Graph Implementation #1: Edge List

Data Structures:
- Vertex Collection:
- Edge Collection:

Operations on an Edge List implementation:
- `insertVertex(K key)`:
  - What needs to be done?
- `removeVertex(Vertex v)`:
  - What needs to be done?
- `incidentEdges(Vertex v)`:
  - What needs to be done?
- `areAdjacent(Vertex v1, Vertex v2)`:
  - Can this be faster than `G.incidentEdges(v1).contains(v2)`?
- `insertEdge(Vertex v1, Vertex v2, K key)`:
  - What needs to be done?

Graph Implementation #2: Adjacency Matrix

Data Structures:

Operations on an Adjacency Matrix implementation:
- `insertVertex(K key)`:
  - What needs to be done?
- `removeVertex(Vertex v)`:
  - What needs to be done?
- `incidentEdges(Vertex v)`:
  - What needs to be done?
- `areAdjacent(Vertex v1, Vertex v2)`:
  - Can this be faster than `G.incidentEdges(v1).contains(v2)`?
- `insertEdge(Vertex v1, Vertex v2, K key)`:
  - What needs to be done?
Graph Implementation #3: Adjacency List

![Graph Diagram]

<table>
<thead>
<tr>
<th>Vertex List</th>
<th>Edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>u</td>
<td>a</td>
</tr>
<tr>
<td>v</td>
<td>b</td>
</tr>
<tr>
<td>w</td>
<td>c</td>
</tr>
<tr>
<td>z</td>
<td>d</td>
</tr>
</tbody>
</table>

Operations on an Adjacency Matrix implementation:

insertVertex(K key):

removeVertex(Vertex v):

incidentEdges(Vertex v):

areAdjacent(Vertex v1, Vertex v2):

insertEdge(Vertex v1, Vertex v2, K key):

Running Times of Classical Graph Implementations

<table>
<thead>
<tr>
<th></th>
<th>Edge List</th>
<th>Adj. Matrix</th>
<th>Adj. List</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space</td>
<td>n + m</td>
<td>n²</td>
<td>n + m</td>
</tr>
<tr>
<td>insertVertex</td>
<td>1</td>
<td>n</td>
<td>1</td>
</tr>
<tr>
<td>removeVertex</td>
<td>m</td>
<td>n</td>
<td>deg(v)</td>
</tr>
<tr>
<td>insertEdge</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>removeEdge</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>incidentEdges</td>
<td>m</td>
<td>n</td>
<td>deg(v)</td>
</tr>
<tr>
<td>areAdjacent</td>
<td>m</td>
<td>1</td>
<td>min( deg(v), deg(w) )</td>
</tr>
</tbody>
</table>

Q: If we consider implementations of simple, connected graphs, what relationship between n and m?

- On connected graphs, is there one algorithm that underperforms the other two implementations?

Q: Is there clearly a single best implementation?

- Optimized for fast construction:

- Optimized for areAdjacent operations:

CS 225 – Things To Be Doing:

1. lab_heap due on Sunday, Nov. 7
2. mp_mosaic ec due Monday Nov. 8
3. Final Project proposal and contract due Monday Nov. 8
4. POTD today!