Force-Directed Methods [Chapter 10: Graph Drawing by Di Battista, Eades, Tamassia, Tollis]

- use a physical analogy to draw graphs
- the algorithm seeks a configuration of the bodies with locally minimal energy (the sum of the forces on each body is zero) $\longrightarrow$ equilibrium configuration

forced-directed methods in general have two parts:

- The model: A force system defined by the vertices and edges, which provides a physical model for the graph
- The algorithm: A technique for finding an equilibrium state of the force system that is a position for each vertex, such that the total force on every vertex is zero. This state defines a drawing of the graph.

10.1 Springs and Electrical Forces

- edges are modeled as springs
- vertices are equally charged particles which repel each other

the force on vertex v is:

$$F(v) = \underbrace{\sum_{(u,v)\in E} f_{uv}}_{\text{spring}} + \underbrace{\sum_{(u,v)\in V\times V} g_{uv}}_{\text{electrical repulsion}}$$

spring $\xrightarrow{\text{follows}}$ Hooke's law

electrical repulsion $\xrightarrow{\text{follows}}$ inverse square law

- $d(p,q)$: Euclidean distance between points p and q
- $P_v(x_v, y_v)$: the position of vertex v

So

x component of the force $F(v)$

$$= \sum_{(u,v)\in E} k^{(1)}_{uv}\left(d(p_u,p_v) - l_{uv}\right)\frac{x_v - x_u}{d(p_u,p_v)} + \sum_{(u,v)\in V\times V} \frac{k^{(2)}_{uv}}{(d(p_u,p_v))^2}\cdot\frac{x_v - x_u}{d(p_u,p_v)}$$

- l_{uv} : the natural (zero energy) length of the string between u and v is l_{uv}.
- $k^{(1)}_{uv}$: the stiffnes of the string between u and v
- $k^{(2)}_{uv}$: the strength of the electrical repulsion between u and v.

benefits (aims to satisfy important aesthetics)

- The spring force is aimed to ensure that the distance between adjacent vertices u and v is approximately equal to l_{uv}.
- The electrical force aims to ensure that vertices should not be close together
- Under certain assumptions the drawing tends to be symmetric

If we use logarithmic springs rather than Hooke's law springs, then

$$x \text{ component of } f_{uv} = k^{(1)}_{uv} \log\left(\frac{d(p_u, p_v)}{l_{uv}}\right) \frac{x_v - x_u}{d(p_u, p_v)}$$

"follow your nose" algorithm

Vertices are initially placed at random locations.
At each iteration, the force $F(v)$ on each vertex is computed, and each vertex v is moved in the direction of $F(v)$ by a small amount proportional to the magnitude of $F(v)$

10.2 The Barycenter Method.

Tutte's algorithm: $l_{uv} = 0$. $k^{(1)}_{uv} = 1$. $k^{(2)}_{uv} = 0$

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

- problem. trivial solution $p_v = 0$ for all v.
- to avoid: the vertex set V is partitioned into two sets, a set of at least three fixed vertices, and a set of free vertices

We choose p_v so that $F(v) = 0$ for each free vertex v:

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

Let $N_0(v)$ denote the set of fixed neighbours of v
Let $N_1(v)$ denote the set of free neighbours of v.

Solving them amounts to placing each free vertex at the barycenter of it neighbours

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

(x_w^*, y_w^*) is a position of a fixed vertex

$\deg(v)$ degree of v.

- The matrix is diagonally dominant. In practice, a simple Newton-Raphson iteration converges quickly.
- For planar graphs, the matrix is sparse and it is possible to solve the equations in $O(n^{1.5})$ [LRT79]

Algorithm 10.1 Barycenter-Draw

Input: graph $G=(V,E)$; a partition $V = V_0 \cup V_1$ of V into a set V_0 of at least 3 fixed vertices and a set V_1 of free vertices; a strictly convex polygon P with $|V_0|$ vertices

Output: a position p_v for each vertex of V, such that the fixed vertices form a convex polygon P.

1. Place each fixed vertex $u \in V_0$ at a vertex of P, and each free vertex at the origin
2. repeat

   foreach free vertex v do

   $$x_v = \frac{1}{deg(v)} \sum_{(u,v) \in E} x_u$$

   $$y_v = \frac{1}{deg(v)} \sum_{(u,v) \in E} y_u$$

   until x_v and y_v converge for all free vertices v.

Theorem 10.1 Suppose that G is a triconnected planar graph, f is a face in a planar embedding of G, and P is a strictly convex planar drawing of f. Then applying the barycenter algorithm, with the vertices of f fixed and positioned according to P, yields a convex planar drawing of G.

- the resolution is poor ?
- for every $n > 1$ there is a graph G s.t. the barycentric method outputs a drawing exponential area for any resolution rule.
- can be generalized to drawings obtained with a more complex energy function. For Barycenter-Draw, the energy of a drawing is the sum of the squares of the lengths of the edges.
- more generally, we can define the energy of a drawing as the sum of pth powers of the edge lengths.

10.3 Forces Simulating Graph Theoretic Distances

[KS80] J.B. Kruskal and J.B. Seery, Designing Network Diagrams

[KK89] T. Kamada and S. Kawai, An algorithm for Drawing General Undirected Graphs

graph theoretic distance $\delta(u,v)$

$G=(V,E)$ is a connected graph, $u,v \in V$, the graph theoretic distance, denoted by $\delta(u,v)$, is the number of edges on a shortest path between u and v

aim : find a drawing in which, for each pair u,v of vertices, the Euclidean distance $d(p_u,p_v)$ between u and v is approximately proportional to $\delta(u,v)$ between all pairs u and v of G.

Thus the system has a force proportional to $d(p_u,p_v) - \delta(u,v)$ between vertices u and v

Kamada and Kawai

take an energy view of this intuition

The potential energy in the spring between u and v is the intergral of the force that the spring exerts, that is

$$\frac{1}{2} \underbrace{k_{uv}}_{\text{stiffness parameter}} (d(p_u,p_v) - \delta(u,v))^2.$$

Kamada chooses $k_{uv} = \dfrac{k}{\delta(u,v)^2}$ for a constant k.

energy in (u,v) is $\eta = \frac{k}{2} \left(\dfrac{d(p_u,p_v)}{\delta(u,v)} - 1 \right)^2$

The energy η in the whole drawing is the sum of these individual energies,

$$\eta = \frac{k}{2} \sum_{u \neq v \in V} \left(\frac{d(p_u,p_v)}{\delta(u,v)} - 1 \right)^2$$

minima occur when the partial derivatives of η, with respect to x_v and y_v are zero.

This gives a set of $2|V|$ equations

$$\underbrace{\frac{\partial \eta}{\partial x_v} = 0, \quad \frac{\partial \eta}{\partial y_v} = 0, \quad v \in V}_{\text{nonlinear}}$$

An iterative approach may be used to solve them.

At each step, a vertex is moved to a position that minimizes energy, while all other vertices remain fixed.

The vertex to be moved is chosen as the one that has the largest force acting on it, that is, $\sqrt{\left(\frac{\partial \eta}{\partial x_v}\right)^2 + \left(\frac{\partial \eta}{\partial y_v}\right)^2}$ is maximized over all $v \in V$.

10.4 Magnetic Fields

Sugiyama and Misue [SM95a, SM95b]

"Graph Drawing by Magnetic-Spring Model"

"A Simple and Unified Method for Drawing Graphs: Magnetic-Spring Algorithm"

A model in which some or all of the springs are magnetized and there is a global magnetic field that acts on the springs.

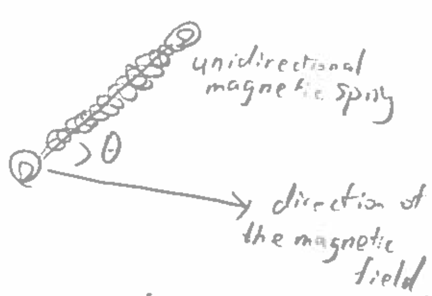
three basic types of magnetic field:

- Parallel: All magnetic forces operate in the same direction 
- Radial. The forces operate radially outward from a point 
- Concentric: The forces operate in concentric circles. 

the strings can be magnetized in 2 ways.

- Unidirectional
- Bidirectional

furthermore, a string may not be magnetized at all.



for a unidirectionally magnetized spring representing the edge (u,v), the force is proportional to $d(p_u, p_v)^\alpha \theta^\beta$, α, β are constants.

* the magnetic spring model is able to handle directed graphs ⟵ arcs point downward / outward / counterclockwise

- the method has also been applied with some success to orthogonal drawings (close to an hv drawing)

mixed graphs : graphs with both directed and undirected edges.

10.5 General Energy Functions

- including discrete energy functions

for example, for each drawing, we can define:

- the number of crossings
- the number of horizontal and vertical edges
- the number of bends in edges

main problem:

* it may be computationally expensive to find a minimum energy state.

we must resort to very general optimization methods such as simulated annealing and genetic algorithms.?

aesthetics conflict with each other, we cannot expect to optimize several criteria simultaneously

we can use an energy function that linearly combines a number of measures:

$$\eta = \lambda_1 \eta_1 + \lambda_2 \eta_2 + \dots + \lambda_k \eta_k$$

η measures the "ugliness" of the drawing and a drawing of minimum energy has maximum beauty.

Davidson and Harel [DH96]: Drawing Graphics Nicely Using Simulated Annealing

$$\eta = \lambda_1 \eta_1 + \lambda_2 \eta_2 + \lambda_3 \eta_3 + \lambda_4 \eta_4$$

$\eta_1 = \sum_{u,v \in V} \frac{1}{d(p_u,p_v)^2}$: similar electrical repulsion

aims to ensure the vertices do not come too close together

$\eta_2 = \sum_{u \in V} \left(\frac{1}{r_u^2} + \frac{1}{l_u^2} + \frac{1}{t_u^2} + \frac{1}{b_u^2} \right)$

r_u, l_u, t_u, b_u are the Euclidean distances between vertex u and the four sidelines of the rectangular frame in which the graph is drawn.

ensures vertices do not come to close to the borders of the screen.

$\eta_3 = \sum_{(u,v) \in E} d(p_u, p_v)^2$ edges do not become too long

$\eta_4 =$ the number of edge crossings in the drawing ? (How to minimize?).

The flexibility of general energy function methods allows a variety of aesthetics to be used by adjusting the coefficients λ_i.

[BBS 97] J. Branke, F. Bucher and H. Schmeck.

Using genetic algorithms for drawing undirected graphs.

10.6 Constraints

Force-directed methods can handle:

- Position constraints
- Fixed-subgraph constraints
- Constraints that can be expressed by forces or energy functions

- Iterative methods can confine the movement of vertices to the prescribed region at each iteration.

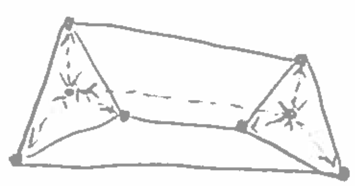
- the barycenter method may be seen as a force-directed method that constraints a set of vertices to a polygon shape.

Constraints that can be expressed by forces include:

- Orientaton of directed edges in a given direction. e.g., horizontal or vertical
- Geometric clustering of specidied sets vertices
- Alignment of vertices

Achieve clustering:

1. For each set C of vertices which need to be clustered, add to the graph a dummy "attractor" vertex v_C.
2. Add attractive forces between an attractor v_C and each vertex in C.
3. Add repulsive forces between pairs of attractors and between attractors and vertices not in any cluster.



[ECH97] P. Eades, R. Cohen and M. Huang. Online animated graph drawing for web navigation.

10.7 Remarks

[Ost 96] 1) Equations describing the minimal energy states are stiff? for some graphs of low connectivity.
2) Replacing the cliques of a graph by stars can improve the speed of spring algorithms for dense graphs.

Force-directed algorithms are heuristics which are best analyzed empirically.

[Ost 96] D. Ostry. Drawing Graphs on Convex Surfaces. Master's thesis.