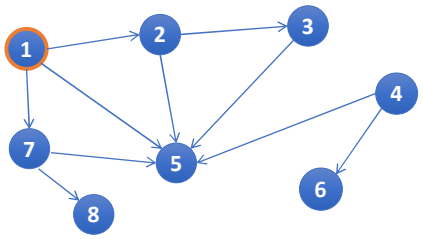
DFS Pseudocode

```
Mark start as discovered

For each successor of start:
    If successor is undiscovered
        Visit successor (recursive call)
Mark start as settled
```

- Exhaustively search starting from one neighbor before moving on to next neighbor



1

Discovery, visitation, settlement

- A vertex is discovered when it is first encountered as a neighbor of another vertex
- A vertex is being visited while its neighbors are iterated over
  - May visit other vertices during a visit
- A vertex is settled after it has been visited
  - Implies all neighbors have been discovered

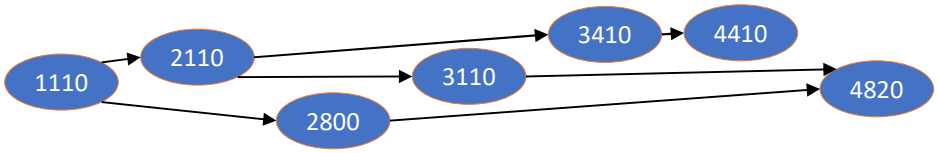
- In recursive DFS, nodes are discovered and visited in the same order, but settlement order is different
  - May not settle until neighbors have been visited
  - Settlement order is *not* reverse visitation order

2

Topological order

A **topological order** of directed graph G is an ordering of its nodes as v_1, v_2, …, v_n, such that for every edge (v_i, v_j), it holds that i < j.

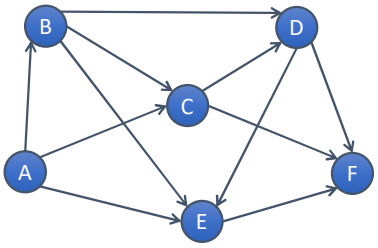
Intuition: line up the nodes with all edges pointing left to right.



Other applications: robot planning, job scheduling, compilers

3

Approach 2: DFS insight



Mark start as discovered

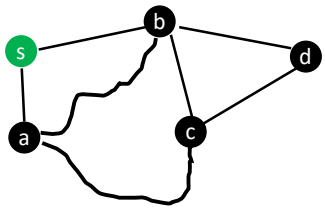
For each successor of start:
  If successor is undiscovered
    Visit successor (recursive call)
  Else if successor is not settled
    Cycle detected!
Mark start as settled
Prepend to topological order

4

Quick Warmup (Unweighted shortest paths)

Fill in the frontier queue and the vertex layers starting at s.

This time, also store the predecessor (the node that you discovered the node from).



Frontier queue				
s				

Vertex state

Vertex	s	a	b	c	d
Layer	0				
Pred					

5

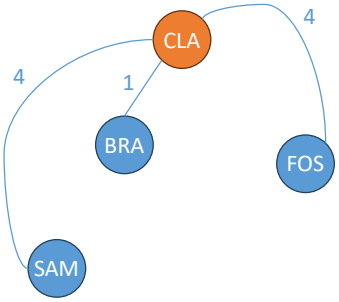
Weighted shortest paths example

Best known path

Vertex	Distance

Best possible path

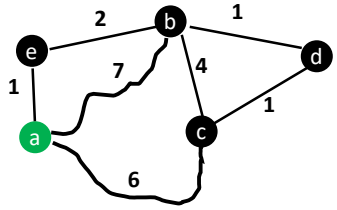
Vertex	Distance
CLA	0



6

When do we know that a path is the shortest possible?

- Shortest path to any undiscovered vertex must be longer than the *smallest* candidate path to a vertex in the frontier
- Therefore, the smallest candidate path in the frontier must be the shortest possible path to that vertex
  - Any new path discovered in the future must go through a vertex currently in the frontier and would therefore be longer



Frontier					
~~a~~					

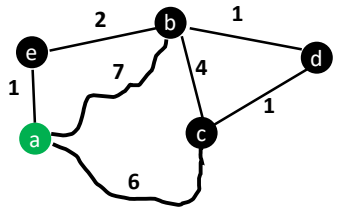
Vertex state

Vertex	a	b	c	d	e
Dist	0				
Pred	null				

7

Exercise: Tracing Dijkstra's algorithm

```
start.discovered = true;
start.dist = 0;
frontier.add(start, start.dist);
while (!frontier.isEmpty()) {
  Vertex v = frontier.removeMin();
  for (Edge e : v.outgoing) {
    Vertex neighbor = e.to;
    double dist = v.dist + e.weight;
    if (!neighbor.discovered || dist < neighbor.dist) {
      neighbor.discovered = true;
      neighbor.dist = dist;
      frontier.addOrUpdate(neighbor, dist);
    }
  }
}
```



Frontier (priority queue)					
a					

Vertex state

Vertex	a	b	c	d	e
Dist	0				
Pred	null				

8