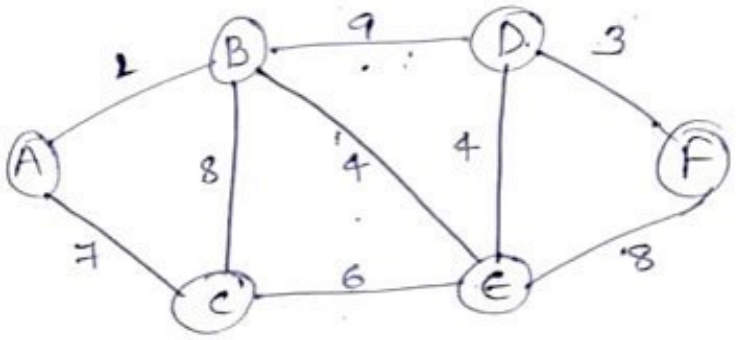
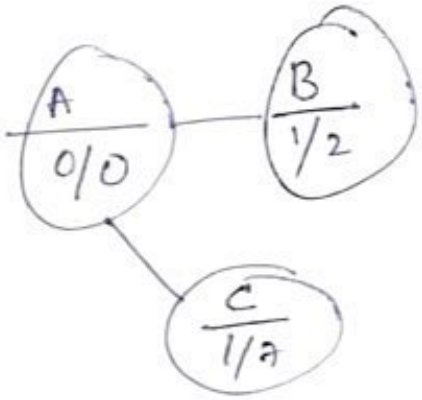
Sai Parthish Mandumula
1002022847

Task 1



Breadth - first search

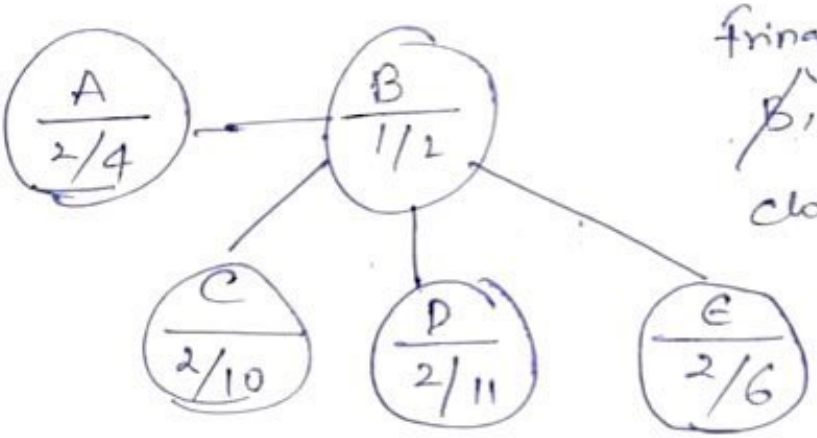
Expand node (A) =>



fringe :- ~~A~~, B, C

closed set - (A)

=> Next Expand B as the element is present in fringe



fringe :-

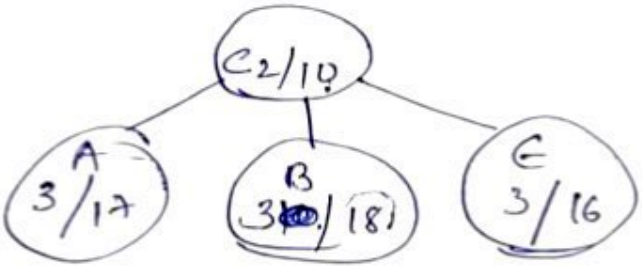
~~B~~, C, A, C, E, D

closed set A, B,

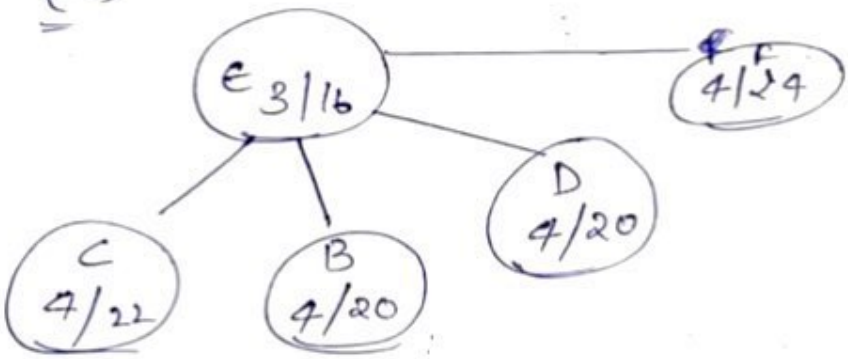
C →

fringe :-
~~C~~, ~~A~~, ~~C~~, E, D, A, B, E

closed set - A, B, C



E →

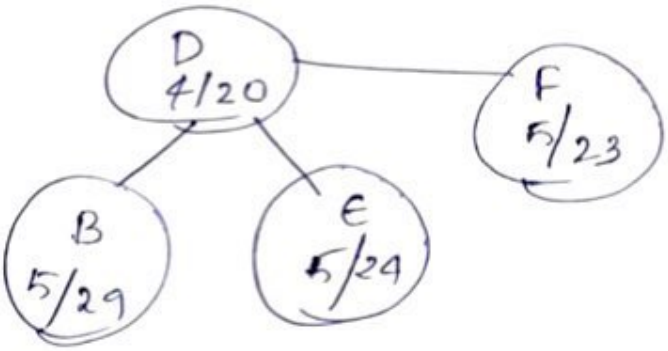


fringe

~~C~~, D, A, B, E, C, B, D, f.

closet set

A, B, C, E.

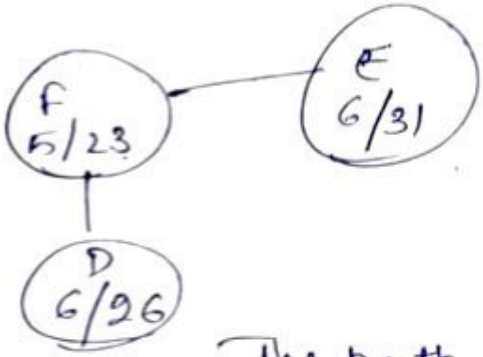


fringe

~~B~~, ~~A~~, ~~B~~, ~~E~~, ~~C~~, ~~B~~, f, B, E f

closet ~~set~~

A, B, C, E, D,



fringe ~~E~~, B, E, f, E, D

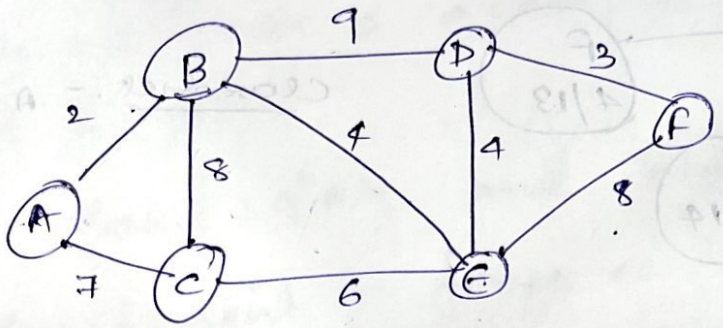
closet ⇒ A, B, C, E, D, f.

The path from start A to end f is

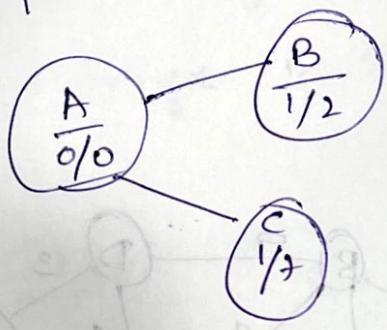
A → B → E → E → D → f.

Sai Parthish Mandumula
1002022847

depth-first-search [LIFO]



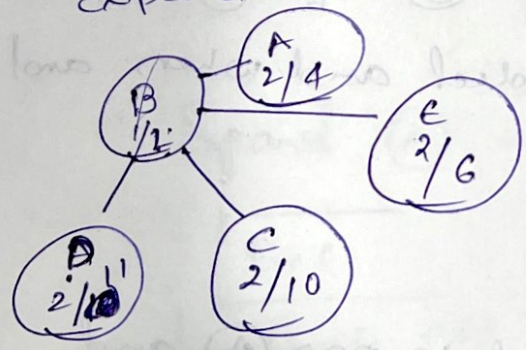
Expand node :A



Fringe :- B C ~~A~~

closed set :- ~~A~~

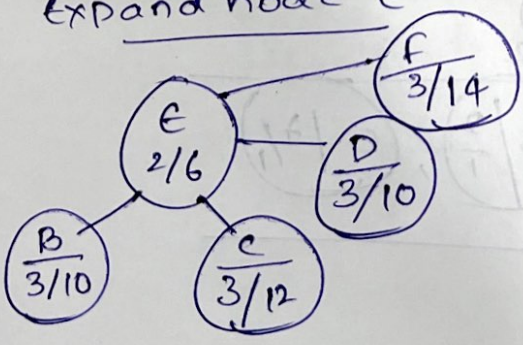
Expand node B :-



Fringe :- E A C ~~B~~ ~~B~~ C ~~A~~

closedset :- A, B

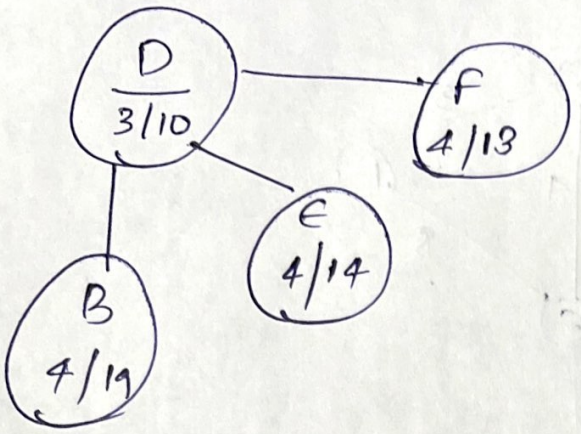
Expand node E



fringe ~~D~~, f, B, C, ~~E~~ A C D ~~B~~ C ~~A~~

closet A, B, E, D.

Expand node D



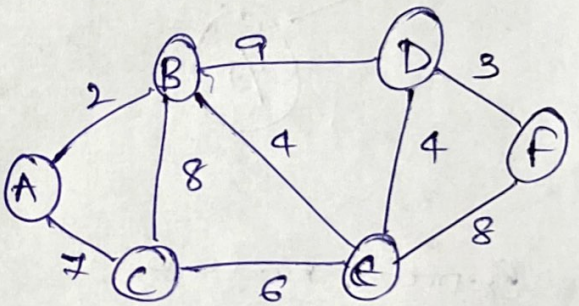
Fringe:-
F, E, B, ~~D~~, F, B, ~~C~~ E A C D ~~B~~ C ~~A~~

closedlist :- A, B, E, D, F.

The dFs Path is A B E D F

- Iterative deepening Search.

Add (A) to fringe

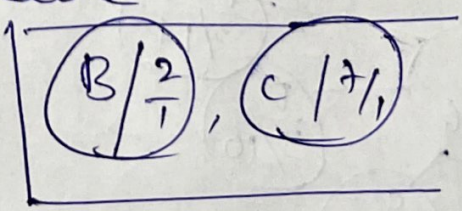


On the graph pop (A) and nothing is added and when and the fringe is Empty at 0.

Dtl:

Add (A)

As we visited A we need to pop (A) and then write the connecting nodes to A so now in fringe we have



Now we Expand C.

So while we take C and we add nothing to fringe

closed : {A, B}

Now B also present in fringe as it is already visited by A and closed : {A, B}.

fringe is empty and search failed.

It2 :-

for the iterative deepening search

Add (A) to fringe

(A)

Expand (A) and Successors to fringe and

B, C

and B, C are connected to A and Expand

(C) So on Expand and then add its Successors

B, A, B, E

closed set : {A, C}

Expand E. Expand B Expand A and then is added to fringe.

Expand B and now we need to add its successors.

A, C, D, E

Expand E, Expand D, Expand C Expand A

and add nothing to fringe.

on the 1 to 3:-

Add (A) and add the successors of A to fringe (B), (C)

C:s n = {A}

Expand (C) and add its successors to fringe

B, A, B, E

where we add {A, C}

we have the nodes to C are B, A, E

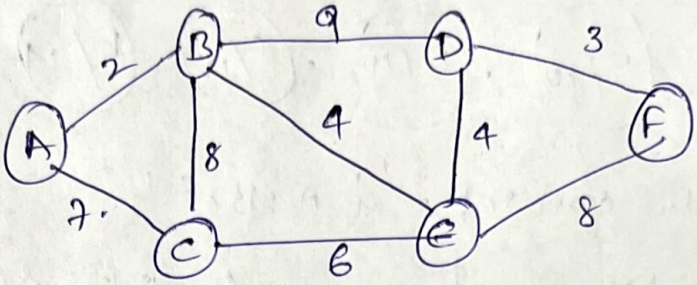
B, A, B, C, D, B, f

C:s :→ {A, C, E}

we the final f is the goal state

A — C — E — F

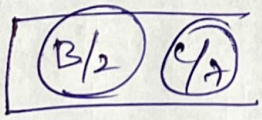
④ uniform cost Search.



For the given graph of uniform cost Search, the fringe is sorted based on the cumulative cost we will start A as it is the start point

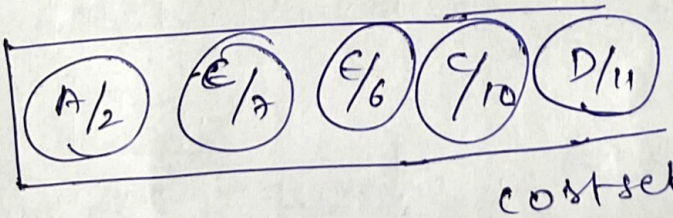
Add (A) to fringe and we just connect the joints or linking to nodes that are B and C.

On Expand A



and now the costset is {A}

Now we Expand B and add Successors so we have B and C will go with B. to B



costset is {A, B}.

as the A is already in fringe we need to pop A. costset {A, B}

Now Expand (E) and add the Successors.

(E/7) . (C/10) . (D/11) . (C/12) (F/14) (D/10)

Now the closet set = { A, B, E }.

pop (E/7) and generate its successors.

C/10 . D/11 , C/12 . D/10 , F/14 . A/14 , B/15, C/13

and the closet = { A, B, E, C }.

and we need to pop the elements in fringe

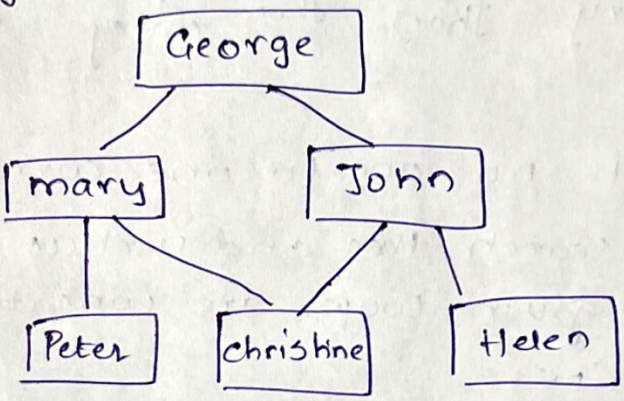
pop (C/10) (D/11) (C/12) (E/13) (F/14) and we know the final goal is F.

the order is A → B → E → D → F.

Sai Parthish Mandumula
1002022840

Task-2

Given a social network graph (SNG) is a graph where each vertex is a person and each edge represents an acquaintance and the given graph is

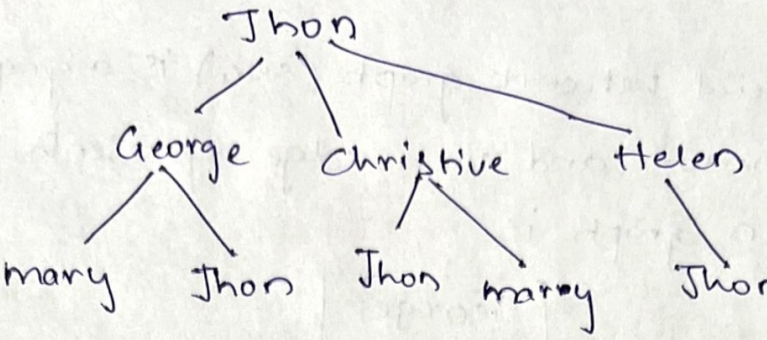


(i) from the among general tree search tree, using breadth first search, depth-first search, iterative deepening search. The degree of separation between connected two people can found by Breadth-first search, as this search results in optimal solution path towards the goal. We cannot guarntee the finding correct number of degree of separation of any two people, By using depth-first search. The DFS will loop infinetly to searching goal node so, we cannot use DFS.

By using Iterative deepening search, we can find the correct number of degree of separation, because the iterative will search for the shortest path which will lead to optimal solution.

If we uniform cost search, we can find the number of separation because of its optimal solution by searching shortest node.

(ii) let us take Jhon as starting point,

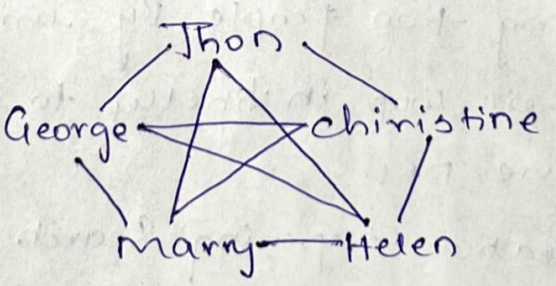


(iii) No, there is no one-to-one correspondence between nodes, in the search tree and vertices in the SNG. The reason is the given loops are concating i.e, Jhon, mary, christine.

(iv) An SNG containing Exactly 5 people, where at least two people have 4-degree of Separation between them is

Jhon — George — christine — mary — Helen.

(v) An SNG containing 5 people, where all people have 1 degree of separation between them is



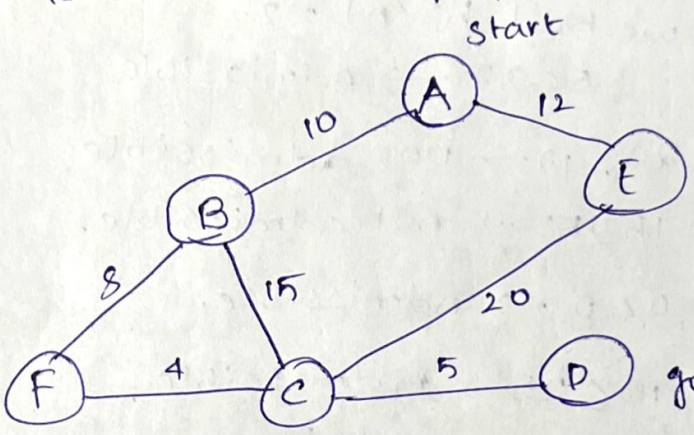
(vi) In an implementation of breadth-first tree search for finding degree of Separation, Suppose that Every node in the search tree takes of 1KB of memory. at each node and there are millions and millions people. this leads to much memory constraint.

Sai Parthish
1002022847

BFS will avoid nodes that visited perviously, then it can be retrived in closed set to achieve one to one correspondance goal node.

Task-3

A search problem showing states and coasts of moving from one state to another. the graph is



By modifying heuristic we can admissible for given fig

$h^*(A)$

$A \rightarrow B \rightarrow F \rightarrow C \rightarrow D$

$10+8+4+5 = 27$

$h^*(E)$

$E \rightarrow C \rightarrow D$

$20+5 \Rightarrow 25$

$h^*(B)$

$B \rightarrow F \rightarrow C \rightarrow D$

$8+4+5 \Rightarrow 17$

$h^*(F)$

$F \rightarrow C \rightarrow D$

$4+5 \Rightarrow 9$

$h^*(C)$

$C \rightarrow D$

$5 \Rightarrow 5$

$h(n)$ is admissible if

$h(n) \leq h^*(n)$

$h^*(D)$ D

$0 \Rightarrow 0$

Heuristic 1:

$h(A) = 5$

$h(B) = 20$

$h(C) = 15$

$h(D) = 0$

$h(E) = 10$

$h(F) = 0$

$h(A) = 5$, we had $h^*(A) = 27$

$5 \leq 27 \rightarrow$ Admissible

$h(B) = 20$; $20 \geq 17 \rightarrow$ not admissible.

$h(C) = 15$; $15 \geq 15 \rightarrow$ not admissible.

$h(D) = 0$; $0 \leq 0 \rightarrow$ Admissible.

$h(E) = 10$; $10 \leq 25 \rightarrow$ Admissible,

$h(F) = 0$; $0 \leq 9 \rightarrow$ Admissible.

In heuristic 1. $h(C)$ is not admissible because of over cost estimation.

Heuristic 2:-

$h(A) = 45$

$h(B) = 45$

$h(C) = 45$

$h(D) = 45$

$h(E) = 45$

$h(F) = 45$.

Sai Parthish . m
1002022843.

$h(A) = 45 \quad \geq 27$

$h(B) = 45 \quad \geq 17$

$h(C) = 45 \quad \geq 5$

$h(D) = 45 \quad \geq 0$

$h(E) = 45 \quad \geq 25$

$h(F) = 45 \quad \geq 9$

not admissible because of over cost Estimate.

Heuristic-3

$h(A) = 10$

$h(B) = 15$

$h(C) = 0$

$h(D) = 0$

$h(E) = 25$

$h(F) = 5.$

solution

$h(A) = 10 \leq 27$

$h(B) = 15 \leq 17$

$h(C) = 0 \leq 5$

$h(D) = 0 \leq 0$

$h(E) = 25 \leq 25$

$h(F) = 5 \leq 9.$

Heuristic-3 is admissible as it satisfy

$h(n) \leq h^*(n)$

Heuristic-4 :-

$h(A) = 0$

$h(B) = 0$

$h(C) = 0$

$h(D) = 0$

$h(E) = 0$

$h(F) = 0$

Sol

$h(A) = 0 \leq 27$

$h(B) = 0 \leq 17$

$h(C) = 0 \leq 5$

$h(D) = 0 \leq 0$

$h(E) = 0 \leq 25$

$h(F) = 0 \leq 9$

This is admissible as it satisfy $h(n) \leq h^*(n)$.

Task- 4

Consider a search space, according to given conditions the o

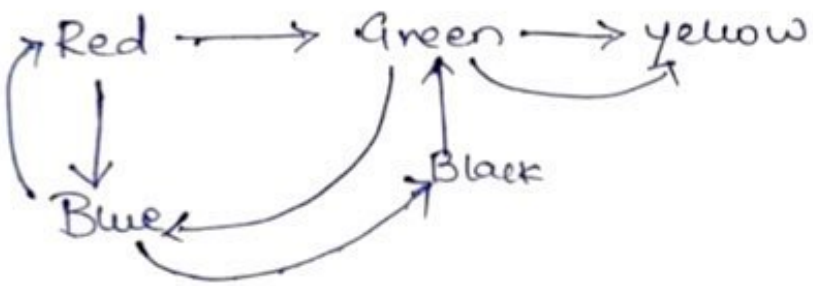
Red states can only have green or blue children

Blue states can only have red or black children

Green states can only have blue or yellow children

Yellow states can only have yellow or red children

Black states can only have green or black children.



Heuristic will be as

h(Red) = 2

h(Blue) = 1

h(Green) = 2

h(yellow) = 3

h(Black) = 0.

Sai Parthiph. M
1002022893.

Task. 5

According to given maps where all towns are on a grid. and on each town T has coordinates where we have Ti and Tj and it follows as

and now for A* we can search f(n) and it will be addition of g(n) and h(n) where

f(n) = g(n) + h(n) and now we have figure 4 and figure 5 so, let us take figure 4.

when source and distination that are indigonal and A* will Expand more number of nodes wher than greedy on Same cases.

Source → distination So,
(3,3)

greedy ⇒ (1,1), (1,2), (2,2), (3,2), (3,3) where as we take A* → (1,1), (1,2), (2,1) (2,2). (2,3) and (3,2), (3,3). where in Some as A*

now the greedy always performs more or Equal as A* Search.

Sai Parthisth. M
1002022847.

Figure. 5

From the given map cities and on partially connected grid. and we can tell that greedy is abnormal. well and the condition it gone perform sometimes and same as. If source is (3,1) and destination as (3,3). In this case greedy will take much longer time and A* because of it gona perform loop and In other case, let take (3,6) and (5,8) as source & destination where the things will expand more number of nodes and than greedy. Here will expand more number of nodes than greedy. so, greedy is better than A* in this case. In order case Consider (0,4) as source & (0,7) as destinatio In this case both will perform same. so, based on source and destination it varies between A* and greedy search.