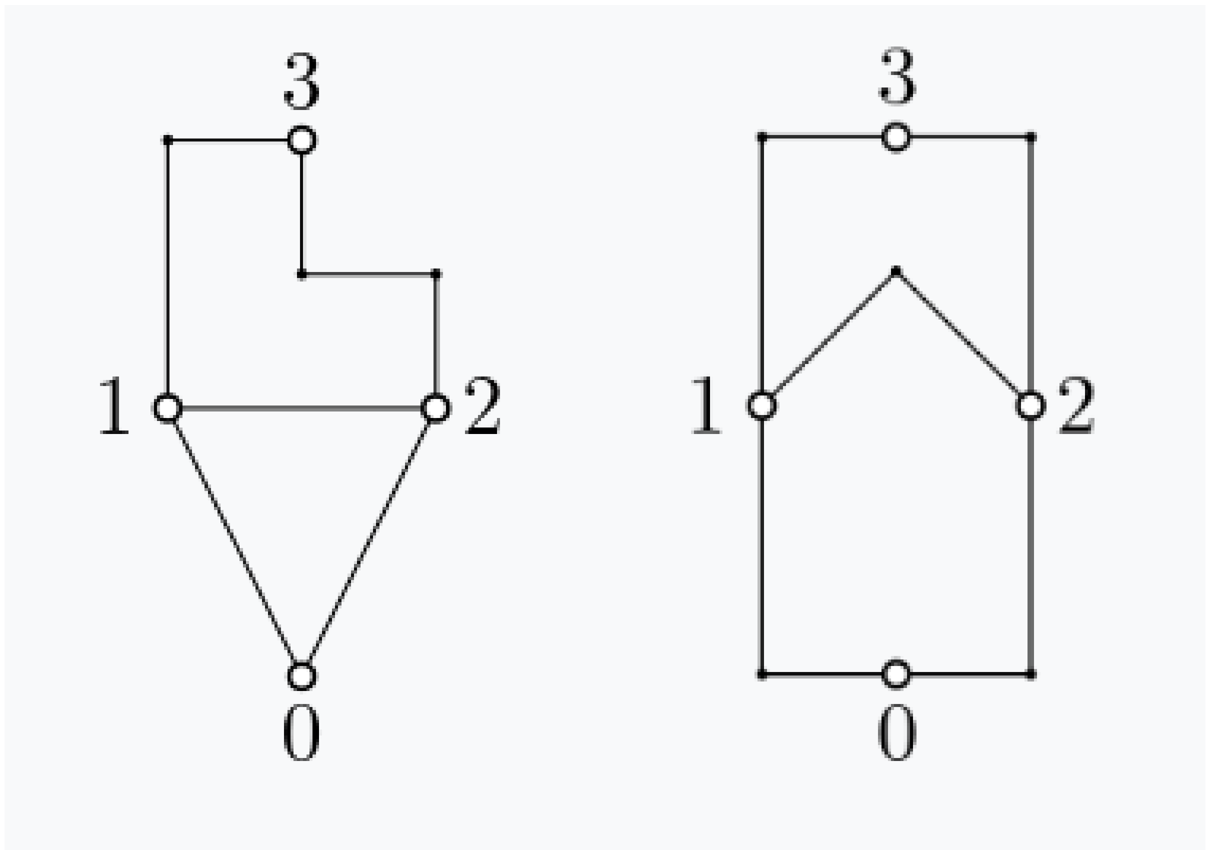
INF562 Project : Graph Drawing Live Challenge

Capucine Leroux and Nissim Maruani

March 2021



Contents

1 Problem presentation

The problem consists of finding a planar drawing of an input graph that minimizes the polyline edge length ratio : $\frac{\text{maximum edge length}}{\text{minimum edge length}}$. The graph has to be on a bounded grid with integer coordinates, it can have bended edges and the number of bends can be bounded too. The details of the challenge are written on the website : *http://mozart.diei.unipg.it/gdcontest/contest2021/index.php?id=live-challenge.*

2 Our method

2.1 Planar drawing

The first issue to tackle is to make sure we start with a valid planar drawing of the graph. Some inputs do not necessarily have an initial valid position for the nodes.

One very well known way to draw a planar graph with no edge crossing is called the Tutte method. It consists in :
- Choosing an outer face and put it in the shape of a polygon
- Expressing all the interior vertices as a barycenter of their neighbors

There are two different possible strategies to find the barycentric coordinates, either iteratively apply :

$$\forall \text{ interior } v, v = \frac{1}{deg(v)} \sum_{(u,v)\in E} u$$

Or directly solve :

$$\begin{aligned} (I-W)x &= b_x \\ (I-W)y &= b_y \end{aligned}$$

Because it was quicker to solve the system once and because the ratio score was similar for both methods, we eventually decided to use the solving method.

The difficult part of Tutte's algorithm is the choice of the outer face. First, since we don't use any combinatorial embedding there, we don't have direct access to faces so we have to find a different way to select a proper face, and second, a good outer face can have a big influence on the ratio we want to minimize. To find the outer face, we selected all the cycles C of the graph G using a BFS method, we selected the ones where $G \backslash C$ was still connected. Among those cycles, we found that some of them still could not be eligible for outer faces, (See Figure 1) so we applied the Tutte method on each one of them and kept the one giving the best valid solution (best as in the one with minimum ratio).

Once the outer face is found, we put the outer vertices on a regular polygon of maximum size to spread the interior vertices as much as possible so they don't all get into the same position.

For some graphs, we could not find a good outer face that gave us a valid solution, in those cases, we used the python networkx library that gives a planar drawing in linear time (See Figure 2). We did not always use this library since Tutte's method could sometimes give better ratio results.

Now that we have a valid planar drawing for our graph, we want to try and improve the ratio with different approaches.

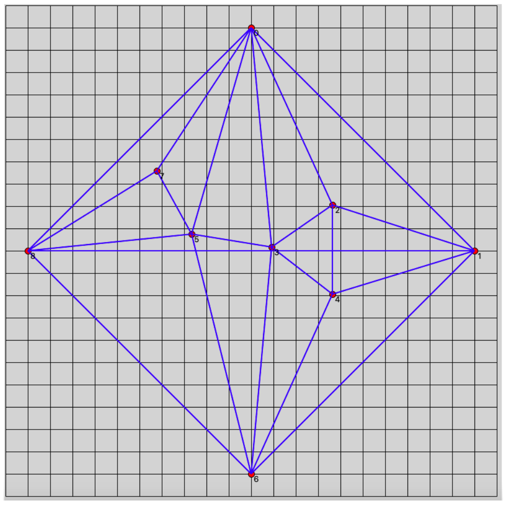


Figure 1: Wrong choice of outer face

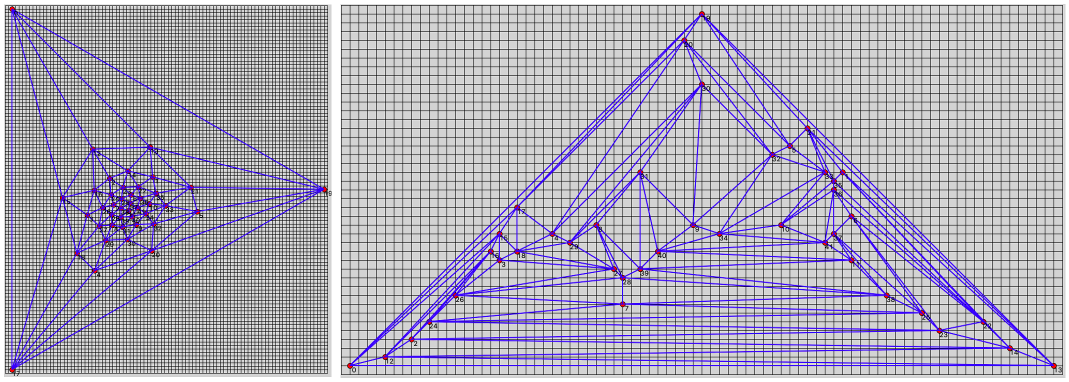


Figure 2: Difference between Tutte (Left) and Networkx (Right) on the same graph

2.2 Random switch

Having also followed INF561, we wanted to try the following idea: pick a random vertex and move it a bit. If the new graph is still planar, and if its polyline edge length ratio is smaller than before, we keep this configuration (See Figure 3).

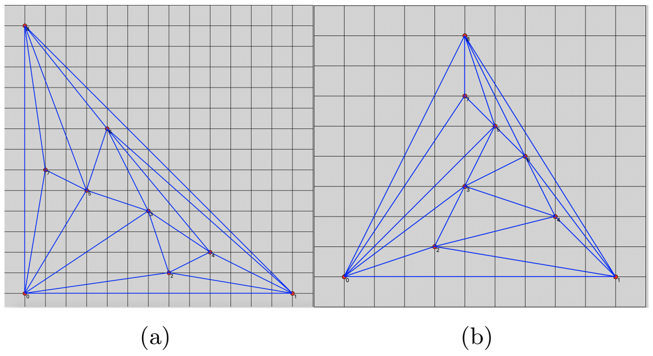


Figure 3: The graph before and after 80 random switches on the 'small' instance

After just 80 steps (computed in 3 seconds), the polyline edge ratio is divided by 2 (see Figure 4).

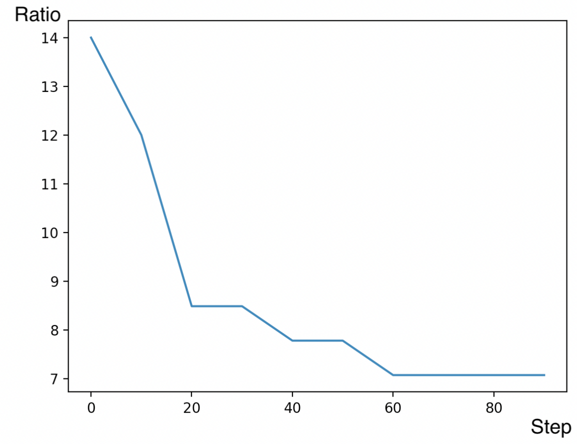


Figure 4: Polyline edge ratio evolution

One advantage of this step is it can be applied at anytime and does not compromise the validity of the solution.

2.3 Force simulation

Another idea was to use the physical concepts seen in the course to better our ratio. We fix the vertices of the outer face, and act as if each edge were a spring of equilibrium length 0. During each step, the vertices are moved slightly.

There are several limitations to this step. Indeed, with the networkx library, the initial graph does not necessarily have a convex outer face, which can compromise the planarity of the drawing after a few steps. To tackle this issue, we fixed the positions of the points of the convex hull instead of the outer face when the outer face was not convex.
Moreover, this step does not assure us to have a decreasing ratio.
However, it tends to be very useful with Tutte planar drawings where the vertices of the middle tend to be too close to each other. This force step helps separating the inside vertices more and therefore, compensate the big discrepancy between the outer lengths and the inner lengths (See figure 5). To get a more realistic result when using the Tutte drawings, we put a stronger spring force on the vertices connected to the outer face to wide out the middle even more.

2.4 Bends

We were also allowed to bend some edges, usually with a maximum number of bends per edge. The strategy here is to iteratively bend the first edge of minimum length that is still allowed bending. Adding a bend to the shortest edge can help decrease the ratio, but because the bend does not always have an obvious valid position, we had to recompute the drawing with the added bend each time. It turns out to be quite helpful when applying Tutte method because it tends to stretch the bended edge in the new drawing. Yet, this step does not always decrease the ratio either.

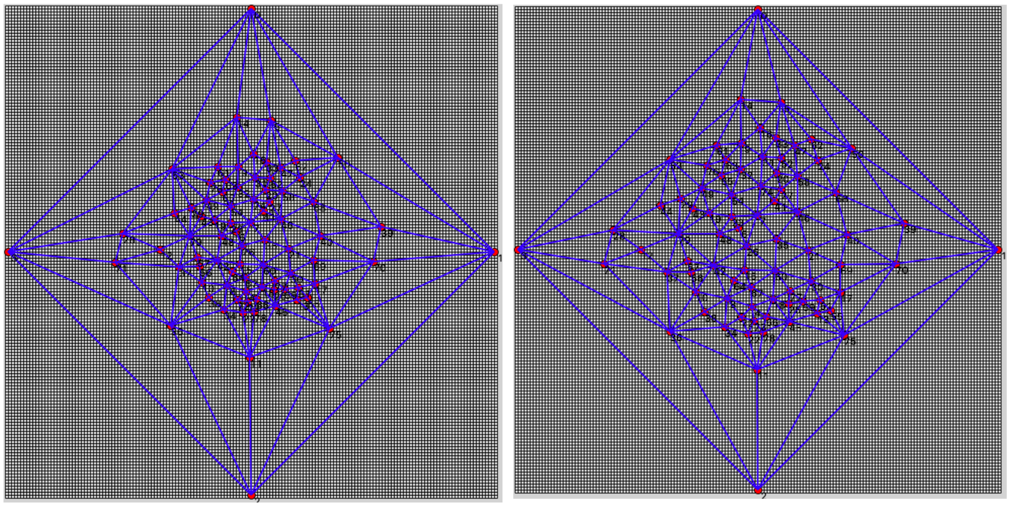


Figure 5: Before and after 130 force steps, going from ratio 37 to 27

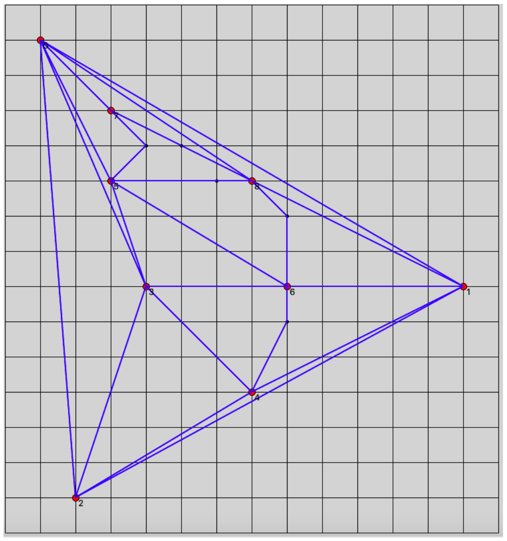


Figure 6: Example of added bends

2.5 Global strategy

For each graph, we tested the two different initial planar drawings, then all the improvements such as bending the small edges, simulating a force spring several times or randomly move the vertices a little if it reduces the ratio. After testing different configurations, we kept the better solution. According to the type of graph, the number of edges, the size of the grid and so on, a given configuration can work very well on a graph and very poorly on another one.

3 Results and limits

We implemented our solution for one type of input format only : the json format. We tested the different performances of our method on the inputs we had, and documented in an excel file the different ratio and time scores. Some ratio scores have been calculated even though the corresponding solution might not be valid, that is why the final ratio is not always the minimum one written.

3.1 Time performance

The tutte embedding is slower than the networkx library solution. Tutte takes between 0.02 seconds for 9 nodes up to 7 seconds for 299 nodes, whereas the networkx algorithm takes respectively 0.002 and 0.1 seconds. It is logical since Tutte theoretically has a $O(n^3)$ complexity against $O(n)$ for networkx, and we search tutte's best solution between several possible cycles. Yet, both solutions give a planar solution in reasonable time.

Checking the validity of an instance takes up to 0.8 seconds for 299 nodes, and getting the polyline edge length ratio takes up to 0.001 seconds. Both are quick enough for our use.

3.2 Ratio performance

The initial scores can be drastically different with Tutte and Networkx. Generally, when the graph is too big or when there is no good outer face, Tutte has a much bigger ratio, in the other case it is much better, and it is more easily improved by the different approaches we mentionned, up to twice as low at the end of the process, whereas in Networkx's case, the different following steps don't make much difference on the ratio afterward.

3.3 Approximating float coordinates

One limit of our work is that for many steps we approximate float coordinates with integer coordinates. However, it would require a much more complex algorithm to handle this approximation properly. Right now, we only round the coordinates to the closest grid point within the maximum grid, this means that two points can be rounded to the same place, or that a valid solution can be transformed into a non valid solution. We probably lost many possible good solutions in the process, by approximating them and making them non valid. Our approach could be much more efficient with a good approximation algorithm. In particular, some final solutions did not even use any of the bends/force/random tools because after those steps, the instances were not valid anymore.