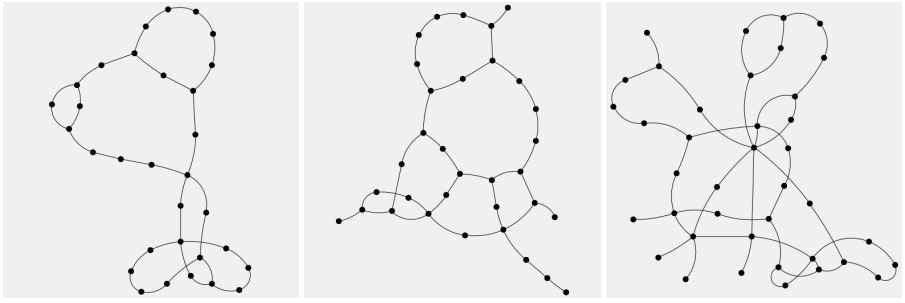
Lombardi Spring Embedder (Short Demo)

Roman Chernobelskiy, Kathryn Cunningham, and Stephen G. Kobourov

Department of Computer Science, University of Arizona Tucson, AZ, USA

Abstract. A *Lombardi drawing* of a graph is defined as one in which vertices are represented as points, edges are represented as circular arcs between their endpoints, and every vertex has perfect angular resolution, as measured by the angle formed by the tangents to the edges at the vertex. We describe an algorithm that creates Lombardi or *near-Lombardi* graph drawings, in which all edges are still circular arcs, but some vertices may not have perfect angular resolution. Our algorithm has a fully functional implementation; source code, images and movies illustrating the system can be found at http://lombardi.cs.arizona.edu.



1 Introduction

The readability of a graph drawing depends on many factors, with several experimental studies pointing to edge crossings and angular resolution as having significant impact [19, 20]. One of the features that stands out in American abstract artist Mark Lombardi's work is the even distribution of edges around vertices [16]. This even spacing of edges around each vertex, together with the use of circular arcs for edges, define *Lombardi drawings* of graphs [7]. Force-directed layout algorithms, also known as spring embedders, are well-known for the "organic" type of drawings they produce. However, perfectly straight-line segments rarely occur in nature. Combining force-directed algorithms with the idea of even distribution of circular arc edges results in drawings that appear even more "alive" than usual.

1.1 Related Work

Early work on angular resolution for graph drawing addresses the problem in the straight-line setting [5, 11, 17] or polyline setting [13, 15]. Even with circular-arc edges, maintaining bounded angular resolution leads to exponential area [4]. Force-directed algorithms have been used to maximize the angular resolution using curved edges [9] and for fixed position drawings with cubic Bézier curves [1, 3]. Curved edges and polyline

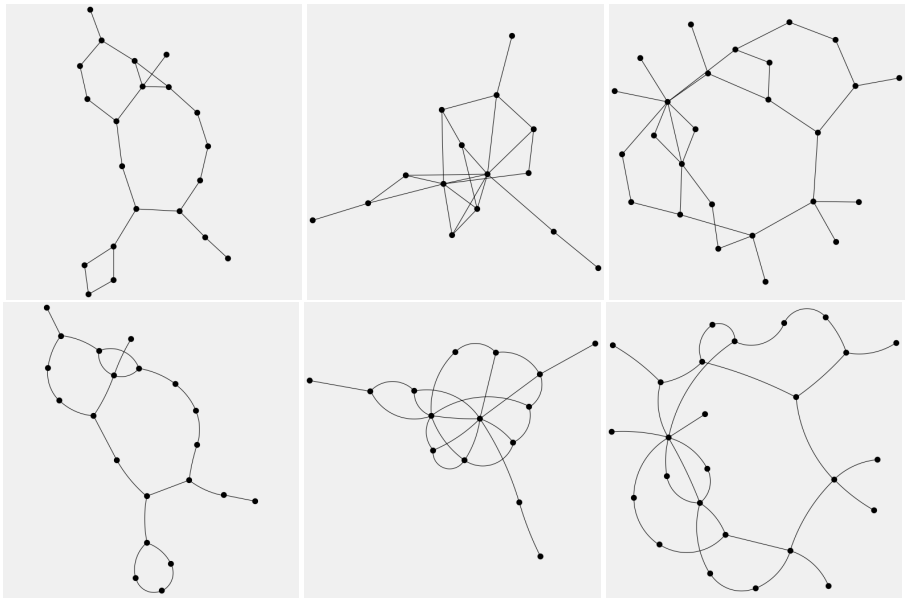


Fig. 1. Examples of standard straight-line and Lombardi drawings.

edges have been studied in the context of drawing planar graphs with good angular resolution [12, 14]. Rotating optimal angular resolution templates for each vertex in the fixed position setting has also been studied [2]. An algorithm for force-directed Lombardi graph construction has been proposed, but not implemented [18].

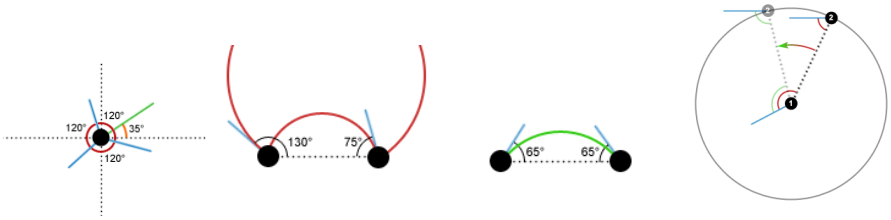
1.2 Our Results

We describe a force-directed algorithm that produces Lombardi (near-Lombardi) drawings of graphs, where every edge is represented by a circular arc and every vertex has a perfect (or near-perfect) angular resolution. While not all graphs allow for Lombardi realizations, many do. As our algorithm follows the spring-embedder paradigm, it is not guaranteed to find such a drawing even if one exists. In such cases, we opt for a near-Lombardi drawing in which all edges are still circular arcs, but some vertices may not have perfect angular resolution; see Fig. 1. We implemented the algorithm and tested it on several graph libraries and with several graph generators. A web-enabled demo, as well as complete source code, image libraries, and several movies illustrating the algorithm at work can be found at http://lombardi.cs.arizona.edu.

2 Lombardi Spring Embedder Formulation

Force-directed algorithms treat a given graph as a physical system, where the vertices represent points in an N-body mechanics problem. In the system set up by Eades, vertices are treated as steel rings and the edges are springs that obey Hooke's Law [8]. Fruchterman and Reingold describe a model in which a strong nuclear force attracts two protons within the atomic nucleus at close range, while an electrical force repels them at farther range [10]. Although inspired by physics, most force-directed algorithms do not attempt to mimic physical laws precisely.

Similar to most force-directed layout algorithms, our Lombardi spring embedder assigns a force to each vertex and aims to minimize the overall energy of the system.



(a) (b) (c) (d)

Fig. 2. Lombardi forces move and rotate vertices so that all corresponding tangents have matching angles, allowing for feasible circular arcs. (a) An illustration of pre-assigned tangents for a given degree-3 vertex. The angles between the tangents are equal and remain fixed, while the vertex itself can be rotated by changing its orientation with respect to the origin (currently 35° as indicated by the green line); (b) If the angles differ, then there cannot be a common circular arc between them; (c) If the angles are equal, there is a unique circular arc between them; (d) The tangential force for vertex 2 with respect to vertex 1 attempts to achieve matching outgoing angles from the two vertices.

There are three forces which affect vertex position, and one force which affects the radius of the circular arcs between a pair of vertices connected by an edge.

The attractive force, F_a, pulls vertices connected by edges closer together. It is applied to every pair of vertices connected by an edge as follows:$F_a = (d-k)/d$, where d is the current distance between the two vertices and k is a constant representing the ideal spring length.

The repulsive force, F_r, pushes vertices apart. It is applied to every pair of vertices using the following formula: $F_r = k^2/d^3$.

The tangential and rotational forces make it possible for circular arcs to be drawn between vertices, while maintaining a perfect (near-perfect) angular resolution. To help compute the two forces, we augment each vertex with an orientation and fixed tangents, which dictate how to draw the arc. The angles between the tangents are equal and remain fixed, while the vertex itself can be rotated by changing its orientation with respect to the origin. Note that the angle of a tangent at one vertex must equal the angle of a tangent at the other vertex for an arc to be possible between them. Here, angles are measured with respect to the segment connecting two vertices; see Fig. 2.

The tangential force, F_t, attempts to move vertices in such a way as to make a circular arc possible between any pair of vertices connected by an edge. To compute this force we need to find the optimal position of a vertex with respect to its neighbor. The magnitude of the force is proportional to the distance between this optimal position and the current position, and the direction is towards the optimal position. It is applied to every pair of vertices that share an edge as follows: $F_t = A \times d$, where d is the distance between current and optimal positions, and A is the tangential force constant.

The rotational force, F_ρ, does not attempt to move vertices, but to rotate a vertex and its tangent template so as to make the tangent angles match, thereby making the arc between two vertices possible. To compute the rotational force we find the optimal angle of a tangent and subtract the current angle as follows:$F_\rho = B \times \Delta angle$, where $\Delta angle$ is the rotation required and B is a small rotational constant.

For each vertex v, the three appropriately scaled movement forces are added together to the rotational force in order to determine the overall force acting on the vertex: $F(v) = F_a + F_r + F_t + F_\rho$.

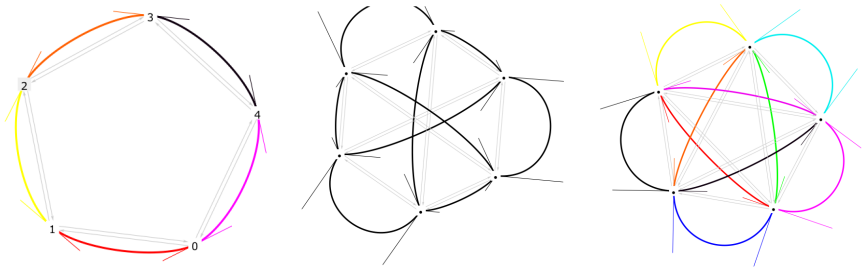


Fig. 3. Perfect Lombardi drawings of C_5, $K_{3,3}$ and K_5, with shown tangents.

The following cooling function is used to determine the magnitude of the force in terms of the number of iterations: $T(i) = (T_0 * (M - i))/M$, where i is the iteration number and M is the maximum number of iterations. T_0 is calculated as follows: $T_0 = K * \sqrt{n}/5$, where K is the ideal spring length and n is the number of vertices. Experimentally determined values for the constants we use are $K = 0.3$, $M = 600$, $A = 0.9$, $B = 0.5$; see Fig. 3 for examples of simple graphs where we obtain perfect Lombardi drawings.

Note that as described the algorithm assumes a fixed order of the neighbors around each vertex. Fixing such an order can be very limiting in computing a Lombardi drawing. With this in mind we allow for modifying the order of adjacencies by shuffling the tangents at the beginning of each iteration to find a lower energy state. If the number of tangents (i.e., degree of a vertex) is small, we try every permutation to find the one with the minimal energy. If the number of tangents is large we move one tangent at a time. The energy function that we try to minimize with this shuffling is similar to the rotational force. Recall that when we computed the rotational force, we were interested in the net force for each vertex. With this shuffling force, however, we are interested in the total force (the sum of absolute values of each contributing rotation).

3 Near-Lombardi Spring Embedder

As not all graphs are Lombardi graphs [6], and our algorithm cannot guarantee that it will find a Lombardi drawing even if one does exist, when needed we relax the perfect angular resolution constraint. If the Lombardi Spring Embedder has failed to find optimal positions for every vertex, we modify the tangents of vertices which have infeasible edge constraints.

For each tangent pair (between two connected vertices with infeasible edge constraints), we place the tangents at an angle equal to the average of the two. Now we can draw circular arcs between all connected vertices with minimal loss of angular resolution. As the change in tangents may be too drastic, we improve the angular resolution by another round of force-directed simulation. Rather than change the position and rotation of vertices as we did before, here we only modify tangent angles (which were fixed before). This new force is based on the observation that a tangent should be in the middle of two tangents clockwise and counter-clockwise from it. We find this midpoint and compute the force which is proportional to the required rotation to get the tangent there: $F_{NL} = C \times \Delta angle$, where $\Delta angle$ is the difference between current angle and optimal angle, and C is a constant. In order to maintain the "equal tangent angles" rule,

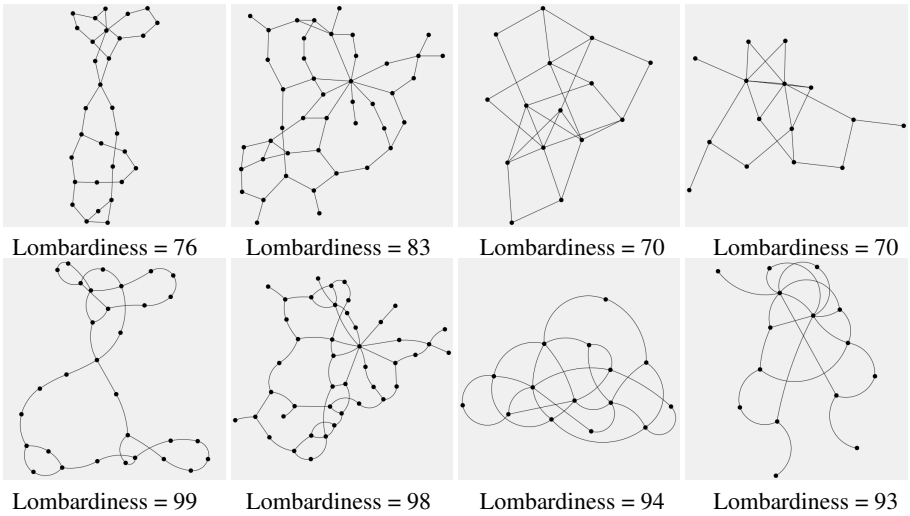


Fig. 4. Standard force-directed and near-Lombardi drawings.

we compute the force using both of the tangents along an edge, and apply it equally to both of them.

For near-Lombardi drawings we need a measure of quality. As all edges are still circular arcs, the only violations are at vertices where perfect angular resolution could not be achieved. With this in mind, we define the *Lombardiness* of a drawing to be a number in the range 0 to 100, defined by the average deviation from perfect angular resolution. To compute this value we add the deviations $\sum_{v \in V} \Delta angle/2|E|$ and scale to the 0-100 range dividing by 1.8 (as the maximum value of $\Delta angle$ is 180).

Note that this measure of Lombardiness can be applied to all drawings we compute, as well as to straight-line drawings computed by a standard force-directed method (after all, straight-line segments are circular arcs with radius infinity). Figure 4 shows several pairs of graphs drawn with a standard force directed embedder and with our Lombardi spring embedder, along with their Lombardiness scores. For 80% of the 5451 graphs in the Rome library with 50 vertices or less, we obtain Lombardiness scores of 98 or higher, while very few have scores in the low 90's.

4 Conclusion and Future Work

We demonstrated a simple extension of the spring-embedder paradigm for creating Lombardi and near-Lombardi drawings. A feature that can often be seen in Mark Lombardi's art is that many vertices follow common trajectories. This feature is not included in the definition of a Lombardi drawing [7], but does occur frequently in drawings obtained by our spring embedder. While previous work on using cubic Bézier curves for good angular resolution is similar in spirit, the resulting drawings do not have vertices following common trajectories.

Before arriving at the spring embedder described here, we considered several other ideas, including those described by Matsakis [18], but did not obtain a functioning prototype. There are several natural directions to explore in future work, including alternative formulation of the spring forces, a multi-level version that would scale to larger graphs, as well as possible use of this approach along with confluent drawing and edge

bundling. A very informal user feedback indicates some aesthetic appeal of the drawings produced by the Lombardi spring embedder. Some keywords and phrases associated with these types of drawings were "more natural, " "balloon animals," "blobby," "cute and cuddly," in contrast with the traditional straight-line realizations which were more "jagged" and "angular"; see Fig 8. In a formal evaluation of the effectiveness of Lombardi-style visualization it would be interesting to study the engagement level and the time spent examining the two different styles of drawings.

Acknowledgments We would like to thank Ulrik Brandes and Alexander Wolff for useful discussions and suggestions.

References

1. U. Brandes and B. Schlieper. Angle and distance constraints on tree drawings. In *Proc. 14th Int. Symp. on Graph Drawing (GD 2006)*, pages 54–65, London, UK, 2007.
2. U. Brandes, G. Shubina, and R. Tamassia. Improving angular resolution in visualizations of geographic networks. In *2nd TCVG Symp. Visualization (VisSym '00)*, pages 23–32, 2000.
3. U. Brandes and D. Wagner. Using Graph Layout to Visualize Train Interconnection Data. *J. Graph Algorithms Appl.*, 4(3):135–155, 2000.
4. C. C. Cheng, C. A. Duncan, M. T. Goodrich, and S. G. Kobourov. Drawing planar graphs with circular arcs. *Discrete Comput. Geom.*, 25(3):405–418, 2001.
5. G. Di Battista and L. Vismara. Angles of planar triangular graphs. *SIAM J. Discrete Math.*, 9(3):349–359, 1996.
6. C. A. Duncan, D. Eppstein, M. T. Goodrich, S. G. Kobourov, and M. Löffler. Planar and Poly-Arc Lombardi Drawings. Submitted to the 19th Symp. on Graph Drawing. 2011.
7. C. A. Duncan, D. Eppstein, M. T. Goodrich, S. G. Kobourov, and M. Nöllenburg. Lombardi Drawings of Graphs. In *Proc. 18th Int. Symp. on Graph Drawing (GD 2010)*, 2010.
8. P. Eades. A heuristic for graph drawing. *Congressus Numerantium*, 42:149–160, 1984.
9. B. Finkel and R. Tamassia. Curvilinear Graph Drawing Using the Force-Directed Method. In *Proc. 12th Int. Symp. on Graph Drawing (GD 2004)*, pages 448–453. 2005.
10. T. Fruchterman and E. Reingold. Graph drawing by force-directed placement. *Softw. – Pract. Exp.*, 21(11):1129–1164, 1991.
11. A. Garg and R. Tamassia. Planar drawings and angular resolution: algorithms and bounds. In *2nd European Symposium on Algorithms*, pages 12–23, London, UK, 1994.
12. M. T. Goodrich and C. G. Wagner. A framework for drawing planar graphs with curves and polylines. *J. Algorithms*, 37(2):399–421, 2000.
13. C. Gutwenger and P. Mutzel. Planar polyline drawings with good angular resolution. In *Proc. 6th Int. Symp. on Graph Drawing (GD'98)*, pages 167–182, 1998.
14. C. Gutwenger and P. Mutzel. Planar polyline drawings with good angular resolution. In *6th International Symposium on Graph Drawing*, pages 167–182, 1998.
15. G. Kant. Drawing planar graphs using the canonical ordering. *Algorithmica*, 16:4–32, 1996.
16. M. Lombardi and R. Hobbs. *Mark Lombardi: Global Networks*. Independent Curators, 2003.
17. S. Malitz and A. Papakostas. On the angular resolution of planar graphs. *SIAM J. Discrete Math.*, 7(2):172–183, 1994.
18. N. Matsakis. Transforming a random graph drawing into a lombardi drawing. *CoRR*, abs/1012.2202, 2010.
19. H. Purchase. Which aesthetic has the greatest effect on human understanding? In *Proceedings of the 5th Symposium on Graph Drawing (GD)*, pages 248–261, 1998.
20. H. C. Purchase, R. F. Cohen, and M. James. Validating graph drawing aesthetics. In *Proceedings of the 3rd Symposium on Graph Drawing (GD)*, pages 435–446, 1996.

Appendix

Here we include some examples of Mark Lombardi's art and some more drawings generated by our Lombardi Spring Embedder. In Fig. 5 we can see that near-perfect angular resolution is a good approximation to what the artist used. In Fig 6 we can also see that groups of vertices follow well-defined trajectories.

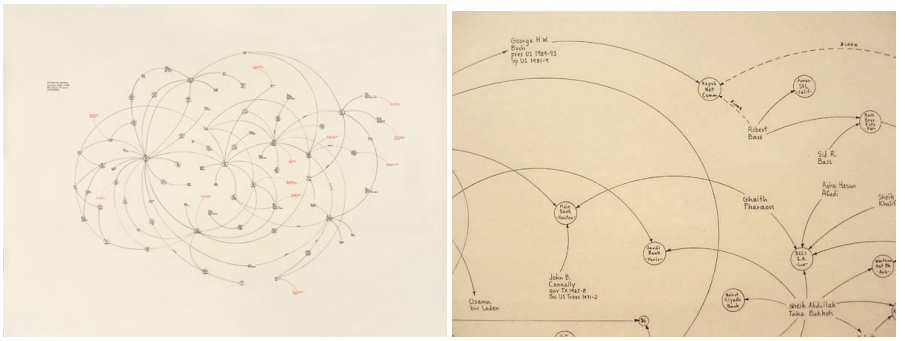


Fig. 5. Examples of Mark Lombardi's art; (a) The topmost vertex could probably be realized with perfect angular, but it is not, probably to allow for clear and well-defined trajectories of a number of vertices in the actual drawing. (b) Several vertices deviate from the normal perfect angular resolution, probably to accommodate more important nodes, which often have very well distributed vertices, unless there is a "direction" in the underlying story.

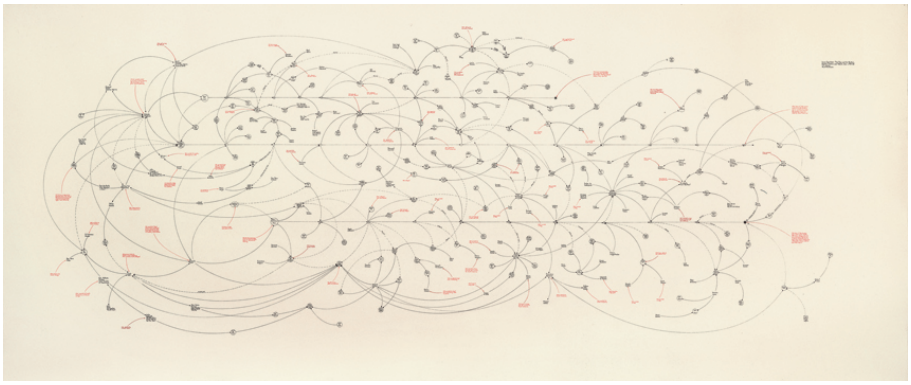


Fig. 6. A large example of Mark Lombardi's, showing several multi-vertex trajectories and hard-wired left-to-right time component.

We include a few more examples of our Lombardi spring embedder, including some graphs from the Rome library that give a similar overall expression to that of Mark Lombardi's drawings; see Fig 7. We also include some standards such as dodecahedron and icosahedron; see Fig. 8.

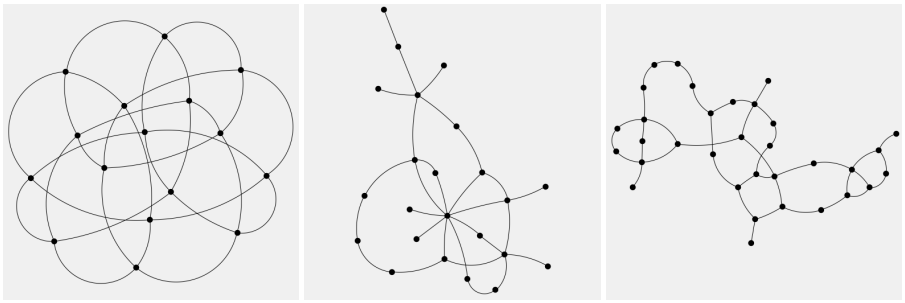


Fig. 7. Rome library graphs realized by the Lombardi spring embedder.

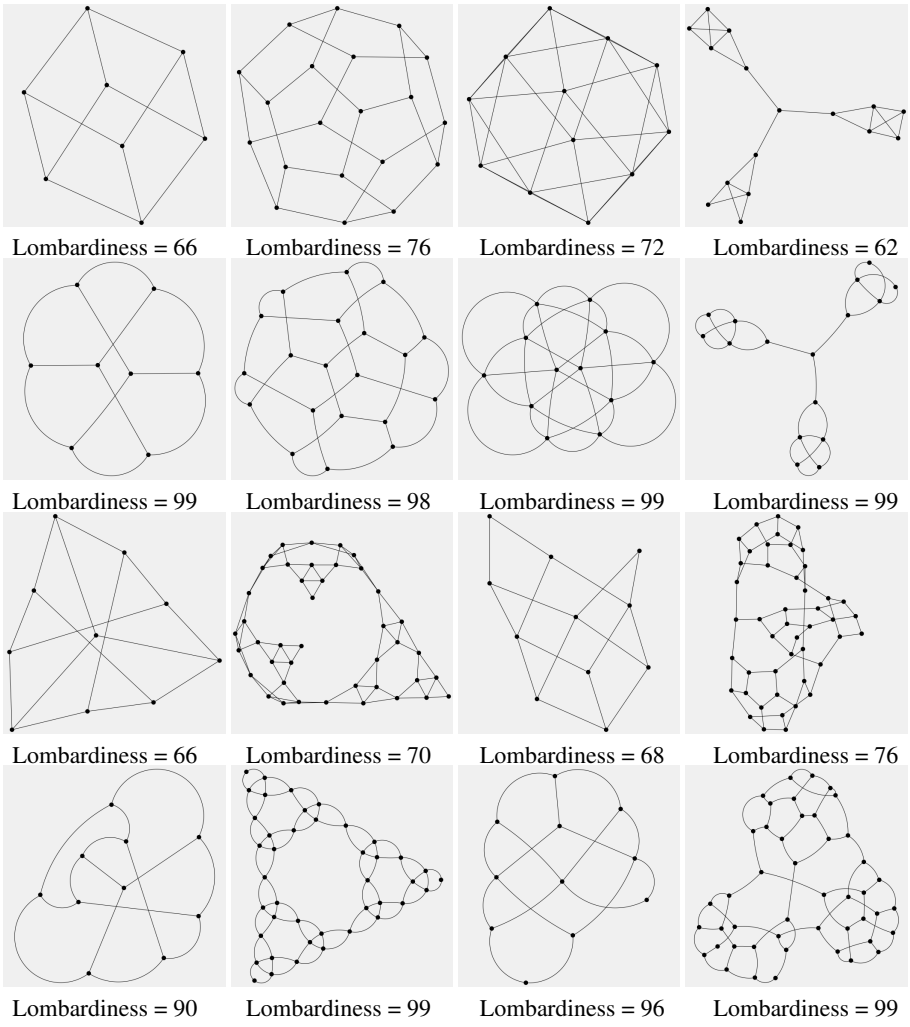


Fig. 8. Standard force-directed and near-Lombardi drawings for some standard graphs: Cube, Dodecahedron, Icosahedron, NoPerfectMatching, Petersen, Sierpinski, Herschel, and Tutte.