Computational Geometry: Lecture 12

Don Sheehy

March 5, 2010

1 Introduction and Recap

Recall the two key facts that we have proven about planar graphs thus far.

1. Every planar graph admits a straight-line embedding.
2. Every embedding of a planar, 3-connected graph is topologically equivalent.

Today, we're going to talk about how to **find** a straight-line embedding of a planar 3-connected graph.

2 How to Draw a Graph

In the 1960's, William Thomas Tutte wrote a paper called *How to Draw a Graph*. With a title like that, you might think that this paper must have solved the problem completely. Although his solution to this problem is very nice, there has been and still is a great deal of interest in finding nice embeddings of graphs. There is conference every year on graph drawing where researchers from all over the world contribute new work in this area. Today, we'll look at Tutte's method which is still commonly used in practice.

Tutte's Idea: Edges as Springs The key insight that Tutte brought to this problem was to imagine that all the edges were springs. Then they can just pull themselves into a nice configuration...maybe.

Here is an idea that doesn't work: Connect up the nodes of the graph so that every edge pulls on its endpoints with a constant spring constant and no minimum length. Take for the drawing the stable configuration of this system.

The reason this doesn't work should be obvious from our physical intuition. With all the springs pulling together, the equilibrium state will only occur when all of the vertices have converged to a single point. This is not a good drawing by any stretch of the imagination.

Let's tweak this only just slightly. Remember that we can find a face of a 3-connected planar graph even before we have a drawing. Let's pick one face, place its vertices in convex position. We turn the edges to springs again, but this time, we don't let the vertices of that one face move. All of the other vertices are left to find a static equilibrium.

Surprisingly, **this little tweak to the method works** to give a nice straight-line drawing. Today we'll show how to compute the final positions of all the vertices. Next time, we'll show why such an embedding has no crossings.

3 Static Equilibrium

Let's start with a single spring connecting two vertices, u and v sitting in $\mathbb{R}^2$. The force of v on u is the vector $k(v-u)$ where k is the spring constant. For the rest of the class, we'll set all spring constants to 1.

So, if we have a vertex q in the graph with neighbors $p_1, \ldots, p_h$, the total force acting on q is

$$F_q = \sum_{i=1}^{h} (p_i - q).$$

If the point q is at static equilibrium, then $F_q = 0$ and thus we can rearrange the preceding equation as follows.

$$F_q = \sum_{i=1}^{h} (p_i - q) = 0 \tag{1}$$

$$kq = \sum_{i=1}^{h} (p_i) \tag{2}$$

$$q = \frac{1}{k} \sum_{i=1}^{h} (p_i). \tag{3}$$

This should look familiar to you. It is the formula for the centroid. It means that if the vertices are at static equilibrium, then they are located at the centroid or barycenter of their neighbors.

Let F_i be the force exerted on vertex p_i. We have seen that this can be written as

$$F_i = \sum_{p_j \sim p_i} (p_i - p_j) = (\deg p_i) p_i - \sum_{p_j \sim p_i} p_j.$$

This is the same thing we did before, with a slightly more general notation. Here, $p_j \sim p_i$ means that p_i and p_j are adjacent in the graph.

Key Fact: F_i is a linear combination of the positions of the other points.

This means that we can express the static equilibrium in terms of a system of linear equations. Let me write down the form of this linear system and then we'll see why it correctly describes our situation.

Let L be an $n \times n$ matrix. For the diagonal entries, let $L_{ii} = \deg p_i$. For the off-diagonals ($i \neq j$), let $L_{ij} = -1$ if $p_i \sim p_j$ and let $L_{ij} = 0$ otherwise. This matrix will look very familiar to many of you. It is known as the *Laplacian* of G.

We just need to observe that it does the right the thing for computing forces. That is, $F_i = \sum_{i=1}^{n} L_{ij} p_j$.

So, if we want to set all of the forces to 0 then we would just need to solve the system $LP = 0$ where P is the $n \times 2$ matrix of the p_i's as row vectors. There is a solution to this system but it's not the one we want. Every such solution is the one that puts all of the points at the same location. This makes sense from a physical perspective because as we mentioned before, we need to do something special to the outer face in order to keep the springs from collapsing the whole graph to a point.

Let k be the number of vertices in some face. We can break up the matrix P in to parts $P = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix}$ so that P_1 is the first k rows of $P4$ and represents the vertices of our outer face. We can break up L similarly.

$$L = \begin{bmatrix} L_1 & B^T \\ B & L_2 \end{bmatrix} \tag{4}$$

Now, we only care about achieving static equilibrium at the interior vertices. The residual force on the outer face can be nonzero. Thus, we care about solving the following linear system.

$$\begin{bmatrix} L_1 & B^T \\ B & L_2 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} = \begin{bmatrix} F' \\ 0 \end{bmatrix} \tag{5}$$

We choose P_1 when we decide where to pin down the outer face. We can find P_2 by solving a linear system because the above equation implies that $BP_1 + L_2 P_2 = 0$ and thus $P_2 = -L_2^{-1} B P_1$.

But wait! You cheated! Why can we simply assume that L_2 is invertible? If L_2 is singular then we can't do this computation, so let's prove that L_2 is invertible.

There is a modern approach to proving this type of matrix is invertible. It involves using some more or less standard tools from linear algebra to observe that L_2 is *positive definite* (all eigenvalues are positive). All positive definite matrices are invertible, so that suffices. I'm not going to prove it that way. Instead, I'm going to show you how Tutte proved it.

Tutte's proof that L_2 is invertible used a really powerful tool known as the matrix-tree theorem.

Theorem 3.1 (The Matrix Tree Theorem)**.** *If L is the Laplacian of a graph and L' is derived from L by deleting the first row and column, then* $\det L'$ *is exactly the number of spanning trees of G.*

This Theorem should blow your mind at least a little bit. We're all computational geometers now so we love determinants. We're computer scientists so we love graphs and spanning trees. We're good computer scientists so we love Laplacians of graphs. Here they all are tied up in a beautiful little package. In fact, this Theorem is true even if we allow multiple edges (i.e. the graph is not *simple*). We will use this stronger version.

Modify our edge contraction so that any vertex previously adjacent to both endpoints will have 2 rather than just one edge going into the new contracted vertex. Now, if we contract the edges of the outer face to a single vertex in this way, the Laplacian of the new graph still has L_2 in its lower right corner. Now, applying the matrix-tree theorem implies that $\det L_2$ is number of spanning trees of the contracted graph. Since contractions cannot separate a connected graph, there will be at least one spanning tree. Thus $\det L_2 > 0$ and therefore it is invertible as desired.