Biograph Layout Algorithm

Wei Jing (Acer)

The layout algorithm we create is a hybrid between spring model and hierarchical model. It generates graph using spring model, and in the meanwhile, it provides hierarchical layout for individual reactions. The hierarchy for individual reaction is: substrates at top, products at bottom, and other compounds on the left or right. In addition, the nodes must also conform to compartment rule. A compartment is a box (in this algorithm, we actually only considered the upper and lower boundaries), and a node must reside in the compartment which it is assigned to. The compartments can make the hierarchy property for individual reactions more obvious.

We divide our algorithm into four phases: initialization, free layout, constrained layout, and post-processing.

Initialization is to quickly provide an initial layout, which includes three steps. Firstly, we generate the initial coordinates of nodes according to the edges. For each edge, we place its two ends according to the edge type, for example, we place the substrate above a reaction if it is a substrate edge. After that, a node swapping step is done to minimize the initial system force. In the third step, again we try to get the adjacent nodes at the correct relative location. But this time we adopted the spring model to relax the positions of nodes. After the initialization, the relative position for most node pairs should be satisfied. However, there may be many node-overlapping.

The second phase is free layout: adjusting the node positions without compartment constraints. We use spring model to adjust node positions from the previous step (the third step in initialization) onwards. In our spring model, the ideal distances between adjacent nodes are proportional to the size of the node pairs (0.6 times the sum of the size). The minimum distances between non-adjacent pairs we used are 1.8 times the sizes of the node pairs. Distances between adjacent nodes are shorter because we want the related nodes (compounds and reactions) to be close to each other. As for the force, we have distantal force and angular force (which is for adjacent nodes only). Distantal force is proportional to the square of (actual distance minus ideal/minimum distance). Angular force is proportional to the product of (square of distance) and (sine of a half angle difference). In the free layout part, we also adopted a method to firmly distribute the edges around a node. After the free layout, the nodes should be spread out, and there should be no or few node-overlapping. The layout generated is similar to that generated from Neato.

In the third phase, we bring in compartments and use it to constrain the node positions. This is composed of two parts: initializing compartments and compartment-constrained layout. Actually, the methods we adopt to deal with compartments are simple. For the initialization, we calculated the average y-coordinates for all nodes inside a Compartment-x (denoted as ymid[x]). Then we sort the ymid’s in increasing order. After that we set the lower boundary and upper boundary of Compartment-x to be (ymid[x-1]+ymid[x])/2 and (y[mid]+y[mid+1])/2. The left and right boundaries are negative infinity and positive infinity. In the constrained-layout, in addition to the methods used in the free layout, we adopted two other methods: force due to compartments, and adjusting compartments. When a node is outside is compartment, we bring it to the nearest point inside the corresponding compartment. The force induced is the square of the displacement. In the adjusting part, a compartment is trying to become the smallest rectangle enclosing the nodes belonging to it.

After the constrained layout, the compartments are satisfied, and the hierarchy properties are stood out for individual reactions. However, there could be few node-overlaps. There are still optimizations we can do: further reduce the edge lengths to make the graph more contracted, reduce edge-crossing and remove node-overlaps. Thus we adopt a post-processing phase in the end.

As is described above, the post-processing phase consists of three steps: minimizing edge-crossing, shrinking edge lengths, and removing node-overlaps. But before these, we have another step—Step 0: calculating post-processing distances. In order to shrink the edges, only reducing the ideal edge lengths is not enough, since the large repulsive force between non-adjacent nodes (which connecting to the same neighbor) we prohibit the edge from shrinking. Therefore, we should reduce the non-adjacent distance for node-pairs with common neighbor. In the first step, once we detected an edge crossing (this is done using cross produce of vectors), we rotate the lowest-connected node such that the two edges become parallel to each other. In edge-shrinking step, we reduce the ideal-edge length if the adjacent pairs are separated far enough. In the node-overlap-removal step, nodes moves faster to get separated from one another. In the node post-processing step (more precisely, the edge-shrink step and the overlap-removal step), the force calculation is different from the previous. The force here is calculated by an exponential function, which we think will force the long edges to become shorter and overlapping nodes to separate.

After the fourth phase, the layout graph is expected to be pretty readable: there exist no node-overlap (or perhaps some minor ones which does not affect readability), few edge-crossing, connected nodes lies close to each other and all nodes conforms to compartments. In such layout graph, the hierarchy property for individual reactions is highlighted by compartments.