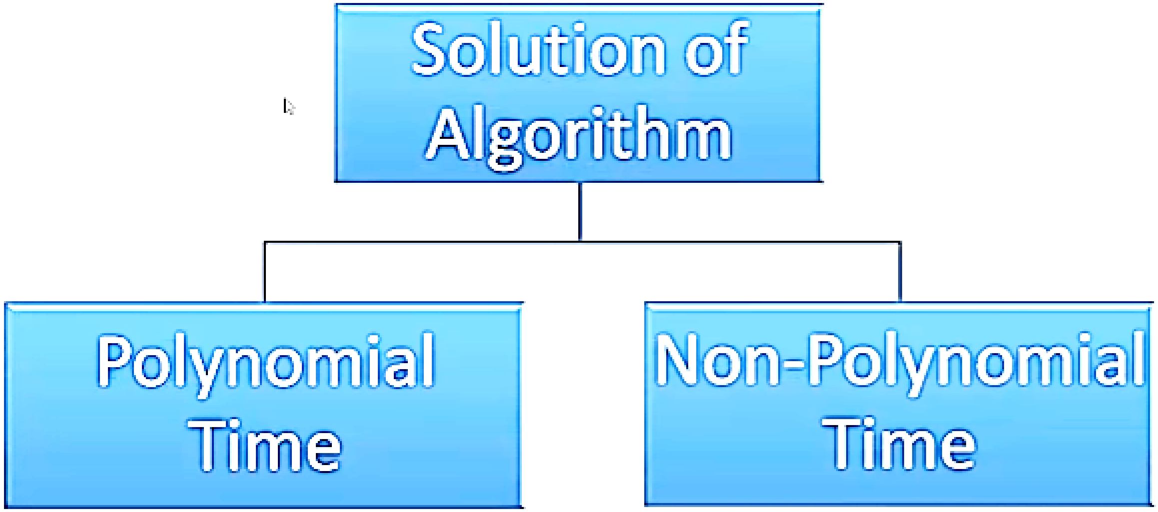
Artificial Intelligence

Combinatorial Explosion

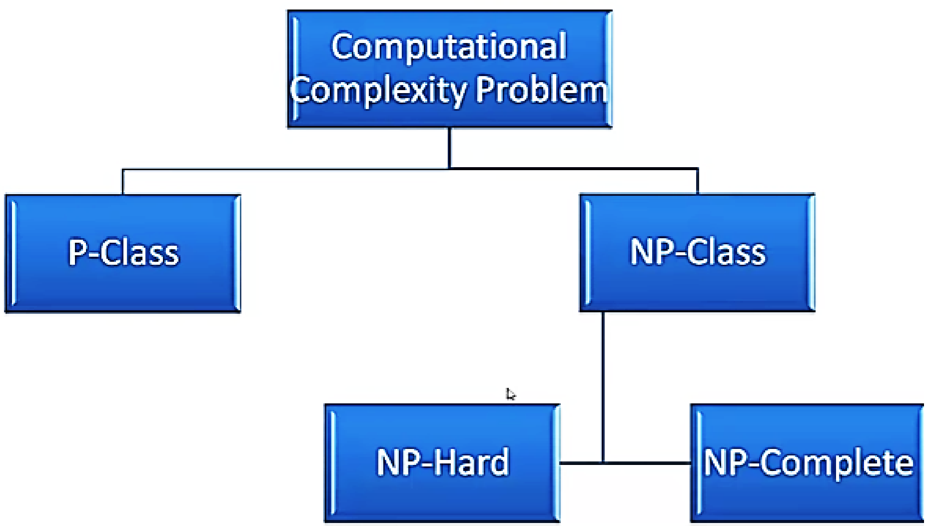
- Problems that involve assigning values to a set of variables can grow exponentially with the number of variables. This is the problem of combinatorial explosion.
- Some such problems can be extremely hard to solve (NP-Complete, NP-Hard).
- Selecting the correct representation can help to reduce this, as can using heuristics

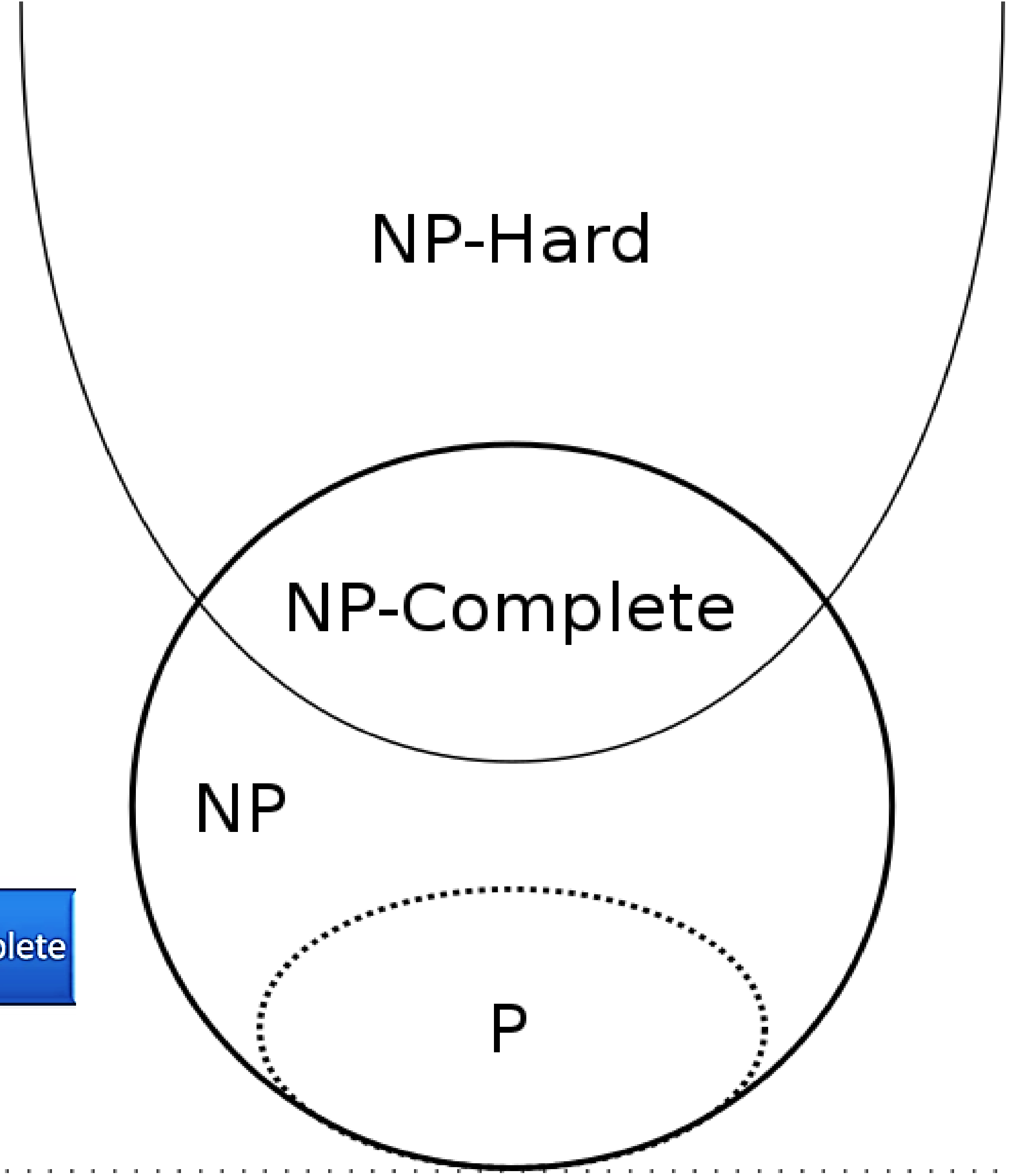
Note that


Solution of Algorithm
Polynomial Time
Non-Polynomial Time

HW

- **P** problems that are solvable in polynomial time. (sorting & searching algorithms
- **NP** = Non-Deterministic polynomial time. NP means verifiable in polynomial time.(hard to solve and easy to verify) (TSP, scheduling and Sudoku)
- **NP-Hard** Problems, there is no known polynomial time solution
- **NP-Complete**, NP & NP-Hard
- Note that : all NP-complete problems are NP-Hard Problems but all NP-Hard problems are not NP-complete.

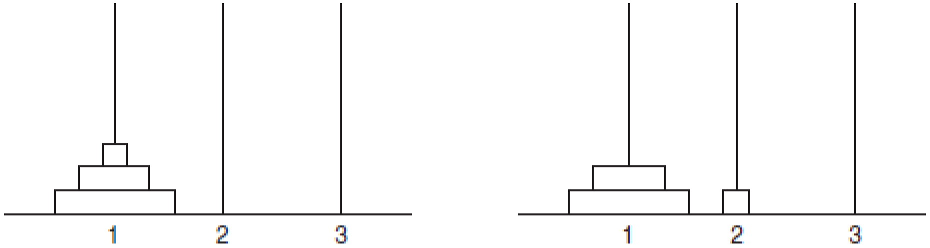

Computational Complexity Problem
P-Class
NP-Class
NP-Hard
NP-Complete


NP-Hard
NP-Complete
NP
P

Problem Reduction

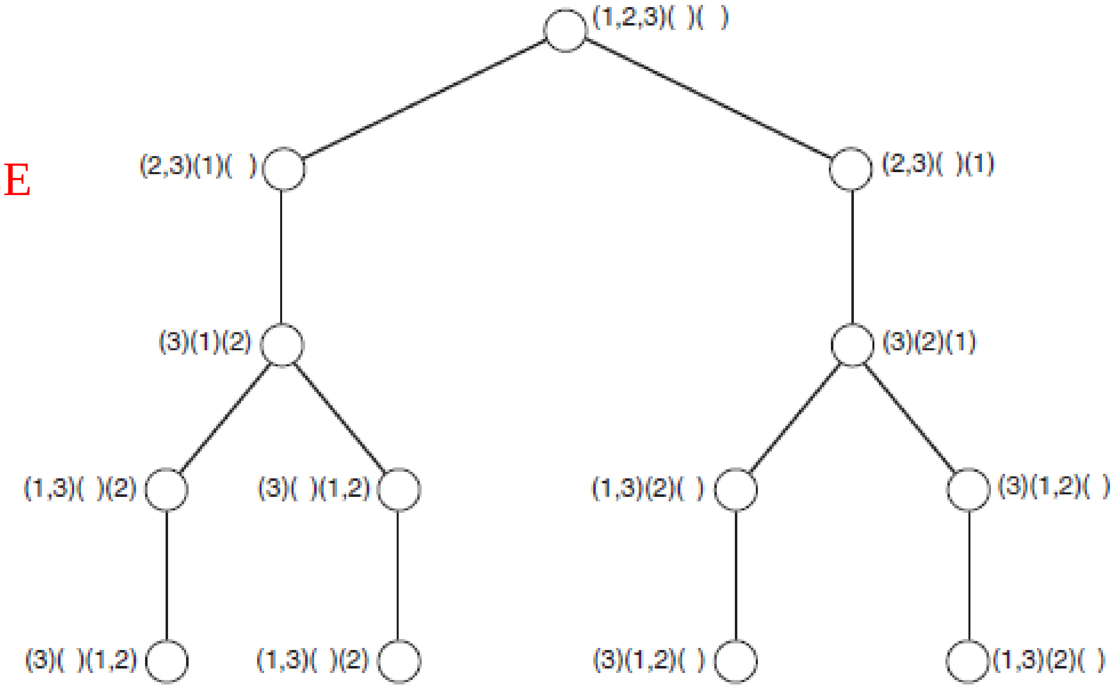
- Breaking a problem down into smaller sub-problems (or sub-goals).
- Can be represented using goal trees (or and-or trees).
- Nodes in the tree represent sub-problems.
- The root node represents the overall problem.
- Some nodes are AND nodes, meaning all their children must be solved.

Example: Towers of Hanoi Problem



$2^n - 1$

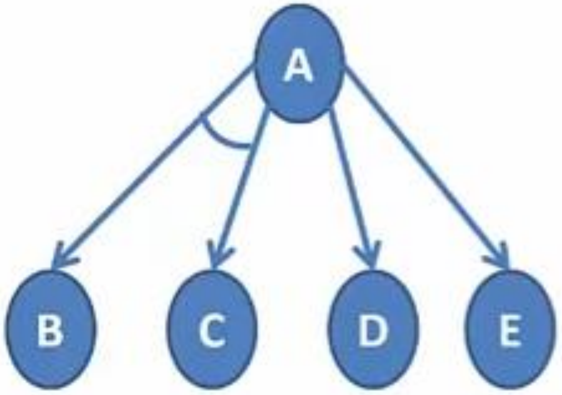
TREE



Goal Trees

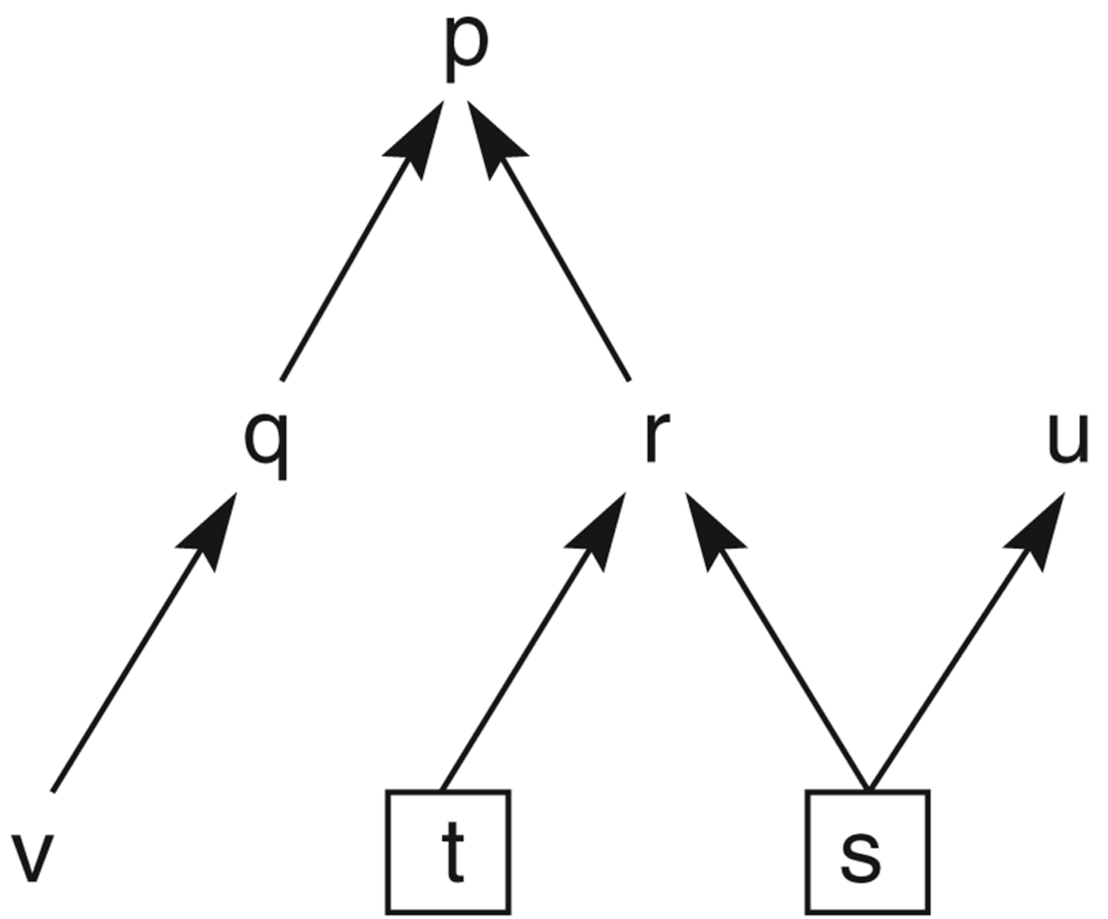
- A **goal tree** also called an **and-or tree.**
- Useful for representing the solution of problems that can be solved by decomposing them into a set of smaller problems, all of which must than be solved.
- **and-or tree** is a graphical representation of the reduction of problems (goals) to AND /OR of sub-problems (sub-goals) [represents the search space for solving the problem]

Example #1



Example #2

q $\rightarrow$ **p**
r $\rightarrow$ **p**
v $\rightarrow$ **q**
s $\rightarrow$ **r**
t $\rightarrow$ **r**
s $\rightarrow$ **u**
s
t



Example #3

a

b

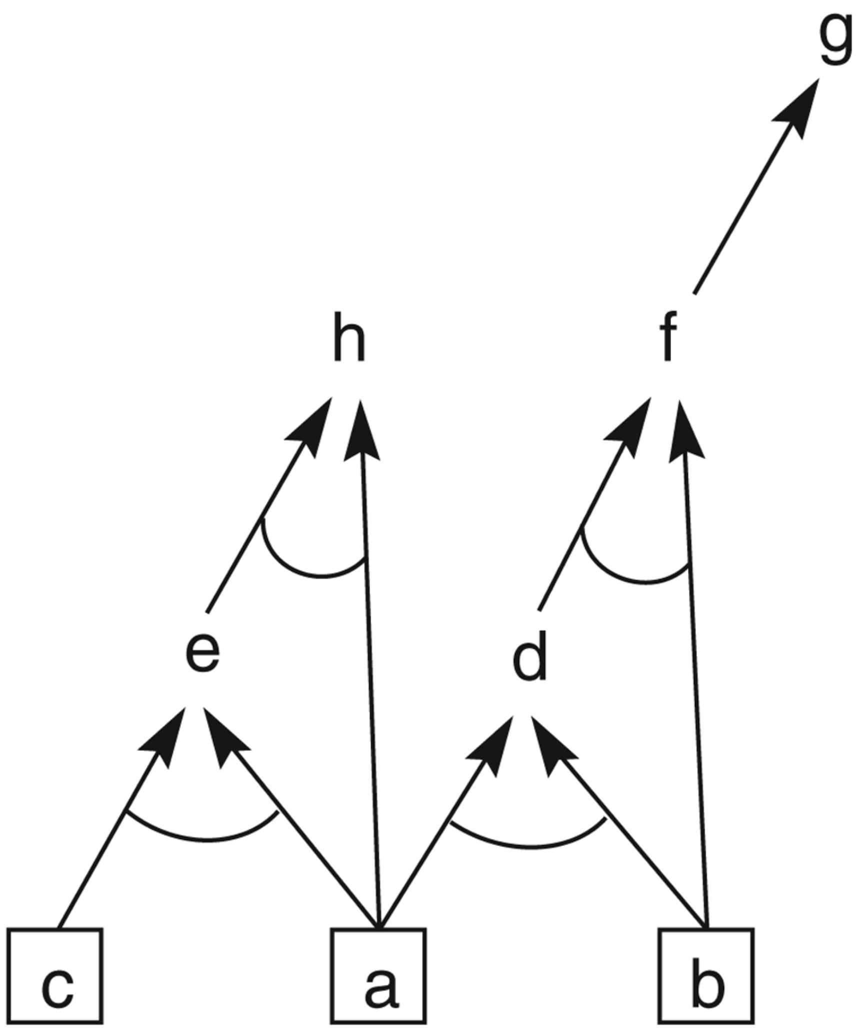
c

$\mathbf{a} \wedge \mathbf{b} \rightarrow \mathbf{d}$

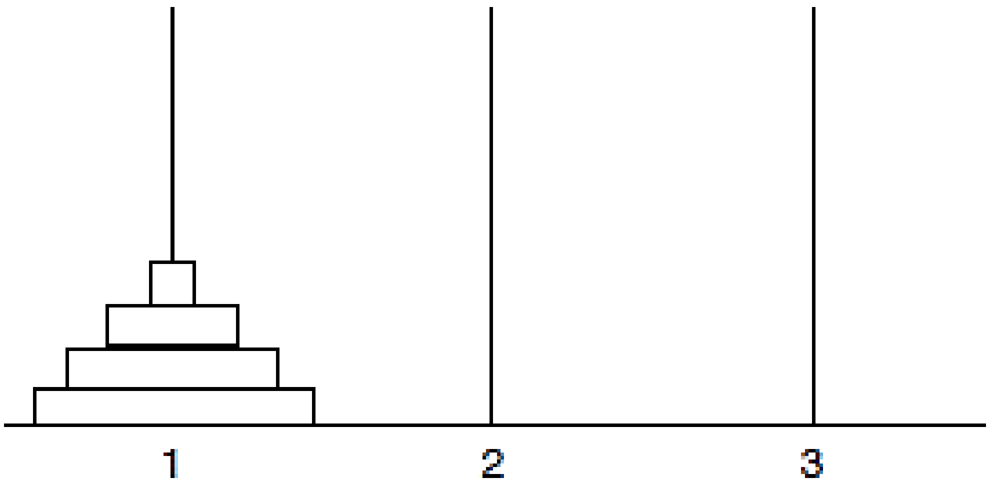
$\mathbf{a} \wedge \mathbf{c} \rightarrow \mathbf{e}$

$\mathbf{b} \wedge \mathbf{d} \rightarrow \mathbf{f}$

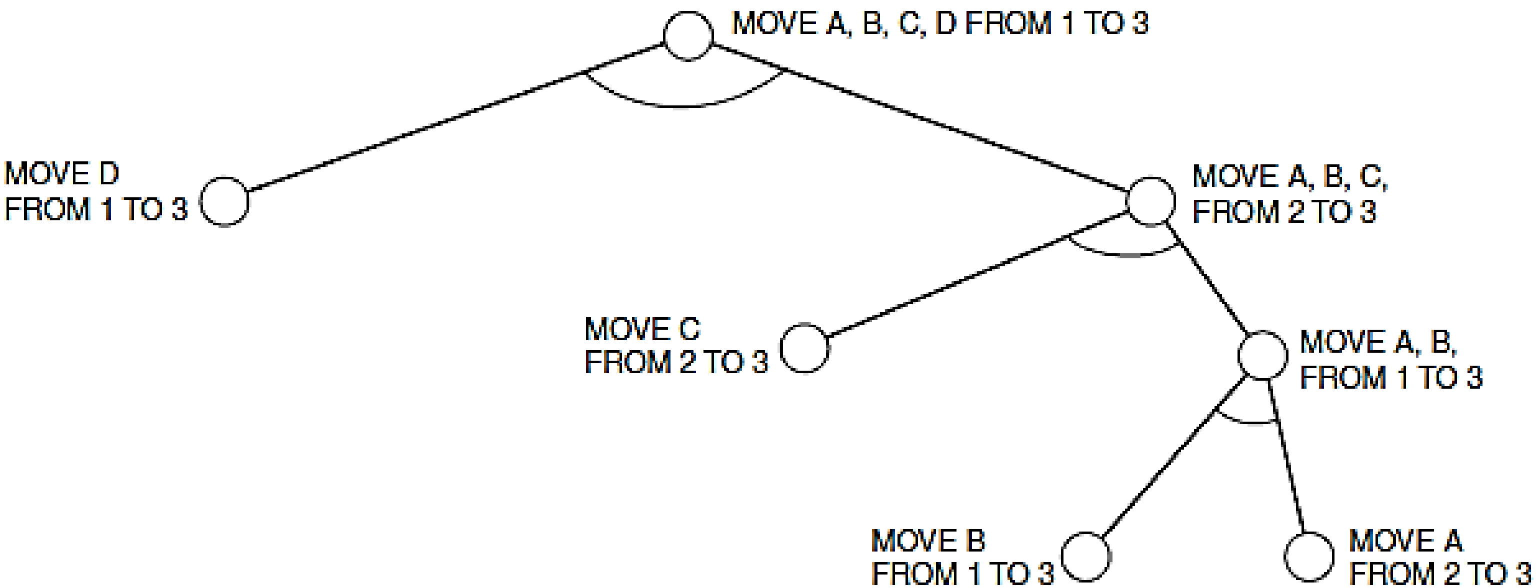
$\mathbf{f} \rightarrow \mathbf{g}$



Question



Solution


MOVE A, B, C, D FROM 1 TO 3
MOVE D
FROM 1 TO 3
MOVE A, B, C,
FROM 2 TO 3
MOVE C
FROM 2 TO 3
MOVE A, B,
FROM 1 TO 3
MOVE B
FROM 1 TO 3
MOVE A
FROM 2 TO 3

Cryptarithmetic Problem

- Find an assignment of digits (0, ..., 9) to letters so that a given arithmetic expression is true. examples: SEND + MORE = MONEY
- and

```
 FORTY     Solution:  29786
+  TEN                  850
+  TEN                  850
 -----                -----
 SIXTY                31486
F=2, O=9, R=7, etc.
```

- Note: In this problem, the solution is NOT a sequence of actions that transforms the initial state into the goal state; rather, the solution is a goal node that includes an assignment of digits to each of the distinct letters in the given problem.

Example

Solve the following Expression

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

Solution

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

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

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

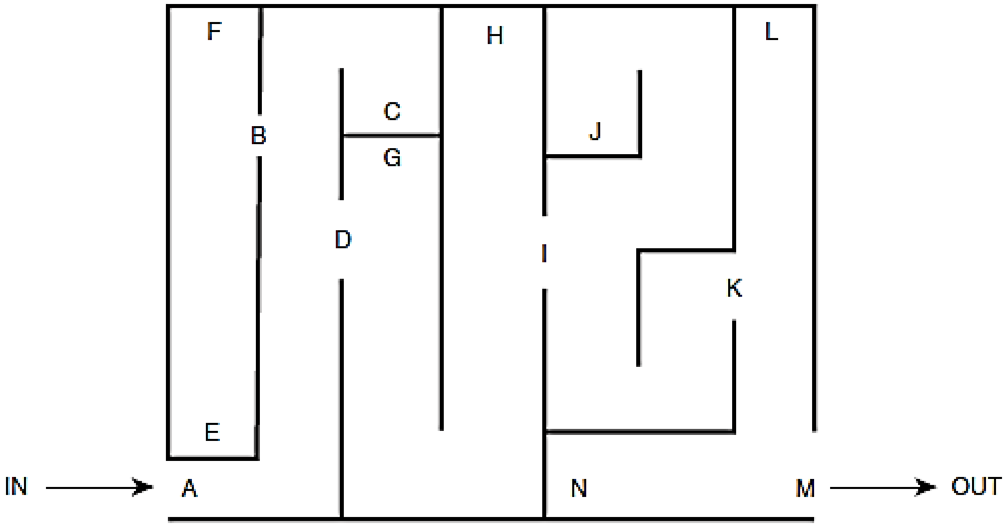
HW

```
      B   A   S   E
+     B   A   L   L
-----------------------
  G   A   M   E   S
```

Character	Code
B	7
A	4
S	8
E	3
L	5
M	9
G	1

Quiz #1

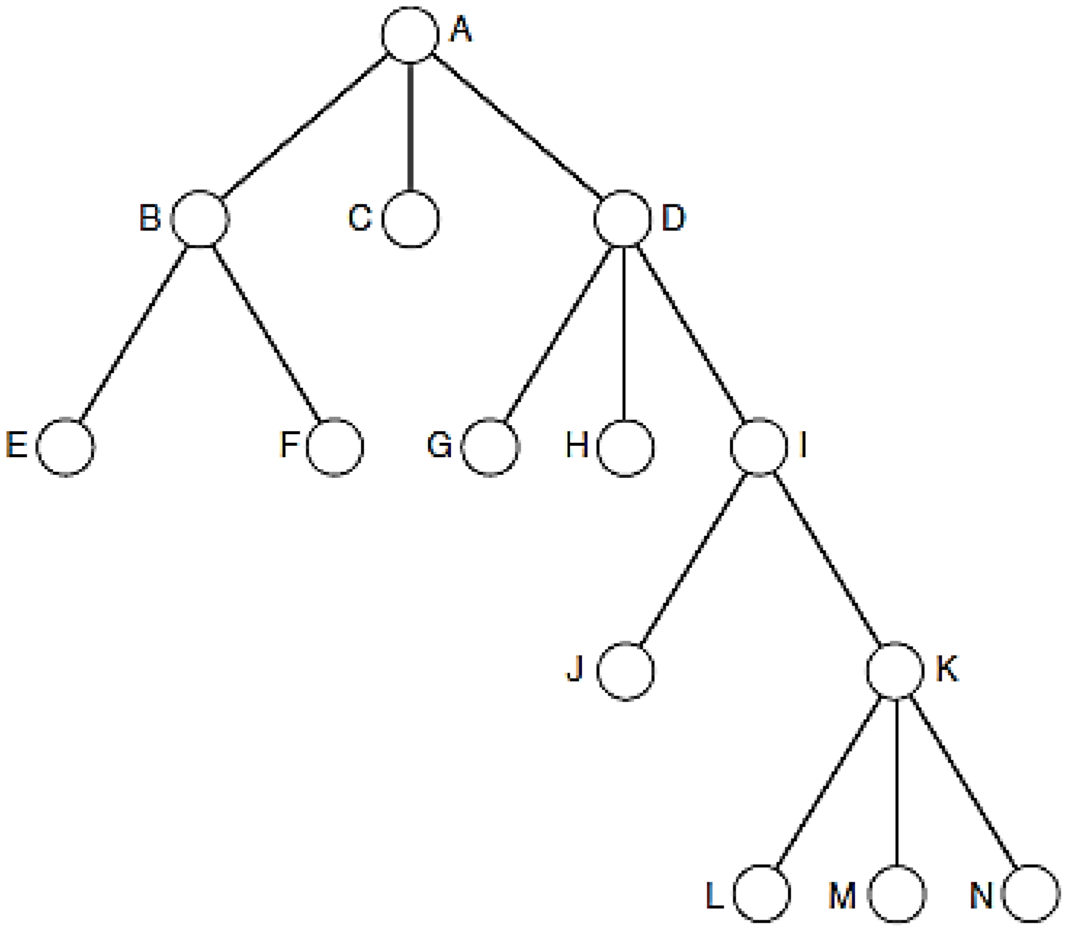
Q1: Draw a search tree for the below problem.



<u>Q2:</u> With reference to the Travelling Salesman Problem explain what is meant by combinatorial explosion and what effect it has in finding an optimal solution?

<u>Q3:</u> Briefly describe the Turing Test

Solution



Production Systems

- The production system is a model of computation that has proved particularly important in AI, both for implementing search algorithms and for modeling human problem solving.
- A production system provides pattern-directed control of a problem-solving process and consists of a set of production rules, a working memory, and a recognize-act control cycle
- Production system are rules of the form x→y , where LHS is known as condition and RHS is action , LHS describe applicability of the rule & RHS describes operation to be performed

Example

- **Trace of a simple production system used for sorting a string composed of letters a,b and c.**

Production set:

1. ba $\rightarrow$ ab
2. ca $\rightarrow$ ac
3. cb $\rightarrow$ bc

Iteration #	Working memory	Conflict set	Rule fired
0	cbaca	1, 2, 3	1
1	cabca	2	2
2	acbca	2, 3	2
3	acbac	1, 3	1
4	acabc	2	2
5	aacbc	3	3
6	aabcc	Ø	Halt

Control of Search in Production Systems

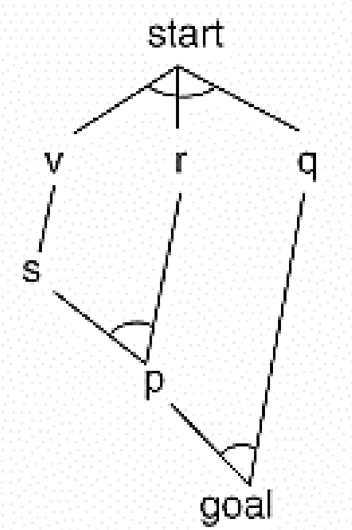
Data-Driven search (forward chaining)

- A set of data-driven search on a set of productions expressed as propositional calculus implications
- The conflict resolution strategy is a simple one of choosing the enabled rule that has fired least recently
- in the case of tie, the first rule is chosen
- Execution halts when a goal is reached

Example#1

Production set:

1. p ∧ q → goal
2. r ∧ s → p
3. w ∧ r → q
4. t ∧ u → q
5. v → s
6. start → v ∧ r ∧ q



Trace of execution:

Iteration #	Working memory	Conflict set	Rule fired
0	start	6	6
1	start, v, r, q	6, 5	5
2	start, v, r, q, s	6, 5, 2	2
3	start, v, r, q, s, p	6, 5, 2, 1	1
4	start, v, r, q, s, p, goal	6, 5, 2, 1	halt

Example 2

- Given

a

b

c

a→ d

b^c→ r

b v f→k

r ^ k→h

Data driven state graph

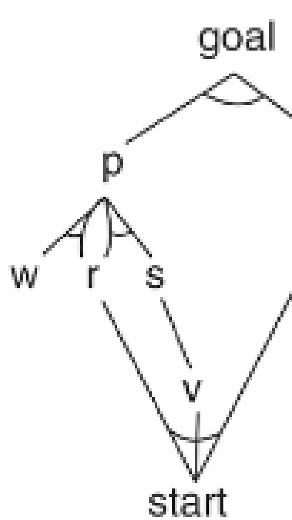
Goal-Driven search (**Backward chaining**)

- begins with a goal
- works backward to the facts of the problem to satisfy that goal
- the goal is placed in working memory and
- matched against the ACTIONs of the production rules
- all production rules whose conclusion ACTIONs match the goal form the conflict set

Example #1

Production set:

1. p ∧ q → goal
2. r ∧ s → p
3. w ∧ r → p
4. t ∧ u → q
5. v → s
6. start → v ∧ r ∧ q



Trace of execution:

Iteration #	Working memory	Conflict set	Rule fired
0	goal	1	1
1	goal, p, q	1, 2, 3, 4	2
2	goal, p, q, r, s	1, 2, 3, 4, 5	3
3	goal, p, q, r, s, w	1, 2, 3, 4, 5	4
4	goal, p, q, r, s, w, t, u	1, 2, 3, 4, 5	5
5	goal, p, q, r, s, w, t, u, v	1, 2, 3, 4, 5, 6	6
6	goal, p, q, r, s, w, t, u, v, start	1, 2, 3, 4, 5, 6	halt

Example #2

- Given

a

b

c

a→ d

b^c→ r

b v f→k

r ^ k→h

Goal driven state graph