Algorithms and Data Structures for Data Science

Lab_adjList

CS 277
Brad Solomon

April 21, 2023

Department of Computer Science
Learning Objectives

Implement core graph ADT functions using adjacency lists

Introduce modifications to graphs for storing weights
Graph ADT

Find
- getVertices() — return the list of vertices in a graph
- getEdges(v) — return the list of edges that touch the vertex v
- areAdjacent(u, v) — returns a bool based on if an edge from u to v exists

Insert
- insertVertex(v) — adds a vertex to the graph
- insertEdge(u, v) — adds an edge to the graph

Remove
- removeVertex(v) — removes a vertex from the graph
- removeEdge(u, v) — removes an edge from the graph
Weighted Adjacency List

Vertex Storage:

Edge Storage:
Graph Implementation: Adjacency List

**insertVertex(v):**

```
<table>
<thead>
<tr>
<th>Vertex</th>
<th>Adjacencies</th>
</tr>
</thead>
<tbody>
<tr>
<td>u</td>
<td>v, 1, w, 2</td>
</tr>
<tr>
<td>v</td>
<td>u, 1, w, 1</td>
</tr>
<tr>
<td>w</td>
<td>v, 1, u, 2, z, 3</td>
</tr>
<tr>
<td>z</td>
<td>w, 3</td>
</tr>
</tbody>
</table>
```

```
Graph:
- u
  - 1
  - 2
- v
  - 1
- w
  - 1
  - 3
- z
  - d=1
```

```
```

```
```
Graph Implementation: Adjacency List

```
insertEdge(u, v, w):
```

![Graph Diagram]

- **Adjacency List:**
  - `u` (d=2) → `v, 1` → `w, 2`
  - `v` (d=2) → `u, 1` → `w, 1`
  - `w` (d=3) → `v, 1` → `u, 2` → `z, 3`
  - `z` (d=1) → `w, 3`
Graph Implementation: Adjacency List

removeVertex(v):

removeEdge(u, v):
Graph Implementation: Adjacency List

getAllEdges(w):

```
v, 1  w, 2
u, 1  w, 1
v, 1  u, 2  z, 3
w, 3
```

Legend:
- **u**: d=2
- **v**: d=2
- **w**: d=3
- **z**: d=1