Math 607: Computation and Combinatorics
Fall 2019
October 18 (Friday of Week 3)

At the beginning of class, we reviewed Wilson's algorithm from the previous lecture but didn't add any new information. To summarize: suppose $G = (V, E)$ is a graph from which we want to pick a spanning tree uniformly at random. We can do this using Wilson's algorithm, which says

- Randomly pick a root vertex r from the set of vertices and set $T_0 = \{r\}$.
- For incrementing i, repeat the following:
  - Pick a uniformly random vertex v_i not in T_i
  - Perform a loop-erased random walk in G until we get a path P_i from v_i to T_i
  - Set $T_{i+1} = T_i \cup P_i$
- Continue until T_k spans G.

Equivalently, we can think of Wilson's algorithm in the following way:

- On each vertex $v \in V$ except for a randomly chosen root vertex, place a stack of cards where each card has a uniformly random neighbor of v written on it, and the ith card in each stack has color i. The cards are independent of one another.
- If the top cards on the stacks form a cycle $v_1, v_2, \ldots, v_t = v_1$, remove the top card on each vertex in the cycle. Regardless of the order in which we remove the cycles, the same cycles are removed. Repeat until there are no cycles.
- If there are infinitely many cycles to be removed, this procedure never finishes. However, this is a probability 0 event, and the expected running time of the algorithm is finite.
- If the number of cycles to be removed is a finite integer k, then the cycles $C_1, \ldots, C_k$ come out in some order, leaving a uniformly random tree T. Wilson's algorithm "discovers" T by performing a loop-erased random walk.

Adjacency Matrices of Graphs

Let G be an undirected graph with no loops and no multiple edges, with n vertices $v_1, \ldots, v_n$.

Definition. The **adjacency matrix** for the graph G is the matrix $A = [a_{ij}]_{i,j}$ where

$$\begin{cases} 1 & \text{if } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases}$$

An **eigenvalue/eigenvector of the graph** G is an eigenvalue/eigenvector of its adjacency matrix A.

Since A is real and symmetric,

- The eigenvalues of A are real
- A gives an orthogonal eigenbasis
- A is diagonalizable over $\mathbb{C}$.

When we square the matrix A, we get

$$A^2 = \left[\sum_\ell a_{i\ell}a_{\ell j}\right]_{i,j} = \left[\#\text{ walks } v_i \to \star \to v_j\right]_{i,j}.$$

The trace of an $n \times n$ matrix M is defined as the sum of its diagonal entries, and is equal to the sum of its eigenvalues counted with multiplicity. That is,

$$\operatorname{tr}(M) = \sum_{1\le i\le n} a_{ii} = \sum_i \lambda_i.$$

If x is a formal variable, then

$$\operatorname{tr}(1 - xA)^{-1} = \operatorname{tr}(1 + xA + x^2A^2 + \dots)$$

is the generating function for the number of closed walks (walks which start and end at the same vertex) rooted at vertex i, with x marking the length of the walk.

We can find the eigenvalues of the adjacency matrix A using the **power method**:

- Since A is symmetric, we can write $A = PDP^{-1}$, where D is the diagonal matrix of eigenvalues of A, and P is the matrix of eigenvectors for A where the ith column of P corresponds to the (i,i)-entry of D. Assume the largest eigenvalue has multiplicity 1 and $|\lambda_1| > |\lambda_2| \ge \cdots \ge |\lambda_n|$.
- Then
$$A^k = P \begin{bmatrix} \lambda_1^k & & \\ & \ddots & \\ & & \lambda_n^k \end{bmatrix} P^{-1}. \tag{1}$$
For large k, λ_1^k dominates all other entries in the diagonal matrix.
- Given any vector v in $\mathbb{C}^n$, you can express v in the eigenbasis. Applying (1), $A^k v$ is approximately an eigenvector for λ_1 when k is large. In practice, to find the eigenvector corresponding to the largest eigenvalue of A, we can repeatedly set
$$v_i = \frac{Av_{i-1}}{|Av_{i-1}|},$$
starting with an arbitrary nonzero vector v_0. As i increases, v_i gets closer and closer to being an eigenvector corresponding to λ_1.