17 Combinatorics

17.4 Connectivity

This project is based on material found in the Part II Graph Theory course.
In this project you will need to be able to generate graphs from $G(n,p)$, the space of graphs with $n$ labelled vertices, edges appearing independently and at random with probability $p$.

A connected graph is one in which there is a path between any two vertices. A component of a graph is a maximal connected subgraph. In this project we examine some algorithms for finding the components of a graph.

1 The adjacency matrix

Let $A$ be the adjacency matrix of a graph $G$ of order $n$ and let $D = A + I$.

**Question 1** What interpretation can you give to the entries $(D^2)_{ij}$, $(D^3)_{ij}$, ...? Describe a simple algorithm, based on this observation, for finding the components of a graph. What is the complexity of your algorithm?

**Question 2** Implement your algorithm and try out your program on several graphs from $G(80,p)$. Plot the number of components versus $p$ and comment on your results.

2 Direct algorithms

An alternative way to find the components is as follows. Start by labelling the vertices with numbers 1 to $n$. Then examine all the edges in turn; whenever an edge is found joining vertices with different labels, say 3 and 5, all vertices with label 5 are relabelled 3 (or vice versa, depending on whether there are more vertices labelled 3 than 5).

A third way to find the components is to “grow” the component containing a given vertex $v$ by constructing the sets $S_j$ of vertices distance exactly $j$ from $v$; having found $S_0 = \{v\}, S_1, \ldots , S_j$, the set $S_{j+1}$ can be found by examining $S_j$. If the component containing $v$ is not the whole graph, the procedure can be reapplied starting from some vertex $w$ not so far reached, and so on until all components are found.

**Question 3** Implement these algorithms and compare them to each other and to the first algorithm. What are their complexities?

An edge list is just a list of pairs of vertices joined by edges; an adjacency list is a collection of lists, one list for each vertex listing its neighbours.

**Question 4** How might the performance of these algorithms be affected if the graphs were stored not as adjacency matrices but as edge lists or as adjacency lists?
3 Isthmi

An isthmus in a graph is an edge whose removal increases the number of components. Alternatively, it is an edge contained in no cycle.

Question 5 Adapt one of the above algorithms to find the isthmi of a graph. Test your program on graphs in $G(80, p)$.

4 The incidence matrix

The vertex space $V$ of a graph of order $n$ is the $n$-dimensional space over GF(2) whose coordinates are labelled by the vertices. The scalar field GF(2) is just the set $\{0, 1\}$ with arithmetic performed (mod 2); this makes life very easy as every vector is just a 0-1 vector and corresponds in a natural way to a subset of the vertex set. The edge space $E$ is the corresponding $m$-dimensional space spanned by the edges, where $m = e(G)$.

The incidence matrix of $G$ is the $n \times m$ matrix whose columns are labelled by the edges and whose rows are labelled by the vertices; the $(ij)$ entry is 1 if vertex $i$ meets edge $j$ and is 0 otherwise. The incidence matrix gives a linear map $\partial : E \rightarrow V$ called the boundary map, and the transpose of the incidence matrix gives a linear map $\delta : V \rightarrow E$ called the coboundary map.

Question 6 Show that an edge is an isthmus if and only if it is not contained in any vector of the kernel of $\partial$. Show that a set of vertices is a union of components if and only if it is in the kernel of $\delta$.

Question 7 Write a procedure, using Gaussian elimination over GF(2), to find the kernel of a linear map specified by a matrix. Hence write a procedure to find the isthmi and the number of components of a graph. How does this method compare with the previous one?

Question 8 Comment briefly on the graph theoretic significance of the images of $\partial$ and $\delta$.

References