Large scale networks fingerprinting and visualization using the k-core decomposition

J. Ignacio Alvarez-Hamelin[*]
LPT (UMR du CNRS 8627),
Université de Paris-Sud,
91405 ORSAY Cedex France
`Ignacio.Alvarez-Hamelin@lri.fr`

Luca Dall'Asta
LPT (UMR du CNRS 8627),
Université de Paris-Sud,
91405 ORSAY Cedex France
`Luca.Dallasta@th.u-psud.fr`

Alain Barrat
LPT (UMR du CNRS 8627),
Université de Paris-Sud,
91405 ORSAY Cedex France
`Alain.Barrat@th.u-psud.fr`

Alessandro Vespignani
School of Informatics,
Indiana University,
Bloomington, IN 47408, USA
`alexv@indiana.edu`

Abstract

We use the k-core decomposition to develop algorithms for the analysis of large scale complex networks. This decomposition, based on a recursive pruning of the least connected vertices, allows to disentangle the hierarchical structure of networks by progressively focusing on their central cores. By using this strategy we develop a general visualization algorithm that can be used to compare the structural properties of various networks and highlight their hierarchical structure. The low computational complexity of the algorithm, $\mathcal{O}(n+e)$, where n is the size of the network, and e is the number of edges, makes it suitable for the visualization of very large sparse networks. We show how the proposed visualization tool allows to find specific structural fingerprints of networks.

1 Introduction

In recent times, the possibility of accessing, handling and mining large-scale networks datasets has revamped the interest in their investigation and theoretical characterization along with the definition of new modeling frameworks. In particular, mapping projects of the World Wide Web and the physical Internet offered the first chance to study topology and traffic of large-scale networks. Other studies followed describing population networks of practical interest in social science, critical infrastructures and epidemiology [1, 2, 3]. The study of large scale networks, however, faces us with an array of new challenges. The definitions of centrality, hierarchies and structural organizations are hindered by the large size of these networks and the complex interplay of connectivity patterns, traffic flows and geographical, social and economical attributes characterizing their basic elements. In this

[*]Further author information: J.I.A-H. is also with Facultad de Ingeniería, Universidad de Buenos Aires, Paseo Colón 850, C 1063 ACV Buenos Aires, Argentina.

context, a large research effort is devoted to provide effective visualization and analysis tools able to cope with graphs whose size may easily reach millions of vertices.

In this paper, we propose a visualization algorithm based on the k-core decomposition able to uncover in a two-dimensional layout several topological and hierarchical properties of large scale networks. The k-core decomposition [4] consists in identifying particular subsets of the graph, called k-cores, each one obtained by recursively removing all the vertices of degree smaller than k, until the degree of all remaining vertices is larger than or equal to k. Larger values of the index k clearly correspond to vertices with larger degree and more central position in the network's structure.

This visualization tool allows the identification of real or computer-generated networks' fingerprints, according to properties such as hierarchical arrangement, degree correlations and centrality. The distinction between networks with seemingly similar properties is achieved by inspecting the different layouts generated by the visualization algorithm. In addition, the running time of the algorithm grows only linearly with the size of the network, granting the scalability needed for the visualization of very large sparse networks. The proposed (publicly available [5]) algorithm appears therefore as a convenient method for the general analysis of large scale complex networks and the study of their architecture.

The paper is organized as follows: after a brief survey on k-core studies (section 2), we present the basic definitions and the graphical algorithms in section 3 along with the basic features of the visualization layout. Section 4 shows how the visualizations obtained with the present algorithm may be used for network fingerprinting, and presents two examples of visualization of real networks.

2 Related work

While a large number of algorithms aimed at the visualization of large scale networks have been developed (e.g., see [6]), only a few consider explicitly the k-core decomposition. Vladimir Batagelj *et al.* [7] studied the k-core decomposition applied to visualization problems, introducing some graphical tools to analyse the cores, mainly based on the visualization of the adjacency matrix of certain k-cores. To the best of our knowledge, the algorithm presented by Baur *et al.* in [8] is the only one completely based on a k-core analysis and directly targeted at the study of large information networks. This algorithm uses a spectral layout to place vertices having the largest shell index. A combination of barycentric and iteratively directed-forces allows to place the vertices of each k-shell, in decreasing order. Finally, the network is drawn in three dimensions, using the z axis to place each shell in a distinct horizontal layer. Note that the spectral layout is not able to distinguish two or more disconnected components. The algorithm by Baur *et al.* is also tuned for representing AS graphs and its total complexity depends on the size of the highest k-core (see [9] for more details on spectral layout), making the computation time of this proposal largely variable. In this respect, the algorithm presented here is different in that it can represent networks in which k-cores are composed by several connected components. Another difference is that representations in 2D are more suited for information visualization than other representations (see [10] and references therein). Finally, the algorithm parameters can be universally defined, yielding a fast and general tool for analyzing all types of networks.

It is interesting to note that the notion of k-cores has been recently used in biologically related contexts, where it was applied to the analysis of protein interaction networks [11] or in the prediction of protein functions [12, 13]. Further applications in Internet-related areas can be found in [14], where the k-core decomposition is used for filtering out peripheral Autonomous Systems (ASes), and in [15] where the scale invariant structure of degree correlations and mapping biases in AS maps is shown. Finally in [16, 17], an interesting approach based on the k-core decomposition has been used to provide a conceptual and structural model of the Internet; the so-called medusa model for the Internet.

3 Graphical representation

Let us consider a graph $G = (V, E)$ of $|V| = n$ vertices and $|E| = e$ edges; a k-core is defined as follows [4]:

-A subgraph $H = (C, E|C)$ induced by the set $C \subseteq V$ is a k-*core* or a core of order k iff $\forall v \in C : \texttt{degree}_H(v) \geq k$, and H is the maximum subgraph with this property.

A k-core of G can therefore be obtained by recursively removing all the vertices of degree less than k, until all vertices in the remaining graph have at least degree k. Furthermore, we will use the following definitions:

-A vertex i has *shell index* c if it belongs to the c-core but not to $(c+1)$-core. We denote by c_i the shell index of vertex i.

-A shell C_c is composed by all the vertices whose shell index is c. The maximum value c such that C_c is not empty is denoted $c_{\max}$. The k-core is thus the union of all shells C_c with $c \geq k$.

-Each connected set of vertices having the same shell index c is a *cluster* Q^c. Each shell C_c is thus composed by clusters Q^c_m, such that $C_c = \cup_{1 \leq m \leq q^c_{\max}} Q^c_m$, where $q^c_{\max}$ is the number of clusters in C_c.

The visualization algorithm we propose places vertices in 2 dimensions, the position of each vertex depending on its shell index and on the index of its neighbors. A color code allows for the identification of shell indices, while the vertex's original degree is provided by its size that depends logarithmically on the degree. For the sake of clarity, our algorithm represents a small percentage of the edges, chosen uniformly at random. As mentioned, a central role in our visualization method is played by multi-components representation of k-cores. In the most general situation, indeed, the recursive removal of vertices having degree less than a given k can break the original network into various connected components, each of which might even be once again broken by the subsequent decomposition. Our method takes into account this possibility, however we will first present the algorithm in the simplified case, in which none of the k-cores is fragmented. Then, this algorithm will be used as a subroutine for treating the general case (Table 1).

3.1 Drawing algorithm for k-cores with single connected component

k-core decomposition. The shell index of each vertex is computed and stored in a vector $\mathcal{C}$, along with the shells C_c and the maximum index c_{max}. Each shell is then decomposed into clusters Q^c_m of connected vertices, and each vertex i is labeled by its shell index c_i and by a number q_i representing the cluster it belongs to.

The two dimensional graphical layout. The visualization is obtained assigning to each vertex i a couple of polar coordinates (ρ_i, α_i): the radius ρ_i is a function of the shell index of the vertex i and of its neighbors; the angle α_i depends on the cluster number q_i. In this way, k-shells are displayed as layers with the form of circular shells, the innermost one corresponding to the set of vertices with highest shell index. A vertex i belongs to the $c_{\max} - c_i$ layer from the center.

More precisely, ρ_i is computed according to the following formula:

$$\rho_i = (1-\epsilon)(c_{\max} - c_i) + \frac{\epsilon}{|V_{c_j \geq c_i}(i)|} \sum_{j \in V_{c_j \geq c_i}(i)} (c_{\max} - c_j) \ , \qquad (1)$$

$V_{c_j \geq c_i}(i)$ is the set of neighbors of i having shell index c_j larger or equal to c_i. The parameter ϵ controls the possibility of rings overlapping, and is one of the only three external parameters required to tune image's rendering.

Inside a given shell, the angle α_i of a vertex i is computed as follow:

$$\alpha_i = 2\pi \sum_{1 \leq m < q_i} \frac{|Q_m|}{|C_{c_i}|} + \mathbf{N}\left(\frac{|Q_{q_i}|}{2|C_{c_i}|} \ , \ \pi \cdot \frac{|Q_{q_i}|}{|C_{c_i}|} \right) \ , \tag{2}$$

where Q_{q_i} and C_{c_i} are respectively the cluster q_i and c_i-shell the vertex belongs to, $\mathbf{N}$ is a normal distribution of mean $|Q_{q_i}|/(2|C_{c_i}|)$ and width $2\pi|Q_{q_i}|/|C_{c_i}|$. Since we are interested in distinguishing different clusters in the same shell, the first term on the right side of Eq. 2, referring to clusters with $m < q_i$, allows to allocate a correct partition of the angular sector to each cluster. The second term on the right side of Eq. 2, on the other hand, specifies a random position for the vertex i in the sector assigned to the cluster Q_{q_i}.

Colors and size of vertices. Colors depend on the shell index: vertices with shell index 1 are violet, and the maximum shell index vertices are red, following the rainbow color scale. The diameter of each vertex corresponds to the logarithm of its degree, giving a further information on vertex's properties. The vertices with largest shell index are placed uniformly in a disk of radius u, which is the unit length ($u = 1$ for this reduced algorithm).

3.2 Extended algorithm for networks with many k-cores components

The algorithm presented in the previous section can be used as the basic routine to define an extended algorithm aimed at the visualization of networks for which some k-cores are fragmented; i.e. made by more than one connected component. This issue is solved by assigning to each connected component of a k-core a center and a size, which depends on the relative sizes of the various components. Larger components are put closer to the global center of the representation (which has Cartesian coordinates $(0, 0)$), and have larger sizes.

The algorithm begins with the center at the origin $(0, 0)$. Whenever a connected component of a k-core, whose center p had coordinates (X_p, Y_p), is broken into several components by removing all vertices of degree k, i.e. by applying the next decomposition step, a new center is computed for each new component. The center of the component h has coordinates (X_h, Y_h), defined by

$$X_h = X_p + \delta(c_{\max} - c_h) \cdot u_p \cdot \varrho_h \cdot \cos(\phi_h) \ ; \ Y_h = \ Y_p + \delta(c_{\max} - c_h) \cdot u_p \cdot \varrho_h \cdot \sin(\phi_h) \ , \tag{3}$$

where δ scales the distance between components, $c_{\max}$ is the maximum shell index and c_h is the core number of component h (the components are numbered by $h = 1, \cdots, h_{max}$ in an arbitrary order), u_p is the unit length of its parent component, ϱ_h and ϕ_h are the radial and angular coordinates of the new center with respect to the parent center (X_p, Y_p). We define ϱ_h and ϕ_h as follows:

$$\varrho_h = 1 - \frac{|S_h|}{\sum_{1 \leq j \leq h_{max}} |S_j|} \ ; \ \phi_h = \phi_{ini} + \frac{2\pi}{\sum_{1 \leq j \leq h_{max}} |S_j|} \sum_{1 \leq j \leq h} |S_j| \ , \tag{4}$$

where S_h is the set of vertices in the component h, $\sum_j |S_j|$ is the sum of the sizes of all components having the same parent component. In this way, larger components will be closer to the original parent component's center p. The angle ϕ_h has two contributions. The initial angle ϕ_{ini} is chosen uniformly at random[1], while the angle sector is the sum of component angles whose number is less than or equal to the actual component number h.

[1]Note that if ϕ_{ini} is fixed, all the centers of the various components are aligned in the final representation.

Algorithm 1

```
1  $k := 1$ and $end :=$ false
2  while not $end$ do
3      $(end, \mathcal{C}) \leftarrow$ make_core $k$
4      $(\mathcal{Q}, \mathcal{T}) \leftarrow$ compute_clusters $k-1$, if $k > 1$
5      $\mathcal{S} \leftarrow$ compute_components $k$
6      $(X, Y) \leftarrow$ compute_origin_coordinates_cmp $k$ (Eqs. from 3 to 4)
7      $U \leftarrow$ compute_unit_size_cmp $k$ (Eq. 5)
8      $k := k + 1$
9  for each node $i$ do
10     if $c_i == c_{\max}$ then
11         set $\rho_i$ and $\alpha_i$ according to a uniform distribution in the disk of radius $u$ ($u$ is the core
           representation unit size)
12     else
13         set $\rho_i$ and $\alpha_i$ according to Eqs. 1 and 2
14 $(\mathcal{X}, \mathcal{Y}) \leftarrow$ compute_final_coordinates $\rho\ \alpha\ U\ X\ Y$ (Eq. 6)
```

Table 1: Algorithm for the representation of networks using k-cores decomposition

Finally, the unit length u_h of a component h is computed as

$$u_h = \frac{|S_h|}{\sum_{1 \le j \le h_{max}} |S_j|} \cdot u_p \ , \tag{5}$$

where u_p is the unit length of its parent component. Larger unit length and size are therefore attributed to larger components.

For each vertex i, radial and angular coordinates are computed by equations 1 and 2 as in the previous algorithm. These coordinates are then considered as relative to the center (X_h, Y_h) of the component to which i belongs. The position of i is thus given by

$$x_i = X_h + \gamma \cdot u_h \cdot \rho_i \cdot \cos(\alpha_i); \ \ y_i = \ Y_h + \gamma \cdot u_h \cdot \rho_i \cdot \sin(\alpha_i) \tag{6}$$

where γ is a parameter controlling the component's diameter.

The global algorithm is formally presented in Table 1. The main loop is composed by the following functions. First, the function $\{(end, \mathcal{C}) \leftarrow$ make_core $k\}$ recursively removes all vertices of degree $k-1$, obtaining the k-core, and stores into $\mathcal{C}$ the shell index $k-1$ of the removed vertices. The boolean variable end is set to $true$ if the k-core is empty, otherwise it is set to $false$. The function $\{(\mathcal{Q}, \mathcal{T}) \leftarrow$ compute_clusters $k-1\}$ operates the decomposition of the $(k-1)$-shell into clusters, storing for each vertex the cluster label into the vector $\mathcal{Q}$, and filling table $\mathcal{T}$, which is indexed by the shell index c and cluster label q: $\mathcal{T}(c,q) = (\sum_{1 \le m < q} |Q_m|/|C_c|, |Q_q|/|C_c|)$. The possible decomposition of the k-core into connected components is determined by function $\{\mathcal{S} \leftarrow$ compute_components $k\}$, that also collects into a vector $\mathcal{S}$ the number of vertices contained in each component. At the following step, functions $\{(X, Y) \leftarrow$compute_origin_coordinates_cmp $k\}$ and $\{U \leftarrow$compute_unit_size_cmp $k\}$ get, respectively, the center and size of each component of the k-core, gathering them in vectors X, Y and U. Finally, the coordinates of each vertex are computed and stored in the vectors $\mathcal{X}$ and $\mathcal{Y}$.

Algorithm complexity. Batagelj and Zversnik [18] present an algorithm to perform the k-core decomposition, and show that its time complexity is $\mathcal{O}(e)$ (where e is the number of edges) for a connected graph. For a general graph it is $\mathcal{O}(n+e)$, where n is the number of nodes, which makes the algorithm very efficient for sparse graphs where e is of order n.

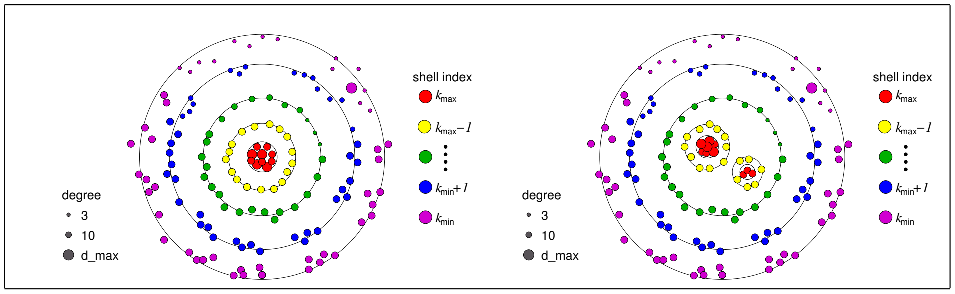

Figure 1: Structure of a typical layout in two important cases: on the left, all k-cores are connected; on the right, some k-cores are composed by more than one connected component. The vertices are arranged in a series of concentric shells corresponding to the various k-shells. The diameter of each shell depends on both the shell index and, in case of multiple components (right) also on the relative fraction of vertices belonging to the different components.

3.3 Basic features of the visualization's layout

The main features of the layout's structure obtained with the above algorithms are visible in Fig.1 where, for the sake of simplicity, we do not show any edge.

The two-dimensional layout is composed of a series of concentric *circular shells*. Each shell corresponds to a single *shell index* and all vertices in it are therefore drawn with the *same color*. A color scale allows to distinguish different *shell indices*: the violet is used for the minimum shell index k_{min}, then we use a graduated rainbow scale for higher and higher shell indices up to the maximum value k_{max} that is colored in red. The diameter of each k-shell depends on the *shell index* k, and is proportional to $k_{max} - k$ (In Fig.1, the position of each shell is identified by a circle having the corresponding diameter). The presence of a trivial order relation in the shell indices ensures that all shells are placed in a concentric arrangement. On the other hand, when a k-core is fragmented in two or more components, the diameters of the different components depend also on the relative number of vertices belonging to each of them, i.e. the fraction between the number of vertices belonging to that component and the total number of vertices in that core. This is a very important information, providing a way to distinguish between multiple components at a given shell index. Finally, the size of each node is proportional to the *original degree* of that vertex; we use a logarithmic scale for the size of the drawn bullets.

4 Network fingerprinting

The k-core decomposition peels the network layer by layer, revealing the structure of the different shells from the outmost one to the more internal ones. The algorithm provides a direct way to distinguish the network's different hierarchies and structural organization by means of some simple quantities: the radial width of the shells, the presence and size of clusters of vertices in the shells, the correlations between degree and shell index, the distribution of the edges interconnecting vertices of different shells, etc.

1) ***Shells Width:*** The thickness of a shell depends on the shell index properties of the neighbors of the vertices in the corresponding shell. For a given shell-diameter (black circle in the median position of shells in Fig.2), each vertex can be placed more internal or more external with respect to this reference. Nodes with more neighbors in higher shells are closer to the center and viceversa: in Fig.2, node y is more internal than node x because it

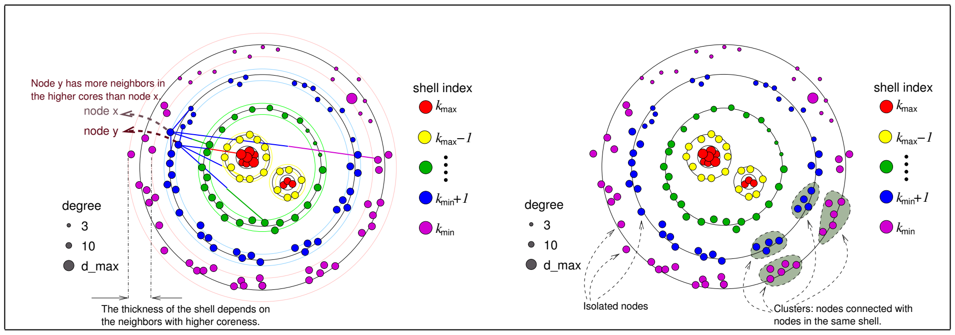
Node y has more neighbors in the higher cores than node x.
node x
node y
shell index
$k_{\max}$
$k_{\max}-1$
$k_{\min}+1$
$k_{\min}$
degree
3
10
d_max
The thickness of the shell depends on the neighbors with higher coreness.
shell index
$k_{\max}$
$k_{\max}-1$
$k_{\min}+1$
$k_{\min}$
degree
3
10
d_max
Isolated nodes
Clusters: nodes connected with nodes in the same shell.

Figure 2: Left: each shell has a certain radial width. This width depends on the correlation's properties of the vertices in the shell. In the second shell, we have pinpointed two nodes x and y. Node y is more internal than x because a larger part of its neighbors belongs to higher k-shells compared to x's neighbors. The figure on the right shows the clustering properties of nodes in the same k-shell. In each k-shell, nodes that are directly connected between them (in the original graph) are drawn close one to the other, as in a cluster. Some of these sets of nodes are circled and highlighted in gray. Three examples of isolated nodes are also indicated; these nodes have no connections with the others of the same shell.

has three edges towards higher index nodes, while x has only one. The maximum thickness of the shells is controlled by the ϵ parameter (Eq. 1).

2) *Shell Clusters:* The angular distribution of vertices in the shells is not completely homogeneous. Fig.2 shows that clusters of vertices can be observed. The idea is to group together all nodes of the same shell that are directly linked in the original graph and to represent them close one to another. Thus, a shell is divided in many angular sectors, each containing a cluster of vertices. This feature allows to figure out at a glance if the shells are composed of a single large connected component rather than divided into many small clusters, or even if there are isolated vertices (i.e. disconnected from all other nodes in the shell, not from the rest of the k-core!).

3) *Degree-Shell index Correlation:* Another property that can be studied from the obtained layouts is the correlation between the degree of the nodes and the shell index. Both quantities are centrality measures and the nature of their correlations is a very important feature characterizing a network's topology. The nodes displayed in the most internal shells are those forming the central core of the network; the presence of degree-index correlations then corresponds to the fact that the central nodes are most likely high-degree hubs of the network. This effect is observed in many real communication networks with a clear hierarchical structure, such as the Internet at the Autonomous System level or the World Wide Air-transportation network [5]. On the contrary, the presence of hubs in external shells is typical of less hierarchically structured networks such as the World-Wide Web or the Internet Router Level. In this case, star-like configurations appear with high degree vertices connected only to very low degree vertices. These vertices are rapidly pruned out in the k-core decomposition even if they have a very high degree, leading to the presence of local hubs in external shells, as in Fig. 3.

4) *Edges:* The visualization shows only a homogeneously randomly sampled fraction of the edges, which can be tuned in order to get the better trade-off between the clarity of visualization and the necessity of giving information on the way the nodes are mainly connected. Edge-reduction techniques can be implemented to improve the algorithm's capacity in representing edges; however, a homogeneous sampling does not alter the extraction of topological information, ensuring a low computational cost. Finally, the two halves of each

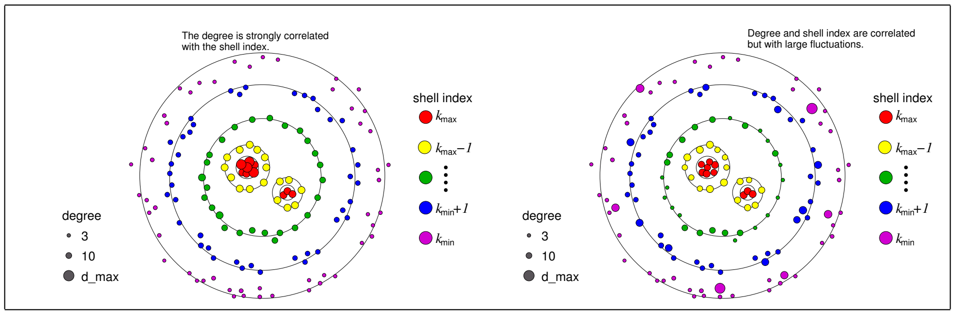
The degree is strongly correlated with the shell index.
Degree and shell index are correlated but with large fluctuations.
shell index
k_{max}
$k_{max}-1$
$k_{min}+1$
k_{min}
degree
3
10
d_max

Figure 3: Correlations between shell index and degree. On the left, we report a graph with strong correlation: the size of the nodes grows from the periphery to the center, in correspondence with the shell index. In the right-hand case, the degree-index correlations are blurred by large fluctuations, as stressed by the presence of hubs in the external shells.

edge are colored with the color of the corresponding extremities to emphasize the connection among vertices in different shells.

5) *Disconnected components:* The fragmentation of any given k-core in two or more disconnected components is represented by the presence of a corresponding number of circular shells with different centers (Fig. 1). The diameter of these circles is related with the number of nodes of each component and modulated by the γ parameter (Eq. 6). The distance between components is controlled by the δ parameter (Eq. 3).

In summary, the proposed algorithm makes possible a direct, visual investigation of a series of properties: hierarchical structures of networks, connectivity and clustering properties inside a given shell; relations and interconnectivity between different levels of the hierarchy, correlations between degree and shell index, i.e. between different measures of centrality.

Numerous examples of the application of this tool to the visualization of real and computer generated networks can be found on the web page of the publicly available tool [5]. For example, the lack of hierarchy and structure of the Erdös-Rényi random graph is clearly identified. Similarly the time correlations present in the Barabási-Albert network find a clear fingerprint in our visualization layout. Here we display another interesting illustration of the use and capabilities of the proposed algorithm in the analysis of large sparse graphs: the identification of the different hierarchical arrangement of the Internet network when visualized at the Autonomous system (AS) and the Router (IR) levels [2]. The AS level is represented by collected routes of *Oregon route-views* [19] project, from May 2001. For the IR level, we use the graph obtained by an exploration of Govindan and Tangmunarunkit [20] in 2000. These networks are composed respectively by about 11500 and 200000 nodes.

Figures 4 and 5 display the representations of these two different maps of Internet. At the AS level, all shells are populated, and, for any given shell, the vertices are distributed on a relatively large range of the radial coordinate, which means that their neighborhoods are variously composed. The shell index and the degree are very correlated, with a clear hierarchical structure, and links go principally from one shell to another. The hierarchical structure exhibited by our analysis of the AS level is a striking property; for instance, one might exploit it for showing that in the Internet high-degree vertices are naturally (as an implicit result of the self-organizing growth) placed in the innermost structure. At higher resolution, i.e. at the IR level, Internet's properties are less structured: external layers, of

[2]The parameters are here set to the values $\epsilon = 0.18$, $\delta = 1.3$ and $\gamma = 1.5$.

lowest shell index, contain vertices with large degree. For instance, we find 20 vertices with degree larger than 100 but index smaller than 6. The correlation between shell index and degree is thus clearly of a very different nature in the maps of Internet obtained at different granularities.

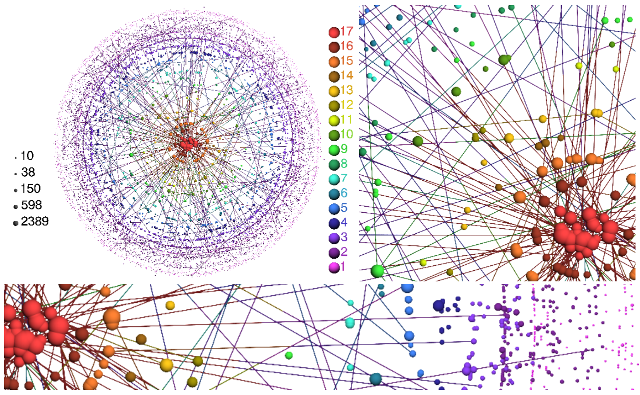

Figure 4: Graphical representation of the AS network. The three snapshots correspond to the full network (top left), with the color scale of the shell index and the size scale for the nodes' degrees, and to two magnifications showing respectively a more central part (top right) and a radial slice of the layout (bottom).

5 Conclusions

Exploiting k-core decomposition, and the corresponding natural hierarchical structures, we develop a visualization algorithm that yields a layout encoding a considerable amount of the information needed for network fingerprinting in the simplicity of a 2D representation. One can easily read basic features of the graph (degree, hierarchical structure, etc.) as well as more entangled features, e.g. the relation between a vertex and the hierarchical position of its neighbors. The present visualization strategy is a useful tool to discriminate between networks with different topological properties and structural arrangement, and may be also used for comparison of models with real data, providing a further interesting tool for model

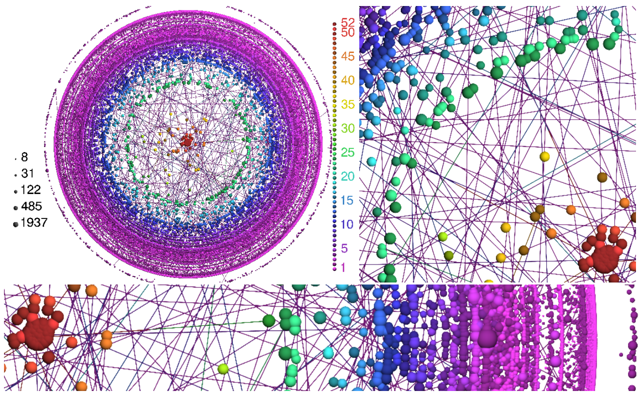

Figure 5: Same as Fig. 4, for the graphical representation of the IR network.

validation. Finally, we also provide a publicly available tool for visualizing networks [5].

Acknowledgments: This work has been partially funded by the European Commission - Fet Open project COSIN IST-2001-33555 and contract 001907 (DELIS).

References

[1] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Rev. Mod. Phys.* **74**, pp. 47–97, 2000.

[2] S. N. Dorogovtsev and J. F. F. Mendes, *Evolution of networks: From biological nets to the Internet and WWW*, Oxford University Press, 2003.

[3] R. Pastor-Satorras and A. Vespignani, *Evolution and structure of the Internet: A statistical physics approach*, Cambridge University Press, 2004.

[4] V. Batagelj and M. Zaversnik, "Generalized Cores," *cs.DS/0202039* , 2002.

[5] LArge NETwork VIsualization tool. `http://xavier.informatics.indiana.edu/lanet-vi/`.

[6] `http://http://i11www.ira.uka.de/cosin/tools/index.php`.

[7] V. Batagelj, A. Mrvar, and M. Zaversnik, "Partitioning Approach to Visualization of Large Networks," in *Graph Drawing '99, Castle Stirin, Czech Republic, LNCS 1731*, pp. 90–98, 1999.

[8] M. Baur, U. Brandes, M. Gaertler, and D. Wagner, "Drawing the AS Graph in 2.5 Dimensions," in *"12th International Symposium on Graph Drawing, Springer-Verlag editor"*, pp. 43–48, 2004.

[9] U. Brandes and S. Cornelsen, "Visual Ranking of Link Structures," *Journal of Graph Algorithms and Applications* **7**(2), pp. 181–201, 2003.

[10] B. Shneiderman, "Why not make interfaces better than 3d reality?," *IEEE Computer Graphics and Applications* **23**, pp. 12–15, November/December 2003.

[11] G. D. Bader and C. W. V. Hogue, "An automated method for finding molecular complexes in large protein interaction networks," *BMC Bioinformatics* **4**(2), 2003.

[12] M. Altaf-Ul-Amin, K. Nishikata, T. Koma, T. Miyasato, Y. Shinbo, M. Arifuzzaman, C. Wada, M. Maeda, T. Oshima, H. Mori, and S. Kanaya, "Prediction of Protein Functions Based on K-Cores of Protein-Protein Interaction Networks and Amino Acid Sequences," *Genome Informatics* **14**, pp. 498–499, 2003.

[13] S. Wuchty and E. Almaas, "Peeling the Yeast protein network," *Proteomics. 2005 Feb;5(2):444-9.* **5**(2), pp. 444–449, 2005.

[14] M. Gaertler and M. Patrignani, "Dynamic Analysis of the Autonomous System Graph," in *IPS 2004, International Workshop on Inter-domain Performance and Simulation, Budapest, Hungary*, pp. 13–24, 2004.

[15] I. Alvarez-Hamelin, L. Dall'Asta, A. Barrat, and A. Vespignani, "k-core decomposition: a tool for the analysis of large scale internet graphs," *cs.NI/0511007* .

[16] S. Carmi, S. Havlin, S. Kirkpatrick, Y. Shavitt, and E. Shir, 2005. `http://www.cs.huji.ac.il/~kirk/Jellyfish_Dimes.ppt`.

[17] S. Carmi, S. Havlin, S. Kirkpatrick, Y. Shavitt, and E. Shir, "Medusa - new model of internet topology using k-shell decomposition," *cond-mat/0601240* .

[18] V. Batagelj and M. Zaversnik, "An O(m) Algorithm for Cores Decomposition of Networks," *cs.DS/0310049* , 2003.

[19] University of Oregon Route Views Project. `http://www.routeviews.org/`.

[20] R. Govindan and H. Tangmunarunkit, "Heuristics for Internet Map Discovery," in *IEEE INFOCOM 2000*, pp. 1371–1380, IEEE, (Tel Aviv, Israel), March 2000.