Graph Traversal

http://www.cs.unh.edu/~ruml/cs758
Graph Traversal
directed, arc/edge, weighted, labeled. drawings
representation
relations $\rightarrow$ edges
cycle, DAG, tree, planar
Breadth-first Search

Graph Traversal

traversal: Graph → forest
The Algorithm

Graph Traversal

1. foreach vertex, label it undiscovered and \( v.d \) \(\leftarrow\) \(\infty\)
2. start's label \(\leftarrow\) discovered, \( d \) \(\leftarrow\) 0, \( \pi \) \(\leftarrow\) nil
3. \(\{\text{start}\}\) \(\leftarrow\) \(\emptyset\)
4. while \(\emptyset\) not empty
   4.1. label \( u \) finished
   5. \(\emptyset\) enqueue \( u \)
   6. foreach neighbor \( v \) of \( u \)
   7. if \( v \) is undiscovered
   8. label \( v \) discovered, \( v.d \) \(\leftarrow\) \( u.d + 1\), \( v.\pi \) \(\leftarrow\) \( u\)
   9. enqueue \( v \) in \(\emptyset\)
10. label \( u \) finished

What's the time complexity?
Do we really need all the labels?
Which vertices does \(\emptyset\) hold (at line 4)?

\[ n \rightarrow \nu, \quad v.\nu + p.n \rightarrow p.\nu \]

Full Algorithm

Graph Traversal

Wheeler Ruml (UNH) Class 13, CS 758 - 5 / 12
Class 13, CS 758 – 6 / 12

The Full Algorithm

1. foreach vertex, label it undiscovered and
   \( v.d \leftarrow \infty \)
2. foreach vertex
3. if \( s \) is undiscovered
4. label \( s \) discovered, \( s.d \leftarrow 0, s.\pi \leftarrow \text{null} \)
5. \( \{s\} \rightarrow Q \)
6. while \( Q \) not empty
7. \( n \rightarrow \mathcal{O} \)
8. foreach neighbor \( v \) of \( n \)
9. if \( v \) is undiscovered
10. label \( v \) discovered, \( v.d \leftarrow u.d + 1, v.\pi \leftarrow u \)
11. enqueue \( v \) in \( Q \)
12. label \( u \) finished

What’s the time complexity?
Join ACM for $19
wildcard vote in one month
midterm
ass 8
ass 7
4. At termination, \( v.d = (n,s)q = p \cdot n \cdot \delta(s,v) \); shortest path length

\[
\text{last.d} \leq \text{removed.d} + 1 = \text{new.d}.
\]

\[
\text{EOLQs: new.d} = \text{removed.d} + 1 > \text{first.d} + 1 \text{ and}
\]

Preserved by dequeue.

Proof: By induction. True when queue is \( s \).

First by at most \( t \).

3. \( p \) values in queue are nondecreasing and last in queue exceeds

\[
(n,s)q \leq I + (n,s)q \leq I + p \cdot n \cdot \delta(s,u) + 1 \geq \delta(s,v).
\]

Proof: By induction over iterations:

\[
(n,s)q \leq p \cdot n \cdot \delta(s,v).
\]

via induction over iterations.

2. \( v.d \) is updated to

Start: \( v.d \) is updated to \( (n,s)q \leq I + p \cdot n \).

Proof: Show \( \forall (n,s)q \leq p \cdot n \).

1. Distances we assign always stay the same or go down.
Claim: at termination, \( v.d = \delta(s, v) \), shortest path length.

*Proof:* Consider \( v \) with minimum incorrect distance, and let \( u \) be predecessor. If \( u \) is already finished, then \( v.d \leq u.d \), contradiction.

If \( u \) is undiscovered, it would then be correct, contradiction.

Contradiction:

\[
\text{Proof:}\quad p.n + 1 + p.m = p.v + p.m = p.w + (n)q < p.w + 1
\]
Depth-first Search

Vs breadth-first?

Discovery and finish times are parenthesized.

What's the time complexity?

DFS

1. forall vertices, label ← undiscovered
2. DFS-visit(start)
   2. DFS-visit(start)
   3. label n discovered
5. if v is undiscovered
4. foreach neighbor v of n
   4. foreach neighbor v of n
   5. if v is undiscovered
   6. v.π ← u
   7. n → • v
6. n → • v
7. DFS-visit(v)
8. label n finished

What's the time complexity?

Discovery and finish times are parenthesized.
When edge is explored, label of arc dest gives type:

- **tree** when edge is explored, label of arc dest gives type
- **forward** connects to ancestor in tree
- **back** in depth-first tree
- **cross** non-tree edge connecting to descendant in tree
- **other** non-ancestors/non-decendants or different DFS
Thanks!

and put it in the box on your way out.

Please write down your most pressing question about algorithms:

What would you like to hear more about?

What question didn’t you get to ask today?

What’s still confusing?

For example: