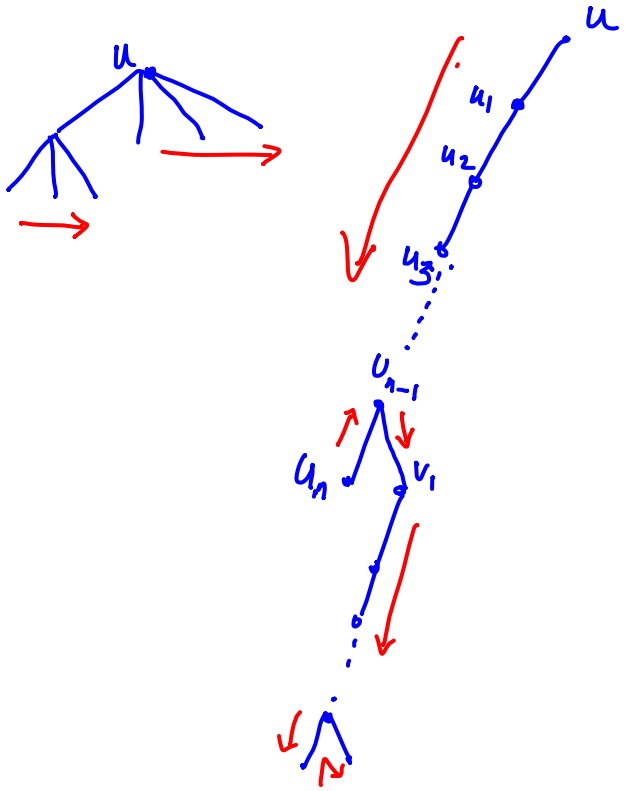
Depth First Search (DFS)



Precise description is given by means of an algorithm (as in the case of BFS)

$v.col$ — White, Grey, Black

$v.\pi$ — Set of DFS trees (DFS forest)

$v.d$, $v.f$ — Timestamps $\in \{1, \ldots, 2|V|\}$

$v.d$: discovery of vertex v

$v.f$: Finishing time of vertex v (Backtracking from)

White

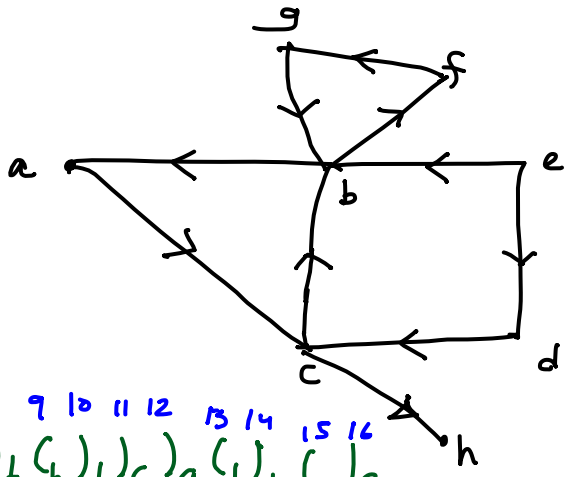
Grey

Black

```
DFS(G)          // G = (V, E)
  for v ∈ V do
      v.col = White
      v.π   = nil
  time = 0
  for v ∈ V do
     if (v.col == White)
          DFS-Visit(G, v)
```

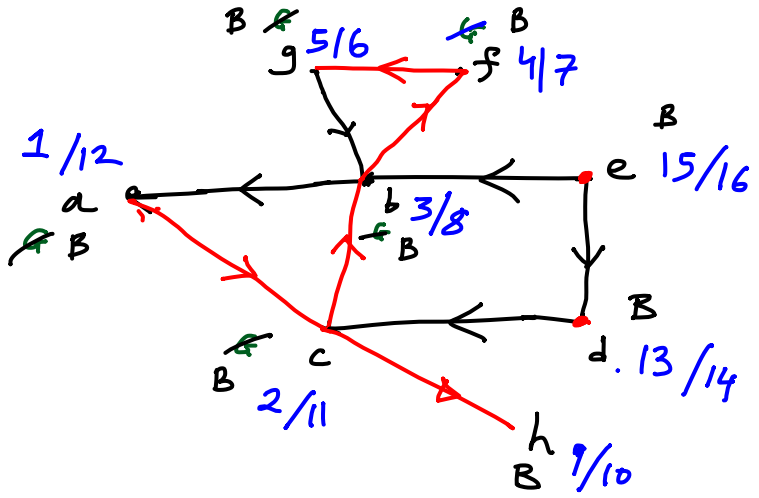
```
DFS-Visit(G, v)
     v.col = Grey
     time = time+1
     v.d  = time
     for w s.t. (v,w) ∈ E do
         if (w.col == White)
              w.π = v
              DFS-visit(G, w)
     v.col = Black
     time = time+1
     v.f  = time
```

// DFS(G)



1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

$(_a(_c(_b.(_f(_g)_g)_f)_b(_h)_h)_c)_a(_d)_d(_e)_e$



Time Complexity

Aggregate Analysis

- DFS-visit(G, v) is called exactly once from each $v \in V$
- In DFS-visit(G, v), the for loop is executed

$$\sum_{v \in V} |\{(v,w) \in E\}| \text{ time}$$

$$\sum_{v \in V} \text{outdeg}(v) = |E| \quad \text{(in directed graph)}$$

$$= 2 \cdot |E| \quad \text{(in Undirected graph)}$$

Total time in

$|V|$ (initialization)

$|V|$ (total DFS-visit(G, v) instructions)

$|E| + 6|V|$ (for all instructions in DFS-visit(G,w) excluding DFS-visit(G, w) instructions, summed over all $v \in V$)

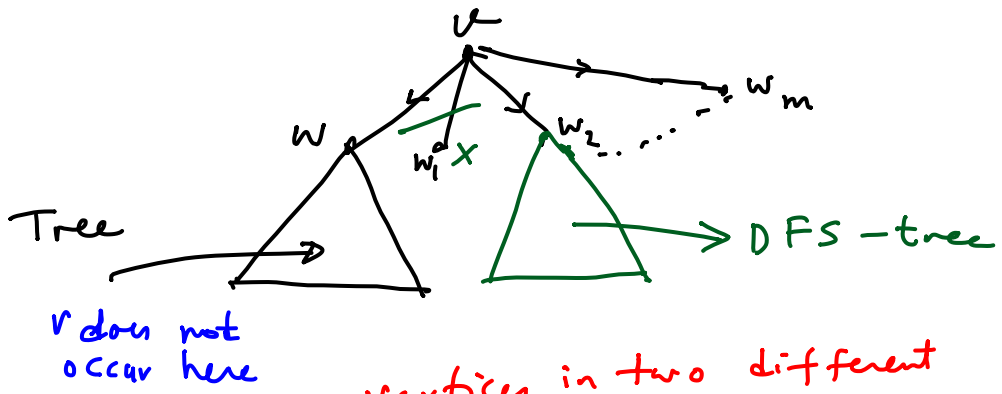
$O(|V| + |E|)$

Properties of DFS

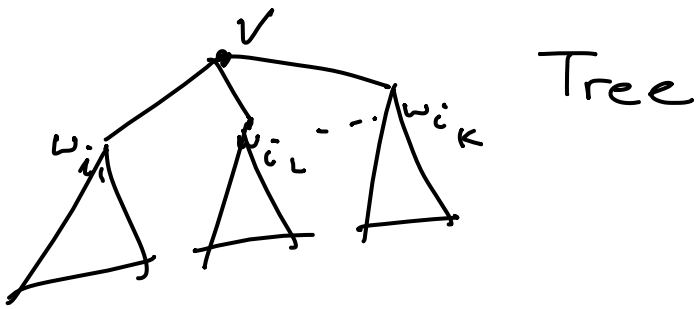
$G_\pi = (V, E_\pi)$, where $E_\pi = \{ (v.\pi, v) \mid v \in V, v.\pi \neq nil \}$

- G_π is a forest of trees.

Any DFS-visit(G,v) call from program DFS(G), makes v as a root ($v.\pi = nil$)



An edge (v,w) is created in G_π iff a call is made to DFS-visit(G,w) from DFS-visit(G,v).



vertices in two different subtrees w_i, w_j are disjoint because call to each DFS(G, w_i) is made exactly once.

Each call to DFS-visit(G, v) from DFS(G) results in a single DFS tree.

All calls to DFS-visit(G, v) from DFS(G) result in a forest of trees (called DFS forest)

- A vertex s is a descendent of v iff s is discovered while v.col is Grey.
  (Whole construction of tree rooted at v happens when v is Grey)

- If we denote by '$(r$' discovery of vertex r and by '$)_r$' the finishing (or backtracking) of r and list all such events in temporal order then resulting sequence is a well formed parathesized expression.

In our example this sequence is shown along with the example.

An equivalent way to formulate parenthesis property in the following Lemma.

Lemma: For any $u, v \in V$ exactly one of the following holds.

(i) $(u.d, u.f)$ is contained in $(v.d, v.f)$. In this case u is a descendent of v.

(ii) $(v.d, v.f)$ is contained in $(u.d, u.f)$. [$u.d < v.d < v.f < u.f$] In this case v is a descendent of u.

(iii) $(u.d, u.f)$ is disjoint from $(v.d, v.f)$.

Proof: Let $u.d < v.d$ (other case is symmetrical)

Now consider two subcases:

(1) $u.f < v.d$ (2) $u.f > v.d$

Subcase 1: $u.d < u.f < v.d < v.f$

Vertex v is discovered when u is finished

neither u is descendent of v nor v is descendent of u.

$(u.d, u.f) \cap (v.d, v.f)$ are disjoint.

Subcase 2: $u.d < v.d < u.f$

v is discovered while u is grey.

$\Rightarrow$ Call to DFS-visit(G, v) is made while executing Call to DFS-visit(G, u)

$\Rightarrow$ DFS-visit(G, v) must finish before DFS-visit(G, u)

$\Rightarrow$ $u.d < v.d < v.f < u.f$

$\Rightarrow$ $(v.d, v.f)$ is contained in $(u.d, u.f)$.

Further, v is a descendent of u.

□