

Figure 1: Example 1

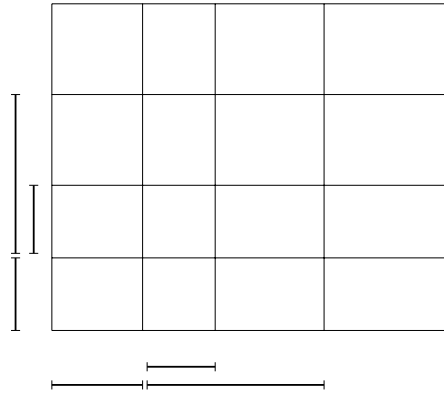


Figure 2: Example 2

This write-up has some new proposals for the traffic visualization tool we discussed as well as a more clear description of some things I might have been unclear about. First I go into some detail about the rendering of overlapping 2-D rectangles representing the two dimensional traffic clusters. Next I give some new ideas about what we can put *inside* these rectangles. I conclude with some open questions (which does not mean that the rest of the stuff here is not open problems, it only means that I have a fuzzier image about the things covered at the end).

1 Drawing the rectangles

Unidimensional clusters (e.g. source and destination prefixes) are represented by horizontal and vertical line segments and the bidimensional clusters are the rectangles defined by these segments. Longer lines can include shorter ones (larger networks include smaller ones), but they cannot overlap arbitrarily. With the rectangles we have more complicated overlap relations. I believe that our rendering of the rectangles should provide visual cues about these overlap relations. For example figures 1 and 2 show that even though the segments along the vertical dimension are different, the 2 dimensional clusters look the same.

To exemplify the inclusion relationship of the rectangles, we can use a stacking metaphor (without do-

ing any 3D rendering – see figure 3): a small rectangle included in a bigger one will be visibly stacked on a larger one that contains it. We can achieve this by making it slightly smaller along the dimension they have the same definition, so that there is a narrow brim outside the inner rectangle. Since there is a bound on the depth of nesting along each dimension, we can limit the screen real estate consumed by these margins.

There will be cases where two rectangles overlap without one including the other. We need not make any size adjustments for this case, but we must decide in which order we stack them. It really doesn't matter, but we should be consistent: say always put the rectangle larger in the vertical dimension on top of the horizontal one.

1.1 Brightness of rectangles

The brightness is a logarithmic unexpectedness label, with light gray representing 100%, darker colors representing higher values and brighter ones smaller values. It is clear what the color of a rectangle that does not include others should be. What is the color of a rectangle that includes others? We compute the unexpectedness label for *the rest of the traffic* and use that as the color.

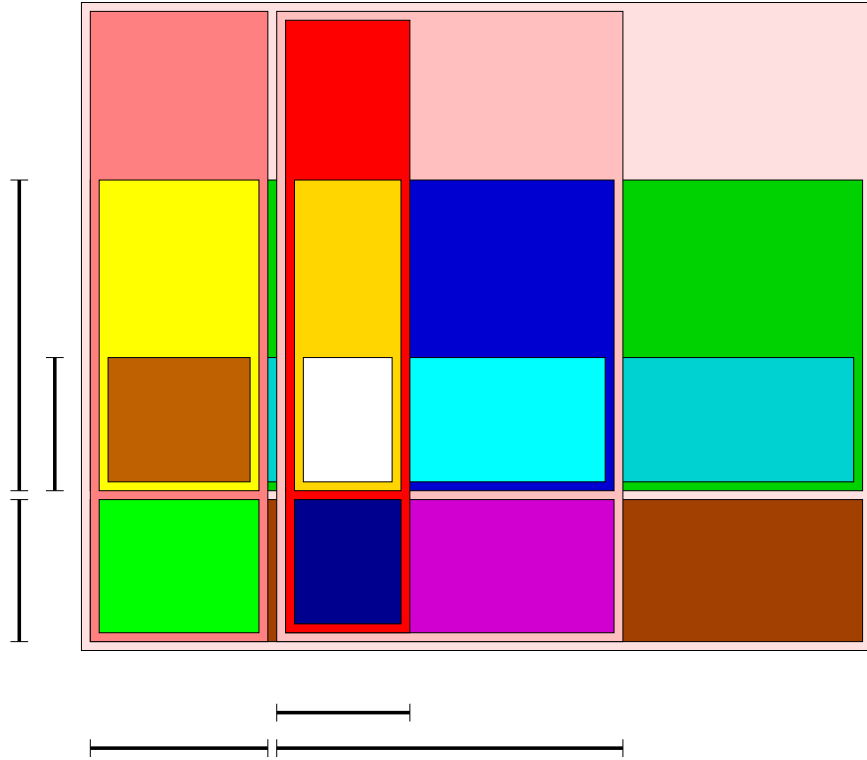


Figure 3: The rectangles are not supposed to be colored, they should only differ in brightness. Maybe the margins are too wide.

2 What to put into the rectangles

The size of the segments representing networks is proportional to the traffic they send/receive. The threshold parameter used by the algorithm gives a lower bound the size of these segments. The order of the segments should reflect their position in the address space, and the inclusion should reflect the inclusion relationships. But what should the offsets be at which these segments are placed?

I proposed placing the segments included in a larger one so that the space not covered by them is equally distributed *between* the shorter segments. Now I incline to think that it makes more sense to leave the “remaining traffic” to the end and push all

the included segments to the beginning. This way each longer segment that includes shorter ones will be divided into one slice for each included segment plus one slice representing the part of the address space not covered by the subnetworks explicitly described by included segments. Fortunately the threshold that gives a minimum size for the segments also constitutes the minimum size for any of these “remaining traffic” segments.

We will have two types of visible rectangular surfaces: rectangles representing traffic clusters that include no smaller clusters, and rectangles representing the uncovered area of larger clusters not covered by more specific (smaller) clusters stacked on top of them. Because all segments have a certain minimum size, all these visible rectangles are beyond a mini-

mum size along both dimensions, so by choosing a suitably large threshold, we can ensure enough real estate to put some readable text or even a pie chart inside these visible rectangles.

We can let the user choose between a few predetermined values for the threshold. At the lowest value, we get small segments and this results in a detailed description of the traffic mix, but the visible rectangles can get so small that we have no place to put anything in them. With a larger threshold we probably have space to put some numbers giving the total traffic (and the remaining traffic in parentheses for the visual rectangles of the second type that do not represent a cluster, but what's left of a cluster after part of it is covered by a more specific one)¹. At an even larger threshold we can probably afford to include the other two measures of the traffic (e.g. not just bytes, but also packets and flows) and derived measures such as average packet size and average flow length (in packets).

Instead of adding these details the user could choose to get, colored pie charts that expresses the division of the traffic along the third dimension (e.g. application). The colors in this pie charts should be consistent over all pie charts displayed together (one for each visible rectangle that has enough traffic). We can easily use the threshold to bound the maximum number of colors per one pie. Note that assigning the colors consistently might require a very large number of colors because it can happen that we get different applications in each pie (but I do not expect that to be the common case). Furthermore these colors should be assigned automatically. Any ideas? Ideally we would want the pie chart to express inclusion relations among its slices, just like the segments express inclusion relations. Is there such a thing as a hierarchical pie chart?

3 Open questions

It is clear how one can drill down by clicking onto the right rectangles. Is there a convenient way to show the drill-down action through some kind of animation

¹Or would it be better to first give the “remaining traffic” and the total in parentheses?

(as opposed to directly switching to the new view)? For example we can keep the same rectangles as before, only stretched and with different colors². Maybe this is a good transition phase on the way to the final view after the drill-down. After this maybe instead of recomputing the segments for both dimensions at a time, we can do it in two steps. On the other hand if it is this hard to explain I wonder how easy it will be for the user to understand what's going on. Your thoughts?

How about only making the line segments clickable? This means that the users will be able to refine only one direction at a time and it might be easier to follow.

Should there be a non-graphical drill down? I think that allowing the user to specify the source and destination prefixes he wants to drill-down to is a good way of accomplishing this. And it can happen that the prefix the user is interested in is not visualized due to compression or low traffic.

How do we do the inverse of drill-down? What if the user wants to see a broader view (i.e. larger network)? It would be nice to also have a visual metaphor for this “zoom out” operation but I have absolutely no ideas.

The simplest drill-down specifies a prefix that we focus on for each dimension. Would it be useful to allow a list of prefixes instead? Or to allow the user to *exclude* specific prefixes? Wouldn't this make the whole GUI terribly cumbersome to use and hard to understand?

What about the rest of the interface? We probably need some more navigation functionality (e.g. navigating through time intervals). Copying the AutoFocus GUI is easy, but probably there are better solutions. Let me just point out here that AutoFocus places large emphasis on precomputed traffic reports and thus it cannot show a view of say the current day until after the end of the day. Since we do not com-

²It might be surprising, but by keeping the same rectangles, we lost the lower bound on the size of a segment. Say we drill down from a view of all the traffic to the traffic going to one network. All the destination network segments will stretch, but the source network segments can go either way. For example if a source sends 10% of the total traffic it can send more than 10% of the traffic received by this destination network. Just as well it might happen that it sends no traffic to this one

pute multidimensional traffic reports like AutoFocus, we can shift more emphasis on user-driven computations (since they are relatively cheap) and provide an almost real-time view (we can refresh say every 5 or every 60 seconds).

Can we find a similar metaphor for differences between two traffic mixes? I do know for sure that network operators often resort to comparisons against previous traffic mixes. With the textual traffic reports of AutoFocus, “delta reports” are possible, but I couldn’t find an easy way to add them to the GUI (and they were too expensive to precompute which won’t be a problem for us here). I would be really really cool to have this one. Any ideas?

Should we leave any time series plots in? They seem to be very popular and network operators like and understand them. Could we combine them with the interface we discuss? Maybe we could leave in an option for replacing the pie charts with time series plots (although it would be hard to implement it).

Should we keep the idea of traffic categories? Would it work well with the rest of the system?