

# Multi-level Graph Drawing using Infomap Clustering <sup>\*</sup>

Seok-Hee Hong<sup>1</sup>, Peter Eades<sup>1</sup>, Marnijati Torkel<sup>1</sup>, Ziyang Wang<sup>1</sup>,  
David Chae<sup>1</sup>, Sungpack Hong<sup>2</sup>, Daniel Langerenken<sup>2</sup>, and Hassan Chafi<sup>2</sup>

<sup>1</sup> University of Sydney, Australia

{seokhee.hong,peter.eades,mtor0581,zwan0130,min.chae}@sydney.edu.au

<sup>2</sup> Oracle Research lab, US

{sungpack.hong,daniel.langerenken,hassan.chafi}@oracle.com

**Abstract.** *Infomap clustering* finds the community structures that minimize the expected description length of a random walk trajectory; algorithms for infomap clustering run fast in practice for large graphs. In this paper we leverage the effectiveness of Infomap clustering combined with the multi-level graph drawing paradigm. Experiments show that our new Infomap based multi-level algorithm produces good visualization of large and complex networks, with significant improvement in quality metrics.

## 1 Introduction

The *multi-level graph drawing* is a popular approach to visualize large and complex graphs to improve the quality of drawings. It recursively coarsens the graph and then uncoarsens the drawing using layout refinement. There are a number of multi-level graph drawing algorithms available [7,9,11,14,17,18,20]. They mainly differ in the *coarsening* method.

Clustering is a widely used analysis method for identifying groups with strong similarity, or communities in the data. Graph clustering is to partition a graph such that vertices in the same cluster are more interconnected. *Infomap clustering* computes clusters by translating a graph into a map, which decomposes the myriad nodes and links into modules that represent the graph [19]. It maximizes an objective function called the minimum description length of a random walk trajectory, where the approximation to the optimal solution can be computed quickly. Infomap performed the best in community finding experiments [15].

In this paper, we present a new multi-level graph drawing algorithm based on Infomap clustering. More specifically, we leverage the effectiveness of Infomap clustering, combined with the multi-level graph drawing paradigm. Experiments with real-world large and complex networks such as protein-protein interaction networks, Facebook graph, Autonomous Systems (AS) graphs as well as benchmark graphs show that our new multi-level algorithms produce good visualization with significant improvement in quality metrics, including shape-based metrics [4], edge crossing and stress. It also requires a small number of coarsening steps for medium to large graphs, which makes it fast to run.

<sup>\*</sup> Research supported by ARC Linkage Grant with Oracle labs.

## 2 Related Work

Hadany and Harel presented the multi-scale method using an edge contraction based coarsening method and a force-directed layout preserving topological properties such as cluster size and vertex degree [10]. Koren and Harel presented FMS, which used a  $k$ -center approximation based coarsening method and a force-directed layout with a beautification [14].

Walshaw presented a multi-level algorithm using a matching, by repeatedly collapsing maximal independent subsets of graph edges, and a grid variant of Fruchterman-Reingold [6] layout [20]. Gajer *et al.* presented GRIP using a maximum independent set filtration based coarsening method, and an intelligent initial placement of vertices based on both graph and Euclidean distances [7].

Quigley and Eades presented FADE using the quad tree, and Barnes-Hut  $n$ -body method [1] for approximation of the repulsive force computation in a force-directed layout [18]. Hachul and Junger presented FM<sup>3</sup> using similar method to compute the repulsive forces between vertices, where subgraphs with small diameter, called solar system, are partitioned and collapsed to obtain a multi-level representation [9]. Hu presented the *sfdp* layout, also using the Barnes-Hut approximation method [11]. Frishman and Tal [5] presented a multi-level force directed graph layout on the GPU, based on spectral partitioning and Kamada-Kawai layout [12]. Bartel *et al.* presented an experimental study for extensive comparison of various multi-level algorithms, using a combination of coarsening methods, initial placement and graph layout methods [2].

More recently, Meyerhenke *et al.* presented a multi-level algorithm using a label propagation method for the coarsening step, and Maxent stress optimisation layout [8] on shared memory parallelization [16]. Nguyen and Hong used fast  $k$ -core coarsening method, which can be computed in linear time [17].

## 3 Infomap based Multi-level Algorithm

The multi-level graph drawing algorithm is an iterative process consisting of the following three steps: *coarsening*, *initialization* (or *placement*), and *graph layout* (or *refinement*). Roughly speaking, the coarsening step is to cluster vertices to define a smaller graph, recursively until the size of the graph falls below the threshold, resulting in a coarse graph hierarchy,  $G_0, G_1, \dots, G_L$ . The layout of graph  $G_L$  is then extended to the layout of graph  $G_{L-1}$  by *placement* (i.e., add vertices back to the layout) and *refinement* step. Recursively, these steps extend the layout of graph  $G_L$  to  $G_0$  by repetitively interpolating from  $G_i$  to  $G_{i-1}$ . In each iteration, the layout of  $G_i$  is used to compute an initial placement of  $G_{i-1}$ , and then the layout algorithm is applied to refine the layout.

### 3.1 Coarsening: Infomap Clustering

Let  $G = (V, E)$  be a graph with vertex set  $V$  and edge set  $E$ . The coarsening step computes a graph level hierarchy by iteratively computing a sequence of smaller graphs  $G_0, G_1, G_2, \dots, G_L$ , where the original graph  $G = G_0$ . At each level, a coarser graph (or clustered graph) is computed by combining a sets of

vertices belong to the same cluster in  $G_i$  and replacing into a single vertex in  $G_{i+1}$ , recursively until the predefined stop criterion is satisfied.

*Infomap clustering* finds community structure that minimizes the expected description length of a random walk trajectory [19]. It computes clusters by translating a graph into a map, which decomposes the myriad nodes and links into modules that represent the graph. The algorithm maximizes an objective function called the Minimum Description Length.

We first compute the Infomap clustering of  $G$ , and partition the vertex set  $V$  into  $V_i$  based on the clusters. More specifically, we define a clustered graph with a weighted vertex set (i.e., the number of vertices belong to each cluster) and a weighted edge set (i.e., the number of edges between the partitioned vertex set). The vertices  $u_1, u_2, \dots, u_k \in V_i$  are merged to form a new cluster vertex  $v \in V_{i+1}$ , where the weight of  $v$  is computed as  $|v| = |u_1| + |u_2| + \dots + |u_k|$ . Similarly, the weight of the collapsed edges are computed as the sum of the weights of the edges that it replaces. This coarsening phase stops when the resulting clustered graph has a small size (say 50) or there is no reduction in terms of size.

### 3.2 Initialization: Placement

This step aims to compute a good initial layout of  $G_{i-1}$  using the layout of  $G_i$ . Let  $v_i \in V_i$  of  $G_i$  corresponds to a cluster of vertices  $u_1, u_2, \dots, u_k \in V_{i-1}$  of  $G_{i-1}$ . We add back vertices  $u_j, j = 1, \dots, k$  to the layout of  $G_i$  by initializing the positions of  $u_j$  using the position of  $v_i$ . Here we use the following three variations.

- Circle placement: It places all  $u_j, j = 1, \dots, k$  at the circle with a small radius, where the center of the circle is the location of  $v_i$ .
- Barycenter placement: It places each vertex at the barycenter of its neighbors [7].
- Zero placement: It places all  $u_j, j = 1, \dots, k$  at the same position as  $v_i$  with small perturbation [20].

### 3.3 Refinement: Force-directed Layout

The initial layout of  $G_{i-1}$  is recursively refined at each level using a force-directed algorithm. We use layout algorithms, previously used in other multi-level graph drawing algorithm experiments [2]:

- FR: Fruchterman and Reingold layout [6].
- FRG: grid variant of Fruchterman and Reingold layout, used in [20].
- FME (Fast-Multipole Embedder): an improvement of NME (New Multipole Method) layout of FM<sup>3</sup> [9], designed for a multi-level method in [2].

## 4 Experiments

We implemented Infomap clustering based multi-level algorithm using OGDF [3], which was used in the comparison experiments of multi-level algorithms [2]. We used a standard Dell laptop with Intel Core i7, 16 GB RAM.

We first experimented with three different placement methods, and found that there is no significant difference in terms of layout quality. We choose the

barycenter placement, which shows slightly better performance, with three layouts FR, FRG and FME for comparison.

More specifically, we have the following variations for comparison:

- InfomapFR: Infomap multi-level algorithm with FR layout
- InfomapFRG: Infomap multi-level algorithm with FR grid variant layout
- InfomapFME: Infomap multi-level algorithm with FME layout

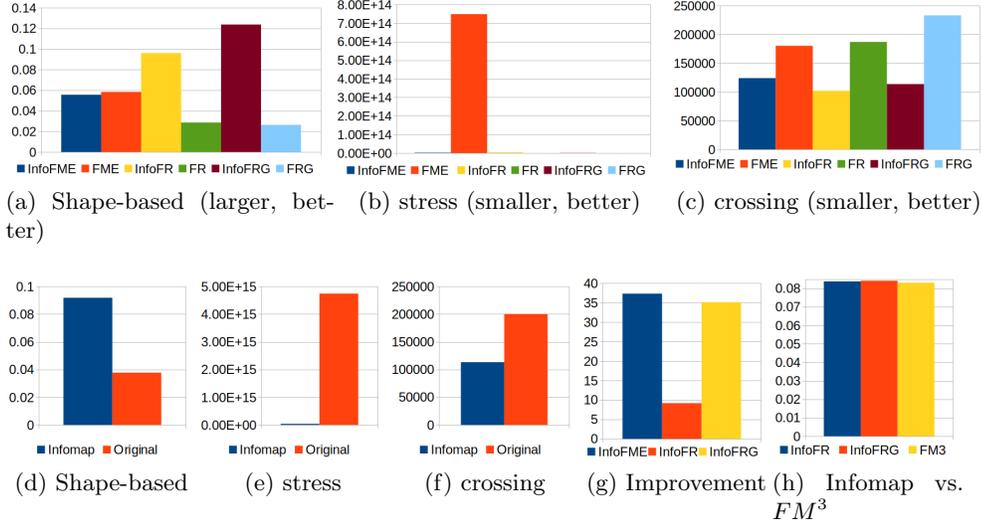
The experiment was conducted with real-world benchmark data sets including social networks such as facebook, biological networks such as protein-protein interaction networks, and benchmark graphs used in previous work [2,17,20].

Table 1 shows the details of the data sets, the number of coarsening levels and runtime (seconds), where  $D$  represents the density of a graph  $G$  and  $L$  represents the number of levels. We can clearly see that the Infomap coarsening method produced small number of levels such as 2 or 3 for most of data sets. Overall, Infomap clustering runs quite fast for medium size graphs.

Graph $G$	$ V_0 $	$ E_0 $	$D$	$L$	Time	$ V_1 $	$ E_1 $	$ V_2 $	$ E_2 $	$ V_3 $	$ E_3 $
$G_{15_0}$	1785	20459	11.5	2	0.02	59	100	9	8		
$nasa1824$	1824	18692	10.3	2	0.02	53	217	5	7		
$G_{4_0}$	2075	4769	2.3	2	0.02	89	326	8	11		
$yeastppi$	2361	7182	3.0	2	0.04	302	1923	101	0		
$soc_h$	2426	11630	4.8	2	0.02	301	1088	149	1		
$oflights$	2939	15677	5.3	2	0.03	170	477	19	24		
$ecolippi$	3796	78120	20.6	2	0.03	245	2453	53	1		
$facebook$	4039	88234	21.9	2	0.02	93	272	7	11		
$3elt$	4720	13722	2.9	2	0.05	189	489	17	35		
$USpowerGrid$	4941	6594	1.3	2	0.18	489	963	44	104		
$as19990606$	5188	10974	2.1	2	0.17	368	2034	12	38		
$commanche_dual$	7920	19800	2.5	2	0.24	503	1365	34	71		
$p2p-Gnutella05$	8846	31839	3.6	2	0.20	830	18154	3	0		
$astroph2001$	16046	121251	7.6	3	0.61	1219	9333	395	68	369	0
$condmat2001$	16264	47594	2.9	3	1.33	1720	4574	798	774	726	0
$crack_dual$	20141	30043	1.5	3	1.16	1357	3633	84	216	10	18
$bcsstk31$	35588	608502	17.1	2	0.36	453	2295	25	44		
$shock-9$	36476	71290	2.0	3	1.17	1351	3852	74	191	8	14
$del16$	65536	196575	3.0	3	1.95	1981	5921	101	290	8	16

**Table 1.** Data sets, size, number of levels ( $L$ ) and runtime.

**Comparison of Quality Metrics:** For large and complex graphs, edge crossing may not a suitable metric to measure the quality of drawings [4,13]. We used the *shape-based metrics* [4]; this is a new graph drawing quality measure specially designed for large graphs. Roughly speaking, the shape-based metrics measure the *faithfulness* of graph drawing, i.e., how well the *shape* of the drawing represents the structure (or shape) of the graph.



**Fig. 1.** Significant improvement in quality metrics: (a)(b)(c) Average of metrics per layout: InfomapFME (blue), FME (red), InfomapFR (yellow), FR (green), InfomapFRG (brown), FRG (cyan); (d)(e)(f) Average of metrics: Infomap (blue) vs. Original (red); (g) Average of improvement by Infomap over Original in Shape-based metrics: InfomapFME (blue), InfomapFR (red), InfomapFRG (yellow); (h) Average of shape-based metrics: InfomapFR (blue), InfomapFRG (red),  $FM^3$  (yellow).

Figures 1(a), (b) and (c) show the comparison of *average metrics* between six layouts (i.e., Infomap multi-level vs. FME, FRG, FR original layouts) using *shape-based quality metrics* ( $Q$ ), *stress* and *edge crossings*. Clearly, we can see that Infomap multi-level layouts perform significantly better than the original layouts. In general, InfomapFR and InfomapFRG perform better than InfomapFME. Figures 1(d), (e) and (f) show the *average metrics* between Infomap multi-level and original layouts. Overall, we can see that Infomap multi-level layouts outperform original layouts. Figure 1(g) shows the *average improvement* by Infomap multi-level layouts over original layouts in shape-based metrics (i.e.,  $(Q_{Infomap}/Q_{Original} - 1)$ ). Clearly, significant improvement was achieved by InfomapFME and InfomapFRG.

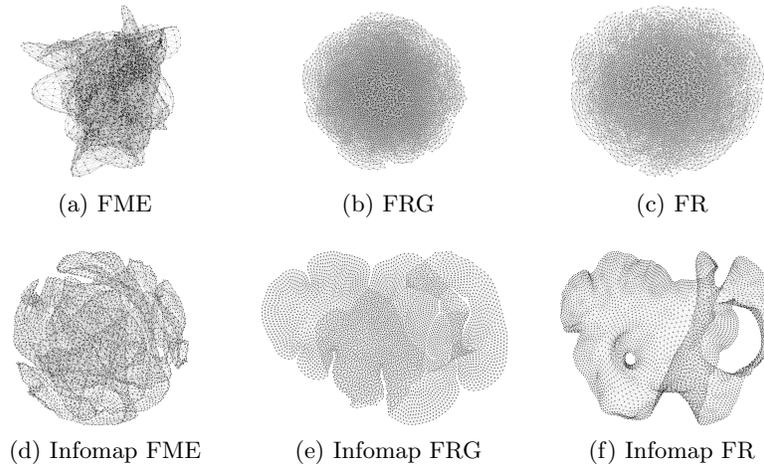
**Visual Comparison:** Overall, Infomap multi-level layouts perform significantly better than original layouts. In general, InfomapFR and InfomapFRG perform significantly better than other layouts, and InfomapFME achieved the most significant improvement over FME. For example, Figure 2 shows visual comparison between layouts of *3elt*.

**Comparison with  $FM^3$ :** Figure 1(h) shows *average shape-based metrics* between InfomapFR, InfomapFRG and  $FM^3$ , excluding the outlier. Clearly, we can see that InfomapFR and InfomapFRG perform similar to  $FM^3$  in shape-

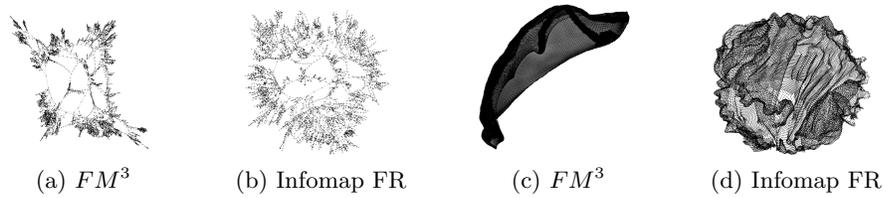
based metrics. For *layout comparison*, see Figures 3. We can see that InfomapFR perform similar to  $FM^3$ , and for some instances perform better than  $FM^3$ .

**Summary:** Our experimental results provide strong evidence that our Infomap based multi-level algorithm performs considerably well for real-world social networks, biological networks and benchmark graphs.

- Overall, Infomap multi-level layouts perform significantly better than original layouts in terms of quality metrics and visualisation.
- Metric wise, InfomapFR and InfomapFRG perform better than InfomapFME.
- InfomapFME achieved the most significant improvement.
- InfomapFR and InfomapFRG perform similar to  $FM^3$  in terms of shape-based metrics and visual comparison.



**Fig. 2.** Visual comparison of *3elt*.



**Fig. 3.** Comparison with  $FM^3$ : (a)(b) *USpowerGrid*; (c)(d) *shock-9*

## References

1. Barnes, J., Hut, P.: A hierarchical  $O(n \log n)$  force-calculation algorithm. *Nature* **324** (1986)
2. Bartel, G., Gutwenger, C., Klein, K., Mutzel, P.: An experimental evaluation of multilevel layout methods. In: *Graph Drawing - 18th International Symposium, GD 2010, Konstanz, Germany, September 21-24, 2010. Revised Selected Papers.* pp. 80–91 (2010)
3. Chimani, M., Gutwenger, C., Jünger, M., Klau, G.W., Klein, K., Mutzel, P.: The open graph drawing framework (OGDF). In: *Handbook on Graph Drawing and Visualization.*, pp. 543–569 (2013)
4. Eades, P., Hong, S., Klein, K., Nguyen, A.: Shape-based quality metrics for large graph visualization. In: *Graph Drawing and Network Visualization - 23rd International Symposium, GD 2015.* pp. 502–514 (2015)
5. Frishman, Y., Tal, A.: Multi-level graph layout on the GPU. *IEEE Trans. Vis. Comput. Graph.* **13**(6), 1310–1319 (2007)
6. Fruchterman, T.M., Reingold, E.M.: Graph drawing by force-directed placement. *Software: Practice and experience* **21**(11), 1129–1164 (1991)
7. Gajer, P., Kobourov, S.G.: Grip: Graph drawing with intelligent placement. *J. Graph Algorithms Appl.* **6**(3), 203–224 (2002)
8. Gansner, E.R., Hu, Y., North, S.C.: A maxent-stress model for graph layout. *IEEE Trans. Vis. Comput. Graph.* **19**(6), 927–940 (2013)
9. Hachul, S., Jünger, M.: Drawing large graphs with a potential-field-based multilevel algorithm. In: *Graph Drawing.* pp. 285–295. Springer (2005)
10. Hadany, R., Harel, D.: A multi-scale algorithm for drawing graphs nicely. *Discrete Applied Mathematics* **113**(1), 3–21 (2001)
11. Hu, Y.: Efficient, high-quality force-directed graph drawing. *Mathematica Journal* **10**(1), 37–71 (2005)
12. Kamada, T., Kawai, S.: An algorithm for drawing general undirected graphs. *Information processing letters* **31**(1), 7–15 (1989)
13. Kobourov, S.G., Pupyrev, S., Saket, B.: Are crossings important for drawing large graphs? In: *Graph Drawing - 22nd International Symposium, GD 2014, Würzburg, Germany, September 24-26, 2014, Revised Selected Papers.* pp. 234–245 (2014)
14. Koren, D., Harel, Y.: A fast multi-scale method for drawing large graphs. *Journal of graph algorithms and applications* **6**(3) (2002)
15. Lancichinetti, A., Fortunato, S.: Community detection algorithms: A comparative analysis. *Phys. Rev. E* **80**, 056117 (Nov 2009)
16. Meyerhenke, H., Nöllenburg, M., Schulz, C.: Drawing large graphs by multilevel maxent-stress optimization. In: *Graph Drawing and Network Visualization - 23rd International Symposium, GD 2015.* pp. 30–43 (2015)
17. Nguyen, A., Hong, S.: k-core based multi-level graph visualization for scale-free networks. In: *2017 IEEE Pacific Visualization Symposium, PacificVis 2017, Seoul, South Korea, April 18-21, 2017.* pp. 21–25 (2017)
18. Quigley, A., Eades, P.: *Fade: Graph drawing, clustering, and visual abstraction.* In: *Graph Drawing.* pp. 197–210. Springer (2001)
19. Rosvall, M., Bergstrom, C.T.: Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences* **105**(4), 1118–1123 (2008)
20. Walshaw, C., et al.: A multilevel algorithm for force-directed graph-drawing. *J. Graph Algorithms Appl.* **7**(3), 253–285 (2003)