Lecture 15

Graph Searching: Depth-First Search and Topological Sort
Depth-First Search

- Graph $G=(V,E)$ directed or undirected
- Adjacency list representation
- **Goal**: Systematically explore every vertex and every edge
- **Idea**: search deeper whenever possible
  - Using a LIFO queue (Stack; FIFO queue used in BFS)
Depth-First Search

• Maintains several fields for each $v \in V$
• Like BFS, **colors** the vertices to indicate their states. Each vertex is
  – Initially *white*,
  – *grayed* when discovered,
  – *blackened* when finished
• Like BFS, records **discovery** of a white $v$ during scanning $\text{Adj}[u]$ by $\pi[v] \leftarrow u$
Depth-First Search

• Unlike BFS, predecessor graph $G_\pi$ produced by DFS forms spanning forest

• $G_\pi=(V,E_\pi)$ where

$$E_\pi = \{(\pi[v], v) : v \in V \text{ and } \pi[v] \neq \text{NIL}\}$$

• $G_\pi = \text{depth-first forest (DFF) is composed of disjoint depth-first trees (DFTs)}$
Depth-First Search

- DFS also timestamps each vertex with two timestamps.
  - \(d[v]\): records when \(v\) is first discovered and grayed.
  - \(f[v]\): records when \(v\) is finished and blackened.
- Since there is only one discovery event and finishing event for each vertex we have \(1 \leq d[v] < f[v] \leq 2|V|\).

Time axis for the color of a vertex:
Depth-First Search

DFS(G)

for each \( u \in V \) do

\[ \text{color}[u] \leftarrow \text{white} \]

\[ \pi[u] \leftarrow \text{NIL} \]

\[ \text{time} \leftarrow 0 \]

for each \( u \in V \) do

if \( \text{color}[u] = \text{white} \) then

DFS-VISIT(G, u)

DFS-VISIT(G, u)

DFS-VISIT(G, u)

\[ \text{color}[u] \leftarrow \text{gray} \]

\[ \text{d}[u] \leftarrow \text{time} \leftarrow \text{time} + 1 \]

for each \( v \in \text{Adj}[u] \) do

if \( \text{color}[v] = \text{white} \) then

\[ \pi[v] \leftarrow u \]

DFS-VISIT(G, v)

\[ \text{color}[u] \leftarrow \text{black} \]

\[ f[u] \leftarrow \text{time} \leftarrow \text{time} + 1 \]
Depth-First Search

- Running time: $\Theta(V+E)$
- Initialization loop in $\text{DFS}$: $\Theta(V)$
- Main loop in $\text{DFS}$: $\Theta(V)$ exclusive of time to execute calls to $\text{DFS-VISIT}$
- $\text{DFS-VISIT}$ is called exactly once for each $v \in V$ since
  - $\text{DFS-VISIT}$ is invoked only on white vertices and
  - $\text{DFS-VISIT}(G, u)$ immediately colors $u$ as gray
- For loop of $\text{DFS-VISIT}(G, u)$ is executed $|\text{Adj}[u]|$ time
- Since $\sum |\text{Adj}[u]| = E$, total cost of executing loop of $\text{DFS-VISIT}$ is $\Theta(E)$
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example

Graph with nodes labeled 1, 2, 3, 4, x, y, z, s, w, v, u, and t.
Depth-First Search: Example

Diagram showing a graph with nodes labeled from s to z and edges connecting them.
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example

[Diagram of a graph with nodes labeled s, x, y, z, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, w, v, t, u, and arrows connecting them.]

Depth-first search (DFS) is a graph traversal algorithm that explores as far as possible along each branch before backtracking. It starts at a given node (the root) and explores as far as possible along each branch before backtracking.
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example

Diagram of a graph with nodes labeled s, t, u, v, w, x, y, z, and numbers 1 through 12. The graph shows directed edges between the nodes.
Depth-First Search: Example

The image shows a graph with nodes labeled from 's' to 'u', connected by directed edges. The nodes contain numbers, and the edges indicate the direction of traversal. The graph starts at node 's' and explores as deeply as possible along each branch before backtracking.
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
## Depth-First Search: Example

<table>
<thead>
<tr>
<th>DFS(G) terminated</th>
<th>Depth-first forest (DFF)</th>
</tr>
</thead>
</table>

DFS \((G)\) terminated

Depth-first forest (DFF)
DFS: Parenthesis Theorem

Thm: In any DFS of $G=(V,E)$, let $\text{int}[v] = [d[v], f[v]]$ then exactly one of the following holds for any $u$ and $v \in V$

• $\text{int}[u]$ and $\text{int}[v]$ are entirely disjoint

• $\text{int}[v]$ is entirely contained in $\text{int}[u]$ and $v$ is a descendant of $u$ in a DFT

• $\text{int}[u]$ is entirely contained in $\text{int}[v]$ and $u$ is a descendant of $v$ in a DFT
Parenthesis Thm
(proof for the case $d[u] < d[v]$)

Subcase $d[v] < f[u]$ (int[$u$] and int[$v$] are overlapping)
- $v$ was discovered while $u$ was still GRAY
- This implies that $v$ is a descendant of $u$
- So search returns back to $u$ and finishes $u$ after finishing $v$
- i.e., $d[v] < f[u] \Rightarrow \text{int}[v]$ is entirely contained in int[$u$]

Subcase $d[v] > f[u] \Rightarrow \text{int}[v]$ and int[$u$] are entirely disjoint

Proof for the case $d[v] < d[u]$ is similar (dual) QED
Nesting of Descendents’ Intervals

Corollary 1 (Nesting of Descendents’ Intervals):

\( v \) is a descendant of \( u \) if and only if

\[ d[u] < d[v] < f[v] < f[u] \]

Proof: immediate from the Parenthesis Thrm

QED
Parenthesis Theorem

\[ (x \ (s \ (w \ w)) \ (v \ v) \ s) \ (y \ (t \ t) \ y) \ x) \ (z \ (u \ u) \ z) \]
Edge Classification in a DFF

Tree Edge: discover a new (WHITE) vertex
▷ GRAY to WHITE ◁

Back Edge: from a descendent to an ancestor in DFT
▷ GRAY to GRAY ◁

Forward Edge: from ancestor to descendent in DFT
▷ GRAY to BLACK ◁

Cross Edge: remaining edges (btwn trees and subtrees)
▷ GRAY to BLACK ◁

Note: ancestor/descendent is wrt Tree Edges
Edge Classification in a DFF

• How to decide which GRAY to BLACK edges are forward, which are cross
Let BLACK vertex $v \in \text{Adj}[u]$ is encountered while processing GRAY vertex $u$
  – $(u,v)$ is a forward edge if $d[u] < d[v]$
  – $(u,v)$ is a cross edge if $d[u] > d[v]$
Depth-First Search: Example
Depth-First Search: Example

Graph:
- Nodes: s, w, v, u, x, y, z, t
- Edges: s→1, 1→w, w→v, v→t, t→z
- s→x, x→y, y→z
- s→w, w→v, v→u

Depth-first traversal order: s, w, v, t, z, x, y
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example

![Depth-First Search Diagram]
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
DFS on Undirected Graphs

- Ambiguity in edge classification, since $(u, v)$ and $(v, u)$ are the same edge
  - First classification is valid (whichever of $(u, v)$ or $(v, u)$ is explored first)

**Lemma 1**: any DFS on an undirected graph produces only Tree and Back edges
Lemma 1: Proof

Assume \((x,z)\) is a \(F\) (F?)
But \((x,z)\) must be a \(B\), since DFS must finish \(z\) before resuming \(x\)

Assume \((u,v)\) is a \(C\) (C?) btw subtrees
But \((y,u)\) & \((y,v)\) cannot be both \(T\); one must be a \(B\) and \((u,v)\) must be a \(T\)
If \((u,v)\) is first explored while processing \(u/v\), \((y,v)\) / \((y,u)\) must be a \(B\)
DFS on Undirected Graphs

Lemma 2: an undirected graph is acyclic (i.e. a forest) iff DFS yields no Back edges

Proof

(acyclic ⇒ no Back edges; by contradiction):
Let (u,v) be a Back then color[u] = color[v] = GRAY
⇒ there exists a path between u and v
So, (u,v) will complete a cycle (Back edge ⇒ cycle)

(no Back edges ⇒ acyclic):
If there are no Back edges then there are only T edges by Lemma 1 ⇒ forest ⇒ acyclic

QED
DFS on Undirected Graphs

How to determine whether an undirected graph $G=(V,E)$ is acyclic

- Run a DFS on $G$: if a Back edge is found then there is a cycle
- Running time: $O(V)$, not $O(V + E)$
  - If ever seen $|V|$ distinct edges, must have seen a back edge ($|E| \leq |V| - 1$ in a forest)
DFS: White Path Theorem

WPT: In a DFS of $G$, $v$ is a descendent of $u$ iff at time $d[u]$, $v$ can be reached from $u$ along a WHITE path.

Proof ($\Rightarrow$): assume $v$ is a descendent of $u$

Let $w$ be any vertex on the path from $u$ to $v$ in the DFT.

So, $w$ is a descendent of $u$ $\Rightarrow$ $d[u] < d[w]$

(by Corollary 1 nesting of descendents’ intervals)

Hence, $w$ is white at time $d[u]$
DFS: White Path Theorem

Proof ($\iff$) assume a white path $p(u,v)$ at time $d[u]$ but $v$ does not become a descendent of $u$ in the DFT (contradiction):

Assume every other vertex along $p$ becomes a descendent of $u$ in the DFT
DFS: White Path Theorem

otherwise let \( v \) be the closest vertex to \( u \) along \( p \) that does not become a descendent.

Let \( w \) be predecessor of \( v \) along \( p(u,v) \):

1. \( d[u] < d[w] < f[w] < f[u] \) by Corollary 1
2. Since, \( v \) was WHITE at time \( d[u] \) (\( u \) was GRAY) \( d[u] < d[v] \)
   Since, \( w \) is a descendent of \( u \) but \( v \) is not
3. \( d[w] < d[v] \Rightarrow d[v] < f[w] \)

So by Parenthesis Thm \( \text{int}[v] \) is within \( \text{int}[u] \), \( v \) is descendent of \( u \)  
QED
Directed Acyclic Graphs (DAG)

No directed cycles

Example:
Directed Acyclic Graphs (DAG)

Theorem: a directed graph $G$ is acyclic iff DFS on $G$ yields no Back edges

Proof (acyclic $\Rightarrow$ no Back edges; by contradiction):
Let $(v,u)$ be a Back edge visited during scanning $\text{Adj}[v]$

$\Rightarrow \text{color}[v] = \text{color}[u] = \text{GRAY}$ and $d[u] < d[v]$

$\Rightarrow \text{int}[v]$ is contained in $\text{int}[u] \Rightarrow v$ is descendent of $u$

$\Rightarrow \exists$ a path from $u$ to $v$ in a DFT and hence in $G$

$\therefore$ edge $(v,u)$ will create a cycle (Back edge $\Rightarrow$ cycle)

path from $u$ to $v$ in a DFT and hence in $G$
acyclic iff no **Back** edges

**Proof** (no **Back** edges $\Rightarrow$ acyclic):

Suppose $G$ contains a cycle $C$ (Show that a DFS on $G$ yields a **Back** edge; proof by contradiction)

Let $v$ be the first vertex discovered in $C$ and let $(u,v)$ be proceeding edge in $C$

At time $d[v]$: $\exists$ a white path from $v$ to $u$ along $C$

By **White Path** Thrm $u$ becomes a descendent of $v$ in a DFT

Therefore $(u,v)$ is a **Back** edge (descendent to ancestor)
Topological Sort of a DAG

• Linear ordering ‘<’ of $V$ such that

$(u,v) \in E \Rightarrow u < v$ in ordering

– Ordering may not be unique
– i.e., mapping the partial ordering to total ordering may yield more than one orderings
Topological Sort of a DAG

Example: Getting dressed

Diagram of a directed acyclic graph (DAG) showing the order in which clothing items should be put on. The example includes:
- Under short
- Socks
- Pants
- Shoes
- Shirt
- Belt
- Tie
- Jacket

The diagram illustrates the dependencies between the items, showing which items must be on before others. The numbers likely represent the order in which the items should be put on.
Topological Sort of a DAG

Algorithm

run DFS(G)
when a vertex finished, output it
vertices output in reverse topologically sorted order

Runs in $O(V+E)$ time
Correctness of the Algorithm

Claim: \((u,v) \in E \Rightarrow f[u] > f[v]\)

Proof: consider any edge \((u,v)\) explored by DFS when \((u,v)\) is explored, \(u\) is \text{GRAY}

- if \(v\) is \text{GRAY}, \((u,v)\) is a \text{Back} edge (contradicting acyclic theorem)
- if \(v\) is \text{WHITE}, \(v\) becomes a descendent of \(u\) (b WPT)
  \[\Rightarrow f[v] < f[u]\]
- if \(v\) is \text{BLACK}, \(f[v] < d[u] \Rightarrow f[v] < f[u]\)

QED