Graphs

this is a flow chart, but it is similar to a graph

Snarf today’s code

Today

• Our last topic
  • Intro to graphs
  • Coding with graphs
Graphs

- set of vertices
  - \{1, 2, 3, 4, 5, 6\}
- set of edges
  - \{(1, 2), (1, 4), (2, 5), (5, 3), (5, 6)\}

\\

Graphs

- directed graphs* – edge sets are ordered
  - \{(1, 2), (1, 4), (2, 5), (5, 3), (5, 6)\}
  - 1 points to 2 – notice the arrow
  - (2, 1) is not an edge

*a.k.a. digraphs
Graphs

• undirected graphs – edge sets are not ordered
  • \{(1, 2), (1, 4), (2, 5), (5, 3), (5, 6)\}
  • (1, 2) is the same as (2, 1)

Graphs

• edges can have weights
Graphs

• Why do you care?

Facebook

• Kevin Bacon
Graphs

- Traveling salesperson problem
  - Given a list of cities and the distance between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the original city?

```
BRUTE-FORCE SOLUTION: O(n!)

DYNAMIC PROGRAMMING ALGORITHMS: O(n^2!

SELLING ON EBAY: O(1)
```

Graphs

- Depth-first-search
  - explore as far as possible before backtracking

Start at root

dfs(vertex)
  if(visited vertex) return;
  visit vertex
  for(adjacent vertices to vertex)
    dfs(adjacent vertex)
Graphs

• Depth-first-search
  • explore as far as possible before backtracking

Start at root

dfs(vertex)
  if(visited vertex) return;
  visit vertex
  for(adjacent vertices to vertex)
    dfs(adjacent vertex)
Graphs

• Depth-first-search
  • explore as far as possible before backtracking

Start at root

dfs(vertex)
  if(visited vertex) return;
  visit vertex
  for(adjacent vertices to vertex)
    dfs(adjacent vertex)

A B D

Graphs

• Depth-first-search
  • explore as far as possible before backtracking

Start at root

dfs(vertex)
  if(visited vertex) return;
  visit vertex
  for(adjacent vertices to vertex)
    dfs(adjacent vertex)

A B D F
**Graphs**

- Depth-first-search
  - explore as far as possible before backtracking

Start at root

dfs(vertex)
  if(visited vertex) return;
  visit vertex
  for(adjacent vertices to vertex)
    dfs(adjacent vertex)

A B D F E
Graphs

• Depth-first-search
  • explore as far as possible before backtracking

Start at root

dfs(vertex)
  if(visited vertex) return;
  visit vertex
  for(adjacent vertices to vertex)
    dfs(adjacent vertex)

A B D F E C G

Graphs

• Breadth-first-search
  • explore as far as possible before backtracking

Start at root

bfs(vertex)
  myQ.enqueue(vertex)

  while(!myQ.isEmpty())
    v = myQ.dequeue
    for(adj vertices of v)
      if(adj not visited)
        myQ.enqueue(adj)
Graphs

- Breadth-first-search
  - explore as far as possible before backtracking

Start at root

```java
bfs(root)
  myQ.enqueue(root)
  
  while(!myQ.isEmpty())
    v = myQ.dequeue
    for(adj vertices of v)
      if(adj not visited)
        myQ.enqueue(adj)
```

A B C E D F G

```java
bfs(root)
  myQ.enqueue(root)
  
  while(!myQ.isEmpty())
    v = myQ.dequeue
    for(adj vertices of v)
      if(adj not visited)
        myQ.enqueue(adj)
```

A B C E
Graphs

• Breadth-first-search
  • explore as far as possible before backtracking

Start at root

```java
bfs(root)
  myQ.enqueue(root)

while(!myQ.isEmpty())
  v = myQ.dequeue
  for(adj vertices of v)
    if(adj not visited)
      myQ.enqueue(adj)
```

A B C E D F G

Graphs

• Breadth-first-search
  • explore as far as possible before backtracking

Start at root

```java
bfs(root)
  myQ.enqueue(root)

while(!myQ.isEmpty())
  v = myQ.dequeue
  for(adj vertices of v)
    if(adj not visited)
      myQ.enqueue(adj)
```

A B C E D F G
**Code time**

- snarf today’s code
  - this will be helpful for the graph APTs in set 6

---

**Exam**

- Pick up your exam
- Check that we summed your grade correctly
- Check that your grade is correct in Sakai

- If there is a problem on your exam:
  - Write the problem and section # on the front page
  - Hand your exam back BEFORE leaving class
    - We will not regrade exams that have left this room