Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks[1]

Reviewed by
Ryan Marcotte and Ming Zhi Yu

Summary

This work proposes Sprout, an end-to-end transport protocol for interactive applications intended to promote high throughput with low delay over cellular links. Whereas many traditional transport protocols control transmission rates in a reactive manner, responding to packet loss events that have already occurred, Sprout seeks to set transmission rates proactively. It accomplishes this through an inference process that uses packet arrival times to estimate the state of the link, both at the current time and into the future. Critically, Sprout models the link as a stochastic process that is subject to uncertainty, allowing the system to incorporate varying amounts of conservatism depending on the needs of the application. In experiments conducted on packet traces collected from real-world cellular data, Sprout is shown to perform well compared to its competitors (TCP variants, commercial video calling protocols) in terms of both the delay induced by the system and the overall throughput.

Highlights

The first thing we like about the paper is that the protocol proposed by the authors adheres to the end-to-end argument principle mentioned in the first lecture. The benefits of doing so are evident. On one hand, such an end-to-end protocol obviates the need to make modifications to network equipment such as LTE infrastructure, baseband modems, IP-layer queues, etc. where queues might build up. For example, the authors give the example of CoDel and argue that changes would need to be made in the above-mentioned equipment in order for the protocol to monitor queues and emit a congestion signal. On the other hand, Sprout, as an end-to-end protocol, gives applications the freedom to set a balance between throughput and delay. CoDel fails to provide such freedom as it was shown by the authors that it slowed down a bulk TCP transfer which had a link to itself.

Another advantage of Sprout is its resilience to packet loss. As we know, packet loss is considered as a signal of congestion by a TCP-style congestion control mechanism. To react, TCP's congestion control

[1] Winstein, K., Sivaraman, A., and Balakrishnan, H., "Stochastic forecasts achieve high throughput and low delay over cellular networks," Proc. of the 10th USENIX Conf. on NSDI, pp. 459-472, 2013.

would narrow the sender window size and sometimes such a slowdown would reach an unacceptable level. In contrast, experiments carried out by the authors demonstrated that even under a 10 percent one-way packet loss, Sprout still has a relatively high throughput.

In addition to the advantages of Sprout, we appreciate some novel methods used by the authors when they designed the protocol and when they conducted the experiment. For example, we like the idea of constructing a doubly-stochastic model based on packet arrival times and making a cautious forecast on future packet deliveries. By constructing a mathematical model, the authors transform such an abstract concept as link capacity into something which could be quantified and which could be used to make probabilistic inference. The idea of using the 5th percentile to make cautious forecasts also impresses us. At first, we did not understand why the authors made such a conservative forecast. However, after seeing the comparison between Sprout and Sprout-EWMA, we understand that such a specially designed forecast mechanism is fundamental in reducing the average delay by 200ms. And for interactive applications which are sensitive to packet delays, such a difference in delay could affect user experience.

We also like the idea of using trace data to drive the experiment. One of the unique characteristics of a cellular link is its varying capacity which causes the bufferbloat problem and motivated the authors to design Sprout. As a result, it is of great importance to have such characteristic preserved during the experiment to see if the new protocol solves the problem or not. By collecting the trace from two smartphones while driving around a city, the authors made sure the settings of the experiment were as close to a real world experiment as possible, while still allowing for offline simulation of the system. Further, the fact that the trace is a ground truth allows us to visualize the direct cause, which is the overshooting of link capacity, to the substantial delay experienced by some other interactive applications. Although we can still do a side-by-side comparison on average delays without the ground truth, having this ground truth really helped us to understand the effect of bufferbloat (see Figure 1 of the paper).

One final thing we commend the authors for is their distribution of an open-source implementation of their system along with their published paper. Making research software available in this manner has many benefits, including aiding in reproduction and extension efforts. Furthermore, for a system such as Sprout that appears to have real-world utility, the open source software could be developed further and used in actual applications.

Improvements/Extensions

The authors make the task of critiquing the deficiencies of their work more challenging as they include a substantial treatment of their own observations of their technique's shortcomings. Such an effort is commendable as it helps other researchers direct future inquiries and appreciate the contributions of the presented work. We also feel that such an honest acknowledgment of a work's limitations would actually make for a stronger paper in the peer review process.

Several times, the authors reference "reactive" techniques that use metrics such as packet losses, one-way delays, and round-trip delays in order to adjust transmission rates. Their objection to such an approach is its reactive nature that fails to anticipate changes to the network that might affect the rate at which the system should send data. This shortcoming of the other techniques is certainly valid, but the authors seem to conflate this reactive approach with the metrics they use. Packet loss and delay are not inherently tied to a reactive approach, just the same as the authors' preferred metric of packet arrival times. It is well known that estimating link quality or capacity, especially into the future, is very challenging. Perhaps this work could utilize more of the metrics that might be available to it (packet loss, delay, or even signal strength in a wireless network) to make better estimates. The current system has the merit of being relatively simple, but incorporating additional features would likely lead to a better predictor and result in better overall performance.

This work models the link as a doubly-stochastic process, which the authors argue is a reasonable and principled choice. Though we do not contest their arguments or the selection of this model, an interesting extension to this work might be to explore alternative ways of modeling the link to see what results in the best predictions and performance. The authors could have aided this sort of future work by providing an evaluation of how well their proposed model and parameters captured the datasets they collected. Furthermore, it would be useful to have insight into how the parameters for these models could be tuned in a principled manner. The authors refer to an empirical selection process they used initially to settle on parameters for their model. Though this process is perfectly acceptable in the context of this paper, it seems that this parameter estimation could be improved by using a data-driven or statistical approach.

We commend the authors' inclusion of Sprout-EWMA as a point of comparison to the proposed method, as it helps to lend insight into the utility of the more complex inference process used in Sprout. Unsurprisingly, the results show that Sprout-EWMA generally provides higher throughput while the more conservative Sprout performs better on self-inflicted delay. The intriguing aspect of these results comes as the authors vary the confidence parameter in Sprout, which gives them a mechanism for trading delay for throughput. Though the curve that this parameter sweep yields has a predictable shape, it is interesting to note that it fails to achieve the simultaneous performance along both axes of Sprout-EWMA. The authors note this result without much explanation, leaving that to future work. We posit that perhaps this could be explained by weaknesses in the model the authors use for the link, particularly in its estimates of uncertainty. Performing inference over future link states using estimated uncertainty does seem to have great potential, but it is not apparent that Sprout really is superior to the much simpler Sprout-EWMA. While Sprout does yield lower delay than Sprout-EWMA, this could easily be due to the conservative nature of the former. If Sprout-EWMA were modified to be more conservative (e.g. by biasing its estimates in the conservative direction), perhaps it would outperform Sprout along the delay metric as well. Indeed, this does seem to be an interesting avenue for future work in order to better understand the characteristics of the inference-based approach of Sprout.

The authors observe that the datasets they utilized in their work did not exhibit significant data loss, though this characteristic is not true of cellular or wireless links in general. To test the performance of Sprout under lossy conditions, the authors intentionally induce simulated Bernoulli packet losses of 5 and 10 percent in some of their experiments. The results of these tests are promising showing that Sprout can continue to provide good throughput even in the face of packet loss, something that is not true of TCP in general. What is not clear from the results is how much packet loss Sprout can tolerate before it too breaks down. Ten percent packet loss is significant for some applications, but there are others in which packet loss may occur at even greater rates. For example, in a link exhibiting 20 or even 50 percent packet loss, would Sprout fail to function entirely? Or can it tolerate arbitrarily large amount of packet loss?

In some ways, Sprout bears resemblance to an adaptive forward error correction system Ryan has previously developed[2]. In that system, a modified form of packet interleaving is performed on groups of packets in order to yield redundancy packets that are transmitted along with the normal data packets. Any subset of the redundancy and data packets can be used by the receiver to recover the transmitted data packets. The system modifies the amount of redundancy to encode in an online fashion based on the estimated state of the link, both at the time of transmission and in the future. It also uses an estimate of the uncertainty of the prediction to make more conservative selections when uncertainty is large. Whereas the goal of Sprout is to provide high throughput and low self-inflicted delay, the adaptive FEC system seeks to maximize the packet delivery ratio. Of course, the use case for the two systems is slightly different: Sprout operates on saturated links while the adaptive FEC system requires that the data packets

[2] Marcotte, R. J., & Olson, E. (2016, May). Adaptive forward error correction with adjustable-latency QoS for robotic networks. In *2016 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 5283-5288). IEEE.

themselves leave enough link capacity to be able to include redundancy packets. However, a hybrid based off of the two systems could perform quite well for certain applications. For example, in Ryan's semester project on image streaming over lossy wireless links, a combination of Sprout and adaptive FEC could be effective. The forward error correction system would help boost the packet reception ratio while Sprout controlled the transmission rate. If implemented in a complementary fashion, such a system may result in better performance at the application than either would be capable of individually.