WFS marks every vertex reachable from \( s \).

\[
O(V+E)
\]

WFS assigns a parent to every vertex reachable from \( s \) (except \( s \) itself).

\[
\text{DFS (stack)} \quad \text{BFS (queue)} \rightarrow \text{shortest path tree}
\]

\[
\text{``Best-first``: Dijkstra \rightarrow priority queue}
\]

\[
\text{Prims/Sarnik} \rightarrow \text{MST widest path}
\]

---

**Algorithm:** WhateverFirstSearch(s)

1. put \((\emptyset, s)\) in bag
2. while the bag is not empty
   1. take \((p, v)\) from the bag
   2. if \( v \) is unmarked
      1. mark \( v \)
      2. \( v, \text{parent} \leftarrow p \)
      3. for each edge \( vw \)
         1. \( (v, w) \) into the bag

---

\( O(1) \)
Undirected connectivity $O(V+E)$.

Components $O(V+E)$.

Shortest paths:
- Breadth First Search $O(V+E)$
- Dijkstra's Algorithm $O(E \log V)$

**WFSAll(G):**

for all vertices $v$
  unmark $v$

for all vertices $v$
  if $v$ is unmarked
    WhicheverFirstSearch($v$)

**COUNTAndLABEL(G):**

$count \leftarrow 0$

for all vertices $v$
  unmark $v$

for all vertices $v$
  if $v$ is unmarked
    $count \leftarrow count + 1$
    LABELONE($v$, count)

return $count$

**LABELONE(v, count):**

while the bag is not empty
  take $v$ from the bag
  if $v$ is unmarked
    mark $v$
    $v.comp \leftarrow count$
  for each edge $vw$
    put $w$ into the bag
Directed graphs

```
WhateverFirstSearch(s):
put s into the bag
while the bag is not empty
    take v from the bag
    if v is unmarked
        mark v
        for each edge v→w
            put w into the bag

reachability
O(V+E)

strong connectivity
O(V+E)

Entire graph strongly connected? O(V+E)

Acyclic? DAG? O(V+E) DFS

strong components O(V+E) DFS
```

```
DFS(v):
mark v
preVisit(v)
for each edge vw
    if w is unmarked
        parent(w) ← v
        DFS(w)
postVisit(v)
```

DFSALL(G):

```
Preprocess(G)
for all vertices v
    unmark v
for all vertices v
    if v is unmarked
        DFS(v)
```

WFS builds a spanning tree of vertices reachable from s

v can reach u
"v is reachable from u"

O(V+E)

count+ + v.pre+ count

count+ + v.post+ count

count ← 0
**DFSALL(G):**
\( \text{clock} \leftarrow 0 \)
for all vertices \( v \)
unmark \( v \)
for all vertices \( v \)
if \( v \) is unmarked
\( \text{clock} \leftarrow \text{DFS}(v, \text{clock}) \)

**DFS(v,clock):**
mark \( v \)
\( \text{clock} \leftarrow \text{clock} + 1; \; v.pre \leftarrow \text{clock} \)
for each edge \( v \rightarrow w \)
if \( w \) is unmarked
\( w.parent \leftarrow v \)
\( \text{clock} \leftarrow \text{DFS}(w, \text{clock}) \)
\( \text{clock} \leftarrow \text{clock} + 1; \; v.post \leftarrow \text{clock} \)
return \( \text{clock} \)

Pre: abfghdlokpeinjm
Post: dkoaplhcgfba mjni

**Lemma:** \( G \) has a cycle iff
For some edge \( u \rightarrow w \)
we have \( u.post < w.post \)
**TopologicalSort(G):**

\[
\text{clock} \leftarrow V \\
\text{for all vertices } v \text{ in postorder} \\
S[\text{clock}] \leftarrow v \\
\text{clock} \leftarrow \text{clock} - 1 \\
\text{return } S[1..V]
\]