MINOR-II

Date 13/02/19

Saathi

PREDICATE LOGIC:

Limitations of Propositional Logic:-

(i) No provision to generalise the things:

To overcome such problems:- we use Predicate logic consisting Quantifiers.

→ Universal (∀) (for all) → satisfied by all the members of group.

→ Existential (∃):- at least one member of the group satisfy the property

Sol:- If 'x' is a CS Upitt graduate then x has passed CS441.

∃):- for at least one value of x

newcar(x) ⟶ registered(x).

1. All new cars must be registered:- property.

Object

⇒ If x is a new car, then x must be registered

(P) newcar ⟶ must be registered(Q) (ie) P → Q

∀x newcar(x) → registered(x)

(P complete subset of Q)

2. Some of the CS graduates graduate with honours.

(x) property

(and).

honours → x:(CS graduate)

∃x CS graduate(x) → graduate with honours(x)

for at least one value of x, there exist an x such that x is CS graduate and x has graduated with honours.

(not complete subset).

Page No.

Date ___ / ___ / _____

Saa

Sols to problem of Propositional logic:

a) Explicity models objects & their properties
b) Allows to make statements with variables & quantify them.

Basic buildering block of predicate logic

1. Constant → models a specific object
   eg:- '7', France, John, etc.
2. Variables → represents object of specific type.
   eg:- (universe of discourse can be people, students, nos.)
3. Predicate :- over one, two or many variable or constants
   - Represent properties or relations among objects

(Relationship)) Predicate (— , —)

eg: John is the brother of Mary.
Brother (John, Mary).

Ann is the wife of John.
Wife (Ann, John). — Constant

John is the husband of Ann
Husband (John, Ann).

Q:→ Predicate where both objects are variables.

Older (John, Peter)
→ It is a proposition

Older (x, y)
→ not a proposition, but can become if we substitute the values.

Date ___/___/_____

Predicates can have more arguments which represents the relations between objects

eg:- Let Q(x, y) denote $x+5>y$.
→ Not a proposition.

eg:- •) Student (Ann) ∧ Student (Jane)
Both Ann and Jane are student (It is a proposition)

•) Country (Sierra) ∨ River (Sierra)
Sierra is either a country or a river. (")

•)

⇒ Construct predicates:- 5
All CS graduate have to pass CS441

Q:- 1. Every daughter is younger than her mother.
2. Every student is younger than some teacher.
3. All cats are animals.
4. All cats and dogs are animals.
5. Humans are mammels.
6. All mammels are not humans.
7. Everyone loves someone:
8. All football players are tall. $\forall x$ football player(x) → tall(x)

$\forall x$

~~$\forall x$ football player(x) → tall(x)~~

1. $\forall x$ daughter(x) → younger than mother(x).
2.
3. $\forall x$ cat(x) → animal(x)
5. $\forall x$ humans(x) → Mammels(x)

Page No.

⇒ Some football players are tall

$\exists x$: football player $(x) \wedge$ tall (x).

⇒ John loves someone.

x : someone

$\exists(x)$: loves (John, x)

or $\exists(x)$: human $(x) \wedge$ loves (John, x)

⇒ everyone loves someone.

~~$\forall x$: human $(x) \rightarrow \exists(y)$: loves (x, y)~~

~~$\forall(x)$: human $(x) \rightarrow \exists(y)$: human $(y) \wedge$ loves (x, y)~~

or $\forall(x) \exists(y)$: loves (x, y)

or $\forall(x) \exists(y)$: human $(x) \wedge$ human $(y) \rightarrow$ loves (x, y)

⇒ Someone loves everyone

$\exists(x) \forall(y)$: loves (x, y)

$\exists(x) \forall(y)$: human $(x) \wedge$ human $(y) \rightarrow$ loves (x, y)

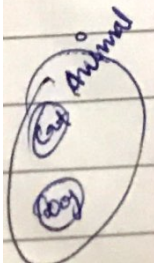
~~→ There is a person.~~

⇒ Some CS UP IIT graduates are also honours student

$\exists(x)$: CS IIT graduate $(x) \wedge$ honors student (x)

⇒ All cats and dogs are animals.

~~$\forall x \forall y$: cat $(x) \vee$ dog $(y) \rightarrow$ animal $(x$~~



$\forall(x)$: cat $(x) \vee$ dog $(x) \rightarrow$ Animal (x).

Saathi

Date

⇒ All cows, buffaloes, goats are herbivores

$\forall x$: cows(x) ∨ buffaloes(x) ∨ goats(x) ⟶ Herbivore(x).

⇒ Every student is younger than some teacher.

$\forall x \exists y$: student(x) ∧ teacher(y) ⟶ younger(x, y)

⇒ Every son of my father is my brother.

$\forall x$: son of my father(x) ⟶ brother(x, me)

Let p(x) denote the predicate that x is genious
Give the english interpretation of each of following :-

1) $\forall(x) : p(x)$

Everyone is genius

2) $\exists(x) : p(x)$

Someone is genuis or some people are genius.

3) $\neg\forall(x) : p(x)$

Not everyone is a genius.

4. $\neg\exists(x) : p(x)$

Someone is not genius.

or some people are not genius.

Q:- what is the difference in the scope of x & y. in $\forall(x)$, $\exists(x)$ & $\forall(y)$, $\exists(y)$ respectively ?

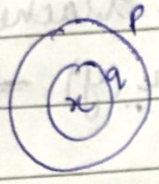
Date 25 / 02 / 19

Q:- What is a literal, clause & CNF. ?

↔ Literal is an atomic formula or a negated atomic formula.
(It is not possible to break it into sub-atomic formula)

p, p & q are literal

x is an element not literal



→ Clause :- Group of literals (V) or
It is either a literal or disjunction of literals.

eg :- P V Q.
~P V ~Q V R.

P V Q ∧ R ← not a clause.

→ CNF (Conjunctive Normal Form) :- It is composed of clauses in conjunctive form.

eg :→ (P V Q) ∧ (Q V ~S V T) ∧ (~R V T).

Q :→ convert the following to CNF.

Q :→ Convert into CNF.

1. P ≡ (R ∧ S)
2. P ↔ (R ∧ S)
3. [P → (R ∧ S)] ∧ [(R ∧ S) → P]
4. [~P V (R ∧ S)] ∧ [~(R ∧ S) V P]
5. (~P V R) ∧ (~P V S) ∧ [~R V ~S V P)

Remove Double Implication & equivalence with Implication.

Date ___/___/_____

Q:→ $P \equiv (R \wedge S)$

$P \leftrightarrow (R \wedge S)$

① Remove double implication or equivalence with implication

$[P \rightarrow (R \wedge S)] \wedge [(R \wedge S) \rightarrow P)$

② Apply Implication Equivalence

$[\sim P \vee (R \wedge S)] \wedge [\sim (R \wedge S) \vee P]$

③ Apply Distribution on '1st' and Apply Demorgan on 2nd. of above eqn.

$(\sim P \vee R) \wedge (\sim P \vee S) \wedge (\sim R \vee \sim S \vee P)$.

Q:→ Convert into CNF.

~~Q V (P ∧ (S → R))~~

~~(Q V P) ∧ (~~

$[(P \wedge (S \rightarrow R)) \vee Q] \wedge (R \rightarrow T)$.

$[(P \vee Q) \wedge [(S \rightarrow R) \vee Q]] \wedge (\sim R \vee T)$

$(P \vee Q) \wedge (\sim S \vee R \vee Q) \wedge (\sim R \vee T)$.

Horn Clause :- Clause that have almost one positive literal. This simplifies the logic buildings.

eg:→ Not human 'x' or mortal 'x'.

$\sim$human(x) or mortal (x)

⇒ $\forall(x)$ human (x) → mortal (x).

eg:→ $\forall y [\forall x (\text{taller}(y, x) \vee \text{wise}(x) \rightarrow \text{wise}(y)]$

$\sim$ taller $(y, x(y)) \vee$ wise $y)$

$\vee$ wise $(x(y)) \vee$ wise (y).

This is Yas Skolemization

Page No.

Date ___ / ___ / _____

Skolemization :→ is to remove the existential quantifier. The existential quantified variable is replaced by a skolem function. The dependent variable is treated as the function of the independent variables.

eg:- There exist an x such that eve is the mother of x.

$\exists x$ female $(x) \longrightarrow$ mother (x, eve)

John's Mother and Father are colleagues.

If Alisa and Richet are friends, then Alisa likes Richet.

John and Marry are colleagues.

3. Colleagues (John, Marry).

Mother $(x, John) \wedge$ Father $(y, John) \longrightarrow$ Colleges (x, y).

Colleges (Mother (John), Father (John))

Mother (John)

arity = 1 ⇒ No. of elements in a predicate.

Friends (Alisa, Richet) → Likes (Alisa, Richet)

Imp:- Universal Quantifier can be drop.
But existential Quantifier cannot be drop. It can be converted to universal Quantifier & then drop. or we use the process of skolemization.

Saathi

Date ___/___/___

a) everyone likes icecream

~~$\forall x$: likes(x, icecream)~~.

$\forall x$: Person(x) $\rightarrow$ Likes(x, icecream)

b) Someone likes icecream.

$\exists x$: Person(x) $\wedge$ likes (x, icecream)

c) everyone likes icecream (Existential form).

$\neg \exists x$: Person(x) $\wedge$ $\sim$ likes(x, icecream).

NOTE $\Leftrightarrow$ $\boxed{\forall x\, S \equiv \sim \exists x \sim S}$

d) No one likes football.

a $\forall x$: Person(x) $\rightarrow$ $\sim$ Likes(x, football).

$\neg \exists x$: Person(x) $\wedge$ Likes(x, football).

Q.) write a statement corresponding to

$\sim \forall x \sim S$

No one dislikes football.

Q All men are mortal — 1

Socrates is a man — 2

Prove that socrates is mortal

Conversion in predicate logic :- $\forall x$: Man(x) $\rightarrow$ Mortal(x) — 1

Man(Socrates) — 2

~~Apply Modus Ponens.~~

$\Rightarrow$ Drop the Quantifier. Man(x) $\rightarrow$ Mortal(x) — 1

Man(Socrates) — 2

$\Rightarrow$ Universal Instantiation

where (x) socrates is an instance of x

(1) and (2) become

Page No.

Date ___/___/___

Man (socrates) ⟶ Marital socrates —3

Man (socrates) —2

Applying Modus Ponen to ③ & ② , where it states

$P \to Q$ is True

P is True

$\therefore Q$ is True

⇒ Marital (Socrates) is True.

Q:- If it is not raining, Tom will go to mountains.

UNIFICATION

sometimes $\exists x$ is dependent on the universal quantifier

eg. Everyone like someone and someone is liked by everyone.

* Everyone like someone

$\forall x \exists y$: Person(x) ⟶ likes(x, y).

* Someone likes everyone

$\exists x \forall y$: Person(y) ⟶ likes(y, x).

eg * y is the mother of x.

mother(y, x)

$\forall x : \exists y$: mother(y, x)

Date ___ / ___ / ___

→ ~~Drop Universal Quantifier~~

Using Skolemization

$\forall x$: mother ($f(x)$, x)

Drop Universal Quantifier

mother ($f(x)$, x).

~~Restrictions on substitution~~

Substitution.

① The variables can be substituted by other variables.

② goo (x, a , moo (y))

p/x z/y.

⇒ goo (p, a , moo (z))

② Replacing variables by constants

Jack (x & bill (y))

goo (Jack a , moo (bill)

③ We can substitute

$f(y)/x$, $f(z)/y$.

Restriction on substitution ↔

(i) diff. constants cannot replace for same variable in a sentence.

(ii) Constants should not be replaced by other constants, there is no fun of it

(iii) whenever a variable is to be replaced, never skolemize it with a function having same variable because it becomes skolemize.

Q:→ $P \vee Q$, $P \to R$, $Q \to R$ ⊢ R

Proof R. is True

1. $P \vee Q$
2. $P \to R$
3. $Q \to R$
4. $\sim P \to Q$ (1, Implication)
5. $\sim P \to R$ (4,3, Syllogism)
6. $\sim R \to \sim P$ (2, Transportation)
7. $\sim R \to R$ (6,5, syllogism)
8. $R \vee R$ (7, Implication)
9. R (8, Tautology)

(1) Q:- 1. Every child loves candy
2. Anyone who loves some candy is not a nutrition fanatic
3. ~~Every~~ Anyone who eats any pumpkin is a nutrition fanatic
4. Anyone who buys any pumpkin either carves it or eat it
5. John buys a pumpkin
6. Life savours is a candy.
Prove 7. John is a child, then John carves some pumpkin.

OR

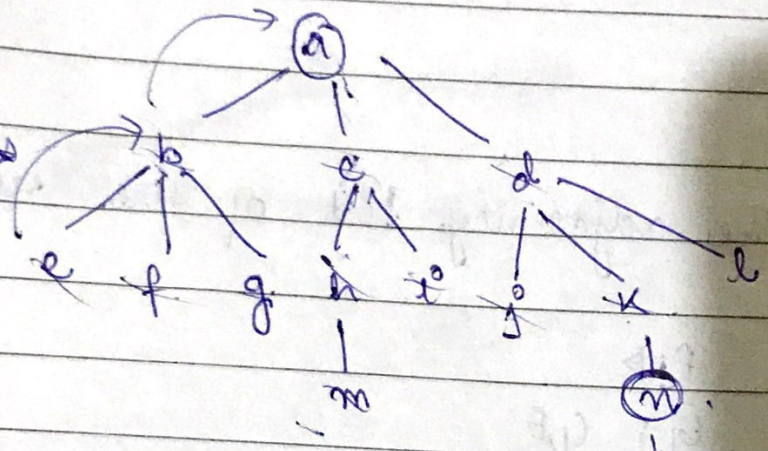
$\sim(A \wedge B)$

$\sim A \vee \sim B$

(2) Q: 1. Every child sees some witch
2. No witch has both a black cat and a pointed hat
3. Every witch is good or bad.
4. Every child who sees any good which gets a candy.
5. Every which that is bad has a black cat
Conclude If every which that is seen by any child that has a pointed hat, then every child gets a candy

(1) (A) 1. ~~(a)~~ $\forall x$: Child $(x) \longrightarrow$ loves $(x,$ candy$)$.
~~2. $\forall y$: loves $(y,$ candy$) \longrightarrow \sim$ nutrition fanatic~~
~~3. $\forall z$:~~
2. $\forall x$: child$(x) \wedge$ loves $(x,$ candy$) \rightarrow \sim$ Nutrition fanatic(x).
3. $\forall x$: child$(x) \wedge$ eats $(x,$ pumpkin$) \rightarrow$ Nutrition Fanatic(x)
4. $\forall x$: Buys $(x,$ pumpkin$) \rightarrow$ Carves $(x,$ pumpkin$) \vee$ eats $(x,$ pu
5. buys (John, pumpkin)
6. Candy (Life savours)
7. Child(John) $\rightarrow$ Carves(John, Pumpkin).

Q:→ Back Tracking →



Initial state - a

Goal state - n.

Two Methods :-

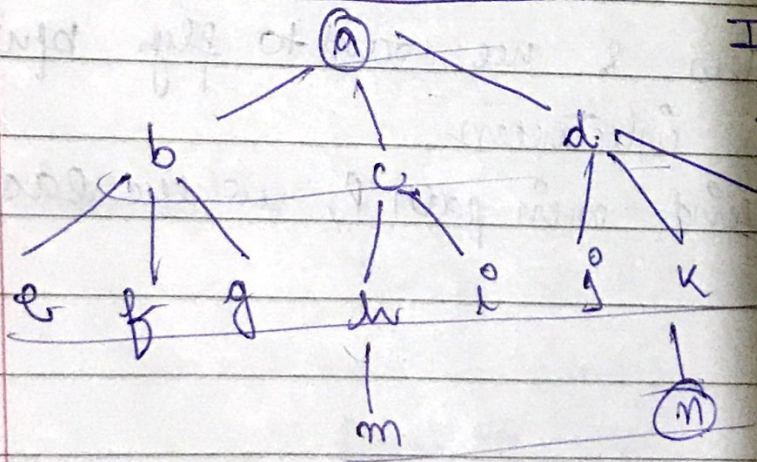
→ BFS (Breadth First

→ DFS (Depth First Se

used (Depth first search)

Reached.

Breadth search first.



Imp

Q:→ Differentiate b/w BFS and DFS.

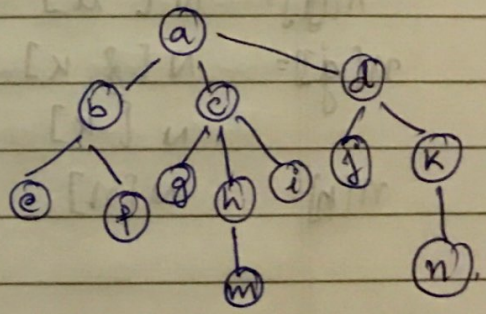
In terms of complexity, back tracking i

→ Exhaustive / Blind Search Algorithms ← about which we don't have apriori information.
we don't know in which direction we are moving.

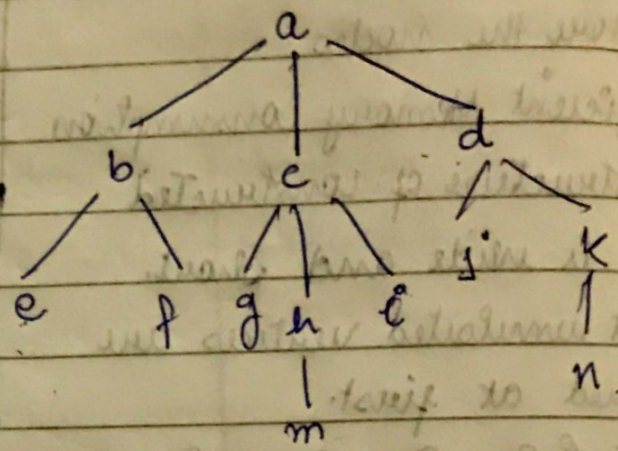
→ ~~Heuristic~~

(i) DFS N[n].

1. We will assign the 1ST node (root node) 'n' to the list of nodes [N].
2. If n is goal, stop & signal success.
3. Else, Generate the children of n &
4. If children of n = [ϕ]
   Stop and signal failure.
5. Else Add children of n to front of N and remove n from N.
6.



Given :-
Find the path generated in reaching to the Goal node (n) using DFS.



N[n].

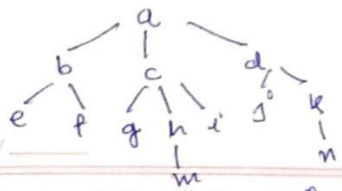
1. Assign 1st root node to the list of nodes [N].

n=a 1. N[a]
Generate children
N[a]=2. N[b, c, d]
3. Generate children of b.
n=[e] N[e, f, c, d]
Generate children of e
n=[f] = N[f, c, d]
n[c] N[c, d]
n[g] = N[g, h, i, d]
n[h] N[k, i, d]
n[m] = N[m, i, d]
n[i] = N[i, d]
n[d] = N[d]
n[j] = N[j, k]
N[k]
n[k] = N[n]

Path

a → b → e → f → c → g → h → m → i → d → j → k → n

* Children are given priority.



Q:- Reach the goal state [n] using BFS ?

1. Initialize N with the ~~root~~ node n
2. If N is empty, stop & signal ~~failure~~
3. Else assign the first node of N to n & remove n from the list N.
4. Examine n, If n is goal STOP & signal success.
5. Else generate children of n & ~~Add~~ to the N.
6. Go to Step 2.

N = [a]

n=a	N [b, c, d]
n=b	N [c, d]
~~n=c~~	N [c, d, e, f]
n=c	N [d, e, f]
~~n=d~~	N [d, e, f, g, h, i]
n=d	N [e, f, g, h, i, j, k]
n=e	N [f, g, h, i, j, k]
n=f	N [g, h, i, j, k]
n=g	N [h, i, j, k ~~n~~]
n=h	N [i, j, k, m]
n=i	N [j, k, m]
n=j	N [k, m]
n=k	N [m, n]
n=m	[n]
~~n=n~~	n=n [n]

Path = 13.

a → b → c → d → e → f → g → h → i → j → k → m → n

* The data we are storing for BFS is more.

Q: Difference in the Time complexity of BFS & DFS and how you arrive at the time complexity.

Time Complexity of both BFS & DFS will be O(V+E), where, ~~E~~
V = no. of vertices & E = no. of Edges.
This again depends on the data structure. If it is an adjacency matrix, it will be O(V^2). If we use an adjacency list, it will be O(V+E).

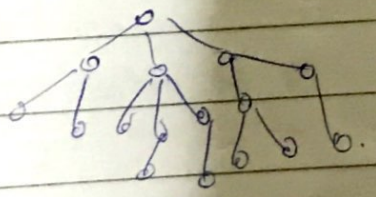
Assignment

Explain Game Theory and the application of AI to it.

9.

Bidirectional Search :-

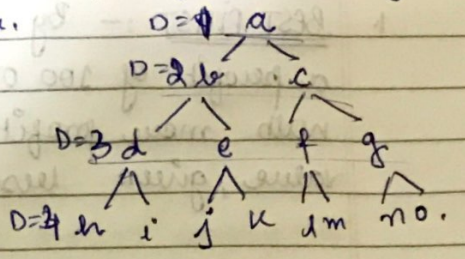
- A search Technique where search procedure is carried out by route node as well as goal node. (end node).
- Search proceeds forward from route node and backward from goal node
- when search meets at a point during the procedure, we say that the goal has been reach.
- By adopting this search, we have reduced the space.



Iterative Deepening :-

- In this method, the depth of tree is increased by a level x. (may be 1 or 2).
- After each iteration, we start with the level 0, by increasing the depth at each iteration, we try to find the goal using DFS.
- There is another technique for Iterative Broadening. This is used because memory req. is lesser than DFS.

Q:→ The given binary tree shall be searched for goal(k) using Iterative deepening. Write the sequence.



D=0 N[a]
D=1 N[a]
n=a N[b, c]
~~n=b N[b, c]~~
~~n=d N[d, e, c]~~
~~n=d N[h, i, e, c]~~
d=2 ~~N[e, e, c]~~ N[a]

Path :- (a, a, b, c, a, b, d, e, c, f, g, a, b, d, h, i, e, j, k).

let D=1

HURISTIC SEARCH

→ that has some amount of randomness. (stochastic)
→ It makes the search fast, gives result in reasonable time.

Stochasticity induces some drawbacks :-
→ they may settle for a less than best option.

Trade off :→
Optimality is reduced.

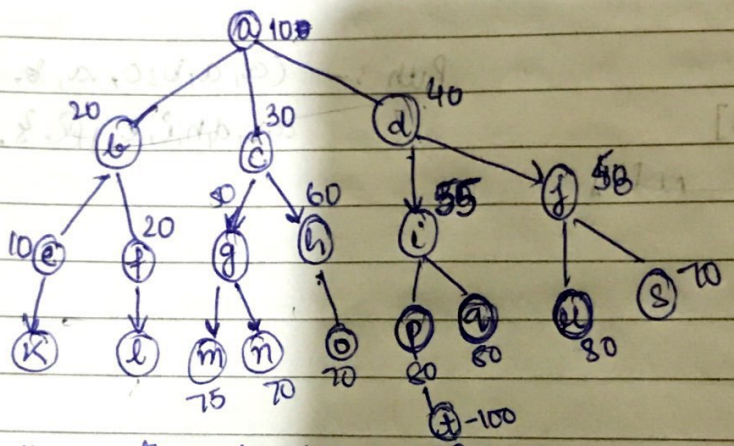
We have some apriori info. about nodes.
∴ They explore these nodes.

Imp Diff. bw Exhaustive & Heuristic Search.

Date ___/___/_____

HURISTIC

1. BEST FIRST :- By using this technique, find the node with a profit of 100 or more by assuming that the nodes with more profit values is nearer to the goal and the value, given besides the nodes are the profits.



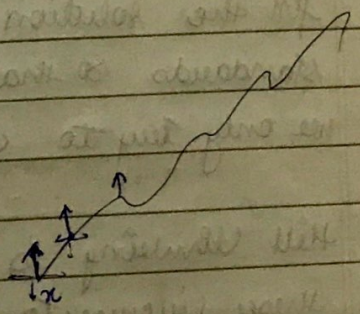
1. Assign the route node to the list [N].
2. If N is empty, there exist an report failure.
   else
3. Take the first node from N.
   assign to variable N & remove it from the list
4. examine ~~the goal~~ n, if n is goal
   Signal success & exist.
5. else Generate the children of n and add to N.
6. Re-arrange the children, based on their estimated goal and ~~Claims~~ ~~the~~ place the node with least distance in front.

Drawback :-

1) Sorting / Re-arranging is complex.

3. [b-20, c-30, d-40]
4. [d-40, c-30, b-20]
5. d-40 [i-55, j-58, c-30, b-20]
[j-58, i-55, c-30, b-20]
6. j-58 [k-80, s-70, i-55, c-30, b-20]
7. k-80 [s-70, i-55, c-30, b-20]
8. s-70 [i-55, c-30, b-20]
9. i-55 [p-80, q-80, c-30, b-20]
10. p-80 [t-100, q-80, c-30, b-20]
11. [t-100] ✓ Goal Reached.

2. Hill Climbing :-



Local optima →
In whatever region you are searching, the highest value of that region is the optimum point.

Drawback
→ This search is blind. It cannot go beyond its region.
*. It get stuck to local optima
*.

Global optima
Highest value of the whole search.

To get over this problem, we will randomly place the blind person and will try to reach the optimum point. Then several optimum point will be compared, Best will be accepted.

→ Has Random Initialization Technique

⇒ Another Method Hill

Sometimes the aim of the search is to find the maximum or the min. value of the given function. However, it may not always be possible to find max or min value of function.

For eg:- It may not be possible to schedule the airlines in such a fashion that may not cause congestion at any airport, under any circumstances.

Under such a situation, the goal of the search may be to fix the solution on standard considering the present standards so that instead of minimising or maximising, we only try to optimise the solution.

Hill climbing is one of the most eff. search techniques under these circumstances.

Algorithm:-

1.) Assign the root node to n.
2. Generate the children of n.
3. If n is greater than all its children, signal 'n' as soln. & stop.
3. Else, assign the greatest value child to n.
4. Go to step 2.

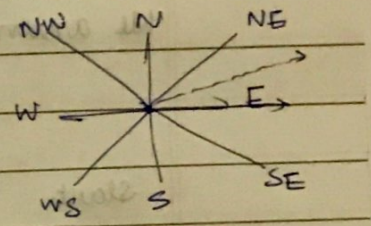
This algo aims to find the min or least value or the max. value depending on the initial or start node to it.

Drawback:-

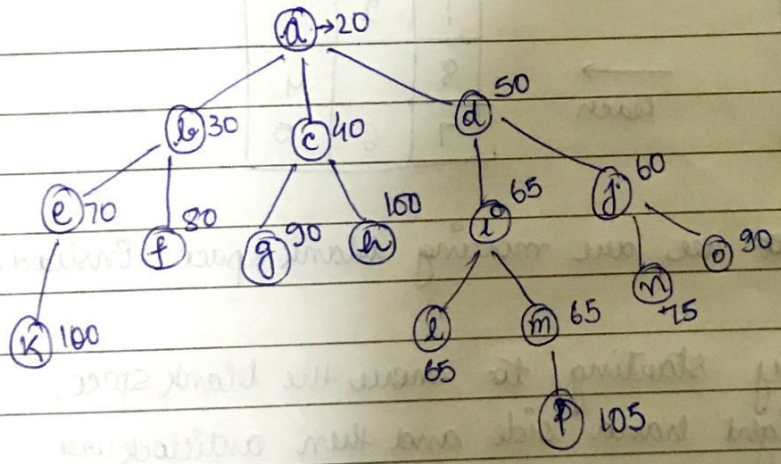
1. If we reach foot hill, we have to stop
   (ii) confined to local Optima.

2. In case we are getting same value in all the directions, the search technique gets confused and it starts ~~toggles~~ ~~and~~ or randomly choose the value. The condition is ~~kya~~ plateau.
   This problem can be overcomed.

3. Ridged :- A ridge is a steep hill going in one direction which may not figure in the search technique.



eg:→



Q:- find a node with value of 100 or more using Hill climbing

n = a-20

n = [b-30, c-40, d-50]

n[d] = [b-30, c-40, d-65, j-60]

n[i] = [b-30, c-40, l-65, m-65, j-60]] Plateau, search becomes confused.

Random n[m-65] = [b-30, c-40, l-65, p-105, j-60]

Goal is achieved

Date ___/___/___

A* :- This algo is one of the most searched tech in game playing.

Here the Evaluation function $f(n)$ is equal to the sum of estimated distance and actual distance covered.

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

(Evaluation function) — $h(n)$: estimated distance of n (current state from goal); $g(n)$: actual distance covered from root node till n.

We assume that $h(n) \leq h'(n)$ which is actual distance till the goal.

Start.

2	8	3
1	6	4
7		5

→ Reach

1	2	3
8		4
7	6	5

* Assume that we are moving blank space instead of tiles.
* We give priority by starting to move the blank space, towards the right hand side and then anticlockwise
* Estimated distance from the goal is equal to the no. of tiles in their place while comparing to goal.

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

2	8	3
1	6	4
7		5

$f(n) = 5.$

→

2	8	3
1		4
7	6	5

$f(n) = 3 + 1 = 4.$

$h(n) = 5$, $g(n) = 0$

→

2		3
1	8	4
7	6	5

$f(n) = 4 + 2 = 6$

↓

	2	3
1	8	4
7	6	5

$f(n) = 3 + 3 = 6$

←

1	2	3
	8	4
7	6	5

$f(n) = 2 + 4 = 6$

←

1	2	3
8		4
7	6	5

$f(n) = 0 + 5 = 5$

we have to generate possible child calculate $f(n)$ and choose the least $f(n)$.