Basic Graph Algorithms

- Exploring Graphs
- Depth First Search
- DFS Forest - Start and Finish Time
- DAG, Topological Sorting
- Strongly Connected Components
- Breadth First Search
- Bipartite Graphs

Imdad ullah Khan

Reachability

Given a graph $G = (V, E)$

A vertex v is reachable form u, if there exists a path from u to v

- $R(u)$: the set of vertices reachable from u

$$R(u) \ : \ \{ v : \exists \text{ a path from } u \text{ to } v \ \}$$

Fundamental graph exploration problems

- Given s and u, is $u \in R(s)$?
- Given s, find $R(s)$

A more constructive/algorithmic definition of reachability:

v is reachable form s, if v is a neighbor of s or v is reachable from a neighbor of s

$$R(s) \ = \ \{s\} \bigcup_{x \in N(s)} R(x)$$

Explore

Input: A graph $G = (V, E)$ and a node $s \in V$
Output: All nodes that are reachable from s, $R(s)$

- $R(s)$ is saved as a bit-vector $visited[1 \dots n]$ $\triangleright$ fixed order
- $visited[i] = 1 \Leftrightarrow v_i \in R(s)$
- Populate $visited[\cdot]$ using $R(s) = \{s\} \bigcup_{x \in N(s)} R(x)$

Algorithm REC-EXPLORE(s)

```
visited[·] ← ZEROS(n)
function EXPLORE(G,s)
    visited[s] ← 1
    for x ∈ N(s) do
        if visited[x] = 0 then
            EXPLORE(G,x)
```

Algorithm ITR-EXPLORE(s)

```
visited ← ZEROS(n)
INSERT(todo, s)
while todo ≠ ∅ do
    u ← REMOVE(todo)
    visited[u] ← 1
    for x ∈ N(u) do
        if visited[x] = 0 then
            INSERT(todo, x)
```

Exploring the whole graph

- EXPLORE is still under-specified to analyze its runtime
- The EXPLORE procedure visits only the portion of graph that is reachable from the given source s
- To examine the rest of the graph, we restart the procedure elsewhere, at some unvisited vertex
- What if we start EXPLORE from some already visited vertex?
- Notice that algorithm readily extends to directed graphs

Depth First Search

Input: A graph $G = (V, E)$

Output: *'Explore'* the graph G

Algorithm $\textsc{dfs}(G)$

```
visited ← ZEROS(n)
for all s ∈ V do
    if visited[s] = 0 then
        DFS-EXPLORE(s)
```

Algorithm $\textsc{dfs-explore}(s)$

```
function DFS-EXPLORE(s)
    visited[s] ← 1
    for x ∈ N(s) do
        if visited[x] = 0 then
            DFS-EXPLORE(x)
```

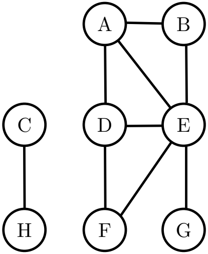
Depth First Search

Algorithm $\text{DFS}(G)$

$visited \leftarrow \text{ZEROS}(n)$
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    $\text{DFS-EXPLORE}(s)$

Algorithm $\text{DFS-EXPLORE}(s)$

function $\text{DFS-EXPLORE}(s)$
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      $\text{DFS-EXPLORE}(x)$



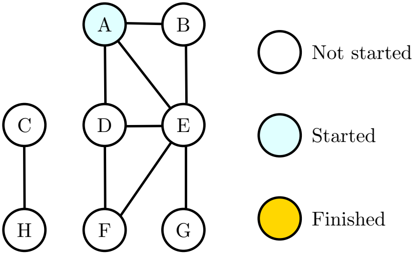
Depth First Search

Algorithm DFS(G)

$visited \leftarrow$ ZEROS(n)
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

function DFS-EXPLORE(s)
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      DFS-EXPLORE(x)



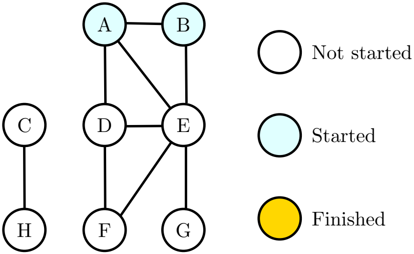
Depth First Search

Algorithm DFS(G)

$visited \leftarrow$ ZEROS(n)
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

function DFS-EXPLORE(s)
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      DFS-EXPLORE(x)



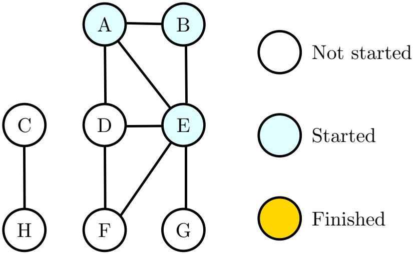
Depth First Search

Algorithm $\textsc{dfs}(G)$

$visited \leftarrow \textsc{zeros}(n)$
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    $\textsc{dfs-explore}(s)$

Algorithm $\textsc{dfs-explore}(s)$

function $\textsc{dfs-explore}(s)$
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      $\textsc{dfs-explore}(x)$



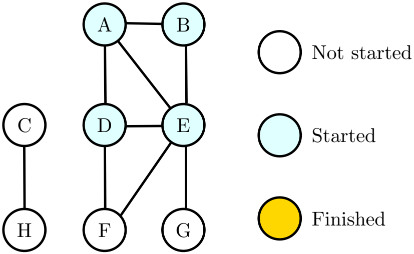
Depth First Search

Algorithm DFS(G)

$visited \leftarrow$ ZEROS(n)
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

function DFS-EXPLORE(s)
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      DFS-EXPLORE(x)



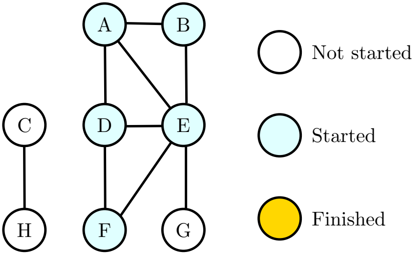
Depth First Search

Algorithm DFS(G)

$visited \leftarrow$ ZEROS(n)
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

function DFS-EXPLORE(s)
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      DFS-EXPLORE(x)



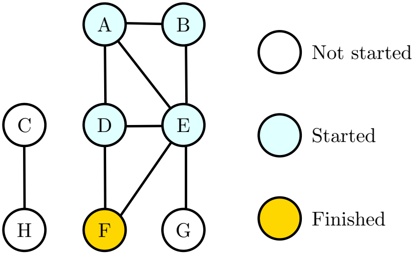
Depth First Search

Algorithm DFS(G)

visited $\leftarrow$ ZEROS(n)
for all $s \in V$ **do**
  if *visited*$[s] = 0$ **then**
    DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

function DFS-EXPLORE(s)
  visited$[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if *visited*$[x] = 0$ **then**
      DFS-EXPLORE(x)



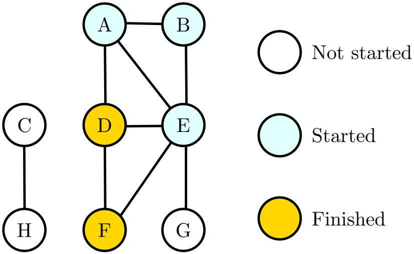
Depth First Search

Algorithm $\textsc{dfs}(G)$

$visited \leftarrow \textsc{zeros}(n)$
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    $\textsc{dfs-explore}(s)$

Algorithm $\textsc{dfs-explore}(s)$

function $\textsc{dfs-explore}(s)$
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      $\textsc{dfs-explore}(x)$



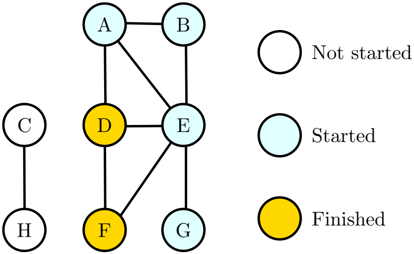
Depth First Search

Algorithm $\textsc{dfs}(G)$

visited $\leftarrow$ $\textsc{zeros}(n)$
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    $\textsc{dfs-explore}(s)$

Algorithm $\textsc{dfs-explore}(s)$

function $\textsc{dfs-explore}(s)$
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      $\textsc{dfs-explore}(x)$



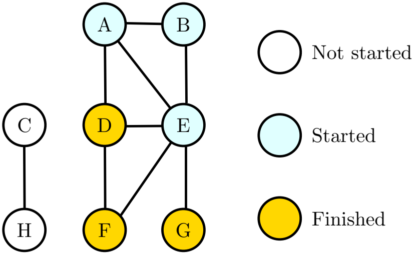
Depth First Search

Algorithm DFS(G)

$visited \leftarrow$ ZEROS(n)
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

function DFS-EXPLORE(s)
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      DFS-EXPLORE(x)



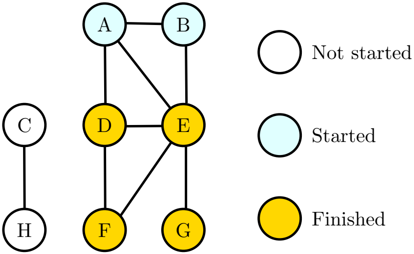
Depth First Search

Algorithm DFS(G)

> *visited* $\leftarrow$ ZEROS(n)
> **for** all $s \in V$ **do**
>   **if** $visited[s] = 0$ **then**
>   DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

> **function** DFS-EXPLORE(s)
> $visited[s] \leftarrow 1$
>   **for** $x \in N(s)$ **do**
>   **if** $visited[x] = 0$ **then**
>   DFS-EXPLORE(x)



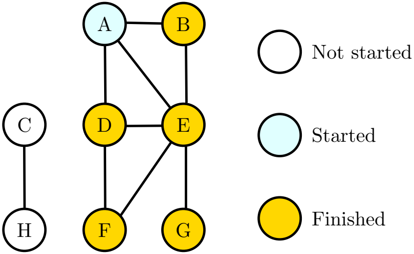
Depth First Search

Algorithm $\text{DFS}(G)$

$visited \leftarrow \text{ZEROS}(n)$
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    $\text{DFS-EXPLORE}(s)$

Algorithm $\text{DFS-EXPLORE}(s)$

function $\text{DFS-EXPLORE}(s)$
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      $\text{DFS-EXPLORE}(x)$



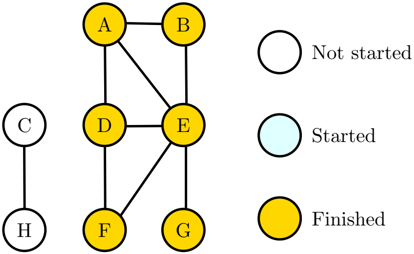
Depth First Search

Algorithm $\text{DFS}(G)$

$visited \leftarrow \text{ZEROS}(n)$
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    $\text{DFS-EXPLORE}(s)$

Algorithm $\text{DFS-EXPLORE}(s)$

function $\text{DFS-EXPLORE}(s)$
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      $\text{DFS-EXPLORE}(x)$



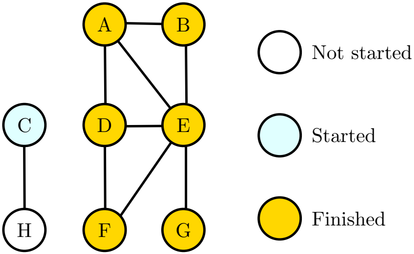
Depth First Search

Algorithm DFS(G)

$visited \leftarrow$ ZEROS(n)
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

function DFS-EXPLORE(s)
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      DFS-EXPLORE(x)



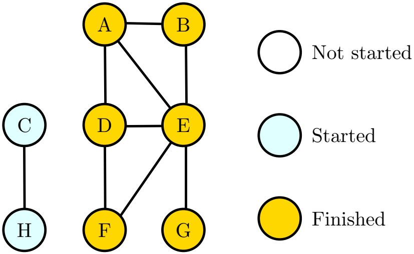
Depth First Search

Algorithm $\text{DFS}(G)$

$visited \leftarrow \text{ZEROS}(n)$
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    $\text{DFS-EXPLORE}(s)$

Algorithm $\text{DFS-EXPLORE}(s)$

function $\text{DFS-EXPLORE}(s)$
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      $\text{DFS-EXPLORE}(x)$



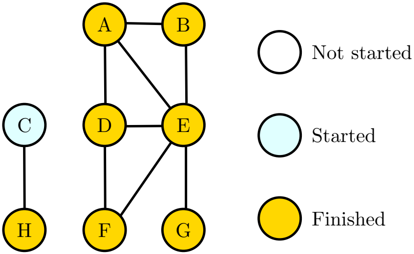
Depth First Search

Algorithm DFS(G)

$visited \leftarrow$ ZEROS(n)
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

function DFS-EXPLORE(s)
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      DFS-EXPLORE(x)



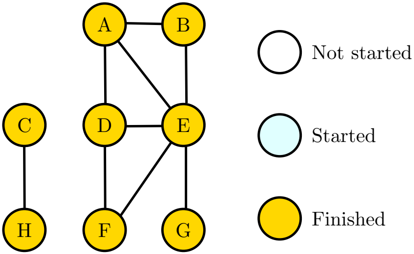
Depth First Search

Algorithm DFS(G)

$visited \leftarrow$ ZEROS(n)
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

function DFS-EXPLORE(s)
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      DFS-EXPLORE(x)



Depth First Search: Forest

Algorithm DFS(G)

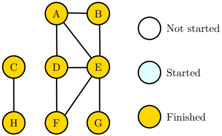
$visited \leftarrow$ ZEROS(n)
for all $s \in V$ **do**
  if $visited[s] = 0$ **then**
    DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

function DFS-EXPLORE(s)
  $visited[s] \leftarrow 1$
  for $x \in N(s)$ **do**
    if $visited[x] = 0$ **then**
      DFS-EXPLORE(x)

When call to DFS-EXPLORE(s) is executed, all vertices in $R(s)$ are visited

- When DFS-EXPLORE(u) is finished one 'DFS *tree*' is formed containing all vertices reachable from u
- The next DFS-EXPLORE(v) called from outer loop forms a new tree



Depth First Search: Runtime

Algorithm DFS(G)

> $visited \leftarrow$ ZEROS(n)
> **for** all $s \in V$ **do**
>   **if** $visited[s] = 0$ **then**
>   DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

> **function** DFS-EXPLORE(s)
> $visited[s] \leftarrow 1$
>   **for** $x \in N(s)$ **do**
>   **if** $visited[x] = 0$ **then**
>   DFS-EXPLORE(x)

- $G = (V, E)$, $|V| = n$, $|E| = m$
- Initialization and visited checking takes $O(n)$ time

Total runtime within all EXPLORE calls is $$\sum_{i=1}^{n} deg(v_i) = 2m$$

- Total runtime is $O(n + m)$

DFS: Summary

Algorithm DFS(G)

visited $\leftarrow$ ZEROS(n)
for all $s \in V$ **do**
  if visited[s] $= 0$ **then**
    DFS-EXPLORE(s)

Algorithm DFS-EXPLORE(s)

function DFS-EXPLORE(s)
  visited[s] $\leftarrow 1$
  for $x \in N(s)$ **do**
    if visited[x] $= 0$ **then**
      DFS-EXPLORE(x)

- DFS explores the entire graph
- Explores one neighbor first (in depth), before going to next neighbor
- Works both for undirected and directed graphs
- Runtime of $O(n + m)$ is equal to reading the input graph
- DFS doesn't add any cost (asymptotically) to any graph algorithm, we typically do it as a pre-processing step
- Answers many questions
  - is graph connected, how many components in the graph, find $R(s)$, $\cdots$
- A fundamental algorithm with many applications; we will discuss some