16.3
Depth First Search (DFS)

16.3.1
Depth First Search (DFS) in Undirected Graphs

Depth First Search

1. **DFS** special case of Basic Search.
2. **DFS** is useful in understanding graph structure.
3. **DFS** used to obtain linear time ($O(m + n)$) algorithms for
    1. Finding cut-edges and cut-vertices of undirected graphs
    2. Finding strong connected components of directed graphs
4. ...many other applications as well.

DFS in Undirected Graphs

Recursive version. Easier to understand some properties.

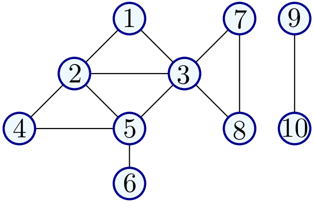
```
DFS(G)
    for all u ∈ V(G) do
        Mark u as unvisited
        Set pred(u) to null
    T is set to ∅
    while ∃ unvisited u do
        DFS(u)
    Output T
```

```
DFS(u)
    Mark u as visited
    for each uv in Out(u) do
        if v is not visited then
            add edge uv to T
            set pred(v) to u
            DFS(v)
```

Implemented using a global array ***Visited*** for all recursive calls.

T is the search tree/forest.

Example



Edges classified into two types: $uv \in E$ is a

1. tree edge: belongs to T
2. non-tree edge: does not belong to T

Properties of DFS tree

Proposition

1. T *is a forest*
2. *connected components of* T *are same as those of* G.
3. *If* $uv \in E$ *is a non-tree edge then, in* T, *either:*
    1. u *is an ancestor of* v, *or*
    2. v *is an ancestor of* u.

Question: Why are there no cross-edges?

Exercise

Prove that **DFS** of a graph G with n vertices and m edges takes $O(n + m)$ time.

THE END

...

(for now)