Visualizing Article Similarities in Wikipedia

Patrick Riehmann, Martin Potthast, Henning Gruendl, Johannes Kiesel, Dean Jürges, Giuliano Castiglia, Bagrat Ter-Akopyan and Bernd Froehlich

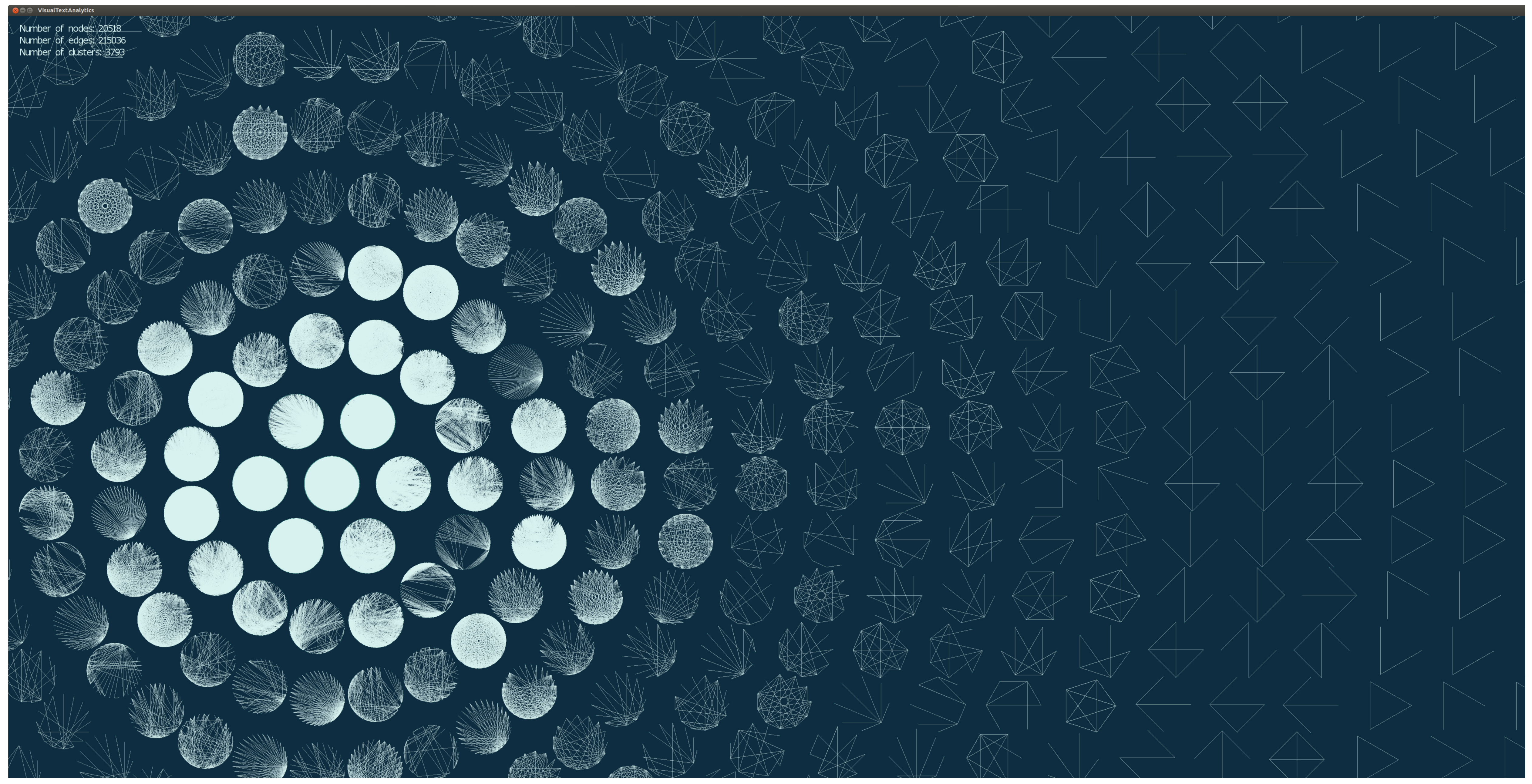

Figure 1: The radial layout places the subgraph containing the most articles in the center and orders the subgraphs with decreasing numbers of vertices more distally by growing the diameter level-wise.

Abstract

In this poster we present intermediate results regarding visual text analytics on Wikipedia. We implemented a visualization providing insight about similarities among Wikipedia articles in terms of structure as well as content. The presented data was gathered and processed via a pairwise comparison of all Wikipedia articles. Comparisons were appropriately pruned due to time and memory reasons when providing our in-memory database with the computed similarity values for visualization.

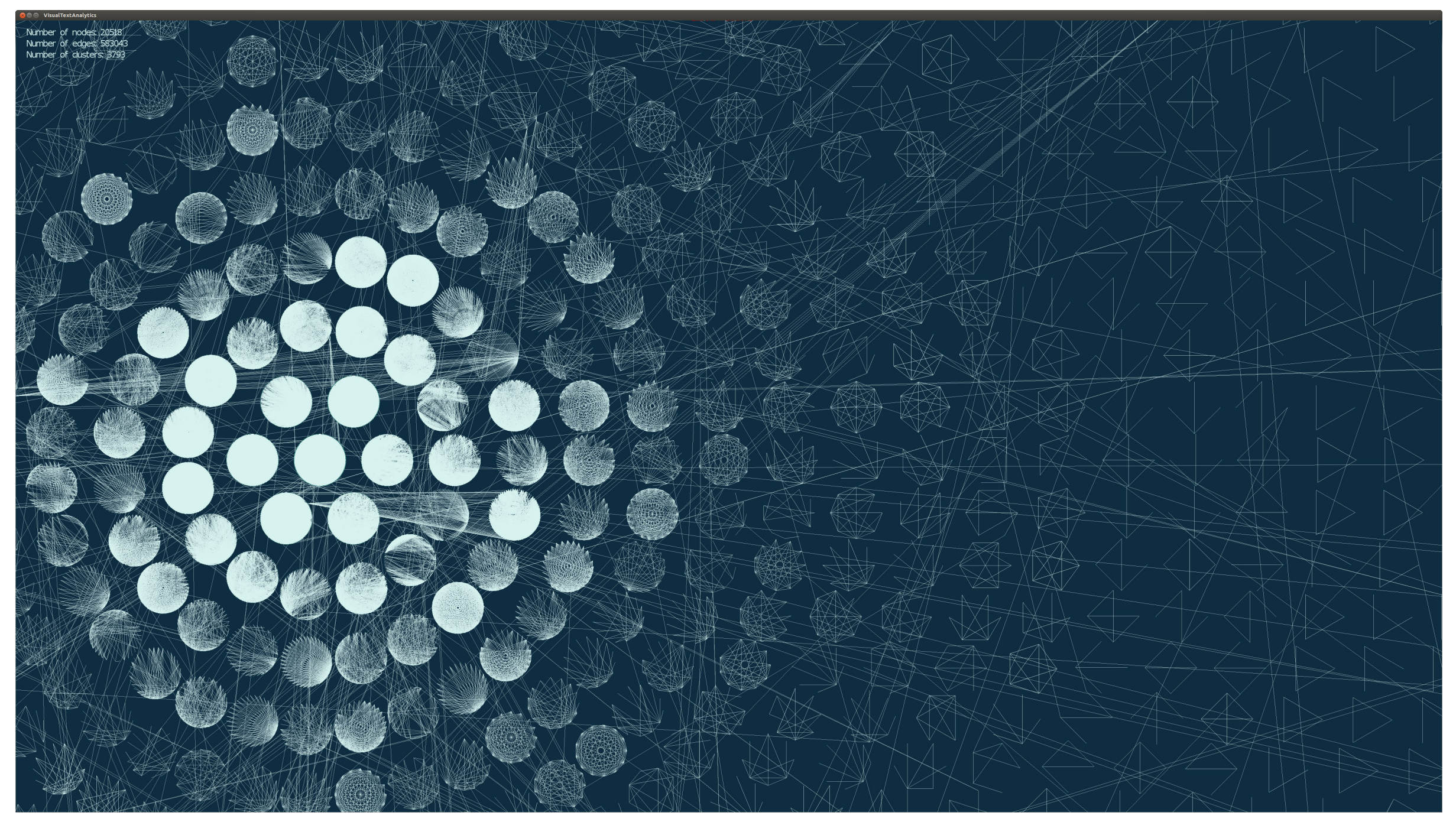

Figure 2: Additional edges appear within as well as across the subgraphs by lowering the global similarity threshold.

Data

From a recent dump of the English Wikipedia, all articles were extracted (excluding meta pages such as user pages, help pages, and talk pages, etc.) and transpiled to plain text using Sweble. Under the vector space model with a *tf-idf* weighting scheme, stemming, and stop word removal, each text document d was represented as a high-dimensional vector **d**. Given two vectors **d1** and **d2**, the semantic similarity of their original documents *d1* and *d2* can be estimated using the cosine similarity measure where a score of 0 means no similarity, and 1 maximum similarity. All articles' vectors combined form a term-document matrix which was distributed column-wise across four graphics cards (NVidia GTX 480) using CUDA. This way, computing the cosine similarity of a given article vector to all of Wikipedia took about 0.5 milliseconds. We compared every Wikipedia article in turn in decreasing order of length (where length served as an indicator of importance) to all others, recording the computed cosine similarities. To save space, we discarded all similarities of 0.1 or less as well as all similarities beyond the top 100,000 per article. To save time, we stopped computations once articles of length 50 terms or less were reached, since these articles are mostly superficial. Note in this respect, that the short articles still formed part of the aforementioned term-document matrix, so that all longer articles have been com- pared to the short ones, whereas the short ones have not been com- pared among themselves. Table 1 gives an overview of the similarity distribution; a total of 65 billion similarities have been computed in about 22 days, stored in 402 GB of disk space for a total 3.8 million articles disregarding the short ones.

Visualization and Interaction

Our visualization depicts articles as vertices and similarity relations as edges. Since an edge is only constituted if the similarity of the two adjacent articles' vertices exceeds a given threshold, the resulting graph consists of a large number of isolated subgraphs (aka graph components). At first, the vertices of each subgraph are arranged locally circle-wise (Figure 1) or as ring metaphor by placing the articles that are in sum incident to more than 50% of the edges on an additional inner circle (Figure 4 right). Globally, the subgraphs are arranged by number of articles contained; either in rectangular (Figure 5) or radial layout (Figure 1). Besides common interactions like zooming and panning most interactions are subgraph-centered such as dragging, dropping and highlighting subgraphs (Figure 3). Semantically zooming on particular subgraphs provides article titles and similarity values, whereas subgraph merging (Figure 4 left) gathers all subgraphs that become connected to the current one when lowering the similarity threshold (Figure 2). Due to the increasing number of edges going along with lowering the similarity threshold, recreating the entire layout instantly for billions of edges was infeasible. Thus, a base drawing consisting of all edges exceeding the initial threshold is generated. Changing the threshold entails the appearance and disappearance of edges (Figure 2vvv), however, some nodes may not emerge until the base layout is recreated.

Implementation was done completely in C++ and OpenGL by conveniently using gloost [WB], imgui [Corb] and FastDB [Kni] (an in-memory database for C++). We evaluated access times of custom-made, tailored data structures against FastDB, the former being faster by a factor of 2. We still opted in favor of FastDB for its additional object-relational features, especially, since practical render performance was barely effected.

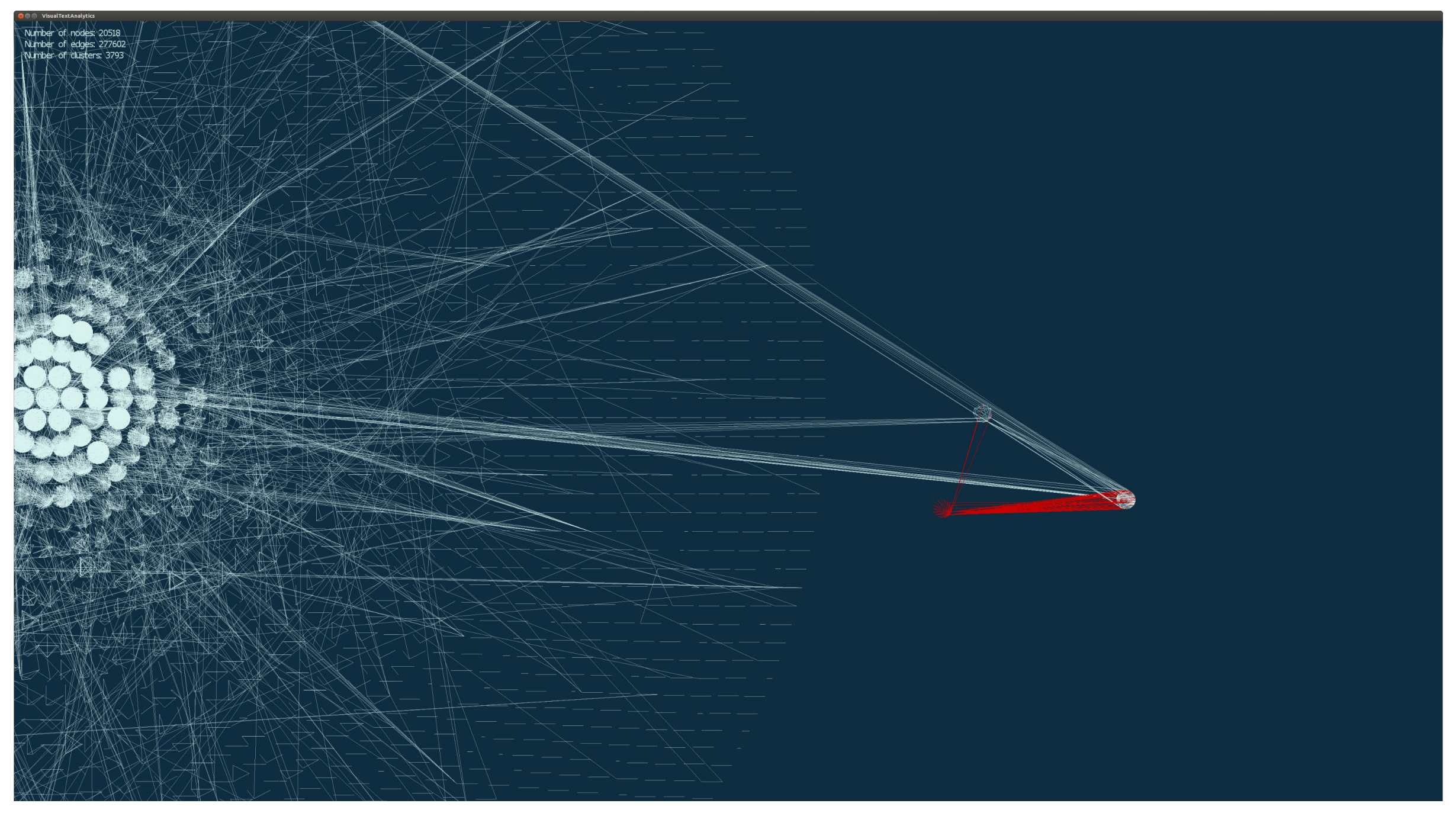

Figure 3: Interaction capabilities like zooming, panning and dragging allow paying attention to particular subgraphs.

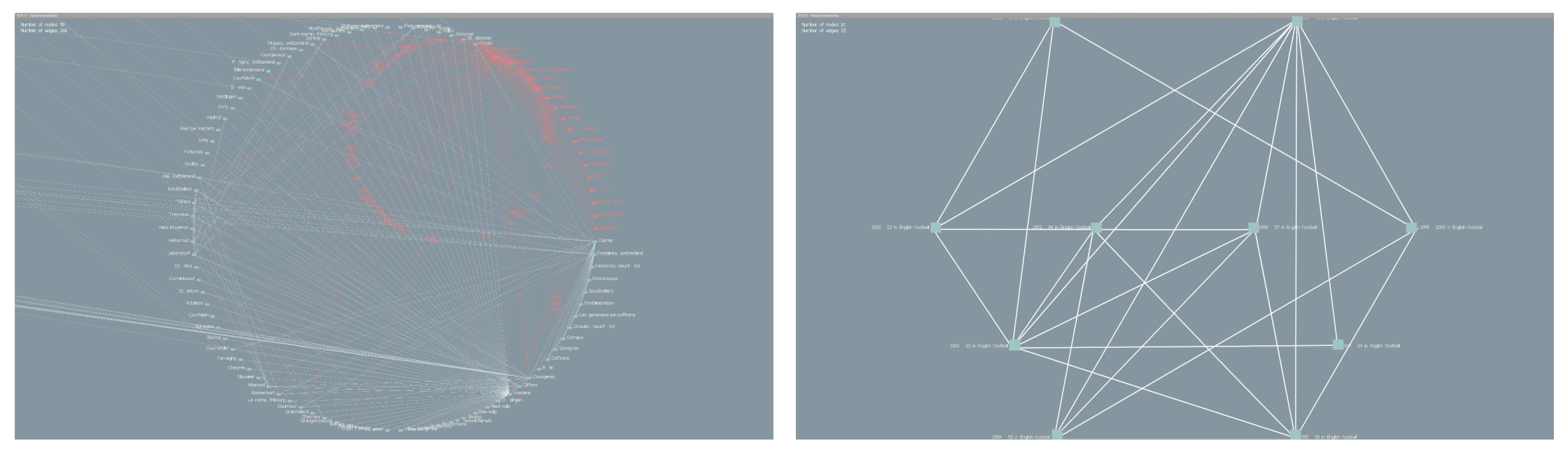

Figure 4: Subgraph merging (left). Local ring layout (right).

Similarity	Number of comparisons	Percentage	Size
[1.0, 0.9)	215,221	0.00033	1.35 MB
[0.9, 0.8)	1,125,858	0.00173	7.09 MB
[0.8, 0.7)	4,762,437	0.00732	29.98 MB
[0.7, 0.6)	16,598,675	0.0255	104.57 MB
[0.6, 0.5)	71,085,794	0.1093	1.17 GB
[0.5, 0.4)	329,595,171	0.5069	2.02 GB
[0.4, 0.3)	1,442,942,719	2.2	8.80 GB
[0.3, 0.2)	6,918,854,819	10.6	42.40 GB
[0.2, 0.1)	56,558,065,056	86.9	347.60 GB
> 0.1	65,344,422,345	100	402.14 GB

Table 1: Distribution of Wikipedia article similarities under the vector space model

Discussion and Future Work

Providing threshold-adaptive layout; less rigid and static than before is crucial. A dynamic layout should at least manage slight changes of the threshold at run time by rearranging the node positions appropriately in local regions without distracting the larger picture and without loosing the user's mental map. For larger changes, however, we intend to create a preprocessed acceleration structure that provides initial positions or proximity information about clusters and articles for the entire threshold range.

Providing a proper subgraph labeling derived from the content of the articles assigned to a particular one will ease inspecting striking subgraphs. In our explorations we found that our similarity measures tends to focus more on structural similarity such as subgraphs of cities, countries, years, events, etc.; maybe due to the fact that Wikipedia articles discussing such entities tend to be structured homogeneously. In order to ameliorate the recognition of similarities in the actual running text, the data cleaning has to be improved as well as the similarity measures' weighting scheme. A comparison between different similarity measures that may be superimposed on top of each other is another interesting possibility.

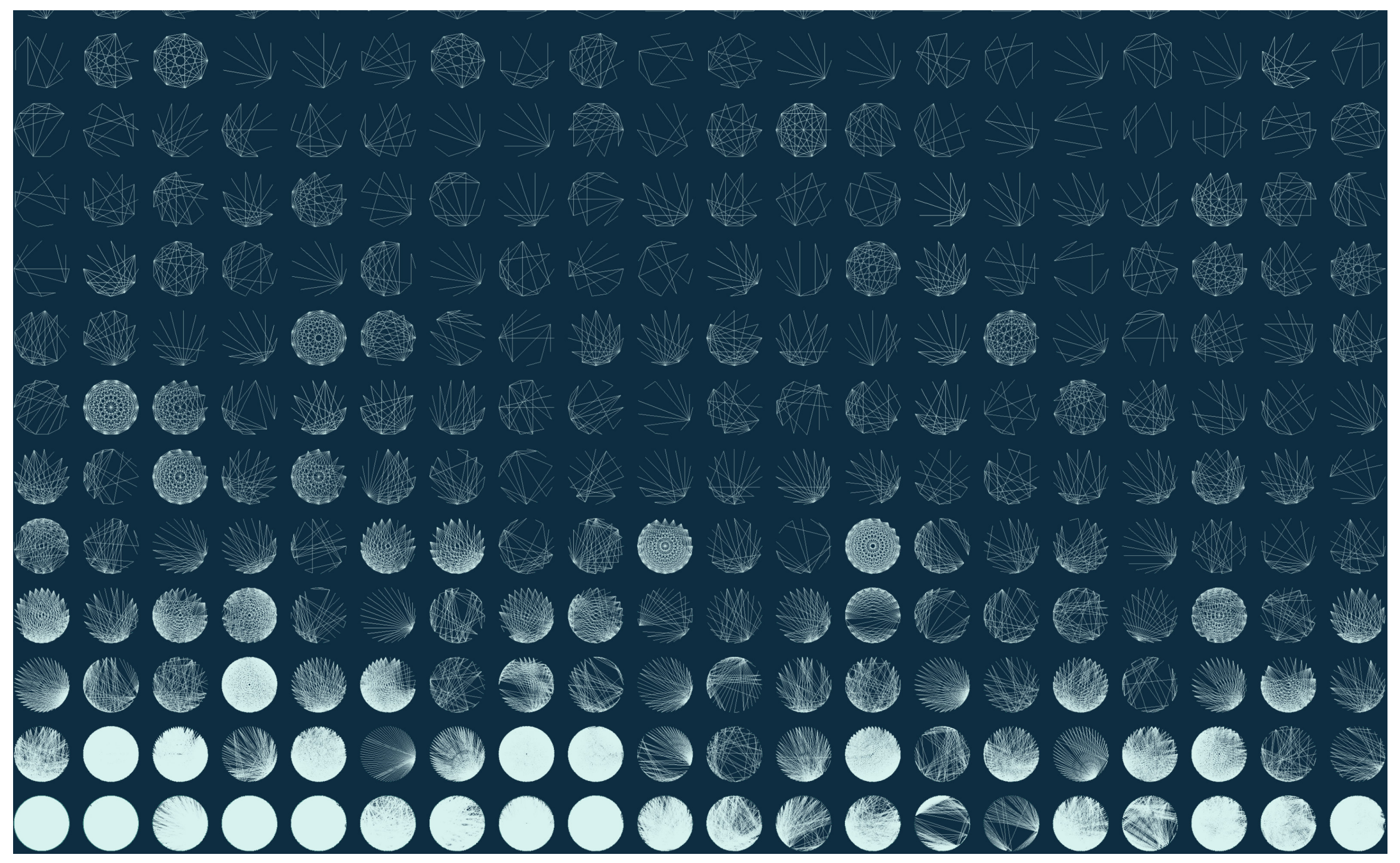

Figure 5: A rectangular placement arranges the subgraphs (in local circle layout) row-wise from left to right starting with the one that contains the the most articles in the lower left corner

VR & Visualization Research
Bauhaus-Universität Weimar