Week 12 (Notes)

Graphs

- Terminology and Definitions
- Representations
- Algorithms

Graphs

First use of graphs in 1736 when Euler tried to solve **"The Konigsberg Bridge Problem"**

4 land areas: A, B, C, D

7 bridges: a, b, c, d, e, f, g

Problem: Is there a way to start out from one land area and walk across each of the bridges exactly once, returning to the original land area?

A Graphical Representation of the Map

- Each land area is represented as a vertex
- Each bridge is represented as an edge.

A, B, C, D are **vertices** or **nodes**.

a, b, c, d, e, f, g are **edges** or **arcs**.

(Multigraph)

Terminology and Definitions

Degree:

Definition: A vertex is incident to an edge if it is one of the two vertices in the ordered pair of nodes comprising the edge.

Euler showed that there is a walk starting at any vertex going through each edge exactly once and returning to the same vertex when the degree of each vertex is even. Such a walk is know as an **Eulerian walk**.

Does the graph of Konigsberg have an Eulerian walk?

How about this?

Applications of Graphs

(1) Network analysis

(2) Project planning (PERT diagrams)

(3) Shortest route planning

(4) VLSI

(5) Natural Language processing

.

.

.

Terminology and Definitions

Graph: A Graph, G, consists of two sets, V & E.

- V is a finite, nonempty set of vertices (or nodes).

- E is a set (possibly empty) of pairs of vertices called edges (or arcs).

G = < V , E >, Note: A tree is a special case of G.

Edges are usually represented as an ordered pair (V i , V j).

Note: If (V 1 , V 2) E then V1 V2 and
(V1 , V 2) occurs only once in the set.

Example:

Terminology and Definitions

Undirected Graph: An undirected graph does not distinguish (V1, V2) from (V2, V1)

Directed Graph (Digraph): A directed graph does.

Example:

Terminology and Definitions

Degree of a Digraph:

We need to distinguish edges with Vi as head from edges with V i as tail.

1. **indegree** is the number of edges incident to Vi with Vi as head.

2. **outdegree** is the number of edges incident to Vi with Vi as tail.

Example:

Indegree (A) = 1 Outdegree (A) = 3

Indegree (B) = 2 Outdegree (B) = 1

Indegree (C) = 1 Outdegree (C) = 2

Indegree (D) = 2 Outdegree (D) = 0

Terminology and Definitions

Note: the number of vertices in an undirected graph with odd degree in a graph must always be even. Why?

If d_i is the degree of vertex i in graph G with n vertices and m edges, then

m =

Now add an edge (in green)

Terminology and Definitions

Multigraph: Graph with loops (or self-loops)

Complete Undirected Graph: Each node is connected to every other node. The number of edges *m* are given as:

m =

Examples:

Terminology and Definitions

Subgraph: A subgraph of graph G, G', has the following 2 properties:

(1) V (G')   V (G)

(2) E (G')   E (G).

Path: A path from vertex V_f to vertex V_L in G is a sequence of vertices

such that

are in E(G).

Terminology and Definitions

Path Length: A path of length k from V_f to V_L is a sequence of k + 1 vertices such that the first vertex is V_f and the (k +1)st is V_L.

The length of the path is the number of edges in it. A simple path is a path in which all vertices, except for possibly the first and the last, are distinct.

Example:

- (1,2), (2,4), (4,3) ---> 1, 2, 4, 3
  is simple
- (1,2), (2,4), (4,2) ---> 1, 2, 4, 2
  is not simple.

Cycle: A cycle is a simple path where $V_f = V_L$.

Terminology and Definitions

Connected: An undirected graph is said to be connected if for every pair of distinct vertices, there is a path between them in G.

Examples:

Terminology and Definitions

Strongly Connected: A directed graph, G, is strongly connected if for every pair of vertices, Vi and Vj , in V(G), there is a path from Vi to Vj and from Vj to Vi .

Examples:

(1)

Strongly Connected

(2)

Not Strongly Connected -- no path from 3 to 1

Terminology and Definitions

Definition: A **strongly connected component** is a maximal subgraph which is strongly connected.

Weighted Graph (Network): When numbers are associated with an edge in a graph, the graph is a weighted graph.

General problem: Find a path with smallest sum of weights to meet some set of criteria.

Directed Acyclic Graph (DAG): Directed graph without cycles.

Example: Shortest path problem

Try to find shortest path from Boston to LA for example.

We will explore how to solve this problem, but first we must develop data structures for graphs.

Graph Representations

(1) Adjacency Matrix

(2) Adjacency Lists

(3) Adjacency Multilists

Choice depends upon the application.

1. Adjacency Matrix

If $G = < V , E >$ where V contains n vertices,

$n \geq 1$, the adjacency matrix can be used to represent a graph where

$A[i][j] = 1$ when (V_i , V_j) $E(G)$

and $A[i][j] = 0$ when (V_i , V_j) $E(G)$.

Note: When G is undirected (V_i , V_j) $E(G)$ sets both $A[i][j]$ and $A[j][i]$ to 1.

Example:

Using Adjacency Matrix Representation

Row sum =

Column sum =

Storage Requirement:

- To determine the number of edges or G is connected: All the entries, except the diagonal need to be examined -----> O(???)

2. Adjacency List

Each row of the adjacency matrix is represented as a linked list.

- Number of nodes in the list gives the degree (undirected graph) of a node.

- To determine the number of edges O(n + e) where e is the number of edges. Good if graph is sparse, that is, e << $n^2/2$

Graph Algorithms

- Transitive Closure,
- Shortest Path
- Searches (Depth–First, Breadth–First)
- Spanning Trees
- Topological Sort
- Network Flow

Transitive Closure

A general graphical problem:

- Is there any path between two cities A and B?
- If more than one path from A to B, which is the shortest?

Definition: A (0-1)-matrix which describes whether or not there is a path between two given vertices (i,j).

Method to Compute Transitive Closure

Use logical AND with the adjacency matrix.

Example: Let A be the adjacency matrix.

If A[i][j] **AND** A[j][k] is true
then there is a path (of length 2) between nodes i and k.

Let A_2[i][j] be such a matrix.

Example:

Method to Compute Transitive Closure

Generalize to $A_n[i][j]$.

Transitive Closure: Matrix obtained by **OR**ing all such matrices.

T.C: $A[i][j]$ **OR** $A_2[i][j]$ **OR** $A_3[i][j]$ . . . $A_n[i][j]$.

Complexity: O(???).

An improvement (Warshall's Algorithm):
Complexity: O( ).

Shortest Path Algorithm in Networks
(by Dijkstra)

Example: Single Source All Destinations.

Weight Adjacency Matrix is given.

Paths are generated in a non-decreasing order.

Shortest Path Algorithm in Networks (by Dijkstra)

Let **Vi** be the starting vertex and **S** denote the set of vertices (including **Vi**) to which shortest paths from **Vi** have already been found.

Observations:

1. If the next shortest path is to vertex **U**, then the path from **Vi** to **U** must go through only those vertices which are in **S**.

2. The destination node for which the next path is being generated, must be the chosen among those vertices which are not in **S** in such a way that it has the minimum distance from **Vi** (minimum taken over all the distances of nodes which are not in **S**).

3. If the addition of node **U** of (Step 2) causes the distance between **Vi** and another node **W**, not in **S**, to decrease then there is an edge between **U** and **W**.

Example