Graph Generators

- Given the diverse nature of graphs as combinatorial structures we can define numerous types of graphs
  - Some drawn from applications
  - Others have mathematical properties found in real graphs
- Graph generators can be used to test the algorithms that we write

Random Edges

- Simplest graph to generate
  - Given \( V \) vertices, generate pairs of numbers between 0 and \( V-1 \)
    - Likely result in a graph with self-loops and parallel edges (multigraph)
    - If we limit edges to those in a simple graph, generate many more edges
      - Only good for a sparse graph
  - Generate \( E \) edges
- This is the generator we’ve used so far

Random Graphs

- Considers all possible edges and include each in the graph with a fixed probability \( p \)
  - \( p \) is calculate such that \( p \times \text{total number of possible edges} \) is \( E \), the number of edges in the graph
  - Given some \( E \) and \( V \), what is the formula for \( p \)?

- Code generates close to \( E \) edges, but may not have exactly \( E \) edges - Why?

```c
Graph graphRandom(int vertices, int edges) {
    int i, j;
    Edge theEdge;
    Graph theGraph = graphInit(vertices);
    double p =

    for (i = 0 ; i < vertices; i++)
        for (j = 0; j < i; j++) {
            if (graphInsert(theGraph,createEdge(i, j, &theEdge)));
        }
    return theGraph;
}
```
Pros and Cons of Random Graph Generator

- **Pro**
  - Algorithms well-studied
  - Easy to implement

- **Con**
  - Do not necessarily have properties similar to ones we see in real world applications
    - Graphs usually model maps, circuits, schedules, transactions, etc.

k-Neighbor Graph

- Modification of a random-edge graph
  1. Randomly pick first vertex \( v \)
  2. Randomly pick second from among those whose indices are within a fixed constant \( k \) of \( v \)

- Graphs exhibit locality not found in random graphs

Euclidian Neighbor Graph

- First generates \( V \) points on a plane with random coordinates between 0 and 1
- Then generated edges
  - Connecting any two points within distance \( d \) of one another
    - If \( d \) is small,
    - If \( d \) is large,
- Models graphs that represent maps, circuits, etc.
- Drawbacks:

Transaction Graphs (Example)

- Consider Graph where
  - the vertices are all the phone numbers associated with Loyola (410-617-XXXX) and the numbers called on those phones
  - an edge represents a phone call placed at a Loyola phone to another phone
- What would the graph look like (dense or sparse)?

- What would need to be added to our collection of graph functions to deal with phone numbers representing vertices?
Function Call Graphs
- Vertex represents a function in a program
- In the static version, an edge represents a function that calls another in the text of the program
  - What does this study?

- In the dynamic version, an edge represents a function that calls another at run time
  - What does this study?

Other Graphs
- Interval Graph
  - Vertex represents an interval on the real line
  - Edge exists if two intervals intersect
- de Bruijin Graph
  - Number of vertices is power of 2
  - For each vertex $i$ there edge from $i$ to $2i$ and an edge from $i$ to $(2i + 1) \mod V$