Md. Sahil, 00171050102?
287mdsahil@gmail.com
8972931876

1.

a) The Turing Test is a test developed ~~by~~ to test a machine's ability to immitate ~~a~~ human-like behaviour.

It involves 3 rooms containing a person, a computer and an interrogator. The interrogator can communicate with the other two by teletype. The interrogator tries to distinguish between the person and the machine. If the machiny is succesfull in fooling the interrogator into believing that it is a person than the machine passes the test and is considered intelligent.

The turing test is a one-sided test, Even if the machines are not actually "intelligent", they can still use some kind of trickery to pass the test. ~~and~~

Most programs stress on simple syntactical analysis and generation of sentences using pattern matching ~~with~~ with known sentences, vocabulary and key words.

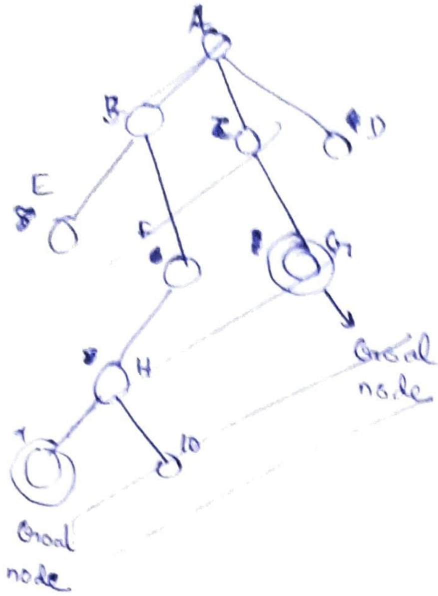
Thus ~~the~~ Turing test is not a good way to ~~&~~ judge a rational agent.

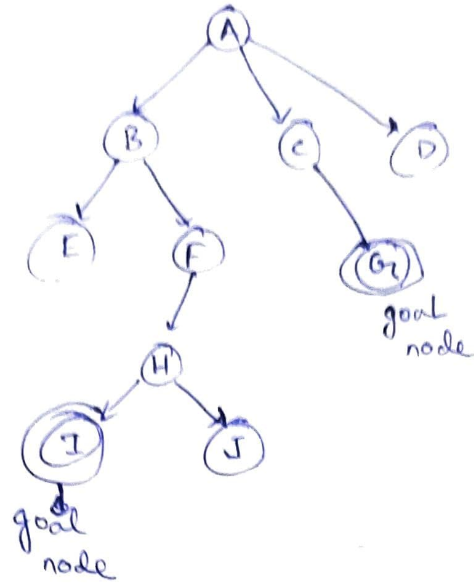
2.

a) Iterative Deepening search involves performing ~~a~~ depth Limited Search ~~at~~ ~~iaa~~ with ~~increasing~~ interatively increasing depth until a solution is found.

Depth Limited DFS it self is not optimal. The goal node encountered may not be the optimal solution ~~A~~, as shown in the diagram.

Md. Sahil





Here normal DFS → A B E F H I → goal achieved is not optimal.
(depth limit = 5)

But in case of IDS → A
A B C
A B E F C G → goal achieved is optimal.

Since IDS increases the depth limit by 1 at each iteration, if a solution is found, it is guaranteed to be optimal (given all the paths have the same cost).

2.

b). If we consider the time complexities O-notation of the three, BFS, DFS & IDS has $O(b^d)$ time complexity. where

b → branching factor
d → depth.

2 (c) Iterative broadening search is preferred over iterative deepening in the case, when an average node of a tree has a large no. of child nodes. i.e. if branching factor is large. On the contrary if branching factor is low & depth factor is high, Iterative deepening is preferred.

Md. Sahid

But if we consider the exact average case ~~values~~ ~~for numbers~~:-

For number of nodes examined :-

~~BFS~~

$$\text{BFS} \rightarrow \frac{b^{d+1} + b^{d} + b - 3}{2(b-1)}$$

$$\text{DFS} \rightarrow \frac{b^{d+1} + bd + b - d - 2}{2(b-1)}$$

$$\text{IDS} \rightarrow \frac{b^{d+2} + b^{d+1} + b^{2} - 4bd - 5b + 3d + 2}{2(b-1)^{2}}$$

Comparing BFS & DFS

$$\frac{b^{d}}{2(b-1)} > \frac{bd}{2(b-1)} \Rightarrow \text{BFS} > \text{DFS}$$

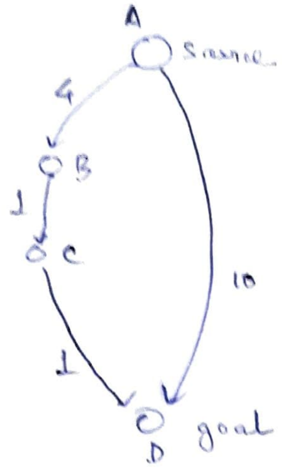
$\therefore$ DFS makes lesser number of node ~~com~~ explorations.

Comparing DFS & IDS

$$\text{IDS} = \frac{b+1}{b-1} \text{ BFS}.$$

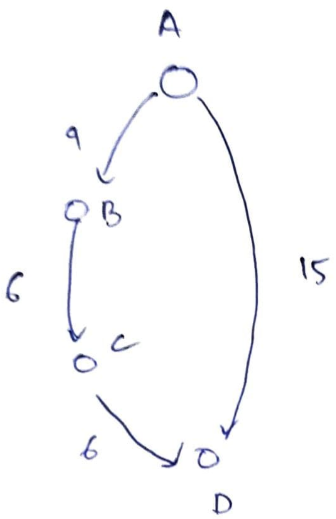
$\therefore$ IDS makes more node exploration than BFS before goal node is acheived.

2 d). Consider the case :-



we see here that ~~that~~ the optimal path is A → B → C → D

~~&~~ lets increase all the path costs by k=5



Now the cost of the path A → B → C → D is = 9+6+6
= 21

where as A → D = 15

∴ the new optimal path is A → D.

∴ The increasing the cost of each edge by a constant value can ~~co~~ chance the optimal solution.

3. a) True

3. b) General Graph Search Algorithm :-

1. ~~Create a~~ Initiate the search tree Tr, consisting solely of the start node ~~no~~ ~~on~~ n_0 ~~on~~ on ordered list called OPEN.

2. Create an list called CLOSED that is empty.

3. If OPEN is empty exit with failure.

4. Select the first node in OPEN put in on CLOSED, call this node n_{cur}.

5. If n_{cur} is goal node, exith with success with the solution by the tracing a path backward along the arcs in Tr from n_{cur} to n_0. ~~(Arcs are~~

6. Expand node ~~n~~ n_{cur}, generating a set ~~of~~ M of successors. ~~Insta~~ Install M as successor of n in the tree.

   Add ~~ncur~~ ~~to~~ M to OPEN.

7. Reorder the list OPEN according to some scheme or heuristic ~~not~~ merit.

8. ~~go~~ Goto ~~sep~~ step 3.

In step 7 ~~the~~ the reordering scheme or the heuristic scheme determines the ~~charact~~ characteristics of the graph.

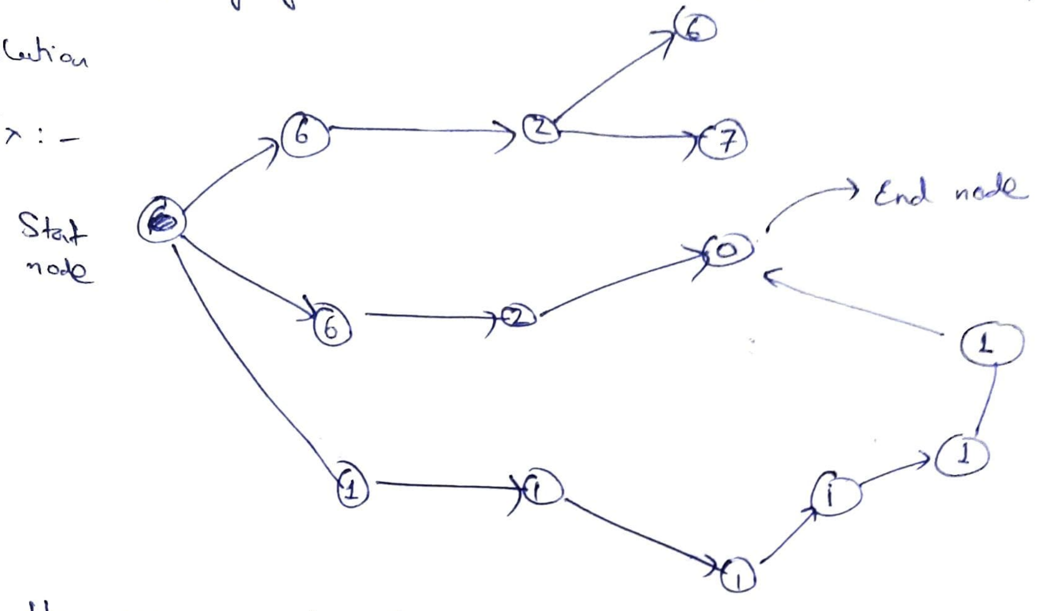
~~The~~ If the reordering scheme is LIFO ~~the~~ the ~~alg~~ algorithm becomes DFS; if the reordering scheme is FIFO the ~~scheme is~~ ~~algor~~ algorithm is BFS. If we use a priority queue with some sorting heuristic, we ~~g~~ will get informed search like Best-first search or A* search.

Md. Sabir

b). False.

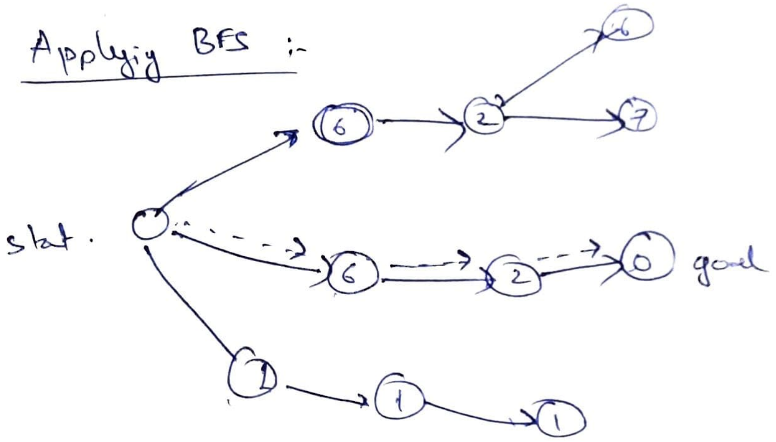
If time complexity of a search algo is higher, it may provide optimal solution

Ex:-



consider the ~~previ~~ graph depicting on state space graph.
The values in each node depicts the particular state's heuristics value.

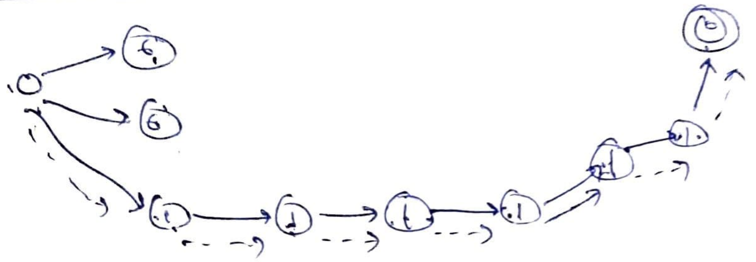
Applying BFS :-



No. of nodes travelled = 11

No. of nodes in the resultant path = 4.

Applying heuristics :-



No of nodes travelled = 10

No. of nodes in ~~the~~ resultant path = 8

3.
c). True

Heuristics search processes are better than blind search techniques.

Blind search :.

→ Brute force in nature → lengthy process
→ larger ~~mem~~ memory requirements as the process remebers ~~ex~~ all the ~~c~~ ~~e~~ nodes the unwanted nodes which are of no use at all.

Heuristic search :-

→ uses some kind of information of the problem to reduce search space.

→ less memory required.
→ Heuristic functions are used for searching.

→ Quicker process.

Ex → 15 - puzzle problem :-
If we apply blind search technique, total number of states to be explored $\simeq 10^{13}$. (~~polynomial~~ non-polynomial algoritm).
Applying heuristics such as manhattan distance or the number of misplaced tiles, the search space reduces drastically.

Hence heuristics approach are better.

5.

a) Normal hill climbing method suffers from problems like getting stuck at plateaus or ridges or getting stuck at local optimas. If the method is conducted only once the chance of facing these issues is considerable.

If the algorithm is ~~restarted~~ run multiple times the chance of hitting local optimas or plateau or ridges regions decreases considerably. Thus the chance of attaining a ~~solution~~ good solution becomes high. Hence random restart hill climbing is better.

b). Simulated annealing algorithm :-

```
begin
  t = max
  Select a random state sc and compute its functional value f(sc)
  while (t >= tmin)
  for i = 1 to n
    begin
    pick an adjacent state sn from sc at random and
    comute f(sn)
    if f(sc) >= f(sn) set sc = sn
    else set sc = sn with probability p = exp[-(f(sn) - f(sc))/t]
    end if
  decay t.
  end while
end.
```

8

Md. Sakib

If we look at the probability function ~~P~~ $P(t) = \exp\left(\frac{-(f(s_c) - f(s_n))}{t}\right)$

When ~~te~~ ~~when~~ ~~tept~~ temperate t is high, the value of the probability is high.

∴ The chances that ~~the~~ S_n is set as S_c is ~~do~~ also high, ~~which~~ ~~be~~.

∴ At high temperatures the algorithms behaves like a random search algorithm.

At lower temperatures the probability becomes low. thus only when $f(s_n) \geq f(s_n)$ s_n is set as S_c.

8
(c). In linear Normalization, we normalize the fitness values into a range of max to min ~~(e.g.~~. Say (1 to 100)

In Roulette wheel selection, however, we select the new generation probabilistically with greater preference to higher fitness values.

As such it is possible to lose diversity via this probability selection process.

In linear normalization, we can control the degree of variety in the ~~pro~~ population as the selection is on us.

Md. Sahil

7. a). $Q\{g(x,y), h(x,y)\}, Q\{g(z,u), h(w,u)\}, Q\{g(t,t), h(v,f(v))\}$

Disagreement set : $\{x, z, t\} \rightarrow z/x$.

$Q\{g(z,y), h(z,y)\}, Q\{g(z,u), h(w,u)\}, Q\{g(t,t), h(v,f(v))\}$

Disagreement set $\{x, t\} \longrightarrow z/t$

$Q\{g(z,y), h(z,y)\}, Q\{g(z,u), h(w,u)\}, Q(g(z,z), h(v,f(v)))\}$

Disagreement set $\{y, u, z\} \rightarrow y/u$

$Q\{g(z,y), h(z,y)\}, Q\{g(z,y), h(w,y)\}, Q(g(z,z), h(v,f(v)))\}$

Disagreement set $\{y, z\} \rightarrow y/z$

$Q\{g(y,y), h(y,y)\}, Q\{g(y,y), h(w,y)\}, Q\{g(y,y), h(v,f(v))\}$

Disagreement set $\{y, w, v\} \rightarrow y/w$

$Q\{g(y,y), h(y,y)\}, Q\{g(y,y), h(y,y)\}, Q\{g(y,y), h(v,f(v))\}$

Disagreement set $\{y, v\} \rightarrow y/v$

$Q\{g(y,y), h(y,y)\}, Q\{g(y,y), h(y,y)\}, Q\{g(y,y), h(y,F(y))\}$

Disagreement set - $\{y, F(y)\}$

Term $F(y)$ includes variable y, hence not unifiable.

Md. Sakib

7. (b) $\neg(\forall x)\{P(x) \rightarrow \{(\forall y)[P(y) \rightarrow P(f(x,y))] \wedge \neg(\forall y)[Q(x,y) \rightarrow P(y)]\}\}$

Eliminate $\rightarrow$

$\neg(\forall x)\{\neg P(x) \vee \{(\forall y)[\neg P(y) \vee P(F(x,y))] \wedge \neg(\forall y)[\neg Q(x,y) \vee P(y)]\}\}$

Reduce scope of $\neg$

$(\exists x)\{P(x) \wedge \{(\exists y)[P(y) \wedge \neg P(F(x,y))] \vee (\forall y)[\neg Q(x,y) \vee P(y)]\}$

Rename variables.

$(\exists x)\{P(x) \wedge \{(\exists x)[P(y) \wedge \neg P(F(x,y))] \vee (\forall z)[\neg Q(x,y) \vee P(z)]\}$

Remove existential quantifier, $g(x)$ is skolem function.
A is skolem constant.

$\{P(A) \wedge \{[P(g(x))) \wedge \neg P(F(A, g(x)))] \vee (\forall z)[\neg Q(A, z) \vee P(z)]\}\}$

Prenex normal form.

$(\forall z)\{P(A) \wedge \{[P(g(x)) \wedge \neg P(F(A, g(x)))] \vee [\neg Q(A, z) \vee P(z)]\}\}$

Remove left portion, and distributive law, CNF.

$P(A) \wedge (P(g(x)) \vee \neg Q(A, z) \vee P(x)) \wedge (\neg P(F(A, g(x))) \vee \neg Q(A, z) \vee P(z))$

clauses:-

1. $P(A)$
2. $P(g(x_2)) \vee \neg Q(A, z_2) \vee P(z_2)$
3. $\neg P(F(A, g(x_3))) \vee \neg Q(A, z_3) \vee P(z_3)$

Md. Sakil

7(c) $(\forall x)\ (P(x) \Rightarrow q(x))$

$(\forall x)\ (\neg P(x) \vee q(x))$

$\neg P(x) \vee q(x)$

$\therefore$ given clauses are :

1. $\neg P(x_1) \vee q(x_1)$
2. $\neg q(x_2) \vee r(x_2)$

To prove $(\forall x)\ (P(x) \Rightarrow r(x))$

Negate $\neg (\forall x)\ (P(x) \Rightarrow r(x))$

$(\exists x)\ \neg(\neg P(x) \vee r(x))$

$(\exists x)\ P(x) \wedge \neg r(x)$

$P(A) \wedge \neg r(A)$

clauses :

3. $P(A)$
4. $\neg r(A)$

$1 + 2 \xrightarrow{x_1/x_2} \neg P(x_1) \vee r(x_1)$ — (5)

$5 + 3 \xrightarrow{A/x_1} r(A)$ — (6)

$6 + 4 \longrightarrow$ NIL hence proved.

7.d) Resolution Refutation is ~~sound~~ sound & complete. Hence if a contradiction exists in the initial clause set we can definitely reach a ~~NE~~ NIL state via refutation.

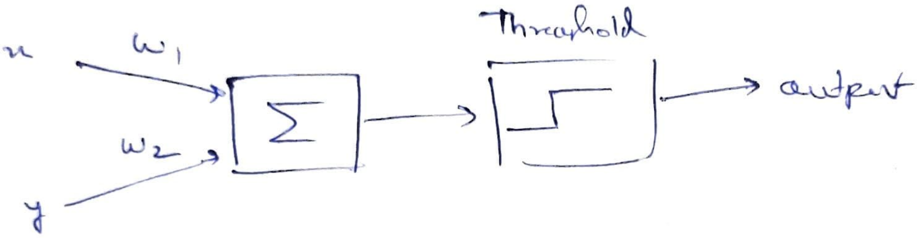
If no such contradiction exists, we will get all possible information and the algorithm will ultimately terminate as the set of resolvents is finite.

Md. Sahil

001710501029 (13)

9. Perception :- An artificial neurone is a system that tries to mimic a biological neurone. It takes input from its surroundings processes it ~~in~~ in its body. and spits the output to many other ~~neurons~~ neurons that may request it.

A single neurone connected by weights to a set of inputs ~~producing~~ producing a single output is known as a perception..



~~Fig. 9.2~~ - Two input perception

→ let x & y be two input and w_1, w_2 be the weights

→ If $w_1 x + w_2 y > \theta$, then the output is 1 else 0, where

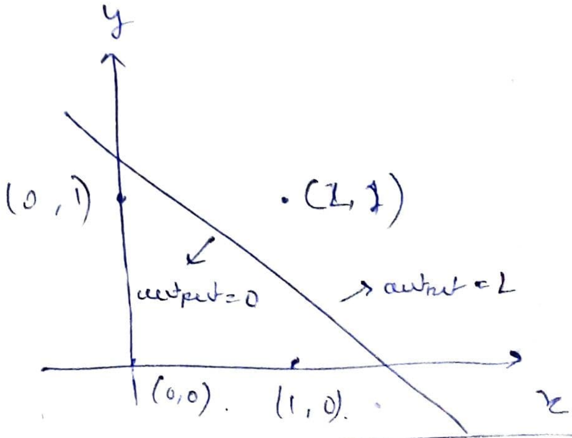
θ = threashold.

→ $w_1 x + w_2 y = \theta$ is the separating line.

For an AND function we have

x	y	output
0	0	0
0	1	0
1	0	0
1	1	1

In order to use a perception as an AND function, we must select θ such that the separating line comes as shown in the figure.



Md. Sahil

Considering $w_1 = 1$ & $w_2 = 1$

we have

$$w_1 x + w_2 y = \theta$$

$$x + y = \theta$$

For AND function, we must have

$1 + 1 > \theta$ (truth value)

$$\left.\begin{array}{l} 0 + 1 < \theta \\ 1 + 0 < \theta \\ 0 + 0 < \theta \end{array}\right\} \text{(false values)}$$

Hence selecting $1 < \theta < 2$ and $w_1 = 1$, $w_2 = 1$,

AND function is modelled, one possible solution $\theta = 1.5$

The seperating line is $\rightarrow$ $x + y = 1.5$

Limitations of a single layer perceptrom :-

A single layer perceptron can only be used in cases where the resultat classes are linearly separable.

Hence, functions like XOR are not implementable by a single layer ~~per~~ perceptron.