Lecture 19: Second Midterm Exam

(Tuesday, April 7, 1998)

Second midterm exam today. No lecture.

Lecture 20: Introduction to Graphs

(Thursday, April 9, 1998)

Read: Sections 5.4, 5.5.

Graph Algorithms: For the next few weeks, we will be discussing a number of various topics. One involves algorithms on *graphs*. Intuitively, a graph is a collection of vertices or nodes, connected by a collection of edges. Graphs are very important discrete structures because they are a very flexible mathematical model for many application problems. Basically, any time you have a set of objects, and there is some "connection" or "relationship" or "interaction" between pairs of objects, a graph is a good way to model this. Examples of graphs in application include *communication* and *transportation networks*, *VLSI* and other sorts of *logic circuits*, *surface meshes* used for shape description in computer-aided design and geographic information systems, *precedence constraints* in scheduling systems. The list of application is almost too long to even consider enumerating it.

Most of the problems in computational graph theory that we will consider arise because they are of importance to one or more of these application areas. Furthermore, many of these problems form the basic building blocks from which more complex algorithms are then built.

Graphs and Digraphs: A *directed graph* (or digraph) $G = (V, E)$ consists of a finite set of *vertices* V (also called *nodes*) and E is a binary relation on V (i.e. a set of *ordered* pairs from V) called the *edges*.

For example, the figure below (left) shows a directed graph. Observe that *self-loops* are allowed by this definition. Some definitions of graphs disallow this. Multiple edges are not permitted (although the edges (v, w) and (w, v) are distinct). This shows the graph $G = (V, E)$ where $V = \{1, 2, 3\}$ and $E = \{(1, 1), (1, 2), (2, 3), (3, 2), (1, 3)\}$.

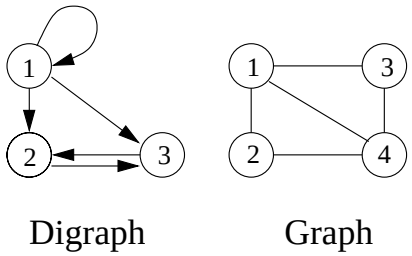

Figure 18: Digraph and graph example.

In an *undirected graph* (or just *graph*) $G = (V, E)$ the edge set consists of unordered pairs of distinct vertices (thus self-loops are not allowed). The figure above (right) shows the graph $G = (V, E)$, where $V = \{1, 2, 3, 4\}$ and the edge set is $E = \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 4\}, \{3, 4\}\}$.

We say that vertex w is *adjacent* to vertex v if there is an edge from v to w. In an undirected graph, we say that an edge is *incident* on a vertex if the vertex is an endpoint of the edge. In a directed graph we will often say that an edge either *leaves* or *enters* a vertex.

A digraph or undirected graph is said to be *weighted* if its edges are labelled with numeric weights. The meaning of the weight is dependent on the application, e.g. distance between vertices or flow capacity through the edge.

Observe that directed graphs and undirected graphs are different (but similar) objects mathematically. Certain notions (such as path) are defined for both, but other notions (such as connectivity) are only defined for one.

In a digraph, the number of edges coming out of a vertex is called the *out-degree* of that vertex, and the number of edges coming in is called the *in-degree*. In an undirected graph we just talk about the *degree* of a vertex, as the number of edges which are *incident* on this vertex. By the *degree* of a graph, we usually mean the maximum degree of its vertices.

In a directed graph, each edge contributes 1 to the in-degree of a vertex and contributes one to the out-degree of each vertex, and thus we have

Observation: For a digraph $G = (V, E)$,

$$\sum_{v \in V} \textit{in-deg}(v) = \sum_{v \in V} \textit{out-deg}(v) = |E|.$$

($|E|$ means the cardinality of the set E, i.e. the number of edges).

In an undirected graph each edge contributes once to the outdegree of two different edges and thus we have

Observation: For an undirected graph $G = (V, E)$,

$$\sum_{v \in V} \textit{deg}(v) = 2|E|.$$

Lecture 21: More on Graphs

(Tuesday, April 14, 1998)

Read: Sections 5.4, 5.5.

Graphs: Last time we introduced the notion of a graph (undirected) and a digraph (directed). We defined vertices, edges, and the notion of degrees of vertices. Today we continue this discussion. Recall that graphs and digraphs both consist of two objects, a set of vertices and a set of edges. For graphs edges are undirected and for graphs they are directed.

Paths and Cycles: Let's concentrate on directed graphs for the moment. A *path* in a directed graph is a sequence of vertices $\langle v_0, v_1, \ldots, v_k \rangle$ such that (v_{i-1}, v_i) is an edge for $i = 1, 2, \ldots, k$. The *length* of the path is the number of edges, k. We say that w is *reachable* from u if there is a path from u to w. Note that every vertex is reachable from itself by a path that uses zero edges. A path is *simple* if all vertices (except possibly the first and last) are distinct.

A *cycle* in a digraph is a path containing at least one edge and for which $v_0 = v_k$. A cycle is *simple* if, in addition, $v_1, \ldots, v_k$ are distinct. (Note: A self-loop counts as a simple cycle of length 1).

In undirected graphs we define path and cycle exactly the same, but for a *simple cycle* we add the requirement that the cycle visit at least three distinct vertices. This is to rule out the degenerate cycle $\langle u, w, u \rangle$, which simply jumps back and forth along a single edge.

There are two interesting classes cycles. A *Hamiltonian cycle* is a cycle that visits every vertex in a graph exactly once. A *Eulerian cycle* is a cycle (not necessarily simple) that visits every edge of a graph exactly once. (By the way, this is pronounced "Oiler-ian" and not "Yooler-ian".) There are also "path" versions in which you need not return to the starting vertex.