Unit 2 4 hrs.

Problem Solving

2.1 Defining problems as a State Space Search

"It is complete set of states including start and goal states, where the answer of the problem is to be searched".

Problem:

"It is the question which is to be solved. For solving the problem, it needs to be precisely defined. The definition means, defining the start state, goal state, other valid states and transitions".

A state space representation allows for the formal definition of a problem which makes the movement from initial state to the goal state quite easily. So, we can say that various problems like planning, learning, theorem proving etc. are all essentially search problems only.

State space search:

State space search is a process used in the field of computer science, including artificial intelligence (AI), in which successive configurations or ***states*** of an instance are considered, with the intention of finding a ***goal state*** with a desired property.

Problems are often modelled as a state space, a set of ***states*** that a problem can be in. The set of states forms a graph where two states are connected if there is an ***operation*** that can be performed to transform the first state into the second.

© Deepak Bhatta Kaji (MCA)
Department of Computer Science – DEC [NAST]

State space search often differs from traditional computer science search methods because the state space is *implicit*: the typical state space graph is much too large to generate and store in memory. Instead, nodes are generated as they are explored, and typically discarded thereafter. A solution to a combinatorial search instance may consist of the goal state itself, or of a path from some *initial state* to the goal state.

Basic Search Problem:

In state space search a state space is formally represented as a tuple.

Given, [S, s, O, G]   // 4Tuples

Where,
S is the (implicitly specified) set of states.
s is the start state.
O is the set of state transition operators.
G is the set of goal state.

2.2 Problem formulation and Problem types

Problem Formulation:

It is all about deciding what action and states to be consider.

- Agent sensor give it enough information to tell exactly which states it is in currently.
- It knows exactly what each of its action does.
- Then it can calculate exactly which state it will be in after any sequence of action.

1. Initial State
2. Actions
3. Transition Model
4. Goal Test
5. Path Cost

The Problem: 8- Puzzle Problem

The eight-tile or 8-puzzle consist of a 3 by 3 (3*3) square frame board which holds 8 movable tiles numbered 1 to 8. One square is empty, allowing the adjacent tiles to be shifted. The objective of the puzzle is to find a sequence of tile movements that leads from a starting configuration to a goal configuration.

The states of 8 tile puzzle are the different permutations of the tiles within frame.

Let's do a standard formulation of this problem now.

States: It specifies the location of each of the 8 tiles and the blank in one of the nice squares.

Initial state : Any state can be designated as the initial state.

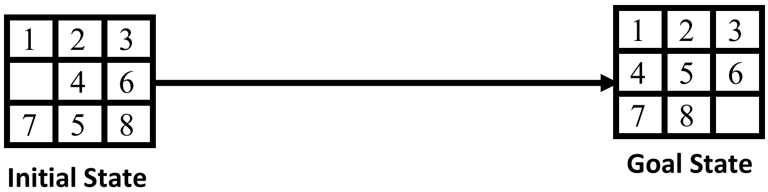
Goal : Many goal configurations are possible one such is shown in the figure

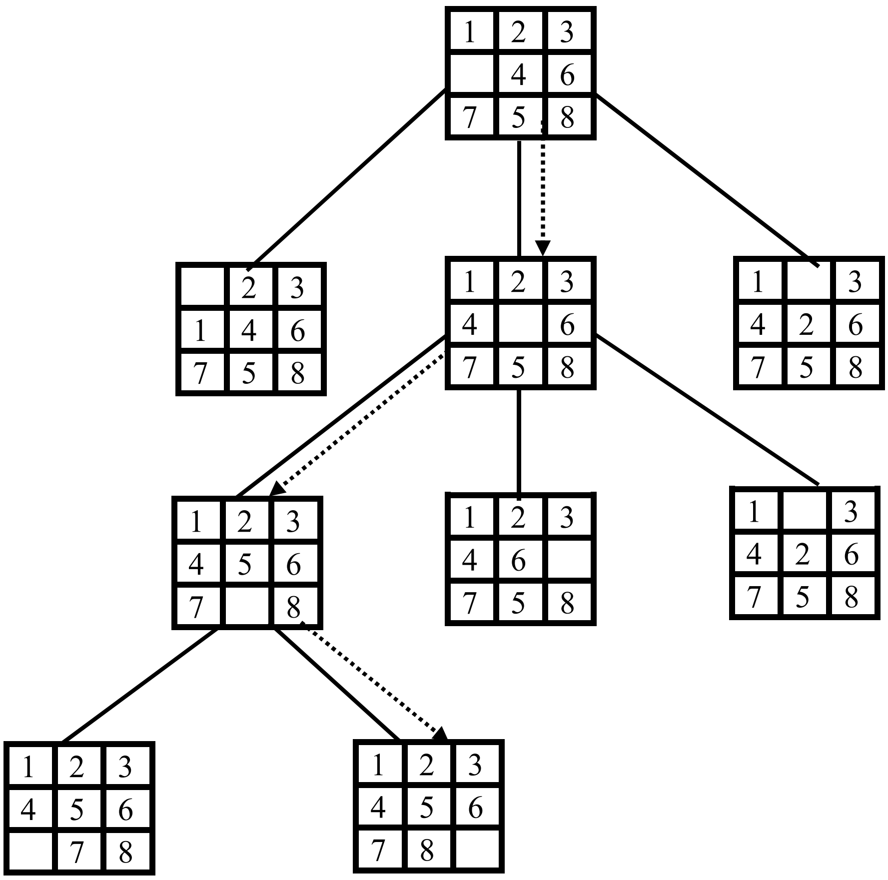
Legal moves (or state) : They generate legal states that result from trying the four actions-

- Blank moves left
- Blank moves right
- Blank moves up
- Blank moves down

Path cost: Each step costs 1, so the path cost is the number of steps in the path.

The tree diagram showing the search space is shown in figure.





Here, we Choose only those states which has minimum path cost.

Department of Computer Science – DEC [NAST]

Problem Types

There are mainly two types of problems:

1. Single state problem.
2. Multiple state problem.

Single State Problem:

When the environment is completely accessible and the agent can calculate its state after any sequence of action, we call it a single-state problem.

Multiple State problem:

When the environment is not fully accessible, the agent must reason about sets of states that it might get to, rather than single states. We call this a multiple-state problem.

The Problem: 8-Queen Puzzle

The eight queen (8-queen) puzzle is the problem of placing eight chess queens on an 8x8 chessboard so that no two queens attack each other; thus, a solution requires that no two queens share the same row, column, or diagonal. The eight queens puzzle is an example of the more general N- queen problem of placing N non-attacking queens on an NxN chessboard, for which solutions exist for all-natural numbers with the exception of n=2 and n=3.

Solutions:

The eight queens puzzle has 94 distinct solutions. If solutions that differ only by the symmetry operations of rotation and reflection of the board are counted as one, the puzzle has 12 solutions. These are called Fundamental solutions.

A fundamental solution usually has eight variants obtained by rotating 90°, 180°, 270° and then reflecting each of the four rotational variants in a mirror in a fixed position. Thus, the number of distinct solutions is $11\times8 + 1\times4 = 92$.

		Q					
				Q			
	Q						
							Q
Q							
						Q	
			Q				
					Q		

Note: For more detail N-Queen Problem is discussed in CSP problem.

The Problem: Missionaries and Cannibals

The Missionaries and Cannibals problem is a classic AI-Puzzle that can be defined as follows:

On One bank of river are three Missionaries and three cannibals. There is one boat available that can hold up to two people and that would like to use boat to cross the river. If the Cannibals ever outnumbers the Missionaries on either of the river's banks, the Missionaries will get eaten or Missionaries may die.

How can the boat be used to safely carry all the Missionaries and Cannibals across the river?

There are 3 missionaries, 3 cannibals, and 1 boat that can carry up to two people on one side of a river.

Goal: Move all the missionaries and cannibals across the river.

Constraint: Missionaries can never be outnumbered by cannibals on either side of river, or else the missionaries are killed.

State: Configuration of missionaries and cannibals and boat on each side of river.

Operators: Move boat containing some set of occupants across the river (in either direction) to the other side.

Department of Computer Science – DEC [NAST]

Rules for solving Missionaries and Cannibal Problem:

#Rules	Action	Meaning
1.	(0, M)	One Missionary Sailing the boat from Bank 1 to Bank 2.
2.	(M, 0)	One Missionary Sailing the boat from Bank 2 to Bank 1.
3.	(M, M)	Two Missionary Sailing the boat from Bank 1 to Bank 2.
4.	(M, M)	Two Missionary Sailing the boat from Bank 2 to Bank 1.
5.	(M, C)	One Missionary & One Cannibal Sailing from Bank 1 to Bank 2.
6.	(C, M)	One Missionary & One Cannibal Sailing from Bank 2 to Bank 1.
7.	(C, C)	Two Cannibal Sailing the boat from Bank 1 to Bank 2.
8.	(C, C)	Two Cannibal Sailing the boat from Bank 2 to Bank 1.
9.	(0, C)	One Cannibal Sailing the boat from Bank 1 to Bank 2.
10.	(C, 0)	One Cannibal Sailing the boat from Bank 2 to Bank 1.

Table: Production Rules for the Missionaries and Cannibal Problem

Applying the rules: Formulation I

Rule Applied	Person in the River Bank 1	Person in the River Bank 2	Boat Position
Initial State	***M, M, M, C, C, C***	***0***	***Bank 1***
5	M, M, C, C	M, C	Bank 2
2	M, M, C, C, M	C	Bank 1
7	M, M, M	C, C, C	Bank 2
10	M, M, M, C	C, C	Bank 1
3	M, C	C, C, M, M	Bank 2
6	M, C, C, M	C, M	Bank 1
3	C, C	C, M, M, M	Bank 2
10	C, C, C	M, M, M	Bank 1
7	C	M, M, M, C, C	Bank 2
10	C, C	M, M, M, C	Bank 1
7	0	M, M, M, C, C, C	Bank 2
Goal State	***0***	***M, M, M, C, C, C***	***Bank 2***

Applying the rules: Formulation II

Rule Applied	Person in the River Bank 1	Person in the River Bank 2	Boat Position
Initial State	***M, M, M, C, C, C***	***0***	***Bank 1***
7	M, M, M, C	C, C	Bank 2
9	M, M, M, C, C	C	Bank 1
7	M, M, M	C, C, C	Bank 2
10	M, M, M, C	C, C	Bank 1
3	M, C	C, C, M, M	Bank 2
6	M, C, C, M	C, M	Bank 1
3	C, C	C, M, M, M	Bank 2
10	C, C, C	M, M, M	Bank 1
7	C	M, M, M, C, C	Bank 2
2	C, M	M, M, C, C	Bank 1
5	0	M, M, M, C, C, C	Bank 2
Goal State	***0***	***M, M, M, C, C, C***	***Bank 2***

Try Yourself

Try to solve this problem same like Missionaries and Cannibal Problem.

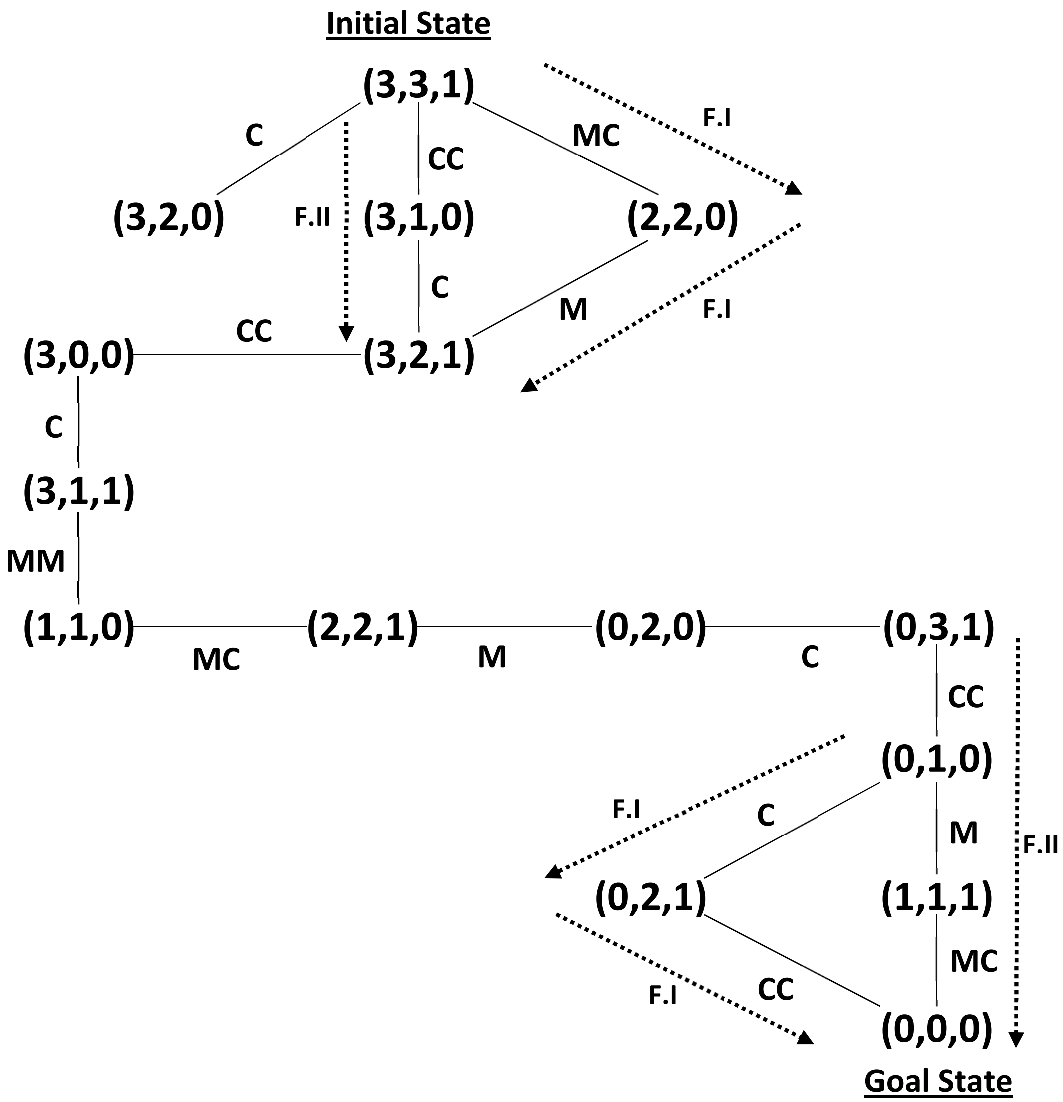
Read the problem carefully to solve gently.

A farmer with wolf, goat & cabbage come to the edge of river. They want to cross the river. There is a boat at river edge but only the farmer can row, the boat can carry two things at a time. If the wolf is ever left alone with the goat, the wolf will eat the goat, similarly if the goat is left alone with cabbage, the goat will eat cabbage. Schedule the things so that all four characters arrive safely on the other side of the river.

Man → M **Wolf → W** **Goat → G** **Cabbage → C**

Department of Computer Science – DEC [NAST]

State Space Diagram for Missionaries and Cannibals Problem Using Formulation I and Formulation II :



Here,

Flow of:

F.I = Formulation I

F.II = Formulation II

Department of Computer Science – DEC [NAST]

The Problem: Water Jug Problem

A Water Jug Problem: You are given two jugs, a 4-Liter one and a 3-Liter one, a pump which has unlimited water which you can use to fill the jug, and the ground on which water may be poured. Neither jug has any measuring markings on it.
How can you get exactly 2-Liter of water in the 4-Liter jug?

<u>Solution:</u>

State Representation and Initial State = we will represent a state of the problem as a tuple (x, y) where x represents the amount of water in the 4-gallon jug and y represents the amount of water in the 3-Liter jug. Note $0 \leq x \leq 4$ and $0 \leq y \leq 3$.

Our initial state: (0,0)

Goal Predicate: state = $(2, y)$ where $0 \leq y \leq 3$.

Operators: we must define a set of operators that will take us from one state to another:

#Rule	Meaning	Check Condition	Calculation
1.	Fill 4-Liter jug	$(x, y) \rightarrow x < 4$	$(4, y)$
2.	Fill 3-Liter jug	$(x, y) \rightarrow y < 3$	$(x, 3)$
3.	Empty 4-Liter jug on the ground	$(x, y) \rightarrow x > 0$	$(0, y)$
4.	Empty 3-Liter jug on the ground	$(x, y) \rightarrow y > 0$	$(x, 0)$
5.	Pour water from 3-Liter jug to fill 4-Liter jug	$(x, y) \rightarrow$ $0 < x + y \geq 4$ and $(y > 0)$	$(4, y - (4 - x))$
6.	Pour water from 4-Liter jug to fill 3-Liter jug	$(x, y) \rightarrow$ $0 < x + y \geq 3$ and $(x > 0)$	$(x - (3 - y), 3)$
7.	Pour all of water from 3-Liter jug into 4-Liter jug	$(x, y) \rightarrow$ $0 < x + y \leq 4$ and $(y > 0)$	$(x + y, 0)$
8.	Pour all of water from 4-Liter jug into 3-Liter jug	$(x, y) \rightarrow$ $0 < x + y \leq 3$ and $(x > 0)$	$(0, x + y)$

Table: Production Rules for the Water Jug Problem

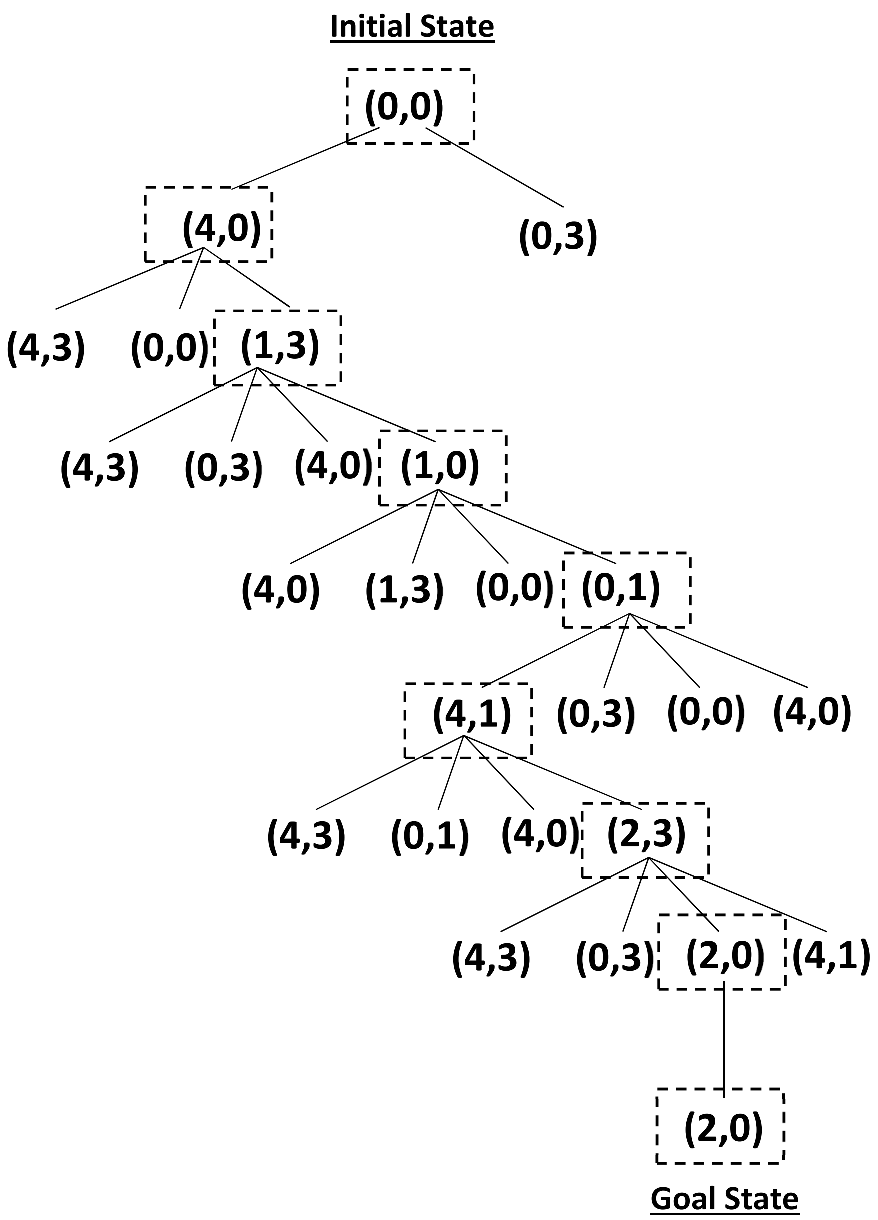
Formulation I.

Liters in 4-Liter jug	Liters in 3-Liter jug	Rule Applied	Meaning
0	0	-	Initial State
0	3	2	Fill 3-Liter jug.
3	0	7	Pour water from 3-Liter jug to 4-Liter jug.
3	3	2	Fill 3-Liter jug.
4	2	5	Pour water from 3-Liter jug to 4-Liter jug until 4-Liter jug is full.
0	2	3	Empty 4-Liter jug on ground.
2	0	8	Pour water from 3-Liter jug to 4-Liter jug.
2	0	-	Goal State Reached.

Formulation II.

Liters in 4-Liter jug	Liters in 3-Liter jug	Rule Applied	Meaning
0	0	-	Initial State
4	0	1	Fill 4-Liter jug.
1	3	6	Pour water from 4-Liter jug to 3-Liter jug.
1	0	4	Empty 3-Liter jug on ground.
0	1	8	Pour all the water from 4-Liter jug to 3-Liter jug.
4	1	1	Fill 4-Liter jug.
2	3	6	Pour water from 4-Liter jug to 3-Liter jug until 3-Liter jug is full.
2	0	4	Empty 3-Liter jug on ground.
2	0	-	Goal State Reached.

State Space Tree Diagram for formulation II:



Try Yourself

Try to draw state space tree diagram for formulation I:

2.3 Well defined problems

Well-Defined Problem: - Clear, definite and well-formed problem.

Problem Solving by Search
An important aspect of intelligence is *goal-based* problem solving.

The **well-defined problems** have specific goals, **clearly defined** solution paths, and clear expected solutions. For example, 8-Puzzle Problem, Water Jug Problem, Tower of Hanoi etc.

The solution of many problems (e.g. Tic Tac Toe (noughts 0 and crosses x), timetabling, chess) can be described by finding a sequence of actions that lead to a desirable goal. Each action changes the *state* and the aim is to find the sequence of actions and states that lead from the initial (start) state to a final (goal) state.

A well-defined problem can be described by:

- **Initial state**
- **Operator or successor function** - for any state x returns s(x), the set of states reachable from x with one action
- **State space** - all states reachable from initial by any sequence of actions
- **Path** - sequence through state space
- **Path cost** - function that assigns a cost to a path. Cost of a path is the sum of costs of individual actions along the path
- **Goal test** - test to determine if at goal state

Extra:

Ill-defined Problem: - Not Clear, indefinite and ill-formed or not well-formed problem.

A problem that lacks one or more of these specified properties is an ill-defined problem, and most problems that are encountered in everyday life fall into this category.

- **Well-defined vs. ill-defined problems**: Problems where the goal or solution is recognizable--where there is a right answer (ex. a math or physics problem) vs. problems where there is no "right" answer but a range of more or less acceptable answers.

Department of Computer Science – DEC [NAST]

2.4 Constraint Satisfaction Problem

What is CSP ?

- Finite set of variables V_1, V_2, V_3, …….. V_n
- Finite set of Constraints C_1, C_2, C_3, …….. C_n
- Non Empty Domain of possible values for each variable D_1, D_2, D_3, …….. D_n
- Each Constraint C_1 limits the values that variables can take, e.g. $V_1 \neq V_2$

A state is defined as an assignment of values to some or all variables.

Solution to a CSP is an assignment to each variable such that each constraint is satisfied.

Some Examples of CSP problems are as follows:

- **8-Puzzle Problem**
- **N-Queen Problem (4x4, 8x8)**
- **Graph Coloring/ Map Coloring/ 3-Color Problem**
- **Crypt-Arithmetic Problem**
- **Crossword**
- **Sudoku (9x9)**

N Queen Problem:

N-Queens problem is a well-known Constraint Satisfactory Problem of Artificial Intelligence. In this problem, we have an **NxN** square grid board and we have **N queens** which need to be placed on them. The queens should be placed on the board in such a way so that it satisfies the below-mentioned constraints:

1 No row should contain more than one queen placed in it
2 No column should contain more than one queen placed in it.
3 Not more than one queen should be placed in the single diagonal.
4 No row or column should be left without any queen placed in it.

On summing up all the constraints, we can conclude that each row and each column should contain exactly one queen in them, neither more nor less than that.

In this series of problems, mostly there are grids whose size is even in number, like 4, 6, 8 and so on. It should be noted that the minimum number of the grid that we can have in this problem is 4, not less than that.

Here the **4-Queen** problem and the 8-Queen problem are the most popular in the N-Queen problem series.

Department of Computer Science – DEC [NAST]

There can exist many solutions for solving this problem, which mean that the solution to these problems is not unique. Yet, one of those solutions to both these types are given below:

4-Queens problem

In the **4-Queens problem**, we have a **4x4 grid** and we have **4 queens** to place on it. The layout for the **4-Queens problem** while satisfying all the constraints is as follows:

		Q	
Q			
			Q
	Q		

	Q		
			Q
Q			
		Q	

8-Queens problem

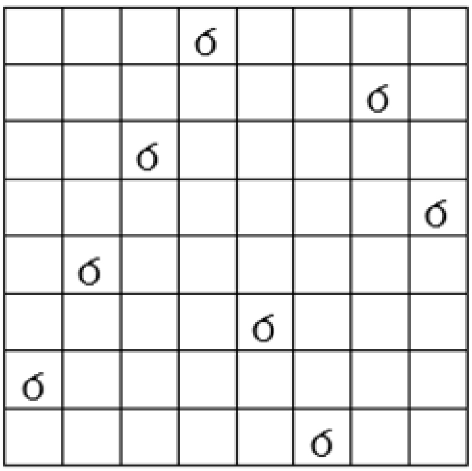
In the **8-Queens problem**, we have an **8x8 grid** and we have **8-queens** to place on it. The layout for the **8-Queens problem** while satisfying all the constraints is as follows:

					Q		
Q							
				Q			
	Q						
							Q
		Q					
						Q	
			Q				

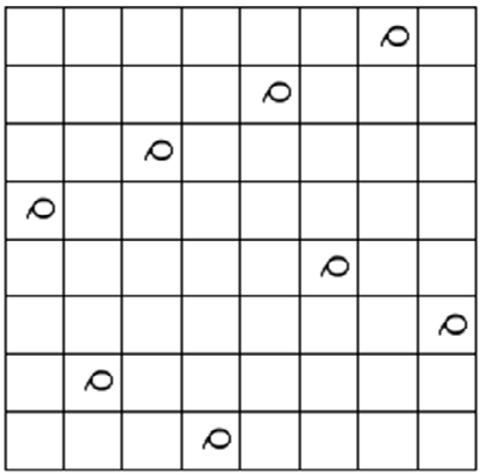
Original

		Q					
							Q
			Q				
						Q	
Q							
					Q		
	Q						
				Q			

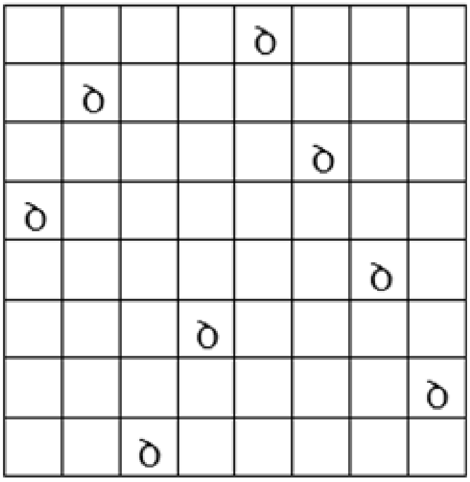
Mirroring(L=>R or R=>L)



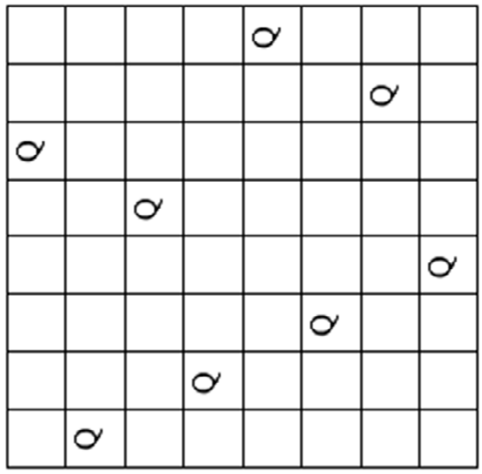
Mirroring(U=>D or D=>U)



90° Rotation

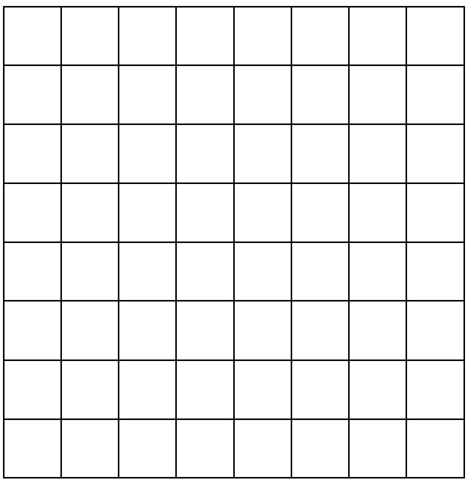


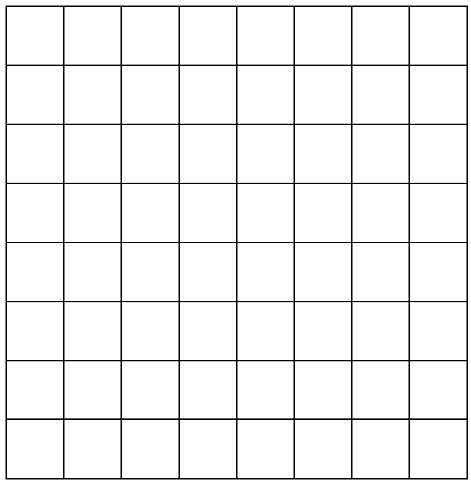
180° Rotation

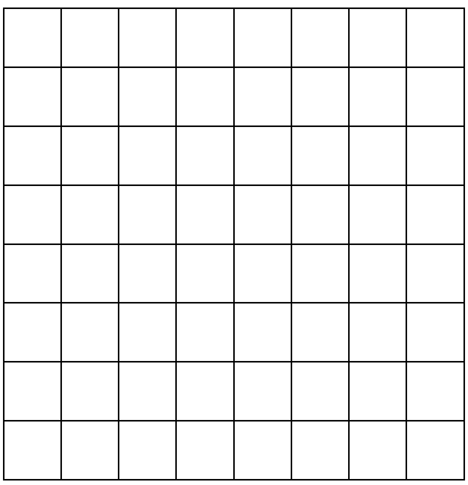


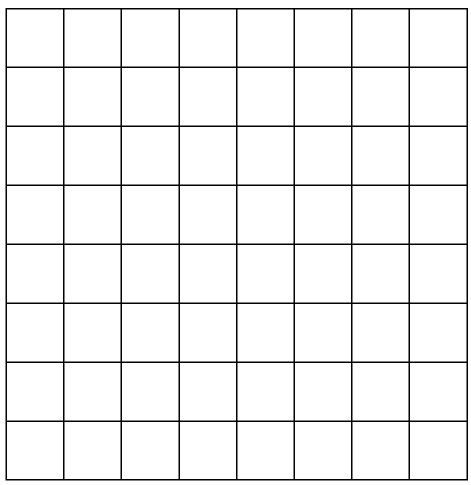
270° Rotation

Please try to solve other ways: *(Use Pencil and Eraser)*



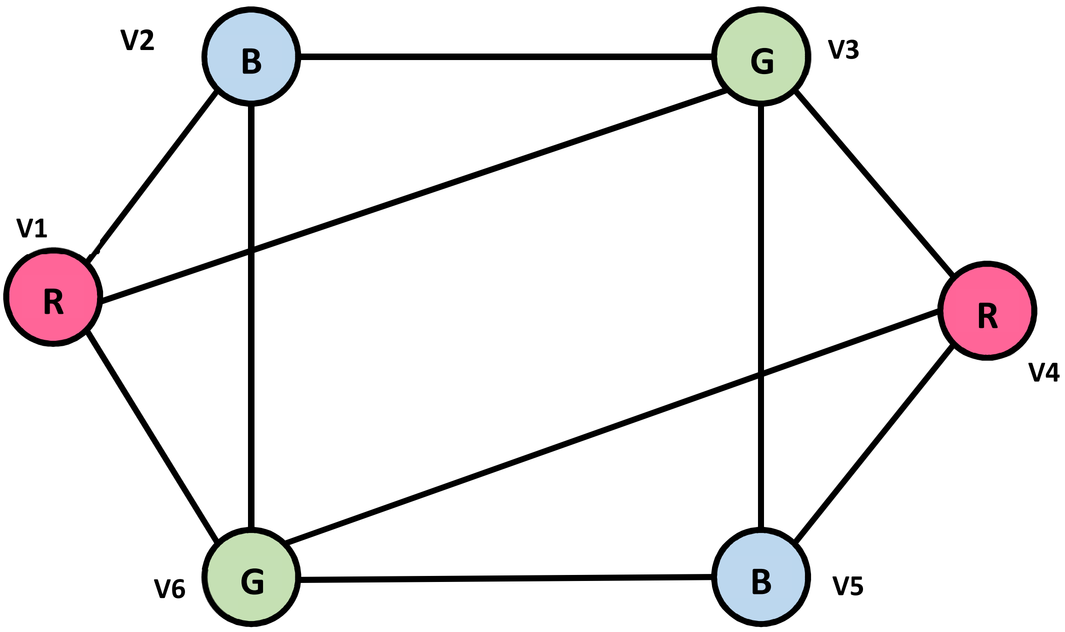






Graph Coloring Problem/ Map Coloring Problem:

A classic CSP is the problem of coloring a map so that no adjacent regions have the same color.



Variables = V_1, V_2, V_3, V_4, V_6

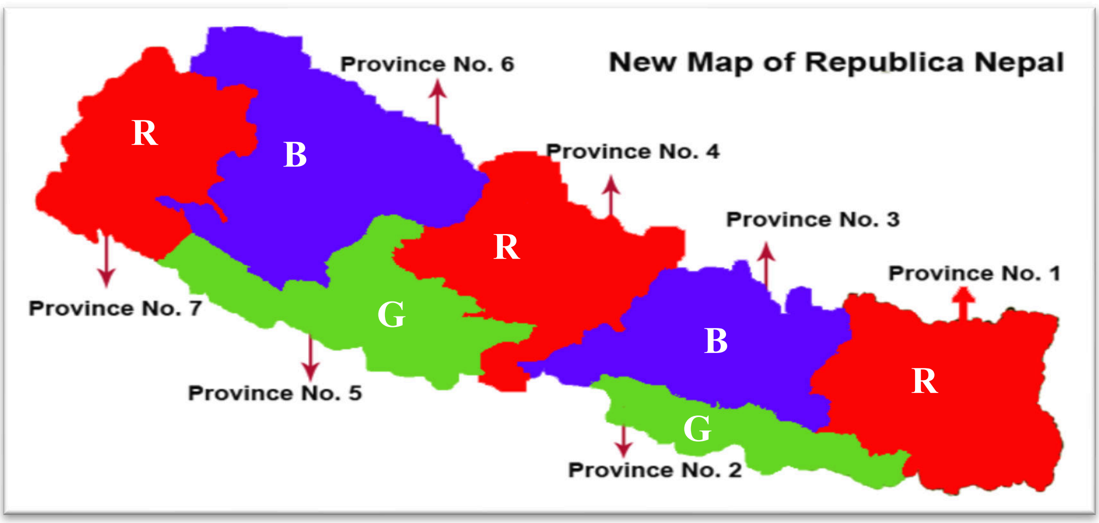
Domain = R, G, B

Constraints = adjacent regions must have different colors

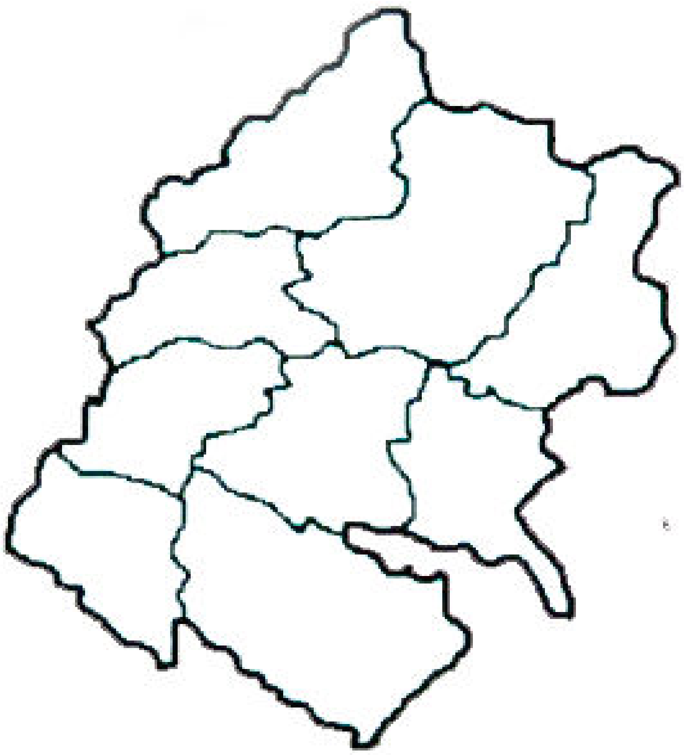
E.g: $V_1 \neq V_2$, $V_1 \neq V_6$, $V_1 \neq V3$

Means, $V_1 \neq V_2$, or ($V_1 \neq V_2$) in {(R,G), (R,B), (G,R), (G,B), (B,R), (B,G)}

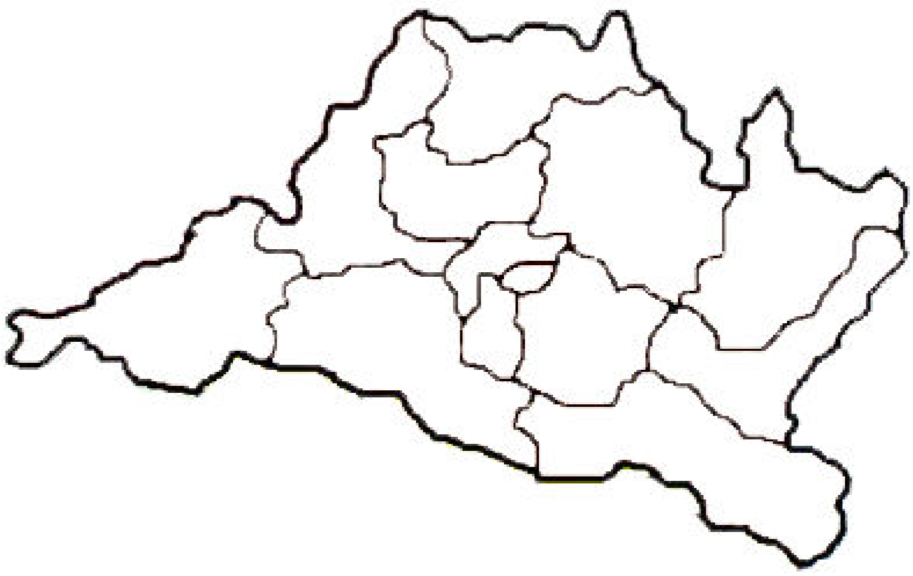
The Map of Nepal can be colored with 3 colors (R, G, B).



Q1. Can the map of Sudur Paschim Pradesh (Province 7) be colored with 3 Color? If possible, try it out.



Q2. Can the map of (Province 3) be colored with 3 Color?.



Department of Computer Science – DEC [NAST]

Crypt-Arithmetic Problem:

The **Crypt-Arithmetic problem in** Artificial Intelligence is a type of encryption problem in which the written message in an alphabetical form which is easily readable and understandable is converted into a numeric form which is neither easily readable nor understandable. In simpler words, the crypt-arithmetic problem deals with the converting of the message from the readable plain text to the non-readable ciphertext. The constraints which this problem follows during the conversion is as follows:

1. A number 0-9 is assigned to a particular alphabet.
2. Each different alphabet has a unique number.
3. All the same, alphabets have the same numbers.
4. The numbers should satisfy all the operations that any normal number does.

Let us take an example of the message: SEND + MORE = MONEY.

Here, to convert it into numeric form, we first split each word separately and represent it as follows:

```
      S   E   N   D
      M   O   R   E
-------------------------------
  M   O   N   E   Y
```

These alphabets then are replaced by numbers such that all the constraints are satisfied. So initially we have all blank spaces.

We first look for the MSB in the last word which is **'M'** in the word **'MONEY'** here. It is the letter which is generated by carrying. So, carry generated can be only one. SO, we have **M=1**.

Now, we have **S+M=O** in the second column from the left side. Here **M=1**. Therefore, we have, **S+1=O**. So, we need a number for **S** such that it generates a carry when added with **1**. And such a number is **9**. Therefore, we have **S=9** and **O=0**.

Now, in the next column from the same side we have **E+O=N**. Here we have **O=0**. Which means **E+0=N** which is not possible. This means a carry was generated by the lower place digits. So, we have:

1+E=N ----------(i)

Next alphabets that we have are **N+R=E -------(ii)**

So, for satisfying both equations **(i) and (ii)**, we get **E=5 and N=6**.

Now, **R** should be **9**, but **9** is already assigned to **S**, So, **R=8** and we have **1** as a carry which is generated from the lower place digits.

Department of Computer Science – DEC [NAST]

Now, we have **D+5=Y** and this should generate a carry. Therefore, **D** should be greater than **4**. As **5, 6, 8** and **9** are already assigned, we have **D=7** and therefore **Y=2**.

Therefore, the **solution to the given Crypt-Arithmetic problem is**:

S=9; E=5; N=6; D=7; M=1; O=0; R=8; Y=2

Which can be shown in layout form as:

```
      9   5   6   7
      1   0   8   5
  --------------------------------
  1   0   6   5   2
```

Some Examples of Cryptarithmetic Problem with Solution

(ADDITION)

1. LET + LEE = ALL

```
      L   E   T              1   5   6
      L   E   E              1   5   5
  ----------------------   ----------------------
      A   L   L              3   1   1
```

Hint: If Two characters are same then the result will be Even Number.

Here, L + L = any even number, so 1 + 1 = 2, *(if it takes carry then add it)*
E + E = any even number, so 5 + 5 = 10 *(0 is the even number)*

2. KANSAS + OHIO = OREGON

```
K   A   N   S   A   S          4   9   7   2   9   2
        O   H   I   O                  5   8   6   5
--------------------------    --------------------------
O   R   E   G   O   N          5   0   3   1   5   7
```

3. HERE + SHE = COMES

	H	E	R	E			9	4	5	4
		S	H	E				8	9	4
C	O	M	E	S		1	0	3	4	8

E + E = 8
then,,E = 4 AND S = 8,
E + S = M
4 + 8 = 12
M = 2
H + 1 = 10 so, O = 0 C = 1,
now R + H = E = 4
H = 9
so R has to be 5 as 5 + 9 = 14 which leaves carry 1..
so M = 3
so R + H + O = 5 + 9 + 0 = 14

4. POINT + ZERO = ENERGY

	P	O	I	N	T			9	8	5	0	4
		Z	E	R	O				3	1	6	8
E	N	E	R	G	Y		1	0	1	6	7	2

5. GO + TO = OUT

	G	O			8	1
	T	O			2	1
O	U	T		1	0	2

Clearly, O = 1., as it is the carry generated by G + T. a number cannot start from 0 in cryptarithmetic addition.

Since O = 1, O + O = 1 + 1 =2. So, T = 2.
G + 2 = 10 + U.
If G = 9, U = 1. Which is not valid since O = 1.
So, G = 8 and U = 0.
Hence, O + U + T = 1 + 0 + 2 = 3

6. USA + USSR = PEACE

```
        U   S   A                 9   3   2
    U   S   S   R             9   3   3   8
--------------------       --------------------
P   E   A   C   E          1  0   2   7   0
```

USA + USSR = PEACE
Here P is carry , P = 1
when P = 1, E = 0 with carry 1 AND U = 9
A + R = E = 0 with carry 1.
so, A = 2 and R = 8
U + S = A = 2 with carry 1, S = 3
S + S + 1 = C, 3 + 3 + 1 = C = 7

7. EVER + SINCE = DARWIN

```
        E   V   E   R              5   6   5   3
    S   I   N   C   E          9   7   8   2   5
------------------------    ------------------------
D   A   R   W   I   N       1  0   3   4   7   8
```

8. EAT + EAT + EAT = BEET

```
     E   A   T            4   8   0
     E   A   T            4   8   0
     E   A   T            4   8   0
 ----------------     ----------------
 B   E   E   T        1   4   4   0
```

8 + 8 + 8 = 24
2 carry over
4 + 4 + 4 = 12 + 2 = 14
So, BEET = 1440
Note, In 3 word addition if there is carry over then count only 2

9. E A T + T H A T = A P P L E

```
         E   A   T              8   1   9
     T   H   A   T          9   2   1   9
 --------------------     --------------------
 A   P   P   L   E      1   0   0   3   8
```

digit number (EAT) + 4 digit number(THAT) = 5 digit number(APPLE)
If so, then A can be 1 and P can be 0.
Again here T is yielding a two digit number(10).
So there must be a carry 1 and T = 9.
Now the expression becomes
E 1 9
9 H 1 9
———————-
1 0 0 L E
Now it is clear that E = 8 and L = 3

10. NINE + FINE = WIVES

	N	I	N	E
	F	I	N	E
W	I	V	E	S

	N	I	N	E
	F	I	N	E
W	I	V	E	S

11. WAIT + ALL = GIFTS

	W	A	I	T
		A	L	L
G	I	F	T	S

	9	6	0	8
		6	7	7
1	0	2	8	5

12. FORTY + TEN + TEN = SIXTY

F	O	R	T	Y
		T	E	N
		T	E	N
S	I	X	T	Y

2	9	7	8	6
		8	5	0
		8	5	0
3	1	4	8	6

13. SCOOBY + DOOO = BUSTED

S	C	O	O	B	Y
		D	O	O	O
B	U	S	T	E	D

1	9	4	4	2	3
		7	4	4	4
2	0	1	8	6	7

14. S C O O B Y + D O O = B L I N K S

S	C	O	O	B	Y
			D	O	O
B	L	I	N	K	S

3	6	O	O		1
			D	2	2
3	6	2	9	5	3

Y + O = S, B + O = K, O + D = N , O = I , C = L ,S = B
so, we can take values S&B = 3, C&L = 6, O&I = 2 Y = 1, D = 7,N = 9,K = 5 .

15. BANANA + GUAVA = ORANGE

B	A	N	A	N	A
	G	U	A	V	A
O	R	A	N	G	E

2	4	9	4	9	4
	6	5	4	7	4
3	1	4	9	6	8

16. HOW + MUCH = POWER

		H	O	W
	M	U	C	H
P	O	W	E	R

		7	0	5
	9	8	3	7
1	0	5	4	2

17. DAYS + TOO = SHORT

	D	A	Y	S
		T	O	O
S	H	O	R	T

	9	7	4	3
		5	2	2
1	0	2	6	5

18. TWO + DAYS = MORE

	T	W	O			8	9	3
D	A	Y	S		6	4	1	2
----------------------------					----------------------------			
M	O	R	E		7	3	0	5

19. CROSS + ROADS = DANGER

	C	R	O	S	S			9	6	2	3	3
	R	O	A	D	S			6	2	5	1	3
		----------------------------						----------------------------				
D	A	N	G	E	R		1	5	8	7	4	6

20. YOUR + YOU = HEART

	Y	O	U	R			9	4	2	6
		Y	O	U				9	4	2
----------------------------						----------------------------				
H	E	A	R	T		1	0	3	6	8

21. TWO + TWO = FOUR

	T	W	O			9	3	8
	T	W	O			9	3	8
----------------------------					----------------------------			
F	O	U	R		1	8	7	6

Here are other possibilities:

938+938=1876
928+928=1856
867+867=1734
846+846=1692
836+836=1672
765+765=1530
734+734=1468

22. ABC + DEF = GHI

A	B	C		2	6	9
D	E	F		1	3	8
G	H	I		4	0	7

This question may have thousand solutions because the characters are distinct. Make sure A + D must not generate a carry.

23. AB + CD = EF

A	B		4	9
C	D		1	8
E	F		6	7

This question may have more than hundreds of solutions because the characters are distinct. Make sure A + C must not generate a carry.

Department of Computer Science – DEC [NAST]

2.5 Game Playing

A Game can be defined by the following:

- ✓ initial state (How the board is setup).
- ✓ the operation (which define the legal moves.
- ✓ a terminal test (which says when the game is over). and
- ✓ a utility or payoff function (which says who won and by how much).

Perfect Information: Information is *shown* or clear. See the exact state to the game. Chess is an example of a game with perfect information as each player can see all the pieces on the board at all times. Other examples of games with perfect information include tic-tac-toe, checkers, infinite chess, and Go

Imperfect Information: Information is Hidden. Card games where each player's cards are *hidden* from other players such as poker and bridge are examples of games with imperfect information.

Game Playing Problem:

- ✓ States where the game has ended are called terminal states.
- ✓ In two-player game, assume one is called MAX (tries maximize utility) and one is called MIN.
- ✓ In the search tree, first layer is more by MAX, next layer by MIN, and alternate to terminal state.
- ✓ Each layer in the search is called a ply.
  *a **ply** is one turn taken by one of the players.*

Game Playing is an important domain of artificial intelligence. Games don't require much knowledge; the only knowledge we need to provide is the rules, legal moves and the conditions of winning or losing the game.

Both players try to win the game. So, both of them try to make the best move possible at each turn. Searching techniques like BFS (Breadth First Search) are not accurate for this as the branching factor is very high, so searching will take a lot of time.

So, we need another search procedure that improve –

Generate procedure so that only good moves are generated.

Test procedure so that the best move can be explored first.

Sometimes Generate and Test procedure is known as **Hit and Trial**.

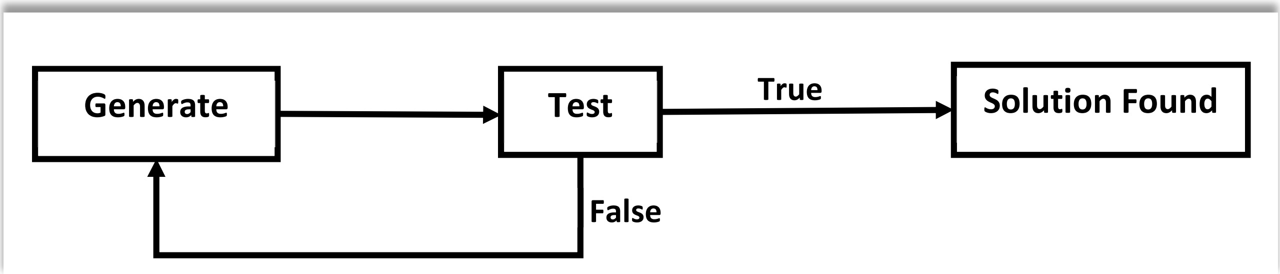


Fig.: Flow of Generate and Test Procedure

Algorithm:

1. Generate a possible solution.
2. Test to see if this is actually a solution.
3. Quit if a solution has been found otherwise - return to step 1.

The most common search technique in game playing is Minimax search procedure. It is depth-first depth-limited search procedure. It is used for games like chess and tic-tac-toe.

Examples: MiniMax algorithm, Alpha Beta Pruning.

These examples will be discussed in the next chapter

2.6 Production System

Rule

Situation/Condition ➔ **Action**

LHS (Left Hand Side) ***RHS (Right Hand Side)***

A production system is nothing but a set of rules, each rule consists of LHS and RHS where, LHS patterns determines the applicability of rules and a RHS describes the operations to be performed if the rule is applied.

Simple, Production system are the rules of the form C ➔ A, where LHS is known Condition and RHS is known as Action.

LHS described as Applicability of Rule.

RHS described Operation to be Performed.

For Example. Water Jug Problem, Missionaries and Cannibal ***(These problem production rule systems are already done. See above)***

More Generalize Definition:

If one adopts a system with production rule and rule Interpreter then that system is known as Production system.

Production System is a model of computation that provides pattern directed search control using set of:

1. **Production Rules**
   - ✓ $C_i \Rightarrow A_i$
     1. C_i = Conditional Part (If…. Else)
     2. A_i = Action Part
        - ✓ Primitive to Action
        - ✓ A call to another production rule
        - ✓ Set of instructions/actions/programs
2. **Working Memory (Knowledge Database)**
   - ✓ Information appropriate for particular task.
3. **Control Strategy**
   - ✓ Specifies the order in which rule will be applied and it resolves conflicts if any
   - ✓
4. **Rule Applier**
   - ✓ Checks current state with LHS of rule in Knowledge Database and finds appropriate rule to apply

Types of Production System or Characteristics:

- ✓ Monotonic Production System
- ✓ Non-Monotonic Production System
- ✓ Partially Commutative Production System
- ✓ Commutative Production System

1. **Monotonic Production System:** In monotonic production system all conclusions are still valid after adding information to the existing information. Example: Theorem Proving (Pythagoras Theorem)

2. **Non-Monotonic Production System:** In non-monotonic production system some conclusion can be invalidated by adding more information to the existing information. Example: Robot Navigation (GPS). Like we human beings even change our conclusion if new information is added.

3. **Partially Commutative Production System:** A system in which application of a particular sequence of rules transforms state x to state y then any permutation of these

rules that is allowable also transform state x to state y. Example: ABCD is applied then it takes to x=> y If I use Permutation of ABCD or Combination of ABCD like BCDA then also it takes stare x to state y. || AxB = BxA

4. **Commutative Production System:** Combination of Monotonic + Partially

Q. Define and describe the difference between knowledge, belief, hypothesis and data.

Ans:

Knowledge: can be defined as the body of facts and principles accumulated by humankind or the act, fact, or state of knowing.

Belief: It is defined as essentially any meaningful and coherent expression that can be represented. Thus, a belief can be true or false.

Hypothesis: It is defined as justified belief that is not known to be true. Thus, a hypothesis is a belief that is backed up with some supporting evidence, but it may still be false. In other words, it is a preliminary assumption or tentative explanation that accounts for a set of facts, taken to be true for the purpose of investigation and testing.

Data: Data in computer terminology mean raw facts and figures. For example, 'DBK', 11716713, 'A' are data. Data are processed to form information.

Information: Data arranged in useful and meaningful form is known as information. For example, 'DBK, whose roll number is 11716713, has got grade A' is an information as it is conveying some meaning.

THE END