Presenting your data

Hello!

Welcome to the module on presenting your data.

Slide 2: Presenting your data

There are two major reasons why you are interested in presenting your data. The first of these is, of course, that you want to visualize the data yourself because you want to understand your data you want to listen to your data. And you want to understand how to exploit your data, etc. And the other is that you want to be able to present your data to your audience because you want to explain the differences that you found and you want to facilitate the audience understanding what it is that you found in the data and perhaps even making use of your data and building upon it themselves

Slide 3: Presenting information with images

Well, one of the reasons that we present data with images is - as the popular saying goes, a picture is worth a thousand words. As we said before, we can use pictures, graphs, flowcharts, UML, state machines, etc. because they can all convey an enormous amount of information if they are used well. Consider "a wink" at a party - how much information that carries.

Slide 4: Why use graphical presentations?

So why do we use graphical presentations? And the reason - first and foremost - is they are very compact you can put a lot of data in a very small amount of space - in comparison to the table which takes more space for the same amount of data. It also makes it easy to bring out differences and therefore facilitates the viewer being able to do comparisons. And often is the case that we can take an abstract view with the picture, and therefore we can make it much easier to understand the concept that we are trying to convey that we've learned from our data. Of course, many people are very good at seeing patterns in visual scenes, and thus it's very useful for being able to present the data visually to these people. And we, of course, use visual presentations/images to be able to provide clarity and objectivity. You can look at the data yourself and come to your own interpretation based upon what you see in the data. And of course, one of the reasons we also use it is to support our written text because we to tell the particular story that we want to convey, and therefore we need to add the figures at just the appropriate places to support that story.

Slide 5: A graph is a encoding, when you look at it you need to visually decode it

Now, William S. Cleveland's book, "The Elements of Graphing data", has a preface in it. Where he really talks about the fact that when you make a graph, you're taking the qualitative and categorical information and your including it using the display method, then he says, the processes of course now the viewer is going to look at it, and they're going to visually decode it, and he emphasizes how that's the very very vital link. So no matter how impressive the technology for the encoding, in the end, the user needs to be able to efficiently and accurately decode what it is that you presented. And in his book, he goes on to display that there are some fundamental elements to how to encode the data to facilitate the use of being able to decode it.

Slide 6: Edward Tufte's books

As I mentioned in one of the earlier modules: Edward Tufte has done a number of books on how to present information well and not only well but actually present beautifully. And I encourage you to read his books.

Slide 7: Measuring a FASP file transfer

Now, I was inspired by the National Center for Biotechnology Information's "Aspera Transfer Guide", because a master's thesis student who was using this Aspera protocol for transferring files. This protocol designed to be able to rapidly transfer very large files. So I said, "okay, let's do an experiment to see how fast it does transfers" and "can I understand what's going on in this protocol even though the protocol is proprietary". So I set up the data collection using a one-gigabyte test file, and I use tcpdump to capture the traffic as described here on interface 0, and I wrote it to a temporary file.

Slide 8: Start ascp to transfer 1G from test server

So I started the file transfer, I looked to see where the executable are that I had downloaded from the website. And now I started the particular program, saying I wanted to get their test data of a one-gigabyte file, I wanted to put in my temp directory, and I specified the maximum rate to be used was a hundred megabits per second. And I've disabled encryption so I could later see the plain text of the actual messages being sent.

Slide 9: FASP session starts

So it output this log information, and the session started, and we get some additional log information, and we see that the initial path MTU was 142 bytes and the starting estimate of the round trip time was a hundred and 174 milliseconds - since the site was located in California and I was running this on a machine here at KTH.

Slide 10: Intermediate output

So it proceeded to transfer blocks of data, and here are some intermediate log data at 0%, 22%, 45%, 68%, 91%, and 100% of the transfer of the file.

Slide 11: FASP transfer stops

Eventually, the file transfer stops, and we got a bunch of data out of that, and we can see it transferred this file which was a 1024 times a thousand kilobytes - so one gigabyte. It was transferred in 87 seconds. For a total of 95875 kilobits per second. The actual measured delay was 176 ms. And the average transmission rate was 98.63 megabits per second - so just slightly under the 100 megabits per second that I specified. And if you look at the sender control information being sent from their server to my client, we see that it sent 112 control messages of which all of them were received whereas the receiver sent traffic in the reverse direction and it sent 879 control messages, but two of those were lost, so we have a 0.23% loss ratio for those control messages and the

Slide 12: Final transfer statistics

And the log also continues with some transfer statistics.

Slide 13: Wireshark: UDP conversion

Now, I took the tcpdump file and loaded it into Wireshark, and we can look at the traffic in terms of the data from my client located at this address at KTH and this port number to the server at that address and port number, and we see the number of UDP datagrams sent by the client was 703,728 packers and this total number of bytes were transferred. Now, we can break that down into packets from the client to the server and for the server to the client, and we see that there were very few messages sent from the client and a very large number packets sent from the server to the client. We see the amount in bytes, and we see the bit rates. We see that the client is sending at a very, very low bit rate. In total, we saw that the entire gigabyte file was transferred.

Slide 14: 33001 is the source UDP port of server 42132 is the source UDP port of the client

Now, if we do a plot looking at packet rates. You can see that the 33001, which is the source port of the server, here's the rate at which it is sending packets. So sometimes it decreases but mostly it stays along at about that rate. And in comparison, we can see down here the UDP traffic sent by the client at a very very low rate.

Slide 15: 33001 is the source UDP port of server 42132 is the source UDP port of the client

So let's zoom in on that a little bit. What do we notice here? We noticed that the rate that the server sends is initially slow; it linearly ramps up until it is sending out this higher rate, but

every once in a while, there are examples such as this, and here and here where the rate it is sending at is much lower. And we know that, of course, it is having to compete for some traffic with other nodes who are sending traffic along the same path.

Slide 16: Number of UDP packets from source ⇒ client since last UDP packet client ⇒ source

So I got interested and said, Hum! Is there a pattern and I said, Well obviously the client is sending packets back to the server once in a while to give it information and then the server is deciding how many packets it should send because it wants to send packets at no faster rate than they successfully make it there, if packets are lost then it needs to reduce its rate. So here's the relative time between when the last UDP packet was sent from the client to the source and the number of UDP packets sent from the source to the client. And we see as expected that initially, they're both exchanging packets and the number of that are being sent by the server is slowly increasing until it reaches this plateau, and mostly it sends along at that rate, but sometimes it sends faster and we can see these examples of cases where it's such a burst of packets right after the clients had sent a UDP packet to it. And then we see this interesting looking what looks like a mess down here. Right. But if we actually look at that we can start to see "hum" there is a pattern to that.

Slide 17: Connecting the points with lines to see their order

Well, it becomes much clearer whenever we plot the points interconnecting them with lines as we saw once in an earlier module, and this shows the sequence of the packets being sent. So you can actually see that most of the time, once again, the packets are being sent with these long bursts, but it's very clear that there's a pattern to these smaller bursts. So the question is: What's the pattern?

Slide 18: Zooming in more we can see very periodic behavior

Well, if we zoom in on that, you start to see some very periodic behavior; we see this occurring rather periodically. Here's the case for the first 330 bursts.

Slide 19: Is the burst length periodicity a result of sever using different periodic processes that are out of phase wrt each others?

So could we explain that? Well, if I remove the lines that are connecting them.

Slide 20: Two sinograms – out of phase with each other – shaping the short burst lengths?

I start to see a pattern that I recognize. And this is a pattern that is basically very commonly used; it is an "eye pattern" whenever you have two different phase sine waves that are interacting - such as you might see in the case of a transmitter where you have one signal in-

phase and another signal out of phase. So we actually compute - if we can fit these two sine waves - so we see the "a" component here, and we see points following along that, and we see the "b" component here delayed a hundred eighty degrees out of phase with it. We can see the scale of them, they each have a scale of about 410. And one of the fascinating things is if we add together the sine and cosine, what do we always get? We get one. And so if you multiply that one times 410 plus 410 sure enough that gives us a rate that's just under the rate at which it is regularly sending these bursts - in the lengths of sending these burst and of course this leads us to hypothesize that these ones that are longer bursts are probing to see if we have more bandwidth. While these two other phase signals are basically trying to transfer UDP messages from the server to the client but by using different burst lengths we are able to see: Are the queues along the way nearly full or not? So are we causing congestion or not? Because if we send a large burst of UDP datagrams and the queue is quite full, of course, some will fall off the end; but we just wasted network resources sending those packets because of course they're not making it to the final destination, and therefore we send a smaller burst of packets which is going to make it to the destination but in order to find out what the actual fill level of all of these queues along the way, we need to be probing with longer and shorter bursts and that - I believe this is what we can see from this diagram.

Slide 21: Source to client UDP 4 different datagram sizes

We can also see that there are four different sizes of UDP datagrams sent: some small ones at 42 bytes long, very few of those only 5 of them, 83 forty-six bytes long, and 20 other seventy-eight bytes long, but most of them are fourteen hundred seventy-two bytes long. So those are 1472 bytes plus our header, and that's, of course, sending the bulk of the data as we expect to transfer in the file. Now.

Slide 22: When are these shorter UDP datagrams sent?

When are these shorter UDP datagrams sent? We can do a plot of time here's the transfer time along the X-axis, and we can see the band showing that they're really fairly regularly sent. And we see here that the square boxes are the server to the client whereas the triangular - Oh sorry! The diamond shape boxes are the client to the server. And we see that they are very regularly sent but looking at this information we might be able to vary our traffic and get a better sense of exactly what they are doing in their protocol.

Slide 23: There is also TCP traffic

We also note that in addition to this UDP traffic there is some TCP traffic and we can suspect that is probably control traffic only - as the client to the server traffic the length of those frames the IP packets, and we see the length [of packets] from the server to the client, and we see those lengths as a function of time, and we also notice that the spacing is very very uniform at 5.196 seconds between these TCP bursts. So, some of you were taking a look at the FASP protocol, and I hope you will find further insight but plotting your data varying the parameters and getting your data to talk to you.

Slide 24: Another experiment: Sending packets through chains of network functions

Now we will take a look for just a few moments at another experiment. In this case, the student is sending packets through a chain of network functions. The reason is that the student wants to understand what is the performance when we take multiple network functions, such as we might use in network virtualization, where we might have a NAT device feeding into a router feeding into a load balancer, etc. So he takes the simple case of one network function, then the case of two network functions where we have one of these followed by another one and then repeating that up to the case we seven network functions in a row. And then he looks at what is the cache miss rate in the processor. So he has all this running inside one particular core on one particular module of a multi CPU system.

Slide 25: Importance of visualizing outliers

And he plots it and then tries to fit some curves to it. And here we can see the data showing which colors correspond to which algorithms and which cache (the layer one cache and the layer three cache) but if we zoom in a little bit this day was provided by Georgios Katsikas, a doctoral student at the network systems lab here at KTH

Slide 26: Importance of visualizing outliers

By zooming in we can see the [miss rate of the] level one cache as a function of the life of the length of the chain for three different packet sizes, and he has spaced the data from the three different packet sizes apart, and he's put them in different colors and used different symbols. And now, we can see here the box plot of data from the two different data plane implementations, and we can see some outliers. And this is really fascinating because we can see these outliers down here, which are nearly an order of magnitude lower cache miss rate.

Slide 27: Importance of outliers

And what does that tell us? As soon as I saw this data, I said, "hold on just a minute" these outliers are getting lucky somehow as they were being processed; they were all in memory, and so we didn't have any misses of the cache. And so, of course, an obvious question becomes: How did they get lucky? So not only the layer one cache but if we zoom in and look at the layer three cache, we also see the case down here that there are a number of cases for which (yes) these packets got lucky so even though the median and the first and third quartiles are way up here and of course the ones that got unlucky that had very high miss rate - these got lucky. And then looking into the "**why** of this", lead Georgios and others, including myself and his advisor Dejan Kostic - to dig deep into exactly what is going on. Because, of course, what we would like to do is move the median performance down here to this level so that in fact all the packets or at least most of the packets are getting lucky and actually experience a low cache miss rate - this is one of the major reasons why it's very important in your box plots to pay attention to your outliers. Georgios initially thought, "Ah! I should just ignore those - as those are due to some sort of experimental error - probably and

aren't really important" - where in fact, they turned out to be the most important data points in this particular plot.

Slide 28: Lots of sources for more information

Now, there lots more sources of information you might look at Ray Lyons "Best Practices in Graphical Data Presentation" And Dona Wong, who has written a book called, "The Wall Street Journal guide to information graphics the do's and don'ts of presenting data facts and figures". And I should point out that Edward Tufte was Dona Wong's thesis adviser. So not surprisingly, she's taken many of the things that she learned from Tufte and applied them in her role as the main person changing the way the Wall Street Journal presented its data visually.

Slide 29: References

There are lots more references to read. I certainly recommend William Cleveland's "Visualizing Data" and all the Tufte books.

Slide 30: ¿Questions?

Best of success in presenting your data visually and having your viewers understand the data that you have presented.