Global placement with spectral methods

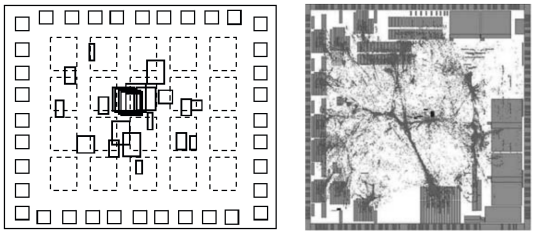
Rodrigo Arias Mallo

January 21, 2018

1 Background

In the process of design and synthesis of an electronic circuit, several steps are involved. The placement step determines the position of each cell or logic element in the circuit in two stages: global and detailed placement.

The global placement determines the approximate position, while the detailed placement refines and legalizes the final positions. A simple global placement is shown in the figure 1a. We see the cells are not legally positioned yet, as they overlap and are not in the allowed positions, shown with discontinuous line. A more complex placement is shown in the figure 1b.



(a) simple (b) complex

Figure 1: Global placement

The placer estimates some quality metrics like wire-length, wire congestion or signal delay. Analytics techniques can be used to model the problems as a minimization of an objective function via mathematical analysis.

2 Motivation

The cells and nets can be modeled with undirected graphs, were nodes represent the cells, and edges the wires. The spectral methods presented in the Koren [2] paper use some properties of graphs to place each node, while a estimation of the wire-length is minimized.

The spectral methods present two major advantages compared with force directed methods. The global optimum can be computed efficiently, and the energy function contains $O(|E|)$ instead of $O(|V|^2)$ entries.

3 Methods

Two main methods are presented, using the Laplacian matrix L and a normalized version. Let the undirected graph $G = (V, E)$ be represented by the adjacency matrix A, defined as

$$A(i,j) = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are connected} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Let the degree of a node i be defined as d_i, and the degree matrix D with the degree of each node in the diagonal, $D(i,i) = d_i$ and zeros elsewhere.

3.1 Method 1: Laplacian matrix

The problem is then formulated as the minimization of the squared wire-length. Let $x(i)$ be the position in the horizontal axis of the node i.

$$\begin{aligned} &\min_x E(x) = \sum_{(i,j)\in E} (x(i) - x(j))^2 \\ &\text{given } \operatorname{Var}(x) = 1 \end{aligned} \tag{2}$$

The energy to be minimized $E(x)$ tries to reduce the distance between nodes, while the variance $\operatorname{Var}(x)$ keeps the nodes from collapsing into a single point. The problem can be rewritten using the Laplacian matrix L, defined as $L = D - A$ or also

written

$$L(i,j) = \begin{cases} d_i & \text{if } i = j \\ -1 & \text{if } i \text{ and } j \text{ are connected} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

So we finally get

$$\begin{gathered} \min_x x^T L x \\ \text{given } x^T x = 1 \\ \text{in the subspace } x^T \cdot 1_n = 0 \end{gathered} \quad (4)$$

Were the solution for x is computed by solving the eigenproblem $Lu_i = \lambda_i u_i$. The horizontal positions of the nodes are given by $x = u_2$.

The vertical coordinates can be obtained by using the next eigenvector $y = u_3$, which is orthogonal to x.

3.2 Method 2: Normalization

A slight modification of the model can lead to more interesting results. Nodes with high degree should be placed at the center, in order to reduce the costs of their multiple edges. In the following model

$$\begin{gathered} \min_x \frac{x^T L x}{x^T D x} \\ \text{in the subspace } x^T D 1_n = 0 \end{gathered} \quad (5)$$

the numerator places the high degree nodes close to the center, while the denominator tries to scatter them away. As a result, the graph is more balanced, avoiding low degree nodes placed far away.

The solution of this new model can be obtained by solving the following generalized eigenproblem:

$$Lu_i = \lambda_i D u_i \quad (6)$$

And using the associated eigenvector u_2 of the second lowest eigenvalue λ_2 we get the horizontal positions of the nodes $x = u_2$. For the vertical positions we use the third eigenvector, $y = u_3$.

4 Results

In order to test and compare the results of the two algorithms, the method of force-directed placement [1] using attractive and repulsive forces was also included for comparison.

Three graphs of increasing size were used. Graph A was hand generated, to test the placement by inspecting the intermediate matrices. The graph B is a random Erdös-Renyí graph with 100 nodes and $p = 0.1$. Finally, a real circuit was generated from a Verilog example, and synthesized by the open tool Yosys [3] of the IceStorm project, using the Lattice iCE40 FPGA as the target. This final graph C contains 345 nodes.

The wire-length was computed as the sum of Euclidean length of the edges, and the positions have been scaled to have standard deviation $\sigma = 1$, so that we can compare wire-length under the same dimensions. The method of using a square bounding box of unit area to scale the graph was rejected, as it can lead to great reduction in wire-length if a few nodes are placed very far away.

Results are shown for the three graphs in the figures 2 3 and 4, using the three methods previously described.

5 Discussion

In the graph A show in the figure 2, we see how the three methods produce similar results, with the Laplacian method giving the lower wire-length.

The figure 3 shows a more interesting pattern. All high degree nodes are placed in the center, in the three methods. However, the Laplacian method gives a layout with a lot of overlapping nodes, with high variability in the node edges. A few low degree nodes were placed very far away. The method 2, deals with this problem, by placing the nodes in a more uniform way, while keeping the high degree nodes in the center. In terms of wire-length, the Laplacian is still getting the best result, but methods 2 and 3 seem more reasonable in a realistic problem.

The final graph C, shown in figure 4 shows a different network, with large chains of low degree nodes. Similarly, we see a lot of overlapping nodes with the first method, but more sparse using the method 2, while the 3 has a more uniform distance between nodes, but the worst wire-length.

We see some nodes in the placements given by the Laplacian method, about 8 units away from the center $(0, 0)$, for graphs B and C. However, if we scale the graphs to fit the unit square, we reduce the wire-length up to half the size, compared to the

method 2. Because of this problem, the graphs were scaled to have standard deviation $\sigma = 1$, allowing some outlier nodes to be far away from the graph.

6 Conclusions

We see that the Laplacian method produces a very short wire-length solution, but by using a lot of overlapping nodes. The normalized method has a behaviour closer to the real limitations, similarly to the force-directed method, while keeping the wire-length shorter.

Additionally, the algorithm provided in the Koren paper [2], based on the power iteration, can be used to extract only the needed eigenvectors, leading to an efficient method of obtaining the global minimum.

Followed by a fine placement step, the normalized method seems to be suitable for larger graphs.

References

[1] T. M. J. Fruchterman and E. M. Reingold, *Graph drawing by force-directed placement*, Softw. Pract. Exper., 21 (1991), pp. 1129–1164.

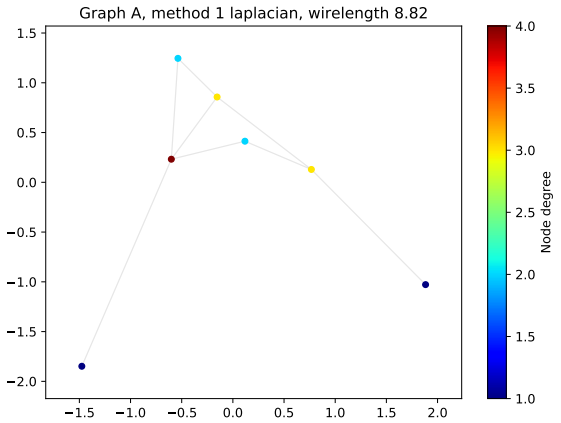
[2] Y. Koren, *On spectral graph drawing*, in Proceedings of the 9th Annual International Conference on Computing and Combinatorics, COCOON'03, Berlin, Heidelberg, 2003, Springer-Verlag, pp. 496–508.

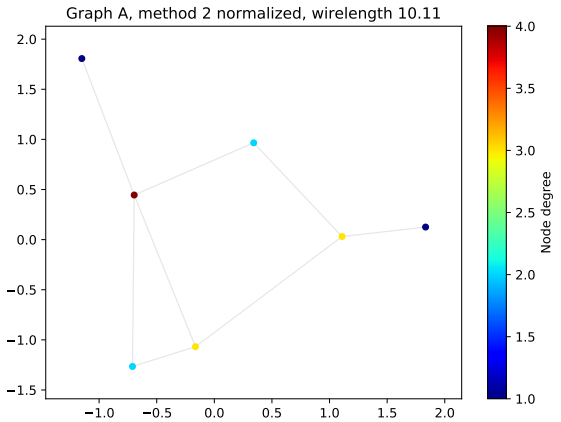
[3] C. Wolf and J. Glaser, *Yosys - a free verilog synthesis suite*, in Austrochip Workshop on Microelectronics 2013, 10 2013, pp. 47–52.

A Comment

In the presentation of this project, the code used to plot the normalized Laplacian method was incorrect. As a result, the graphs shown had nodes with high degrees away from the center.

Two different methods were used to test for consistent results, with two different bugs that lead to the same wrong graph. One of the method presented used the normalized Laplacian matrix $\mathcal{L}$, but was excluded from this report, as the actual explanation is simpler. After a long debugging process, both bugs were found and fixed. The presented graphs now are consistent with the expected results, giving a placement with high degree nodes in the center.





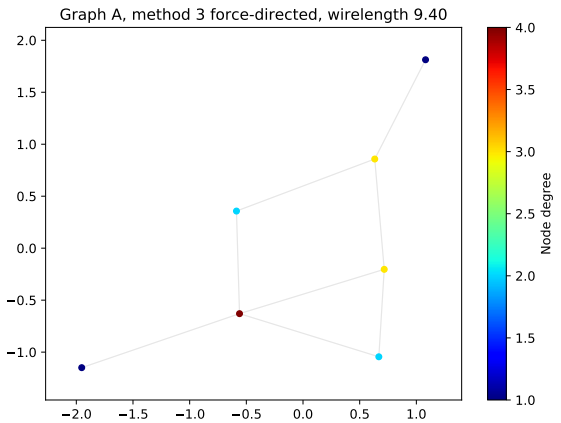
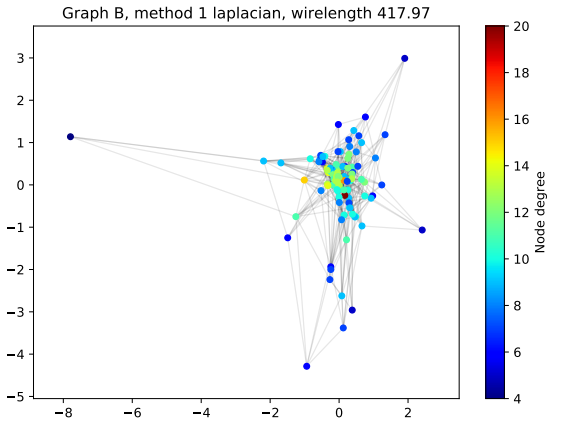
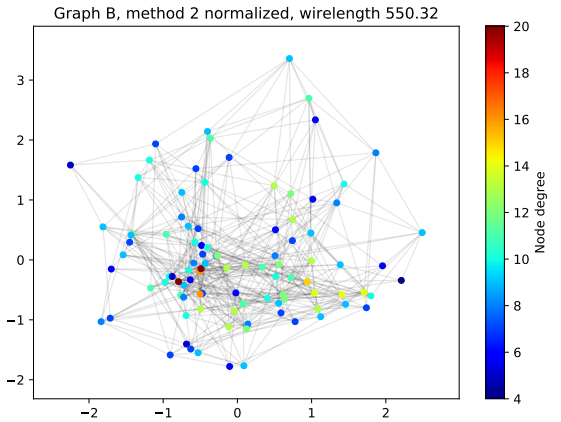


Figure 2: Placement of graph A





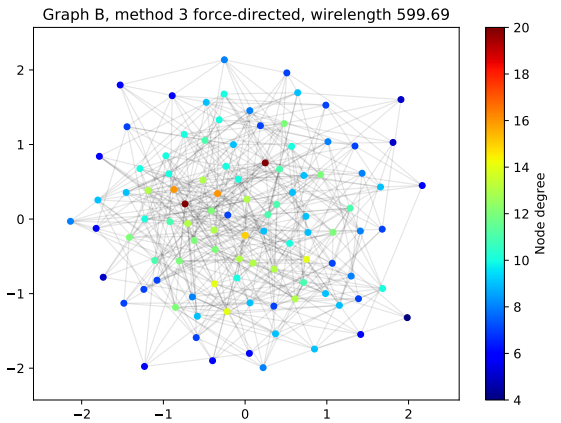


Figure 3: Placement of graph B

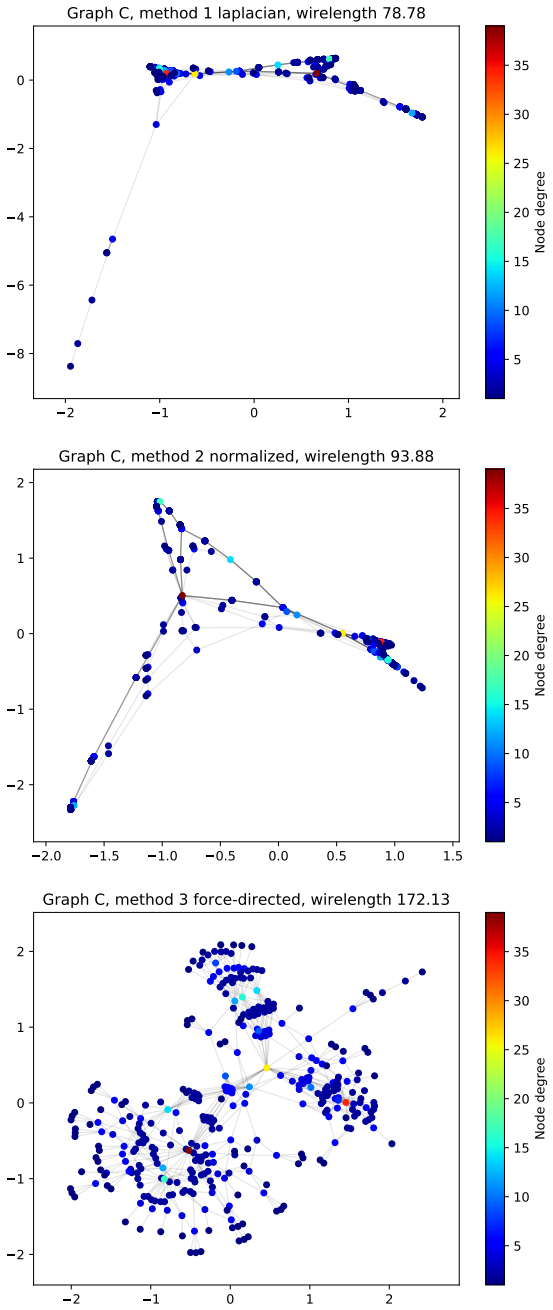


Figure 4: Placement of graph C