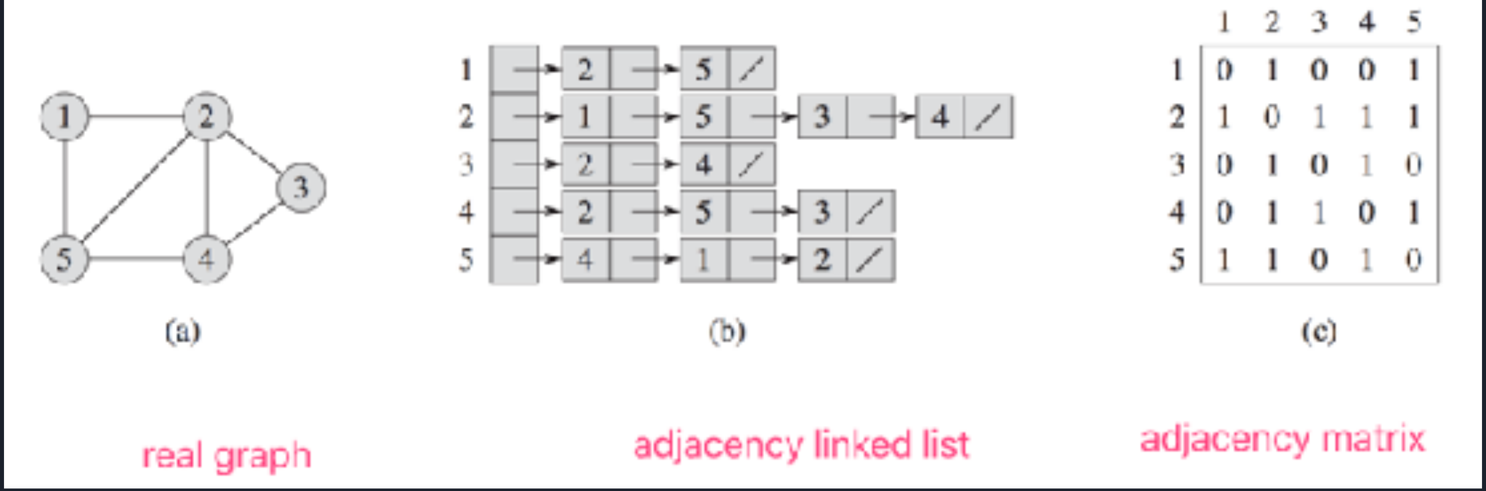
COMP 345 Week 4

Haotao Lai (Eric)
h_lai@encs.concordia.ca
http://laihaotao.me/ta

Graph

How can we represent a graph



$$
\begin{array}{c|ccccc}
 & 1 & 2 & 3 & 4 & 5 \\
\hline
1 & 0 & 1 & 0 & 0 & 1 \\
2 & 1 & 0 & 1 & 1 & 1 \\
3 & 0 & 1 & 0 & 1 & 0 \\
4 & 0 & 1 & 1 & 0 & 1 \\
5 & 1 & 1 & 0 & 1 & 0
\end{array}
$$

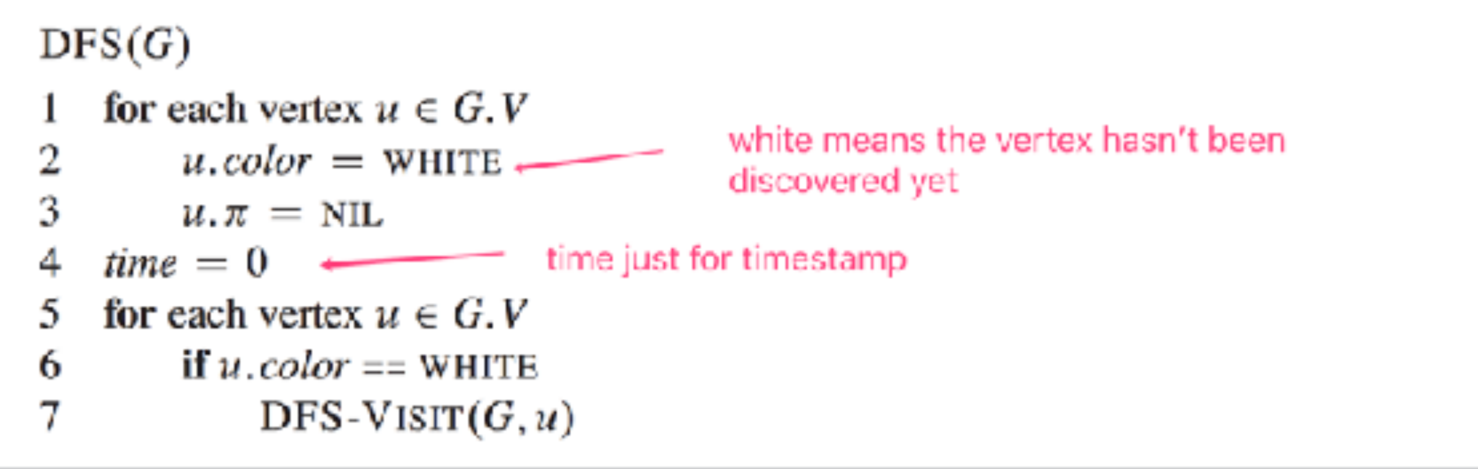
How to traverse a graph

There are a lot of ways to do it, the most common two is DFS and BFS.

You are not restricted in this two ways, during your demo ! ! !

Take DFS as an Example

Recursion

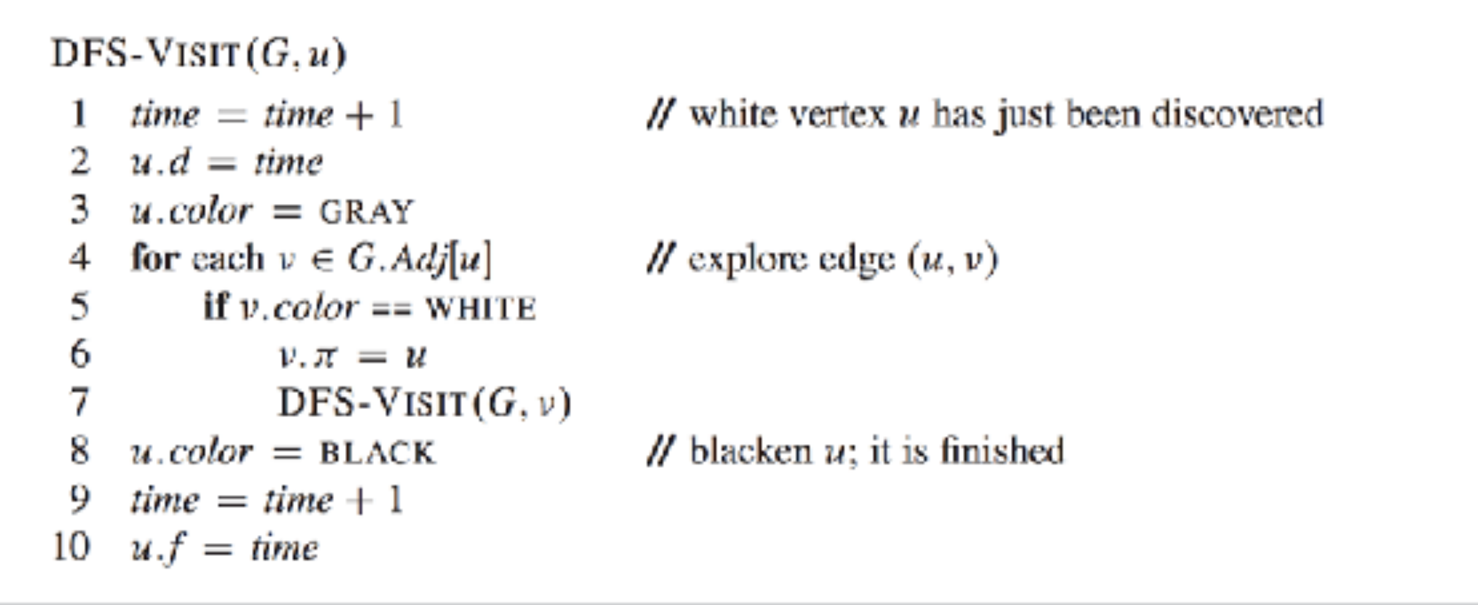


```
DFS(G)
for each vertex $u \in G.V$
    $u.color$ = WHITE
    $u.\pi$ = NIL
$time$ = 0
for each vertex $u \in G.V$
    if $u.color$ == WHITE
        DFS-VISIT$(G, u)$
```

white means the vertex hasn't been discovered yet

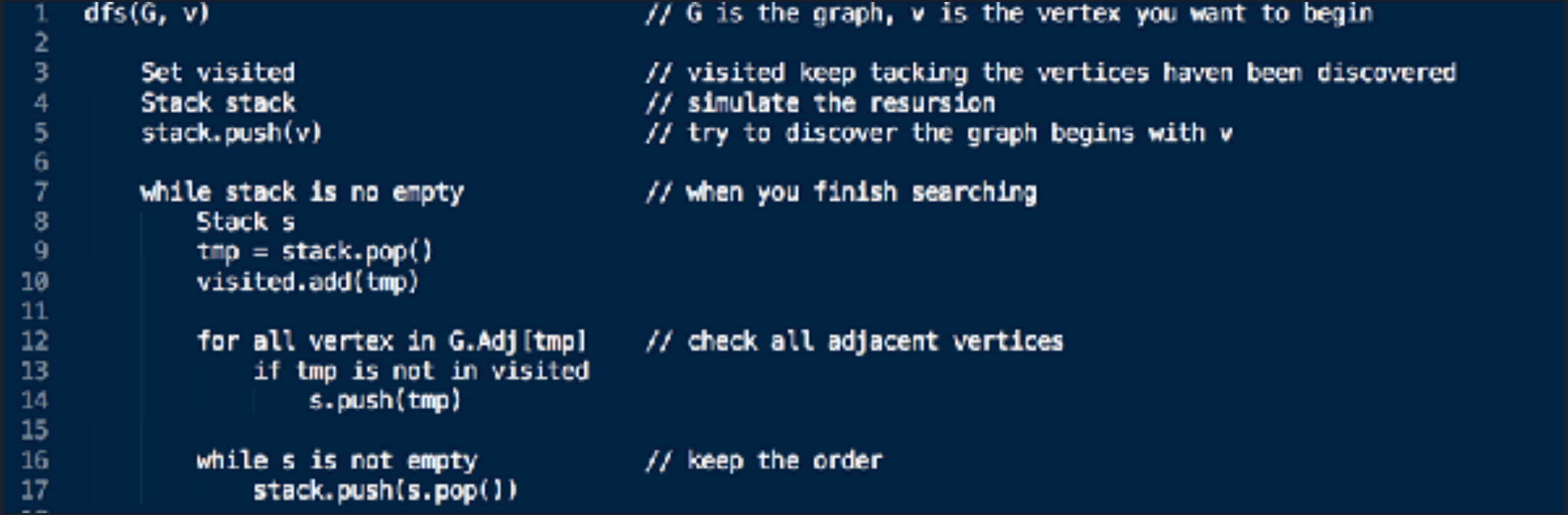
time just for timestamp

Recursion (continue)



```
DFS-VISIT$(G, u)$
time = time + 1                 // white vertex $u$ has just been discovered
$u.d$ = time
$u.color$ = GRAY
for each $v \in G.Adj[u]$       // explore edge $(u, v)$
    if $v.color$ == WHITE
        $v.\pi = u$
        DFS-VISIT$(G, v)$
$u.color$ = BLACK               // blacken $u$; it is finished
time = time + 1
$u.f$ = time
```

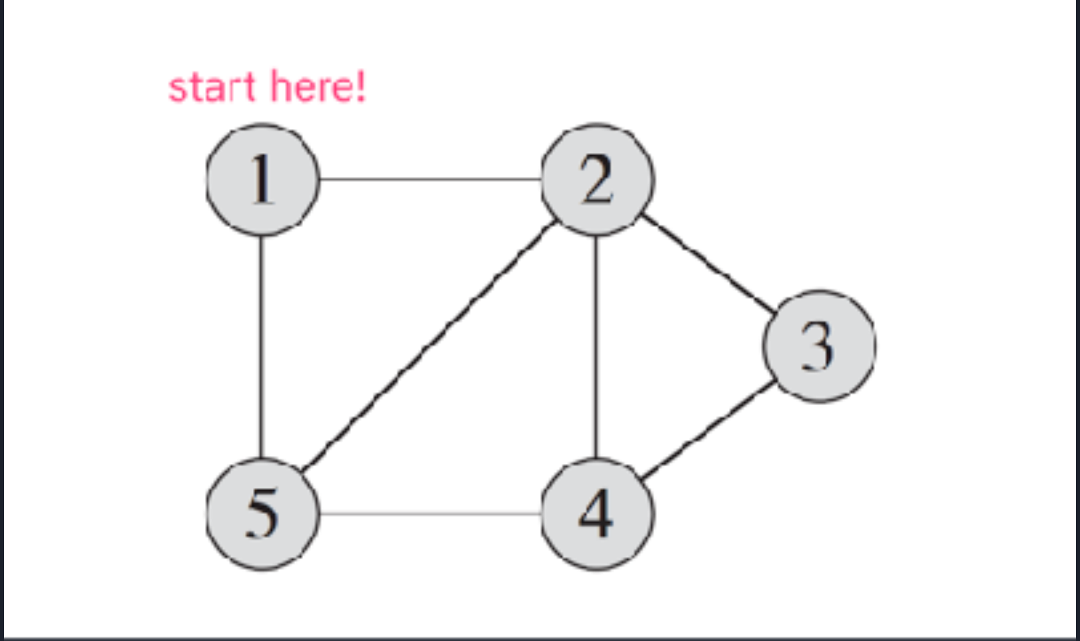
Loop



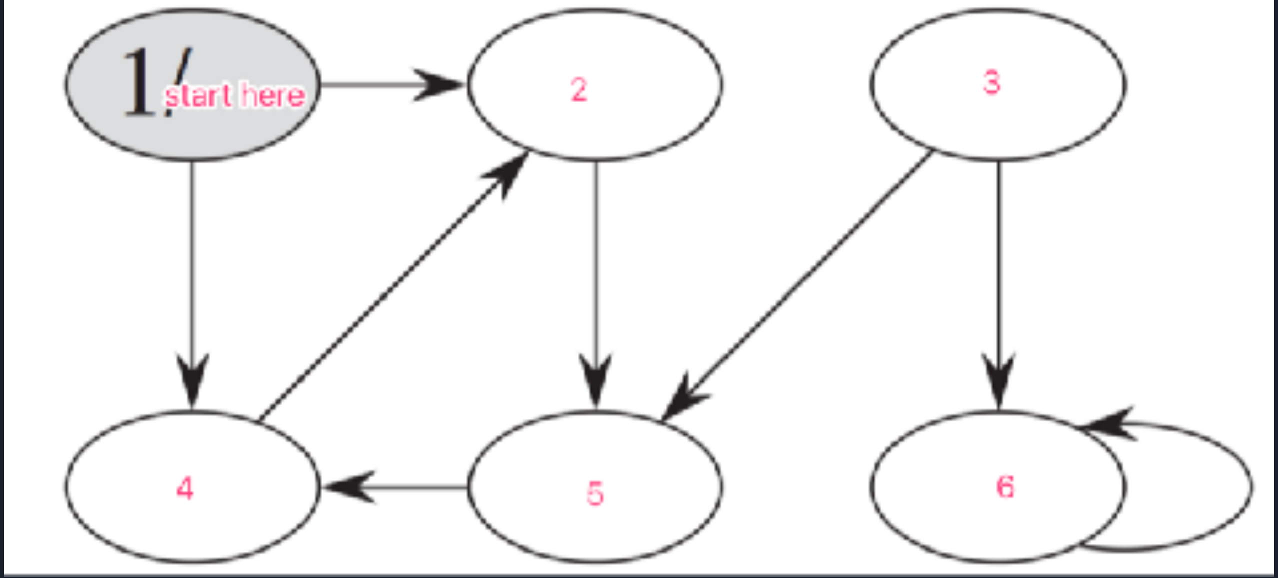
```
dfs(G, v)                           // G is the graph, v is the vertex you want to begin

    Set visited                     // visited keep tacking the vertices haven been discovered
    Stack stack                     // simulate the resursion
    stack.push(v)                   // try to discover the graph begins with v

    while stack is no empty         // when you finish searching
        Stack s
        tmp = stack.pop()
        visited.add(tmp)

        for all vertex in G.Adj[tmp]    // check all adjacent vertices
            if tmp is not in visited
                s.push(tmp)

        while s is not empty        // keep the order
            stack.push(s.pop())
```

Let’s do an example

Example 1 Undirected Graph



Example 2 Directed Graph



1/
start here
2
3
4
5
6