[320] Search Order and Queue Structures

Yiyin Shen

Tracing DFS

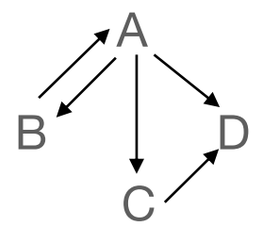
```
def dfs_search(self, dst):
    self.graph.visited.add(self)

    if self == dst:
        return (self.name,)

    for child in self.children:
        if not child in self.graph.visited:
            childpath = child. dfs_search(dst)
            if childpath:
                return (self.name,) + childpath

    return None
```

A.find(D)



Tracing BFS

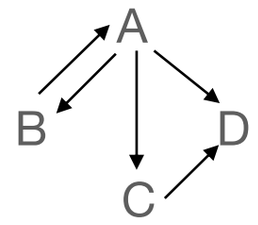
```
def bfs_search(self, dst):
        to_visit = deque([self])
        added = {self}

        while len(to_visit) > 0:
            curr_node = to_visit.popleft()
            if curr_node == dst:
                return self.backtrace(dst)

            for child in curr_node.children:
                if child not in added:
                    to_visit.append(child)
                    added.add(child)
                    child.finder = curr_node

    return None
```

A.find(D)



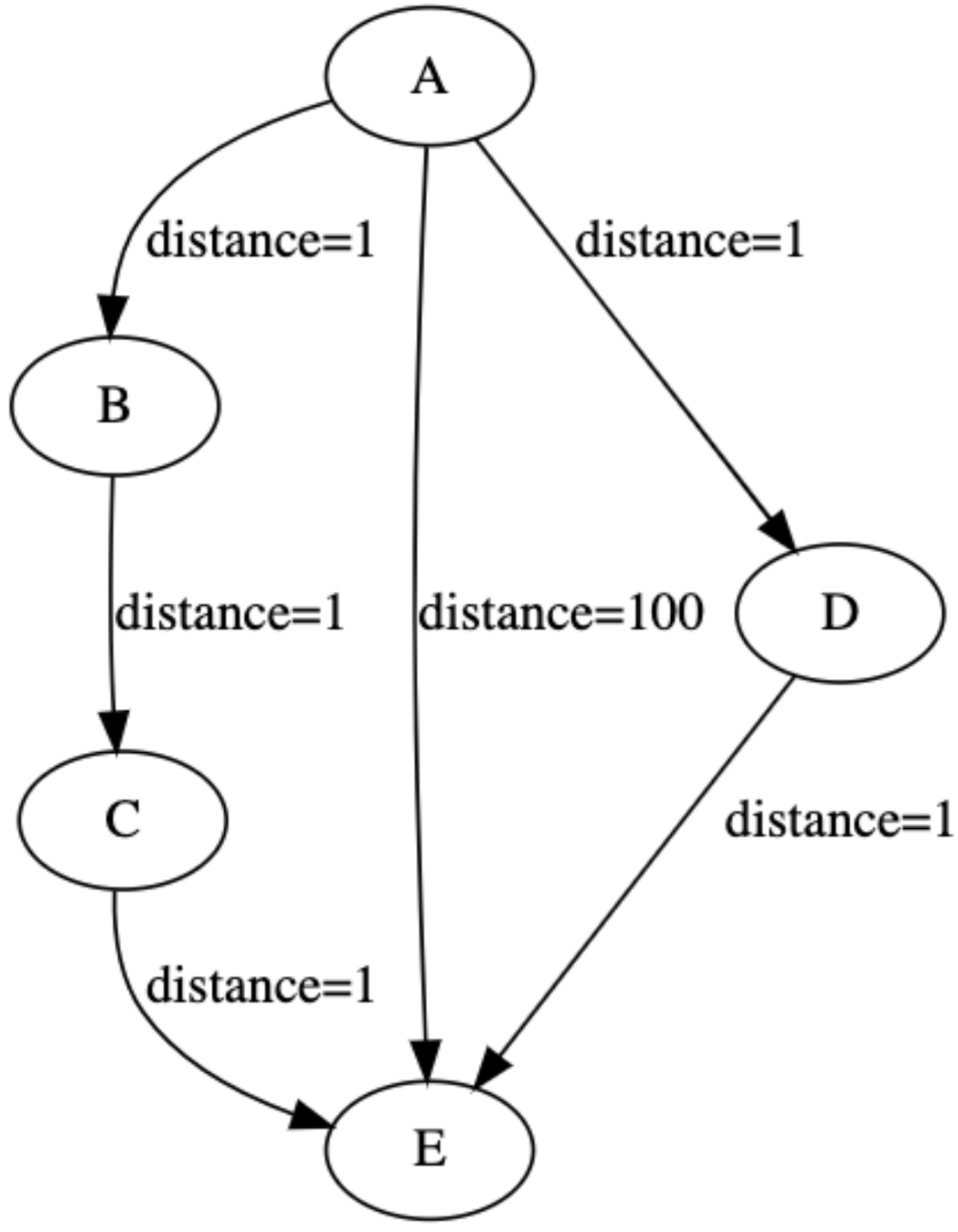
Queueing Structures

Shortest Weighted Path

What path will DFS choose?

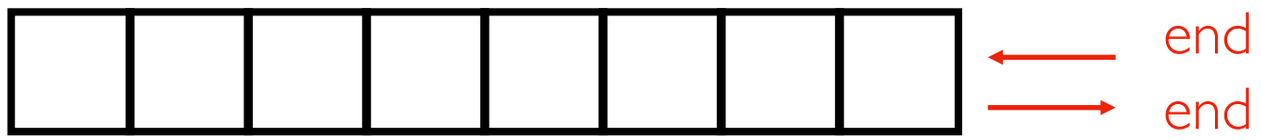
What path will BFS choose?

What path would you choose?

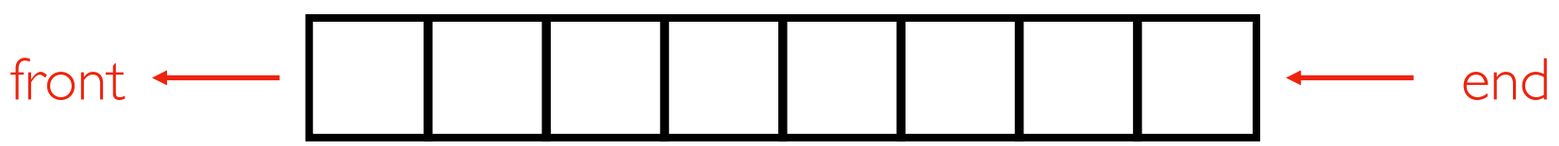


Work queues ("to do" list)

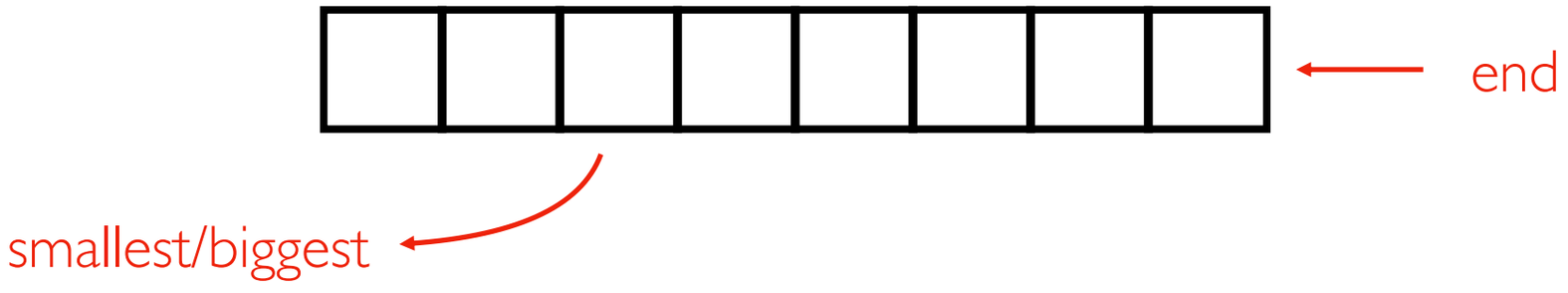
Stack
(FILO)



Queue
(FIFO)

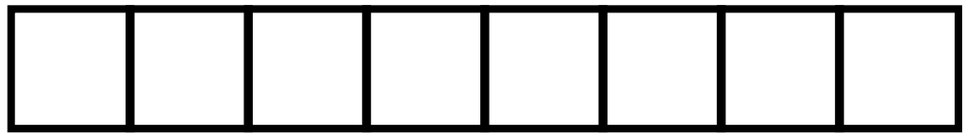


Priority Queue



Work queues ("to do" list)

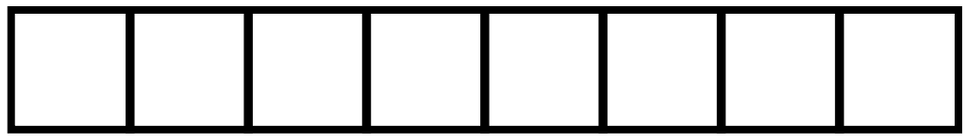
Stack
(FILO)




```
L.append(x)
```

```
x = L.pop(-1)
```

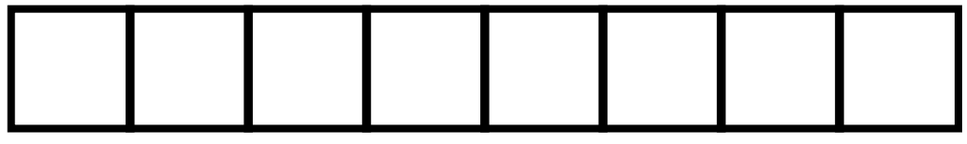
Queue
(FIFO)




```
x = L.pop(0)
```

```
L.append(x)
```

Priority Queue



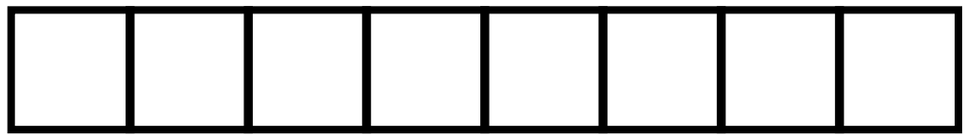

```
L.append(x)
```

```
L.sort()
x = L.pop(-1)
```

what operations are slow?

Work queues ("to do" list)

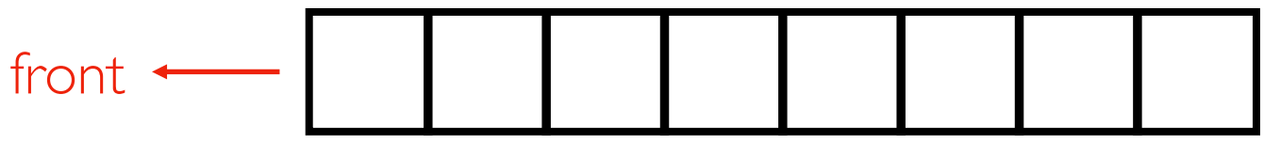
Stack
(FILO)




```
L.append(x)
```

```
x = L.pop(-1)
```

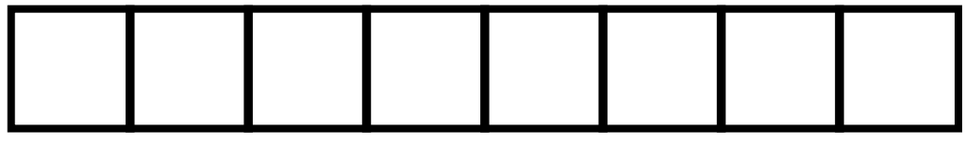
Queue
(FIFO)



❗
```
x = L.pop(0)
```

```
L.append(x)
```

Priority Queue




```
L.append(x)
```

❗
```
L.sort()
x = L.pop(-1)
```

what operations are slow?