First publ. in: IEEE transactions on visualization and computer graphics 13 (2007), 6

Visual Analysis of Network Traffic for Resource Planning, Interactive Monitoring, and Interpretation of Security Threats

Florian Mansmann, Daniel A. Keim, Stephen C. North, Brian Rexroad, and Daniel Sheleheda

Abstract— The Internet has become a wild place: malicious code is spread on personal computers across the world, deploying botnets ready to attack the network infrastructure. The vast number of security incidents and other anomalies overwhelms attempts at manual analysis, especially when monitoring service provider backbone links.

We present an approach to interactive visualization with a case study indicating that interactive visualization can be applied to gain more insight into these large data sets. We superimpose a hierarchy on IP address space, and study the suitability of Treemap variants for each hierarchy level. Because viewing the whole IP hierarchy at once is not practical for most tasks, we evaluate layout stability when eliding large parts of the hierarchy, while maintaining the visibility and ordering of the data of interest.

Index Terms—Information visualization, network security, network monitoring, treemap

1 Introduction

Without firewall protection, computers on the Internet are continually scanned by malicious hosts for potential vulnerabilities. Internet service providers combat this and protect their customers by identifying and blocking emerging attacks. In practice, large networks with hundreds of thousands of hosts are monitored by integrating the logs from gateway routers, firewalls, and intrusion detection systems. This data is analyzed with statistical and signature-based methods to detect changes and anomalies that point to emerging attacks. Due to well-known trends, networks are not only already huge but are still growing, which means coping with both more legitimate traffic and more malicious attacks. There are too many potential security incidents and other anomalies to deal with each individually, so more sophisticated methods are needed. Our objective is to show how visual analysis can foster better insight in this area.

The main focus of this study is to propose interactive Hierarchical Network Maps (HNMaps) that support a mental model of global Internet measurements. HNMaps are applied to depict a hierarchy of 7 continents, 190 countries, 23054 autonomous systems and 197427 IP prefixes. We adopt the Treemap approach [13]: each node in the hierarchy is drawn as a box placed inside its parent. Node sizes are proportional to the number of items contained. Popup and fixed text labels are used to identify nodes. An adjustable color scale also encodes an attribute under inspection.

We provide filtering and zooming to interactively explore this hierarchy. These operations may force re-layout of the part of the hierarchy under investigation. So an important aspect of this study is an assessment of the layout stability of Treemap algorithms under large adjustments in the display set. We compare two layout algorithms for ordered data, *split-by-middle* and *strip Treemaps*, with respect to squareness, locality preservation and running time. A case study conducted jointly with security experts is presented as evidence of the suitability of HNMaps for visually analyzing security-related data. To the best of our knowledge, this is the first proposed method to visualize large-scale network data aggregated by prefix, autonomous system, country, and continent.

This paper is organized as follows: we introduce relevant networking and data warehouse terminology and review related work. Next, characteristics of different levels of the hierarchy and their implications on visualization are examined. We review several Treemap layout variants, and formally evaluate HistoMap 1D and Strip Treemaps. To demonstrate the usefulness of the proposed visualization to network resource monitoring, planning, and security, we present three small case studies, before assessing the overall contribution.

2 A data warehouse approach for network monitoring and security

We begin by defining several networking terms.

- An *IP address* is a 32-bit number (in IPv4) which uniquely identifies a host interface in the Internet. For example, 134.34.240.69 is the IP address of a University of Konstanz web server in dot-decimal notation.
- A *prefix* is a range of IP addresses, and corresponds to one or more networks [10]. For instance, 134.34.0.0/16 specifies the 65536 IP addresses assigned to the University of Konstanz, Germany. Prefixes underlie Classless Inter-Domain Routing, described elsewhere [1].
- An *Autonomous System* (AS) is a connected group of one or more IP prefixes (networks) run by one or more network operators, and has a single and clearly defined routing policy [10]. They are indexed by a 16-bit *Autonomous System Number* (ASN). Usually, an AS belongs to a local, regional or global service provider, or to a large customer that subscribes to multiple IP service providers.

Interactively visualizing large-scale network and security data imposes high demands on an underlying database system. Data warehouses are often employed to integrate data from multiple sources and formats. The *OLAP* (On-Line Analytical Processing) architecture [6] is a good option for this. The underlying *multidimensional data model* [17] allows mapping of detailed transactional data, denoted *facts*, arranged in a multidimensional space or *hypercube*. Numerical values under analysis, termed *measures*, are characterized by descriptive values drawn from a number of *dimensions*. The values in any single dimension can be further organized in a containment-type hierarchy, thus, analytical queries can efficiently compute measure aggregates at multiple levels of detail. The test data sets in this study have the dimensions *IP address* and *time*. We define hierarchies on these dimensions that are appropriate for the proposed application.

- *Florian Mansmann and Daniel A. Keim are with University of Konstanz, Germany, E-mail: {mansmann, keim}@inf.uni-konstanz.de.*
- *Stephen North, Brian Rexroad, and Daniel Sheleheda are with AT&T, E-mail: north@research.att.com, {dsheleheda, brexroad}@att.com.*

- *IP addresses* are grouped by *IP prefix* → *autonomous system* → *country* → *continent* and we thus obtain the following hierarchy:

```
7 continents
  190 countries
    23054 autonomous systems
      197427 prefixes
```

- *Timestamps* are aggregated by *millisecond* → *second* → *minute* → *hour* → *day* → *month* → *year*.

A global map from IP prefixes to origin (primary) AS numbers and names can be a bit subtle to obtain. Though a local map can be extracted from any border gateway router, because of route aggregation, it is unlikely to list many worldwide origin ASes. Therefore, maps must be derived from data collected at multiple vantage points in the Internet. This presents problems of consistency and completeness, especially in the presence of intentional efforts to spoof AS identifiers. This problem has been well studied and there are effective heuristics based on dynamic programming [15] yielding public prefix-to-ASN tables. Unfortunately, the available tables date from 2004[1] and we therefore reverted to the data we extracted from a single routing table in September 2006 [1]. Additionally, to determine the country of an autonomous system, we look up the country of each IP address within the associated prefixes and choose the prevailing one. For this, we rely on the GeoIP database of Maxmind, Ltd. [16] (claimed to be 99% accurate).

In the proposed visualization, the measure of interest is encoded with color, using a predefined color scale and (usually) logarithmic normalization: $colorindex(v) = log(v+1)/log(v_{max+1})$. In some cases, analyzing an absolute measure of network traffic (such as number of connections or bytes transferred) yields hardly any insight in time-varying dynamics of the data. It may be more informative to run a backend database query to calculate a first or second derivative over time, and to visualize the result.

3 Related Work

Visualization and analysis of the internet backbone is an active research area. So far, a number of researchers and practitioners have used geographic, 3D, or abstract graph layouts to represent the network infrastructure [7]. Cheswick et al. [4], for instance, employed a spring-embedder graph layout to show connectivity in the internet graph. Another example is the AS level graph of Claffy [5] that uses polar coordinates to integrate both geography and connectivity into its layout. For each AS, the longitude of its headquarter is mapped to the angular position, while the outdegree of the connectivity graph determines its radial position. While these studies convey interesting technical information related to connectivity, our proposal aims at communicating the intensity of traffic measurements at prefix, AS, country, and continent level.

As mentioned before, a common way of displaying hierarchical data is with layouts where child nodes are placed inside their parents. Such layouts provide spatial locality for nodes under the same parent and visually emphasize the sizes of sets at all levels in the hierarchy. Usually, leaf nodes have labels or additional statistical attributes that are encoded graphically as relative object size or color.

The most important layouts of this type are *Treemaps* – space-filling layouts of nested rectangles, of which there are several main variants. The earliest type was the *slice-and-dice Treemap* [13]. Here, display space is partitioned into slices whose sizes are proportional to to the sum of the nodes they contain. At each hierarchy level this procedure is repeated recursively, rendering child nodes inside parent rectangles while alternating between horizontal and vertical partitioning. This is not difficult to program and can run fast, but long, thin rectangles arise, which are hard to perceive and compare visually. *Squarified Treemaps* [3] remedy this deficiency by using rectangles with controlled aspect ratios. Rectangles are prioritized by size, so large ones are given precedence in the layout. This improves the Treemap's appearance, but does not preserve input node order, which is a problem in some applications. This drawback was noticed, and *Ordered Treemaps* [2] were devised to address it. Most Ordered Treemap variants are pivot-based. Unfortunately, in our application we often deal with rectangles of highly varying sizes, and the resulting pivot rectangles can become highly deformed. We therefore investigated an adaption of pivot-based Ordered Treemaps, which we named HistoMap 1D, and compared it to *Strip Treemaps*. Similar to the *Million Items Treemap* [8], our goal is to handle a large hierarchy (more than 200,000 nodes). Unlike the latter, we do not intend to animate size changes, but instead focus on layout stability when redistributing the layout space after filtering out substantial parts of the hierarchy.

An alternative, non space-filling layout algorithm was proposed by Itoh et al. and applied to visualizing computer security data [12]. This method maps hosts to randomly placed rectangles, and subnets to larger enclosing rectangles. Another layout of IPv4 address space is a quadtree decomposition by consecutive pairs of IP address bits [20], (equivalent to "stacked coordinates") which was applied to analyzing route changes. *Root Polar Layout* [9] emphasizes the distinction between internal and external network addresses through their distance from the center. In contrast with these methods, ours needs to integrate geographic and abstract layout in the same view, while scaling up to the total IP address space.[2]

A related area is recent work on cartographic layouts, particularly, rectangular cartograms [11] that optimize the layout of rectangles with respect to area, shape, topology, relative position, and display space utilization. A genetic algorithm has been applied to find a good compromise between these objectives. This method renders layouts offline, not interactively, and does not yet exploit hierarchical structures. An overview of rectangular cartograms can be found in [21].

4 Space-filling layouts for diverse data characteristics

Each level of the network address hierarchy has unique characteristics that imply certain layout requirements. The continent and country level layouts should respect relative geographic positions (though some compromises must be made to make arbitrary space-filling layouts). At the AS level, we would like to place nodes with similar children close to each other, by arranging them in order of the middle IP address in each AS. On the IP prefix (lowest) level, ordering becomes even more important, as many adjacent IP prefixes share common owners and routing policies, or are semantically related (i.e., two universities that joined the Internet at roughly the same time). Table 1 summarizes these observations.

Table 1. Data characteristics of the IP hierarchy

Level	Data used for layout	Dimensionality
continent	center coordinate of each continent	2
country	center coordinate of each country	2
AS	middle IP address of contained prefixes	1
prefix	middle IP address	1

In contrast with classical Treemap algorithms, we apply different layout algorithms to various levels. A geographic layout method, *HistoMap*, is proposed for continents and countries; a fast one-dimensional version of HistoMap for autonomous systems aims at preserving neighborhoods in 1D; and a Strip Treemap [2] approach that preserves the input data order is applied to IP prefixes at the lowest level. The following sections describe this is more detail.

[1] A side-effect here could be that large ASes then contained fewer prefixes than they do today, making the AS level easier to render.

[2] Though this study considers only IPv4 networks, IPv6 addressing has a similar hierarchical prefix structure, though based on longer, variable-length prefixes. As HNMap does not display objects in the raw address space, the number of address bits itself is not an issue.

4.1 Geographic HistoMap layout

The upper two levels of the AS/IP hierarchy contain geographic entities. Geographic visualization can be very compelling – two-dimensional maps are a very familiar convention for representing three-dimensional reality. Mental models derived from maps are remarkably effective for many tasks, even when extreme scales and nonlinear transformations are involved. Numerous approaches have been investigated to showing geographically-related and more abstract information on maps [7].

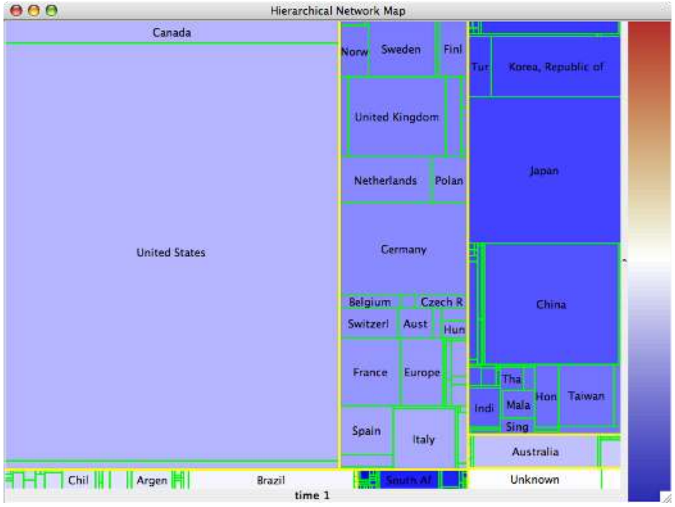


Fig. 1. Geographic HistoMap layout of the upper two levels of the IP hierarchy. Size represents the number of IP addresses assigned to each country. A seventh continent is placed below Australia to visualize ASes without country reference, anonymous proxies, and satellite providers.

To meet the challenging visibility goals of our application, our geographic maps rely on several kinds of abstraction: a) we omit oceans, b) we represent countries as rectangles, c) we size these rectangles proportionally to the number of IP addresses in the ASes assigned to those countries, and d) we reposition these geographic entities spatially while seeking to preserve neighborhoods.

Figure 1 shows the result of applying these abstractions within the HistoMap algorithm. HistoMap operates on four arrays containing latitudes, longitudes, weights, and an index P of the geographic entities. To avoid deep unbalanced recursive calls while operating on these large arrays, we split the arrays in a way similar to *Pivot-by-middle Ordered Treemap* using the partitioning algorithm sketched in Figure 2. As is conventional, the result of a split is that all elements on one side of the pivot element are less than or equal to the pivot element, and all elements on the other side are greater than or equal. This is achieved by choosing an arbitrary pivot from the array, and applying a modified version of *Quicksort*. In our algorithm, each successive recursive call to `partition` is limited to that part of the array that includes the middle position.

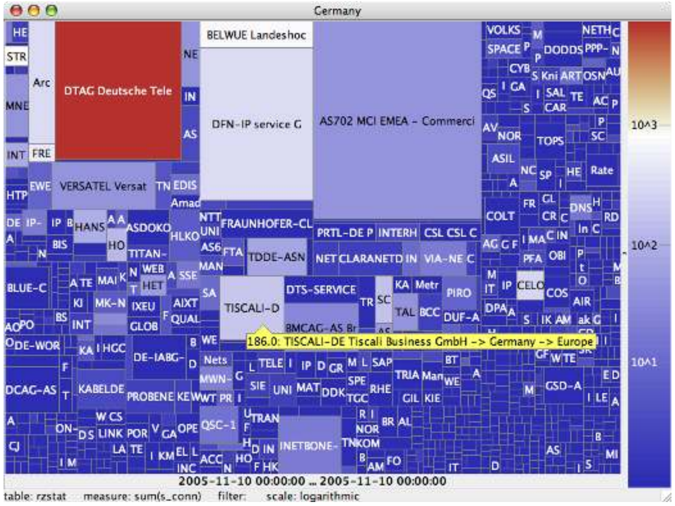
To test the quality of the horizontal and vertical splits in each call of *splitRect*, we first semi-sort the data according to longitude and then latitude. In fact, we only need to guarantee that all elements left (or right) of the middle element are smaller (or greater, respectively). Obviously, the second sort destroys the order of the first. Since we only operate on the given arrays, we define an absolute ordering on the data set through an index array to guarantee the reproducibility of the partitioning. In case of equality, the higher index determines which element is greater. For speed, our quality function assesses the squareness of the two rectangles in a greedy fashion without testing the effects of further splits through look-aheads. This approach is not limited to two-way splits, but can be extended to three-or-more-way splits. For simplicity, though, we used the 2-bin variant in all our experiments.

```
procedure partition ($P$)
begin
    if $|P| > 1$ then
        ($P_1$,$P_2$) ← splitRect ($P$);
        partition ($P_1$);
        partition ($P_2$);
    else
        drawRect ($P$);
end
procedure splitRect ($P$)
begin
    ($P_1$,$P_2$) = splitHorizontal ($P$);
    ($P_3$,$P_4$) = splitVertical ($P$);
    if quality ($P_1$,$P_2$) > quality ($P_3$,$P_4$) then
        return ($P_1$,$P_2$)
    else
        return ($P_3$,$P_4$)
end
```

Fig. 2. HistoMap partitioning algorithm

4.2 One-dimensional HistoMap layout

To place AS nodes in a spatially coherent way, we first attempted using AS numbers as a sorting criterion, but found that this reveals few interesting patterns. As an alternative, we calculate the median IP address of all prefixes advertised in an AS. When applying the HistoMap 1D layout, ASes that predominantly contain low IP prefixes (like 4.0.0.0/8) are placed near the upper left corner, and those containing high prefixes (like 239.254.0.0/16) are placed near the lower right corner. Figure 3 demonstrates the result of this layout algorithm run on a list of autonomous systems in Germany.



186.0: TISCALI-DE Tiscali Business GmbH -> Germany -> Europe

2005-11-10 00:00:00 .. 2005-11-10 00:00:00

table: rzstat measure: sum(s_conn) filter: scale: logarithmic

Fig. 3. HistoMap 1D layout of all autonomous systems in Germany. The measure (number of incoming connections) of each item is expressed through color.

One-dimensional HistoMap layout is a simplification of the geographic HistoMap algorithm. As with the geographic version, the effects of vertical and horizontal splits of the available display space are assessed, but the split through the data array is the same for both directions. A speed-up of 2.5 is obtained by avoiding the work of resorting. It is also possible to apply splitting to multidimensional data. Each split then represents a hyperplane orthogonal to the split dimension. (A look-ahead function can also be applied to make better layouts, but we did not implement this.)

4.3 Strip Treemap layout

Strip Treemap was chosen for the lowest-level layout for several reasons: a) its linear layout (Fig. 4) preserves the order of the rectan-

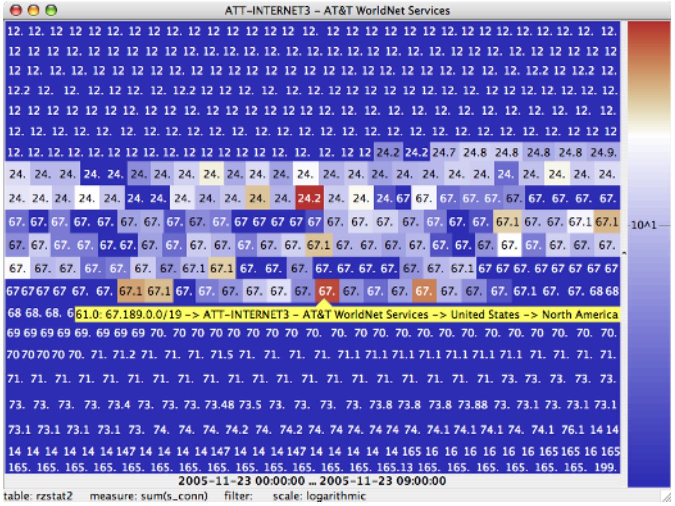

ATT-INTERNET3 – AT&T WorldNet Services
61.0: 67.189.0.0/19 -> ATT-INTERNET3 – AT&T WorldNet Services -> United States -> North America
2005-11-23 00:00:00 ... 2005-11-23 09:00:00
10^1

table: rzstat2 measure: sum(s_conn) filter: scale: logarithmic

Fig. 4. Strip Treemap layout emphasizing the order of prefixes in the autonomous system ATT-INTERNET3.

gles which aids readability, b) it can be implemented efficiently, c) it usually yields good aspect ratios, and d) it is relatively robust against changes [2].

In this algorithm, input items are first sorted by an index. Then, iteratively, items are added to the current strip. If, after recalculating sizes, the average aspect ratio of the rectangles within this strip increases, the candidate rectangle is removed from the strip, all other rectangles are finalized on the screen, and a new strip is initialized and made current. The algorithm terminates after processing all rectangles.

To obtain better aspect ratios of the rectangles, especially to avoid long, skinny rectangles in the final strip, we apply an optimization proposed by Bederson et al. [2], of maintaining a look-ahead strip and moving items from this strip to the current strip if the combined aspect ratio improves.

5 Evaluation of Data-Driven Layout Adaption

For experiments, we used the previously-described layout algorithms. We measured 1.5 seconds to render the complete IP hierarchy using HistoMap 1D, 1.9 sec. for Strip / HistoMap 1D, 3.1 sec. for HistoMap 1D / Strip and 4.6 sec. for Strip layout. We still see some further potential for optimization by applying array-based sorting to our implementation of Strip Treemap layout.

In the rest of this section, we will examine rectangles on the 3^{rd} (AS) and 4^{th} (prefix) levels with respect to a) visibility, b) average aspect ratio, and c) layout preservation when applying the HistoMap 1D, the Strip layout, or their combinations (at the prefix level only). All measurements in our experiments refer to the currently deepest visualized hierarchy level.

5.1 Visibility

Showing all data in a large hierarchy simultaneously is difficult, especially if there is large variance in data element size. The dilemma is that on the one hand, we would like to accurately convey sizes of all parts of the hierarchy. On the other hand, we would like to legibly show as many items as possible, including small ones. An obvious approach is to simply allocate screen space by the number of addresses in each prefix, which determines all node sizes. However, this approach does not cope well with large variances in prefix lengths, ranging from 2^0 (subnet mask /32) to 2^{24} (subnet mask /8).

One improvement is based on noting that many IP prefixes are contained in others. Borders for deeply nested hierarchy levels are costly in display space [18]. Instead, we can render overlapping prefixes adjacent to each other, assigning values (such as packet counts) to the most specific prefix, a common practice in analyzing routing data. If we have to assign display space to approximately 2 billion IP addresses, this still works out to half a pixel per 1000 IP addresses on a megapixel display or about 5 pixels on a 9 megapixel powerwall.

Further, we can normalize the size of the nodes at IP prefix level, using square root or logarithmic scaling. Experimentally, with square root normalization we were unable to place 657 prefixes using our best layout on all levels on a 1920 x 1200 pixel screen (net resolution: 1856 x 1132 pixels). Logarithmic normalization $f(size) = log_2(size+1)$ combined with certain heuristics can show all prefixes, and was thus the method of choice for all subsequent experiments. The heuristics are to assign a minimum width and height of 1 pixel to each rectangle, to give precedence to borders of larger rectangles, and to omit borders of smaller rectangles if there is insufficient space. As static labeling is not possible for many rectangles on the screen, detailed information, such as the value of statistical attributes and the path to the root of the hierarchy are shown when the mouse hovers over a rectangle.

Finally, it is admittedly very hard to see many randomly-placed one-pixel rectangles. By removing insignificant nodes (of very small area) we can give more space to other items. Figure 5 demonstrates the result of the algorithms when showing the IPv4 address space with logarithmically scaled network sizes using HistoMap 1D for the 3^{rd} and 4^{th} hierarchy levels. Note that large parts of the screen are blue, because there is no traffic to these networks and we inhibit the drawing of borders on the lowest level. Table 2 shows the detailed results of our experiments.

Table 2. Percentage of invisible rectangles of the IP hierarchy

Algorithm ($3^{rd}/4^{th}$ level)	AS (23054)	Prefixes (197427)
HistoMap 1D	0.00 %	0.00 %
Strip	0.03 %	0.46 %
Strip / HistoMap 1D	-	0.02 %
HistoMap 1D / Strip	-	0.16 %

5.2 Average rectangle aspect ratio

To evaluate squareness in a layout, we take the unweighted arithmetic average of the aspect ratios:

$$aspect\_ratio = \frac{1}{N}\sum_i \frac{max(w_i,h_i)}{min(w_i,h_i)} \quad (1)$$

Table 3 shows that the AS-level hierarchy is more difficult to render with respect to squareness. This may be explained by the large variances in size: though we logarithmically scale node sizes at the lowest level, the size of each higher level node is calculated by summing the sizes of all its child nodes (1495 nodes maximum). Overall, better results are achieved by HistoMap 1D layout, but the combination of Strip and HistoMap 1D offers improved aspect ratios.

Table 3. Average aspect ratio of rectangles of the IP hierarchy

Algorithm ($3^{rd}/4^{th}$ level)	AS (23054)	Prefixes (197427)
HistoMap 1D	3.077	1.749
Strip	5.754	3.106
Strip / HistoMap 1D	-	1.960
HistoMap 1D / Strip	-	2.218

5.3 Layout preservation

Showing all details at once is not always advisable from a perceptual point of view. Thus, after loading a large batch of raw data (such as a day's worth of traffic from a particular gateway) we prune the hierarchy by removing nodes in the order of least traffic and within that by least index, but keeping empty nodes in proportion to the total nodes

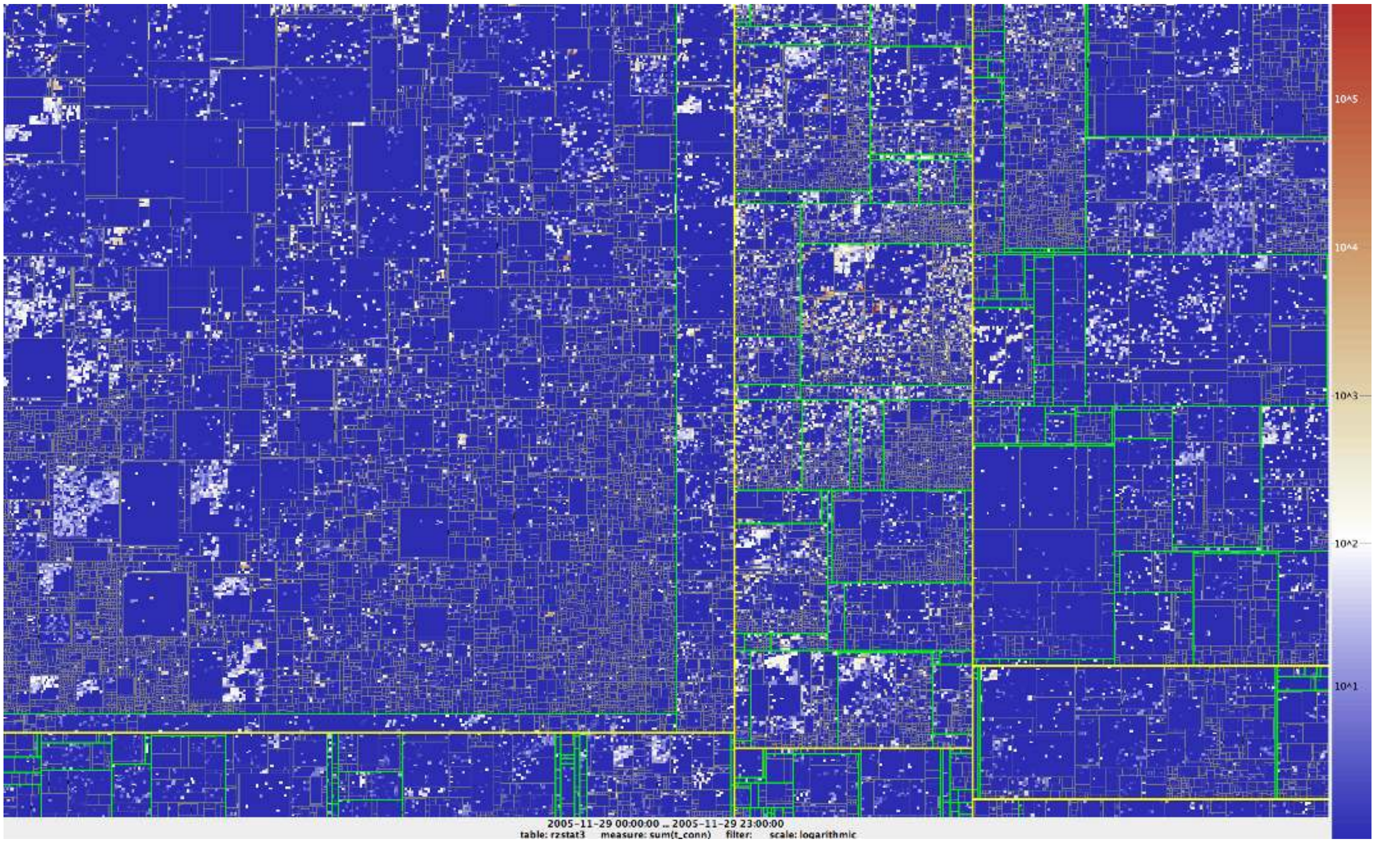


Fig. 5. Anonymized outgoing traffic connections from our university gateway on November 29th, 2005 showing all 197427 IP prefixes.

within a given parent. This helps stay close to the original data distribution while avoiding randomized sampling, and defines a precise node removal order that can be tested with all the layouts. Because the size of the higher level nodes is defined as the sum of its children, removing child nodes also effects the higher hierarchy levels. Subfigures 7(b) and 7(c) demonstrate the effect of removing nodes where traffic falls below a given threshold.

Following Bederson et al. [2], we calculate the unweighted average of the layout distance change metric $d(r_1,r_2)$ to numerically assess layout changes between each rectangle r_2 in the projected layout of the reduced hierarchy and its representation r_1 in the original layout. This metric takes into account both positional and absolute side changes.

$$d(r_1,r_2)=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2+(w_1-w_2)^2+(h_1-h_2)^2} \quad (2)$$

The strong correlation of the red and blue as well as of the green and black dashed lines in Figure 6 shows the higher impact of the layout algorithm on the 3^{rd} towards the 4^{th} hierarchy level. HistoMap 1D layout is thus preferred over Strip layout for the 3^{rd} (AS) level, as it better preserves the original layout. In the layout for the 4^{th} (prefix) hierarchy level, we could not find any significant difference between the two. The good results for 50% of the prefix nodes (dashed lines) under all layout algorithms can be explained through a major change towards 40 % in the geographic layout of the 1^{st} and 2^{nd} level due to unavoidable recalculation of the weights of the upper levels.

5.4 Summary

We demonstrated the benefits of the HistoMap 1D over Strip Treemap layout in several ways. First the latter is not capable of showing all nodes of our test data hierarchy at the target screen resolution, due to its sequential display space partitioning. Second, HistoMap 1D yields better average rectangle aspect ratio, on both the AS and prefix levels.

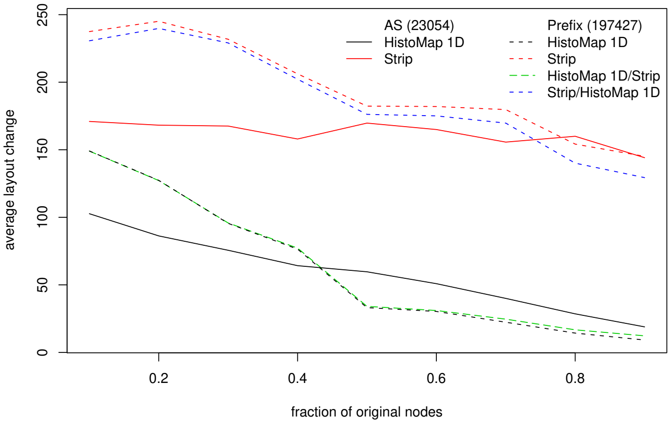


Fig. 6. Average layout change

Third, when filtering out parts of the hierarchy to focus on an area of interest, HistoMap 1D is better at preserving the layout.

We also showed that a combination of these two algorithms is promising, particularly due to the linear reading order of the Strip layout. In the case of a semantic ordering like the order of the prefixes, such a layout could reveal sequential scanning patterns or failures (continuous or discontinuous values in subsequent prefixes).

6 Case Studies

To demonstrate the fitness of our proposal, we conducted case studies with data from a production web server, a university gateway, and a large service provider. In all scenarios, the *Hierarchical Network Map* was the central exploratory tool. Other graphical representations were generated from data specified on the map.

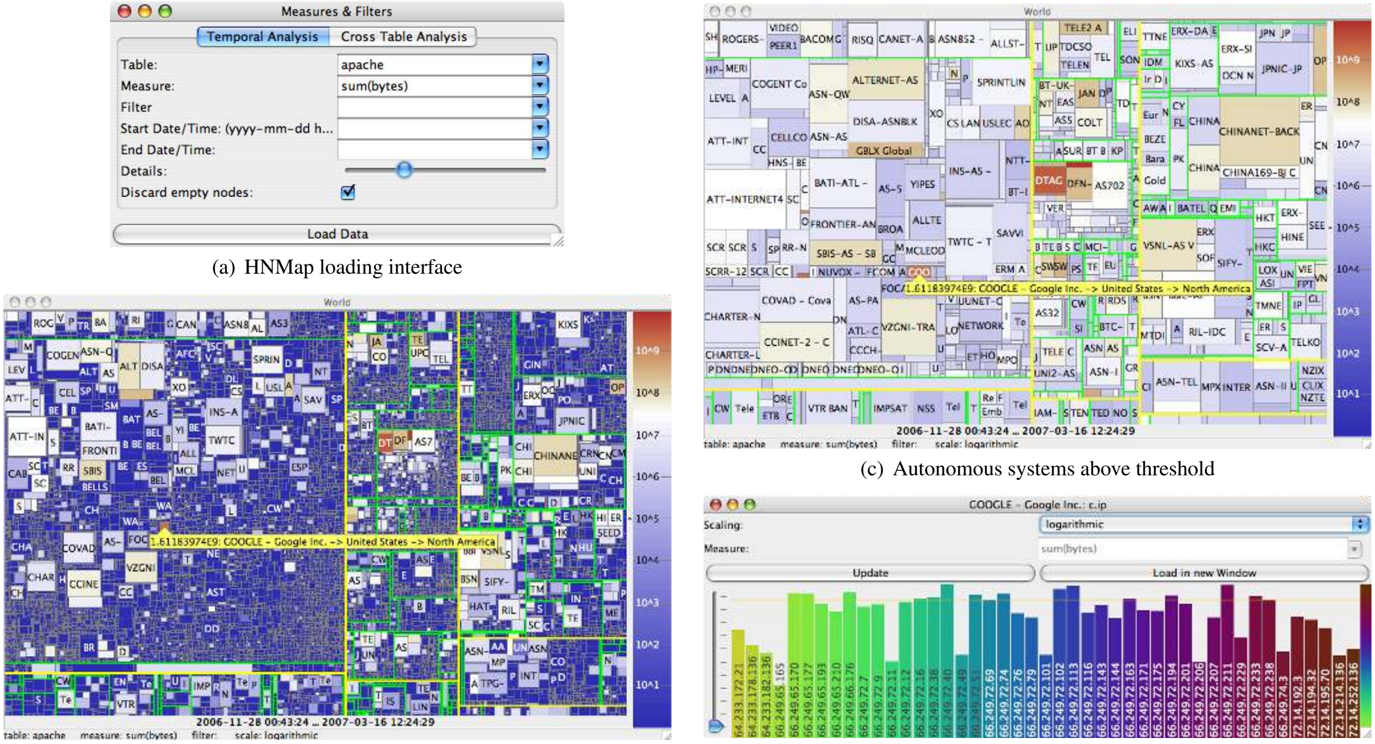


Table: apache
Measure: sum(bytes)
Filter
Start Date/Time: (yyyy-mm-dd h...
End Date/Time:
Details:
Discard empty nodes: ☑

(a) HNMap loading interface

2006-11-28 00:43:24 ... 2007-03-16 12:24:29
table: apache measure: sum(bytes) filter: scale: logarithmic

(b) All autonomous systems

(c) Autonomous systems above threshold

Scaling: logarithmic
Measure: sum(bytes)

(d) Interactive bar chart

Fig. 7. Visual exploration process: (a) Specifying visualization parameters in the HNMap loading interface (b) + (c) Removing nodes with little or no traffic from the hierarchy gives more space to the important. These two subfigures show traffic from our web server from 28th November 2006 till 16th March 2007 grouped by autonomous systems using HistoMap 1D layout. (d) Interactive bar chart with a detail slider to hide low value entries.

6.1 Resource location planning

A positive customer experience is crucial to the success of most businesses. Today, off-the-shelf web analytics software can show the geolocations of customers that visit a web site. This can be helpful for inferring customer demographics to optimize logistics or marketing strategies. Logically, we would also be interested in the ASes from which customers access a web server, so we could study the consequences of the placement of customer-facing web servers.

To conduct this analysis, we processed the Apache log files of our web server and loaded the IP address, a timestamp, and the transferred byte count of each web request into our system. The system connects to a database within a server and probes the set of available tables, which are prejoined with the IP network tables to save time during interactive exploration. Choosing the most appropriate measure is key to any analysis, and in this case we used the sum of transferred bytes to weight IP addresses. An optional filter can be invoked to ignore large transactions such as huge multimedia downloads that might otherwise skew the results. A checkbox enables removal of ASes without significant traffic. Other potentially distracting details can be reduced with the *detail slider* (see subfigure 7(a)).

Subfigures 7(b) and 7(c) show IP addresses of clients that visited our web server between 28 November 2006 and 16 March 2007, aggregated by AS. It clearly highlights the significance of the BELWUE and the DTAG systems, both grouped in the green country rectangle representing Germany. At the same time, the high volume requested by webcrawlers such as Google and Yahoo (Inktomi) as identified by their ASes are obvious. A right-click on any node brings up a context menu to either navigate within the hierarchy or trigger other graphical representations of detailed data. Figure 7(d) shows a logarithmically scaled bar chart using the previously specified measure on the data specified on the map.

This kind of analysis may be conducted in many other resource planning scenarios, such as choosing an optimal Internet service provider (ISP) for the intranet of a widely-spread company, or placing a shared database server at a favorable location.

6.2 Monitoring large-scale traffic changes

Another application for HNMap is monitoring network traffic. The two main interests here are to track and predict traffic volumes within one or several interoperating networks to keep the network infrastructure running well, and to react quickly to recover from failures.

For this case study, we used one day of Netflows captured at our university gateway, anonymized by removing the internal IP address to ensure privacy. We started by specifying the relevant data, such as the number of failed connections (including those that did not transfer any data). This gives an indication of malfunctions or scanning activities – many IP addresses within a probed network are unassigned or protected by a firewall and thus do not reply to the packets sent by scanning computers. Figure 8 shows the first derivative of failed connections over time aggregated on countries. The intense colors of the rectangles in Asia (upper right) attract attention and raise the question of what caused the high number of failed connections in the early morning. In this scenario, the color was chosen to show change: white for little or no change, blue for negative changes, and red for positive.

After opening bar charts for each of the suspicious countries, a conspicuously large number of different *destination ports* characterized the traffic from China, which would under normal circumstances mean that a large variety of application programs were used. For a more thorough analysis, we opened the *Radial Traffic Analyzer* (RTA) [14] to correlate the *source IP addresses* as the origin of this traffic with the used ports. After removing the IP addresses with relatively small shares of traffic through a slider, Figure 9 stresses that two IP addresses (218.56.57.58 and 202.102.134.68) were involved in port scanning activities, as seen on the colorful patterns on the outer ring. The RTA interface is capable of drawing one ring for each analyzed dimension (e.g., source & destination IP, application port, event type, etc.) of the data set. All rings are grouped by the dimensions of the inner rings and

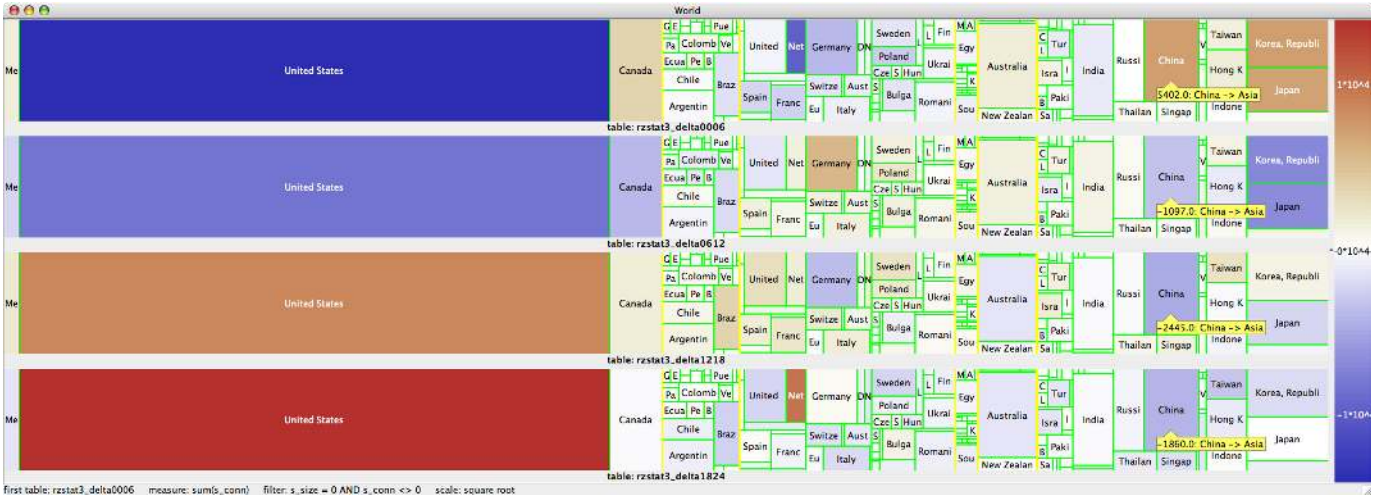
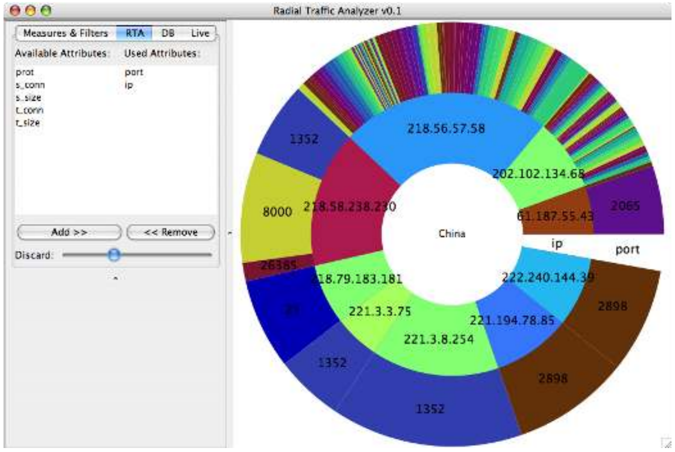


table: rzstat3_delta0006
table: rzstat3_delta0612
table: rzstat3_delta1218
table: rzstat3_delta1824

first table: rzstat3_delta0006 measure: sum(s_conn) filter: s_size = 0 AND s_conn <> 0 scale: square root

Fig. 8. Monitoring traffic changes: Failed incoming connections on our university gateway on November 29, 2005. The four map instances show for each country the increase (red) or decrease (blue) of traffic within the time span in comparison to the previous one. Note the use of square root normalization for the coloring to decrease the visual effect of outliers.

sorted according to their own dimensional values. Interactively rearranging these rings helps the analyst to explore the data and to gain deeper insight.



Available Attributes: prot s_conn s_size t_conn t_size
Used Attributes: port ip

Fig. 9. Radial Traffic Analyzer to find dependencies between dimensions

Such a scenario indicates that the HNMap is capable of showing the aggregates of a selected measure, or alternatively its first or second derivative over time. Navigational operations on the map, such as drilling down to deeper hierarchy nodes or rolling-up to see higher level aggregates, are limited to the IP dimension. Navigating along the IP dimension is often not adequate to find a “needle in the haystack.” We therefore supplement HNMap with graphical data representations such as bar charts and the RTA.

6.3 Botnet spread propagation

Between July and December 2006, Symantec observed an average of 63,912 active bot-infected computers per day [19]. Compared with the total number of computers on the Internet, this number is low. However, one should not forget the potential damage these hijacked computers might inflict when remotely controlled to collectively attack a commercial or governmental web server.

For this case study, we used a signature-based detection algorithm, which exploits the knowledge about known botnet servers, to collect bot-infected IP addresses. Without the visualization, we could only see that the list of IP addresses was continually changing, but we could neither map this change to the network infrastructure nor build a mental model of what was going on. Therefore, the goal of our analysis was to identify the IP prefixes, ASes, or countries with a high infection rate and to investigate whether they group on a higher hierarchy level.

The captured data was stored in a database system as before, and loaded into HNMap. We can see the severity of the spread and attention is drawn to the red rectangle representing China in the country view. Animating the map over time (one image per day) helps to confirm the dynamics of the developments in China. Figure 10 shows the results of zooming in to detailed IP address ranges grouped by AS (yellow borders) to identify which prefixes and ASes played a major role in the infection. We can observe that the infection was widely spread in the large AS on the upper left and in the smaller and thinner AS on the lower left. Furthermore, a few red colored prefixes outside these ASes probably contributed considerably to the spread of the worm.

With this knowledge, network operators can adapt firewall configurations to block or filter traffic from the relevant IP prefixes.

6.4 Expert knowledge

Informal evaluation by visualization and domain experts who tested HNMap yielded many improvements, such as an interactive feature to move the transition point in a bi-color scale, time-series animation, linked parallel instances of the map, and intelligent placement of popup labels, to name a few.

7 Conclusion

In this study, we examined various space-filling layout algorithms, and demonstrated combining them to display the *IP/AS hierarchy*. We compared HistoMap 1D and Strip Treemap layouts with respect to rectangle visibility, rectangle aspect ratios, order and layout stability when filtering large parts of the hierarchy. At the AS level, we conclude that HistoMap 1D layout is preferable in scalability, squareness, and stability. In the IP level, Strip Treemaps are most appropriate due to their maintaining the input order of nodes, though some compromises are made.

In three case studies, we explored the usefulness of *Hierarchical Network Maps* and resulting insights into data sets for network resource planning, monitoring, and security. To the best of our knowledge, our tool is the first that visually compares large-scale network data aggregated by prefix, autonomous system, country, and continent.

In the future, we hope to improve layouts at the AS level by exploiting information about border gateway router connectivity. Methods for avoiding deformation of small rectangles are placed next to large ones in the Geographic HistoMap and HistoMap 1D layout may also be beneficial. Because this study emphasized the importance of the upper hierarchy levels to layout stability, we will also consider more closely to what extent sacrificing squareness in the geographic hierarchy levels improves layout stability.

Acknowledgements

This work has been funded by the German Research Society (DFG) under the grant GK-1042, Explorative Analysis and Visualization of Large Information Spaces, Konstanz, and by the Information and Software Systems Research Lab in AT&T. We thank the anonymous reviewers of IEEE InfoVis 2007 for their valuable comments.

References

[1] T. Bates, P. Smith, and G. Huston. CIDR Report, September 2006. `http://bgp.potaroo.net/cidr/`.

[2] B. B. Bederson, B. Shneiderman, and M. Wattenberg. Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies. *ACM Trans. Graph.*, 21(4):833–854, 2002.

[3] M. Bruls, K. Huizing, and J. J. Van Wijk. Squarified treemaps. In *Proceedings of the Joint Eurographics and IEEE TCVG Symposium on Visualization*, pages 33–42, 2000.

[4] B. Cheswick, H. Burch, and S. Branigan. Mapping and visualizing the internet. In *Proc. 2000 USENIX Annual Techincal Conference*, pages 1–12, 2000.

[5] K. C. Claffy. Caida: Visualizing the internet. *IEEE Internet Computing*, 05(1):88, 2001.

[6] E. F. Codd, S. B. Codd, and C. T. Salley. Providing OLAP (on-line analytical processing) to user-analysts: An IT mandate. *Technical report, E.F.Codd & Associates*, 1993.

[7] M. Dodge and R. Kitchin. *Atlas of Cyberspace*. Addison-Wesley, 2001.

[8] J.-D. Fekete and C. Plaisant. Interactive information visualization of a million items. In *InfoVis 2002, IEEE Symposium on Information Visualization*, pages 117–124, Los Alamitos, CA, USA, 2002. IEEE Computer Society.

[9] G. A. Fink and C. North. Root polar layout of internet address data for security administration. In *Proc. IEEE Workshop on Visualization for Computer Security (VizSEC)*, pages 55–64, October 2005.

[10] J. Hawkinson and T. Bates. RFC 1930 Guidelines for creation, selection, and registration of an Autonomous System (AS), March 1996.

[11] R. Heilmann, D. A. Keim, C. Panse, and M. Sips. RecMap: Rectangular Map Approximations. In *InfoVis 2004, IEEE Symposium on Information Visualization, Austin, Texas*, pages 33–40, October 2004.

[12] T. Itoh, H. Takakura, A. Sawada, and K. Koyamada. Hierarchical visualization of network intrusion detection data. *IEEE Computer Graphics and Applications*, 26(02):40–47, 2006.

[13] B. Johnson and B. Shneiderman. Tree-maps: A space filling approach to the visualization of hierarchical information structures. In *VIS '91: Proceedings of the 2nd IEEE Conference on Visualization*, pages 284–291, 1991.

[14] D. A. Keim, F. Mansmann, J. Schneidewind, and T. Schreck. Monitoring network traffic with radial traffic analyzer. In *Proc. of IEEE Symposium on Visual Analytics Science and Technology 2006 (VAST 2006)*, pages 123–128, 2006.

[15] Z. M. Mao, D. Johnson, J. Rexford, J. Wang, and R. H. Katz. Scalable and accurate identification of as-level forwarding paths. In *INFOCOM*, volume 3, pages 1605–1615, 2004.

[16] Maxmind, Ltd. Geoip database, 2007. `http://www.maxmind.com`.

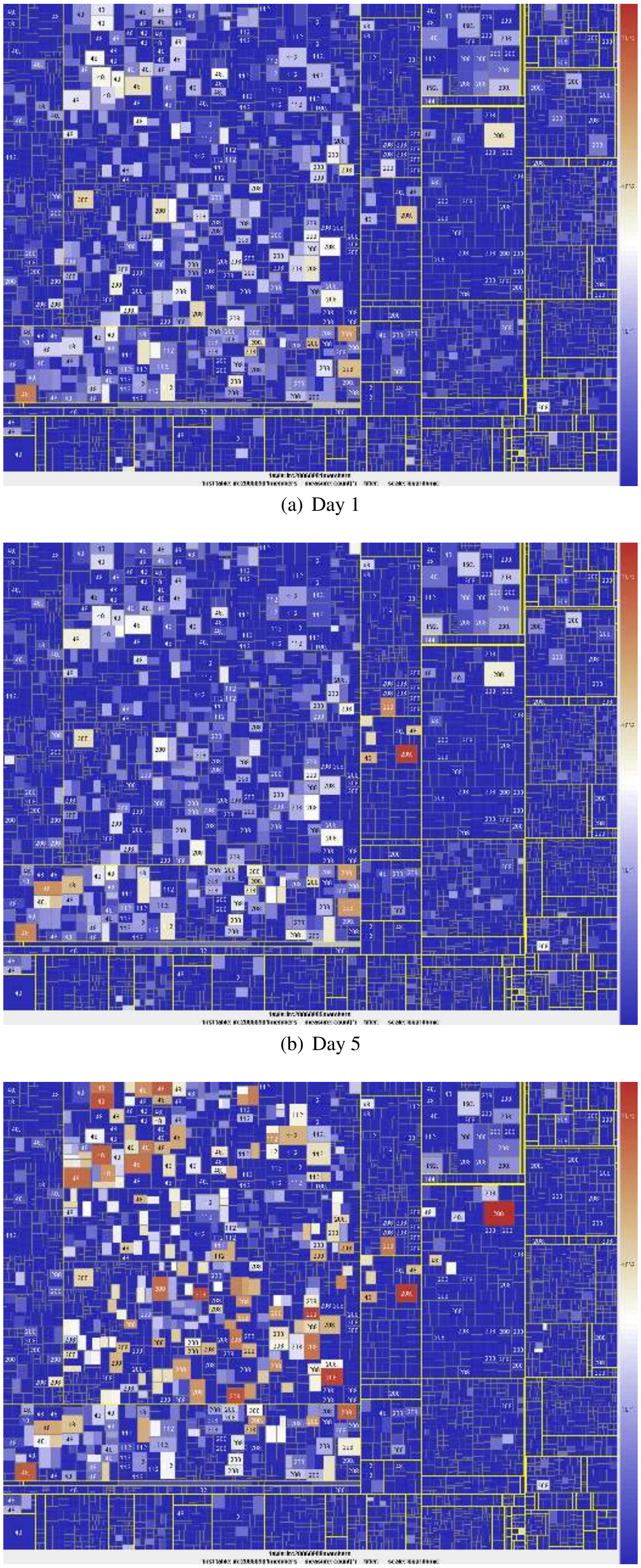
[17] T. B. Pedersen and C. S. Jensen. Multidimensional database technology. *IEEE Computer*, 34(12):40–46, 2001.

[18] T. Schreck, D. A. Keim, and F. Mansmann. Regular treemap layouts for visual analysis of hierarchical data. In *Spring Conference on Computer Graphics (SCCG'2006), April 20-22, Casta Papiernicka, Slovak Republic*, pages 184–191. ACM Siggraph, 2006.

[19] Symantec. Symantec Internet Security Threat Report: Trends for July-December 06, March 2007. Volume XI.

[20] S. T. Teoh, K.-L. Ma, S. F. Wu, and T. Jankun-Kelly. Detecting flaws and intruders with visual data analysis. *IEEE Computer Graphics and Applications*, 24(5):27–35, 2004.

[21] M. van Krevelda and B. Speckmann. On rectangular cartograms. *Computational Geometry*, 37(3):175–187, 2007.



(a) Day 1

(b) Day 5

(c) Day 9

Fig. 10. Rapid spread of botnet computers in China in August 2006 as seen from the perspective of a large service provider. The yellow boxes group prefixes by ASes. We anonymized the prefix labels because of privacy concerns.