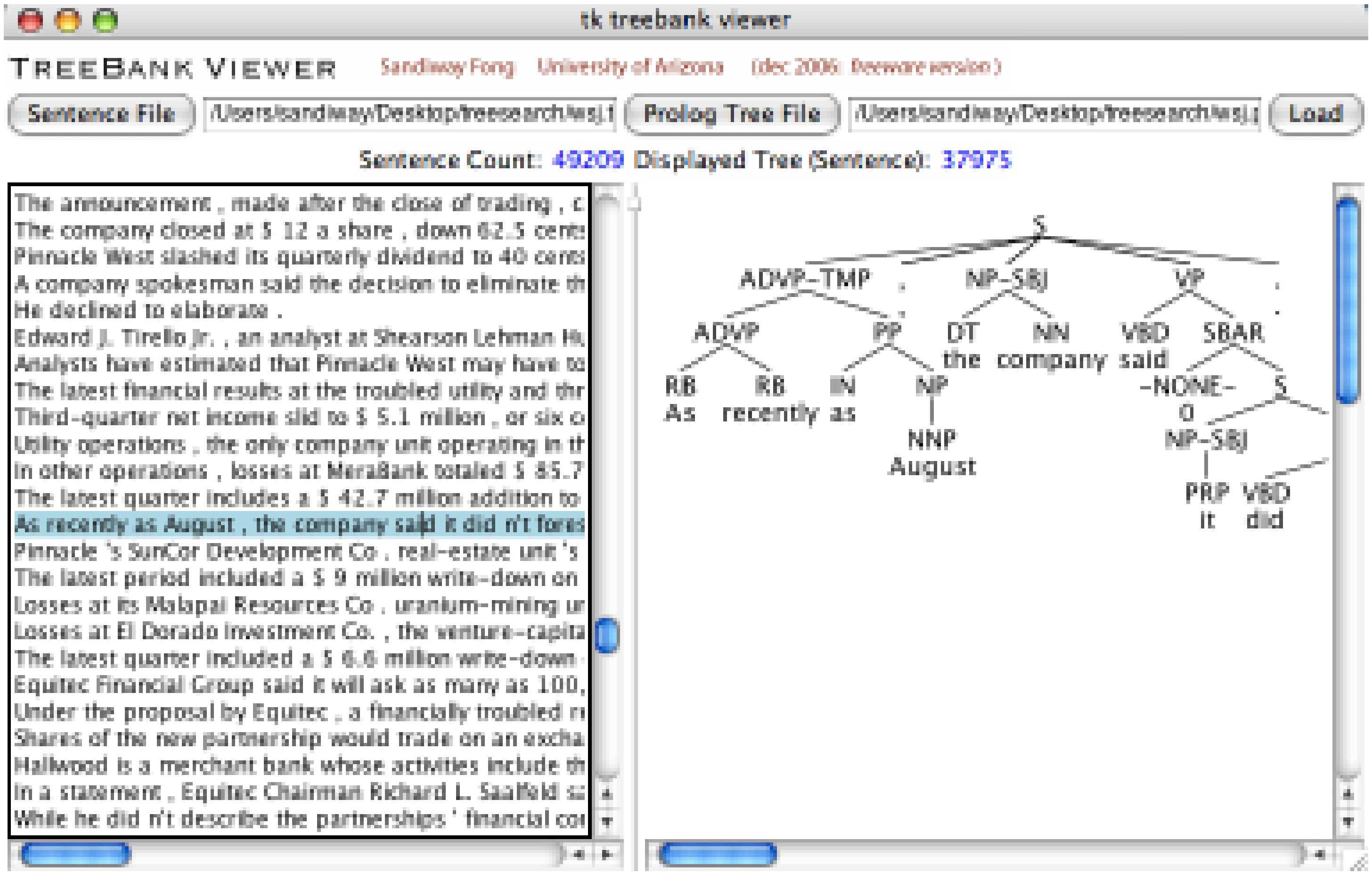


n	h(n)
1	60
2	120
3	30
4	40
5	0

(1)
(1 3) (1 4) (1 2)
(1 3 5) (1 3 4)

C(1-3-5) = 100 + 125 =225
Expanded : 2
Generated : 6
Max Queue Length : 3

Case Study – 1 Search in Treebanks

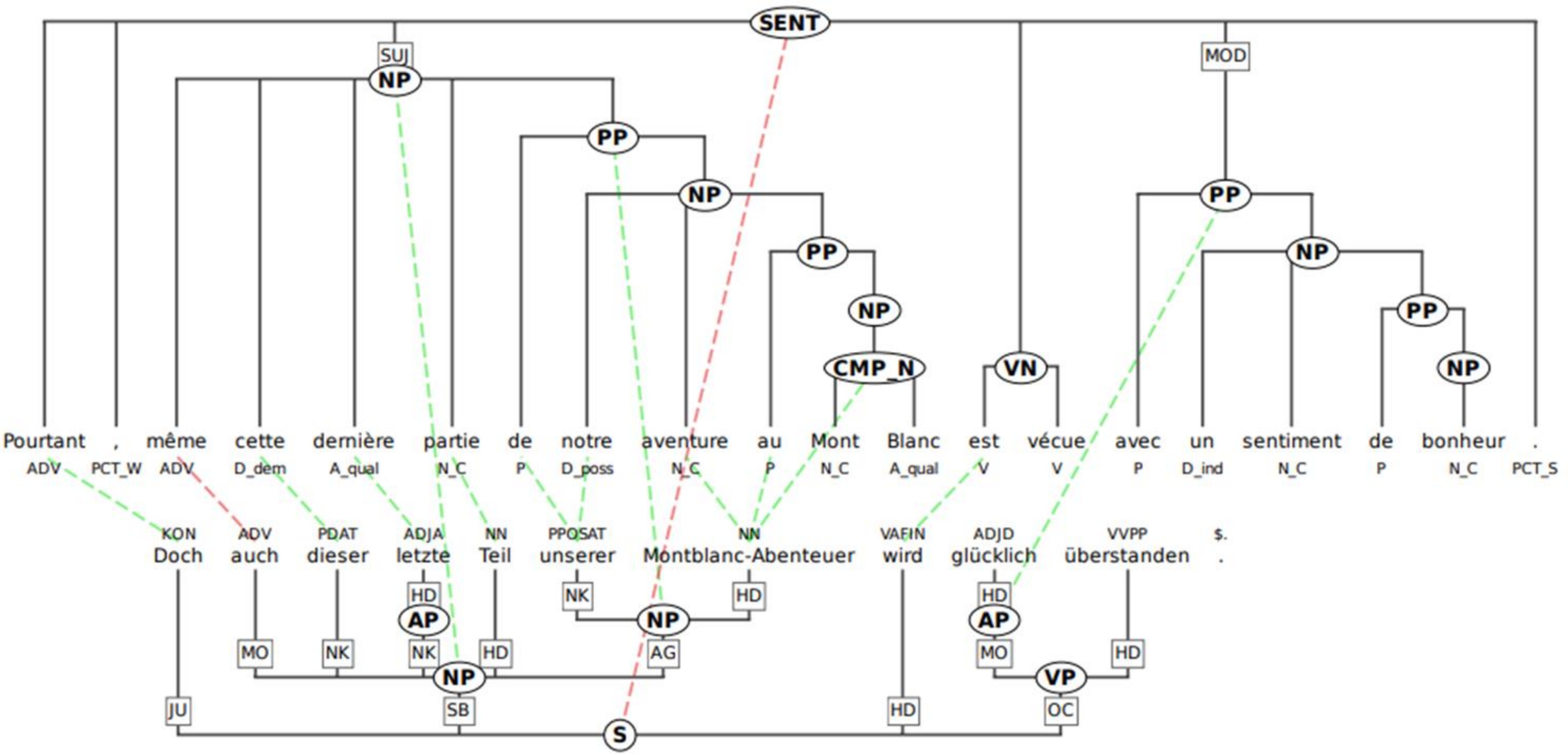


Source Credit :

https://catalog.ldc.upenn.edu/docs/LDC95T7/cl93.html

https://ufal.mff.cuni.cz/pdt3.5

Case Study – 1 Search in Treebanks



Source Credit :

https://catalog.ldc.upenn.edu/docs/LDC95T7/cl93.html

https://ufal.mff.cuni.cz/pdt3.5

A* Search

Expands the node which lies in the closest path (estimated cheapest path) to the goal

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

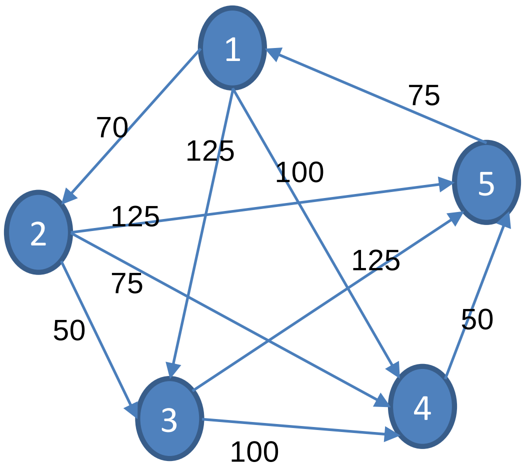
g(n) – the cost to reach the node

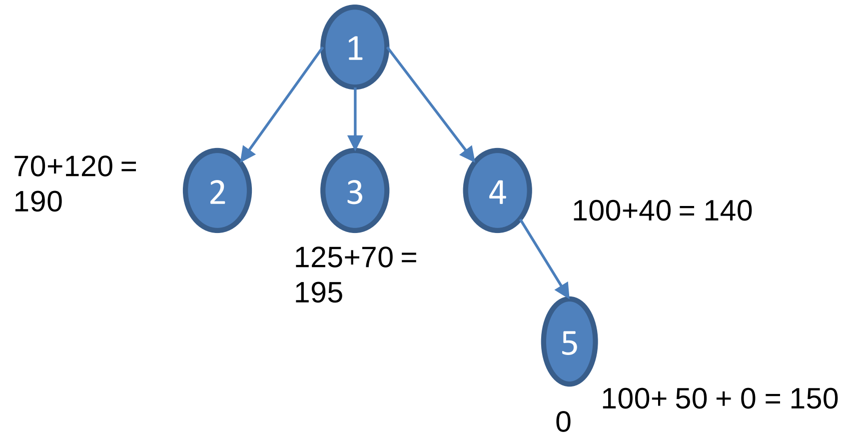
h(n) – the expected cost to go from node to goal

f(n) – estimated cost of cheapest path through node n

Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Drobeta	242	**Pitesti**	100
Eforie	161	**Rimnicu Vilcea**	193
Fagaras	176	**Sibiu**	253
Giurgiu	77	**Timisoara**	329
Hirsova	151	**Urziceni**	80
Iasi	226	**Vaslui**	199
Lugoj	244	**Zerind**	374

A* Search



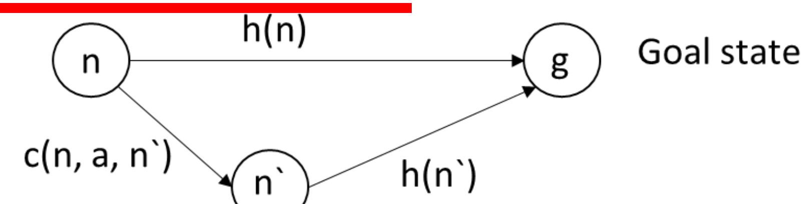


n	h(n)
1	60
2	120
3	70
4	40
5	0

(1)
(1 4) (1 2) (1 3)
(1 4 5) (1 2) (1 3)

C(1-4-5) = 100 + 150 =150
Expanded : 2
Generated : 5
Max Queue Length : 3

A* Search



<u>Optimal on condition</u>

h(n) must satisfies two conditions:

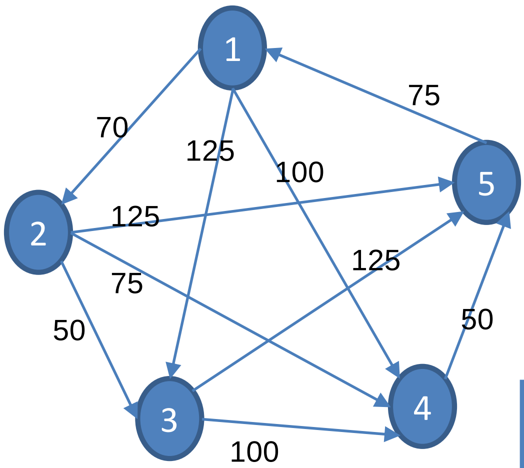
- Admissible Heuristic – one that never overestimates the cost to reach the goal
- Consistency – A heuristic is consistent if for every node n and every successor node n` of n generated by action a,  h(n) <= c(n, a, n`) + h(n`)

<u>Complete</u>

- If the number of nodes with cost <=C* is finite
- If the branching factor is finite
- A* expands no nodes with f(n) > C*, known as pruning

<u>Time Complexity</u> - $\mathcal{O}\left(b^{\Delta}\right)$ where the absolute error $\Delta = h^{*} - h$

Is the heuristic designed leads to optimal solution?

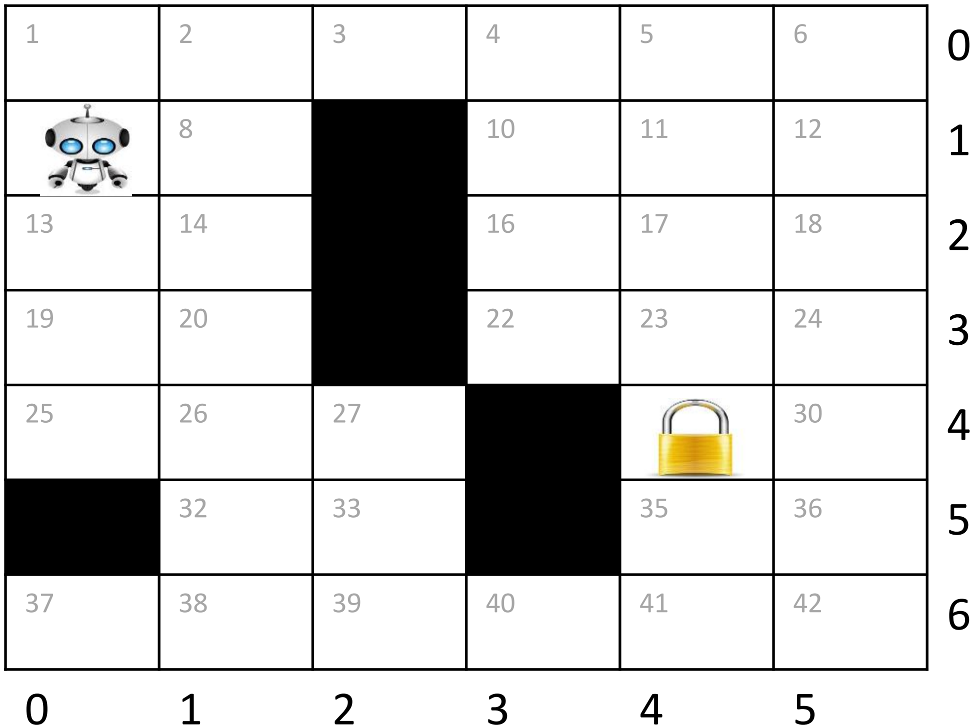


Assuming node 3 as goal, taking only sample edges per node below is checked for consistency

n	h(n)	Is Admissible? h(n) <= h*(n)	Is Consistent? For every arc (i,j): h(i) <= g(i,j) + h(j)
1	80	Y	N (5→1) : 190<=155
2	60	N	Y (1→2) : 80<=130
3	0	Y	
4	200	Y	Y (1→4) : 80<=300 Y (2→4) : 60<=275
5	190	Y	Y (2→5) : 60<=315 Y (4→5) : 200<=240

Path finding Robot – Sample Planning Agent Design

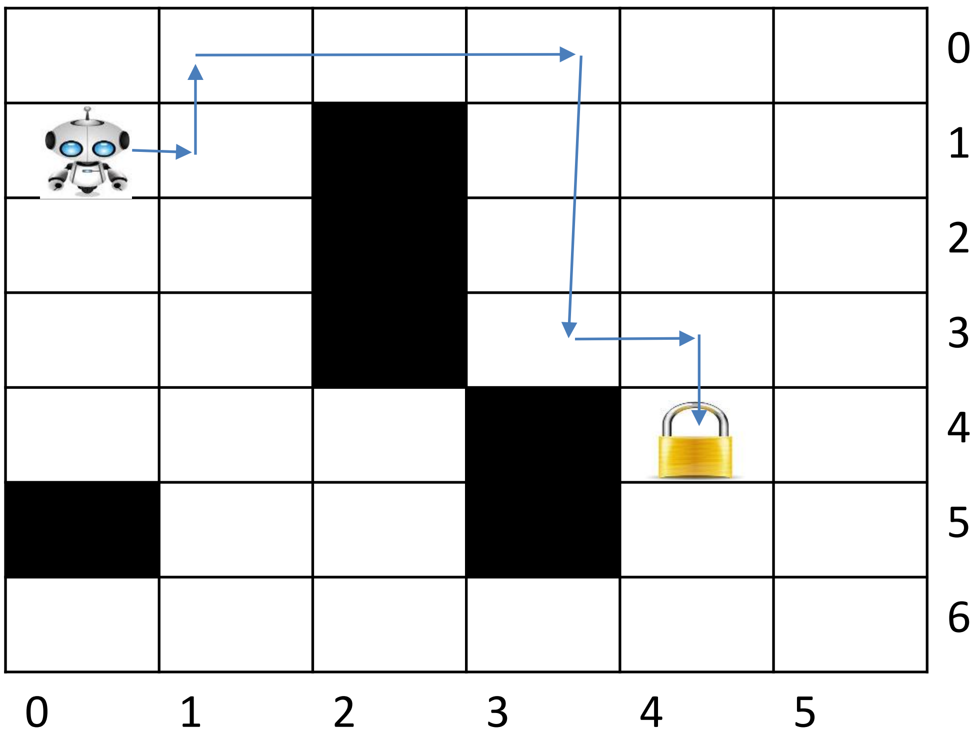
Successor Function Design



N-W-E-S

A*

Demo

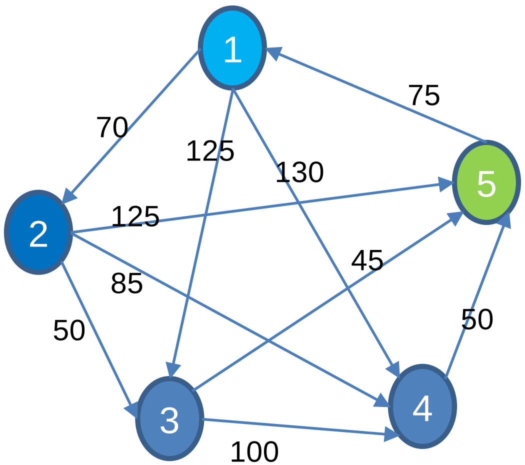


Variations of A*

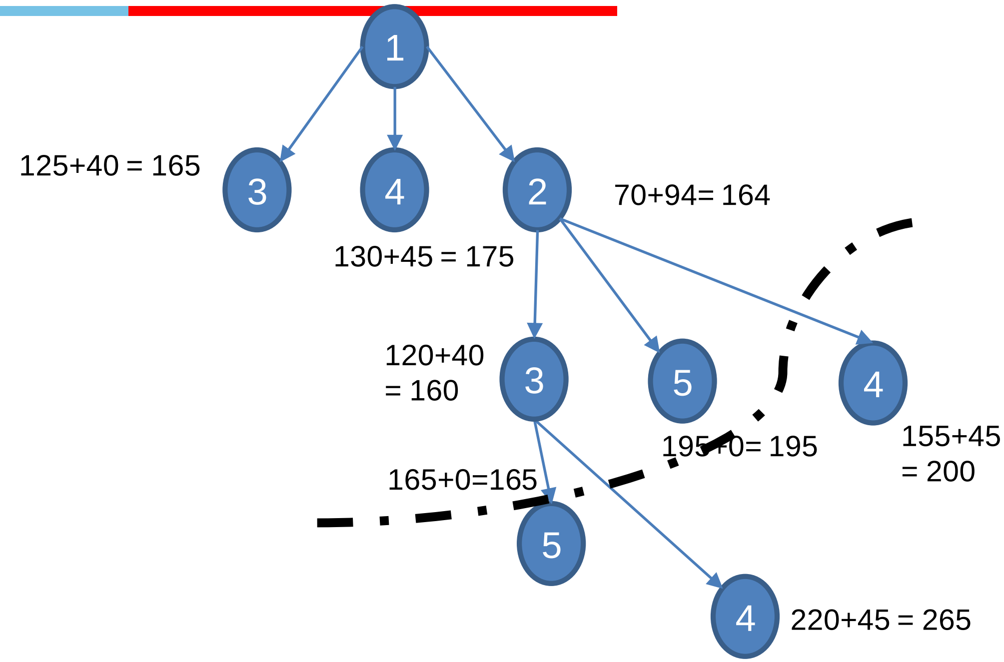
Memory Bounded Heuristics

Iterative Deepening A*

Set limit for f(n)



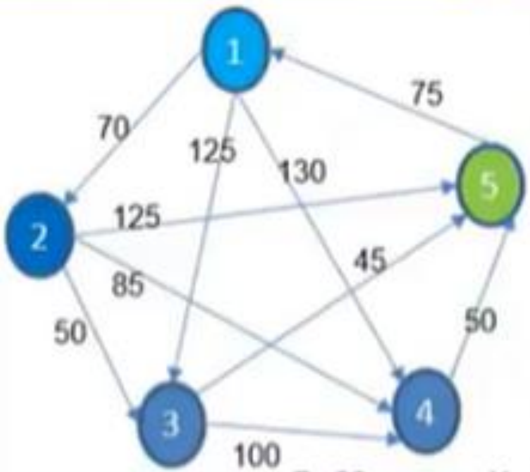
n	h(n)
1	60
2	94
3	40
4	45
5	0



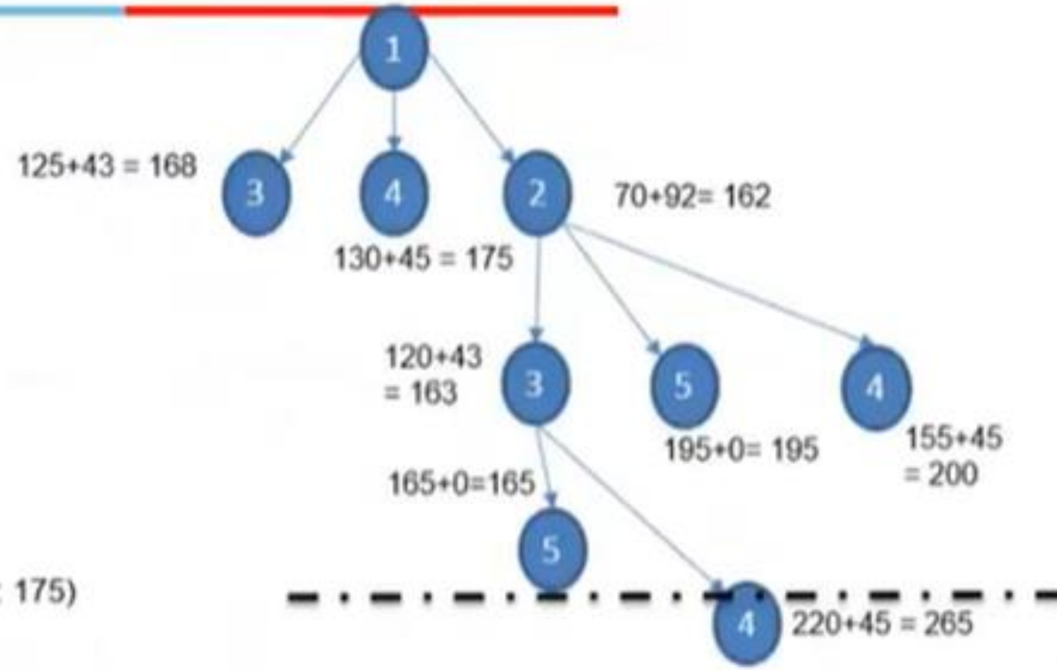
Cut off value is the smallest of f-cost of any node that exceeds the cutoff on previous iterations

Iterative Limit : Eg
f(n) = 180
f(n) = 195
f(n) = 200
.
.
.
.

Iterative Deepening A*



Set limit for f(n)



n	h(n)
1	60
2	92
3	43
4	45
5	0

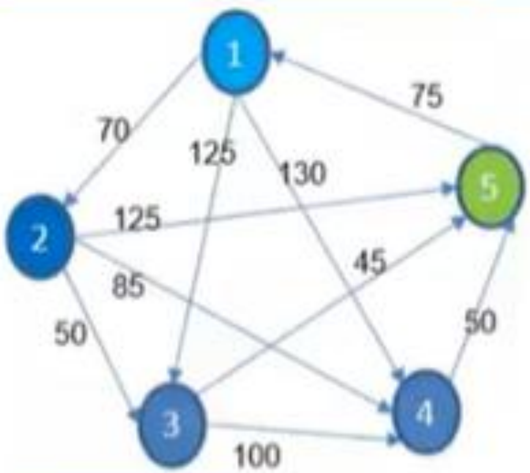
B=60
(1: 60)
TEST-F
(1 2: 162) (1 3: 168) (1 4: 175)

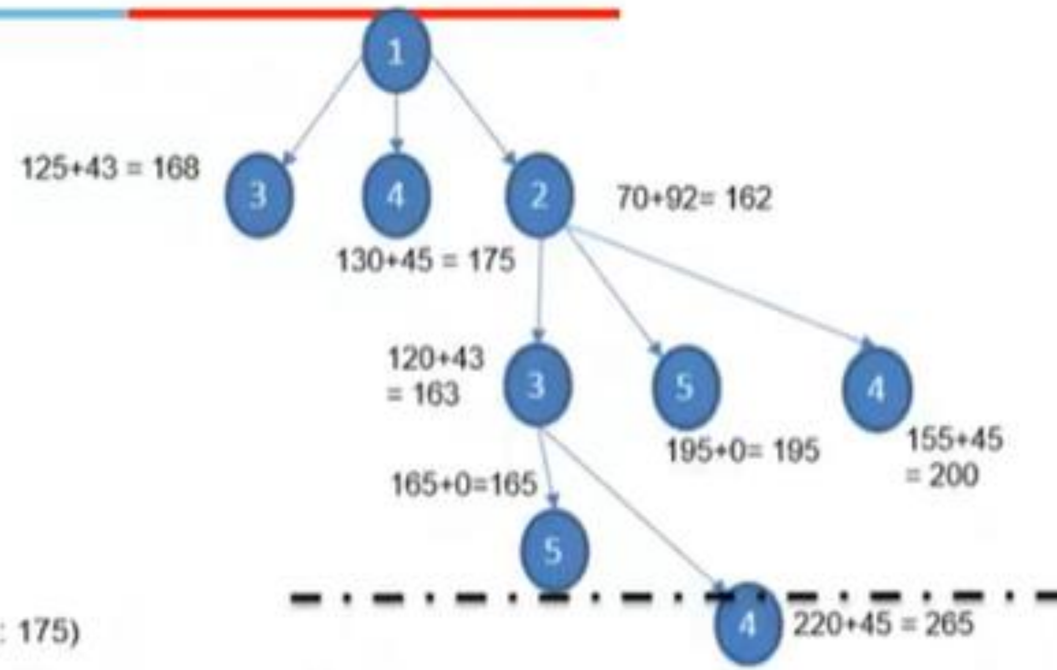
B=162
(1: 60)
TEST-F
(1 2: 162) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3: 163) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)

B=163
(1: 60)
TEST-F
(1 2: 162) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3: 163) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)
TEST-F

Iterative Deepening A*

Set limit for f(n)





n	h(n)
1	60
2	92
3	43
4	45
5	0

B=163

(1: 60)
TEST-F
(1 2: 162) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3: 163) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3 5: 165) (1 2 3 4: 265) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)

B=165

(1: 60)
TEST-F
(1 2: 162) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3: 163) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3 5: 165) (1 2 3 4: 265) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)

Required Reading: AIMA - Chapter #1, 2, 3.1, 3.2, 3.3, 3.4, 3.5

Next Class Plan:

- Another algorithm : Variation of A*
- Heuristic Design
- Local Search Algorithms

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

Artificial & Computational Intelligence

AIML CLZG557

M2 : Problem Solving Agent using Search

Raja vadhana P

Assistant Professor,

BITS - CSIS

BITS Pilani

Pilani Campus

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

M5 Probabilistic Representation and Reasoning

M6 Reasoning over time

M7 Ethics in AI

Module 2 : Problem Solving Agent using Search

A. Uninformed Search

B. Informed Search

C. Heuristic Functions

D. Local Search Algorithms & Optimization Problems

Learning Objective

At the end of this class , students Should be able to:

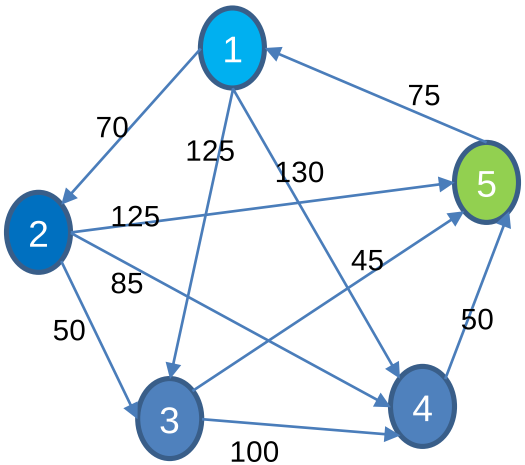
1. Apply A* variations algorithms to the given problem
2. Compare given heuristics for a problem and analyze which is the best fit
3. Differentiate between informed and local search requirements
4. Design relaxed problem with appropriate heuristic design
5. Understand the notion of Local Search algorithms
6. Design fitness function for a problem

Variations of A*

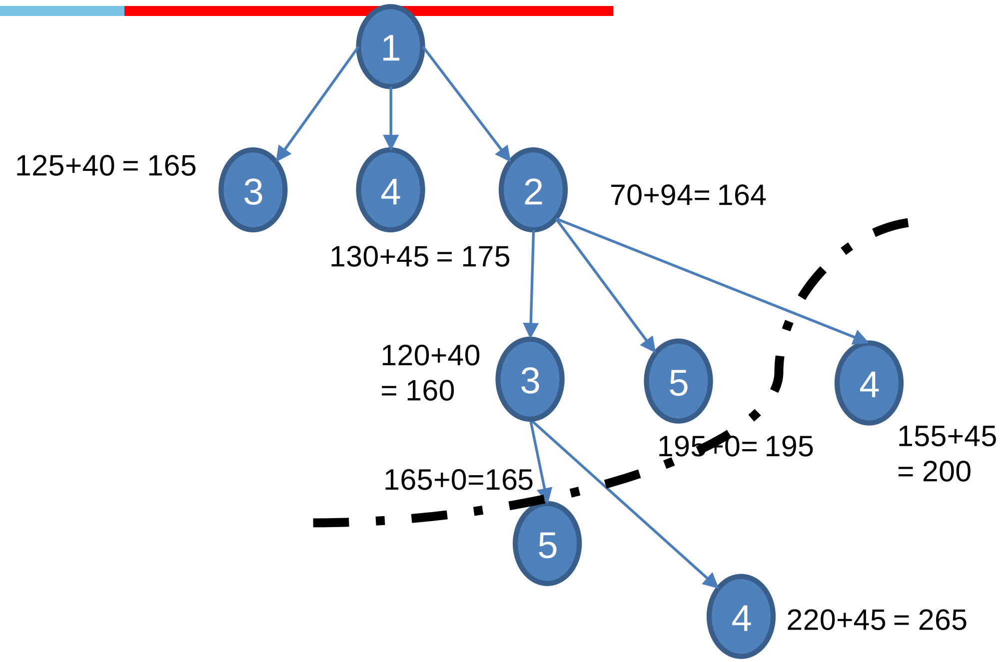
Memory Bounded Heuristics

Iterative Deepening A*

Set limit for f(n)



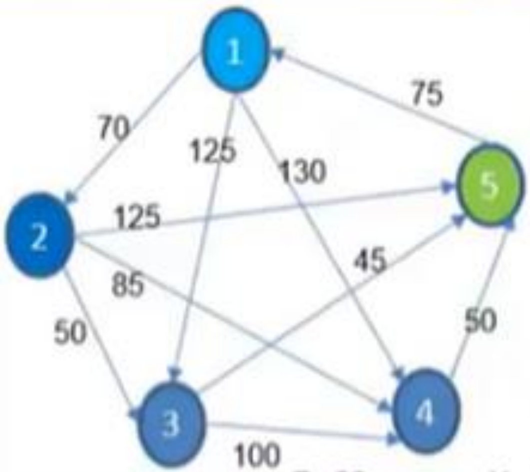
n	h(n)
1	60
2	94
3	40
4	45
5	0

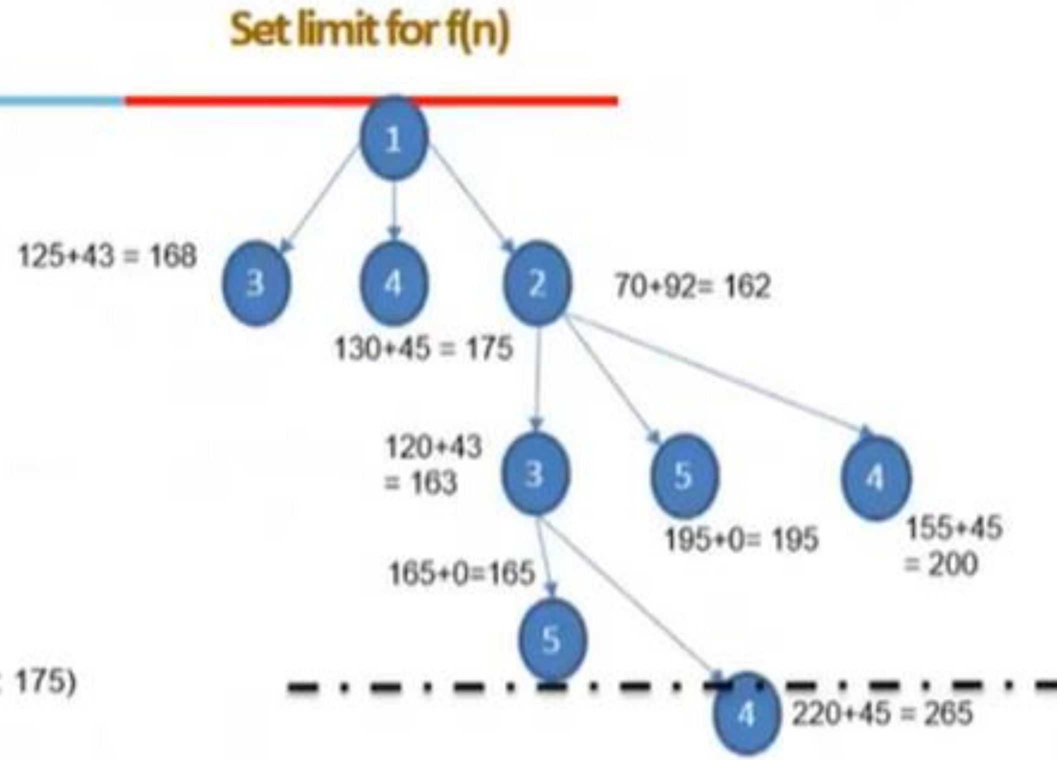


Cut off value is the smallest of f-cost of any node that exceeds the cutoff on previous iterations

Iterative Limit : Eg
f(n) = 180
f(n) = 195
f(n) = 200
.
.
.
.

Iterative Deepening A*





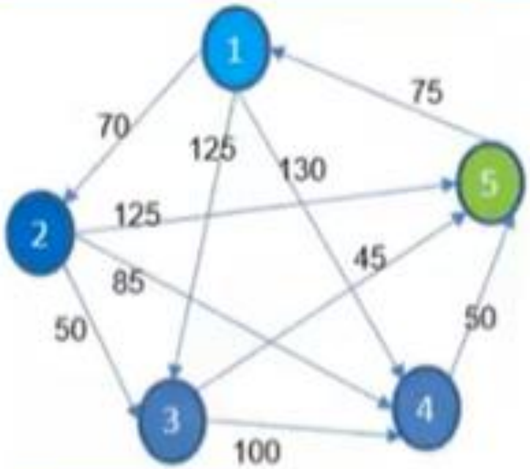
n	h(n)
1	60
2	92
3	43
4	45
5	0

B=60
(1: 60)
TEST-F
(1 2: 162) (1 3: 168) (1 4: 175)

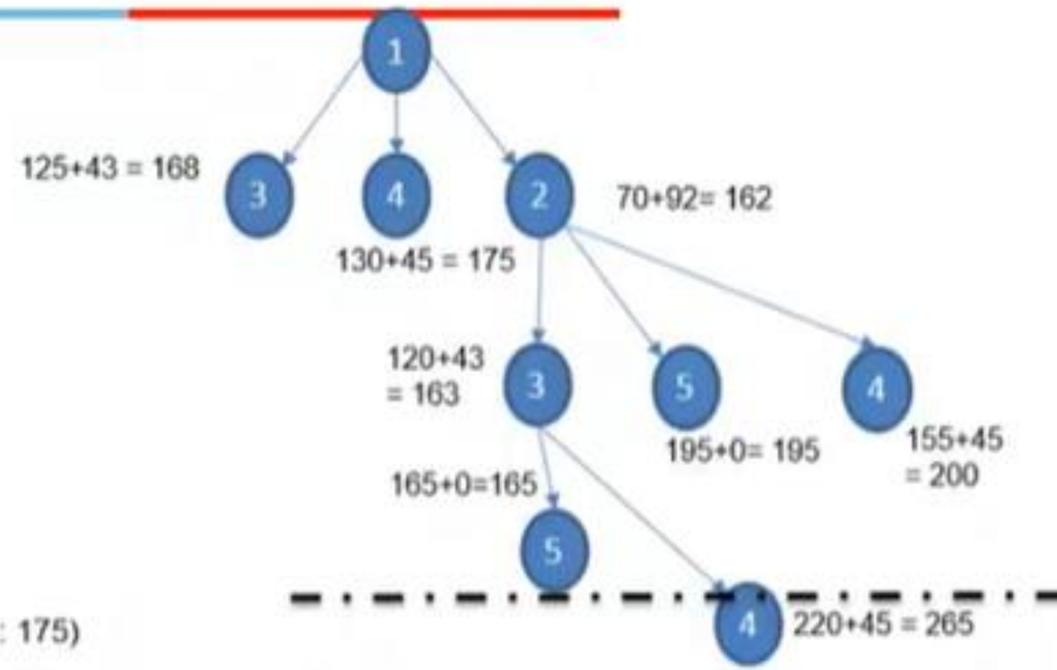
B=162
(1: 60)
TEST-F
(1 2: 162) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3: 163) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)

B=163
(1: 60)
TEST-F
(1 2: 162) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3: 163) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)
TEST-F

Iterative Deepening A*



Set limit for f(n)



n	h(n)
1	60
2	92
3	43
4	45
5	0

B=163

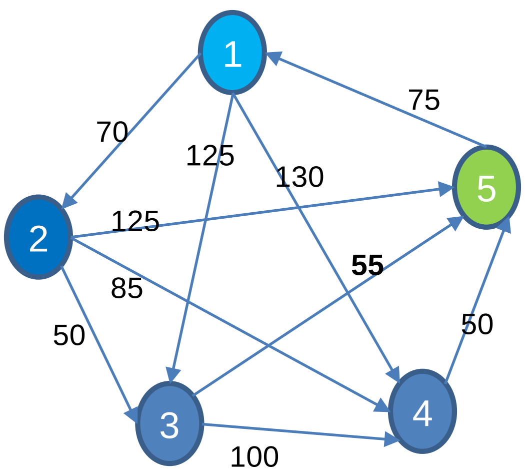
(1: 60)
TEST-F
(1 2: 162) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3: 163) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3 5: 165) (1 2 3 4: 265) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)

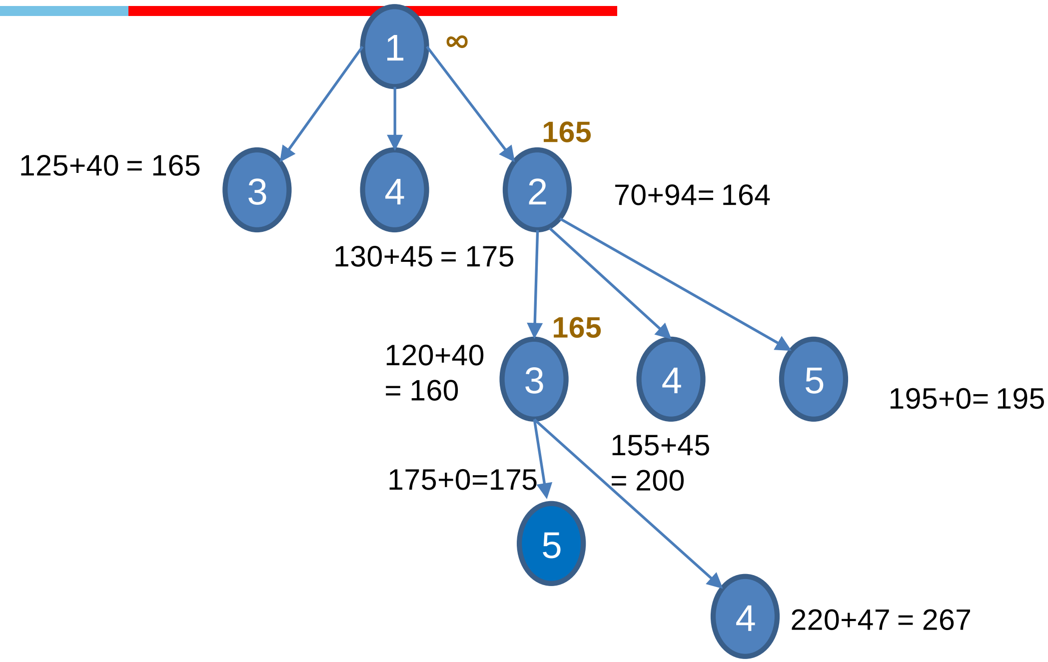
B=165

(1: 60)
TEST-F
(1 2: 162) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3: 163) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)
TEST-F
(1 2 3 5: 165) (1 2 3 4: 265) (1 2 4: 200) (1 2 5: 195) (1 3: 168) (1 4: 175)

Recursive Best First Search A*

Remember the next best alternative f-Cost to regenerate





n	h(n)
1	60
2	94
3	40
4	45
5	0

(1, 60)
(1 2 | 164) (1 3 | 165) (1 4 | 175)

(1 2 | 175) (1 4 | 175) (1 3 | 180)

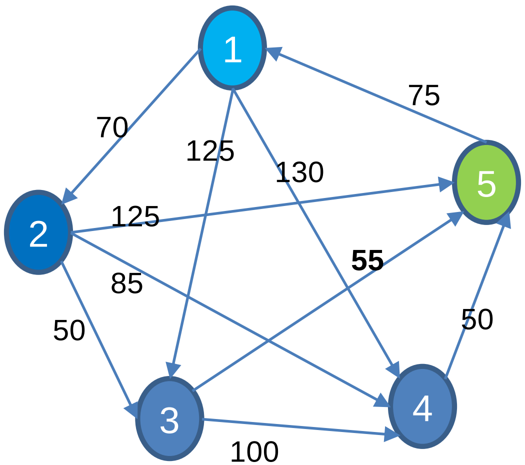
(1 2 3 | 175) (1 4 | 175) (1 3 | 180) (1 2 5 | 195) (1 2 4 | 200)

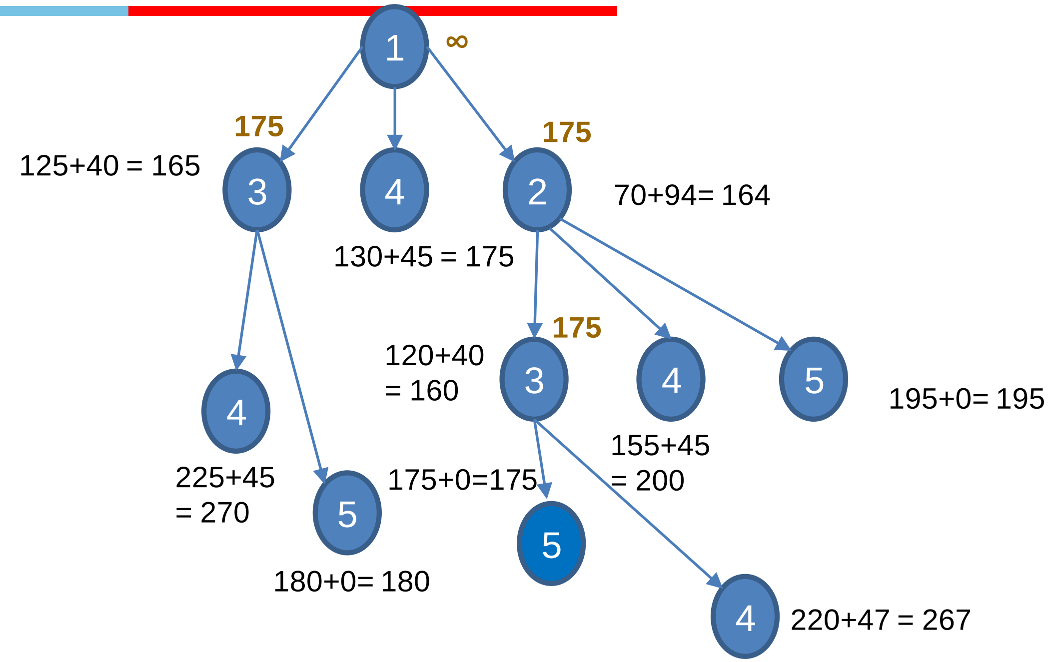
(1 2 3 5 | 175) (1 4 | 175) (1 3 | 180) (1 2 5 | 195) (1 2 4 | 200) (1 2 3 4 | 267)

PASS

Recursive Best First Search A*

Remember the next best alternative f-Cost to regenerate





n	h(n)
1	60
2	94
3	40
4	45
5	0

(1, 60)
(1 2 | 164) (1 3 | 165) (1 4 | 175)

(1 2 | 175) (1 4 | 175) (1 3 | 180)

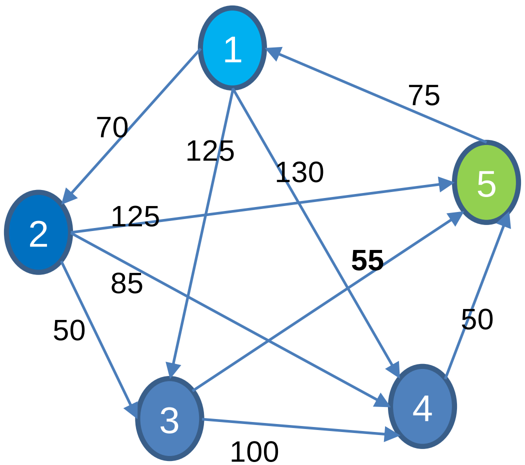
(1 2 3 | 175) (1 4 | 175) (1 3 | 180) (1 2 5 | 195) (1 2 4 | 200)

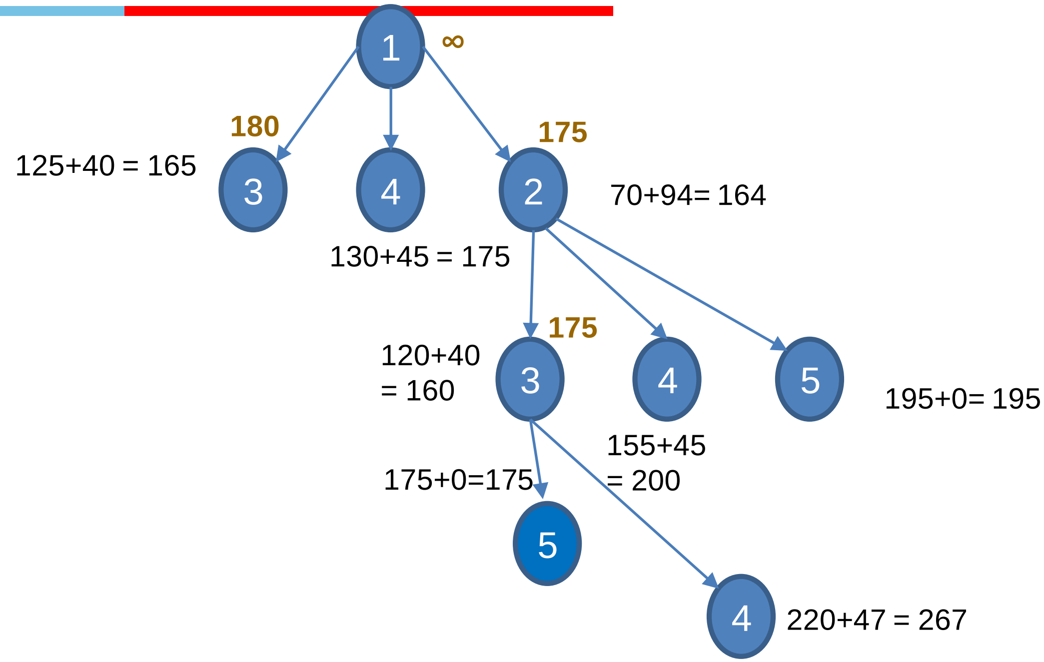
(1 2 3 5 | 175) (1 4 | 175) (1 3 | 180) (1 2 5 | 195) (1 2 4 | 200) (1 2 3 4 | 267)

PASS

Recursive Best First Search A*

Remember the next best alternative f-Cost to regenerate





n	h(n)
1	60
2	94
3	40
4	45
5	0

(1, 60)

(1 2 | 164) (1 3 | 165) (1 4 | 175)

(1 2 | 175) (1 4 | 175) (1 3 | 180)

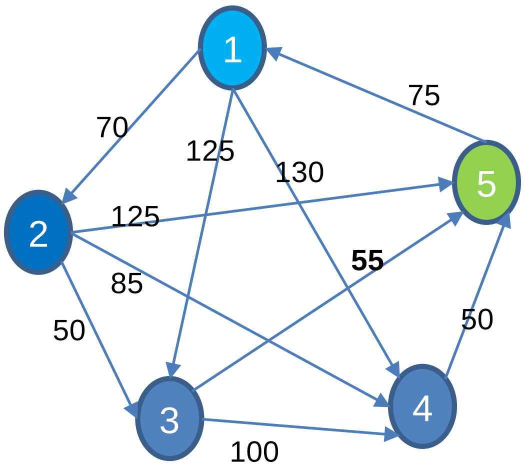
(1 2 3 | 175) (1 4 | 175) (1 3 | 180) (1 2 5 | 195) (1 2 4 | 200)

(1 2 3 5 | 175) (1 4 | 175) (1 3 | 180) (1 2 5 | 195) (1 2 4 | 200) (1 2 3 4 | 267)

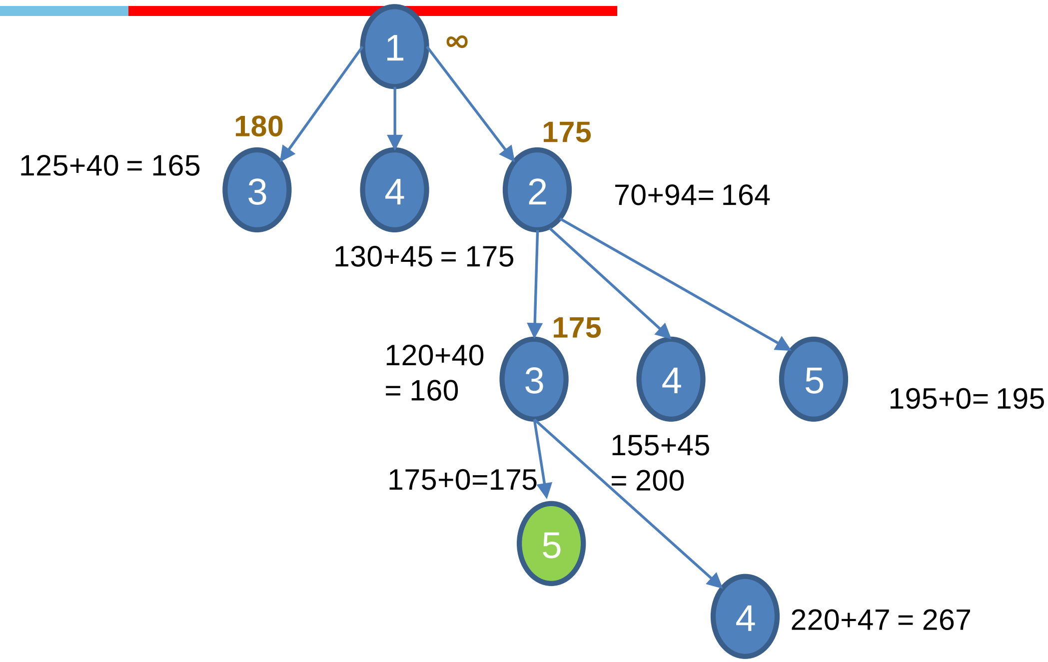
PASS

Recursive Best First Search A*

Remember the next best alternative f-Cost to regenerate



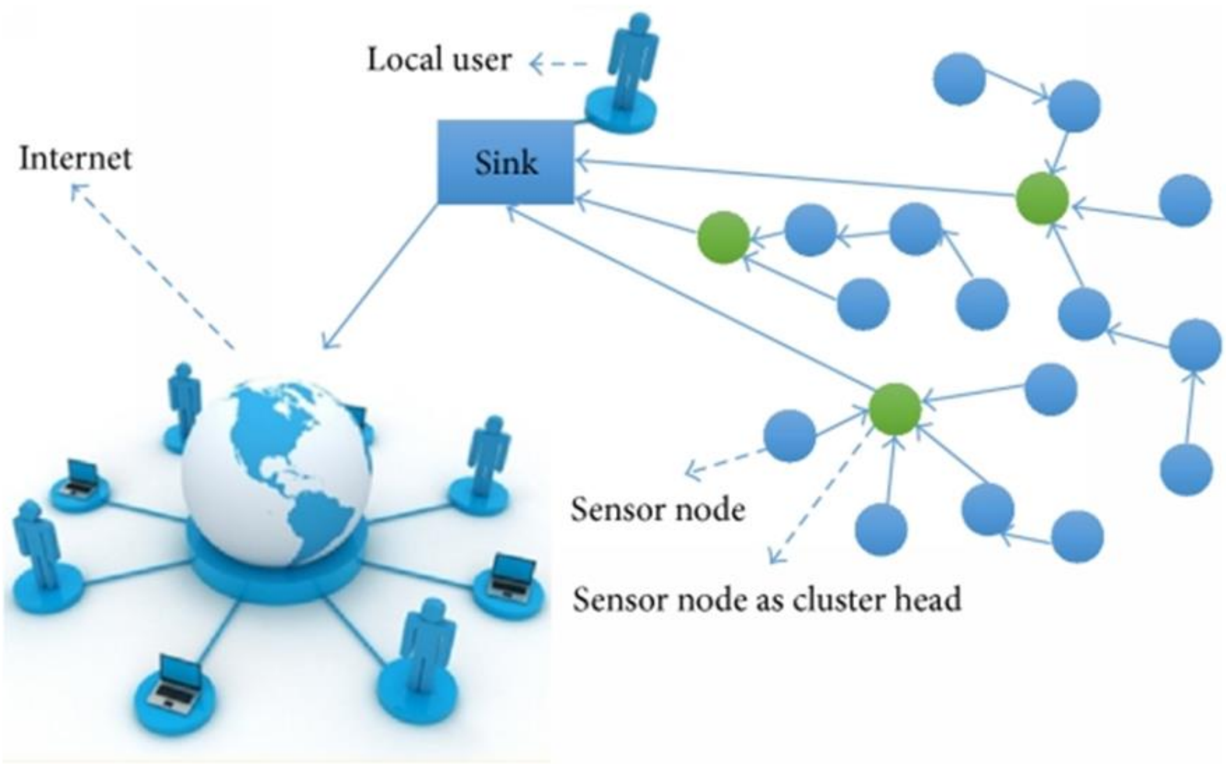
n	h(n)
1	60
2	94
3	40
4	45
5	0



If the current best leaf value > best alternative path
Best leaf value of the forgotten subtree is backed up to the ancestors
Recursion unwinds
Else
Continue expansion

Space Usage = O(bd) very less

Case Study – Search in Network Routing

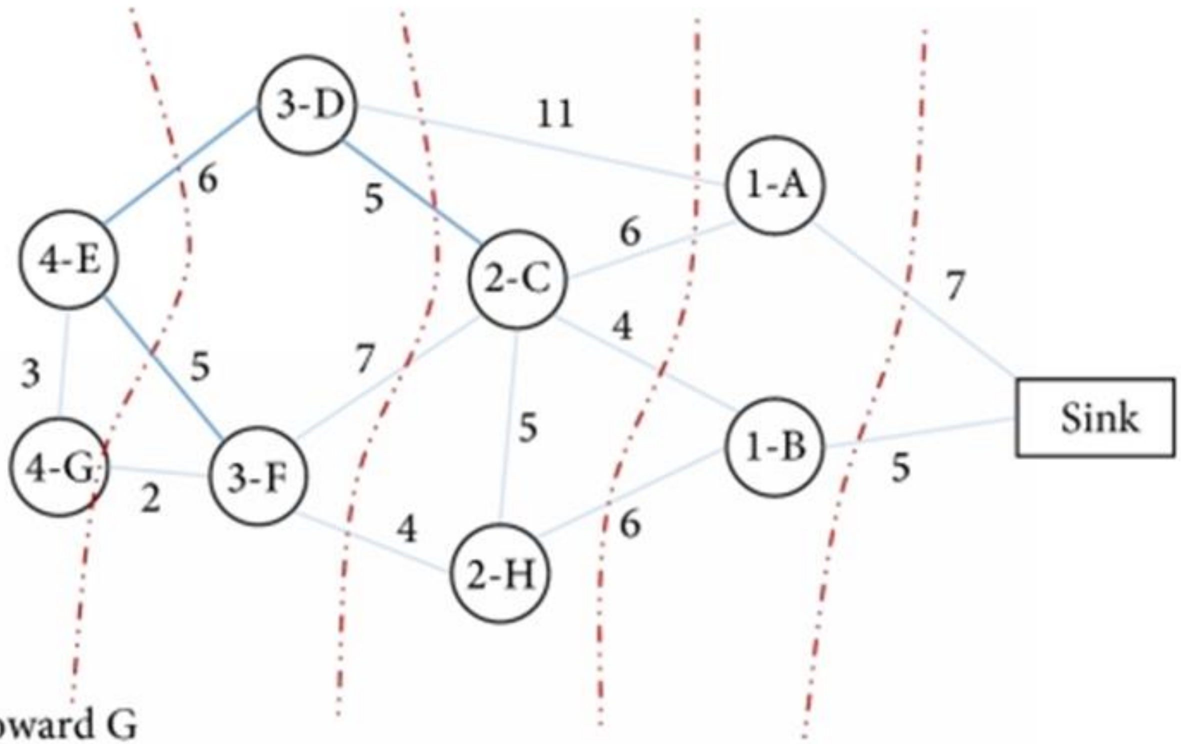


Source Credit :

AR-RBFS: Aware-Routing Protocol Based on Recursive Best-First Search Algorithm for Wireless Sensor Networks
https://doi.org/10.1155/2016/8743927

Case Study – Search in Network Routing

A	14.1
B	11.3
C	8.2
H	6.6
F	2
E	3
D	4.8



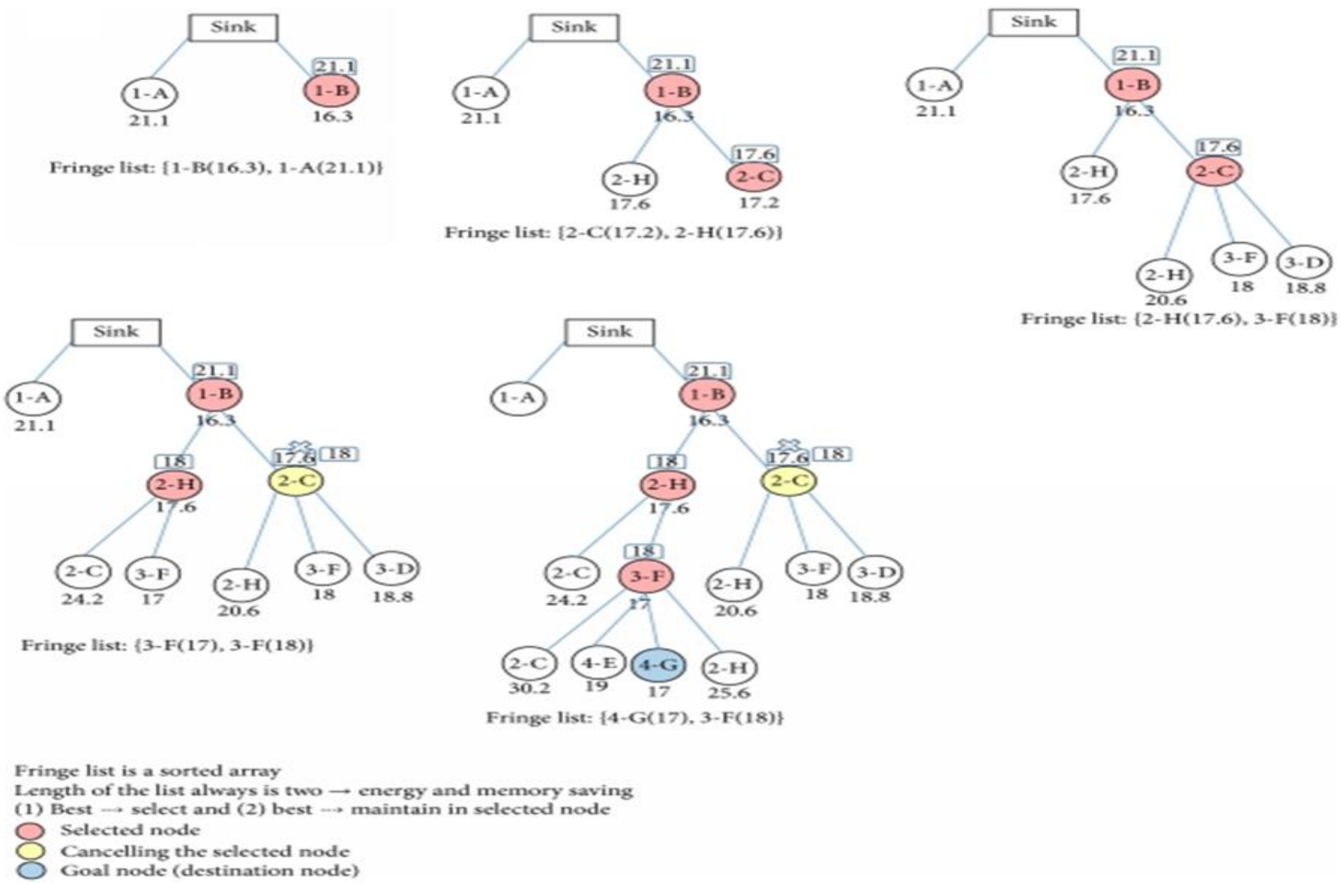
Heuristic values toward G

Source Credit :

AR-RBFS: Aware-Routing Protocol Based on Recursive Best-First Search Algorithm for Wireless Sensor Networks

https://doi.org/10.1155/2016/8743927

Case Study – Search in Network Routing



Source Credit :

AR-RBFS: Aware-Routing Protocol Based on Recursive Best-First Search Algorithm for Wireless Sensor Networks
https://doi.org/10.1155/2016/8743927

Design of Heuristics

Heuristic Design

- **Effective Branching Factor**
- Good Heuristics
- Notion of Relaxed Problems
- Generating Admissible Heuristics

Effective branching factor (b^*):

If the algorithm generates N number of nodes and the solution is found at depth d, then

$$N + 1 = 1 + (b^*) + (b^*)^2 + (b^*)^3 + \ldots + (b^*)^d$$

Heuristic Design

- Effective Branching Factor
- Good Heuristics
- **Notion of Relaxed Problems**
- Generating Admissible Heuristics

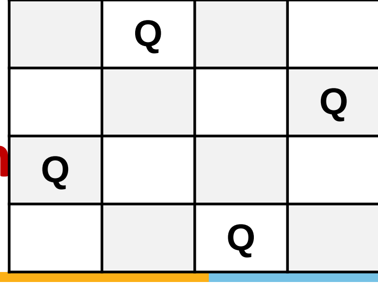
Simplify the problem

Assume no constraints

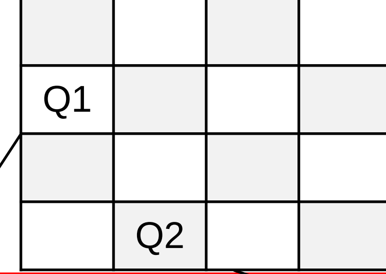
Cost of optimal solution to relaxed problem ≤ Cost of optimal solution for real problem

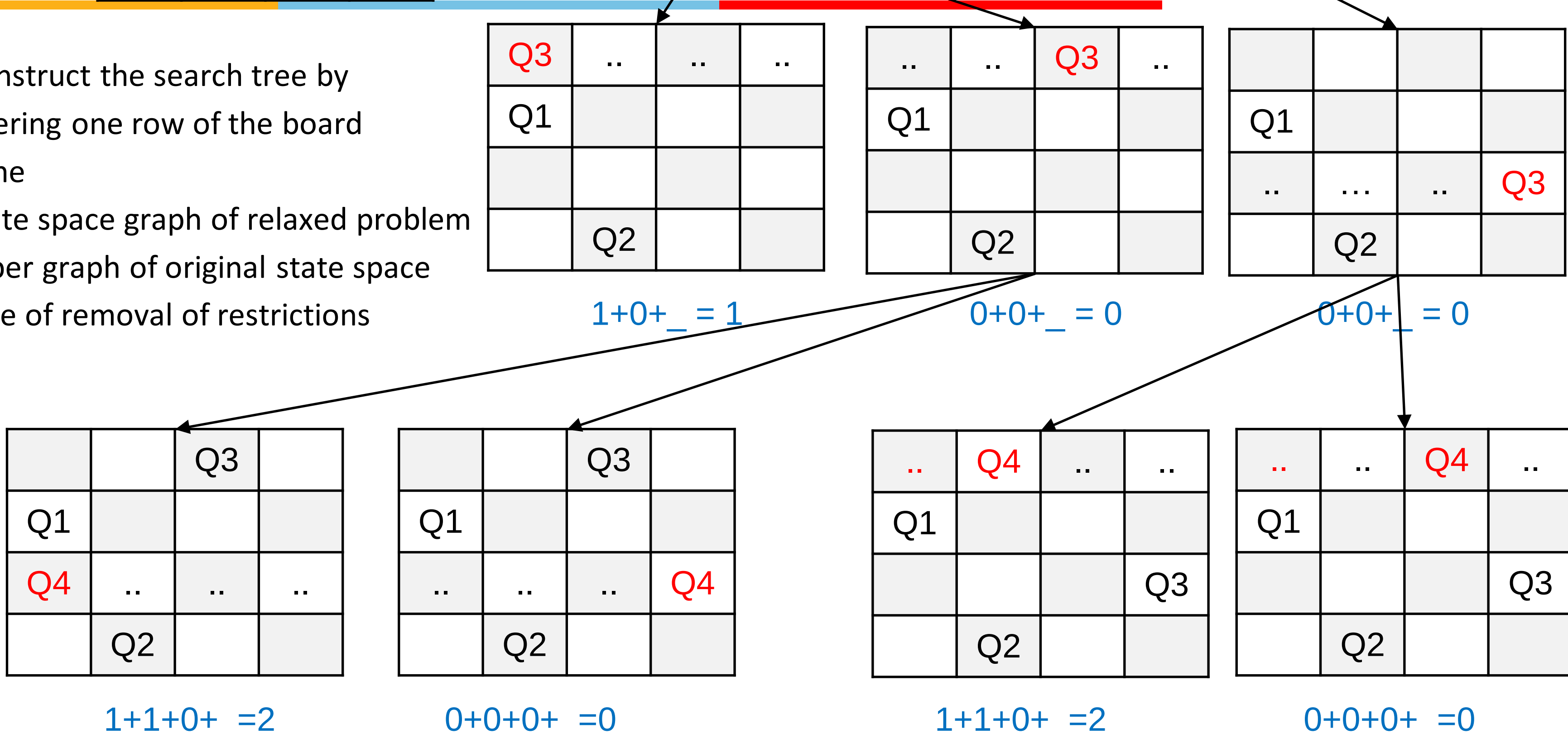
Design of Heuristics

N-Queen



- Construct the search tree by considering one row of the board at a time
- State space graph of relaxed problem is a super graph of original state space because of removal of restrictions

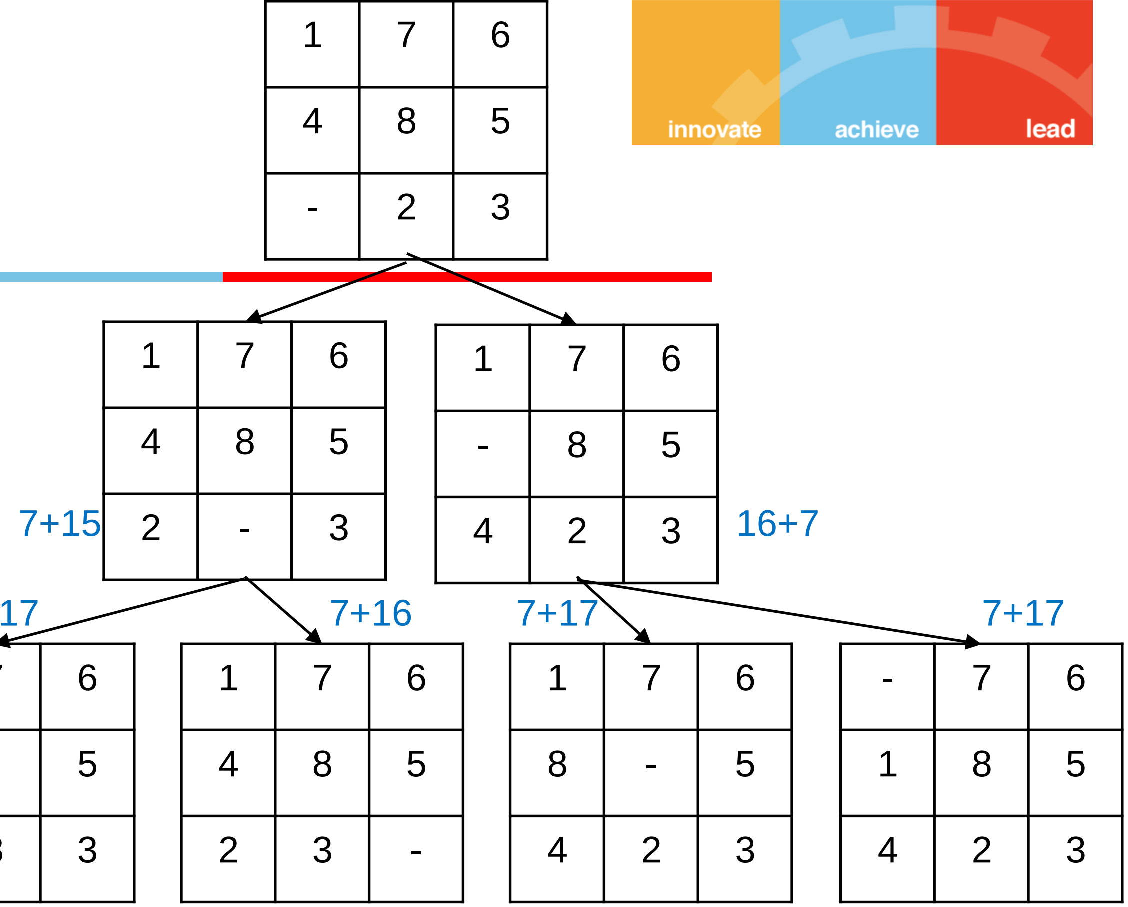




Initial State	Possible Actions	Transition Model	Goal Test	Path Cost	No.Of.States
< Xi , Yi >	Place in any non-occupied row in board		isValid Non-Attacking	Transition + Valid Queens	n!

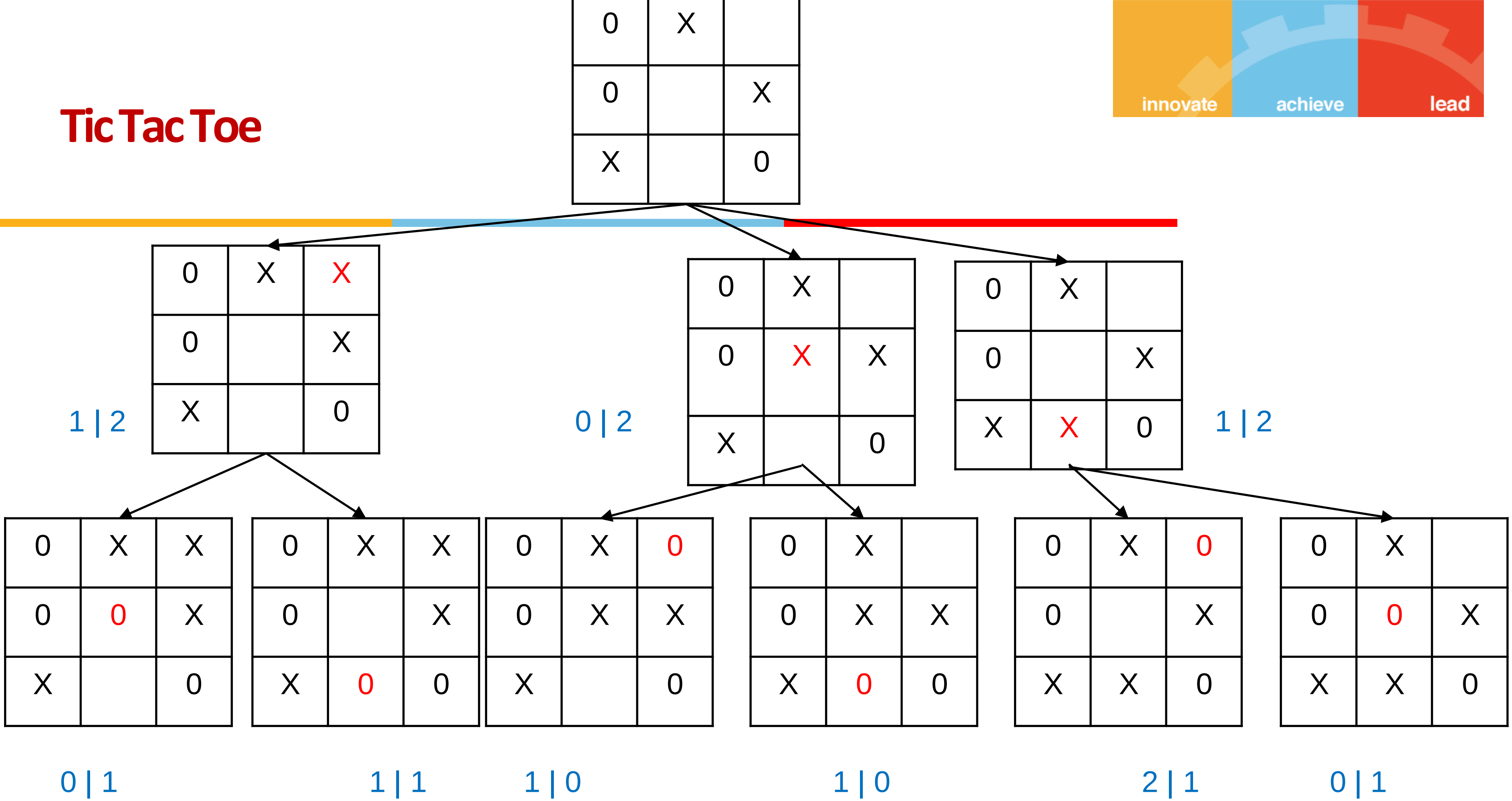
N-Tile

-	1	2
3	4	5
6	7	8



Initial State	Possible Actions	Transition Model	Goal Test	Path Cost	No.Of.States
<LOC, ID>	Move Empty to near by Tile		ID=LOC+1	Transition + Positional + Distance+ Other approaches	9!

Tic Tac Toe



Opposite Win | Player Win

Initial State	Possible Actions	Transition Model	Goal Test	Path Cost	No.Of.States
([Xij], [Yij])	Place a coin in unoccupied (i,j)		N : i's N : j's N : i=j	No.of.Steps + Opp.Win + (N-1-Curr.Win)	$19{,}683=3^9$

Learn from experience

-	1	2
3	4	5
6	7	8

Trail / Puzzle	X1(n) : No.of.Misplaced Tiles	X2(n): Pair of adjacent tiles that are not in goal	X3(n): Position of the empty tile	…………h`(n)
Example 1	7	10	7	…………
Example 2	5	6	6	…………
………	..	..	..	…………

1	7	6
4	8	5
-	2	3

Create a suitable model:
h(n) = c1*X1(n) + c2*X2(n) + ………..

Local Search & Optimization

Task Environment

Goal Formulation

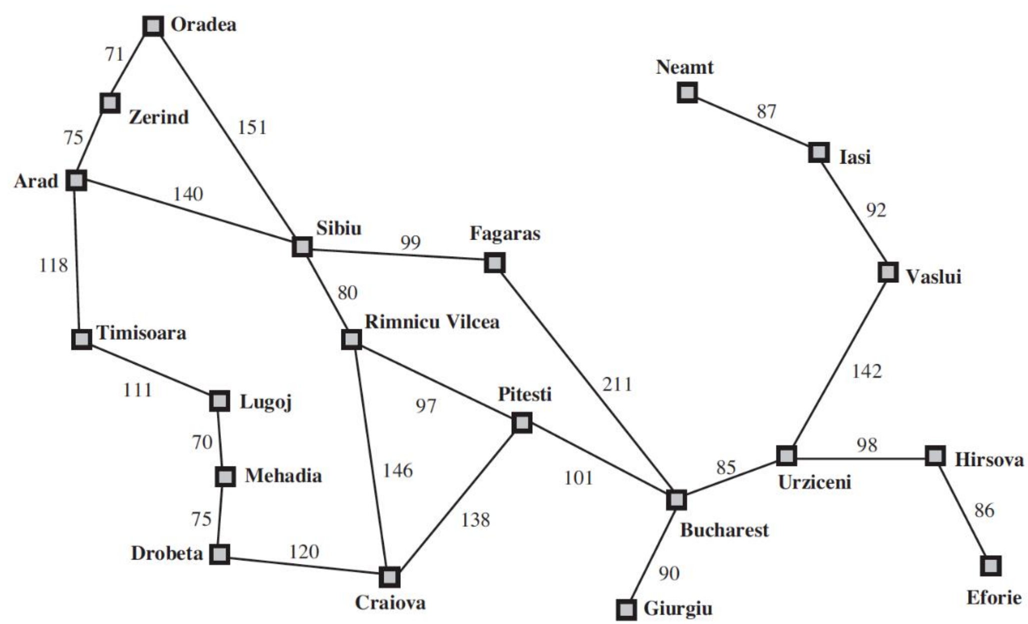
Problem Formulation

Search Phase

Execution Phase

Phases of Solution Search by PSA

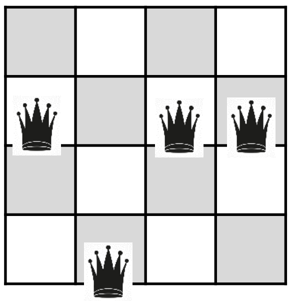
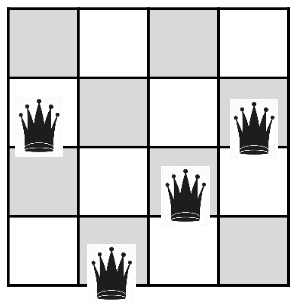
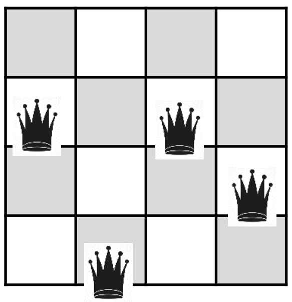
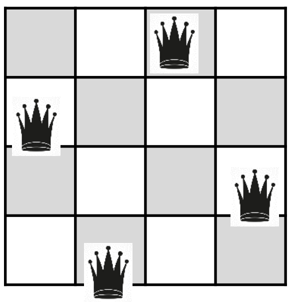
Assumptions – Environment :
Static (4.5)
Observable
Discrete (4.4)
Deterministic (MDP)



Local Search

Terminology

Local Search : Search in the state-space in the neighbourhood of current position until an optimal solution is found

Feasible State/Solution **Neighboring States** **Optimal Solution**

Fitness Value:

h(n) = 4 h(n) = 4 h(n) = 2 h(n) = 0

h(n) = No.of.Conflicting **pairs** of queens

h(n) = 2 h(n) = 2 h(n) = 4 h(n) = 6

h(n) = No.of.**Non**-Conflicting **pairs** of queens.

Required Reading: AIMA - Chapter 3.3, 3.4, 3.5,3.6, 4.1, 4.2, 4.3

Next Session Plan: Local Search & Optimization Algorithms

- Hill Climbing Search
- Local Beam Search
- Simulated Annealing
- Genetic Algorithm
- Particle Swarm Optimization

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

BITS Pilani
Pilani Campus

Artificial & Computational Intelligence

AIML CLZG557

M2 : Problem Solving Agent using Search

Raja vadhana P

Assistant Professor,

BITS - CSIS

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

M5 Probabilistic Representation and Reasoning

M6 Reasoning over time

M7 Ethics in AI

Module 2 : Problem Solving Agent using Search

A. Uninformed Search

B. Informed Search

C. Heuristic Functions

D. Local Search Algorithms & Optimization Problems

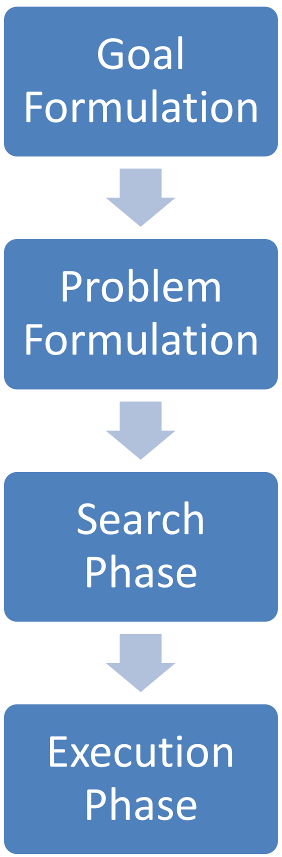
Learning Objective

At the end of this class , students Should be able to:

1. Differentiate which local search is best suitable for given problem
2. Design fitness function for a problem
3. Construct a search tree
4. Apply appropriate local search and show the working of algorithm at least for first 2 iterations with atleast four next level successor generation(if search tree is large)
5. Design and show local search Algorithm steps for a given problem

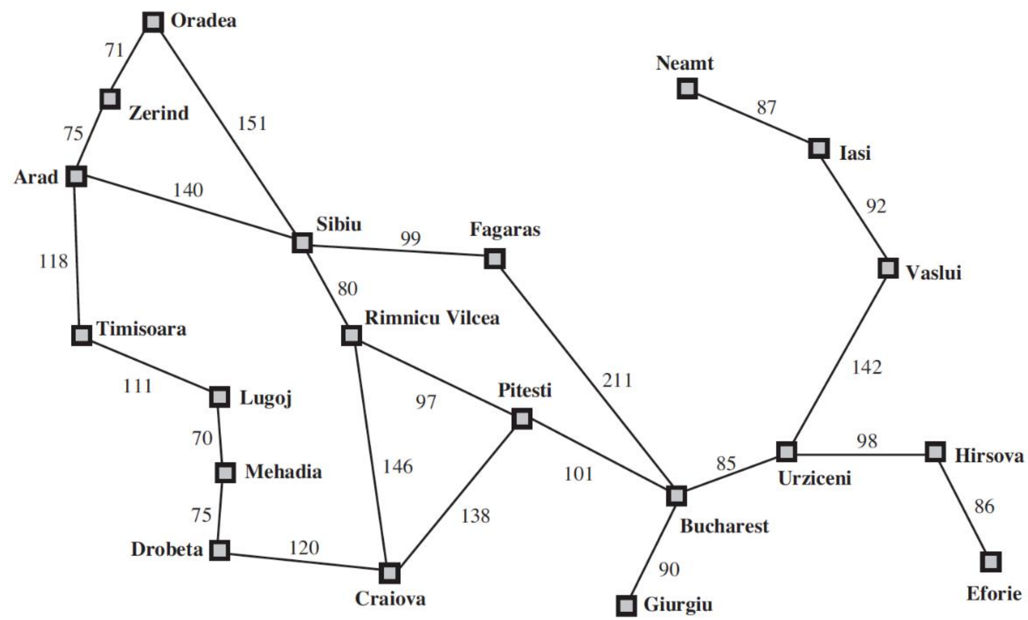
Local Search & Optimization

Task Environment



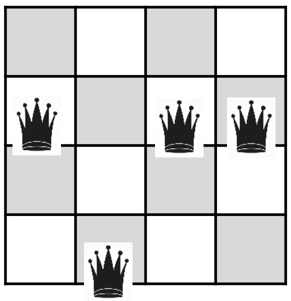
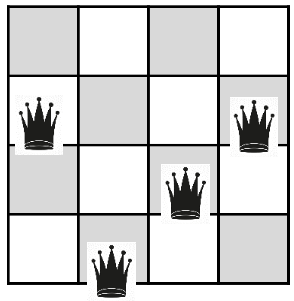
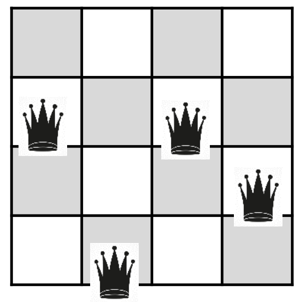
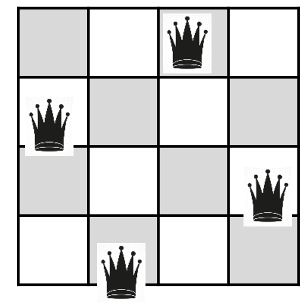
Phases of Solution Search by PSA

Assumptions – Environment :
Static (4.5)
Observable
Discrete (4.4)
Deterministic (MDP)



Terminology

Local Search : Search in the state-space in the neighbourhood of current position until an optimal solution is found

Feasible State/Solution **Neighboring States** **Optimal Solution**

Fitness Value:

h(n) = 4 h(n) = 4 h(n) = 2 h(n) = 0

h(n) = No.of.Conflicting **pairs** of queens

h(n) = 2 h(n) = 2 h(n) = 4 h(n) = 6

h(n) = No.of.**Non**-Conflicting **pairs** of queens.

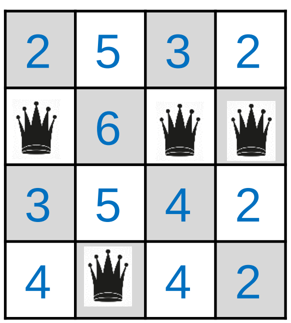
Local Search

Terminology

Local Search : Search in the state-space in the neighbourhood of current position until an optimal solution is found

Algorithms:

- Choice of Neighbour
- Looping Condition
- Termination Condition



innovate achieve lead

Optimization Problem

Goal : Navigate through a state space for a given problem such that an optimal solution can be found

Objective : Minimize or Maximize the objective evaluation function value

Scope : Local

Objective Function : Fitness Value evaluates the goodness of current solution

Local Search : Search in the state-space in the neighbourhood of current position until an optimal solution is found

Single Instance Based	Multiple Instance Based
Hill Climbing	Genetic Algorithm
Simulated Annealing	Particle Swarm Optimization
Local Beam Search	Ant Colony Optimization
Tabu Search	

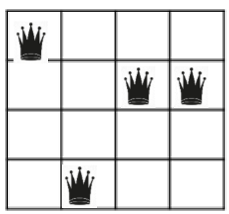
Hill Climbing

Hill Climbing

1. Select a random state
2. Evaluate the fitness scores for all the successors of the state
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. Repeat from Step 2

h(n) = No.of non-conflicting pairs of queens in the board.

Q1-Q2
Q1-Q3
Q1-Q4
Q2-Q3
Q2-Q4
Q3-Q4



1	4	2	2	4

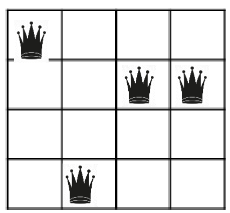
Note : Steps 3 & 4 in the above algorithm will be a part of variation of Hill climbing

Hill Climbing

1. Select a random state
2. Evaluate the fitness scores for all the successors of the state
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. Repeat from Step 2

h(n) = No.of non-conflicting pairs of queens in the board.

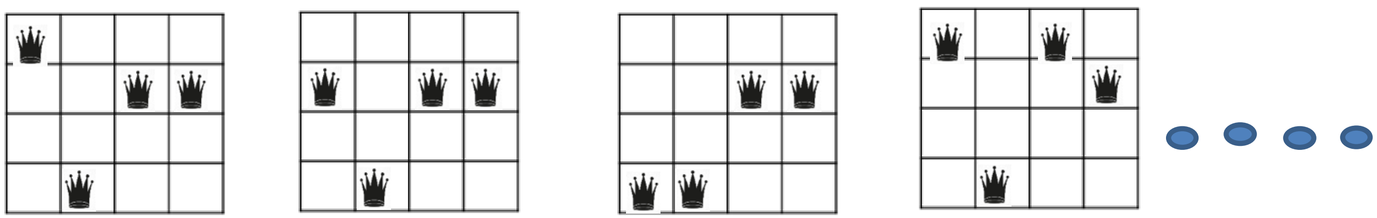
Q1-Q2
Q1-Q3
Q1-Q4
Q2-Q3
Q2-Q4
Q3-Q4



1	4	2	2	4

Hill Climbing

1. Select a random state
2. Evaluate the fitness scores for all the successors of the state
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. Repeat from Step 2

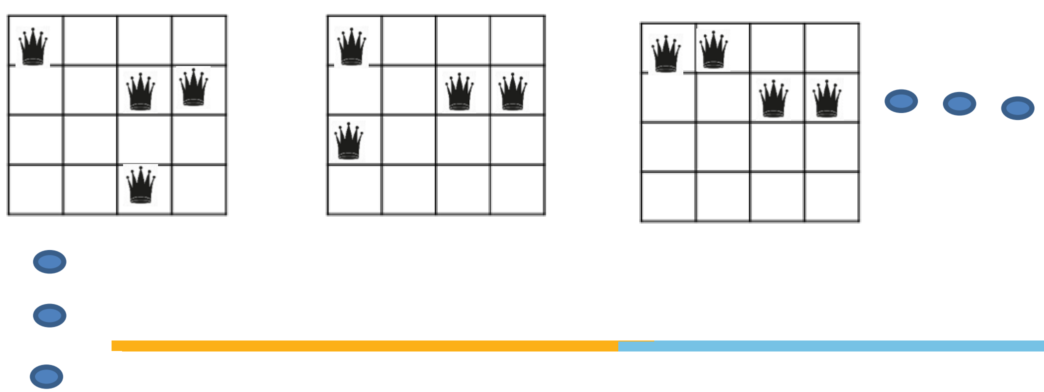


1	4	2	2

2	4	2	2

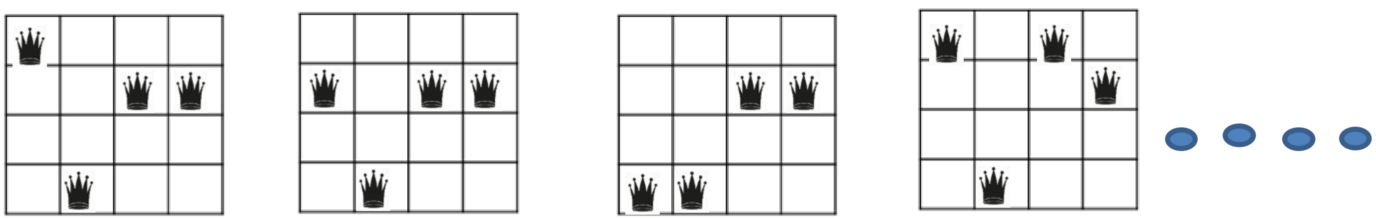
4	4	2	2

1	4	1	2



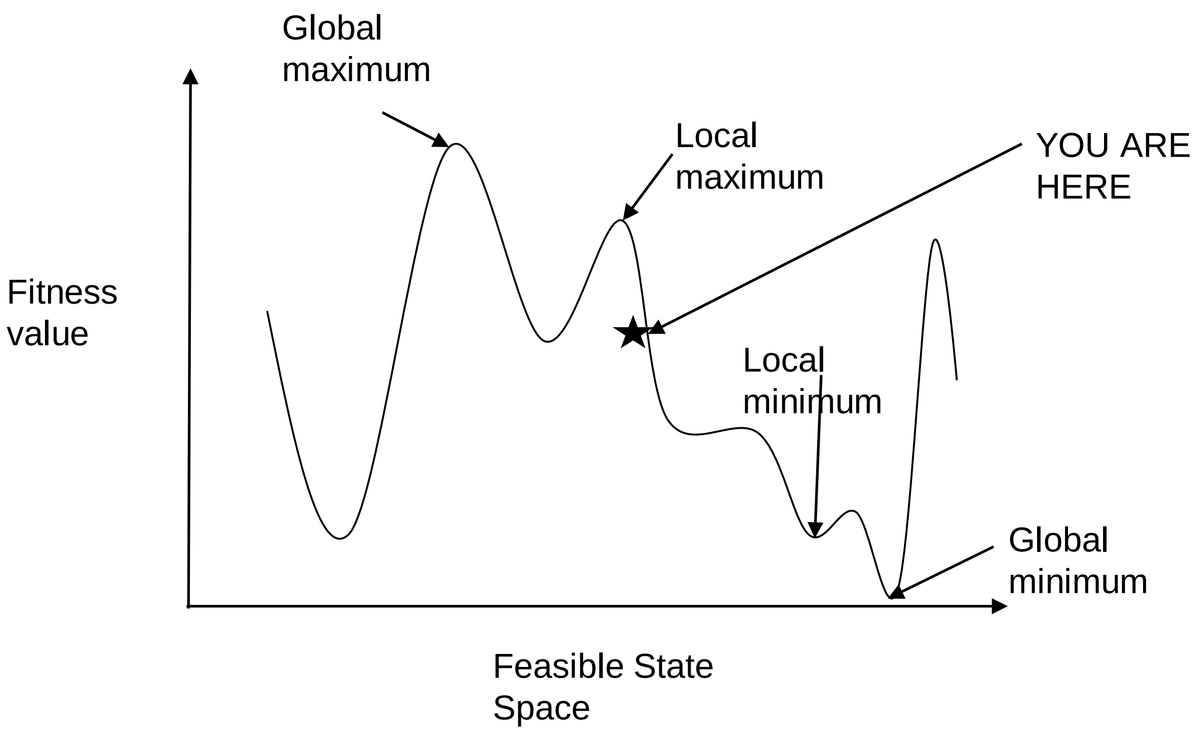
Hill Climbing

1. Select a random state
2. Evaluate the fitness scores for all the successors of the state
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. Repeat from Step 2



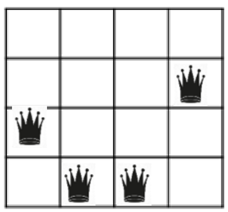
1	4	2	2	4
2	4	2	2	2
4	4	2	2	2
1	4	1	2	3

Hill Climbing



Random Restart

1. Select a random state
2. Evaluate the fitness scores for all the successors of the state
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. Repeat from Step 2



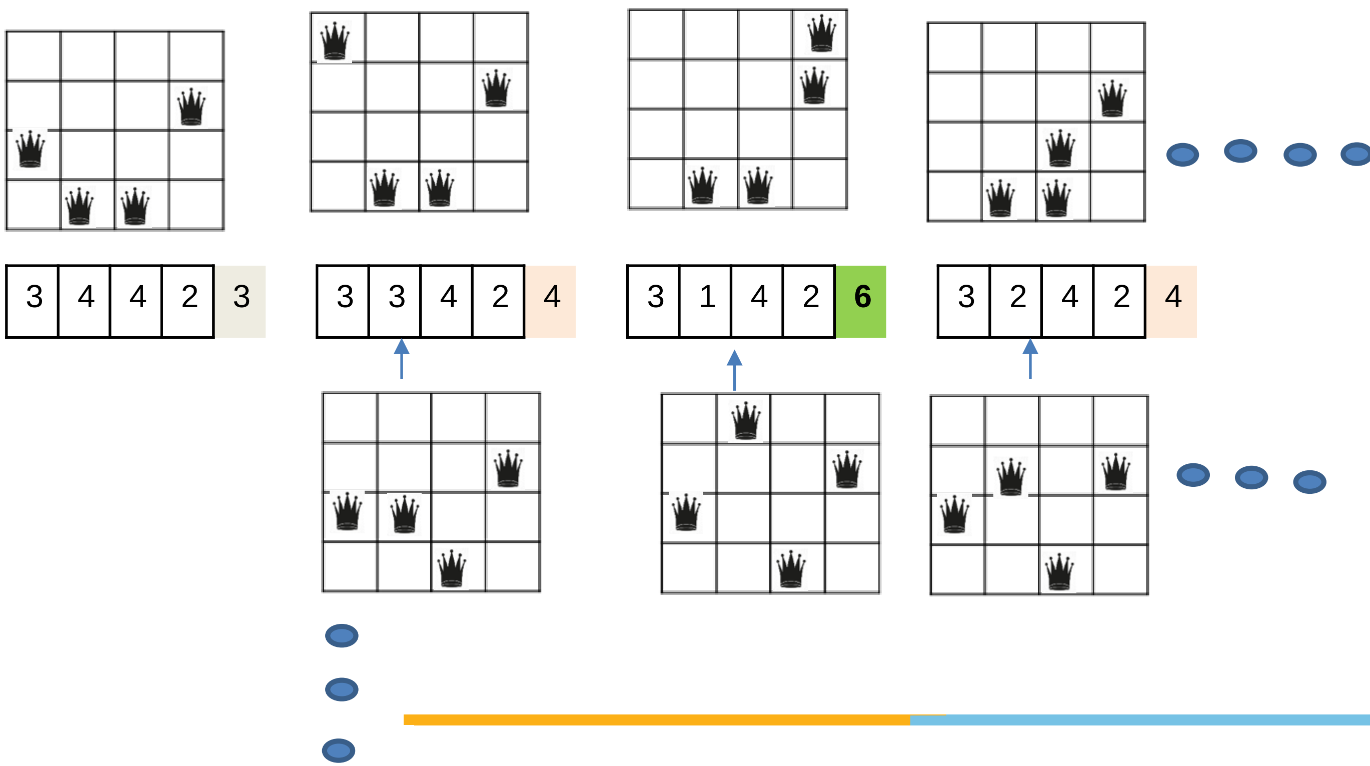
3	4	4	2	3

```
function HILL-CLIMBING(problem) returns a state that is a local maximum

  current ← MAKE-NODE(problem.INITIAL-STATE)
  loop do
      neighbor ← a highest-valued successor of current
      if neighbor.VALUE ≤ current.VALUE then return current.STATE
      current ← neighbor
```

Hill Climbing

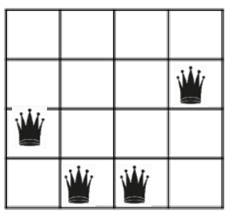
1. Select a random state
2. Evaluate the fitness scores for all the successors of the state
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. Repeat from Step 2



Hill Climbing

Random Restart

1. Select a random state
2. Evaluate the fitness scores for all the successors of the state
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. Repeat from Step 2



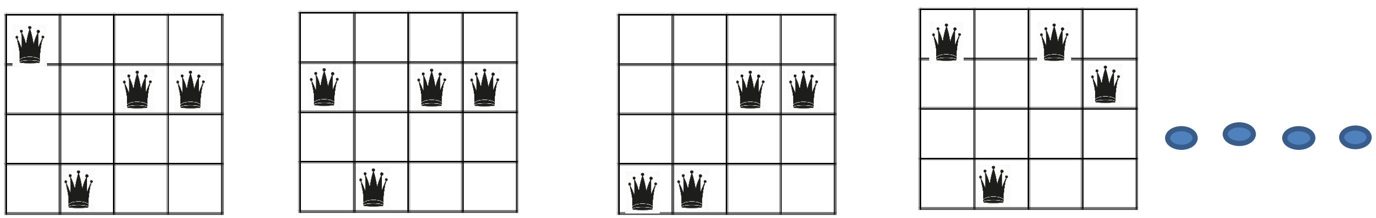
3	4	4	2	3

function HILL-CLIMBING(*problem*) **returns** a state that is a local maximum

current ← MAKE-NODE(*problem*.INITIAL-STATE)
loop do
  neighbor ← a highest-valued successor of *current*
  if neighbor.VALUE ≤ current.VALUE **then return** *current*.STATE
  current ← *neighbor*

Stochastic Hill Climbing

1. Select a random state
2. Evaluate the fitness scores for all the successors of the state
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. Repeat from Step 2



1	4	2	2	4

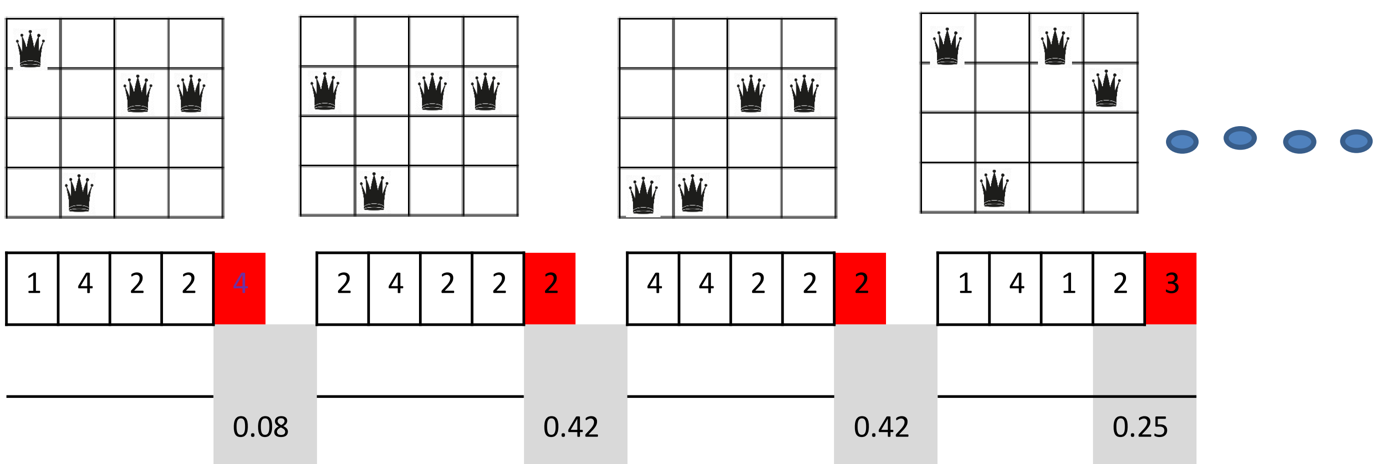
2	4	2	2	2

4	4	2	2	2

1	4	1	2	3

Stochastic Hill Climbing

1. Select a random state
2. Evaluate the fitness scores for all the successors of the state
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. Repeat from Step 2



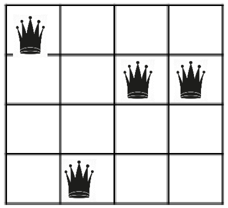
12 N = {4,2,2,3,3,2,2,0,2,1,3,0}

next ← a randomly selected successor of *current*
ΔE ← *next*.VALUE – *current*.VALUE
if $\Delta E > 0$ **then** *current* ← *next*
else *current* ← *next* only with probability $e^{\Delta E/T}$

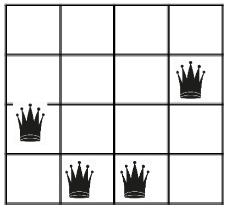
Local Beam Search

Beam Search

1. Initialize k random state
2. Evaluate the fitness scores for all the successors of the k states
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. If the goal is not found, Select the next ‘k’ states randomly based on the probabilit
6. Repeat from Step 2



1	4	2	2	4

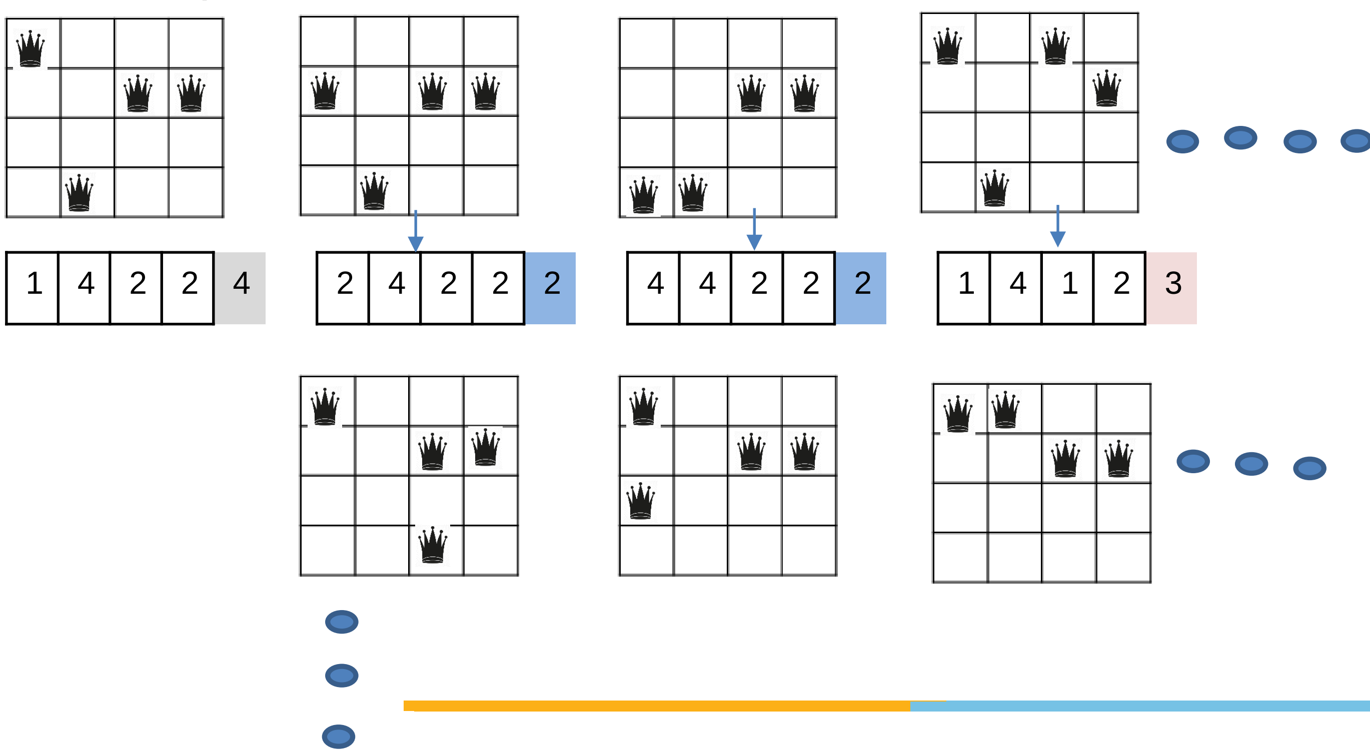


3	4	4	2	3

Beam Search

1st State

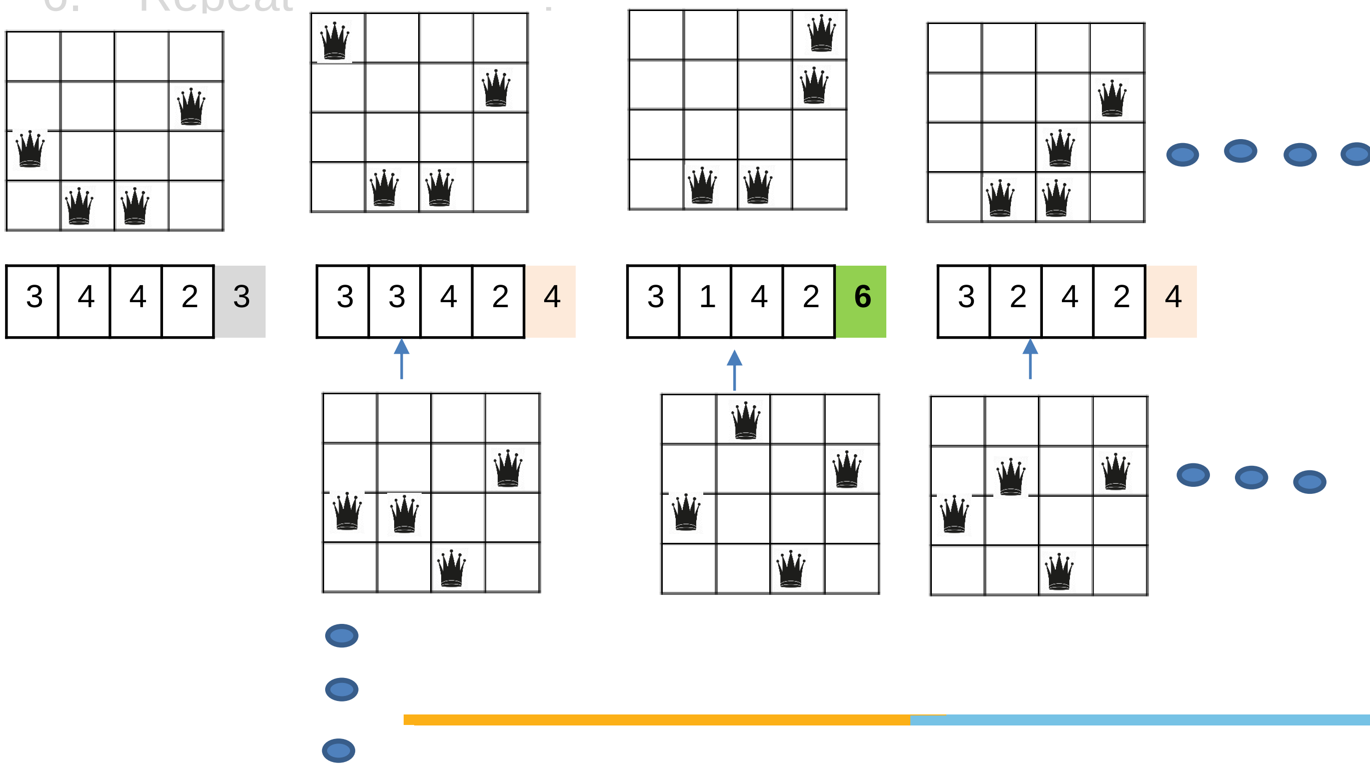
1. Initialize k random state
2. Evaluate the fitness scores for all the successors of the k states
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. If the goal is not found, Select the next ‘k’ states randomly based on the probability
6. Repeat from Step 2



Beam Search

2nd State

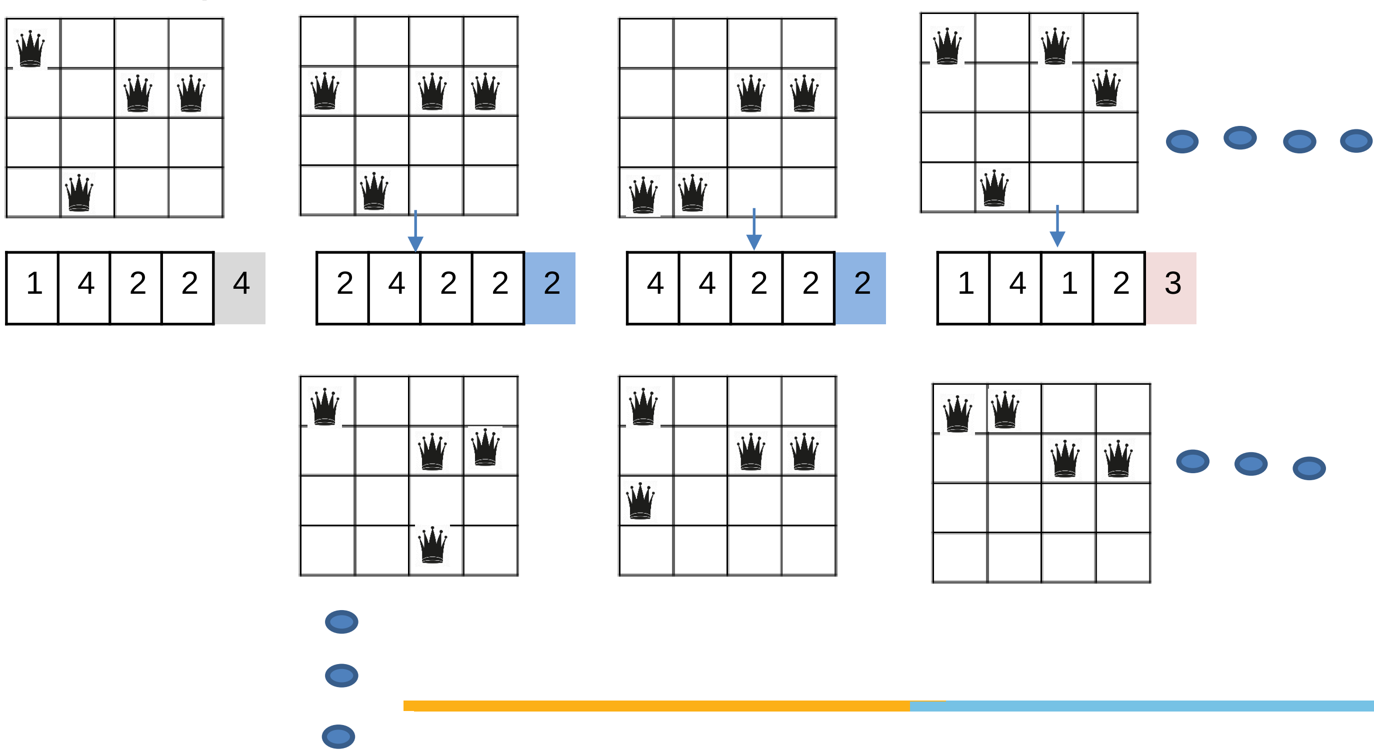
1. Initialize k random state
2. Evaluate the fitness scores for all the successors of the k states
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. If the goal is not found, Select the next 'k' states randomly based on the probability
6. Repeat from Step 2



Stochastic Beam Search

Sample from 1st State

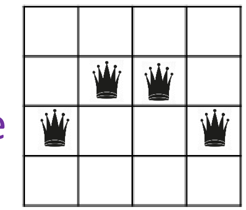
1. Initialize k random state
2. Evaluate the fitness scores for all the successors of the k states
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. If the goal is not found, Select the next 'k' states randomly based on the probability
6. Repeat from Step 2

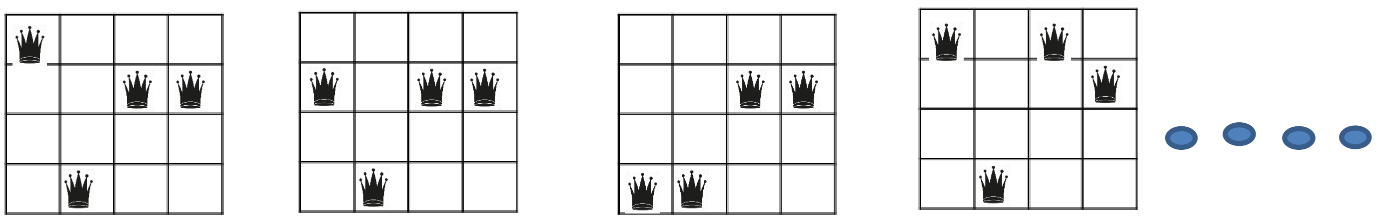


Simulated Annealing

Simulated Annealing

1. Select a random state
2. Evaluate the fitness scores for all the successors of the state
3. Calculate the probability of selecting a successor based on fitness score
4. Select the next state based on the highest probability
5. Repeat from Step 2





1	4	2	2	4	0.55
2	4	2	2	2	0.5
4	4	3	3	2	0.5
1	4	1	2	3	0.525

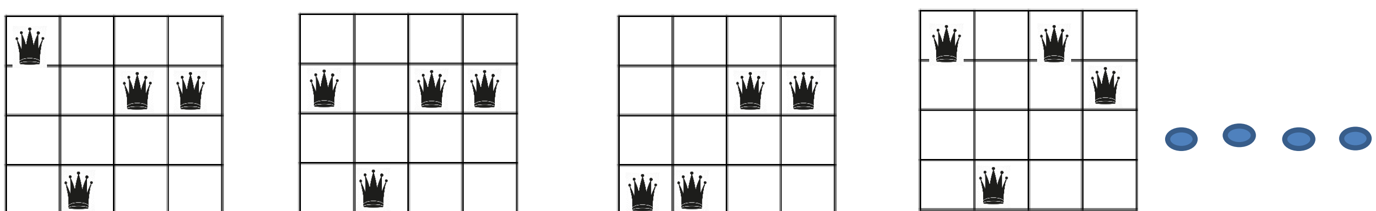
12 N = {4,2,2,3,3,2,1,3,2,1,3,2}

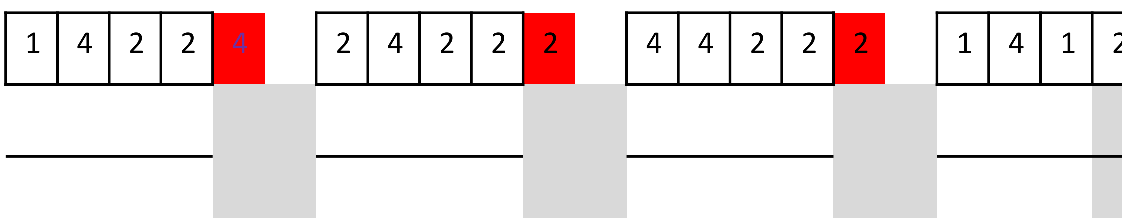
Init = 2

Simulated Annealing

function SIMULATED-ANNEALING(*problem*, *schedule*) **returns** a solution state
  inputs: *problem*, a problem
    schedule, a mapping from time to "temperature"

  current ← MAKE-NODE(*problem*.INITIAL-STATE)
  for t = 1 **to** ∞ **do**
    T ← *schedule*(t)
    if $T = 0$ **then return** *current*
    next ← a randomly selected successor of *current*
    ΔE ← *next*.VALUE – *current*.VALUE
    if $\Delta E > 0$ **then** *current* ← *next*
    else *current* ← *next* only with probability $e^{\Delta E/T}$





Next Value	ΔE	ΔE/t	$e^{\Delta E/t}$	$\frac{1}{1+e^{\Delta E/t}}$
1	-1	-0.1	0.904	0.525
2	0	0	1	0.5
3	1	0.1	1.105	0.47
4	2	0.2	1.221	0.45

Simulated Annealing

Current Value = 4 (Local Maxima)

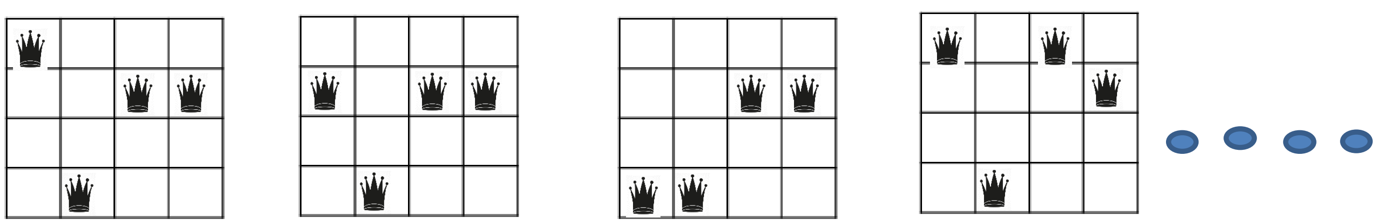
Global Maxima = 6

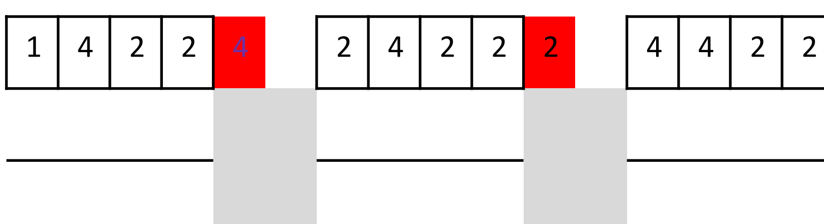
Next Value	ΔE	ΔE/t	$e^{\Delta E/t}$	$\frac{1}{1+e^{\Delta E/t}}$	ΔE/t	$e^{\Delta E/t}$	$\frac{1}{1+e^{\Delta E/t}}$
2	2	0.1	1.12	0.47	0.4	1.49	0.40
3	1	0.05	1.05	0.49	0.2	1.22	0.45
5	-1	-0.05	0.95	0.51	-0.2	0.82	0.55

Simulated Annealing

function SIMULATED-ANNEALING(*problem*, *schedule*) **returns** a solution state
  inputs: *problem*, a problem
    schedule, a mapping from time to "temperature"

  current ← MAKE-NODE(*problem*.INITIAL-STATE)
  for $t = 1$ **to** ∞ **do**
    $T \leftarrow$ *schedule*(t)
    if $T = 0$ **then return** *current*
    next ← a randomly selected successor of *current*
    $\Delta E \leftarrow$ *next*.VALUE – *current*.VALUE
    if $\Delta E > 0$ **then** *current* ← *next*
    else *current* ← *next* only with probability $e^{\Delta E/T}$





Next Value	ΔE	ΔE/t	$e^{\Delta E/t}$	$\frac{1}{1+e^{\Delta E/t}}$	$e^{-\Delta E/t}$	$\frac{1}{1+e^{-\Delta E/t}}$
1	-1	-0.1	0.904	0.525	1.105	0.47
2	0	0	1	0.5	0	0.5
3	1	0.1	1.105	0.47	0.904	0.525
4	2	0.2	1.221	0.45	0.819	0.55

Simulated Annealing

Maximization problem design to achieve global minima

Set Temp to very high temp t

Set n as number of iteration to be performed at a particular t

L1: Randomly select a random neighbour

Calculate Energy barrier E = f(N)-f(C)

If **E > 0** then its a good move

    Move ahead for next tree search level

Else

    Create a random number r:[0-1]

    If **$r < e^{-E/t}$**

        Choose this bad state & move downhill

    Else

        Go to L1.

If Goal is reached or {acceptable goal(set criteria to check)node is reached & t is small END}

Else

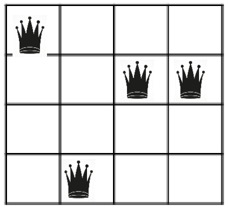
    If no.of.neighbors explored has reached a threshold >=n

        then Lower t and go to L1.

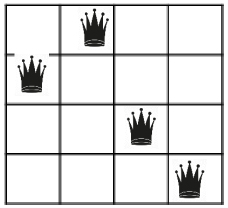
Genetic Algorithm

Genetic Algorithm

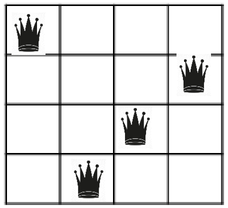
1. Select 'k' random states – **Initialization : k=4**
2. Evaluate the fitness value all states
3. If anyone of the state's has achieved the threshold fitness value or threshold new states or no change is seen than previous iteration then the algorithm stops
4. Else, use roulette wheel mechanism to select pair/s
5. Pairs selected produces new state (successor) by crossover
6. Successor is allowed to mutate
7. Repeat from Step 2



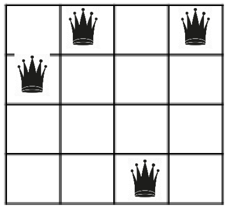
1	4	2	2



2	1	3	4



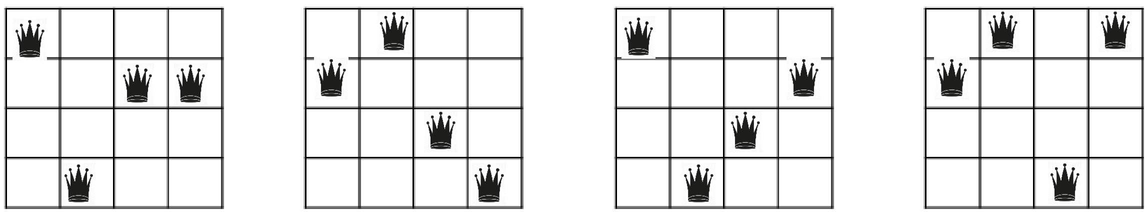
1	4	3	2



2	1	4	1

Genetic Algorithm

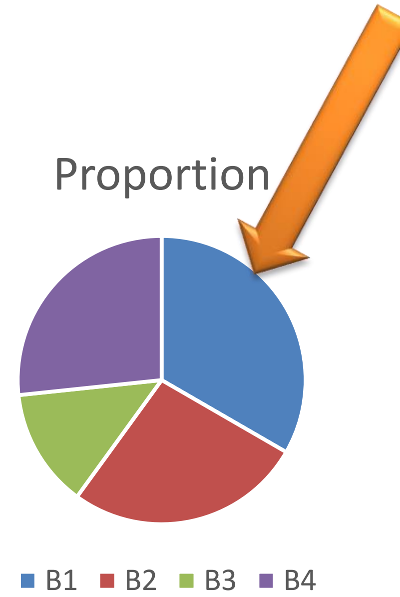
1. Select ‘k’ random states – **Initialization : k=4**
2. Evaluate the fitness value all states : Maximizing function : No.of.Non-attacking pairs Queens → Threshold = 6
3. If anyone of the state’s has achieved the threshold fitness value or threshold new states or no change is seen than previous iteration then the algorithm stops
4. Else, use roulette wheel mechanism to select pair/s
5. Pairs selected produces new state (successor) by crossover
6. Successor is allowed to mutate
7. Repeat from Step 2

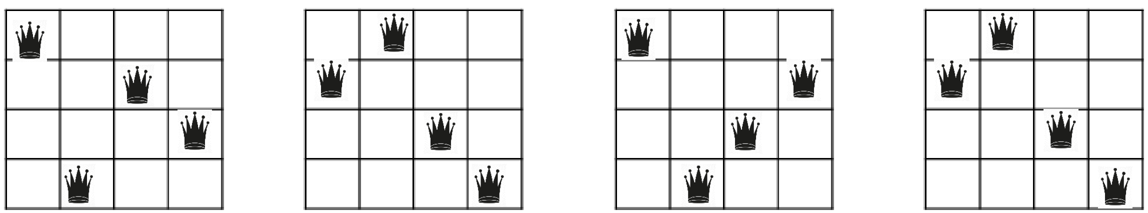


1	4	2	2	4	2	1	3	4	4	1	4	3	2	2	2	1	4	1	3

Genetic Algorithm

Eg., use roulette wheel mechanism to select pair/s





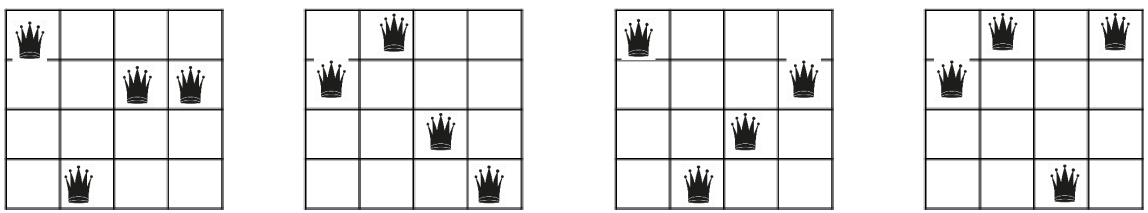
1	4	2	3	5	2	1	3	4	4	1	4	3	2	2	2	1	3	4	4
				0.33					0.27					0.13					0.27

2	1	3	4	1	4	2	3	1	4	2	3	1	4	3	2

Sample winners of game -1 ,2,3,4 : B4, B1, B1, B3

Genetic Algorithm

1. Select 'k' random states – **Initialization : k=4**
2. Evaluate the fitness value all states : Maximizing function : No.of.Non-attacking pairs Queens → Threshold = 6
3. ~~If anyone of the state's has achieved the threshold fitness value or threshold new states or no change is seen than previous iteration then the algorithm stops~~
4. Else, use roulette wheel mechanism to select pair/s
5. Pairs selected produces new state (successor) by crossover
6. Successor is allowed to mutate
7. Repeat from Step 2



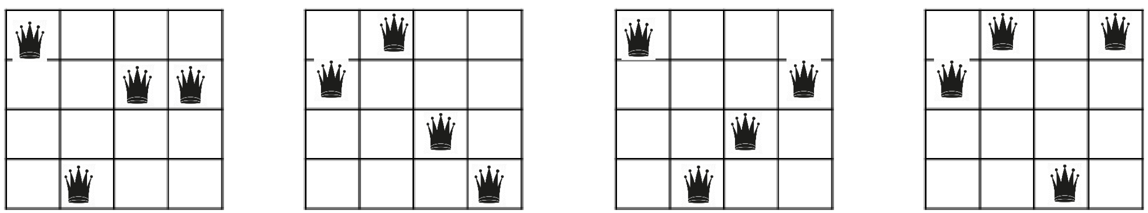
1	4	2	2	4	0.31
2	1	3	4	4	0.31
1	4	3	2	2	0.15
2	1	4	1	3	0.23

2	1	4	1
1	4	2	2
1	4	2	2
1	4	3	2

Sample winners of game -1 ,2,3,4 : B4, B1, B1, B3

Genetic Algorithm

1. Select 'k' random states – **Initialization : k=4**
2. Evaluate the fitness value all states : Maximizing function : No.of.Non-attacking pairs Queens → Threshold = 6
3. ~~If anyone of the state's has achieved the threshold fitness value or threshold new states or no change is seen than previous iteration then the algorithm stops~~
4. Else, use roulette wheel mechanism to select pair/s
5. Pairs selected produces new state (successor) by crossover
6. Successor is allowed to mutate
7. Repeat from Step 2



State	Fitness	Probability
1 4 2 2	4	0.31
2 1 3 4	4	0.31
1 4 3 2	2	0.15
2 1 4 1	3	0.23

Parent 1	⟺	Parent 2
2 | 1 4 1	⟺	1 | 4 2 2
1 | 4 2 | 2	⟺	1 | 4 3 | 2

Successor 1	Successor 2
2 4 2 2	1 1 4 1
1 4 3 2	1 4 2 2

Genetic Algorithm – Reference to alternative approaches of crossover

2	4	2	2

1	1	4	1

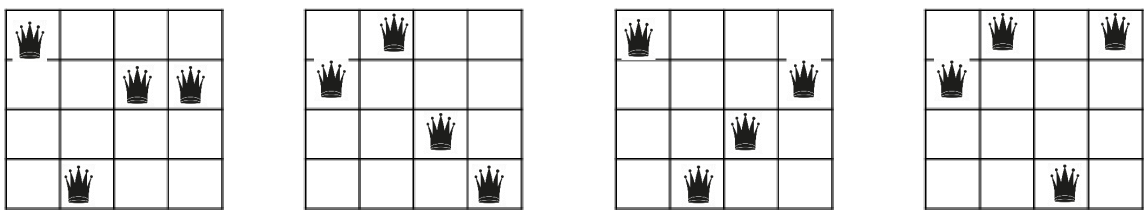
1	4	3	2

1	4	2	2

http://ictactjournals.in/paper/IJSC_V6_I1_paper_4_pp_1083_1092.pdf

Genetic Algorithm

1. Select ‘k’ random states – **Initialization : k=4**
2. Evaluate the fitness value all states : Maximizing function : No.of.Non-attacking pairs Queens → Threshold = 6
3. ~~If anyone of the state’s has achieved the threshold fitness value or threshold new states or no change is seen than previous iteration then the algorithm stops~~
4. Else, use roulette wheel mechanism to select pair/s
5. Pairs selected produces new state (successor) by crossover
6. Successor is allowed to mutate
7. Repeat from Step 2



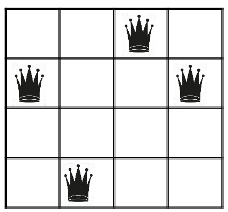
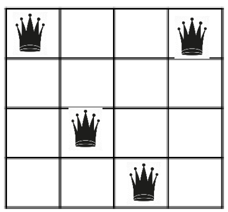
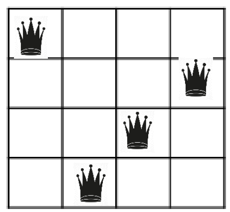
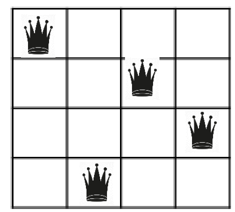
1	4	2	2	4	0.31
2	1	3	4	4	0.31
1	4	3	2	2	0.15
2	1	4	1	3	0.23

2	1	4	1	⟺	1	4	2	2	
1	4	2	2	⟺	1	4	3	2	

2	4	2	2
1	1	4	1
1	4	3	2
1	4	2	2

Genetic Algorithm

1. Select ‘k’ random states – **Initialization : k=4**
2. Evaluate the fitness value all states : Maximizing function : No.of.Non-attacking pairs Queens → Threshold = 6
3. ~~If anyone of the state’s has achieved the threshold fitness value or threshold new states or no change is seen than previous iteration then the algorithm stops~~
4. Else, use roulette wheel mechanism to select pair/s
5. Pairs selected produces new state (successor) by crossover
6. Successor is allowed to mutate
7. Repeat from Step 2

2	4	2	2

1	1	4	1

1	4	3	2

1	4	2	2

2	4	1	2

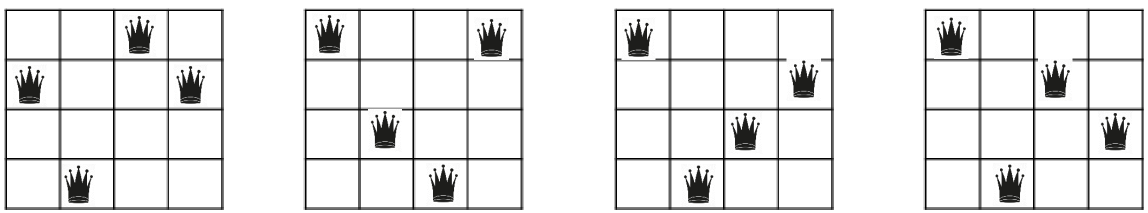
1	3	4	1

1	4	3	2

1	4	2	3

Genetic Algorithm

1. Select 'k' random states – **Initialization : k=4**
2. Evaluate the fitness value all states : Maximizing function : No.of.Non-attacking pairs Queens → Threshold = 6
3. If anyone of the state's has achieved the threshold fitness value or threshold new states or no change is seen than previous iteration then the algorithm stops
4. Else, use roulette wheel mechanism to select pair/s
5. Pairs selected produces new state (successor) by crossover
6. Successor is allowed to mutate
7. Repeat from Step 2



2	4	2	2

1	1	4	1

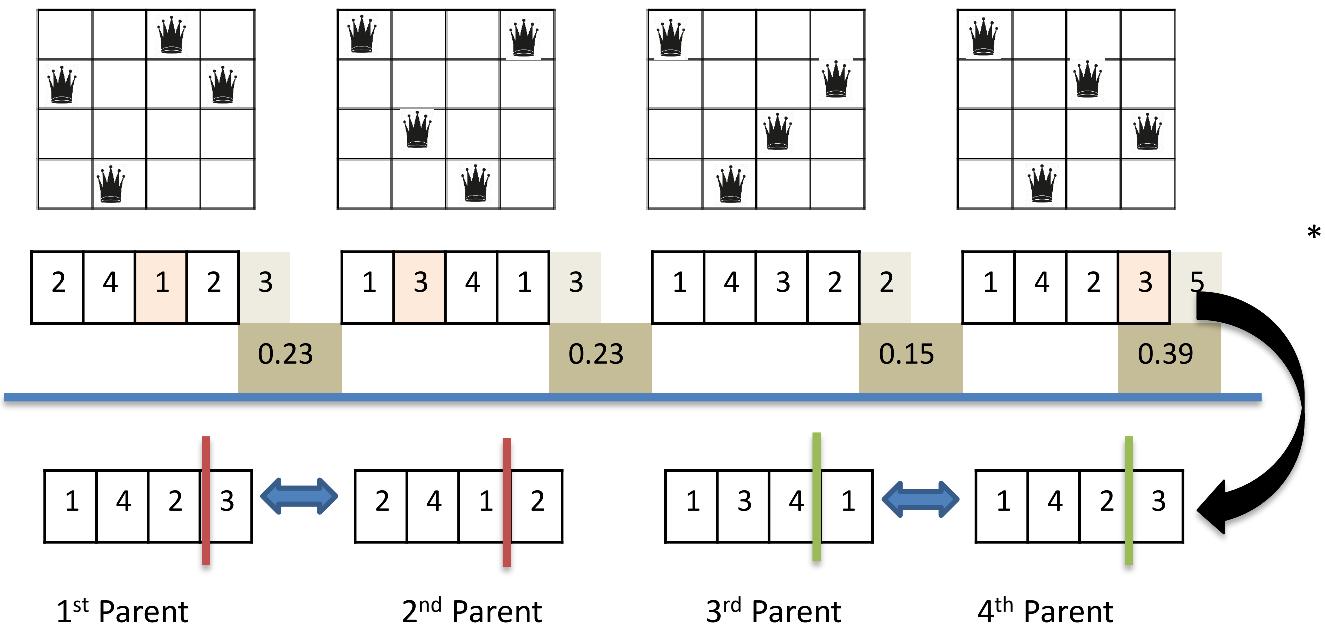
1	4	3	2

1	4	2	2

2	4	1	2	3	0.23
1	3	4	1	3	0.23
1	4	3	2	2	0.15
1	4	2	3	5	0.39

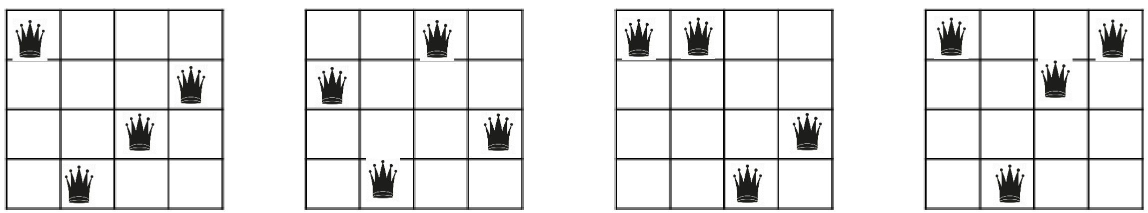
Genetic Algorithm

1. Select ‘k’ random states – **Initialization : k=4**
2. Evaluate the fitness value all states : Maximizing function : No.of.Non-attacking pairs Queens → Threshold = 6
3. If anyone of the state’s has achieved the threshold fitness value or threshold new states or no change is seen than previous iteration then the algorithm stops
4. Else, use roulette wheel mechanism to select pair/s
5. Pairs selected produces new state (successor) by crossover
6. Successor is allowed to mutate
7. Repeat from Step 2



Genetic Algorithm

1. Select 'k' random states – **Initialization : k=4**
2. Evaluate the fitness value all states : Maximizing function : No.of.Non-attacking pairs Queens → Threshold = 6
3. If anyone of the state's has achieved the threshold fitness value or threshold new states or no change is seen than previous iteration then the algorithm stops
4. Else, use roulette wheel mechanism to select pair/s
5. Pairs selected produces new state (successor) by crossover
6. Successor is allowed to mutate
7. Repeat from Step 2



1	4	2	3		2	4	1	2		1	3	4	1		1	4	2	3

1	4	2	2		2	4	1	3		1	3	4	3		1	4	2	1

1	4	3	2	2		2	4	1	3	6		1	1	4	3	3		1	4	2	1	4

Genetic Algorithm

Techniques:

1. Design of the fitness function
2. Diversity in the population to be accounted
3. Randomization

Application:

- Creative tasks
- Exploratory in nature
- Planning problem
- Static Applications

Genetic Algorithm

innovate achieve lead

```
function GENETIC-ALGORITHM(population, FITNESS-FN) returns an individual
  inputs: population, a set of individuals
          FITNESS-FN, a function that measures the fitness of an individual

  repeat
     new_population ← empty set
     for i = 1 to SIZE(population) do
        x ← RANDOM-SELECTION(population, FITNESS-FN)
        y ← RANDOM-SELECTION(population, FITNESS-FN)
        child ← REPRODUCE(x, y)
        if (small random probability) then child ← MUTATE(child)
        add child to new_population
     population ← new_population
  until some individual is fit enough, or enough time has elapsed
  return the best individual in population, according to FITNESS-FN
```

```
function REPRODUCE(x, y) returns an individual
  inputs: x, y, parent individuals

  n ← LENGTH(x); c ← random number from 1 to n
  return APPEND(SUBSTRING(x, 1, c), SUBSTRING(y, c + 1, n))
```

Required Reading: AIMA - Chapter 3.3, 3.4, 3.5,3.6, 4.1, 4.2, 4.3

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

BITS Pilani
Pilani Campus

Artificial & Computational Intelligence

AIML CLZG557

M3 : Game Playing

Raja vadhana P
Assistant Professor,
BITS - CSIS

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

M5 Probabilistic Representation and Reasoning

M6 Reasoning over time

M7 Ethics in AI

Module 3 : Searching to play games

A. Minimax Algorithm

B. Alpha-Beta Pruning

C. Making imperfect real time decisions

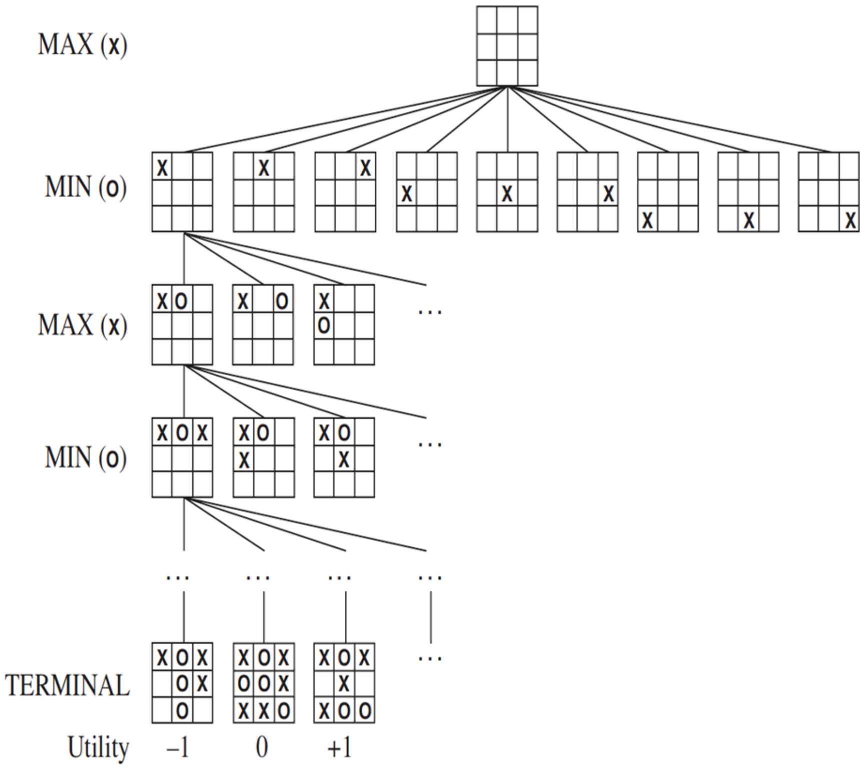
Learning Objective

At the end of this class , students Should be able to:

1. Convert a given problem into adversarial search problem
2. Formulate the problem solving agent components
3. Design static evaluation function value for a problem
4. Construct a Game tree
5. Apply Min-Max
6. Apply and list nodes pruned by alpha pruning and nodes pruned by beta pruning

Task Environment

Phases of Solution Search by PSA



Assumptions – Environment :

Static (4.5)

Observable

Discrete (4.4)

Deterministic

Number of Agents

Game Problem

Study & design of games enables the computers to model ways in which humans think & act hence simulating human intelligence.

<u>AI for Gaming:</u>

- Interesting & Challenging Problem
- Larger Search Space Vs Smaller Solutions
- Explore to better the Human Computer Interaction

<u>Characteristics of Games:</u>

- Observability
- Stochasticity
- Time granularity
- Number of players

<u>Adversarial Games:</u>

Goals of agents are in conflict where one's optimized step would reduce the utility value of the other.

Single Player Game

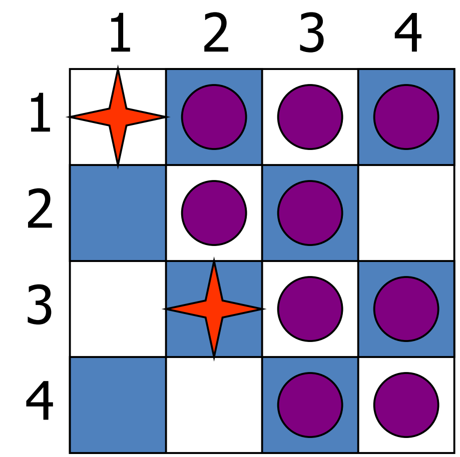
as Constraint Satisfaction Problem
An Overview - Sudoku

Problem Formulation

- Total variables = 81
  - One for each square
  - X = {A1, A2, A3, ..., A9, B1, B2, ...}
- Domains
  - Empty Squares has {1, 2, 3, 4, 5, 6, 7, 8, 9}
  - Filled Squares has only one value as provided
- Constraints
  - A1 – A9 should all be distinct, ...
  - A1 – I1 should all be distinct, ...
  - A1-3, B1-3, C1-3 should all be distinct

	1	2	3	4	5	6	7	8	9
A			3		2		6		
B	9			3		5			1
C			1	8		6	4		
D			8	1		2	9		
E	7								8
F			6	7		8	2		
G			2	6		9	5		
H	8			2		3			9
I			5		1		3		

Local Search Technique



	1	2	3	4	5	6	7	8	9
A			3		2		6		
B	9			3		5			1
C			1	8		6	4		
D			8	1		2	9		
E	7								8
F			6	7		8	2		
G			2	6		9	5		
H	8			2		3			9
I			5		1		3		

Sample fitness value :
No.of.Contradicting constraints or No.of,satisfying constraints

Constraint Satisfaction Graph - Subgraph

Sudoku

	1	2	3	4
A		2	1	
B	4			
C				2
D		3		

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	1	3	2	3

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	1	3	1	3

2+2

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	1	3	2	3

1+2

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	1	3	3	3

3+2

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	1	3	4	3

2

Sudoku

A		2	1	
B	4			
C				2
D		3		

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	1	3	2	3

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	1	3	1	3

2+2

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	1	3	2	3

1+2

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	1	3	3	3

3+2

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	1	3	4	3

2

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	1	3	4	3

0

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	2	3	4	3

2

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	3	3	4	3

3+2

	1	2	3	4
A	3	2	1	3
B	4	2	3	4
C	2	4	4	2
D	4	3	4	3

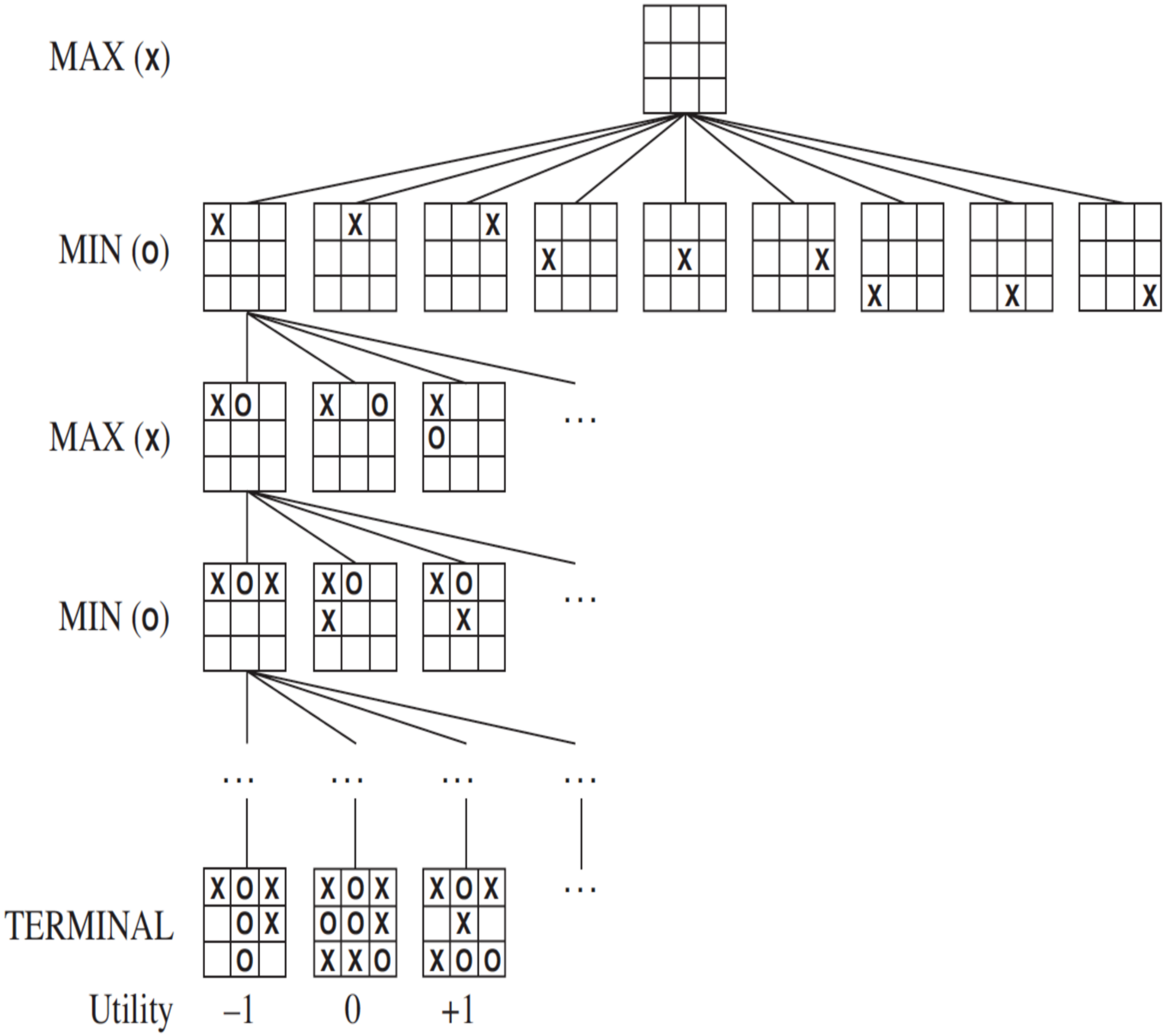
3+2

Games as Search Problem

PSA : Representation of Game:

- INITIAL STATE: S0
- PLAYER(s)
- ACTIONS(s)
- RESULT(s, a)
- TERMINAL-TEST(s)
- UTILITY(s, p)

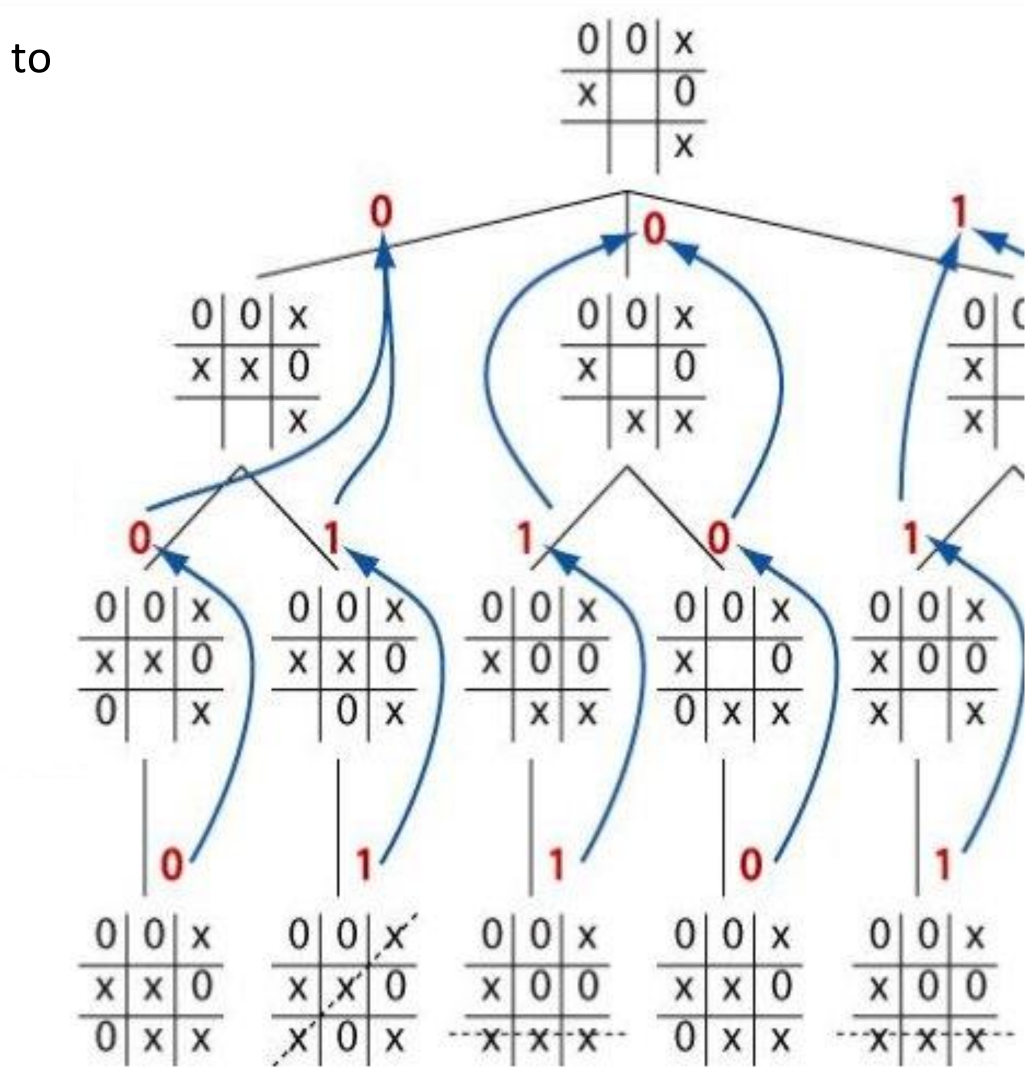
Eg., Tic Tac Toe

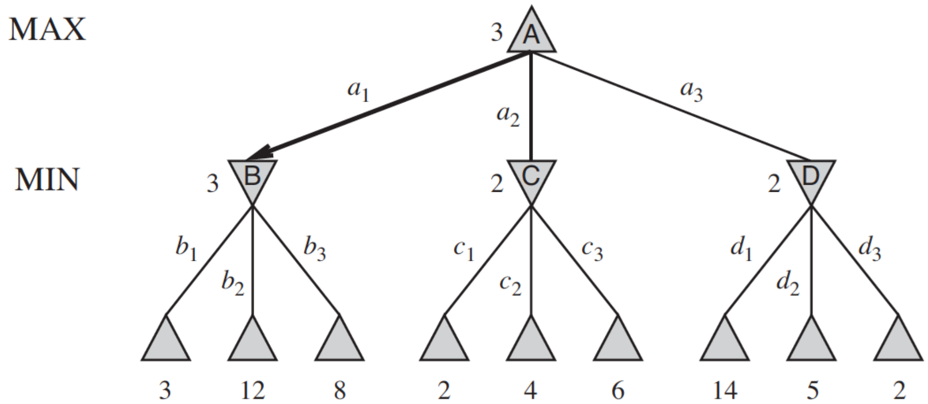


Min-Max Algorithm

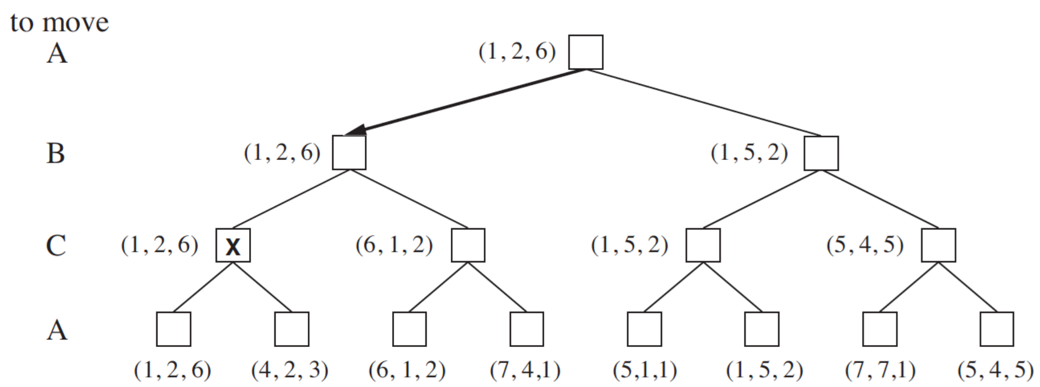
Idea: Uses Depth – First search exploration to decide the move

Let
start Player = MAX
Depth m =3





Two Player Game : 1-Ply Game



Multiplayer Game

Min-Max Algorithm

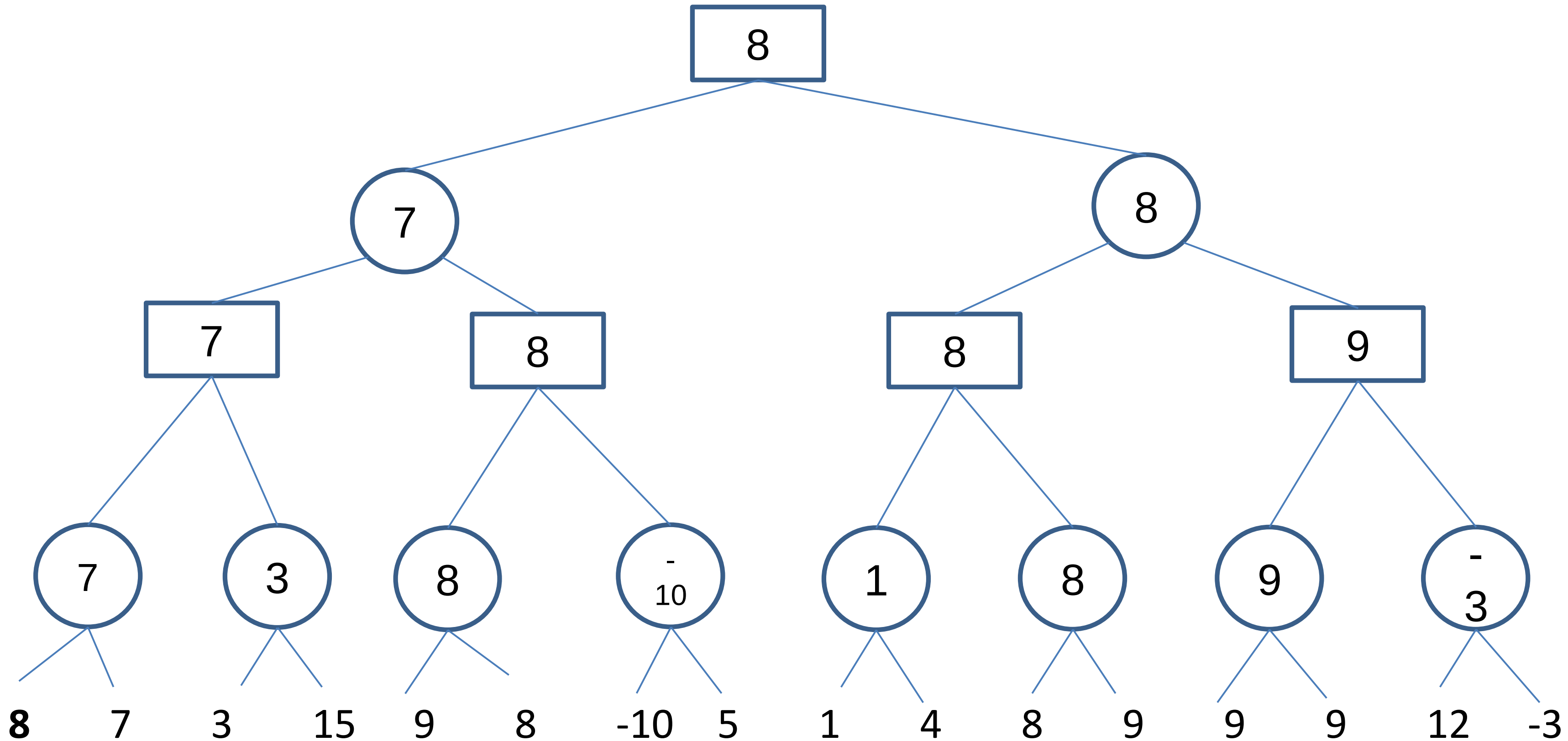
```
function MINIMAX-DECISION(state) returns an action
  return arg max_{a ∈ ACTIONS(s)} MIN-VALUE(RESULT(state, a))
```

```
function MAX-VALUE(state) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← −∞
  for each a in ACTIONS(state) do
    v ← MAX(v, MIN-VALUE(RESULT(s, a)))
  return v
```

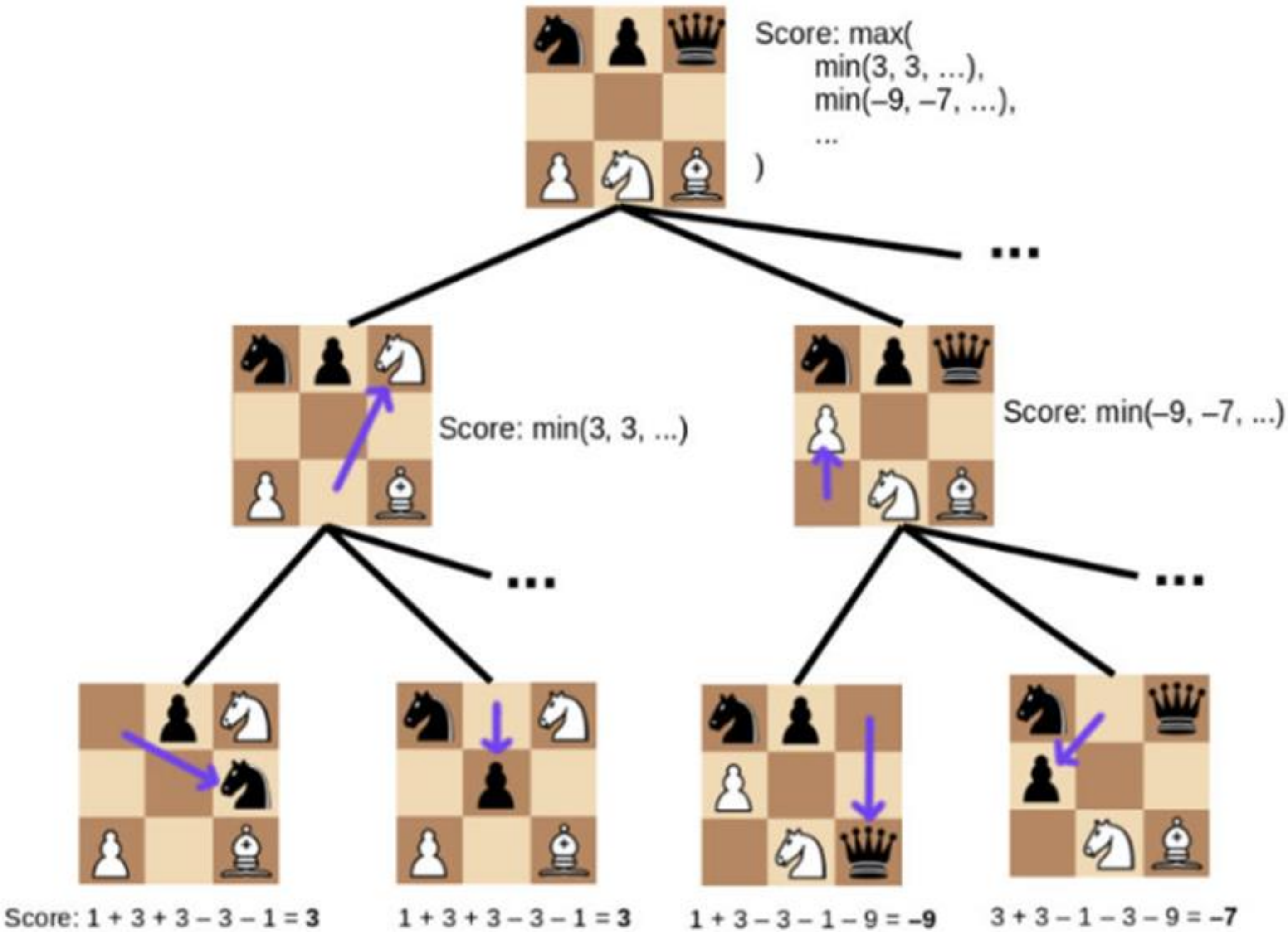
```
function MIN-VALUE(state) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← ∞
  for each a in ACTIONS(state) do
    v ← MIN(v, MAX-VALUE(RESULT(s, a)))
  return v
```

Min-Max Algorithm – Example -1

Squares represent MAX nodes
Circles represent MIN nodes

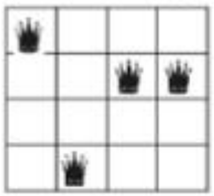


Design of Static Evaluation Values



Design of Static Evaluation Values

N-Queens



1	4	2	2	4

Tic-Tac-Toe

0	0	x
x		0
		x

Max's Share	2
Min's Share	1
Board Value	1

N-Tile

2	8	3
1	6	4
7		5

1	2	3
8		4
7	6	5

No.of.Tiles Out of Place	5

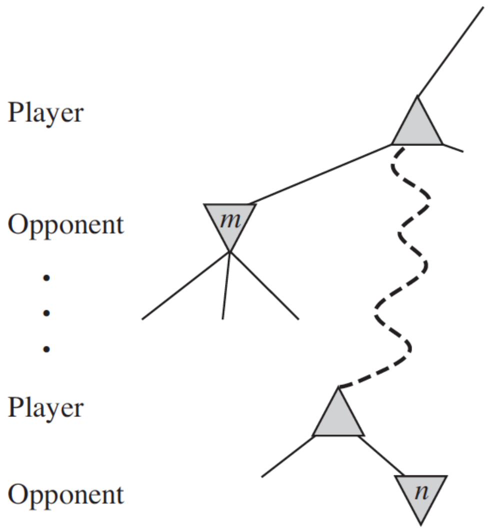
$$\text{Eval}(S) = w_1 f_1(s) + w_2 f_2(s) + \ldots\ldots + w_n f_n(s)$$

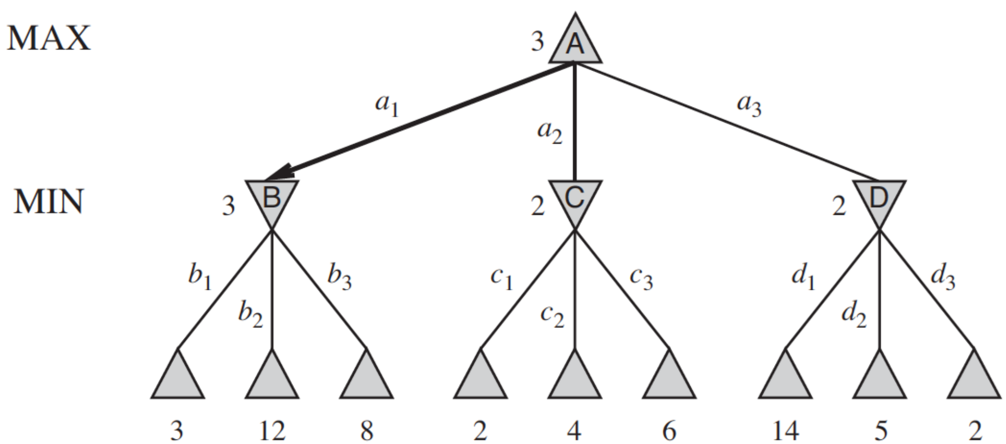
$$= 0.6\,(\text{MaxChance} - \text{MinChance}) + 0.4\,(\text{MaxPairs} - \text{MinPairs})$$

Alpha – beta Pruning

General Principle:

At a node ***n*** if a player has better option at the parent of ***n*** or further up, then ***n*** node will never be reached .Hence the entire subtree pruned

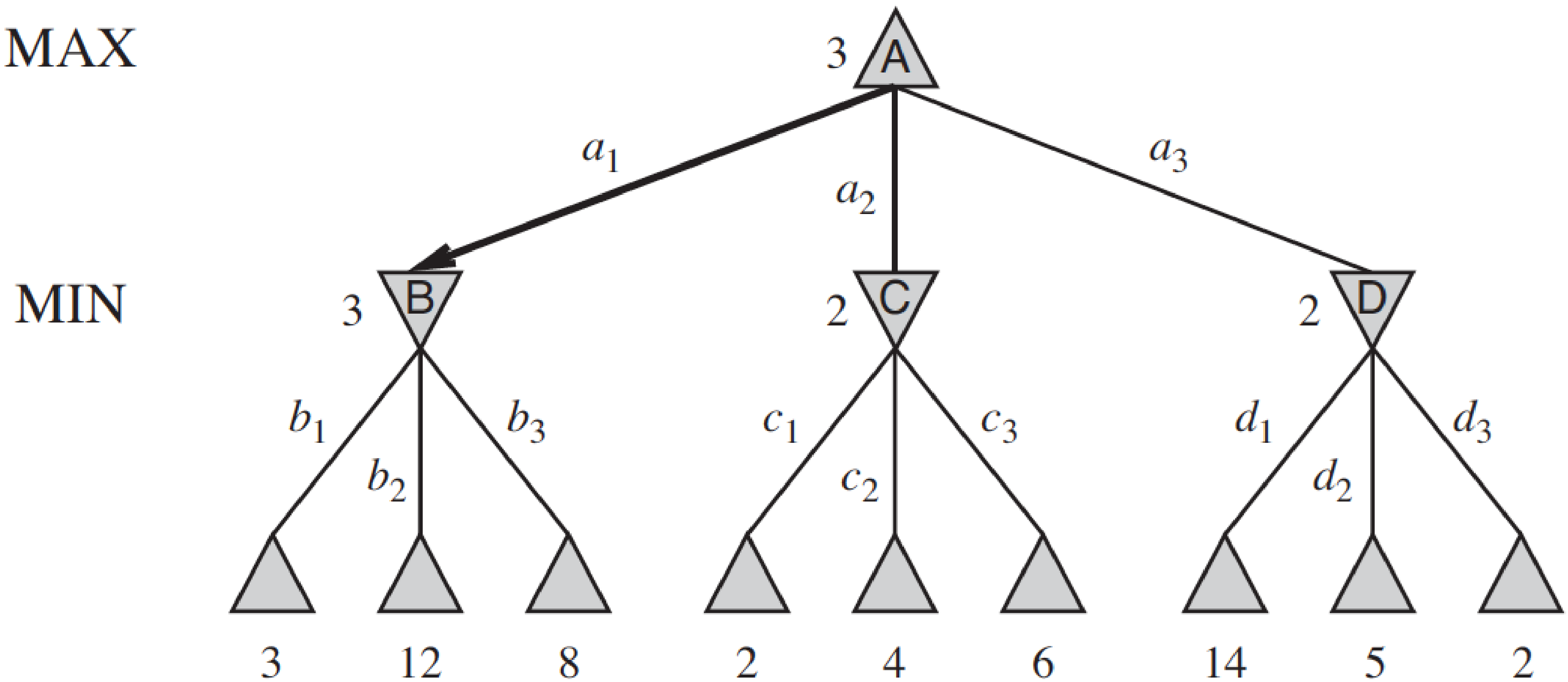




$$
\begin{aligned}
\textsc{Minimax}(root) &= \max(\min(3,12,8), \min(2,x,y), \min(14,5,2)) \\
&= \max(3, \min(2,x,y), 2) \\
&= \max(3, z, 2) \qquad \text{where } z = \min(2,x,y) \leq 2 \\
&= 3.
\end{aligned}
$$

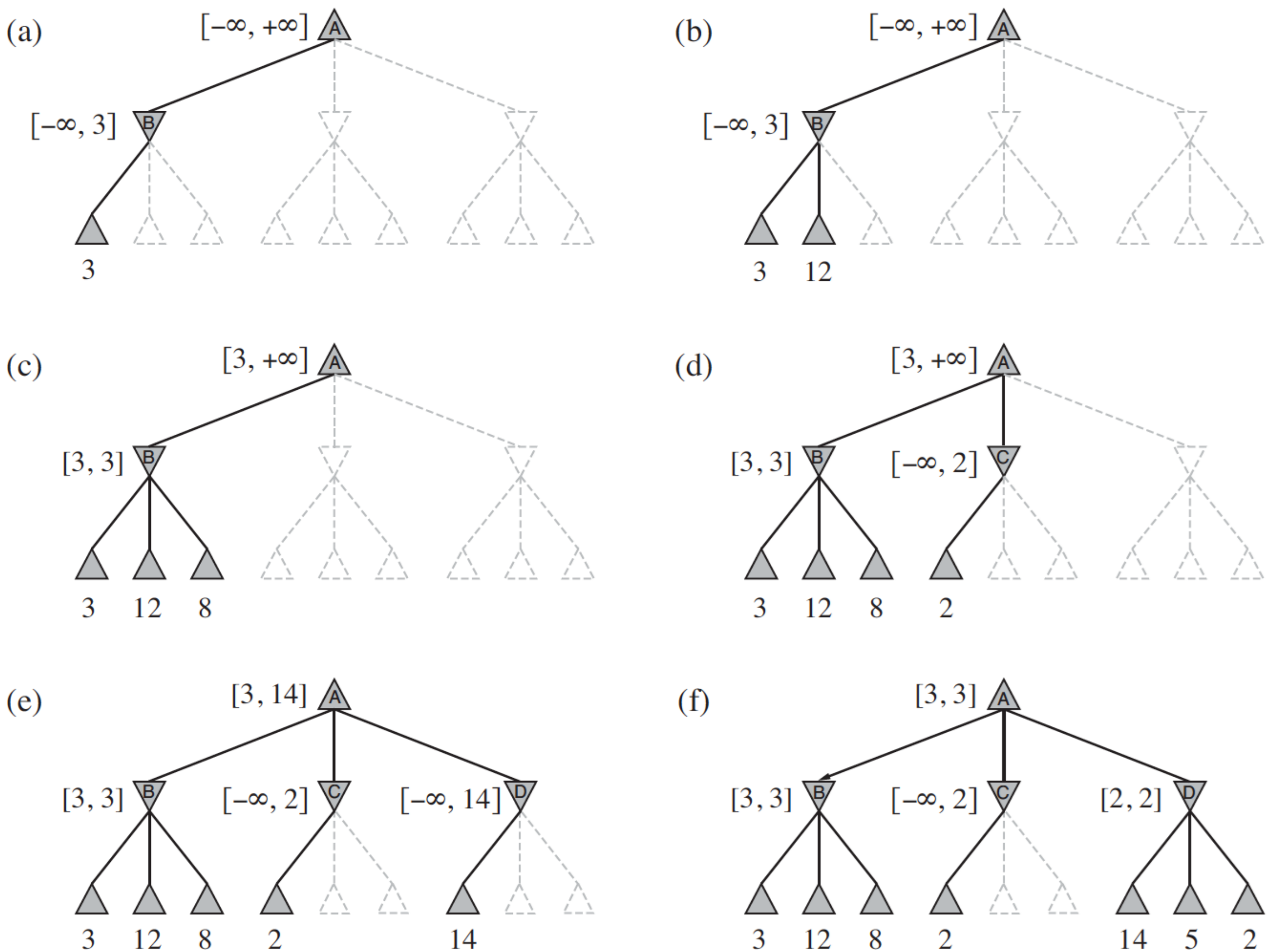
Alpha Beta Pruning

Book Example



Alpha Beta Pruning

Book Example



Min-Max Algorithm

Alpha beta Modifications

```
function ALPHA-BETA-SEARCH(state) returns an action
   v ← MAX-VALUE(state, −∞, +∞)
   return the action in ACTIONS(state) with value v
```

```
function MAX-VALUE(state, α, β) returns a utility value
   if TERMINAL-TEST(state) then return UTILITY(state)
   v ← −∞
   for each a in ACTIONS(state) do
      v ← MAX(v, MIN-VALUE(RESULT(s,a), α, β))
      if v ≥ β then return v
      α ← MAX(α, v)
   return v
```

```
function MIN-VALUE(state, α, β) returns a utility value
   if TERMINAL-TEST(state) then return UTILITY(state)
   v ← +∞
   for each a in ACTIONS(state) do
      v ← MIN(v, MAX-VALUE(RESULT(s,a) , α, β))
      if v ≤ α then return v
      β ← MIN(β, v)
   return v
```

Is it possible to compute the minimax decision for a node without looking at every successor node?

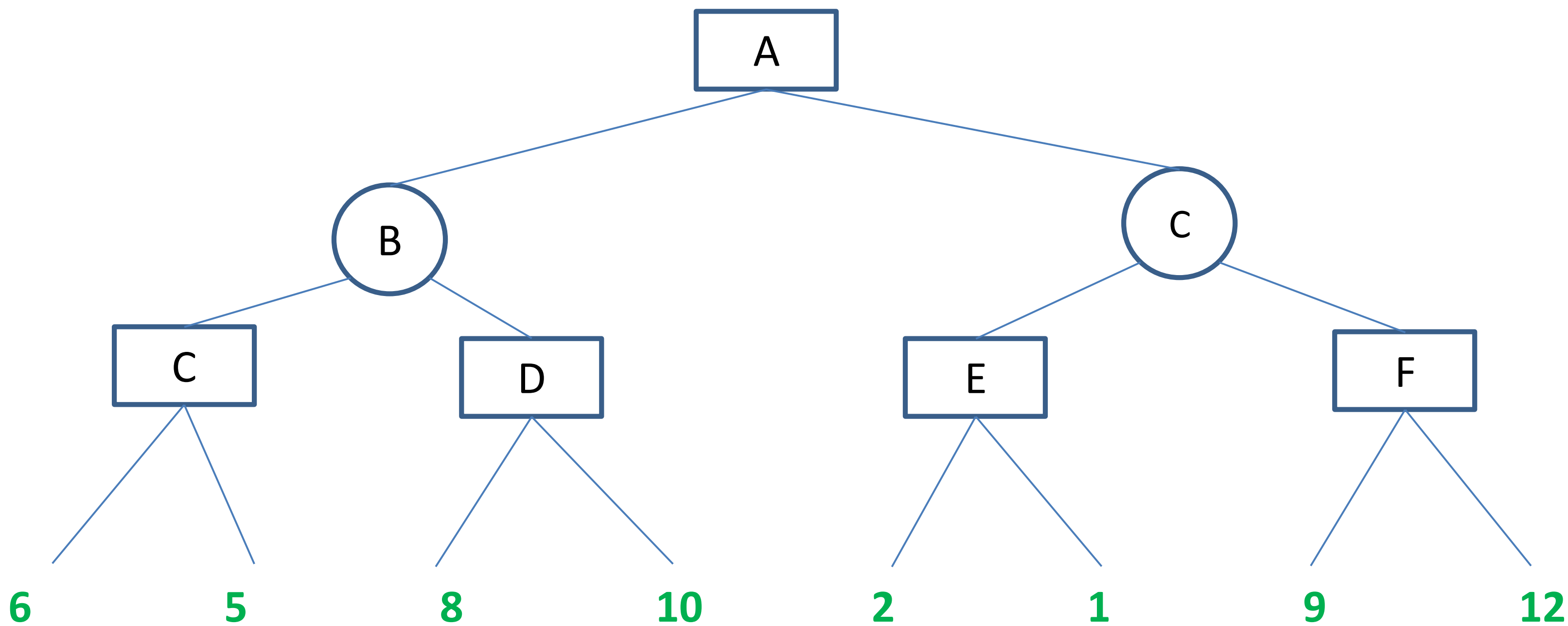
Alpha – beta Pruning

Steps in Alpha – Beta Pruning:

1. At root initialize alpha = -∞ and beta = + ∞. This is to set the worst case boundary to start the algorithm which aims to increase alpha and decrease beta as much as optimally possible
2. Navigate till the depth / limit specified and get the static evaluated numeric value.
3. For every value VAL being analyzed : Loop till all the leaf/terminal/specified state level nodes are analyzed & accounted for OR until **beta <= alpha.**
   1. If the player is MAX :
      1. If VAL > alpha
      2. then reset alpha = VAL
      3. also check **if** beta <=alpha **then** tag the path as unpromising (TO BE AVOIDED) **and** prune the branch from game tree. Rest of their siblings are not considered for analysis
   2. Else if the player is MIN:
      1. If VAL < beta
      2. then reset beta = VAL
      3. also check **if** beta <=alpha **then** tag the path as unpromising (TO BE AVOIDED) **and** prune the branch from game tree. Rest of their siblings are not considered for analysis

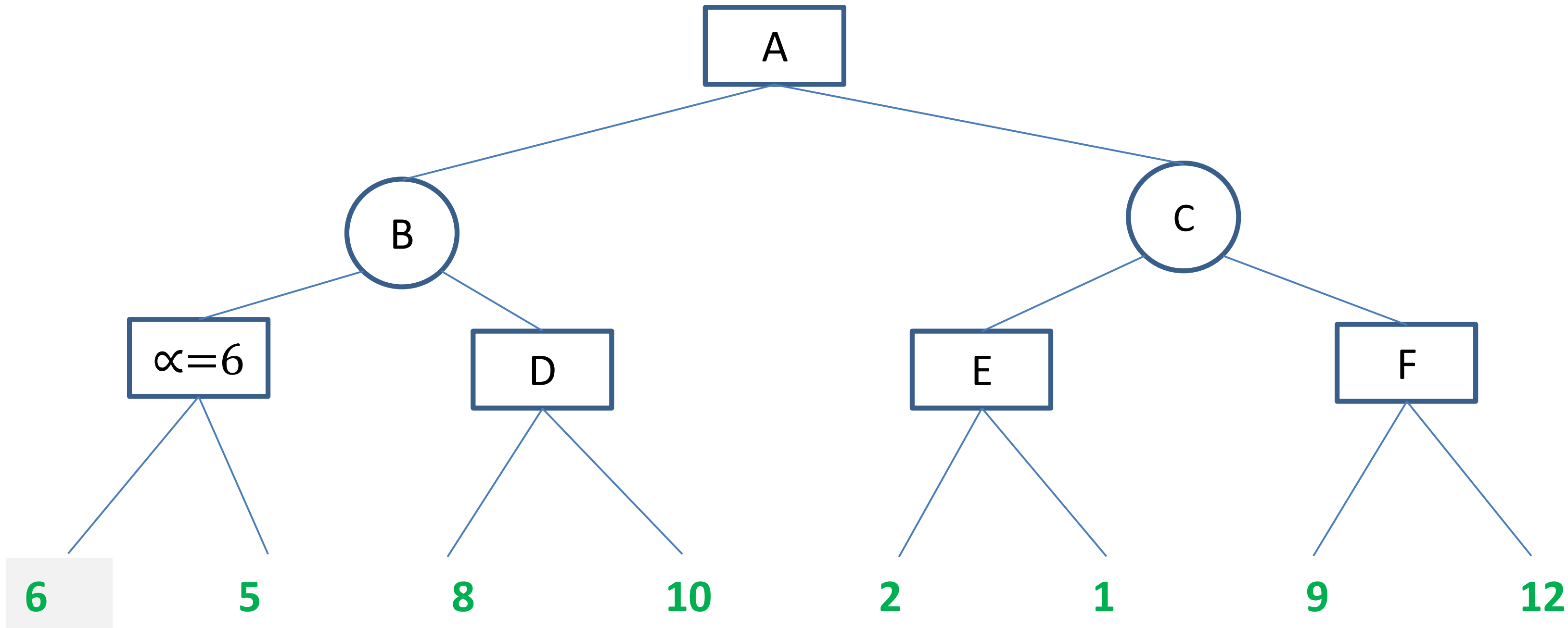
Alpha Beta Pruning - Another Example

Idea –Pruning



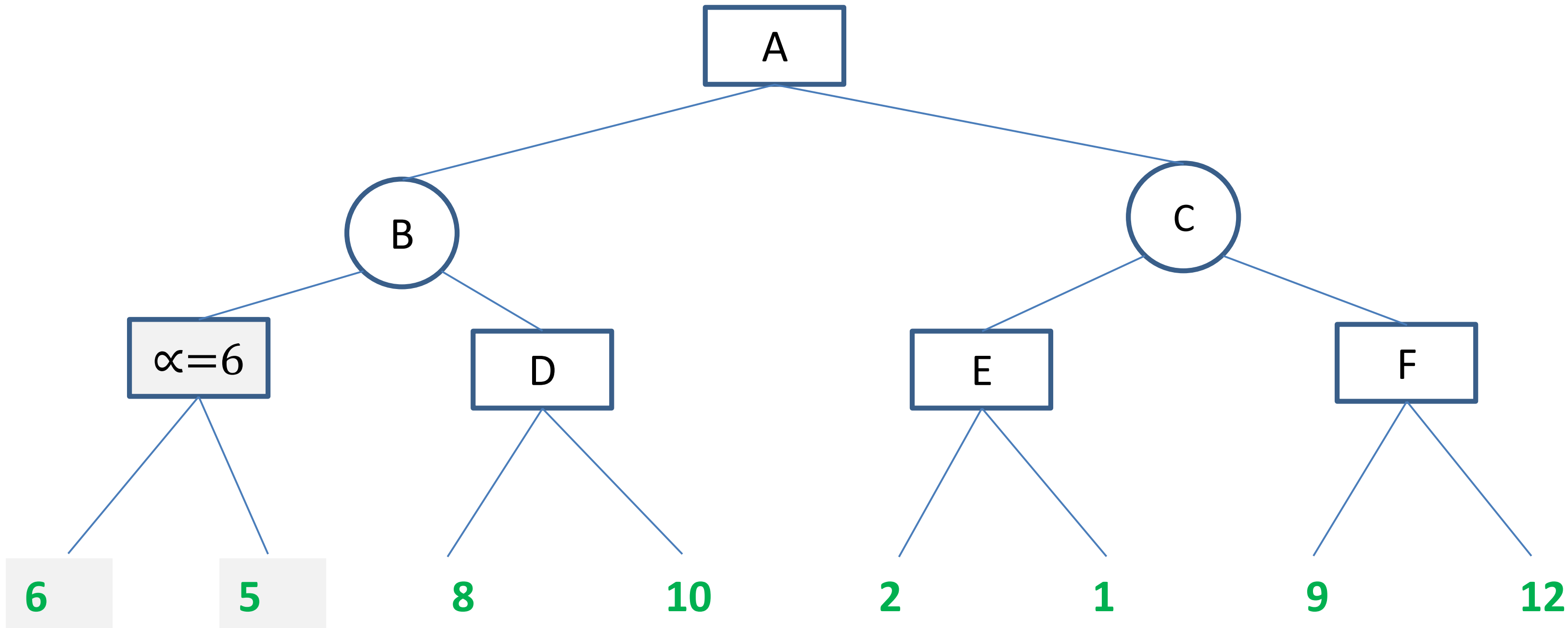
Alpha Beta Pruning

Idea –Pruning



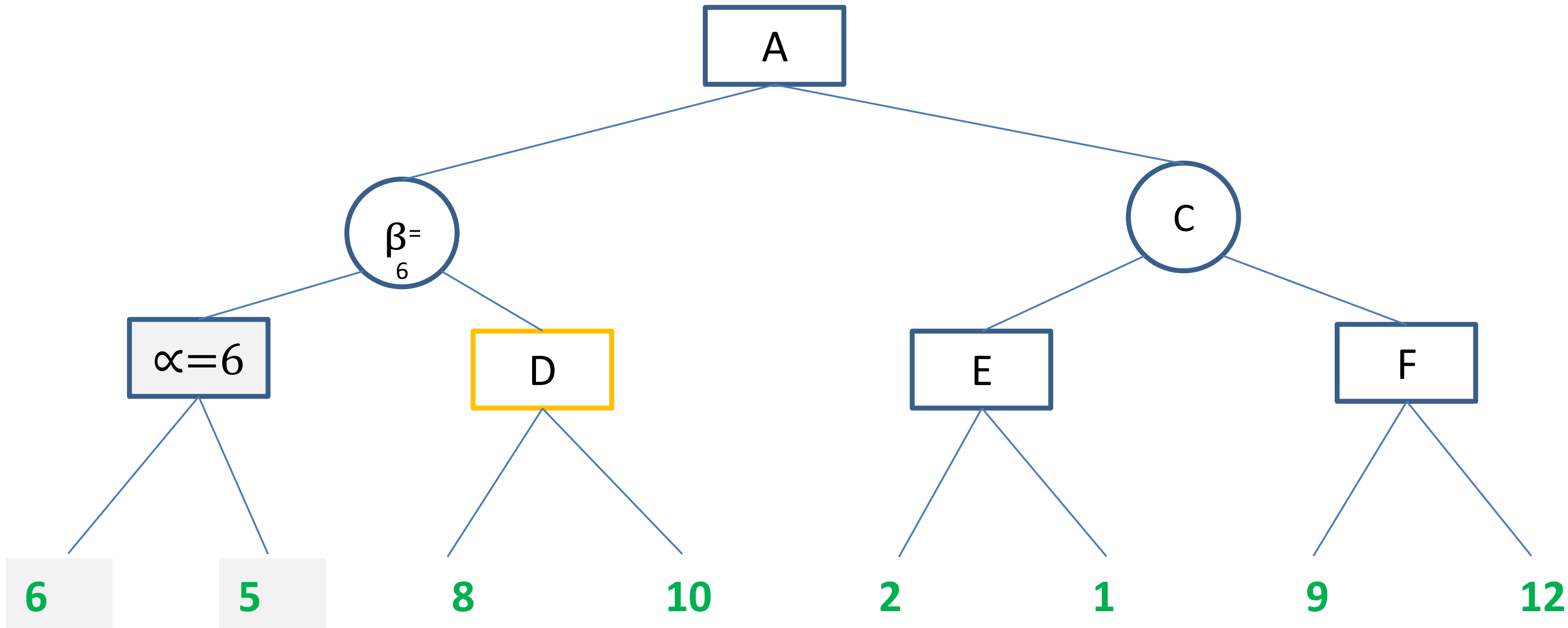
Alpha Beta Pruning

Idea –Pruning



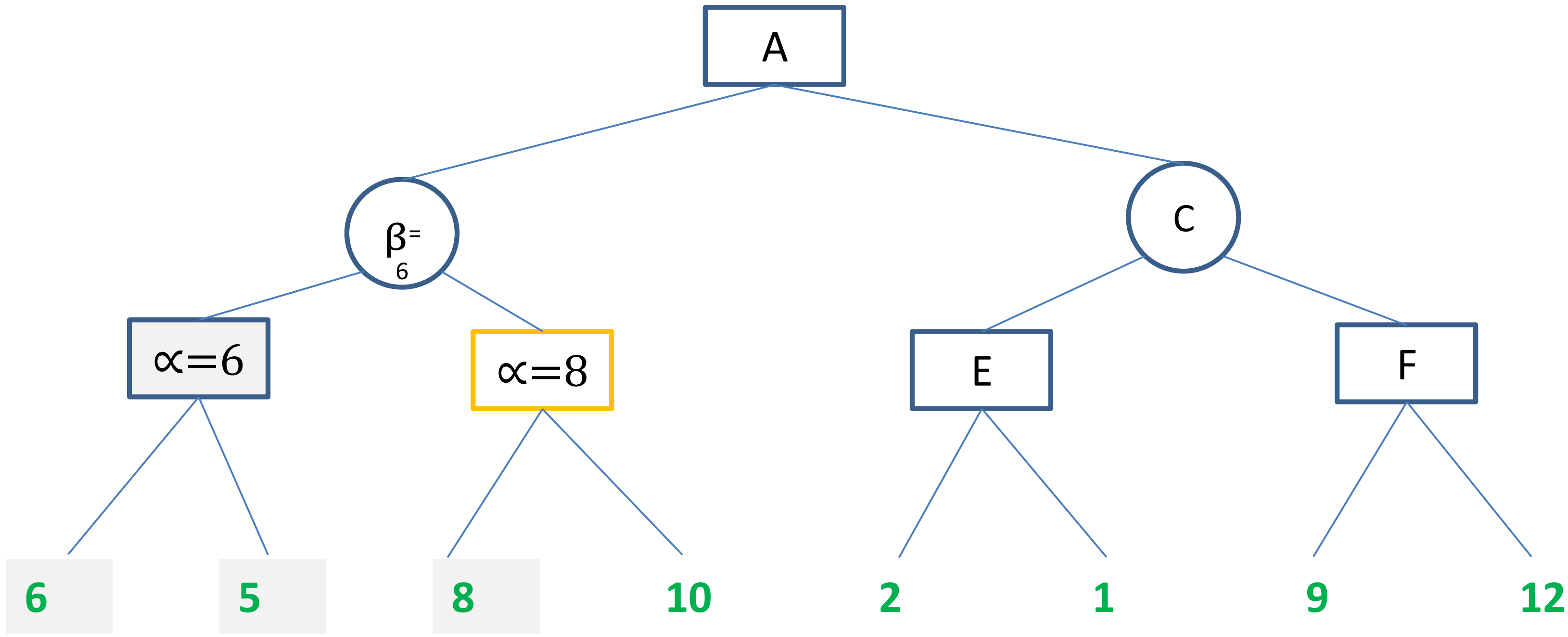
Alpha Beta Pruning

Idea –Pruning

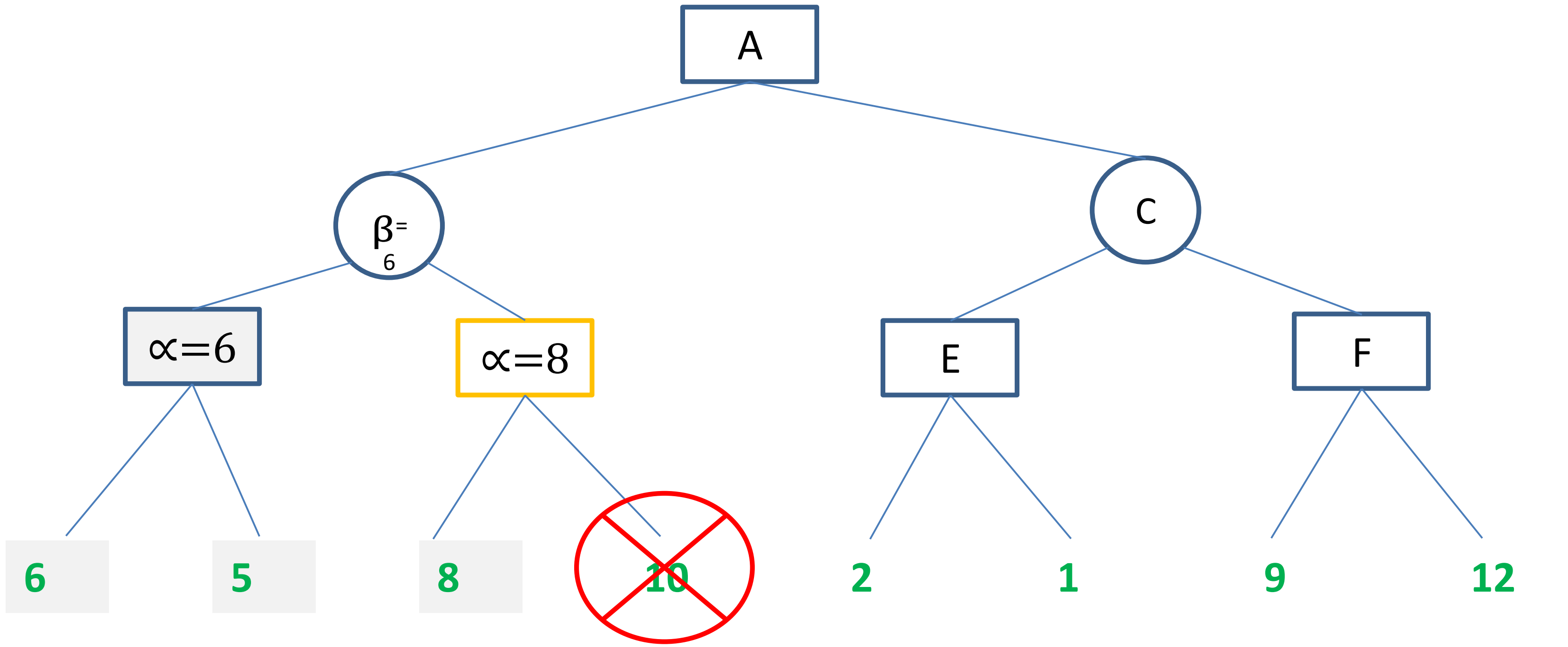


Alpha Beta Pruning

Idea – Alpha Pruning

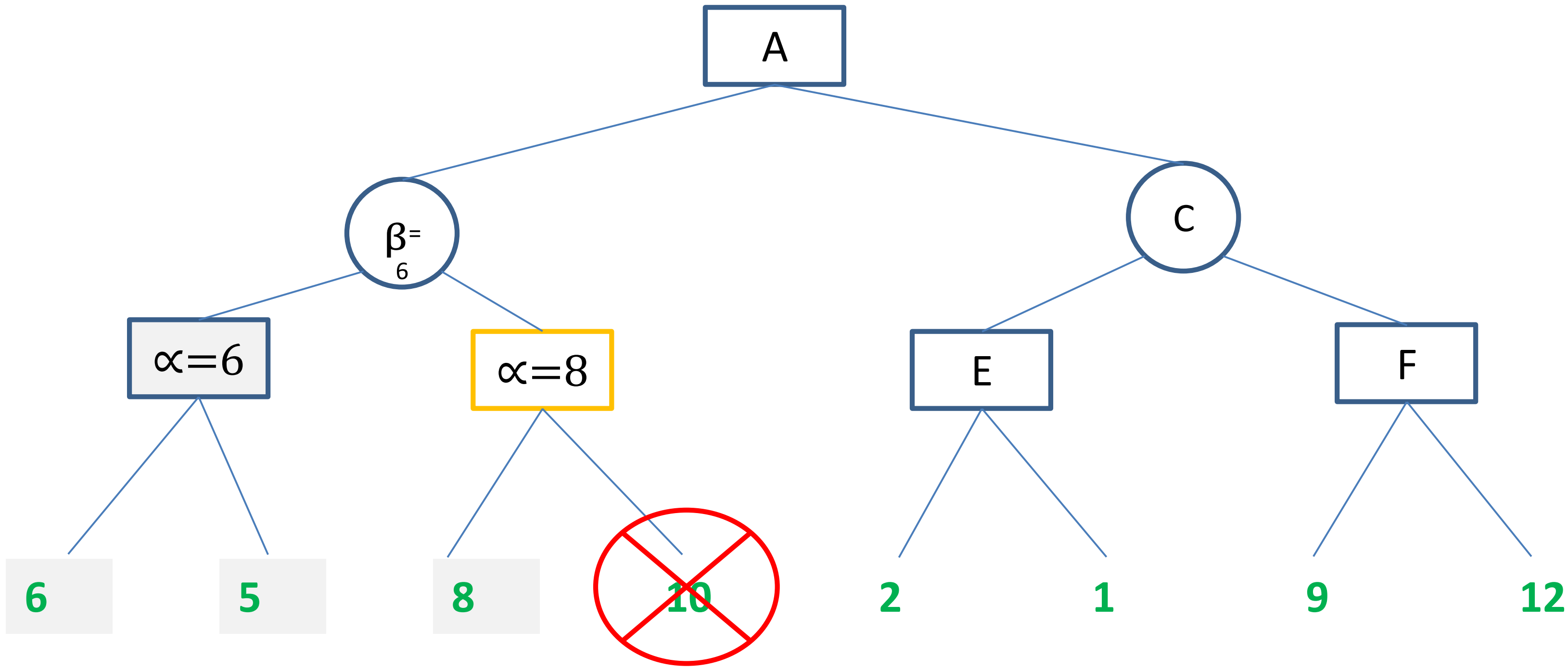


Idea – Beta Pruning

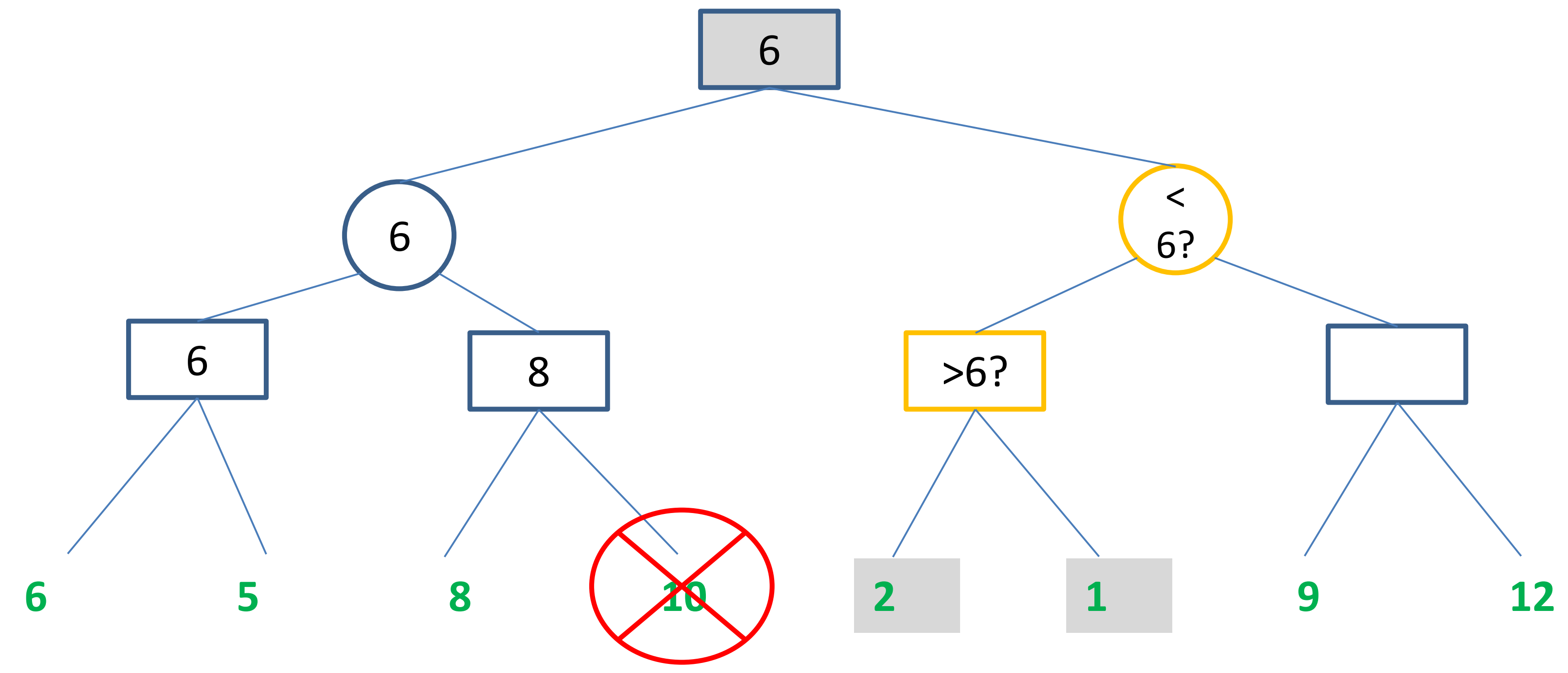


Alpha Beta Pruning

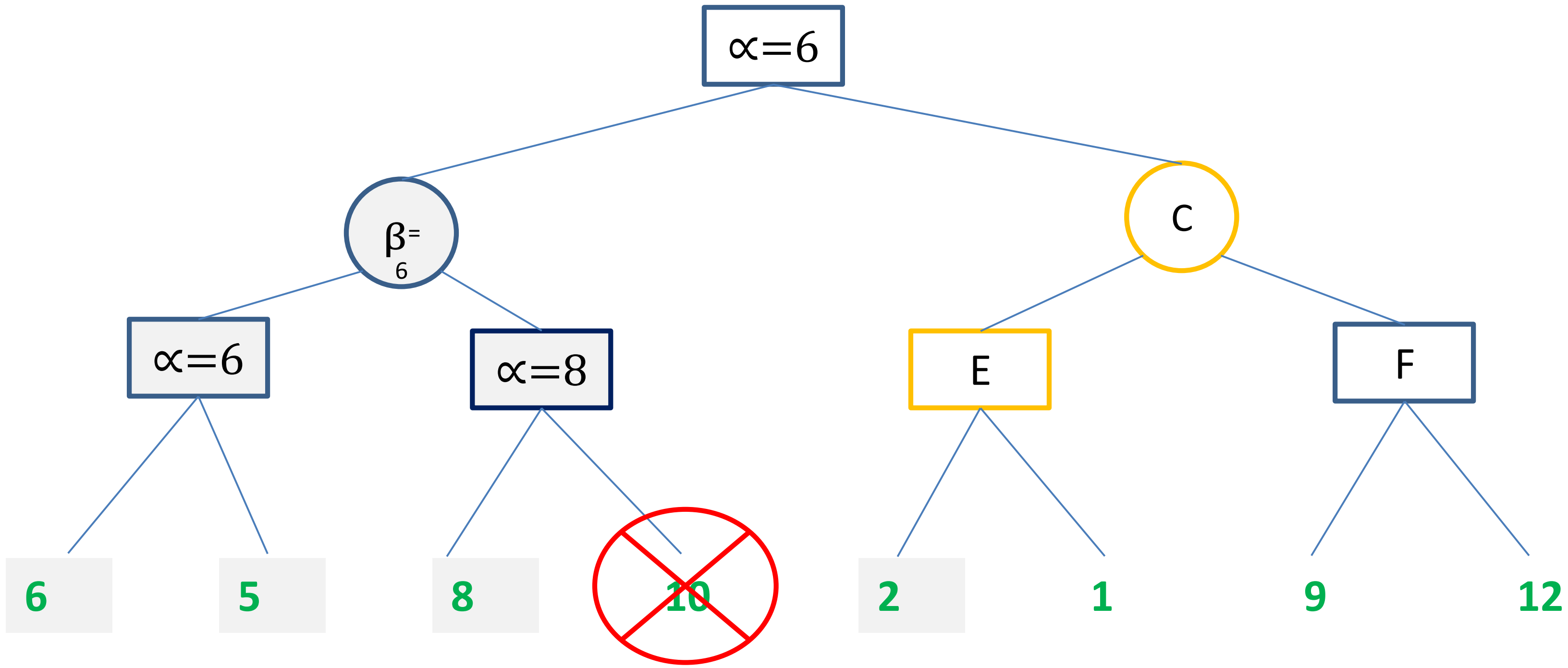
Idea –Pruning



Idea –Pruning

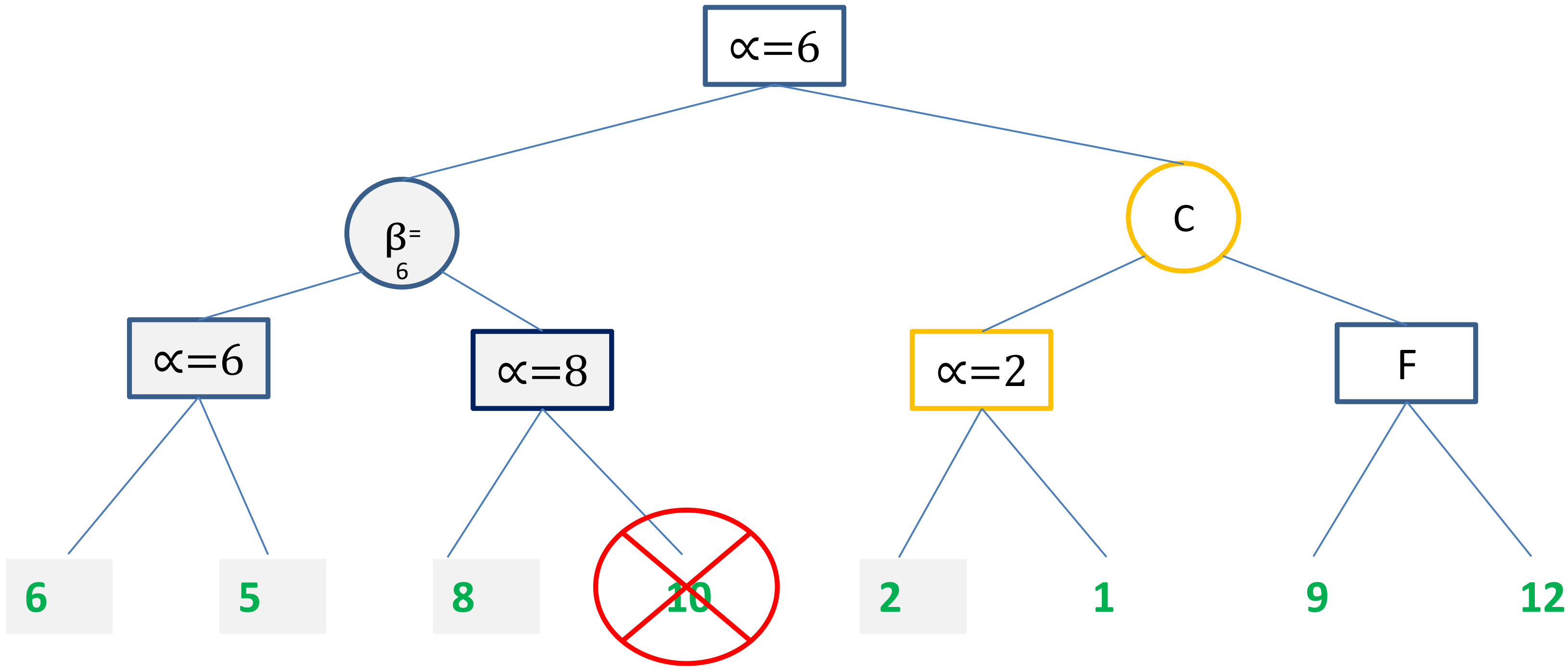


Idea –Pruning

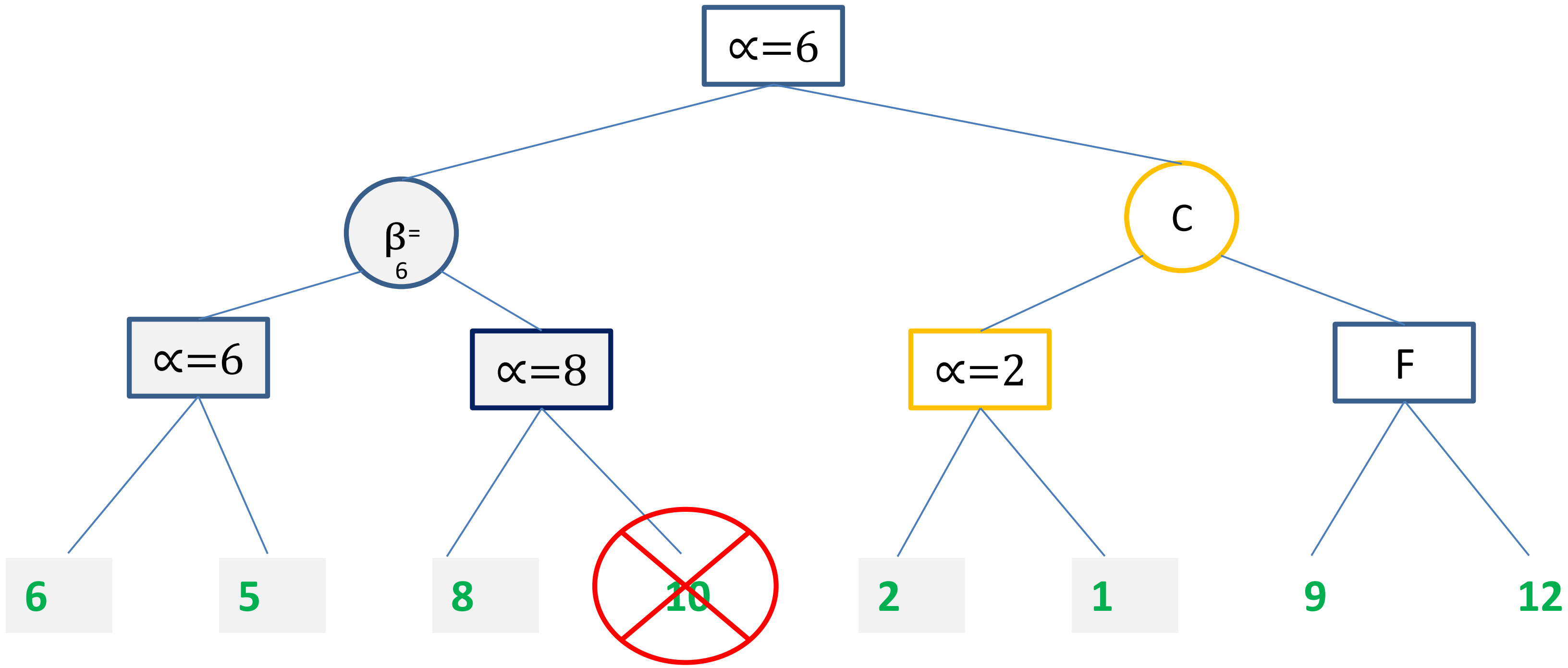


Alpha Beta Pruning

Idea –Pruning

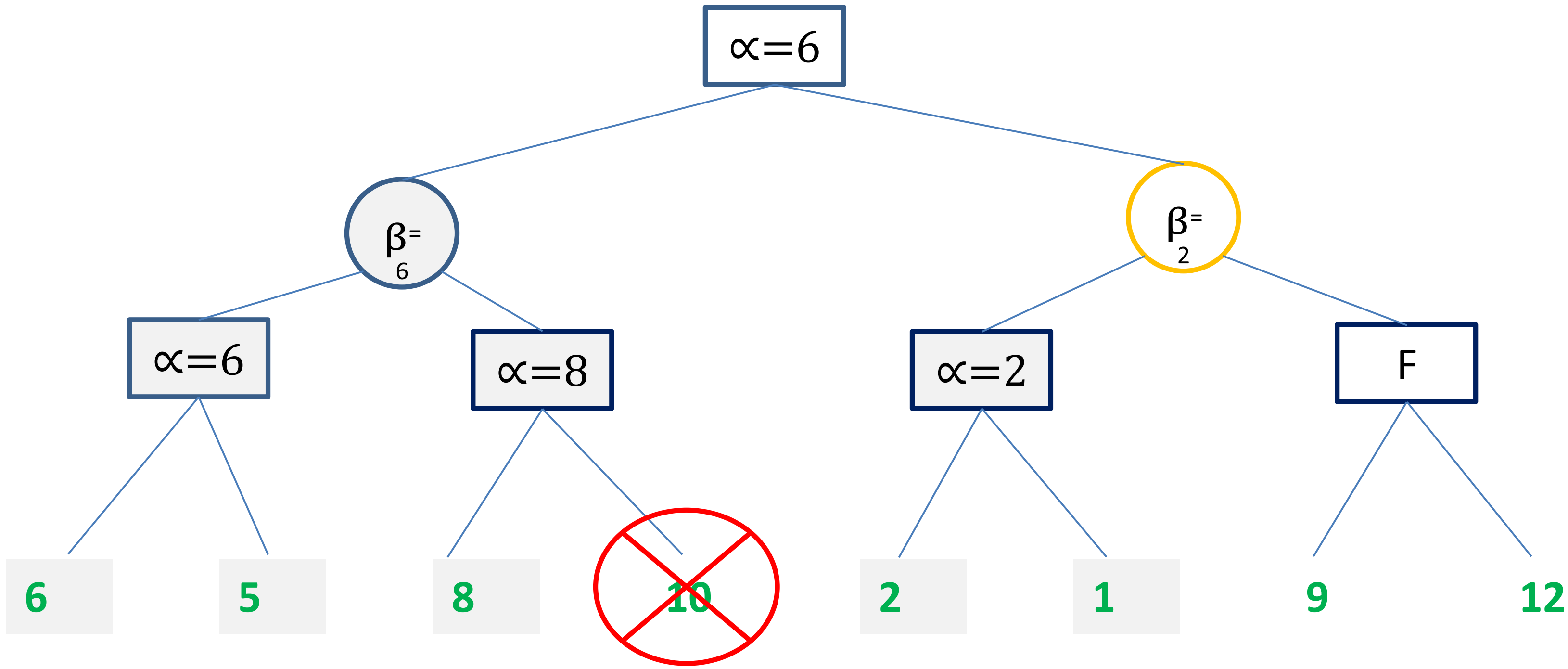


Idea –Pruning

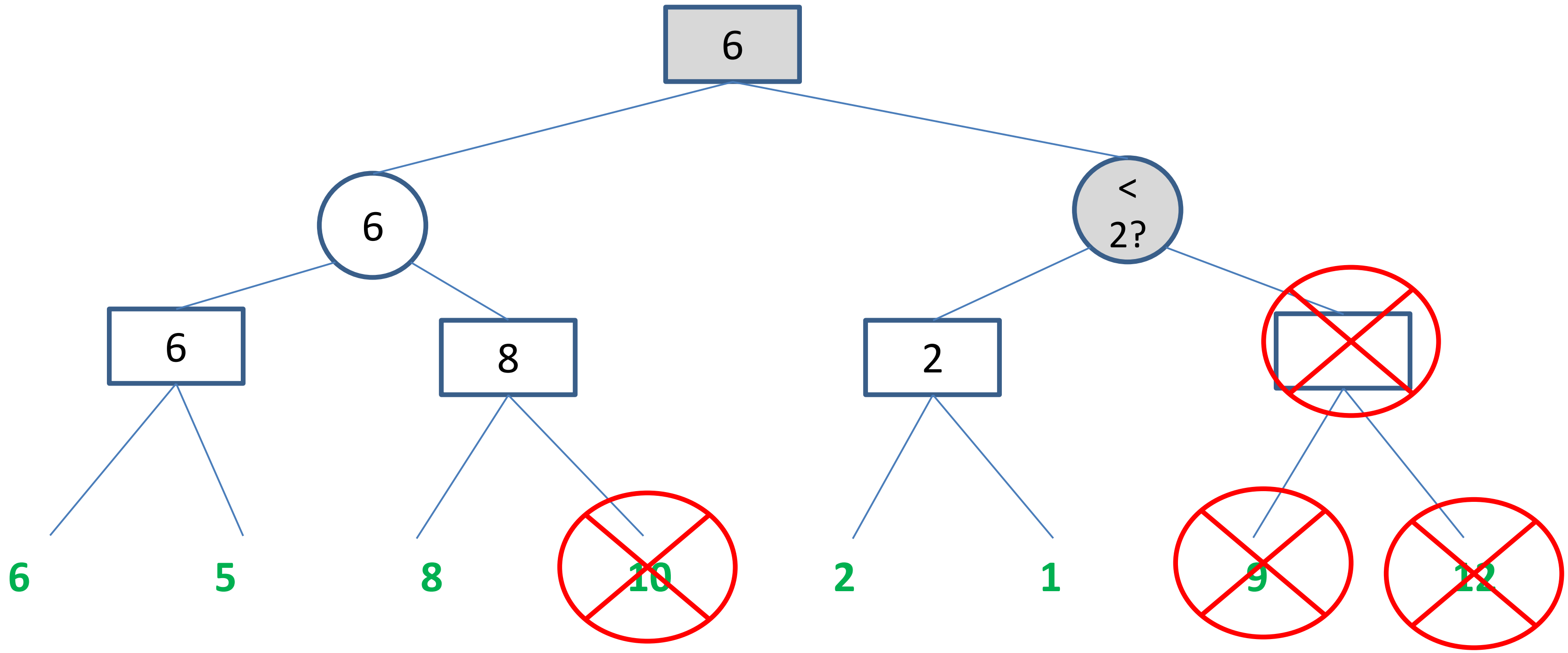


Alpha Beta Pruning

Idea –Pruning



Idea – Alpha Pruning



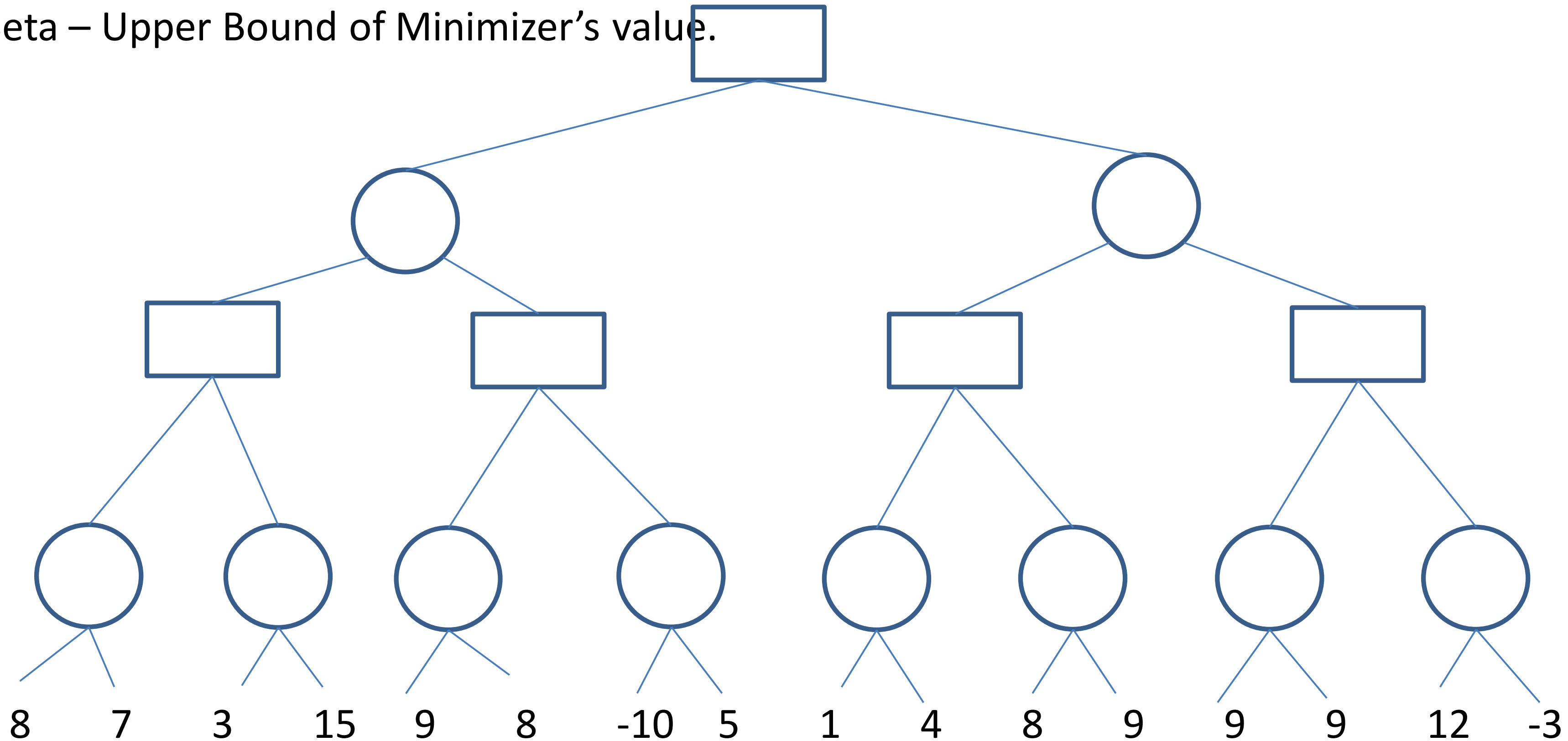
Alpha – Lower bound of Maximizer's value. Perceived value that Maximizer hopes to against a competitive Minimizer

Alpha – beta Pruning – Example -4

Do for practice.

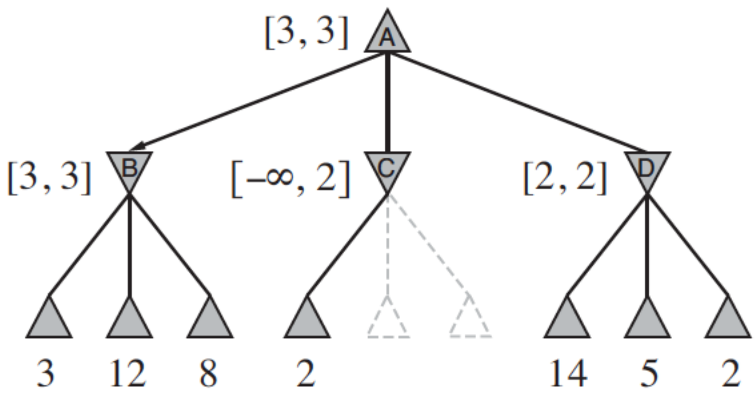
Alpha – Lower bound of Maximizer's value. Perceived value that Maximizer hopes to get with a competitive Minimizer

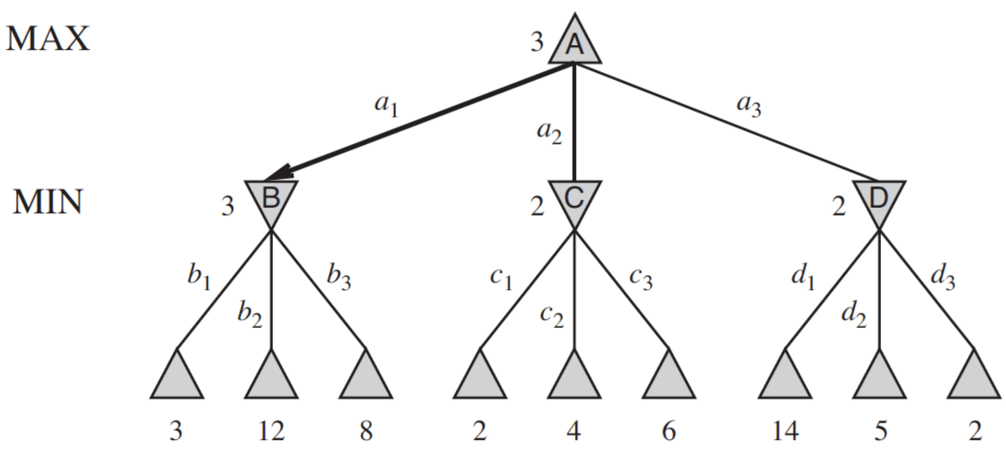
Beta – Upper Bound of Minimizer's value.



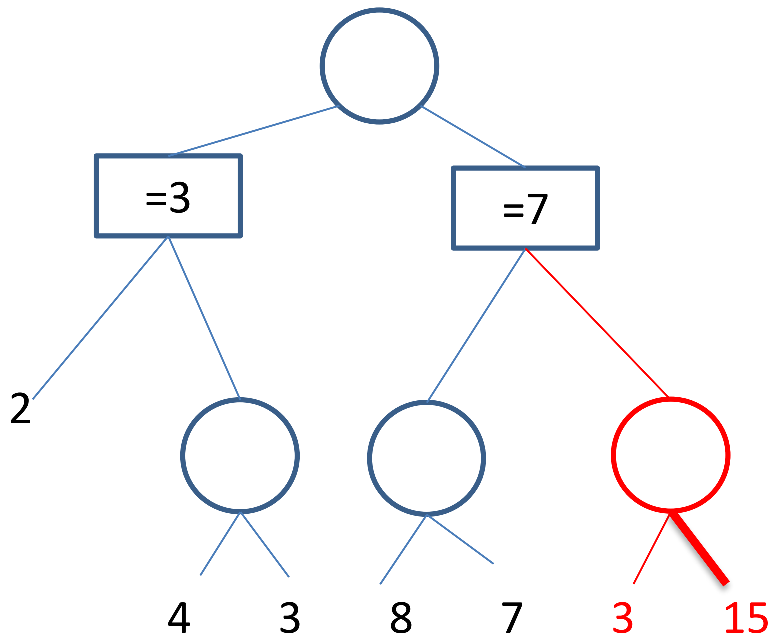
Computational Efficiency

How to reduce the move generations better along while doing Alpha-Beta Pruning?

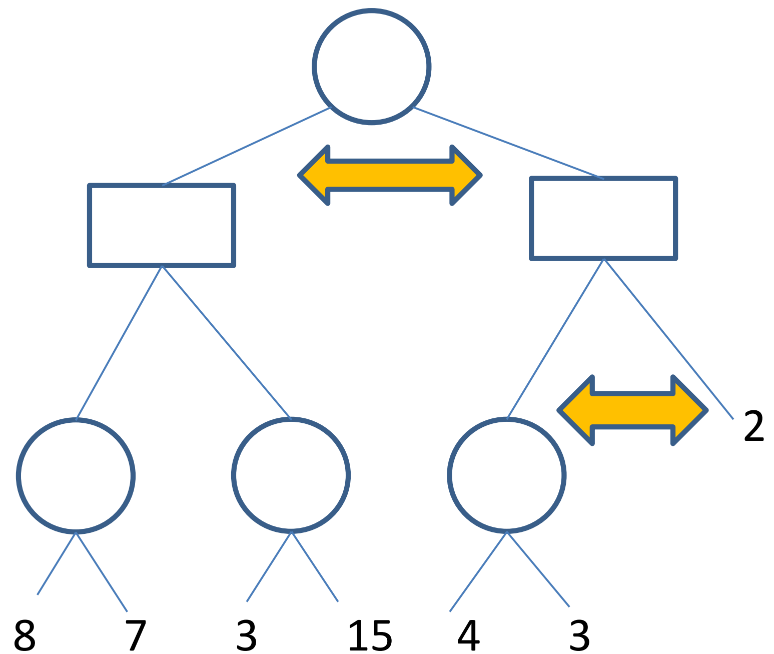




After Move Ordering



Before Move Ordering



Required Reading: AIMA - Chapter # 4.1, #4.2, #5.1

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

Artificial & Computational Intelligence

AIML CLZG557

M4 : Knowledge Representation using Logics

Raja vadhana P

Assistant Professor,

BITS - CSIS

BITS Pilani

Pilani Campus

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

M5 Probabilistic Representation and Reasoning

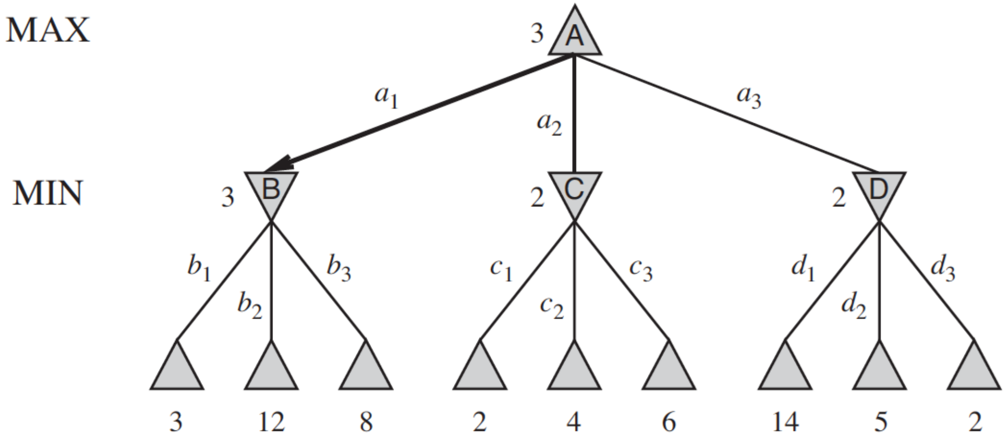
M6 Reasoning over time

M7 Ethics in AI

Gaming
(Imperfect Decisions)

Computational Efficiency

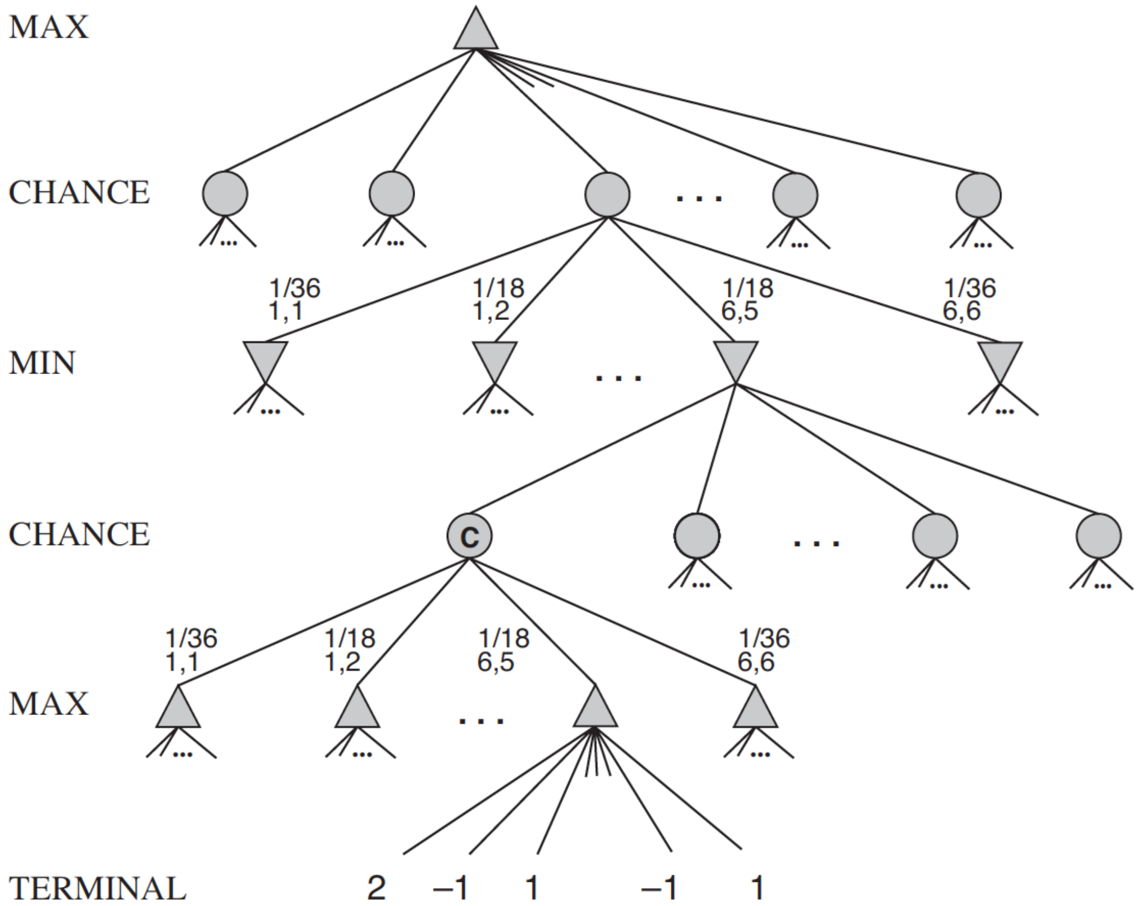
How games can be designed to handle imperfect decisions in real-time?



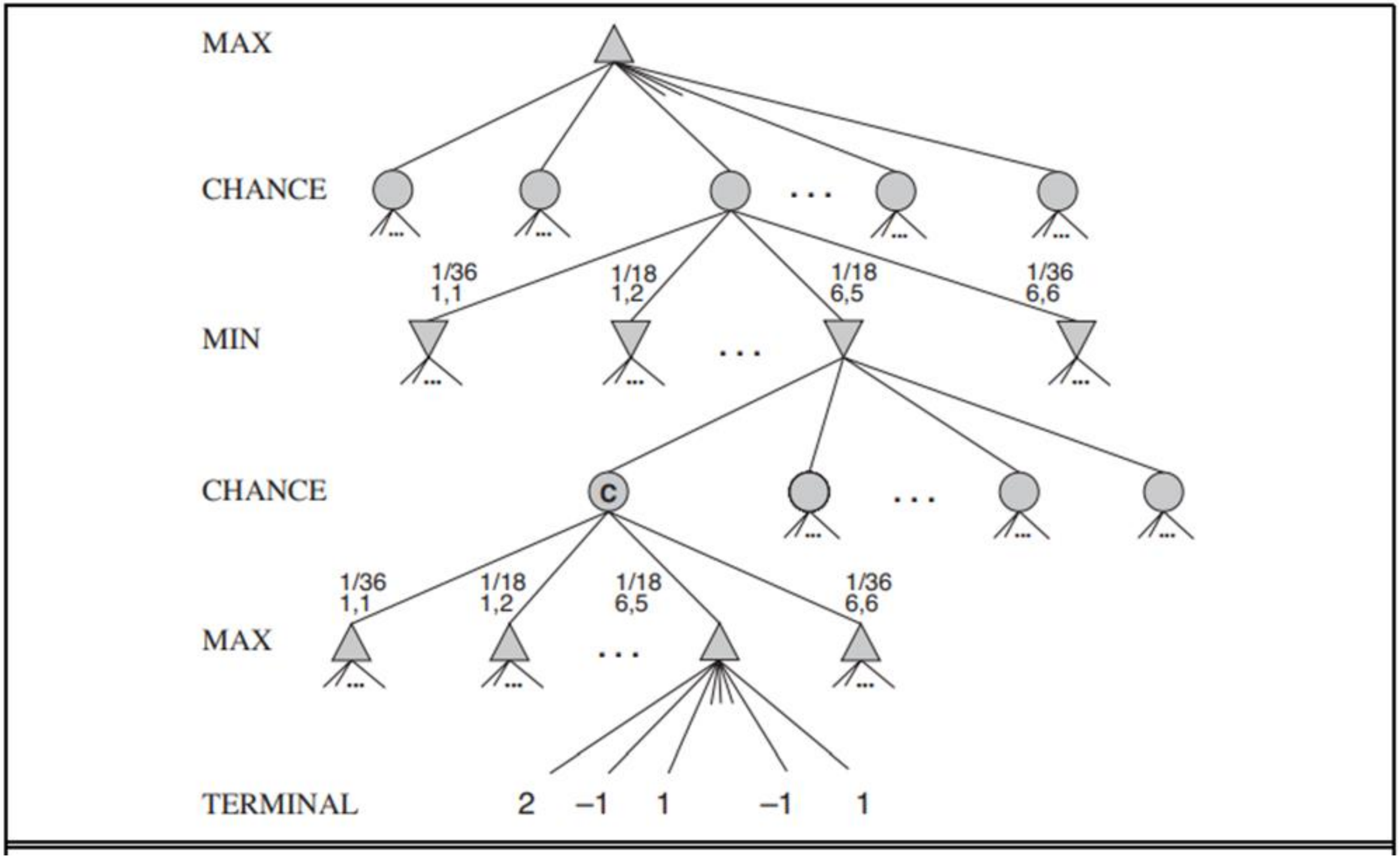
Computational Efficiency

Idea : Chance Node:

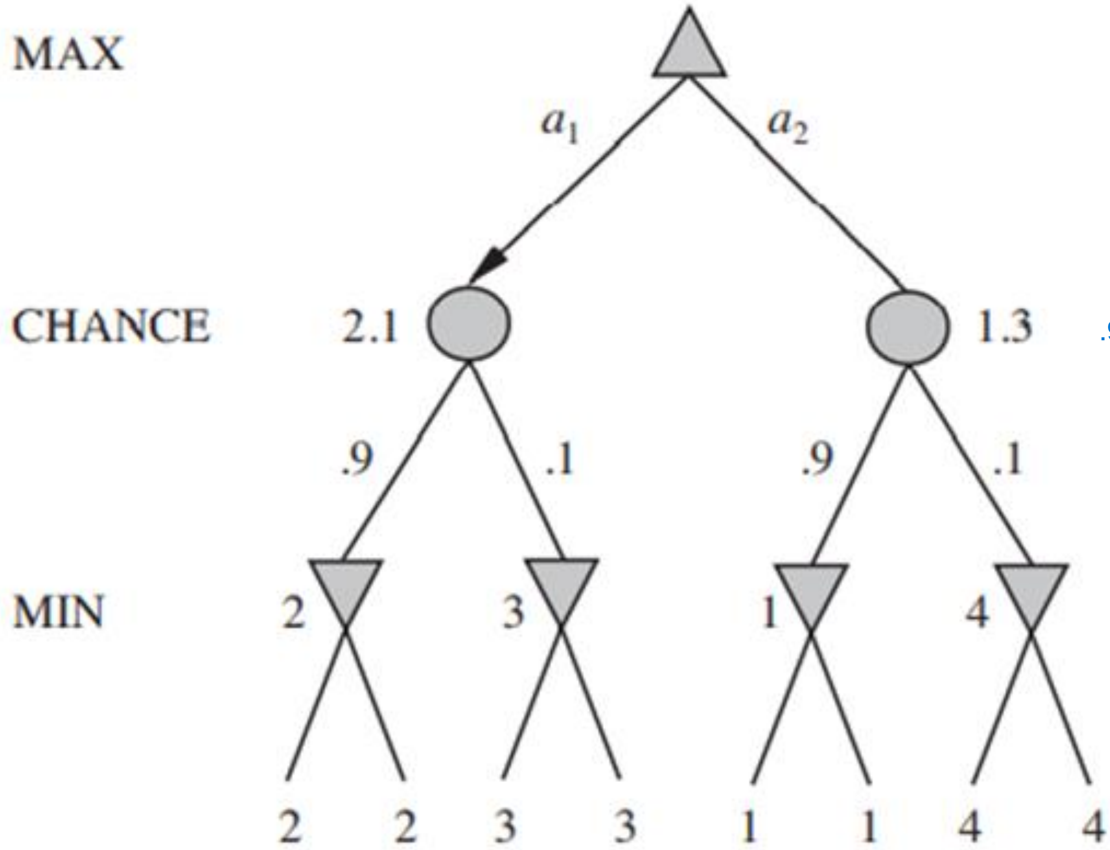
Holds the expected values that are computed as a sum of all outcomes weighted by their probability (of dice roll)



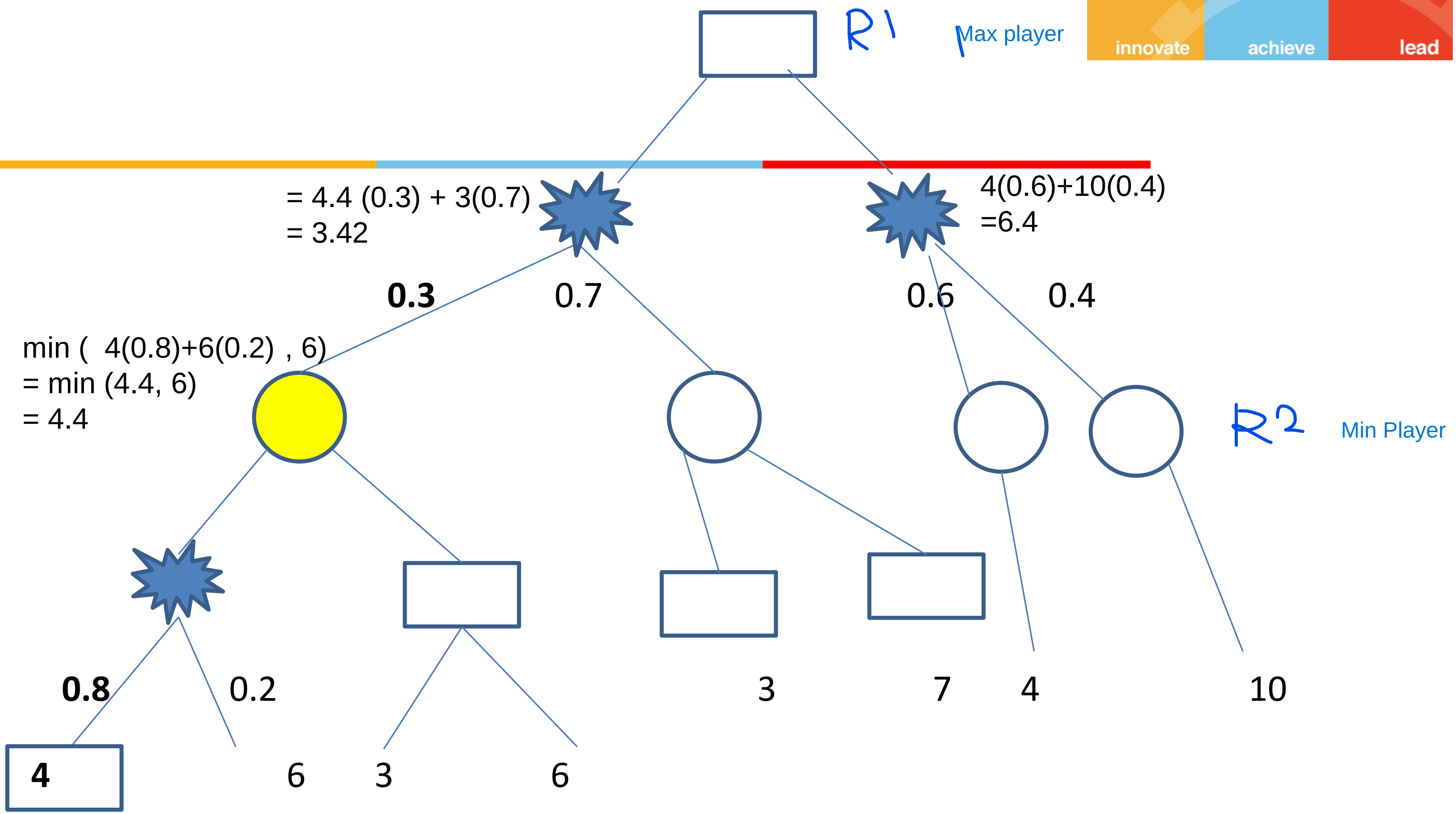
Expecti Mini Max Algorithm



Expecti Mini Max Algorithm

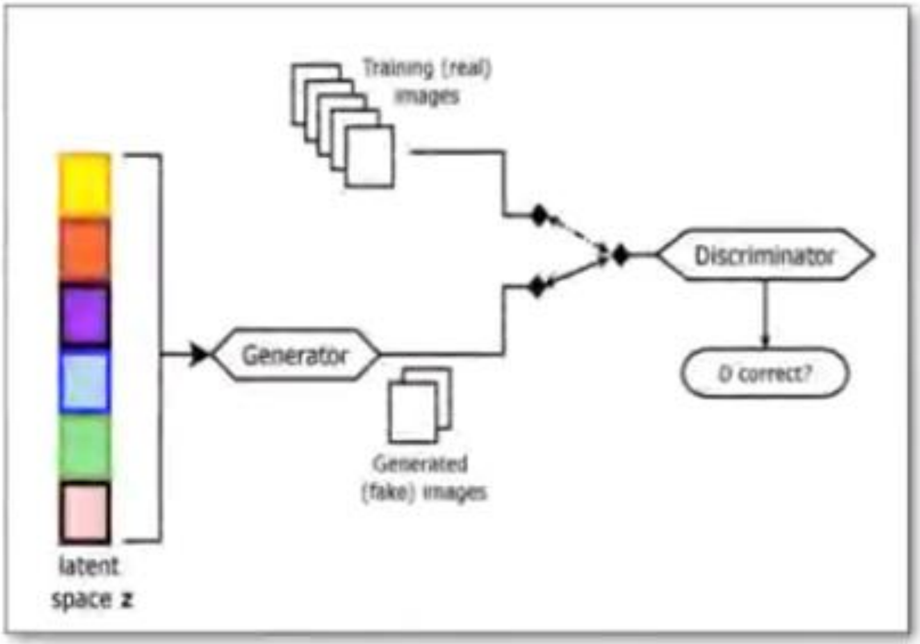


.9X1 + .1X4 = 1.3


R1
Max player
= 4.4 (0.3) + 3(0.7)
= 3.42
4(0.6)+10(0.4)
=6.4
0.3
0.7
0.6
0.4
min (4(0.8)+6(0.2) , 6)
= min (4.4, 6)
= 4.4
R2
Min Player
0.8
0.2
3
7
4
10
4
6
3
6

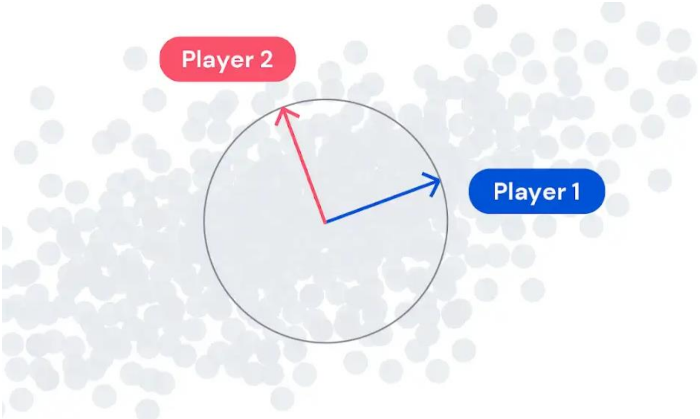
Game Playing
(Interesting Case Studies)

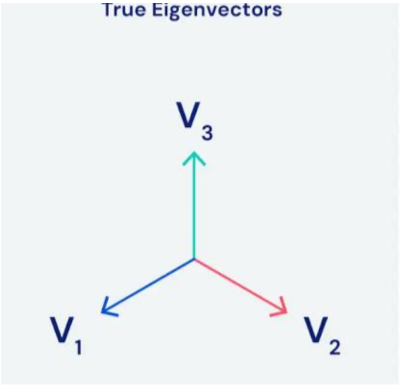
Games in Image Processing

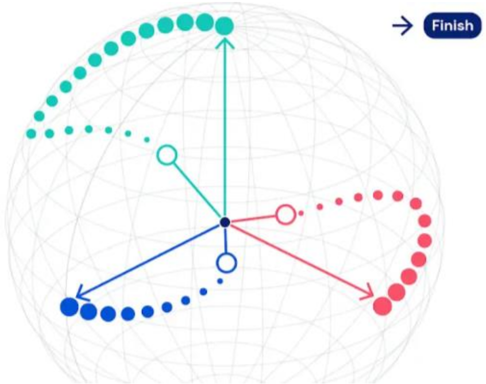


Source Credit:
[2019 - Analyzing and Improving the Image Quality of StyleGAN](#)
[Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, Timo Aila](#)
https://thispersondoesnotexist.com/

Games in Feature Engineering



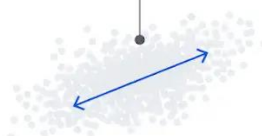


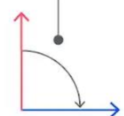


Source Credit:
https://deepmind.com/blog/article/EigenGame
2021 - EigenGame: PCA as a Nash Equilibrium , Ian Gemp, Brian McWilliams, Claire Vernade, Thore Graepel

Games in Feature Engineering

$$\mathtt{Utility}(v_i \mid v_{j<i}) = \boxed{\mathtt{Var}(v_i)} - \sum_{j<i} \boxed{\mathtt{Align}}(v_i, v_j)$$





Source Credit:
https://deepmind.com/blog/article/EigenGame
[2021 - EigenGame: PCA as a Nash Equilibrium , Ian Gemp, Brian McWilliams, Claire Vernade, Thore Graepel]()

Knowledge Representation Using Logics

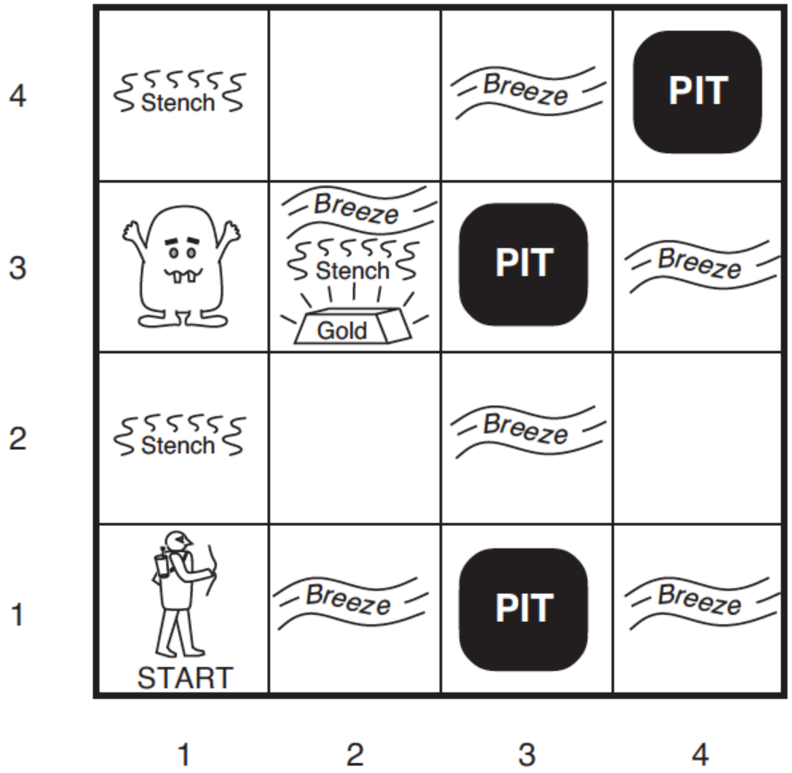
Learning Objective

At the end of this class , students Should be able to:

1. Represent a given knowledge base into logic formulation
2. Infer facts from KB using Resolution
3. Infer facts from KB using Forward Chaining
4. Infer facts from KB using Backward Chaining

Knowledge based Agent : Model & Represent

Concepts, logic Representation of a sample agent



Wumpus World Problem:

This example -Single agent deterministic Sequential Static partial observability task environment

PEAS:

Performance Measure:

+1000 for climbing out with gold,

-1000 for falling into a pit or being eaten by Wumpus,

-1 for each action taken and

-10 for using an arrow

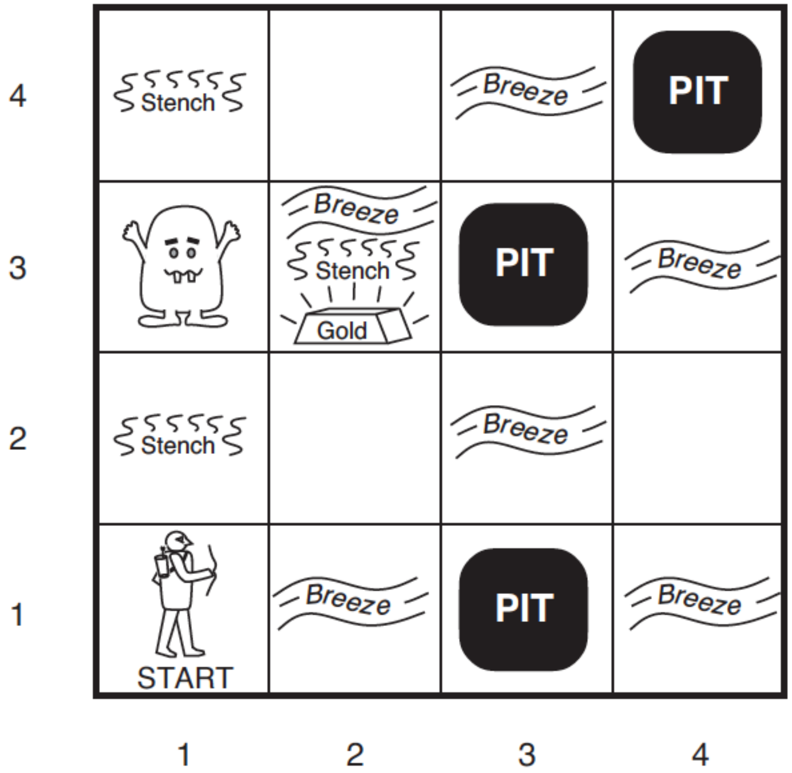
Environment: 4x4 grid of rooms. Always starts at [1, 1] facing right.

The location of Wumpus and Gold are random.

Agent dies if entered a pit or live Wumpus.

Knowledge based Agent : Model & Represent

Concepts, logic Representation of a sample agent



Wumpus World Problem:

PEAS:

Actuators –

Forward,

TurnLeft by 90,

TurnRight by 90,

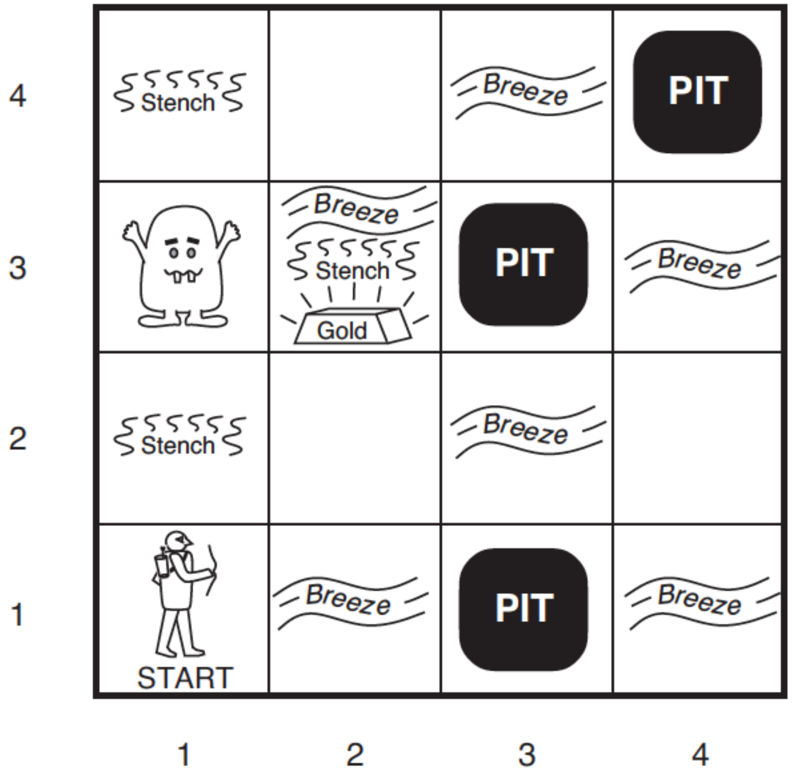
Grab – pick gold if present,

Shoot – fire an arrow, it either hits a wall or kills wumpus. Agent has only one arrow.

Climb – Used to climb out of cave, only from [1, 1]

Knowledge based Agent : Model & Represent

Concepts, logic Representation of a sample agent

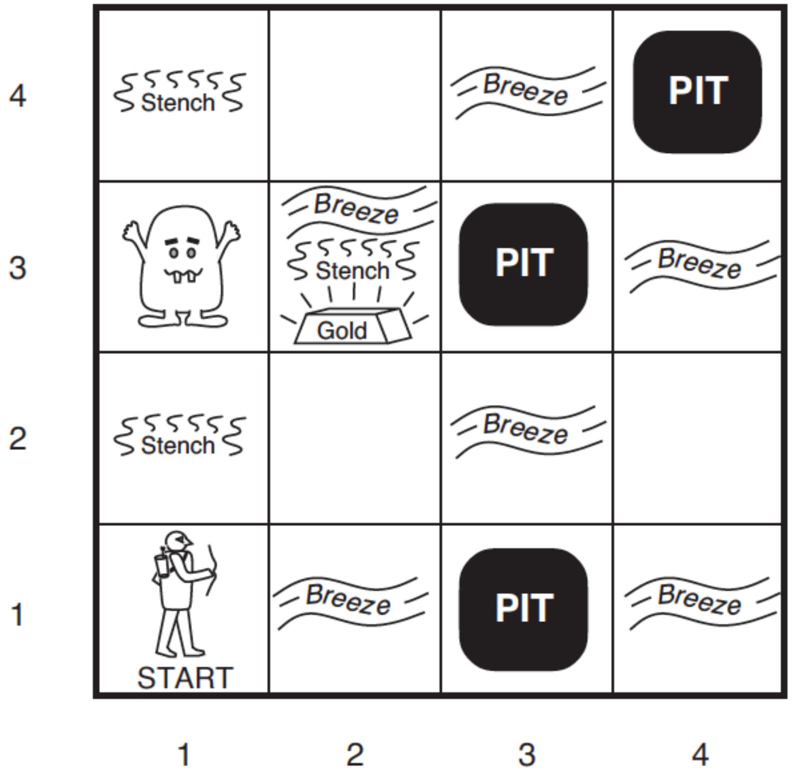


Why do we need Factored representation

- To reason about steps
- To learn new knowledge about the environment
- To adapt to changes to the existing knowledge
- Accept new tasks in the form of explicit goals
- To overcome partial observability of environment

Knowledge based Agent : Model & Represent

Concepts, logic Representation of a sample agent



Wumpus World Problem:

PEAS:

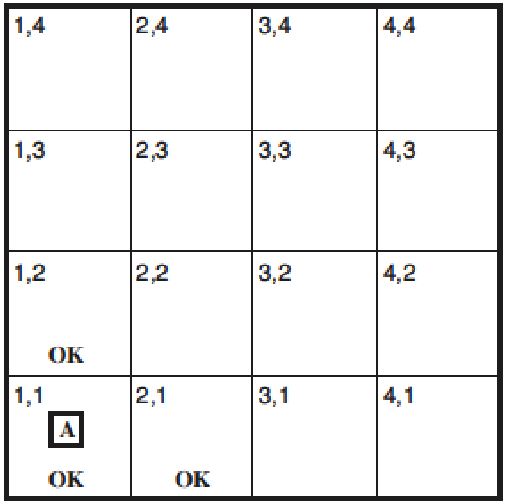
Sensors. The agent has five sensors

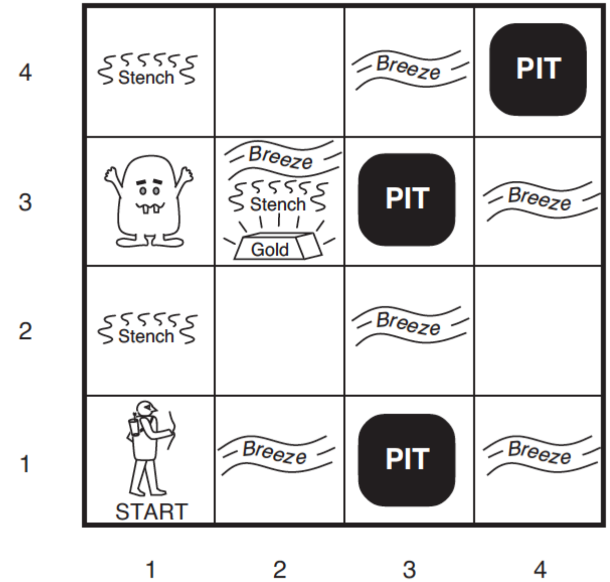
- **S**tench: In all adjacent (but not diagonal) squares of Wumpus
- **B**reeze: In all adjacent (but not diagonal) squares of a pit
- **G**litter: In the square where gold is
- **B**ump: If agent walks into a wall
- **S**cream: When Wumpus is killed, it can be perceived everywhere

Percept Format:
[Stench?, Breeze?, Glitter?, Bump?, Scream?]
E.g., [Stench, Breeze, None, None, None]

Percept 1: [None, None, None, None, None]

Action: Forward



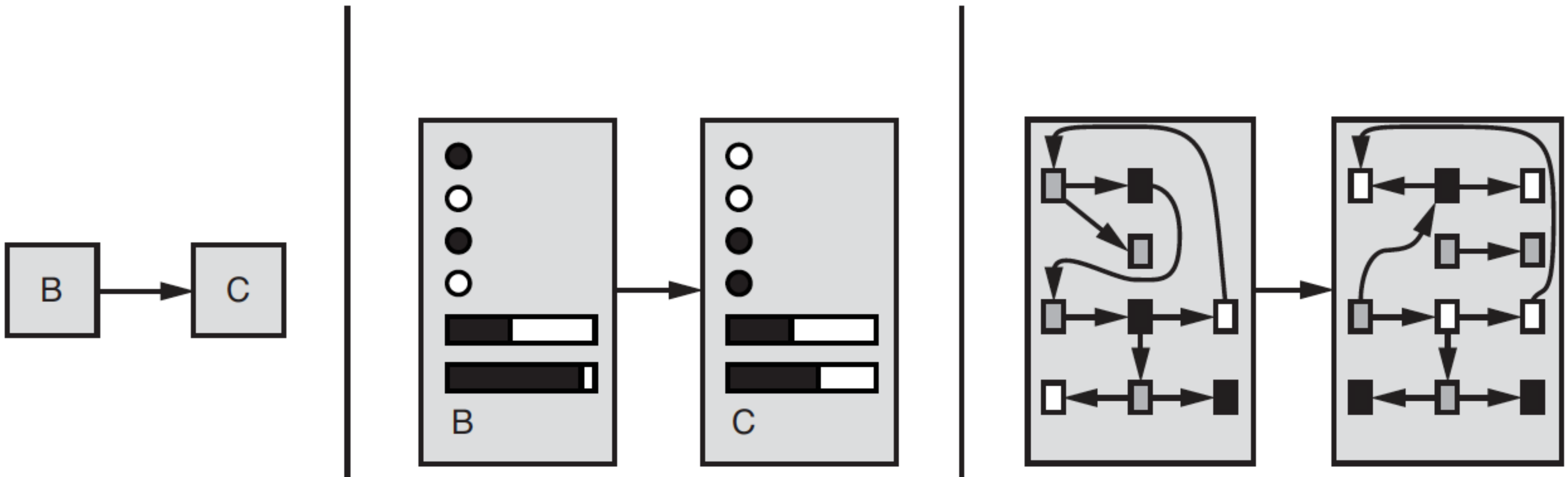


Percept Format:
[Stench?, Breeze?, Glitter?, Bump?, Scream?]

Representation

Agents based on Propositional logic ,TT-Entail for inference from truth table

Syntax
Semantics → Logic → Propositional Logic
Model → Predicate Logic



(a) Atomic (b) Factored (b) Structured

Search Strategies Propositional Logic First Order Logic

Propositional Logic

Agents based on Propositional logic, TT-Entail for inference from truth table

A simple representation language for building knowledge-based agents

Proposition Symbol – A symbol that stands for a proposition.

E.g., W1,3 – “Wumpus in [1,3]” is a proposition and $W_{1,3}$ is the symbol

Proposition can be true or false

Atomic : $W_{1,3}$

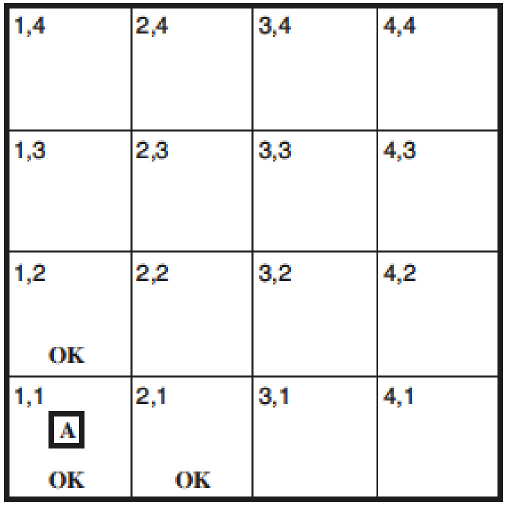
Conjuncts : $W_{1,3} \wedge P_{3,1}$

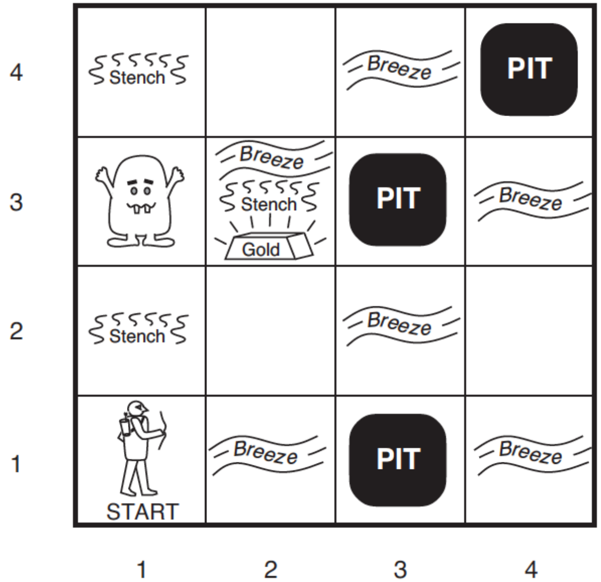
Disjuncts : $W_{1,3} \vee P_{3,1}$

Implications :

$(W_{1,3} \wedge P_{3,1}) \Rightarrow \neg W_{2,2}$

Biconditional : $W_{1,3} \Leftrightarrow \neg W_{2,2}$





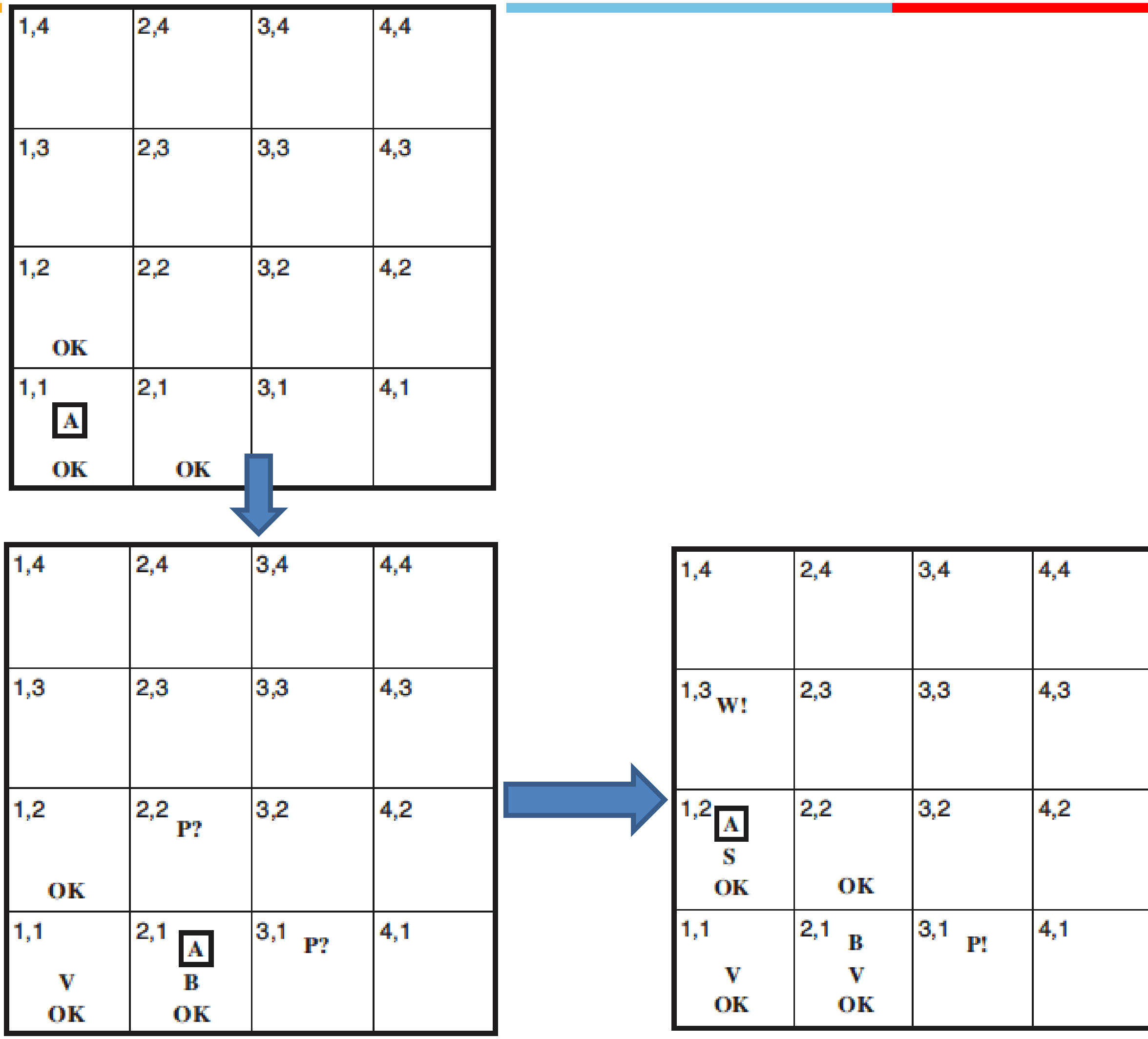
Tie break in search:

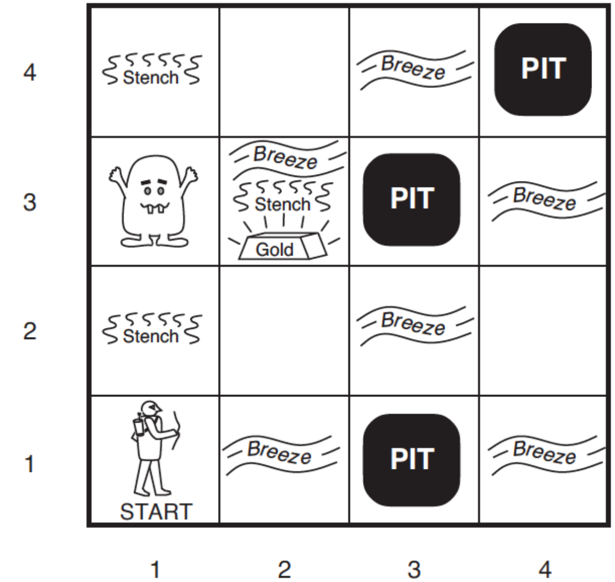
¬ , ∧ , ∨, ⇒ , ⇔

$(\neg A) \wedge B$ has precedence over $\neg (A \wedge B)$

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

Percept 3: [Stench, None, None, None, None]
Action: Move to [2, 2]
Remembers (2,2) as possible PIT and no Stench.





Representation by Propositional Logic

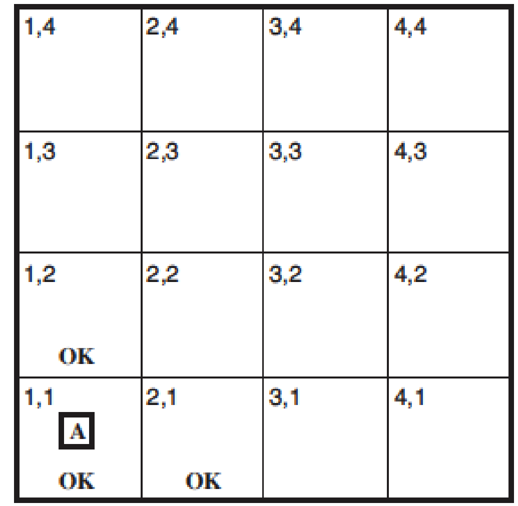
For each [x, y] location

$P_{x,y}$ is true if there is a pit in [x, y]

$W_{x,y}$ is true if there is a wumpus in [x, y]

$B_{x,y}$ is true if agent perceives a breeze in [x, y]

$S_{x,y}$ is true if agent perceives a stench in [x, y]



----------------------- R is the sentence in KB

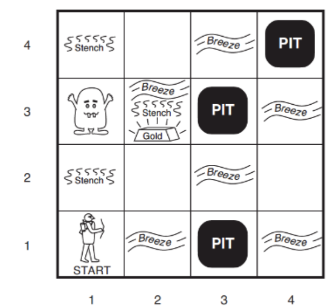
$R_1 : \neg P_{1,1}$

$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

$R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

$R_4 : \neg B_{1,1}$

$R_5 : B_{2,1}$



$Query : \neg P_{1,2}$ entailed by our KB?

TT – Entails Inference – Example

$\neg P_{1,2}$ entailed by our KB?

Way – 1 :

1. Get sufficient information $B_{1,1}$, $B_{2,1}$, $P_{1,1}$, $P_{1,2}$, $P_{2,1}$, $P_{2,2}$, $P_{3,1}$
2. Enumerate all models with combination of truth values to propositional symbols
3. In all the models, find those models where KB is true, i.e., every sentence R_1, R_2, R_3, R_4, R_5 are true
4. In those models where KB is true, find if query sentence $\neg P_{1,2}$ is true
5. If query sentence $\neg P_{1,2}$ is true in all models where KB is true, then it entails, otherwise it won't

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	*false*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*false*	*false*
false	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*false*	*true*	*false*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	*true*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*false*	*true*	*true*	*false*
false	*true*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	<u>*true*</u>
false	*true*	*false*	*false*	*false*	*true*	*false*	*true*	*true*	*true*	*true*	*true*	<u>*true*</u>
false	*true*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*	<u>*true*</u>
false	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*true*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	*true*	*true*	*true*	*true*	*true*	*true*	*false*	*true*	*true*	*false*	*true*	*false*

TT – Entails Inference – Example

Agents based on Propositional logic, TT-Entail for inference from truth table

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	*false*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*false*	*false*
false	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*false*	*true*	*false*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	*true*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*false*	*true*	*true*	*false*
false	*true*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*false*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*true*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	*true*	*true*	*true*	*true*	*true*	*true*	*false*	*true*	*true*	*false*	*true*	*false*

Inference : Properties

1. Entailment : $\alpha \models \beta$
2. Equivalence : $\alpha \equiv \beta$ if and only if $\alpha \models \beta$ and $\beta \models \alpha$
3. Validity
4. Satisfiability

Inference : Example – Theorem Proving (Self Study)

Propositional theorem proving - Proof by resolution

Logical Equivalence rules can be used as inference rules

$$
\begin{aligned}
(\alpha \wedge \beta) &\equiv (\beta \wedge \alpha) && \text{commutativity of } \wedge \\
(\alpha \vee \beta) &\equiv (\beta \vee \alpha) && \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) &\equiv (\alpha \wedge (\beta \wedge \gamma)) && \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) &\equiv (\alpha \vee (\beta \vee \gamma)) && \text{associativity of } \vee \\
\neg(\neg\alpha) &\equiv \alpha && \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) && \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \vee \beta) && \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) && \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) &\equiv (\neg\alpha \vee \neg\beta) && \text{De Morgan} \\
\neg(\alpha \vee \beta) &\equiv (\neg\alpha \wedge \neg\beta) && \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) && \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) &\equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) && \text{distributivity of } \vee \text{ over } \wedge
\end{aligned}
$$

Inference : Example – Theorem Proving

$$
\begin{aligned}
(\alpha \wedge \beta) &\equiv (\beta \wedge \alpha) && \text{commutativity of } \wedge \\
(\alpha \vee \beta) &\equiv (\beta \vee \alpha) && \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) &\equiv (\alpha \wedge (\beta \wedge \gamma)) && \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) &\equiv (\alpha \vee (\beta \vee \gamma)) && \text{associativity of } \vee \\
\neg(\neg\alpha) &\equiv \alpha && \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) && \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \vee \beta) && \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) && \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) &\equiv (\neg\alpha \vee \neg\beta) && \text{De Morgan} \\
\neg(\alpha \vee \beta) &\equiv (\neg\alpha \wedge \neg\beta) && \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) && \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) &\equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) && \text{distributivity of } \vee \text{ over } \wedge
\end{aligned}
$$

1. **Modes Ponens**
2. AND Elimination

α : I walk in rain without the umbrella

β : I get wet

α → β   **α**

β

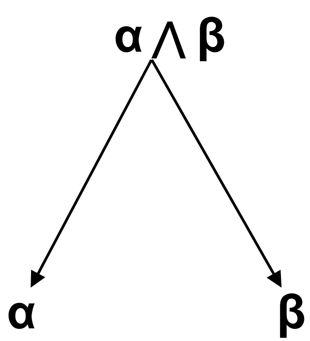
Inference : Example – Theorem Proving

1. Modes Ponens
2. **AND Elimination**

$$
\begin{array}{rcll}
(\alpha \wedge \beta) & \equiv & (\beta \wedge \alpha) & \text{commutativity of } \wedge \\
(\alpha \vee \beta) & \equiv & (\beta \vee \alpha) & \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) & \equiv & (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) & \equiv & (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\
\neg(\neg\alpha) & \equiv & \alpha & \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\beta \Rightarrow \neg\alpha) & \text{contraposition} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\alpha \vee \beta) & \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) & \equiv & ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) & \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) & \equiv & (\neg\alpha \vee \neg\beta) & \text{De Morgan} \\
\neg(\alpha \vee \beta) & \equiv & (\neg\alpha \wedge \neg\beta) & \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) & \equiv & ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) & \equiv & ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge
\end{array}
$$

α : I walk in rain without the umbrella

β : I get wet



Inference : Example – Theorem Proving

$R_1 : \neg P_{1,1}$

$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$

$R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1})$

$R_4 : \neg B_{1,1}$

$R_5 : B_{2,1}$

$$
\begin{aligned}
(\alpha \land \beta) &\equiv (\beta \land \alpha) && \text{commutativity of } \land \\
(\alpha \lor \beta) &\equiv (\beta \lor \alpha) && \text{commutativity of } \lor \\
((\alpha \land \beta) \land \gamma) &\equiv (\alpha \land (\beta \land \gamma)) && \text{associativity of } \land \\
((\alpha \lor \beta) \lor \gamma) &\equiv (\alpha \lor (\beta \lor \gamma)) && \text{associativity of } \lor \\
\neg(\neg\alpha) &\equiv \alpha && \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) && \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \lor \beta) && \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)) && \text{biconditional elimination} \\
\neg(\alpha \land \beta) &\equiv (\neg\alpha \lor \neg\beta) && \text{De Morgan} \\
\neg(\alpha \lor \beta) &\equiv (\neg\alpha \land \neg\beta) && \text{De Morgan} \\
(\alpha \land (\beta \lor \gamma)) &\equiv ((\alpha \land \beta) \lor (\alpha \land \gamma)) && \text{distributivity of } \land \text{ over } \lor \\
(\alpha \lor (\beta \land \gamma)) &\equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma)) && \text{distributivity of } \lor \text{ over } \land
\end{aligned}
$$

Query: $\neg P_{1,2}$. Can we prove if this sentence be entailed from KB using inference rules?---------------------

$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$

$R_6 : (B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$ Biconditional Elimination

$R_7 : ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$ And Elimination

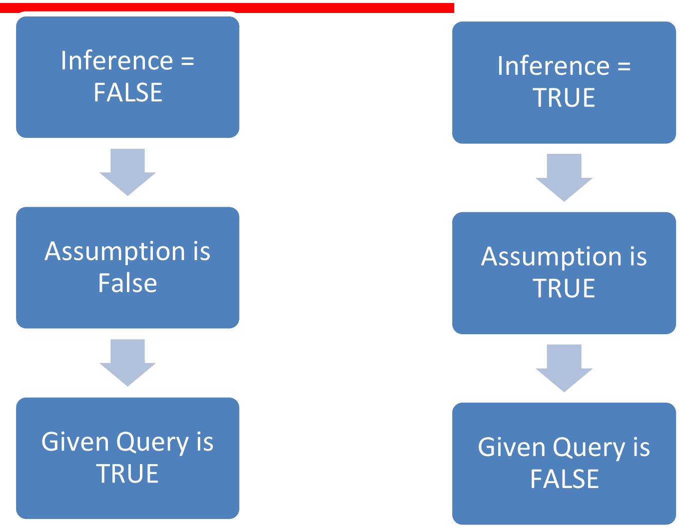
$R_8 : (\neg B_{1,1} \Rightarrow \neg (P_{1,2} \lor P_{2,1}))$ Contraposition

$R_9 : \neg (P_{1,2} \lor P_{2,1})$ Modus Ponens

$R_{10} : \neg P_{1,2} \land \neg P_{2,1}$ Demorgans

R11: $\neg P_{1,2}$ And Elimination

Proof by Contradiction



Required Reading: AIMA - Chapter # 4.1, #4.2, #5.1, #9

Next Session Plan:

- (Prerequisite Reading : Refresh the basics of probability , Bayes Theorem , Conditional Probability, Product Rule, Conditional Independence, Chain Rule)

- Inferences using Logic (Forward / Backward Chaining / DPLL algorithm)
- Bayesian Network
- Representation
- Inferences (Exact and approximate-only Direct sampling)

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

BITS Pilani
Pilani Campus

Artificial & Computational Intelligence

AIML CLZG557

M4 : Knowledge Representation using Logics

Raja vadhana P
Assistant Professor,
BITS - CSIS

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

M5 Probabilistic Representation and Reasoning

M6 Reasoning over time

M7 Ethics in AI

Knowledge Representation Using Logics

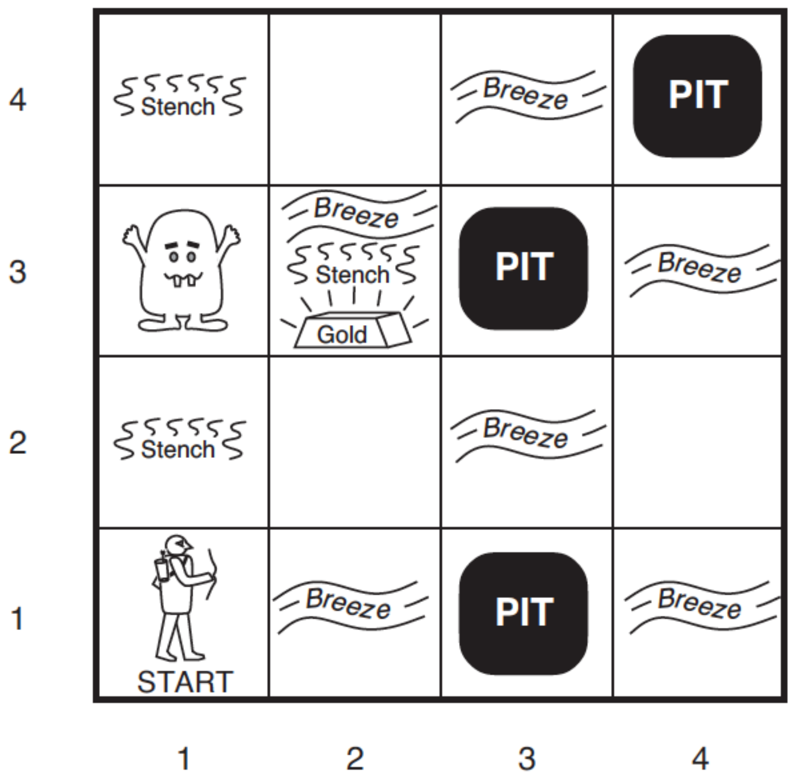
Learning Objective

At the end of this class , students Should be able to:

1. Represent a given knowledge base into logic formulation
2. Infer facts from KB using Resolution
3. Infer facts from KB using Forward Chaining
4. Infer facts from KB using Backward Chaining

Knowledge based Agent : Model & Represent

Concepts, logic Representation of a sample agent



Wumpus World Problem:

PEAS:

Performance Measure:

+1000 for climbing out with gold,

-1000 for falling into a pit or being eaten by Wumpus,

-1 for each action taken and

-10 for using an arrow

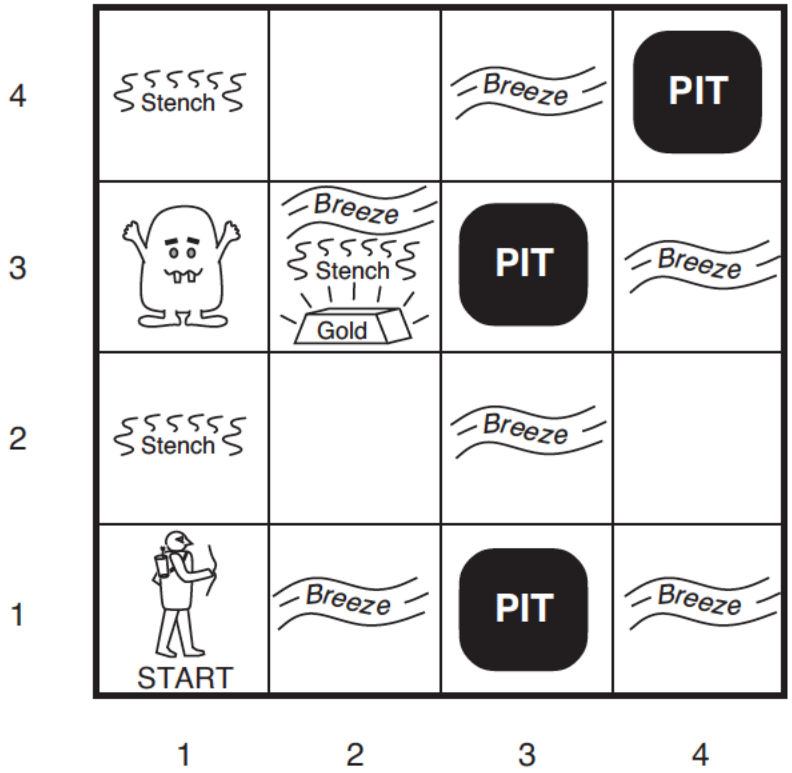
Environment: 4x4 grid of rooms. Always starts at [1, 1] facing right.

The location of Wumpus and Gold are random.

Agent dies if entered a pit or live Wumpus.

Knowledge based Agent : Model & Represent

Concepts, logic Representation of a sample agent



Wumpus World Problem:

PEAS:

Actuators –

Forward,

TurnLeft by 90,

TurnRight by 90,

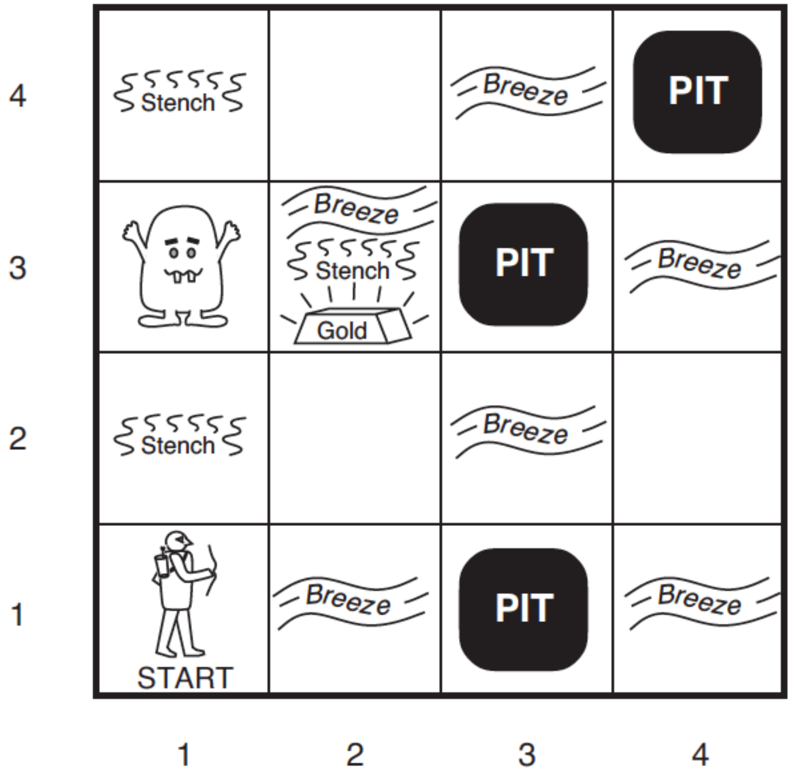
Grab – pick gold if present,

Shoot – fire an arrow, it either hits a wall or kills wumpus. Agent has only one arrow.

Climb – Used to climb out of cave, only from [1, 1]

Knowledge based Agent : Model & Represent

Concepts, logic Representation of a sample agent

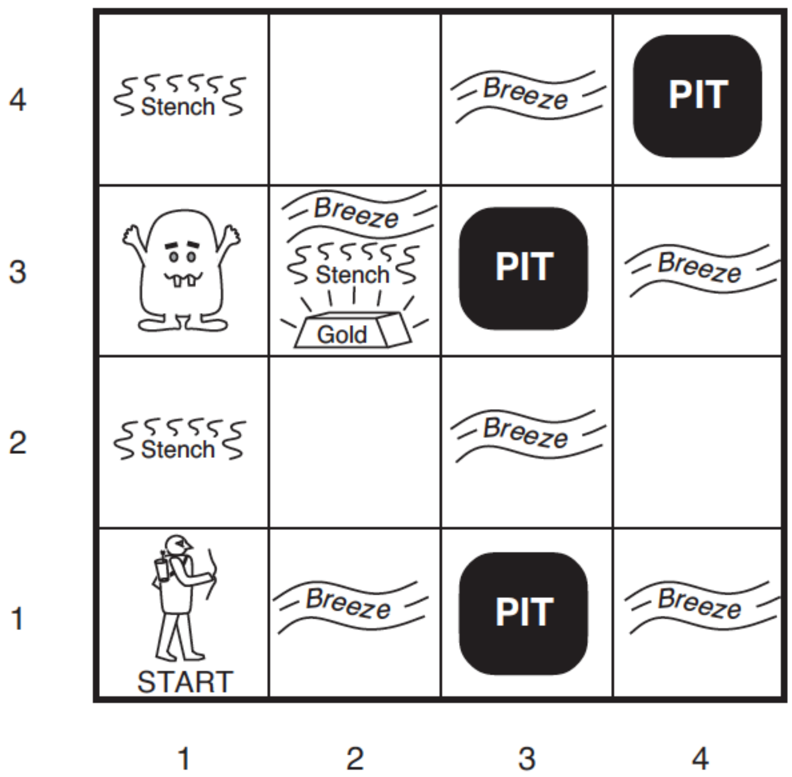


Why do we need Factored representation

- To reason about steps
- To learn new knowledge about the environment
- To adapt to changes to the existing knowledge
- Accept new tasks in the form of explicit goals
- To overcome partial observability of environment

Knowledge based Agent : Model & Represent

Concepts, logic Representation of a sample agent



Wumpus World Problem:

PEAS:

Sensors. The agent has five sensors

- **S**tench: In all adjacent (but not diagonal) squares of Wumpus
- **B**reeze: In all adjacent (but not diagonal) squares of a pit
- **G**litter: In the square where gold is
- **B**ump: If agent walks into a wall
- **S**cream: When Wumpus is killed, it can be perceived everywhere

Percept Format:
[Stench?, Breeze?, Glitter?, Bump?, Scream?]
E.g., [Stench, Breeze, None, None, None]

Propositional Logic

Agents based on Propositional logic, TT-Entail for inference from truth table

A simple representation language for building knowledge-based agents

Proposition Symbol – A symbol that stands for a proposition.

E.g., W1,3 – "Wumpus in [1,3]" is a proposition and $W_{1,3}$ is the symbol

Proposition can be true or false

Atomic : $W_{1,3}$

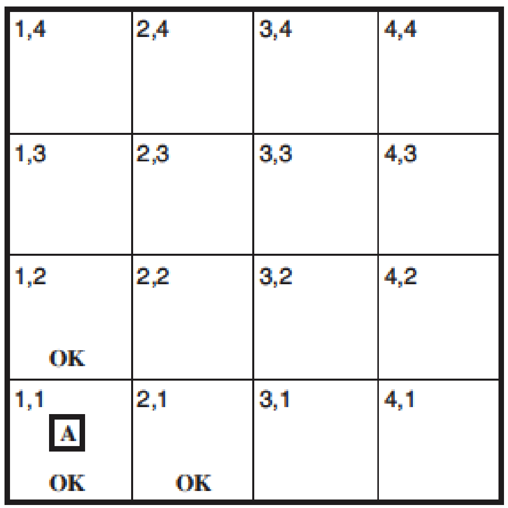
Conjuncts : $W_{1,3} \wedge P_{3,1}$

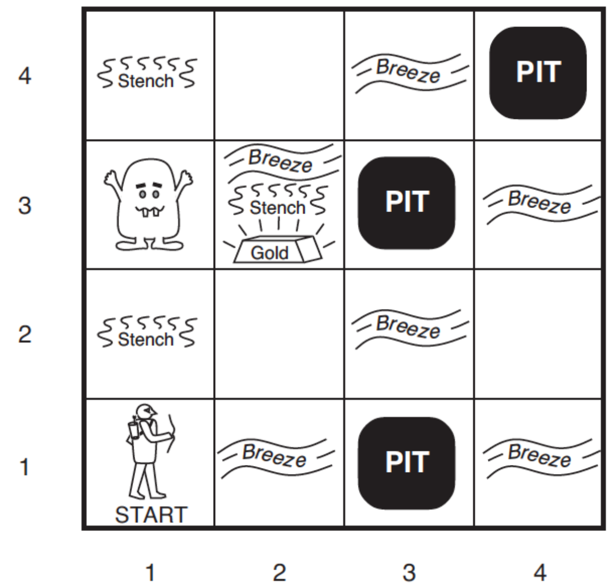
Disjuncts : $W_{1,3} \vee P_{3,1}$

Implications :

$(W_{1,3} \wedge P_{3,1}) \Rightarrow \neg W_{2,2}$

Biconditional : $W_{1,3} \Leftrightarrow \neg W_{2,2}$

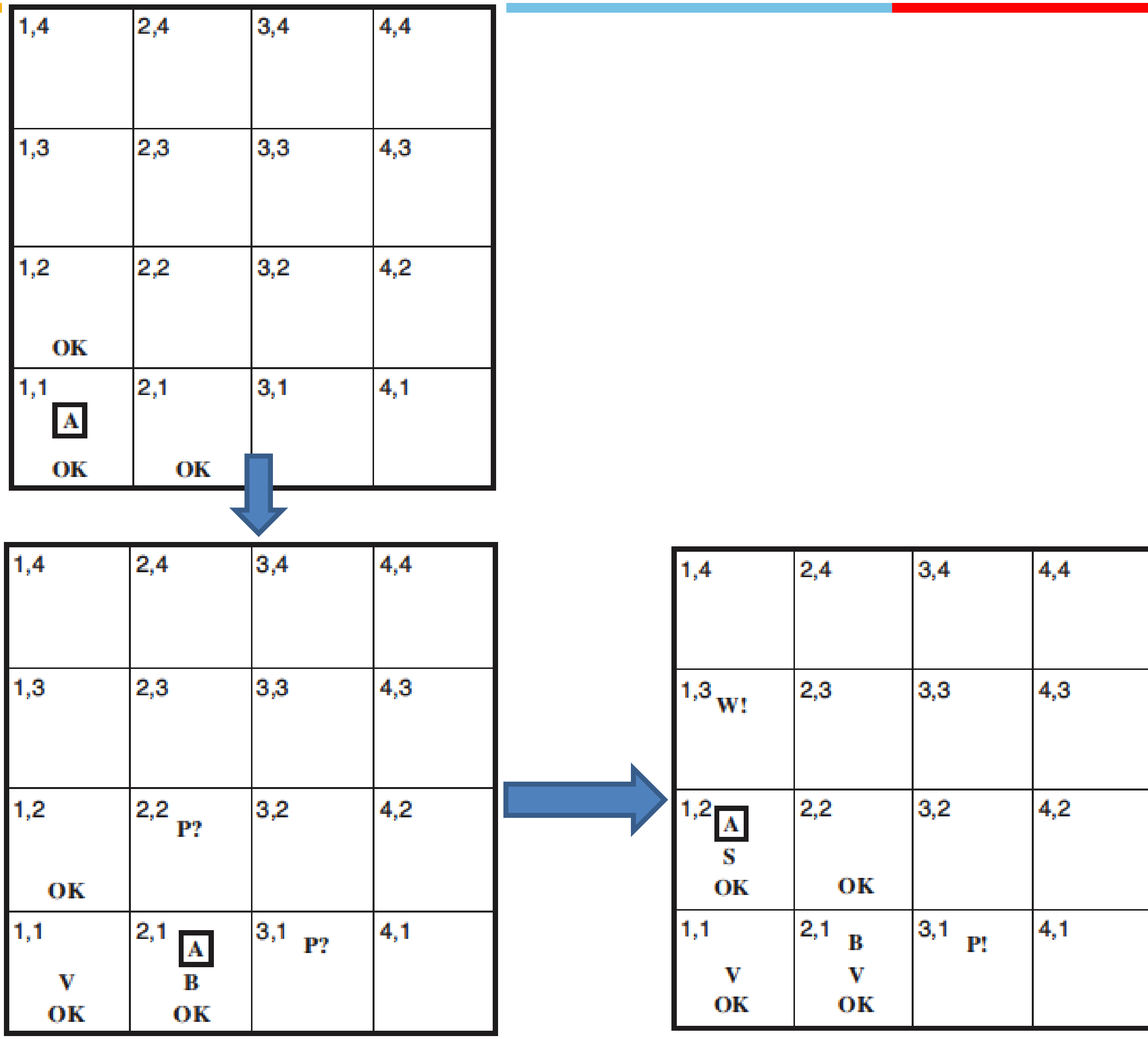


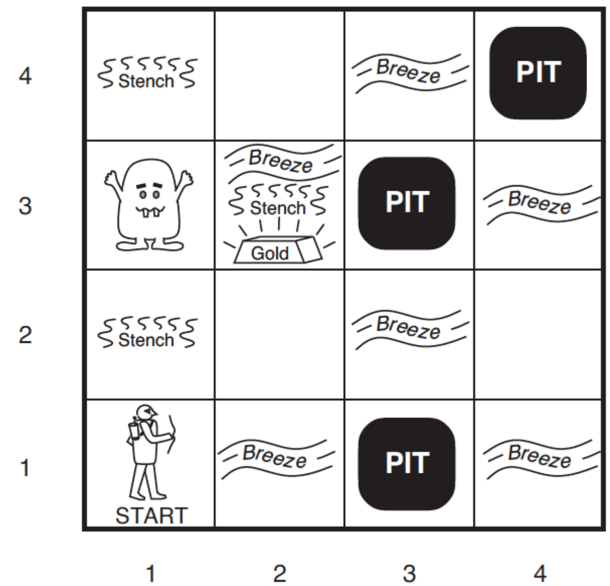


Percept 3: [Stench, None, None, None, None]

Action: Move to [2, 2]

Remembers (2,2) as possible PIT and no Stench.

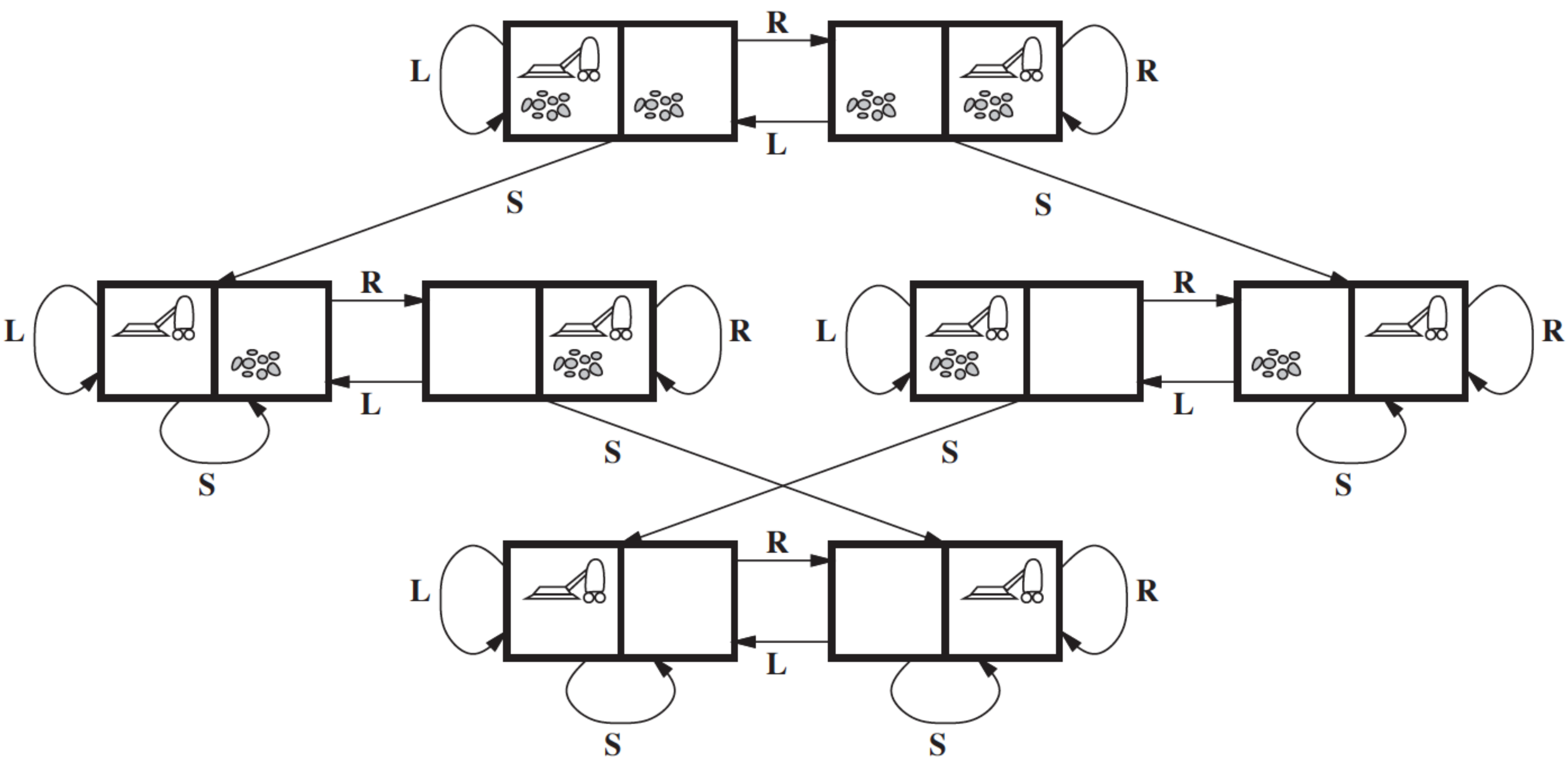




Sample Result – Toy Problem

innovate achieve lead

Agents based on Propositional logic, TT-Entail for inference from truth table, Propositional theorem proving



Representation by Propositional Logic

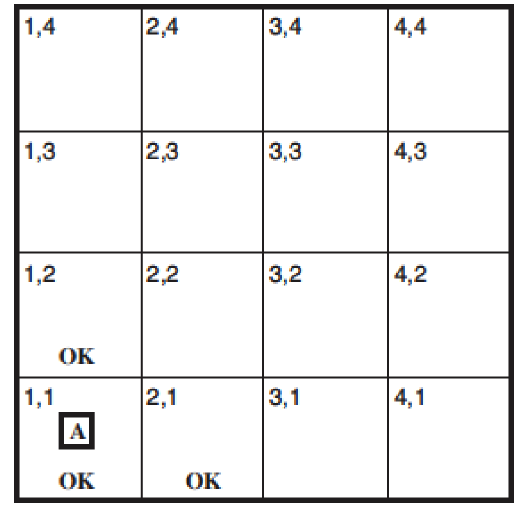
For each [x, y] location

$P_{x,y}$ is true if there is a pit in [x, y]

$W_{x,y}$ is true if there is a wumpus in [x, y]

$B_{x,y}$ is true if agent perceives a breeze in [x, y]

$S_{x,y}$ is true if agent perceives a stench in [x, y]



----------------------- R is the sentence in KB

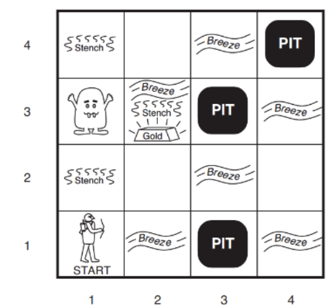
$R_1 : \neg P_{1,1}$

$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

$R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

$R_4 : \neg B_{1,1}$

$R_5 : B_{2,1}$



$Query$: $\neg P_{1,2}$ entailed by our KB?

TT – Entails Inference – Example

¬ $P_{1,2}$ entailed by our KB?

Way – 1 :

1. Get sufficient information $B_{1,1}$, $B_{2,1}$, $P_{1,1}$, $P_{1,2}$, $P_{2,1}$, $P_{2,2}$, $P_{3,1}$
2. Enumerate all models with combination of truth values to propositional symbols
3. In all the models, find those models where KB is true, i.e., every sentence R_1, R_2, R_3, R_4, R_5 are true
4. In those models where KB is true, find if query sentence ¬ $P_{1,2}$ is true
5. If query sentence ¬ $P_{1,2}$ is true in all models where KB is true, then it entails, otherwise it won't

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	*false*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*false*	*false*
false	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*false*	*true*	*false*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	*true*	*false*	*false*	*false*	*false*	*false*	*true*	*true*	*false*	*true*	*true*	*false*
false	*true*	*false*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*false*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*true*	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*false*	*false*	*true*	*true*	*false*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	*true*	*true*	*true*	*true*	*true*	*true*	*false*	*true*	*true*	*false*	*true*	*false*

Inference : Example – Theorem Proving

1. **Modes Ponens**
2. AND Elimination

α : I walk in rain without the umbrella

β : I get wet

α → β **α**

β

$$
\begin{array}{rcll}
(\alpha \wedge \beta) & \equiv & (\beta \wedge \alpha) & \text{commutativity of } \wedge \\
(\alpha \vee \beta) & \equiv & (\beta \vee \alpha) & \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) & \equiv & (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) & \equiv & (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\
\neg(\neg\alpha) & \equiv & \alpha & \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\beta \Rightarrow \neg\alpha) & \text{contraposition} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\alpha \vee \beta) & \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) & \equiv & ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) & \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) & \equiv & (\neg\alpha \vee \neg\beta) & \text{De Morgan} \\
\neg(\alpha \vee \beta) & \equiv & (\neg\alpha \wedge \neg\beta) & \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) & \equiv & ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) & \equiv & ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge
\end{array}
$$

Inference : Example – Theorem Proving

1. Modes Ponens
2. **AND Elimination**

$$
\begin{array}{rcll}
(\alpha \wedge \beta) & \equiv & (\beta \wedge \alpha) & \text{commutativity of } \wedge \\
(\alpha \vee \beta) & \equiv & (\beta \vee \alpha) & \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) & \equiv & (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) & \equiv & (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\
\neg(\neg\alpha) & \equiv & \alpha & \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\beta \Rightarrow \neg\alpha) & \text{contraposition} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\alpha \vee \beta) & \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) & \equiv & ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) & \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) & \equiv & (\neg\alpha \vee \neg\beta) & \text{De Morgan} \\
\neg(\alpha \vee \beta) & \equiv & (\neg\alpha \wedge \neg\beta) & \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) & \equiv & ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) & \equiv & ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge
\end{array}
$$

α : I walk in rain without the umbrella

β : I get wet

$$
\begin{tikzcd}
 & \boldsymbol{\alpha \wedge \beta} \arrow[dl] \arrow[dr] & \\
\boldsymbol{\alpha} & & \boldsymbol{\beta}
\end{tikzcd}
$$

Inference : Example – Theorem Proving

$R_1 : \neg P_{1,1}$

$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$

$R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1})$

$R_4 : \neg B_{1,1}$

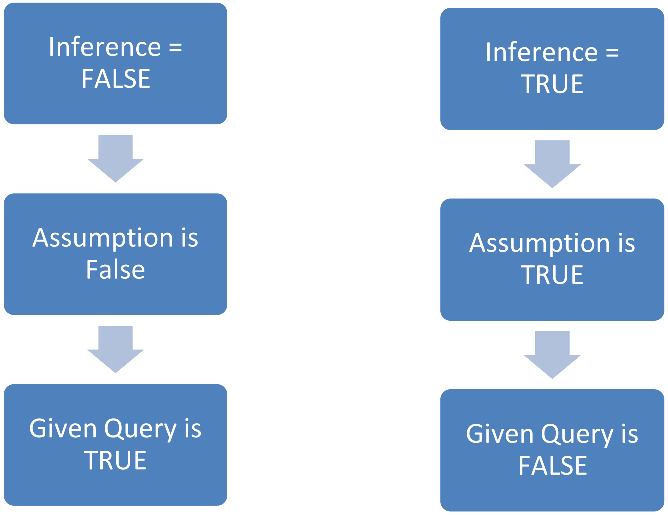
$R_5 : B_{2,1}$

$(\alpha \land \beta) \equiv (\beta \land \alpha)$	commutativity of $\land$
$(\alpha \lor \beta) \equiv (\beta \lor \alpha)$	commutativity of $\lor$
$((\alpha \land \beta) \land \gamma) \equiv (\alpha \land (\beta \land \gamma))$	associativity of $\land$
$((\alpha \lor \beta) \lor \gamma) \equiv (\alpha \lor (\beta \lor \gamma))$	associativity of $\lor$
$\neg(\neg\alpha) \equiv \alpha$	double-negation elimination
$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$	contraposition
$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \lor \beta)$	implication elimination
$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha))$	biconditional elimination
$\neg(\alpha \land \beta) \equiv (\neg\alpha \lor \neg\beta)$	De Morgan
$\neg(\alpha \lor \beta) \equiv (\neg\alpha \land \neg\beta)$	De Morgan
$(\alpha \land (\beta \lor \gamma)) \equiv ((\alpha \land \beta) \lor (\alpha \land \gamma))$	distributivity of $\land$ over $\lor$
$(\alpha \lor (\beta \land \gamma)) \equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma))$	distributivity of $\lor$ over $\land$

Query: $\neg P_{1,2}$. Can we prove if this sentence be entailed from KB using inference rules?---------------------

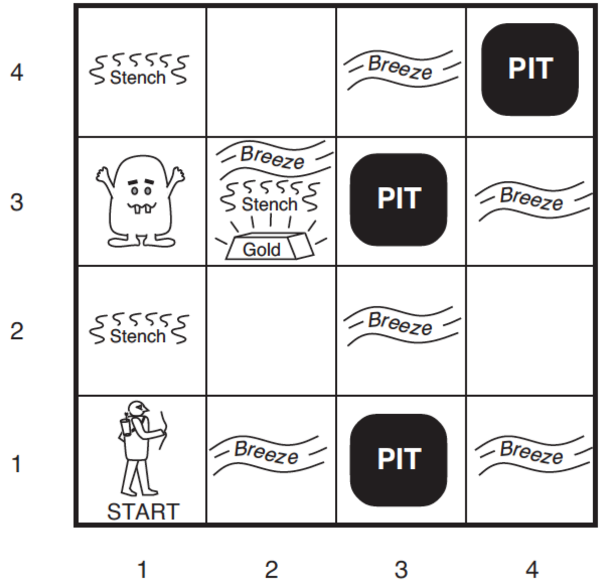
$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$	
$R_6 : (B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$	Biconditional Elimination
$R_7 : ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$	And Elimination
$R_8 : (\neg B_{1,1} \Rightarrow \neg (P_{1,2} \lor P_{2,1}))$	Contraposition
$R_9 : \neg (P_{1,2} \lor P_{2,1})$	Modus Ponens
$R_{10} : \neg P_{1,2} \land \neg P_{2,1}$	Demorgans
R11: $\neg \mathbf{P_{1,2}}$	And Elimination

Proof by Contradiction



Horn Clause

1. Definite Clause : A horn clause with exactly one positive literal
2. Fact : Definite clause with no negative literal / assertion
3. Rule
4. Inference by Chaining



PL-Resolution : CNF conversion

Wumpus world Book example

$R_1 : \neg P_{1,1}$

$\mathbf{R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})}$

$\mathbf{R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})}$

$R_4 : \neg B_{1,1}$

$R_5 : B_{2,1}$

Query: $\neg P_{1,2}$

Conjunctive Normal Form :

(A V ~B) ^ (A V B V ~C) ^ ~A

Unit Resolution : ~A

Query : Is 'C' true?

PL-Resolution

$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$ commutativity of $\wedge$
$(\alpha \vee \beta) \equiv (\beta \vee \alpha)$ commutativity of $\vee$
$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$ associativity of $\wedge$
$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$ associativity of $\vee$
$\neg(\neg\alpha) \equiv \alpha$ double-negation elimination
$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$ contraposition
$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$ implication elimination
$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$ biconditional elimination
$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$ De Morgan
$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$ De Morgan
$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$ distributivity of $\wedge$ over $\vee$
$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$ distributivity of $\vee$ over $\wedge$

$R_1 : \neg P_{1,1}$
$\cancel{R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})}$
$\cancel{\mathbf{R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})}}$
$R_4 : \neg B_{1,1}$
$R_5 : B_{2,1}$
Query: $\neg P_{1,2}$

$R_6 : \neg B_{1,1} \vee P_{1,2} \vee P_{2,1}$
$R_7 : \neg P_{1,2} \vee B_{1,1}$
$R_8 : \neg P_{2,1} \vee B_{1,1}$
$R_9 : \neg B_{2,1} \vee P_{1,1} \vee P_{2,2} \vee P_{3,1}$
$R_{10} : \neg P_{1,1} \vee B_{2,1}$
$R_{11} : \neg P_{2,2} \vee B_{2,1}$
$R_{12} : \neg P_{3,1} \vee B_{2,1}$

Eliminate		$\mathbf{R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})}$	$\mathbf{R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})}$
←→	Biconditional Elimination	$(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$	$(B_{2,1} \Rightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})) \wedge ((P_{1,1} \vee P_{2,2} \vee P_{3,1}) \Rightarrow B_{2,1})$
→	Implication Elimination	$\neg B_{1,1} \vee (P_{1,2} \vee P_{2,1})$ $\mathbf{\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1}}$	$\neg B_{2,1} \vee (P_{1,1} \vee P_{2,2} \vee P_{3,1})$ $\mathbf{\neg(P_{1,1} \vee P_{2,2} \vee P_{3,1}) \vee B_{2,1}}$
Clause level ¬	De Morgan	$\mathbf{(\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1}}$	$\mathbf{(\neg P_{1,1} \wedge \neg P_{2,2} \wedge \neg P_{3,1}) \vee B_{2,1}}$
CNF Form	Distributivity of ∨ over ∧	$\mathbf{(\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})}$	$\mathbf{(\neg P_{1,1} \vee B_{2,1}) \wedge (\neg P_{2,2} \vee B_{2,1}) \wedge (\neg P_{3,1} \vee B_{2,1})}$

PL-Resolution

Unit Resolution: Query: $\neg P_{1,2}$

To find: Is there a pit in location (1,2) using the CNF obtained in previous slide

Inference = FALSE

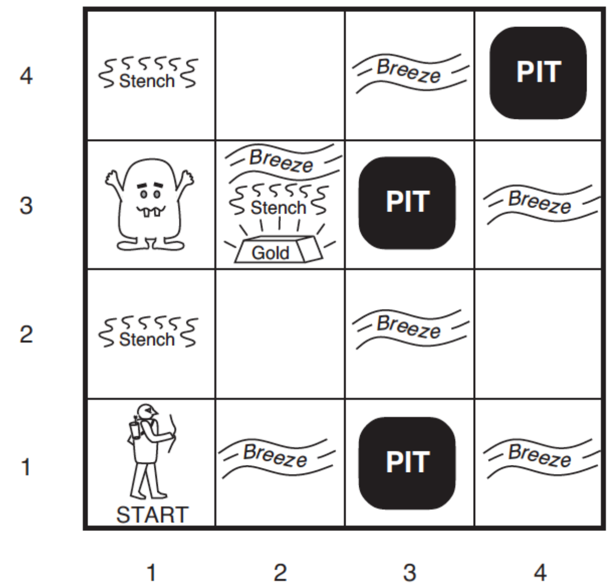
Assumption is False

Given Query is TRUE

$\neg B_{1,1}$ $\quad$ $\neg P_{1,2} \vee B_{1,1}$

$P_{1,2}$ $\quad$ $\neg P_{1,2}$

{ }



M I

Student likes maths course and likes interesting courses

Student like datamining course D S

If student likes math course then they like statistics course $M \to S$

If student likes statistics and datamining then student is good in data analysis

$S \land D \longrightarrow A$

$R1 = M \land I$

$R2 = D$

$R3 = M \to S$

$R4 : (S \land D) \to A$

$$
\begin{aligned}
(\alpha \wedge \beta) &\equiv (\beta \wedge \alpha) && \text{commutativity of } \wedge \\
(\alpha \vee \beta) &\equiv (\beta \vee \alpha) && \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) &\equiv (\alpha \wedge (\beta \wedge \gamma)) && \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) &\equiv (\alpha \vee (\beta \vee \gamma)) && \text{associativity of } \vee \\
\neg(\neg\alpha) &\equiv \alpha && \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) && \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \vee \beta) && \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) && \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) &\equiv (\neg\alpha \vee \neg\beta) && \text{De Morgan} \\
\neg(\alpha \vee \beta) &\equiv (\neg\alpha \wedge \neg\beta) && \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) && \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) &\equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) && \text{distributivity of } \vee \text{ over } \wedge
\end{aligned}
$$

$[a \wedge (a \to b)] \equiv b$

Is the student good at analytics? A

$\downarrow$

R5: $\boxed{\sim A}$

Step	Premises	Justification – Rule Applied
1	$(S \wedge D) \to A$	
2	$\sim(S \wedge D) \vee A$	
3	$\sim S \vee \sim D \vee A$	
4		
5	M.A.I —	Both true
6	M	
6		

DPLL Algorithm

In logic and computer science, the Davis–Putnam–Logemann–Loveland (**DPLL**) **algorithm** is a complete, backtracking-based search **algorithm** for deciding the satisfiability of propositional logic formulae in conjunctive normal form

DFS

Improvements:

1. Early Termination
2. Pure Symbolic Heuristic
3. Unit Clause Heuristic

DPLL Algorithm

$R_7 : \neg P_{1,2} \vee B_{1,1}$

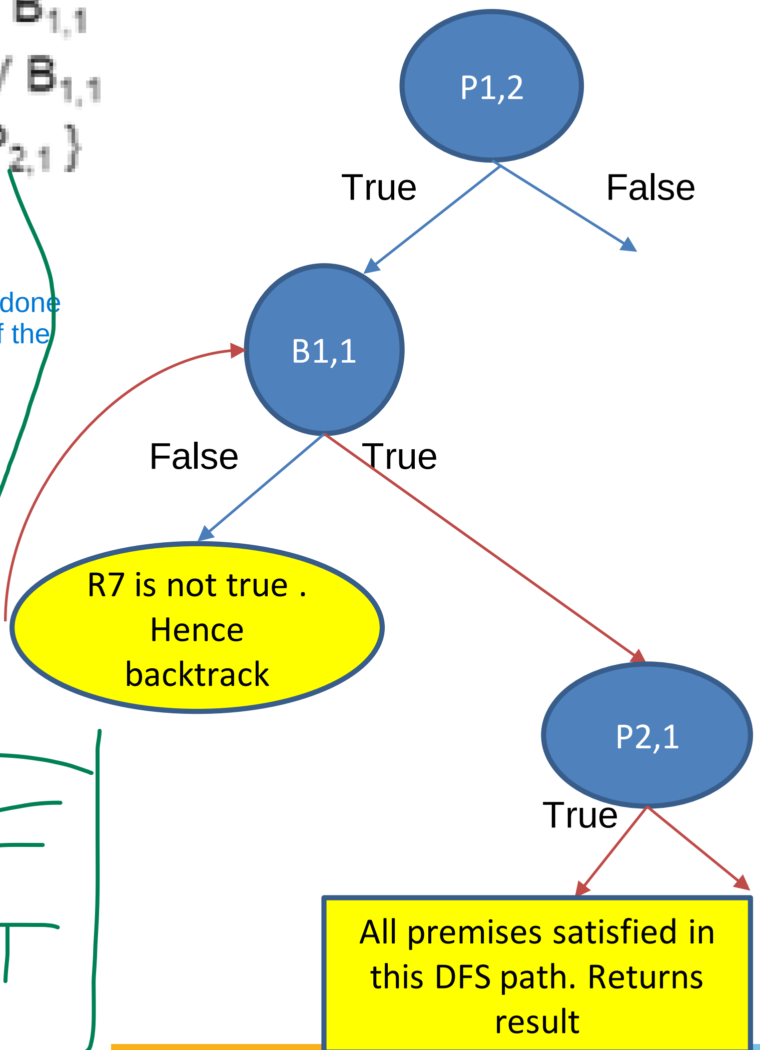
$R_8 : \neg P_{2,1} \vee B_{1,1}$

$\{P_{1,2}, B_{1,1}, P_{2,1}\}$

Three Unique literal

Truth table can be done as well. The size of the table will be 2^3=8

T	T	F
T	T	T



It is using DFS.

$R_7 : \neg \cancel{P_{1,2}} \vee B_{1,1}$

$R_8 : \neg P_{2,1} \vee B_{1,1}$

$R_7 : \neg \cancel{P_{1,2}} \vee \cancel{B_{1,1}}$

$R_8 : \neg P_{2,1} \vee \cancel{B_{1,1}}$

$R_7 : \cancel{\neg P_{1,2}} \vee B_{1,1}$

$R_8 : \cancel{\neg P_{2,1}} \vee B_{1,1}$

Towards Predicate Logic

Unary predicate. Only one variable.

$\forall$ $O(x)$ $I(x)$

All courses are offered and interesting $\forall x\, O(x) \wedge I(x)$

First order logic $f(v_1, v_2, \ldots, v_n)$

All offered courses are interesting $\forall x\, O(x) \rightarrow I(x)$

$\exists$

Some of the courses are offered and interesting [Atleast one of the offered courses is interesting] $\exists x [O(x) \wedge I(x)]$

Some of the offered courses are interesting

$\exists x\, O(x) \rightarrow I(x)$

$\exists x [O(x) \wedge I(x)]$

Predicate Logic

Squares neighboring the wumpus are smelly

Objects: squares, wumpus

Unary Relation (properties of an object): smelly

N-ary Relation (between objects): neighboring

Function: -

Primary difference between propositional and first-order logic lies in “ontological commitment” – the assumption about the nature of reality.

Predicate Logic – Sample Modelling

1. "Squares neighboring the wumpus are smelly"

$\forall x,y\ Neighbour(x,y) \wedge Wumpus(y) \Rightarrow Smelly(x)$

2. "Everybody loves somebody"

$\forall x\ \exists y\ Loves(x, y)$

∃ or [illegible]

3. "There is someone who is loved by everyone"

$\exists y\ \forall x\ Loves(x, y)$

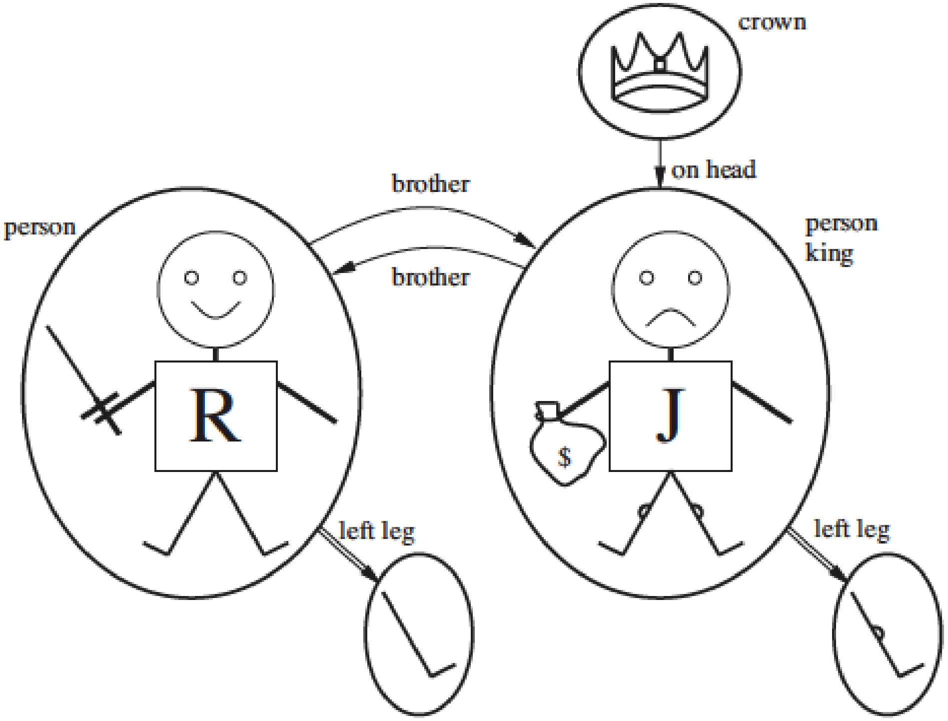
Order of quantifiers is important

Predicate Logic – Sample Modelling

Brother(Richard, John) ∧ Brother(John, Richard)

King(Richard) ∨ King(John)

¬King(Richard) ⟹ King(John)

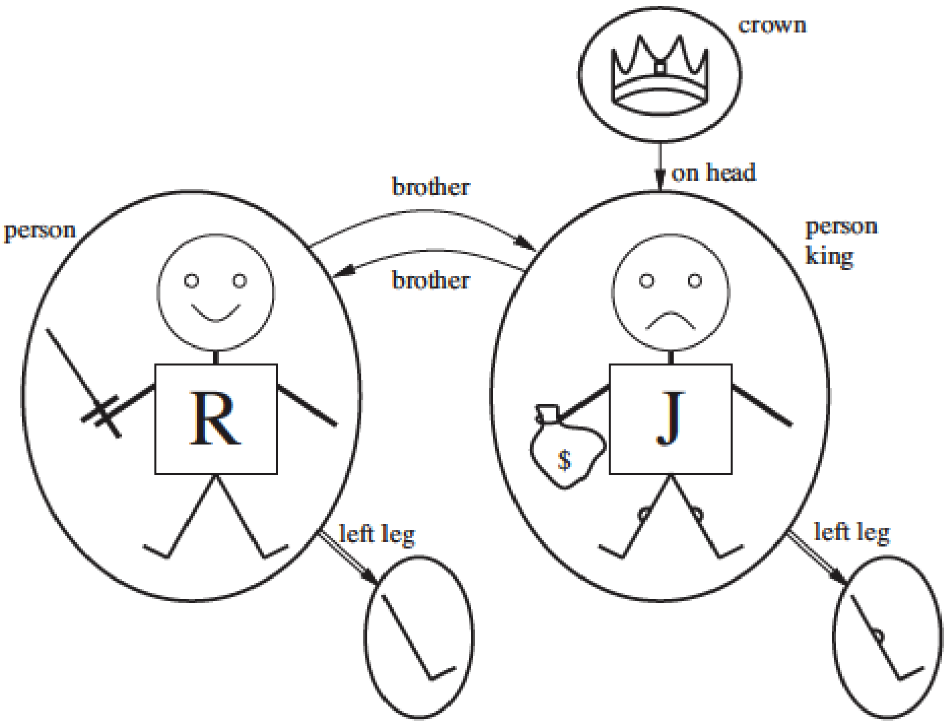


Unification & Lifting

Brother(Richard, John) ∧ Brother(John, Richard)

King(Richard) ∨ King(John)

¬King(Richard) ⟹ King(John)

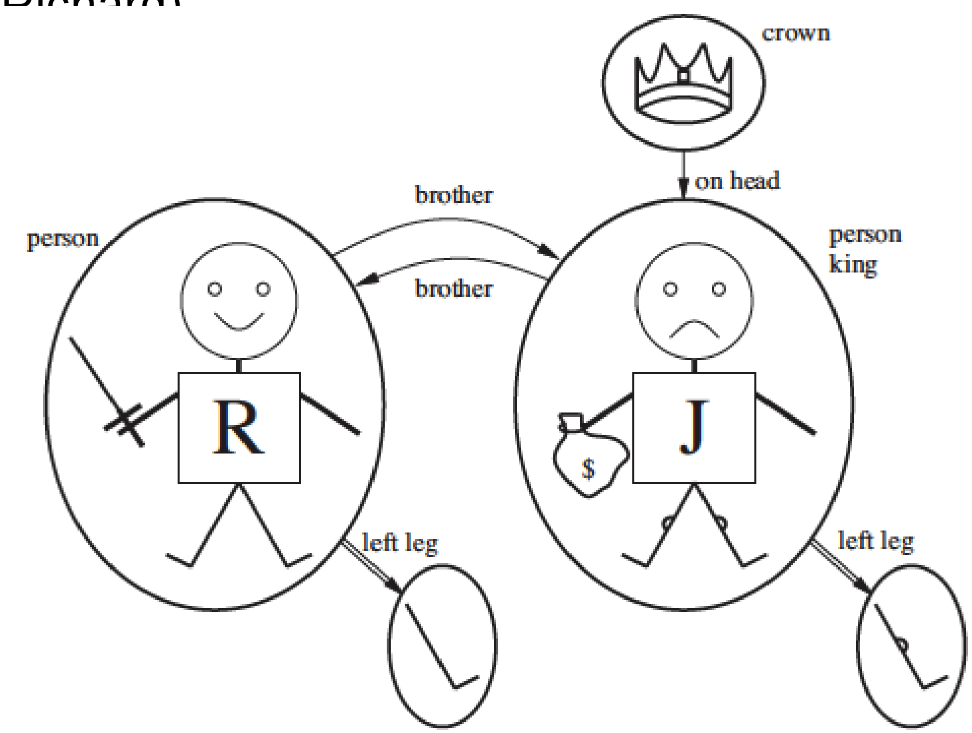


Predicate Logic – Sample Modelling Quantifiers

Brother(Richard, John) $\wedge$ Brother(John, Richard)

King(Richard) $\vee$ King(John)

$\neg$King(Richard) $\Rightarrow$ King(John)



”All Kings are persons”

$\forall x\ King(x) \Rightarrow Person(x)$

“King John has a crown on his head”

$\exists x\ Crown(x)\ \ \wedge OnHead(x, John)$

Ground Term: A term with no variables. E.g., King(Richard)

Predicate Logic – Inference

1. Substitute for Quantifiers
2. Convert into Propositional Logic
3. Apply inference tech

$\forall x \; King(x) \Longrightarrow Person(x)$

Richard the Lionheart is a king ⟹ Richard the Lionheart is a person

King John is a king ⟹ King John is a person

$\exists x \; Crown(x) \;\; \land OnHead(x, John)$

Crown(C_1) ∧ OnHead(C_1, John) ||C1 is imputed assumed fact

Forward Chaining

- Consider the following problem:

The law says it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

P P P P P P F F P P

- We will prove that West is a criminal

Forward Chaining

- First, we will represent the facts in First Order Definite Clauses

" … it is a crime for an American to sell weapons to hostile nations"

$$American(x) \land Weapon(y) \land Sells(x, y, z) \land Hostile(z) \Rightarrow Criminal(x)$$

"Nono … has some missiles"

$$\exists x \; Owns(Nono, x) \land Missile(x)$$

is transformed into two definite clauses by Existential Instantiation

$$Owns(Nono, M_1)$$

$$Missile(M_1)$$

Forward Chaining

- "All of its missiles were sold to it by Colonel West"

$$Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$$

- Missiles are weapons

$$Missile(x) \Rightarrow Weapon(x)$$

- Hostile means enemy

$$Enemy(x, America) \Rightarrow Hostile(x)$$

- "West, who is American"

$$American(West)$$

- "The country Nono, an enemy of America"

$$Enemy(Nono, America)$$

Forward Chaining

1. $American(x) \wedge Weapon(y) \wedge Sells(x,y,z) \wedge Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \wedge Owns(Nono,x) \Rightarrow Sells(West,x,Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x,America) \Rightarrow Hostile(x)$

Missile(M1)

Owns(Nono, M1)

American (West)

Enemy (Nono, America)

Required Reading: AIMA - Chapter # 4.1, #4.2, #5.1, #9

Next Session Plan:

- (Prerequisite Reading : Refresh the basics of probability , Bayes Theorem , Conditional Probability, Product Rule, Conditional Independence, Chain Rule)

- Bayesian Network
- Representation
- Inferences (Exact and approximate-only Direct sampling)

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

Artificial & Computational Intelligence

AIML CLZG557

M4 : Knowledge Representation using Logics

&

M5 : Probabilistic Representation and Reasoning

Raja vadhana P

Assistant Professor,

BITS - CSIS

BITS Pilani

Pilani Campus

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

M5 Probabilistic Representation and Reasoning

M6 Reasoning over time

M7 Ethics in AI

Knowledge Representation Using Logics

Learning Objective

At the end of this class , students Should be able to:

1. Represent a given knowledge base into logic formulation
2. Infer facts from KB using Resolution
3. Infer facts from KB using Forward Chaining
4. Infer facts from KB using Backward Chaining

Forward Chaining

- Consider the following problem:

The law says it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

- We will prove that West is a criminal

Forward Chaining

- First, we will represent the facts in First Order Definite Clauses

" … it is a crime for an American to sell weapons to hostile nations"

$$American(x) \wedge Weapon(y) \wedge Sells(x,y,z) \wedge Hostile(z) \Rightarrow Criminal(x)$$

"Nono … has some missiles"

$$\exists x\ Owns(Nono, x) \wedge Missile(x)$$

is transformed into two definite clauses by Existential Instantiation

$$Owns(Nono, M_1)$$

$$Missile(M_1)$$

Forward Chaining

- "All of its missiles were sold to it by Colonel West"

$$Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$$

- Missiles are weapons

$$Missile(x) \Rightarrow Weapon(x)$$

- Hostile means enemy

$$Enemy(x, America) \Rightarrow Hostile(x)$$

- "West, who is American"

$$American(West)$$

- "The country Nono, an enemy of America"

$$Enemy(Nono, America)$$

Forward Chaining

1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$

Missile(M1)

Owns(Nono, M1)

American (West)

Enemy (Nono, America)

Forward Chaining

- Consider the following problem:

The law says it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

- We will prove that West is a criminal

Algorithm:

1. Start from the facts
2. Trigger all rules whose premises are satisfied
3. Add the conclusion to known facts
4. Repeat the steps till no new facts are added or the query is answered

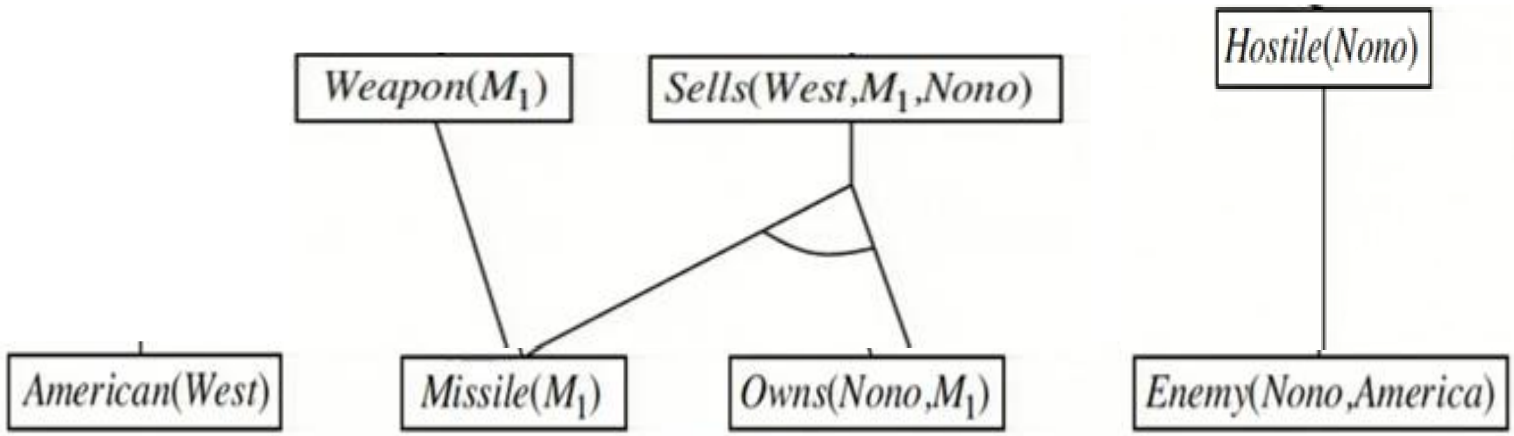
Forward Chaining

1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$

Missile(M1)
Owns(Nono, M1)
American (West)
Enemy (Nono, America)

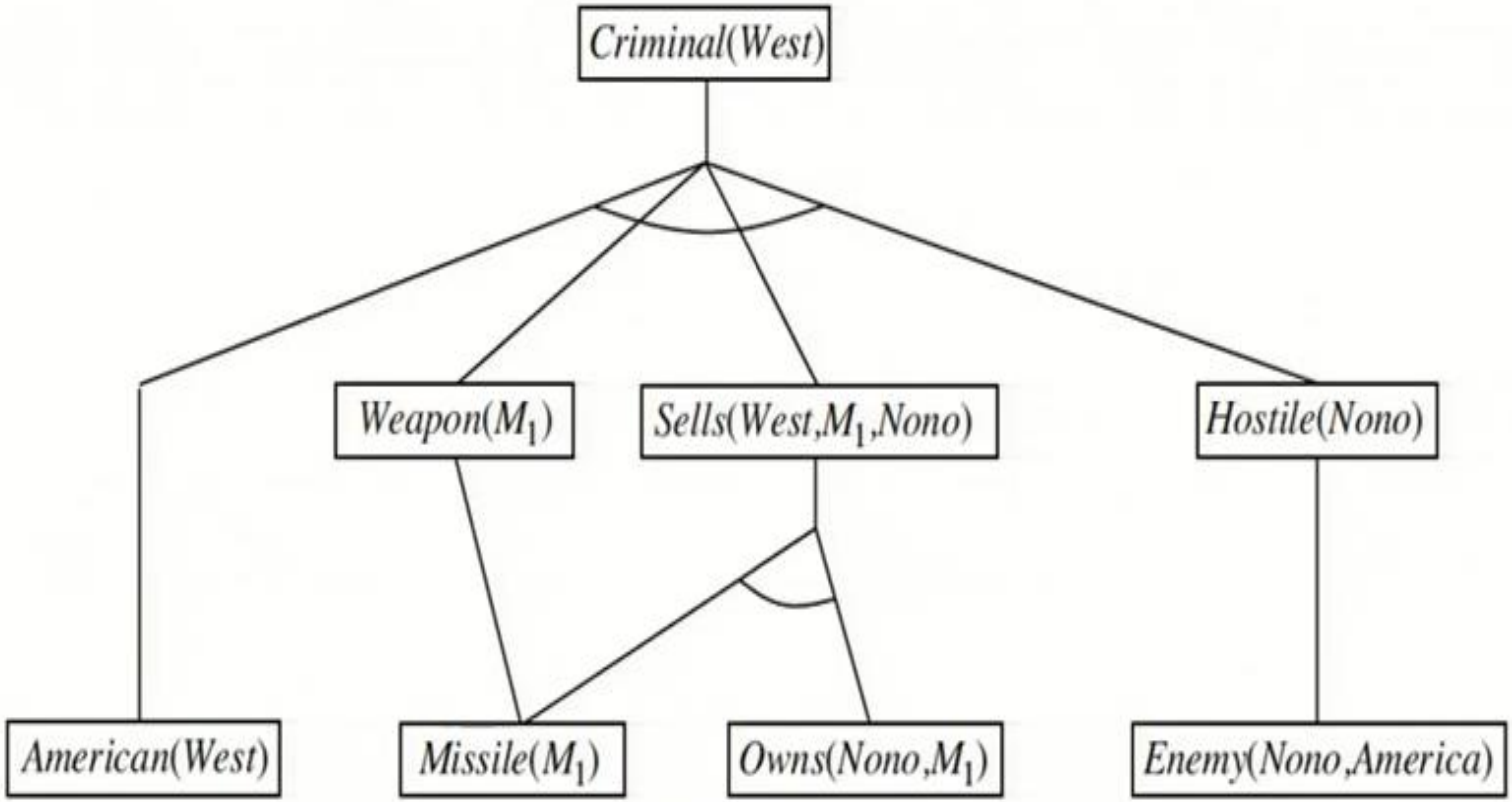
Forward Chaining

1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$



Forward Chaining

1. $American(x) \wedge Weapon(y) \wedge Sells(x,y,z) \wedge Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \wedge Owns(Nono,x) \Rightarrow Sells(West,x,Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x,America) \Rightarrow Hostile(x)$



Forward Chaining

Algorithm:

1. Start from the facts - Conjunct Ordering
2. Trigger all rules whose premises are satisfied - Pattern Matching
3. Add the conclusion to known facts – **Irrelevant Facts**
4. Repeat the steps till no new facts are added or the query is answered – Redundant Rule Matching

Backward Chaining

Algorithm:

1. Form Definite Clause
2. Start from the Goals
3. Search through rules to find the fact that support the proof
4. If it stops in the fact which is to be proved → Empty Set- LHS

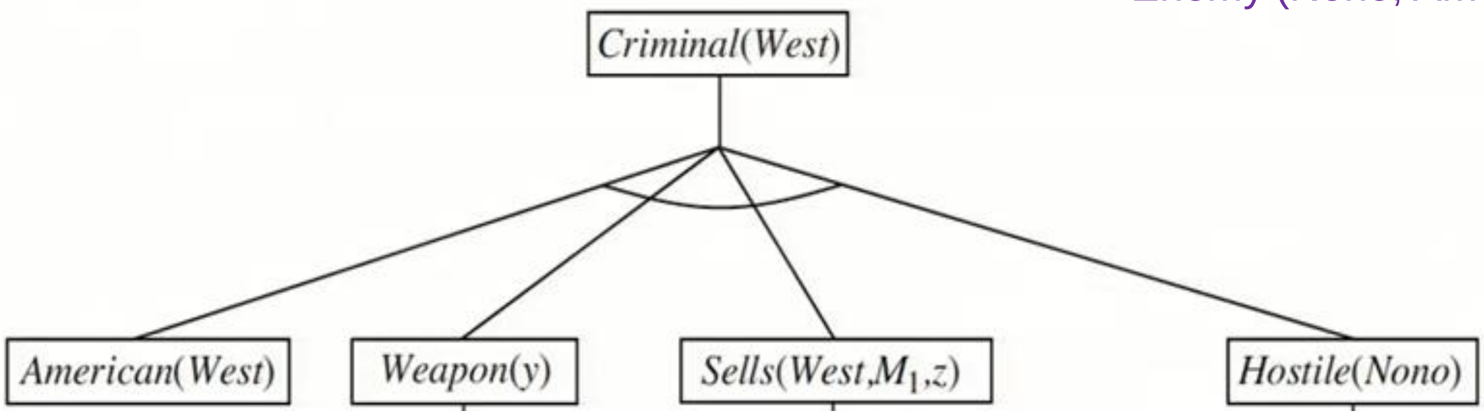
Divide & Conquer Strategy

Substitution by Unification

Backward Chaining

1. $American(x) \wedge Weapon(y) \wedge Sells(x,y,z) \wedge Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \wedge Owns(Nono,x) \Rightarrow Sells(West,x,Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$

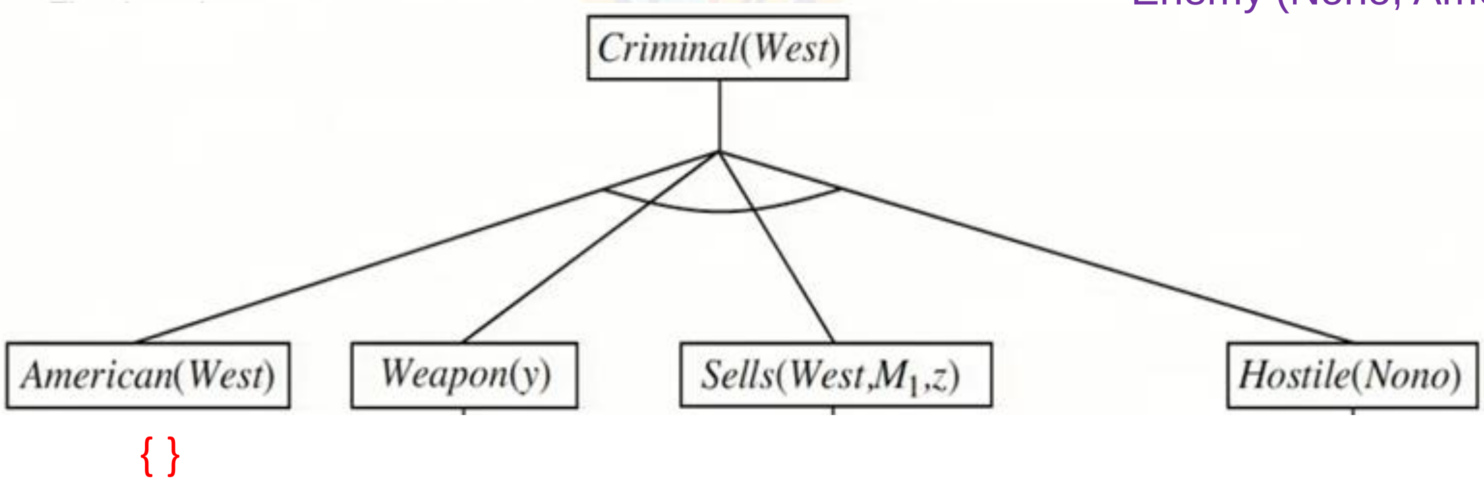
Missile(M1)
Owns(Nono, M1)
American (West)
Enemy (Nono, America)



Backward Chaining

1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$

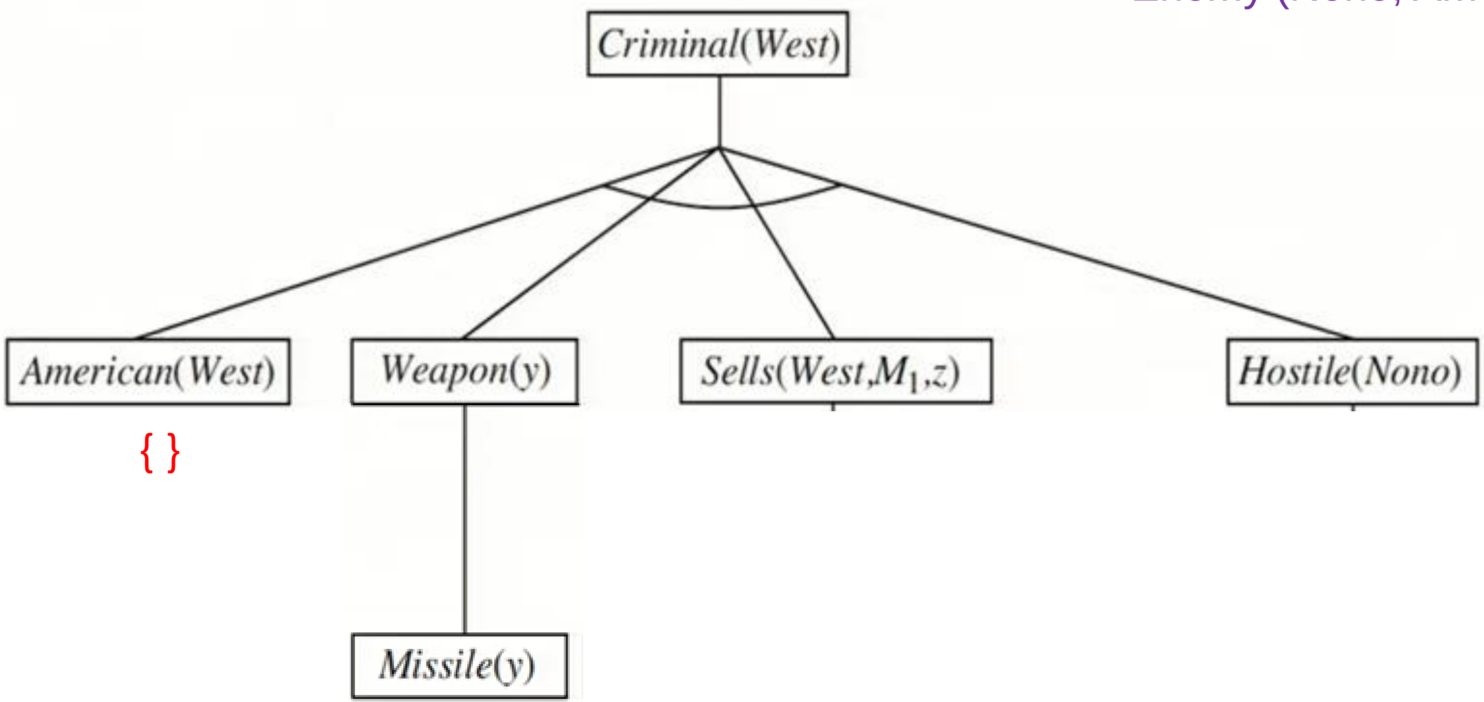
Missile(M1)
Owns(Nono, M1)
American (West)
Enemy (Nono, America)



Backward Chaining

1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$

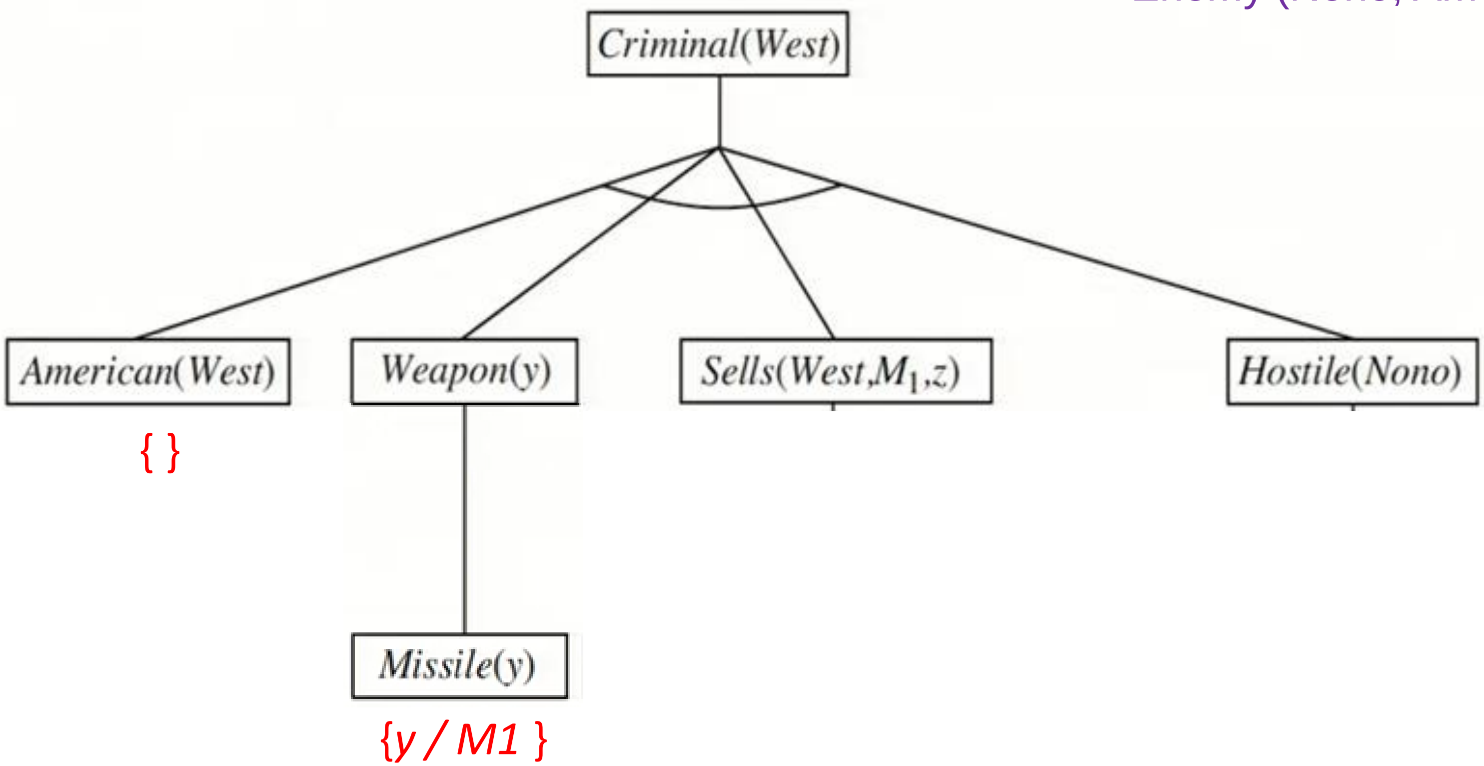
Missile(M1)
Owns(Nono, M1)
American (West)
Enemy (Nono, America)



Backward Chaining

1. $American(x) \land Weapon(y) \land Sells(x, y, z) \land Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \land Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$

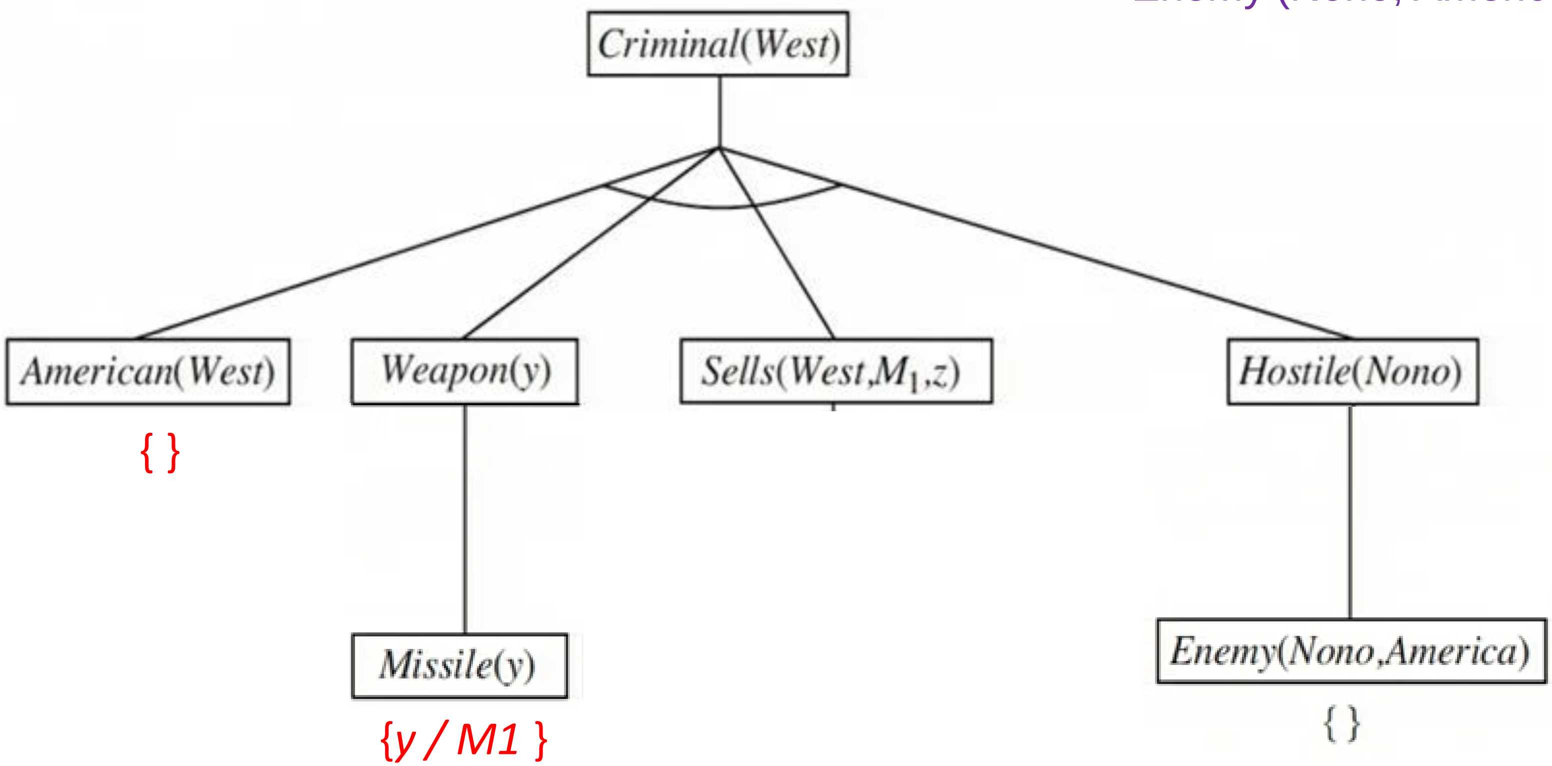
Missile(M1)
Owns(Nono, M1)
American (West)
Enemy (Nono, America)



Backward Chaining

1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$

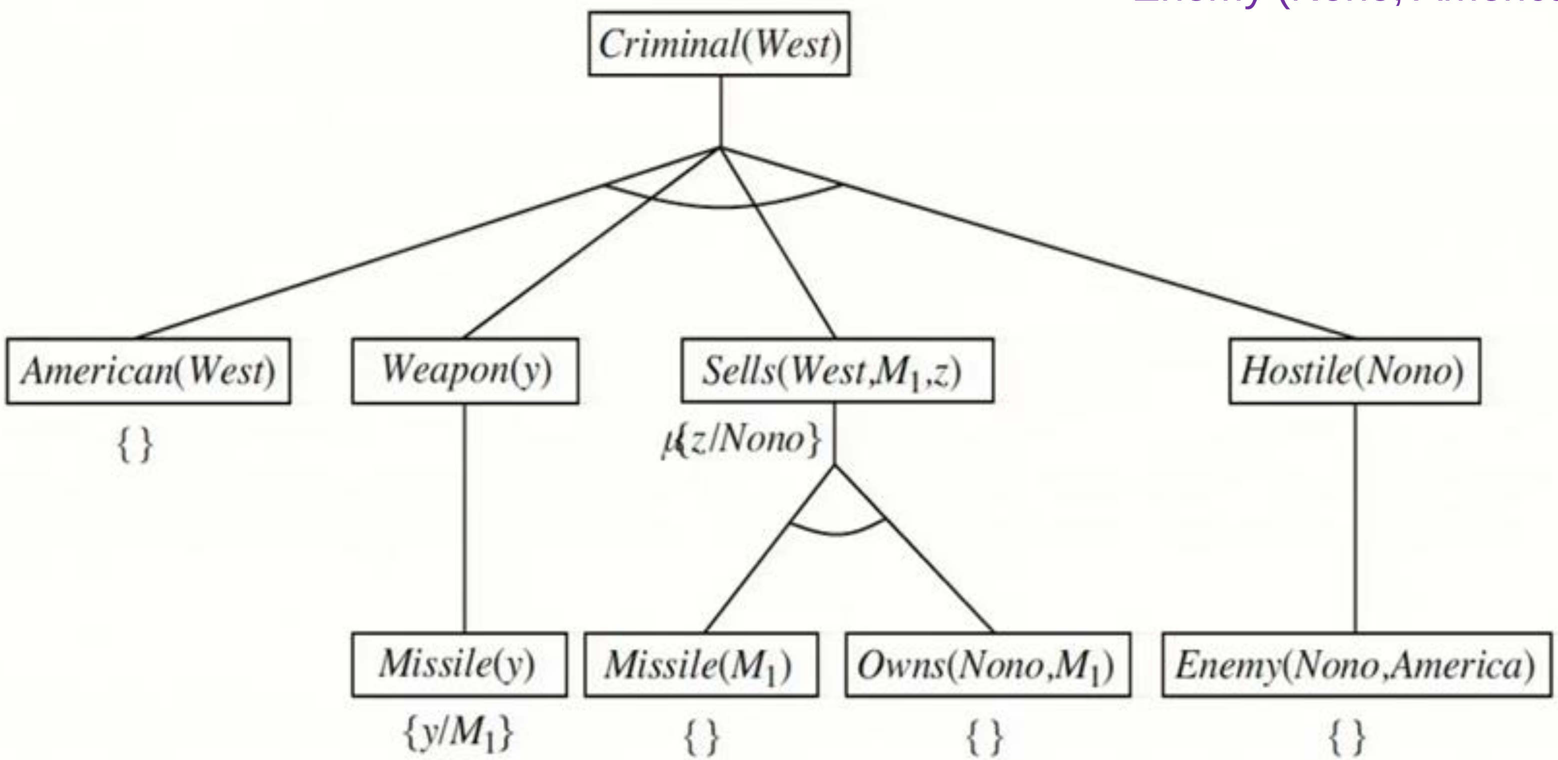
Missile(M1)
Owns(Nono, M1)
American (West)
Enemy (Nono, America)



Backward Chaining

1. $American(x) \land Weapon(y) \land Sells(x, y, z) \land Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \land Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$

Missile(M1)
Owns(Nono, M1)
American (West)
Enemy (Nono, America)



Exercise:

All courses offered are interesting	O(Z) → I(Z)
Students like easy to score courses	E(A) → L(s, A)
Data Mining is a Compute Science course	C(dm, cs)
Some of the easy courses are interesting	E(B) → I(B)
All Math are interesting course	C(E,math) → I(E)
Statistics is a math course	C(stat, math)
All computer science course are easy	C(D,cs) → E(D)

Apply chaining for below :

Q1: Do students like statistics?

Q2: What course does students like?

Predicate Logic Representation
Quantifier Instantiation
Theorem Proving for question Q1 (Proof by Direct & Indirect Proof by Contradiction & Proof by Resolution)
Reasoning using Chaining for Q1 and Q2

Predicate Logic

(Interesting Case Studies)

Robotic Process Automation

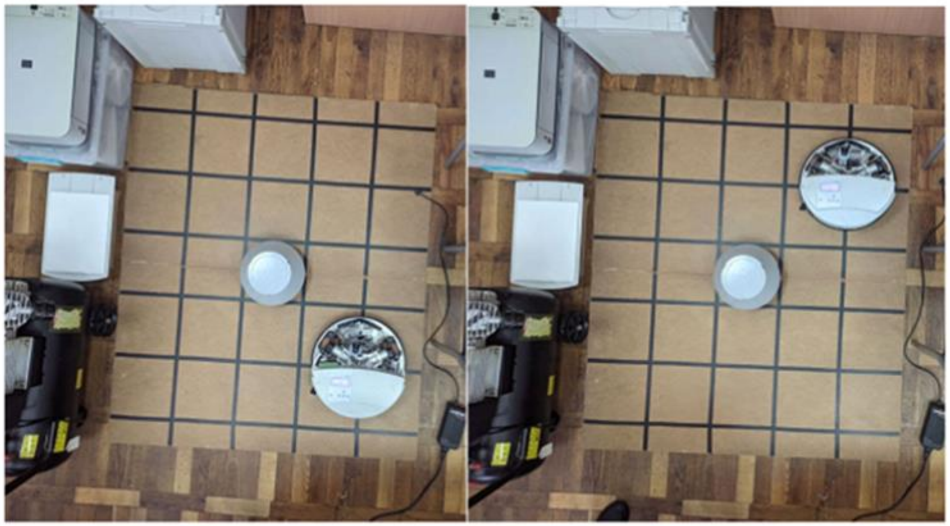


Figure 3. Example of discrete workspace for the Festo Robotino.

States:
is_at(robot, door45, now)
is_with(robot, box1, now))
stands(door12, closed, now)
is_in(box1, room1. now))
Actions:
“go_to”;“open_door”; “take_box”; “push_box”;

$$pg^0(pr,ps) \leftarrow pg_0^0(pr_0,ps_0,pa_0) \wedge pg_1^0(pr_1,ps_1,pa_1) \wedge \ldots \wedge pg_{n-1}^0(pr_{n-1},ps_{n-1},pa_{n-1}) = \wedge_{i=0}^{n-1} pg_i^0(pr_i,ps_i,pa_i)$$

$$pg^m(pr,ps) \leftarrow pg_0^m(pr_0,ps_0,pa_0) \wedge pg_1^m(pr_1,ps_1,pa_1) \wedge \ldots \wedge pg_{n-1}^m(pr_{n-1},ps_{n-1},pa_{n-1}) = \wedge_{i=0}^{n-1} pg_i^m(pr_i,ps_i,pa_i).$$

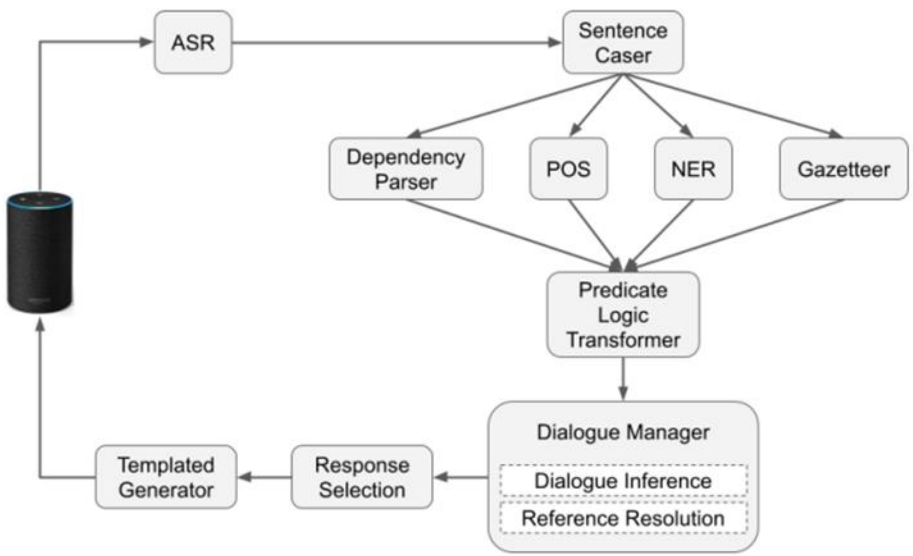
, the global (final) goal is defined as follows:

$$pg^{total}(pr, ps) \leftarrow \vee_{j=0}^{m-1} \wedge_{i=0}^{n-1} pg_i^m(pr_i, ps_i, pa_i).$$

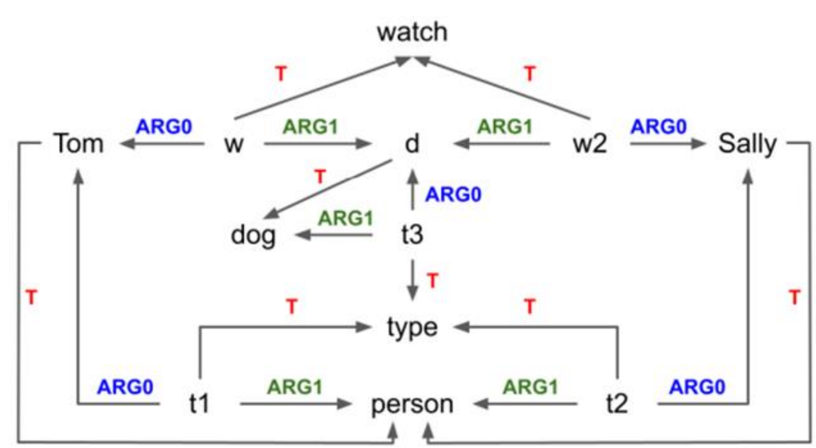
Source Credit:Tsymbal, O.; Mercorelli, P.; Sergiyenko, O.
Model of Problem-Solving for Robotic Actions Planning. Mathematics 2021, 9, 3044.

innovate achieve lead

Natural Language – Chat bot



English Sentence	Predicate Logic
I ran on the treadmill.	r/run(user) time(r, past) on(r, t) type(t, treadmill)
I like watching action movies.	l/like(user, watch(user, m)) time(l, now) type(m, movie) type(m, group) action(m)
Your dog is sweet.	s/sweet(d) time(s, now) type(d, dog) possess(bot, d)
I am a math teacher.	b/be(user, t) time(b, now) type(t, teacher) of(t, math)
My grades fell quickly after I stopped studying.	f/fall(g) time(f, past) type(g, grade) type(g, group) possess(user, g) quick(f) after(f, s/stop(user, study(user)) time(s, past)
I didn't eat lunch yet.	e/eat(user, l) type(l, lunch) not(e) time(e, past)
I should eat lunch.	e/eat(user, l) type(l, lunch) should(e) time(e, now)
What musical instrument do you play?	p/play(bot, i) type(i, musical_instrument) time(p, now) request(user, i)
Did you like the book I gave you?	l/like(bot, b) type(b, book) time(l, past) g/give(user, b) recipient(g, bot) time(g, past) request_truth(user, l)



Source Credit: [An Approach to Inference-Driven Dialogue Management within a Social Chatbot](#)
[4th Proceedings of Alexa Prize (Alexa Prize 2020)](#)

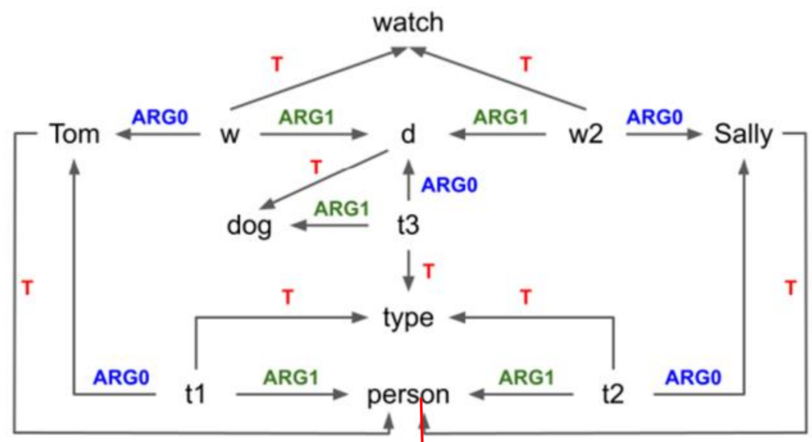
Natural Language – Chat bot

I/like(person(), movie())
→
cause(I, reason())

(a) Inference rule representing the common sense notion that a person likes a movie for a reason

I/like(user, X/item())
cause(I, reason())
→
What do you like about X?

(b) Template rule transforming the logical forms from (a) to an appropriate natural language expression



Rule	Solution	Span Attachments	Concept Attachments
(a)	X: "watched" Y: "Tom"	("watched", ARG0, "Tom")	(w, ARG0, Tom)
(b)	X: "watched" Y: "dog"	("watched", ARG1, "dog")	(w, ARG1, d)
(c)	X: "watched" Y: "stop" Z: "by"	("by", ARG0, "watched") ("by", ARG1, "stop")	(b, ARG0, w) (b, ARG1, bs)
(c)	X: "stop" Y: "park" Z: "near"	("near", ARG0, "stop") ("near", ARG1, "park")	(n, ARG0, bs) (n, ARG1, cp)

Span	Concept
Tom	Tom
watched	$w * /watch(-,-)$
dog	$type(d*, dog)$
by	$b * /by(-,-)$
bus stop	$type(bs*, bus\_stop)$
near	$n * /near(-,-)$

(a) Concepts from Gazetteer

Span	Concept
Tom	$type(t*, per)$
Central Park	$type(cp*, loc)$

(b) Concepts based on NER

Span	Concept
Tom	$type(t*, nnp)$
watched	$type(w*, vbd)$
dog	$type(d*, nn)$
by	$type(b*, in)$
bus	$type(b*, nn)$
stop	$type(s*, nn)$
near	$type(n*, in)$
Central	$type(c*, nnp)$
Park	$type(p*, nnp)$

(c) Concepts based on POS

Source Credit: [An Approach to Inference-Driven Dialogue Management within a Social Chatbot 4th Proceedings of Alexa Prize (Alexa Prize 2020)]

Natural Language – Chat bot

Bot: I'm a big fan of action movies.
User: Yeah, I like the Avengers .
Bot: What do you like about the Avengers?

(a)

User: Let's talk about movies.
Bot: Is there a particular movie that you really like?
User: The Avengers
Bot: What do you like about the Avengers?

(b)

Bot: What did you do this weekend?
User: I watched the Avengers. It's my favorite movie.
Bot: That sounds fun. For my weekend I went hiking.
User: That's cool.
Bot: What do you like about the Avengers?

(c)

Source Credit: [An Approach to Inference-Driven Dialogue Management within a Social Chatbot](#) [4th Proceedings of Alexa Prize (Alexa Prize 2020)](#)

Reasoning

Module 5:

Probabilistic Representation and Reasoning

A. Inference using full joint distribution

B. Bayesian Networks

   I. Knowledge Representation

   II. Conditional Independence

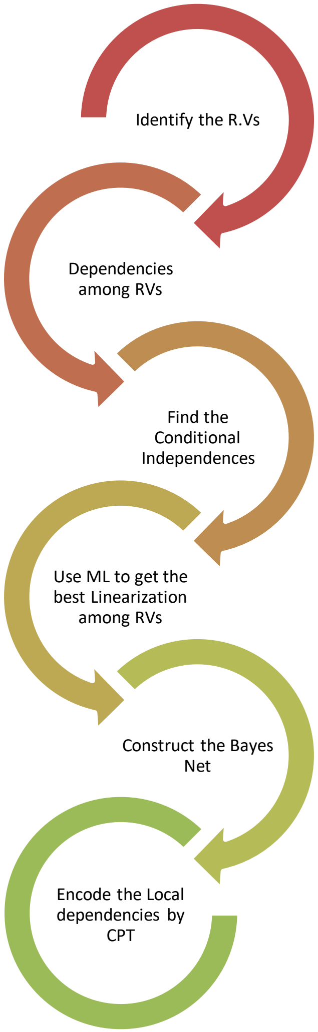
   III. Exact Inference

   IV. Introduction to Approximate Inference

Bayesian Net???

Wumpus World Problem

- Wumpus Ghost traces of scent in the visited cell
- Earlier visited cell may become unsafe!!!
- **Problem**: Given the information that there is a possibility of apparition of Wumpus anywhere in the cave, AI agent needs to be safely travel with more caution!!



Reasoning

- Monotonic Reasoning

- Non- Monotonic Reasoning

  Dependency Directed Backtracking: when a statement is deleted as “ no more valid”, other related statements have to be backtracked and they should be either deleted or new proofs have to be found for them. This is called dependency directed backtracking (DDB)

Reasoning

- Monotonic Reasoning
- Non- Monotonic Reasoning

Monotonic	Non-Monotonic
Consistent	Relaxed Consistency
Complete Knowledge	Incomplete Knowledge
Static	Dynamic
Discrete	Continuous & Learning Agent
Predicate Logic	Probabilistic Model

Uncertainty

You can reach Bangalore Airport from MG Road within 90 mins if you go by route A.

- There is uncertainty in this information due to partial observability and non determinism
- Agents should handle such uncertainty

Previous approaches like Logic represent all possible world states

Such approaches can’t be used as multiple possible states need to be enumerated to handle the uncertainty in our information

Uncertainty

You can reach Bangalore Airport from MG Road within 90 mins if you go by route A.

Road Block	Festival Season	Weekend	Observation (20)	Prob
F	F	F	12	0.6
F	F	T	3	0.15
F	T	F	2	0.1
F	T	T	2	0.1
T	F	F	0	0
T	F	T	0	0
T	T	F	1	0.05
T	T	T	0	0
				=1

Belief

You can reach Bangalore Airport from MG Road within 90 mins if you go by route A.

“we are 80% confident that it would be true on any given day”

Augmentation: If we know that it is evening, the Probability of the statement can be 0.4

Probability Basics Refresher – Self Study

Probability Theory

Basics
Conditional Probability
Chain Rule
Independence
Conditional Independence
Belief Nets
Joint Probability distribution

Conditional Probability

Towards Chain Rule:

P(a | b) = P(a,b) / P(b)

P(a, b) = P(a | b) P(b)

P(a, b , c) = P(a , x) where x= b,c

P(a,x) = P(a | x) . P(x)

= P(a | bc) . P(b, c)

= P(a | bc) . P(b | c). P(c)

Hence : P(a,b,c) = P(a | bc) . P(b | c). P(c)

Chain Rule : Generalization

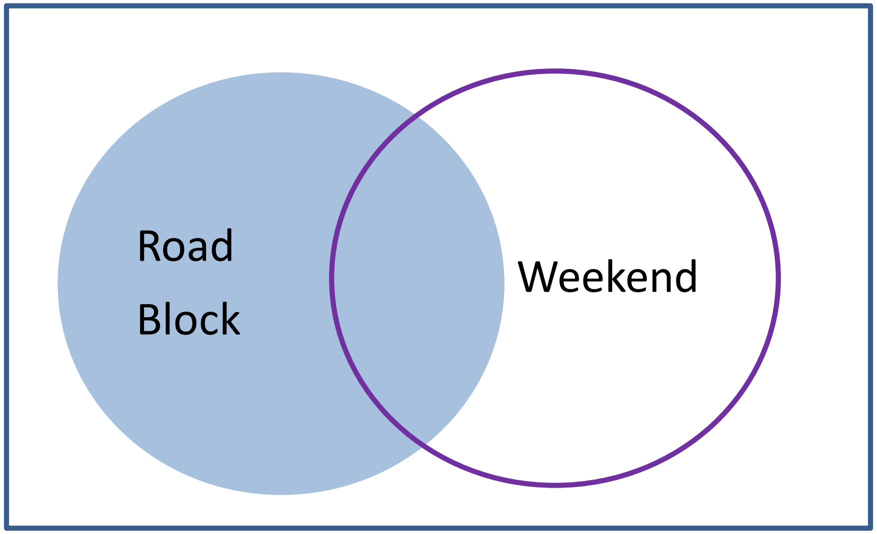
$$\mathbf{P(X_1, X_2, \ldots X_K) = \prod P(X_i \mid X_{i-1}, \ldots X_1)}$$

Where i = k to 1 (reverse)

Probability Theory

<u>Independence</u>

P(a | b) = P(a)



Implication:

P(a | b) = P(a,b) / P(b)

P(a) = P(a,b) / P(b)

P(a,b) = P(a) . P(b)

<u>Conditional Independence</u>

P(a | b c) = P(a | c)

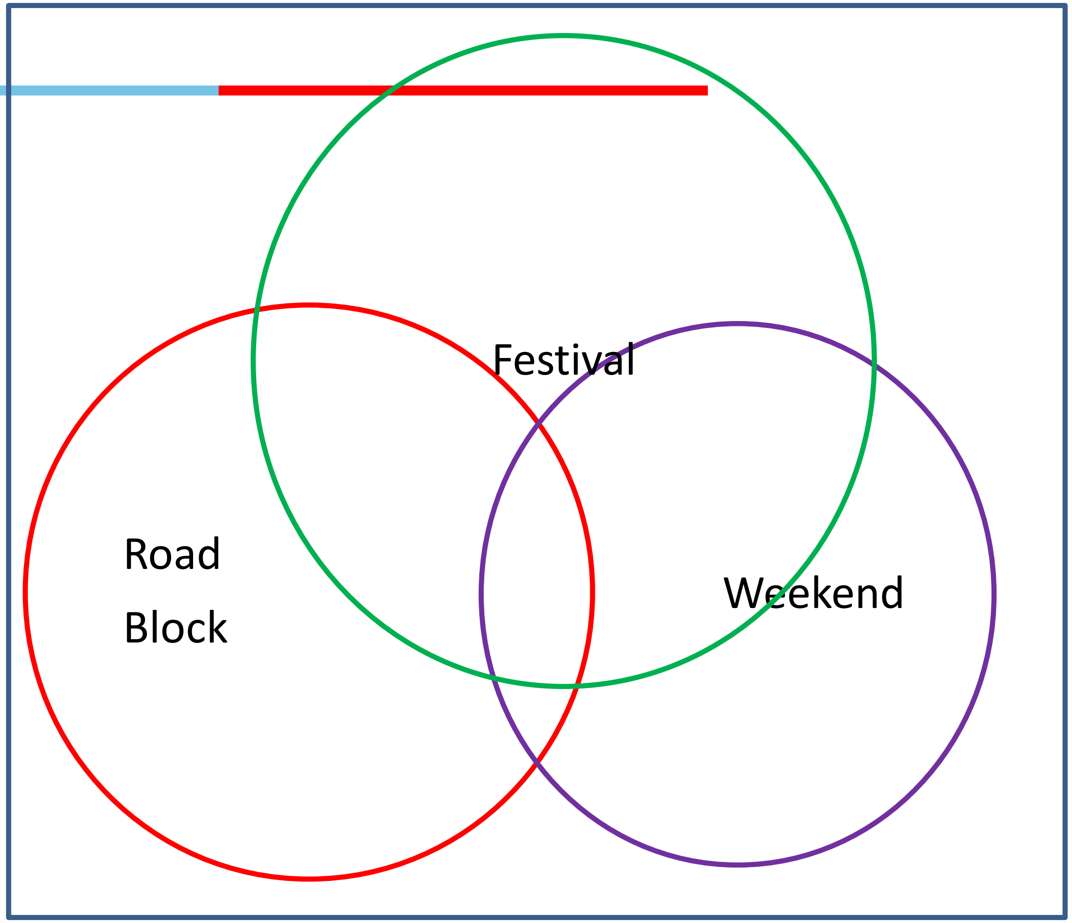
Probability Theory

<u>Conditional Independence</u>

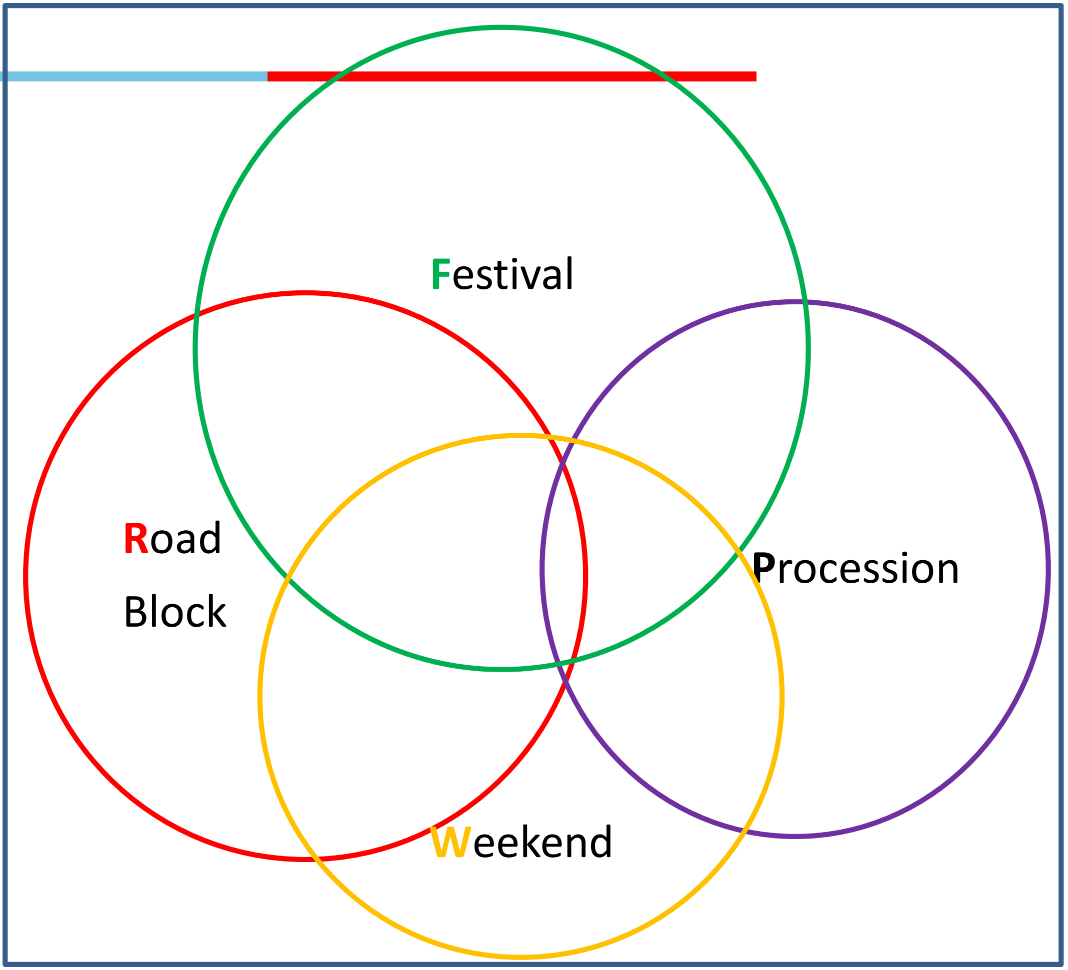
$P(a \mid b\ c) = P(a \mid c)$

Extension:

$P(a\ b \mid c) = P(a \mid c) . P(b \mid c)$



Probability Theory



Independence

$P(a \mid b) = P(a)$

$P(R \mid W) = P(R)$

Conditional Independence

$P(a \mid b\ c) = P(a \mid c)$

$P(R \mid PW) = P(R \mid P)$

Extension:

$P(a\ b \mid c) = P(a \mid c) \cdot P(b \mid c)$

$P(FP \mid R) = P(F \mid R) \cdot P(P \mid R)$

$P(RW \mid P) = P(R \mid P) \cdot P(W \mid P)$

Chain Rule:

$P(a\ b\ c\ d) = P(a \mid b\ c\ d) \cdot P(b \mid c\ d) \cdot P(c \mid d) \cdot P(d)$

$P(R\ P\ F) = P(R \mid P\ F) \cdot P(P \mid F) \cdot P(F)$

Independence

If we have two random variables, TimeToBnlrAirport and HyderabadWeather

P(TimeToBnlrAirport, HyderabadWeather)

To determine their relation, use the product rule

= P(TimeToBnlrAirport | HyderabadWeather) / P(HyderabadWeather)

However, we would argue that HyderabadWeather and TimeToBnlrAirport doesn't have any relation and hence

P(TimeToBnlrAirport | HyderabadWeather) = P(TimeToBnlrAirport)

This is called Independence or Marginal Independence

Independence between propositions a and b can be written as

$$P(a \mid b) = P(a) \quad \text{or} \quad P(b \mid a) = P(b) \quad \text{or} \quad P(a \wedge b) = P(a)P(b)$$

Bayes Rule

Using the product rule for propositions a and b

$$P(a \wedge b) = P(a \mid b)P(b) \quad \text{and} \quad P(a \wedge b) = P(b \mid a)P(a)$$

Equating the right hand sides and dividing by P(a)

$$P(b \mid a) = \frac{P(a \mid b)P(b)}{P(a)}$$

This is called the Bayes Rule

Conditional Independence

2 random variables A and B are conditionally independent given C iff

$$P(a, b \mid c) = P(a \mid c)\, P(b \mid c) \text{ for all values } a, b, c$$

More intuitive (equivalent) conditional formulation

- A and B are conditionally independent given C iff

  $P(a \mid b, c) = P(a \mid c)$ OR $P(b \mid a, c) = P(b \mid c)$, for all values a, b, c

- Intuitive interpretation:

  $P(a \mid b, c) = P(a \mid c)$ tells us that learning about b, given that we already know c, provides no change in our probability for a, i.e., b contains no information about a beyond what c provides

  $P(R \mid F, P) = P(R \mid P)$

Joint Probability Distributions

Instead of distribution over single variable, we can model distribution over multiple variables, separated by comma

E.g., **P**(A, B) = **P**(A | B) . **P**(B)

P(A, B) is the probability distribution over combination of all values of A and B

E.g., if A = Weather and B = Cavity

$$
\begin{aligned}
P(W = sunny \wedge C = true) &= P(W = sunny|C = true)\, P(C = true) \\
P(W = rain \wedge C = true) &= P(W = rain|C = true)\, P(C = true) \\
P(W = cloudy \wedge C = true) &= P(W = cloudy|C = true)\, P(C = true) \\
P(W = snow \wedge C = true) &= P(W = snow|C = true)\, P(C = true) \\
P(W = sunny \wedge C = false) &= P(W = sunny|C = false)\, P(C = false) \\
P(W = rain \wedge C = false) &= P(W = rain|C = false)\, P(C = false) \\
P(W = cloudy \wedge C = false) &= P(W = cloudy|C = false)\, P(C = false) \\
P(W = snow \wedge C = false) &= P(W = snow|C = false)\, P(C = false)\, .
\end{aligned}
$$

Probabilistic Inference

Computation of posterior probabilities given observed evidence, i.e., full joint probability distribution

	$toothache$		$\neg toothache$	
	$catch$	$\neg catch$	$catch$	$\neg catch$
$cavity$	0.108	0.012	0.072	0.008
$\neg cavity$	0.016	0.064	0.144	0.576

Query: P(cavity ∨ **toothache)**

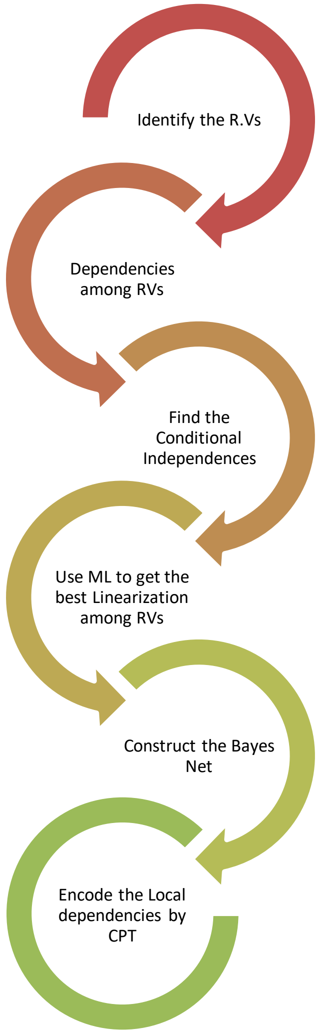
0.108 + 0.012 + 0.072 + 0.008 + 0.016 + 0.064 = 0.28

Building a Bayesian Network

Example Bayesian Net #1

A simple world with four random variables

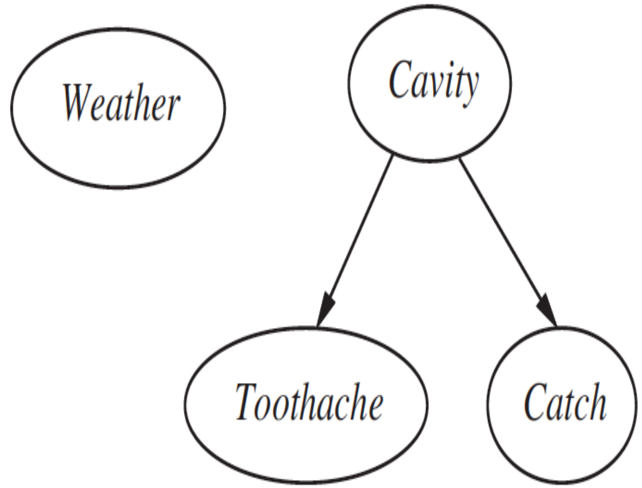
- Weather, Toothache, Cavity, Catch
- Weather is independent of other variables
- Toothache and Catch are conditionally independent given Cavity
- P(Toothache, Catch | Cavity) = P(Toothache | Cavity) . P(Catch | Cavity)
- Cavity is a direct cause of Toothache and Catch
- No direct relation between Toothache and Catch exists

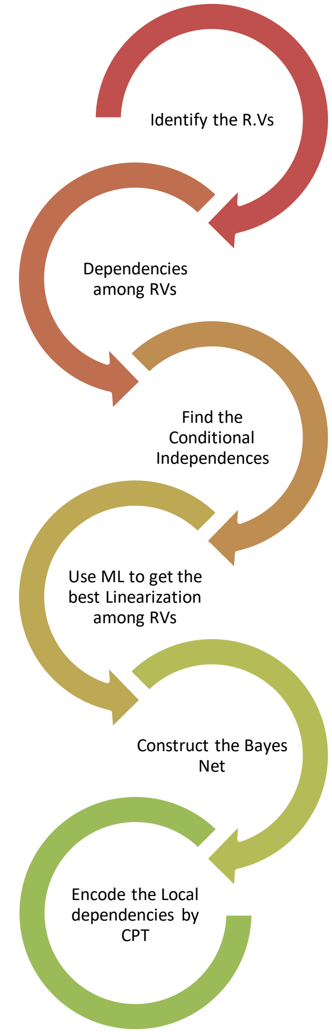


Example Bayesian Net #1

A simple world with four random variables

- Weather, Toothache, Cavity, Catch
- Weather is independent of other variables
- Toothache and Catch are conditionally independent given Cavity
- P(Toothache, Catch | Cavity) = P(Toothache | Cavity) . P(Catch | Cavity)
- Cavity is a direct cause of Toothache and Catch
- No direct relation between Toothache and Catch exists

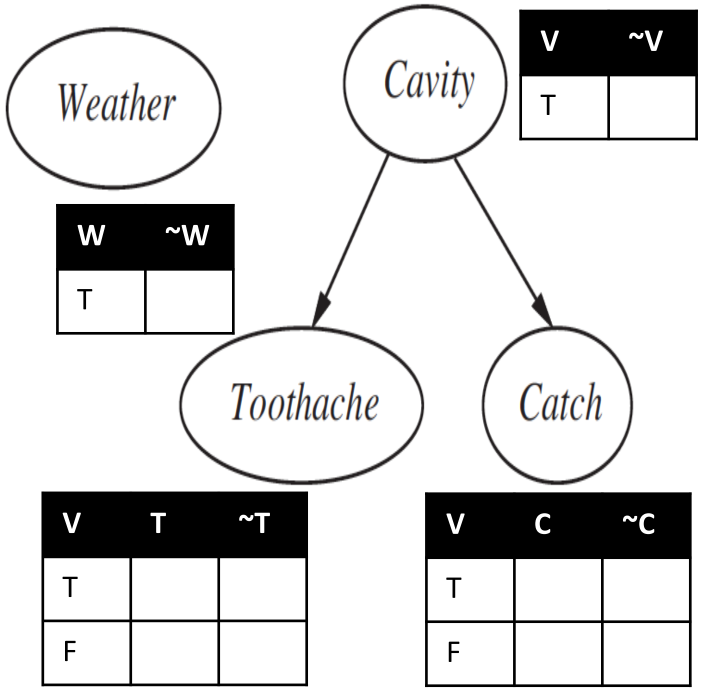


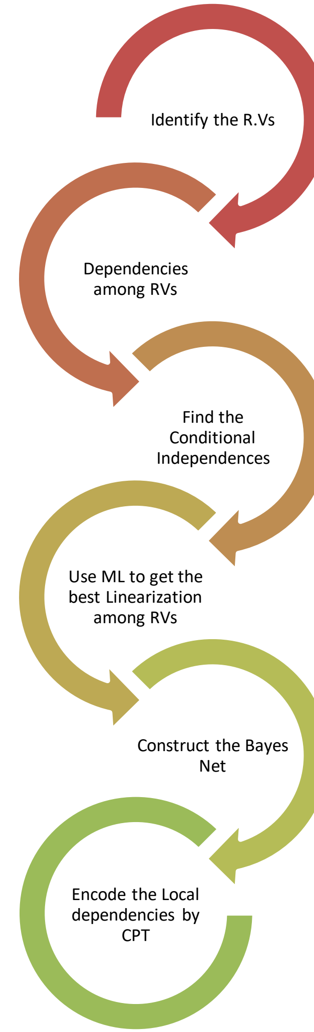


Example Bayesian Net #1

A simple world with four random variables

- Weather, Toothache, Cavity, Catch
- Weather is independent of other variables
- Toothache and Catch are conditionally independent given Cavity
- P(Toothache, Catch | Cavity) = P(Toothache | Cavity) . P(Catch | Cavity)
- Cavity is a direct cause of Toothache and Catch
- No direct relation between Toothache and Catch exists





Required Reading: AIMA - Chapter # 13, 14.1, #14.2, #14.3, #14.4, $14.5

Next Session Plan:

- Bayesian Network – Few more examples & Numerical Problems
- Inferences (Exact and approximate-only Direct sampling)
- Notion of Markov Models

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

Artificial & Computational Intelligence

AIML CLZG557

BITS Pilani
Pilani Campus

M5 : Probabilistic Representation and Reasoning

Raja vadhana P
Assistant Professor,
BITS - CSIS

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

M5 Probabilistic Representation and Reasoning

M6 Reasoning over time

M7 Ethics in AI

Reasoning

Module 5:

Probabilistic Representation and Reasoning

A. Inference using full joint distribution

B. Bayesian Networks

I. Knowledge Representation

II. Conditional Independence

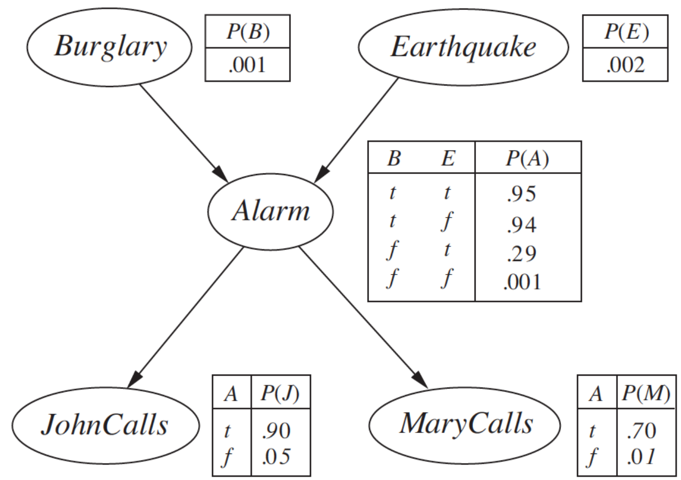
III. Exact Inference

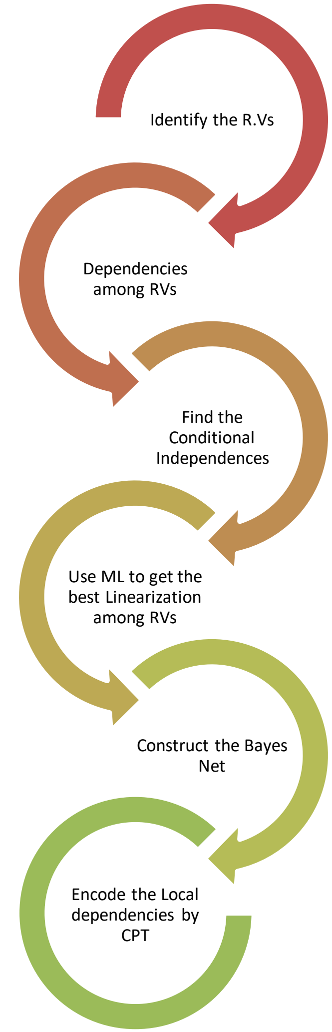
IV. Introduction to Approximate Inference

Example Bayesian Net #2

A Burglary Alarm System

- Fairly reliable on detecting a burglary
- Also responds to earthquakes
- Two neighbors John and Mary ar asked to call you at work when Burglary happens and they hear the Alarm
- John nearly always calls when he hears the alarm, however sometimes confuses the telephone ring with alarm and calls then too
- Mary like loud music and often misses the alarm altogether
- **Problem**: Given the information that who has / has not called we need to estimate the probability of a burglary

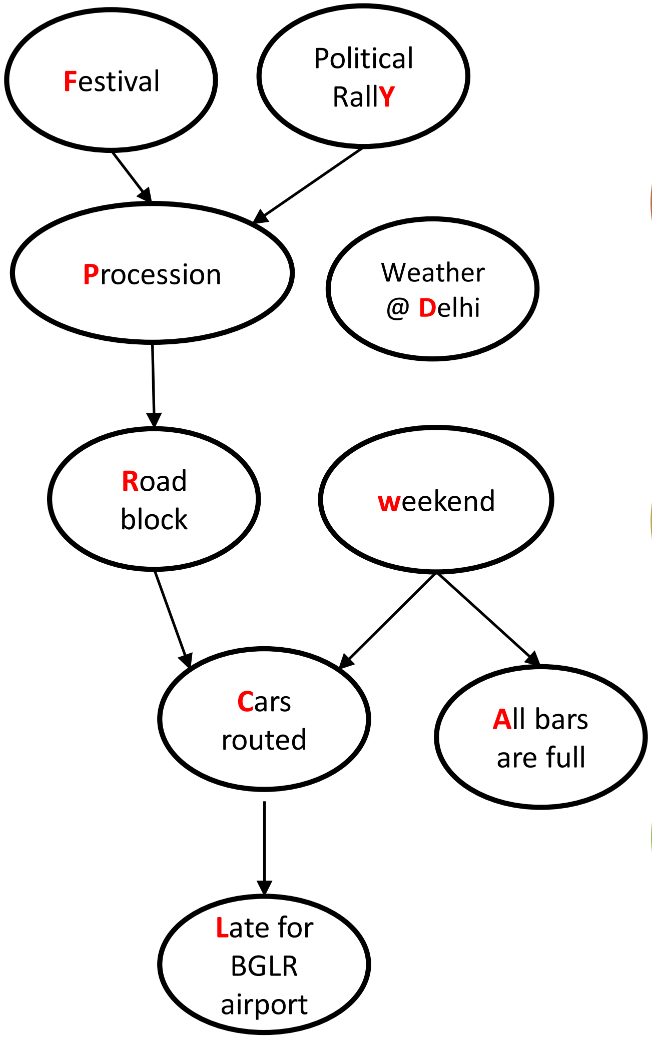


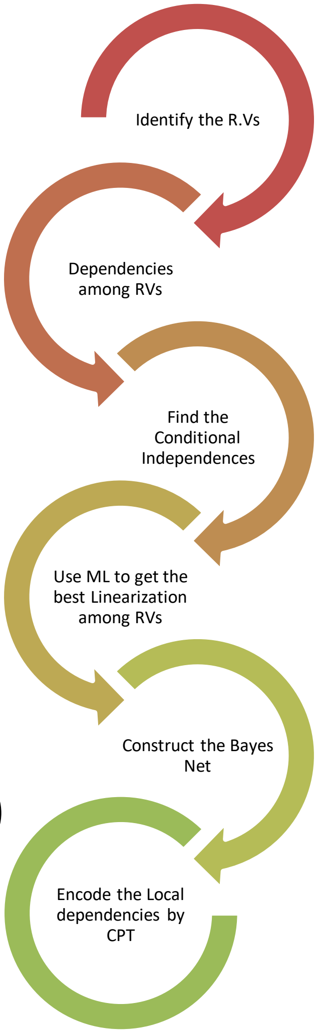


Example Bayesian Net #3

Traffic Prediction -Travel Estimation

- AI system reminds traveler regarding start time
- Travel plan is to reach Delhi and the weather of Delhi may influence the accommodation plans
- Traveler always take car to reach airport
- Car may be rerouted either due to road block or weekday traffic during working hours which delays the arrival to airport
- Bars are always observed to be full on weekends
- Authorities block roads to safe the processions
- Processions observed during festive season or due to the political rally.
- **Problem**: Given the information that there is a political rally expected estimate the probability of late arrival





Example Bayesian Net #3

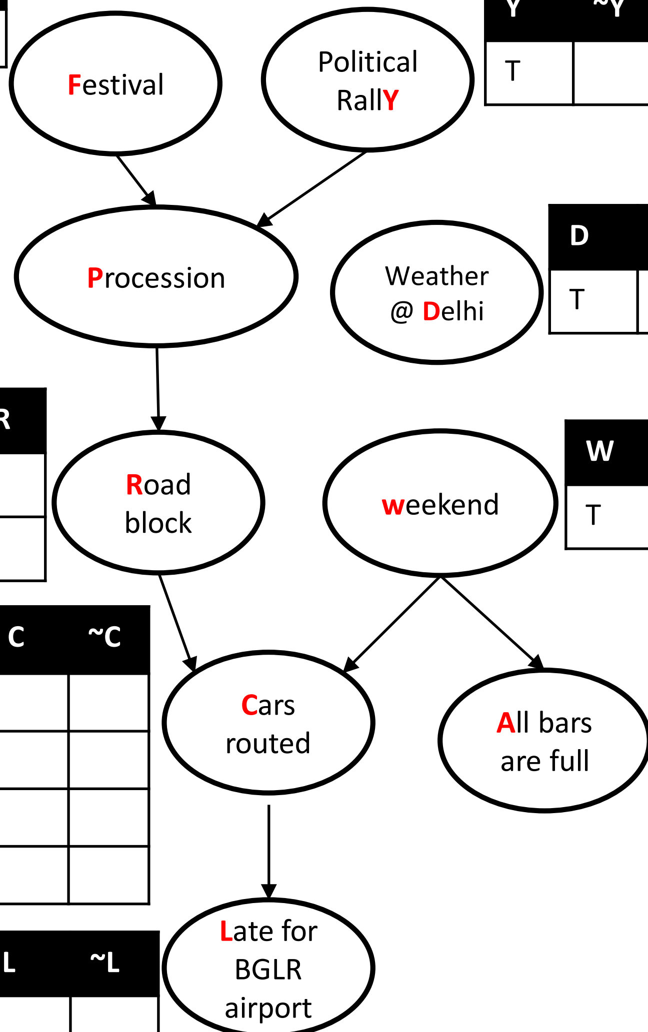
F	~F
T	

F	Y	P	~P
T	T		
T	F		
F	T		
F	F		

P	R	~R
T		
F		

R	W	C	~C
T	T		
T	F		
F	T		
F	F		

C	L	~L
T		
F		

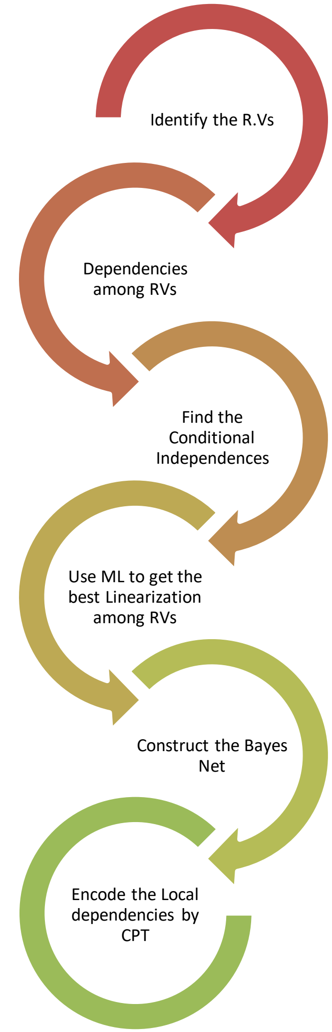


Y	~Y
T	

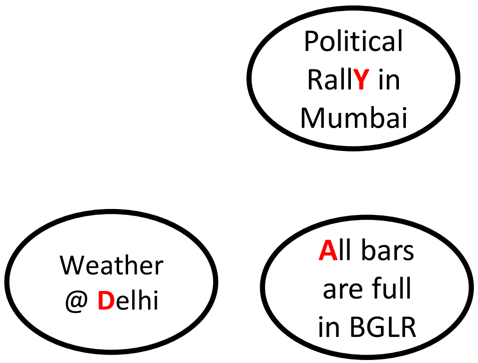
D	~D
T	

W	~W
T	

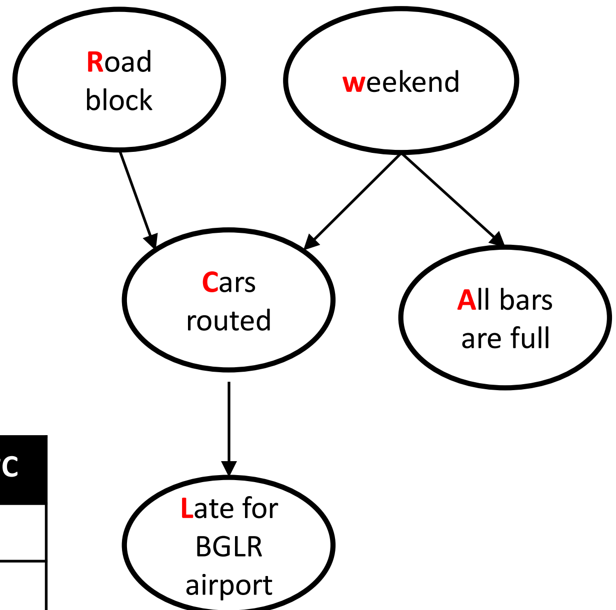
W	A	~A
T		
F		



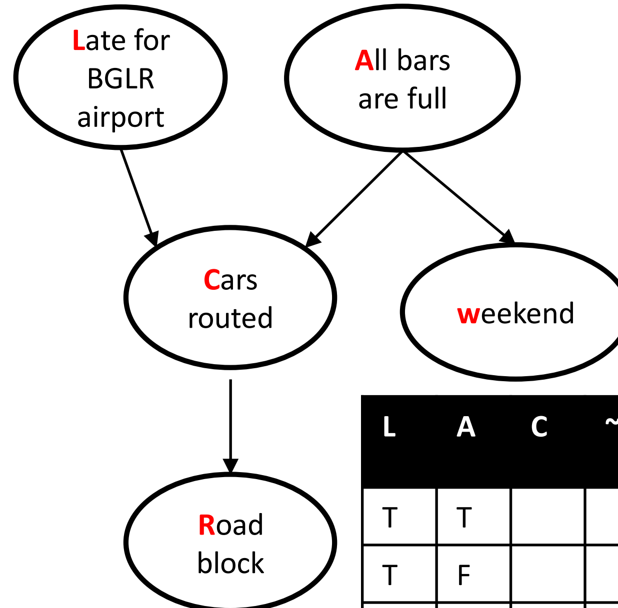
Example Bayesian Nets



R	W	C	~C
T	T		
T	F		
F	T		
F	F		



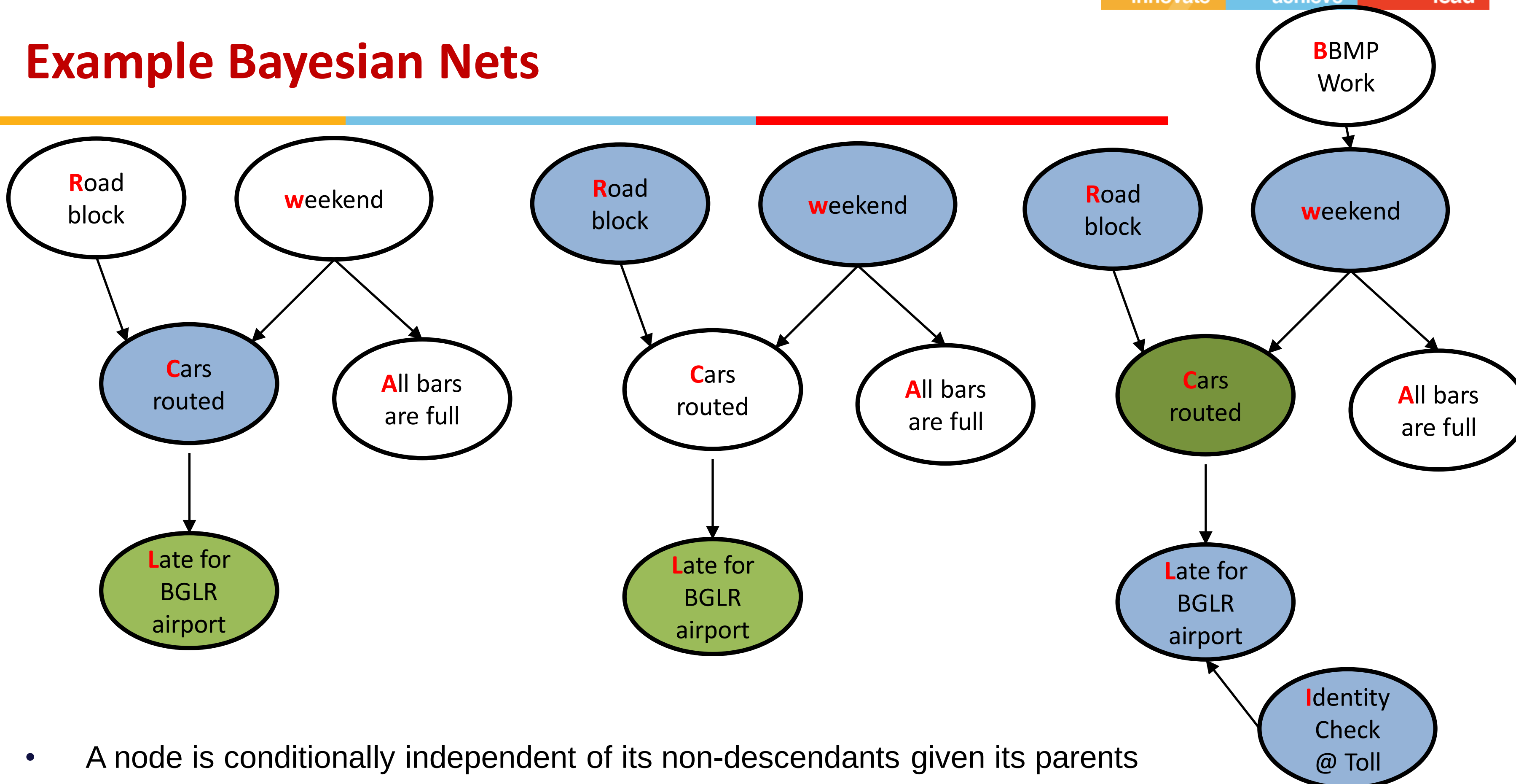
C	L	~L
T		
F		



C	R	~R
T		
F		

L	A	C	~C
T	T		
T	F		
F	T		
F	F		

Example Bayesian Nets

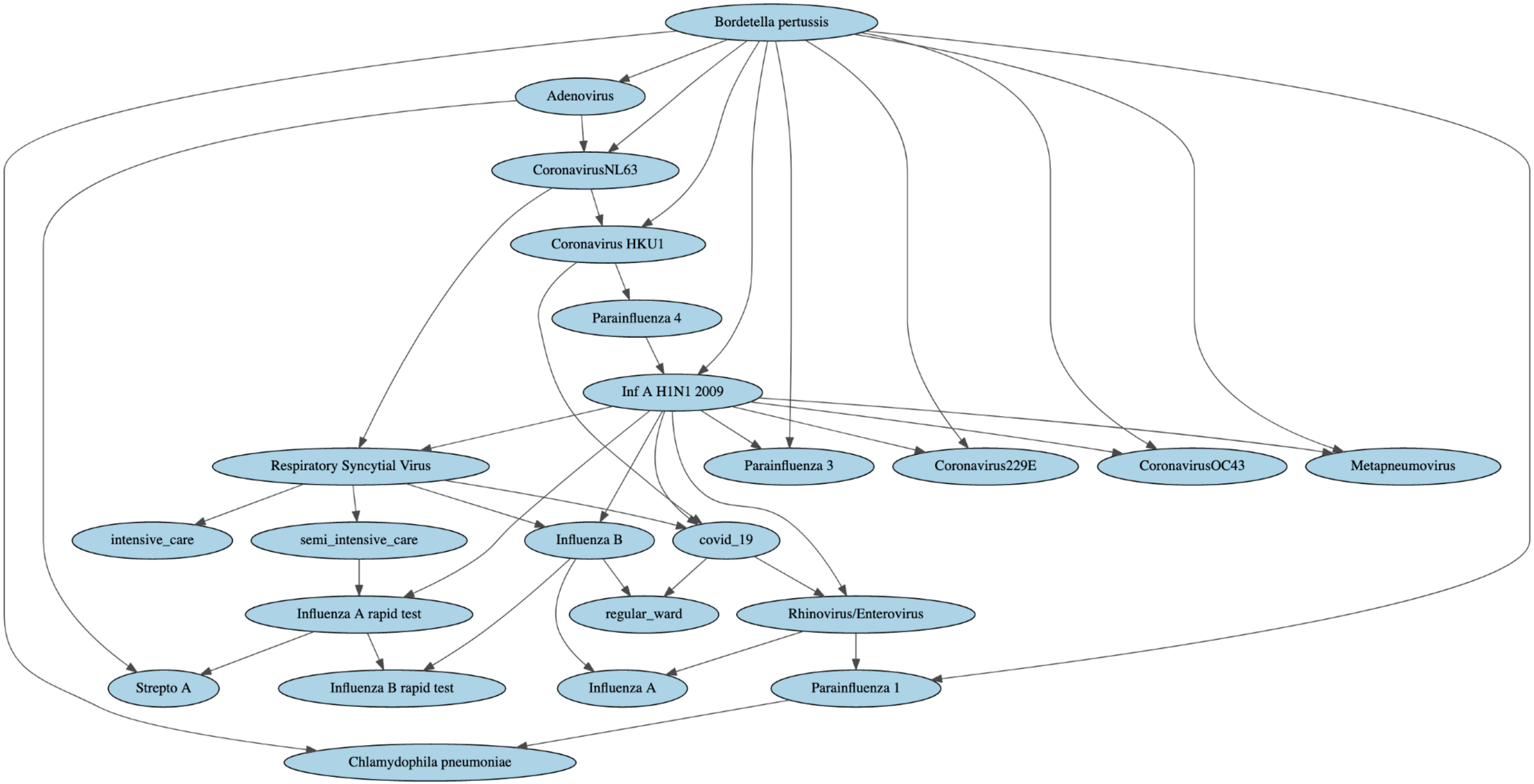


- A node is conditionally independent of its non-descendants given its parents
- A node is conditionally independent of all other nodes in the net , given its parents, children and children's parents.

Bayesian Nets

Interesting Case Study

111-column wide dataset : 6347497291776 entries to store the JPD
817 entries. Memory gain :99.99999998712879%!



Source Credit : https://www.kaggle.com/einsteindata4u/covid19

Bayesian Network

Cyber Security

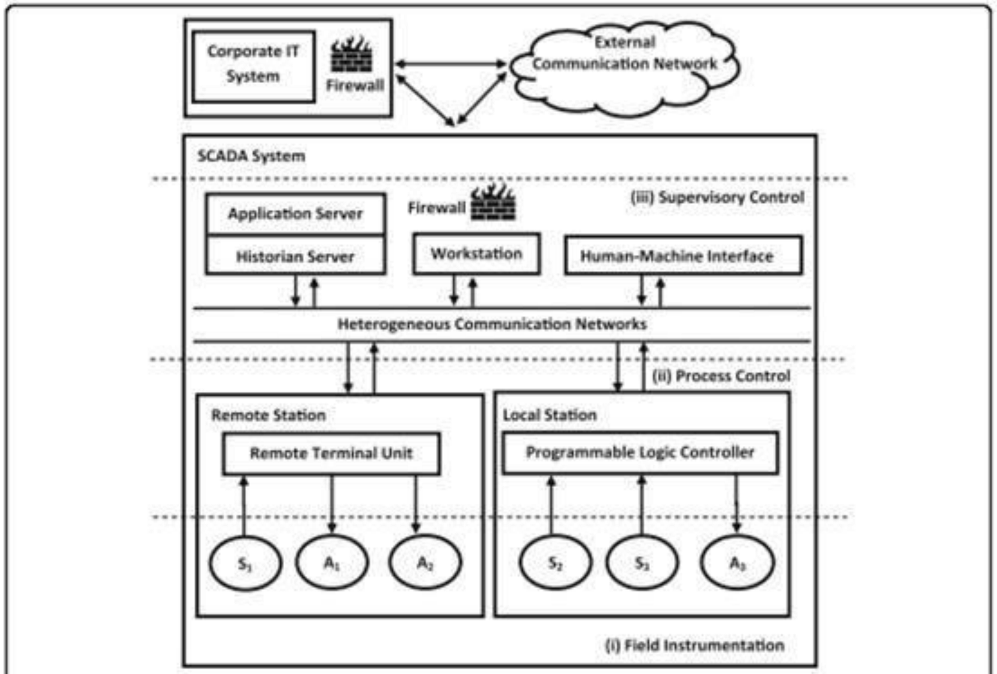


Fig. 1 Typical ICS Architecture and Layers

[Source Credit : 2021 : Chockalingam, S., Pieters, W., Teixeira, A. *et al.* Bayesian network model to distinguish between intentional attacks and accidental technical failures: a case study of floodgates.]

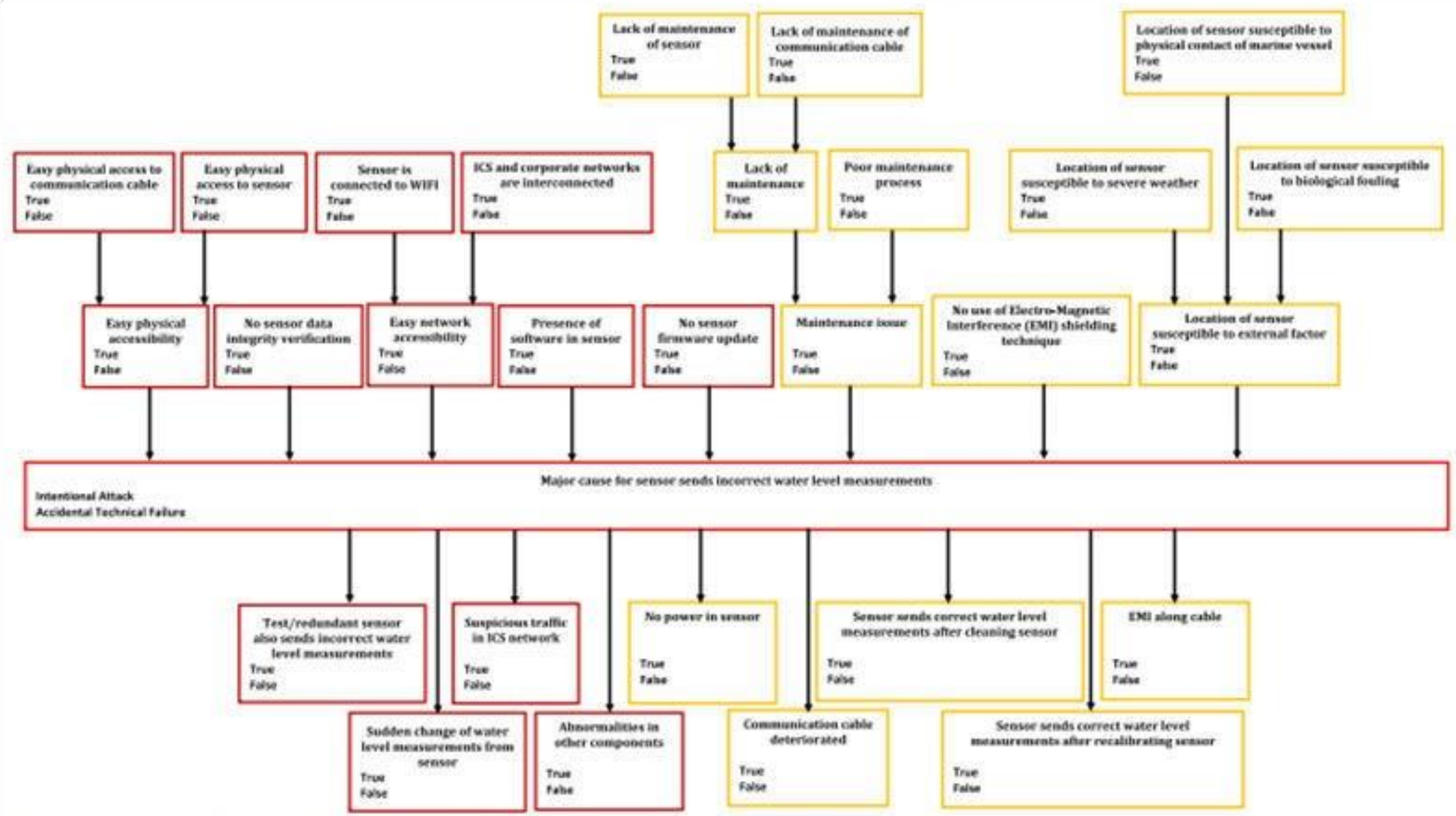


Fig. 4 Constructed Qualitative BN Model. (In this Figure, the presence of contributory factors and observations (or test results) colored in dark red would increase the likelihood of the problem (colored in red) due to an attack on the sensor. Furthermore, the presence of contributory factors and observations (or test results) colored in orange would increase the likelihood of the problem due to sensor failure)

[Source Credit : 2021 : Chockalingam, S., Pieters, W., Teixeira, A. *et al.* Bayesian network model to distinguish between intentional attacks and accidental technical failures: a case study of floodgates.]

Bayesian Network

Cyber Security

Table 2 CPT Excerpt – Problem Variable

C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	Y	
								Attack	Failure
True	True	True	True	True	True	True	True	0.02	0.98
True	True	True	True	True	True	True	False	0.09	0.91
True	True	True	True	True	True	False	True	0.06	0.94
True	True	True	True	True	True	False	False	0.24	0.76
True	True	True	True	True	False	True	True	0.09	0.91
True	True	True	True	True	False	True	False	0.38	0.62
True	True	True	True	True	False	False	True	0.24	0.76
True	True	True	True	True	False	False	False	0.97	0.03
True	True	True	True	False	True	True	True	0.02	0.98
True	True	True	True	False	True	True	False	0.09	0.91

In this table, C_1: Easy physical accessibility, C_2: No sensor data integrity verification, C_3: Easy network accessibility, C_4: Presence of software in sensor, C_5: No sensor firmware update, C_6: Maintenance issue, C_7: No use of EMI shielding technique, C_8: Location of sensor susceptible to external factor and Y: Major cause for sensor sends incorrect water level measurements

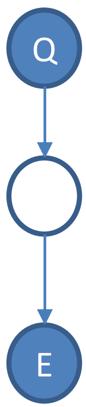
Source Credit : 2021 : Chockalingam, S., Pieters, W., Teixeira, A. *et al.* Bayesian network model to distinguish between intentional attacks and accidental technical failures: a case study of floodgates.

Inferences in Bayesian Nets

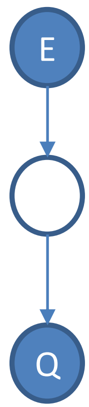
Enumeration

Belief Nets

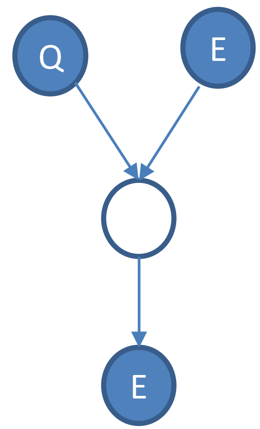
Diagnostic



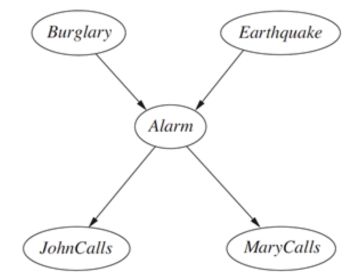
Causal

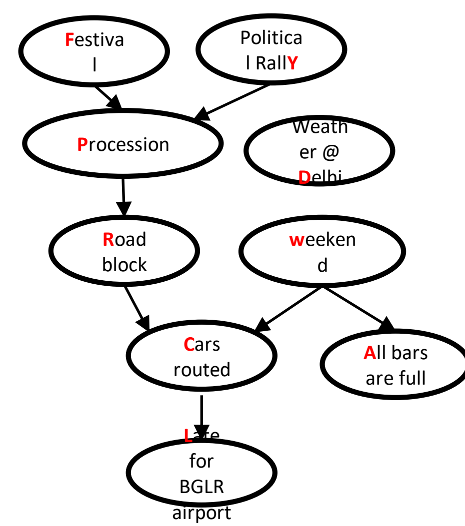


Inter-Casual

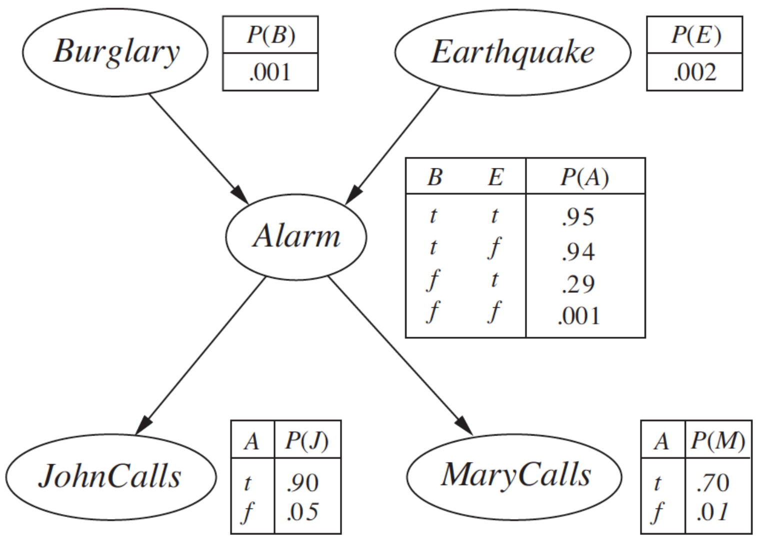


Mixed Inferences





Examples

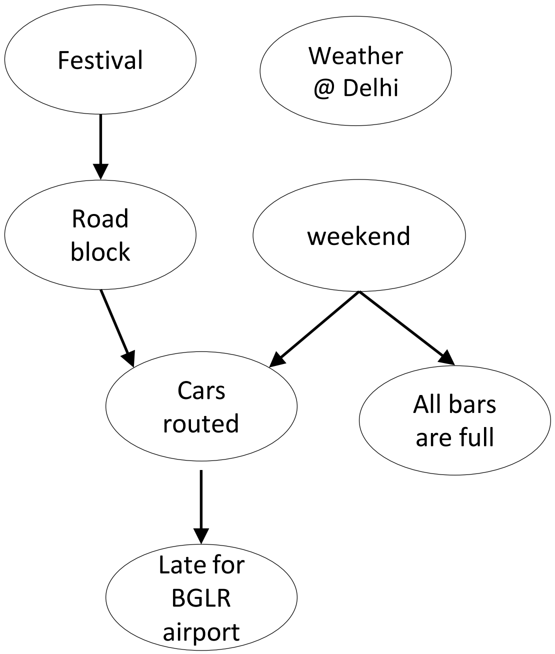


1. Calculate the probability that alarm has sounded, but neither burglary nor earthquake happened, and both John and Mary called

2. What is the probability that Burglary happened given John & Mary called the police

3. What is the probability that John calls given earthquake occurred?

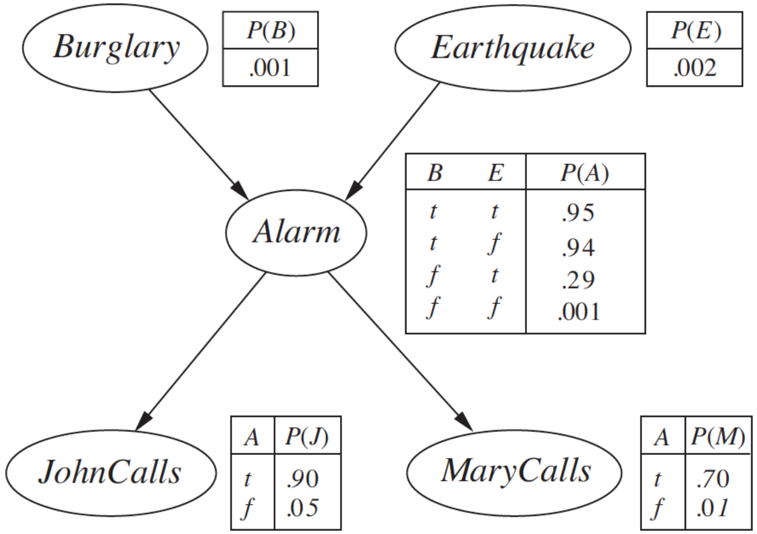
Examples



1. Calculate the probability that arrival at airport was delayed during a weekend but there was no road block or festival and car was not routed anywhere.

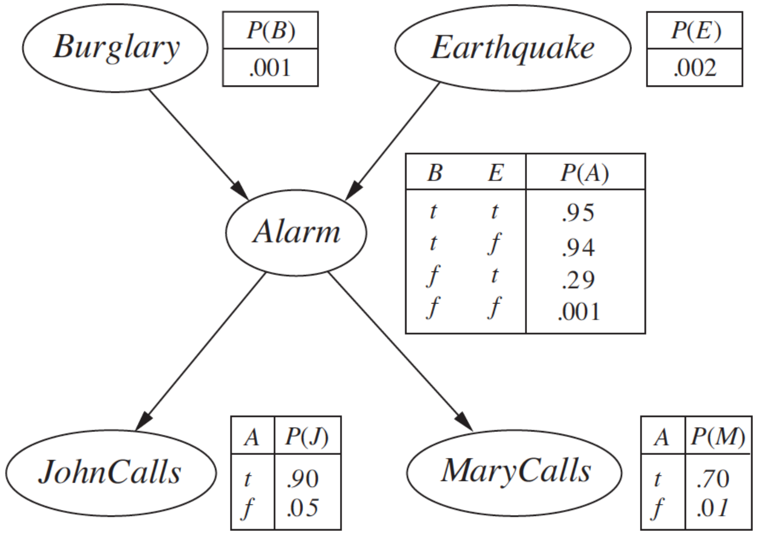
2. What is the probability that it is a festival season given cars where routed?

3. What is the probability that car arrived late at airport given it's a festival day?



1. Calculate the probability that alarm has sounded, but neither burglary nor earthquake happened, and both John and Mary called

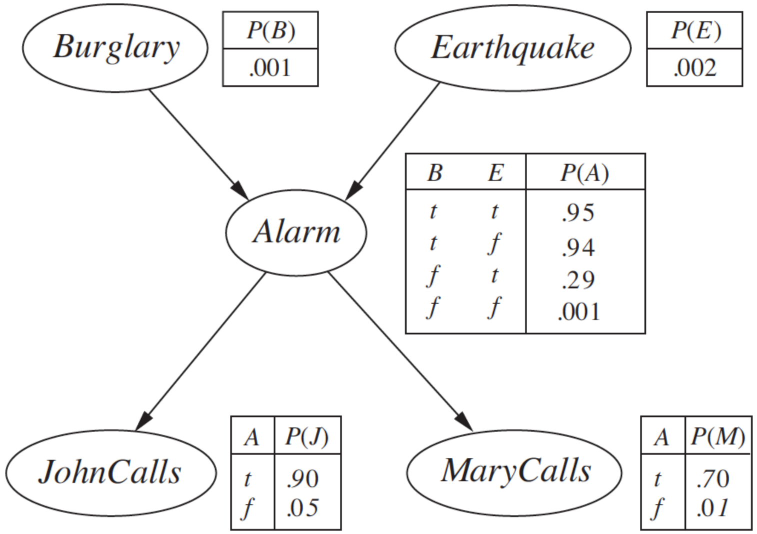
$$
\begin{aligned}
P(j, m, a, \neg b, \neg e) &= P(j \mid a) P(m \mid a) P(a \mid \neg b \wedge \neg e) P(\neg b) P(\neg e) \\
&= 0.90 \times 0.70 \times 0.001 \times 0.999 \times 0.998 = 0.000628
\end{aligned}
$$



2. What is the probability that Burglary happened given John & Mary called the police

$$P(B \mid J,M) = \frac{P(B, J, M)}{P(J, M)}$$

$$P(B \mid J,M) = \frac{\sum_{A,E} P(J, M, A, B, E)}{\sum_{A,B,E} P(J, M, A, B, E)}$$



P(B)
.001

P(E)
.002

B	E	P(A)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

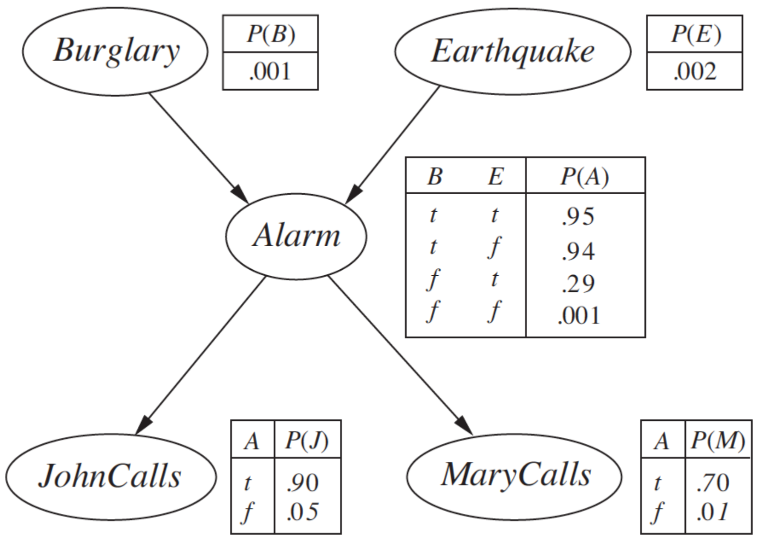
A	P(J)
t	.90
f	.05

A	P(M)
t	.70
f	.01

2. What is the probability that Burglary happened given John & Mary called the police

$$P(B \mid J, M) = \frac{P(B, J, M)}{P(J, M)}$$

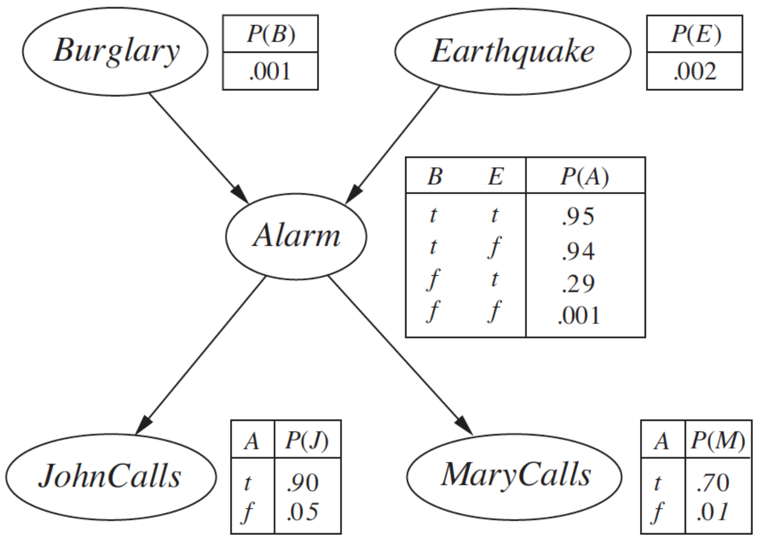
$$P(B \mid J, M) = \frac{\sum_{A, E} P(J, M, A, B, E)}{\sum_{A, B, E} P(J, M, A, B, E)}$$



2. What is the probability that Burglary happened given John & Mary called the police

$$P(B \mid J,M) = \frac{P(B, J, M)}{P(J, M)}$$

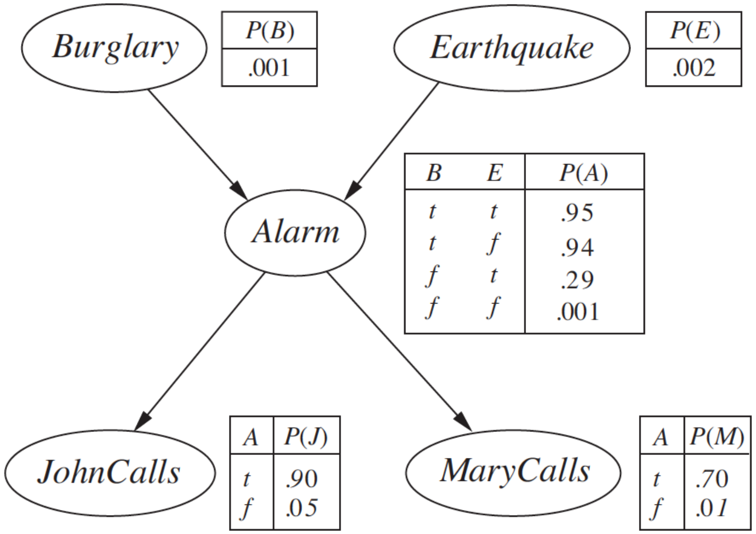
$$P(B \mid J,M) = \frac{\sum_{A, E} P(J, M, A, B, E)}{\sum_{A, B, E} P(J, M, A, B, E)}$$



3. What is the probability that John calls given earthquake occurred?

$$P(J \mid E) = \frac{P(J, E)}{P(E)}$$

$$P(J \mid E) = \frac{\sum_{M, A, B} P(J, M, A, B, E)}{\sum_{J, M, A, B} P(J, M, A, B, E)}$$



3. What is the probability that John calls given earthquake occurred?

$$P(J \mid E) = \frac{P(J, E)}{P(E)}$$

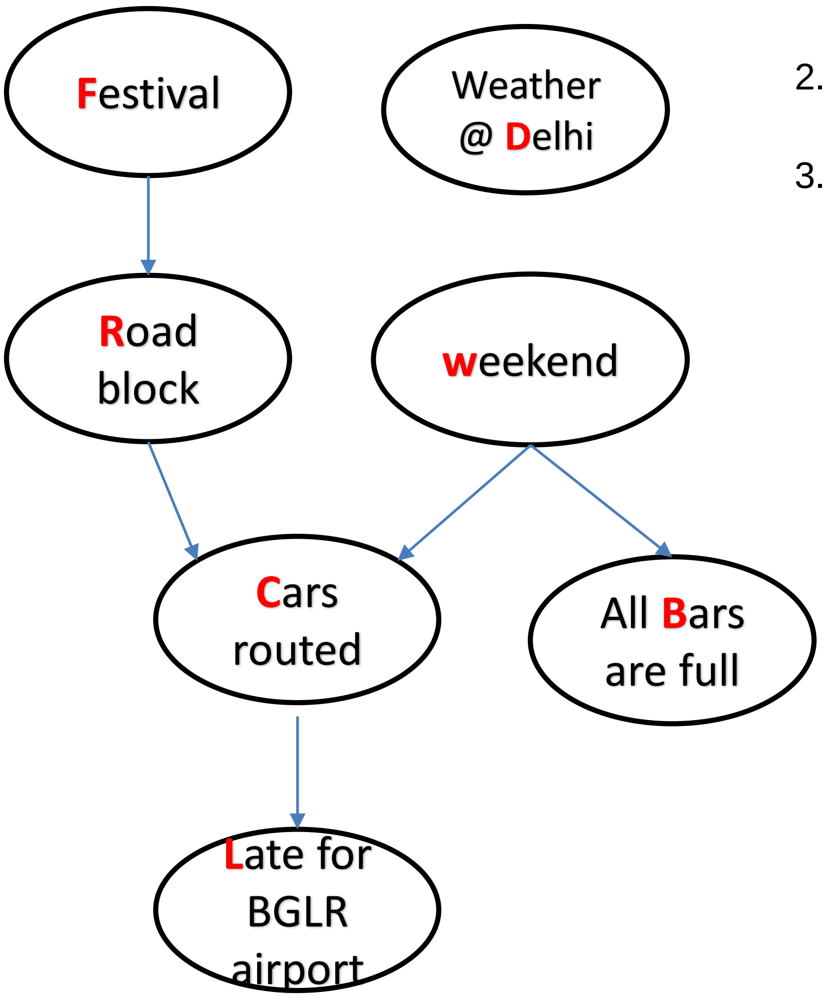
$$P(J \mid E) = \frac{\sum_{M, A, B} P(J, M, A, B, E)}{\sum_{J, M, A, B} P(J, M, A, B, E)}$$

Inferences in Bayesian Nets

Variable Elimination

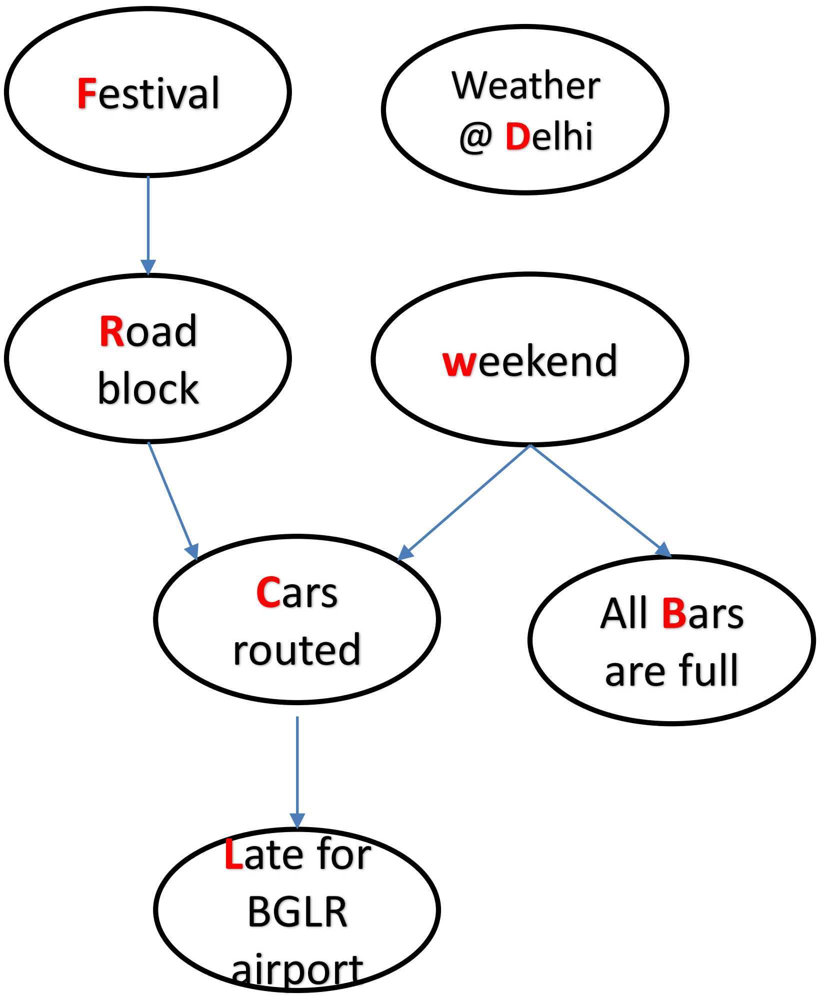
Reduce Guaranteed Independent nodes

D-Connectedness Vs D-Separation



1. Each variable is conditionally independent of its non-descendants, given its parents
2. Eliminate the hidden variables that is neither a query nor an evidence nor the ancestors of {query, eveidences}
3. **Two variables are d-separated if they are conditionally independent given evidences**

D-Separation in Inference



X	Y	Evidence Z	d-sep?
F	W	C	No
L	W	R	No
R	**L**	**C**	**Yes**
B	R	C	No

- P(R | L , C) = P(R| L)

R & L are d-separated ie., conditionally independent given C

Required Reading: AIMA - Chapter # 14.2, #14.3, #14.4, #14.5

Next Session Plan:

- Approximate Inference in Bayesian Network
- Reasoning over Time
- Markov Models

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

Artificial & Computational Intelligence

AIML CLZG557

M5 : Probabilistic Representation and Reasoning

Raja vadhana P

Assistant Professor,

BITS - CSIS

BITS Pilani

Pilani Campus

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

M5 Probabilistic Representation and Reasoning

M6 Reasoning over time

M7 Ethics in AI

Reasoning

Module 5:

Probabilistic Representation and Reasoning

A. Inference using full joint distribution

B. Bayesian Networks

I. Knowledge Representation

II. Conditional Independence

III. Exact Inference

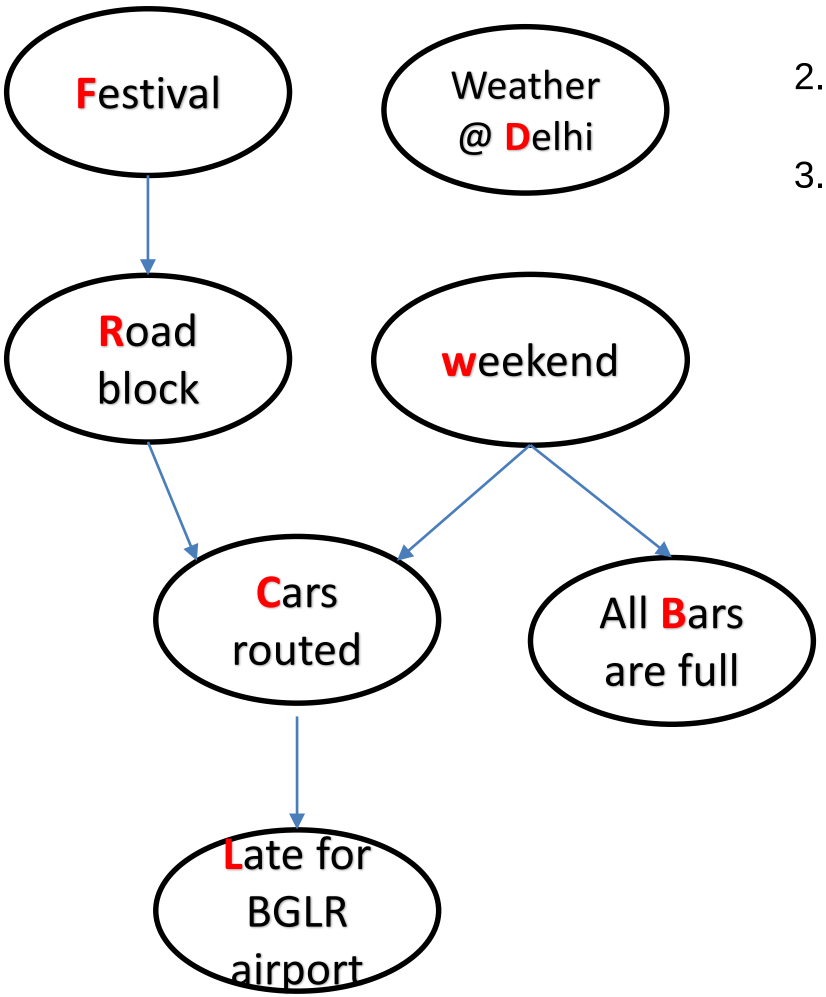
IV. Introduction to Approximate Inference

Inferences in Bayesian Nets

Variable Elimination

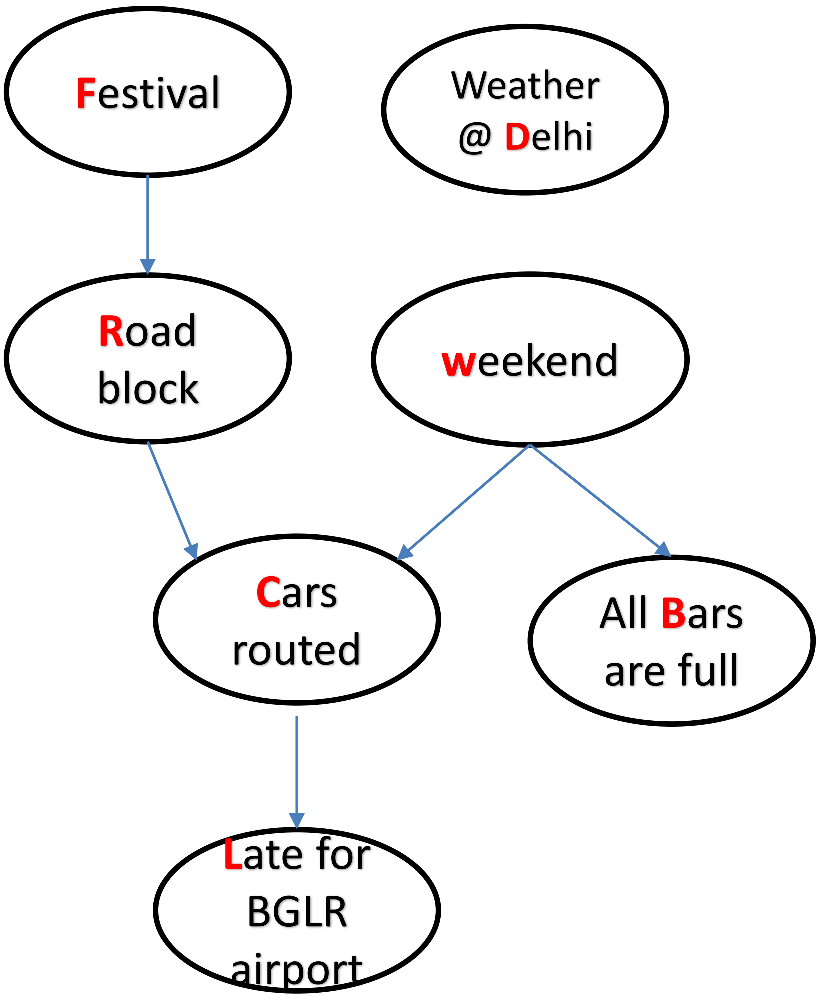
Reduce Guaranteed Independent nodes

D-Connectedness Vs D-Separation



1. Each variable is conditionally independent of its non-descendants, given its parents
2. Eliminate the hidden variables that is neither a query nor an evidence
3. **Two variables are d-separated if they are conditionally independent given evidences**

Try it & Test

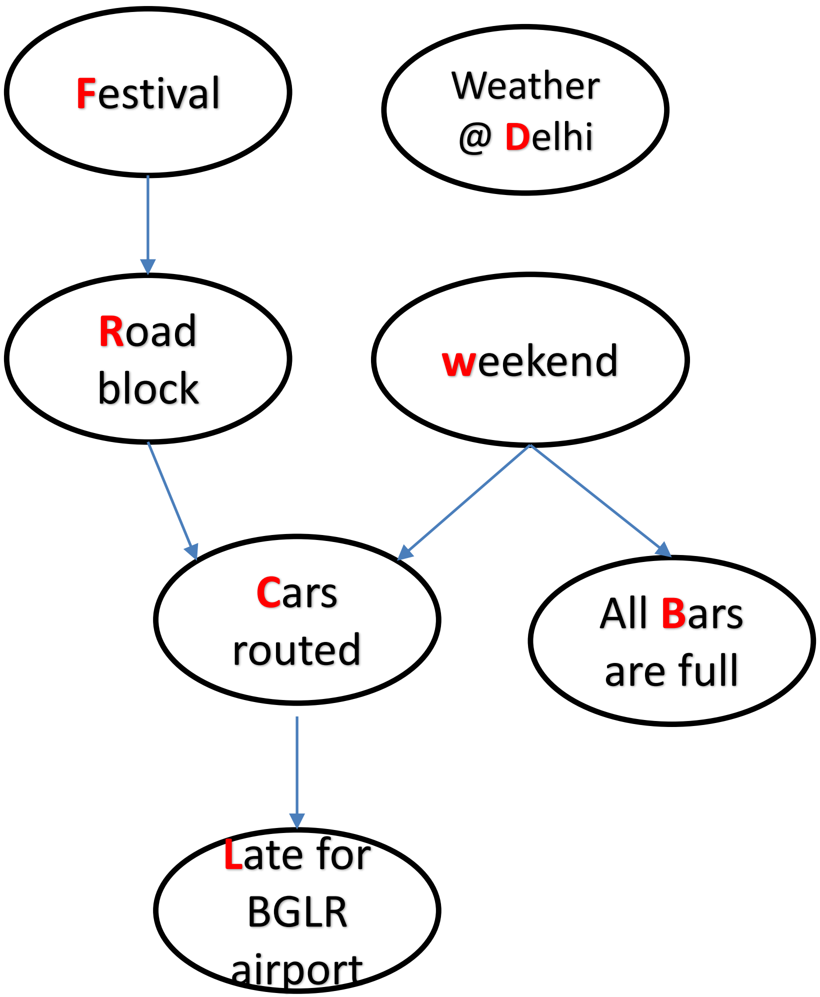


X	Y	Evidence Z	d-sep?
F	W	C	No
L	W	R	No
R	**L**	**C**	**Yes**
B	R	C	No

- P(R | L , C) = P(R| L)

R & L are d-separated ie., conditionally independent given C

D-Separation in Inference



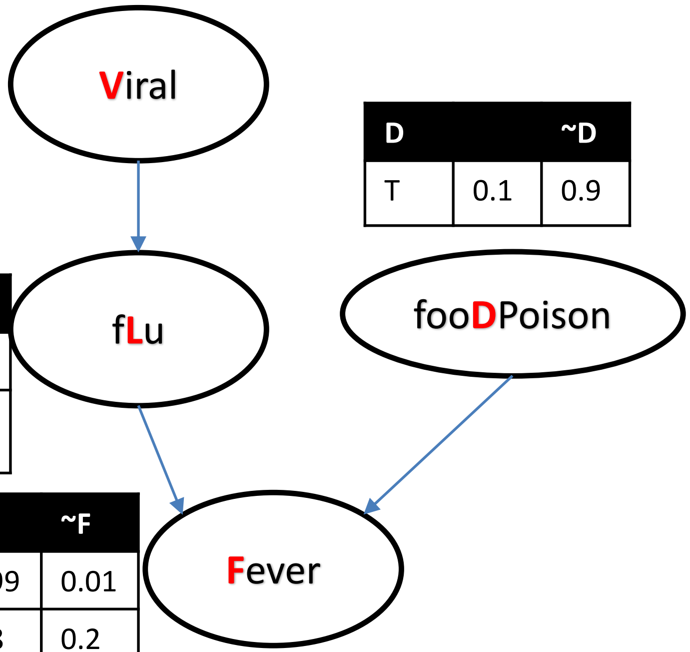
X	Y	Evidence Z	d-sep?
F	W	C	No
L	W	R	No
R	**L**	**C**	**Yes**
B	R	C	No

- P(R | L , C) = P(R| L)

R & L are d-separated ie., conditionally independent given C

D-Separation in Inference

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

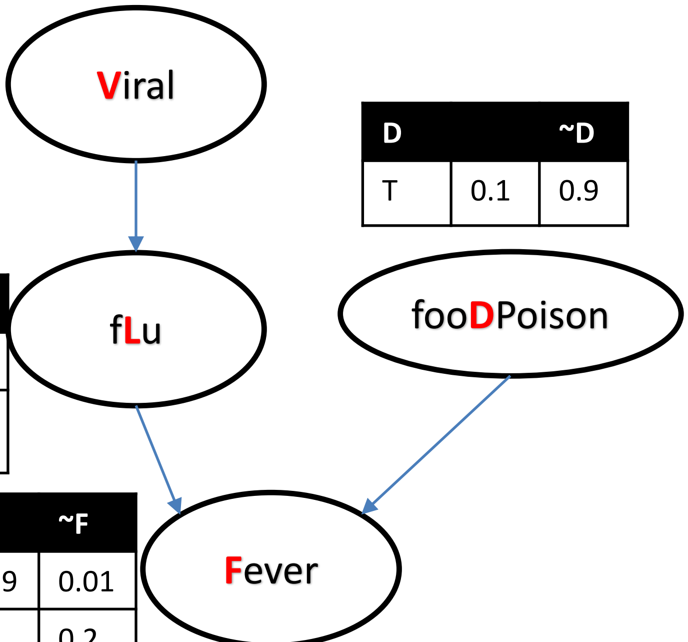
X	Y	Evidence Z	d-sep?
V	F	L	Yes
V	D	L	Yes

- P(V | F , L)
- P(V | D , L)

D-Separation in Inference

Enumeration

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

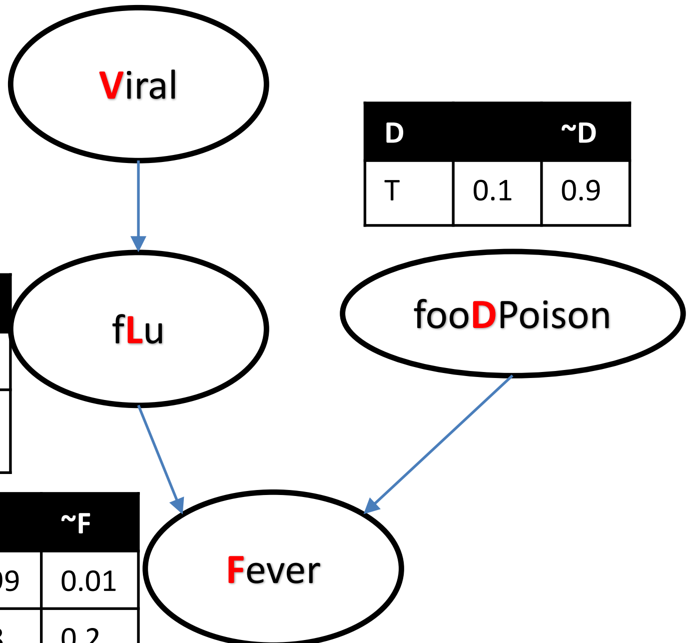
X	Y	Evidence Z	d-sep?
V	F	L	Yes
V	D	L	Yes

- P(V \| F , L)
- P(V \| D , L)

D-Separation in Inference

Variable Elimination

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

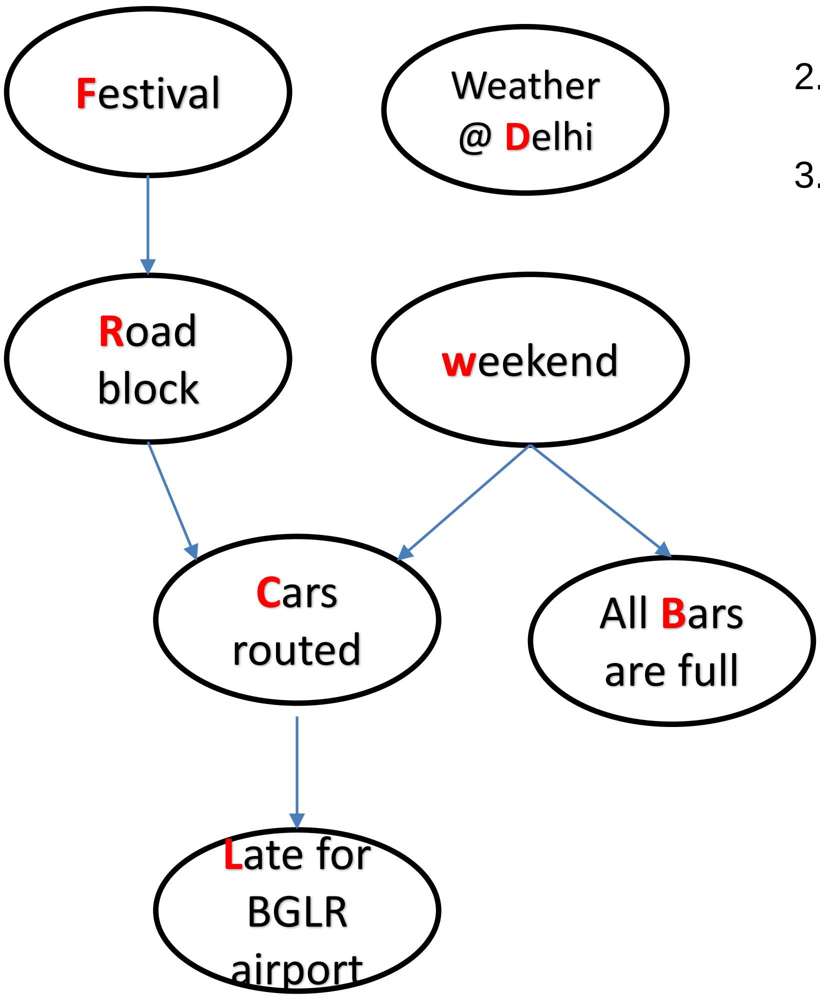
V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

X	Y	Evidence Z	d-sep?
V	F	L	Yes
V	D	L	Yes

- P(V | F , L)
- P(V | D , L)

Variable Elimination



1. Each variable is conditionally independent of its non-descendants, given its parents
2. **Eliminate the hidden variables that is neither a query nor evidence**
3. Two variables are d-separated if they are conditionally independent given evidences

- $P(B) = \sum_{L, B, W, R, F} P(L, C, B, W, R, F)$
  $= \sum_{L} \sum_{B} P(L|C) \, . \, P(B \mid W) \, . \, \sum_{W} P(C| W, R). \sum_{R} P(R \mid F) \, . \, \sum_{F} P(F)$
  $= \mathbf{P(B \mid W)}$

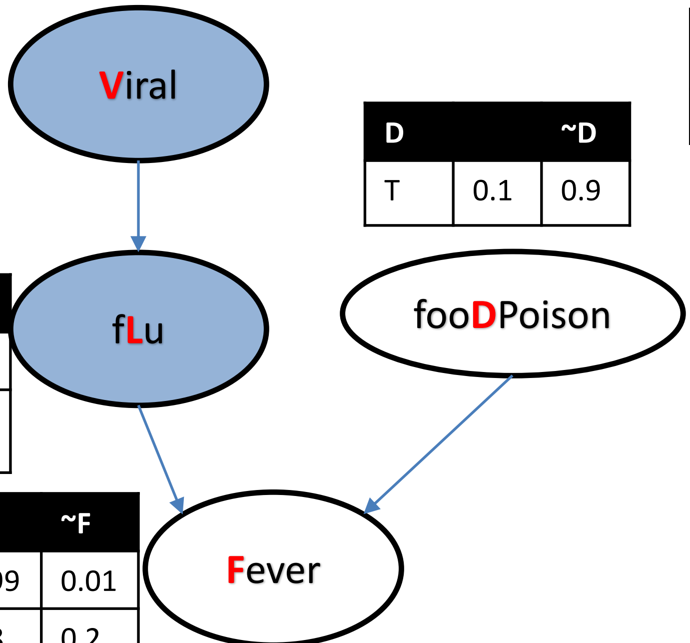
All other variables are hidden w.r.t to B as (L, C, R, F) are neither evidence nor query nor (L, C, R, F) ∈ Ancestors(W, B)

This is variable elimination example targeting irrelevant nodes

Inference

Variable Elimination: V

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V		~V
T	0.6	0.4

V	L	~L
T	0.8	0.2
F	0.01	0.99

V	L	
T	T	0.48
T	F	0.12
F	T	0.004
F	F	0.396

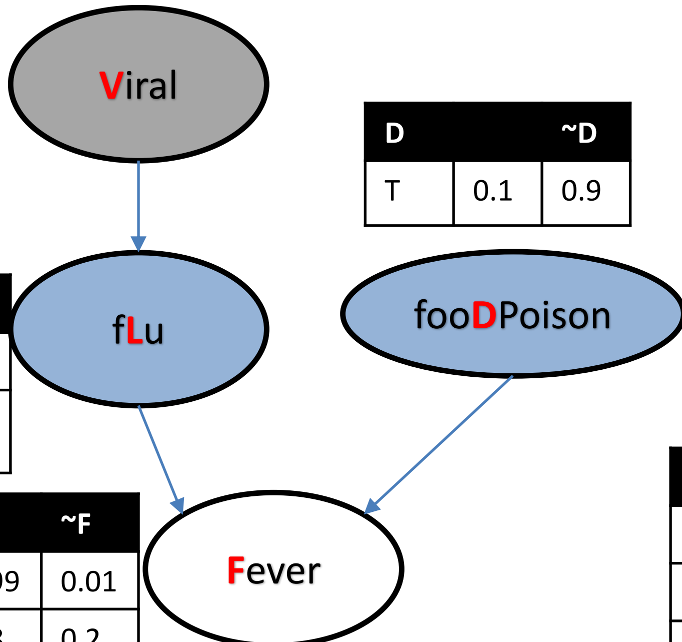
L	
T	0.484
F	0.516

P(V)
P(L|V)
P(D)
P(F|L,D)

Inference

Variable Elimination: L,D

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

P(L)
P(D)
P(F|L,D)

L	
T	0.484
F	0.516

D		~D
T	0.1	0.9

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

D	L	F	
T	T	T	0.048
T	F	T	0.34852
F	T	T	0.04128
F	F	T	0.02322
T	T	F	0.00048
T	F	F	0.087
F	T	F	0.01032
F	F	F	0.44118

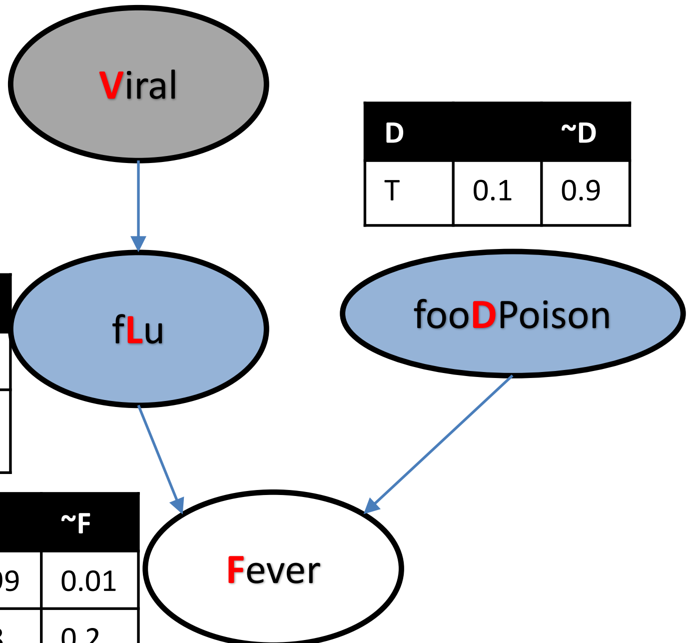
L	F	
T	T	0.08928
F	T	0.37174
T	F	0.0108
F	F	0.52818

F	
T	0.46102
F	0.53898

Inference

Variable Elimination: L

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

L	
T	0.484
F	0.516

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

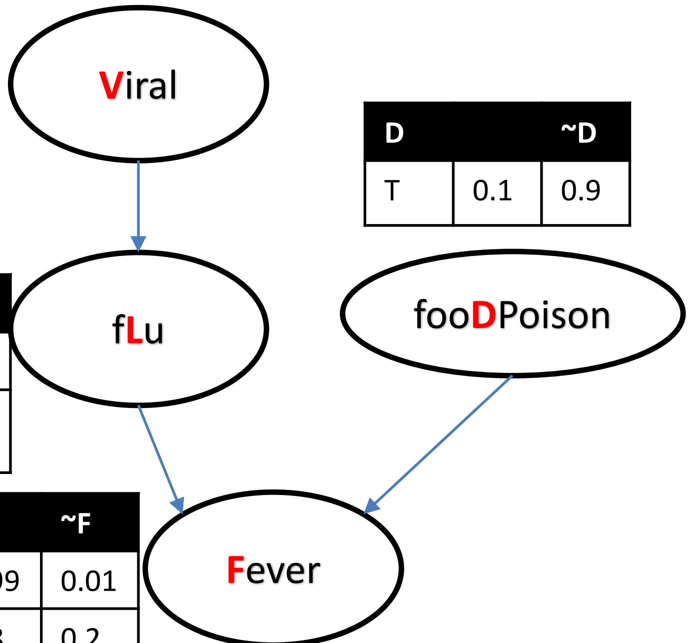
P(L)
P(D)
P(F|L,D)

Approximate Inferences in Bayesian Nets
Introduction

Prior Sampling

Sample Generation by Randomization

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

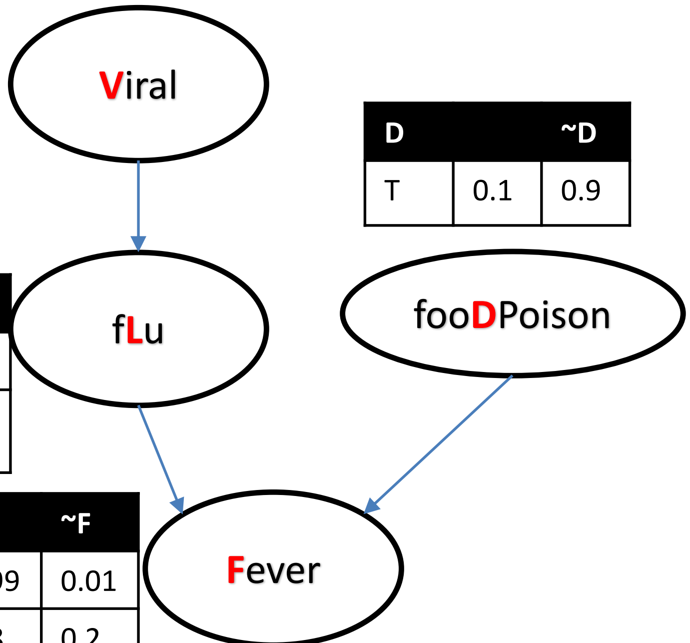
V	L	D	F

0.3, 0.2, 0.6, 0.58, 0.73, 0.87, 0.15, 0.6, 0.57, 0.85, 0.12, 0.004, 0.93 , 0.0002, 0.9,0.55…………..

Prior Sampling

Sample Generation by Randomization

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

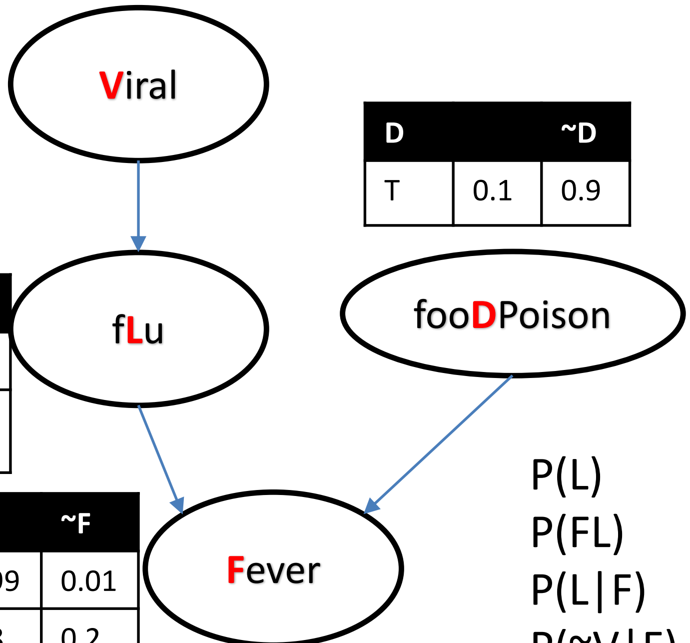
V	L	D	F
T	T	F	T
F	F	F	F
T	F	F	T
F	T	F	T
..			
…..			

0.3, 0.2, 0.6, 0.58, 0.73, 0.87, 0.15, 0.6, 0.57, 0.85, 0.12, 0.004, 0.93 , 0.0002, 0.9,0.55…………..

Prior Sampling

Inference

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	D	F
T	T	F	T
F	F	F	F
T	F	F	T
F	T	F	T
T	T	F	T
T	F	F	F
F	F	F	T
T	F	F	F

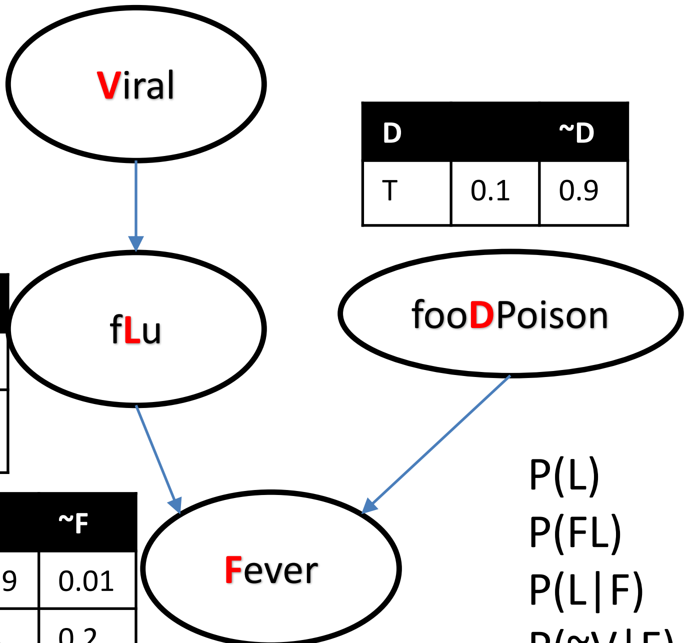
P(L) = 3/8
P(FL) = 3/8
P(L|F) = 3/5
P(~V|F) = 2/5
P(L|V~F) = 0
P(F|D) = ?????

Rejection Sampling

Sample Generation by Randomization

V		~V
T	0.6	0.4

D		~D
T	0.1	0.9



V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	D	F
…..			

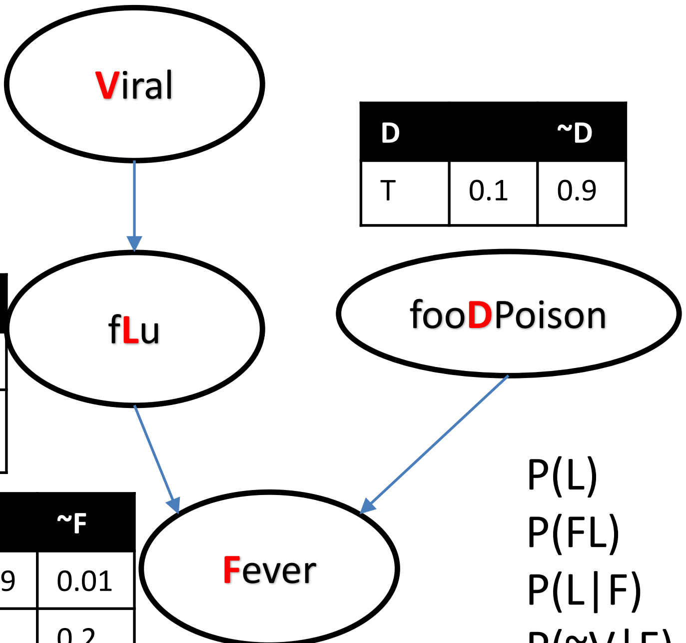
P(L) = 3/8
P(FL) = 3/8
P(L|F) = 3/5
P(~V|F) = 2/5
P(L|V~F) = 0
P(F|D) = ?????

0.3, 0.2, 0.6, 0.58, 0.73, 0.87, 0.15, 0.6, 0.57, 0.85, 0.12, 0.004, 0.93 , 0.0002, 0.9,0.555,0.38…………..

Rejection Sampling

Sample Generation by Randomization

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	D	F
T	T	F	
T	T	F	
T	T	F	
F	F	T	F
T	F	T	T
…..			

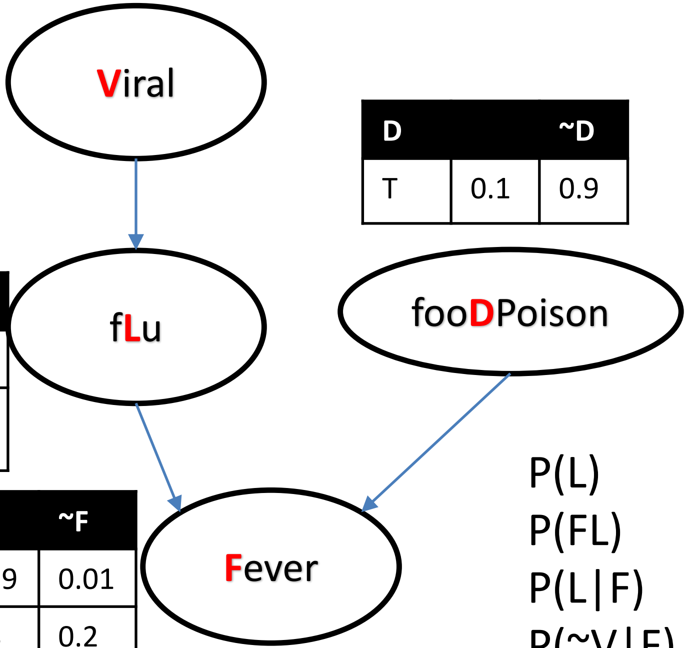
$P(L) = 3/8$
$P(FL) = 3/8$
$P(L|F) = 3/5$
$P(\sim V|F) = 2/5$
$P(L|V\sim F) = 0$
$P(F|D)$ = ?????

0.3, 0.2, 0.6, 0.58, 0.73, 0.87, 0.15, 0.6, 0.57, 0.85, 0.12, 0.004, 0.93 , 0.0002, 0.9,0.555,0.38…………..

Rejection Sampling

Inference

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

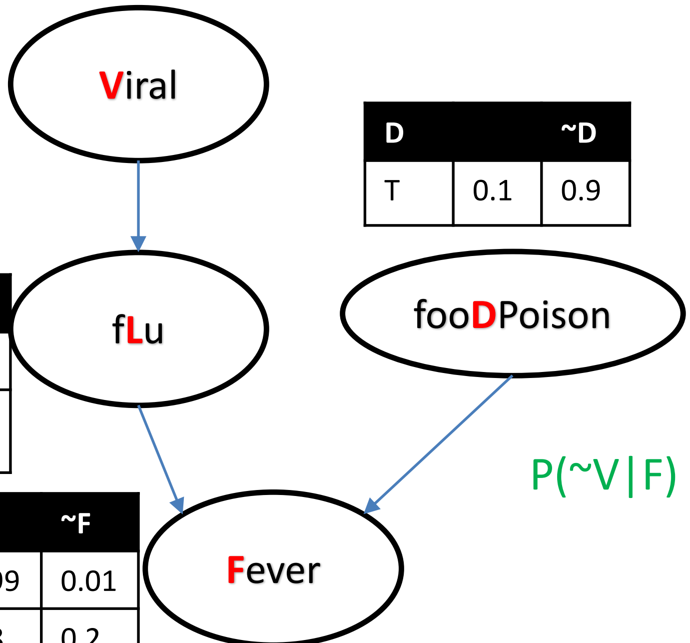
$P(L) = 3/8$
$P(FL) = 3/8$
$P(L|F) = 3/5$
$P(\sim V|F) = 2/5$
$P(L|V\sim F) = 0$
$P(F|D) = 5/8$

V	L	D	F
T	T	T	T
F	F	T	F
T	F	T	T
F	T	T	T
T	T	T	T
T	F	T	F
F	F	T	T
T	F	T	F

Rejection Sampling

Sample Generation by Randomization

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	D	F
T	T	F	T
F	F	F	
T	F	F	T
F	T	F	T
…..			

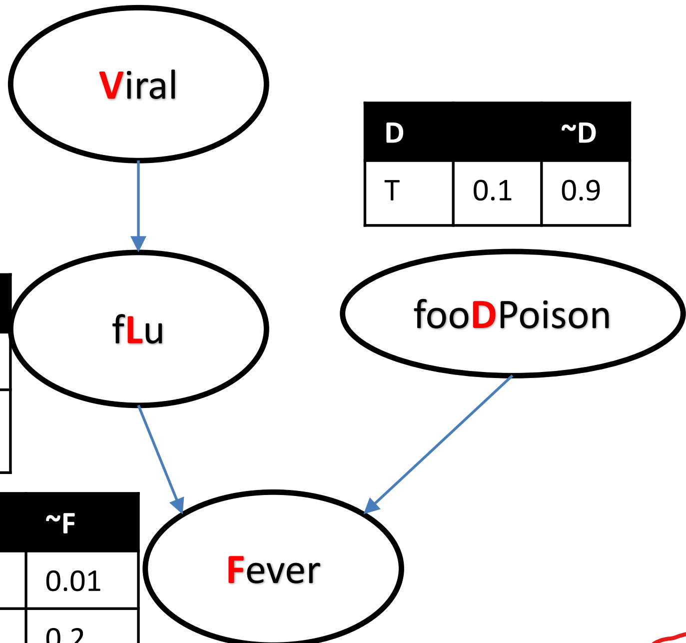
$P(\sim V \mid F) \quad = ?$

0.3, 0.2, 0.6, 0.58, 0.73, 0.87, 0.15, 0.6, 0.57, 0.85, 0.12, 0.004, 0.93 , 0.0002, 0.9,0.555,……….

Likelihood Weighing

Sample Generation by Randomization

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	D	F	wgt
F		T		
F		T		
F		T		
F		T		
F		T		
F		T		
F		T		

Based on the query V and D filled out. The follow the prior approach.

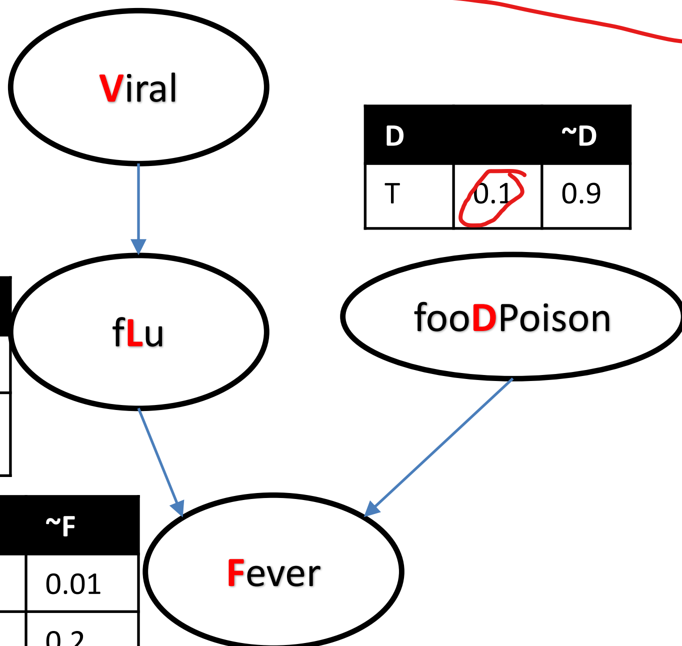
P(~F|D,~V) = 0.04 / 7*0.04

0.3, 0.2, 0.58, 0.73, 0.87, 0.15, 0.6, 0.57, 0.85, 0.12, 0.004, 0.93 , 0.0002, 0.99,,..........

Likelihood Weighing

Sample Generation by Randomization

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	D	F	wgt
F	F	T	T	0.4*1* 0.1 *1=
F	F	T	T	
F	F	T	T	
F	F	T	T	
F	F	T	T	
F	T	T	T	
F	T	T	F	

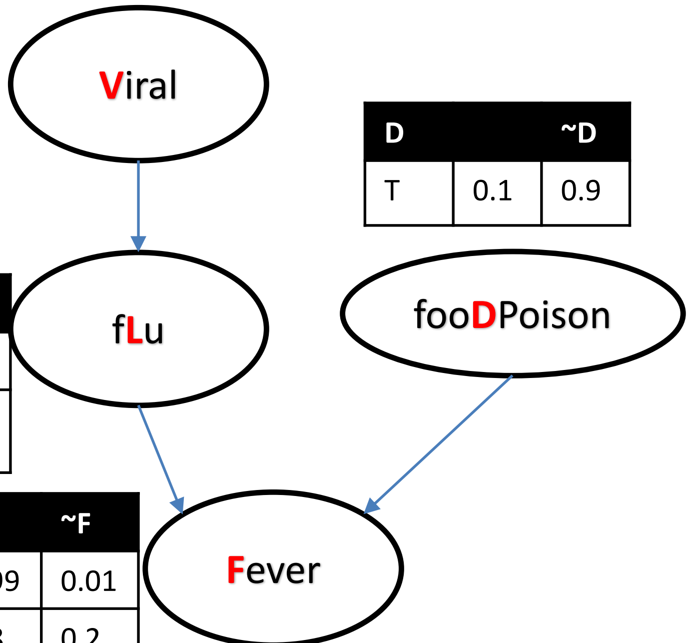
P(~F|D,~V) = 0.04 / 7*0.04

0.3, 0.2, 0.58, 0.73, 0.87, 0.15, 0.6, 0.57, 0.85, 0.12, 0.004, 0.93 , 0.0002, 0.99,,……….

Likelihood Weighing

Inference

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

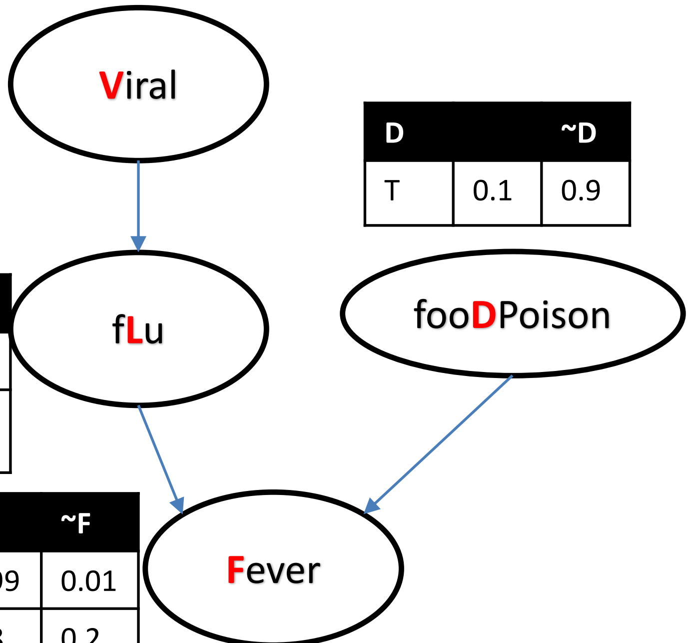
V	L	D	F	wgt
F	F	T	F	0.4*1* 0.1 *1=
F	T	T	T	0.4*1* 0.1 *1=
F	F	T	T	0.4*1* 0.1 *1=
F	F	T	F	0.4*1* 0.1 *1=

P(F|D,~V) = 0.04+0.04 / 4*0.04

Likelihood Weighing

Inference

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

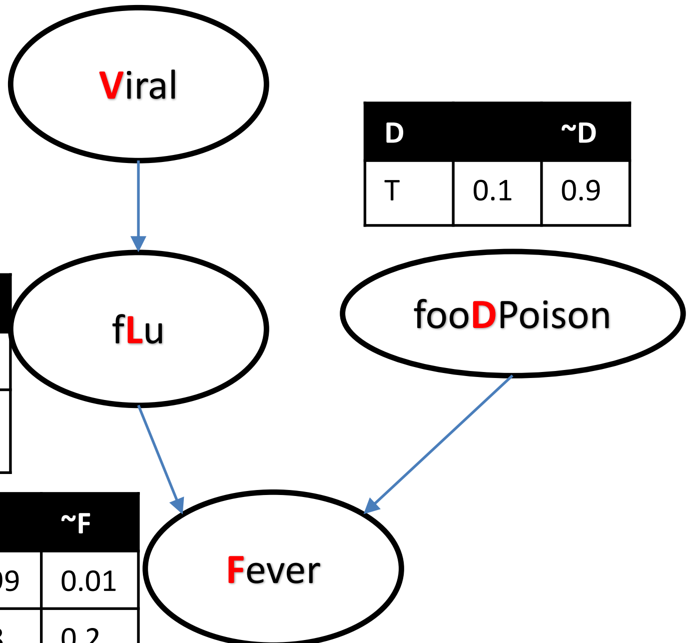
V	L	D	F	wgt
F	F	T	F	
F	F	T	T	
F	F	T	T	
T	F	T	F	

P(F|D,~L) = 0.099+0.099 / (3*0.099 + 0.02)

Likelihood Weighing

Inference

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

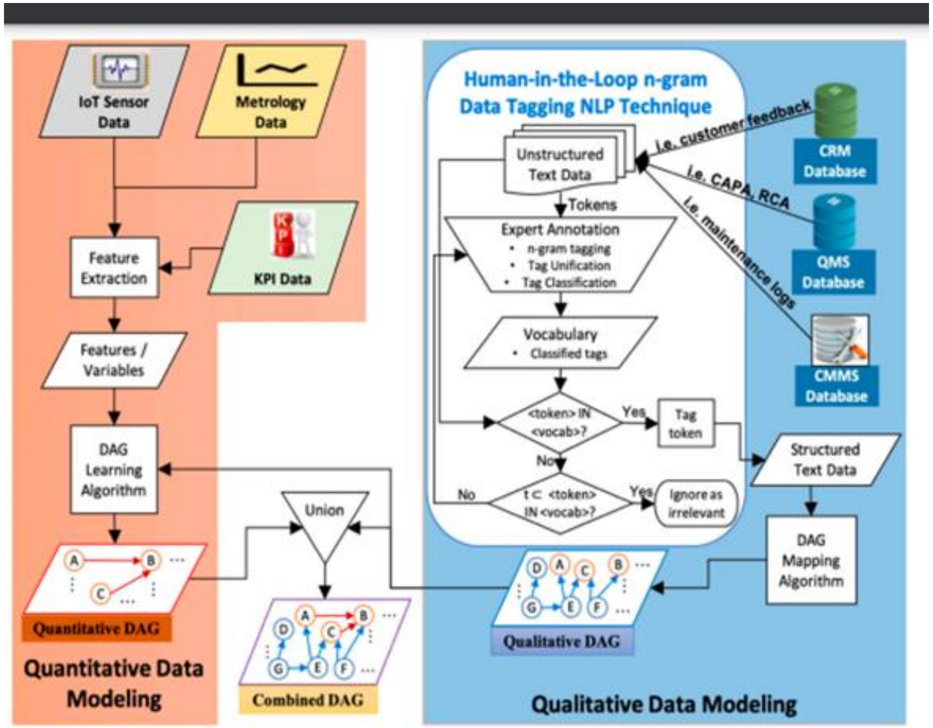
L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	D	F	wgt
F	F	T	F	1*0.99* 0.1 *1=
F	F	T	T	1*0.99* 0.1 *1=
F	F	T	T	1*0.99* 0.1 *1=
T	F	T	F	1*0.2* 0.1 *1=

P(F|D,~L) = 0.099+0.099 / (3*0.099 + 0.02)

Bayesian Network

Fault Diagnostic System

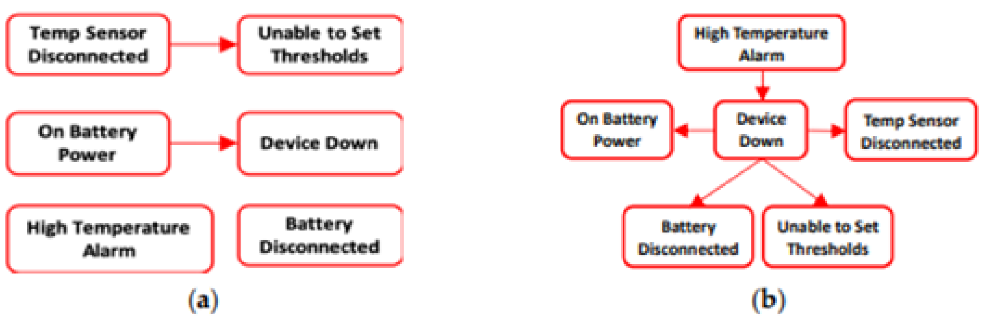


Source Credit : [Sensors 2021 : Fusion-Learning of Bayesian Network Models for Fault Diagnostics]

Bayesian Network

Fault Diagnostic System

Raw Data	Short Description		Resolution Notes			
	On battery power		Power outage due to transformer fire			
Classified Tags	Symptom		Cause(s)		Link	
	on_battery_power		power_outage, transformer_fire		due_to	
BN Mapping	Child Variable	Child State	Parent Variable	Parent State	Ancestor Variable	Ancestor State
	on_battery_power	yes	power_outage	yes	transformer	Fire



Source Credit : [Sensors 2021 : Fusion-Learning of Bayesian Network Models for Fault Diagnostics]

Bayesian Network

Fault Diagnostic System

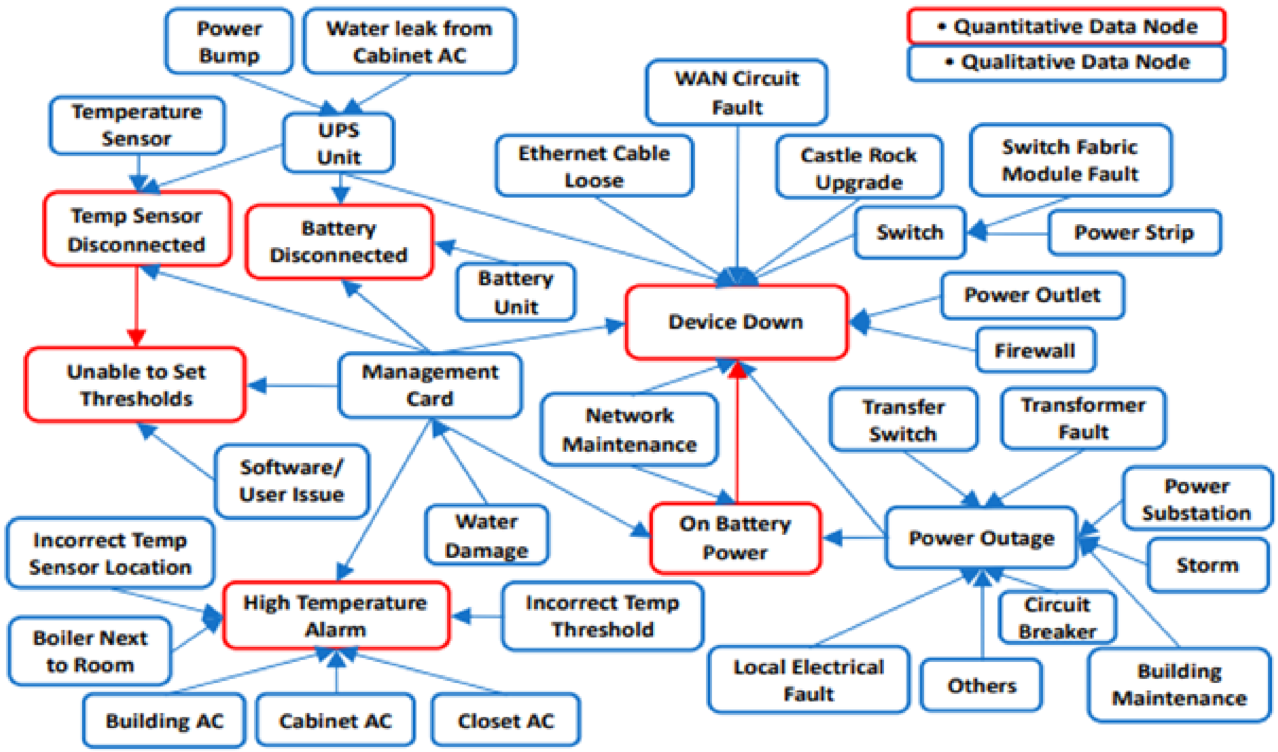
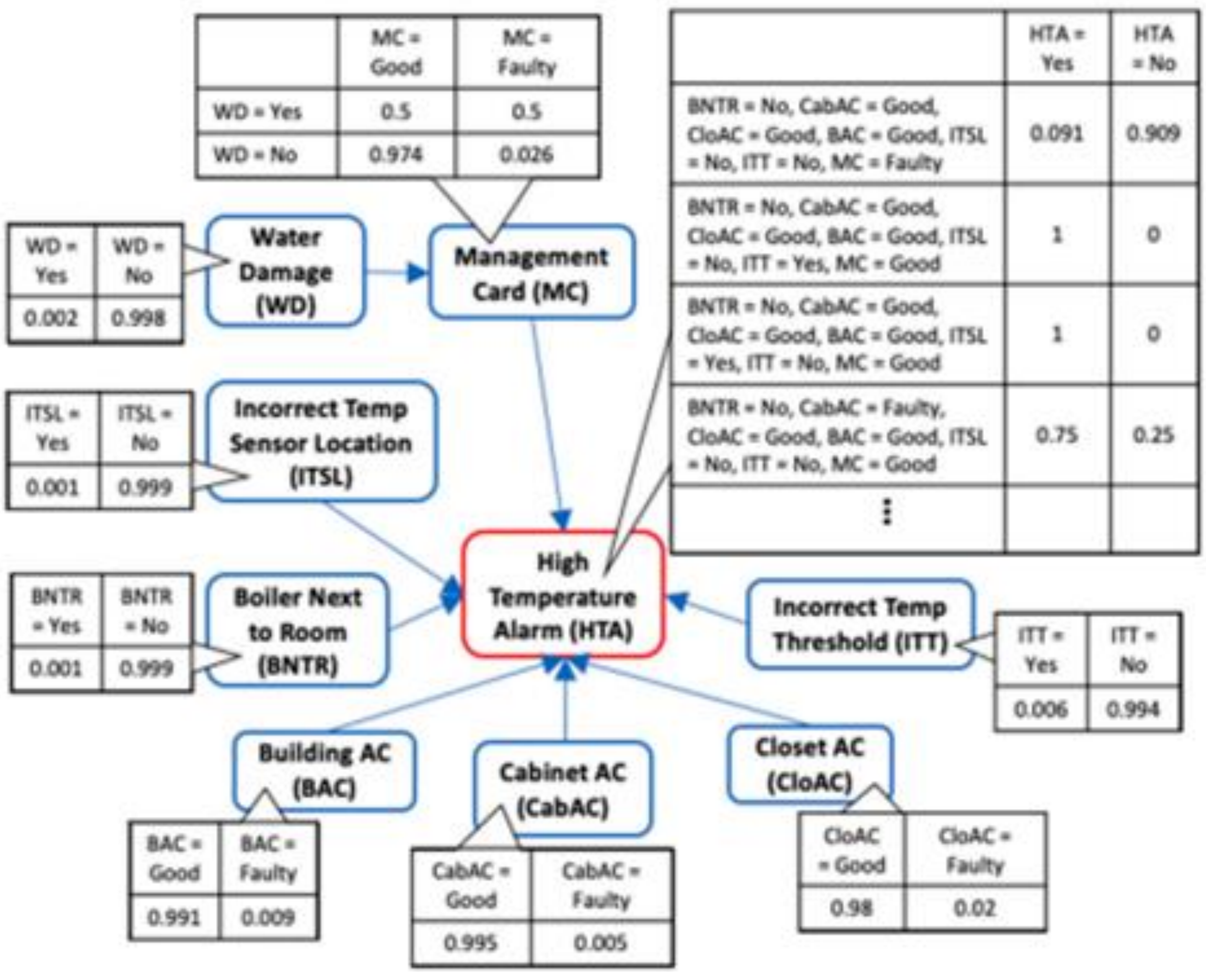


Figure 8. Fused Bayesian Network structure for top six occurring UPS messages.

Source Credit : [Sensors 2021 : Fusion-Learning of Bayesian Network Models for Fault Diagnostics]

Bayesian Network

Fault Diagnostic System



Source Credit : [Sensors 2021 : Fusion-Learning of Bayesian Network Models for Fault Diagnostics]

Module 6:
Reasoning over time

Reasoning Over Time

A. Time and Uncertainty

B. Inference in temporal models

C. Introduction to Hidden Markov Model

D. Applications of HMM

Time & Uncertainty

Learning Objective

1. Understand the relationship between Time & Uncertainty

1. Recognize the transition model of Markov Model

1. Relate to the application of the Hidden Markov Model

Sequential Decision Problems & Markov Decision Process

Markov Decision Process

Sequential Problem | Partial Observability | Belief System

Modelling sequences of random events and transitions between states over time is known as Markov chain

Agents in partially observable environment should keep a track of current state to the extent allowed by sensors

E.g., Robot moving in a new maze

Agent maintains a **belief state** representing the current possible world states

Transition Model / Probability Matrix :
Using belief state and transition model,
the agent can how the world might evolve in
next time step. To capture the degree of belief
we will use Probability Theory. We model the change in world
using a variable for each aspect of state and
at each point in time.
Current state depends only finite number of previous states.

$$\begin{array}{cc}
\begin{array}{cc} \mathbf{C} & \mathbf{M} \end{array} & \\
\begin{vmatrix} 0.40 & 0.20 \\ 0.60 & 0.80 \end{vmatrix} & \begin{array}{c} \mathbf{C} \\ \mathbf{M} \end{array}
\end{array}$$

Markov Decision Process

Time - Uncertainty | States - Observations

Static World: Each random variable would have a single fixed value

E.g., Diagnosing a broken car

Dynamic World: The state information keeps changing with time

E.g., treating a diabetic patient, tracking the location of robot, tracking economic activity of a nation

Time slices: World is observed in time slices. Each slice has a set of random variables, some observable and some not.

Assumption: We will assume same subset of random variables are observable in each time slice

$\mathbf{E_t}$ – set of observable random variables at time t

$\mathbf{X_t}$ – set of unobserved random variables at time t

$$\begin{array}{cc} \begin{array}{cc} \mathbf{C} & \mathbf{M} \end{array} & \\ \left| \begin{array}{cc} 0.40 & 0.20 \\ 0.60 & 0.80 \end{array} \right| & \begin{array}{c} \mathbf{C} \\ \mathbf{M} \end{array} \end{array}$$

Markov Process

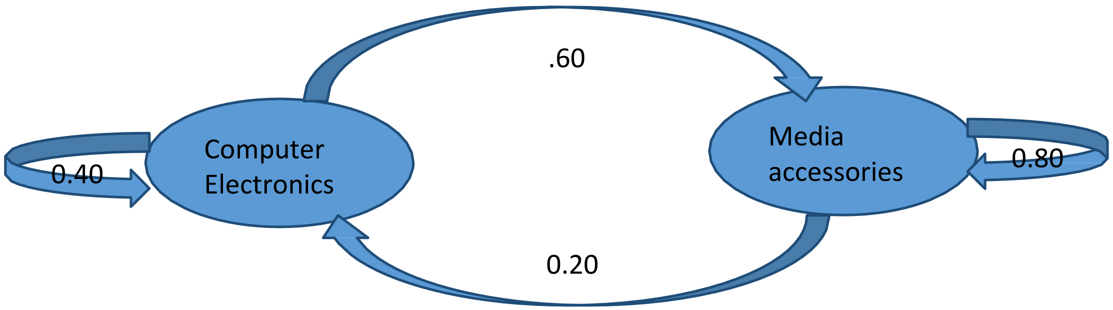
States | Observations | Assumptions

Modelling sequences of random events and transitions between states over time is known as Markov chain

Transition Model / Probability Matrix :
Current state depends only finite number of previous states. :

$$\begin{array}{ccc} C & M & \\ \left|\begin{matrix} 0.40 & 0.20 \\ 0.60 & 0.80 \end{matrix}\right| & & \begin{matrix} C \\ M \end{matrix} \end{array}$$

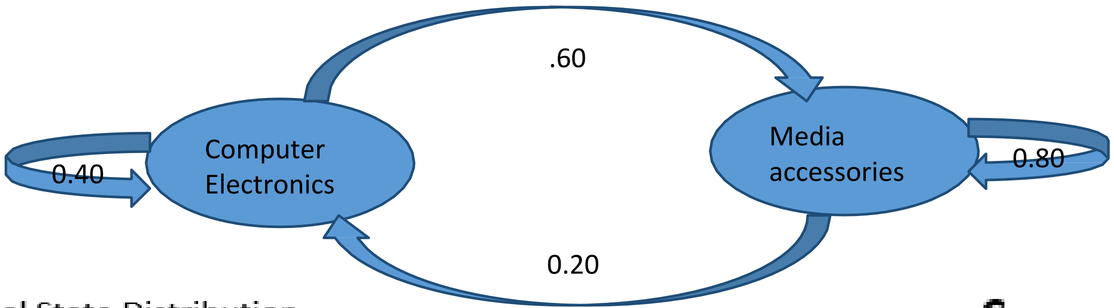
Markov Model- Example 1



Transition Model

$$\begin{array}{ccc} C & M & \\ \begin{array}{|c} 0.40 \\ 0.60 \end{array} & \begin{array}{c|} 0.20 \\ 0.80 \end{array} & \begin{array}{c} C \\ M \end{array} \end{array}$$

Markov Model



Current State: Initial State Distribution

$$\begin{array}{c|l} \begin{vmatrix} 1 \\ 0 \end{vmatrix} & \begin{array}{l} \mathbf{C} \\ \mathbf{M} \end{array} \end{array}$$

$$\begin{array}{cc|l} \mathbf{C} & \mathbf{M} & \\ \hline \multicolumn{2}{c}{\begin{vmatrix} 0.40 & 0.20 \\ 0.60 & 0.80 \end{vmatrix}} & \begin{array}{l} \mathbf{C} \\ \mathbf{M} \end{array} \end{array}$$

Next State : Likely to buy Media accessories on next visit

$$\begin{array}{c|l} \begin{vmatrix} 0.40 \\ 0.60 \end{vmatrix} & \begin{array}{l} \mathbf{C} \\ \mathbf{M} \end{array} \end{array}$$

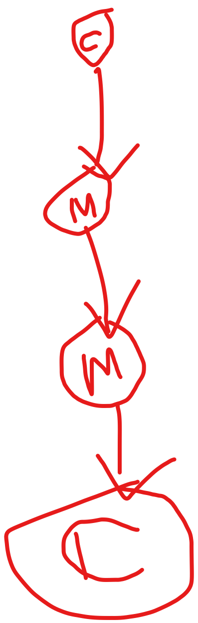
Next State : Likely to buy Media accessories on next visit

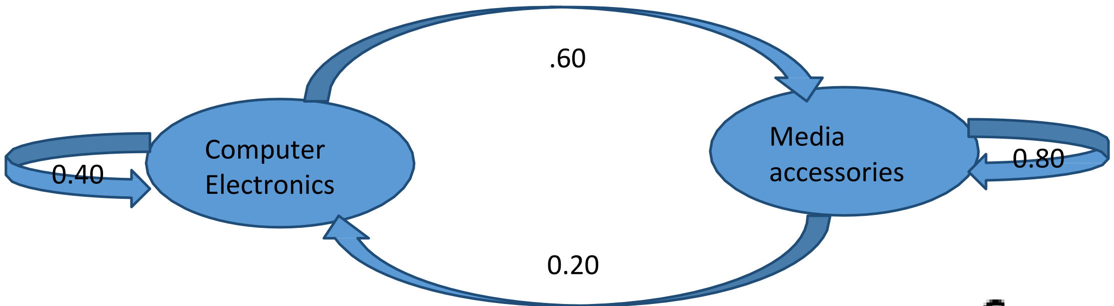
$$\begin{array}{c|l} \begin{vmatrix} 0.28 \\ 0.72 \end{vmatrix} & \begin{array}{l} \mathbf{C} \\ \mathbf{M} \end{array} \end{array}$$

Inference in temporal Models

Markov Model

Inference Type 1





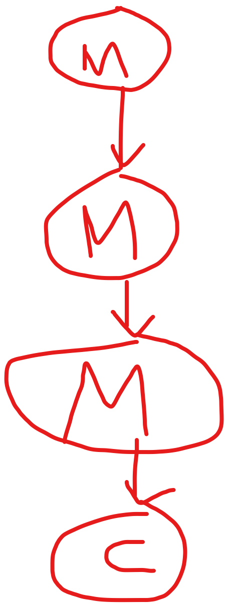
$$
\begin{array}{ccc}
C & M & \\
\left|\begin{matrix} 0.40 & 0.20 \\ 0.60 & 0.80 \end{matrix}\right| & & \begin{matrix} C \\ M \end{matrix}
\end{array}
$$

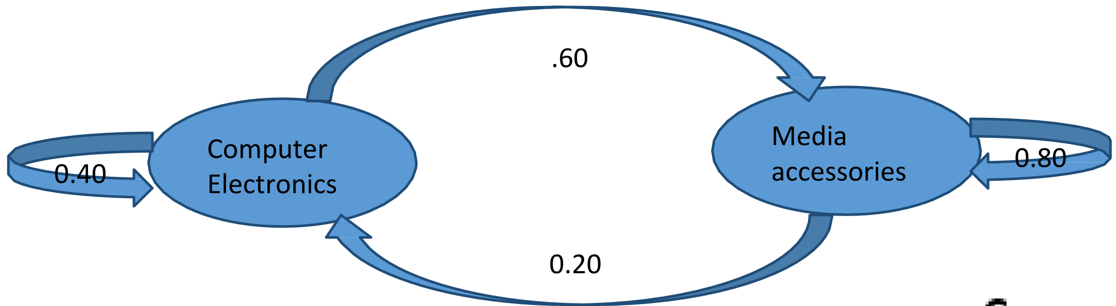
What is the probability that the purchasing behaviour of the customer is in below sequential order only?Initial Probability Matrix is P(C) = 1 , P(M) = 0
(Computer, Media, Media, Computer)

Apply Bayes chain rule:
P(Computer, Media, Media, Computer) = P(C) * P(M|C) * P(M|M) * P(C|M) = 0.096

Markov Model

Inference Type 2





	C	M	
	0.40	0.20	**C**
	0.60	0.80	**M**

What is the probability that the customer who purchased Media accessories will keep coming back to purchase media accessories in the next 2 consecutive visits only?

<u>Derive Initial prob values & Apply Bayes chain rule on the pattern exhibited:</u>

Initial Probability Matrix is P(M) = 1 , P(C) = 0

P(Media, Media, Media, Computer) = P(M) * P(M|M) * P(M|M) * P(C|M) = 0.128

Markov Model

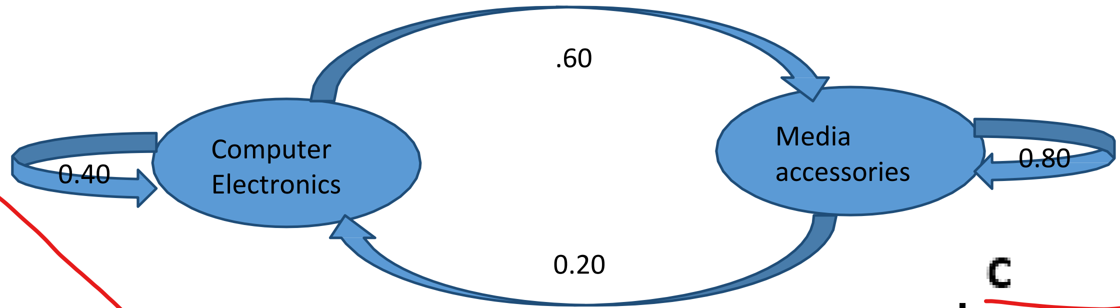
Inference Type 3

C

? M

? M

? M



C < M

	C	M	
	0.40	0.20	C
	0.60	0.80	M

Given the evidence that the customer walked into the store and bought a computer electronics, find the expected purchase pattern in the next 3 visits

<u>Derive Initial prob values & Apply Bayes chain rule and reverse predict the combination on the most likely pattern (Similar to Viterbi Algorithm):</u>

Initial Probability Matrix is P(C) = 1 , P(M) = 0

P(Computer , X, Y, Z) = P(Computer) * P(X|Computer) * P(Y|X) * P(Z|X) =

1*0.6*0.8*0.8 → Produces max values

Ans : Pattern = (Computer, Media, Media, Media)

Required Reading: AIMA - Chapter # 4.1, #4.2, #5.1, #9 Refer to the handout

Next Session Plan:

- Hidden Markov Models

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

Artificial & Computational Intelligence

AIML CLZG557

M6: Reasoning over time

Raja vadhana P
Assistant Professor,
BITS - CSIS

BITS Pilani
Pilani Campus

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

M5 Probabilistic Representation and Reasoning

M6 Reasoning over time

M7 Ethics in AI

Module 6:
Reasoning over time

innovate achieve lead

Reasoning Over Time

A. Time and Uncertainty

B. Inference in temporal models

C. Overview of HMM

D. Learning HMM Parameters using EM Algorithm

E. Applications of HMM

Reasoning Over Time

Learning Objective

1. Understand the relationship between Time & Uncertainty

1. Recognize the transition model of Markov Model

1. Relate to the application of the Hidden Markov Model

Markov Process

States | Observations | Assumptions

Modelling sequences of random events and transitions between states over time is known as Morkov chain

Transition Model / Probability Matrix :
Current state depends only finite number of previous states. :

$$
\begin{array}{cc}
\begin{array}{cc} \mathbf{C} & \mathbf{M} \end{array} & \\
\left|\begin{array}{cc} 0.40 & 0.20 \\ 0.60 & 0.80 \end{array}\right| & \begin{array}{c} \mathbf{C} \\ \mathbf{M} \end{array}
\end{array}
$$

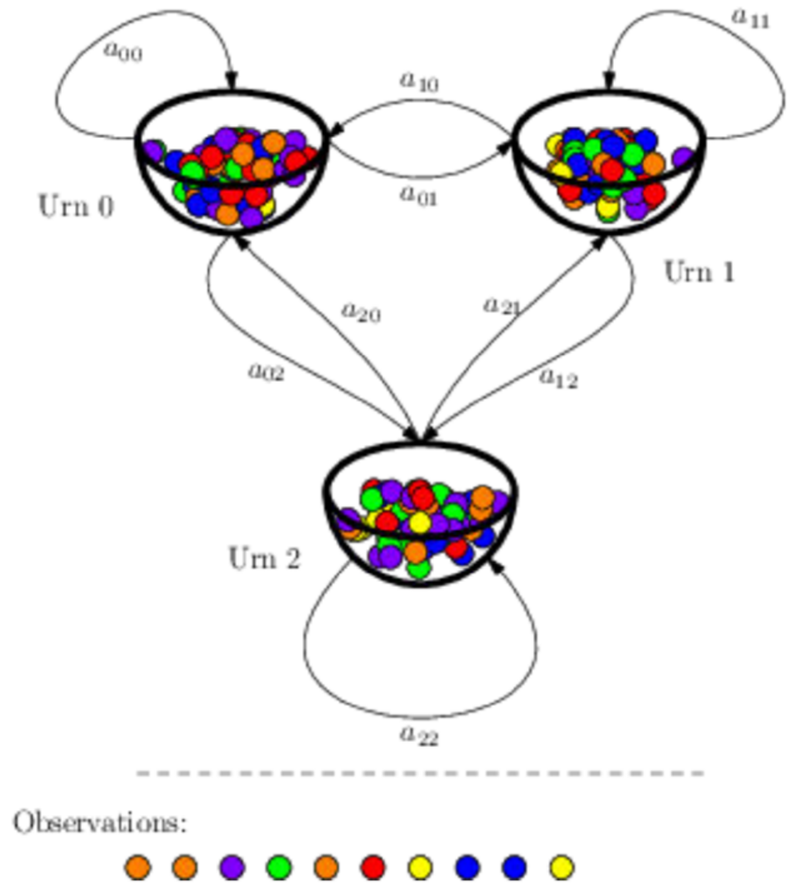
HMM

Markov Process

States | Observations | Assumptions

Standard Mathematical Example:

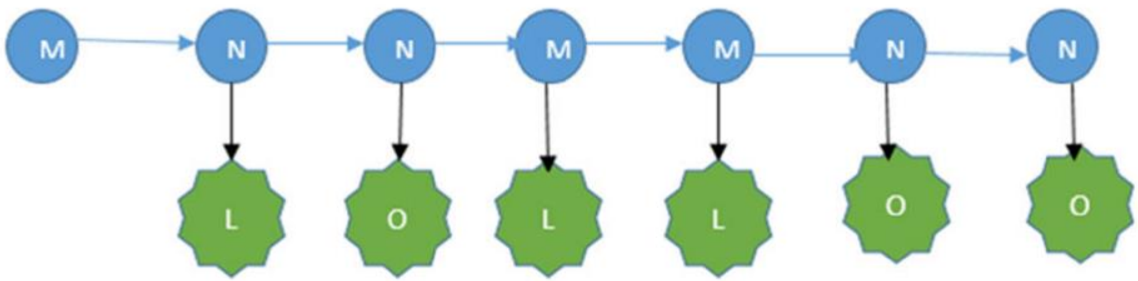
Urn & Ball Model



Hidden Markov Model

States | Observations | Assumptions

Time Slice **(t)**	0	1	2	3	4	5	6	P(**O_t | O_{t-1}**)
Observed Evidence **(O_t/ E_t)**	-	Late	OnTime	Late	Late	Ontime	Ontime	……
Unobserved State **(U_t /X_t/ Q_t)**	Meeting	No Meeting	No Meeting	Meeting	Meeting	No Meeting	No Meeting	……



Transition Model / Probability Matrix

P(**U_{t-1} = No Meeting**)	P(**U_{t-1} = Meeting**)	←Previous
0.5	0.67	P(**U_t = No Meeting**)
0.5	0.33	P(**U_t = Meeting**)

Evidence / Sensor Model/ Emission Probability Matrix

P(**U_t = No Meeting**)	P(**U_t = Meeting**)	←Unobserved Evidence v
0.9	0.3	P(**O_t = OnTime**)
0.1	0.7	P(**O_t = Late**)

Hidden Markov Process

States | Observations | Assumptions

Modelling sequences of random events and transitions between states over time is known as Morkov chain

X_t

Hidden Markov Process models events as the state sequences that are not directly observable but only be approximated from the sequence of observations produced by the system

E_t

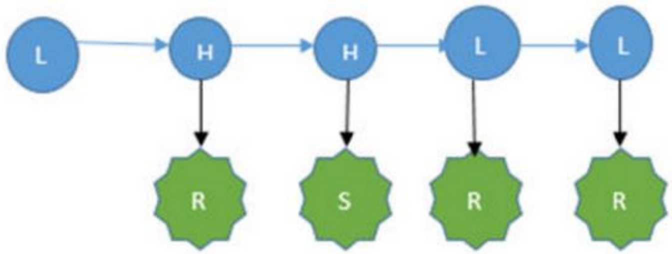
Transition Model / Probability Matrix :
Current state depends only finite number of previous states. :

Evidence / Sensor Model/ Emission Probability Matrix :
Current Evidence or Observation depends Current State of the world. Given the Current State Knowledge of the world, observation doesn't depend on history:

Hidden Morkov Model

States | Observations | Assumptions

Time Slice (t)	0	1	2	3	4	....	$P(O_t \mid O_{t-1}, O_{t-2})$
Observed Evidence (O_t)	-	Rainy	Sunny	Rainy	Rainy		
Unobserved State(U_t)	Low Pressure	High Pressure	High Pressure	Low Pressure	Low Pressure		



Transition Model / Probability Matrix

$P(U_{t-2} = LP, U_{t-1} = HP)$	$P(U_{t-2} = HP, U_{t-1} = HP)$	$P(U_{t-2} = HP, U_{t-1} = LP)$	$P(U_{t-2} = LP, U_{t-1} = LP)$	←Previous
0.2	0.40	0.85	0.5	$P(U_t = LP)$
0.8	0.60	0.15	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

Hidden Morkov Model

Filtering

P(L $_3$ | R-S-R-R)

$P(X_t \mid E_{1\ldots t})$

Prediction

P(L $_3$ | R-S)

$P(X_{t+k} \mid E_{1\ldots t})$

Smoothing

P(H $_2$ | R-S-R-R)

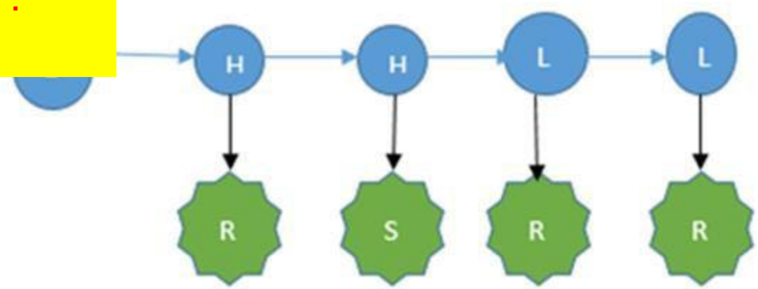
$P(X_{k,\ o>k>t} \mid E_{1\ldots t})$

Most Likely Explanation

P(H-H-L-L | R-S-R-R)

$\operatorname{argmax} X_{1\ldots t} : P(X_{1\ldots t} \mid E_{1\ldots t})$

In your Text book another example for these inferences is explained "Task of predicting the weather condition by a security personnel sitting in an underground secret installation by observing the state of an employee who either umbrella or don't". Kindly check it and work it out as additional practice

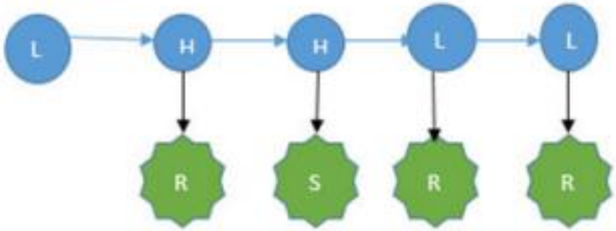


Hidden Morkov Model

Inference: Type -1

Sequence Evaluation : Likely hood Computation : Forward Algorithm

Find the probability of occurrence of
this weather sequence observation: **S-S-R**



Intuition: $P(E_{1\ldots t}) = \sum_{i=1}^{N} P(E_{1\ldots t} \mid X_{1\ldots t}) * P(X_{1\ldots t}) =$

$= \sum_{i=1}^{N} \prod_{j=1}^{t} P(Ej \mid Xj) * P(Xj \mid Xj_{-1})$

Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = \text{Rainy})$
0.2	0.6	$P(E_t = \text{Sunny})$

P(SSR)

$$= \sum_{X} P(SSR\,,\; X) = \sum_{X} P(SSR\,, X_1 X_2 X_3)$$

$$= \sum_{X} P(R, X_3, S, X_2, S, X_1) = \sum_{X} P(R \mid X_3) * P(X_3 \mid X_2) * P(S \mid X_2) * P(X_2 \mid X_1) * P(S \mid X_1) * P(X_1 \mid X_0)$$

$$= \sum_{X} P(R \mid X_3) * P(S \mid X_2) * P(S \mid X_1) * P(X_3 \mid X_2) * P(X_2 \mid X_1) * P(X_1 \mid X_0)$$

$$= \sum_{X} \prod_{j=1}^{t} P(Ej \mid X_j) * P(Xj \mid X_{j-1})$$

Hidden Morkov Model

Forward Propagation Algorithm

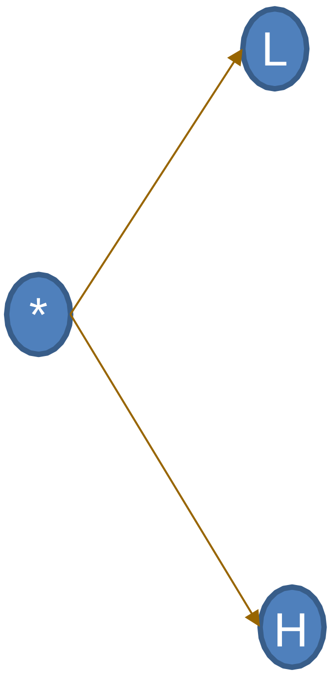
Find the probability of occurrence of this Pressure sequence observation: **S-S-R**

<u>Initialization Phase:</u>

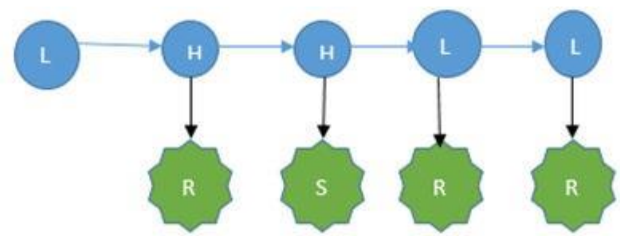
P(L)*P(S|L) = 0.5*0.2 = 0.1 □ 0.25

$$\frac{0.1}{0.3+0.1}$$

P(H)	0.5
P(L)	0.5



P(H)*P(S|H) = 0.5*0.6 = 0.3 □ 0.75



Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

Hidden Morkov Model

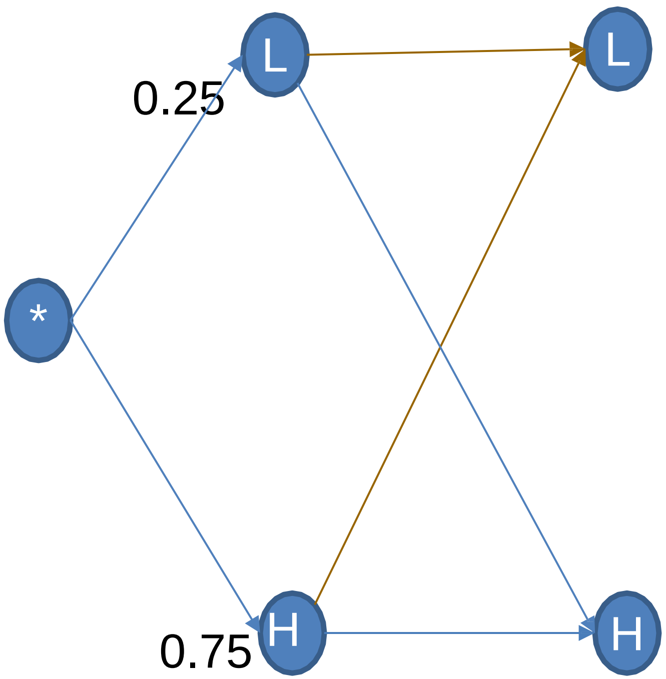
Forward Propagation Algorithm : S-S-R

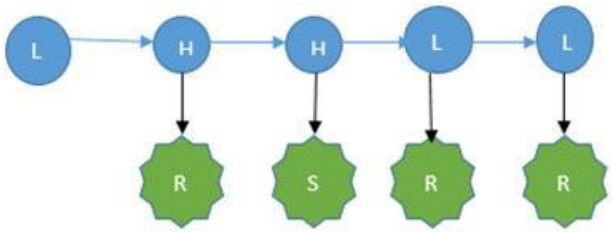
P(L)*P(L|L)*P(S|L) = 0.25*0.5*0.2 = **0.025**

P(H)*P(L|H)*P(S|L) = 0.75*0.2*0.2 = **0.03**

0.055

marginal





P(L)*P(H|L)*P(S|H) = 0.25*0.5*0.6 = **0.075**

P(H)*P(H|H)*P(S|H) = 0.75*0.8*0.6 = **0.36**

Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

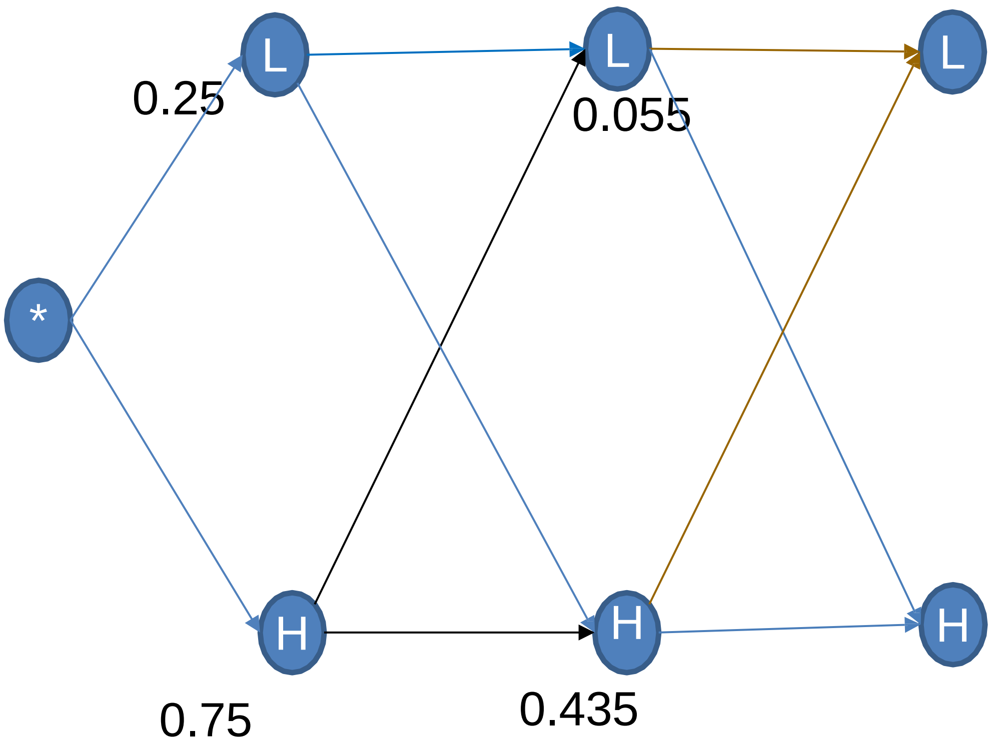
$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

Hidden Morkov Model

Forward Propagation Algorithm : S-S-R

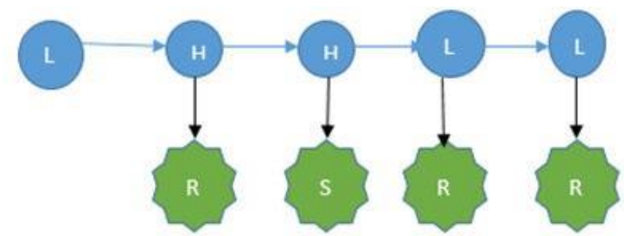
P(L)*P(L|L)*P(R|L) = 0.055*0.5*0.8 = **0.022**

P(H)*P(L|H)*P(R|L) = 0.435*0.2*0.8 = **0.0696**



P(L)*P(H|L)*P(R|H) = 0.055*0.5*0.4 = **0.011**

P(H)*P(H|H)*P(R|H) = 0.435*0.8*0.4 = **0.1392**



Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

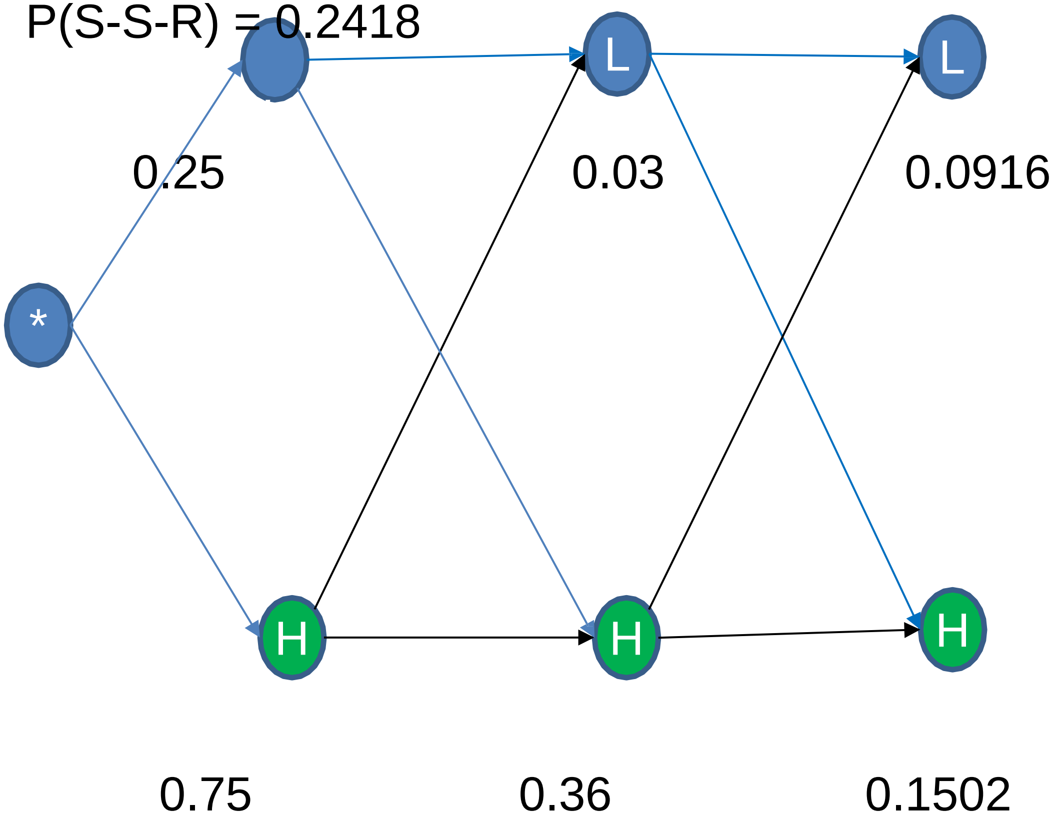
$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

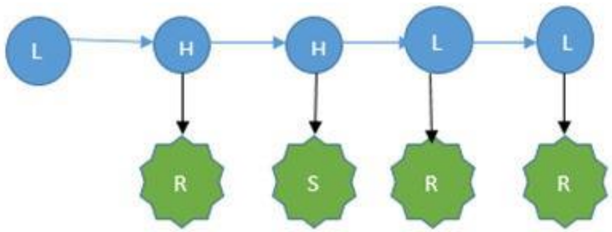
Hidden Morkov Model

Forward Propagation Algorithm : S-S-R

Termination Phase:

P(S-S-R) = 0.2418





Transition Model / Probability Matrix

P(U_{t-1} = HP)	P(U_{t-1} = LP)	←Previous
0.2	0.5	P(U_t = LP)
0.8	0.5	P(U_t = HP)

Evidence / Sensor Model/ Emission Probability Matrix

P(X_t = LP)	P(X_t = HP)	←Unobserved Evidence v
0.8	0.4	P(E_t = Rainy)
0.2	0.6	P(E_t = Sunny)

Hidden Morkov Model

Filtering : P(SecondUrnIsSelected $_3$ | Red-Blue-Blue-Yellow)

$P(X_t \mid E_{1\ldots t})$

Prediction: P(FirstUrnWillbeSelected $_3$ | Red-Yellow)

$P(X_{t+k} \mid E_{1\ldots t})$

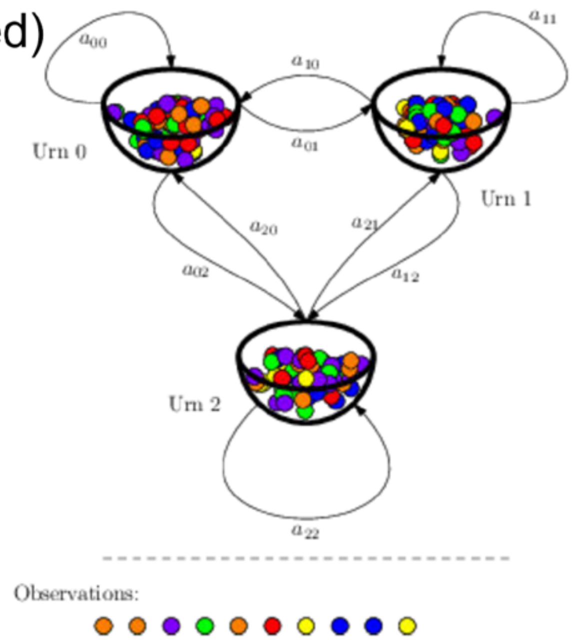
Smoothing: P(ThirdUrnWasSelected $_2$ | Red-Yellow-Red-Red)

$P(X_{k,\ o>k>t} \mid E_{1\ldots t})$

Most Likely Explanation (or) Viterbi Algorithm

P(Urn1-Urn2-Urn1 | Red-Yellow-Yellow)

$\text{argmax } X_{1\ldots t} : P(X_{1\ldots t} \mid E_{1\ldots t})$



Hidden Morkov Model

Inference: Type -2

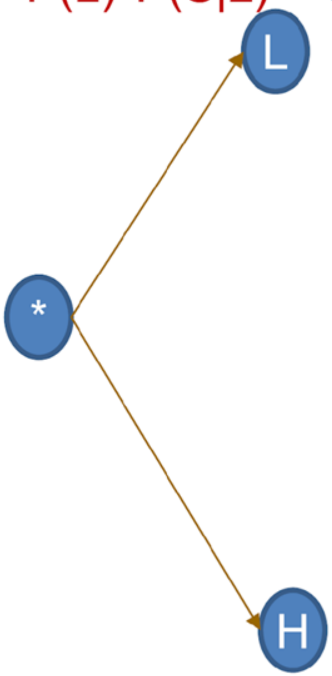
Most Likely Explanation : Veterbi Algorithm

Find the pattern in pressure that might have caused this observation: **S-S-R**

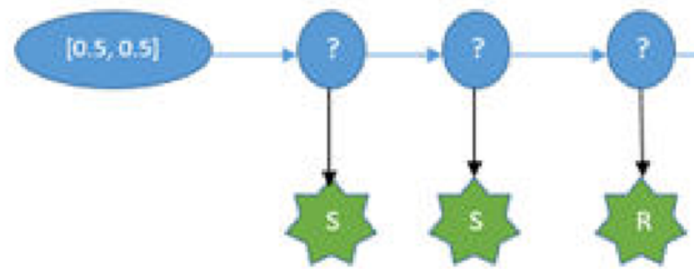
argmax $X_{1...t}$: $P(X_{1...t} \mid E_{1...t})$

MM Inf

P(L)*P(S|L) = 0.5*0.2 = 0.1 → 0.25



Normalization is only in 1st step.



P(H)*P(S|H) = 0.5*0.6 = 0.3 → 0.75

Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

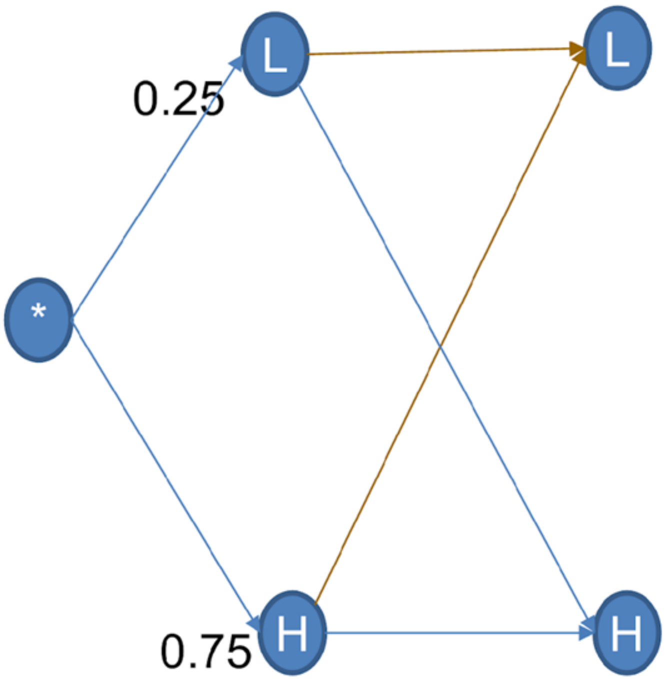
Hidden Morkov Model

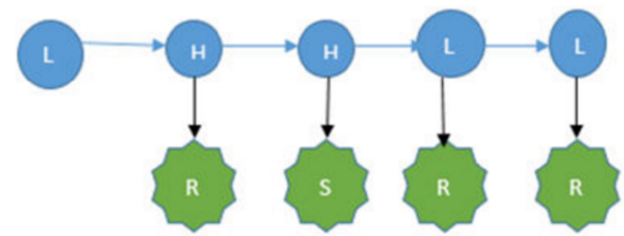
Veterbi Algorithm : S-S-R

P(L)*P(L|L)*P(S|L) = 0.25*0.5*0.2 = 0.025

P(H)*P(L|H)*P(S|L) = 0.75*0.2*0.2 = 0.03

No marginalization happen here. Only highest value is selected





Transition Model / Probability Matrix

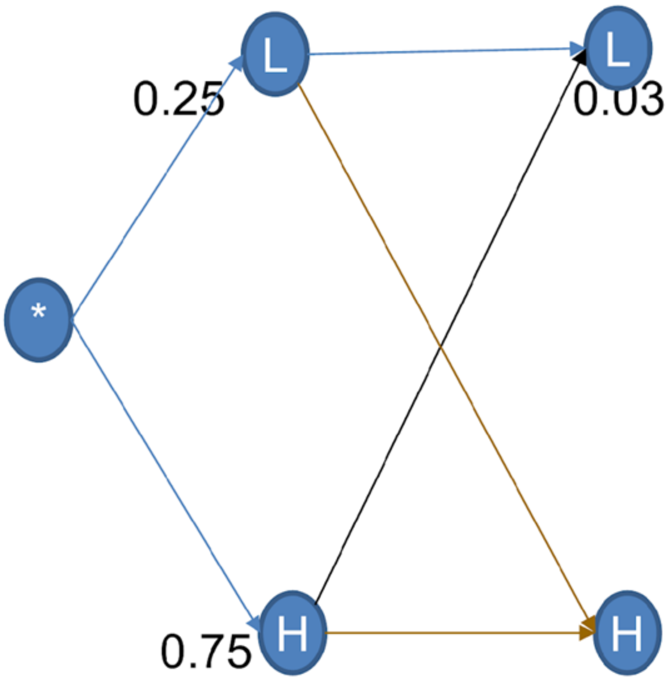
$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

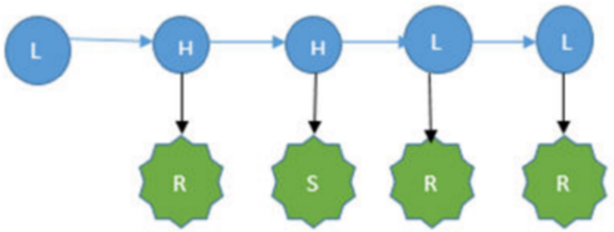
Hidden Morkov Model

Veterbi Algorithm : S-S-R



$P(L)*P(H|L)*P(S|H) = 0.25*0.5*0.6 = 0.075$

$P(H)*P(H|H)*P(S|H) = 0.75*0.8*0.6 = 0.36$



Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

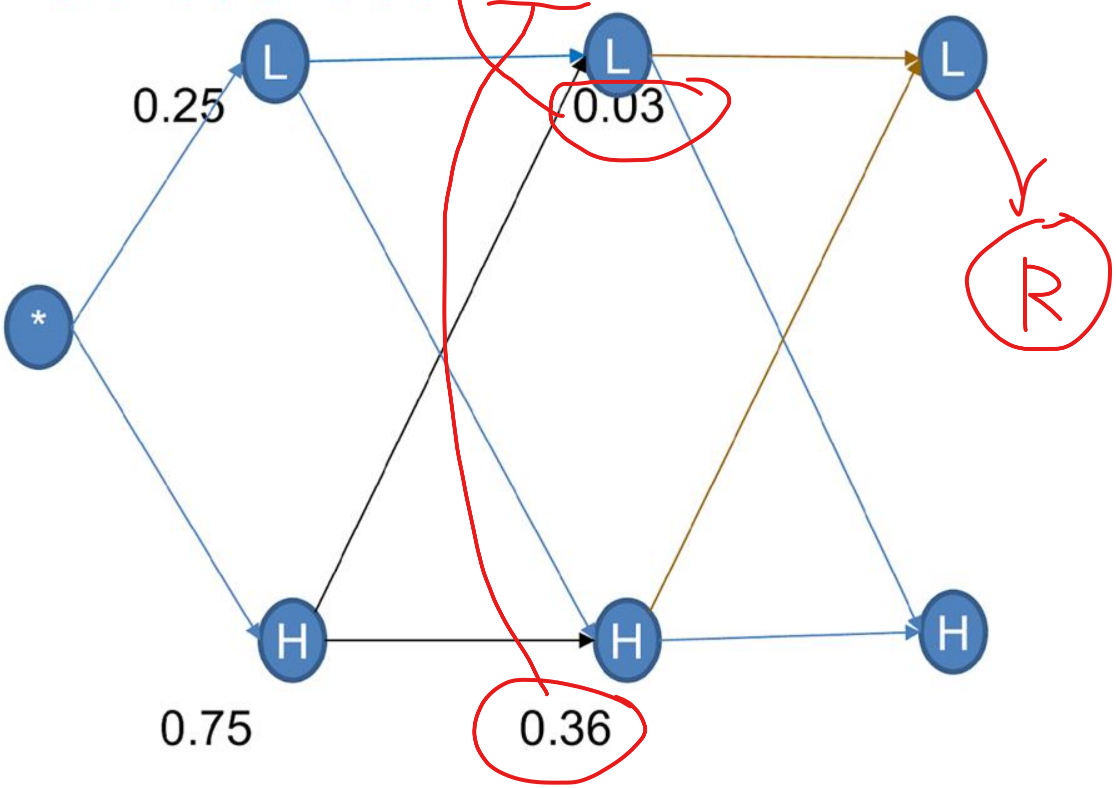
$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

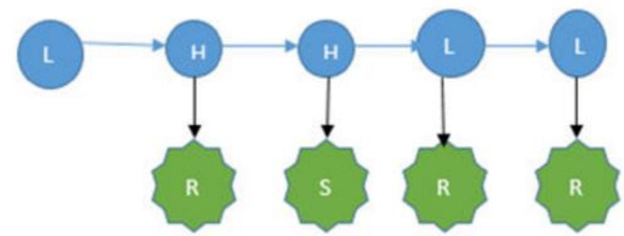
Hidden Morkov Model

Veterbi Algorithm : S-S-R

P(L)*P(L|L)*P(R|L) = 0.03*0.5*0.8 = 0.012

P(H)*P(L|H)*P(R|L) = 0.36*0.2*0.8 = **0.0576**





Transition Model / Probability Matrix

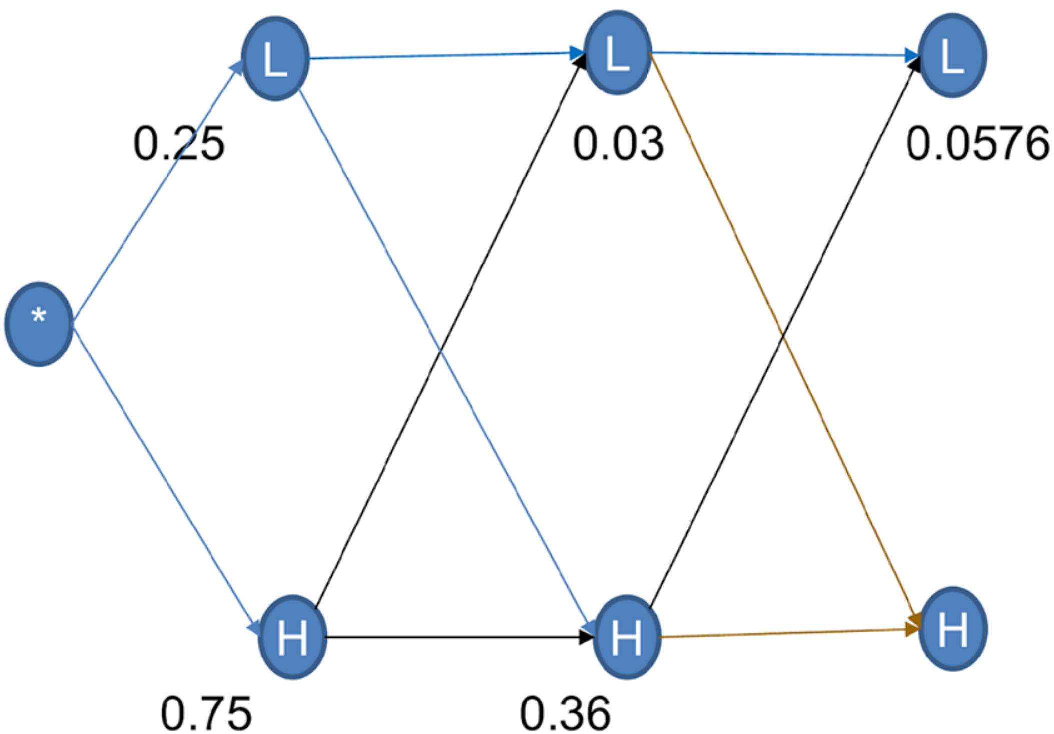
$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

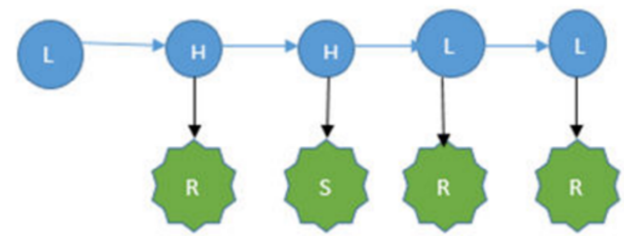
Hidden Morkov Model

Veterbi Algorithm : S-S-R



P(L)*P(H|L)*P(R|H) = 0.03*0.5*0.4 = 0.006

P(H)*P(H|H)*P(R|H) = 0.36*0.8*0.4 = 0.1152



Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

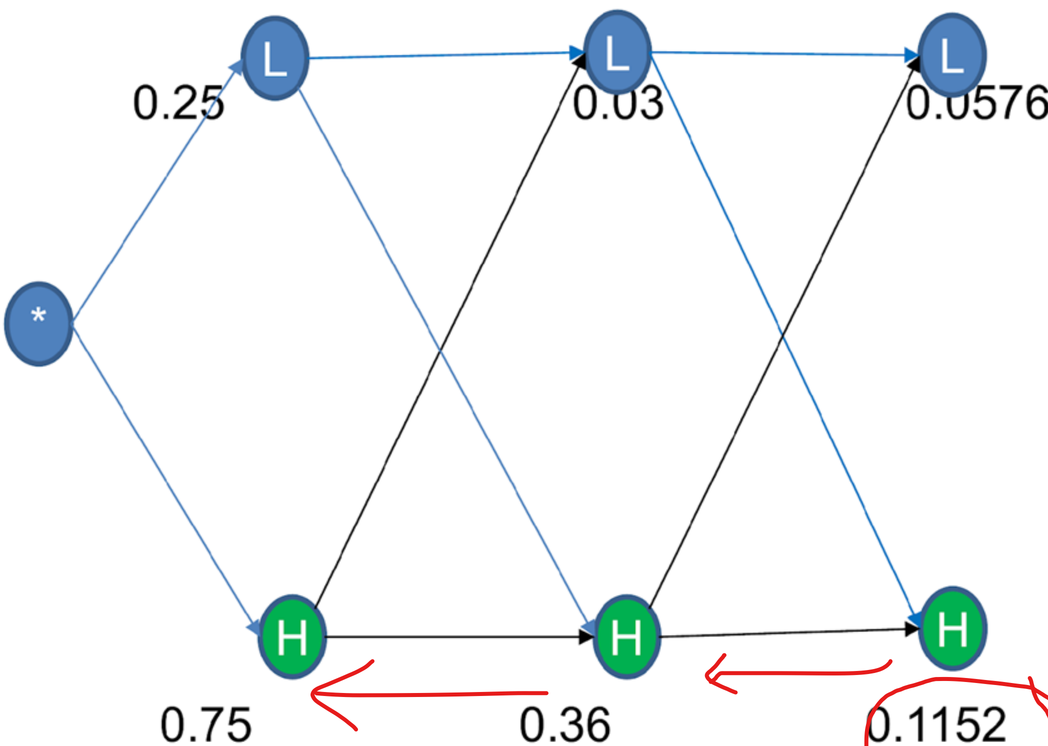
Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

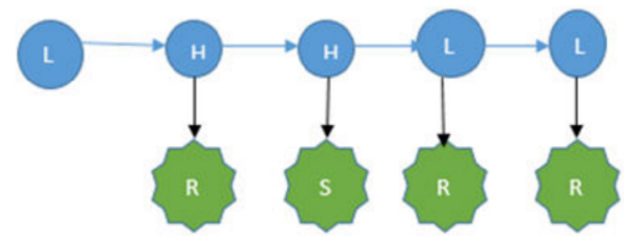
Hidden Morkov Model

Veterbi Algorithm : S-S-R

It keeps all result and finally back tracks and provide the sequence for maximum value. This will return the pattern.



$P(HHH \mid SSR) = 0.1152$



Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

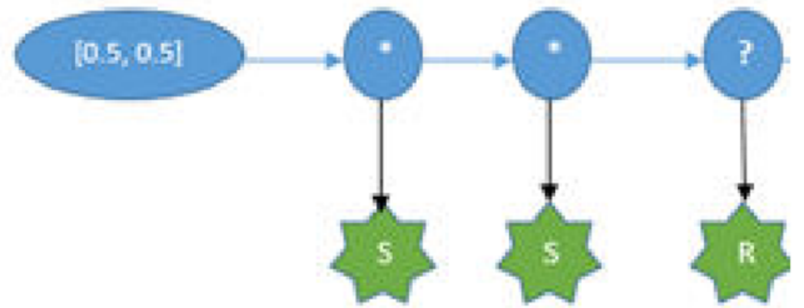
Hidden Morkov Model

Inference: Type -3

Filtering : Forward Propagation Algorithm

Find the Current Pressure if sequence of weather observations recorded are: **S-S-R**

Intuition: $P(E_{1\ldots t}) = \sum_{i=1}^{N} P(E_{1\ldots t} \mid X_{1\ldots t}) * P(X_{1\ldots t}) = \sum_{i=1}^{N} \prod_{j=1}^{t} P(Ej \mid Xj) * P(Xj \mid Xj_{-1})$



Transition Model / Probability Matrix

P(U_{t-1} = HP)	P(U_{t-1} = LP)	←Previous
0.2	0.5	P(U_t = LP)
0.8	0.5	P(U_t = HP)

Evidence / Sensor Model/ Emission Probability Matrix

P(X_t = LP)	P(X_t = HP)	←Unobserved Evidence v
0.8	0.4	P(E_t = Rainy)
0.2	0.6	P(E_t = Sunny)

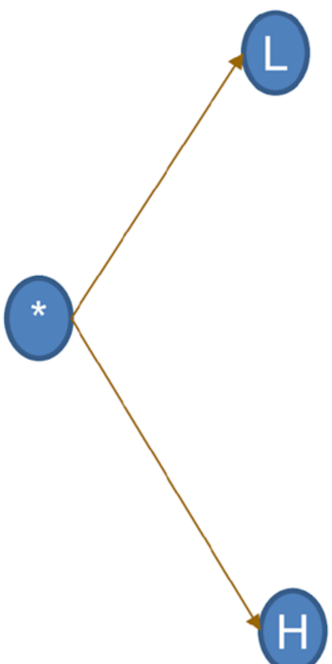
Hidden Morkov Model

Forward Propagation Algorithm

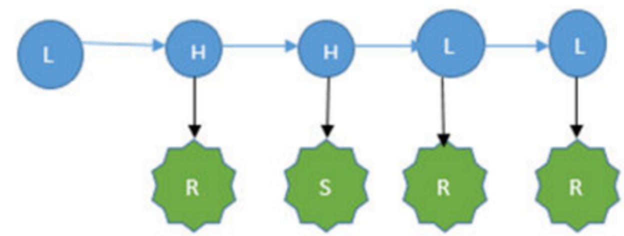
Pressure sequence observation: **S-S-R**

Initialization Phase:

P(L)*P(S|L) = 0.5*0.2 = 0.1 → 0.25



P(H)*P(S|H) = 0.5*0.6 = 0.3 → 0.75



Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

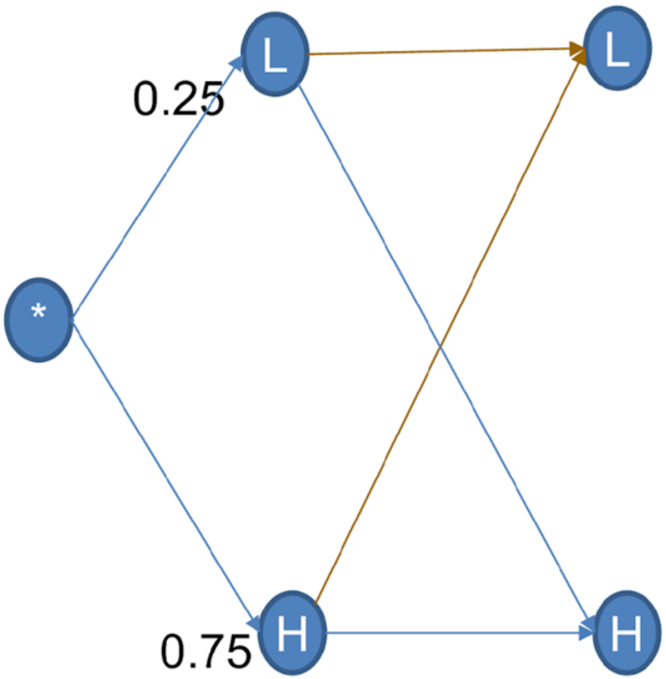
$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

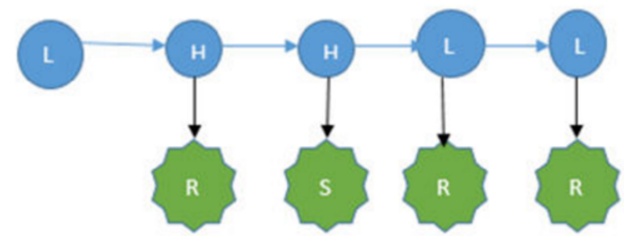
Hidden Morkov Model

Forward Propagation Algorithm : S-S-R

$P(L)*P(L|L)*P(S|L) = 0.25*0.5*0.2 = \mathbf{0.025}$

$P(H)*P(L|H)*P(S|L) = 0.75*0.2*0.2 = \mathbf{0.03}$





Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

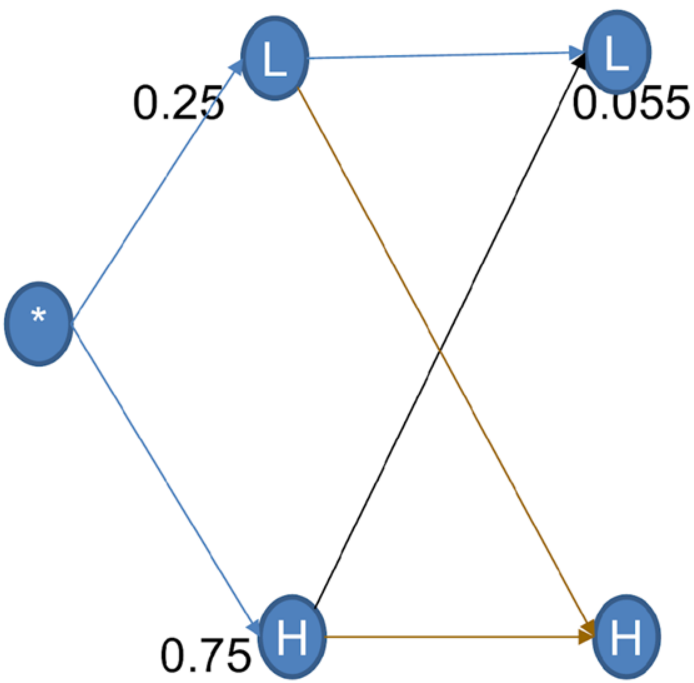
Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

<u>Recursion Phase:</u>

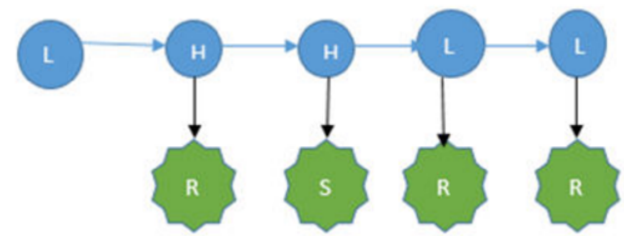
Hidden Morkov Model

Forward Propagation Algorithm : S-S-R



P(L)*P(H|L)*P(S|H) = 0.25*0.5*0.6 = **0.075**

P(H)*P(H|H)*P(S|H) = 0.75*0.8*0.6 = **0.36**



Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

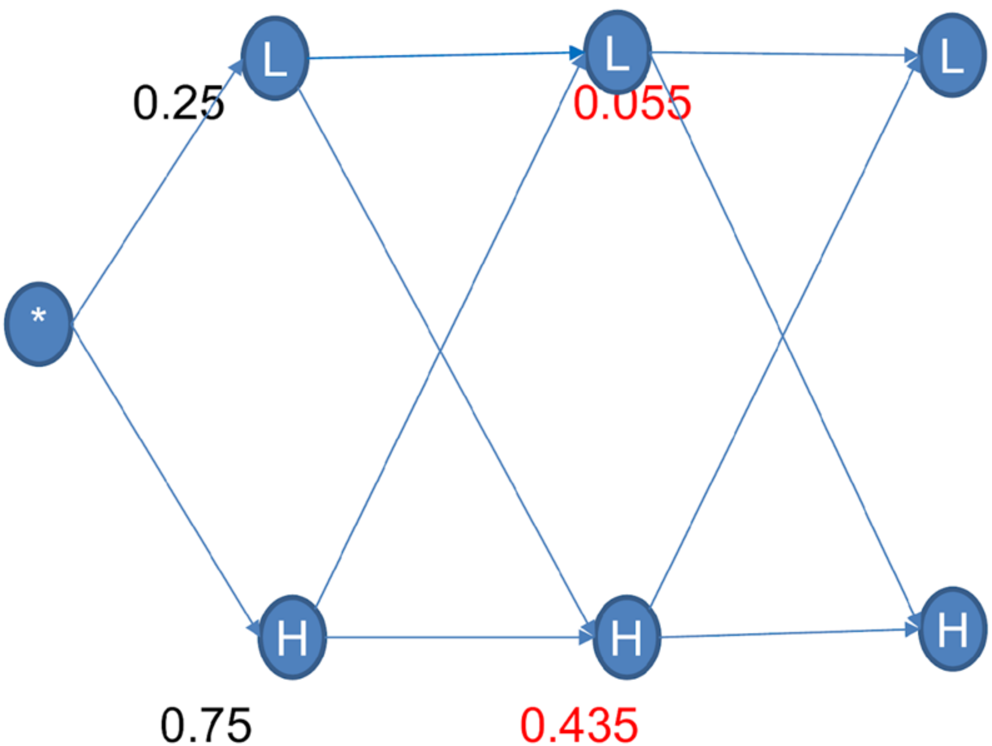
Evidence / Sensor Model/ Emission Probability Matrix

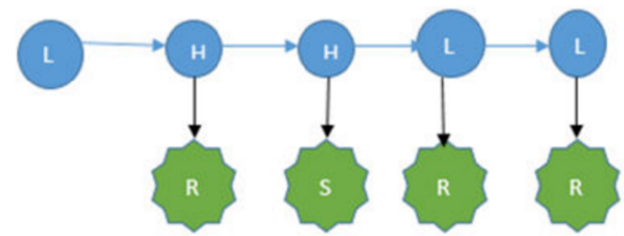
$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

Hidden Morkov Model

Forward Propagation Algorithm : S-S-R

Normalization require here





Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

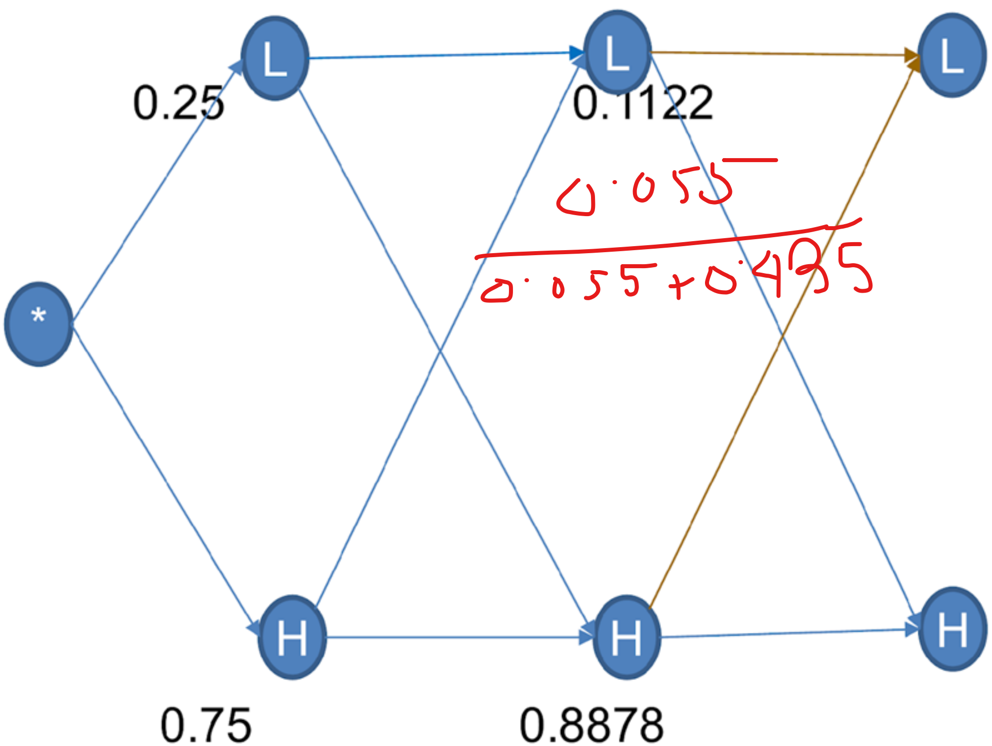
$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

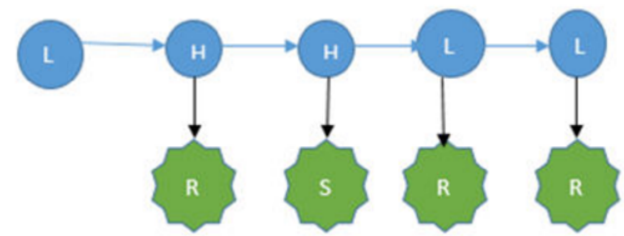
Hidden Morkov Model

Forward Propagation Algorithm : S-S-R

P(L)*P(L|L)*P(R|L) = 0.1122*0.5*0.8 = **0.04488**

P(H)*P(L|H)*P(R|L) = 0.8878*0.2*0.8 = **0.142048**





Transition Model / Probability Matrix

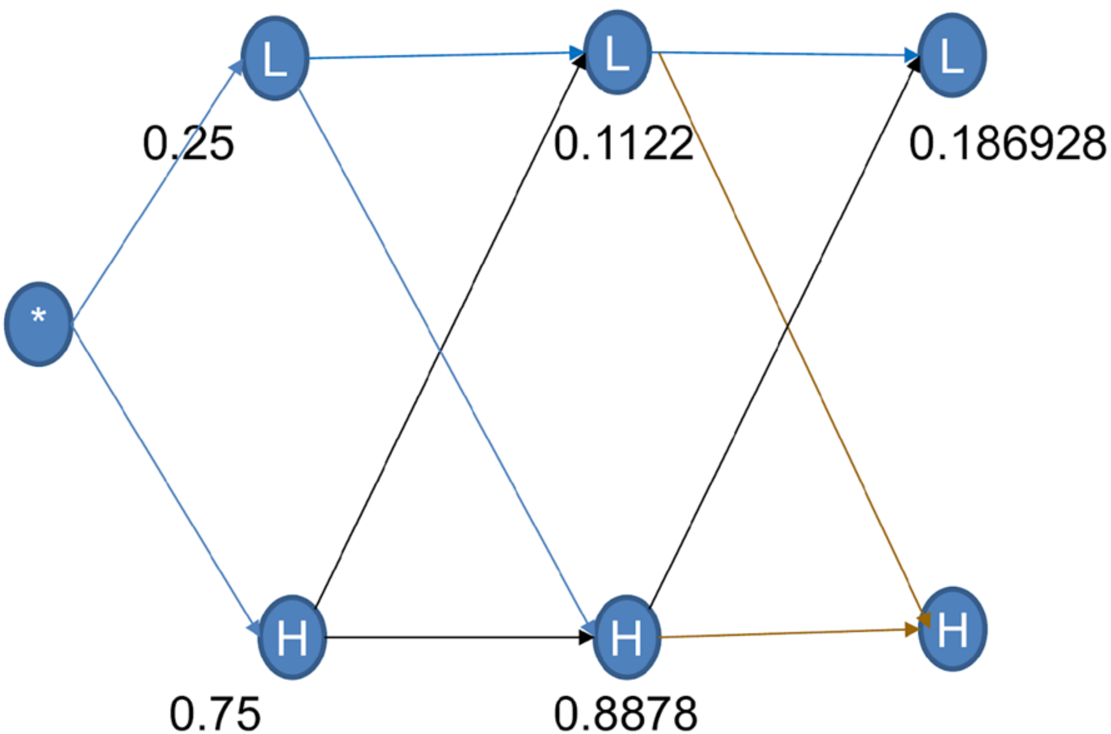
P(U_{t-1} = HP)	P(U_{t-1} = LP)	←Previous
0.2	0.5	P(U_t = LP)
0.8	0.5	P(U_t = HP)

Evidence / Sensor Model/ Emission Probability Matrix

P(X_t = LP)	P(X_t = HP)	←Unobserved Evidence v
0.8	0.4	P(E_t = Rainy)
0.2	0.6	P(E_t = Sunny)

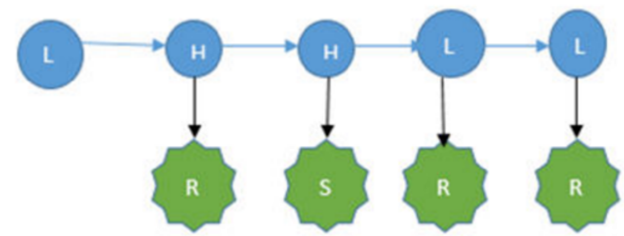
Hidden Morkov Model

Forward Propagation Algorithm : S-S-R



P(L)*P(H|L)*P(R|H) = 0.1122*0.5*0.4 = **0.02244**

P(H)*P(H|H)*P(R|H) = 0.8878*0.8*0.4 = **0.284096**



Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

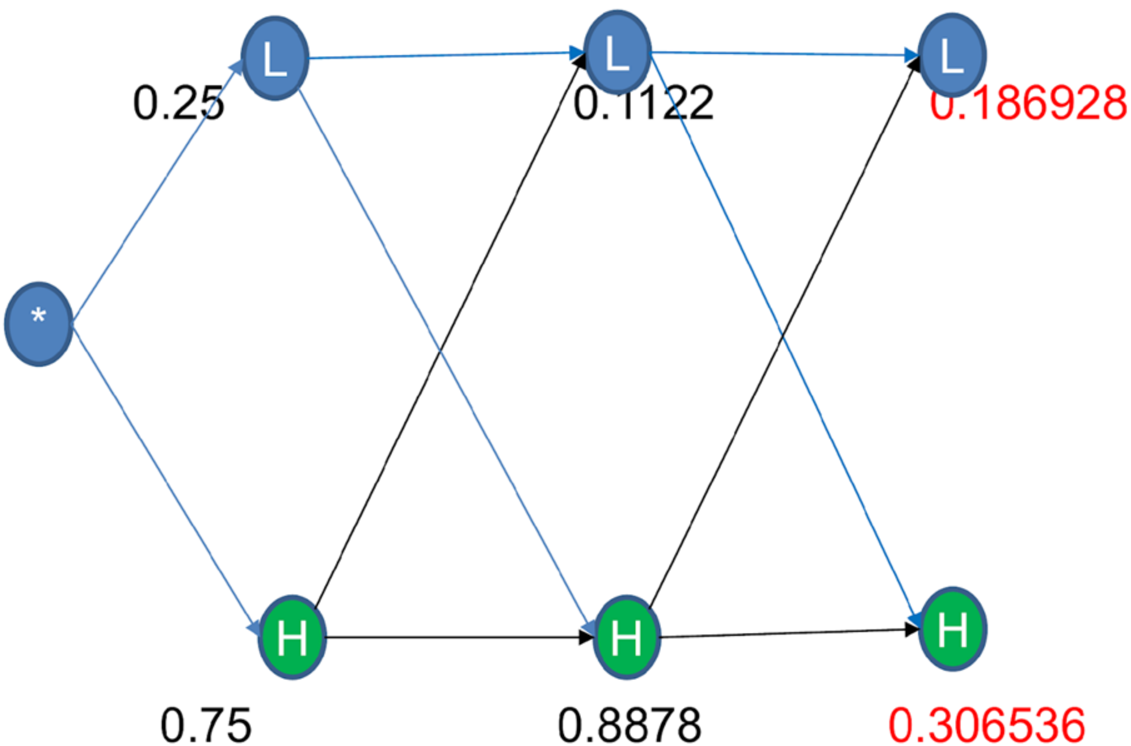
Evidence / Sensor Model/ Emission Probability Matrix

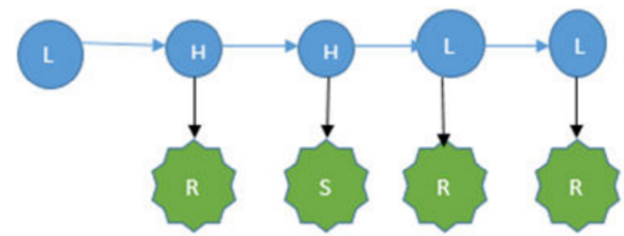
$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

Hidden Morkov Model

Forward Propagation Algorithm : S-S-R

Termination Phase:





Transition Model / Probability Matrix

P(U$_{t-1}$ = HP)	P(U$_{t-1}$ = LP)	←Previous
0.2	0.5	P(U$_t$ = LP)
0.8	0.5	P(U$_t$ = HP)

Evidence / Sensor Model/ Emission Probability Matrix

P(X$_t$ = LP)	P(X$_t$ = HP)	←Unobserved Evidence v
0.8	0.4	P(E$_t$ = Rainy)
0.2	0.6	P(E$_t$ = Sunny)

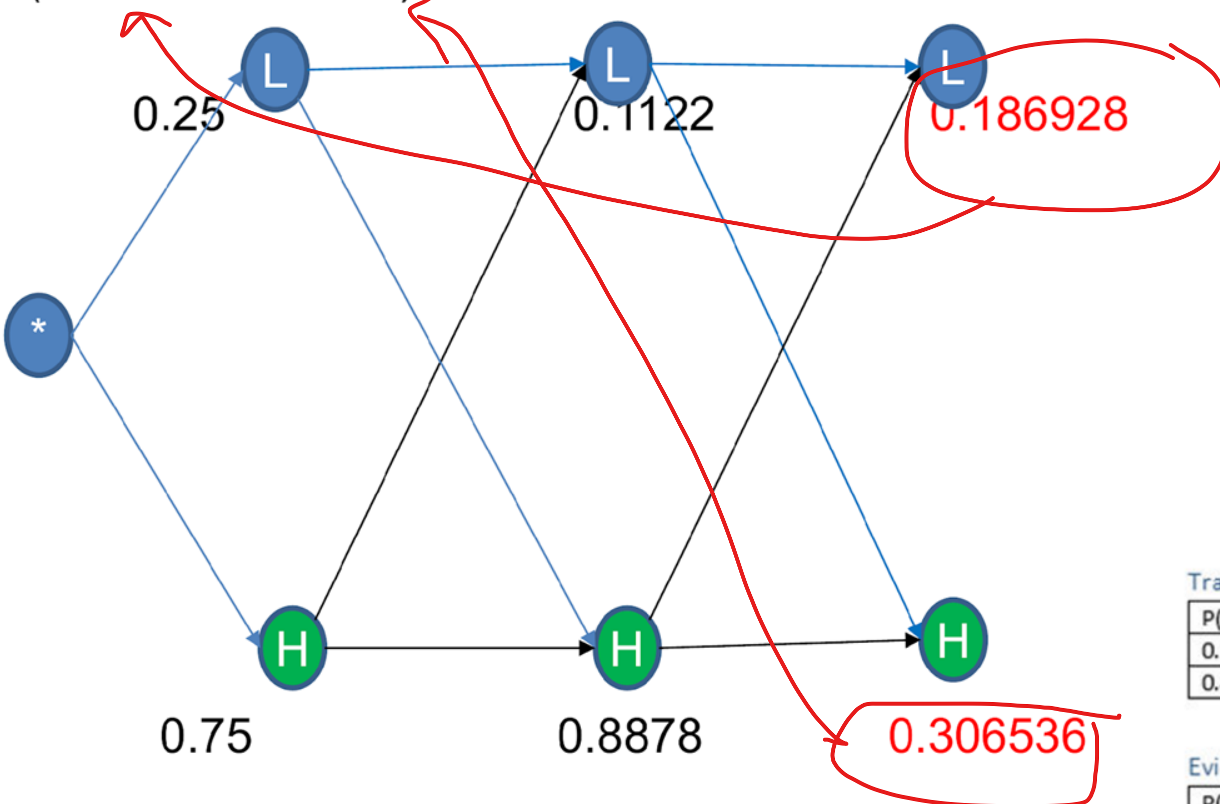
Hidden Morkov Model

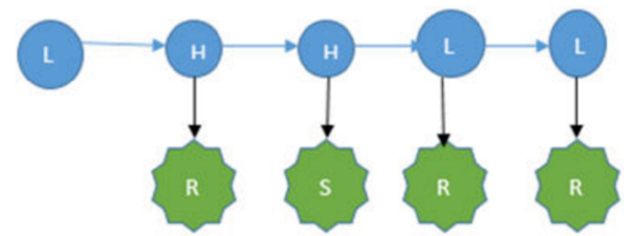
Forward Propagation Algorithm : S-S-R

Termination Phase:

(0.37881,**0.62119**)

After normalized.





Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

Hidden Morkov Model

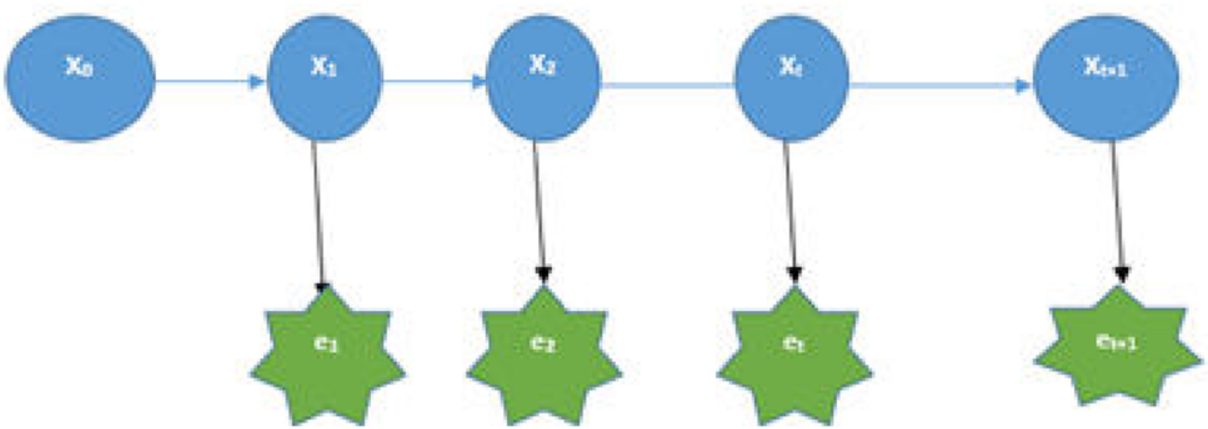
Inference: Type -3

Filtering : Forward Propagation Algorithm

Find the Current Pressure if sequence of weather observations recorded are: **S-S-R**

Intuition: $P(Xt_{+1}\,|\,E_{1\ldots t+1}) = \boldsymbol{\alpha}\, P(et_{+1}\,|\,Xt_{+1}) * \sum_{X_t} P(X_{t+1}|X_t) * \boldsymbol{P}(\mathbf{X_t}|\,\mathbf{E_{1..t}})$

$$P(Xt_{+1}\,|\,E_{1\ldots t+1}) = \boldsymbol{\alpha}\, P(e_{t+1}\,|\,Xt_{+1}) * \sum_{X_t} P(X_{t+1}|X_t) * \boldsymbol{P}(\mathbf{X_t}|\,\mathbf{E_{1..t}})$$



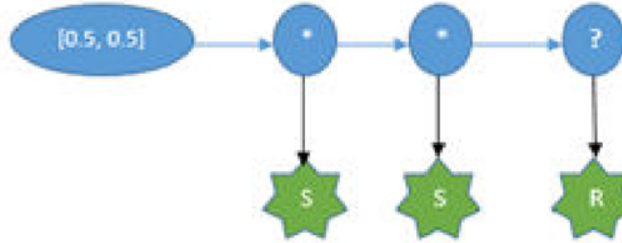
Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

Hidden Morkov Model



-3

Filtering : Forward Propagation Algorithm

Find the Current Pressure if sequence of weather observations recorded are: **S-S-R**

Intuition: $P(Xt_{+1} \mid E_{1\ldots t+1}) = \boldsymbol{\alpha}\, P(et_{+1} \mid Xt_{+1}) * \sum_{X_t} P(X_{t+1} \mid X_t) * \boldsymbol{P(X_t \mid E_{1..t})}$

$P(X_3 \mid SSR) = P(X_3 \mid S,S,R)$

$$= \frac{\boldsymbol{P(R \mid X_3, S, S)} * P(X_3 \mid S, S)}{P(R)}$$

$$= \frac{P(R \mid X_3) * \boldsymbol{P(X_3 \mid S, S)}}{P(R)}$$

$$= \frac{P(R \mid X_3) * \{\sum_{X_2} P(X_3 \mid X_2) * \boldsymbol{P(X_2 \mid S, S)}\}}{P(R)}$$

$$= \frac{P(R \mid X_3) * \{\sum_{X_2} P(X_3 \mid X_2) * \overset{S \mid X2}{P(\cancel{R} \mid \cancel{X_3})} * \{\sum_{X_1} P(X_2 \mid X_1) * \boldsymbol{P(X_1 \mid S)}\}\}}{P(R) * P(S)}$$

Transition Model / Probability Matrix

P(Ut-1 = HP)	P(Ut-1 = LP)	←Previous
0.2	0.5	P(Ut = LP)
0.8	0.5	P(Ut = HP)

$$P(Xt_{+1} \mid E_{1\ldots t+1}) = \boldsymbol{\alpha}\, P(e_{t+1} \mid Xt_{+1}) * \sum_{X_t} P(X_{t+1} \mid X_t) * \boldsymbol{P(X_t \mid E_{1..t})}$$

Evidence / Sensor Model/ Emission Probability Matrix

P(Xt = LP)	P(Xt = HP)	←Unobserved Evidence v
0.8	0.4	P(Et = Rainy)
0.2	0.6	P(Et = Sunny)

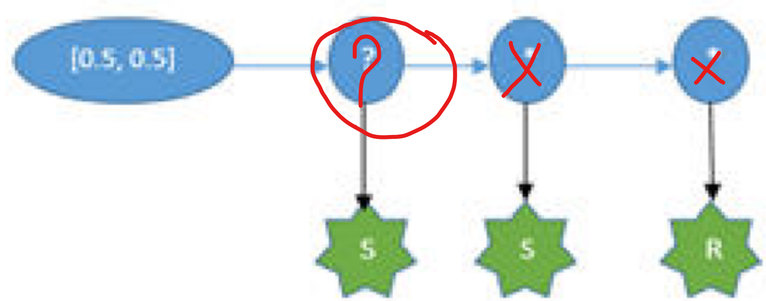
Hidden Morkov Model

Inference: Type -4

Smoothing : Backward Propagation Algorithm (Most Likely State Estimation)

Find the Pressure in past instance of time if sequence of following future weather observations recorded are: **S-S-R**

Intuition: $P(E_{1\ldots t}) = \sum_{i=1}^{N} P(E_{1\ldots t} \mid X_{1\ldots t}) * P(X_{1\ldots t}) = \sum_{i=1}^{N} \prod_{j=1}^{t} P(E_j \mid X_j) * P(X_j \mid X_{j-1})$



Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = \text{Rainy})$
0.2	0.6	$P(E_t = \text{Sunny})$

Hidden Morkov Model

Inference: Type -4

Smoothing : Backward Propagation Algorithm

Find the Pressure in past instance of time if sequence of following future weather observations recorded are: **S-S-R**

Intuition: $P(Xt_{+1} | E_{1...t+1}) = \boldsymbol{\alpha}\, P(et_{+1} | Xt_{+1}) * \sum_{X_t} P(X_{t+1}|X_t) * \boldsymbol{P}(\mathbf{X_t} | \mathbf{E_{1..t}})$

$P(X_1 | SSR) = P(X_1 | S,S,R)$

$$= \frac{\boldsymbol{P}(\boldsymbol{SR} | \mathbf{X_1}\, \boldsymbol{S}) * P(X_1 | S)}{P(SR)}$$

$$= \frac{P(X_1 | S) * \{\sum_{X_2} P(X_2|X_1) * \boldsymbol{P}(\mathbf{SR} | \mathbf{X_2}\, \mathbf{X_1})\}}{P(SR)}$$

$$= \frac{P(X_1 | S) * \{\sum_{X_2} P(X_2|X_1) * \boldsymbol{P}(\mathbf{SR} | \mathbf{X_2})\}}{P(SR)}$$

$$= \frac{P(X_1 | S) * \{\sum_{X_2} P(X_2|X_1) * P(S | X_2) * \boldsymbol{P}(\mathbf{R} | \mathbf{X_2})\}}{P(SR)}$$

$$= \frac{P(X_1 | S) * \{\sum_{X_2} P(X_2|X_1) * P(S | X_2) * \{\sum_{X_3} P(X_3|X_2) * P(R | X_3) * \boldsymbol{P}(. | \mathbf{X_3})\}\}}{P(SR)}$$

Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probab

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

$$P(Xt | E_{t+1,\, t+2\, ...\, z}) = \boldsymbol{\alpha} * \text{fwd msg} * \sum_{X_{t+1}} P(X_{t+1}|X_t) * P(e_{t+1}|X_{t+1}) * \boldsymbol{P}(\mathbf{E_{t+2..z}} | \mathbf{X_{t+1}})$$

Hidden Morkov Model

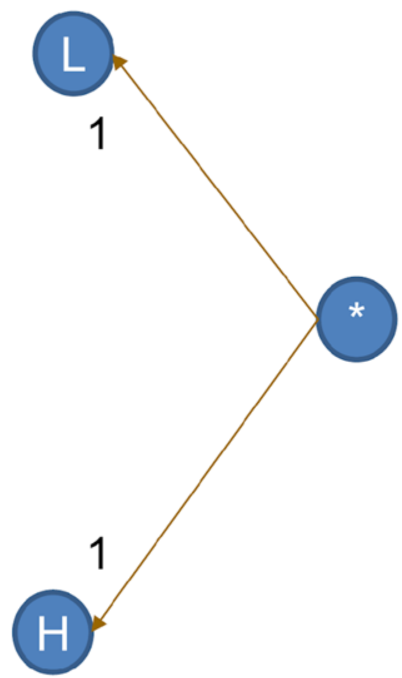
Backward Propagation Algorithm

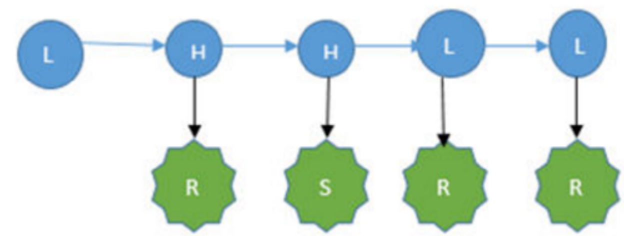
Pressure sequence observation: **S-S-R**

Initialization Phase: Set value 1 for the terminal state

P(L|L)*P(R|L)*P(.|L) = 0.5*0.8 * 1= 0.40

P(H|L)*P(R|H)*P(.|H) = 0.5*0.4 *1= 0.2





Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

P(L|H)*P(R|L)*P(.|L) = 0.2*0.8 * 1= 0.16

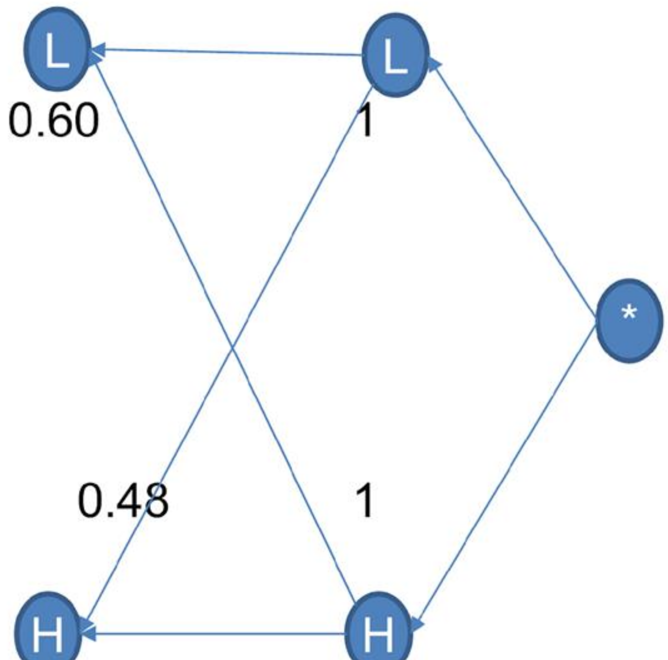
P(H|H)*P(R|H)*P(.|H) = 0.8*0.4 *1= 0.32

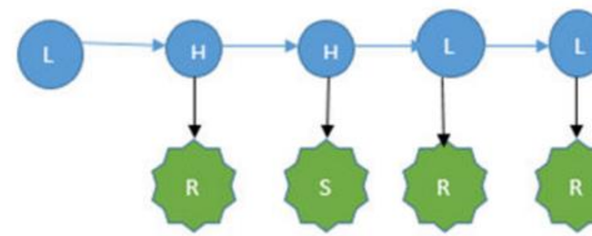
Hidden Morkov Model

Backward Propagation Algorithm : S-S-R

P(L|L)*P(S|L)*MSG(L`) = 0.5*0.2 * 0.60= 0.06

P(H|L)*P(S|H)*MSG(H`) = 0.5*0.6*0.48= 0.144





Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

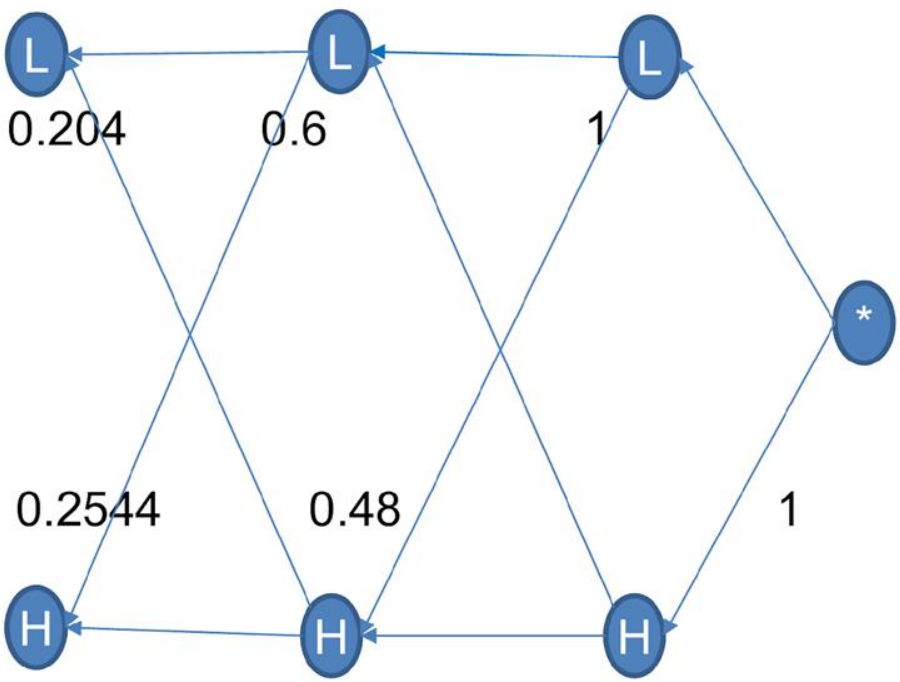
P(L|H)*P(S|L)*MSG(L`) = 0.2*0.2 * 0.6= 0.024

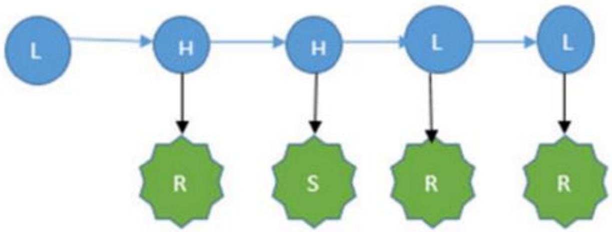
P(H|H)*P(S|H)*MSG(H`) = 0.8*0.6 *0.48= 0.2304

<u>Recursion Phase:</u>

Hidden Morkov Model

Backward Propagation Algorithm : S-S-R





Recursion Phase: If it continues if needed !!!!

Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

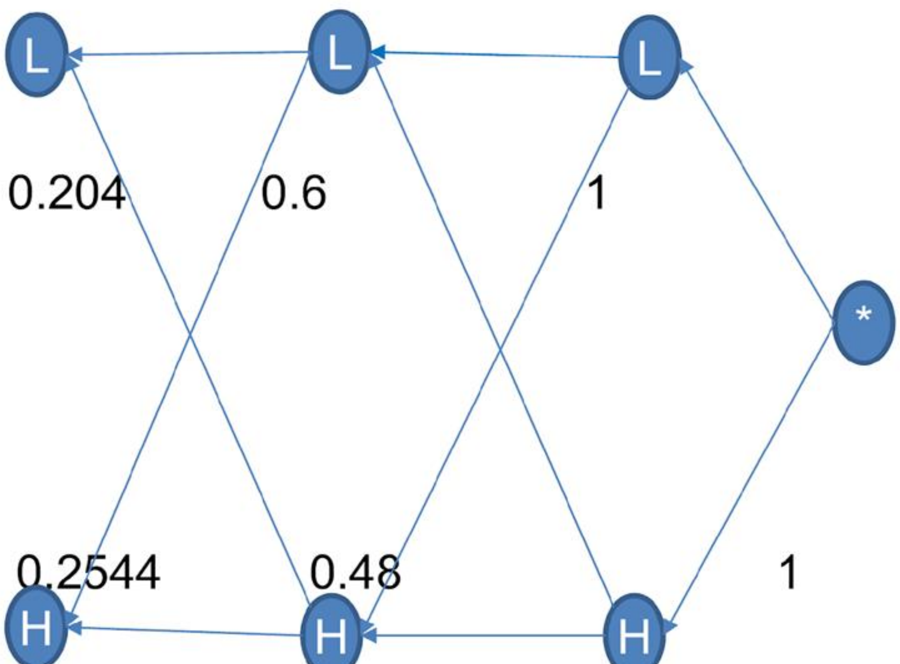
Hidden Morkov Model

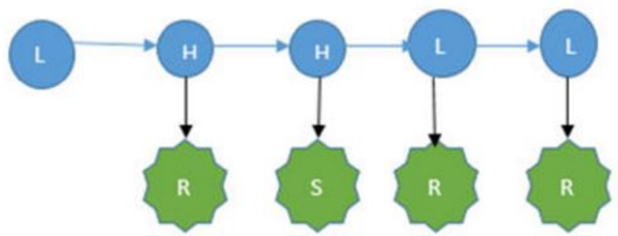
Backward Propagation Algorithm : S-S-R

P(H|SSR) > P(L|SSR)

P(L)*P(S|L)*MSG(L`) = 0.5*0.2 * 0.204= 0.0204

Normalize at the end.





Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = \text{Rainy})$
0.2	0.6	$P(E_t = \text{Sunny})$

P(H)*P(S|H)*MSG(H`) = 0.5*0.6 * 0.2544= 0.07632

Termination Phase: (0.2109,0.7891)

Normalize :Initial value * Emission at start* backMsg

function FORWARD-BACKWARD(**ev**, *prior*) **returns** a vector of probability distributions
  inputs: **ev**, a vector of evidence values for steps $1, \ldots, t$
    prior, the prior distribution on the initial state, $\mathbf{P}(\mathbf{X}_0)$
  local variables: **fv**, a vector of forward messages for steps $0, \ldots, t$
    b, a representation of the backward message, initially all 1s
    sv, a vector of smoothed estimates for steps $1, \ldots, t$

  $\mathbf{fv}[0] \leftarrow prior$
  for $i = 1$ **to** t **do**
    $\mathbf{fv}[i] \leftarrow$ FORWARD($\mathbf{fv}[i-1], \mathbf{ev}[i]$)
  for $i = t$ **downto** 1 **do**
    $\mathbf{sv}[i] \leftarrow$ NORMALIZE($\mathbf{fv}[i] \times \mathbf{b}$)
    $\mathbf{b} \leftarrow$ BACKWARD($\mathbf{b}, \mathbf{ev}[i]$)
  return sv

Figure 15.4 The forward–backward algorithm for smoothing: computing posterior probabilities of a sequence of states given a sequence of observations. The FORWARD and BACKWARD operators are defined by Equations (15.5) and (15.9), respectively.

Forward Path Probability

$\alpha_t(j) \quad = \alpha_{t-1(}i)\ ai_{,j}\ bj_{(}o_{t)}$

i

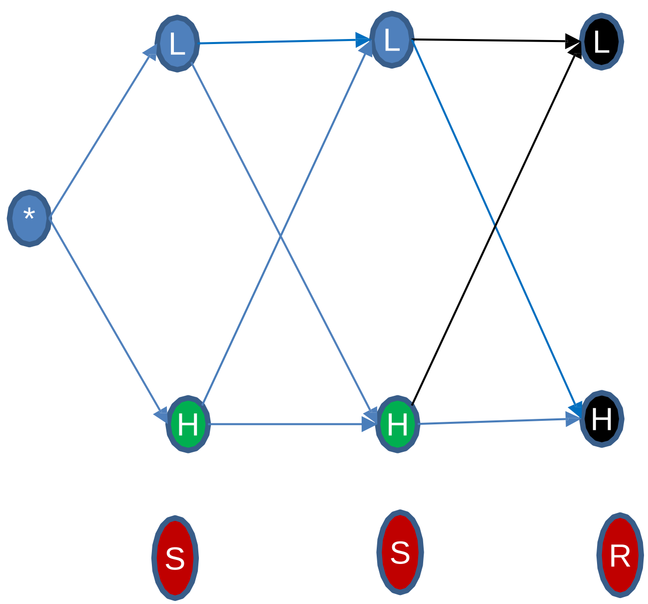
$P(O_{1..t} \mid \lambda)$

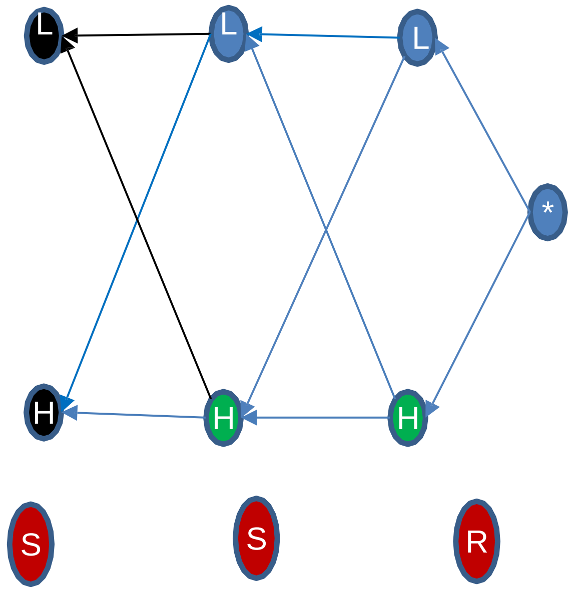
Backward Path Probability

$\beta_t(i) =_j \quad \beta_{t+1}(j)\ a_{i,j}\ bj(o_{t+1})$

$P(O_{t+1, t+2..t+k} \mid \lambda)$

$\gamma_t(i) = P(X_t \mid O_{1...t\ ...t+1,\ t+2..t+k} \mid \lambda)$: Forward – Backward Algorithm





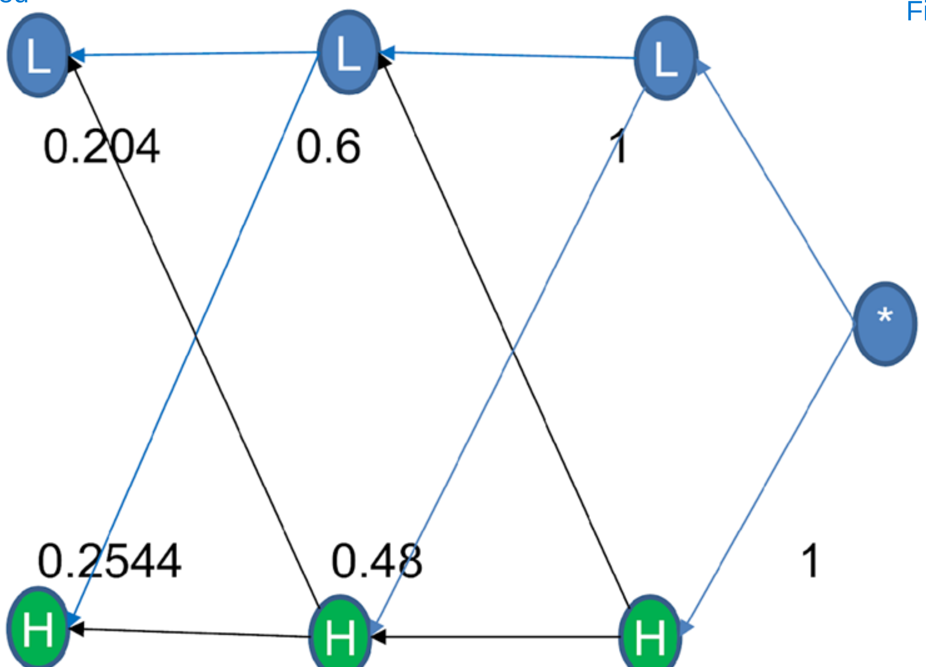
Hidden Morkov Model

Forward Backward Propagation Algorithm : S-S-R

$P(X2 \mid SSR) = \alpha * P(X2|SS) * P(R|X2)$

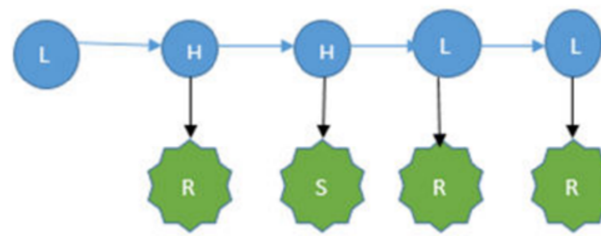
$P(X2 \mid SSR) = \alpha * (\underbrace{0.1122 , 0.8878}_{\text{Forward propagation data}})* (\underbrace{0.6, 0.48}_{\text{Backward propagation}}) = (0.06732, 0.426144) = (0.14,0.86)$

Normalized



Termination Phase: $X2$ = ??? → **X2 = H**

Find the 2nd instance given the observation is S-S-R?



Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability Matrix

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

Required Reading: AIMA - Chapter #15.1, #15.2, #15.3, #20.3.3

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

BITS Pilani
Pilani Campus

Artificial & Computational Intelligence

AIML CLZG557

M6: Reasoning over time & M7:Ethics in AI

Raja vadhana P
Assistant Professor,
BITS - CSIS

Course Plan

M1 Introduction to AI

M2 Problem Solving Agent using Search

M3 Game Playing

M4 Knowledge Representation using Logics

M5 Probabilistic Representation and Reasoning

M6 Reasoning over time

M7 Ethics in AI

Module 6: Reasoning over time

Reasoning Over Time

A. Time and Uncertainty

B. Inference in temporal models

C. Overview of HMM

D. Learning HMM Parameters using EM Algorithm

E. Applications of HMM

Reasoning Over Time

Learning Objective

1. Relate to the application of the Hidden Markov Model

2. Machine Learning for estimating the parameters of HMM

3. Understand the connect between the ethical impact and design of Intelligent agents

4. Understand the explainabilty/interpretability of Intelligent systems

5. Relate the use of logics in the explainability of complex systems

Forward Path Probability

$\alpha_t(j) = \alpha_{t-1(}i)\ a i_{,j}\ b j\ _{(}o\ _{t\ )}$

i

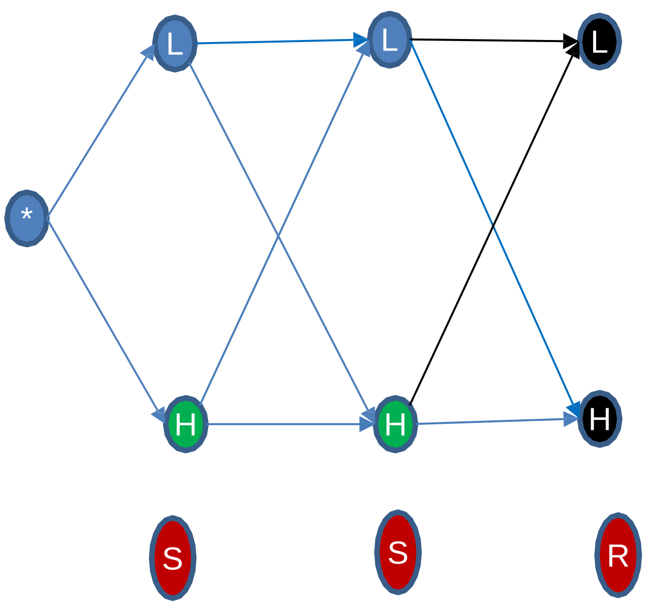
$P(O_{1..t} \mid \boldsymbol{\lambda}\ )$

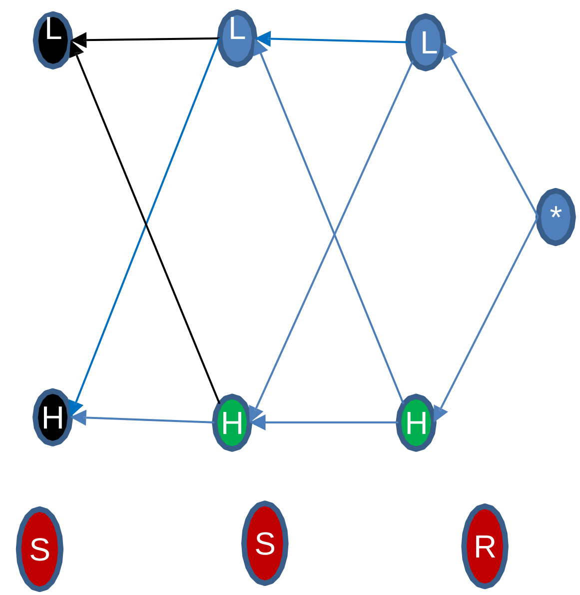
Backward Path Probability

$\beta_t(i) = _{j}\quad \beta_{t+1}(j)\ a_{i,\ j}\ bj(o\ _{t+1})$

$P(O_{t+1,\ t+2..t+k} \mid \boldsymbol{\lambda}\ )$

$\boldsymbol{\gamma}_t(i) = P(X_t \mid O_{1\ldots t\ \ldots t+1,\ t+2..t+k} \mid \boldsymbol{\lambda})$: Forward – Backward Algorithm





Text & Natural Language Processing

HMM Application

Initial	Prob
N	0.67 4/6
D	0.33 2/6
V	0
J	0
A	0
P	0

	N	D	V	J	A	P
N		01675		0.67		1
D			0.571			
V	0.63	0.1675	0.143			
J		0.33	0.143			
A						
P		0.1675				
End	0.37	0.1675	0.143	0.33		

$P(N_{t+1} | N_t)$

Given the corpus with tags to build training data.

1. Create initial probability matrix.
2. Transition probability matrix
3. Emission probability matrix
4. Use HMM Veterbi algorithm to predict the sequence of PoS Tags for given test data / sentence.

In the HMM model , the PoS tags act as the hidden states and the word in the given test sentence as the observed states.

- Boys are taller.
  N V J
- This is the tree.
  D V D N
- She is a tall girl.
  N V D J N
- Trees are more.
  N V D
- Girls are more than boys.
  N V D P N
- The tall tree is falling.
  D J N V V

Initial	Prob
N	0.67
D	0.33
V	0
J	0
A	0
P	0

N	D	V	J	A	P	
	01675		0.67		1	N
		0.571				
0.63	0.1675	0.143				
	0.33	0.143				
	0.1675					
0.37	0.1675	0.143	0.33			

Emission Table

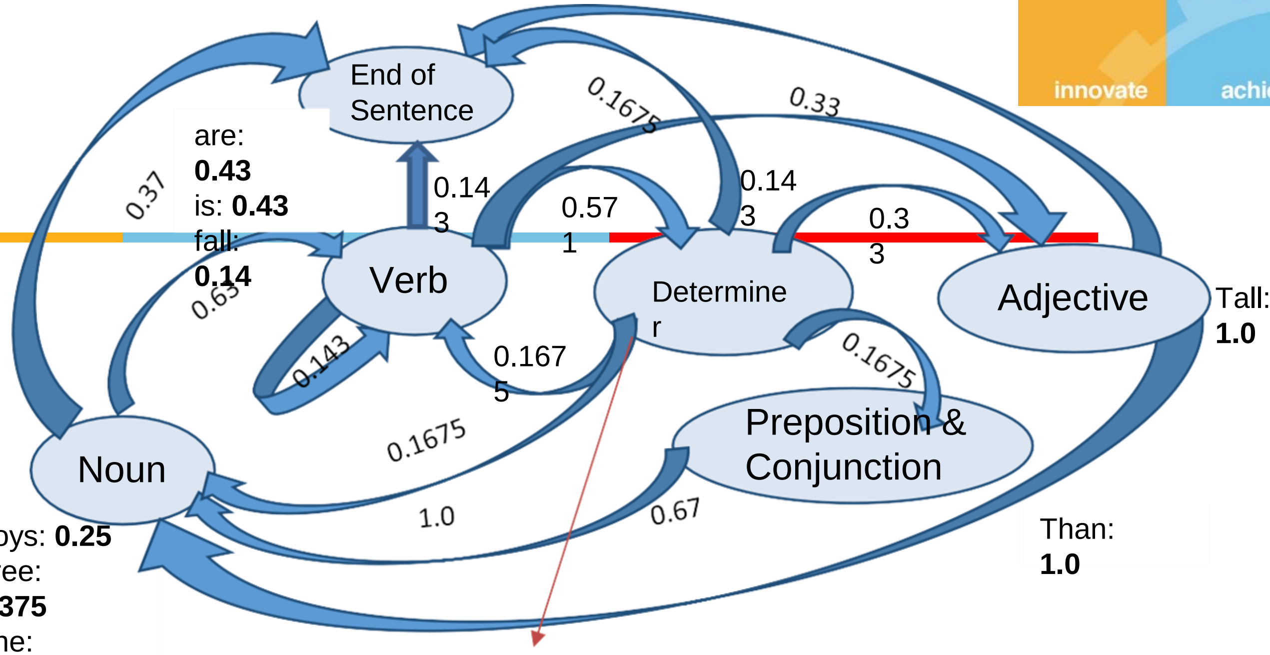
N	D	V	J	A	P	
0.25						Boys
		0.43				Are
			1			Tall
	0.17					This
		0.43				Is
	0.33					The
0.375						Tree
0.125						She
	0.17					A
0.25						Girl
	0.33					More
					1	Than
		0.14				fall

Exercise :

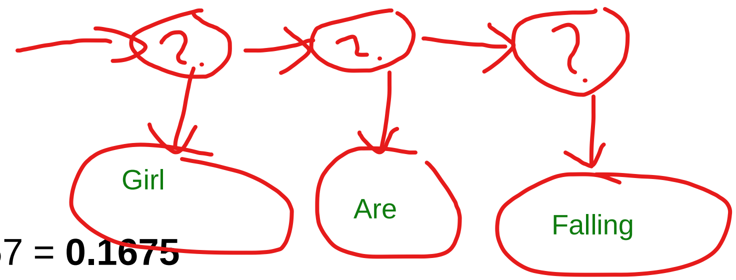
For the below test data/sentence, using the tables constructed using training data, predict the PoS tags.

"Girls are falling"

Initi	
N	0.67
D	0.33
V	0
J	0
A	0
P	0



"Girls are falling"



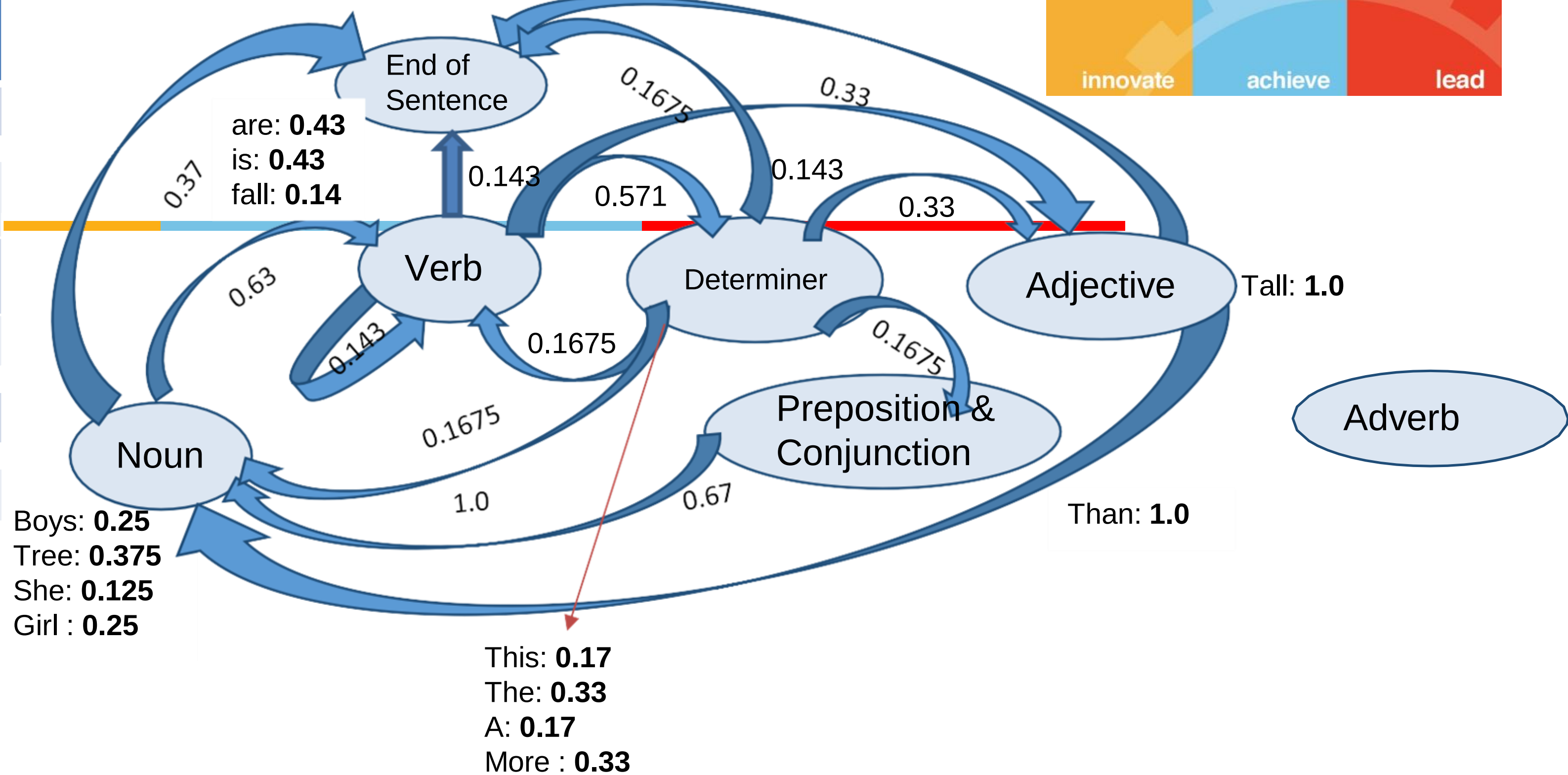
P(Girls , Noun) = P(Girl | Noun) * P(Noun | StartState) = 0.25 * 0.67 = **0.1675**

P(Girls , Verb) = P(Girl | Verb) * P(Verb | StartState) = 0 * 0 = 0 (Ideally with better corpus and the KB , for
most cases it might not be 0 but too low like 0.00000000000…..001.)

P(Girls , Determiner) = P(Girls , Adverb) = P(Girls , Adjective) =P(Girls , Preposition/Conjunction) = 0

StartState → Noun

Initi	
N	0.67
D	0.33
V	0
J	0
A	0
P	0



"Girls are falling"

If Sequence is StartState → Noun

P(are , Verb)

= P(are| Verb) * P(Verb | Noun) * **P(Girls | Noun)** * **P(Noun | StartState)**

= 0.43 * 0.63 * **0.1675**

= 0.04537

P(are, Noun) = P(are| Noun) * P(Noun | Noun) * **Noun** = 0 * 0 = 0

P(are, Determiner) = P(are, Adverb) = P(are , Adjective) =P(are , Preposition/Conjunction) = 0

StartState →Noun →Verb

If Sequence is StartState →Verb

P(are , Verb)

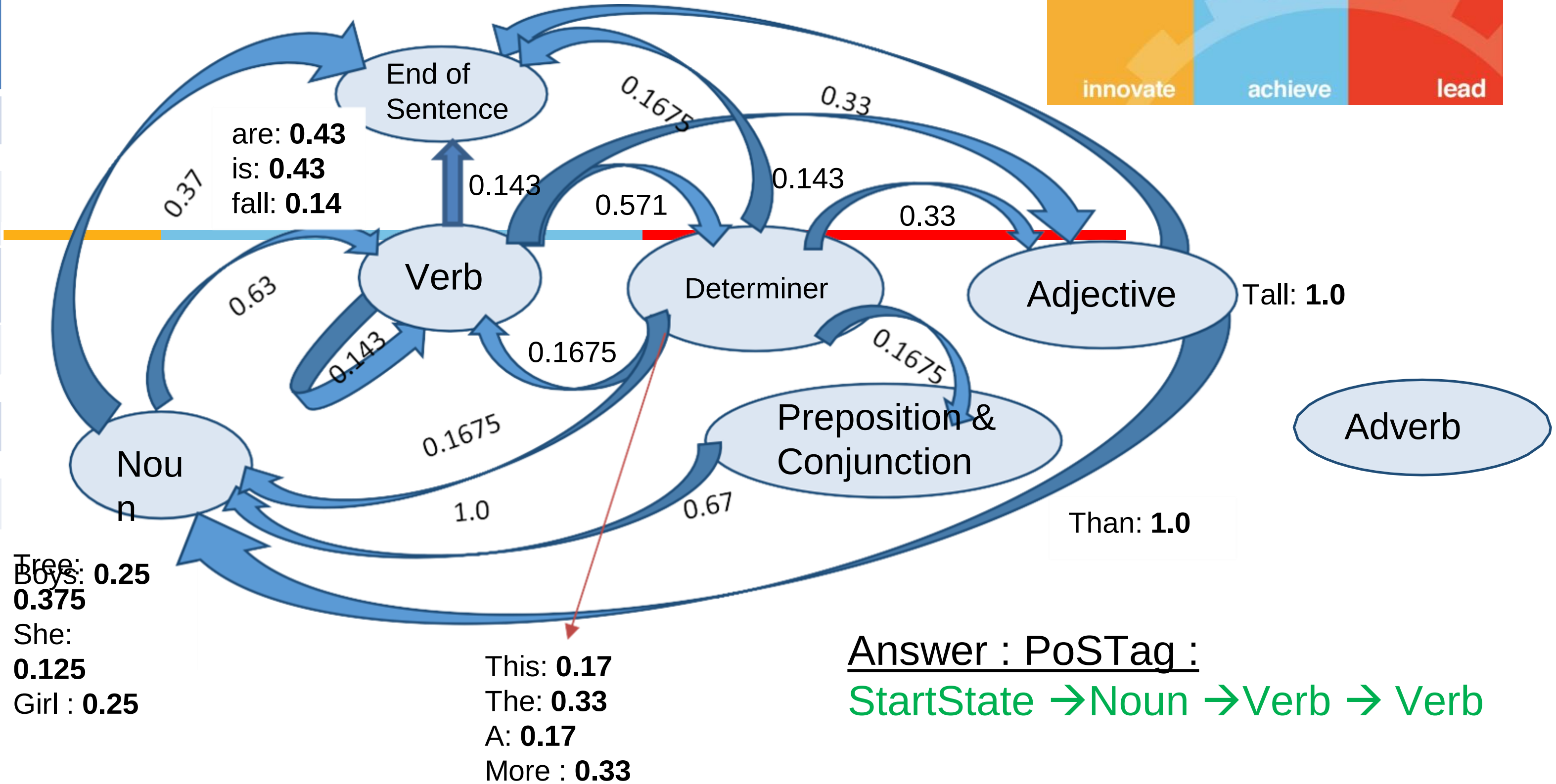
= P(are| Verb) * P(Verb | Verb) * **P(Girls | Verb)** ***** **P(Verb | StartState)**

= 0.43 * 0.143 * **0 = 0**

P(are, Noun) = P(are| Noun) * P(Noun | Noun) * **Verb**= 0 * 0 = 0

P(are, Determiner) = P(are, Adverb) = P(are , Adjective) =P(are , Preposition/Conjunction) = 0

Initi	
N	0.67
D	0.33
V	0
J	0
A	0
P	0



"Girls are falling"

<u>Answer : PoSTag :</u>

StartState →Noun →Verb → Verb

If Sequence is StartState → Noun→Verb

P(falling, Verb)

= P(falling|Verb)*P(Verb|Verb)***P(are| Verb) * P(Verb | Noun)** * **P(Girls | Noun) * P(Noun | StartState)**

= 0.14 * 0.143 * **0.04537**

= 0.000908

P(are, Noun) = P(are| Noun) * P(Noun | Noun) * **Noun** = 0 * 0 = 0

P(are, Determiner) = P(are, Adverb) = P(are , Adjective) =P(are , Preposition/Conjunction) = 0

If Sequence is StartState →Verb →Adjective

P(falling , Verb)

= P(falling | Verb) * P(Verb | Verb) * P(are| Adjective) * P(Adjective | Verb) * P(Girls | Verb)

* P(Verb | StartState)

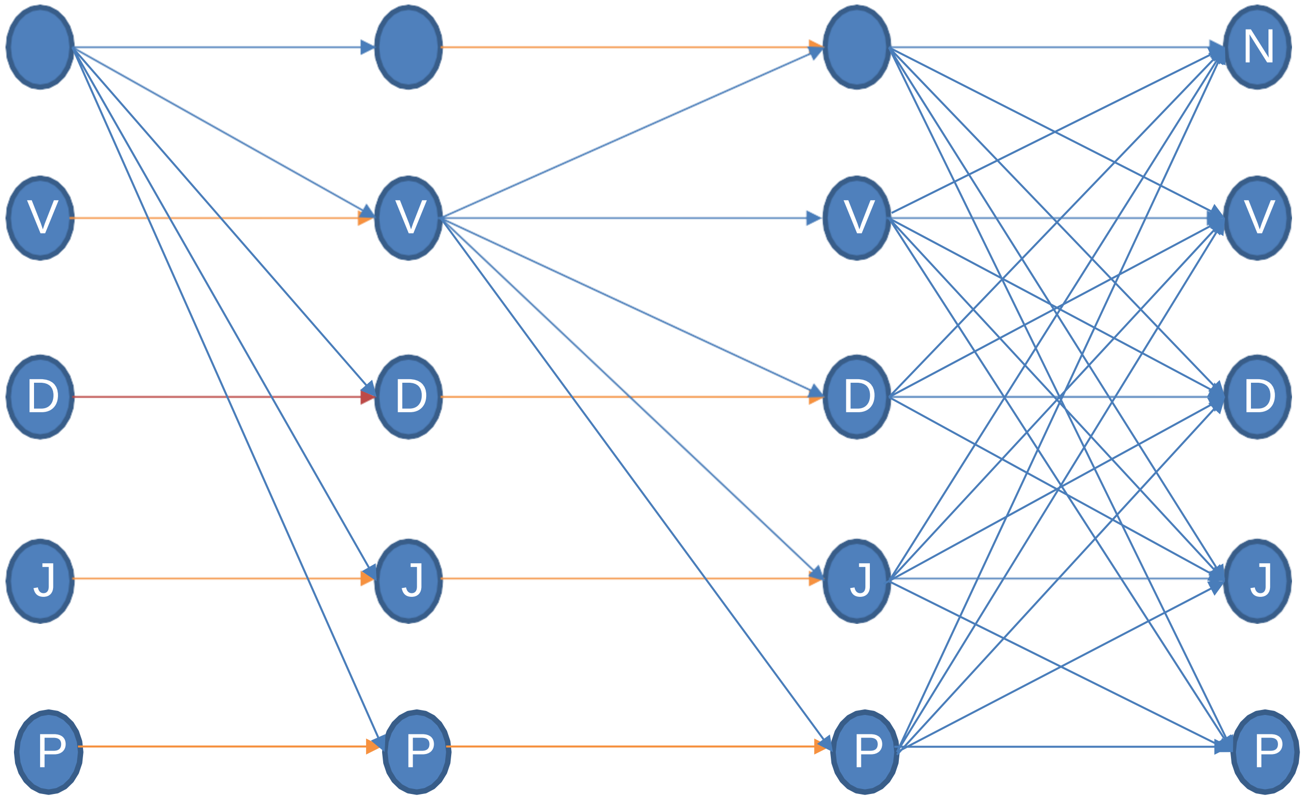
= 0.14 * 0.143 * **0 = 0**

P(falling, Noun) = P(falling| Noun) * P(Noun | Adjective) * **Verb**= 0 * 0 = 0

P(falling, Determiner) = P(falling, Adverb) = P(falling , Adjective) =P(falling , Preposition/Conjunction) = 0

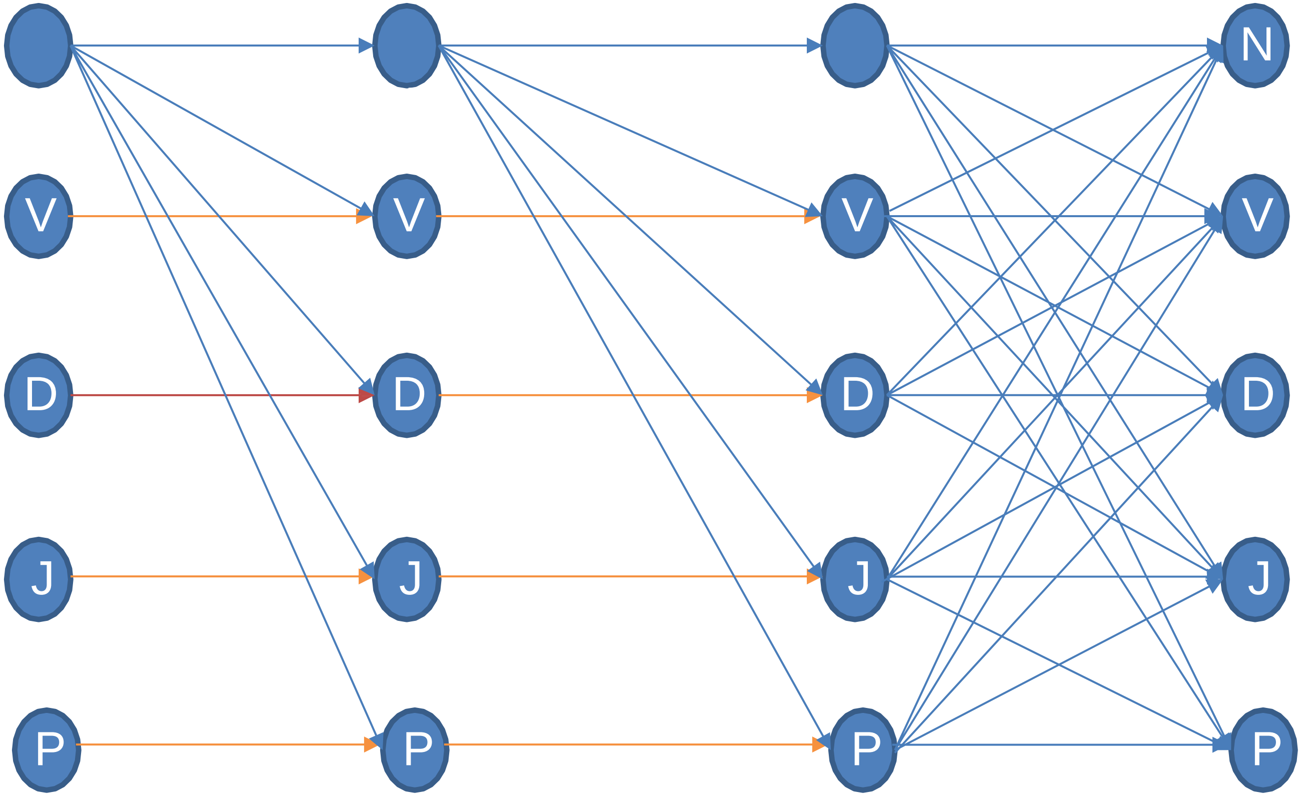
Sample Sequence under Test: Start →Noun →Verb →……….

Assume Noun→Verb is the maximum Value



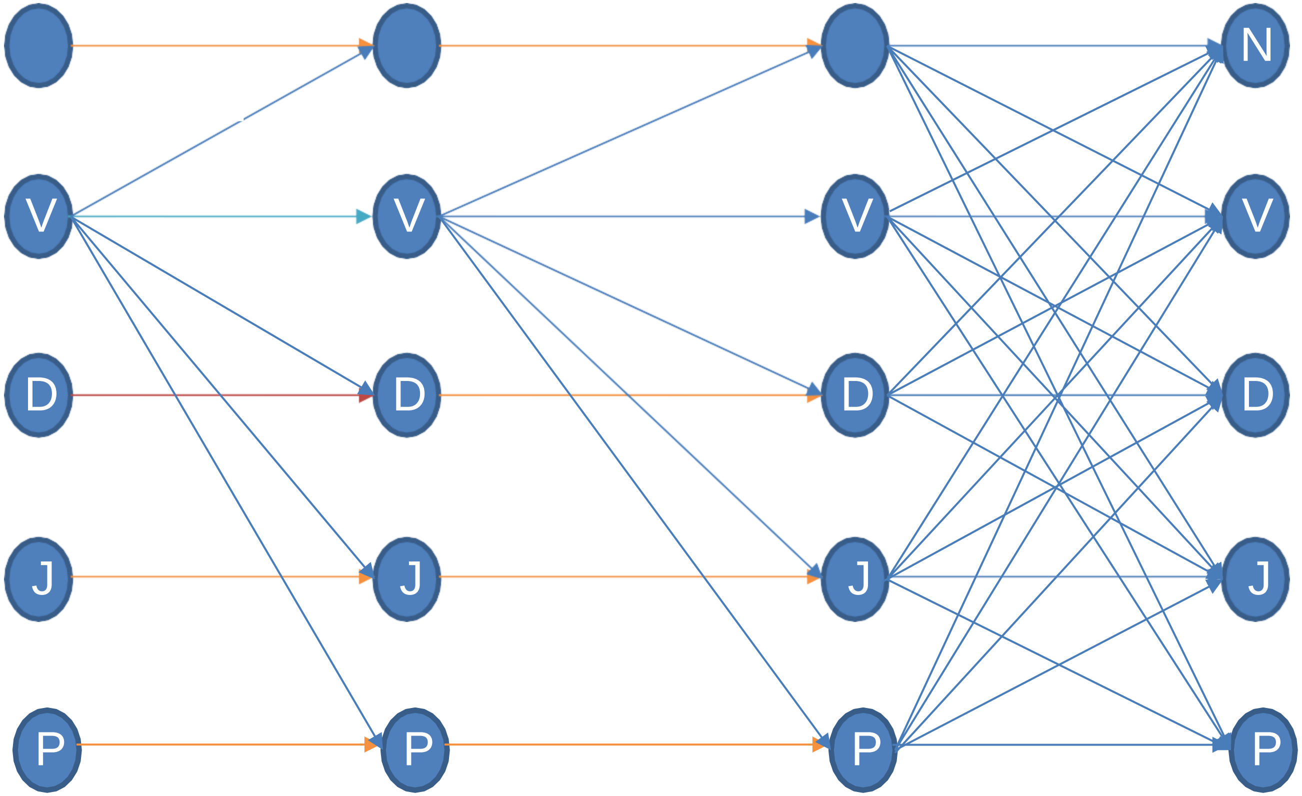
Sample Sequence under Test: Start →Noun →Noun→ ……….

Assume Noun→Noun is the maximum Value



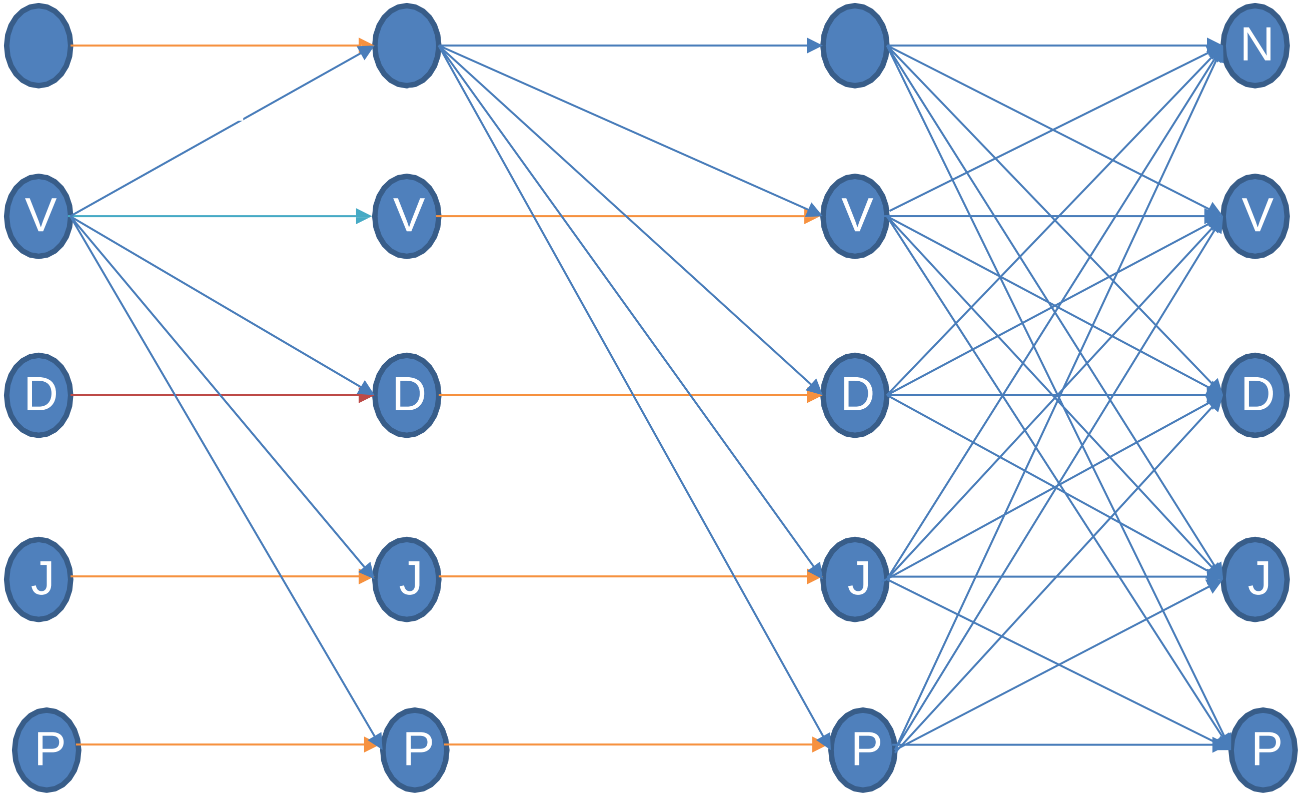
Sample Sequence under Test: Start →Verb→Verb → ……….

Assume Verb→Verb is the maximum Value



Sample Sequence under Test: Start →Verb→Noun→……….

Assume Verb→Noun is the maximum Value



Learning HMM Parameters

Parameter Estimation by EM Algorithm

(Baum-Welch re-estimation procedure)

Parameter Estimation

innovate achieve lead

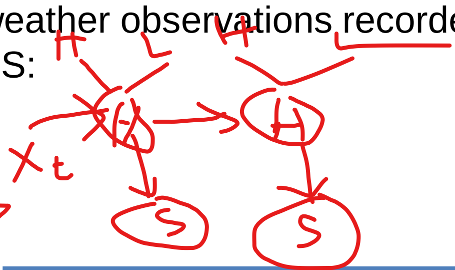
Learning Approach

Baum-Welch re-estimation procedure: Backward Propagation Algorithm

Given an observation sequence O(Evidence) and the set of possible states in the HMM, learn the HMM parameters A(Transition) and B(Emission).

Given set of weather observations recorded estimate the PARAMATERS:

{SS, SR, RR}



P(H).P(S|H).P(H|L).P(S|L)

	HH	HL	LH	LL
SS	**(0.5).(0.6).(0.8)(0.6) = 0.1440** P(H).P(S|H).P(H|H).P(S|H)	0.0120	0.03	0.01
SR	**0.0960**	0.048	0.02	0.04
RR	0.064	0.032	0.12	**0.16**
Total	0.304	0.092	0.17	0.21

Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probabi

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

Parameter Estimation

Learning Approach

Baum-Welch re-estimation procedure: Backward Propagation Algorithm

Given an observation sequence O(Evidence) and the set of possible states in the HMM, learn the HMM parameters A(Transition) and B(Emission).

Given set of weather observations recorded estimate the PARAMATERS:

{SS, SR, RR}

	HH	HL	LH	LL	Best Seq	P(Best)
SS	**0.1440**	0.0120	0.03	0.01	HH	0.144
SR	**0.0960**	0.048	0.02	0.04	HH	0.096
RR	0.064	0.032	0.12	**0.16**	LL	0.16
Total	0.304	0.092	0.17	0.21		0.4
Normalize	0.76	0.23	0.425	0.525		

normalized

t

HP	LP	
0.232323232	0.5526316	LP
0.767676768	0.4473684	HP

t+1

New transition table

Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probabi

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

Parameter Estimation

innovate achieve lead

Learning Approach

Baum-Welch re-estimation procedure: Backward Propagation Algorithm

Given an observation sequence O(Evidence) and the set of possible states in the HMM, learn the HMM parameters A(Transition) and B(Emission).

Find set of weather observations recorded estimate the parAMATERS:

{SS, SR, RR}

LP	HP	
0.615384615	0.4	R
0.384615385	0.6	S

	H□S	L□S	H□R	L□R	Best Seq	P(Seq)
SS	**0.1440**	0.01			HH	0.144
SR	0.0960	0.04	0.096	0.048	HH	0.096
RR			0.064	0.0320	LL	0.16
Total	0.24	0.05	0.16	0.08		
Normalize	0.6	0.125	0.4	0.2		

Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probabi

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

Parameter Estimation

Learning Approach

Baum-Welch re-estimation procedure: Backward Propagation Algorithm

Given an observation sequence O(Evidence) and the set of possible states in the HMM, learn the HMM parameters A(Transition) and B(Emission).

Find set of weather observations recorded estimate the parAMATERS:

{SS, SR, RR}

After this step for the second iteration
Use the optimized tables
(Initial, Transition , Emission)
and repeat the algorithm till convergence

	Start(H)	Start(L)	Best Seq	P(Best)
SS	**0.1440**	0.03	HH	0.144
SR	0.0960	0.04	HH	0.096
RR	0.064	0.16	LL	0.16
	0.304	0.23		
Normalize	0.76	0.575		

HP	LP
0.56929	0.4307

Transition Model / Probability Matrix

$P(U_{t-1} = HP)$	$P(U_{t-1} = LP)$	←Previous
0.2	0.5	$P(U_t = LP)$
0.8	0.5	$P(U_t = HP)$

Evidence / Sensor Model/ Emission Probability

$P(X_t = LP)$	$P(X_t = HP)$	←Unobserved Evidence v
0.8	0.4	$P(E_t = Rainy)$
0.2	0.6	$P(E_t = Sunny)$

HMM in Prevention of Network Security Threat

(Interesting Case Studies)

Hidden Morkov Model

Cyber Security

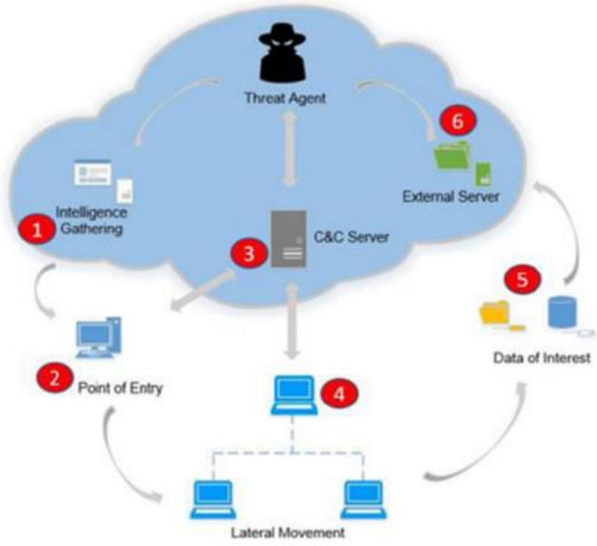
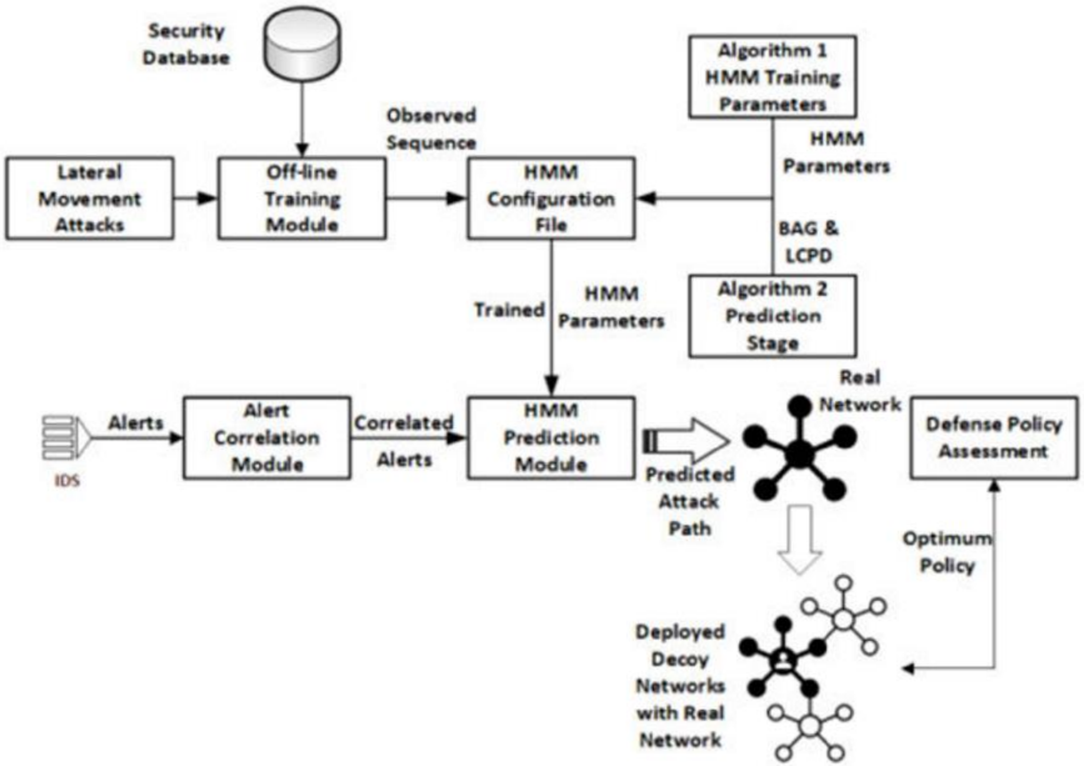


FIGURE 1. **Typical stages of APT attack.**

Source Credit : [2021 : Hidden Markov Model and Cyber Deception for the Prevention of Adversarial Lateral Movement]

Hidden Morkov Model

Cyber Security



Source Credit : [2021 : Hidden Markov Model and Cyber Deception for the Prevention of Adversarial Lateral Movement]

Hidden Morkov Model

Cyber Security

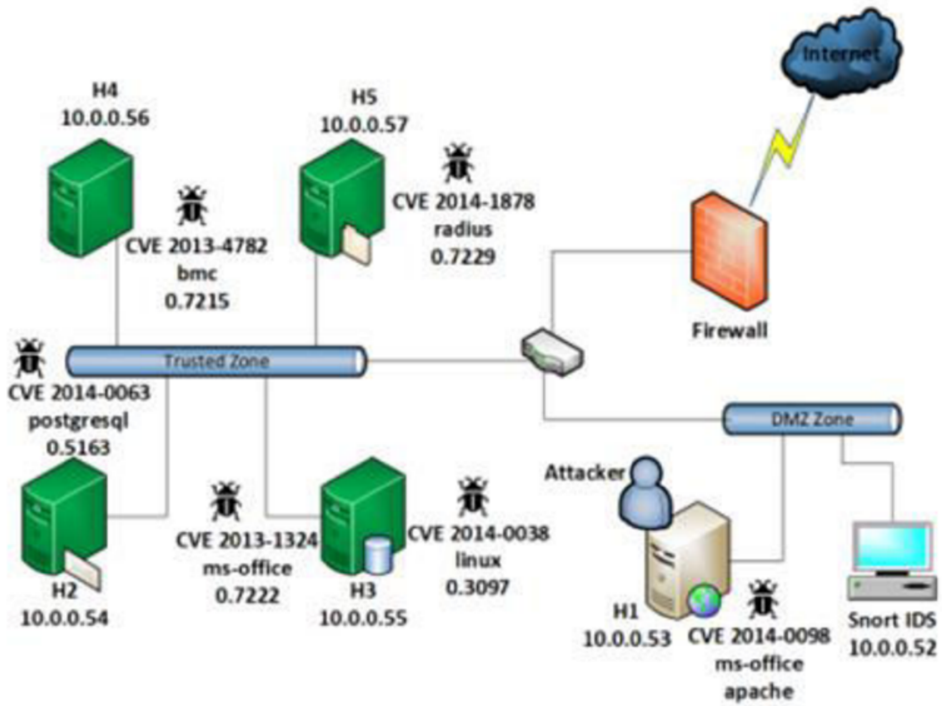


FIGURE 9. Experimental network topology.

Source Credit : [2021 : Hidden Markov Model and Cyber Deception for the Prevention of Adversarial Lateral Movement]()

Hidden Morkov Model

Cyber Security

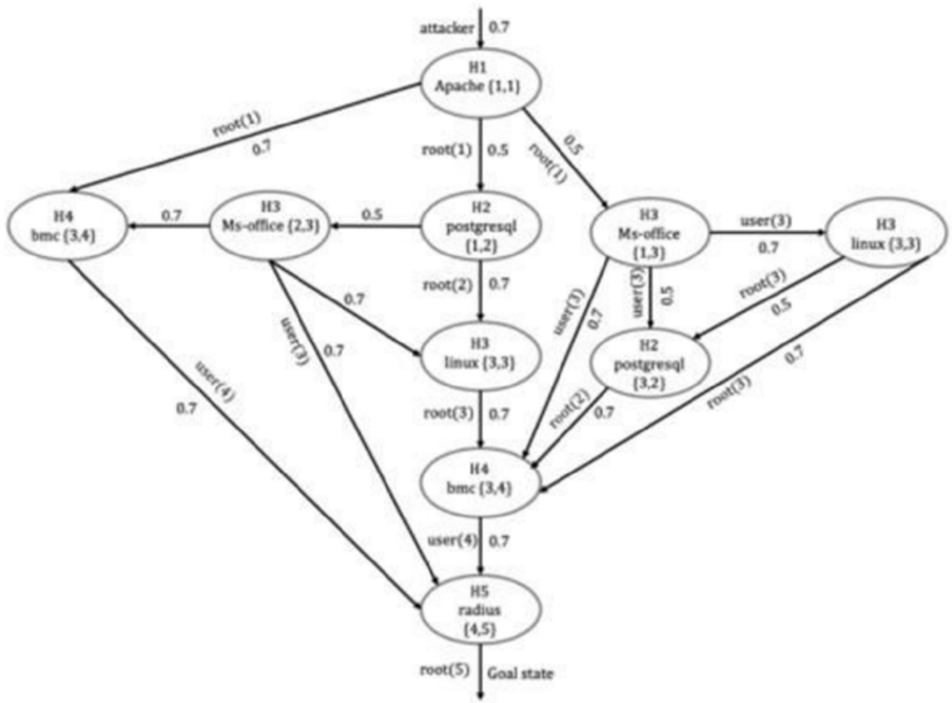


FIGURE 10. Attack graph of the experimental network.

Attack states description.

State	Description
S_1	Initial State
S_2	(H_1,root)
S_3	(H_2,root)
S_4	(H_3,user)
S_5	(H_3,root)
S_6	(H_4,user)
S_7	(H_5,root)

Source Credit : [2021 : Hidden Markov Model and Cyber Deception for the Prevention of Adversarial Lateral Movement]

Hidden Morkov Model

Cyber Security

Attack states description.

State	Description
S_1	Initial State
S_2	$(H_1$,root)
S_3	$(H_2$,root)
S_4	$(H_3$,user)
S_5	$(H_3$,root)
S_6	$(H_4$,user)
S_7	$(H_5$,root)

TABLE 6. Possible attack paths.

Path Number	Attack Path
1	$S_1 \rightarrow S_2 \rightarrow S_6 \rightarrow S_7$
2	$S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_4 \rightarrow S_7$
3	$S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_6 \rightarrow S_7$
4	$S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_4 \rightarrow S_5 \rightarrow S_6 \rightarrow S_7$
5	$S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_5 \rightarrow S_6 \rightarrow S_7$
6	$S_1 \rightarrow S_2 \rightarrow S_4 \rightarrow S_6 \rightarrow S_7$
7	$S_1 \rightarrow S_2 \rightarrow S_4 \rightarrow S_3 \rightarrow S_6 \rightarrow S_7$
8	$S_1 \rightarrow S_2 \rightarrow S_4 \rightarrow S_5 \rightarrow S_3 \rightarrow S_6 \rightarrow S_7$
9	$S_1 \rightarrow S_2 \rightarrow S_4 \rightarrow S_5 \rightarrow S_6 \rightarrow S_7$

Source Credit : [2021 : Hidden Markov Model and Cyber Deception for the Prevention of Adversarial Lateral Movement]

Ethics in Artificial Intelligence

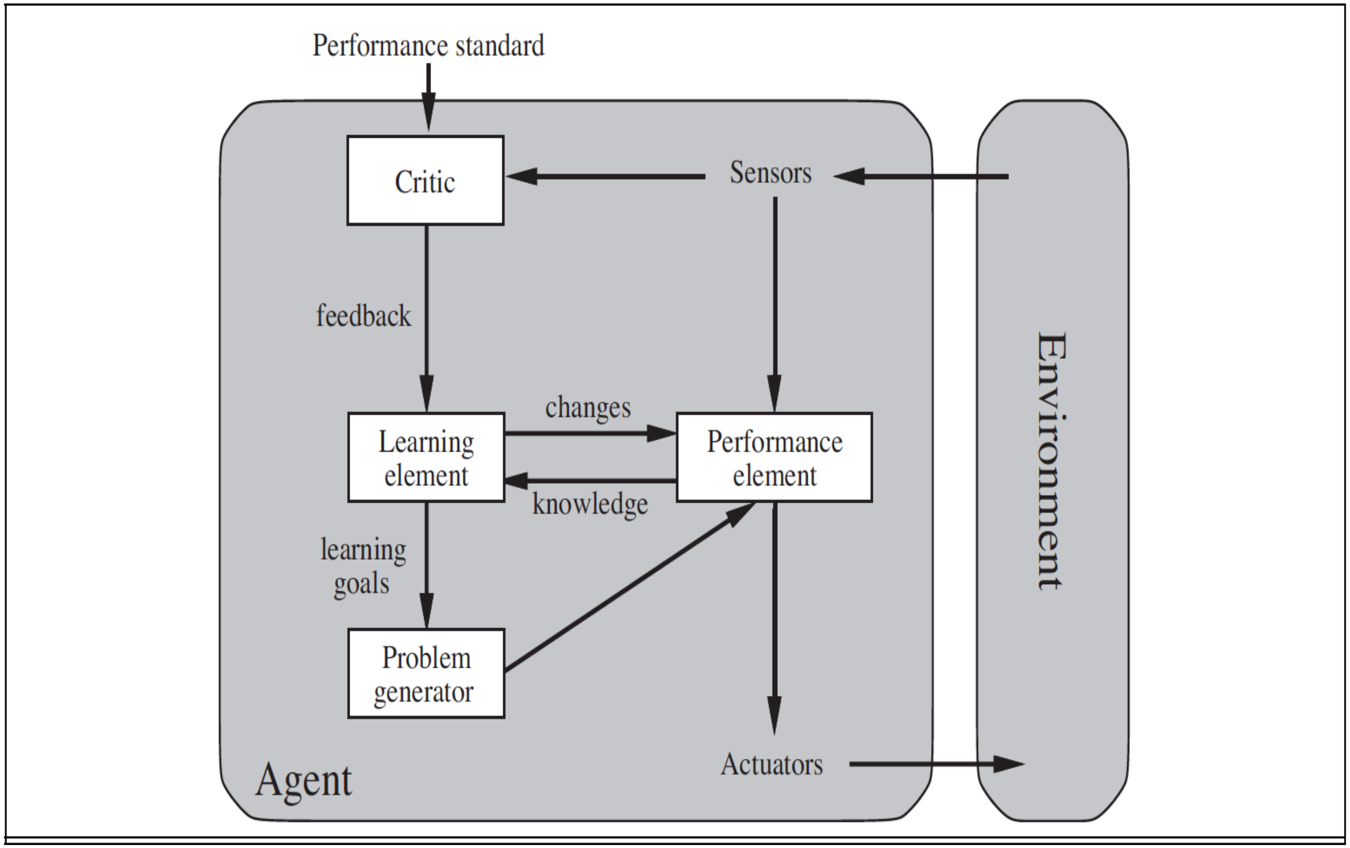
Module 7 : Ethics in AI

A. Explainable AI

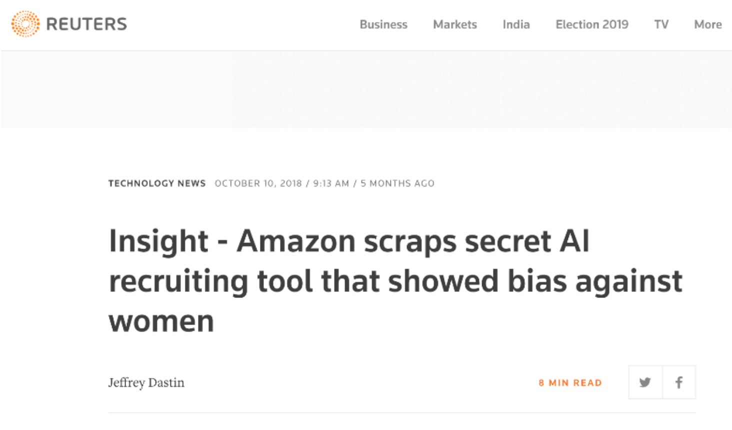
B. Logically Explained Network

C. Explainable Bayesian Network

Shortcomings of AI


Performance standard
Critic
Sensors
feedback
changes
Learning element
Performance element
knowledge
learning goals
Problem generator
Actuators
Agent
Environment

Recommendation System


REUTERS Business Markets India Election 2019 TV More

TECHNOLOGY NEWS OCTOBER 10, 2018 / 9:13 AM / 5 MONTHS AGO

Insight - Amazon scraps secret AI recruiting tool that showed bias against women

Jeffrey Dastin 8 MIN READ

Amazon's Edinburgh engineering hub's goal was to develop AI that could rapidly crawl the web and spot candidates worth recruiting

Fairness : The absence of bias towards an individual or a group

Are the predictions _____ ?

- Fair
- Unbiased

Object Recognition System

Forbes Billionaires Innovation Leadership Money Consumer Industry

44,931 views | Jul 1, 2015, 01:42pm

Google Photos Tags Two African-Americans As Gorillas Through Facial Recognition Software

Maggie Zhang Forbes Staff

I write about technology, innovation, and startups.

-

Are the Inferences _____ ?

- Correct
- Unbiased

Are the Predictions _____?

- Fair
- Universally Applicable

Natural Language Processing system

Microsoft silences its new A.I. bot Tay, after Twitter users teach it racism [Updated]

Sarah Perez

@sarahintampa / 3 years ago

Comment

Tay is Microsoft's conversational bot powered by NLP & ML.

Are the interpretations _____ ?

- Fair
- Legal
- Socially Ethical

Building a Fair Model

No artificial model is a perfect one. But every model significantly influence the social , economic, cultural ethics impacting humanity.

Justify the design modelled & metric used to validate the model, is in fact the right choices fit in the context.

1. Is it fair to make an AI-ML system?
2. Is there a better technical approach to convert an existing AI system fair?
3. Are the results obtained by the AI system fair?

Interpretable Models

Are the results obtained by the AI system fair?

Interpretable models helps to trust the AI system by answering transparently to the specific questions like “Why the system is behaving under certain scenarios?”

- If a loan gets rejected, do we know the reasons?
- If a job application is accepted, is it biased towards a gender?
- If a bail is granted to an accused, is it based on their race?
- If a patient is diagnosed with a disease, what factors made the algorithm to classify it?

Interpretable Models

Example Based Explanations:

If SymptomInX ≡ SymptomInY

if DiseaseA infected X

then probably DiseaseB might have infected Y

If CustomerX ≡ CustomerY

if CustomerX purchased P1

then probably CustomerY will purchase P1

Counterfactual Explanations:

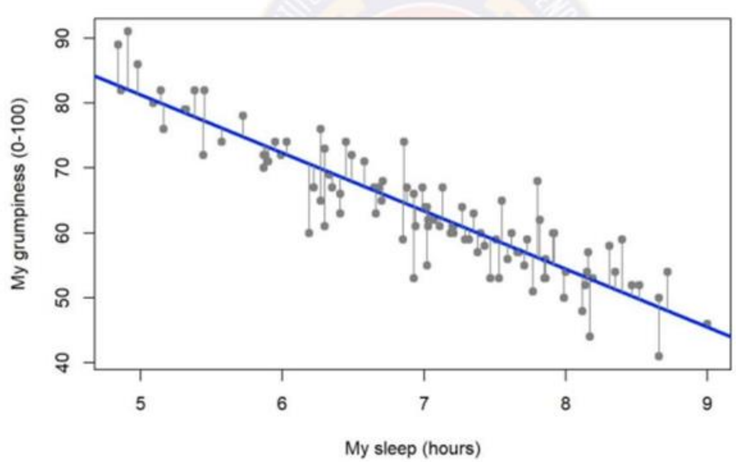
If customerX's income level had not been less than L3

then the customer's Loan might not have been rejected

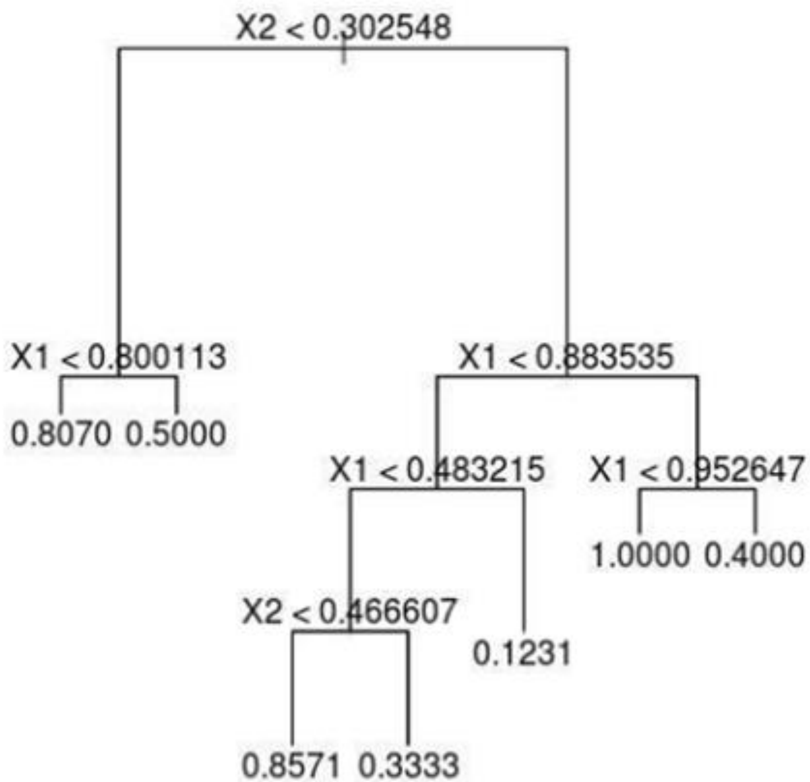
Interpretable Models

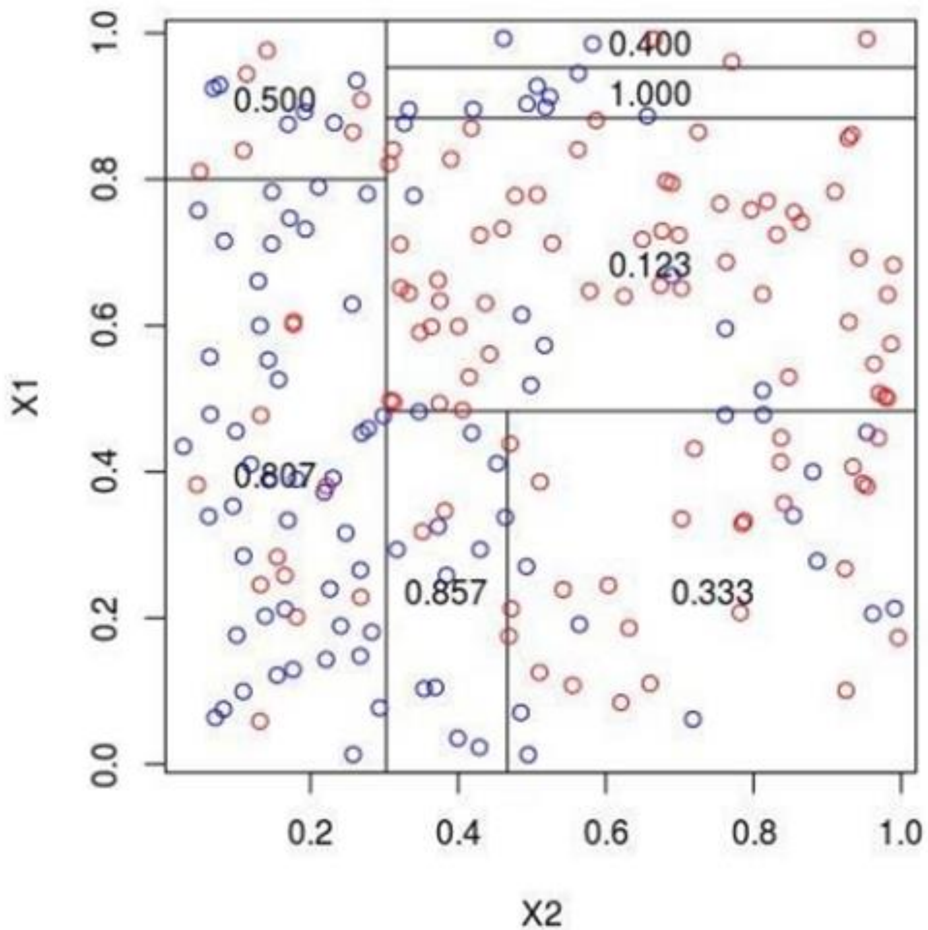
$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i = \mathbf{x}_i^\mathsf{T} \boldsymbol\beta + \varepsilon_i, \qquad i = 1, \ldots, n,$$

interpretable components

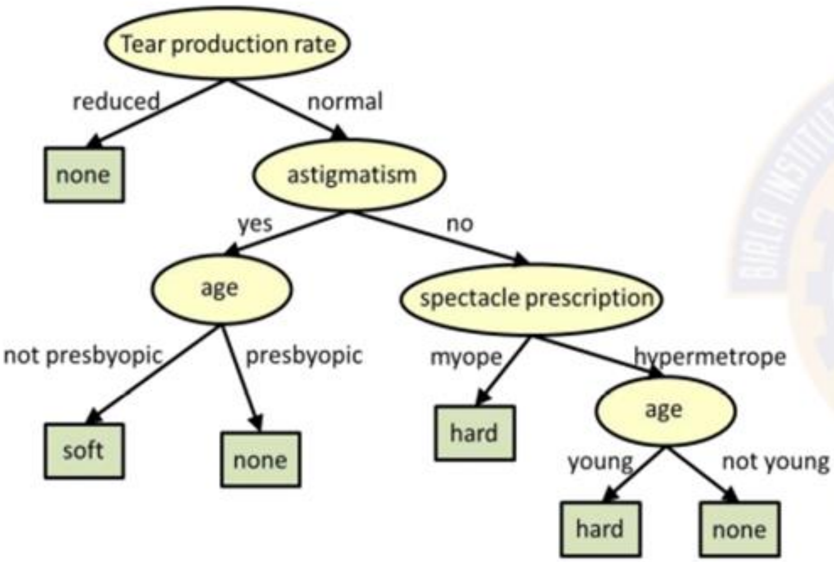


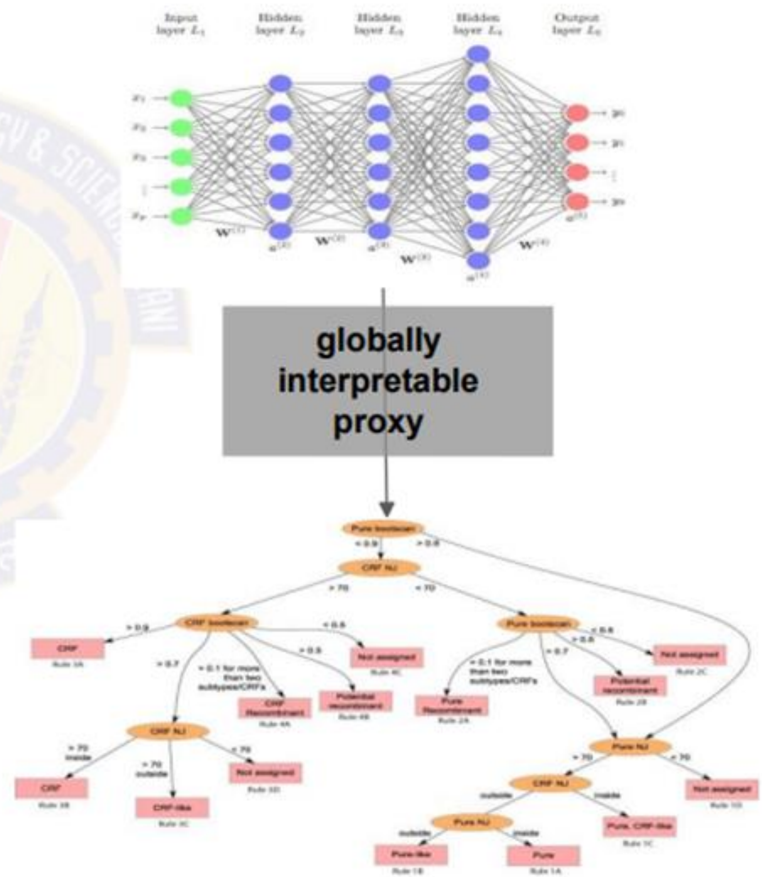
Interpretable Models





Interpretable Models

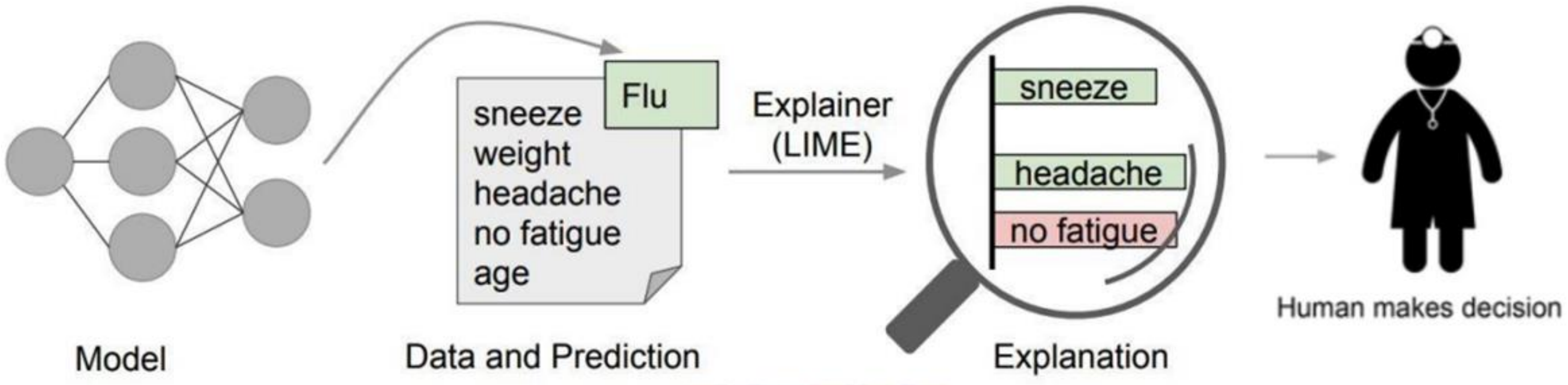




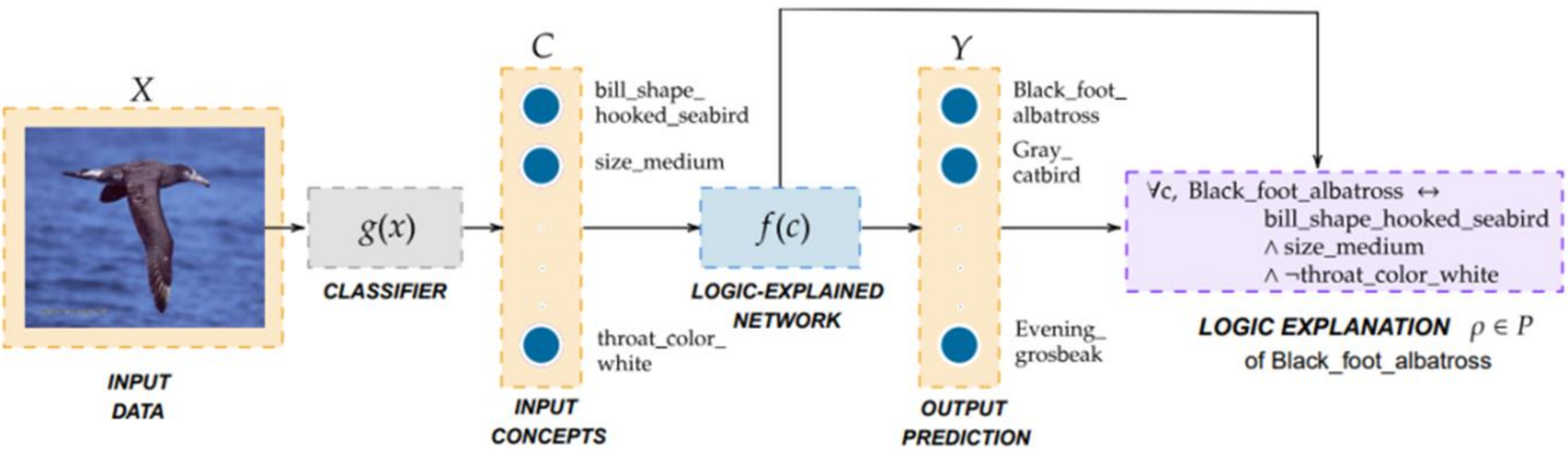
innovate achieve lead

Autonomous AI

Interpretable Models



In Deep Learning



Source Credit: [2021: **Logic Explained Networks**]

XAI

- Bayesian Networks

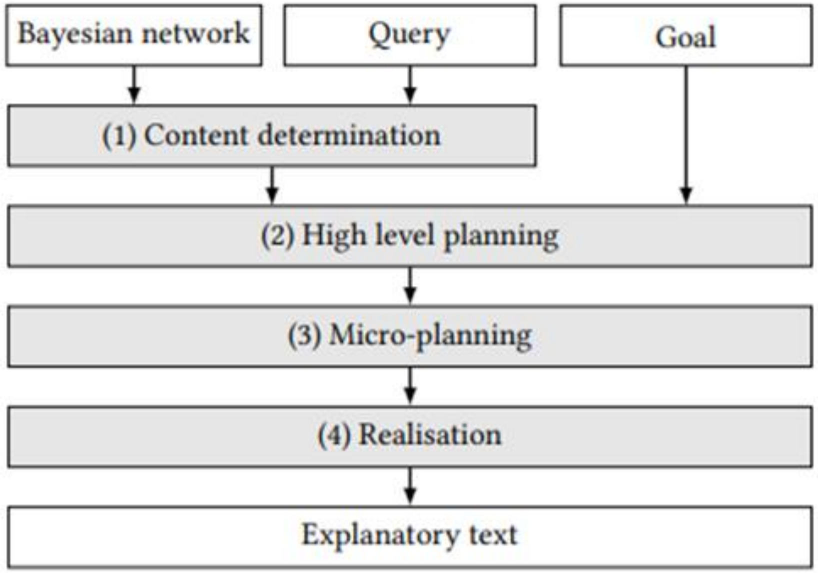
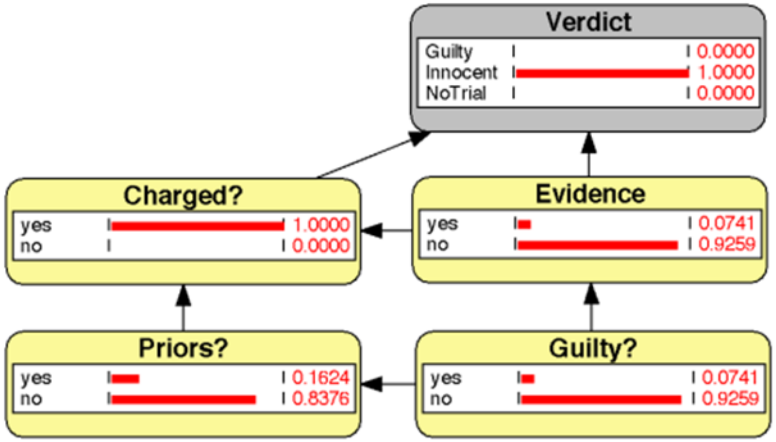


Figure 4: Pipeline architecture of the explanation system

Source Credit :
June 2019: **Explainable Bayesian Network Query Results via Natural Language Generation Systems**
International Conference on Artificial Intelligence and Law
https://doi.org/10.1145/3322640.3326716

XAI

- Bayesian Networks



$Pr(x)$	Label
1	Certain
[0.85, 1)	Almost certain
[0.75, 0.85)	Probable
(0.5, 0.75)	Expected
0.5	Fifty-fifty
[0.25, 0.5)	Uncertain
[0.15, 0.25)	Improbable
(0, 0.15)	Almost impossible
0	Impossible

(a) Verbal-numerical scale expressing probabilities [25]

$Pr(x \mid C, o) - Pr(x \mid C)$	Label
(0.3, 1]	considerable increase
(0.15, 0.3]	substantial increase
(0.05, 0.15]	moderate increase
(0.01, 0.05]	slight increase
(0, 0.01]	inconsequential increase
0	unchanged
[−0.01, 0)	inconsequential decrease
[−0.05, −0.01)	slight decrease
[−0.15. − 0.05)	moderate decrease
[−0.3, −0.15)	substantial decrease
[−1, −0.3)	considerable decrease

(b) Verbal-numerical scale expressing changes in probability

Label	Effect	Description
+4	(0.5, 1]	strongly positive effect
+3	(0.25, 0.5]	moderate positive effect
+2	(0.125, 0.25]	fair positive effect
+1	(0, 0.125]	slight positive effect
0	0	neutral
−1	[−0.125, 0)	slight negative effect'
−2	[−0.25, −0.125)	fair negative effect
−3	[−0.5, −0.25)	moderate negative effect
−4	[−1, −0.5)	strong negative effect

(c) Verbal-numerical scale expressing magnitudes of inference effect

<u>Source Credit :</u>

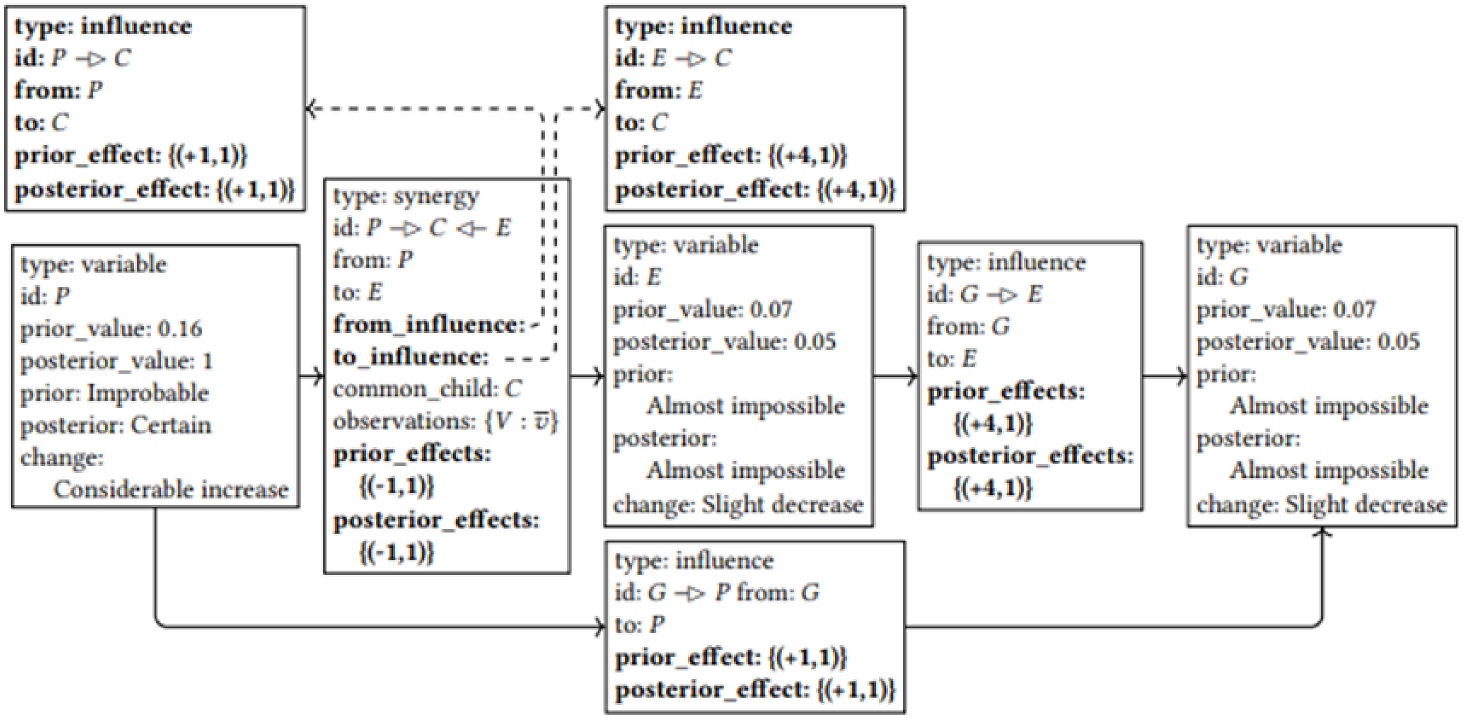
June 2019: **Explainable Bayesian Network Query Results via Natural Language Generation Systems**

International Conference on Artificial Intelligence and Law

https://doi.org/10.1145/3322640.3326716

XAI

➢ Bayesian Networks



Source Credit :
June 2019: **Explainable Bayesian Network Query Results via Natural Language Generation Systems**
International Conference on Artificial Intelligence and Law
https://doi.org/10.1145/3322640.3326716

innovate achieve lead

➢ Bayesian Networks

Variable	Parsed proposition
G	(S (NP (DT *the*) (NN *defendant*)) (VP (VBZ *is*) (ADJP (JJ *guilty*))))
P	(S (NP (DT *the*) (NN *defendant*)) (VP (VBD *committed*) (NP (JJ *prior*) (NNS *offences*))))
E	(S (NP (EX *there*)) (VP (VBZ *is*) (NP (NP (JJ *hard*) (NN *evidence*)) (VP (VBG *supporting*) (NP (DT *the*) (NN *defendant's*) (NN *guilt*))))))
C	(S (NP (DT *the*) (NN *defendant*)) (VP (VBZ *is*) (VP (VBN *charged*))))
V	(S (NP (DT *the*) (NN *defendant*)) (VP (VBZ *is*) (VP (VBN *found*) (S (ADJP (JJ *guilty*))))))

Sentence template 2. Consistently strong effect NP

Result	effect_np(*I*)
Conditions	*I*.type ∈ {influence,synergy}
	Pr(*I*.prior_effects = +4) = 1
	Pr(*I*.posterior_effects = +4) = 1
Template	(NP (DT *a*) (ADJP (RB *consistently*) (JJ *strong*)) (JJ *positive*) (NN *effect*))

(1) *The defendant is found not guilty.*
(2) As a consequence of this, it is certain that *the defendant is charged.*
(3) There are two variables that help explain why *the defendant is charged* as the likelihood of this event increases with the probability that:
(4) *the defendant committed prior offences* and
(5) *there is hard evidence supporting the defendant's guilt.*
(6) Either of these explanations makes the other less necessary to explain that the defendant is charged.
(7) Therefore, an increase in the probability that *the defendant has committed prior offences* has a consistently slight negative effect on the probability that *there is hard evidence supporting the defendant's guilt.*

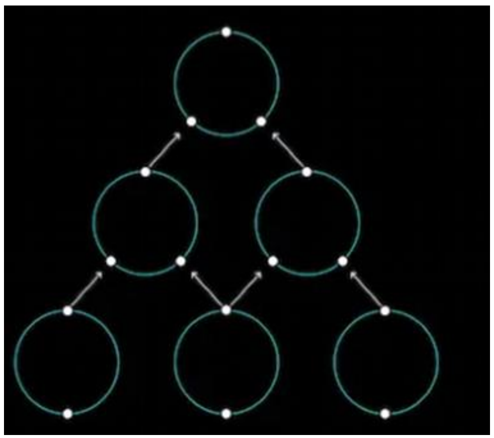
Source Credit :

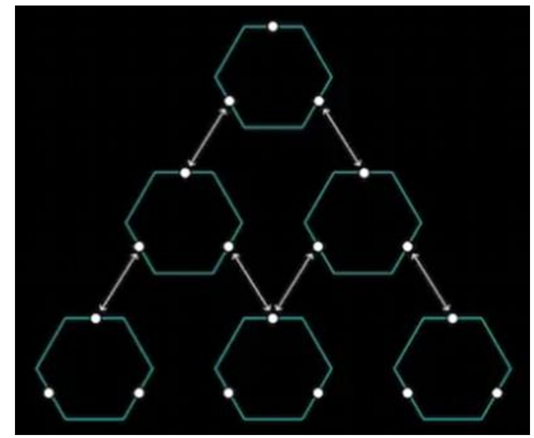
June 2019: **Explainable Bayesian Network Query Results via Natural Language Generation Systems**

International Conference on Artificial Intelligence and Law

https://doi.org/10.1145/3322640.3326716

- Logic based Neural Network (LNN) in KBQA

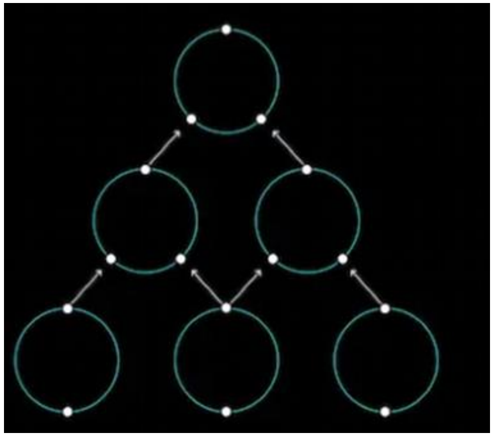


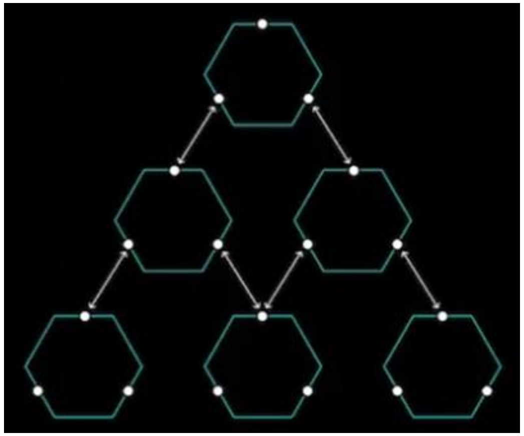


Source Credit :
2020: https://research.ibm.com/blog/ai-neurosymbolic-common-sense

XAI

➢ Logic based Neural Network (LNN) in KBQA

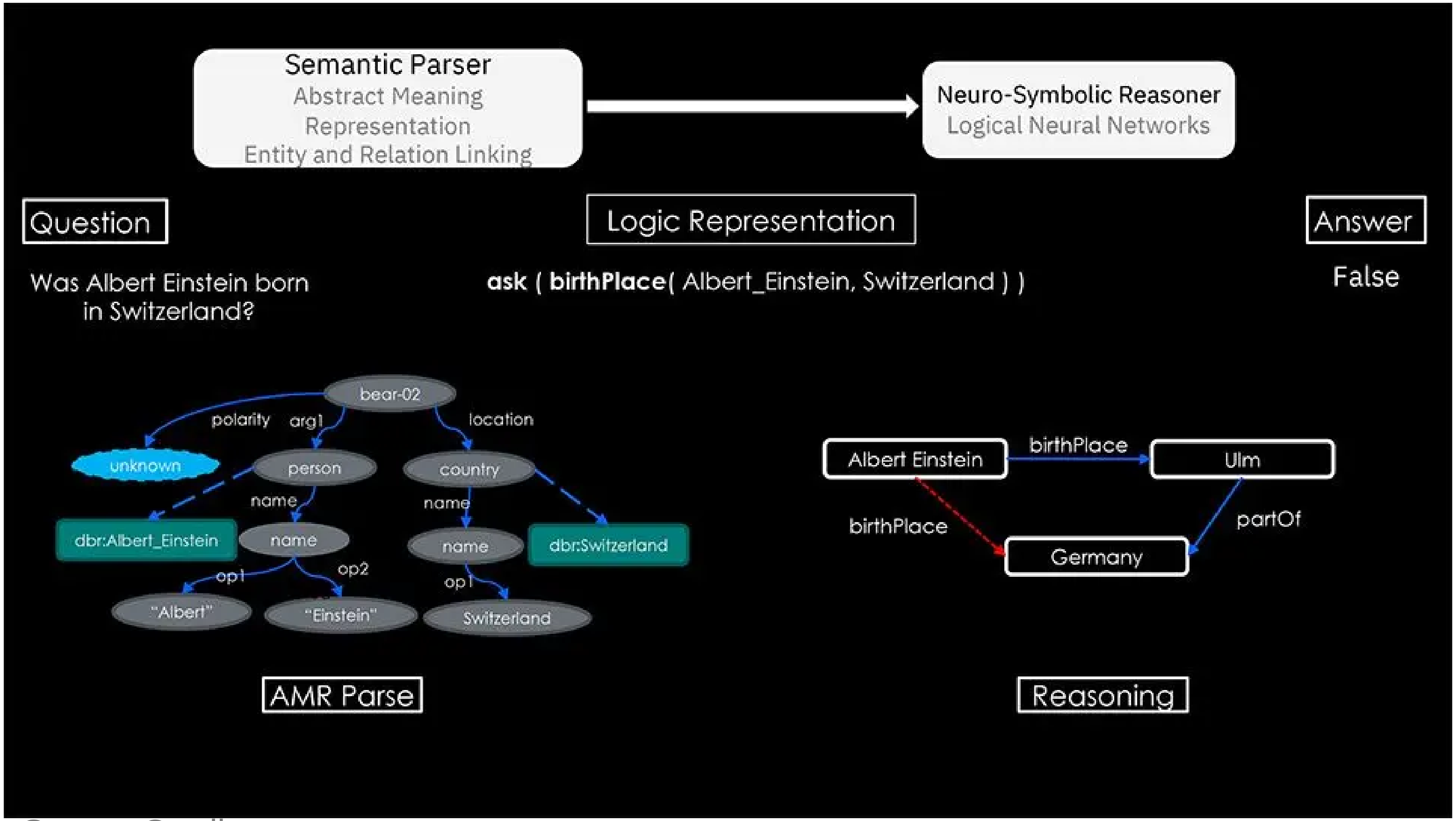




Source Credit :
2020: https://research.ibm.com/blog/ai-neurosymbolic-common-sense

XAI

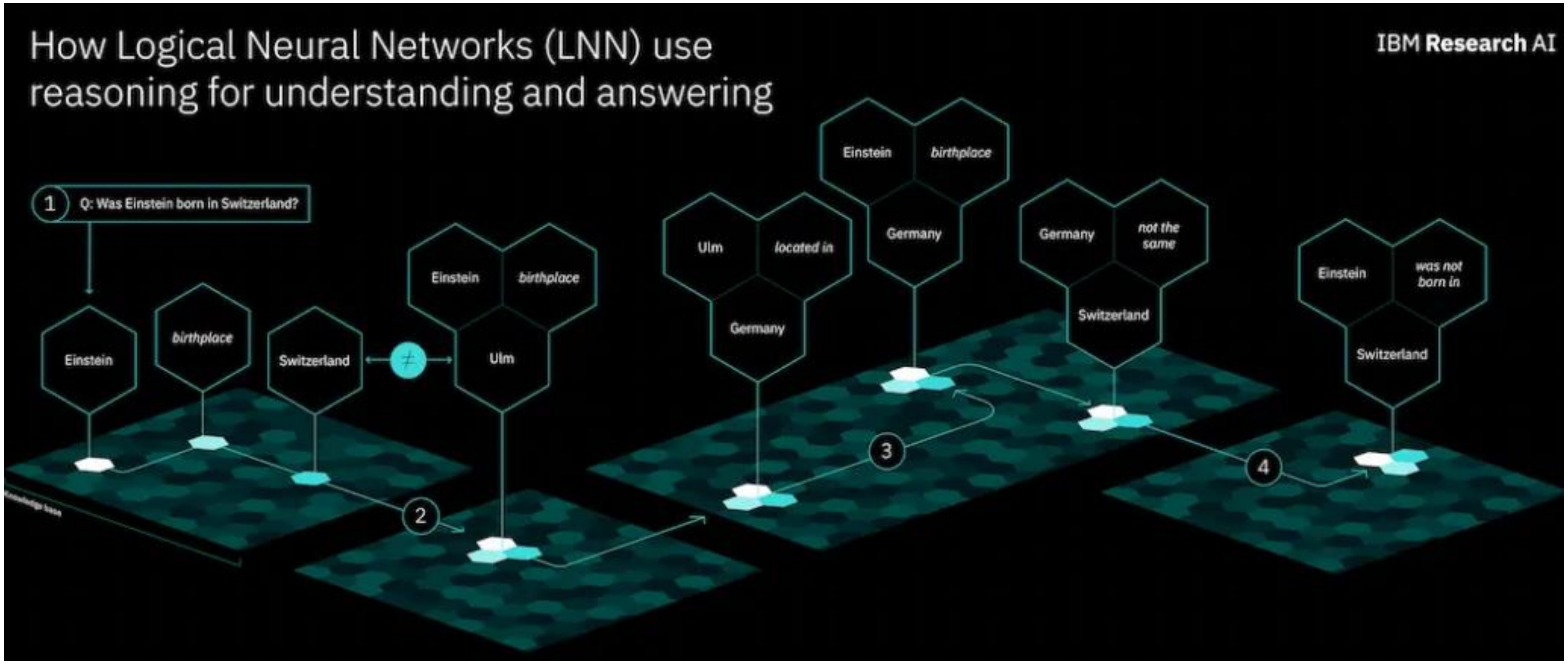
- Logic based Neural Network (LNN) in KBQA



Source Credit :
2020: https://research.ibm.com/blog/ai-neurosymbolic-common-sense

XAI

➢ Logic based Neural Network (LNN) in KBQA



Source Credit :
2020: https://research.ibm.com/blog/ai-neurosymbolic-common-sense

Required Reading: AIMA - Chapter #15.1, #15.2, #15.3, #20.3.3 and references in the document to the external sources

Thank You for all your Attention

Note : Some of the slides are adopted from AIMA TB materials

BITS Pilani
Pilani Campus

Artificial & Computational Intelligence

****** CLZG557**

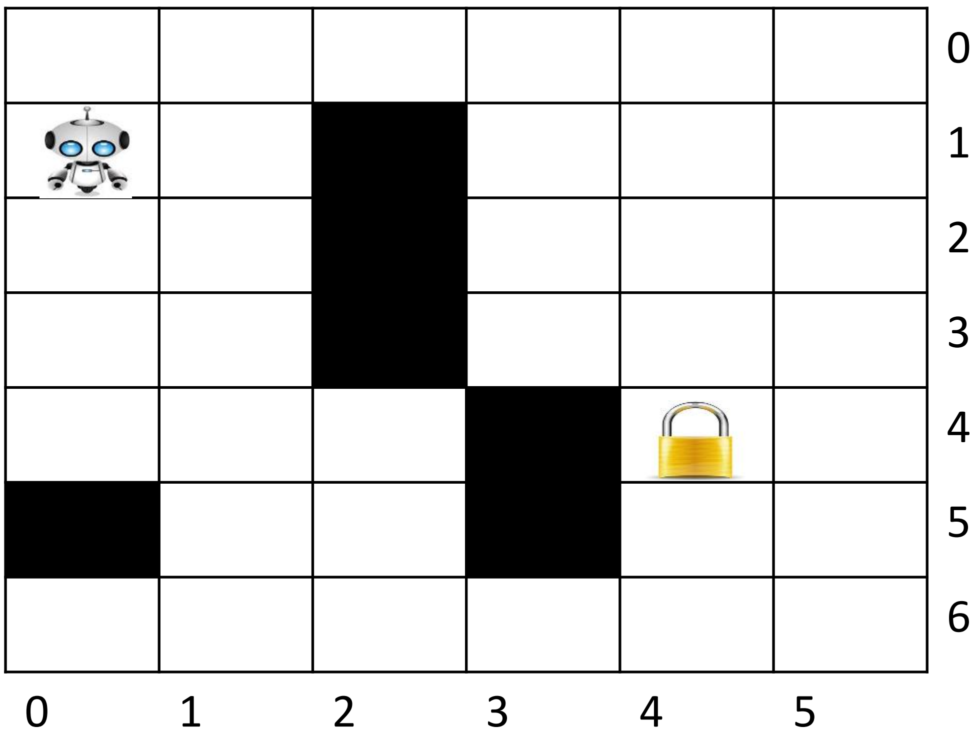
Review

Raja vadhana P

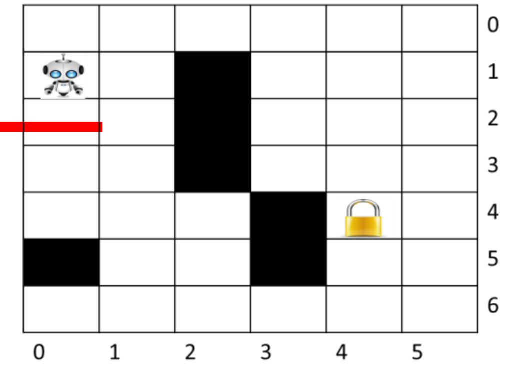
Assistant Professor,

BITS - CSIS

Path finding Robot - Lab Example

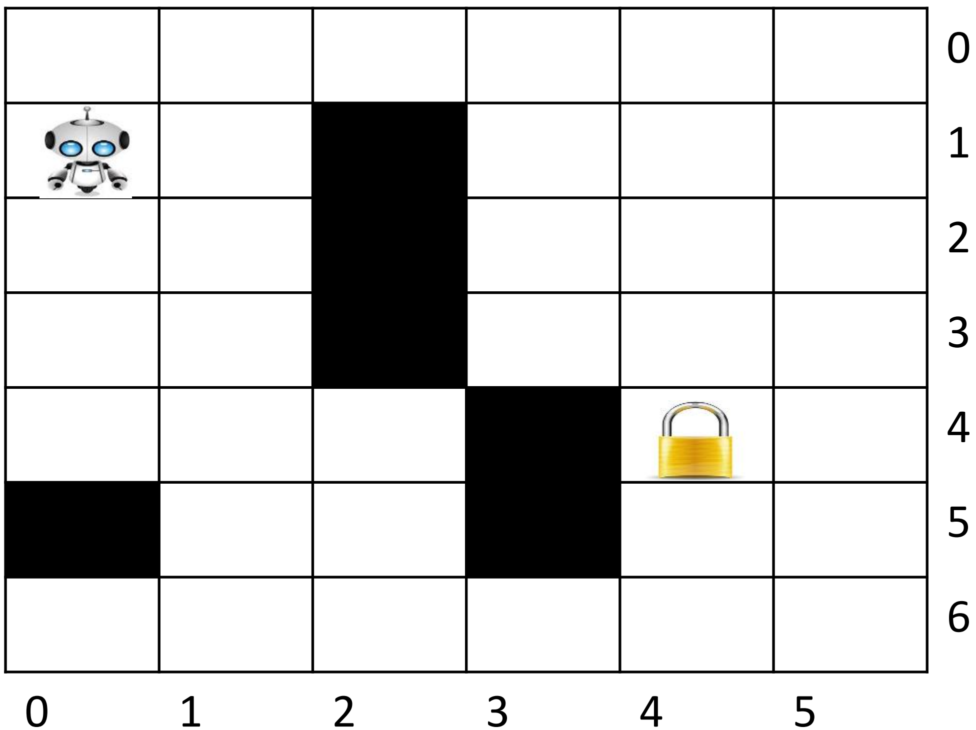


PEAS Environment



Agent
Performance
Environment
Sensors
Actuators

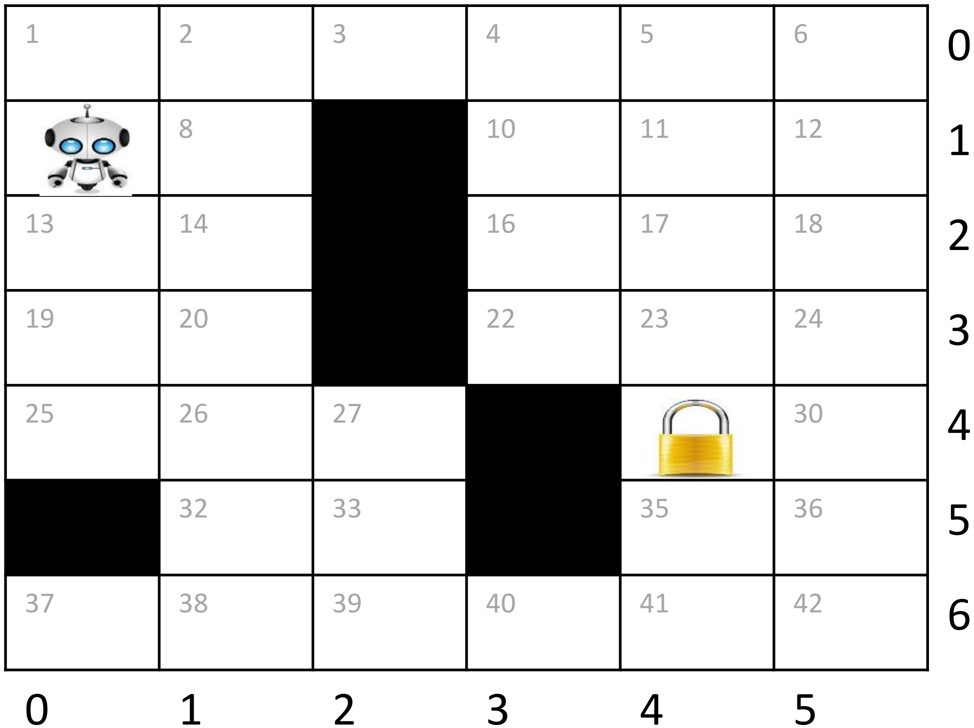
Dimensions of Task Environment



Agent
Observability
No.of.Agents
No.of.States
Determinism
Dynamicity
Output Dependency

Problem Solving Agent Design

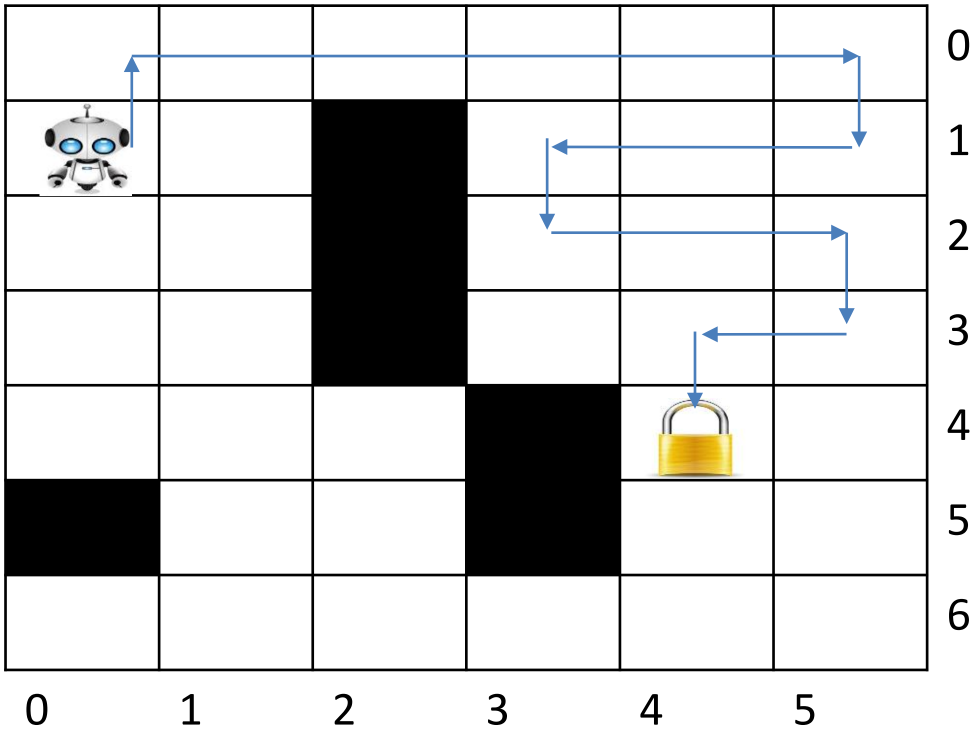
Successor Function Design



N-W-E-S

PSA
Initial State
Actions , Success Function , Transition Model
Path Cost
Goal Test

DFS : – CSIS Virtual Lab Example Results



NWES:
S(1,0) G(4,4)

(1,0)---(0,0)---(0,1)---(0,2)---(0,3)---(0,4)---(0,5)---(1,5)---(1,4)---(1,3)---(2,3)---(2,4)---(2,5)---(3,5)---(3,4)---(4,4)

Greedy BFS: – CSIS Virtual Lab Example Results

Demo

8	7	6	5	4	5	0
	6		4	3	4	1
6	5		3	2	3	2
5	4		3	2	3	3
4	3	2			1	4
	4	3		1	2	5
6	5	4	3	2	3	6
0	1	2	3	4	5	

NWES:
S(1,0) G(4,4)

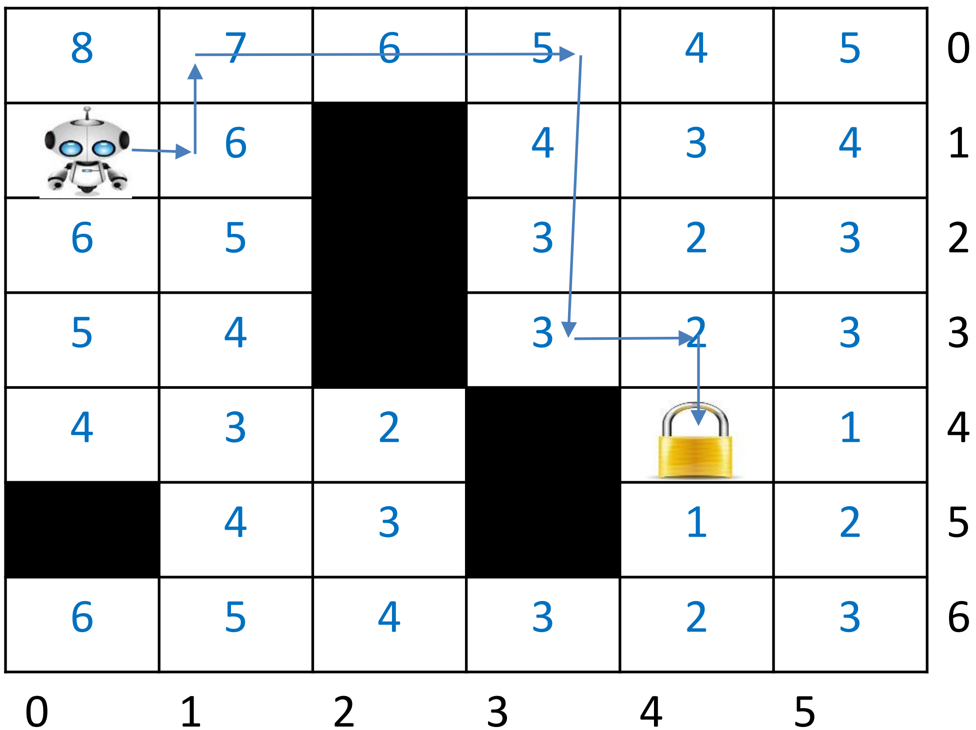
$h(n) =$ Manhattan Distance

RULE for Heuristic= Admissibility and consistence

Completeness

A*: – CSIS Virtual Lab Example Results

Demo

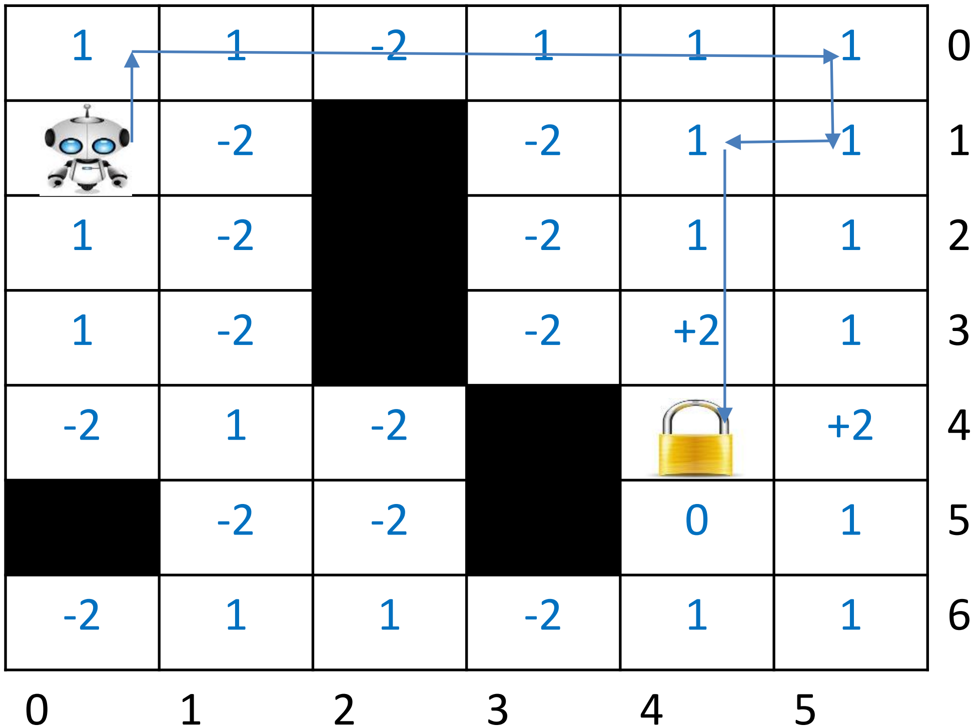


NWES:
S(1,0) G(4,4)

[(1, 0), (1, 1), (0, 1), (0, 2), (0, 3), (1, 3), (2, 3), (3, 3), (3, 4), (4, 4)]

Hill Climbing

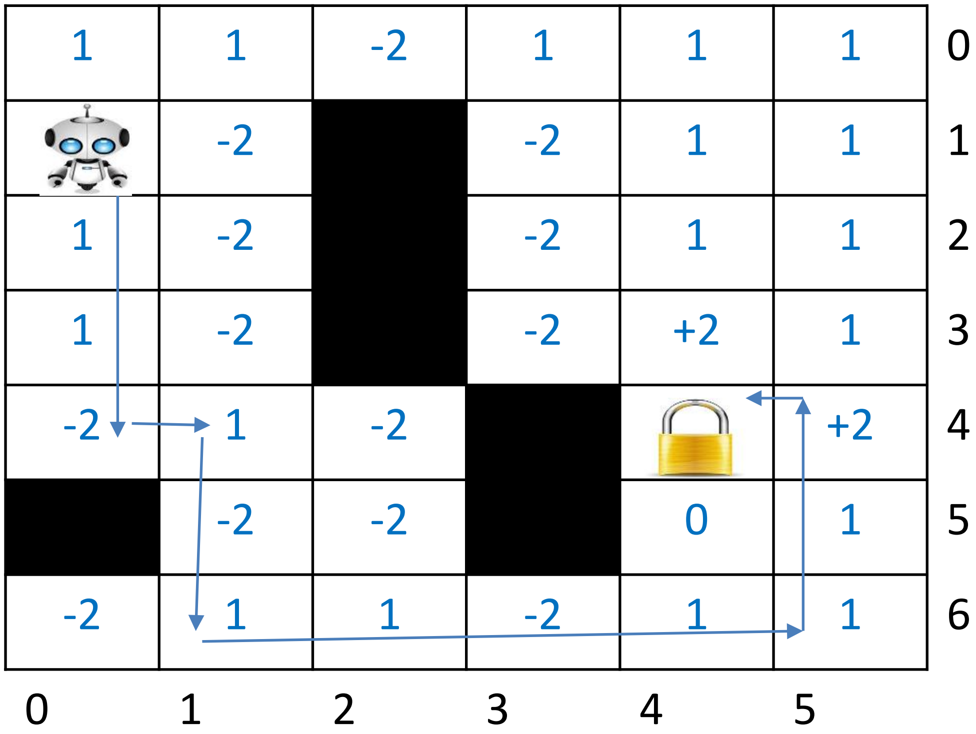
Demo



NWES:
S(1,0)

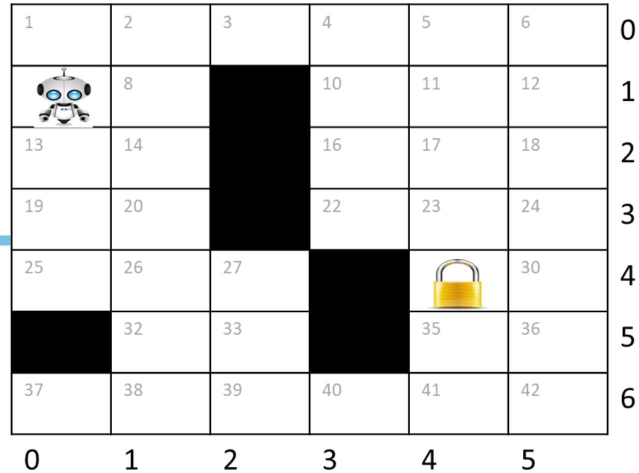
Hill Climbing

Demo



SENW:
S(1,0)

Genetic Algorithm



Chromosome / String

'N' Location & Fitness Value

Cross over function

Double Split Sequence Cross over

Mutation

Random 2 cell mutation , valid with sequence propagation

Fitness Value Design

Total cell value . HTB

Heuristic values

7	8	2	3	4	5	6	12	18	24	....
1	-2	1	-2	1	1	1	1	1	1	...

14	8	2	3	4	5	11	17	23	29	...
-2	-2	1	-2	1	1	1	1	+2	+3	...

7	8	2	3	4	10	11	12	18	24	..
1	1	-2	1	-2	1	1	1	1	1	...

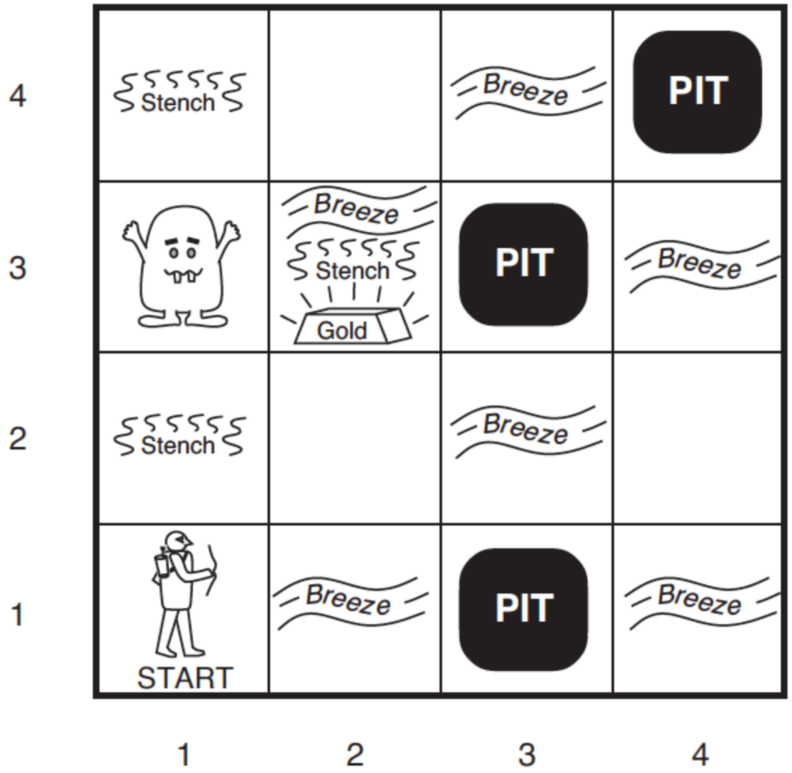
4	10	11	5	6	12	18	24	30	29	...
1	-2	1	1	1	1	1	1	+2	+3	...

4	10	11	12	18	24	30	29	35	41	...
1	-2	1	1	1	1	+2	+3	0	1	...

4	10	11	17	23	24	30	29	35	41	....
1	-2	1	1	+2	1	+2	+3	0	1	....

~10 (based on visible subset alone

AI Navigation



Wumpus World Problem:

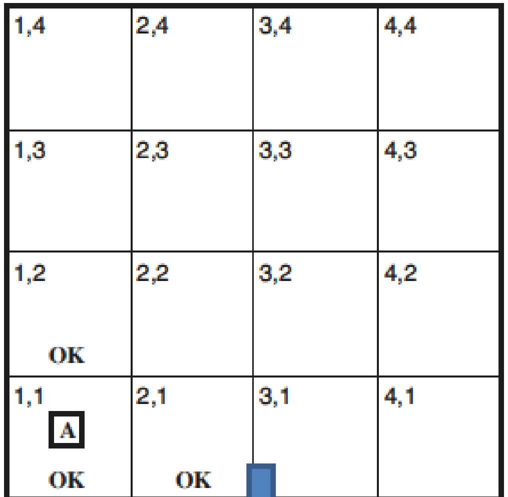
PEAS:

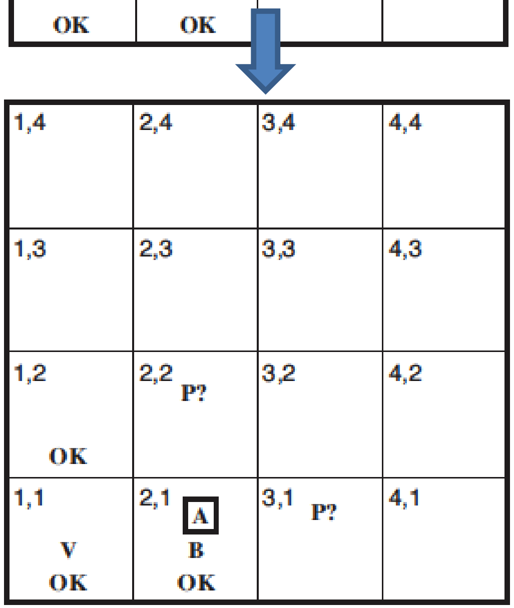
Performance Measure:
+100 for climbing out with gold,
-100 for falling into a pit or being eaten by Wumpus,
-5 for each action taken and
-10 for hitting a wall

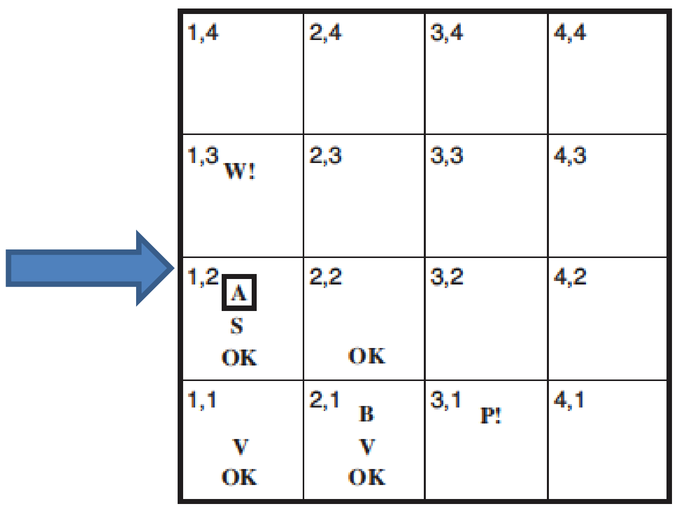
Environment: 4x4 grid of rooms. Always starts at [1, 1] facing right.
The location of Wumpus and Gold are random.
Agent dies if entered a pit or live Wumpus.

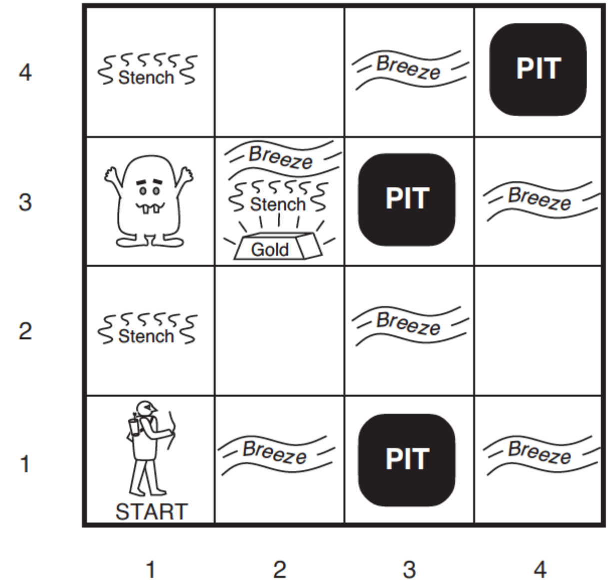
Knowledge Representation

Percept Format:
[Stench?, Breeze?, Glitter?, Bump?, Scream?]

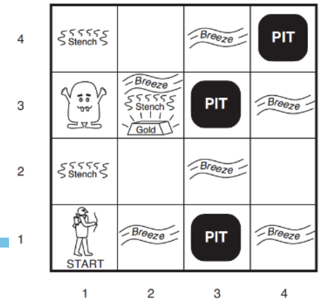








Propositional Logic - Modelling



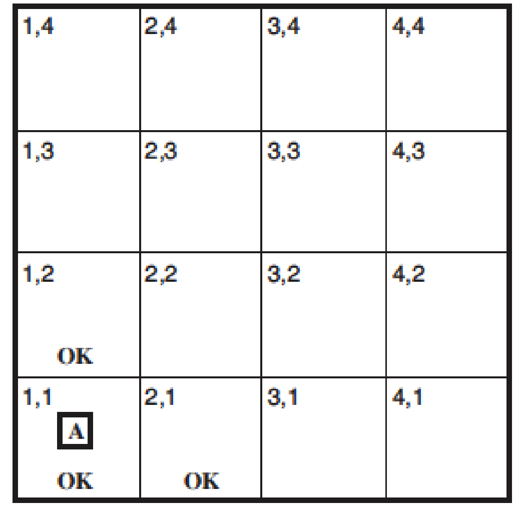
For each [x, y] location

$P_{x,y}$ is true if there is a pit in [x, y]

$W_{x,y}$ is true if there is a wumpus in [x, y]

$B_{x,y}$ is true if agent perceives a breeze in [x, y]

$S_{x,y}$ is true if agent perceives a stench in [x, y]



----------------------- R is the sentence in KB

$R_1 : \neg P_{1,1}$

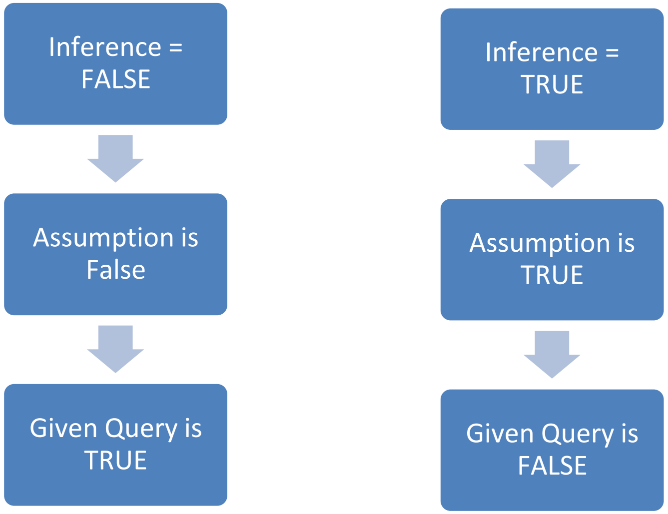
$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

$R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

$R_4 : \neg B_{1,1}$

$R_5 : B_{2,1}$

Proof by Contradiction



Inference : Example – Theorem Proving

$R_1 : \neg P_{1,1}$

$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

$R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

$R_4 : \neg B_{1,1}$

$R_5 : B_{2,1}$

$$
\begin{array}{rcll}
(\alpha \wedge \beta) & \equiv & (\beta \wedge \alpha) & \text{commutativity of } \wedge \\
(\alpha \vee \beta) & \equiv & (\beta \vee \alpha) & \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) & \equiv & (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) & \equiv & (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\
\neg(\neg\alpha) & \equiv & \alpha & \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\beta \Rightarrow \neg\alpha) & \text{contraposition} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\alpha \vee \beta) & \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) & \equiv & ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) & \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) & \equiv & (\neg\alpha \vee \neg\beta) & \text{De Morgan} \\
\neg(\alpha \vee \beta) & \equiv & (\neg\alpha \wedge \neg\beta) & \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) & \equiv & ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) & \equiv & ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge
\end{array}
$$

Query: $\neg P_{1,2}$. Can we prove if this sentence be entailed from KB using inference rules?---------------------

$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$	
$R_6 : (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$	Biconditional Elimination
$R_7 : ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$	And Elimination
$R_8 : (\neg B_{1,1} \Rightarrow \neg (P_{1,2} \vee P_{2,1}))$	Contraposition
$R_9 : \neg (P_{1,2} \vee P_{2,1})$	Modus Ponens
$R_{10} : \neg P_{1,2} \wedge \neg P_{2,1}$	Demorgans
R11: $\neg P_{1,2}$	And Elimination

PL-Resolution

$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$ commutativity of $\wedge$
$(\alpha \vee \beta) \equiv (\beta \vee \alpha)$ commutativity of $\vee$
$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$ associativity of $\wedge$
$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$ associativity of $\vee$
$\neg(\neg\alpha) \equiv \alpha$ double-negation elimination
$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$ contraposition
$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$ implication elimination
$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$ biconditional elimination
$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$ De Morgan
$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$ De Morgan
$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$ distributivity of $\wedge$ over $\vee$
$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$ distributivity of $\vee$ over $\wedge$

$R_1 : \neg P_{1,1}$
$\cancel{R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})}$
$\cancel{\mathbf{R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})}}$
$R_4 : \neg B_{1,1}$
$R_5 : B_{2,1}$
Query: $\neg P_{1,2}$

$R_6 : \neg B_{1,1} \vee P_{1,2} \vee P_{2,1}$
$R_7 : \neg P_{1,2} \vee B_{1,1}$
$R_8 : \neg P_{2,1} \vee B_{1,1}$
$R_9 : \neg B_{2,1} \vee P_{1,1} \vee P_{2,2} \vee P_{3,1}$
$R_{10} : \neg P_{1,1} \vee B_{2,1}$
$R_{11} : \neg P_{2,2} \vee B_{2,1}$
$R_{12} : \neg P_{3,1} \vee B_{2,1}$

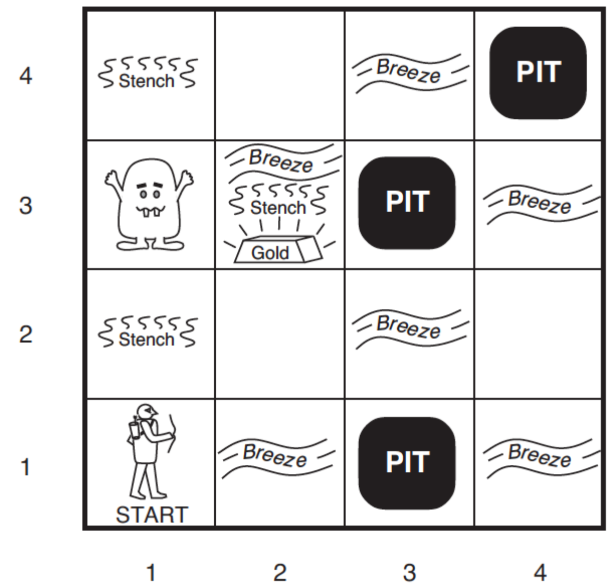
Eliminate		$\mathbf{R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})}$	$\mathbf{R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})}$
$\leftrightarrow$	Biconditional Elimination	$(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$	$(B_{2,1} \Rightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})) \wedge ((P_{1,1} \vee P_{2,2} \vee P_{3,1}) \Rightarrow B_{2,1})$
$\rightarrow$	Implication Elimination	$\neg B_{1,1} \vee (P_{1,2} \vee P_{2,1})$ $\mathbf{\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1}}$	$\neg B_{2,1} \vee (P_{1,1} \vee P_{2,2} \vee P_{3,1})$ $\mathbf{\neg(P_{1,1} \vee P_{2,2} \vee P_{3,1}) \vee B_{2,1}}$
Clause level $\neg$	De Morgan	$\mathbf{(\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1}}$	$\mathbf{(\neg P_{1,1} \wedge \neg P_{2,2} \wedge \neg P_{3,1}) \vee B_{2,1}}$
CNF Form	Distributivity of $\vee$ over $\wedge$	$\mathbf{(\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})}$	$\mathbf{(\neg P_{1,1} \vee B_{2,1}) \wedge (\neg P_{2,2} \vee B_{2,1}) \wedge (\neg P_{3,1} \vee B_{2,1})}$

PL-Resolution

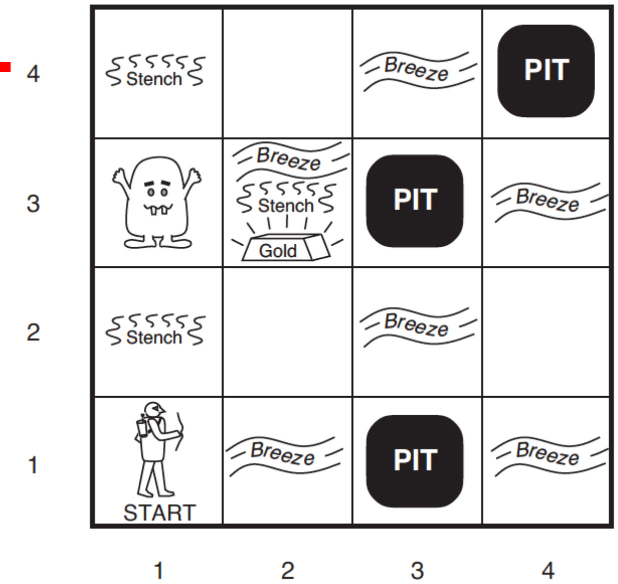
Unit Resolution: Query: $\neg P_{1,2}$

To find: Is there a pit in location (1,2) using the CNF obtained in previous slide

$$
\begin{tikzcd}
\neg B_{1,1} \arrow[dr] & & \neg P_{1,2} \vee B_{1,1} \arrow[dl] \\
P_{1,2} \arrow[dr] & \neg P_{1,2} \arrow[d] & \\
& \{\ \} &
\end{tikzcd}
$$



Predicate Logic – Sample Modelling



"Squares neighboring the wumpus are smelly"
$\forall x$,y *Neighbour*(x,y) ∧ *Wumpus*(y)⟹Smelly(x)

"Squares neighboring the pit are breezy"
$\forall x$,y *Neighbour*(x,y) ∧ *Pit*(y)⟹Breezy(x)

"Squares that are breezy and smelly are riskier"
$\forall x$ *Smelly*(x) ∧ *Breezy*(x)⟹*Risk(x)*

"Squares that are neither smelly or breezy are safer"
$\forall x$ ~*Smelly*(x) ∧ ~*Breezy*(x)⟹*Safe(x)*

"Neighbors of squares without breeze do not have pit"
$\forall x$,y *Neighbour*(x,y) ∧ ~Breezy(x) ⟹ ~ *Pit*(y)

"Neighbors of squares without smell do not have wumpus"
$\forall x$,y *Neighbour*(x,y) ∧ ~ Smelly(x) ⟹ ~ *Wumpus*(y)

Order of quantifiers is important

Forward Chaining

- Consider the following problem:

The law says it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

- We will prove that West is a criminal

Forward Chaining

1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$

Missile(M1)

Owns(Nono, M1)

American (West)

Enemy (Nono, America)

Forward Chaining

- Consider the following problem:

The law says it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

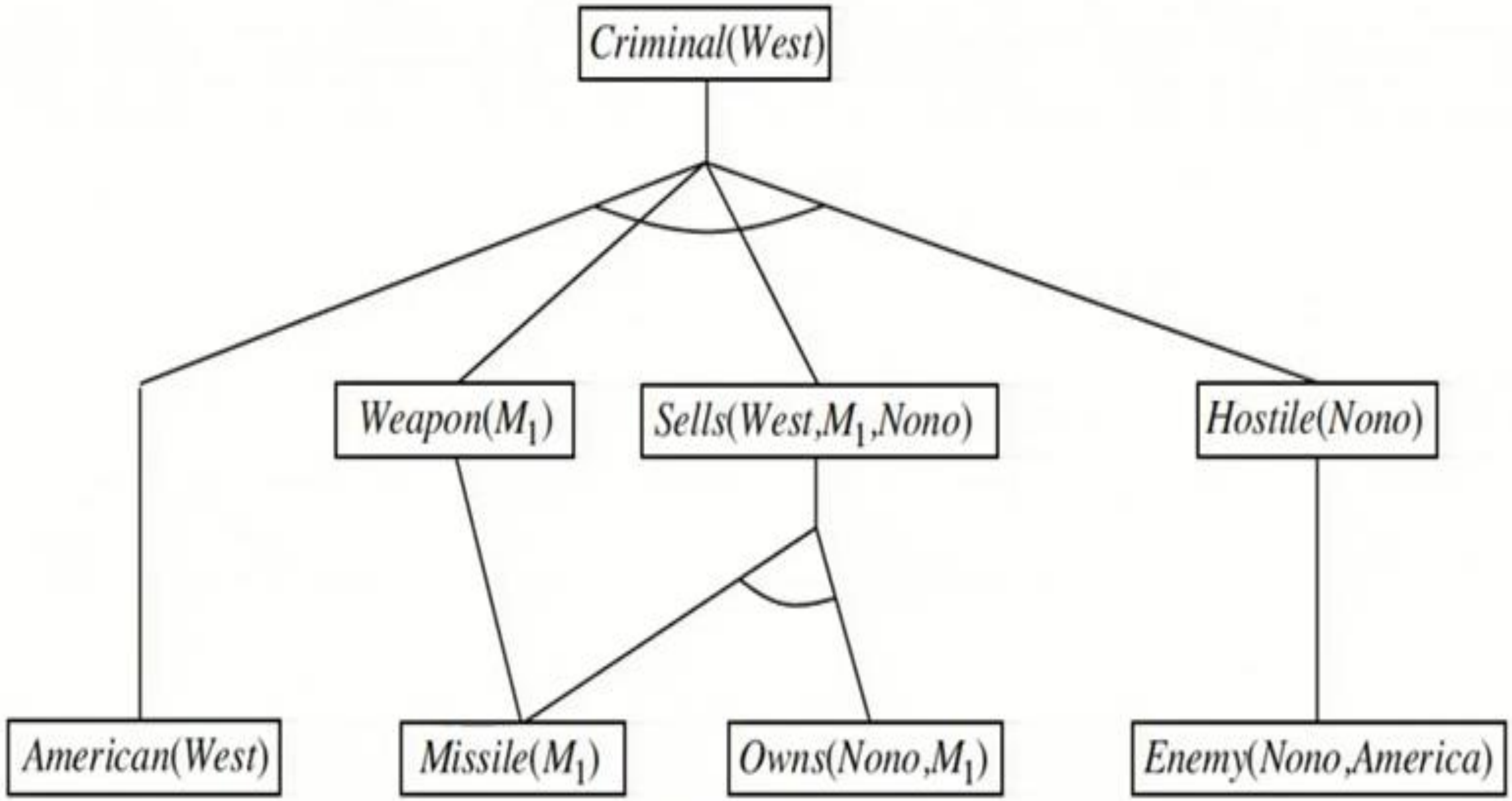
- We will prove that West is a criminal

Algorithm:

1. Start from the facts
2. Trigger all rules whose premises are satisfied
3. Add the conclusion to known facts
4. Repeat the steps till no new facts are added or the query is answered

Forward Chaining

1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$



Backward Chaining

Algorithm:

1. Form Definite Clause
2. Start from the Goals
3. Search through rules to find the fact that support the proof
4. If it stops in the fact which is to be proved → Empty Set- LHS

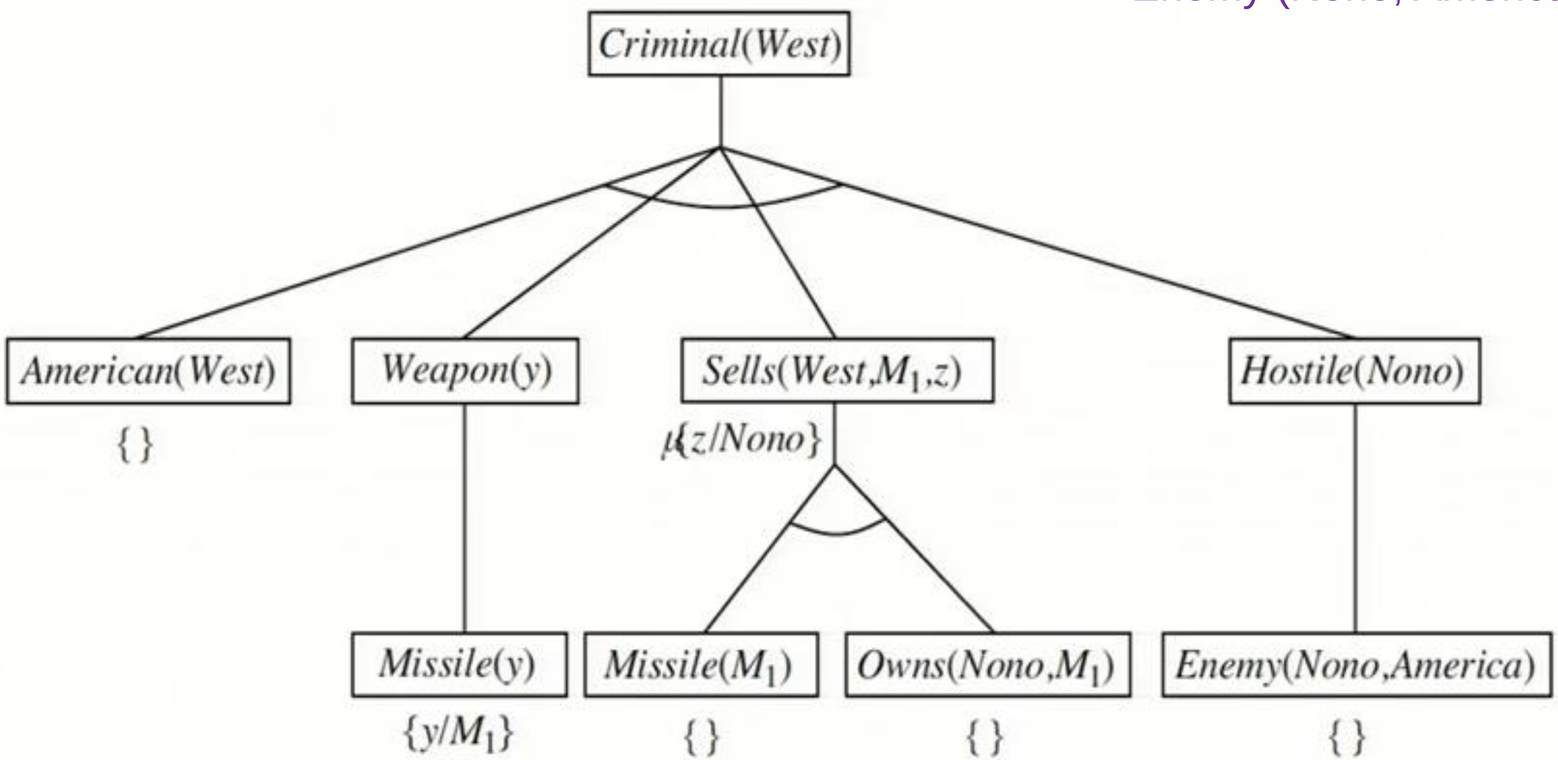
Divide & Conquer Strategy

Substitution by Unification

Backward Chaining

1. $American(x) \land Weapon(y) \land Sells(x, y, z) \land Hostile(z) \Rightarrow Criminal(x)$
2. $Missile(x) \land Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$
3. $Missile(x) \Rightarrow Weapon(x)$
4. $Enemy(x, America) \Rightarrow Hostile(x)$

Missile(M1)
Owns(Nono, M1)
American (West)
Enemy (Nono, America)

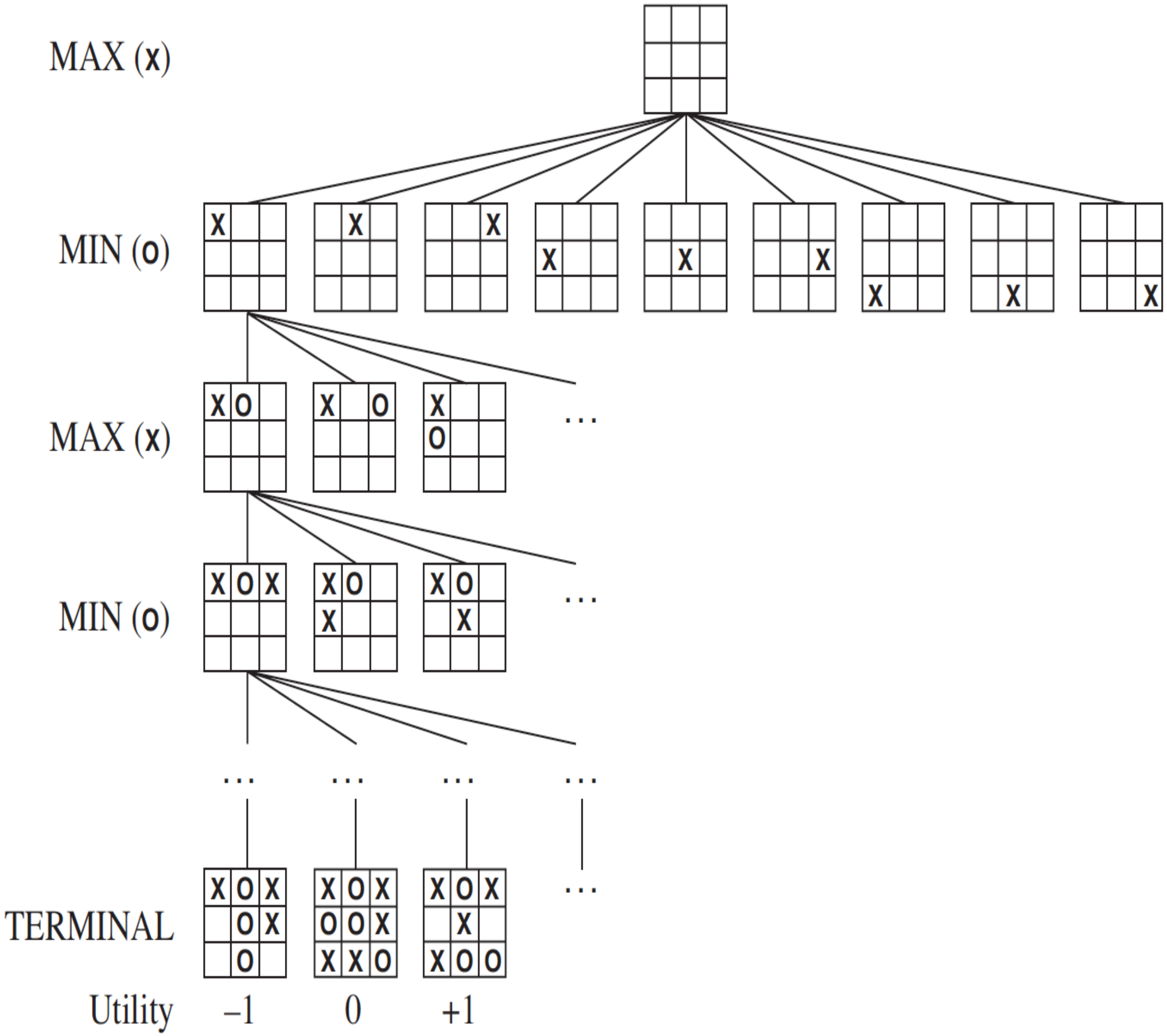


Games as Search Problem

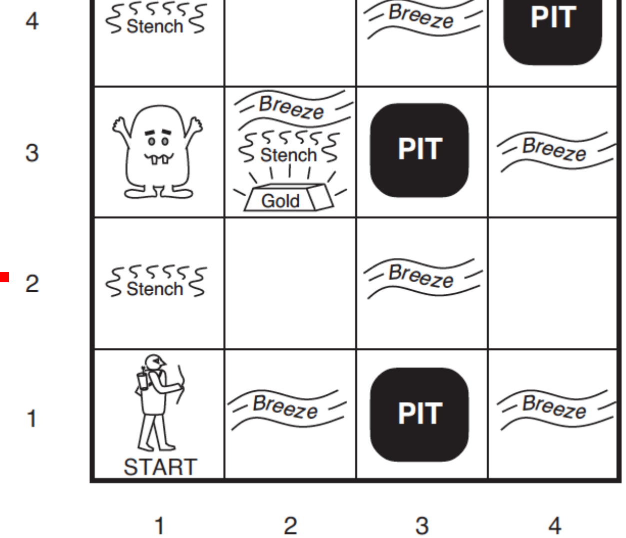
PSA : Representation of Game:

- INITIAL STATE: S0
- PLAYER(s)
- ACTIONS(s)
- RESULT(s, a)
- TERMINAL-TEST(s)
- UTILITY(s, p)

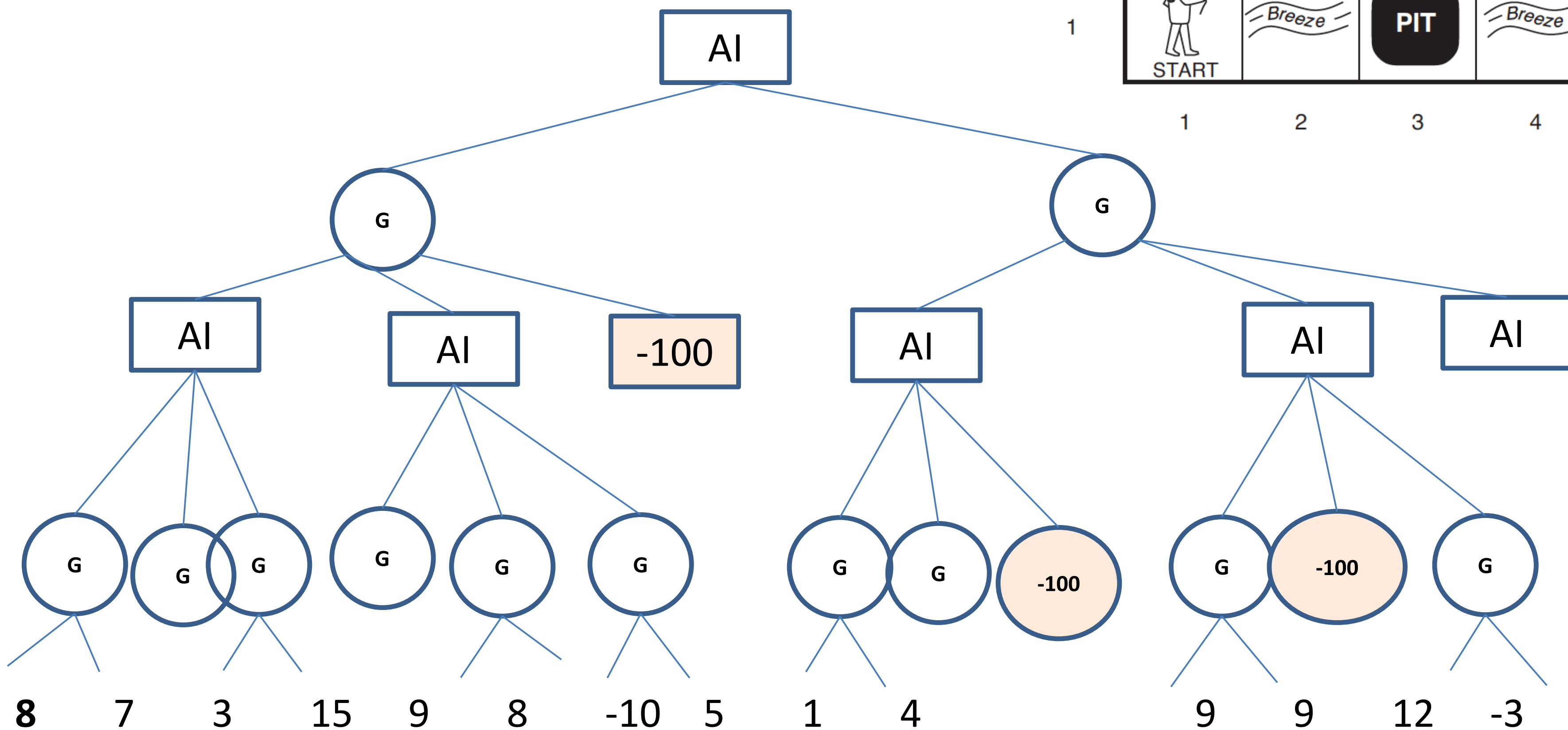
Eg., Tic Tac Toe



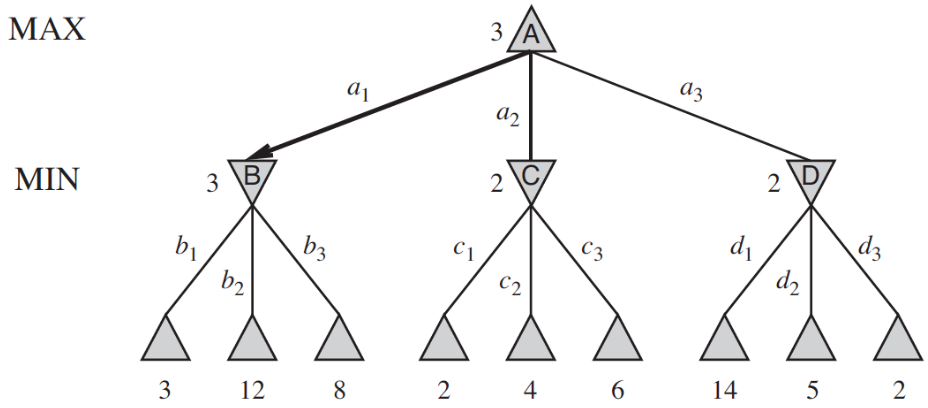
Games



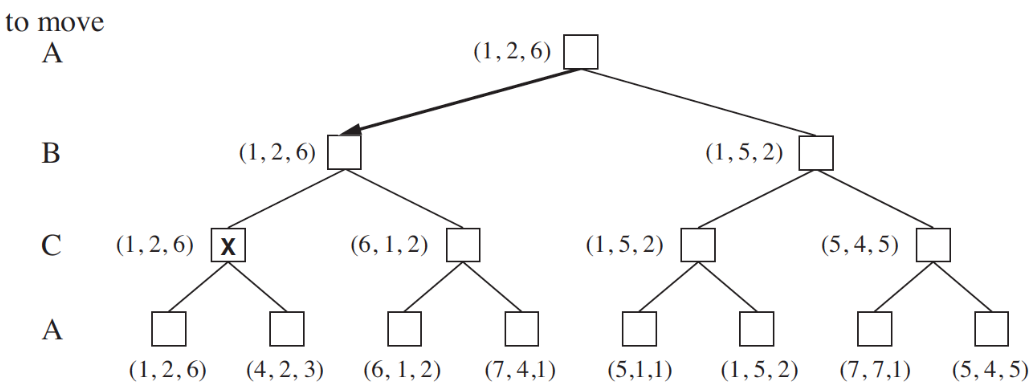
4
3
2
1
Stench
Breeze
PIT
Stench
Gold
Breeze
Stench
START
1
2
3
4



AI
G
-100
-100
8 7 3 15 9 8 -10 5 1 4 9 9 12 -3

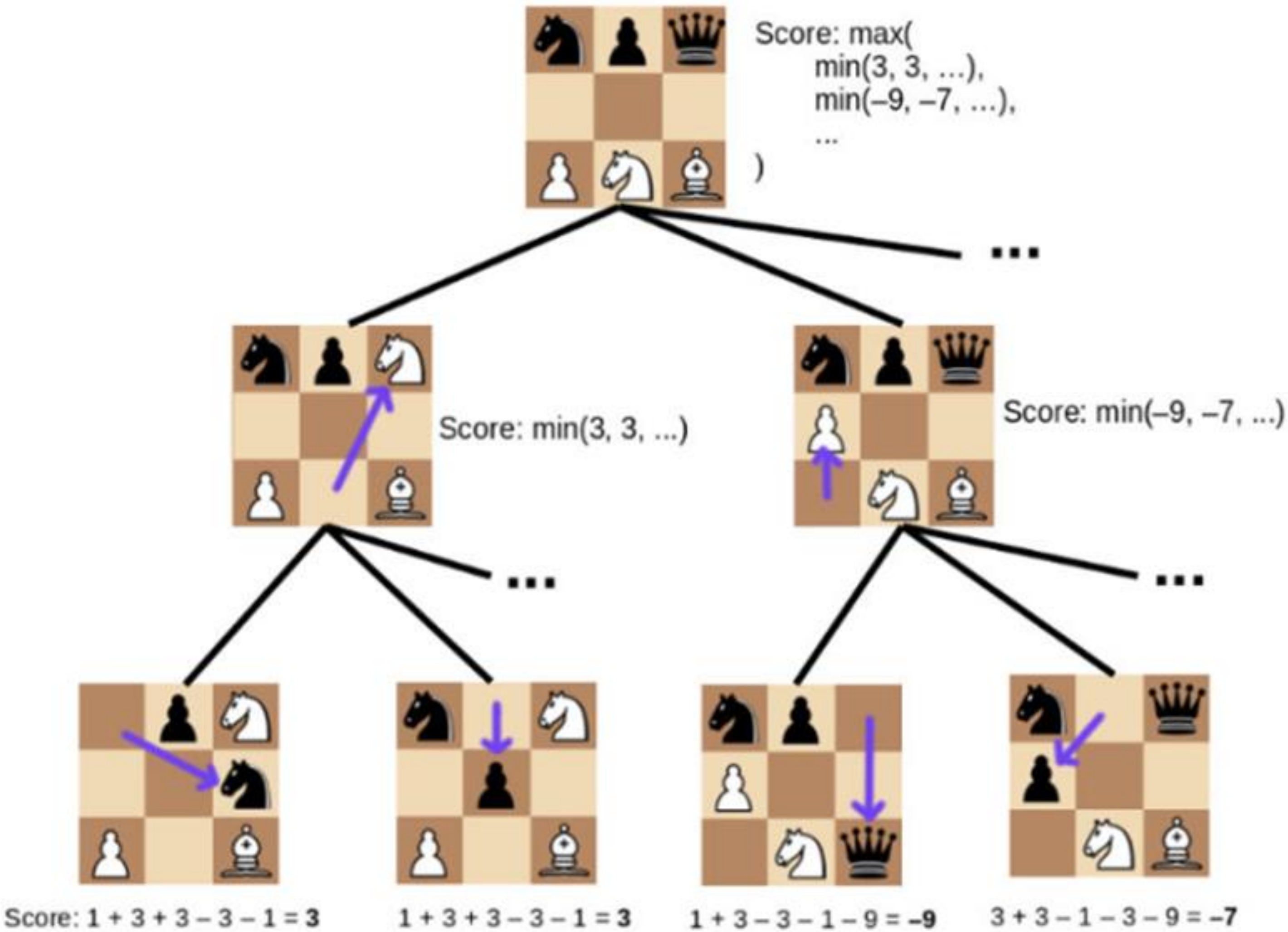


Two Player Game : 1-Ply Game



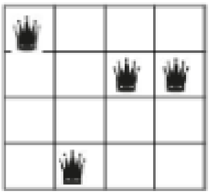
Multiplayer Game

Design of Static Evaluation Values



Design of Static Evaluation Values

N-Queens



1	4	2	2	4

Tic-Tac-Toe

0	0	x
x		0
		x

Max's Share	2
Min's Share	1
Board Value	1

N-Tile

2	8	3
1	6	4
7		5

1	2	3
8		4
7	6	5

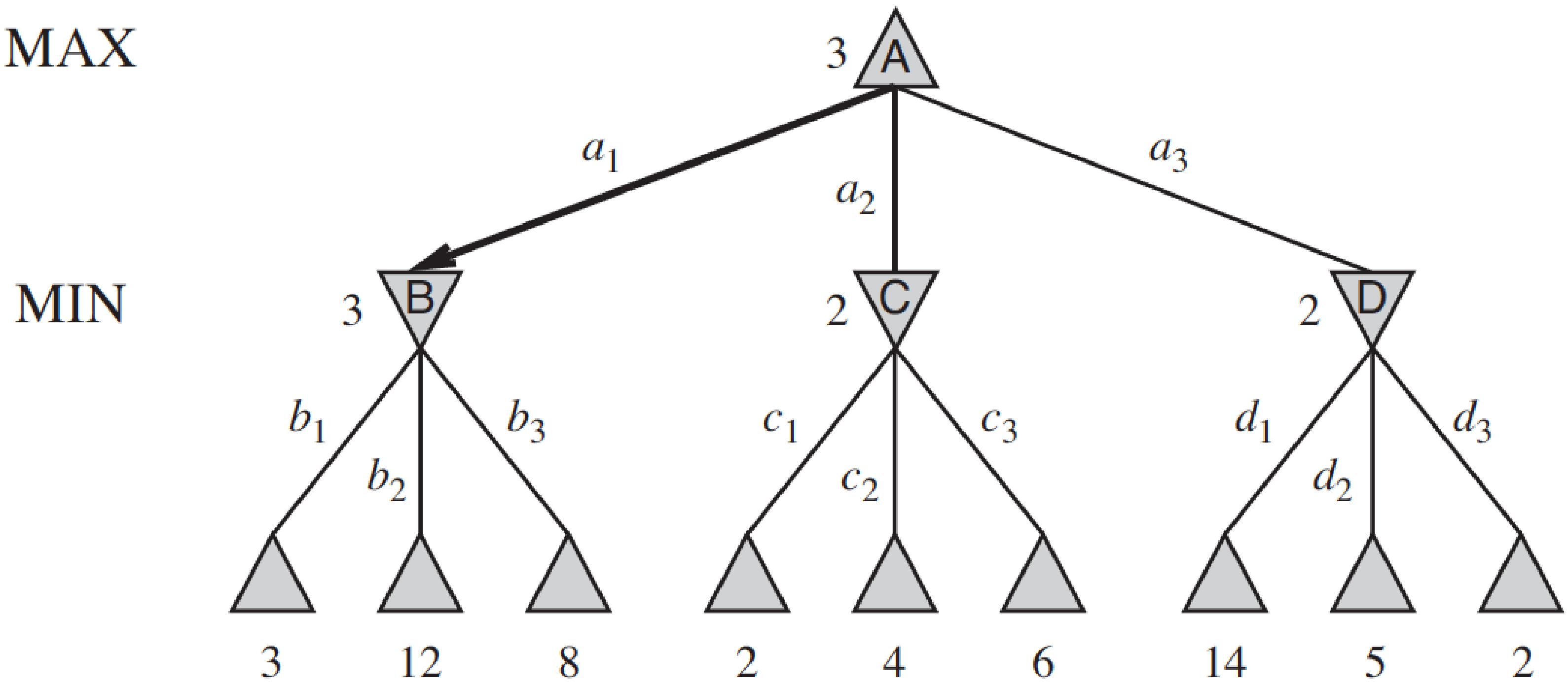
No.of.Tiles Out of Place	5

$$Eval\ (S) = w_1 f_1 (s) + w_2 f_2 (s) + \ldots\ldots + w_n f_n (s)$$

$$= 0.6\ (MaxChance - MinChance) + 0.4\ (MaxPairs - MinPairs)$$

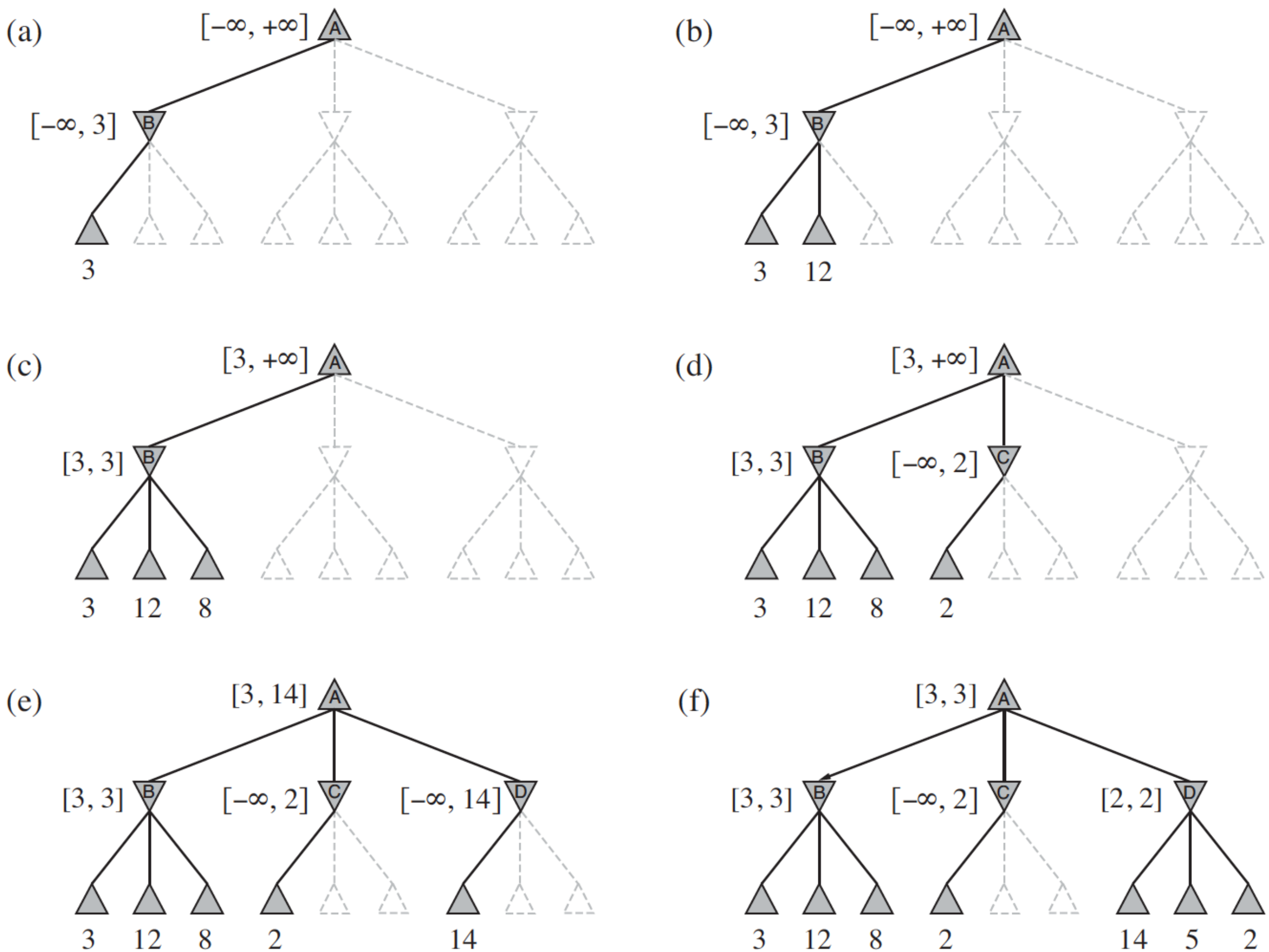
Alpha Beta Pruning

Book Example



Alpha Beta Pruning

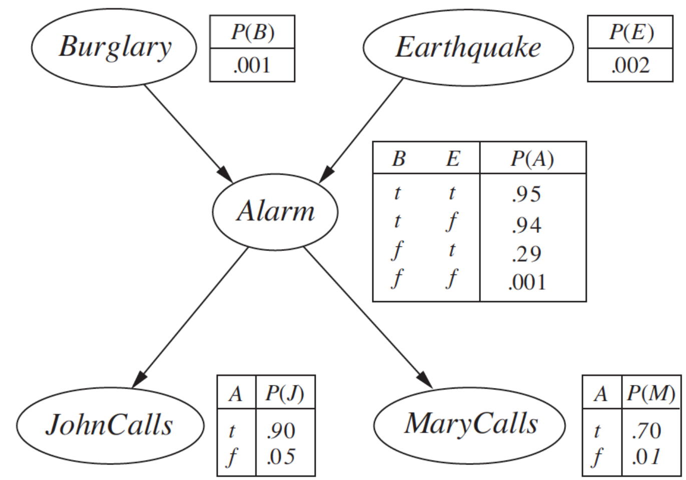
Book Example

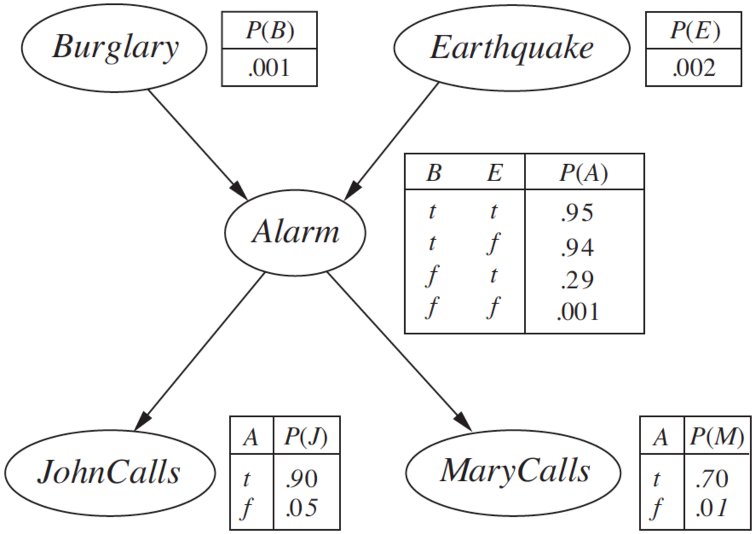


Bayesian Network

A Burglary Alarm System

- Fairly reliable on detecting a burglary
- Also responds to earthquakes
- Two neighbors John and Mary are asked to call you at work when Burglary happens and they hear the Alarm
- John nearly always calls when he hears the alarm, however sometimes confuses the telephone ring with alarm and calls then too
- Mary like loud music and often misses the alarm altogether
- **Problem**: Given the information that who has / has not called we need to estimate the probability of a burglary





P(B)
.001

P(E)
.002

B	E	P(A)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

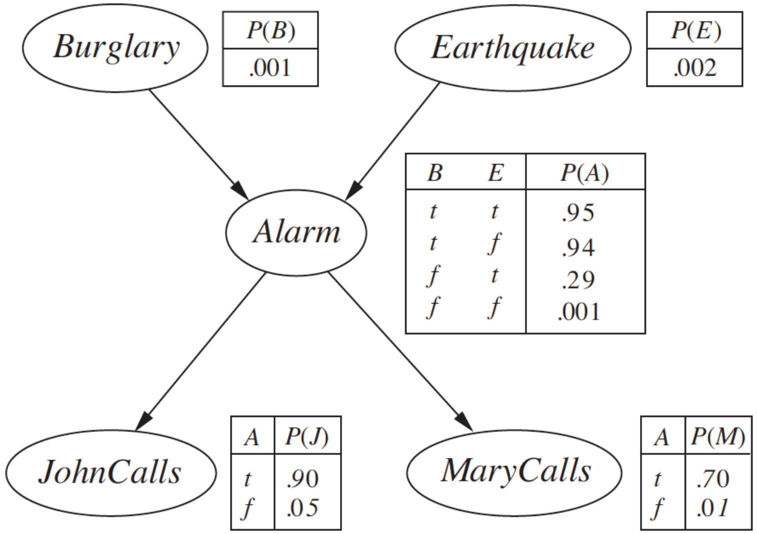
A	P(J)
t	.90
f	.05

A	P(M)
t	.70
f	.01

2. What is the probability that Burglary happened given John & Mary called the police

$$P(B \mid J,M) = \frac{P(B, J, M)}{P(J, M)}$$

$$P(B \mid J,M) = \frac{\sum_{A,\,E} P(J, M, A, B, E)}{\sum_{A,\,B,\,E} P(J, M, A, B, E)}$$



$P(B)$
.001

$P(E)$
.002

B	E	$P(A)$
t	t	.95
t	f	.94
f	t	.29
f	f	.001

A	$P(J)$
t	.90
f	.05

A	$P(M)$
t	.70
f	.01

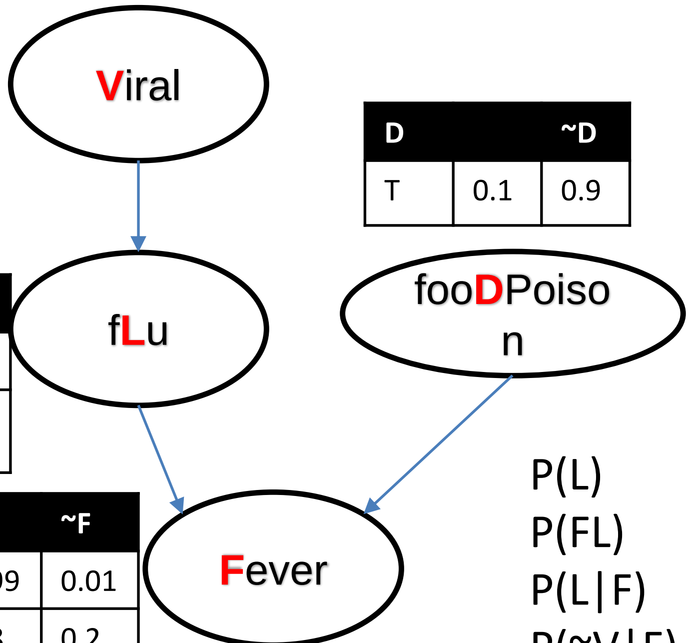
2. What is the probability that Burglary happened given John & Mary called the police

$$P(B \mid J,M) = \frac{P(B, J, M)}{P(J, M)}$$

$$P(B \mid J,M) = \frac{\sum_{A, E} P(J, M, A, B, E)}{\sum_{A, B, E} P(J, M, A, B, E)}$$

Prior Sampling

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	D	F
T	T	F	T
F	F	F	F
T	F	F	T
F	T	F	T
T	T	F	T
T	F	F	F
F	F	F	T
T	F	F	F

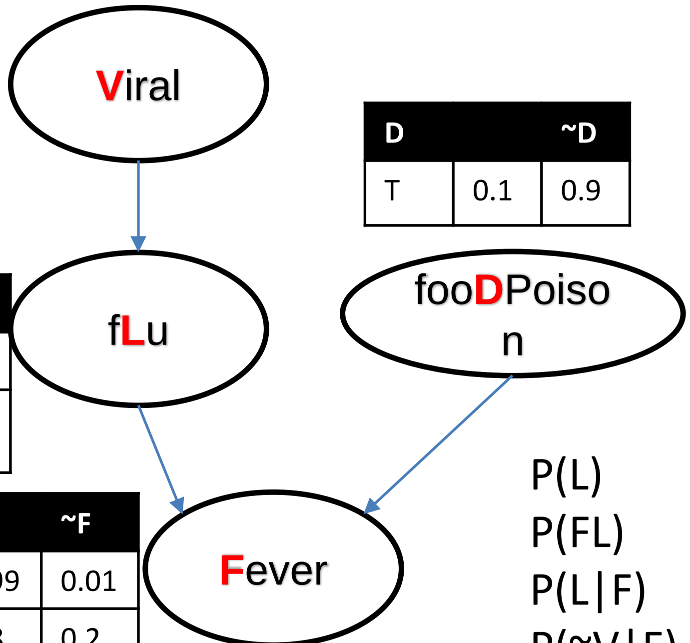
P(L) = 3/8
P(FL) = 3/8
P(L|F) = 3/5
P(~V|F) = 2/5
P(L|V~F) = 0
P(F|D) = ?????

Random numbers generated which is used for sampling based on the Bayes net distribution
0.3, 0.2, 0.58, 0.73, 0.87, 0.15, 0.6, 0.57, 0.85, 0.12, 0.004, 0.93 , 0.0002, 0.99,,………..

Rejection Sampling

Inference

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	D	F
T	T	T	T
F	F	T	F
T	F	T	T
F	T	T	T
T	T	T	T
T	F	T	F
F	F	T	T
T	F	T	F

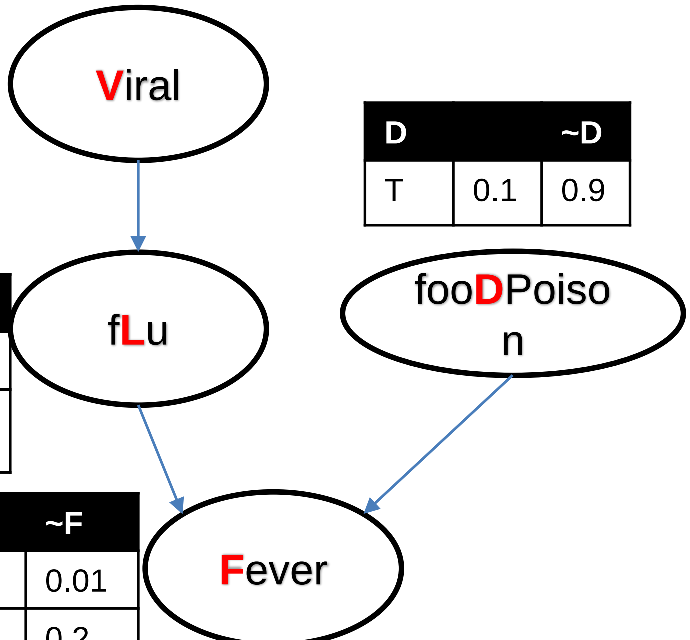
P(L) = 3/8
P(FL) = 3/8
P(L|F) = 3/5
P(~V|F) = 2/5
P(L|V~F) = 0
P(F|D) = 5/8

0.3, 0.2, 0.58, 0.73, 0.87, 0.15, 0.6, 0.57, 0.85, 0.12, 0.004, 0.93 , 0.0002, 0.99,,..........

Likelihood Weighing

Sample Generation by Randomization

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	D	F	wgt
F	F	T	T	0.4*1* 0.1 *1=
F	F	T	T	
F	F	T	T	
F	F	T	T	
F	F	T	T	
F	T	T	T	
F	T	T	F	

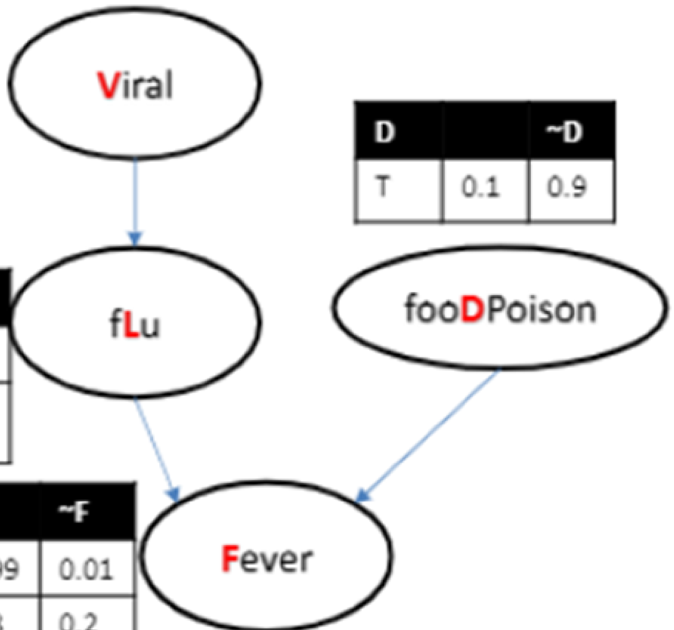
P(~F|D,~V) = 0.04 / 7*0.04

0.3, 0.2, 0.58, 0.73, 0.87, 0.15, 0.6, 0.57, 0.85, 0.12, 0.004, 0.93 , 0.0002, 0.99,,………..

Likelihood Weighing

Sample Generation by Randomization

V		~V
T	0.6	0.4



D		~D
T	0.1	0.9

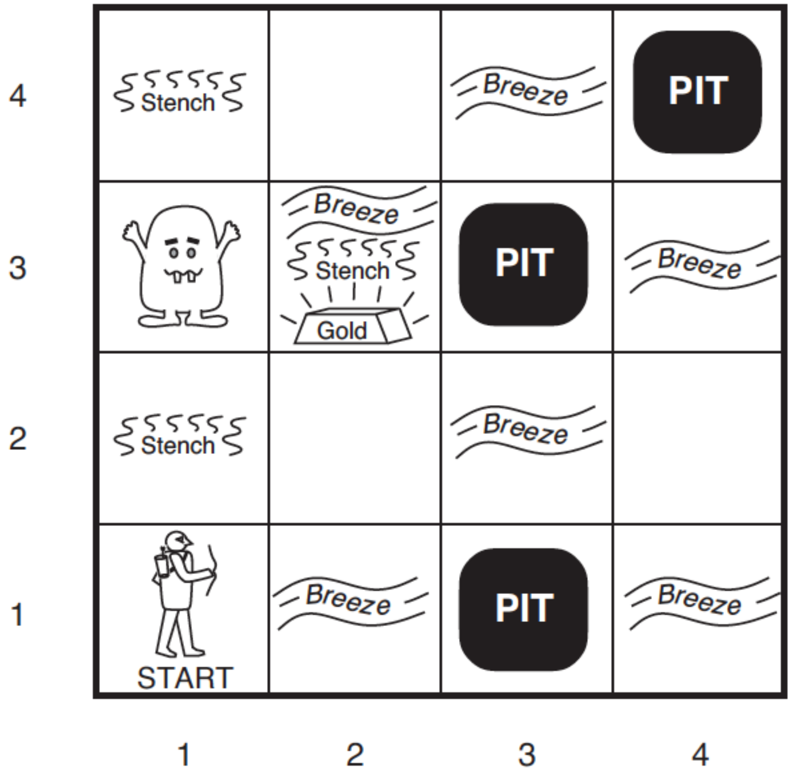
V	L	~L
T	0.8	0.2
F	0.01	0.99

L	D	F	~F
T	T	0.99	0.01
T	F	0.8	0.2
F	T	0.8	0.2
F	F	0.05	0.95

V	L	D	F	wgt
F	F	T	F	1*0.99* 0.1 *1=
F	F	T	T	1*0.99* 0.1 *1=
F	F	T	T	1*0.99* 0.1 *1=
T	F	T	F	1*0.2* 0.1 *1=

$P(F|D,\sim L) \quad = 0.099+0.099 \;\; / \; (3*0.099 + 0.02)$

Hidden Markov Model



Wumpus World Problem:

Sensors :

Stench (S)

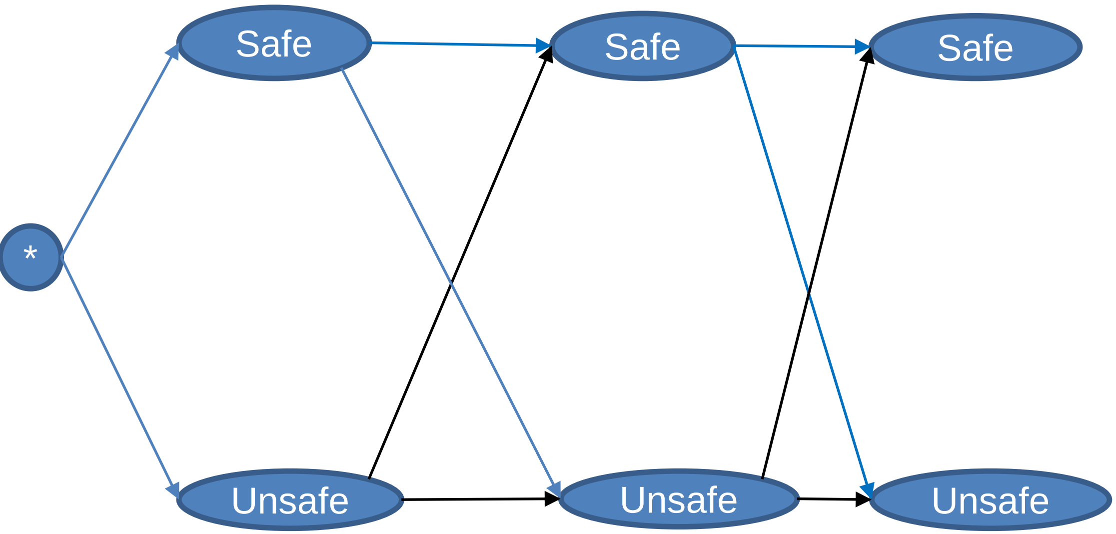
Breeze (B)

Glitter (G)

Hidden Markov Model

Initial probability dist table

P(S)	.5
P(~S)	.5



Wumpus World Problem:

Sensors :

Stench (S)

Breeze (B)

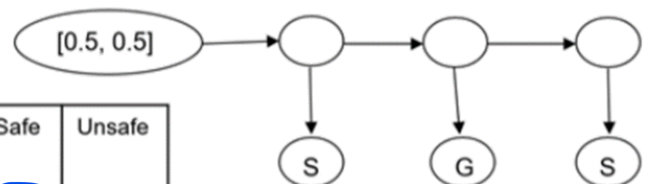
Glitter (G)

Transition table

Unobserved events

Safe_t	Unsafe_{t+1}
True	0.7
False	0.4

Most likely algo.(viterbi algo)



Safe	Unsafe	
0.3	0.2	Glitter
0.4	0.6	Stench

Emission Table

safe	Unsafe	
.4	.6	S
0.3	0.2	B
.3	.2	G

t

S	~S	
0.3	0.6	S
0.7	0.4	~S

t+1

Sample Problem Types

Search Problem Design

A IPL matches – Stadium – Umpire scheduling needs to be automatically done based on dependency of Match Vs Stadium (should be read as assignment of a stadium venue for a match), Match Vs Umpire, Stadium Vs Weekday for a IPL with multiple cricket teams paired & categorized into 3 PlayGroups. There are three types of matches planned for : M1, M2, M3. M1 should be conducted for all 3 groups G1, G2, G3 and they could be umpired by X and Y. M2 should be conducted only 2 groups G1, G2 and they could be umpired by Y and Z. M3 must be conducted for 2 groups G2, G3 and they could be umpired by X and Z. There are only 2 stadiums available. Matches should be run in either of the four slots Slot1: Monday , Slot2: Tuesday and Slot3: Wednesday , Slot4: Thursday in the same week. Umpire X can't take consecutive days and need a break of atleast a day before next match. X and Z doesn't prefer to take multiple matches involving same group.

Local search is best when any assignment-based question asked.

Assignment Problem
State/Strings
Variable : Match&Group Values : Slot&Umpire&Venue
Action /Transition
Assignment of a combination of value tuples to a variable in no particular order
Heuristics / Fitness function
No.of.steps.in.Assignments are less relevant. No.of.conditions in the problems that are satisfied or not satisfied (with or without weighs based on the importance of conditions)
Goal State/Completion
All the variables have valid & consistent assignment. Path becomes less relevant in these cases.

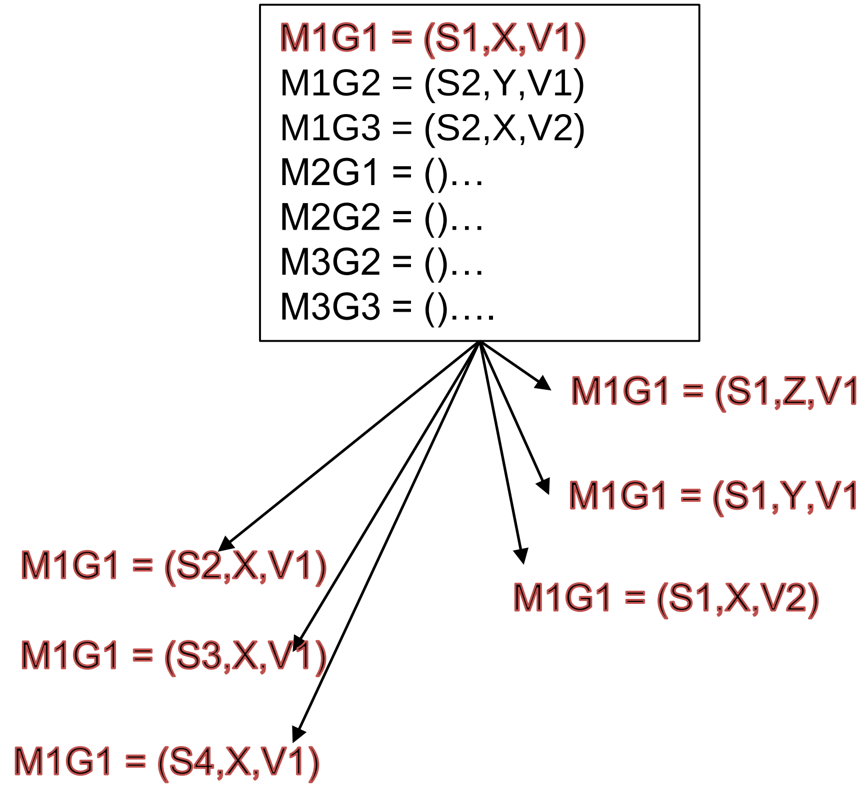
Search Problem Design

A IPL matches – Stadium – Umpire scheduling needs to be automatically done based on dependency of Match Vs Stadium (should be read as assignment of a stadium venue for a match), Match Vs Umpire, Stadium Vs Weekday for a IPL with multiple cricket teams paired & categorized into 3 PlayGroups. There are three types of matches planned for : M1, M2, M3. M1 should be conducted for all 3 groups G1, G2, G3 and they could be umpired by X and Y. M2 should be conducted only 2 groups G1, G2 and they could be umpired by Y and Z. M3 must be conducted for 2 groups G2, G3 and they could be umpired by X and Z. There are only 2 stadiums available. Matches should be run in either of the four slots Slot1: Monday , Slot2: Tuesday and Slot3: Wednesday , Slot4: Thursday in the same week. Umpire X can't take consecutive days and need a break of atleast a day before next match. X and Z doesn't prefer to take multiple matches involving same group.

Agent
Observability
Fully Observable or Partially observable for both cases its suitable
No.of.Agents
Single
No.of.States
Discrete
Determinism
Deterministic in the absence of explicit mention of such environmental influences
Dynamicity
Static in the absence of explicit mention of such environmental influences
Output Dependency
Sequential

Search Problem Design

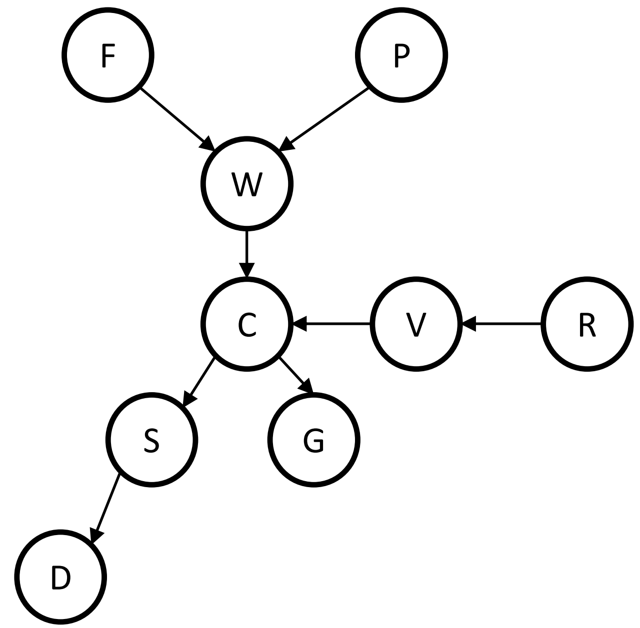
A IPL matches – Stadium – Umpire scheduling needs to be automatically done based on dependency of Match Vs Stadium (should be read as assignment of a stadium venue for a match), Match Vs Umpire, Stadium Vs Weekday for a IPL with multiple cricket teams paired & categorized into 3 PlayGroups. There are three types of matches planned for : M1, M2, M3. M1 should be conducted for all 3 groups G1, G2, G3 and they could be umpired by X and Y. M2 should be conducted only 2 groups G1, G2 and they could be umpired by Y and Z. M3 must be conducted for 2 groups G2, G3 and they could be umpired by X and Z. There are only 2 stadiums venue available **= {v1, v2}**. Matches should be run in either of the four slots Slot1 **(S1)**: Monday , Slot2 **(S2)**: Tuesday and Slot3**(S3)**: Wednesday , Slot4**(S4)**: Thursday in the same week. Umpire X can't take consecutive days and need a break of atleast a day before next match. X and Z doesn't prefer to take multiple matches involving same group.



Partial search tree : Take one variable at a time and assign other possible values to generate children keeping other variables the same at that level of search tree. Here the value is a triplet of (Slot , Umpire, Venue) and varying a value is analogous to changing one of the value in the triplet . No.of.Conditions not satisfied by all the variables put together can be used as heuristics / fitness.

Bayesian Network

Most of the WILP students are fans (F) of cricket irrespective of their gender. With the new season of IPL (Indian Premier League) having started on the exam month almost every cricket fans spend time to watch(W) the live play. Sometimes being a parent (P) reduces the probability of watching the IPL live season. A likely consequence of watching matches is reduced concentration(C) on the following day/s. A consequence of the reduced concentration is increased stress(S) with work environment leading to reduced productivity (D) in project. Lack of concentration might also be caused by viral (V) infection, which is common in this rainy season(R). WILP students have the comprehensive exams and reduced concentration would reduce the probability of good grades (G) in the exam which reflects the performance of students in examination. Assume an AI agent is fed this information and it answers to certain queries that can be inferred. Assume all the events(conditional or unconditional) are equally likely to occur:



Hidden variable R,C, D

Example Joint Prob.Distribution Query : $\sum P(F, P, W, \sim S, V, G)$

What is the chance that "an ardent fan of cricket who is a parent of two kids, never misses an IPL match, doesn't get stressed in work environment, is affected by viral infection and performs well in the comprehensive examination"?

D-Seperation: Performance of in the examination is independent of stress in work environment given its known that the student is affected by viral infection $(G \perp S \mid V)$

Markov Models

Unobserved events. Thats why it is in transition table.

In general it's observed that low concentration/focus among students leads to low grades in 80% of time and high concentration/focus produces good grades in 95% of time. 60% of time students realize that low concentration had led to challenges in completed exams and they decide to concentrate more in the upcoming next exams. 30% of time students are always prudent and proactively prepare with concentration for every exams. At the start of the course enrolment, usually 98% of students are willing to concentrate more for evaluations.

AI agent is designed to perform learner behavioral analysis based on given model. With inputs of grade scored by students for few semesters, how the AI system answers the query, where B=Scoring Low Grade, G= Scoring High Grade.

Observed events. So it is Emission

Transition Model Concentration level		←Previous Exam
Low	High	V current exam
0.4	0.7	concentration Low
0.6	0.3	concentration High

Emission Model		
concentration Low	concentration High	
0.8	0.05	B Less Grades
0.2	0.95	G Good Grades

Initial Probability:
P(High Concentration) = 0.98
P(Low Concentration) = 0.02

Example Markov Model Query :

"What is the most likely explanation of the student behavior if the grade score sequence pattern observed is B-G-B"? – Apply Veterbi Algorithm