6331 - Algorithms, Spring 2014, CSE, OSU
Elementary graph algorithms

Instructor: Anastasios Sidiropoulos
Many problems can be phrased as graph problems.
Many problems can be phrased as *graph problems*.

Input: Graph $G = (V, E)$. 
Graph problems

- Many problems can be phrased as *graph problems*.
- Input: Graph $G = (V, E)$.
- The running time is measured in terms of $|V|$, and $|E|$.
Representing a graph

Adjacency-matrix for a graph $G = (V, E)$.

$|V| \times |V|$ matrix $A = (a_{ij})$, where

$$a_{ij} = \begin{cases} 1 & \text{if } \{i, j\} \in E \\ 0 & \text{if } \{i, j\} \notin E \end{cases}$$

Storage space $= \Theta(|V|^2)$. 
Representing a graph

Adjacency-matrix for a graph $G = (V, E)$.

$|V| \times |V|$ matrix $A = (a_{ij})$, where

$$a_{ij} = \begin{cases} 
1 & \text{if } \{i, j\} \in E \\
0 & \text{if } \{i, j\} \notin E
\end{cases}$$

Storage space $= \Theta(|V|^2)$. 
Representing a graph

The adjacency-list for a graph \( G = (V, E) \) is an array \( Adj \) of size \( |V| \).
Representing a graph

The adjacency-list for a graph $G = (V, E)$ is an array $Adj$ of size $|V|$. For each $u \in V$, $Adj[u]$ is a list that contains all $v \in V$, with $\{u, v\} \in E$. 

Storage space $= \Theta(|V| + |E|)$. Much smaller space when $|E| \ll |V|^2$. 
Representing a graph

The adjacency-list for a graph $G = (V, E)$ is an array $Adj$ of size $|V|$.

For each $u \in V$, $Adj[u]$ is a list that contains all $v \in V$, with $\{u, v\} \in E$.

Storage space $= \Theta(|V| + |E|)$. 
Representing a graph

The adjacency-list for a graph $G = (V, E)$ is an array $Adj$ of size $|V|$. For each $u \in V$, $Adj[u]$ is a list that contains all $v \in V$, with $\{u, v\} \in E$.

Storage space = $\Theta(|V| + |E|)$.

Much smaller space when $|E| \ll |V|^2$. 
Breadth-first search

An algorithm for “exploring” a graph, starting from the given vertex $s$. 

Breadth-first search

BFS($G, s$)

for each $u \in G.V - \{s\}$
  
  $u.color = WHITE$
  
  $u.d = \infty$
  
  $u.\pi = NIL$

$s.color = GRAY$
$s.d = 0$

$s.\pi = NIL$

$Q = \emptyset$

ENQUEUE($Q, s$) //FIFO queue

while $Q \neq \emptyset$

  $u = DEQUEUE(Q)$

  for each $v \in G.Adj[u]$
    
    if $v.color = WHITE$
      
      $v.color = GRAY$
      
      $v.d = u.d + 1$
      
      $v.\pi = u$

      ENQUEUE($Q, v$)

  $u.color = BLACK$
Running time of BFS

- How many DEQUEUE operations?
Running time of BFS

- How many DEQUEUE operations? A non-white vertex never becomes white.
Running time of BFS

- How many DEQUEUE operations? A non-white vertex never becomes white. Every vertex is enqueued at most once.
Running time of BFS

- How many DEQUEUE operations? A non-white vertex never becomes white. Every vertex is enqueued at most once. At most $O(|V|)$ DEQUEUE operations.
Running time of BFS

- How many DEQUEUE operations? A non-white vertex never becomes white. Every vertex is enqueued at most once. At most $O(|V|)$ DEQUEUE operations.
- For every dequeued vertex $u$, we spend $O(|G.Adj[u]|)$ time.
Running time of BFS

- How many DEQUEUE operations? A non-white vertex never becomes white. Every vertex is enqueued at most once. At most $O(|V|)$ DEQUEUE operations.

- For every dequeued vertex $u$, we spend $O(|G.Adj[u]|)$ time. Total length of all adjacency-lists is $O(|E|)$. 
Running time of BFS

- How many DEQUEUE operations? A non-white vertex never becomes white. Every vertex is enqueued at most once. At most $O(|V|)$ DEQUEUE operations.
- For every dequeued vertex $u$, we spend $O(|G.\text{Adj}[u]|)$ time. Total length of all adjacency-lists is $O(|E|)$.
- Total running time $O(|V| + |E|)$. 
Shortest paths

For $u, v \in V$, let $\delta(u, v)$ be the minimum number of edges in a path between $u$ and $v$ in $G$, and $\infty$ if no such path exists.
For $u, v \in V$, let $\delta(u, v)$ be the minimum number of edges in a path between $u$ and $v$ in $G$, and $\infty$ if no such path exists.

I.e., $\delta(u, v)$ is the **shortest path distance** between $u$ and $v$ in $G$. 
Shortest paths

For $u, v \in V$, let $\delta(u, v)$ be the minimum number of edges in a path between $u$ and $v$ in $G$, and $\infty$ if no such path exists.

I.e., $\delta(u, v)$ is the shortest path distance between $u$ and $v$ in $G$.

A path between $u$ and $v$ in $G$ of length $\delta(u, v)$ is called a shortest-path.
Analysis of BFS

Lemma

For any \{u, v\} \in E, we have

$$\delta(s, v) \leq \delta(s, u) + 1.$$
Analysis of BFS

Lemma

For any \( \{u, v\} \in E \), we have

\[
\delta(s, v) \leq \delta(s, u) + 1.
\]

Why?
Lemma

*After the termination of BFS, for each* \( v \in V \), *we have*

\[ v.d \geq \delta(s,v). \]
Analysis of BFS

Lemma

*After the termination of BFS, for each* $v \in V$, we have

$$v.d \geq \delta(s, v).$$

Proof.

Induction on the number of ENQUEUE operations.
Analysis of BFS

Lemma

*After the termination of BFS, for each* \( v \in V \), *we have*

\[
v \cdot d \geq \delta(s, v).
\]

Proof.

*Induction on the number of ENQUEUE operations.*

*Inductive hypothesis:* For all \( v \in V \), we have \( v \cdot d \geq \delta(s, v) \).
Analysis of BFS

Lemma
After the termination of BFS, for each \( v \in V \), we have

\[ v.d \geq \delta(s, v). \]

Proof.
Induction on the number of ENQUEUE operations.
Inductive hypothesis: For all \( v \in V \), we have \( v.d \geq \delta(s, v) \).
Basis of the induction: \( s.d = 0 \), and \( v.d = \infty \) for all \( v \neq s \).
Analysis of BFS

Lemma
After the termination of BFS, for each \( v \in V \), we have

\[ v.d \geq \delta(s, v). \]

Proof.
Induction on the number of ENQUEUE operations.
Inductive hypothesis: For all \( v \in V \), we have \( v.d \geq \delta(s, v) \).
Basis of the induction: \( s.d = 0 \), and \( v.d = \infty \) for all \( v \neq s \).
Consider some \( v \in G.\text{Adj}[u] \), immediately after dequeueing \( u \).
Analysis of BFS

Lemma
After the termination of BFS, for each $v \in V$, we have

$$v.d \geq \delta(s, v).$$

Proof.
Induction on the number of ENQUEUE operations.
Inductive hypothesis: For all $v \in V$, we have $v.d \geq \delta(s, v)$.
Basis of the induction: $s.d = 0$, and $v.d = \infty$ for all $v \neq s$.
Consider some $v \in G.\text{Adj}[u]$, immediately after dequeueing $u$.

$$v.d = u.d + 1$$

$$\geq \delta(s, u) + 1$$

$$\geq \delta(s, v) \quad \text{(by the previous Lemma)}$$
Analysis of BFS

Lemma
Suppose during the execution, \( Q = (v_1, \ldots, v_r) \), where \( v_1 = \text{head} \), \( v_r = \text{tail} \). Then for all \( i \in \{1, \ldots, r - 1\} \)

\[ v_i.d \leq v_{i+1}.d, \]

and

\[ v_r.d \leq v_1.d + 1. \]
Lemma

Suppose during the execution, $Q = (v_1, \ldots, v_r)$, where $v_1 = \text{head}$, $v_r = \text{tail}$. Then for all $i \in \{1, \ldots, r - 1\}$

$$v_i.d \leq v_{i+1}.d,$$

and

$$v_r.d \leq v_1.d + 1.$$

Why?
Analysis of BFS

Lemma

Suppose during the execution, both \( v_i \) and \( v_j \) are enqueued, and \( v_i \) is enqueued before \( v_j \). Then, \( v_i.d \leq v_j.d \) when \( v_j \) is enqueued.

Why?
Analysis of BFS

Lemma

Suppose during the execution, both $v_i$ and $v_j$ are enqueued, and $v_i$ is enqueued before $v_j$. Then, $v_i \cdot d \leq v_j \cdot d$ when $v_j$ is enqueued.

Why?
Analysis of BFS

Theorem

After termination, for all $v \in V$, we have

$$v.d = \delta(s, v).$$

Moreover, for any $v$ that is reachable from $s$, there exists a shortest path from $s$ to $v$ that consists of a shortest path from $s$ to $v.\pi$, followed by the edge $\{v.\pi, v\}$. 
Proof sketch

Suppose for the purpose of contradiction that there exists $v$ with $v.d \neq \delta(s,v)$. 
Proof sketch

Suppose for the purpose of contradiction that there exists \( v \) with \( v.d \neq \delta(s, v) \).

Pick such a \( v \) so that \( \delta(s, v) \) is minimized.

If \( v \) is WHITE, then \( v.d = u.d + 1 \), a contradiction.

If \( v \) is BLACK, then it is already dequeued, so by the above Lemma \( v.d \leq u.d \), a contradiction.

If \( v \) is GRAY, then it was painted GRAY after dequeuing some vertex \( w \), so \( v.d = w.d + 1 \leq u.d + 1 \), a contradiction.
Proof sketch

Suppose for the purpose of contradiction that there exists $v$ with $v.d \neq \delta(s, v)$. Pick such a $v$ so that $\delta(s, v)$ is minimized. By the above Lemma, $v.d > \delta(s, v)$. 
Proof sketch

Suppose for the purpose of contradiction that there exists $v$ with $v.d \neq \delta(s, v)$.
Pick such a $v$ so that $\delta(s, v)$ is minimized.
By the above Lemma, $v.d > \delta(s, v)$.
Let $u$ be the vertex preceding $v$ in a shortest path from $s$ to $v$. We have
\[
v.d > \delta(s, v) = \delta(s, u) + 1 = u.d + 1.
\]
Proof sketch

Suppose for the purpose of contradiction that there exists $v$ with $v.d \neq \delta(s, v)$.
Pick such a $v$ so that $\delta(s, v)$ is minimized.
By the above Lemma, $v.d > \delta(s, v)$.
Let $u$ be the vertex preceding $v$ in a shortest path from $s$ to $v$. We have
\[ v.d > \delta(s, v) = \delta(s, u) + 1 = u.d + 1. \]
Consider the time immediately after dequeueing $u$. 
Proof sketch

Suppose for the purpose of contradiction that there exists \( v \) with \( v.d \neq \delta(s, v) \).
Pick such a \( v \) so that \( \delta(s, v) \) is minimized.
By the above Lemma, \( v.d > \delta(s, v) \).
Let \( u \) be the vertex preceding \( v \) in a shortest path from \( s \) to \( v \). We have
\[
v.d > \delta(s, v) = \delta(s, u) + 1 = u.d + 1.
\]
Consider the time immediately after dequeueing \( u \).
- If \( v \) is WHITE, then \( v.d = u.d + 1 \), a contradiction.
Proof sketch

Suppose for the purpose of contradiction that there exists $v$ with $v.d \neq \delta(s, v)$. Pick such a $v$ so that $\delta(s, v)$ is minimized. By the above Lemma, $v.d > \delta(s, v)$. Let $u$ be the vertex preceding $v$ in a shortest path from $s$ to $v$. We have

$$v.d > \delta(s, v) = \delta(s, u) + 1 = u.d + 1.$$ 

Consider the time immediately after dequeueing $u$.

- If $v$ is WHITE, then $v.d = u.d + 1$, a contradiction.
- If $v$ is BLACK, then it is already dequeued, so by the above Lemma $v.d \leq u.d$, a contradiction.
Proof sketch

Suppose for the purpose of contradiction that there exists \( v \) with \( v.d \neq \delta(s, v) \). Pick such a \( v \) so that \( \delta(s, v) \) is minimized. By the above Lemma, \( v.d > \delta(s, v) \).

Let \( u \) be the vertex preceding \( v \) in a shortest path from \( s \) to \( v \). We have

\[
v.d > \delta(s, v) = \delta(s, u) + 1 = u.d + 1.
\]

Consider the time immediately after dequeueing \( u \).

- If \( v \) is WHITE, then \( v.d = u.d + 1 \), a contradiction.
- If \( v \) is BLACK, then it is already dequeued, so by the above Lemma \( v.d \leq u.d \), a contradiction.
- If \( v \) is GRAY, then it was painted GRAY after dequeueing some vertex \( w \), so \( v.d = w.d + 1 \leq u.d + 1 \), a contradiction.
Proof sketch (cont.)

So, $v.d = \delta(s, v)$ for all $v \in V$. 
Proof sketch (cont.)

So, \( v \cdot d = \delta(s, v) \) for all \( v \in V \).

For the last part of the theorem, if \( u = v \cdot \pi \), then \( v \cdot d = u \cdot d + 1 \). The assertion follows by induction.
Breadth-first trees

We define the **predecessor graph** as \( G_\pi = (V_\pi, E_\pi) \), where

\[
V_\pi = \{ v \in V : v.\pi \neq NIL \} \cup \{ s \}
\]

\[
E_\pi = \{ (v.\pi, v) : v \in V_s \setminus \{ s \} \}
\]

\( G_\pi \) is a breadth-first tree if \( V_\pi \) consists of the vertices reachable from \( s \) and for all \( v \in V_\pi \), \( G_\pi \) contains a unique simple path from \( s \) to \( v \) that is also a shortest path from \( s \) to \( v \) in \( G \).
Breadth-first trees

We define the **predecessor graph** as $G_\pi = (V_\pi, E_\pi)$, where

$$V_\pi = \{ v \in V : v.\pi \neq NIL \} \cup \{ s \}$$

$$E_\pi = \{ (v.\pi, v) : v \in V_s \setminus \{ s \} \}$$

$G_\pi$ is a **breadth-first tree** if $V_\pi$ consists of the vertices reachable from $s$ and for all $v \in V_\pi$, $G_\pi$ contains a unique simple path from $s$ to $v$ that is also a shortest path from $s$ to $v$ in $G$. 
Breadth-first trees

We define the **predecessor graph** as $G_{\pi} = (V_{\pi}, E_{\pi})$, where

$$V_{\pi} = \{v \in V : v.\pi \neq NIL\} \cup \{s\}$$

$$E_{\pi} = \{(v.\pi, v) : v \in V_s \setminus \{s\}\}$$

$G_{\pi}$ is a **breadth-first tree** if $V_{\pi}$ consists of the vertices reachable from $s$ and for all $v \in V_{\pi}$, $G_{\pi}$ contains a unique simple path from $s$ to $v$ that is also a shortest path from $s$ to $v$ in $G$.

**Lemma**

*After the execution of BFS, the predecessor graph $G_{\pi}$ is a breadth-first tree.*