An Experimental Study on the Ply Number of Straight-line Drawings: *

F. De Luca[1], E. Di Giacomo[1], W. Didimo[1], S. Kobourov[2], G. Liotta[1]

[1] Università degli Studi di Perugia, Italy
felice.deluca@studenti.unipg.it,
{emilio.digiacomo,walter.didimo,giuseppe.liotta}@unipg.it
[2] University of Arizona, USA
kobourov@cs.arizona.edu

Abstract. The *ply number* of a drawing is a new criterion of interest for graph drawing. Informally, the ply number of a straight-line drawing of a graph is defined as the maximum number of overlapping disks, where each disk is associated with a vertex and has a radius that is half the length of the longest edge incident to that vertex. This paper reports the results of an extensive experimental study that attempts to estimate correlations between ply numbers and other aesthetic quality metrics for a graph layout, such as stress, edge-length uniformity, and edge crossings. We also investigate the performances of several graph drawing algorithms in terms of ply number, and provides new insights on the theoretical gap between lower and upper bounds on the ply number of k-ary trees.

1 Introduction

Graphs occur naturally in many domains: from sociology and biology, to software engineering and transportation. When the vertices and edges of the given graph have no inherent geographical locations, graph layout algorithms are used to try to capture the underlying relationships in the data (see, e.g., [8,10,22,32]).

In order to make the graph layout readable for the user, such algorithms are designed to optimize several quality metrics, like minimizing the number of edge crossings, striving for uniform edge lengths, or maximizing the vertex angular resolution [8]). *Force-directed* methods are among the most flexible and popular graph layout algorithms [24]. They tend to compute drawings that are aesthetically pleasing, exhibit symmetries, and contain no, or a few, edge crossings when the graph is planar. Classic examples include the spring layout method of Eades [12] and the algorithm of Fruchterman and Reingold [16], both of which rely on spring forces, similar to those in Hooke's law. In these methods, there are repulsive forces between all nodes and attractive forces between nodes that are adjacent. Alternatively, forces between the nodes can be computed

* Research supported in part by the MIUR project AMANDA "Algorithmics for MAssive and Networked DAta", prot. 2012C4E3KT_001. Work on this problem began at the NII Shonan Meeting *Big Graph Drawing: Metrics and Methods*, Jan. 12-15, 2015. We thank M. Kaufmann and his staff in the University of Tübingen for sharing their code to compute ply number. We also thank A. Wolff and F. Montecchiani for many useful discussions.

based on their graph theoretic distances, determined by the lengths of shortest paths between them. For instance, the algorithm of Kamada and Kawai [23] uses spring forces proportional to the graph theoretic distances. In general, force-directed methods define an objective function which maps each graph layout into a number in $\mathbb{R}^+$ representing the *energy* of the layout. This function is defined in such a way that low energies correspond to layouts in which adjacent nodes are near some pre-specified distance from each other, and in which non-adjacent nodes are well-spaced. The notion of the energy of the system is related to the notion of *stress* in multidimensional scaling [26]. A careful look into stress-based layout algorithms and force-directed algorithms shows that despite some similarities, they optimize a spectrum of different functions. For example, methods such as Kamada-Kawai optimize long graph distances, while Noack's LinLog layout [27] optimizes short graph distances. A later study by Chen and Buja explores further the differences between the energy models [5].

Recently, a new parameter, called *ply number*, has been proposed as a quality metric for graph layouts [9], partly inspired by the fact that real road networks have small values of such a parameter [13]. Roughly speaking, a drawing has a small ply number if some, suitably defined, regions of influence of the vertices in the drawing are well spread out. More precisely, let Γ be a straight-line drawing of a graph. For each vertex $v \in \Gamma$, let C_v be the *open* disk centered at v whose radius r_v is half the length of the longest edge incident to v. Denote by S_q the set of disks C_v sharing a point $q \in \mathbb{R}^2$. The *ply number* of Γ is defined as $\mathsf{pn}(\Gamma) = \max_{q \in \mathbb{R}^2} |S_q|$. In other words, the ply number of Γ is the maximum number of disks C_v mutually intersecting in Γ (see, e.g., Fig. 1). The ply number $\mathsf{pn}(G)$ of a graph G is the minimum ply number over all straight-line drawings of G. Figure 2 shows two drawings of the same graph. Intuitively, the drawing to the left is more readable than the drawing to the right and in fact the ply number of the left drawing is significantly smaller than the ply number of the right drawing.

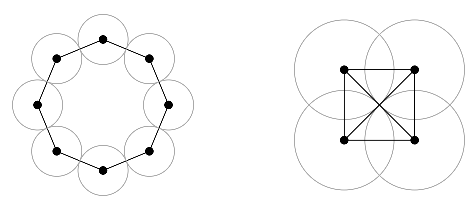


Fig. 1: (a) A drawing with ply number 1. (b) A drawing with ply number 2.

Our Contribution. While preliminary theoretical results about computing drawings with low ply number have already appeared [1,9], our work is an experimental study whose main goals can are: (*i*) to shed more light on the quality of drawings computed by some of the most popular algorithms, and in particular by different types of force-directed methods; (*ii*) to investigate whether the ply number of a drawing can be actually regarded as a quality metric, which possibly encompasses other popular metrics; (*iii*) to guide further theoretical studies of the combinatorial properties of drawings with low ply number. Specifically, our experiments involve several graph layout algorithms and several graph families, and we establish a correlation between ply number and some classical quality metrics like stress and edge length uniformity. Additionally, we give some insights about the known theoretical gap between lower and upper bounds for the ply number of k-ary trees.

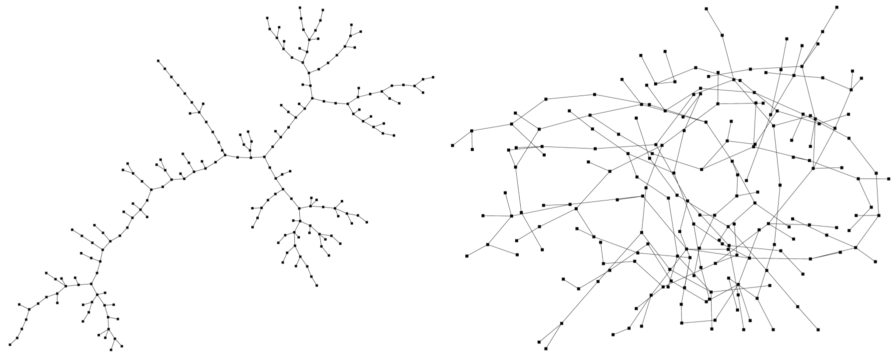


Fig. 2: Two drawings of the same graph with ply number 3 (left) and 12 (right).

The paper is structured as follows. Sec. 2-4 provides details about our experimental questions, setting, and procedures. The results of our study are presented and discussed in Sec. 5. Conclusions and future research directions are given in Sec. 6.

2 Experimental Questions

As mentioned in the introduction, our experiment has the following main objectives: (i) to shed more light on the quality of drawings computed by some of the most popular algorithms, with particular on force-directed methods; (ii) to investigate whether the ply number of a drawing can be regarded as a quality metric; (iii) to guide further theoretical studies of the combinatorial properties of drawings with low ply number. We pose the following experimental questions:

Q1. *How good are the layouts computed by different drawing algorithms in terms of ply number?*
Q2. *How close is the ply number of drawings produced by existing algorithms to the ply number of the input graph (i.e., to the optimum value)?*
Q3. *Does ply number correlate with some other commonly used quality metrics?*
Q4. *Can we establish empirical upper bounds on the ply number of k-ary trees?*

Questions Q1-Q3 are concerned with objective (i) and (ii), while Q4 is relevant for objective (iii). Below, we discuss the motivation behind each question in more detail.

The possibility that force-directed algorithms indirectly optimize the ply number has been suggested in [9]: With Q1 we compare force-directed algorithms based on different force models to experimentally investigate this hypothesis. In [9] it is observed that non-planar drawings may have significantly smaller ply number than planar ones. Hence, for planar graphs, we also consider algorithms that compute straight-line planar drawings in comparison with drawings computed by force-directed algorithms.

In addition to Q1, Question Q2 focuses on the quality of layout algorithms in terms of ply number. More precisely, it aims to estimate the gap between the ply number of drawings computed by existing algorithms and the optimum. However, computing the optimum value for the ply number over all drawings of a graph is NP-hard [9], and the (worst-case) optimum value of ply number is known only for simple graph families, like paths, cycles, binary trees, and caterpillars (whose optimum is either 1 or 2). We then restrict Q2 to these families.

Question Q3 is more focused on understanding whether the ply number can be used as a quality metric for graph layouts, which possibly encompasses several other popular quality measures. We are mainly interested in three measures that are among the most used in graph visualization and that we expect to affect (positively or negatively) the ply number more than others: number of crossings, stress, and edge-length uniformity. It is worth remarking that the number of crossings is widely adopted to evaluate the quality of graph layouts, especially for graphs of small and medium size (see, e.g., [21,29,30,34]). Studying the correlation between ply number and crossings, is further motivated by the fact that, as observed above, the ply number is sometimes reduced at the expense of edge crossings. The *stress* of a graph layout captures how well the realized geometric distances between pairs of vertices reflect their graph-theoretic distances in the graph; in the standard formulation, all edges are assumed to have have about the same length; thus stress is related to edge-length uniformity (see Sec. 3). Recent studies give some evidence that reducing the stress of a graph layout is correlated with improved aesthetics [11,25]. Studying the correlation between ply number, stress, and edge-length uniformity is also motivated by the fact that in a drawing with ply number one, all edges have the same length [9].

Question Q4 is motivated by objective (*iii*) and arises from theoretical results on the ply number of k-ary trees. Namely, it is known that every 2-ary (i.e., binary) tree has ply number at most two [9], while the ply number of 10-ary trees is not bounded by a constant [1]. What happens for values of k in the range $[3,9]$ is an interesting theoretical question. With Q4 we experimentally investigate this question.

3 Experimental Setting

In order to answer Questions Q1–Q4, we selected different graph datasets, algorithms, and measures. In what follows we describe each of these experimental components.

Graph datasets. To understand whether the experimental results are influenced by the structure of the graph we considered several graph families. All graphs are of small or medium size, expressed as the number of their vertices. In some cases, the size and the number of instances used for each graph family depends on the type of question we want to answer (see Sec. 5 for details). We used the following datasets:

Trees. Generated with uniform probability distribution using Prüfer sequences [28].
Planar. Connected simple planar graphs, generated with the OGDF library [6].
General. Connected simple graphs, generated with uniform probability distribution.
Scale-free. Scale-free graphs, generated according to the Barabási-Albert model [2].
Caterpillars. Each caterpillar of n vertices is generated by first creating a path (spine of the caterpillar) of length $k \in [\frac{n}{4}, \frac{n}{2}]$ (randomly chosen), and attaching each remaining vertex to a randomly selected vertex of the spine.
Paths, Cycles. For each desired number of vertices n, there is only one (unlabeled) path and one (unlabeled) cycle of n vertices.

Table 1: Table summarizing which datasets are used to answer each question.

	Q1	Q2	Q3	Q4
Trees	✓		✓	
Planar	✓		✓	
General	✓		✓	
Scale-free	✓		✓	
Caterpillars		✓		
Paths		✓		
Cycles		✓		
k-ary Trees ($k = 2$)		✓		
k-ary Trees ($k = 3, 6, 9$)				✓

k-ary Trees. Rooted trees where each node has either 0 or k children. Each tree is generated by starting with a single vertex and then creating k children of a randomly selected leaf, until the desired number n of vertices is achieved. When n cannot be obtained, we use a value close to it.

Table 1 shows which datasets are used to answer each question. Note that the datasets Caterpillars, Paths, Cycles, and 2-ary Trees are expressly used to answer Q2. All these families (except Cycles) are special cases of trees and we do not use them to answer Q1 and Q3. The dataset of k-ary Trees is expressly designed to answer Q4. See Sec. 4 for more details.

Algorithms. Among the many force-directed algorithms, we considered some of the most popular ones [24]. We used the following algorithms, available in OGDF:

FR. This algorithm is based on the Fruchterman-Reingold model [16], an improvement of the seminal algorithm by Eades [12]. Vertices are viewed as equally-charged electrical particles and edges act similar to springs; electrical charges cause repulsion between vertices and springs cause attraction. It also introduces a temperature function, which reduces the displacement of the vertices as the layout becomes better.

GEM. This algorithm is proposed by Frick et al. [15]. It is a variant of the FR algorithm, which adds several new heuristics to improve the convergence, including local temperatures, gravitational forces, and the detection of rotations and oscillations.

KK. This algorithm is described by Kamada-Kawai [23]. Unlike FR and GEM, it aims to compute a layout where the geometric distance between two vertices equals their graph-theoretic distance in the graph. The energy function minimized by this algorithm is therefore a type of stress function.

SM. This technique is proposed by Gansner et al. [17]. It minimizes a stress function similar to that proposed by KK, which can be minimized more efficiently via majorization.

FM3. The fast multipole multilevel method of Hachul and Jünger [18] is among the most effective force-directed algorithms in the literature [19].

We also used the following algorithm, available in Gephi [3]:

LL. This is a force-directed algorithm based on the LinLog energy model proposed by Noack [27]. It is specifically conceived to emphasize clusters in the graph.

For instances of Planar, Trees, and 2-ary Trees, we also considered planar straight-line drawing algorithms, still using the implementations in OGDF. The algorithms are:

PL. This is an improved version of the planar straight-line drawing algorithm proposed by Chrobak and Kant [7], based on the shift algorithm of de Fraysseix et al. [14].
TR. The tree layout algorithm of Buchheim et al. [4] is an efficient version of Walker's algorithm [33], which in turn is an extension of the Reingold-Tilford algorithm for rooted binary trees [31].

Measures. We considered four measures: ply number (PN), number of crossings (CR), stress (ST), and edge-length uniformity (EU). Let Γ be a straight-line drawing of a graph $G = (V,E)$. EU corresponds to the normalized standard deviation of the edge length, i.e.:

$$\mathtt{EU}(\Gamma) = \sqrt{\sum_{e\in E} \frac{(l_e - l_{avg})^2}{|E| l_{avg}^2}},$$

where l_e is the length of edge e and l_{avg} is the average length of the edges. The stress of Γ is defined as:

$$\mathtt{ST}(\Gamma) = \sum_{i,j\in V} w_{ij}(\| p_i - p_j \| - d_{ij})^2,$$

where $w_{ij} = d_{ij}^{-2}$, p_i and p_j are the positions of i and j in Γ, and d_{ij} is the graph theoretic distance of i and j in G.

4 Experimental Procedures

In the following we describe the different experimental procedures executed to answer each question. In Sec. 5 we present the results and summarize the main findings.

Procedure for Q1. We drew each instance of each dataset with all the algorithms (clearly, PL has been used only for Planar and TR only for Trees and 2-ary Trees), and measured the ply number of each drawing. Both for Trees and for Planar, we generated 10 instances for each fixed number of vertices $n \in \{50, 100, 150, 200, \dots, 450\}$: the average density of the planar graphs is 1.76. In the General dataset we generated 10 instances for each pair $\langle n, d\rangle$, where $n \in \{50, 100, 150, 200, \dots, 450\}$ is still the number of vertices of the graph and $d \in \{1.5, 2.5\}$ is its edge density. Indeed, we want also understand in which measure the ply number is influenced by the density of the graph. In the Scale-free dataset we generated 10 instances for each pair $\langle n, d\rangle$, where $n \in \{50, 100, 150, 200, \dots, 450\}$ and $d \in \{2, 3\}$ (the graph generator that we used required to specify an integer number as edge density [20]).

Procedure for Q2. To answer Q2 we need to compare the ply number of the drawings computed by the various algorithms with the ply number of the input graph (i.e., the optimum value). Hence, we considered families of graphs whose (worst-case optimal) ply number is known. In particular, we considered the Paths and Cycles instances, which have ply number one, and the Caterpillars and 2-ary Trees instances, whose ply number is at most two (see [9]). For each $n \in \{50, 100, 150, 200, \ldots, 450\}$, we generated a single instance in Paths and Cycles, and 5 instances in the Caterpillars and 2-ary Trees datasets. For each instance, we computed 10 different drawings with the algorithms KK, FM3, SM (those with better performances based on the results of Q1). We then took the minimum value of ply number over all the drawings of each instance.

Procedure for Q3. We took a representative instance for each sample (i.e., size or size and density) of each dataset. As reported in Table 1, we used the same datasets as for Q1. For each representative instance we computed a series of 60 different drawings and on this series we measured Spearman's rank correlation coefficient ρ between ply number and all the other quality metrics described in Sec. 3, i.e., ST, CR, and EU. The series of drawings are produced by running 10 times each of the 6 force-directed algorithms, varying the initial layout every time.

Procedure for Q4. We generated a k-ary tree for each pair $\langle n, k \rangle$, where $n = \{100, 150, 200, \ldots, 950, 1000, 2000, 3000, 4000\}$ is the desired number of vertices and $k \in \{3, 6, 9\}$. The choice for the values of k is motivated by the fact that we want to experimentally understand if we can empirically establish a constant upper bound to the ply number of k-trees for $2 < k < 10$. For each instance, we measured the ply number of a drawing computed with SM, which turned out to be the best performing algorithm for this measure, according to the experiment for Question Q1.

5 Experimental Results and Findings

We first report the experimental data, by presenting tables and charts. Then, we list and discuss the main findings.

Data for Q1. Figure 3 reports the average ply number for each sample (number of vertices) of drawings computed by the same algorithm. For Trees and Planar we observed a similar trend. The algorithms that give the lowest values of the ply number are SM, KK, and FM3 (with KK and SM that always have almost identical values and FM3 that is slightly better for larger planar graphs). FR produces drawings with ply number higher than the previous three algorithms, although it has a similar trend. GEM and LL produces drawings with ply number much higher than the other force-directed algorithms, with GEM that becomes worse than LL as n grows. For Trees the TR algorithm has quite good performances (between FM3 and FR), while the algorithm PL (for the Planar dataset) produces the drawings with the worse values of ply, thus confirming that planar drawings often have higher ply number than non-planar ones. For the General and Scale-free datasets, we have a similar situation. The main difference is that GEM performances do not worsen as fast as in the cases of Trees and Planar (in particular, it is always better than LL). For the Scale-free dataset, the values computed by KK and SM increase, as n grows, more than those of FR and FM3. Thus, for larger values of n they are outperformed by FR and FM3, and approach the performances of GEM.

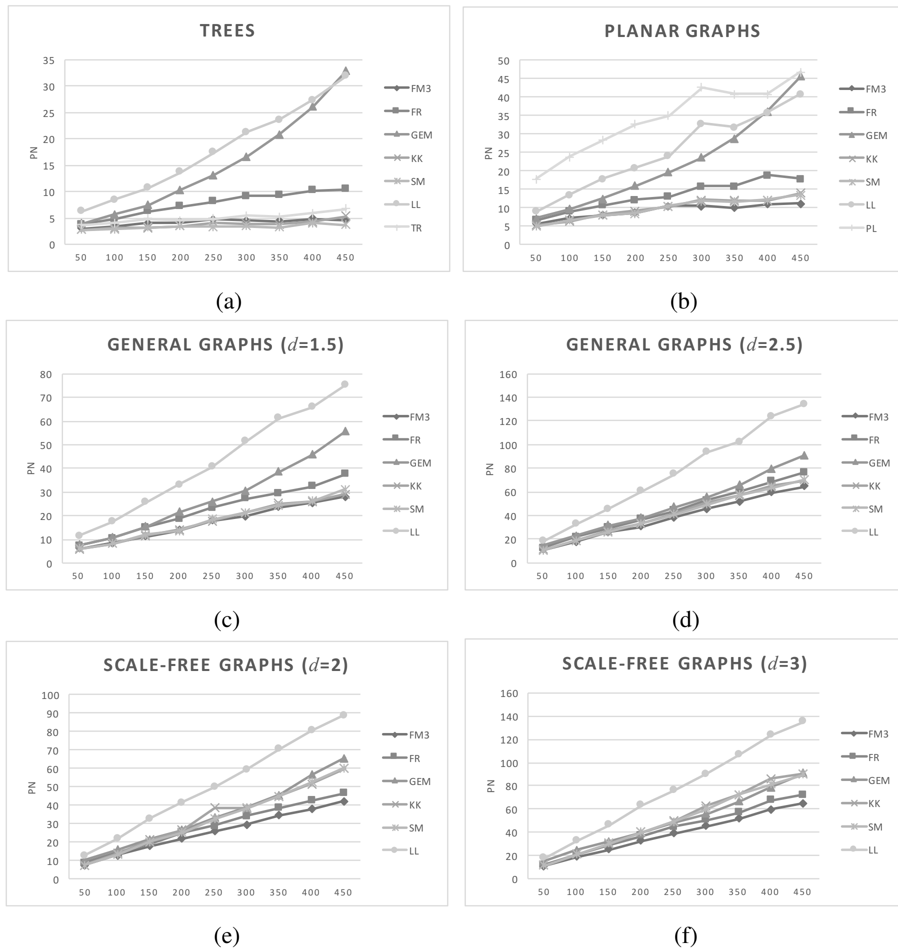


Fig. 3: Average ply number for (a) Trees; (b) Planar; (c) General with density $d = 1.5$; (d) General with density $d = 2.5$; (e) Scale-free with density $d = 2$; (f) Scale-free with density $d = 3$. The x-axis reports the number of vertices.

Data for Q2. For paths and cycles, all three algorithms compute drawings with ply number 2, i.e., just one unit larger than the optimal value, for instances up to 250. For larger sizes, FM3 still computes drawing with ply number 2, while KK and SM produces drawing with ply number 3. It is worth saying that all the drawings computed by FM3 have ply number 2 with the only exception of one drawing of the cycle of size 450. This is consistent with the fact that FM3 (a multilevel algorithm) is less affected by the initial position of the vertices. The maximum value of ply for drawings produced by KK and SM is 6 and 4, respectively. Figure 6(a) (in appendix) shows, for the binary trees, the average ply number of the drawings computed by the different algorithms for the different sizes. In this case, FM3 has the worst performance with an average ply number ranging from 3 to 5.2; the other two algorithms have almost the same values of the average ply number, ranging from 2 to 4.2. In the chart we also show (with error bars) the (average) maximum ply number of each algorithm: FM3 and KK tend to have larger differences between the (average) maximum and the (average) minimum ply number than SM. Figure 6(b) (in appendix) shows the same data for caterpillars. In this case, the three algorithms have similar performances. The average ply number is around 3.5. Also in this case SM is the algorithm with the smallest difference between maximum and minimum error.

Data for Q3. Table 2 shows, for each instance, the values of the Spearman's rank correlation coefficient ρ. We have high values of correlation (i.e., $\rho \geq 0.7$) between ply number and stress for almost every instance. The exceptions are larger scale-free graphs, for which in most cases we have a moderate correlation (i.e., $0.3 \leq \rho < 0.7$). The correlation between PN and EU is high/moderate in all cases. High values of correlation are obtained for the smaller instances of each dataset, with the only exception of scale-free graphs, where there is a high correlation for all sizes. Concerning PN and CR, we have (high/moderate) correlation only for trees, planar graphs, and for general graphs with density 1.5. For denser general graphs and for scale-free graphs there is little correlation between PN and CR. Figure 4 (in appendix) plots the relationships between ply number and stress, for the instances with $n = 100$ of each dataset.

Table 2: Correlation coefficient ρ between PN and ST, CR, EU. The values in bold indicate a strong correlation ($\rho \geq 0.7$).

	n	PN,ST	PN,CR	PN,EU		n	PN,ST	PN,CR	PN,EU
Trees	50	**0.92**	**0.88**	**0.86**	General ($d = 2.5$)	50	**0.80**	0.19	**0.89**
	100	**0.90**	**0.84**	**0.89**		100	**0.80**	0.21	**0.72**
	150	**0.94**	**0.92**	**0.78**		150	**0.78**	0.20	0.62
	200	**0.97**	**0.89**	**0.87**		200	**0.79**	0.04	0.54
	250	**0.96**	**0.91**	0.69		250	**0.78**	0.47	0.58
	300	**0.97**	**0.92**	0.65		300	**0.71**	-0.34	0.63
	350	**0.93**	**0.90**	0.62		350	**0.70**	-0.19	0.59
	400	**0.96**	**0.90**	0.61		400	**0.73**	-0.13	0.55
	450	**0.95**	**0.96**	0.51		450	**0.72**	-0.19	0.54
Planar	50	**0.92**	**0.80**	**0.76**	Scale-free ($d = 2$)	50	**0.86**	0.29	**0.74**
	100	**0.91**	**0.83**	**0.86**		100	**0.83**	0.08	**0.80**
	150	**0.90**	0.66	**0.89**		150	0.32	-0.08	**0.82**
	200	**0.77**	**0.87**	**0.71**		200	0.57	0.33	**0.72**
	250	**0.87**	**0.83**	**0.81**		250	0.38	0.27	**0.90**
	300	**0.93**	**0.80**	**0.80**		300	0.21	0.09	**0.90**
	350	**0.80**	**0.91**	**0.74**		350	0.48	0.53	**0.82**
	400	**0.77**	**0.93**	0.67		400	0.45	0.60	**0.84**
	450	**0.76**	**0.84**	0.58		450	0.50	0.54	**0.80**
General ($d = 1.5$)	50	**0.88**	0.39	**0.87**	Scale-free ($d = 3$)	50	**0.72**	0.15	**0.91**
	100	**0.92**	**0.80**	**0.82**		100	0.67	-0.04	**0.88**
	150	**0.89**	**0.81**	**0.86**		150	0.54	-0.32	**0.87**
	200	**0.87**	**0.83**	**0.89**		200	0.47	-0.16	**0.87**
	250	**0.85**	0.60	**0.84**		250	0.30	0.13	**0.90**
	300	**0.85**	**0.73**	**0.73**		300	0.18	0.27	**0.95**
	350	**0.84**	**0.85**	**0.85**		350	0.14	0.36	**0.96**
	400	**0.85**	**0.79**	0.49		400	0.29	0.22	**0.92**
	450	**0.88**	**0.86**	0.59		450	0.30	0.41	**0.90**

Data for Q4. Figure 7 (in appendix) reports the values of ply number for each k and for increasing values of n. For the sizes of the trees that we considered, we always observed progressively increasing ply numbers, even for $k = 3$, and the ply number function does not exhibit any asymptotic trend towards a constant upper bound.

We summarize the main findings.

F1. Algorithms designed to minimize stress and edge-length uniformity (like SM and KK) compute drawings with smaller values of ply number. This behavior confirms the intuition that low ply number is related to stress and edge uniformity optimization (see also F3). Also multilevel algorithms (like FM3) have good performances and are more stable on denser graphs. We think this is a consequence of their coarsening phase, which indirectly tends to evenly distribute the vertices in the plane, thus producing drawings with good edge length uniformity, independently of the original placement of the nodes. Force-directed algorithms whose energy model is conceived to highlight clusters, such as LL, tend to produce drawings with high ply numbers, as they give rise to very different edge lengths in the same drawing.

F2. The best performing algorithms in terms of ply number very often generate drawings whose ply number is close to the optimum for graphs like paths, cycles, caterpillars, and binary trees. Hence, they can be considered good heuristics to compute graph layouts with minimum ply number, at least for these simple graph families.

F3. There is a strong correlation between ply and stress, and a strong/moderate correlation between ply and edge-length uniformity. For planar graphs and low density graphs, the correlation between ply and crossings is also observed, while ply is definitely non-correlated with edge crossings on denser graphs and, in particular, on scale-free graphs. Overall, these findings indicate that ply number can be often regarded as a unifying quality metric, which encompasses at least stress and edge length uniformity. For very sparse graphs, it also encompasses edge crossings. Note that, the correlation between ply number and stress does not always imply that low ply number equals low stress. For example, Fig. 5 (in appendix) shows a drawing with low ply number and high stress, and a drawing with low stress and high ply number.

F4. We could not observe any asymptotic trend of the ply number towards a constant upper bound for k-ary trees ($k \in \{3,6,9\}$). This indicates that the ply number for such graphs is likely unbounded, which should be confirmed by a theoretical proof.

6 Conclusions and Future Work

Our graph datasets and the data collected in the different experiments are publicly available at `http://www.felicedeluca.com/ply/`. These data answer, or partially answer, several of our initial questions and raise new interesting questions for further study.

We remark that, as in many experimental studies, ours has some limitations and should be interpreted in the context of the specific datasets, layout algorithms, and measurements used. For example, a more complete picture can be obtained with a more diverse set of graphs. In particular, computing the ply number of a drawing in a reliable way requires high arithmetic precision, which is computationally expensive. This limited the sizes of graphs that we could consider. Further, we mostly used algorithms available in OGDF, considering only one other algorithm. Comparing a wider spectrum of algorithms might help to identify what type of stress-minimization and energy minimization functions are best suited to minimize the ply number.

Our study provides answers several of the questions that we asked and also suggests several natural research directions that remain to be explored:

– Can new layout algorithms be developed that directly optimize ply number? Modifying stress-based methods such as Kamada-Kawai would be difficult, but perhaps force-directed methods such as Fruchterman-Reingold can be augmented with additional forces to separate overlapping disks.

– Considering more carefully the variations in the exact functions used in different stress-based methods and force-directed methods and their impact on optimizing ply might lead to better insights about how to compute low-ply layouts.

– There is growing evidence that different types of graph layout algorithms are suited to different types of graphs. A cognitive study could consider the impact of minimizing ply number, compared to the impact of minimizing edge crossings and other aesthetic criteria that are not correlated with the ply number.

– More experiments can be performed to look for possible correlations between ply and other quality metrics, such as angular resolution and drawing symmetries.

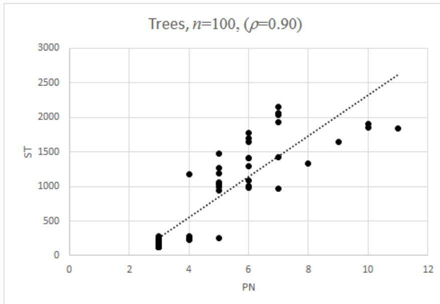
– Further experiments may help to identify graph families with constant or unbounded ply number. Experimental data can then be formally verified. In particular, our experiments indicate that k-ary trees, with $k \in [3,9]$ have unbounded ply number; proving this would close a gap in our theoretical knowledge.

References

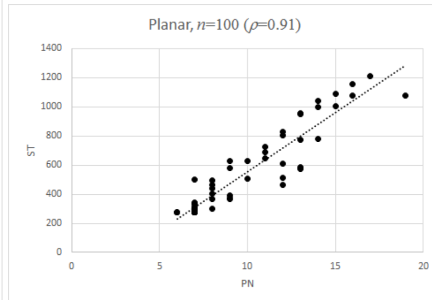
1. Angelini, P., Bekos, M., Bruckdorfer, T., Hančl, J., Kaufmann, M., Kobourov, S., Kratochvíl, J., Symvonis, A., Valtr, P.: Low ply drawings of trees. In: Nöllenburg, M., Hu, Y. (eds.) GD 2016. LNCS (to apper)
2. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science 286(5439), 509–512 (1999)
3. Bastian, M., Heymann, S., Jacomy, M.: Gephi: An open source software for exploring and manipulating networks (2009), `http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154`
4. Buchheim, C., Jünger, M., Leipert, S.: Drawing rooted trees in linear time. Software: Practice and Experience 36(6), 651–665 (2006)
5. Chen, L., Buja, A.: Stress functions for nonlinear dimension reduction, proximity analysis, and graph drawing. The Journal of Machine Learning Research 14(1), 1145–1173 (2013)
6. Chimani, M., Gutwenger, C., Jünger, M., Klau, G.W., Klein, K., Mutzel, P.: The open graph drawing framework (OGDF). In: Tamassia, R. (ed.) Handbook of Graph Drawing and Visualization, pp. 543–569. CRC Press (2013)
7. Chrobak, M., Kant, G.: Convex grid drawings of 3-connected planar graphs. Int. J. of Computational Geometry & Applications 07(03), 211–223 (1997)
8. Di Battista, G., Eades, P., Tamassia, R., Tollis, I.G.: Graph Drawing. Prentice Hall, Upper Saddle River, NJ (1999)
9. Di Giacomo, E., Didimo, W., Hong, S.H., Kaufmann, M., Kobourov, S.G., Liotta, G., Misue, K., Symvonis, A., Yen, H.C.: Low ply graph drawing. In: IISA 2015. pp. 1–6. IEEE (2015)

10. Didimo, W., Liotta, G.: Mining graph data. In: Cook, D.J., Holder, L.B. (eds.) Graph Visualization and Data Mining, pp. 35–64. Wiley (2007)
11. Dwyer, T., Lee, B., Fisher, D., Quinn, K.I., Isenberg, P., Robertson, G.G., North, C.: A comparison of user-generated and automatic graph layouts. IEEE Transanction on Visualization and Computer Graphics 15(6), 961–968 (2009)
12. Eades, P.: A heuristics for graph drawing. Congressus numerantium 42, 146–160 (1984)
13. Eppstein, D., Goodrich, M.T.: Studying (non-planar) road networks through an algorithmic lens. In: GIS 2008. pp. 1–10. ACM (2008)
14. de Fraysseix, H., Pach, J., Pollack, R.: How to draw a planar graph on a grid. Combinatorica 10(1), 41–51 (1990)
15. Frick, A., Ludwig, A., Mehldau, H.: A fast adaptive layout algorithm for undirected graphs. In: Tamassia, R., Tollis, I.G. (eds.) GD ' 94. LNCS, vol. 894, pp. 388–403. Springer (1995)
16. Fruchterman, T.M.J., Reingold, E.M.: Graph drawing by force-directed placement. Software: Practice and Experience 21(11), 1129–1164 (1991)
17. Gansner, E.R., Koren, Y., North, S.C.: Graph drawing by stress majorization. In: Pach, J. (ed.) GD 2004. LNCS, vol. 3383, pp. 239–250. Springer (2004)
18. Hachul, S., Jünger, M.: Drawing large graphs with a potential-field-based multilevel algorithm. In: Pach, J. (ed.) GD 2004. LNCS, vol. 3383, pp. 285–295. Springer (2004)
19. Hachul, S., Jünger, M.: Large-graph layout algorithms at work: An experimental study. Journal of Graph Algorithms and Applications 11(2), 345–369 (2007)
20. Hagberg, A.A., Schult, D.A., Swart, P.J.: Exploring network structure, dynamics, and function using networkx. In: Varoquaux, G., Vaught, T., Millman, J. (eds.) Proceedings of the 7th Python in Science Conference. pp. 11 – 15. Pasadena, CA USA (2008)
21. Huang, W., Hong, S.H., Eades, P.: Effects of sociogram drawing conventions and edge crossings in social network visualization. J. Graph Algorithms Appl. 11(2), 397–429 (2007)
22. Jünger, M., Mutzel, P. (eds.): Graph Drawing Software. Springer (2003)
23. Kamada, T., Kawai, S.: An algorithm for drawing general undirected graphs. Information Processing Letters 31(1), 7 – 15 (1989)
24. Kobourov, S.G.: Force-directed drawing algorithms. In: Tamassia, R. (ed.) Handbook of Graph Drawing and Visualization, pp. 383–408. CRC Press (2013)
25. Kobourov, S.G., Pupyrev, S., Saket, B.: Are crossings important for drawing large graphs? In: GD 2014. LNCS, vol. 8871, pp. 234–245. Springer (2014)
26. Kruskal, J.B., Wish, M.: Multidimensional Scaling. Sage Press (1978)
27. Noack, A.: Energy models for graph clustering. Journal of Graph Algorithms and Applications 11(2), 453–480 (2007)
28. Prüfer, H.: Neuer beweis eines satzesüber permutationen. Arch. Math. Phys. 27, 742–744 (1918)
29. Purchase, H.C.: Effective information visualisation: a study of graph drawing aesthetics and algorithms. Interacting with Computers 13(2), 147–162 (2000)
30. Purchase, H.C., Carrington, D.A., Allder, J.A.: Empirical evaluation of aesthetics-based graph layout. Empirical Software Engineering 7(3), 233–255 (2002)
31. Reingold, E.M., Tilford, J.S.: Tidier drawings of trees. IEEE Transactions on Software Engineering SE-7(2), 223–228 (March 1981)
32. Tamassia, R. (ed.): Handbook of Graph Drawing and Visualization. CRC Press (2013)
33. Walker, J.Q.: A node-positioning algorithm for general trees. Software: Practice and Experience 20(7), 685–705 (1990)
34. Ware, C., Purchase, H.C., Colpoys, L., McGill, M.: Cognitive measurements of graph aesthetics. Information Visualization 1(2), 103–110 (2002)

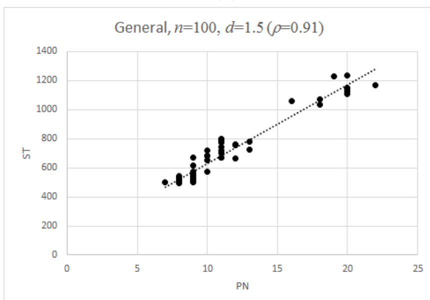
A Additional material



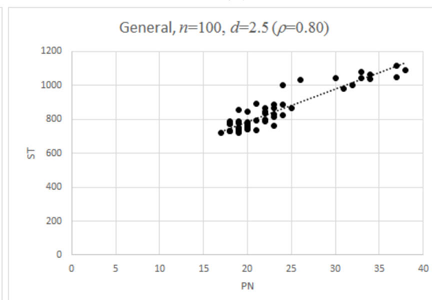
(a)



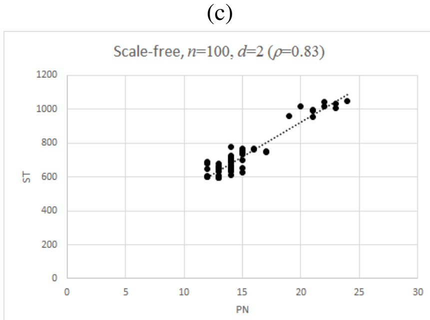
(b)



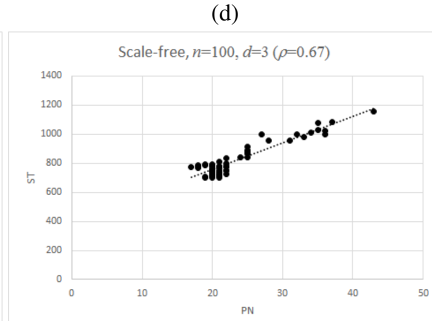
(c)



(d)

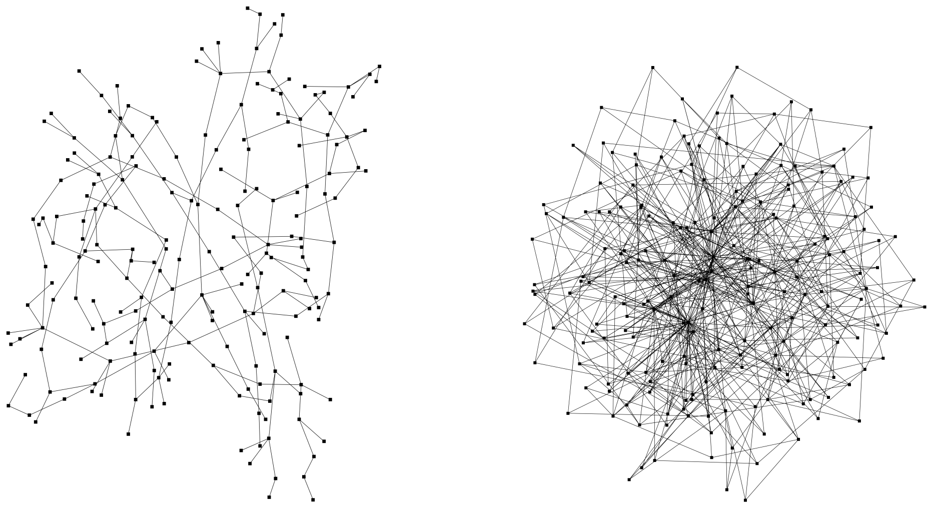


(e)



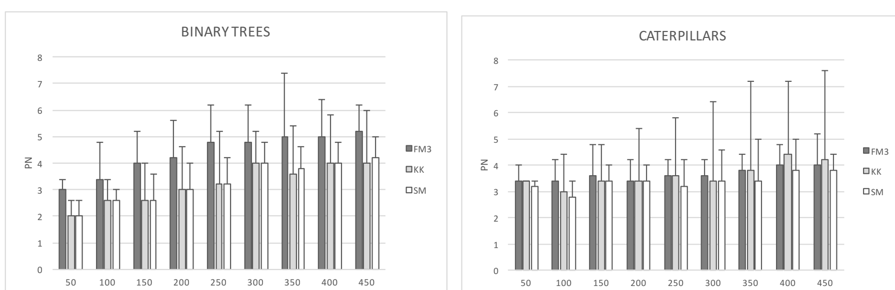
(f)

Fig. 4: Relationships between ply number (PN) and stress (ST), for the instances with $n = 100$ of each dataset.



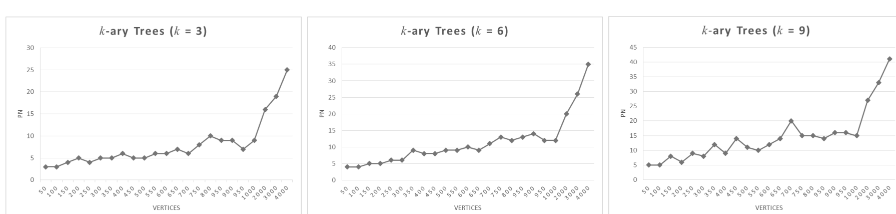
(a) PN = 8; ST = 9178. (b) PN = 47; ST = 3297.

Fig. 5: Drawings of two graphs with 200 vertices. (a) Low ply number and high stress. (b) Low stress and high ply number.



(a) (b)

Fig. 6: Average ply number for (a) Binary trees; (b) Caterpillars;



(a) (b) (c)

Fig. 7: Behavior of the ply numbers on k-ary trees.