Eurographics Conference on Visualization (EuroVis) (2014)
N. Elmqvist, M. Hlawitschka, and J. Kennedy (Editors)

Short Papers

Visualizing Graphs as Maps with Contiguous Regions

Stephen G. Kobourov, Sergey Pupyrev and Paolo Simonetto

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

Abstract

Relational datasets, which include clustering information, can be visualized with tools such as BubbleSets, LineSets, SOM, and GMap. The countries in SOM-based and GMap-based visualizations are fragmented, i.e., they are represented by several disconnected regions. While BubbleSets and LineSets have contiguous regions, these regions may overlap, even when the input clustering is non-overlapping. We describe two methods for creating non-fragmented and non-overlapping maps within the GMap framework. The first approach achieves contiguity by preserving the given embedding and creating a clustering based on geometric proximity. The second approach achieves contiguity by preserving the clustering information. The methods are quantitatively evaluated using embedding and clustering metrics, and their usefulness is demonstrated with several real-world datasets and a fully-functional online system at `gmap.cs.arizona.edu`.

1. Introduction

The geographic map metaphor has been utilized as visual interface for relational data, where objects, relations between objects, and clustering are captured by cities, roads between cities, and countries. Information spatialization is the process of assigning 2D coordinates to abstract data points, ideally such that the spatial mapping has much of the characteristics of the original high-dimensional space. Multidimensional scaling (MDS) and principal component analysis (PCA) are techniques that allow us to spatialize high-dimensional data, resulting in point clouds, node-link diagrams, and maps. In data mining and data analysis, clustering is a very important step and maps are very helpful in dealing with clustered data. First, by explicitly defining the boundary of the clusters and coloring the regions, we make the clustering information clear. Second, as most dimensionality-reduction techniques lead to 2D positioning of the data points, a map is a natural generalization. Finally, while it often takes considerable effort to understand graphs, charts, and tables, a map representation is more intuitive.

While many map-based visualizations have been considered, some of them produce fragmented maps (SOM, GMap), while others have overlapping regions, even when the underlying clusters have no overlaps (BubbleSets, LineSets). We describe methods for creating maps with contiguous and non-overlapping regions, since experimental evidence suggests that this results in better task performance and fewer misinterpretations. The algorithms can be utilized in different scenarios depending on the type of input data. The first algorithm relies on the initial embedding of the input graph and can be applied when nodes have preassigned coordinates. The second algorithm is applicable in scenarios in which the input graph has been clustered in advance. We designed and implemented both approaches and they are available as fully functional online tools.

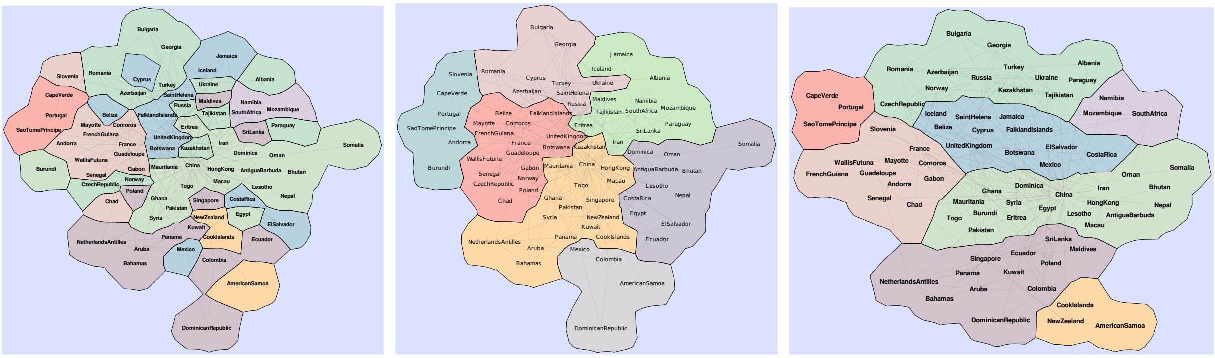
2. Related Work

Using maps to visualize non-cartographic data has been considered in the context of spatialization [SF03, FMM06]. Self organizing maps (SOM), coupled with geographic information systems, render maps of textual documents [Sku02]. Similarly, maps of general science show related groups of scientific disciplines [BKB05]. More recently, maps of computer science (MOCS) [FK14] provide visual exploration of topics in conferences and temporal topic evolution.

Different visualization techniques identify groups by connecting or enclosing related elements: BubbleSets [CPC09], LineSets [ARRC11], and Euler diagrams [RZF08, SAA09]. When providing an underlying overlap-free clustering, the first two techniques would generate contiguous but potentially overlapping regions. Instead, Euler diagram generation methods produce contiguous and non-overlapping regions but ignore the connectivity and the initial embedding of the graph.

The geographic map metaphor was also used for visual-

DOI: 10.2312/eurovisshort.20141153



(a) *GMap* **(b)** *EBA* **(c)** *CBA*

Figure 1: *International trade relationships visualized with (a) the GMap algorithm, our new (b) embedding-based algorithm (EBA) and (c) clustering-based algorithm (CBA). EBA preserves the given graph embedding, while CBA preserves the given clustering of nodes. The countries computed by EBA and CBA always form contiguous regions.*

izing recommendations, where the Graph-to-Map approach (GMap) was introduced [GHKV09]. GMap combines graph layout and graph clustering, together with appropriate coloring of the clusters and creating boundaries based on clusters and connectivity in the original graph. A follow-up paper describes how this approach can be generalized to any relational data set [HGK10], such as books on Amazon, TV-shows and movies, and research collaborations. GMap has also been used to produce maps of computer science based on conference publications or individual publications [FK14]. In summary, GMap generates drawing of clustered graphs so that the groups are depicted as countries in a geographical map. GMap visualizations are based on a modified Voronoi diagram of the nodes, which in turn is determined by the embedding and clustering. However, since graph layout and graph clustering are two separate steps, the result is often fragmented; see Fig. 1.

A recent study compared several techniques for displaying clusters with node-link diagrams [JRHT14]. They found GMap improved performance of searching and exploration tasks, compared to standard node-link diagrams, BubbleSets, and LineSets. On the other hand, GMap worsened performance of group-based tasks, as there is no explicit connection between disjoint regions of the same cluster, and users were unsure whether they belong to the same group or not. Such fragmentation makes it difficult to identify the correct regions and can result in misinterpretation of the map. Our own informal experiments indicate that users prefer contiguous maps, and that the removal of country fragmentation is considered more important than preserving other graph characteristics such as embedding or clustering.

3. Algorithms for Creating Contiguous Maps

We assume that the data is already processed and the input is a graph $G = (V, E)$ with edge weights $w(e) \in \mathbb{R}$ for all $e \in E$. If the graph is unweighted then we assume $w(e) = 1$. The output of our algorithms is a graph layout, i.e., positions $p_v \in \mathbb{R}^2$ for all nodes $v \in V$, and a clustering $C = \{C_1, \dots, C_k\}$ of the nodes so that nodes of the same cluster form a contiguous region. Let $||p_u - p_v||$ be a Euclidean distance between nodes u and v, and $c_v \in C$ be a cluster to which node $v \in V$ is assigned. For an edge (u, v), we define the *ideal edge length* as $d(u, v) = -\log[0.9 \cdot w(u, v) + 0.1]$.

We first describe an *embedding-based algorithm (EBA)*, which preserves a given graph layout. There are applications in which the input specifies positions of the nodes (e.g., MDS, PCA). EBA is well suited for such settings. We then describe a *cluster-based algorithm (CBA)*, which preserves a given graph clustering. CBA is suited for settings where the input specifies the clustering of the nodes (e.g., party affiliation in political networks, countries grouped by continent). Note that when neither clustering nor embedding is given, we can apply both algorithms to create a map. We pair EBA with any embedding algorithm, or CBA with any clustering algorithm. Both produce contiguous maps; see Fig. 1.

Embedding-Based Algorithm. We assume the input graph is drawn with fixed node positions. The graph is processed in the following two steps.

1. Cluster graph nodes using K-means [Llo82]. Here the edges of the graph are ignored; the distance between nodes u and v is the distance between the corresponding points $||p_u - p_v||$. This results in a partition of graph nodes into k clusters so that every node belongs to the cluster with the nearest mean. Since we use Euclidean distances, the clusters form convex (thus, contiguous) regions.

2. Iteratively refine the clustering. For every node v, find its nearest neighbor u in a different cluster, i.e., $C_v \neq C_u$. If v has more edges with nodes of C_u, rather than with the nodes of c_v, i.e., $\sum_{t \in C_u} w(v, t) > \sum_{t \in C_v} w(v, t)$, then reassign v to the cluster C_u and proceed with the next node. The operation is

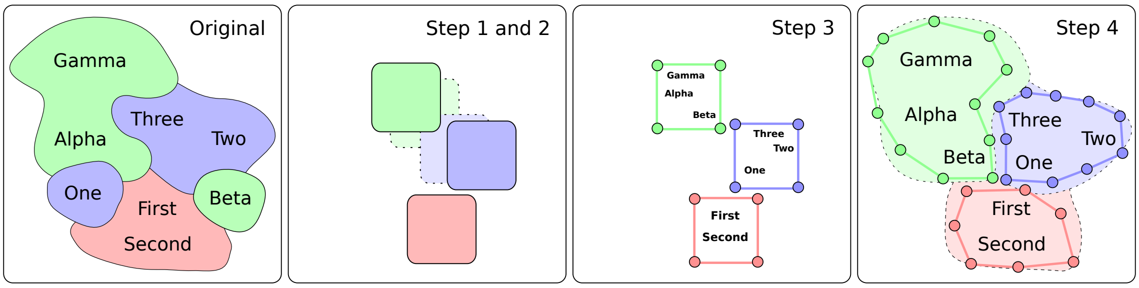


Figure 2: *CBA: initial map;* ***Step 1****: squares are placed at country-barycenters;* ***Step 2****: overlaps are removed;* ***Step 3****: subgraphs induced by countries are scaled to fit inside its square;* ***Step 4****: a spring embedder pulls nodes towards original positions; flexible uncrossable boundaries ensure contiguity and smooth shapes.*

applied only if the new clusters are still contiguous, which can be tested as described in [GHKV09]. The process is repeated until no node can be moved to a new cluster. Note that the process converges, as the sum $\sum_{u,v \in C} w(u,v)$ increases in each step, but we do not have a good bound on the number of iterations required.

Any geometric clustering algorithm can be used in the first step; we choose K-means as it is efficient, simple to implement, steadily produces non-fragmented clusters, and converges quickly for all graphs in our datasets (with the iterative refinement). A disadvantage is that the number of clusters should be specified; but when the number is unknown, we compute a suitable value using standard methods [SJ03].

Clustering-Based Algorithm. We assume the input graph is given with fixed node clustering and some initial layout. The algorithm attempts to maintain the layout, as it helps to preserve the mental map. This is important for applications in which the given embedding is meaningful. The graph is processed in the following four steps; see Fig. 2.

1. Compute country-barycenters in the initial layout. The country-barycenter is the average coordinate of the nodes in that cluster. It is used to keep the position of the country as close as possible to its position in the input layout.

2. Reserve a square for each country. Assign square-nodes of equal size to each country, and then remove overlaps between the squares using a node overlap removal algorithm [DMS07]. This step ensures that each country has a non-overlapping square-region reserved for its nodes.

3. Bound country regions and reposition nodes inside. Enclose each country region by four boundary nodes and edges. Take the layout of the subgraph induced by the nodes of a country and scale it to fit inside its square.

4. Pull nodes towards their original positions with flexible boundaries. This is accomplished by a modified spring embedder with added attractive force pulling nodes towards their position in the original embedding, in addition to the standard attractive and repulsive forces.

For the last step we modify the ImPrEd algorithm, which prevents nodes from crossing edges [SAAB11], with an additional force that attracts nodes to their original positions. The algorithm keeps nodes inside country boundaries, by making the boundary edges flexible so they can expand or contract in order to fit the shape of the growing countries. CBA aims to preserve the original position of the cluster and the relative position of the nodes inside groups; see Fig. 1c.

4. Quantitative Analysis

We evaluate our new methods using real-world datasets and quantitative metrics for graph layout and clustering. The metrics are defined so that the measurement is a real positive number in the range $[0,1]$ with 1 indicating the best value. We use 10 datasets, creating a total of 70 maps. Half of them (Amazon book titles, `last.fm` music bands, coauthorship graph, university similarity, and trade data) are from [HGK10]. The others are extracted from DBLP titles for conferences, journals, and authors using MOCS. The maps contain $|V| \in \{50, 100, \dots, 500\}$ nodes in the underlying graph. First, we describe metrics for graph clustering.

Modularity. In a good clustering, the number of edges within clusters should be high and the number of edges between clusters should be low, as measured by modularity:

$$\frac{1}{2m}\sum_{u,v}\left(w_{uv} - \frac{deg_u deg_v}{2m}\right)\delta(c_u, c_v),$$

where the sum is over all pairs of nodes, $deg_v = \sum_t w(v,t)$ is the weighted degree of node v, $m = \frac{1}{2}\sum_{uv} w(u,v)$, and $\delta(c_i, c_j)$ is 1 if $c_i = c_j$ and 0 otherwise [BGW03].

Conductance. The conductance compares the weight of a cut and the weight of the edges in either of the two subgraphs induced by the cut [BGW03, Sch07]. The conductance of a graph is the average conductance over all cuts.

Coverage. The coverage is given as the fraction of the weight of all intra-cluster edges with respect to the total weight of all edges in the whole graph [Sch07]: $\sum_{uv} w(u,v)\delta(c_u,c_v) / \sum_{uv} w(u,v)$.

Next, we define metrics for graph layout.

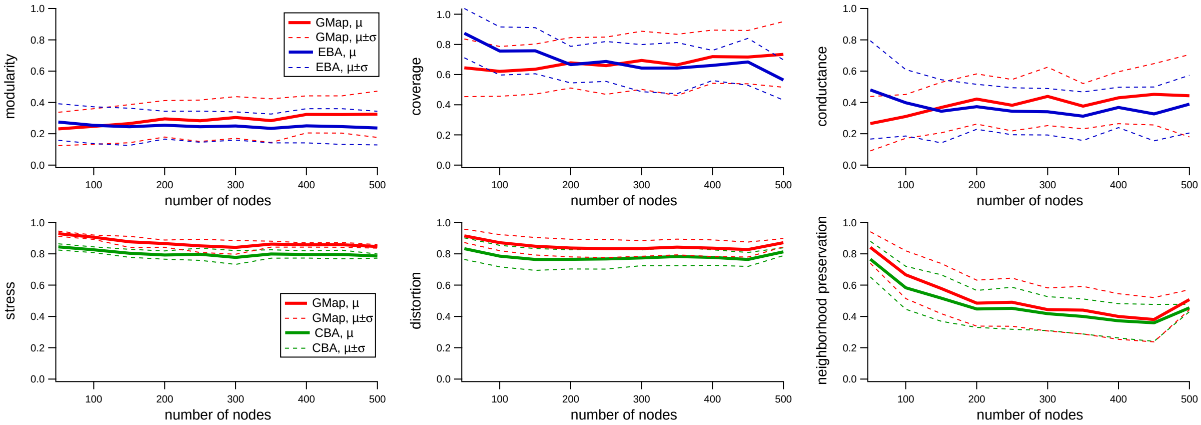


Figure 3: *Comparison of the embedding and clustering metrics over all graphs, where the means μ are solid thick lines and the standard deviations σ are thin dashed lines of the same color; the number of nodes in the graph is* $|V| \in \{50, 100, \ldots, 500\}$.

Stress. The stress of an embedding is a classic MDS metric, which measures the energy of the spring system; we use the normalized version of the metric suggested in [GKN04]:

$$stress = \frac{1}{m} \sum_{u,v \in V} w(u,v) \left(\frac{||p_u - p_v|| - d_{vu}}{\max(||p_u - p_v||, d_{uv})} \right)^2,$$

where $m = \frac{1}{2} \sum_{uv} w(u,v)$. Low stress indicates a good solution; as value 1 is best in all our metrics, we use $1 - stress$.

Distortion. *Distortion* measures whether the distances between pairs of nodes are proportional to the desired distances. Consider a matrix of ideal distances with entry d_{uv} for nodes u and v, and a matrix of actual distances between corresponding points with $\Delta_{uv} = ||p_u - p_v||$. The matrices are seen as two random variables for which the correlation coefficient r is computed. Since r takes values between -1 and 1, the metric is given by $(r+1)/2$.

Neighborhood Preservation. For each node v, T adjacent nodes of v are selected and their Euclidean distances to v are measured. The percentage of the T nodes that are within distance $d(T)$ (the distance of the T-th closest node to v in the graph space), averaged over all nodes v, measures neighborhood preservation [VPN*10]. As suggested in [GHN12], we use $T = 20$ in our experiments.

Since EBA preserves the given embedding, we only report its performance on the clustering metrics; see Fig 3. On average, EBA results in lower modularity, conductance, and coverage compared to the default (modularity-based) clustering of GMap. However, the reductions are less than 20%. It is worth noting that for small maps (50 and 100 nodes) the EBA clustering steadily outperforms the default one. More careful analysis shows that the second step of EBA ("local refinement") is very effective for maps of smaller size.

Since CBA preserves the given clustering, we only report its performance on the embedding metrics; see Fig 3. On average, CBA produces layouts with worse embedding metrics. The average decrease is 13% for stress, 10% for distortion, and 7% for neighborhood preservation. The CBA metrics are very similar to the default GMap values for instances in which the underlying graph is dense and the default map is highly fragmented. For some of datasets (international trade data), CBA had better results than the default (e.g., 9% better neighborhood preservation).

EBA is very fast, taking few milliseconds to process the largest tested graphs. CBA is less efficient producing maps with 100 nodes in a few seconds and maps with 500 nodes in under a minute. Improving performance of CBA (for example, with a hierarchical approach) is a future direction.

5. Conclusion and Future Work

We designed, implemented, and evaluated two approaches for generating contiguous maps. These approaches can be applied in different scenarios depending on the input data and user preferences. The simpler and more efficient EBA tends to produce maps with slightly worse clustering metrics. CBA keeps the clustering and aims to preserve the embedding. Although we utilized the GMap framework, other techniques can also benefit from these methods. For example, applying BubbleSets on top of the results of EBA or CBA may remove overlaps between disjoint clusters. An online tool and source code for the algorithms is available at gmap.cs.arizona.edu.

Although contiguous maps seem more readable, fragmentation might encode important information, e.g., close relationships between the members of a fragmented group with other groups. It would be worthwhile to identify such "meaningful" fragmentation and show it on the map. Similarly interesting would be in-depth user study comparing map-based visualizations and investigating the impact of clustering quality and embedding quality on map comprehension.

References

[ARRC11] Alper B., Riche N., Ramos G., Czerwinski M.: Design study of linesets, a novel set visualization technique. *Visualization and Computer Graphics, IEEE Transactions on 17*, 12 (2011), 2259–2267. 1

[BGW03] Brandes U., Gaertler M., Wagner D.: Experiments on graph clustering algorithms. In *In 11th Europ. Symp. Algorithms* (2003), Springer-Verlag, pp. 568–579. 3

[BKB05] Boyack K. W., Klavans R., Börner K.: Mapping the backbone of science. *Scientometrics 64* (2005), 351–374. 1

[CPC09] Collins C., Penn G., Carpendale S.: Bubble sets: Revealing set relations with isocontours over existing visualizations. *Visualization and Computer Graphics, IEEE Transactions on 15*, 6 (2009), 1009–1016. 1

[DMS07] Dwyer T., Marriott K., Stuckey P. J.: Fast node overlap removal. In *International Symposium on Graph Drawing (GD06)* (2007), Kaufmann M., Wagner D., (Eds.), vol. 4372 of *Lecture Notes in Computer Science*, Springer, pp. 153–164. 3

[FK14] Fried D., Kobourov S. G.: Maps of computer science. In *7th IEEE PacificVis Symposium* (2014). To appear. 1, 2

[FMM06] Fabrikant S., Monteilo D., Mark D. M.: The distance-similarity metaphor in region-display spatializations. *Computer Graphics and Applications, IEEE 26*, 4 (2006), 34–44. 1

[GHKV09] Gansner E., Hu Y., Kobourov S., Volinsky C.: Putting recommendations on the map - visualizing clusters and relations. In *Proc. 3rd ACM Conference on Recommender Systems* (2009), pp. 345–354. 2, 3

[GHN12] Gansner E. R., Hu Y., North S.: A maxent-stress model for graph layout. In *Pacific Visualization Symposium (PacificVis), 2012 IEEE* (2012), pp. 73–80. 4

[GKN04] Gansner E. R., Koren Y., North S.: Graph drawing by stress majorization. In *International Symposium on Graph Drawing (GD04)* (2004), Pach J., (Ed.), vol. 3383 of *Lecture Notes in Computer Science*, Springer, pp. 239–250. 4

[HGK10] Hu Y., Gansner E. R., Kobourov S. G.: Visualizing graphs and clusters as maps. *IEEE Computer Graphics and Applications 30*, 6 (2010), 54–66. 2, 3

[JRHT14] Jianu R., Rusu A., Hu Y., Taggart D.: How to display group information on node-link diagrams: an evaluation. *IEEE Transactions on Visualization and Computer Graphics* (2014). to appear. 2

[Llo82] Lloyd S. P.: Least squares quantization in pcm. *IEEE Transactions on Information Theory 28* (1982), 129–137. 2

[RZF08] Rodgers P., Zhang L., Fish A.: General Euler diagram generation. In *International Conference on Diagrams (Diag08)* (2008), Stapleton G., Howse J., Lee J., (Eds.), vol. 5223 of *Lecture Notes in Computer Science*, Springer, pp. 13–27. 1

[SAA09] Simonetto P., Auber D., Archambault D.: Fully automatic visualisation of overlapping sets. *Computer Graphics Forum (EuroVis09) 28*, 3 (June 2009), 967–974. 1

[SAAB11] Simonetto P., Archambault D., Auber D., Bourqui R.: ImPrEd: An improved force-directed algorithm that prevents nodes from crossing edges. *Computer Graphics Forum (EuroVis11) 30*, 3 (June 2011), 1071–1080. 3

[Sch07] Schaeffer S. E.: Graph clustering. *Computer Science Review 1*, 1 (2007), 27–64. 3

[SF03] Skupin A., Fabrikant S. I.: Spatialization methods: a cartographic research agenda for non-geographic information visualization. *Cartography and Geographic Information Science 30* (2003), 95–119. 1

[SJ03] Sugar C. A., James G. M.: Finding the number of clusters in a dataset. *Journal of the American Statistical Association 98*, 463 (2003), 750–763. 3

[Sku02] Skupin A.: A cartographic approach to visualizing conference abstracts. *IEEE Computer Graphics and Applications 22*, 1 (2002), 50–58. 1

[VPN*10] Venna J., Peltonen J., Nybo K., Aidos H., Kaski S.: Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *J. Mach. Learn. Res. 11* (2010), 451–490. 4