Assignment #6: StreamGraphs

Caitlin Ross & Noah Wolfe
03/21/2016

1 Data Source (ROSS)

In this assignment, we have decided to generate/collect data from our ROSS research projects. ROSS is a massively parallel discrete-event simulator that can process billions of events per second [3], [1]. ROSS models are made up of a collection of logical processes (LPs). Each LP models a distinct component of the system. LPs interact with one another through events in the form of time-stamped messages. An MPI task is abstracted as a processor element (PE) in ROSS. Each PE owns a number of LPs and schedules events in time-stamp order for all LPs assigned to it. Events that are destined for a logical process on another PE (i.e. remote events) are sent as MPI messages.

For this assignment we decided to collect remote event data for each PE in the simulation. Running with 8 total PE's we collect the total number of MPI messages sent per PE at each instance of simulation time. The first dataset is generated with a simulation using the Dragonfly network interconnect model with uniform random traffic using minimal routing. The second dataset is generated using a Slim Fly network interconnect model simulation also using uniform random traffic and minimal routing. The data is extracted into a simple csv format with the following layout: PE#, sim-time, remote-events. We started off collecting 1,000 points in time for each PE but the resulting streamgraphs looked terrible (very discrete and spiky instead of smooth and continuous) so we dropped the number of time samples down to 100 per PE for a total of 800 points in time per graph.

2 Bar Graph Visualization

In order to generate the stacked bar graphs, we decided to use Excel. Excel is very straightforward and has a simple interface with many options for color. Also, our data set was in the csv format so loading into Excel was very easy. The resulting stacked bar graphs are shown in Figure 1. Presenting the data in bar graph form leaves a lot to be desired. On the positive side, one can quickly glean the overall trend of all MPI sends for all PEs in each simulation but that is it. The separation and differences between the different PEs is very difficult to see which is a direct result of trying to smash so much data into bars in a small horizontal area. It forces the width of the bars to be very small. Also, as a result of the small bar widths, color isn't allowed to have a beneficial effect on the data visualization. One of the strong properties of Excel is selection of color schemes but unfortunately the bar graph cancels out the color capability.

3 Stream Graph Visualization

We used William Turman's Stream graph D3 implementation [4] to create the stream graphs of our data. This required very little modification to work. We got our data into a similar format as the example, except that we use an integer to represent simulated time, instead of using a real date as the example does. We had to make some Javascript modifications to appropriately handle the time information. Then we were able to add both data sets to the same page. The stream graphs of both the Slimfly and Dragonfly data are shown in Figure 2. The colors used are from the ColorBrewer website [2].

The graphs are hosted at `https://caitlinross.github.io/vis/streamgraph.html`. There is an interactive aspect that is not shown in the figures. When hovering over a layer, several things occur. The opacity of the other layers are decreased to make the layer selected stand out. A vertical bar moves to where the mouse pointer is. Finally, a tooltip appears that shows the PE id and the value for that PE at that point in time.

4 Bar vs Stream Graph Comparison

Overall, we think it is clear the stream graph is a much better choice for large time-series data sets as compared with stacked bar graphs. Both graph approaches make observation and analysis of global/general features quick and easy. After a quick glance at both sets of visualizations, it is clear the Slim Fly network has a much more consistent MPI send rate while the Dragonfly is more sporadic. However, the similarities between the bar graphs and stream graphs end there. On one hand, the stream graph visualizations perform better in representing the differences in the PEs, clearly showing the separation of the PE data by colors and making it easy to follow along the x-axis. With the bar charts, it's nearly impossible to differentiate between the different PEs and the thin bars makes it impossible for any coloring to help the matter as there is no space to color. On the other hand, the labels on the y-axis don't make much sense in the stream graph, since the baseline is not set to 0. The stacked bar graph chart has a very clear y-axis and thus makes it easier to see the number of remote events for all PEs for a given point in simulated time. These numbers can of course be calculated from the stream graph y-axis with some extra effort. In the end, the stream graph does a much better job of providing detailed analysis when dealing with large amounts of data as shown in this report. The ability to distinguish and follow trends in the data at the global simulation and local PE levels makes the stream graph the preferred choice.

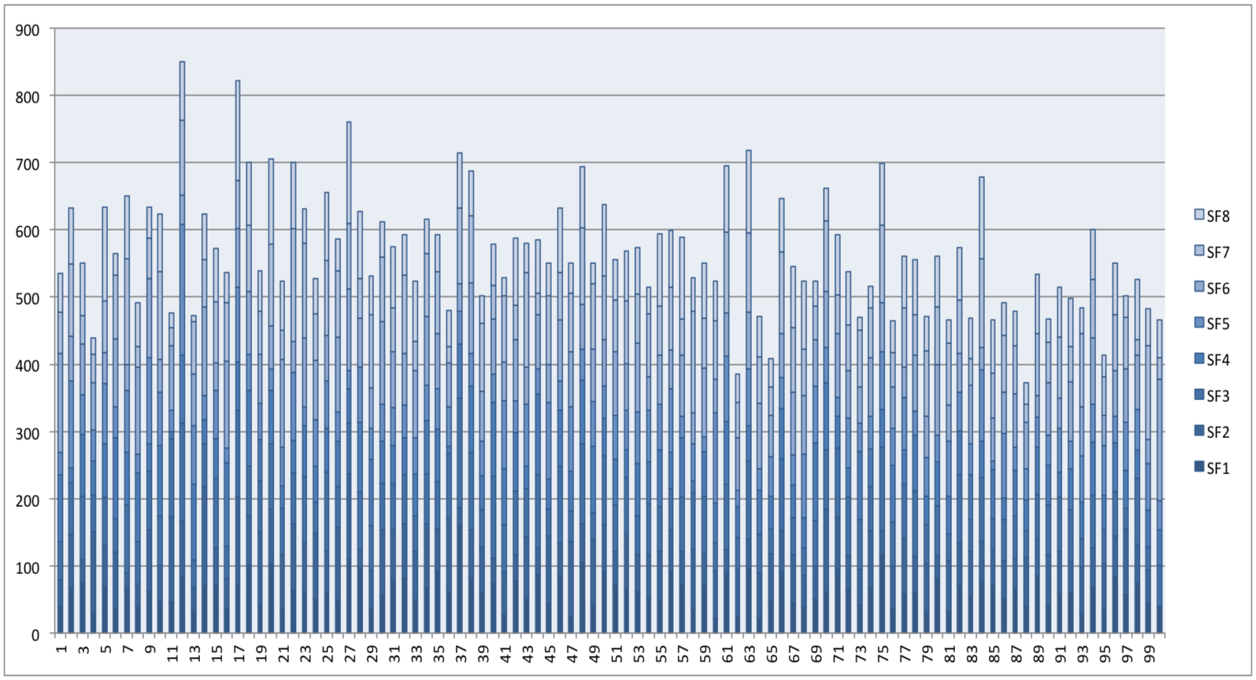
References

[1] D. W. Bauer Jr., C. D. Carothers, and A. Holder. Scalable time warp on blue gene supercomputers. In *Proceedings of the 2009 ACM/IEEE/SCS 23rd Workshop on Principles of Advanced and Distributed Simulation*, PADS '09, pages 35–44, Washington, DC, USA, 2009. IEEE Computer Society.

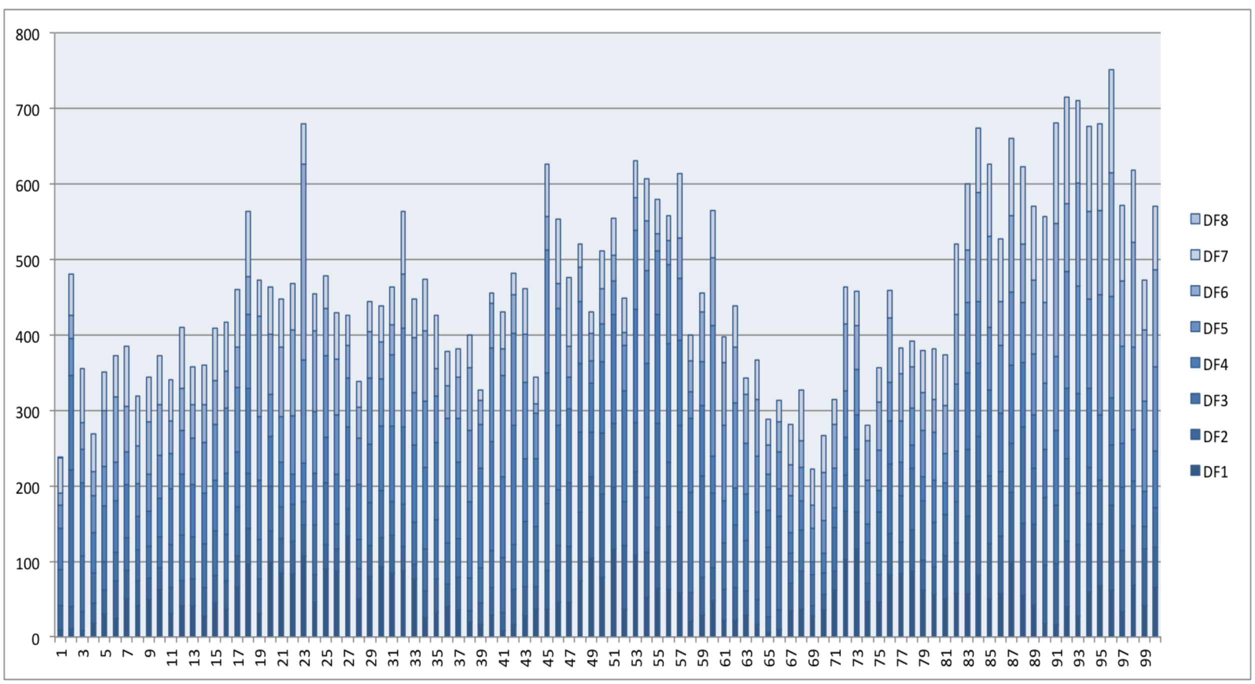
[2] C. Brewer and M. Harrower. Colorbrewer2. `http://colorbrewer2.org/`. Accessed: 2016-03-21.

[3] A. O. Holder and C. D. Carothers. Analysis of time warp on a 32,768 processor ibm blue gene/l supercomputer. In *in 2008 Proceedings European Modeling and Simulation Symposium (EMSS*, 2008.

[4] W. Turman. D3 interactive streamgraph. `http://bl.ocks.org/WillTurman/4631136`. Accessed: 2016-03-18.

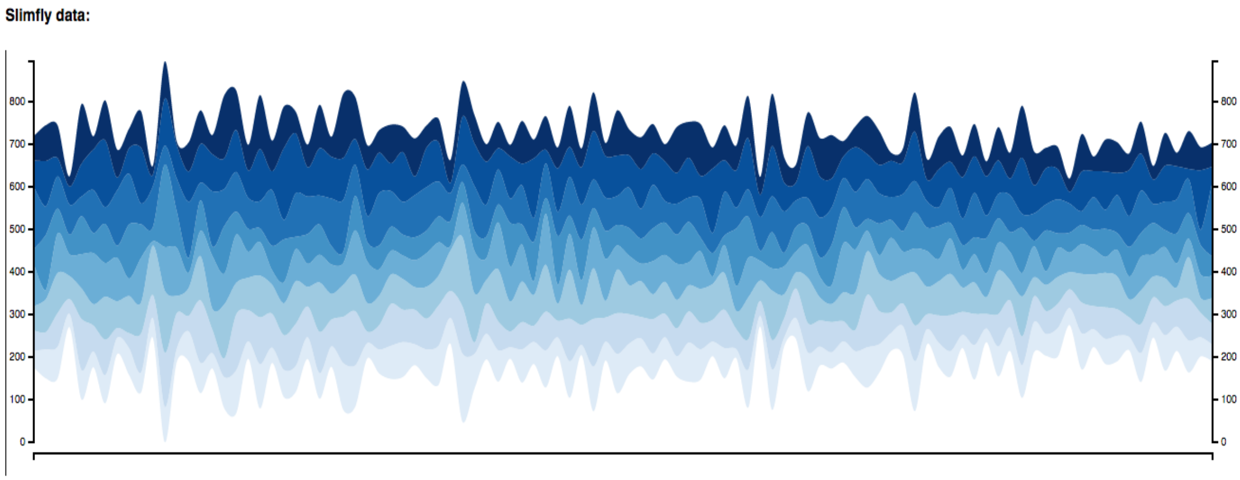


(a) Slim Fly

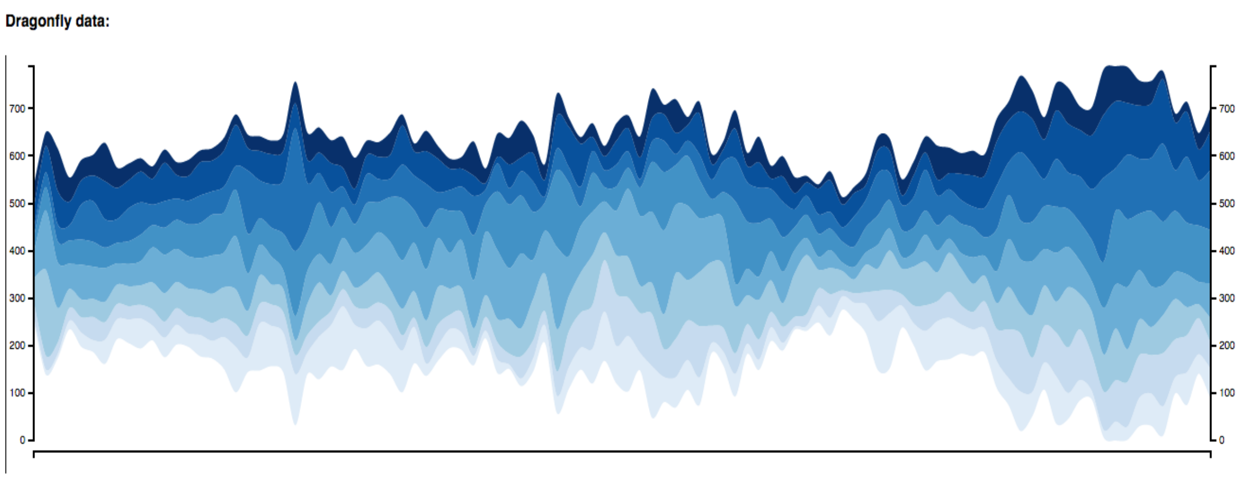


(b) Dragonfly

Figure 1: Stacked bar chart visualizations of the Dragonfly and Slim Fly remote message simulation data sets showing the number of MPI messages being sent from each PE in the simulation over time. Both figures start with PE 1 (”SF1” or ”DF1” in the legends) on the bottom of the stacked bars and the data for the following PEs is stacked on top, in order, and finishing with PE 8 (”SF8” or ”DF8” in the legends) on top. Each bar along the x-axis represents a different point in the simulation time.



(a) Slim Fly



(b) Dragonfly

Figure 2: Stream graph visualizations of the Dragonfly and Slim Fly remote message simulation data sets showing the number of MPI messages being sent from each PE in the simulation over time. Both figures start with PE 1 (lightest blue) on the bottom with the following PEs layered in order on top, and finishing with PE 8 (darkest blue) on top. The x-axis shows simulation time increasing from left to right. The vertical height of a layer (i.e., PE) corresponds to the number of remote events for that PE.