Visualizing Graphs and Clusters as Maps

Article *in* IEEE engineering in medicine and biology magazine: the quarterly magazine of the Engineering in Medicine & Biology Society · November 2010

DOI: 10.1109/MCG.2010.101 · Source: PubMed

3 authors:

Emden Gansner
self
96 PUBLICATIONS **6,084** CITATIONS
SEE PROFILE

Yifan Hu
Yahoo
92 PUBLICATIONS **6,326** CITATIONS
SEE PROFILE

Stephen G. Kobourov
The University of Arizona
285 PUBLICATIONS **4,893** CITATIONS
SEE PROFILE

Visualizing Graphs and Clusters as Maps

Emden R. Gansner*
AT&T Labs - Research, 180 Park Ave, Florham Park, NJ 07932

Yifan Hu†
AT&T Labs - Research, 180 Park Ave, Florham Park, NJ 07932

Stephen G. Kobourov‡
University of Arizona, 1040 E 4th Street, Tucson, AZ 85721

ABSTRACT

Information visualization is essential in making sense out of large data sets. Often, high-dimensional data are visualized as a collection of points in 2-dimensional space through dimensionality reduction techniques. However, these traditional methods often do not capture well the underlying structural information, clustering, and neighborhoods. In this paper, we describe GMap, a practical algorithmic framework for visualizing relational data with geographic-like maps. We illustrate the effectiveness of this approach with examples from several domains.

Keywords: Information Visualization; Clustering; Graph Drawing; Graph Coloring; Maps; Set Visualization.

INTRODUCTION

With the growth of the Internet and scientific and technological advances, the world has seen an explosion in the generation and collection of data. This data is often relational, high-dimensional, or both. With such a wealth of data comes the need to understand it, analyze it, and use it. Visualization can play a key role in all of these steps, but especially so in the exploratory phase.

Many useful techniques for visualizing abstract data sets have been considered in the past. Graphs made of nodes and links (or edges) are often used to capture the relationships between objects, and graph drawing allows us to visualize such relationships. Typically nodes are represented by points in two or three dimensional space, and edges are represented by lines between the corresponding vertices. For an example of a relational data set, consider the Amazon.com graph, where books are nodes and there is an edge between two books if people who have bought one of the books also buy the other.

High-dimensional data sets, on the other hand, are often visualized as point clouds in two or three dimensional space. An example is the listening pattern of users of the `last.fm` website. Each user can be represented by a long vector. The dimension of the vector is the same as the number of musicians available at the website, and the value of each element is proportional to the number of times a user listens to the corresponding musician. In a point cloud visualization, each point is a user and two points are close to each other if the two users have similar music taste.

Unfortunately, these standard approaches of node-link diagrams and point cloud representations often require considerable effort to comprehend. If we desire to make the data accessible to people outside the areas of computer science and statistics, providing more compelling drawings is an important task. This is certainly true for the ordinary user who might be curious about the recommendations made by Amazon.com, but is also true for the biologist who wishes to understand the visualization of a biological experiment that was created by a statistician.

When large data sets are drawn as point clouds or node-link graphs, it is sometimes possible to observe a visual similarity to geographic maps. For example, several small connected components next to a much larger component suggest several small islands next to a big continent. We took the next step in visualizing data directly as a geographic map. A map representation is familiar and intuitive; most people are very familiar with maps and nicely drawn maps often provide hours of enjoyable exploration. Many people take advantage of this familiarity with maps and manually create map-like representations that portray relations among abstract concepts. We were interested in automating the process which begins with the data and ends with a drawing of a map. In this process there are two main competing goals: First, we would like to be faithful to the data, by maintaining the inherent structure and relationships, without adding or implying non-existent structure. Second, we would like the final result to look like a map, using standard cartographic conventions. Our attempt to address these two goals led to a framework for a map-based data representation that we call GMap.

GMAP

The GMap framework allows us to generate map-like representations from an abstract data set. Specifically, given a high-dimensional data set or a graph with edge weights (e.g., the similarity between books as determined by purchase behavior at Amazon.com), it produces a drawing with a map-like look, with countries that enclose similar objects, outer boundaries that follow the outline of the vertex set, and inner boundaries that have the twists and turns found in real maps. A typical example is in Figure 1 and shows just under 1000 books.[1] Our maps also can have lakes, islands, and peninsulas, similar to those found in real geographic maps.

GMap is a framework in the true sense of the word, rather than a specific algorithm. It consists of four main steps, the first two steps can be achieved by a variety existing algorithms. For the last two steps we propose new algorithms.

In the first step we take as input a graph or high-dimensional data set, and embed it into the plane. The statistics and scientific modeling communities have extensively explored this problem and provide many ways of doing this. Possible embedding algorithms include principal component analysis, multidimensional scaling (MDS), force-directed algorithms, or non-linear dimensionality reductions such as Locally Linear Embedding and Isomap.

The second step takes this collection of points in the plane and aggregates them into clusters. Here, it is important to match the clustering algorithm to the embedding algorithm. For example, a geometric clustering algorithm such as k-means may be suitable for an embedding derived from MDS, as the latter tends to place similar points in the same geometric region with good separation between clusters. On the other hand, with an embedding derived from a

*e-mail: erg@research.att.com
†e-mail: yifanhu@research.att.com
‡e-mail: kobourov@cs.arizona.edu. Work done while this author was at AT&T Labs.

[1] Most of the images in this article are available in an interactive form at `http://www.research.att.com/~yifanhu/MAPS/imap.html`.

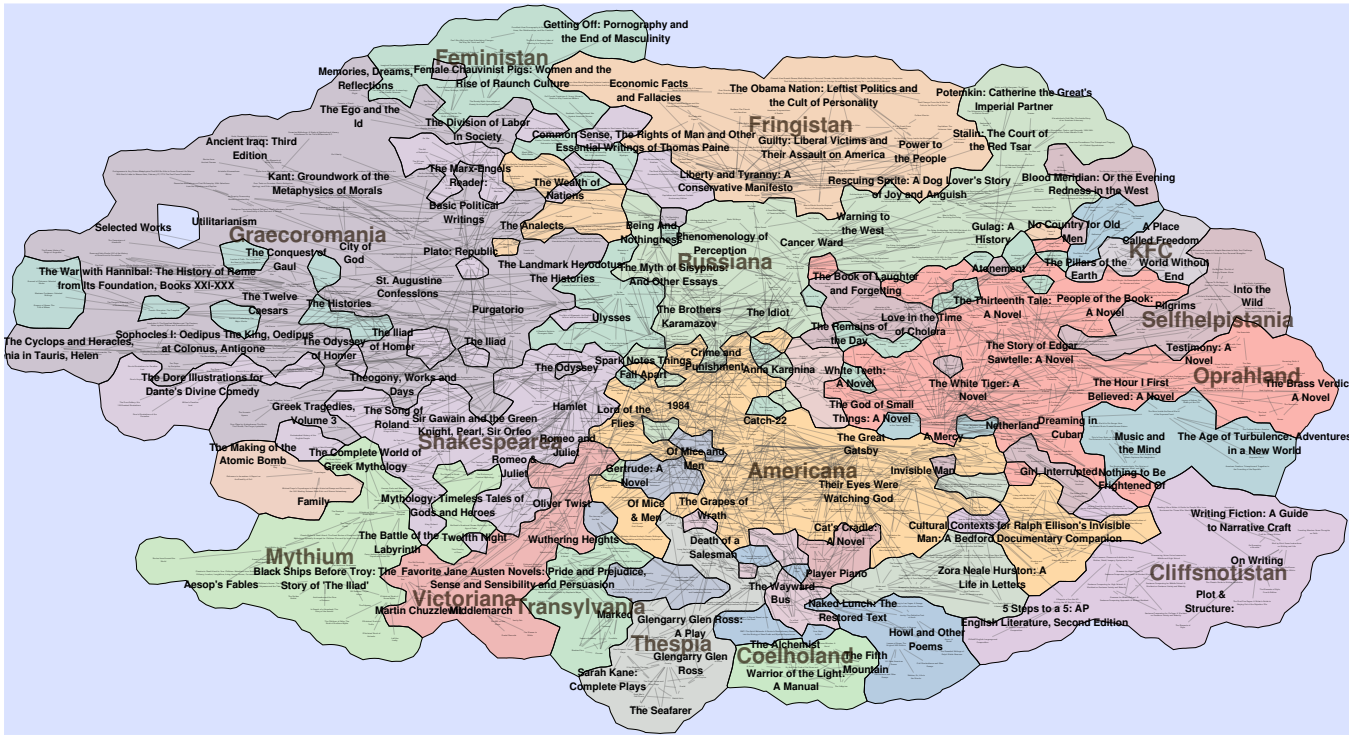

Figure 1: A map of books related to "1984" from Amazon.com

force-directed layout, a modularity based clustering [9] could be a better fit. The two algorithms are strongly related, and therefore we can expect vertices that are in the same cluster to also be physically close to each other in the embedding.

In the third step, we use the two-dimensional embedding together with the clustering to create the actual map by delineating country boundaries, carving continental outlines, and separating islands from continents. This can be accomplished with the help of plane partitioning techniques such as Voronoi diagrams, but with new algorithmic techniques to ensure realistic looking outer and inner boundaries.

In the fourth and final step, we add additional graphical attributes to the drawing in order to enhance its clarity, to serve as keys to the abstract data, or to simply make it more aesthetically appealing. This would involve assigning an appropriate set of colors to the various regions. We propose a spectral algorithm that maximize color difference between neighboring countries. In addition, we could add mountains to the map or overlay it with a heat map to indicate scalar values associated with geographic positions.

While the first two steps have been the subject of hundreds of papers, the last two steps in this process are new. With this in mind, in the next sections we describe how we put together the embedding and clustering information so that we can create the map representation and make it aesthetically appealing. Our presentation is narrative and informal. We refer the interested reader to the article [5] for technical details and more references.

Making the map

Given the placement of the points from the first step, and their clustering from the second, we want to create a map, with inner boundaries separating points not in the same cluster and outer boundaries preferably following the general outline of the point set. A naive approach for creating the map is to form the Voronoi diagram of the vertices based on the embedding information, together with four points on the corners of the bounding box. This is illustrated in Figure 2(a). Such maps often have sharp corners, and angular outer boundaries. We can generate more natural outer boundaries by adding random points to the current embedding. A random point is only accepted if its distance from any of the real points is more than some preset threshold. Note that this step can be implemented efficiently using a suitable space decomposing data structure, such as a quadtree. This leads to boundaries that follow the shape of the point set. In addition, the randomness of the points on the outskirts gives rise to some randomness of the outer boundaries, thus making them more map-like, as seen in Figure 2(b). Furthermore, depending on the value of the threshold, this step can also result in the creation of lakes and fjords in areas where vertices are far apart from each other. Nevertheless, some inner boundaries remain artificially straight.

At this point, we still note the undesirable feature that the "countries" all have roughly the same area (Figure 2(b)), whereas we might prefer some areas to be larger than others (e.g., due to the importance of the entities they represent). As an illustration, in Figure 2, we assume that "node 1" is more important than the other

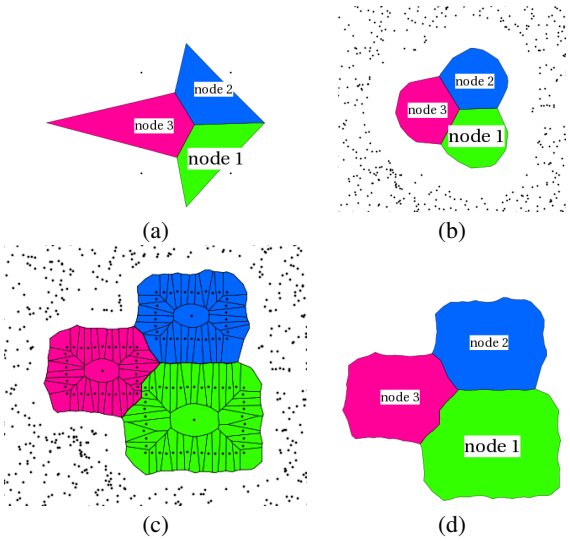

Figure 2: (a) Voronoi diagram of vertices and corners of bounding box; (b) better construction of outer boundaries through placement of random points; (c) Voronoi diagram of vertices and points inserted around the bounding boxes of the labels; (d) the final map.

two nodes, and use a larger label for that area. [2] To make areas follow the shape of the labels, we first generate artificial points along the bounding boxes of the labels as shown in Figure 2(c). To make the inner boundaries less uniform and more map-like, we perturb these points randomly instead of running strictly along the boxes. Here Voronoi cells that belong to the same vertex are colored in the same color, and cells that correspond to the random points on the outskirt are not shown. Cells of the same color are then merged to give the final map in Figure 2(d). Note that instead of the bounding boxes of labels, we could use any 2D shapes, e.g., the outlines of real countries, in order to obtain a desired look and proportion of area, as long as these shapes do not overlap.

We note that not all real maps have complicated boundaries. For example, boundaries of the western states in the United States often have long straight sections. We believe that irregular boundaries are more typical of historical and geographic boundaries, and lead to more map-like results. But this is a matter of personal taste and our technique can generate maps of both styles.

When mapping vertices that contain cluster information, in addition to merging cells that belong to the same vertex, we also merge cells that belong to the same cluster, thus forming regions of complicated shapes, with multiple vertices and labels in each region. At this point we can add more geographic components to strengthen the map metaphor. For instance, in places where there is significant space between vertices in neighboring clusters, we can add lakes, rivers, or mountain ranges to the map to indicate the distance.

With the regions determined, we have a representation of the data in which closely related objects, as determined by the graph topology and possibly edge weights, are drawn closely together. This geometric information is then used to discover clusters among the objects. To emphasize the clusters, each is represented as a collection of geometric regions.

Figure 3 shows the difference between a typical node-link graph layout and the result of applying GMap to the same data. (Indeed, the graph layout is first step of the GMap layout.) The data represents the graph of author collaborations between 1994 and 2004 at the Symposium on Graph Drawing. The upper drawing exhibits the connected components and closely related nodes are indicated by proximity, but cluster structure is only hinted at. In the GMap version, the cluster structure is obvious. Coloring the nodes in the node-link drawing would still only imply the clusters. The GMap figure makes the clusters explicit as well as indicating strong cluster relations where two clusters share a border.

When projecting high dimensional data into low dimensional space, distance distortion is inevitable, and the resulting figure will often have some anomalies and distortions. Thus, some strongly related objects may be separated by seemingly unrelated objects. For example, in Figure 3, we see a cluster of North American authors split into three components in the southwest part of the map (the three beige components). The authors in the central component have had much more collaboration with European authors; those in the bottom-left component much less so. There is a singleton component sitting in the middle, reflecting the fact that this author, "North", collaborated with both the European authors, and with the group of authors that form the bottom-left component. Such fragmentations are inherent in the embedding and clustering algorithms used in the first two steps. However we have proposed ways [5] to use the clustering information to adjust the layout, so that the regions of countries are more contiguous, at the expense of some loss of relational information captured in the original embedding.

We believe that the map-like visualization better captures some of the cluster structure while at the same time providing an illustration that is more attractive to the typical user than the traditional scatterplots and node-link diagrams. Although we have not yet run a formal user study, anecdotal evidence from our customers and colleagues corroborates this claim.

Coloring the map

The final step, annotating the map with graphical attributes, involves assigning good colors to the countries in our maps. The Four Color Theorem states that only four colors are needed to color any map so that no neighboring countries share the same color. It is implicitly assumed that each country forms a connected region. This result, however, is of limited use to us because countries in our maps are not always connected. Therefore, we will have to use one unique color for each cluster to avoid ambiguity. Estimation of the number of colors an "average human" can discriminate, when color pairs are presented side by side, ranges from tens of thousands to a million. On the other hand, it takes much more effort when comparing colors that are similar. A further limiting factor is that 5% of males are color blind, which rules out certain coloring schemes. Finally, the limited palette of map-like coloring schemes reduces the choice of colors even more.

In GMap, we start with a coloring scheme from ColorBrewer (`www.colorbrewer.org`), and generate as many colors as the number of countries by blending the base colors. As a result, our color space is linear and discrete. Because of the blending, any two consecutive colors in the linear array of colors are similar to each other. When applying these colors to the map, we want to avoid coloring neighboring countries with such pairs of colors.[3] With this in mind, we need to solve a discrete optimization problem where we find the best color permutation such that the color difference between neighbors is maximized overall. This is achieved by a combination of two procedures. They both operate on the *country graph*. The country graph is created from the original graph after it has been embedded and clustered. Specifically, the country graph contains a node for each country, with each edge representing two

[2] A weighted Voronoi diagrams can be used to make the area of each Voronoi cell proportional to its weight. We do not use this approach, however, because we want the Voronoi cell to also contain a specific shape, e.g., the bounding box of a label.

[3] Although two non-neighboring countries with similar colors can lead the viewer to believe that they are disjoint regions of the same country, this problem diminishes when the two countries are sufficiently far apart, as it is unlikely that distant regions belong to the same cluster.

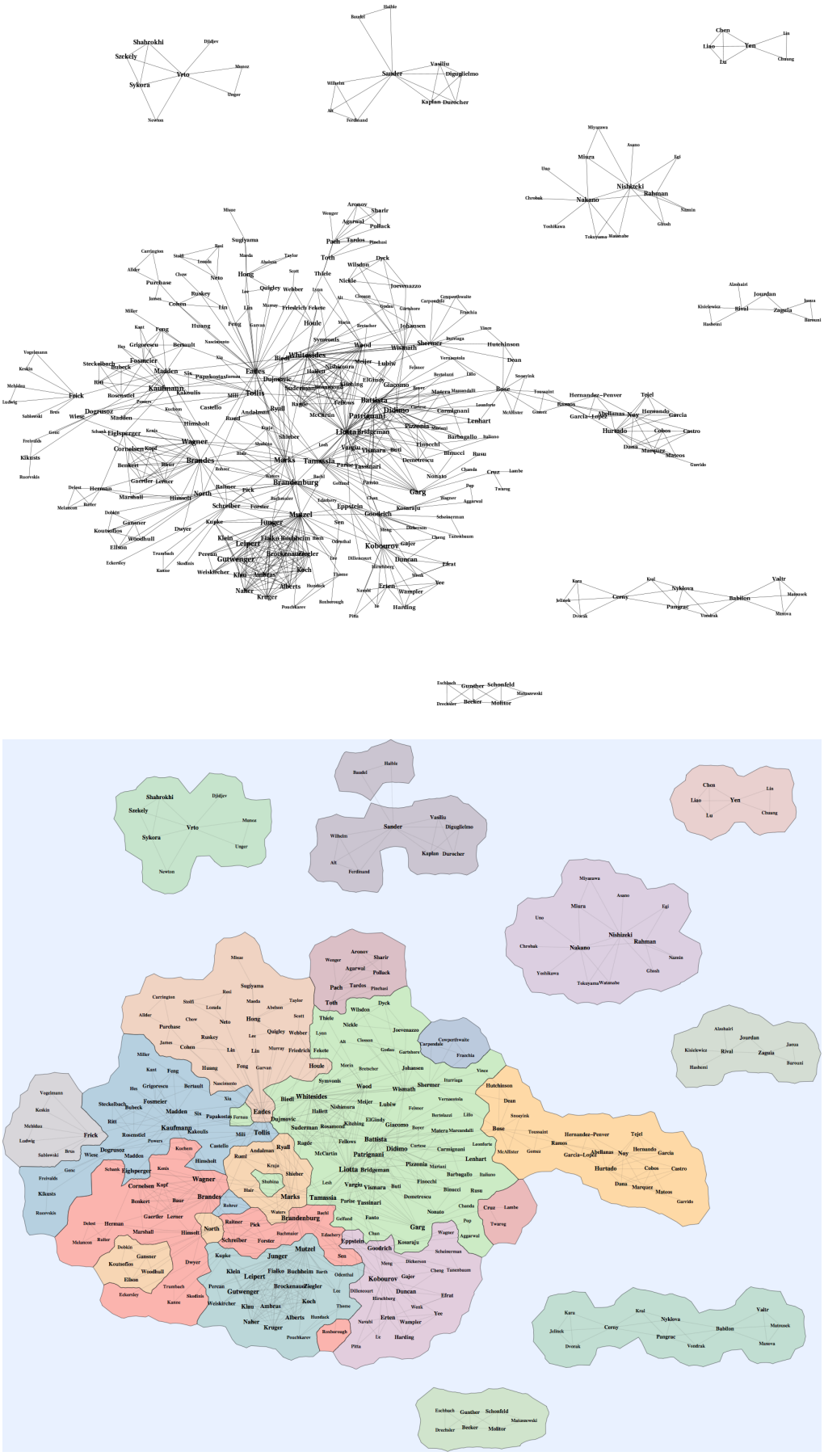

Figure 3: Node-link drawing compared to the GMap representation

countries that share a border. The first procedure turns the map coloring problem into that of a continuous optimization problem on the country graph,

$$\max \sum_{\{i,j\}\in E_c} w_{i,j}(c_i - c_j)^2, \text{ subject to } \sum_{k\in V_c} c_k = 1, \qquad (1)$$

where E_c is the set of edges in the country graph, scalar value c_i is the color (index) assigned to country i, and weighting factor $w_{i,j}$ measures how important it is to promote color difference between country i and country j. The solution is the largest eigenvector of the weighted Laplacian of the country graph. We then use the ordering induced by the values of the eigenvector as the color permutation. This is then followed by a greedy color swapping procedure to see if the color difference can be further improved. This heuristic algorithm improves the color difference between neighbors significantly compared with a random color assignment.

Figure 4 illustrates the difference between a simple, random coloring and a coloring obtained using our approach. Note, for example, that in the random coloring, the countries in the middle are assigned colors in a green-gray palette, making it difficult to distinguish them. In the lower, optimized coloring, the separate regions in the central part are much more evident.

USING GMAP TO VISUALIZE SET RELATIONS

GMap is designed for visualizing cluster relations as maps, where each item is assumed to belong to one cluster and hence one country. However, the same algorithm can be easily adapted to visualize multiple relations among a set of objects. Byelas and Telea [3] proposed visualizing set relations using deformed convex hulls. Collins et al. [4] proposed "bubblesets," based on isocontours. GMap provides a different approach.

To apply GMap for visualizing sets, we work on one set at a time. Items not in the active set are considered obstacles, and treated the same as random points inserted in the GMap algorithm. Applying GMap then gives us a map of one country. By repeating this process for multiple sets, and using transparency to allow overlapping parts of the regions to be seen, we achieve a visualization of multiple sets.

To avoid a country being disconnected, we add edges to link items in the same set. The edge addition process is similar to that of Collins et al. [4]. Edges are then routed as splines to avoid hitting items not in the active set, when possible. Artificial points are inserted along the spline edges. When GMap is applied, distant items are connected by "bridges" along the edges. Figure 5 shows a group of 55 photos from the Yale Face Database [7], representing portraits using different expressions and lightings. The photos are embedded in 2D using MDS. The distance between two photos is calculated using principal component analysis of a matrix of 55 rows, with each row a vector of the pixel values of a photo. Even with careful pre-processing, the embedding does not always put the same persons in the same neighborhood. Without set visualization, it is difficult to identify all photos of the same person. Figure 5 shows three sets of photos, each set belonging to a single person. As can be seen, each country is connected, and each avoids photos not in the set. This example shows that GMap provides a good alternative for set visualization.

GMAP CASE STUDIES

The GMap layout is meant to accentuate clusters in fairly large graphs. Gleaning information from the maps typically involves an interactive, multi-scale process, similar to that used for exploring geographic maps. One views the map at small scale to sense the overall layout, the major regions, and how they relate to each other. One then zooms in to see local detail, and to traverse the map along small features. At some point, one may zoom out again to put the local details into a global context.

Based on this style of use, GMap figures are most effective when displayed as a large image, often a meter or more in width, or via an interactive viewer. In the former case, the user can physically move to change the scale. In the latter case, the viewer provides the scale change and, at the same time, can provide some version of semantic zoom, so that more detail is added the more the user zooms in. In addition, an interactive viewer can provide such additional features as textual search or links connecting a feature on the map to some external information. For example, clicking on a book shown in the BookLand layout might take the user to its entry at Amazon.com.

Given the page size limitations of a traditional journal, the figures included here are scaled to illustrate some features of the GMap layout (Figures 4 and 8) or to give a high-level view of such maps. The reader is encouraged to explore some GMap layouts at the site indicated in Footnote 1.

We now consider how GMap works in practice with various different data sets. In the first, we visualize the "landscape" of music artists. In the second, we visualize the "landscape" of books as implied by user purchase behavior at Amazon.com. In the third, we visualize international trade data. Finally, we look at TV programs based on users' viewing habits. In our implementation we use a scalable force-directed layout algorithm for step one and a modularity-based clustering algorithm for step two.

It is important to note that in all cases the countries and their geography in the resulting maps are not part of the input data, but emerge from the graph layout and clustering algorithms. This gives the user a potential tool to discover structure based solely on local data.

MusicLand

To create a land of music, we collected data from a web crawl of the `last.fm` website. As an Internet radio and music community website, it has over 30 million users, and recommends music based on user profiles. Over several years, the recommender system has collected information about how one band/musician/composer/artist is related to another in terms of how many listeners of one also enjoy the other. For each composer, the last.fm website lists the top 250 related composers. For example, Beethoven is considered to have "super similarity" to Mozart, Bach, and Brahms, "very high similarity" to Mendelssohn, Schumann, and Vivaldi [4] The website also provides the number of listeners of each musician. In April 2009, we crawled the website by starting with Beethoven, and the top 20 musicians most similar to Beethoven, provided that each has at least 100,000 listeners[5]. We then found the top 20 most similar musicians to each of those with at least 100,000 listeners and proceed recursively. Our crawl yielded a graph with 2782 musicians, with edge weights corresponding to the similarity between musicians. We further pruned this graph by only taking edges that have "super similarity". Finally, ignoring singletons, we end up with 2588 vertices. We then laid out the graph, clustered the vertices, and generated the MusicLand map as shown in Figure 6. We next examine some of the countries in MusicLand in more detail.

The Mainland: The vast majority of musicians/bands is located in a single continent. While it is not as easy to spot major trends along the main axes, many of the clusters are well-defined and neighboring clusters make sense from a musical point of view. A cluster of classic rock on the east shore begins with *Eric Clapton* and *Janis Joplin* in the south, goes through the *Who* in the middle

[4]Note that we do not know last.fm's formal definition of "super similarity", "very high similarity", or "high similarity".

[5]The reason we added a cut off of 100,000 listeners is that the number of listeners of musicians seem to follow a power law distribution. Without the cut off, our crawl did not finish after over a week of crawling, during which time we observed well over 1/2 million musicians, many with only a few hundred listeners.

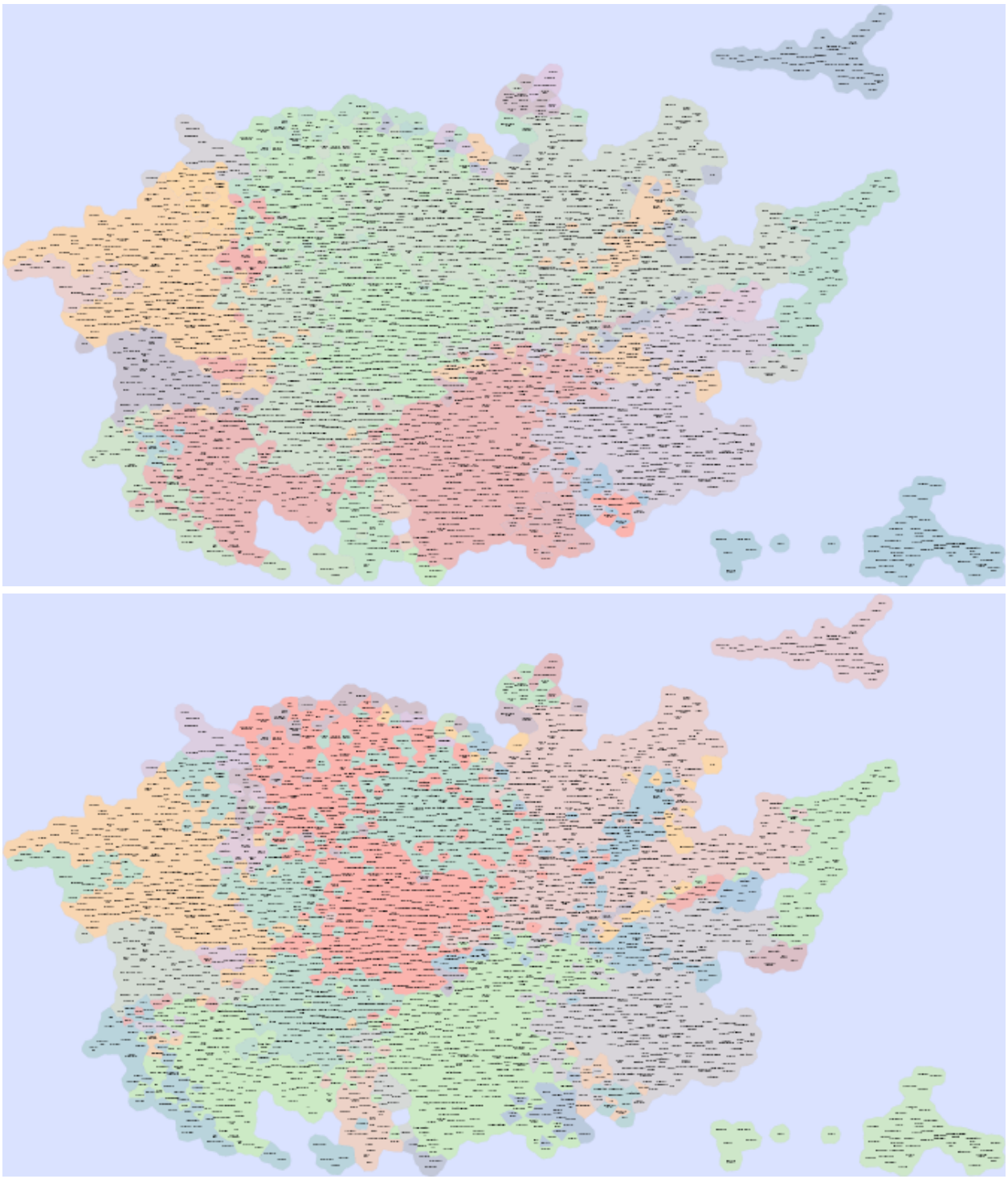

Figure 4: A random color assignment (top) vs. an optimized color assignment (bottom) that maximizes color differences between neighbors

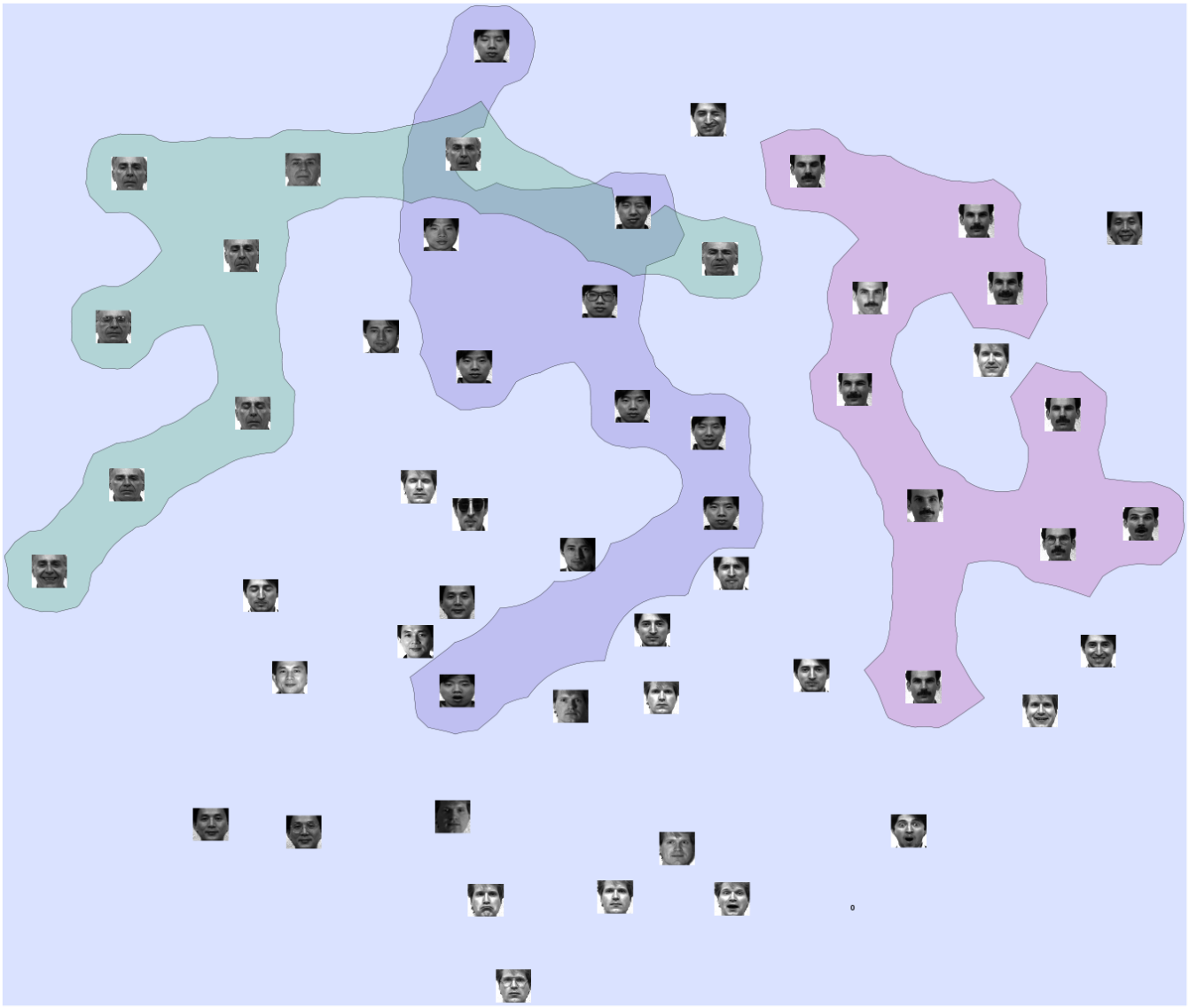

Figure 5: Using GMap to visualize multiple sets. Photos in the same set belongs to the same person.

before reaching the heavy rock cluster of *AC/DC* and *Iron Maiden* in the north. Farther to the north is the grunge cluster anchored by *Nirvana* and *Alice in Chains*.

On the west coast of the main continent, we find a cluster of electronic music, featuring *Fatboy Slim, Daft Punk*, and *DJ Shadow*. To the north are avant-garde electronic bands like *Aphex Twin*, while in the extreme west are electronic classics such as *Vangelis*. To the south is a compact and well-defined cluster of dance music represented by *Paul Oakenfold* and *ATB*.

In the center of the map, there is a well-defined concentration of female singer-songwriters such as *Alanis Morissette, Norah Jones*, and *Amy Winehouse*. Pop music is to the southeast of here with *ABBA* and *Eurythmics*, while hip-hop and rap music is southwest with *Beyonce* and *Alicia Keys*.

The Islands: There are two notable island regions in MusicLand: in the northeast is Reggae island, while the chain of islands off *Rocky Coast* in the southeast make up the *Classical Archipelago*. By zooming in (Figure 7), it is easy to find some general patterns in the layout of the archipelago along the East-West and North-South axes. Along the first axis, the west contains modern composers such as *Ravel, Satie*, and *Pärt*, while the east contains 17th century composers such as *Bach, Handel,* and *Albinoni*. Along the second axis, the north has a high concentration of opera composers such as *Verdi, Rossini*, and *Puccini,* whereas the south has more orchestral and instrumental composers such as *Holst, Elgar*, and *Stravinsky*. Not surprisingly, *Mozart* and *Beethoven* are the most popular composers in the classical music cluster. The islands of *Erik Satie* and *Arvo Pärt* connect the big island in the east with the westernmost island of contemporary classical music represented by minimalists *Philip Glass* and *Michael Nyman.*

BookLand

Many e-commerce websites provide recommendations to allow for exploration of related items. Traditionally this is done in the form of a flat list. For example, Amazon typically lists around 5-6 books under "Customers Who Bought This Item Also Bought", with a clickable arrow to allow a customer to see further related items.

Instead of a flat list, which provides a very limited view of the

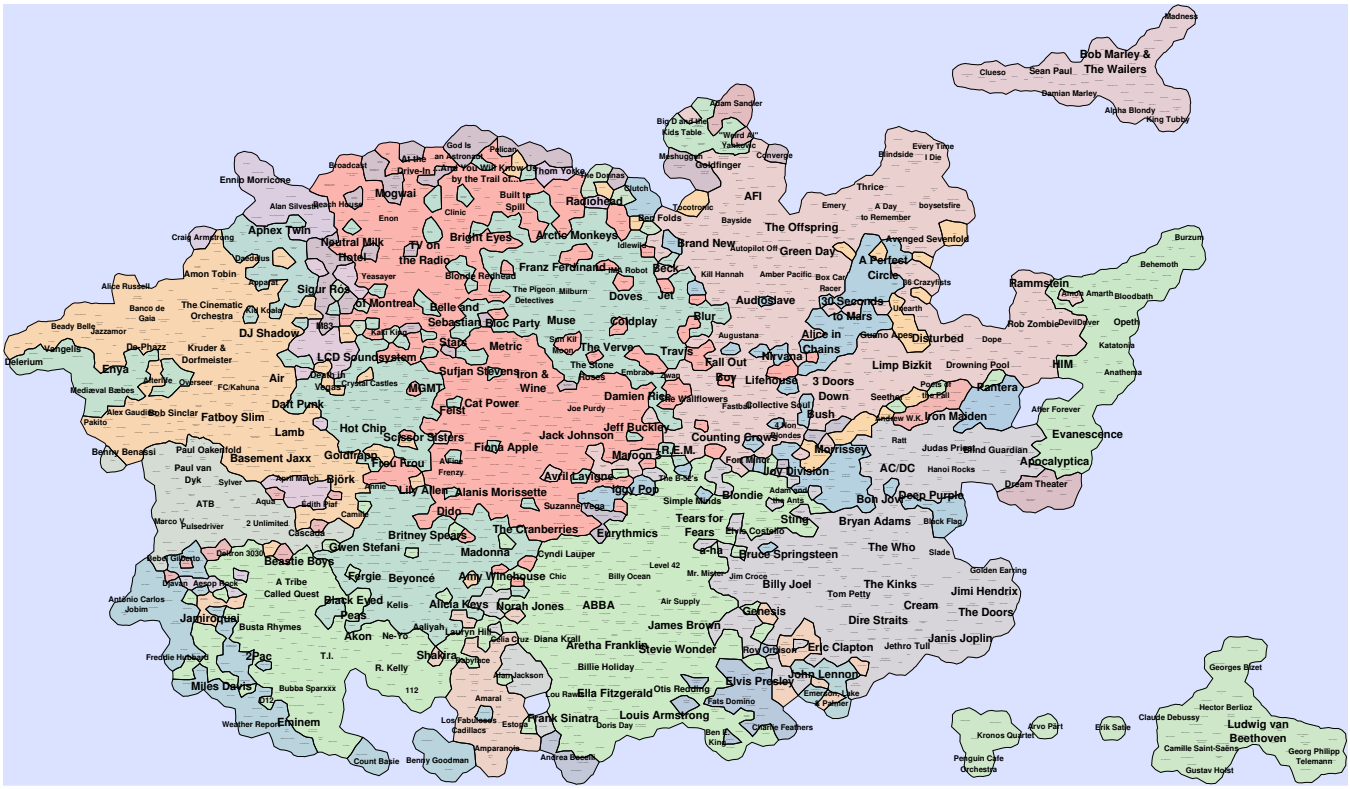

Figure 6: The landscape of music

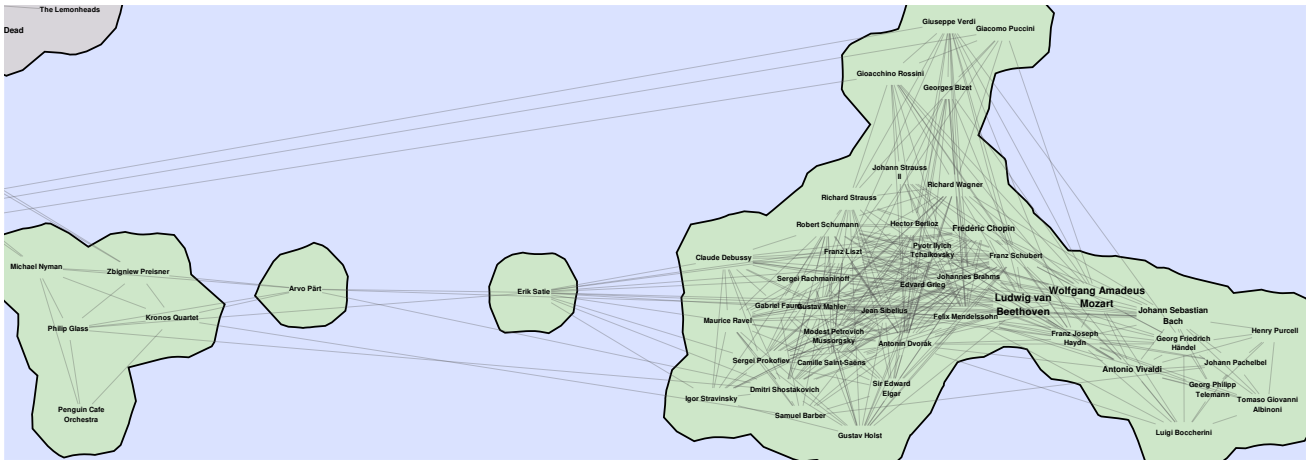

Figure 7: Part of the classical archipelago

Figure 8: Two of the central clusters in BookLand.

neighborhood, there have been attempts to convey the underlining connectivity of the products through graph visualization. None of the existing approaches, however, gives a comprehensive view of the relationship and the clustering structures.

Using our GMap algorithm, we obtained the map in Figure 1. The underlying data is obtained with a breadth-first traversal following Amazon's "Customers Who Bought This Item Also Bought" links, starting from the root node, Orwell's *1984*. Links are followed up to a distance of twelve from the root node. We then trim the graph by keeping only vertices of distance nine or less from the root vertex. We further merge nodes that represent the same book, but with different publishers or different bindings, by matching books with the same title. The underlying graph for this map contains 913 vertices and 3410 edges. With an average degree of nearly eight, peripheral vertices in this map have only a handful of edges while central vertices have more than 20 immediate neighbors. We next examine several of the BookLand countries in more detail (cf. Figures 8 and 1).

Americana: Somewhat surprisingly, Orwell's *1984* along with *Animal Farm* ended up in the west corner of a region populated mostly by American writers. Britain is also represented by Golding's *The Lord of the Flies* and Huxley's *Brave New World* along

with Burgess's *Clockwork Orange*, which connect the British corner of the region to the main part dominated by 20th century American classics. Bradbury's *Fahrenheit 451* and Salinger's *Catcher in the Rye* provide a transition to a variety of well-known novels: Steinbeck's *Grapes of Wrath* and *Of Mice and Men*, Hemingway's *For Whom the Bell Tolls*, Fitzgerald's *Great Gatsby*, Heller's *Catch-22*, and Kesey's *One Flew Over a Cuckoo's Nest*.

Russiana: To the north of Americana lies one of the largest countries in BookLand, dominated by Russian literature and history. The core contains classic novels by Dostoyevsky (*The Brothers Karamazov*), Tolstoy (*War and Peace*), and Solzhenitsyn (*The Gulag Archipelago, Cancer Ward*). In the west, there is, unexpectedly, a cluster of Camus books (*The Stranger, The Plague, The Fall*), all well connected with the Russian classics.

From the map, we can also see some high-level connections. There is the not too surprising proximity of self-help books with books recommended by Oprah Winfrey. We also find the group of recent vampire novels adjacent to the cluster of Victorian novels. A closer investigation shows that vampire novels are only connected to the rest of the graph through Jane Austen.

TradeLand

Figure 9 is a map visualizing the trade relations between all countries. Bilateral trade data between each of the 209 countries and its top trading partners were acquired from Mathematica's `CountryData` package. The font size of a label is proportional to the logarithm of the total trade volume of the country, and the color of a label reflects whether a country has a trade surplus (black) or deficit (red).

The label color gives an easy way to spot the oil-rich countries with large surpluses, which are distributed all over the world as well as in our map: Middle East (Saudi Arabia, Kuwait), Europe (Russia), South America (Venezuela), Africa (Nigeria, Equatorial Guinea). On the other hand, the countries with huge deficits are mostly in Africa (Sierra Leone, Senegal, Ethiopia) with the United States, the clear outlier.

Many countries in close geographic proximity end up close in our map, e.g, Central American countries like Honduras, El Salvador, Nicaragua, Guatemala and Costa Rica are close to each other in the northeast. Similarly the three Baltic republics, Latvia, Lithuania and Estonia, are close to each other in the northwest. This is easily explained by noting that geographically close countries tend to trade with each other. There are easy-to-spot exceptions: North Korea is not near South Korea, Israel is not particularly close to Jordan or Syria.

The G8 countries (Canada, France, Germany, Italy, Japan, Russia, United Kingdom, and the United States) are all in close proximity to each other in the center of the map. Two of the largest and closest countries in our map are China and the United States. Clearly, the proximity is due to the very large trade volume rather than geographic closeness. All these countries are in the largest cluster which is dominated by European countries in the west, Asian countries in the east, and Middle Eastern countries in the south.

Interestingly, we see from the map that African countries are distributed in several clusters in close proximity to China (a major trading partner to many African countries), the United States (trading less with Africa these days), and around former colonizers (e.g., Togo, Cameroon and Senegal, which are all close to France). On the other hand, Caribbean and South and Central American countries form several clusters in the north of the map. In addition, these clusters are mostly contiguous, essentially forming a supercluster. This differentiation between Latin America and Africa is clearly brought out by the GMap figure.

Finally, we note that the periphery of the map contains small countries from around the world, and countries with few trading partners.

TVLand

As a last example, we consider a map derived from TV viewer data. The objects are TV shows, with an edge between two shows if several viewers watched both. The edges are weighted by the number of such viewers. We illustrate how GMap can be used in the context of recommendation systems with the aid of heat map overlays, tailored to an individual viewer.

Figure 10 shows such a map for a typical but fictitious viewer. Based on the observed viewing habits of a specific viewer, and on those of other viewers, a recommendation system can suggest possible new shows. In order for the labels to be readable in a journal format, our map displays a small subset of the TV shows and consists of six countries. The countries capture several types of shows by genre. In the southwest are two countries corresponding to TV shows for children, with the smaller targeting younger kids with *Dora the Explorer* and *Wow Wow Wubbzy*. In the northwest is a cluster of shows about fashion and entertainment such as *Say Yes to Dress* and *E! News*. The large country in the north contains popular sitcoms such as *Seinfeld, Cheers*, and *Frasier*. In the northeast is a cluster of crime shows (*NCIS and Law & Order*). The cluster in the southeast is not as thematically focused but contains popular shows of several types: *The Oprah Winfrey Show, So You Think You Can Dance*, and *Dateline NBC*. In each country the TV shows are colored based on the strength of the recommendation; that is, the lighter the color, the stronger the recommendation, as in a typical geographic map where shading of colors are used to represent hills and valleys.

Maps like this provide a global view not available in the traditional list of recommendations and offer a more appealing presentation than a scatterplot or a node-link diagram. The map provides a context for recommendations, allowing a user to understand the reasoning behind the recommendation: when considering highly-recommended shows the user can check out nearby, related shows; or explore a path of shows to a new area.

Related Work

There are many papers in geography about accurately and appealingly representing a given geographic region, or on re-drawing an existing map subject to additional constraints. Examples of the first kind of problem are found in traditional cartography, e.g., the 1569 Mercator projection of the sphere onto 2D Euclidean space. Cartograms provide an example of the second kind of problem, where the goal is to redraw a map so that the geographical areas are proportional to some metric, an idea which dates back to 1934 and is still popular today (e.g., the New York Times' red-blue maps of the US, showing the presidential election results in 2000 and 2004 with states drawn proportional to population).

Work in information visualization has produced many ways of representing data, some even adopting the name "map." Most of these have little visual connection with geographic maps. The map of science [1] uses vertex coloring in a graph drawing to provide an overview of the scientific landscape, based on citations of journal articles. Treemaps [11], squarified treemaps [2], and the more recent newsmaps represent hierarchical information by means of space-filling tilings, allocating area proportional to some metric.

Concept maps are diagrams showing relationships among concepts [10]. Somewhat similar are cognitive maps and mind-maps used to represent words or ideas linked to and arranged around a central key word.

In self-organizing maps (SOM) [8], an unsupervised learning algorithm places objects on a two-dimensional grid such that similar objects are close to each other. Unlike GMap, SOM creates a "map" by coloring cells of the grid based on a feature value; therefore it

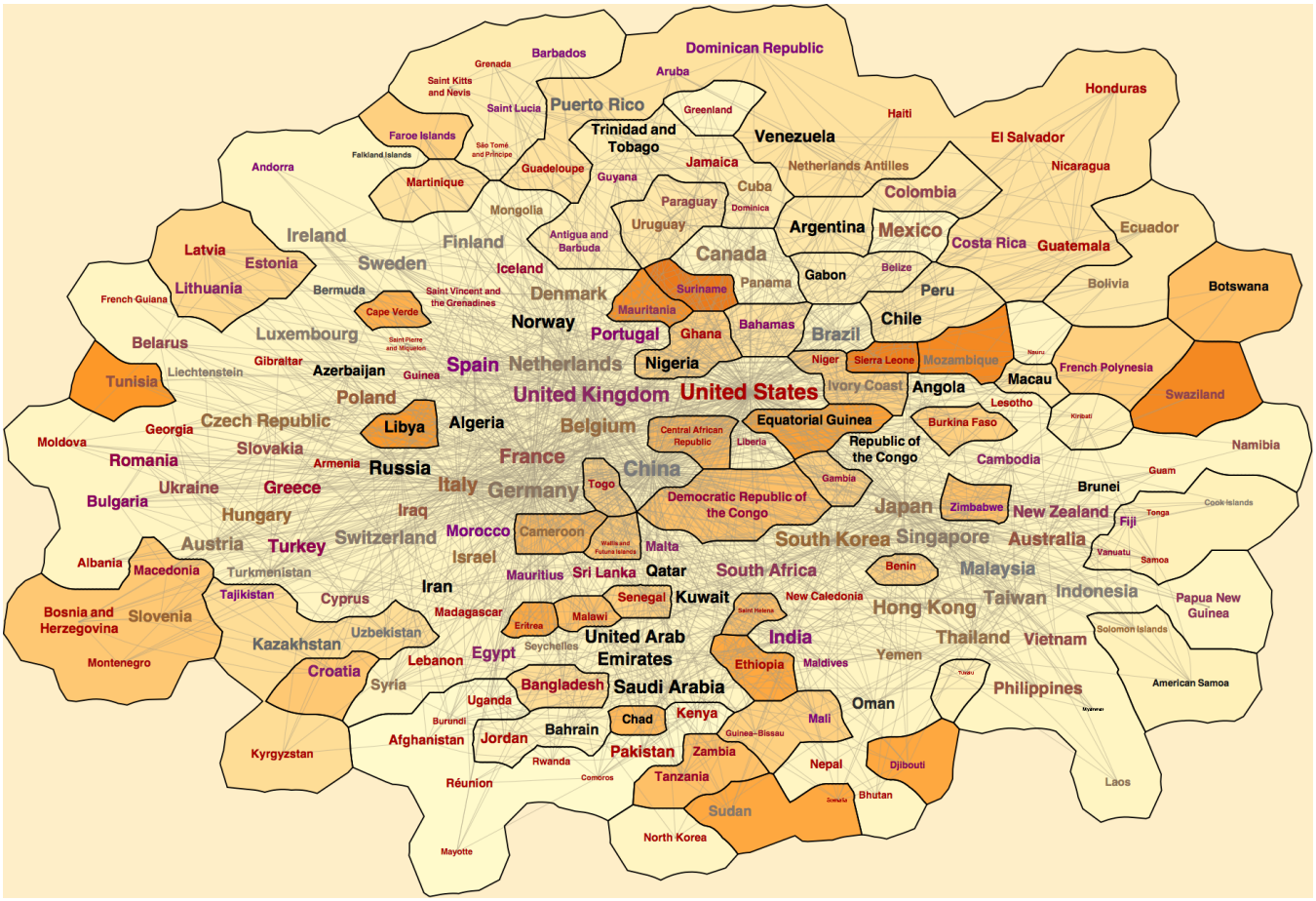

Figure 9: A map of trade relations between countries.

operates on a discrete grid space without a clear inner boundary between "countries". Furthermore, the grid tends to fit a rectangular box, so that the overall outline of the point set often follows that shape.

Also related is work on visualizing subsets of a set of items using geometric regions to indicate the grouping. Byelas and Telea [3] use deformed convex hulls to highlight areas of interest in UML diagrams. Collins et al. [4] use "bubblesets," based on isocontours, to depict multiple relations among a set of objects. Simonetto et al. [12] automatically generate Euler diagrams which provide one of the standard ways, along with Venn diagrams, for visualizing subset relationships. Apart from differences in the algorithms used to generate regions, these works differ from ours in that they create regions that overlap with each other, with the goal of faithfully representing set relationships, while we take the map metaphor seriously, assuming that regions do not overlap and aiming for a cartographic verisimilitude. However, as Figure 5 shows, our approach can also be used for visualizing sets.

Representing imagined places on a map as if they were real countries also has a long history, e.g., the 1930's Map of Middle Earth by Tolkien and the Bücherlandes map by Woelfle from the same period. More recent drawings include maps of programming language concepts and online communities. While most such maps are generated in an *ad hoc* manner by hand and are not strictly based on underlying data, they are often visually appealing.

Generating synthetic geography has a large literature, connected to its use in computer games and movies. Most of the work relies on variations of a fractal model. These techniques could provide additional photo-realism, and may be used in future extension of our work.

Conclusion and Future Work

We believe that the GMap algorithmic framework, by capitalizing on an ancient and familiar visual metaphor, introduces a significant new style of viewing abstract relational and cluster data that users will find aesthetically appealing and helpful in understanding the data. From informal observation, we have found that people, when faced with a traditional graph drawing and a map of the same data, spend significantly longer time studying the map, and they find non-trivial structural information without any prompting. For example, several viewers of BookLand observed that the "gateway" to Fringistan is Rand's *Atlas Shrugged*. We plan to perform formal user studies of the interaction with graphs and maps, in the context of visualizing recommendations [6].

While the approach of visualizing relational information with the aid of geographical maps is general, here we showed one particular implementation, where embedding and clustering algorithms are coupled with novel mapping and coloring algorithms. There are specific idiosyncrasies we would like to address. For example, GMap can produce countries that are disconnected. We have proposed ways [5] to use the cluster information to adjust the layout so that the regions of countries are more contiguous, at the expense of some loss of graph information. We are planning further investigation into balancing country connectedness and the preservation of

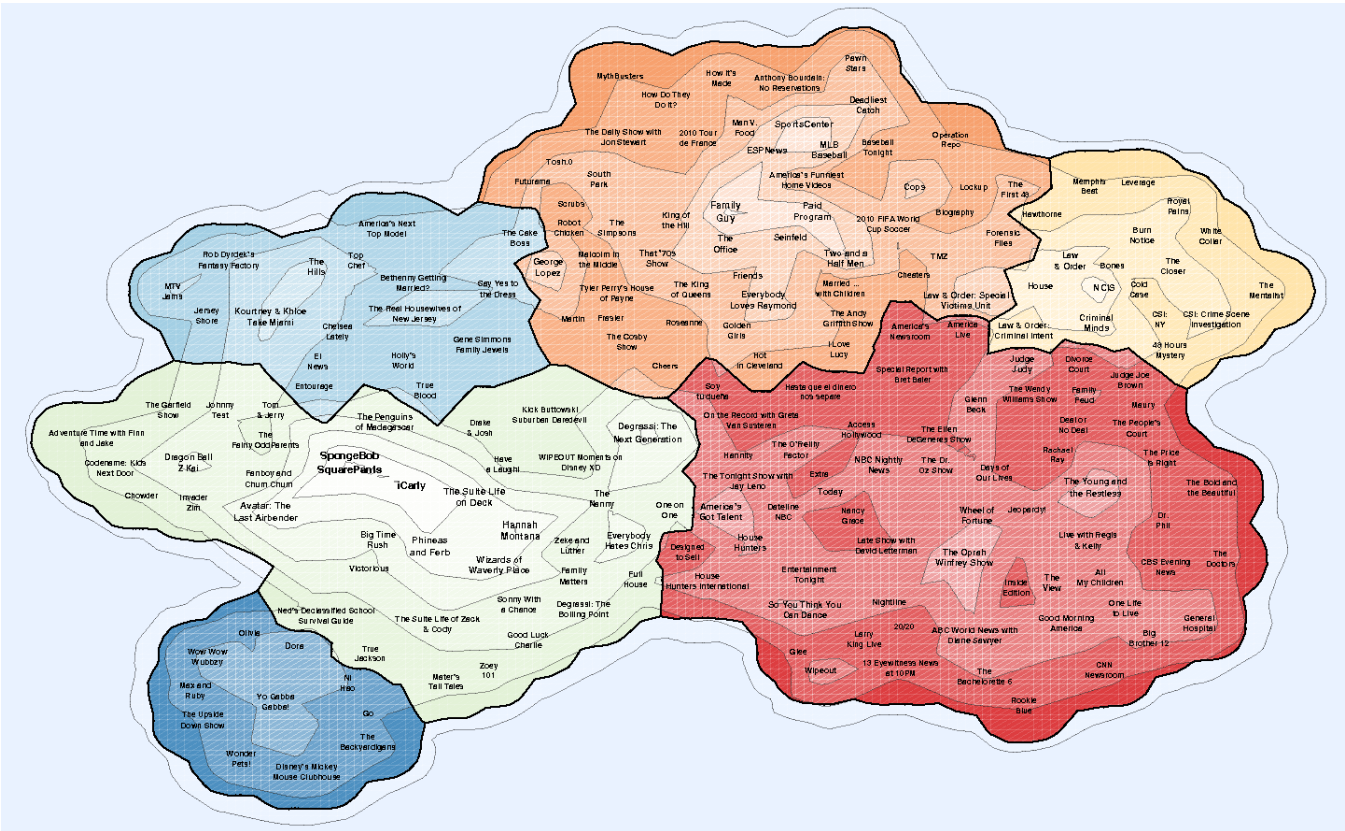

Figure 10: Personalized recommendation heat map for a typical user. Regions where the highest recommended shows sit are lighter. Regions of low scores are darker.

graph information.

Finally, our algorithm is efficient and can handle large graphs. As a reference point, all maps in this paper were generated in a few seconds. Mapping a larger graph with 440,000 vertices took 4 minutes on a typical processor. The resulting maps look best on large, wall-sized posters and display walls. To make such maps more useful for exploration of large data sets on commonly available media, we have developed an interactive interface that can search, zoom and pan easily on the maps (see footnote 1).

Acknowledgments

We would like to thank Stephen North for helpful discussions, and Carlos Scheidegger for help with the image process example.

REFERENCES

[1] K. Boyack, R. Klavans, and K. Borner. Mapping the backbone of science. *Scientometrics*, 64(3):351–374, 2005.

[2] M. Bruls, K. Huizing, and J. van Wijk. Squarified treemaps. In *Joint Eurographics and IEEE TCVG Symposium on Visualization*, pages 33–42. Press, 1999.

[3] H. Byelas and A. Telea. Visualization of areas of interest in software architecture diagrams. In *SoftVis'06: Proceedings of the 2006 ACM Symposium on Software Visualization*, pages 105–114, 2006.

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

[5] E. R. Gansner, Y. F. Hu, and S. G. Kobourov. Gmap: Drawing graphs and clusters as map. In *IEEE Pacific Visualization Symposium*, pages 201 – 208, 2010.

[6] E. R. Gansner, Y. F. Hu, S. G. Kobourov, and C. Volinsky. Putting recommendations on the map - visualizing clusters and relations. In *Proceedings of the 3rd ACM Conference on Recommender Systems*. ACM, 2009.

[7] A. Georghiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 23(6):643–660, 2001.

[8] T. Kohonen. *Self-Organizing Maps*. Springer, 2000.

[9] M. E. J. Newman. Modularity and community structure in networks. *Proc. Natl. Acad. Sci. USA*, 103:8577–8582, 2006.

[10] J. D. Novak and A. J. Cañas. The theory underlying concept maps and how to construct them. Technical report, Institute for Human and Machine Cognition, January 2006.

[11] B. Shneiderman. Tree visualization with tree-maps: A 2-D space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99, 1992.

[12] P. Simonetto, D. Auber, and D. Archambault. Fully automatic visualisation of overlapping sets. *Computer Graphics Forum*, 28:967–974, 2009.