

# Inferring Short Time Scale Traffic Characteristics from Long Time Scale Measurements

Andrew Coyle

Teletraffic Research Centre  
The University of Adelaide  
Adelaide, South Australia  
Andrew.Coyle@adelaide.edu.au

*Abstract* It has long been known that short time scale traffic characteristics such as burstiness can directly impact queueing performance. However, it can be problematic obtaining detailed traffic measurements on such time scales from large, operational networks. More typically, data on 5 minute time scales, obtained via SNMP, is available. A technique that takes two independent sets of measurements of a single traffic stream, and from these measurements creates a single set of measurements at a finer time interval has been developed. The results from this analysis can be used to estimate short time scale characteristics of traffic in a network that cannot normally be measured directly.

## I. INTRODUCTION

Data are often polled from routers in a network on a periodic basis, for example, via SNMP at 5 minute intervals. With respect to traffic loads, SNMP will collect data on all links from all routers, thereby providing two views of traffic carried on a given link, namely, the incoming and outgoing traffic volumes reported by the routers at either end of the link. SNMP data is usually collected via a polling mechanism, and hence the traffic data collected from each end of a specific link may not be synchronised. Being 5 minute measurements, any characteristics of the traffic using the network on a finer time period are not directly available. Whilst at first sight it may appear to be impossible to get information for finer time periods, we can use the fact that traffic is measured both at both ends of the link, and also that these sets of measurements are not synchronised, to determine traffic at sub 5-minute intervals. This can be achieved by observing that each 5-minute period can be split up into two sub 5-minute intervals, and the amount of traffic carried in both of these intervals can be determined. Using a single day of 5-minute measurements we are creating 560 sub 5-minute intervals from two sets of 280 5-minute measurements.

In this paper a technique for generating and analysing sub 5-minute data is described and a simulation used to help explain and validate the technique.

## II. THE BASIC TECHNIQUE

In the following the time interval between  $v$  seconds after midnight and  $w$  seconds after midnight will be represented as  $t=(v, w)$ .

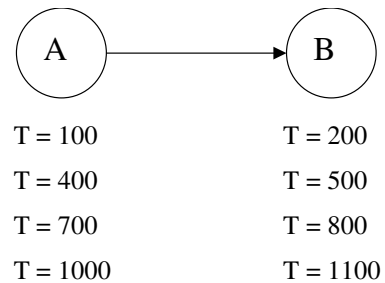


Figure 1. A network link with sample measurement times.

Consider the network link shown in Figure 1. Traffic originating in router A ends up at router B. Both of these routers are measuring all the traffic that enters and leaves all of the router's ports hence the traffic from router A to router B is measured at 2 locations. The polling time between individual measurements at a router are typically about 300 seconds, so that 5-minute data are easily measured and analysed. However when the measurements of the two routers on any link are not synchronised extra, sub 5-minute data, can be determined. Note that if the measurements are synchronous then they will be exactly the same and no extra information can be found. For the purposes of this analysis traffic between any two routers is considered to be instantaneously transferred across the link. The speed of the link means that the link transit time is substantially smaller than the recording intervals; hence this assumption is a reasonable one. Therefore there are 2 measurements of the data for any link, recorded every 5 minutes from both ends of the link.

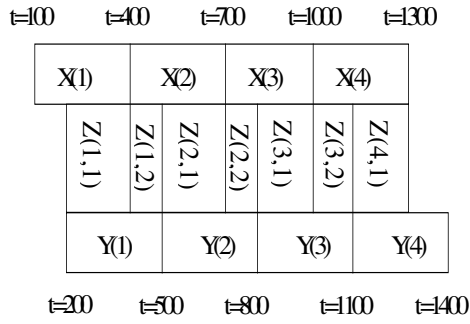


Figure 2. The measurement time intervals

Suppose we have  $n$  periods in the day. We will assume that router A in Figure 1 measures the amount of traffic that leaves for router B at  $t=100, 400, 700, 1000\dots$ , the amount of traffic carried in these 5-minute intervals are  $x(k), k = 1, n$ . Router B records this traffic when it arrives at  $t = 200, 500, 800, 1100\dots$ . The amount of traffic carried in these 5-minute intervals are  $y(k), k = 1, n$ . We will endeavour to determine the amount of traffic carried on link A-B in the intervals  $t=(100,200), t=(200,400), t=(400,500)$  and so on. This traffic will be denoted  $z(k,1), z(k,2), k = 1, n$ . These measurement intervals are shown in Figure 2. The problem could then be stated as:

Find

$$z(k,1), z(k,2), k = 1, \dots, n$$

such that

$$z(k,1) + z(k,2) = y(k), k = 1, \dots, n$$

and

$$z(k,2) + z(k+1,1) = x(k+1), k = 2, \dots, n.$$

If we know  $z(1,1)$  then the problem is easily solved since

$$z(1,2) = y(1) - z(1,1)$$

$$z(2,1) = x(2) - z(1,2)$$

$$z(2,2) = y(2) - z(2,1)$$

and so on.

It remains to find  $z(1,1)$ . In the present example this involves estimating the amount of traffic in the interval  $t=(100,200)$ . To do this it is important to realise how any error in estimating  $z(1,1)$  will propagate through all of the calculated values in the analysis. Suppose that  $z'(1,1)$  is an approximation to  $z(1,1)$  with an error of  $\epsilon$  so that  $z(1,1) = z'(1,1) + \epsilon$ . We can find approximations for the other measurements as follows

$$z'(1,2) = y(1) - z'(1,1)$$

$$z'(2,1) = x(2) - z'(1,2)$$

$$z'(2,2) = y(2) - z'(2,1)$$

and so on.

Then we can easily see that

$$z'(1,2) = y(1) - z'(1,1) = y(1) - z(1,1) + \epsilon = z(1,2) + \epsilon$$

$$z'(2,1) = x(2) - z'(1,2) = x(2) - z(1,2) - \epsilon = z(2,1) - \epsilon$$

and

$$z'(2,2) = y(2) - z'(2,1) = y(2) - z(2,1) + \epsilon = z(2,2) + \epsilon$$

In other words the absolute error for all the other approximate values is the same but with alternating sign. In our example assume that the actual amount of traffic in the first period,  $t=(100,200)$ , is 100 Megabits, and suppose that we have incorrectly calculated this value to be 90 Megabits, so that our calculated value is 10 Megabits less than the actual value. Then when we calculate the traffic in the period  $t=(200,400)$  the calculated value will be 10 Megabits higher than the actual value. We know exactly the traffic in the period  $t=(100,400)$ , have underestimated the traffic in the sub-period  $t=(100,200)$  and so must now overestimate the traffic in the period  $t=(200,400)$ . This logic continues for the entire day, the absolute error in the traffic value for any period neither diminishes nor increases, it is a constant 10 Megabits - alternating between overestimating and underestimating the correct value. A technique to determine this first value accurately can be derived from the problem that eventuates when this value is inaccurate.

It is not unrealistic to assume that the average rate of traffic over all of the even periods should be the same as that for all of the odd periods. Note that if this is not the case then the traffic has some interesting periodic properties and the analysis will be invalid if this was to occur. Therefore we should choose the amount of traffic in our initial interval so that the average traffic rate over all of the odd periods is the same as that for the even periods.

Suppose that the initial guess for  $z(1,1)$  is  $z'(1,1)$ . One way for getting this value is to assume that the traffic  $x(1)$  is uniformly distributed so that

$$z'(1,1) = \frac{x(1)T(z(1,1))}{T(x(1))}.$$

Using this first guess we can then easily find approximate values for the rest of the traffic flows using the recursive process described above. The average rate over all of the odd periods is

$$\sum_k \frac{z(k,1)}{T(z(k,1))}$$

and the average rate over all of the even periods is

$$\sum_k \frac{z(k,2)}{T(z(k,2))}.$$

If we assume these two rates are equivalent then

$$\sum_k \frac{z(k,1)}{T(z(k,1))} = \sum_k \frac{z(k,2)}{T(z(k,2))},$$

so that

$$\sum_k \frac{z'(k,1) + \mathcal{E}'}{T(z(k,1))} = \sum_k \frac{z'(k,2) - \mathcal{E}'}{T(z(k,2))}$$

where  $\mathcal{E}'$  is an approximation to the actual error  $\mathcal{E}$ . Rearranging this we get

$$\begin{aligned} \sum_k T(z(k,2)) \left( \sum_k z'(k,1) + n\mathcal{E}' \right) = \\ \sum_k T(z(k,1)) \left( \sum_k z'(k,2) - n\mathcal{E}' \right) \end{aligned}$$

and so

$$\mathcal{E}' = \left[ \frac{A - B}{n(\sum_k T(z(k,1)) + T(z(k,2)))} \right]$$

where

$$A = \sum_k T(z(k,1)) \sum_k z'(k,2)$$

and

$$B = \sum_k T(z(k,2)) \sum_k z'(k,1).$$

Therefore we can approximate the error in the  $z'$  values by equating the two rates. From this a better approximation to the  $z$  values can be easily found. The accuracy of our results depends primarily on the accuracy of our assumption that the rates over the odd and even periods are the same. If this is true then  $z(1,1) \approx z'(1,1) + \mathcal{E}'$  and the rest of the calculated results will also be accurate. This will be demonstrated using a simple simulation.

### III. SIMULATION

A simulation of a link with traffic will enable us to determine how well the technique works. Consider a single network link. We shall simulate a stream of traffic from the origin to the destination with the traffic calculated at one-second intervals. Traffic measurements will be created at the origin and the destination every 5-minutes, with the origin and destination measurement times offset. For this theoretical example, while we know the exact results for the sub 5-minute intervals we can also calculate them using the technique described above. We can then compare the exact results with the calculated results to determine the efficacy of our procedure.

There are many models of traffic that we can use for this simulation. To test this technique it is not necessary to use a complicated accurate model of the actual data on the network – although if one existed then it could be used. Instead we will use a simple model that nevertheless leads to some interesting results and helps validate the proposed algorithm. We shall assume that there are multiple sources of traffic. A new source of traffic appears each second with probability  $q$ , and a source of traffic stops every second with probability  $p$ . The time that a traffic source is on is geometrically distributed with the average time  $1/p$  seconds. Each source emits a fixed amount of traffic in a single second, if this amount is 1 Megabit and there are  $n$  sources on a given second then the amount of traffic on the link is  $n$  Megabits. It is easily seen that the average traffic rate is  $q/p$  Megabits per second. This traffic model has enough variation in the traffic rates to be able to validate our method.

### IV. SIMULATION RESULTS

We now use the theoretical traffic process described above, considering the case when  $p = 0.01$  and  $q = 0.1$ . On average there are 10 sources on, each source lasts an average of 100 seconds and if when on each source emits 1 Megabit of traffic each then the average traffic carried on the link is 10 Megabits per second. A trace of the traffic created by measuring this process in 5 minute intervals is shown in Figure 3.

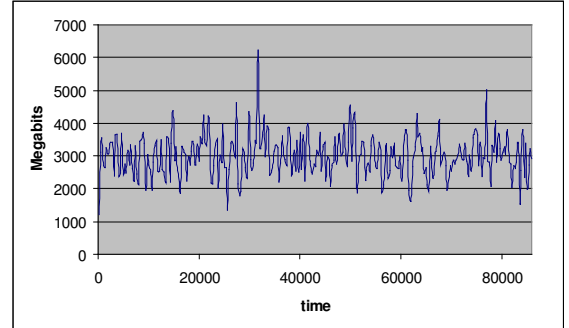


Figure 3 :The On/Off time series

Let the traffic at the source be measured at  $t=100, 400, 700, \dots$  and the traffic at the destination be measured at  $t=200, 500, 800, \dots$ . This means that the intervals for which we calculate the flows are between the times  $t=100, 200, 400, 500, 700, 800, \dots$  so that the intervals' lengths are alternately 100 and 200 seconds. These are the sub 5-minute intervals that we are inferring from the 5-minute observations we are given. Some results are shown in Table 1. These show the start and end times of the measurement intervals, the exact traffic rates for this period, results from the algorithm with an incorrect initial interval value, and results from the algorithm using the average rate equalising procedure. As discussed earlier when there is an error in the first period's value, all of the remaining periods are also in error. The error alternates in sign, and the rates shown in this table show, the size of the error is proportional to the size of the measurement period. In this case the error in the rate is alternately, 0.738 and then

1.477. When the equalising process is used the exact and algorithm values are much closer – the average percentage error is less than 0.2. The results are not exact since the assumption that the 2 average rates are equivalent is not strictly true, they will be out by a small amount. In the case where they are out by a large amount the analysis will be invalid, although this indicates fairly unusual traffic behaviour.

TABLE I. TABLE 1 :SIMULATION RESULTS SHOWING THE EXACT AND CALCULATED LINK RATES.

<i>Start Time</i>	<i>Finish Time</i>	<i>Exact</i>	<i>Algorithm - no equalising</i>	<i>Algorithm - equalising</i>
200	400	10.545	9.806667	10.55643
400	500	11.94	13.41667	11.91713
500	700	8.28	7.541667	8.291434
700	800	8.38	9.856667	8.357133
800	1000	8.87	8.131667	8.881434
1000	1100	11.89	13.36667	11.86713
1100	1300	11.585	10.84667	11.59643
1300	1400	9.78	11.25667	9.757133
1400	1600	8.97	8.231667	8.981434
1600	1700	9.43	10.90667	9.407133
1700	1900	8.31	7.571667	8.321434
1900	2000	11.68	13.15667	11.65713
2000	2200	14.005	13.26667	14.01643
2200	2300	5.87	7.346667	5.847133
2300	2500	10.03	9.291667	10.04143
2500	2600	15.2	16.67667	15.17713
2600	2800	13.25	12.51167	13.26143
2800	2900	12.68	14.15667	12.65713
2900	3100	8.755	8.016667	8.766434
3100	3200	9.19	10.66667	9.167133

The simulation has also been run for the case when the times that the measurements are made at are  $t=200,499,500,799,800,\dots$ . In this extreme scenario we can determine results for intervals as small as one second. Results are shown in Table II. As can be seen the results are about 99% accurate. Due to the traffic model we are using the distribution of the traffic for the single second periods should be approximately Poisson, although only approximately due mainly to correlation effects over time. We have managed to get data for very small time intervals accurately using only the 5-minute data. In practice it would need to be careful when applying this technique to intervals this small due to errors in recording times, however if the data are accurate then inference can be made at this fine period.

TABLE II. SIMULATION RESULTS SHOWING THE EXACT AND CALCULATED RATES.

<i>Start Time</i>	<i>Finish Time</i>	<i>Exact</i>	<i>Algorithm - no equalising</i>	<i>Algorithm - equalising</i>
200	499	11.01338	11.00667	11.01302
499	500	10	12.00667	10.10769
500	799	8.307692	8.300981	8.307332
799	800	10	12.00667	10.10769
800	1099	9.866221	9.859509	9.865861
1099	1100	13	15.00667	13.10769
1100	1399	10.98662	10.97991	10.98626
1399	1400	10	12.00667	10.10769
1400	1699	9.120401	9.11369	9.120041
1699	1700	10	12.00667	10.10769
1700	1999	9.414716	9.408004	9.414356
1999	2000	15	17.00667	15.10769
2000	2299	11.29766	11.29095	11.2973
2299	2300	10	12.00667	10.10769
2300	2599	11.74916	11.74245	11.7488
2599	2600	13	15.00667	13.10769
2600	2899	13.06689	13.06018	13.06653
2899	2900	11	13.00667	11.10769
2900	3199	8.899666	8.892954	8.899305
3199	3200	9	11.00667	9.107692

## V. OPERATIONAL APPLICATIONS

Data offsets may be ad hoc, possibly due to the time routers are started or stopped, so there may be no pattern to the measurements obtained from a network. However if the measurement time periods can be controlled then periods of particular interest could be analysed. So for example if the traffic characteristics at 15 second duration are of particular significance then by adjusting the times at which measurements are made we can get some 15 second data, 288 values on a link every day. These have been determined even though the router is still only measuring data every 5-minutes, which operationally may be the best that are available.

Alternatively a greater spread of measurement intervals may be possible if different routers measure data at different time periods. So for example if one router records data every 300 seconds and another every 315 seconds then a variety of time interval data will be available for the link between these two routers. In this case if the two routers start at the same time then data for intervals that are multiples of 15 seconds are available. The disadvantage with this is that the number of values obtained at any particular time interval would be reduced. For small data sets it is debatable whether enough measurements could be made at any interval for any conclusions to be reached with any degree of certainty. For larger data sets the spread of intervals may give the best results.

## VI. CONCLUSIONS

We have described and tested a technique for inferring data from finer period than these data are polled at. This technique has been described and results from a simulation presented. The results indicate that the technique is able to provide insight into small time scale data traffic characteristics using only coarse, long time scale data.

## ACKNOWLEDGMENT

The author would like to thank Anthony Utano and Tim Neame from Telstra for providing network data on which to test the procedure developed, and for their helpful discussions and advice about this work.