

# Supplemental Material for The Sprawltter Graph Readability Metric: Combining Sprawl and Area-aware Clutter

Online Submission ID: 7108

September 2019

## Contents

<b>S1 Parameter analysis</b>	<b>2</b>
S1.1 Minimum penalty fraction	2
S1.1.1 Theoretical analysis	2
S1.1.2 Computational analysis	2
S1.2 EE curve shape	4
S1.2.1 Theoretical analysis	4
S1.2.2 Computational analysis	4
<b>S2 Pseudo-code for NE and EE area-aware metric</b>	<b>6</b>
<b>S3 List of graph layouts</b>	<b>7</b>
<b>S4 Computational time</b>	<b>8</b>
<b>S5 Computational pipeline</b>	<b>9</b>

## S1 Parameter analysis

We present the analysis for two parameters in the penalty mapping functions that are defined in the paper:

$$f_{v_1, v_2}^{NN}(x) = (1 - \alpha)(2x)^{0.7} + \alpha M_{v_1, v_2}^{0.7} \quad (0 \leq x \leq M_{v_1, v_2}) \quad (1)$$

$$f_{v, e}^{NE}(x) = 2(1 - \alpha)x + \alpha M_{v, e} \quad (0 \leq x \leq M_{v, e}) \quad (2)$$

$$f_G^{EE}(x) = \left(\frac{16}{\pi^2} - 4\alpha\right)x^2 + \alpha \frac{\pi^2}{4} \quad (0 \leq x \leq \frac{\pi}{2}) \quad (3)$$

### S1.1 Minimum penalty fraction: $\alpha$

In Sect 4.4 of the main paper, we describe the semantics of the minimum penalty fraction,  $\alpha$ , that appears in Equation 4. The minimum penalty fraction is a trade-off between the ability to distinguish no overlap from touching overlap and the ability to distinguish the different amount of overlap.

$$\begin{aligned} (NN) \quad \beta &= \left(1 - \frac{1}{.5^{.7}}\right)\alpha + \frac{1}{.5^{.7}} \approx 1.625 - 0.625\alpha \quad (0 < \alpha < 1) \\ (NE) \quad \beta &= 2 - \alpha \quad (0 < \alpha < 1) \\ (EE) \quad \beta &= \frac{16}{\pi^2} - 3\alpha \quad (0 < \alpha < \frac{16 - 2\pi}{3\pi^2} \approx 0.328) \end{aligned} \quad (4)$$

#### S1.1.1 Theoretical analysis

We show the function plot for NN overlap between bigger nodes, as mentioned in Sect.4.4 in the main paper.

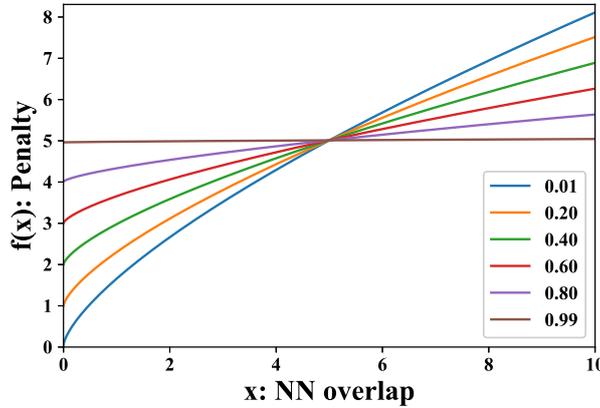


Figure S1: The NN penalty mapping function of a big metanode pair that is 10x the size of the smallest node ( $M = 10$ ), with different minimum penalty fraction ( $\alpha$ ) values.

#### S1.1.2 Computational analysis

We validate the theoretical analysis and generate practical suggestions with our implementation and three groups of synthetic small graph layouts, namely, progression-NN, progression-NE, and progression-EE.

We tested the influence of  $\alpha$  for NN, NE, and EE separately as it is not necessary to use the same  $\alpha$  for all three cases. We used six different  $\alpha$  values within its valid range defined in Equation 4: a near-minimum, a near-maximum, and four evenly-spaced values in between.

The progression-NN has a series of graph layouts with minimum, some, and near-max node-node overlap for either leaf nodes only, metanodes only, or both, as shown in Table S1. For each  $\alpha$ , we can compare the area-aware metrics between different layouts to obtain a practical sense of whether the differences in penalties are too little, too much, or just enough to distinguish the layouts. The same analysis also apply to NE (Table S2) and EE (Table S3).

With the extremely small  $\alpha$  value 0.01, the near-minimum penalty for the touching overlap between leaf nodes are only 0.5 (average penalty is  $P/C = 0.50/4 = 0.125$ ), which is too little to distinguish it from no overlap at all. With  $\alpha$  values that are larger than 0.8, the differences between penalties for touching, some, and near-max overlap of leaf nodes are too tiny ( $3.29 \rightarrow 3.91 \rightarrow 4.24$  when  $\alpha = 0.80$ ), and differences for metanodes are also too tiny ( $12.40 \rightarrow 15.34$ ). Note that the last difference seems sufficient ( $15.34 \rightarrow 27.24$ ) only because the last layout has overlaps of both leaf node pairs and metanode pairs. With  $\alpha$  values in between, the differences between penalties are generally sufficient to tell these layouts apart.

The choice of  $\alpha$  is subjective to user preference, and they can have different  $\alpha$  values for NN, NE, and EE, as long as it is within the range specified by Equation 4. For simplicity, we use  $\alpha = 0.20$  for all three cases.

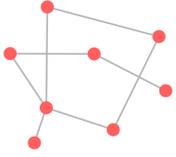
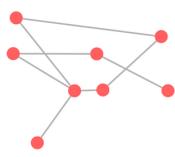
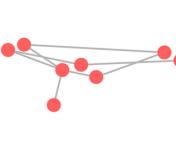
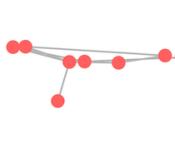
Table S1: Influence of minimum penalty fraction ( $\alpha$ ) tested on a small 2-level synthetic layout with increasing node-node overlap of leaf node pairs and metanode pairs.

Overlap amount		Near-min		Some		Near-max	
Leaf node pairs or metanode pairs		Leaf nodes	Metanodes	Leaf nodes	Metanodes	Leaf nodes	Both
Layout							
Count		4	1	4	1	4	12
Area-aware metrics (variable: $\alpha$ )	0.01	0.50	1.69	3.57	14.05	5.17	31.69
	0.20	1.17	4.27	3.66	14.36	4.95	30.62
	0.40	1.88	6.98	3.74	14.69	4.71	29.49
	0.60	2.59	9.69	3.83	15.01	4.47	28.37
	0.80	3.29	12.40	3.91	15.34	4.24	27.24
	0.99	3.96	14.97	4.00	15.65	4.01	26.17

Table S2: Influence of minimum penalty fraction ( $\alpha$ ) tested on a small 2-level synthetic layout with increasing node-edge overlap of leaf node-edge pairs and metanode-edge pairs.

Overlap amount		Near-min		Some		Near-max	
Leaf node pairs or metanode pairs		Leaf nodes	Metanodes	Leaf nodes	Metanodes	Leaf nodes	Both
Layout							
Count		4	6	5	15	5	36
Area-aware metrics (variable: $\alpha$ )	0.01	3.38	12.15	7.27	56.76	9.27	149.33
	0.2	3.50	12.11	6.83	54.46	8.45	136.72
	0.4	3.62	12.07	6.37	52.04	7.59	123.45
	0.6	3.75	12.03	5.92	49.62	6.72	110.18
	0.8	3.87	11.99	5.46	47.20	5.86	96.91
	0.99	3.99	11.96	5.02	44.90	5.04	84.30

Table S3: Influence of minimum penalty fraction ( $\alpha$ ) tested on a small single-level synthetic layout with decreasing edge-edge crossing angles.

Overlap amount		Near-min	Some	Small angle	Near-max
Layout					
Count		2	2	2	2
Area-aware metrics (variable: $\alpha$ )	0.01	0.05	0.97	4.79	6.67
	0.07	0.35	1.12	4.37	5.96
	0.13	0.64	1.28	3.94	5.25
	0.20	0.99	1.46	3.45	4.42
	0.26	1.28	1.62	3.03	3.72
	0.328	1.62	1.80	2.55	2.91

## S1.2 EE curve shape: $\gamma$

For the EE case, we compare the quadratic function (Equation 3) against a linear function, stated as follows.

$$f_G^{EE}(x) = \left(\frac{4}{\pi} - 2\alpha\right)x + \alpha\frac{\pi}{2} \quad (0 \leq x \leq \frac{\pi}{2}), \quad \text{where } 0 < \alpha < \frac{2}{\pi} \approx 0.637 \quad (5)$$

### S1.2.1 Theoretical analysis

To separate the influence of  $\alpha$  and  $\gamma$ , we compare the EE linear function with the quadratic one using the same  $\alpha$ , but repeat the comparison for multiple different  $\alpha$  values, as shown in Figure S2. Since the domain of  $\alpha$  is different for linear function,  $(0, 0.637)$ , and quadratic function,  $(0, 0.328)$ , we only compare the functions with valid  $\alpha$  values for both, i.e., 0.01, 0.1, 0.2, 0.3.

We observe that the basic shapes and relationships of the two functions stay the same across different  $\alpha$  values, despite that the touching penalty and range of penalty vary (in the way we described in the previous section). The absolute difference between the two functions decreases as  $\alpha$  increases, due to the decrease in the range of penalty. The relative difference between two functions becomes large only when the complementary angle  $x$  is nearly maximum,  $\pi/2$ , i.e., the glancing angle; otherwise, they are very close to each other.

Therefore, we can choose either the linear or quadratic function depending on how heavy the glancing angle should be penalized.

### S1.2.2 Computational analysis

We computed the area-aware metrics using both linear and quadratic penalty mapping functions on all 52 layouts we collected. Figure S3 shows the penalties computed with the linear function (x-axis) compared to that with the quadratic function (y-axis). Each dot represents a graph layout in our dataset. We used log scales for both axes to de-clutter the dots as there are many small synthetic layouts. We can clearly see that the two penalties are linearly correlated. The Pearson correlation between the two penalties is 0.9997, and the slope of the linear regression line is 1.17, which indicates that penalties with the quadratic function are slightly bigger. In general, the difference between the two functions are very small.

We also sorted the layouts by the difference between the penalties using the two functions, and manually inspected a few layouts with the biggest difference. We found out that in these layouts, the angles between crossing edges are smaller than those in other layouts, and there are many near-glancing angles. This discovery also confirms our theoretical analysis: difference only becomes relatively large on small crossing angles (i.e., large complementary angles).

We chose the quadratic function for our analysis in the main paper as Huang et al. found that the quadratic trend is more relevant to user performance in their study [1].

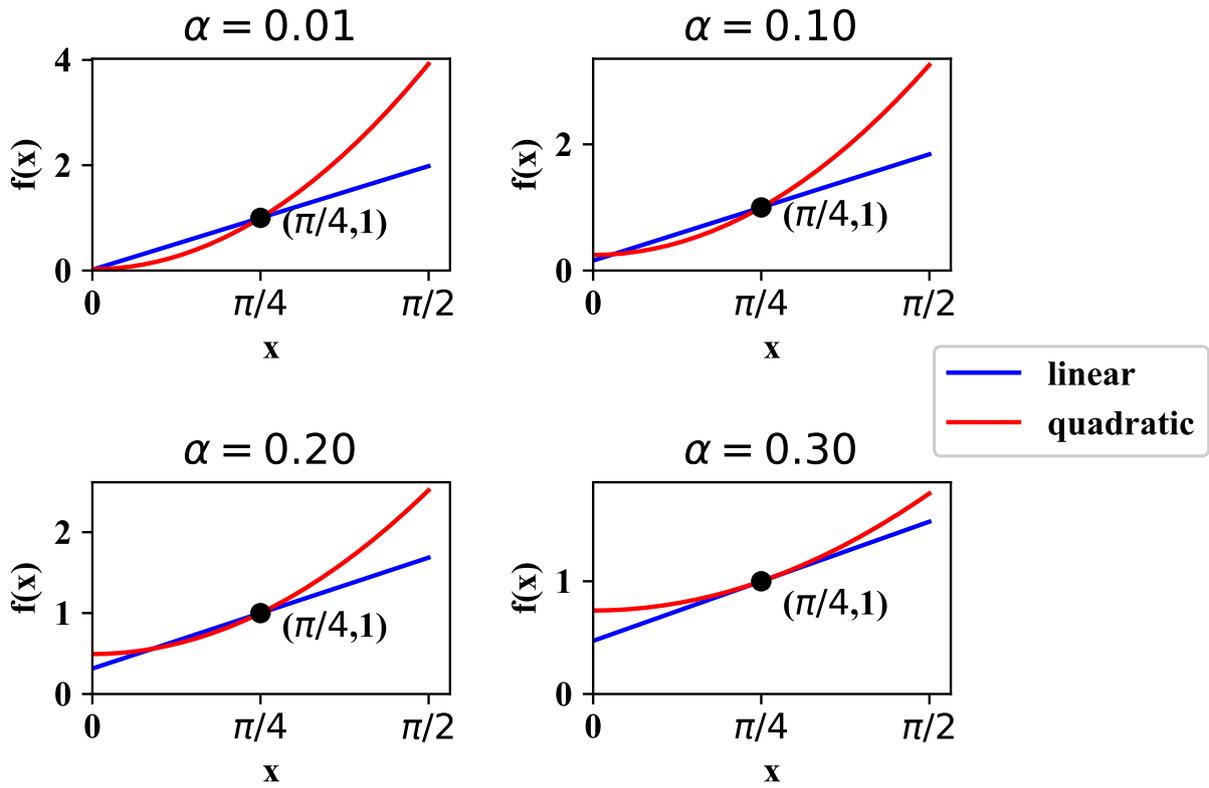


Figure S2: Two different curve shapes (linear and quadratic) of the EE penalty mapping function.

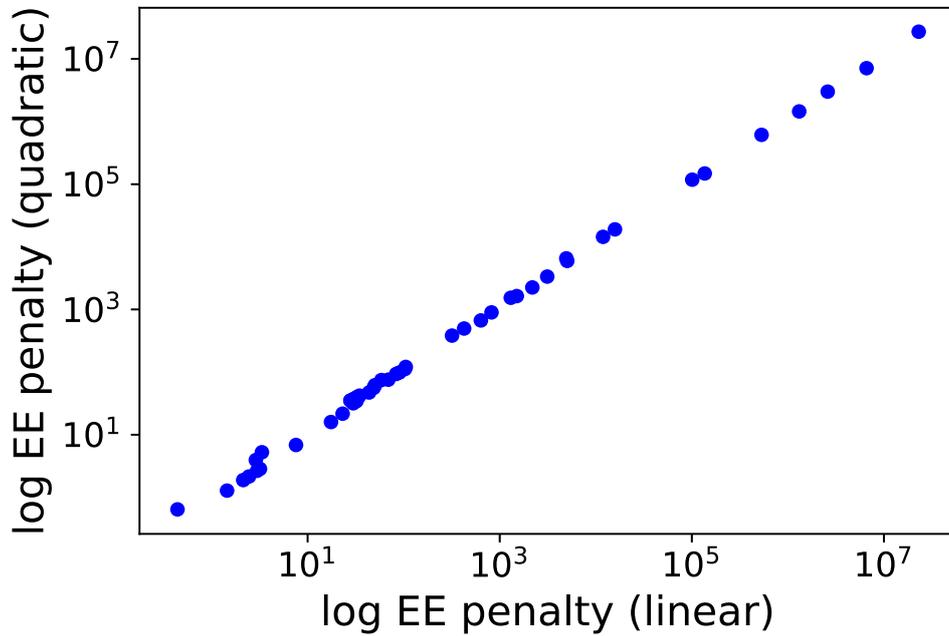


Figure S3: EE area-aware metric with linear penalty mapping function against that with the quadratic function, using log scale on both axes.

## S2 Pseudo-code for NE and EE area-aware metric

---

**Algorithm 1:** Computation of node-edge area-aware metrics.

---

**Input :**  $G = (V, E)$   
**Output:** total penalty  $P^{NE}(G)$  and count  $C^{NE}(G)$

```
1 totalPenalty  $\leftarrow$  0
2 count  $\leftarrow$  0
3 for  $v \in V$  do
4   for  $e \in E$  do
5     if  $v \notin e.ends \ \&\& \ !IsAncOrDesc(v, e.ends)$  then
6       if CheckIntersection( $v, e$ ) then
7          $x \leftarrow$  ComputeOverlapLength( $v, e$ )
8         penalty  $\leftarrow$  PenaltyMapFunc( $x, v, e$ )
9         totalPenalty  $\leftarrow$  totalPenalty + penalty
10        count  $\leftarrow$  count + 1
11 return totalPenalty, count
```

---

---

**Algorithm 2:** Computation of edge-edge area-aware metrics.

---

**Input :**  $G = (V, E)$   
**Output:** total penalty  $P^{EE}(G)$  and count  $C^{EE}(G)$

```
1 totalPenalty  $\leftarrow$  0
2 count  $\leftarrow$  0
3 for  $e1 \in E$  do
4   for  $e2 \in E$  do
5     if  $e1 \neq e2$  then
6       if CheckIntersection( $v, e$ ) then
7          $x \leftarrow$  ComputeCrossingAngle( $e1, e2$ )
8         penalty  $\leftarrow$  PenaltyMapFunc( $x$ )
9         totalPenalty  $\leftarrow$  totalPenalty + penalty
10        count  $\leftarrow$  count + 1
11 return totalPenalty, count
```

---

### S3 List of graph layouts

Table S4: Full list of graph layouts

Type	Name (source)	#Nodes	#Edge	#Layouts	Layout algorithm	Purpose
Synthetic	four-clusters	40	123	10	Manual	debugging
	proxy shape	19	40	2		exploration of proxy shapes
	meta-edges 1	43	114	2		meta-edges overlap
	meta-edges 2	42	82	1		meta-edge and node overlap
	touching	12	18	2		minimum overlap
	progression NN	40	123	7		analysis of minimum penalty fraction ( $\alpha$ ) for NN
	progression NE	40	123	6		analysis of minimum penalty fraction ( $\alpha$ ) for NE
	progression EE	8	8	4		analysis of minimum penalty fraction ( $\alpha$ ) for EE
	single-level variable node-size	31	92	4	Davidson-harel, FM3, Linlog, stress marjorization	check metrics performance on this widely-used type of graph
Real-world	coauthor-big [2]	1538	8040	5	GEM [3], FM3 [4], Grouseflocks [5], Koala [2]	various situation in large layout
	coauthor-small [5]	103	505	4		
	email-eu-core [6]	986	15957	4		
	partition-add32 [7]	4960	9462	5		
Total	13			56		

## S4 Computational time

Table S5: Detailed running times on each layout for three metric families (NN, NE, and EE) and two approaches (AS vs. count), averaged over 4 runs. The slowdown factor is the ratio between running time of AS and count, representing how much the AS metric is slower than traditional count-based metrics. It is computed only if the running time is greater than 1 second, in order to reduce timing errors of the operating system. The average slowdown at the last row is computed over the valid rows.

Layout	NN			NE			EE		
	sprawlter	count	slowdown	sprawlter	count	slowdown	sprawlter	count	ratio
four-clusters-ne0	0.01	0.01	NA	0.06	0.04	NA	0.10	0.06	NA
four-clusters-ne1	0.01	0.01	NA	0.07	0.05	NA	0.12	0.07	NA
four-clusters-nn0	0.01	0.01	NA	0.06	0.04	NA	0.10	0.06	NA
four-clusters-nn1	0.01	0.01	NA	0.08	0.05	NA	0.10	0.06	NA
four-clusters-nn2	0.01	0.01	NA	0.09	0.06	NA	0.10	0.06	NA
four-clusters-nn3	0.01	0.01	NA	0.07	0.04	NA	0.09	0.06	NA
four-clusters-nn4	0.01	0.01	NA	0.08	0.05	NA	0.10	0.06	NA
four-clusters-sprawl	0.01	0.01	NA	0.06	0.04	NA	0.09	0.06	NA
four-clusters-ee0	0.01	0.01	NA	0.10	0.05	NA	0.18	0.07	NA
four-clusters-ee-glancing	0.01	0.01	NA	0.11	0.05	NA	0.18	0.08	NA
special-cases-1	0.00	0.00	NA	0.01	0.01	NA	0.01	0.01	NA
special-cases-2	0.00	0.00	NA	0.01	0.01	NA	0.01	0.01	NA
special-cases-3	0.01	0.01	NA	0.08	0.04	NA	0.10	0.05	NA
special-cases-4	0.01	0.01	NA	0.07	0.04	NA	0.09	0.05	NA
special-cases-5	0.01	0.01	NA	0.05	0.03	NA	0.05	0.03	NA
special-cases-6	0.00	0.00	NA	0.00	0.00	NA	0.00	0.00	NA
special-cases-7	0.00	0.00	NA	0.00	0.00	NA	0.00	0.00	NA
progression-nn-none	0.01	0.01	NA	0.06	0.04	NA	0.10	0.06	NA
progression-nn-touch-leaf	0.01	0.01	NA	0.07	0.06	NA	0.10	0.09	NA
progression-nn-touch-meta	0.01	0.01	NA	0.07	0.04	NA	0.10	0.10	NA
progression-nn-some-leaf	0.01	0.01	NA	0.06	0.07	NA	0.10	0.09	NA
progression-nn-some-meta	0.01	0.01	NA	0.08	0.06	NA	0.09	0.08	NA
progression-nn-near-max-leaf	0.01	0.01	NA	0.06	0.05	NA	0.09	0.07	NA
progression-nn-near-max-meta	0.01	0.01	NA	0.09	0.07	NA	0.11	0.10	NA
progression-ne-touch-leaf	0.01	0.01	NA	0.06	0.06	NA	0.09	0.06	NA
progression-ne-touch-meta	0.01	0.01	NA	0.06	0.04	NA	0.09	0.07	NA
progression-ne-some-leaf	0.01	0.01	NA	0.06	0.06	NA	0.09	0.08	NA
progression-ne-some-meta	0.01	0.01	NA	0.07	0.04	NA	0.11	0.06	NA
progression-ne-near-max-leaf	0.01	0.01	NA	0.06	0.04	NA	0.09	0.06	NA
progression-ne-near-max-meta	0.01	0.01	NA	0.08	0.05	NA	0.13	0.08	NA
progression-ee-ortho	0.00	0.00	NA	0.00	0.00	NA	0.00	0.00	NA
progression-ee-half	0.00	0.00	NA	0.00	0.00	NA	0.00	0.00	NA
progression-ee-near-glancing	0.00	0.00	NA	0.00	0.00	NA	0.00	0.00	NA
progression-ee-glancing	0.00	0.00	NA	0.00	0.00	NA	0.00	0.00	NA
varied-davidson-harel	0.01	0.01	NA	0.04	0.02	NA	0.05	0.03	NA
varied-fast-multipole-emb	0.01	0.00	NA	0.04	0.02	NA	0.06	0.03	NA
varied-linlog	0.01	0.00	NA	0.04	0.02	NA	0.05	0.03	NA
varied-stress-majorization	0.01	0.00	NA	0.03	0.02	NA	0.05	0.04	NA
coauthor-grouseflocks-0	0.01	0.01	NA	0.23	0.12	NA	0.45	0.24	NA
coauthor-grouseflocks-1	0.01	0.01	NA	0.28	0.12	NA	0.43	0.22	NA
coauthor-grouseflocks-2	0.02	0.01	NA	0.94	0.42	NA	2.22	1.19	1.87
coauthor-gem	13.83	9.46	1.46	139.26	97.92	1.42	420.14	293.56	1.43
coauthor-koala	12.02	8.85	1.36	197.67	144.26	1.37	966.99	531.41	1.82
snap-email-grouseflocks-1	0.02	0.01	NA	2.03	1.00	NA	43.93	20.65	2.13
snap-email-grouseflocks-2	0.03	0.02	NA	6.63	3.23	2.05	198.30	86.78	2.29
snap-email-gem	4.97	3.64	1.37	212.53	150.01	1.42	4147.91	2197.89	1.89
snap-email-koala	4.93	3.69	1.34	331.11	240.04	1.38	10252.12	4196.38	2.44
ivOrigins-grouseflocks-2	0.01	0.01	NA	0.25	0.12	NA	1.33	0.61	NA
ivOrigins-grouseflocks-4	0.00	0.00	NA	0.04	0.02	NA	0.03	0.02	NA
ivOrigins-gem	0.05	0.04	NA	0.63	0.48	NA	2.77	1.49	1.86
ivOrigins-koala	0.05	0.04	NA	1.09	0.73	NA	6.64	2.99	2.22
partition-add32-grouseflocks-0	0.21	0.16	NA	1.31	0.82	NA	1.08	0.78	NA
partition-add32-grouseflocks-1	0.25	0.18	NA	2.09	1.38	1.51	1.93	1.40	1.37
partition-add32-grouseflocks-5	0.64	0.40	NA	5.84	3.73	1.57	6.24	4.22	1.48
partition-add32-fm3	127.99	92.38	1.39	504.51	358.67	1.41	510.11	372.16	1.37
partition-add32-koala	130.81	94.17	1.39	773.12	560.11	1.38	1604.56	880.74	1.82
<b>average</b>			1.38			1.50			1.85

## S5 Computational pipeline

The full computational pipeline that we have implemented includes multiple data preparation steps in addition to the sprawlter metrics themselves, plus other metrics for comparative analysis purposes.

1. **Convert graph format** (Python). We convert a variety of input graph formats into Tulip format, as Tulip [8] provides several useful layout algorithms.
2. **Apply layout algorithms**. For the synthetic graphs, we manually position the nodes to create various degree of clutter and sprawl, and also apply four different layout algorithms within the version 5.2.1 of Tulip [8] (fast multipole embedder, Davidson-Harel, Linlog, stress majorization). For the real-world graphs, we apply two layout algorithms in Tulip (GEM, FM3); we use the 2008 version of GrouseFlocks [5], and a version of Koala [2] modified to save in Tulip format. The output in all cases is the geometric information of nodes and edges (position, area, shape, length) in Tulip format.
3. **Extract geometries and node hierarchy** (Python). We then parse the 56 layouts in Tulip format, extract the information of geometry and node hierarchy, and store it in JSON format.
4. **Compute metrics** (Python). We implement the sprawlter metrics for the NN (Algorithm 1 in the main paper), NE, and EE families and also their respective count-based metrics, and also the global readability metrics of node-node overlap and crossing angle of Dunne et al [9]. We use the Shapely Python package [10] for manipulation of geometry.
5. **Display and analyze results for comparison**. We display the computed metrics of each layout in a table with HTML and JavaScript, and we write Python scripts to analyze parameters and computational time.

GrouseFlocks was developed over ten years ago, and while it still functions as an interactive exploration tool for multi-level graph layouts, the coloring of metanodes is no longer correct (possibly due to deprecated dependent libraries). In order to have easy-to-understand results figures, we exported Tulip-format output files from GrouseFlocks into the current Tulip, saved them in SVG format, and placed the metanodes with the correct colors back to their original positions in Adobe Illustrator.

In the supplemental materials, we include all input files (in JSON format after step 3), output files (results of step 4), implementation code, and Python scripts for results analysis.

## References

- [1] Weidong Huang, Seok-Hee Hong, and Peter Eades. Effects of crossing angles. In *Pacific Visualization Symposium (PacificVIS)*, pages 41–46. IEEE, 2008.
- [2] Takayuki Itoh and Karsten Klein. Key-node-separated graph clustering and layouts for human relationship graph visualization. *IEEE computer graphics and applications*, 35(6):30–40, 2015.
- [3] Arne Frick, Andreas Ludwig, and Heiko Mehldau. A fast adaptive layout algorithm for undirected graphs (extended abstract and system demonstration). In *International Symposium on Graph Drawing*, pages 388–403. Springer, 1994.
- [4] Stefan Hachul and Michael Jünger. Drawing large graphs with a potential-field-based multilevel algorithm. In *International Symposium on Graph Drawing*, pages 285–295. Springer, 2004.
- [5] Daniel Archambault, Tamara Munzner, and David Auber. Grouseflocks: Steerable exploration of graph hierarchy space. *IEEE transactions on visualization and computer graphics*, 14(4):900–913, 2008.
- [6] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):2, 2007.
- [7] Alan J Soper, Chris Walshaw, and Mark Cross. A combined evolutionary search and multilevel optimisation approach to graph-partitioning. *Journal of Global Optimization*, 29(2):225–241, 2004.
- [8] David Auber, Daniel Archambault, Romain Bourqui, Maylis Delest, Jonathan Dubois, Antoine Lambert, Patrick Mary, Morgan Mathiaut, Guy Melançon, Bruno Pinaud, Benjamin Renoust, and Jason Vallet. TULIP 5. In Reda Alhajj and Jon Rokne, editors, *Encyclopedia of Social Network Analysis and Mining*, pages 1–28. Springer, August 2017.
- [9] Cody Dunne, Steven I Ross, Ben Shneiderman, and Mauro Martino. Readability metric feedback for aiding node-link visualization designers. *IBM Journal of Research and Development*, 59(2/3):14–1, 2015.
- [10] Shapely contributors. Shapely: manipulation and analysis of geometric objects. <https://shapely.readthedocs.io/en/latest>, 2018. [Online; accessed 28-March-2019].