# Network Science

## Depth-First Search

Joao Meidanis

University of Campinas, Brazil

September 28, 2020

# Summary

# Depth-First Search (DFS) Algorithm

# Depth-First Search

Needs:

- adjacency lists

Provides:

- edge classification
- cycle detection
- topological sort

# Algorithm — Core

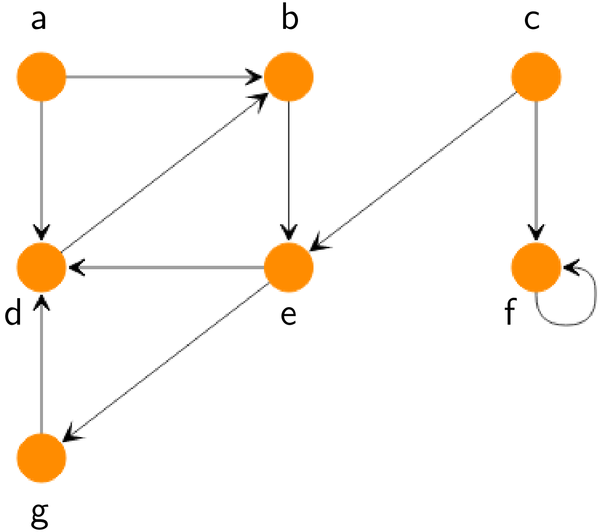
```
function DFS-VISIT(u, Adj)
    for v in Adj[u] do
        if v not in parent then
            parent[v] ← u
            DFS-VISIT(v, Adj)
        end if
    end for
end function
```

# Algorithm — Main

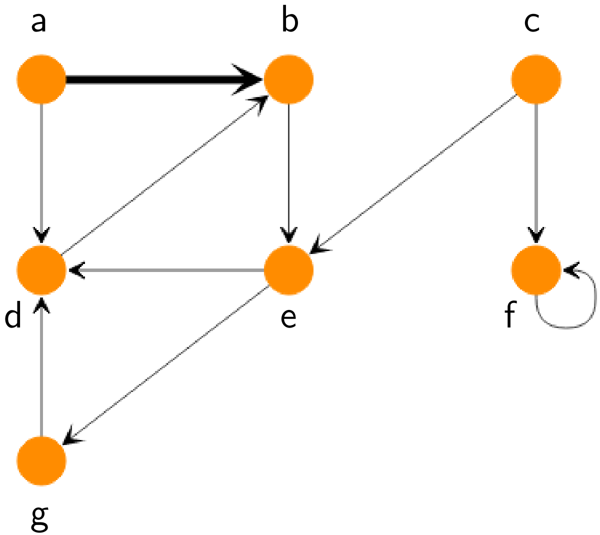
```
function DFS(V, Adj)
    parent ← {}
    for v in V do
        if v not in parent then
            parent[v] ← None
            DFS-Visit(v, Adj)
        end if
    end for
end function
```

# Example

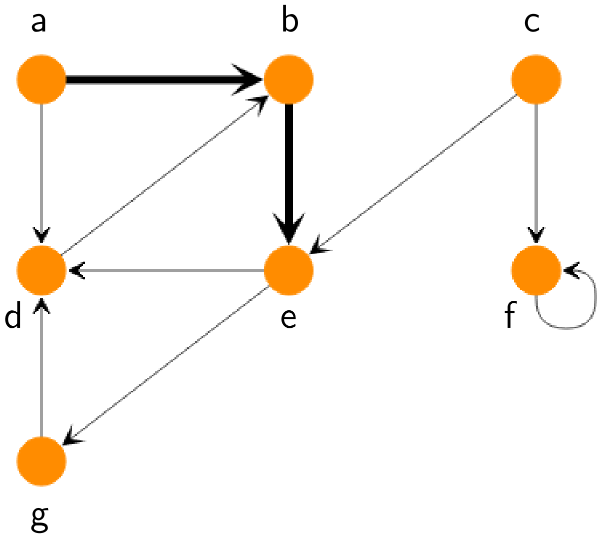
# Example: directed graph



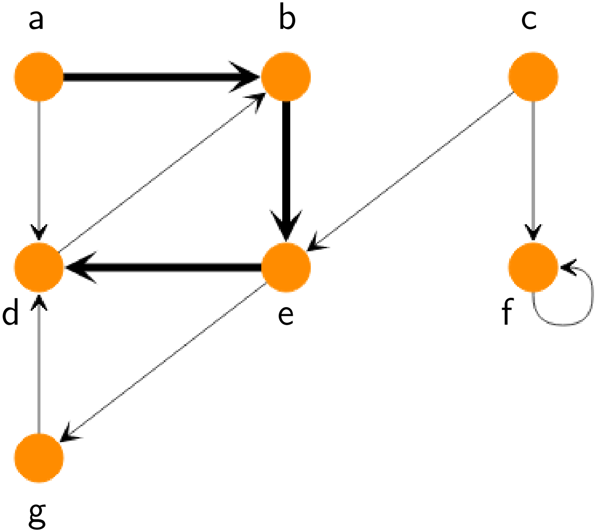
# Example: directed graph



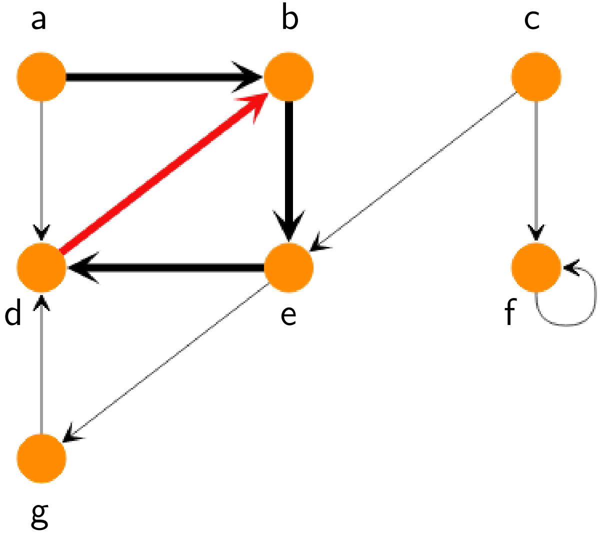
# Example: directed graph



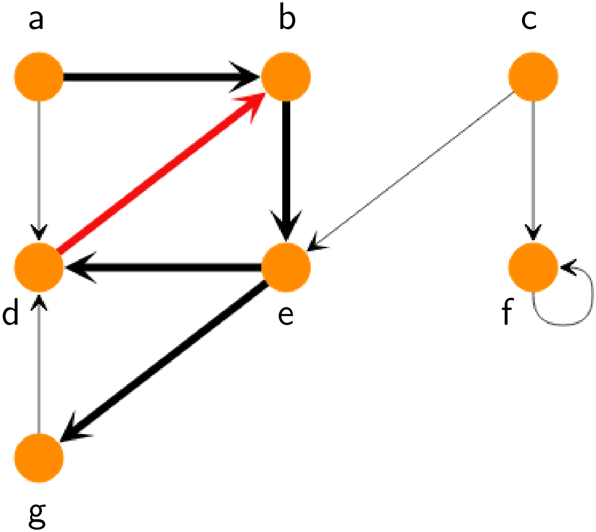
# Example: directed graph



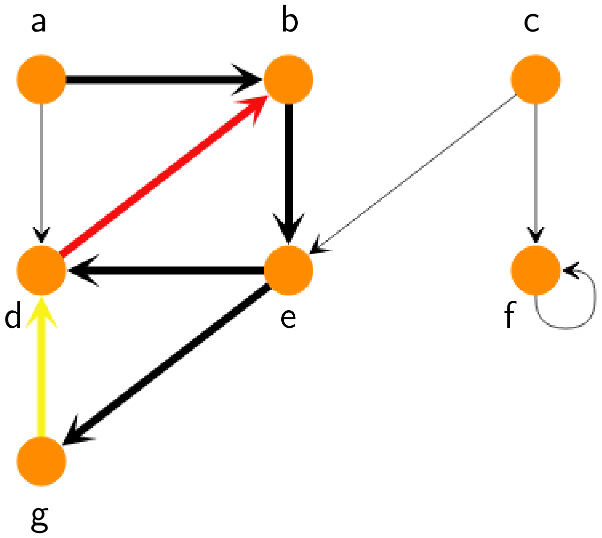
# Example: directed graph



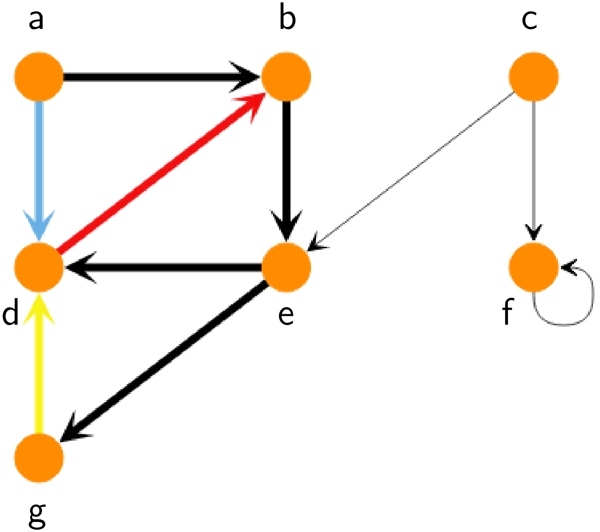
# Example: directed graph



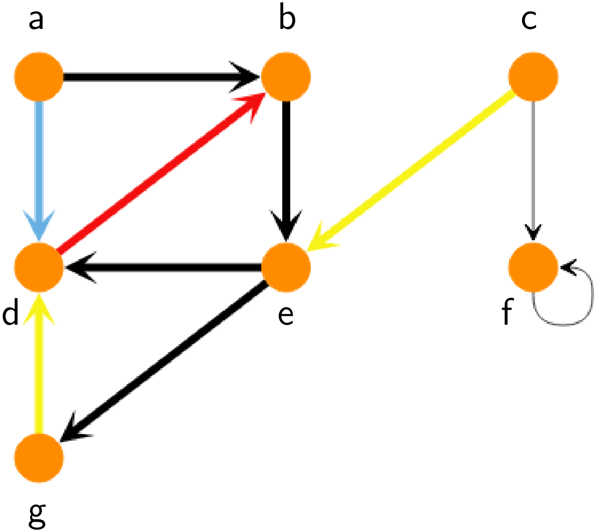
# Example: directed graph



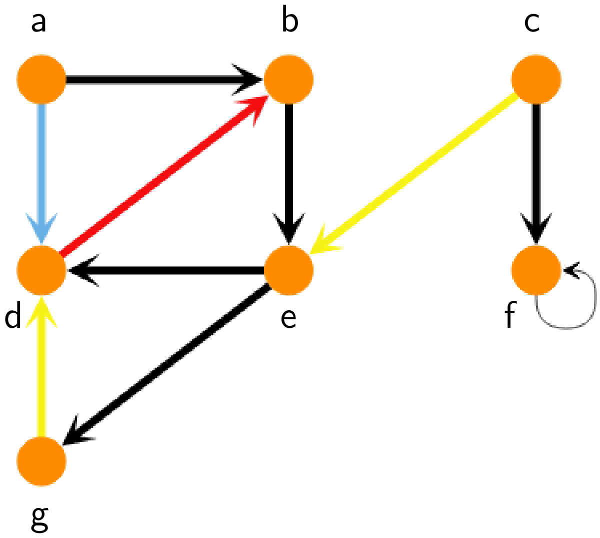
# Example: directed graph



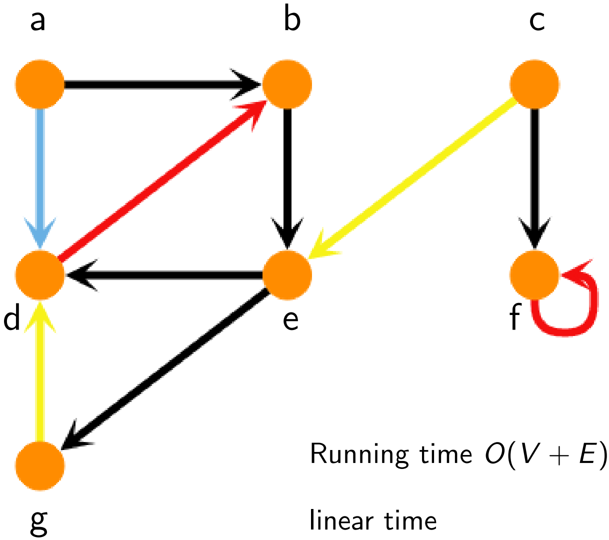
# Example: directed graph



# Example: directed graph



# Example: directed graph




Running time $O(V + E)$

linear time

# Applications

# Edge classification

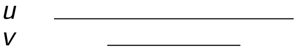
Depends on DFS itself, not just graph

Types of edges:

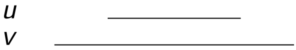
- tree edges
- forward edges
- backward edges
- cross edges

# Algorithm additions

- Starting and ending times
  - useful to classify edges
- forward edges: $u \to v$ with

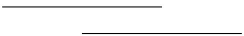



- backward edges: $u \to v$ with




- cross edges: $u \to v$ with


- impossible:



# Undirected graph

- no forward edges
- no cross edges

# Cycle detection

Graph has a cycle $\iff$ DFS has a backward edge

# Topological sorting

Premises:

- Acyclic graphs
- Tasks that depend on one another

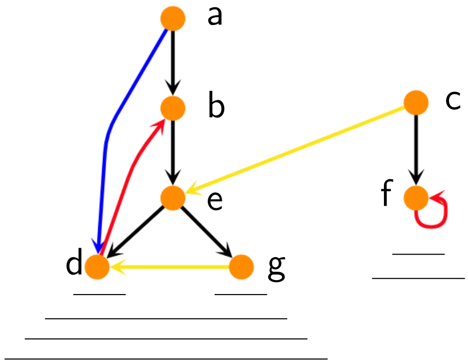
Results:

- Topological sort: Safe order for the tasks
- DFS: reverse order of finishing times

# Cycle detection

DFS has a backward edge $\Longrightarrow$ Graph has a cycle

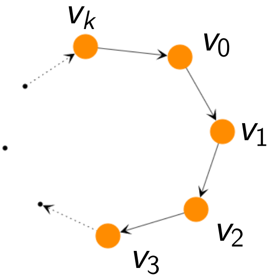
backward edge: $u \to v$ with $u$ / $v$




$u$ starts while $v$ active
$\Longrightarrow$
there is a path from $v$ to $u$

# Cycle detection

Graph has a cycle $\implies$ DFS has a backward edge

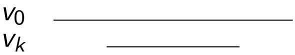



$v_0$: first visited vertex in cycle

$v_1, v_2, v_3, \ldots, v_k$: start after $v_0$

$v_1, v_2, v_3, \ldots, v_k$: start before $v_0$ finishes

Therefore,



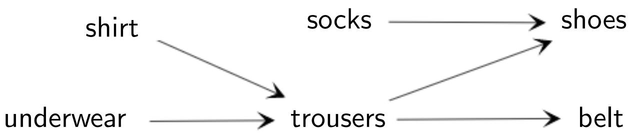

and $v_k \to v_0$ is a backward edge

# Topological sorting

Example: getting dressed

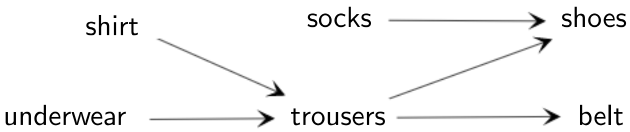
socks
shoes
trousers
belt
shirt
underwear

socks $\rightarrow$ shoes
underwear $\rightarrow$ trousers
shirt $\rightarrow$ trousers
trousers $\rightarrow$ belt
trousers $\rightarrow$ shoes



# Topological sorting

Example: getting dressed




shoes, socks, belt, trousers, underwear, shirt

belt, shoes, trousers, shirt, socks, underwear