Lecture 15

Graph Searching:

Depth-First Search and Topological Sort
DFS: Parenthesis Theorem

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

- $int[u]$ and $int[v]$ are entirely disjoint
- $int[v]$ is entirely contained in $int[u]$ and $v$ is a descendant of $u$ in a DFT
- $int[u]$ is entirely contained in $int[v]$ 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
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

[Diagram of a graph with nodes labeled from 1 to 8 and edges connecting them.]

Nodes: 1, 2, 3, 4, 5, 6, 7, 8
Edges: 1-2, 2-3, 3-4, 4-5, 5-6, 6-7, 7-x, x-y, y-z, s-t, t-u, v-w

Nodes in gray: 1, 5, 6
Nodes in white: 2, 3, 4, 7, 8, x, y, z, s, t, v, w, u

Edge labels: T, B, C
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example
Depth-First Search: Example

Depth-First Search (DFS) is a graph traversal algorithm that explores as far as possible along each branch before backtracking. It starts at the root node (or an arbitrary node) and explores the tree downward always, first. When it reaches a leaf node, it backtracks to the most recent node on its path that still has unexplored adjacent nodes and explores those. This process continues until all the nodes have been visited.

The diagram above illustrates a DFS traversal. The nodes are visited in the order: S, T, B, F, C, C, C, T, T, T, T, T, x, y, z. Each path from the root node S to a leaf node is a DFS path.
Depth-First Search: Example

The diagram illustrates the process of depth-first search (DFS) on a graph. The search starts at node S and explores as far as possible along each branch before backtracking. Nodes are visited in the order S, 2, 7, 3, 4, 5, 6, 8, 11, 9, 10, 13, with the edges marking the order of traversal. The final node reached is Z, which is an example of a depth-first search.
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
Lemma 2: an undirected graph is acyclic (i.e. a forest) iff DFS yields no Back edges

Proof

(acyclic $\Rightarrow$ no Back edges; by contradiction):

Let $(u,v)$ be a Back then color[$u$] = color[$v$] = GRAY

$\Rightarrow$ there exists a path between $u$ and $v$

So, $(u,v)$ will complete a cycle (Back edge $\Rightarrow$ cycle)

(no Back edges $\Rightarrow$ acyclic):

If there are no Back edges then there are only T edges by Lemma 1 $\Rightarrow$ forest $\Rightarrow$ 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 ($\Leftarrow$) 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.

$p(u,v)$ at time $d[u]$
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 \text{WHITE} at time \( d[u] \) \( (u \) was \text{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 \( \text{Adj}[v] \)

\[ \Rightarrow \text{color}[v] = \text{color}[u] = \text{GRAY} \text{ and } d[u] < d[v] \]
\[ \Rightarrow \text{int}[v] \text{ is contained in } \text{int}[u] \Rightarrow v \text{ is descendent of } u \]
\[ \Rightarrow \exists \text{ a path from } u \text{ to } v \text{ in a DFT and hence in } G \]
\[ : \text{ edge } (v,u) \text{ will create a cycle (Back edge } \Rightarrow \text{ cycle) } \]
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
  $$(u,v) \in E \Rightarrow u < v \text{ in ordering}$$
  - 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 showing the topological order of dressing with nodes representing clothing items and edges showing dependencies.

1. Under short
2. Socks
3. Pants
4. Shoes
5. Shirt
6. Belt
7. Tie
8. Jacket

Order: 11/16, 12/15, 6/7, 13/14, 9/10, 1/8, 2/5, 3/4
Topological Sort of a DAG

Algorithm

run \text{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: \((u,v) \in E \implies f[u] > f[v]\)

Proof: consider any edge \((u,v)\) explored by DFS

when \((u,v)\) is explored, \(u\) is GRAY

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

QED