Drawing the AS Graph in Two and a Half Dimensions*

Michael Baur† Ulrik Brandes‡ Marco Gaertler† Dorothea Wagner†

20th July 2004

Abstract

We propose a method for drawing AS graph data using 2.5D graph visualization. In order to bring out the pure graph structure of the AS graph we consider its core hierarchy. The k-cores are represented by 2D layouts whose interdependence for increasing k is displayed by the third dimension. For the core with maximum value a spectral layout is chosen thus emphasizing on the most important part of the AS graph. The lower cores are added iteratively by force-based methods. In contrast to alternative approaches to visualize AS graph data, our method illustrates the entire AS graph structure. Moreover, it is generic with regard to the hierarchy displayed by the third dimension.

1 Introduction

Current research activities in computer science and physics are aiming at understanding the dynamic evolution of large and complex networks like the physical internet, World Wide Web, peer-to-peer systems and the relation between autonomous systems (AS). The design of adequate visualization methods for such networks is an important step towards this aim. As these graphs are on one hand large or even huge, on the other hand changing, i.e. growing within short time, customized visualizations concentrating on their intrinsic structural characteristics are required.

In this paper we concentrate on graphs modeling the relation between autonomous systems (AS graphs). We propose a layout method that brings out the pure structure of an AS graph. More precisely, we focus on the core hierarchy of AS graphs. A 2D layout is obtained by first choosing a spectral layout to display the core with maximum value and then adding the lower cores iteratively by force-based methods. Using 2.5D graph visualization, we then represent the core hierarchy by stacking the induced 2D layouts of the k-cores for increasing k on top of each other in the third dimension.

Two and a half dimensional visualizations have been proposed frequently for network data from various applications. Related methods use the third dimension to display a graph hierarchy [6, 9] or graphs that evolve over time [4]. A few samples of visualizations of AS graphs are already available. However, they either focus on the geographic location of the AS [7], on the tree of routing paths seen from a selected AS [2, 8] or on a high level view created by clustering the nodes [14]. In contrast, our method displays the entire AS graph structure without using external information.

Previous attempts to analyze the structure of the AS graph propose the existence of meaningful central nodes that are highly connected to a large fraction of the graph [11]. It seems that this structural peculiarity is interpreted very well by the notion of k-cores introduced in [15, 1]. This concept is already rudimentary used for initial cleaning in [12]. Accordingly, our approach is based

*The authors gratefully acknowledge financial support from DFG under grant WA 654/13-2 and BR 2158/1-2, and from the European Commission within FET Open Projects COSIN (IST-2001-33555) and DELIS (contract no. 001907).

†University of Karlsruhe, Department of Computer Science, 76128 Karlsruhe, Germany. {baur,gaertler,dwagner}@ilkd.uni-karlsruhe.de

‡University of Konstanz, Department of Computer & Information Science, 78457 Konstanz, Germany. Ulrik.Brandes@uni-konstanz.de

on the hierarchical core decomposition of the AS graph. However, it can be modified with respect to the hierarchy displayed by the third dimension.

We consider various AS graph data sets to demonstrate the usefulness of our method as means for analyzing and supporting the relation between autonomous systems. On one hand, AS graphs from different dates between 2001 and 2003 are studied, provided by the Oregon Routeview Project [13], on the other hand graphs obtained by the Internet Topology Generator INET 3.0 [16] are consulted.

Section 2 introduces the layout paradigm pursued by our approach and the new 2.5D visualization method for AS graphs is explained in Section 3. In Section 4 we present and discuss the results obtained for various AS graph data sets and Section 5 gives the conclusions.

2 Layout Paradigm

As mentioned in the introduction previous attempts to analyze the structure of the AS graph propose the existence of meaningful central nodes that are highly connected to a large fraction of the graph. Accordingly, we assume a hierarchical decomposition. However, abstraction to the levels of hierarchy is normally accompanied by a loss of information. This drawback should be avoided. Therefore, we establish the following layout paradigm:

- All nodes and edges are displayed.
- The levels of hierarchy are emphasized.
- The inter- and intra-level connections are made clear.

In order to extract a hierarchy that resembles the connectivity structure of the AS graph, we consider the k-core concept. The *k-core* of a graph is defined as the unique subgraph obtained by recursively removing all nodes of degree less than k. A node has *coreness* ℓ, if it belongs to the ℓ-core but not to the $(\ell + 1)$-core. The *ℓ-core layer* is the collection of all nodes having coreness ℓ. The *core* of a graph is the k-core such that the $(k + 1)$-core is empty.

3 Layout Method

We propose an incremental algorithm to produce a 2D layout satisfying our layout paradigm. This layout is afterwards transformed into 2.5D in a canonical way using the core hierarchy. The description of the 2D layout algorithm is divided into two parts. At first a generic method to generate a 2D layout of a hierarchical decomposition of the graph is introduced. We then give a specification of parameters that can be chosen to fulfill certain requirements and requests induced by the structure of AS graphs.

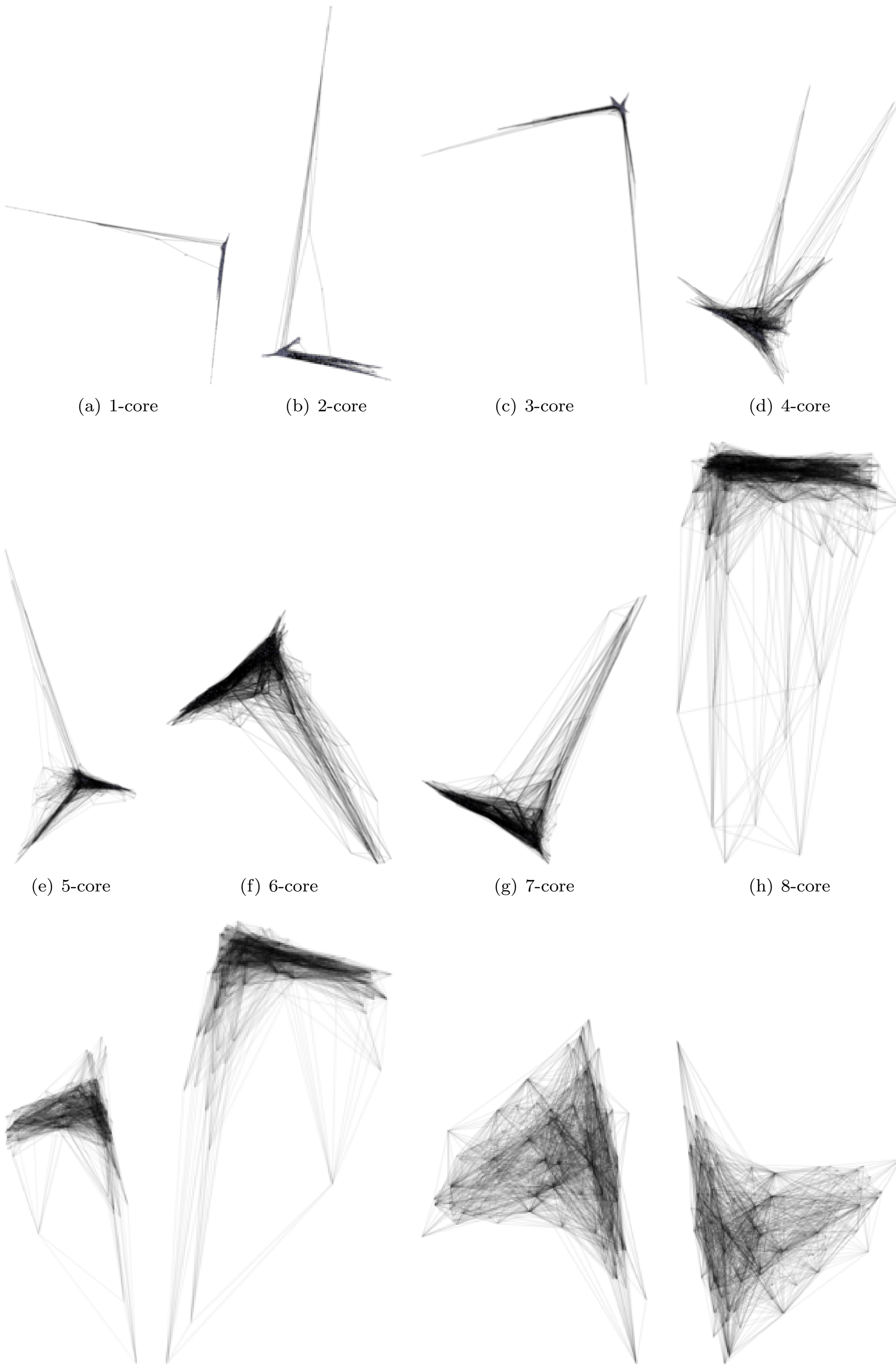
3.1 Generic Algorithm

The first step of the algorithm constructs a spectral layout for the highest level of the hierarchy. Then, iteratively, the lower levels are added using a combination of barycentric and force-directed placement. Algorithm 1 gives a formal description of this procedure based on the core hierarchy.

Preliminary studies indicate that a spectral placement does not lead to a satisfactory layout of the AS graph as a whole. However, the results improve for increasing core value (example shown in Figures 1,2).

We therefore choose a spectral layout as initial placement for the core of the graph. Then, for the iterative addition of nodes with lower coreness to the already computed layout, a combination of a barycentric and a force-directed placement is used.

More precisely, when a new level of hierarchy is added, we first calculate a barycentric placement in which all new nodes are placed in the barycenter of their neighbors in this level. Unfortunately, barycentric layouts also have a number of drawbacks. Firstly, nodes that are structural equivalent



(a) 1-core (b) 2-core (c) 3-core (d) 4-core

(e) 5-core (f) 6-core (g) 7-core (h) 8-core

(i) 9-core (j) 10-core (k) 11-core (l) 12-core

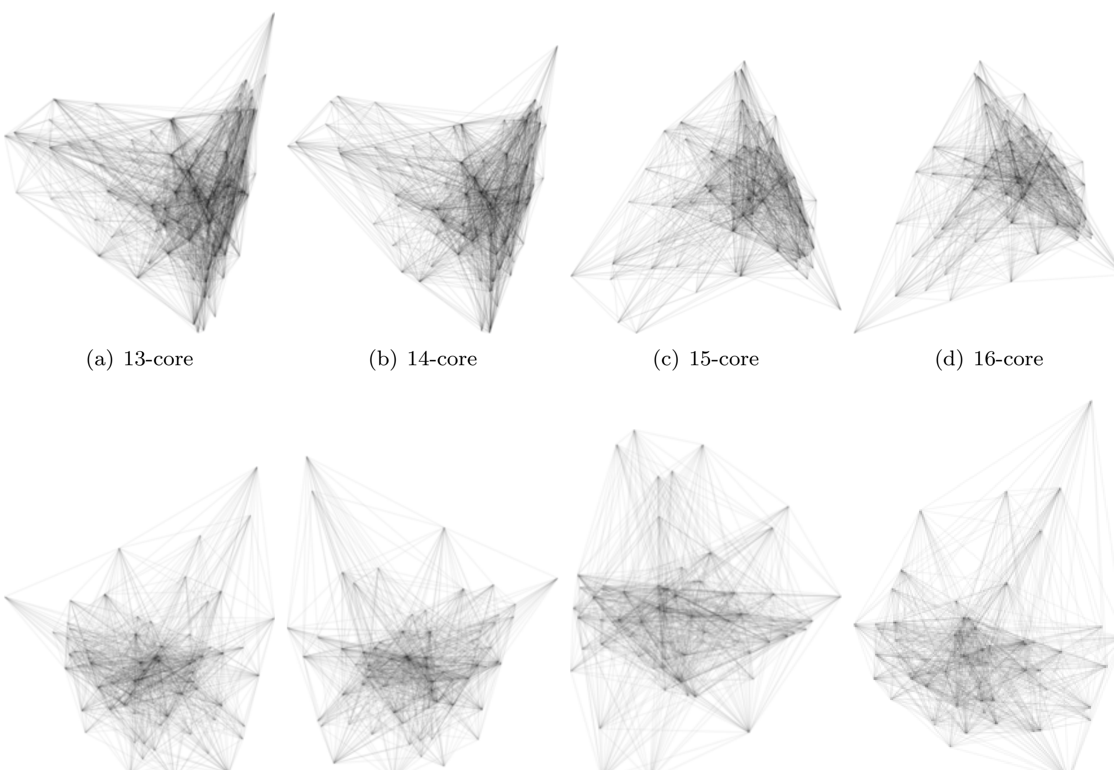
Figure 1: Spectral Layouts (I/II) of the cores of the AS graph (01/01/2003)

Algorithm 1: Generic AS layout algorithm

Input: graph $G = (V, E)$

calculate coreness $c \colon V \to \mathbb{N}$
let $k \leftarrow$ maximum coreness, $G_l \leftarrow$ the l-core, $C_l \leftarrow l$-core layer

calculate spectral layout for G_k
for $l \leftarrow k - 1, \ldots, 1$ **do**
    if $C_l \neq \emptyset$ **then**
        calculate barycentric layout for C_l in G_l, keeping G_{l+1} fixed
        calculate force-directed layout for C_l in G_l, keeping G_{l+1} fixed
        calculate force-directed layout for G_l



(a) 13-core (b) 14-core (c) 15-core (d) 16-core

(e) 17-core (f) 18-core (g) 19-core (h) 20-core

Figure 2: Spectral Layouts (II/II) of the cores of the AS graph (01/01/2003)

in the current subgraph are assigned to the same position. Secondly, all nodes are placed inside the convex hull of the already positioned nodes. In particular this means that the outermost placed nodes are those having highest coreness which is clearly contradictory to the intuition of importance. To overcome these difficulties, we use the barycentric layout as an initial placement for a subsequent force-directed refinement step, where only newly added nodes are displaced. In addition, a force-directed approach is applied for all nodes in order to relax the whole graph layout. However, the number of iterations and the maximal movement of the nodes is carefully restricted not to destroy the previously computed layout. A special feature of this relaxation step is the use of non-uniform natural spring lengths $l(u, v)$, where $l(u, v)$ scales with the smaller core value of the two incident nodes u and v. Thus, the effect of a barycentric layout is modeled, since edges between nodes of high coreness are longer than edges between nodes of low coreness. Accordingly, these springs prevent nodes with high coreness from drifting into the center of the layout.

The generic AS layout algorithm is based on the core decomposition of the graph. In general, the core decomposition can result in disconnected parts. For the AS graph, all k-cores stay connected which is an additional advantage of the core hierarchy in this case. Of course, the algorithm can be applied in the same way for an arbitrary hierarchical decomposition. However, the combination of layout steps as well as the choice of parameters presented in the next paragraph is originated from the core structure of the AS graph.

3.2 Fitting the Parameters

Beside the choice of the underlying hierarchical decomposition, the algorithm offers a few more degrees of freedom that allow an adjustment to a broad range of applications. For the spectral layout we propose a modified Laplacian matrix $L' = 1/4 \cdot D - A$ already considered in [5]. Our experiments showed that the application of the normalized adjacency matrix results in comparably good layouts while the standard Laplacian matrix performs significantly worse.

The force-directed placement is performed by a variant of the algorithm from [10]. Unlike the original algorithm, we calculate the displacement only for one vertex at a time and update its position immediately. Furthermore, we use the original forces but with non-uniform natural edge lengths $l(u, v)$ proportional to $\cdot \min\{\text{level}(u), \text{level}(v)\}^2$. For the refinement step where only new nodes are displaced using forces, we perform at most 50 iterations. The second application of forces to relax the whole layout is already stopped after 20 iterations.

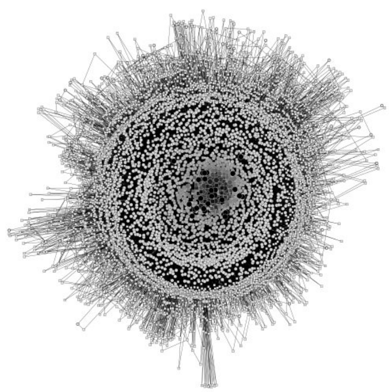
4 Results

We illustrate the results of our method for real AS data sets as well as for generated graphs. For a more detailed discussion, we also refere to [3]. The section is concluded by techniques to aid the human perception and the discussion of some artefacts of the spectral embedding.

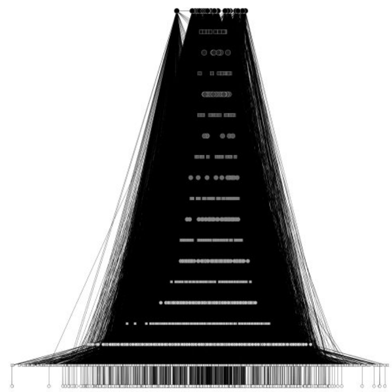
4.1 Data Sets

Our real world data consist of three AS graphs collected by the Oregon Routeview Project at different dates, i.e June, 1st 2001 (11,211 nodes, 23,689 edges, 19 levels), June, 1st 2002 (13,315 nodes, 27,703 edges, 20 levels), and June, 1st 2003 (15,415 nodes, 34,716 edges, 25 levels). Additionally, we used INET 3.0 to generate artifical graphs that should exhibit a similar topology.

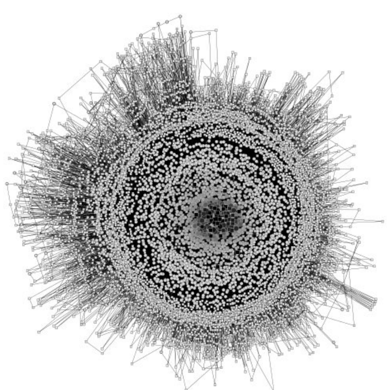
We discuss two different two-dimensional types of figures, the 2D layout produced by Algorithm 1 and the projection of the 2.5D layout into one of the full dimensions, also referred to as *level projection*. Nodes are represented by rectangles (odd coreness) or ellipses (even coreness) of size decreasing according to the coreness and with colors fading from black to white. Edges are always drawn as straight lines.



(a) 2D layout (2001)

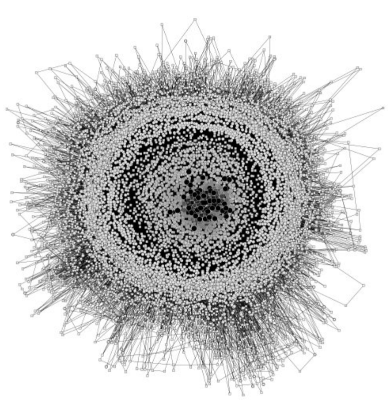


(b) level projection (2001)

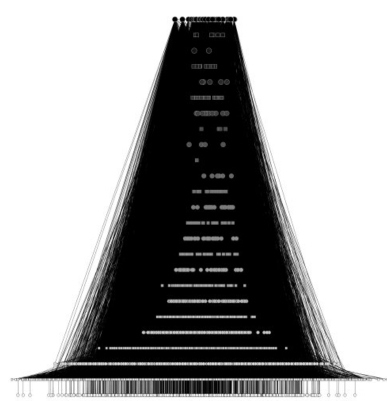


(c) 2D layout (2002)

(d) level projection (2002)



(e) 2D layout (2003)



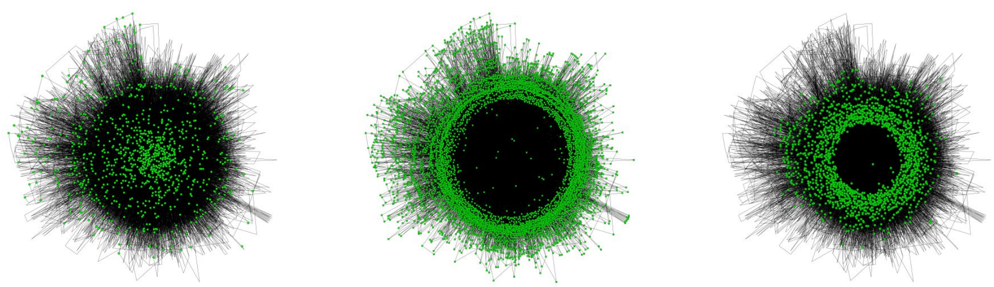
(f) level projection (2003)

Figure 3: 2D layout and level projection of the AS graph

4.1.1 Real AS Graph

The 2D layouts are dominated by the nodes with small coreness. This is reflected in Figures 3(a), 3(c) and 3(e) that show a huge periphery of small and bright nodes. On the other hand, most nodes with higher coreness are contained in the convex hull of the core, which is apparent in Figures 3(b), 3(d) and 3(f).

A closer examination reveals three almost separated radial areas around the center. The first one mainly contains the 3-core layer, while the 2-core layer forms the second and third area that are distinguished by their density. Figures 4(a)-4(c) show the corresponding nodes in the lower layers.



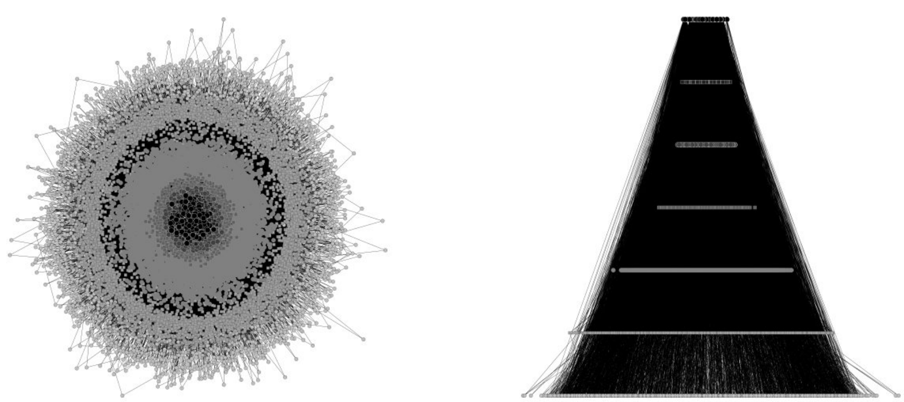
(a) restricted to 1-core layer (b) restricted to 2-core layer (c) restricted to 3-core layer

Figure 4: 2D layout where only nodes in certain cores are drawn (06/01/02)

Especially for the 2- and 3-core layer, we can identify two classes of nodes. In both cases there is a relatively uniform class of nodes drawn towards the center and a second class of nodes in the periphery. In contrast, a large part of the 1-core layer is attracted to the central region. Nodes of medium and large coreness (greater than five) are contained in the convex hull of the core which documents the relation between importance and coreness.

These properties can be observed for all three instances. The well-known growth of the AS graph affects especially the 2- and 3-core layers. We observe that the spatial distances of these two layers decreases over time.

4.1.2 Generated Graphs



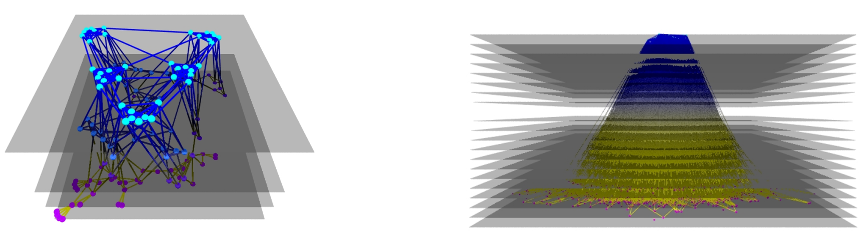
(a) 2D layout (b) level projection

Figure 5: Figures of the generated graph with 11,211 nodes

The 2D layout and the level projection of a generated graph are shown in Figure 5. Let us first mention that there is a significant difference of the generated graphs to the real AS graphs, e.g. the number of edges of a generated graph with 11,211 nodes is much larger than the number of edges of the real AS graph with the same number of nodes (almost 12,000 edges more), while the highest core value is only eight (in contrast to highest core value 19 of the AS graph). An obvious difference of the generated graph induced by this fact is the more uniform distribution of cardinalities of the core layers. Accordingly, in the layouts the separation of the different core layers is less significant. However, the 2D layouts are still satisfactory with respect to the illustration of the core hierarchy.

4.2 Supporting Perception

There are several means for visual aid in 2.5D layouts, i.e. choice of perspective (in 3D), additional geometric objects emphasizing the levels of hierarchy, and colors. The choice of perspective is very powerful. We have already used this feature when presenting only the 2D layout and the level projection respectively. More general, a user can focus on individual aspects, i.e. a global oriented view, a hierarchical version, or a mixture of both. A beneficial consequence might be that unintended information is automatically masked out by the perspective. In order to simplify navigation in the three dimensional space, one can also introduce additional objects that mark the levels of hierarchy, i.e. rectangles, discs, or planes. Transparency or filters might even increase their effectiveness. Color can be used in various ways, to highlight nodes and edges of special interest, to code the levels of hierarchy, or to improve the overall perception. In Figures 6–13



(a) small example

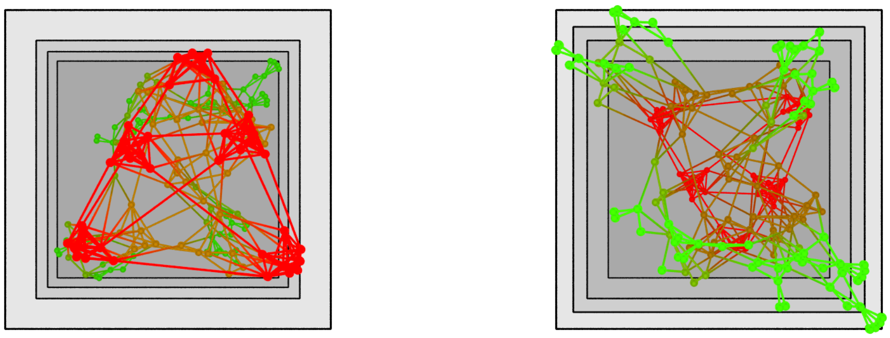
(b) AS graph (06/01/03)

Figure 6: Visual support features

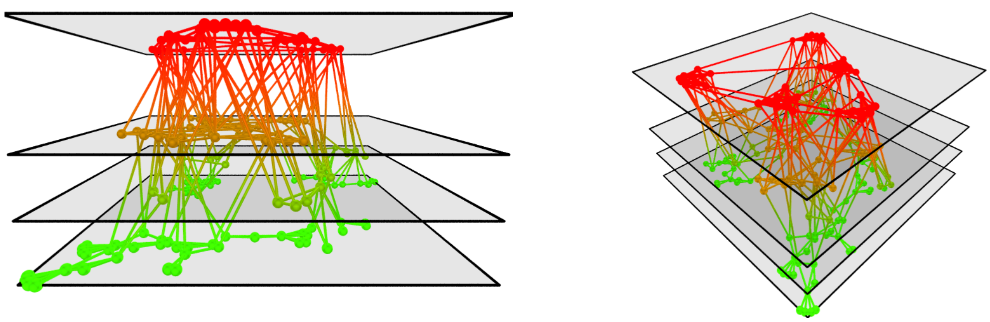
several examples of these techniques are presented. Layers and colors are very effective in real three dimensional drawings. We used transparent rectangles that absorbed light to draw layers and colored the nodes accordingly to their coreness. The color of the edges are determined by a linear interpolation of the colors assigned to their incident nodes.

4.3 Artefacts

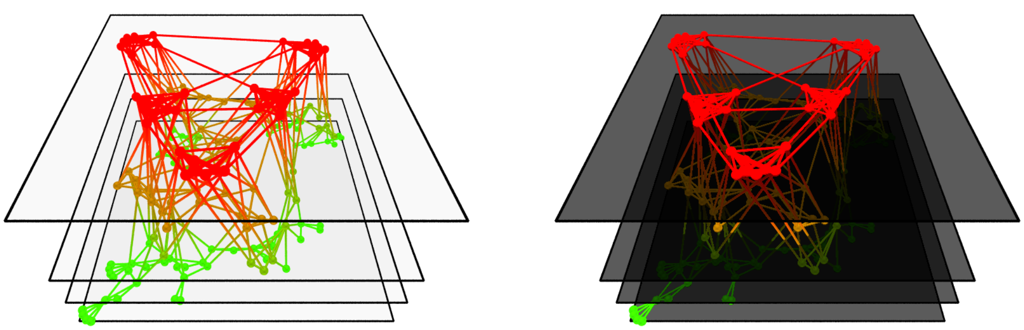
The core of the AS graph is usually very dense and relatively small. In fact the core was very clique-like and had a diameter of two in the beginning of 2001. This has an enormous impact on the quality of the initial spectral layout. There exist different techniques to compensate this fact. Figure 14 presents spectral layouts based on different matrices. The normalized adjacency matrix and the Laplacian matrix emphasize the effect that in the induced spectral layout of a dense graph less connected nodes are pushed towards the periphery. This effect is reduced by the modified Laplacian matrix. At first sight, it is not obvious which matrix should be used. In our opinion, the more uniform layouts induced by the modified Laplacian matrix increases the readability as connections are easier to recognize.



(a) view from above (b) view from below

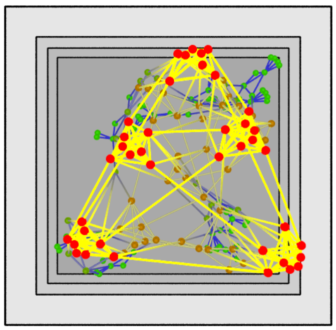


(c) view from one side (d) perspective view

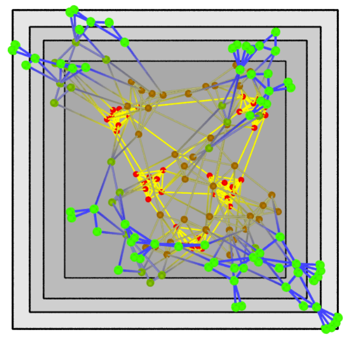


(e) using (very) transparent layers (f) using absorbing layers

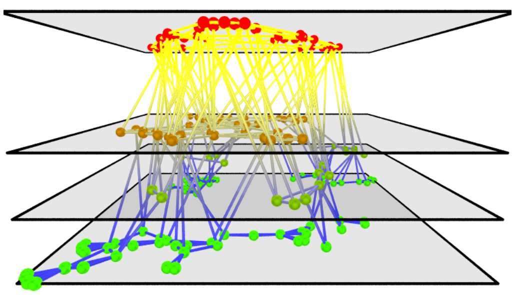
Figure 7: Visual support features using same the color channels for nodes and edges



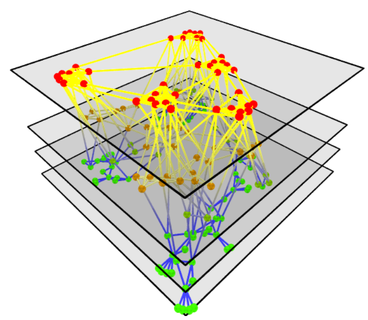
(a) view from above



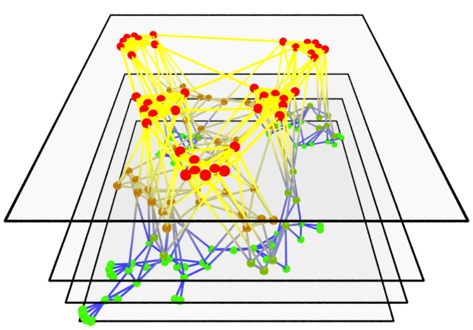
(b) view from below



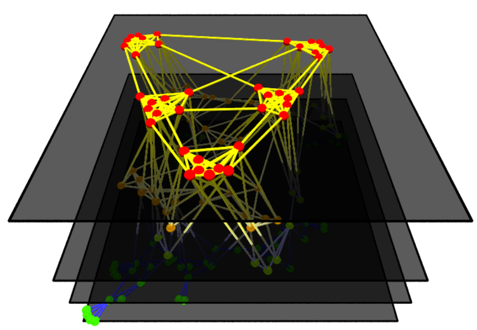
(c) view from one side



(d) perspective view

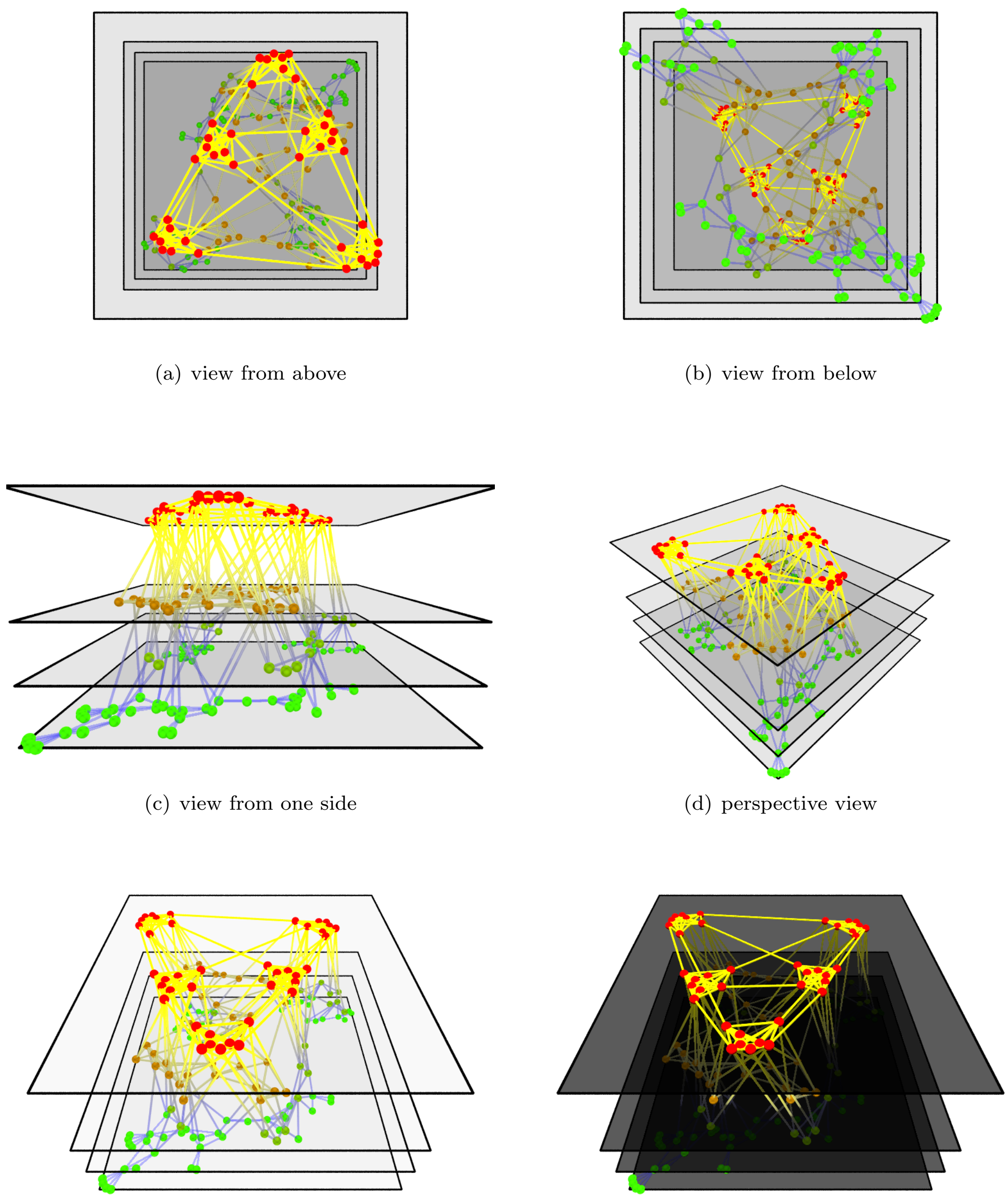


(e) using (very) transparent layers



(f) using absorbing layers

Figure 8: Visual support features using different color channels for nodes and edges



(a) view from above

(b) view from below

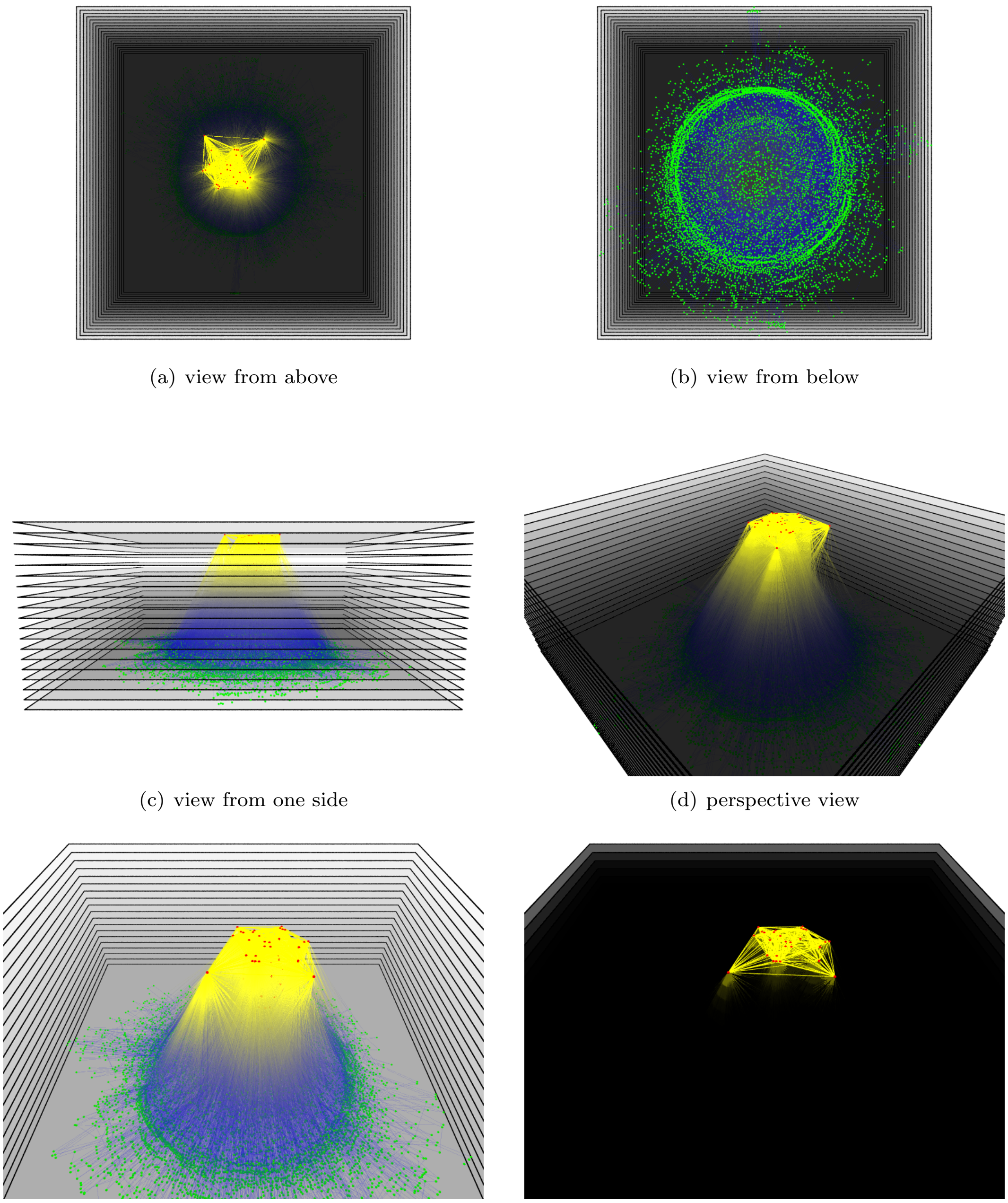
(c) view from one side

(d) perspective view

(e) using (very) transparent layers

(f) using absorbing layers

Figure 9: Visual support features using transparent edges that corresponds to the layer's height



(a) view from above

(b) view from below

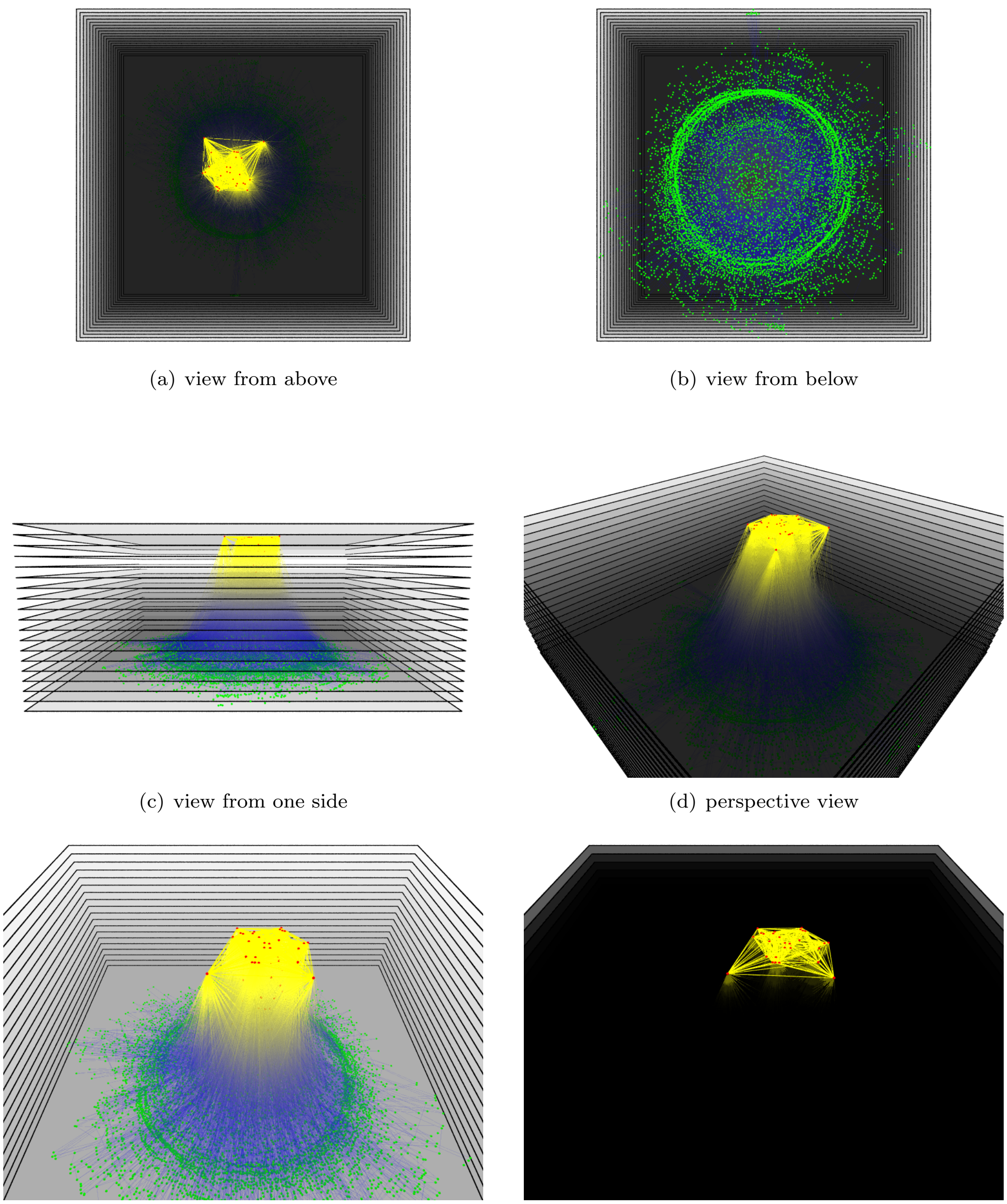
(c) view from one side

(d) perspective view

(e) using (very) transparent layers

(f) using absorbing layers

Figure 10: all edges



(a) view from above

(b) view from below

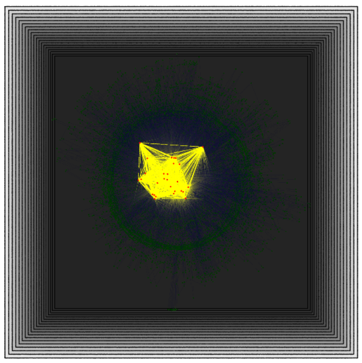
(c) view from one side

(d) perspective view

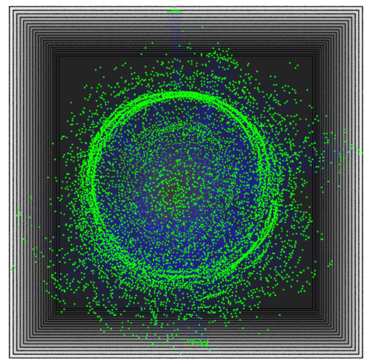
(e) using (very) transparent layers

(f) using absorbing layers

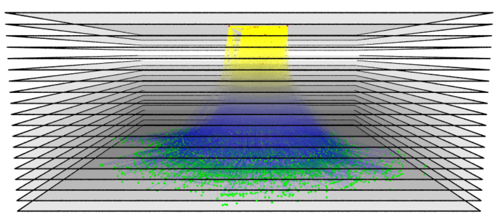
Figure 11: without (2,corenumber) edges



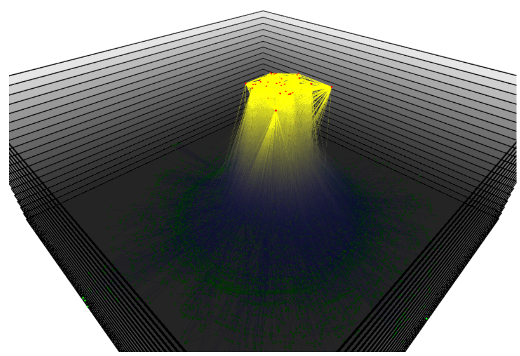
(a) view from above



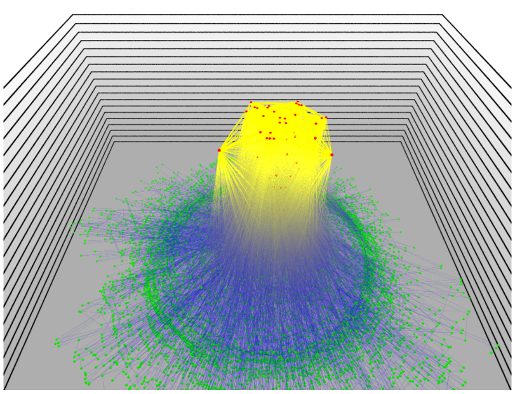
(b) view from below



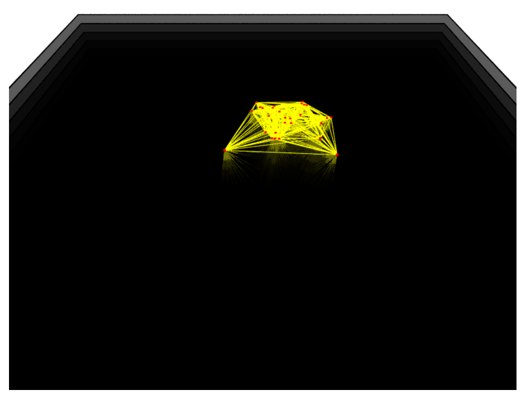
(c) view from one side



(d) perspective view

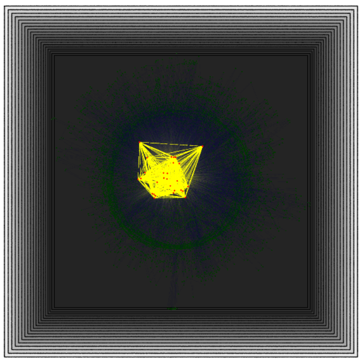


(e) using (very) transparent layers

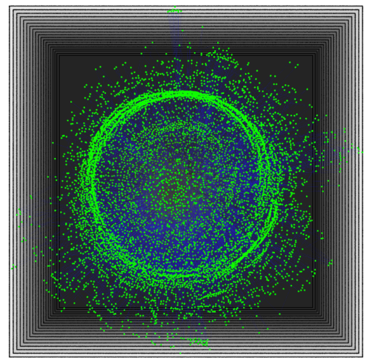


(f) using absorbing layers

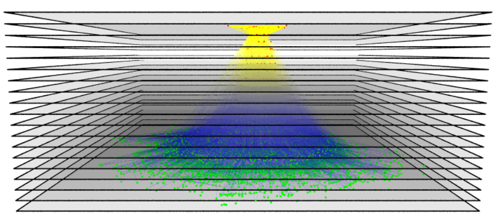
Figure 12: without ({2,3}, corenumber) edges



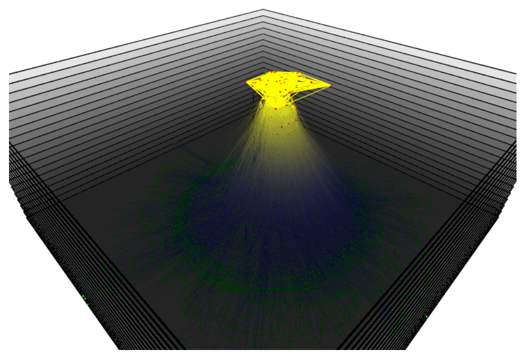
(a) view from above



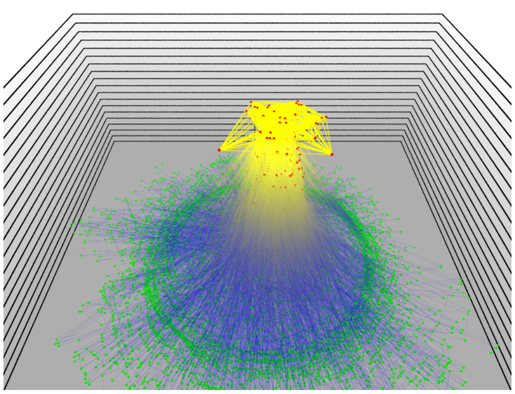
(b) view from below



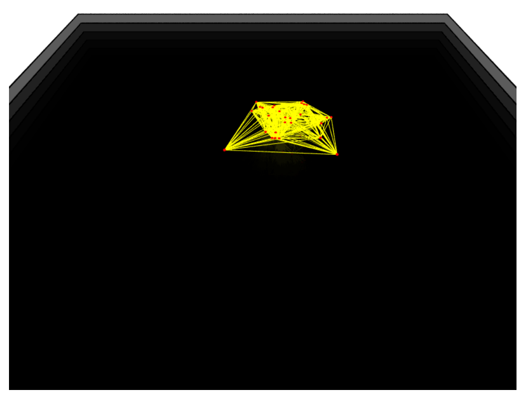
(c) view from one side



(d) perspective view

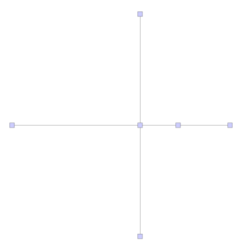


(e) using (very) transparent layers

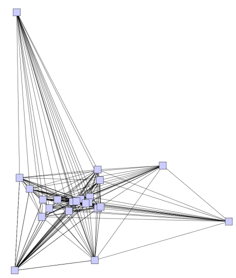


(f) using absorbing layers

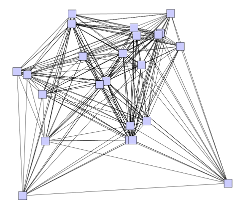
Figure 13: only few core edges left



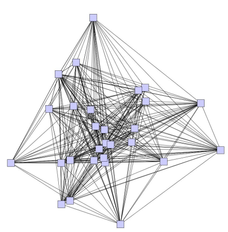
(a) normalized adjacency matrix (06/01/01)



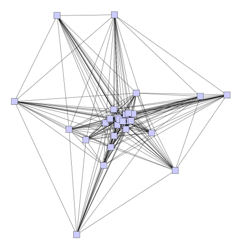
(b) Laplacian matrix (06/01/01)



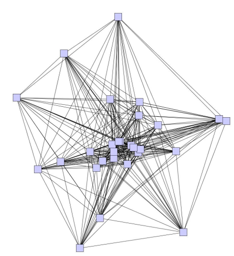
(c) modified Laplacian matrix (06/01/01)



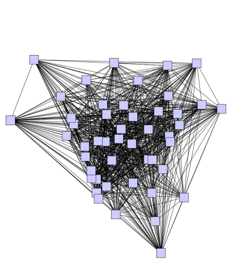
(d) normalized adjacency matrix (06/01/02)



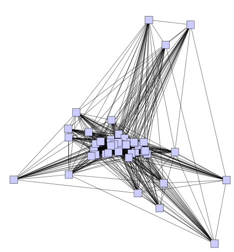
(e) Laplacian matrix (06/01/02)



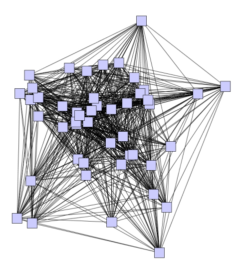
(f) modified Laplacian matrix (06/01/02)



(g) normalized adjacency matrix (06/01/03)



(h) Laplacian matrix (06/01/03)



(i) modified Laplacian matrix (06/01/03)

Figure 14: Different spectral layouts for the core of the AS graph

5 Conclusion

Core based 2.5D visualizations of the AS graph support the recognition of its detailed hierarchy. Especially, it emphasizes the characteristics of the lower core layers and their connections with the highest layers. The evolution of the AS graph has an observable effect on the layout. Also there is a significant difference in the layouts of real AS graphs and generated ones.

References

[1] V. Batagelj and M. Zaveršnik. Generalized cores. Preprint 799, Universtiy of Ljibljana, 2002.

[2] Guiseppe Di Battista, Federico Mariani, Maurizio Patrignani, and Maurizio Pizzonia. Bgplay: A system for visualizing the interdomain routing evolution. In Giuseppe Liotta, editor, *11th International Symposium on Graph Drawing, GD'03*, volume 2912 of *LNCS*, pages 295–306. Springer, 2004.

[3] Ulrik Brandes, Michael Baur, Marco Gaertler, and Dorothea Wagner. Drawing the as graph in two and a half dimensions. Technical report, Faculty of Informatics, University Karlsruhe, 2004.

[4] Ulrik Brandes and Steven R. Corman. Visual unrolling of network evolution and the analysis of dynamic discourse. *Information Visualization*, 2003.

[5] Ulrik Brandes and Sabine Cornelsen. Visual ranking of link structures. *Journal of Graph Algorithms and Applications*, 2003.

[6] Ulrik Brandes, Tim Dwyer, and Falk Schreiber. Visual understanding of metabolic pathways across organisms using layout in two and a half dimensions. *Journal of Integrative Bioinformatics*, 2004.

[7] CAIDA. Visualizing internet topology at a macroscopic scale. Available from World Wide Web: http://www.caida.org/analysis/topology/as_core_network/.

[8] CAIDA. Walrus - graph visualization tool. Available from World Wide Web: http://www.caida.org/tools/visualization/walrus/.

[9] Peter Eades and Qing-Wen Feng. Multilevel visualization of clustered graphs. *Graph Drawing*, 1996.

[10] T. Fruchtermann and E. Reingold. Graph drawing by force-directed placement. *Software - Practice and Experience*, 1991.

[11] Marco Gaertler and Maurizio Patrignani. Dynamic analysis of the autonomous system graph. In *IPS 2004 – Inter-Domain Performance and Simulation*, 2004.

[12] Christos Gkantsidi, Milena Mihail, and Ellen Zegura. Spectral analysis of internet topologies. In *IEEE Infocom 2003*, 2003.

[13] Oregon Routeview Project. Available from World Wide Web: http://www.routeviews.org.

[14] G. Sagie and A. Wool. A clustering approach for exploring the internet structure. In *To appear in Proc. 23rd IEEE Convention of Electrical and Electronics Engineers in Israel (IEEEI)*, 2004.

[15] S. B. Seidman. Network structure and minimum degree. *Social Networks*, 5(5):269–287, 1983.

[16] Jared Winick and Sugih Jamin. Inet-3.0: Internet topology generator. Technical Report UM-CSE-TR-456-02, EECS, University of Michigan, 2002. Available from World Wide Web: citeseer.nj.nec.com/526211.html.