A low complexity visualization tool that helps to perform complex systems analysis

2008 New J. Phys. 10 125003

(http://iopscience.iop.org/1367-2630/10/12/125003)

New Journal of Physics
The open-access journal for physics

A low complexity visualization tool that helps to perform complex systems analysis

M G Beiró[1], J I Alvarez-Hamelin[1,2,3] and J R Busch[1]

[1] Facultad de Ingeniería, Universidad de Buenos Aires, Paseo Colón 850, C1063ACV Buenos Aires, Argentina

[2] CONICET (Argentinian Council of Scientific and Technological Research), Buenos Aires, Argentina

E-mail: ignacio.alvarez-hamelin@cnet.fi.uba.ar

New Journal of Physics **10** (2008) 125003 (18pp)
Received 14 April 2008
Published 1 December 2008
Online at http://www.njp.org/
doi:10.1088/1367-2630/10/12/125003

Abstract. In this paper, we present an extension of large network visualization (LaNet-vi), a tool to visualize large scale networks using the k-core decomposition. One of the new features is how vertices compute their angular position. While in the later version it is done using shell clusters, in this version we use the angular coordinate of vertices in higher k-shells, and arrange the highest shell according to a cliques decomposition. The time complexity goes from $O(n\sqrt{n})$ to $O(n)$ upon bounds on a heavy-tailed degree distribution. The tool also performs a k-core-connectivity analysis, highlighting vertices that are not k-connected; e.g. this property is useful to measure robustness or quality of service (QoS) capabilities in communication networks. Finally, the actual version of LaNet-vi can draw labels and all the edges using transparencies, yielding an accurate visualization. Based on the obtained figure, it is possible to distinguish different sources and types of complex networks at a glance, in a sort of 'network iris-print'.

[3] Author to whom any correspondence should be addressed.

Contents

1. Introduction

The study and analysis of complex systems presents a vast field of research today, covering from biological systems, going through social networks, up to Internet and WWW networks. The distinctive feature of these systems is their emerging behaviour, e.g. the way their members become connected or their self-organized hierarchical structure. Several studies have been done on social sciences, critical infrastructures and epidemiology [1]–[4] among others, where systems are normally represented by a graph, called a complex network.

In this paper, we propose visualization as a tool to show internal properties of complex networks. The first version of our tool, large network visualization (LaNet-vi) [5] presents the network graph in a two-dimensional (2D) figure, where each vertex's position is computed mainly using the k-core decomposition and other algorithms, uncovering the hierarchy and the way that vertices are connected, like a precise fingerprint of the complex network. The main key is the low complexity of the k-core decomposition ($O(e)$: linear in the number of edges), which makes it suitable for visualizing large networks. In the second version (LaNet-vi2), presented in this paper, we preserve a low complexity algorithm ($O(n^{3/B})$ for networks with heavy tailed degree distribution, where n is the number of vertices and $2 \leqslant B \leqslant 3$ is the exponent of a power law bounding the distribution), adding the following new features.

Firstly, we draw the maximum core as a collection of cliques[4] and then the remaining vertices compute their angle from the angle of neighbours in higher shells. This allows to establish an 'order' giving an accurate picture of network structure.

Secondly, this new version also produces scalable vector graphics (SVG) figures. In these figures, we can display all the graph edges without overwhelming the picture, by taking advantage of transparencies and thus showing the complete network connectivity.

Thirdly, LaNet-vi2 allows visualization of vertices not observing the core-connectivity, i.e. that the minimum number of paths edge-disjoint is k, where k is the core that contains both ends.

[4] A *clique* is a complete graph, i.e. all vertices are pairwise adjacent.

We present a theorem that assures k-edge-connectivity, and all vertices infringing this property are highlighted in the visualization. This property can be interpreted as the network robustness, e.g. having k paths to connect two vertices means that the network can support at least k broken edges to become disconnected.

Finally, LaNet-vi2 can include vertex labels, which helps to find vertices in the picture, improving the interpretation of the figure.

The remainder of the paper is organized as follows: a brief state of the art is presented in section 2, then new algorithms are shown in section 3, some examples are given in section 4, and finally section 5 concludes the paper.

2. Related work

To the best of our knowledge, there are a few visualization tools using k-core decomposition. The first approach to use k-core decomposition on visualization is a work by Batagelj *et al* [6], where the adjacency-matrix of some cores has been analysed.

Then, the algorithm presented on 'Drawing the AS graph in 2.5 dimensions' by Baur *et al* [7] is the first one that combines k-cores and visualization of the complete network. This proposal uses directed spectral analysis to draw the maximum core, and then a combination of barycentric and iteratively directed-forces allows them to draw other shells in decreasing order, each in one layer, resulting in a 3D figure. This approach is not able to represent several shell's components.

The next tool is the first version of LaNet-vi [5, 8], which gives a 2D representation of the whole graph, allowing to distinguish between different graphs and also to validate models of complex systems. It is important to stress that 2D representations are more suitable for information visualization than others due to human perception reasons (see [9] and references therein).

The Halfmoon [10] paradigm proposes a visualization where all vertices and edges are drawn, vertices are placed in a half-circle grouped by shell index. Here, transparency is used to show how shells are connected. This visualization cannot distinguish between components on the same core.

Finally, `pajek` [11] is a software package for the analysis and visualization of large networks, which can compute several parameters measured on complex networks (k-core decomposition included).

Visualization is not the unique application of k-core decomposition. It has been used in the analysis of protein interaction networks by Bader and Hogue [12], and in the prediction of protein functions by Altaf-Ul-Amin *et al* [13] and Wuchty and Almaas [14]. Another interesting application was done in the networking area by Gkantsidis *et al* [15] and by Batagelj and Zaversnik [16], who used the k-cores to filter out peripheral autonomous systems for Internet networks. We can add, also in the networking area, a paper of Mahadevan *et al* [17] which depicts the usual parameters for characterizing autonomous systems maps, the k-cores being one of them. The work of Ducheneaut *et al* [18] presents the analysis of a game players' network using the k-core decomposition. Finally, the Complex System SCILAB tool [19], which is a plug-in for SCILAB (http://scilab.org), is capable of computing parameters like those obtained from the k-core decomposition.

3. How LaNet-vi2 works

This tool is an open source project hosted on `sourceforge` [20] running on Linux as well as Windows. It is also included inside Network Workbench (NWB) [21], a toolkit for network research, composed by a framework developed in Java language and tools on binary (pre-compiled) format. There is also available an online [22] version that accepts networks and sends the results by e-mail.

A problem with the first version is that it cannot show the internal organization, which is the central improvement on this new version. Observing that the highest core has diameter 2 in most complex networks, we decomposed it into cliques, drawing them in different sectors of the central circle. This clique decomposition is not unique, but is an interesting way to organize the central core. Then, sons in lower shells can use the fathers' angles to compute their angle, achieving the organization of the whole graph. This method is different from the previous version [5], which grouped vertices by clusters, placing each of them in a random angular sector according to their size. The problem is the high amount of clusters, whose sizes follow a Zipf law resulting in lots of sectors with very different sizes. Now, each vertex chooses its position according to the angles of neighbours in higher shells and also their cluster neighbours. In this way, we eliminate a lot of cross-connections across the centre of each shell, yielding a clearer figure. The rest of the positioning parameters are the same as those in the previous version, that is the radius is computed using neighbours in higher shells and the component information (for more details see [5]).

This new version can draw both kinds of graphs, the old ones using clusters to place shells, or the new approach where the highest core organizes the remaining vertices. In the previous version of LaNet-vi rendering was carried out with PovRay software. As mentioned, compatibility with SVG has recently been added, which is an open language that offers, among other things, an effect of transparency on objects, allowing to smoothly visualize all edges in big networks without hiding vertices. This new version can produce SVG and PovRay files for both visualizations (clusters or cliques), calling the appropriate render to produce final figures in PNG format. The rest of this section is dedicated to presenting new algorithms, to discussing their time complexity, and to giving a formal treatment of the relation between connectivity and the k-shell index.

3.1. New visualization algorithm

The new algorithm keeps on doing the k-core decomposition and components analysis. We will briefly introduce some notations and definitions used through the paper, and then present the algorithm.

An undirected graph is denoted as $G = (V, E)$, where $V = \{v_i\}$ is the set of vertices and $E = \{e_{ij}\}$ is the set of edges, where e_{ij} denotes an edge between v_i and v_j. A vertex degree will be named $d_G(v_i)$, or $d(v_i)$ for short. The highest degree is $d_{\max}$. The number of vertices and edges will be $n = |V|$ and $e = |E|$, respectively. The k-core definition [23] is the following,

Definition 1. *A subgraph $H = (C, E|C)$ induced by the set $C \subseteq V$ is a k-core or a core of order k if and only if the degree of every vertex $v \in C$ induced in H is greater than or equal to k (in symbolic form, this reads $\forall v \in C : d_H(v) \geqslant k$), 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 lower than k, until all vertices in the remaining graph have degree greater than or equal to k. This decomposition can be easily implemented: the algorithm by Batagelj and Zversnik [16] presents a time complexity of order $O(n+e)$ for a general graph. This makes the algorithm very efficient for sparse graphs, where e is in the order of n.

Definition 2. *A vertex i has shell index k, that is $s(i)=k$, if it belongs to the k-core but not to the $(k+1)$-core, and the k-shell S_k is composed by all the vertices whose shell index is k. The maximum value k such that S_k is not empty is denoted $k_{\max}$.*

Notice that the k-core is thus the union of all shells S_s with $s \geqslant k$.

Definition 3. *Every connected component of S_s will be called a cluster Q^s.*

Each shell S_s is thus composed by clusters Q^s_m, such that $S_s = \cup_{1 \leqslant m \leqslant q^s_{\max}} Q^s_m$, where $q^s_{\max}$ is the number of clusters in S_s.

LaNet-vi2 takes the complex network and computes the k-core decomposition, analysing whether each core has multiple components, as is done in the previous version [5]. Then, it picks out the subgraph corresponding to the highest core which is decomposed into cliques.

Once cliques have been built, the central core is decomposed into several circular sectors proportional to the cliques' sizes. For each clique, its vertices are placed uniformly on the periphery of the corresponding sector.

The central core is given a radius $R_{k_{\max}}$ such that its area is proportional to the amount of vertices and their sizes, to assure they do not get overlapped. The remaining space in the image is divided into equidistant rings, one for each core.

Vertices in lower cores are positioned in sectors close to their neighbours of higher cores. In this way the central cliques organize the deployment of the whole graph. The algorithm for positioning each vertex computes a radius ρ and an angle ϕ. The radius is computed by

$$\rho_i = R_{k_{\max}} + \frac{k_{\max} - R_{k_{\max}}}{k_{\max}} f(v_i), \tag{1}$$

where

$$f(v_i) = (1-\epsilon)(k_{\max} - s(i) + 1) + \frac{\epsilon}{|N_{i/\geqslant s(i)}|} \sum_{j \in N_{i/\geqslant s(i)}} (s(j) - s(i)), \tag{2}$$

$N_{i/\geqslant s(i)}$ being the subset of N_i formed by those vertices which are in a shell higher than or equal to v_i.

The angle is a circular average of all its neighbours' angles from higher cores and those from the same core whose angle is already computed (denoted by ϕ^C_j), as

$$\cos\phi_i = \frac{\sum_{j \in N_{i/\geqslant s(i)} \cap \phi^C_j} (s(j) - s(i) + 1) \cos\phi_j}{\sum_{j \in N_{i/\geqslant s(i)} \cap \phi^C_j} (s(j) - s(i) + 1)}, \tag{3}$$

$$\sin\phi_i = \frac{\sum_{j \in N_{i/\geqslant s(i)} \cap \phi^C_j} (s(j) - s(i) + 1) \sin\phi_j}{\sum_{j \in N_{i/\geqslant s(i)} \cap \phi^C_j} (s(j) - s(i) + 1)}. \tag{4}$$

As the main difference between the previous version and the new one is the cliques decomposition, we present the algorithm that we used. It is important to highlight that finding

the maximum clique in a graph is an NP-complete problem [24], and is unfeasible for complex networks, so we used an heuristic that attempts to find the biggest cliques first. We will use the following concepts:

The *neighbourhood* of a vertex $v_i \in V$ is the induced subgraph of G that includes all vertices connected to v_i. It will be denoted by

$$N_i = \left\{ v_j \in V / e_{ij} \in E \right\}. \tag{5}$$

The *set of connections among neighbours* T_i is the set of edges whose vertices belong to v_i's neighbourhood, this is

$$T_i = \left\{ e_{jk} \in E / v_j, v_k \in N_i \right\}. \tag{6}$$

The *clustering coefficient* $cc(v_i)$ for $v_i \in V$ is the relation between the amount of connections among v_i's neighbours $w_1, w_2, \ldots, w_{N_i}$ and the maximum possible connections among them, specifically $\frac{N_i\,(N_i-1)}{2}$. In effect,

$$cc(v_i) = \frac{2\,|T_i|}{|N_i|\,(|N_i|-1)}. \tag{7}$$

The *common neighbourhood* C_{ij} is the induced graph of G formed by every vertex adjacent to both v_i and v_j.

A useful result for our algorithm is that the amount of neighbours' connections $|T_i|$ can be computed having C_{ij} for every v_j adjacent to v_i, in this way: for v_j, v_k adjacent to v_i, if they are connected to each other, then $v_j \in C_{ik}$ and $v_k \in C_{ij}$, so

$$|T_i| = \sum_{j/v_j \in N_i} \frac{|C_{ij}|}{2}. \tag{8}$$

The detailed algorithm is described in table 1. First of all, we isolate the central core, which is a subgraph $G = (V, E)$ of the whole network. The objective consists of finding the main cliques in G. As vertices in the biggest clique should have neighbours with a great number of connections among them, we order V by this parameter, $|T_i|$. We take then the vertex with more connected neighbours, v_j, to form a subset V_1 and proceed to add neighbours subject to the constraint that they are connected to every vertex in this subset, until no more neighbours can be added. The first ones to be added are those that have more shared neighbours with v_j, because they have more possibilities of being part of a big clique. Once no more neighbours can be added the clique is closed and a new one is started taking the first of the remaining vertices. This procedure continues until there are no more vertices left. Finally, a partition composed by cliques $\{V_1, V_2, \ldots, V_n\}$ will be obtained, which are collectively exhaustive and mutually exclusive with respect to V.

As can be seen, the algorithm makes extensive use of the coefficients $|C_{ij}|$, so they will be computed at the beginning.

3.2. Complexity analysis

Naming $n_{k_{\max}}$ the number of vertices on the maximum core $k_{\max}$, and $d_{k_{\max}}$ its maximum degree, we begin the analysis. On lines 7–10 of algorithm 1 (first step) every pair of neighbours v_j, v_k of the same vertex v_i is taken, and as v_i is a neighbour of both, $|C_{jk}|$ is incremented. There is no need to build the set C_{jk}, we only have to know its size. The condition $k > j$ is necessary so that the pair does not appear twice, in different order. There are $d(v_i)\,(d(v_i)-1)/2$ pairs of

Table 1. Algorithm for decomposing the central core into cliques.

Algorithm 1

```
1  input: a graph G = (V, E)
2  output: set of cliques V_1, V_2, ..., V_n that cover V
3  initialise |C_ij| (common neighbourhoods) in 0 for every v_i, v_j ∈ V
4  initialise |T_i| in 0 for every v_i ∈ V
5  initialise n = 0
6
7  for each vertex v_i ∈ V do
8     for each neighbour v_j ∈ N_i do
9        for each neighbour v_k ∈ N_i, k > j do
10          increment |C_jk|
11
12 for each vertex v_i ∈ V do
13    compute connections among neighbours |T_i| using equation (8)
14
15 order V by decreasing |T_i| generating W
16
17 while W is not empty do
18    start clique V_n
19    v_f = {first vertex in W}
20    insert first vertex v_f into clique V_n and remove it from W
21    generate N with all vertices adjacent to v_f that are in W (those without clique yet)
22    order N by decreasing |C_fj|
23    for each vertex v_j ∈ N (in order) do
24       if it's adjacent to every vertex in V_n (V_n ⊂ N_j) then
25          add v_j to clique V_n
26       remove v_j from W
27 n = n + 1
28
29 return {V_i}
```

neighbours for v_i, which is less than $d^2_{k_{\max}}$, so the number of operations for this step is bounded by $n_{k_{\max}} d^2_{k_{\max}}$.

On lines 12–13 (second step) every $|T_i|$ is obtained by adding the $|C_{ij}|$'s for every neighbour v_j, as shown in equation (8). There is one operation per neighbour, so $d(v_i)$ operations for v_i and $2e_{k_{\max}}$ for the whole core (denoting by $e_{k_{\max}}$ the number of edges in the highest core). An upper bound for $e_{k_{\max}}$ is $n_{k_{\max}} d_{k_{\max}}$, then this step is $O(n_{k_{\max}} d_{k_{\max}})$.

Line 15 (third step) orders vertices in V by $|T_i|$. The number of operations here is $O(n_{k_{\max}} \log n_{k_{\max}})$.

Finally, lines 17–27 (fourth step) perform the cliques decomposition. The 'while' loop is executed once per clique. Starting the clique (line 18) and inserting the first vertex (line 20) take constant time. Generating N (line 21) is $O(d(v_f))$ which is bounded by $O(d_{k_{\max}})$. Ordering (22) is $O(d_{k_{\max}} \log d_{k_{\max}})$. The 'for' loop in line 23 is run $d(v_f)$ times. The first time, line 24 implies one verification because there is just one vertex, v_f, in N_j. Second time one or two verifications are made, depending on the result of the previous one. The worst case for this step

is that of a complete graph, in which $1+2+\cdots+d(v_f)$ checks have to be made, and a bound for this is $d^2_{k_{\max}}$. Lines 25–27 take constant time. Then, the complexity of the 'while' loop is $O(n_{\text{cliques}}\, d^2_{k_{\max}})$.

The two steps that govern time are the first and the last ones, and taking into account that the number of cliques n_{cliques} is lower than $n_{k_{\max}}$, we have $O(n(k_{\max})\, d^2_{k_{\max}})$. Now, $d_{k_{\max}}$ is bounded by $d_{\max}$, and $n_{k_{\max}}$ can be bounded also by $d_{\max}$ for most real complex networks (as we verified; for Internet maps see [8]). Therefore, the whole algorithm 1 complexity is

$$O(d^3_{\max}). \tag{9}$$

Now, we need to bound $d_{\max}$ in terms of n. Complex networks usually have a heavy tailed degree distribution [3], which can be bounded by a power law

$$n(d) = A \cdot d^{-B}, \tag{10}$$

where $n(d)$ is the number of vertices with degree d. This is because degree distributions are either a power law or Weibull distribution, with an exponential cutoff (due to the finite size of networks). Some early work on Internet topology [25] reports exponents in the range $2.1 \leqslant B \leqslant 2.5$ for Internet topology, and in general complex networks have exponents in the range $2 \leqslant B \leqslant 3$.

From equation (10) we obtain

$$\frac{n(1)}{n(d)} \propto d^B, \tag{11}$$

from where

$$d \propto \left(\frac{n(1)}{n(d)}\right)^{1/B} < (n(1))^{1/B} < n^{1/B}. \tag{12}$$

Therefore, from equations (9) and (12) the total time complexity to perform the organization of the highest core is $O(n^{3/B})$, considering a degree distribution bounded by a power law with exponent B of equation (10).

3.3. *Connectivity analysis*

Let us introduce the following:

Definition 4. *Considering a graph G and its k-core decomposition,*

(i) given k, two vertices in the k-core are core-connected if there are k edge-disjoint paths between them; and

(ii) the graph is core-connected if given two vertices in a k-core, they are core-connected.

In other words, taking the minimum shell index of a pair of vertices gives a lower bound of edge-connectivity when the graph is k-core-connected. Notice that each core's graph has to be connected, i.e. to have only one component; otherwise it is not possible to find paths.

LaNet-vi2 can evaluate if a graph is k-core-connected for simple graphs having only one component for all k-cores. The Internet is an example of this kind of network, as well as some biological networks [13, 14], the international flights network [26, 27], etc. Examples of other networks that do not belong to this category are scientific collaborations [28] or the World Wide Web [8].

The connectivity is important in terms of what the network represents. For example, considering the Internet, the connectivity expresses the number of independent paths (more precisely edges independent paths for edge-connectivity) between a couple of routers or autonomous systems. In this case, connectivity gives a measure of robustness, if you are considering the probability of isolation; or a measure of the probability to find a path with an specified quality of service (QoS).

The following theorem states conditions for k-edge-connectivity in a graph with minimum degree k. Let $G = (V, E)$ be a simple graph (i.e. without loops, without multiple edges). Let $V_1 \subset V$, $V_2 \doteq V \setminus V_1$ (the complement of V_1 in V), and set $G_1 = (V_1, E|V_1)$, $G_2 = (V_2, E|V_2)$. Let r and r_{G_1} denote the usual distance in G and G_1, respectively. A cut between two sets of vertices A and B is denoted by $[A, B]$. We assume in the following that V_1 and V_2 are nonvoid, and define

$$\delta(x, y) \doteq \min\{r_{G_1}(x, y), r_G(x, V_2) + r_G(y, V_2)\},\ x, y \in V_1,$$

$$\delta(x, y) = \delta(y, x) \doteq r_G(x, V_2),\ x \in V,\ y \in V_2.$$

If $x \in V$ and $A \subset V$, we set $\delta(x, A) \doteq \min_{a \in A} \delta(x, a)$.

We shall also use the notations

$$\partial V_1 \doteq \{x \in V_1 : r_G(x, V_2) = 1\},$$

$$V_1^0 \doteq V_1 \setminus \partial V_1.$$

Theorem 1. *Assume that*

(i) $d_G(x) \geqslant k,\ x \in V,$

(ii) G_2 *is k-edge connected,*

(iii) $\max_{x,y \in V} \delta(x, y) \leqslant 2,$

(iv) One of the following:

(a) $\partial V_1 \geqslant k$ or

(b) $\sum_{x \in V_1} \min\{|[x, V_1^0]|, |[x, V_2]|\} \geqslant k.$

Then G is k-edge-connected.

The proof is presented in [29]. This theorem is related to a well-known theorem of Plesník (see [30], theorem 6), which states that in a simple graph of diameter 2 the edge connectivity is equal to the minimum degree. In fact, under hypothesis (*iv*)*a*, our theorem follows from Plesník's result by 'contracting' V_2 to a vertex. The resulting graph is not simple, but we can avoid this problem by replacing all edges joining one vertex to the contracted one by only one edge.

Therefore, we can apply theorem 1 recursively to all shells of a graph, starting from the highest one, in order to determine when a couple of vertices are core-connected (see definition 4(*i*)). For instance, taking G to be the highest core (i.e., $k_{\max}$-core), and V_2 as a vertex, if G has diameter 2, it is $k_{\max}$-edge-connected. Then, taking each cluster (from those with shell-index $k_{\max}$ to $k_{\min}$), we can apply theorem 1 and thus prove that this cluster is k-edge-connected if it verifies the hypothesis.

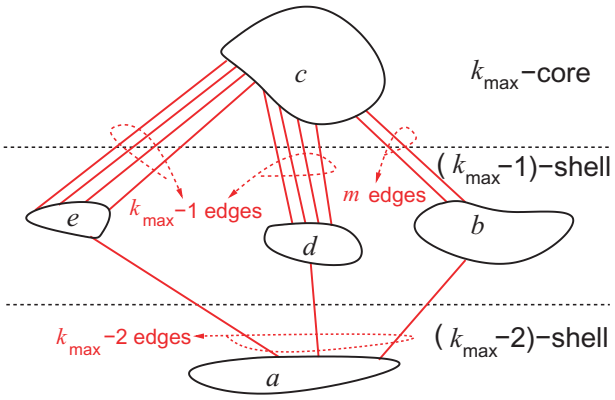

Figure 1. The cluster a is connected through cluster e, d and b to the highest core. Here cluster a has a $k_{\max} - 2$ connectivity, but cluster b has only $m < k_{\max} - 1$ connectivity.

LaNet-vi2 begins from the highest core checking that it has diameter 2. Then, it takes one by one all clusters of shell $k - 1$, where k is the previous shell (first time the highest one), and verifies hypothesis (*iii*) and (*iv*) of theorem 1. Notice that hypothesis (*i*) is assured by the definition of the shell, and hypothesis (*ii*) is guaranteed by the previous shell. However, a certain cluster in the s-shell can have a lower connectivity than s, while their sons (clusters connected to it in lower shells) can have a good connectivity (equal to or larger than its shell index). This case is illustrated in figure 1.

The time complexity is not modified by the verification of hypothesis (*iii*) and (*iv*) of theorem 1. Computing hypothesis (*iii*) requires a breadth-search on the cluster plus edges connecting towards higher shells, which has a complexity of $O(e_c + e_h)$, e_c being the number of edges in the cluster and e_h the number of edges towards higher shells. Testing hypothesis (*iv*) is shorter and requires only $O(e_h)$. This process is done from higher to lower shells, and each edge of the graph is visited a constant number of times. Therefore, the total complexity of this step is $O(e)$, e being the total number of edges, which is the same as for the k-core decomposition.

The picture obtained with Lanet-vi2 picks out the vertices in a k-shell that do not satisfy core-connectivity by drawing them in white or black for colour pictures, or using a square instead of a circle for greyscale images. Therefore, the set of vertices in colour (or greyscale) forms a k-core-connected graph.

3.4. Adding labels

The functionality of visualizing the vertices' names or numbers has been added. We implemented a mesh to control the overlapping of names, in the following way: the image of $W \times H$ pixels is divided into cells of 10×12 pixels, which is the monospaced font size. After each vertex is drawn, the closest cells to it in the mesh are analysed (left-side, right-side, up and right, up and left, down and right and down and left), considering a number of consecutive cells equal to the name string size, and all the cells bordering them. If the number of overlaps (cells that are already occupied by text) is lower than a certain amount (which we defined as one to avoid superpositions almost completely), then the name is stamped and all these consecutive cells on the mesh are marked as occupied. This method does not assure that all the names are stamped, but the stamped ones will certainly be legible. If all names are necessary, using a

bigger resolution will always help, because for greater resolutions the text uses less space and does not cover the vertices.

The user can also select which names to show, by only including in the names file those ones, and omitting the names for the rest of the vertices.

4. Complex networks analysis using LaNet-vi

This section is dedicated to analysing a complex network using real data. We focus on the autonomous systems Internet map (or AS map for short), which is a network composed by autonomous systems as vertices and their peer-relationships as edges. ASes are networks whose administration is centralized. They are used either for connecting institutions or customers to the Internet (called stub ASes), or to interconnect stub ASes and interconnection ASes. The AS map is the top hierarchy view of the Internet, where each AS is identified by a number from 1 to 65 535. One of the most important reasons for such organization is the freedom to apply policies to interconnect different ASes, reflecting commercial agreements. It is important to note that even if these relationships can be directed, it is needed to treat them as undirected ones to apply the k-core decomposition as it is explained here. We are not interested in searching 'commercial relationships' but 'possible connections' because all communication links are bidirectional (i.e. on any link data flows in both directions).

The analysis presented in this paper uses the three following databases: The first one is the Oregon Route Views [31], which collects the BGP [32] routing tables of several BGP routers having a great number of peer connections. This method allows knowledge of the public routes on the Internet, but not all routes are public due to the policies of each AS. The second source is CAIDA [33], that uses a couple of tens of *skitter* probes (a tool based on `traceroute` and `ping`) to survey Internet maps [34]. These probes discover paths by sending packets to Internet addresses; then paths can be merged to obtain a map at different levels, here we focus only on the AS level. The main difference with respect to the first technique, is that information is gathered from real paths, i.e. a path traversed by data packets. This allows us to obtain routes that are not declared on public BGP tables. The third source is the DIMES project [35], that uses a distributed system (based on `traceroute` and `ping`) composed by several thousands of probes to measure Internet maps [36], also using real paths. The main difference between this project and the former is the number of probes, which is a thousand times greater than CAIDA and yields a more detailed map [37]; besides the techniques are different (see [36] for more details).

The new layering algorithm of LaNet-vi maintains some features from the previous version (for these ones see [5]), others have changed (marked as $*$), and there are some new ones (marked with $^{\#}$).

Shells width. Vertices within the same shell are initially placed on a circle, but then they are moved to the centre according to their neighbours on higher shells, producing a thick ring. Its thickness gives an idea of how high their neighbours are, and is controlled by the ϵ parameter [5].

Degree-shell index correlation. Degree and shell-index are centrality measures, so it is very important to study their correlation. Degree is represented by the vertex size (in a logarithmic scale), and shell-index by colours (going from blue for the lowest shell, to red for the highest one); both scales and their maximum values are displayed.

Disconnected components. A k-core can have more than one component, which is highlighted by drawing each component in separated circles, each of them proportional to its component size. A γ parameter controls the diameter and δ controls the distance between components [5].

Shell clusters*. This feature is only available when the cliques decomposition is deactivated. Clusters of vertices having the same shell index are drawn in the same angular sector.

Edges*. Due to layout changes in this new version, edges are placed in a different way, giving a better image. For PovRay-based figures, we continue using a random fraction of edges, because they are solid (not transparent) and drawing all edges would result in a non intelligible figure. Instead, for SVG figures we draw all edges taking advantage of transparency. In both cases, the edges show how shells are connected. Each extreme of an edge is drawn with the same colour as the vertex at the opposite side.

Highest core#. It is displayed as a collection of cliques, assigning a different circular sector of the central core for each one, depending on its size. This feature allows us to observe the proximity of vertices. Since the remaining vertices choose their angle using this information, it is possible to see when a clique or a group of them is more connected to the whole network.

Core-connectivity#. Setting this option, LaNet-vi2 corroborates whether a vertex verifies all conditions for core-connectivity, highlighting those which do not. This property is useful on networks with only one component, but if it is used on other graphs, LaNet-vi2 will apply it only to the greatest component. For example, due to exploration issues it is possible to obtain a network with several components for the Internet, but the only meaningful one is the greatest (the others are too small).

Labels#. In this version, labels scaled to the size of the figure are added and superimposed labels are avoided, i.e. when a label occupies a place already used it is not displayed. The order in which labels are placed is from the highest k-shell to the lowest one, and is random inside a k-shell. Through an input file the user can provide the names of all vertices or just those he is interested in. Anyway, in order to have all labels displayed, it is essential to generate a high resolution image.

Multigraphs#. Networks with several edges between vertices are called multigraphs, and we can compute the k-core decomposition taking these edges into account. For this option, we cannot assure the core-connectivity, so it is set off.

Zoom#. It is possible to draw a portion of the whole image, in order to focus on an interesting part in high resolution.

Detailed information#: This version gives (under user command-line request) a list of vertices by core, the vertices in each clique for the highest core, and the list of vertices that are not core-connected.

In the presented case, AS Internet maps, we cannot show all features LaNet-vi2 is capable of, but we illustrate how to use our tool to analyse these complex networks.

The main characteristics of AS maps are the following (several are obtained from [5]):

- lower shells are mostly connected to the highest one, shown by the predominant red colour (that of the k_{max}-core) of edges at the external radius and violet colour near the central circle;

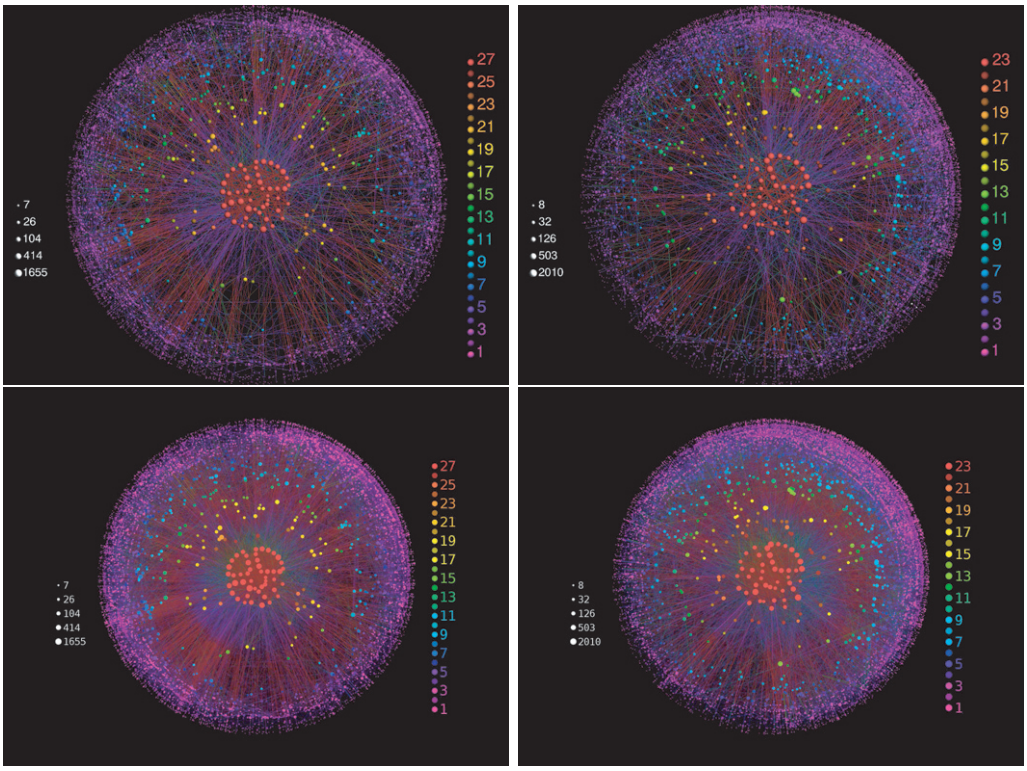

Figure 2. AS obtained from Route Views project [31]. Maps in April 2005 (first column) and February 2008 (second column). Upper figures are rendered with PovRay, bottom ones with SVG.

- high degree-shell index correlation, which means most connected vertices are in the most robust subgraph (or the one with most routing capabilities), i.e. there are no star-like subgraphs;
- distribution of lower shells is sparse (e.g. thick rings), meaning that vertices in these shells are widely connected to different higher shells;
- several cliques exist in the $k_{\max}$-core, where normally one is the biggest;
- they are k-core-connected.

These characteristics identify AS maps from others. It is important to stress that the first and second characteristics together imply a disassortative [38] behaviour, because low degree vertices are connected to high degree ones. For instance, there are other Internet maps at more detailed granularity: Internet Routers (IR), which have different characteristics. As an example, IR maps have shells mostly interconnected without a particular hierarchy, and no degree-shell index correlation [5], yielding a quasi planar (neither assortative nor disassortative) nearest-neighbour distribution [4].

This networks classification, which we call 'network fingerprint', could also be done by the previous LaNet-vi version, but this new version is able to identify also the maps source. We call this feature the 'network iris-print'.

We present images from three AS map sources: Oregon Route Views in figure 2, CAIDA in figure 3 and DIMES in figure 4, all graphics are in high resolution (use the zoom to see details).

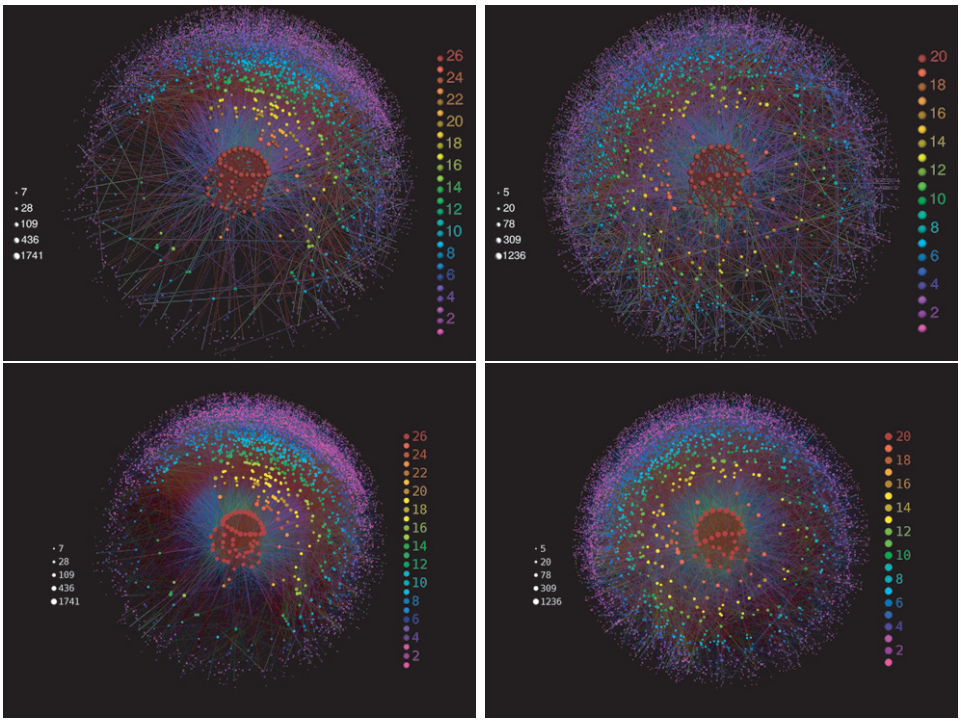

Figure 3. AS obtained from CAIDA project [33]. Maps in April 2005 (first column) and January 2008 (second column). Upper figures are rendered with PovRay, bottom ones with SVG.

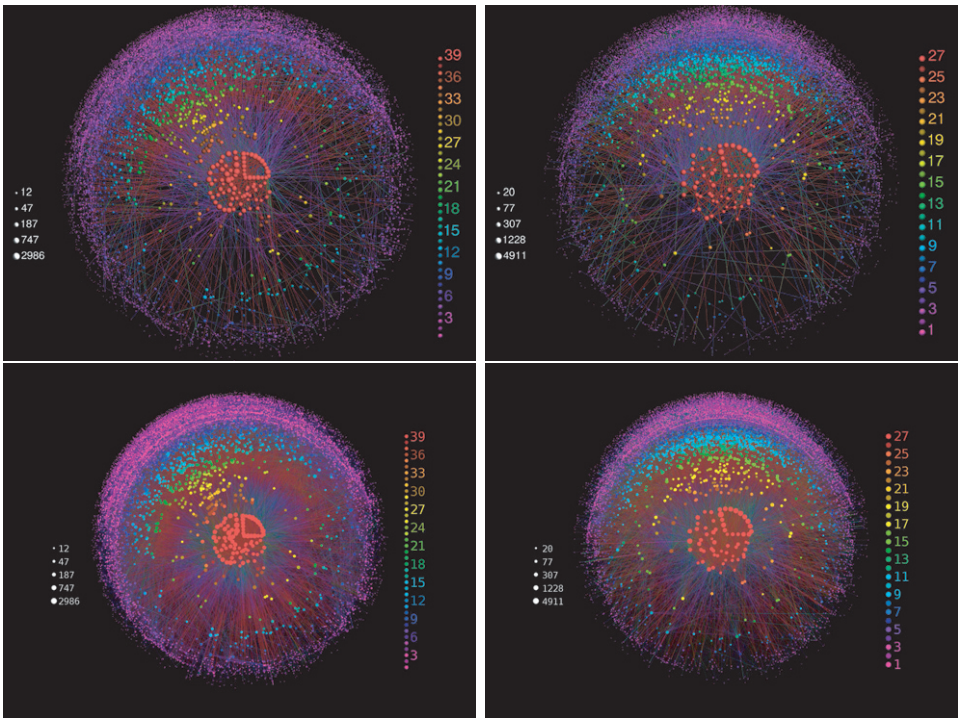

Figure 4. AS obtained from DIMES project [35]. Maps in April 2005 (first column) and September 2007 (second column). Upper figures are rendered with PovRay, bottom ones with SVG.

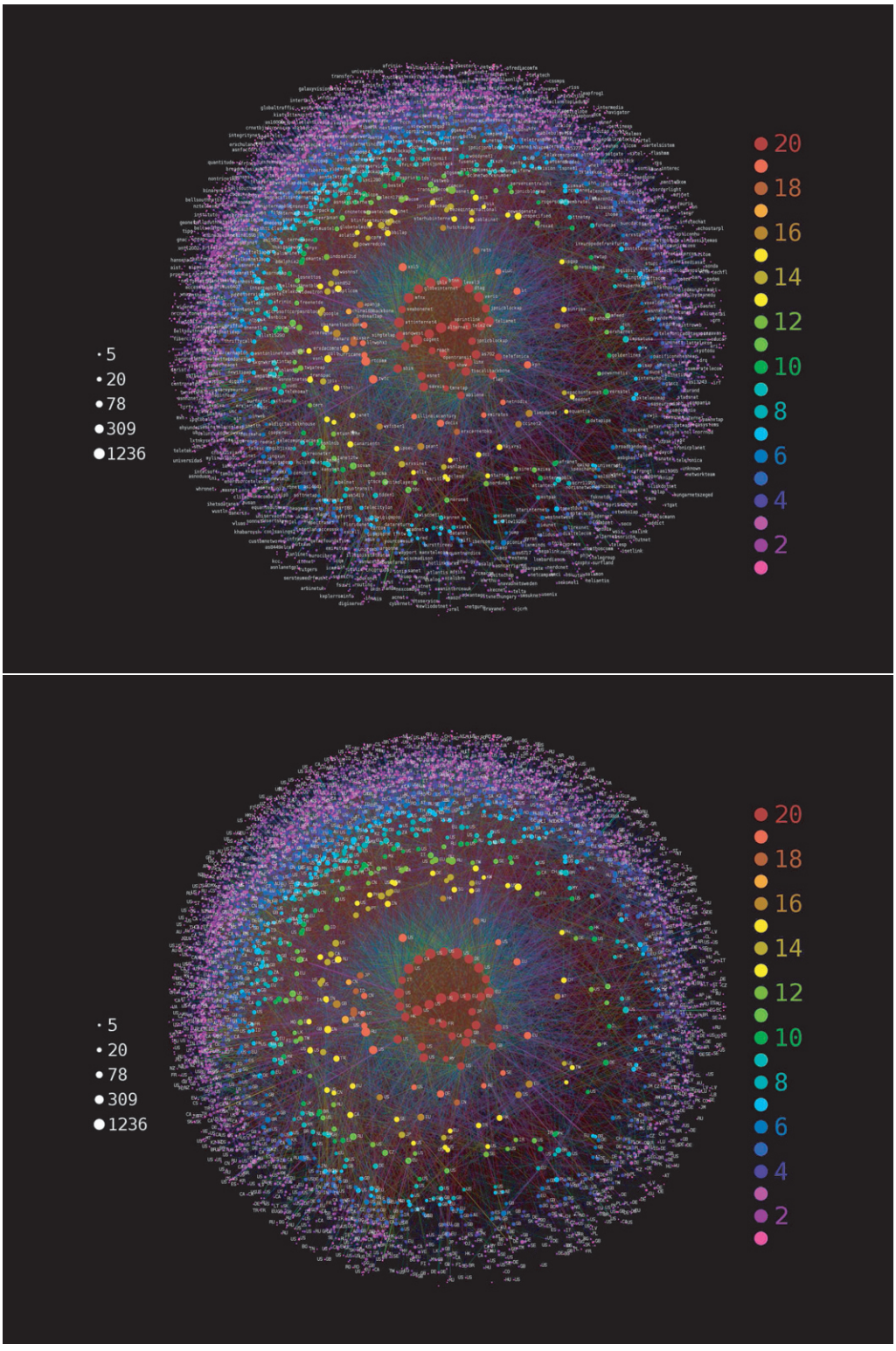

Figure 5. AS obtained from CAIDA project [33] in January 2008. Upper figure has AS names, and bottom figure has two-letter country codes (http://www.iana.org/domains/root/db/).

As a first observation, Oregon Route Views maps have several cliques on their highest core and edges arise from all angles. There is a little concentration of middle shells on first and second quadrant, but not a significant amount. It is interesting to see that even though the maximum degree has increased, the number of shells has decreased from 2005 to 2008. CAIDA maps show a big clique that concentrates vertices of all shells on the first quadrant in 2005 and first and second quadrant in 2008. We appreciate that degree and number of shells are decreasing, justified by the number of probes: 21 in 2005 and 10 in 2008. DIMES maps present one big clique (about a half of the first quadrant) and several small cliques, where most vertices are concentrated in the first quadrant for both years. It is worth remarking that the number of shells is also reduced since the maximum degree is increased.

We observe that CAIDA and DIMES are the most similar, because both exploration methods gather information from real paths, whereas Oregon Route Views uses the public BGP routing tables. The most important difference between CAIDA and DIMES is the number of probes, because DIMES uses several thousand.

Finally, we present visualizations with labels in figure 5. We can observe shortcuts of ASes' names on the top graph, and their origin countries on the bottom one. Notice there are no geographic groups (e.g. we cannot identify Asian ASes in a specific angular sector), mainly due to the highest core composition (most of the ASes are located in the US) and because the larger ASes span over several countries.

5. Conclusions

In this paper, we have presented the second version of LaNet-vi, a Large Network visualization tool, capable not only of identifying different types of networks like AS or IR maps ('network fingerprint'), but also of recognizing its source, providing a useful tool to see at a glance connectivity properties ('network iris-print'). We also performed a core-connectivity analysis, and visualized through LaNet-vi the vertices that do not verify this property. In communication networks, core-connectivity can be interpreted as robustness or QoS capability.

Moreover, LaNet-vi2 is now capable of producing SVG figures, which combined with edge transparency yields a precise visualization. As a final feature, our tool can now display labels, deal with multigraphs and draw a portion of the picture.

Finally, we envisage visualization of weighted graphs as future work, including these edge weights in the graphical algorithm. This might help to interpret the properties of new networks and to better understand their organization.

Acknowledgments

We acknowledge the support of the Facultad de Ingeniería, Universidad de Buenos Aires (Exp.909.325/2007 FIUBA). We also wish to acknowledge Dmitri Krioukov (CAIDA) for help with `skitter` data and AS names; and Fabiana Vega (FIUBA) for her helpful suggestions. Finally, MB acknowledges the support of Telefónica de Argentina (fellowship Res.Dec.2597/07 FIUBA).

References

[1] Albert R and Barabási A L 2000 Statistical mechanics of complex networks *Rev. Mod. Phys.* **74** 47

[2] Amaral L A N, Scala A, Barthélemy M and Stanley H E 2000 Classes of small world networks *Proc. Natl Acad. Sci. USA* **97** 11149–52

[3] Dorogovtsev S N and Mendes J F F 2003 *Evolution of Networks: From Biological Nets to the Internet and WWW* (Oxford: Oxford University Press)

[4] Pastor-Satorras R and Vespignani A 2004 *Evolution and Structure of the Internet: A Statistical Physics Approach* (Cambridge: Cambridge University Press)

[5] Alvarez-Hamelin J I, Dall'Asta L, Barrat A and Vespignani A 2006 Large scale networks fingerprinting and visualization using the *k*-core decomposition *Advances in Neural Information Processing Systems 18* ed Y Weiss, B Schölkopf and J Platt (Cambridge, MA: MIT Press) pp 41–50 http://books.nips.cc/papers/files/nips18/NIPS2005_0999.pdf

[6] Batagelj V, Mrvar A and Zaversnik M 1999 Partitioning approach to visualization of large networks *Graph Drawing '99, Castle Stirin, Czech Republic, LNCS 1731* pp 90–8 doi:10.1007/3-540-46648-7_9

[7] Baur M, Brandes U, Gaertler M and Wagner D 2004 Drawing the AS graph in 2.5 dimensions *12th Int. Symp.on Graph Drawing* (Berlin: Springer) p 43–8 doi:10.1007/b105810

[8] Alvarez-Hamelin J I, Dall'Asta L, Barrat A and Vespignani A 2005 *k*-core decomposition: a tool for the visualization of large scale networks arXiv:cs.NI/0504107

[9] Ben Shneiderman 2003 Why not make interfaces better than 3d reality? *IEEE Comput. Graph. Appl.* **23** 12–5

[10] Alvarez-Hamelin J I, Gaertler M, Görke R and Wagner D 2005 Halfmoon—a new paradigm for complex network visualization *Technical Report* TR-2005-29, Faculty of Informatics, Universität Karlsruhe (TH) http://digbib.ubka.uni-karlsruhe.de/volltexte/1000004127

[11] Batagelj V and Mrvar A, pajek: a large network analysis software. Online at http://vlado.fmf.uni-lj.si/pub/networks/pajek/

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

[13] Altaf-Ul-Amin Md *et al* 2003 Prediction of protein functions based on *K*-cores of protein-protein interaction networks and amino acid sequences *Genome Inform.* **14** 498–9 http://www.jsbi.org/modules/journal1/index.php/GI14.html

[14] Wuchty S and Almaas E 2005 Peeling the yeast protein network *Proteomics* **5** 444–9

[15] Gkantsidis C, Mihail M and Zegura E W 2003 Spectral analysis of internet topologies *INFOCOM* http://www.ieee-infocom.org/2003/papers/09_04.PDF

[16] Batagelj V and Zaversnik M 2003 An $O(m)$ algorithm for cores decomposition of networks arXiv:cs.DS/0310049

[17] Mahadevan P, Krioukov D, Fomenkov M, Huffaker B, Dimitripoulos X, Claffy Kc and Vahdat A 2005 Lessons from three views of the internet topology arXiv:cs.NI/0508033

[18] Ducheneaut N, Yee N, Nickell E and Moore R J 2006 Alone together?: exploring the social dynamics of massively multiplayer online games *Conf. on Human Factors in Computing Systems* pp 407–16 doi:10.1145/1124772.1124834

[19] Orlicky J I, Estrada V and Alvarez-Hamelin J I. Complex Systems SCILAB Toolbox. Online at http://sourceforge.net/projects/complex-sys-sci/

[20] **La**rge **Net**work **vi**sualization tool. http://sourceforge.net/projects/lanet-vi

[21] NWB Team 2006–2008 Network Workbench Tool. Indiana University and Northeastern University. Online at http://nwb.slis.indiana.edu/

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

[23] Batagelj V and Zaversnik M 2002 Generalized cores arXiv:cs.DS/0202039

[24] Cook S A 1971 The complexity of theorem proving procedures *Third Annu. ACM Symp. on Theory of Computing* pp 121–58 doi:10.1145/800157.805047

[25] Faloutsos M, Faloutsos P and Faloutsos C 1999 On power-law relationships of the internet topology *SIGCOMM* pp 251–62 http://www.sigcomm.org/sigcomm99/papers/session7-2.html
[26] Guimerá R and Amaral L A N 2004 Modeling the world-wide airport network *Eur. Phys. J.* B **38** 381–5
[27] Barrat A, Barthélemy M, Pastor-Satorras R and Vespignani A 2004 The architecture of complex weighted networks *Proc. Natl Acad. Sci. USA* **101** 3747
[28] Barabási A L, Jeong H, Néda Z, Ravasz E, Schubert A and Vicsek T 2002 Evolution of the social network of scientific collaborations *Physica* A **311** 590–614
[29] Alvarez-Hamelin J I and Busch J R 2008 Edge connectivity in graphs: an expansion theorem arXiv: 0803.3057v1 [math.GM]
[30] Plesník J 1975 Critical graphs of a given diameter *Acta Fac. Rerum Natur. Univ. Comenian. Math.* **30** 71–93
[31] University of Oregon Route Views Project. Online at http://www.routeviews.org/
[32] Internet Engineering Task Force. RFC 4271: A Border Gateway Protocol 4. Online at http://www.rfc-archive.org/getrfc.php?rfc=4271, http://www.bgp4.as/
[33] Cooperative Association for Internet Data Analysis, Router-Level Topology Measurements. Online at http://www.caida.org/tools/measurement/skitter/
[34] Huffaker B, Fomenkov M, Moore D and Claffy K 2001 Macroscopic analyses of the infrastructure measurement and visualization of Internet connectivity and performance *Passive and Active Measurement* (*PAM*) *Conf.*
[35] Distributed Internet MEasurements and Simulations. Online at http://www.netdimes.org
[36] Shavitt Y and Shir E 2005 Dimes: let the internet measure itself arXiv:cs.NI/0506099
[37] Dall'Asta L, Alvarez-Hamelin J I, Barrat A, Vázquez A and Vespignani A 2006 Exploring networks with traceroute-like probes: theory and simulations *Theor. Comput. Sci.* **355** 6–24
[38] Newman M E J 2002 Assortative mixing in networks *Phys. Rev. Lett.* **89** 208701