Force Directed Drawing

Vincent La

May 5, 2018

1 Introduction

Force directed algorithms attempt to draw graphs by relating them to some physical analogy. For example, we may view vertices as steel balls and the edges between them as springs. One of the earlier force directed algorithms, Tutte's Barycenter Algorithm, attempts to place a graph's nodes along it's "center of mass."

2 Tutte's Barycenter Method

An early force directed drawing method was Tutte's Barycenter Method. In this method, the force on every vertex is given by

$$F(v) = \sum_{(u,v)\in E} (p_u - p_v)$$

Hence, we can ...

$$\sum_{(u,v)\in E} (x_u - x_v) = 0$$
$$\sum_{(u,v)\in E} (y_u - y_v) = 0$$

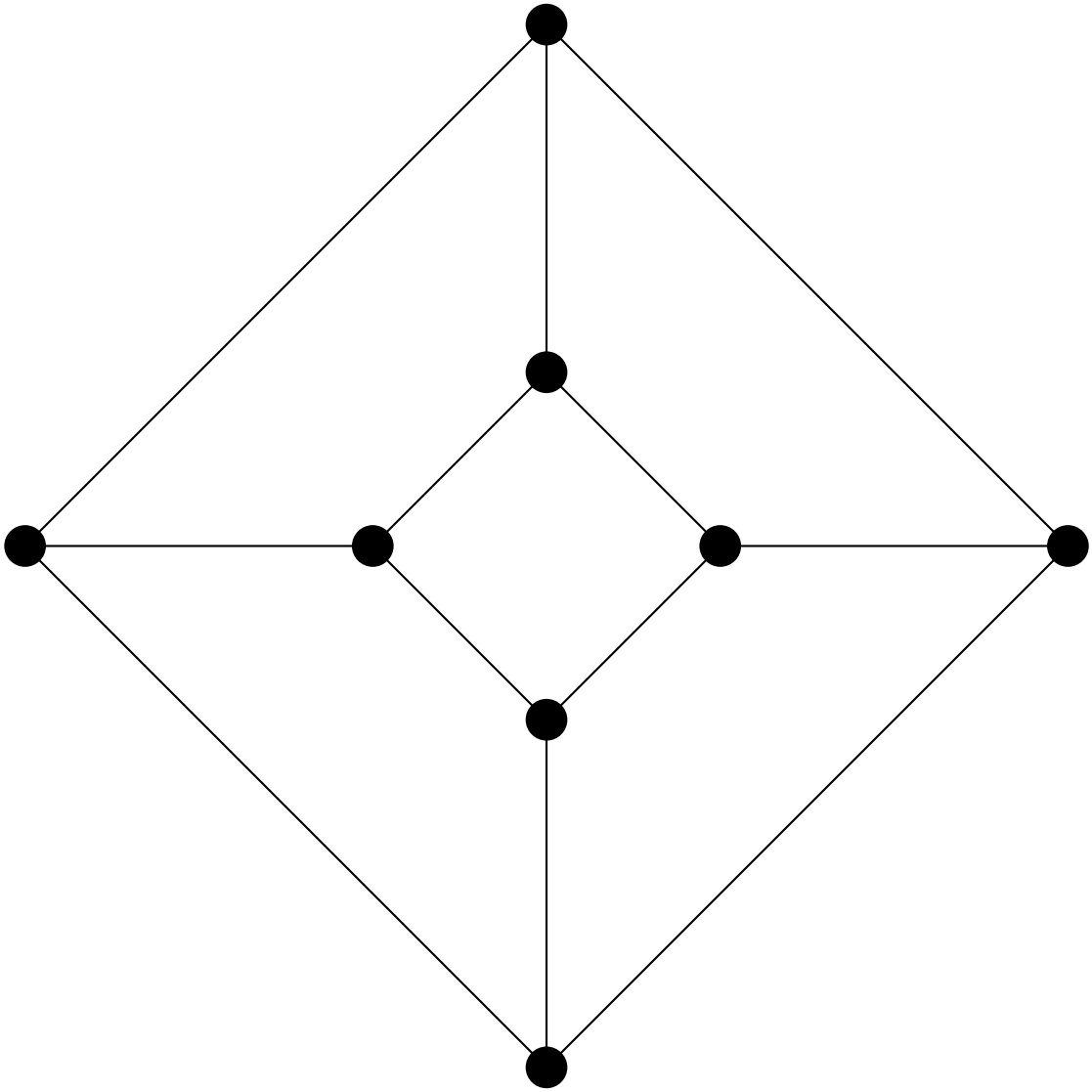
Which we may rewrite as

$$\deg(v)x_v - \sum_{u\in N_1(v)} x_u = \sum_{w\in N_0(v)} x_w^*$$
$$\deg(v)y_v - \sum_{u\in N_1(v)} y_u = \sum_{w\in N_0(v)} y_w^*$$

These equations are linear, and the resulting matrix is diagonally dominant (see Example 1.1). This is because the diagonal consists of vertex degrees, while the other entries a_{ij} are either -1's (if x_i and x_j are neighbors) or 0's if they aren't.

2.1 Example: Hypercube

A simple example for which Tutte's method gives aesthetically pleasing results is the hypercube.



In the image below, the hypercube is placed in 500 x 500 pixel grid. The grid is governed by a simple Cartesian coordinate system, where the top left and bottom right corners have coordinates $(-250, 0)$ and $(250, 250)$ respectively. Four vertices are fixed and laid out into a circle of radius 250 centered at the origin. Hence, the bulk of the work performed algorithm is done in placing the center four free vertices. Labeling the free vertices as x_1, x_2, x_3, x_4, we may represent the task of laying out the free vertices with this matrix

$$\begin{bmatrix} 3 & -1 & 0 & -1 \\ -1 & 3 & -1 & 0 \\ 0 & -1 & 3 & -1 \\ -1 & 0 & -1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 250 \\ 0 \\ -250 \end{bmatrix}$$

The solution to this matrix is given by $x_1 = x_3 = 0, x_2 = \frac{250}{3}, x_4 = -\frac{250}{3}$.

2.2 Algorithms

2.2.1 Newton-Raphson Iteration

Algorithm 1 Barycenter Layout

1: **procedure** BARYCENTER(t)
2: Layout n fixed vertices in a convex polygon
3: Construct an (size of free vertices) matrix A as follows

$$A_{i,j} = \begin{cases} \deg(v) & \text{if } i = j \\ -1 & \text{if adjacent} \\ 0 & \text{otherwise} \end{cases}$$

4: Solve $Mx = f_x$ to find the x-coordinates
5: Solve $My = f_y$ to find the y-coordinates
6: **end procedure**

2.2.2 Linear System

As seen above, the problem of applying the barycenter method can also been modeled with a linear system.

2.3 Prism Graph

The prism graph denoted Π_n is constructed by taking the vertices and edges of an n-prism. If we look below, the distance between vertices gets increasingly smaller as n increases.

2.3.1 Symmetry

Under certain conditions, the barycenter method produces drawings which preserve rotational symmetry.

Theorem: Points along a unit circle form an eigenvector Consider the linear system for coordinates of the free vertices of the prism graph Π_n, and then take its corresponding matrix M. Now, starting at $(1, 0)$, place n points equally along the perimeter of the unit circle. If we create a vector $x = (x_1, ..., x_n)$, then where x_i is the x-coordinate of the i^{th} unit circle point, then x is an eigenvector of M.

Algorithm 2 Barycenter Layout

1: **procedure** BARYCENTER(t)
2: Layout n fixed vertices in a convex polygon
3: Construct an (size of free vertices) matrix A as follows

$$A_{i,j} = \begin{cases} \deg(v) & \text{if } i = j \\ -1 & \text{if adjacent} \\ 0 & \text{otherwise} \end{cases}$$

4: Construct an (number of free vertices) vector x as follows

$$x_i = \sum ...$$

5: Solve $Mx = f_x$ to find the x-coordinates
6: Solve $My = f_y$ to find the y-coordinates
7: **end procedure**

Moreover, the same can be said for the y-coordinates and if we write $Mx = \lambda_x x$, $My = \lambda_y y$, it can be shown that $\lambda_x = \lambda_y$.

Proof. First, notice that $M = 3I + N$, where N is a matrix composed of all of the $-1's$ in M (and is zero everywhere else). Hence, $N\vec{x}$ is of the form

$$\begin{bmatrix} 0 & -1 & 0 & ... & 0 & -1 \\ -1 & \ddots & \ddots & 0 & ... & 0 \\ 0 & \ddots & \ddots & \ddots & ... & 0 \\ 0 & ... & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & ... & \ddots & \ddots & -1 \\ -1 & 0 & ... & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} \cos 0 \\ \cos \frac{2\pi}{n} \\ \cdots \\ \cos \frac{2\pi(n-2)}{n} \\ \cos \frac{2\pi(n-1)}{n} \end{bmatrix}$$

By distributivity,

$$Mx = (3I + N)x = 3Ix + Nx$$

Hence, showing that $Nx = \lambda x$ is equivalent to showing that the following holds for some $\lambda \in \mathbb{R}$.

$$\begin{cases} -\cos \frac{2\pi}{n} - \cos \frac{2\pi(n-1)}{n} = \lambda \cos 0 & \text{Equation for the first row} \\ -\cos \frac{2\pi(i-2)}{n} - \cos \frac{2\pi i}{n} = \lambda \cos \frac{2\pi(i-1)}{n} & \text{Equation for the } i^{th} \text{ row} \end{cases}$$

Now, the first equation implies that

$$\begin{aligned}
\lambda &= -\left[\cos\frac{2\pi}{n} + \cos\frac{2\pi n - 2\pi}{n}\right] \\
&= -2\left[\cos\frac{2\pi + 2\pi n - 2\pi}{2n}\cos\frac{2\pi - 2\pi n + 2\pi}{2n}\right] && \text{Using sum-product identity} \\
&= -2\left[\cos\frac{2\pi n}{2n}\cos\frac{4\pi - 2\pi n}{2n}\right] \\
&= -2\left[\cos\pi\cos\frac{2\pi}{n} - \pi\right] \\
&= 2\left[\cos\pi - \frac{2\pi}{n}\right] && \cos \text{ is an even function} \\
&= -2\left[\cos -\frac{2\pi}{n}\right] = -2\left[\cos\frac{2\pi}{n}\right] && \text{Supplementary angles}
\end{aligned}$$

Now, we just need to show take the equation for the i^{th} row and notice that

$$\begin{aligned}
-\left[\cos\frac{2\pi(i-2)}{n} + \cos\frac{2\pi i}{n}\right] &= -2\left[\cos\frac{2\pi(i-2) + 2\pi i}{2n}\cos\frac{2\pi(i-2) - 2\pi i}{2n}\right] && \text{Sum-product identity} \\
&= -2\left[\cos\frac{\pi i - 2\pi + \pi i}{n}\cos\frac{\pi i + 2\pi - \pi i}{n}\right] \\
&= -2\left[\cos\frac{\pi i - 2\pi + \pi i}{n}\cos\frac{2\pi}{n}\right] \\
&= \lambda\cos\frac{2\pi(i-1)}{n}
\end{aligned}$$

as desired.

□

2.3.2 Resolution

One the the main drawbacks of this algorithm is potentially poor resolution, i.e. the more edges and vertices we add to our graph, the harder it becomes to distinguish the different features of our graph. This is demonstrated best by the prism graph.

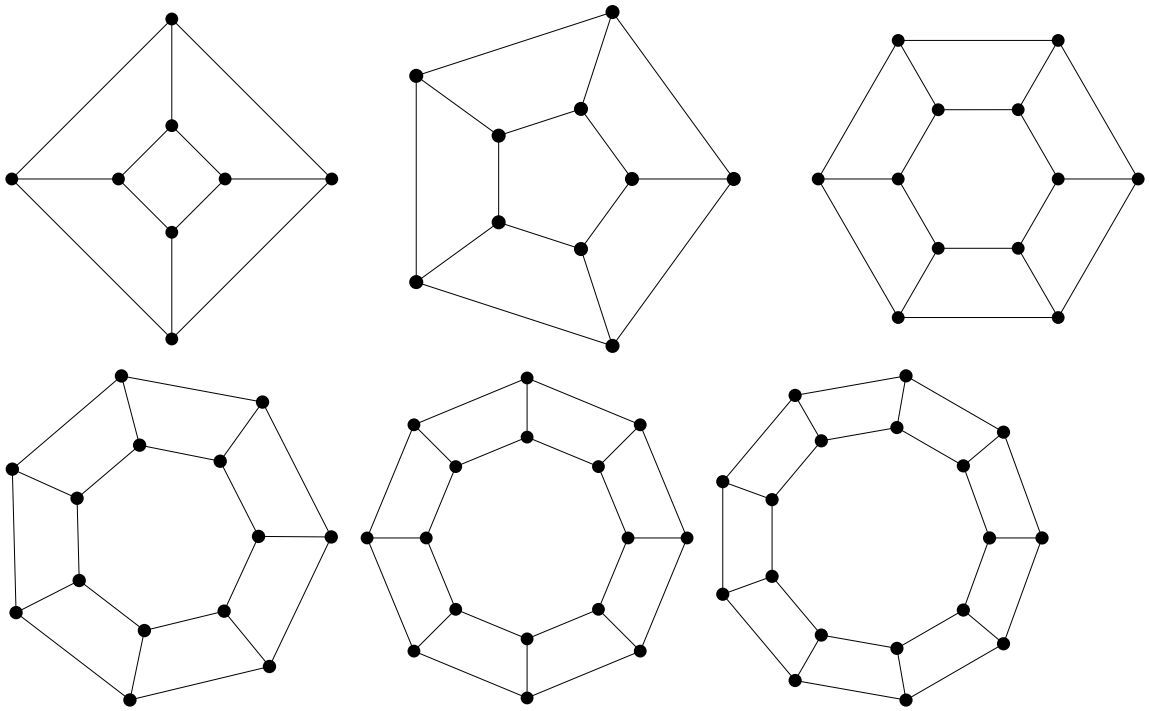


Figure 1: Π_4 through Π_9 as drawn by Tutte's algorithm. Notice that Π_4 is isomorphic to the hypercube Q_2

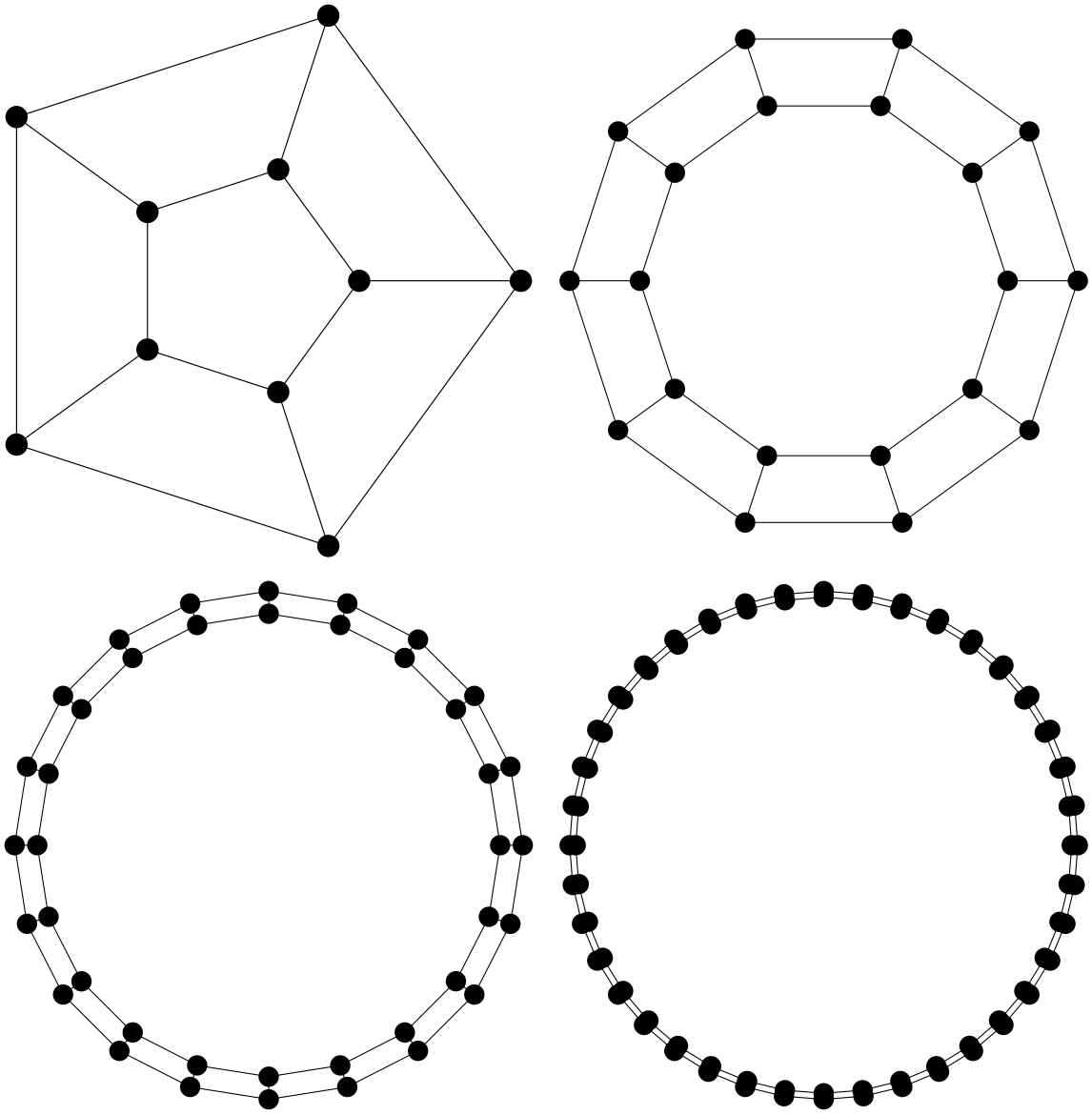


Figure 2: $\Pi_5, \Pi_{10}, \Pi_{20}$ and Π_{40} as drawn by Tutte's algorithm

Theorem For every fixed vertex u in the prism graph Π_n, the distance between it and its adjacent free vertex v is $O(\frac{1}{n})$.

Proof. From the theorem above, we know that

$$v = u \cdot \frac{1}{3 - 2\cos\frac{2\pi}{n}}$$

Hence,

$$\operatorname{dist}(u,v) = \sqrt{\left(u_x - u_x \cdot \frac{1}{3 - 2\cos\frac{2\pi}{n}}\right)^2 + \left(u_y - u_y \cdot \frac{1}{3 - 2\cos\frac{2\pi}{n}}\right)^2}$$

□