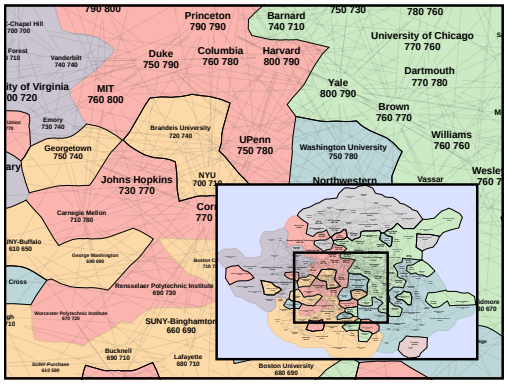
Visualizing Graphs as Maps with Contiguous Regions

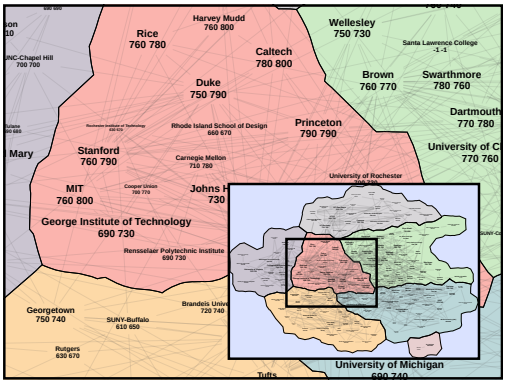
Stephen G. Kobourov Sergey Pupyrev
Paolo Simonetto
Department of Computer Science, University of Arizona, Tucson, AZ, USA

Abstract

Relational datasets, which also 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, that is, they are represented by several disconnected regions. Although countries can be uniquely colored to help with identification, experimental data indicates that such fragmentation makes it more difficult to identify the correct regions. On the other hand, BubbleSet and LineSets visualizations (originally developed to show overlapping sets) have contiguous regions but the 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 in the plane and creating a clustering based on geometric proximity. The second approach achieves contiguity by preserving the clustering information and distorting the given embedding in the plane if it would result in fragmentation. We formally analyze these methods and quantitatively evaluate them using embedding metrics and clustering metrics. We demonstrate the usefulness of the new methods with several datasets, and make them available in an online system at `http://gmap.cs.arizona.edu`.



(a) The original fragmented map



(b) A contiguous map built with our new algorithm

Figure 1: Maps of US universities and their average SAT scores, with similarity-based clustering. In the fragmented map, a country consists of all regions with the same color, whereas in the contiguous map, a country is a single contiguous region.

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

Figure 2: Comparison of the original GMap results and the new contiguous maps for the same graph, showing international trade relationships.

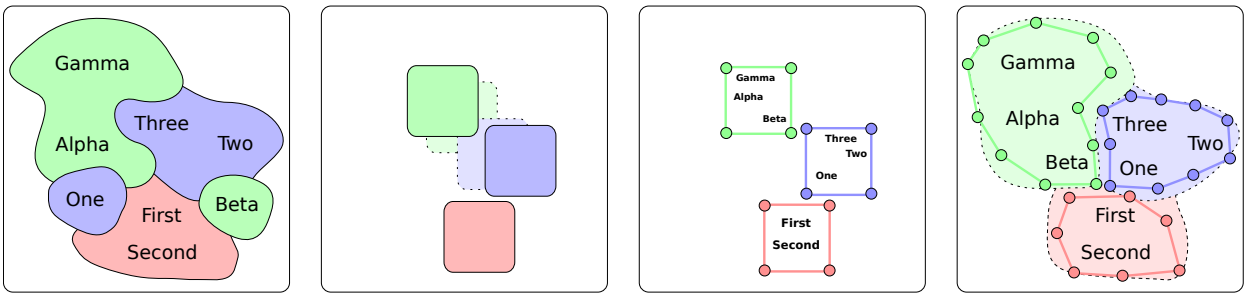
1 Introduction

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. Multi-dimensional scaling and principal component analysis 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 familiar and intuitive as most people are very familiar with maps.

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. 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 fully functional online tools.

We also ran a preliminary user experiment on fragmented and contiguous maps generated for the same dataset. A vast majority of participants prefer the contiguous variant even for relatively small datasets with $50-150$ objects and $5-8$ clusters. Although each cluster is colored uniquely to help with identification, most people are familiar with real geographical maps in which countries are largely contiguous. We observed a significant rate of misinterpretations of the map that occur when countries are fragmented.



(a) Original map **(b)** Step 1 and 2 **(c)** Step 3 **(d)** Step 4

Figure 3: CBA: **(a)** initial map; **(b)** squares are placed at country-barycenters; overlaps are removed; **(c)** a subgraph induced by a country is scaled to fit inside its square; **(d)** a spring embedder pulls nodes towards original positions; flexible boundaries ensure contiguity.

2 Related Work

Using maps to visualize non-cartographic data has been considered in the context of spatialization [21, 7]. Work at PNNL resulted in document visualization systems Themescape [26], and successive systems Spire and In-Spire. Self organizing maps (SOM), coupled with geographic information systems, render 2D maps of textual documents [20]. Similarly, maps of science show groups of scientific disciplines [3]. Recent systems include VOSviewer [23] and the Sci2 system [2]. Maps of computer science (MOCS) [8] provide a way for visual exploration of topics in a particular conference or journal.

Augmenting node-link diagrams with spatial features can improve graph revisitation tasks [12]. This is used in visualizations that explicitly draw boundaries to indicate the grouping: BubbleSets [5], LineSets [1], and Euler diagrams [25, 16, 19]. When providing an underlying overlap-free clustering, the first two techniques would generate contiguous but potentially overlapping regions. Instead, Euler diagram generation methods would produce contiguous and non overlapping regions, but would ignore the connectivity or the initial embedding of the graph.

The geographic map metaphor was also used for visualizing recommendations, where the Graph-to-Map approach (GMap) was introduced [9]. 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 [13]. In GMap the visualization is 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. 2.

A recent study compared several techniques for displaying clusters with node-link diagrams [14]. They found GMap improved performance of searching and exploration tasks, compared to standard node-link diagrams, BubbleSets [5], and LineSets [1]. 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.

3 Algorithms for Creating Contiguous Maps

Given a high-dimensional data (*e.g.*, similarity between universities in Fig. 1, or international trade patterns in Fig. 2), we extract an edge-weighted graph and create a drawing in which similar nodes are placed close to each other, and groups of similar nodes are enclosed in contiguous regions.

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, that is, positions $p_v \in \mathbb{R}^2$ for all nodes $v \in V$, and a clustering $C = \{C_1, \ldots, C_k\}$ 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 $d(u, v) = -\log[0.9 \cdot w(u, v) + 0.1]$. In some scenarios, positions of the nodes can also be a part of input. For example, if the nodes model actual cities, the positions are their geographic coordinates.

We first describe an *embedding-based algorithm (EBA)*, which preserves a given graph layout. There are many applications where the input specifies the 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. There are many applications where the input specifies the clustering of the nodes (e.g., party affiliation in political networks, countries grouped by continent). CBA is well suited for such settings.

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. 2.

3.1 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 [15].* 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. The operation is applied only if the new clusters are still contiguous. 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; in practice the process converges quickly for all graphs in our datasets.

Any geometric clustering algorithm can be used in the first step; we choose K-means as it is efficient, simple to implement, and works well with the iterative refinement in the second step. 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 [22].

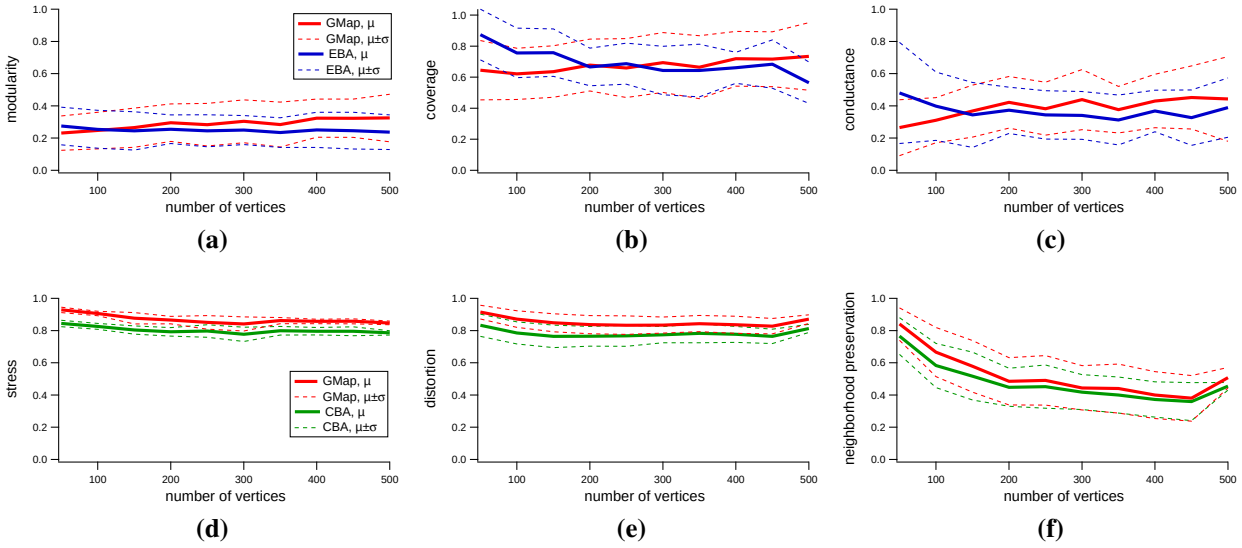


Figure 4: Comparison of the embedding and clustering metrics over all graphs, where the means μ are solid thick lines and the standard deviations σ are thin dashed lines of the same color; the number of nodes in the graph is $|V| \in \{50, 100, \ldots, 500\}$.

3.2 Clustering-Based Algorithm

We assume the input graph is given with fixed node clustering and some initial (non-contiguous) layout. The graph is processed in the following four steps; see Fig. 3b. The algorithm takes as an input graph clustering and computes an original embedding using an existing graph layout algorithms. The nodes are initially placed in positions that ensure cluster contiguity and then subsequently pulled toward their original positions using a force-directed technique. The steps taken by the algorithm are the following.

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 [6]. 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 [18]. 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.

Note that the algorithm can also be applied in the setting with a given clustering and embedding. In most cases the resulting map reproduces positions of nodes of the original layout with good precision. The nodes of small fragments tend to be placed along the border of the country, in the direction of the original fragment; see Figure 3.

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 and 0 indicating the worst one.

4.1 Data Sources

We use 10 datasets, creating a total of 70 maps. Half of them (Amazon crawl for book titles, `last.fm` crawl for music bands, GD co-authorship graph, university similarity, and international trade data) are from [13]. The others are extracted from DBLP titles for conferences, journals, and authors using the MOCS system [8]. The maps contain $|V| \in \{50, 100, \ldots, 500\}$ nodes in the underlying graph.

4.2 Metrics for Graph Layout

Stress: The stress of an embedding is a classic MDS metric, which measures the energy of the spring system [11]:

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

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 random variables with correlation coefficient:

$$r = \frac{\sum_{(u,v)\in E} (\Delta_{uv} - \overline{\Delta})(d_{uv} - \overline{d})}{\sqrt{\sum_{(u,v)\in E} (\Delta_{uv} - \overline{\Delta})^2 \sum_{(u,v)\in E} (d_{uv} - \overline{d})^2}},$$

where $\overline{\Delta}$ and $\overline{d}$ are the mean values of the corresponding distances. Since the correlation coefficient 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 [24]. As suggested in [10], we use $T = 20$ in our experiments.

4.3 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 [4].

Conductance: The conductance of a cut compares the weight of a cut and the weight of the edges in either of the two subgraphs induced by that cut [4, 17]. The conductance $\phi(C)$ of any given cluster C can be obtained as

$$\phi(C) = \frac{\sum_{v\in C}\sum_{u\notin C} w(v,u)}{min(a(C), a(\overline{C}))},$$

where $a(C) = \sum_{v\in C}\sum_{u\in V} w(v,u)$ is the sum of the weights of all edges with at least one endpoint in C. This value represents the cost of one cut that bisects G into two node sets C and $V \setminus C$. The conductance of the clustering is the average value of those cuts: $\phi(G) = 1 - \sum \phi(C)/|C|$.

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 [17]:

$$\frac{\sum_{uv} w(u,v)\delta(c_u,c_v)}{\sum_{uv} w(u,v)}.$$

4.4 Results

Since EBA preserves the given embedding, we only report its performance on the clustering metrics; see Fig 4(a-c). 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 4(d-f). On average, CBA produces layouts with worse embedding metrics. The average the 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 in practice, taking only 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.

A software implementation of the algorithms is available at `http://gmap.cs.arizona.edu`. The website allows to generate maps on custom input and export them in several formats.

5 User Study

In order to confirm our suppositions on the negative effect of map fragmentation, we conducted a user study in which we asked participants to evaluate several map-like visualizations and perform some basic tasks using the maps. The primary hypotheses addressed in the experiment are as follows.

H1 People are confused by fragmentation in maps.

H2 People perform faster and with greater accuracy using contiguous maps for several real-world tasks.

H3 People find contiguous maps more appealing rather than their fragmented equivalencies.

5.1 Experimental Set Up

During the experiment, an unsupervised survey was accessible online (`http://userstudy-maps.appspot.com`). Using social networks and email lists, we asked our students and colleagues to participate. An estimated completion time was about 10 minutes. Every participant was randomly and secretly assigned to one of the two groups. For the first group (referred to as "uninstructed"), we provided brief instructions about the system and the experiment. The uninstructed users were only told that *"the diagrams used in the study are created with a method for representing data as geographical maps. Like a standard geographical map, each of these diagrams contains countries, cities and connections between cities."* In contrast, for the alternative "instructed" group, we additionally explained the map metaphor for graph visualization, the meaning of "countries", "cities", and "road" between them. We also introduced the term *fragmentation*, and showed an example with contiguous and fragmented maps pointing out that "the same country can be split into two or more fragments but countries are always identified by a unique color". This partitioning into instructed and uninstructed groups was designed to assess how the encoding used in these maps coincide with the user intuition.

The study was divided into two parts. In the first one, participants were given a simple real-world task, which may be performed using map-like data visualization. Specifically users were asked the following questions:

Task 1a: *How many countries do you see?*

Task 1b: *How many cities are in the country pointed by the arrow?*

Task 1c: *Which country has more cities?*

For each of the questions, a user was given a map with respectively zero, one or two highlighted countries. We used 5 graphs for each task for which we created 2 versions (contiguous and fragmented) of maps. This yielded 30 questions for the first part of the study in total. In Tasks 1a and 1b, we asked users to count the number of countries and cities and select the relative range (we provided as list of possible answers $0-3$, $4-6$, $7-9$, $10-12$, $13-15$, ≥ 16 for Task 1a, and $0-5$, $6-10$, $11-15$, $16-20$, ≥ 21 for Task 1b). For the Task 1c, where we specified to estimate the country size rather than counting the number of cities, there were only 3 options ("Red Arrow" country is bigger, "Blue Arrow", and "Equal Size"). For each question, the accuracy and response time were recorded. The questions of the Task 1 were designed so that to evaluate Hypotheses H1 and H2.

The second part of the experiment was more open-minded. We asked users to take their time and motivate the answers. We also provided a text field and encouraged participants to explain their decisions. The part consisted of two questions:

Task 2a: *Which of the maps do you like more?*

Task 2b: *Please rate the map.*

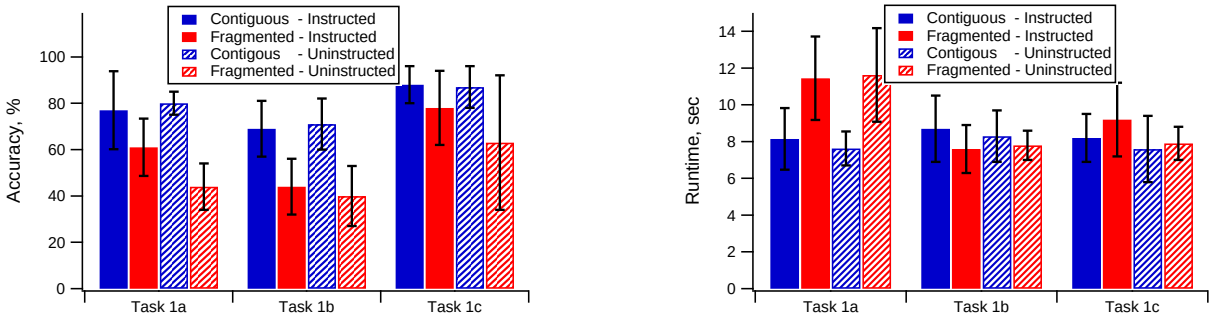


Figure 5: Mean and standard deviation of (a) user accuracy and (b) response time for Task 1.

For the Task 2a, we presented pairs of images with contiguous and fragmented maps constructed for the same graph. There were 10 pairwise comparisons per user. Note that here our purpose was to test the user intuition; hence, we did not define neither explain what map is considered as the ideal one. Task 2b involved selecting a rate for the presented map. Participants could choose between "Poor", "Fair", "Good", "Very good", and "Excellent" options. A collection of 20 images was generated for the question: 5 fragmented ones, and 5 for each of the three our methods described in Section 3. Both questions of the section were designed to assess the hypothesis H3.

The images used in Tasks 1 were generated with one of our algorithms for creating contiguous maps. We used graphs with $50 - 150$ vertices in the experiment from the real-world dataset described in Section 4. Since the questions in Task 1 had the same answers for both the fragmented and the contiguous map of the same graph, participants who would recognisee the particular input instance could answer the second time without actually re-calculating the required value. This would bias the user study results and give an unfair time advantage to the questions that happened to appear later on the user study. In order to make fragmented and contiguous maps of the same graph less recognizable, we applied a series of simple transformations. The contiguous maps were rotated by an angle of

$$\text{randFromTo}(20, 70) + \lfloor \text{randFromTo}(0, 4) \rfloor * 90 \quad \text{degrees},$$

and the colors used for identifying countries were scrambled.

To allow for fair pairwise comparison, the maps used in the Task 2a were not modified. All maps were rendered at low resolution in order to reduce their loading time. However, the countries and cities were still well-distinguishable. The questions were presented ordered by their corresponding task. Within each task, the images were shuffled for every participant to prevent a training effect.

5.2 Results

Of the 64 people who started the study, we processed the results from the 52 who completed all questions. Among the participants, the average age was 26 years and 37 were male.

To investigate hypotheses H1 and H2, we classified user responses for the Task 1 into four groups depending on the type of study (instructed/uninstructed) and map (fragmented/contiguous). For each group, we compute the *mean* and the *standard deviation* of accuracy of user answers; see Figure 5a. Consistently, there was a large difference in accuracy between contiguous and fragmented maps for all three tasks. The

I like *contiguous* maps because...
data fragmentation looks weird
it looks simple
countries as single entity and one shape are easy to read
all regions are contiguous. no disjoint regions
its easier to make quick guesses on graph properties
items in each "country" are consolidated together
I like *fragmented* maps because...
it looks more real
I like neither of maps because...
don't know for what i'm searching

Table 1: User comments for the Task 2a.

difference was statistically significant for the uninstructed group for all three tasks and for the instructed group for the Task 1b ($p < 0.001$, t-test). This result supports our hypothesis H2; hence, maps without fragmentation do provide an advantage for performing particular real-world tasks. Somewhat unexpected, even in the simplest scenario (Task 1a for contiguous maps with $5 - 8$ countries per map) accuracy of user responses reaches only 80%. Our detailed investigation showed that participants were often confused by similar colors, and considered separate regions as two fragments of the same country.

We have not noticed a difference in accuracy between the instructed and uninstructed versions of the study for Task 1 on contiguous maps. In contrast, if the input map is fragmented, then the instructed users perform significantly better. This suggests that hypothesis H1 can be accepted. We stress here that in our experiments we manually chose the maps with high amount of fragmentation (some countries contain up to $10 - 15$ smaller fragments). At the same time, "mostly non-fragmented" maps might be useful and meaningful. Future work might explore to what extent the fragmentation in maps is confusing for users.

The analysis of time needed for a participant to give an answer for Task 1 is in Figure 5b. Before the time analysis, we excluded the responses that required more than 90 seconds. An investigation showed that such long response times indicate connectivity problems with our online system. For the filtered data, an average response time is 8.5 seconds per question with standard deviation 5.7 seconds. The Task 1a appears to be the most difficult in its fragmented variant with over 12 seconds per question. For the Tasks 1b and 1c, participants showed mostly the same performance. All the differences are not statistically significant.

Finally, we analyzed responses for Tasks 2a and 2b to investigate the third hypothesis on aesthetical preferences of contiguous maps. There was a strong agreement among respondents as to what representation appears to be the "nicest". A majority of participants (over 92%) prefer the contiguous variant, while only 3.5% like the fragmented one. For the Task 2a, we collected comments explaining user preferences, see Table 1 for a subset of the most popular user comments. Although we did not explain the participants the research goal of the study, most of them chose the contiguous variant because it had less "islands" and "regions". For the Task 2b, we see a significant preference of both our algorithms over the fragmented version. There is no difference between instructed/uninstructed datasets.

6 Conclusions and Future Work

We suggested and evaluated two approaches for generating fragment-free maps. These approaches can be applied in different scenarios depending on the type of input data and user preferences. The simpler but more efficient embedding-based algorithm tend to produce maps with slightly worsen clustering metrics. The second algorithm keeps the original clustering and also capable to preserve a given embedding. We also developed a publicly available system that can be used to generate contiguous maps for a given input graph. See a video accompanying the submission for more details.

Although contiguous maps are generally more appealing, fragmentation might encode important information. If a map is drawn properly, fragmentation may indicate close relationship of fragmented objects with several countries. An interesting direction for future work is identify cases in which fragmentation is "meaningful" and should be presented on a map. It would be interesting to perform an in-depth user study comparing the algorithms. Such a study would investigate the effect of the quality of clustering and embedding on map comprehension.

Acknowledgements. We thank Joe Fowler and Yifan Hu for productive discussions and help with the implementation of the algorithms and the online tool.

References

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

[2] K. Börner. Plug-and-play macroscopes. *Communications of the ACM*, 54(3):60–69, 2011.

[3] K. W. Boyack, R. Klavans, and K. Börner. Mapping the backbone of science. *Scientometrics*, 64:351–374, 2005.

[4] U. Brandes, M. Gaertler, and D. Wagner. Experiments on graph clustering algorithms. In *In 11th Europ. Symp. Algorithms*, pages 568–579. Springer-Verlag, 2003.

[5] C. Collins, G. Penn, and S. Carpendale. Bubble sets: Revealing set relations with isocontours over existing visualizations. *IEEE Trans. Vis. and Comp. Graphics*, 15(6):1009–1016, 2009.

[6] T. Dwyer, K. Marriott, and P. J. Stuckey. Fast node overlap removal. In M. Kaufmann and D. Wagner, editors, *International Symposium on Graph Drawing (GD06)*, volume 4372 of *Lecture Notes in Computer Science*, pages 153–164. Springer, 2007.

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

[8] D. Fried and S. G. Kobourov. Maps of computer science. In *7th IEEE PacificVis Symposium*, 2014. To appear.

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

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

[11] E. R. Gansner, Y. Koren, and S. North. Graph drawing by stress majorization. In J. Pach, editor, *International Symposium on Graph Drawing (GD04)*, volume 3383 of *Lecture Notes in Computer Science*, pages 239–250. Springer, 2004.

[12] S. Ghani and N. Elmqvist. Improving revisitation in graphs through static spatial features. In *Proceedings of Graphics Interface 2011*, GI '11, pages 175–182, 2011.

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

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

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

[16] P. Rodgers, L. Zhang, and A. Fish. General Euler diagram generation. In G. Stapleton, J. Howse, and J. Lee, editors, *International Conference on Diagrams (Diag08)*, volume 5223 of *Lecture Notes in Computer Science*, pages 13–27. Springer, 2008.

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

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

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

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

[21] A. Skupin and S. I. Fabrikant. Spatialization methods: a cartographic research agenda for non-geographic information visualization. *Cartography and Geographic Inf. Sci.*, 30:95–119, 2003.

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

[23] N. J. van Eck and L. Waltman. Software survey: VOSviewer, a computer program for bibliometric mapping. *Scientometrics*, 84(2):523–538, 2010.

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

[25] A. Verroust and M.-L. Viaud. Ensuring the drawability of extended Euler diagrams for up to 8 sets. In A. F. Blackwell, K. Marriott, and A. Shimojima, editors, *International Conference on Diagrams (Diag04)*, volume 2980 of *Lecture Notes in Computer Science*, pages 128–141. Springer, 2004.

[26] J. A. Wise, J. J. Thomas, K. Pennock, D. Lantrip, M. Pottier, A. Schur, and V. Crow. Visualizing the non-visual: spatial analysis and interaction with information from text documents. In *INFOVIS*, pages 51–58, 1995.