Term Work 6

Problem statement :- Design an algorithm to implement DFS and develop Prolog problem for the same

Theory :-

Depth first search or DFS algorithm is a recurssion algorithm that uses the backtracking principle. It contains conducting exhastive search of all nodes by moving forward if possible & backtracking if necessary to exist the next node, pop the top node from the stack & push all of its nearby node into a stack, Topological Sorting schedule problem group cycle direction of solving puzzles with just one solution. such as a maze or a shadow puzzle.

A standard DFS implementation puts even vertex of the graph into one of two categories

1] visited
2. Non-visited.

The purpose of the algorithm is to move each vertex as visited while avoiding cycle

The DFS algorithm is as follows:-

1] Start by putting any one of the graph's varies on top of a stack
2] Take the top item of the Stak out it the visited list
3] Create a list of that vertex's adjacent nodes. Add the which aren't in the visited list to the top of stack

Program:

```
child(s,c)
Child(S,A)
Child(3,B)
Child(B,A)
Child(A,E)
Child(C,D)
Child(D,E)
Child(E,G)

Path(A,G,[A|Z]):-
  Child node(A,G,Z),
  Child node(A,G,[G]):-

  Child(A G)
  Childnode(A,G,(X|L]):-
  Child(A,X).
  Childnode(X,G,L).
```

Output :-

? : path(A, G, L).
L = [A, C, G].

? : path(a, e, h).
h = [a, b, e]

Tracing :-

path(a, G, [a|z]) :- childnode(a, g, z)
childnode(a, g, [g]) :- child(a, g).
childnode[a, g, [x|L]) :- child(a, x), childnode(a, g, L)
childnode(c, g, [g]) :- child[c g].
L = [a, c, g].

Conclusion :-

In this termwork we learnt the implemention of DFS algorithm in prolog and executed the graph with resulting path to goal node.