Lecture 19: Graph implementation & traversals

- Graph demo
- Depth first & breadth-first search

Announcements:
- Visiting lecture today after class, Gates 114 (reception in 122)
- Project 5
explore every vertex reachable from a given vertex.

idea: make as much progress as possible (completely explore one path before backing out and exploring next)

Depth-first search

- find all vertices
- detect cycles
- determine if a graph is connected: every vertex is reachable from every other vertex.

break a graph up into components, many algorithms only work on connected graphs.
Depth First Search (v)

- Visit v
- For each child of v:
  - Depth First Search child
- If visit all vertices reachable from v without visiting any vertex in "seen" set, update seen w/ all vertices.
- If v is in seen:
  - Return no vertices reachable from v without entering seen.
- Visit v, mark v in seen (add v to set)
- For each neighbor u of v:
  - Depth First Search (u, seen)
- Return vertices reachable from v has been

0(|V|)

Space

0(|V| + |E|)

Run time:
- Do some work for each vertex
- Do some constant work for every edge
- Total storage is constant for each stack frame.
- Longest possible path we would consider?
  - Paths never go through same vertex twice.
  - Longest path $O(|V|)$

In fact, $|V| + 1$
DFS iterative version:

we'll maintain a worklist of vertices still need to visit.

\text{dfs}(v):
\begin{enumerate}
\item \text{Visit}(v);
\item for each child \( u \) of \( v \), add \( u \) to worklist.
\item while worklist is empty (everything has been visited)
\item pop v; take something off of worklist, visit it,
\item add all children to worklist.
\end{enumerate}

want the last entry put in to be first that we process.

\text{use a stack!}

push, adds to end of stack
gulp, removes from end of stack.
Breadth-First Search: go as wide as possible

- Maintain a vertex list to be visited
- Remove a vertex
- Visit it
- Add neighbors to WL
- Repeat

if we get a vector

if we don't get a vector

2 steps away

1 step away

3 steps away

Visited

distance 0

distance 1

distance 2

distance 3