PAI

Gatesmashers

Uninformed search (blind / brute force)	Informed search
• Search without info	• Search with info
• No knowledge	• Use knowledge to find steps to soln
• Time consuming	• Quick soln
• More complexity (time, space)	• Less complexity (time, space)
• Eg - DFS, BFS, Uniform cost search	• Eg - Best first search, A*, Heuristic DFS

→ The process of looking for a sequence of actions that reaches the goal is called SEARCH

→ A problem can be defined formally by 5 components.

1) initial state
2) actions
3) Transition model
4) goal test
5) path cost

(1–3) state space - set of all states reachable from initial state by any sequence of actions.

→ An action sequence that leads from the initial state to goal state is called the SOLUTION

→ OPTIMAL SOLN - has lowest path cost among all solutions.

→ Process of removing detail from representation is called ABSTRACTION

→ Definition of a problem-

Initial state - s^0
Actions - $\{a_1, a_2, a_3, \dots\}$
Result (s, a) - s'
Goal test (s) - True/False
Path cost $(s_i \xrightarrow{a_j} s_{i+1} \xrightarrow{a_{j+1}} s_{i+2}) \rightarrow$ cost value (x)
where $i = 0, 1, \dots$
$j = 1, 2, \dots$

Step cost $(s, a, s') \longrightarrow x$

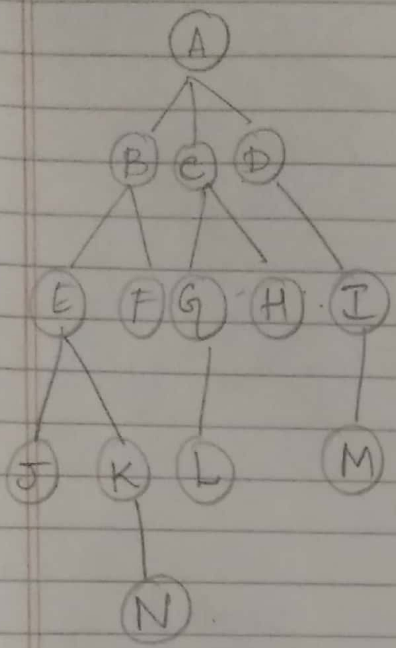
→ Uninformed search have no additional info about state beyond that provided in the problem definition, so they can only proceed by generating successors until they find a goal state

→ Root - initial state
branches - action
nodes - states
leaf node - node with no children
frontier - set of all leaf nodes available for expansion at any given point.

→ In graph-search, the frontier separates the state-space graph into explored & unexplored region, so that every path from initial state to an unexplored state has to pass through a state in the frontier.

Uninformed

BFS - Breadthwise search / level search
- Uninformed search - No info about goal / cost / etc..
- FIFO (Queue)
- Optimal & complete (shortest path & gives answer fo sure)
- ~~time consuming~~



← queue ←

~~A~~ A is taking into BCD

~~B~~ C D B is taking to E F

~~C~~ D E F

~~D~~ E F G H

~~E~~ F G H I

~~F~~ G H I J K

~~G~~ H I J K

~~H~~ I J K L

~~I~~ J K L

~~J~~ K L M

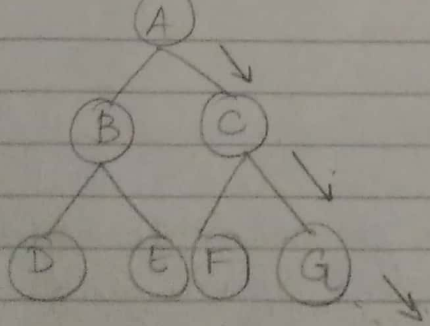
~~K~~ L M

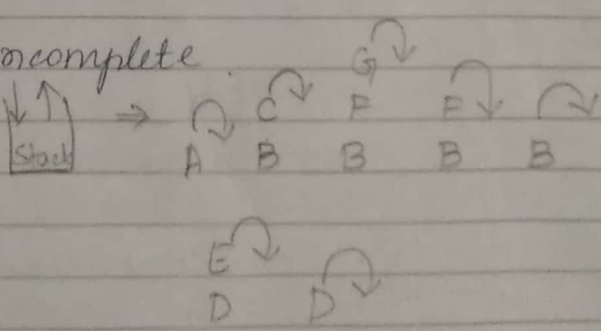
~~L~~ M N

~~M~~ N

N

2) DFS - Depthfirst Search
- Uninformed
- LIFO (Stack)
- non optimal & incomplete

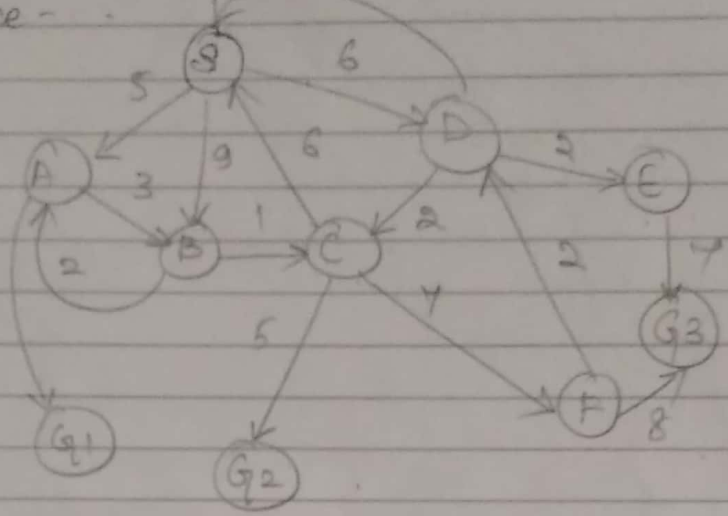


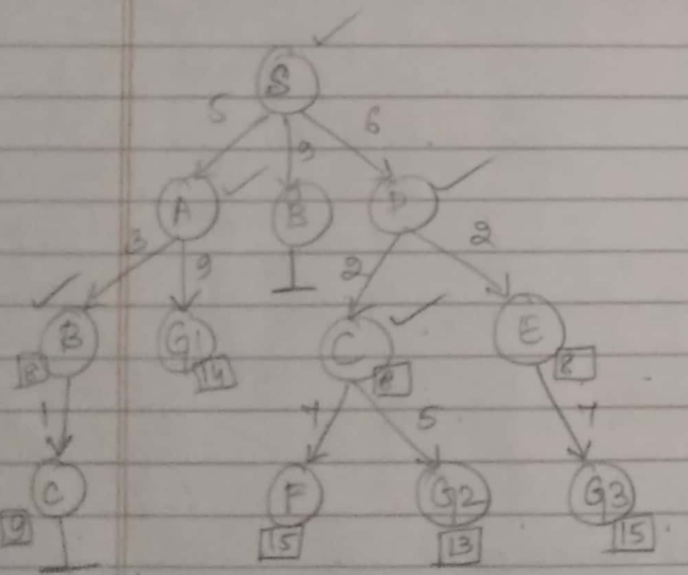


ACGFBED

3) Uniform cost search - move from start state to one of the goal state with minimal total cost on the path.
- cheapest path
- uninformed search..

Search space -





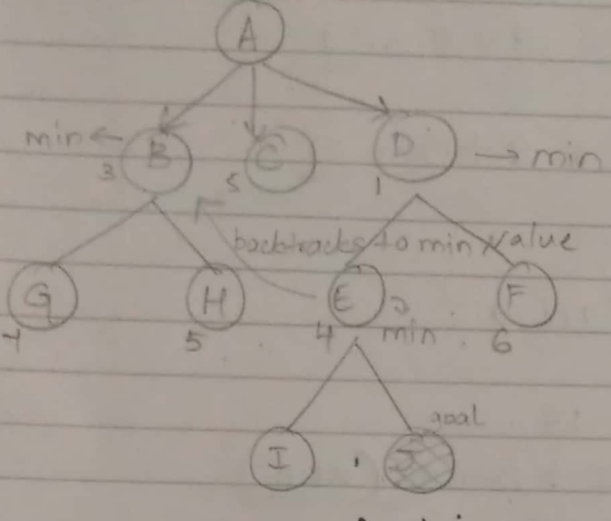
visited
S, A, D, B, C, E

Smallest & Check if its goal
If it is then stop, else gotta expand.

→ Informed search/heuristic search have additional info about search state, so they can guide the search by ranking successors acc to some fitness score, until they find a goal state.

Informed -

1) **Best first search** - depends on heuristic value
- Greedy search
- priority queue is used
  - ↳ closed - already explored/visited
  - ↳ open - not visited



open - ~~A B C D E~~ F G, H

closed - A, B, D, E

Goal

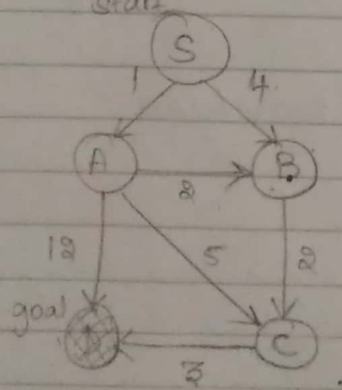
A → D → E → J

2) **A* search** - avoids expanding paths that are already expensive, but expands most promising paths first.

- $f(n) = g(n) + h(n)$ → heuristic value

$f(n)$ → estimated total path to reach goal

$g(n)$ → cost to reach the node



(st. line dist)

heuristic value	
S	7
A	6
B	2
C	1
D	0

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

$S \rightarrow f(A) = 0 + 7 = 7$

$S \rightarrow A \; f(n) = 1 + 6 = 7$

$S \rightarrow B \; f(n) = 4 + 2 = \underline{6}$ (min)

So, $S \rightarrow B \rightarrow C \; f(n) = 6 + 1 = 7$

$S \rightarrow A \rightarrow B = 3 + 2 = \underline{5}$ (min)

$S \rightarrow A \rightarrow C = 6 + 1 = 7$

$S \rightarrow A \rightarrow D = 13 + 0 = 13$

paths - S→A (use this) now (used)

~~S→B~~, S→B→C (used)

(used) ~~S→A→B~~, S→A→C, S→A→D (g = 13)

(use this) → P.T.O

use these P.T.O (used)

(used) ~~S→A→B→C~~

S→A→B→C→D (g = 8)

$S \to A \to B \to C = 5 + 1 = \underline{6}$ (min)

$S \to A \to B \to C \to D = 8 + 0 = \underline{\underline{8}}$

here ↓

$(S \to B \to C) \to D = 9 + 0 = \underline{\underline{9}}$

$(S \to A \to C) \to D = 9 + 0 = \underline{\underline{9}}$

Now,
total goal paths are →

$S \to A \to D = 13$
$S \to A \to B \to C \to D = 8$ (min)
$S \to B \to C \to D = 9$
$S \to A \to C \to D = 9$

So Goal path ↓

$\underline{S \to A \to B \to C \to D}$ $\underline{\underline{\text{cost} = 8}}$

→ Will A^* always find the lowest cost path?
Depends on h value.

→ Condn^s for problem solving by searching

1) fully observable
2) known
3) discrete
4) Deterministic
5) Static

Heuristic search - Guides the search process in the most profitable path among all that are available

informed

7	2	4
5	-	6
8	3	1

Start state

-	1	2
3	4	5
6	7	8

goal state

h_1 - Spaces filled - 8
h_2 - sum of shortest distances to reach goal state

2's shortest dist

$3 + 1 + 2 + 2 + 2 + 3 + 3 + 2 = 18$

1's shortest dist   3's shortest dist

optimal soln - $h_1 + h_2 = 8 + 18 = 26$ moves max needed to solve problem

8 puzzle problem - (without heuristic)

uninformed

1	2	3
4	8	-
7	6	5

Start state

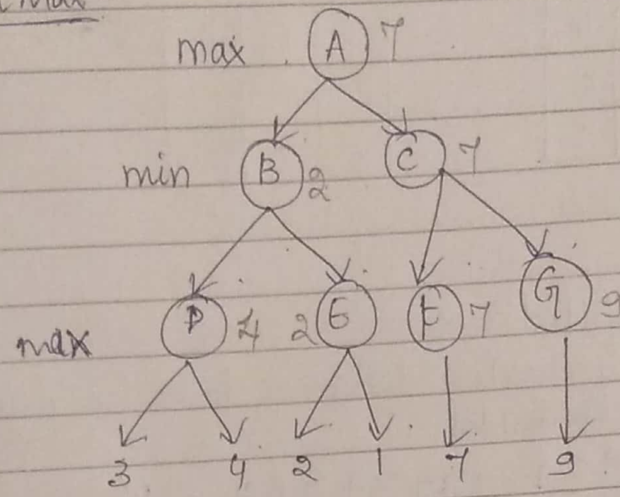
1	2	3
4	5	6
7	8	-

goal state

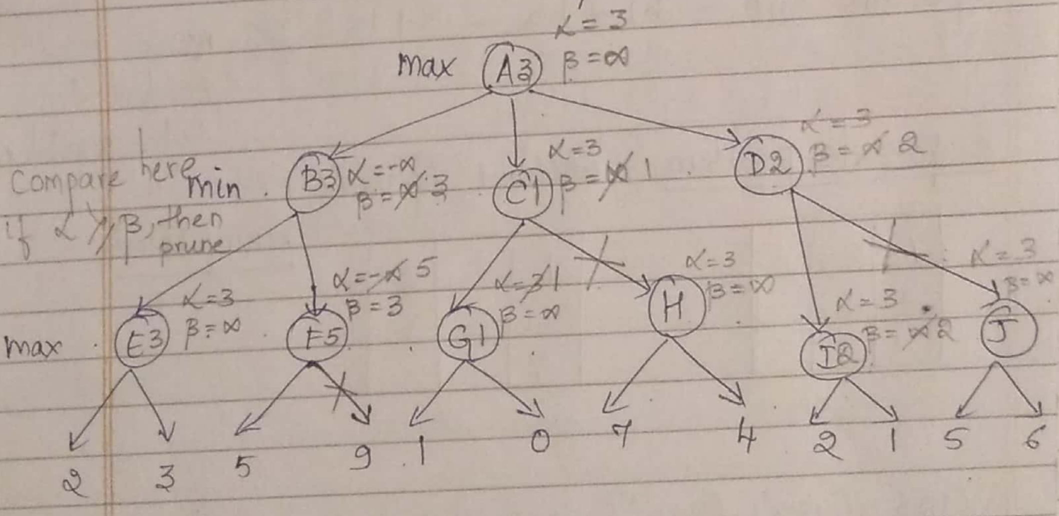
{(123), (480), (765)} → Initial state
{(123), (485), (760)}
{(123), (485), (706)}
{(123), (405), (786)}   Total path cost = 5.
{(123), (450), (786)}
{(123), (456), (780)}

Game tree → Min Max
→ α-β pruning

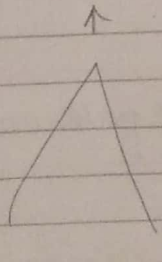
Min Max -



α-β pruning - α $\geq$ β $\alpha = -\infty$ (max)
$\beta = \infty$ (min)



No of α cutoffs - 1
No of β cutoffs - 2



Hidden Markov Model - used in speech recognition

Hidden	Observed
B - Basketball	S - Sunny
F - Football	C - Cloudy
G - Games	R - Rainy

State transition table 1

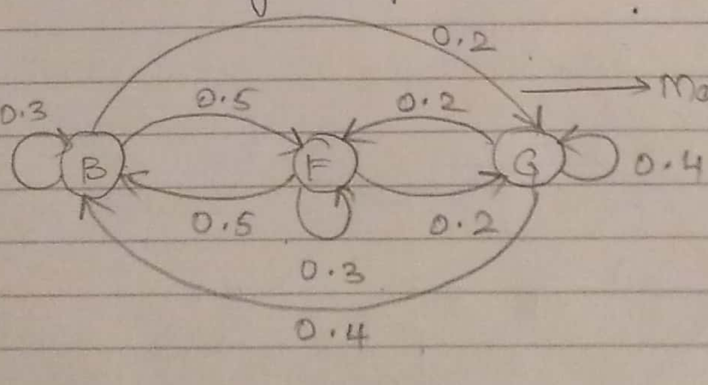
Start/End	B	F	G	Sum
B	0.3	0.5	0.2	1
F	0.5	0.3	0.2	1
G	0.4	0.2	0.4	1

own values but sum should be equal to 1

State transition table 2

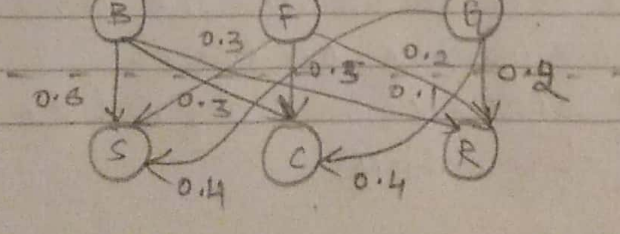
Start/End	S	C	R	Sum
B	0.6	0.3	0.1	1
F	0.3	0.5	0.2	1
G	0.4	0.4	0.2	1

State transition diagram for table 1.



Marko chains.

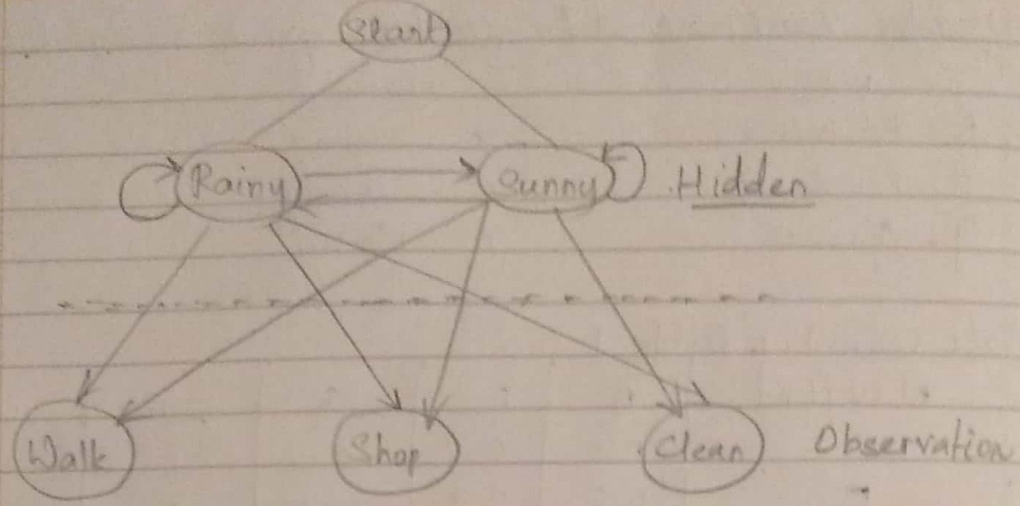
State transition ~~table~~ diagram for table 2 :



Hidden

Observed

2)



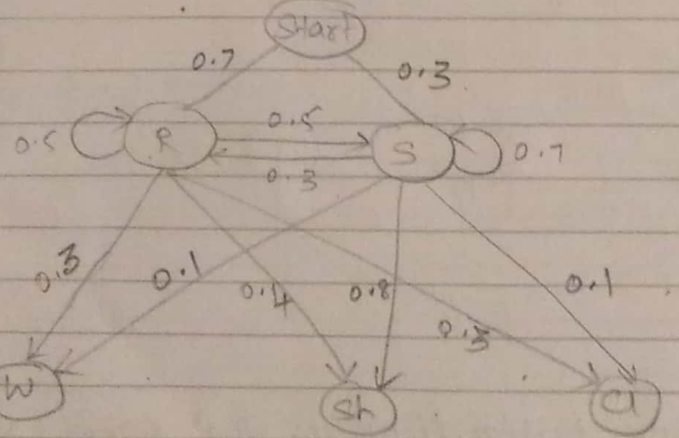
Draw transition tables.

start $\xrightarrow{0.7}$ Rainy
start $\xrightarrow{0.3}$ Sunny

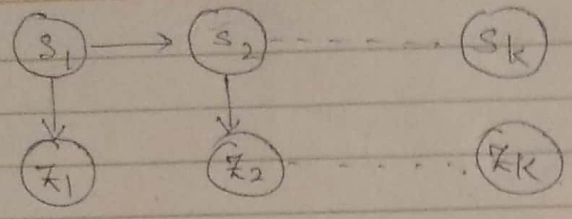
Start/End	Rainy	sunny	Sum
Rainy	0.5	0.5	1
Sunny	0.7	0.3	1

Random values which sum upto 1

Start/End	W. ~~Rainy~~	Sh ~~Sunny~~	Cl ~~Clean~~	Sum
Rainy	0.3	0.4	0.2	1
Sunny	0.1	0.8	1	1
~~Clean~~	~~0.8~~	~~0.2~~		

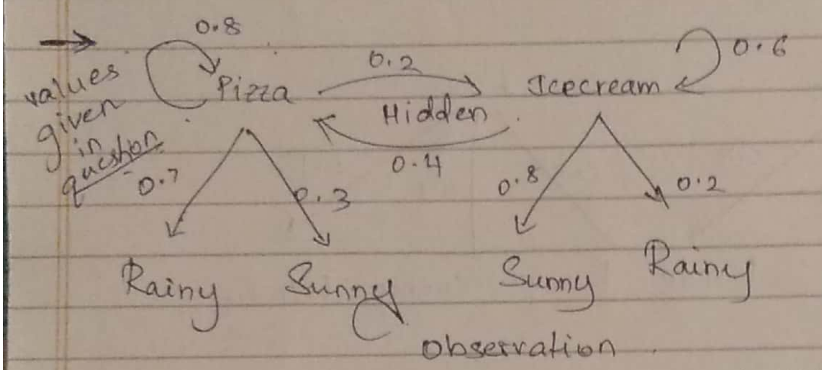


→ Future state depends on current state.
→ States & observations are known.
→ sequence of states is not defined to get particular sequence of observations.



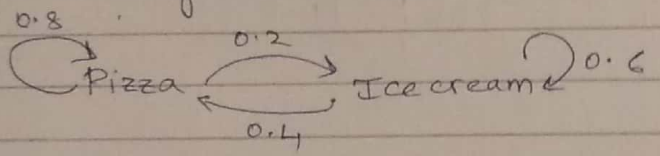
S_{1-k} - States

Z_{1-k} - Observations.

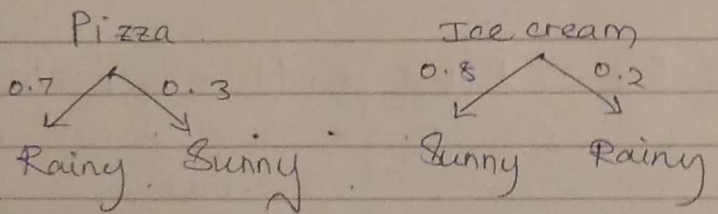


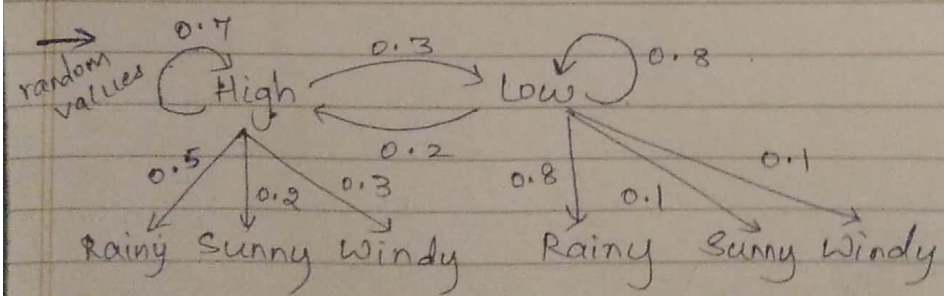
Draw Transition & emission probabilities -

Transition probabilities -

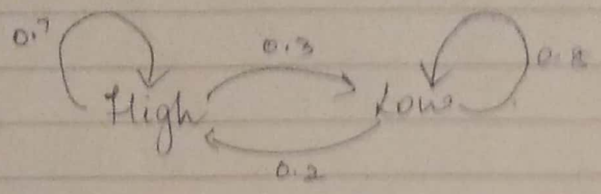


Emission probabilities -

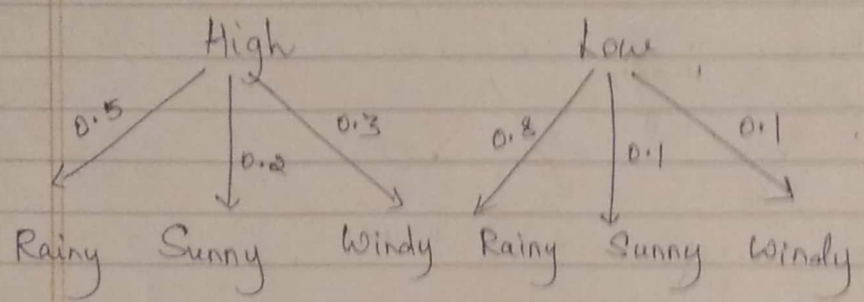




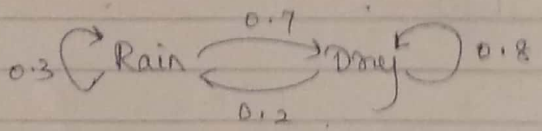
Transition probabilities



Emission probabilities



→



$P(D|D) = 0.8$

$P(R|R) = 0.3$

$P(R|D) = 0.7$

$P(D|R) = 0.2$

$$P(D, D, R, R) = P(R|R) * P(R|D) * P(D|D) * P(D|R)$$
$$= 0.3 * 0.7 * 0.8 * 0.2$$
$$= \underline{0.033}$$