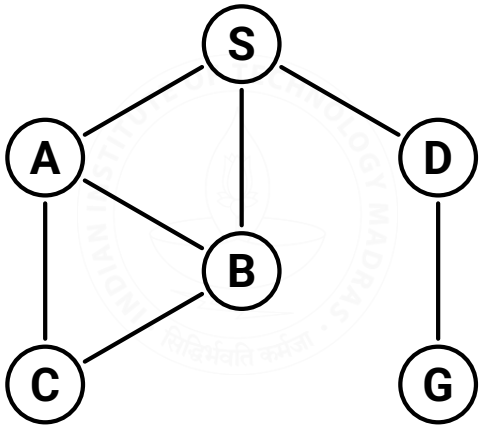
Lecture Example 2
Depth First Search: Cases 1, 2 and 3

Prepared by S. Baskaran

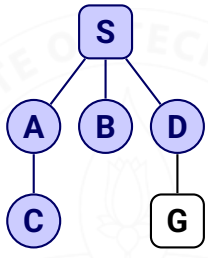
State Space



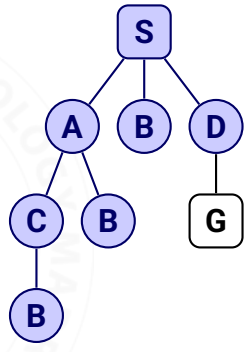
X	MoveGen(X)
S	[A,B,D]
A	[C,B,S]
B	[S,A,C]
C	[B,A]
D	[S,G]
G	[D]

Depth First Search – Search Trees

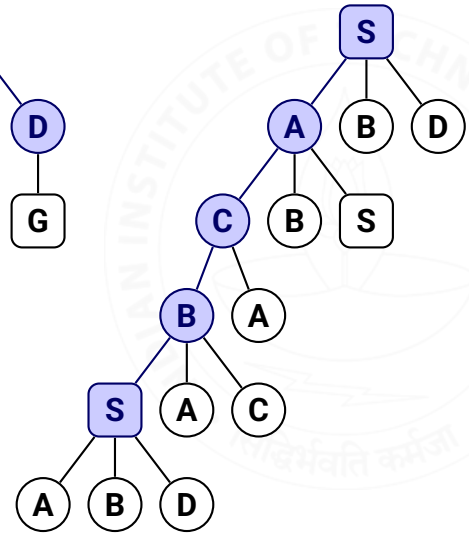
DFS Case 1



DFS Case 2



DFS Case 3



Depth First Search – Case 1

The solution for Case 1 is based on the DFS algorithm given in Week 2 Notes. For the Cases 2 and 3, take DFS Case 1 and make suitable modifications according to what is discussed in the lecture. Try it yourself.

DFS Case 1 removes both OPEN nodes and CLOSED nodes from the output of MoveGen.

```
OPEN and CLOSED carry pairs: (NODE,PARENT)

OPEN        (S,null):[]
CLOSED      []

1.
NODE        S
close       (S,null)
moveGen     A:B:D:[]
newNodes    A:B:D:[]
newPairs    (A,S):(B,S):(D,S):[]

OPEN        (A,S):(B,S):(D,S):[]
CLOSED      (S,null):[]

2.
NODE        A
close       (A,S)
moveGen     C:B:S:[]
newNodes    C:[]
newPairs    (C,A):[]

OPEN        (C,A):(B,S):(D,S):[]
CLOSED      (A,S):(S,null):[]

3.
NODE        C
close       (C,A)
moveGen     B:A:[]
newNodes    []
newPairs    []

OPEN        (B,S):(D,S):[]
CLOSED      (C,A):(A,S):(S,null):[]

4.
NODE        B
close       (B,S)
moveGen     S:A:C:[]
newNodes    []
newPairs    []

OPEN        (D,S):[]
CLOSED      (B,S):(C,A):(A,S):(S,null):[]

5.
NODE        D
close       (D,S)
moveGen     S:G:[]
newNodes    G:[]
newPairs    (G,D):[]

OPEN        (G,D):[]
CLOSED      (D,S):(B,S):(C,A):(A,S):(S,null):[]

6.
NODE        G
GOAL        G

OPEN        (G,D):[]
CLOSED      (D,S):(B,S):(C,A):(A,S):(S,null):[]

PATH        S:D:G:[]
```

Depth First Search – Case 2

The solution for Case 1 is based on the DFS algorithm given in Week 2 Notes. For the Cases 2 and 3, take DFS Case 1 and make suitable modifications according to what is discussed in the lecture. Try it yourself.

DFS Case 2 removes only the CLOSED nodes from the output of MoveGen.

```
OPEN and CLOSED carry pairs: (NODE,PARENT)

OPEN        (S,null):[]
CLOSED      []

1.
NODE        S
close       (S,null)
moveGen     A:B:D:[]
newNodes    A:B:D:[]
newPairs    (A,S):(B,S):(D,S):[]

OPEN        (A,S):(B,S):(D,S):[]
CLOSED      (S,null):[]

2.
NODE        A
close       (A,S)
moveGen     C:B:S:[]
newNodes    C:B:[]
newPairs    (C,A):(B,A):[]

OPEN        (C,A):(B,A):(B,S):(D,S):[]
CLOSED      (A,S):(S,null):[]

3.
NODE        C
close       (C,A)
moveGen     B:A:[]
newNodes    B:[]
newPairs    (B,C):[]

OPEN        (B,C):(B,A):(B,S):(D,S):[]
CLOSED      (C,A):(A,S):(S,null):[]

4.
NODE        B
close       (B,C)
moveGen     S:A:C:[]
newNodes    []
newPairs    []

OPEN        (B,A):(B,S):(D,S):[]
CLOSED      (B,C):(C,A):(A,S):(S,null):[]

5.
NODE        B
close       (B,A)
moveGen     S:A:C:[]
newNodes    []
newPairs    []

OPEN        (B,S):(D,S):[]
CLOSED      (B,A):(B,C):(C,A):(A,S):(S,null):[]

6.
NODE        B
close       (B,S)
moveGen     S:A:C:[]
newNodes    []
newPairs    []

OPEN        (D,S):[]
CLOSED      (B,S):(B,A):(B,C):(C,A):(A,S):
            (S,null):[]

7.
NODE        D
close       (D,S)
moveGen     S:G:[]
newNodes    G:[]
newPairs    (G,D):[]

OPEN        (G,D):[]
CLOSED      (D,S):(B,S):(B,A):(B,C):(C,A):
            (A,S):(S,null):[]

8.
NODE        G
GOAL        G

OPEN        (G,D):[]
CLOSED      (D,S):(B,S):(B,A):(B,C):(C,A):
            (A,S):(S,null):[]

PATH        S:D:G:[]
```

Depth First Search – Case 3

The solution for Case 1 is based on the DFS algorithm given in Week 2 Notes. For the Cases 2 and 3, take DFS Case 1 and make suitable modifications according to what is discussed in the lecture. Try it yourself.

DFS Case 3 does not remove any nodes from the output of MoveGen.

```
OPEN and CLOSED carry pairs: (NODE,PARENT)

OPEN        (S,null):[]
CLOSED      []

1.
NODE        S
close       (S,null)
moveGen     A:B:D:[]
newNodes    A:B:D:[]
newPairs    (A,S):(B,S):(D,S):[]

OPEN        (A,S):(B,S):(D,S):[]
CLOSED      (S,null):[]

2.
NODE        A
close       (A,S)
moveGen     C:B:S:[]
newNodes    C:B:S:[]
newPairs    (C,A):(B,A):(S,A):[]

OPEN        (C,A):(B,A):(S,A):(B,S):(D,S):[]
CLOSED      (A,S):(S,null):[]

3.
NODE        C
close       (C,A)
moveGen     B:A:[]
newNodes    B:A:[]
newPairs    (B,C):(A,C):[]

OPEN        (B,C):(A,C):(B,A):(S,A):(B,S):
            (D,S):[]

CLOSED      (C,A):(A,S):(S,null):[]

4.
NODE        B
close       (B,C)
moveGen     S:A:C:[]
newNodes    S:A:C:[]
newPairs    (S,B):(A,B):(C,B):[]

OPEN        (S,B):(A,B):(C,B):(A,C):(B,A):
            (S,A):(B,S):(D,S):[]

CLOSED      (B,C):(C,A):(A,S):(S,null):[]

5.
NODE        S
close       (S,B)
moveGen     A:B:D:[]
newNodes    A:B:D:[]
newPairs    (A,S):(B,S):(D,S):[]

OPEN        (A,S):(B,S):(D,S):(A,B):(C,B):
            (A,C):(B,A):(S,A):(B,S):(D,S):[]

CLOSED      (S,B):(B,C):(C,A):(A,S):(S,null):[]

... and so on ...
```