Graphs

<table>
<thead>
<tr>
<th></th>
<th>USD</th>
<th>GBP</th>
<th>CAD</th>
<th>EUR</th>
<th>AUD</th>
</tr>
</thead>
<tbody>
<tr>
<td>USD</td>
<td>1.000</td>
<td>0.557569</td>
<td>1.3048</td>
<td>0.826515</td>
<td>1.41563</td>
</tr>
<tr>
<td>GBP</td>
<td>1.79349</td>
<td>1.000</td>
<td>2.34015</td>
<td>1.48235</td>
<td>2.53893</td>
</tr>
<tr>
<td>CAD</td>
<td>0.7764</td>
<td>0.427321</td>
<td>1.000</td>
<td>0.633441</td>
<td>1.08494</td>
</tr>
<tr>
<td>EUR</td>
<td>1.20989</td>
<td>0.674602</td>
<td>1.57867</td>
<td>1.000</td>
<td>1.71276</td>
</tr>
<tr>
<td>AUD</td>
<td>0.706399</td>
<td>0.393866</td>
<td>0.921709</td>
<td>0.583849</td>
<td>1.000</td>
</tr>
</tbody>
</table>

Set (accessed by final currency)

exchange rates in August 2004

What data structure is best if you just want to look up an exchange rate?

What if you want to determine if arbitrage scheme exists?
Graph Representing Exchange Rates

Vertex

Edge (directed)

edge weight

path cycle
Task Scheduling

Precedence Graph for Changing a Flat Tire

- drive off
  - put lug tool away
    - tighten lugs
      - lower car
    - put lug on
      - remove old tire
      - put on spare
        - remove spare from trunk
        - put lugs on
          - remove jack
          - remove lugs
            - get out lug tool
              - get out owner's manual
              - take out jack
            - jack up car
              - place jack under car
Finding Shortest Travel Routes

- Each airport is a vertex.
- Sample edges (flights).
- #edges = #flights.
- Multigraph - multiple edges between vertices.
and many more

- image segmentation
- minimizing infrastructure cost (e.g., laying optical fiber) to allow travel/communication between a set of locations
- executing a makefile

A lot of problems can be formulated as graph problems
Types of graphs

Unweighted

name of edge

Directed

Undirected
Weighted directed multigraph
How can we represent a graph?

What are basic things we might want to do?

Is there an edge from \( V_i \) to \( V_j \)?

Iterate over all edges from \( V_i \)?

Less often, iterate over all edges to \( V_i \)?
\[ V = \text{set of vertices in graph} \]

For Lab 4

\[ \text{STL} \rightarrow \text{airport object} \]

3-letter acronym for airport

Set of airports
Object "STL"

city name St. Louis

time zone location

List of edges (Flights) that originate in St. Louis

Could also keep a list of edges that terminate in St. Louis

Could use a Set versus a list.

Expected constant time search by dest.

Provide an comparator to constructor

List of outgoing edges
Adjacency List

Representation of a graph where for each vertex you store a list of all outgoing edges adjacent (in a directed sense)
Graph

Typical way to draw

\[
\begin{align*}
\varepsilon_1, \varepsilon_3 \\
\varepsilon_2, \varepsilon_2 \\
\varepsilon_4 \\
\end{align*}
\]

\(a: b, \varepsilon\)
\(b: c, \varepsilon, c\)
\(c: \emptyset, \varepsilon_4\)
\(d: \)

Incoming edges

Also we use

\(\begin{align*}
a: \\
b: a \\
c: a, b \\
d: b, c \\
\end{align*}\)

Implicit representation of an edge when no multi-edges
If we keep a set of outgoing edges for each vertex (with comparison defined by dest) answer is edge from Vi to Vj in expected constant time.

Adjacency Set

Augmented Adjacency Set also keep a set of incoming edges for each vertex
Adjacency Matrix

$O(n^2)$ time to iterate over $n$ vertices and $m$ edges (if not multigraph)
$O(m+n)$ time to iterate over all edges

# objects among all lists is $m$
Next time

Summarize time & space complexities

Talk about rep. undirected graph