Can Network Coding Mitigate TCP-induced Queue Oscillation on Narrowband Satellite Links?

Ulrich Speidel[1(✉)], Lei Qian[1], 'Etuate Cocker[1], Péter Vingelmann[2], Janus Heide[2], and Muriel Médard[3]

[1] Department of Computer Science, The University of Auckland, Auckland, New Zealand
{ulrich,lqia012,ecoc005}@cs.auckland.ac.nz
[2] Steinwurf ApS, Aalborg, Denmark
{peter,janus}@steinwurf.com
[3] EECS, Massachusetts Institute of Technology, Cambridge, MA, USA
medard@mit.edu

Abstract. Satellite-based Internet links often feature link bandwidths significantly below those of the ground networks on either side. This represents a considerable bottleneck for traffic between those networks. Excess traffic banks up at IP queues at the satellite gateways, which can prevent conventional TCP connections from reaching a transmission rate equilibrium. This well-known effect, known as *queue oscillation* can leave the satellite link severely underutilised, with a corresponding impact on the goodput of TCP connections across the link. Key to queue oscillation are sustained packet losses from queue overflow at the satellite gateway that the TCP senders cannot detect quickly due to the long satellite latency. Network-coded TCP (TCP/NC) can hide packet loss from TCP senders in such cases, allowing them to reach equilibrium. This paper reports on three scenarios in the Pacific with two geostationary and one medium earth orbit connection. We show by simulation and circumstantial evidence that queue oscillation is common, and demonstrate that tunneling TCP over network coding allows higher link utilisation.

Keywords: Queue oscillation · TCP · Network coding · Satellite links

1 Introduction

A large number of locations around the world rely on satellite links as their only feasible means of connecting to the Internet. Traditionally, this connectivity used to be supplied by geostationary (GEO) satellites, although in the last few years, a medium earth orbit (MEO) satellite operator has entered the market [1]. In practice, this means a connection with long round trip times (typ. >500 ms for GEO, or >125 ms for MEO) and generally expensive and hence often narrow bandwidth. Even though the advent of MEO service has brought some relief

P. Pillai et al. (Eds.): WiSATS 2015, LNICST 154, pp. 301–314, 2015.
DOI: 10.1007/978-3-319-25479-1_23

in this respect, many small and medium-sized communities cannot afford bandwidths comparable to those that connect to the satellite gateways at either end of the link.

The resulting bottleneck presents a significant challenge to the Internet's staple transport protocol, TCP [2], leading to both bandwidth *underutilisation* and slow or even stalling data transfers. These problems have been discussed in the literature for many years [3–5], leading to the development of TCP variants for connections involving wideband satellite links [6,7]. However, these do not necessarily work well in the common narrowband scenarios, where the link is shared with conventional TCP varieties and widespread deployment of satellite-friendly TCP variants is not a feasible option.

This paper presents two main results: Firstly, we show by simulation that native TCP behaviour of multiple parallel TCP flows suffices to produce the effects observed in three satellite-connected Pacific Island locations. Secondly, we show by experiment that network coding of some TCP flows into these locations can result in better goodput in many cases.

The next section discusses the principle behind queue oscillation. Section 3 considers how to model a satellite link and the traffic across it in a simulation, followed by a selection of simulation results in Sect. 4. We then discuss the network coding scheme used in our live experiment across the three actual satellite links in Sect. 5, and provide some experimental evidence on its effectiveness in Sect. 6.

2 TCP-induced Queue Oscillation

The Transmission Control Protocol [2] handles the bulk of Internet traffic. TCP relies on a family of complex flow and congestion control algorithms aimed at achieving reliability, high goodput rates and fair bandwidth sharing among multiple TCP flows. A common feature of all these algorithms is their use of acknowledgment packets (ACK) as feedback to the TCP sender. The sender infers from the ACKs whether packets have been lost or delayed, and how much data it may entrust to the channel with acknowledgments outstanding. Under ideal circumstances, this results in links that carry a maximum of goodput and a minimum of retransmissions.

Roughly speaking, a TCP sender starts data transmission at a slow packet rate to elicit ACKs from the receiver. As long as the receiver returns ACKs and indicates readiness to receive more data, the sender progressively increases the packet rate. This continues either until the receiver throttles the sender through its advertised window in the ACKs, or until packet loss in either direction disrupts the ACK sequence arriving at the sender. In the case of packet loss (missing ACKs), the sender responds with a more-or-less aggressive reduction in packet rate.

Packet loss can occur in the physical layer (interference, noise, fading, etc.) but also in network equipment (routers, switches, modems, ...) unable to buffer sufficient incoming packets in their input queue for dispatch on outgoing links,

dropping excess packets arriving for a full queue. In this paper, we argue that the latter effect suffices to explain the low performance reported across many satellite links.

In the case of satellite links, the satellite bandwidth is generally much smaller than that of the network infrastructure it connects to (typically fibre optic networks). Viewed from both ends of a TCP connection across the satellite, the sat link thus represents a bottleneck. The queue at which traffic banks up when the bottleneck gets congested is the transmission queue at the satellite ground station. We note here as an aside that the bottleneck bandwidth also tends to be the most expensive, meaning that the size of the bottleneck is generally severely constrained by economic factors rather than traffic demand dimensioning.

TCP's congestion control algorithms are of course designed to cope with bottlenecks, albeit those for which the algorithm can obtain quasi-realtime information. However, our satellite transmission queue is both the location at which congestion events are most likely to occur, and the location at which congestion relief through sender response must become effective. Information about packet drops generally propagates in the form of "non-events" via the receiver to the sender: A data packet dropped at the satellite queue subsequently fails to arrive at the receiver, which therefore does not emit an ACK for the packet. The ACK then doesn't return via the satellite link, and only once it is overdue at the sender, the sender will take corrective action by lowering its packet rate. The lower packet rate in turn does not become effective at the queue until the data held back by the sender eases the overflow situation at the queue.

That is, event and response are separated by a whole round-trip-time (RTT), which is particularly long in satellite links. In other words: The TCP sender on a satellite link always works with severely out-of-date congestion information. On links carrying multiple TCP flows, there is an additional problem: Multiple senders are all independently afflicted by the same issue. Sensing capacity on the link (=senders receive regular ACKs for packets that passed through the bottleneck some time ago), the senders *all* more or less simultaneously increase their congestion window and hence their sending rate. As the packet arrival rate at the queue is the sum of the (latency time-shifted) sender rates, this can considerably accelerate the rate at which the queue fills. Now note that this acceleration continues even after the queue overflows – it only ceases once the senders learn about the packet drops through the missing ACKs, and the resulting back-off only has an effect at the queue once the packets sent at the fast rates have all arrived there.

Similarly, when the senders detect packet loss and back off in response, the effect on the queue arrival rate is in effect multiplied by the number of flows involved: The flood now slows to a trickle. It remains a trickle until the senders receive ACKs again, one RTT later. This often allows the queue to drain completely, leaving the link idle. As the ACKs return again, the cycle may repeat.

This effect is known as *queue oscillation* and is well documented in the literature, not just for satellite links. However, satellite links are prime candidates for queue oscillation due to their long latencies and the severe bandwidth bottleneck they often represent.

In principle, we may distinguish three scenarios for a satellite link with multiple-sender TCP traffic:

- **Pre-oscillation:** Low traffic demand; existing TCP flows do not fill the bandwidth of the link and the queue does not overflow. Characteristic for this scenario are low packet loss and low link utilisation.
- **Oscillation:** The queue oscillates between entirely empty and overflow and reaches both extremes in the order of around 2–3 RTT. Characteristic for this scenario are high packet loss with recognisable bursts paired with low link utilisation. The high packet loss limits the goodput on individual connections even if the link as such has plenty of spare capacity.
- **Congestion:** The queue may oscillate between overflow and part-empty, but traffic demand is such that young and short TCP flows appear at a high rate. These flows, which have not responded to packet loss yet, provide sufficient traffic to the queue to prevent it from draining completely. Characteristic for this scenario are high packet loss with recognisable bursts paired with high link utilisation, a high number of concurrent flows, a significant number of old stalled flows, and low per-flow goodput.

The next section discusses how we can replicate these scenarios in a network simulator.

3 Modelling an "Island" Link

In the typical "island" scenario, the bulk of data flow is from the Internet to the island, with typical inbound-to-outbound bandwidth ratios of 4:1. Our simulations consider inbound payload data flow only. Our "satellite link" assumes a one-way delay of 240 ms for GEO and of 125 ms for MEO. On the Internet side, the router at the satellite gateway forms the root of a "binary" tree of depth 2 whose edges are links of $\geq$ 1 Gbps and whose leaf nodes are TCP senders with one-way latencies to the gateway of 11, 20, 60 and 80 ms, respectively. This arrangement lets us simulate a variety of latencies of similar orders of magnitude as one would expect in the real world. At the "island end", we use a single TCP receiver connected via a 1 Gbps link to the router.

The next challenge is to simulate the traffic. On real links, tools such as ntop/nprobe [11] and various others can measure the number of active hosts and/or the number of active flows seen during a certain time period. The number of active hosts does not yield the number of active flows, however, even if it permits estimates of the right order of magnitude. Real-world flows also tend to be short in terms of the size in bytes, and those active at one particular time usually have different start times and congestion windows.

This poses the challenge of getting the "flow mix" at least approximately right, as consideration of the extremes shows. Flows that are too small consist of only a few packets – too little to partake in oscillation over a whole cycle. Too few flows are bound to leave the link underutilised and will never fill the queue, so there will be no packet loss and no oscillation – our pre-oscillation

scenario. At the other end of the spectrum, in the congestion scenario, one creates new flows at a rate and/or of sizes that are too large. In this scenario, old flows will slow to a crawl because of the congestion and will not complete, while the new flows that have yet to experience packet loss ensure that the queue never empties: Here, we expect plenty of packet loss all the time, a permanently overflowing queue, 100 % link utilisation, and a very high combined goodput. However, because of the large and growing number of flows, the average goodput per flow tends to zero. It is between these poles that one expects queue oscillation.

To obtain a realistic flow size distribution, we assume that the island's Internet traffic is predominantly web traffic. For each new flow that we generate, we choose the byte volume of a flow randomly from the first 100,000 flows in a trace taken at the University of Auckland's border gateway in 2005, which consists of around 85 % web traffic. Figure 1 shows the flow size distribution thus obtained.

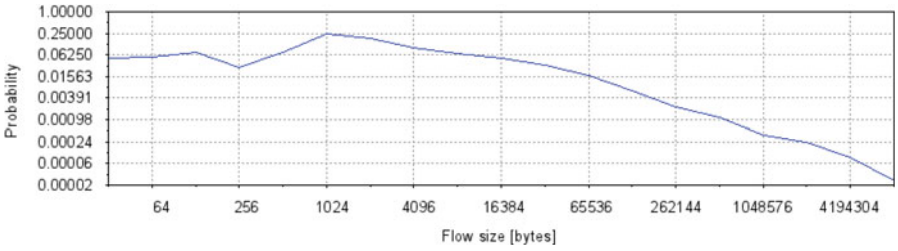

Fig. 1. Flow size distribution for our simulations.

In our simulation, we perform a "sweep" using an estimated flow creation rate. The number of concurrent flows at the beginning of each simulation is zero. It then either grows throughout the simulation or reaches a plateau. Simulations whose number of concurrent flows continues to grow inevitably head for congestion: Older flows stall, and the goodput rate per flow declines. If the number of concurrent flows stabilises, the link can be in either a pre-oscillation or oscillation scenario. We can distinguish between these two scenarios based on the absence or presence of burst packet losses. If the flow creation rate is very low, it is also possible that both the number of concurrent flows and the goodput rate per flow keep growing significantly throughout the experiment, a clear indicator of pre-oscillation.

Another parameter that is difficult to determine is the queue capacity at the satellite gateways, as this information resides with the satellite operators and does not necessarily get documented to their customers. Industry-standard routers typically provision queues for a few to up to several hundred packets, and we tried a number of values in this range. As a general rule, a low capacity queue encourages oscillation as it is easy to fill and easy to clear, whereas a high capacity queue tends to shift the scenario towards congestion.

4 Simulation Results

Our simulation concentrates on the three real-world Pacific Island locations we investigated: Rarotonga in the Cook Islands, Niue, and Funafuti Atoll in Tuvalu. At the time of deployment, Rarotonga had an inbound MEO bandwidth of 160 Mbps, which has since been upgraded to 332 Mbps, and Niue had 8 Mbps from GEO. The exact GEO bandwidth for Funafuti does not seem to be officially known, however our observations lead us to an estimate of 16 Mbps. Initial daytime observations showed that none of the links was in pre-oscillation mode: Rarotonga and Funafuti oscillated, whereas Niue seemed to have reached permanent congestion.

For the "Niue model", Fig. 2 shows that link utilisation reaches levels close to the maximum around 50 s into the simulation for a queue with a capacity of 50 packets and flows created once every 13 ms. In this scenario, almost all data on the link is goodput. An inspection of packet drops over the experiment time in Fig. 3 shows the high packet loss characteristic of the congestion scenario. Note however that lossless time intervals interleave regularly with those featuring high packet loss – a classic sign of queue oscillation. An event-based inspection of the queue length shows that the queue also frequently drains completely. Figure 4 shows that in this scenario, the number of concurrent flows and the instantaneous average goodput per flow reach a noisy equilibrium towards the end of the sweep.

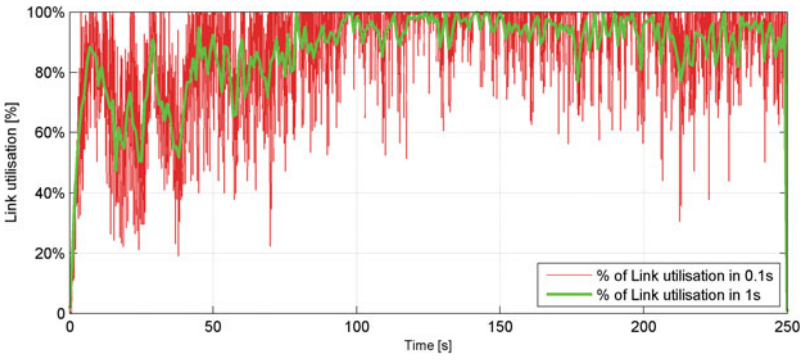

Fig. 2. Simulated link utilisation of the "Niue model". Note that the link is almost fully used after the initial ramp-up.

For the Rarotonga scenario, our initial deployment took place with an inbound satellite link capacity of 160 Mbps. A simulated queue with a capacity of 50 packets also produces oscillation for 160 Mbps if we create a new flow every millisecond. Figure 5 shows that this results in a link utilisation of around 60 %, while still producing significant packet loss. Figure 6 shows the bursty nature of the packet losses: During some time periods, the queue drops dozens of arriving packets, during other time periods it drops none. This effect is present during the entire duration of the experiment. The low link utilisation is evidence of complete queue drain, which we have also been able to observe directly on the queue.

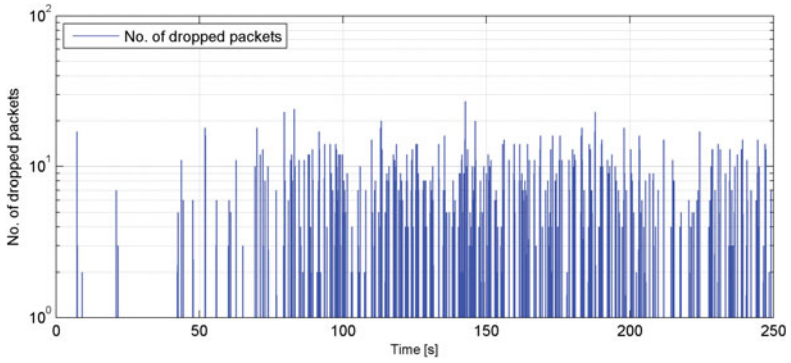

Fig. 3. Bursty packet drops in the simulated "Niue model".

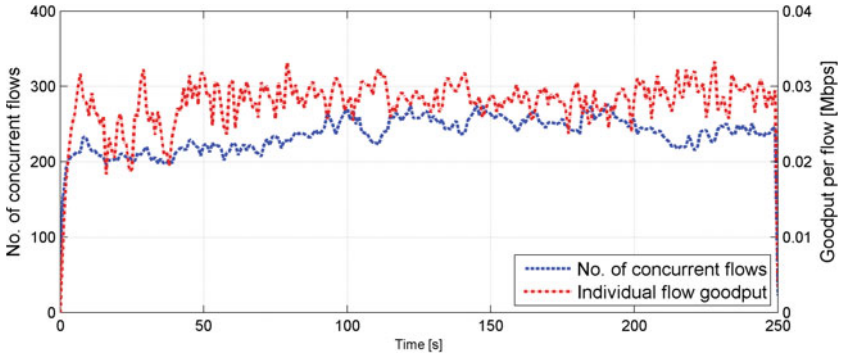

Fig. 4. Simulated number of concurrent flows and average per-flow goodput of the "Niue model": As new flows appear, older flows stall. Higher flow creation rates than the one used in this model lead to a steady rise in concurrent connections and a steady drop in goodput per flow.

Figure 7 shows that number of concurrent flows plateaus, showing that the link copes with demand. The effect of the queue oscillation here is simply a lengthening of flow durations caused by the packet losses.

If one "upgrades" the simulated Rarotonga link to the later value of 332 Mbps and leaves all other parameters (queue capacity, flow creation rate and flow distribution) identical, packet losses *increase* slightly. This is somewhat counter-intuitive, but a possible explanation is that the larger queue drain rate causes TCP flows to have larger congestion windows when the queue eventually overflows. This means that there are more packets "in flight" at overflow time that will subsequently be dropped. Goodput in this scenario remains almost unchanged, and the link utilisation predictably drops to around 30 %.

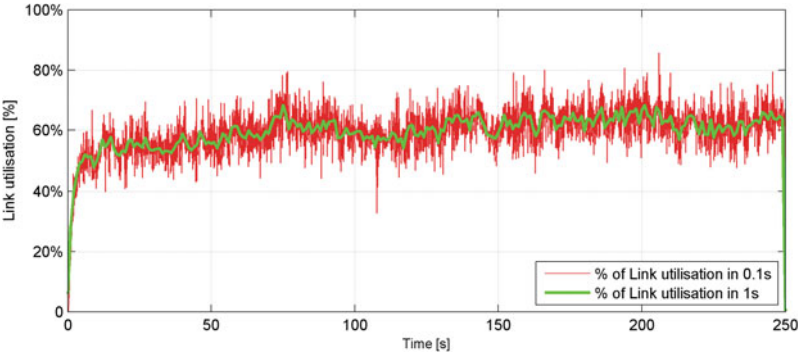

Fig. 5. Simulated link utilisation of the 160 Mbps "Rarotonga model".

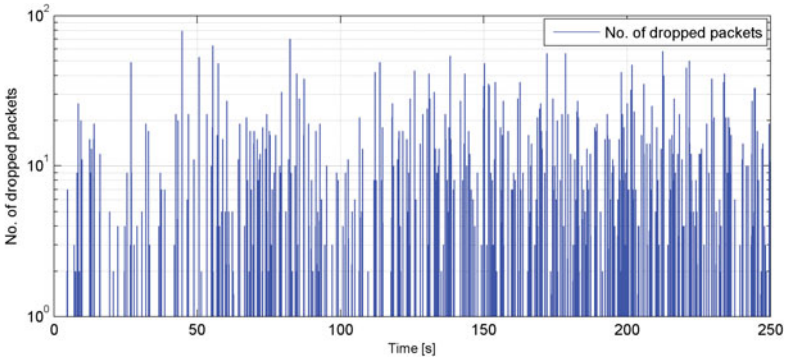

Fig. 6. Bursty packet drops in the simulated 160 Mbps "Rarotonga model".

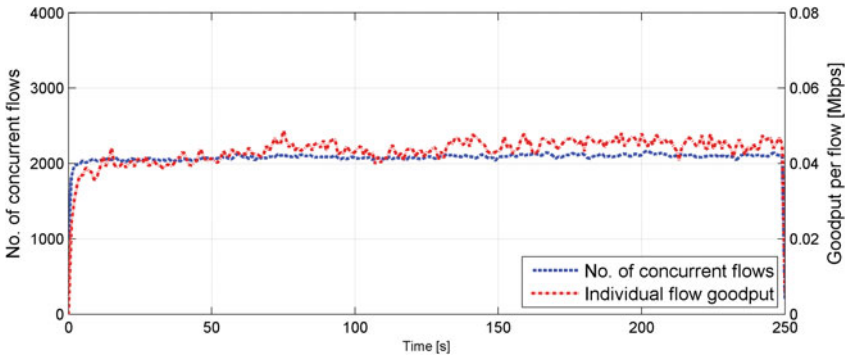

Fig. 7. Simulated number of concurrent flows and average per-flow goodput of the 160 Mbps "Rarotonga model".

5 Tunneling TCP over UDP with Network Coding

Network coding (NC) lets us mask a moderate amount of burst packet loss from the TCP sender and receiver. The type of network coding used in our study is known as *random linear network coding* (RLNC) [8,9].

For simplicity, we discuss the world-to-island traffic direction here only, the opposite direction works analogously. First, we identify an IP network X on the island side that is to benefit from FEC by NC. We then ensure that the off-island NC encoder G_w (which does not need to be placed at or even near the satellite gateway) is the IP gateway to X, i.e., all IP packets to X from anywhere on the Internet are routed to G_w. These packets carry TCP and any other protocols from the TCP/IP suite.

G_w intercepts the packets and groups them into *generations*. A generation is a set of n IP packets $p_1, p_2, \ldots, p_n$ which G_w then turns into $n + \omega$ linear equations with random one-byte coefficients $c_{i,j}$ such that the i-th equation in the set is:

$$\sum_{j=1}^{n} c_{i,j} p_j = r_i,$$

where $c_{i,j} p_j$ is the product of $c_{i,j}$ and each byte in p_j. G_w stores the respective results in each byte of r_i and transmits each such linear equation as a "combination" UDP packet to the on-island decoder G_i. The combination packet contains the $c_{i,j}$, r_i, and other information such as packet length. Without packet loss, G_i thus receives a system of linear equations overdetermined by up to ω equations, whose solution is the generation $p_1, p_2, \ldots, p_n$. G_i solves this system and forwards $p_1, p_2, \ldots, p_n$ to their original destinations in X. Note that more than ω packets need to be lost between G_w and G_i for the generation to become unrecoverable, thus providing FEC for loss bursts of up to length ω.

The UDP combination packets use G_w's IP address as origin and G_i's IP address as destination. Note that the latter, while on the other side of the satellite link, is *not* part of X, but part of a separate IP network Y whose traffic is routed via the satellite gateway. G_i thus needs two interfaces, one with a IP address in X and one with an address in Y. G_w also needs two interfaces: One that receives traffic from the Internet for X, and one that sends UDP combination packets to G_i.

In the reverse direction, G_w and G_i switch roles: G_i acts as the gateway for hosts in X, encoding traffic from X to the Internet into UDP combination packets heading for G_w. Similarly, G_w now decodes these combinations and forwards the packets from X to their destinations on the Internet. This thus establishes a UDP tunnel from the Internet to X and vice versa.

6 Experimental Observations

Our experimental results do not give an entirely uniform picture for the various links:

Niue: The actual scenario on the Niue link has been difficult to measure as the link is also the only electronic means of getting experimental data out. During our visit in December 2014, link utilisation was in excess of 90 % during the day, and packet losses were common. Our simulation indicates that around 200 concurrent flows may be realistic and that typical loss bursts do not exceed a few dozen packets. Distributing these losses across concurrent flows suggests that most flows are unlikely to lose more than one or two packets. This corresponds well to the fact that a generation size of $n = 10$ with $\omega = 2$ was able to mask most of the losses on the real link.

The link into Niue sees sustained peak data rates of around 7.5 Mbps with >7 Mbps recorded for much of the day. Individual conventional TCP connections with 5 MB downloads achieve around 0.3 Mbps. Closer inspection reveals that the actual link transports goodput without redundant retransmissions arriving at the Niue end, which again agrees well with the goodput observations in the simulation above. At the utilisation and data rates observed, the link can thus handle around 25 such parallel connections, note however that most flows are much shorter and take up less bandwidth due to TCP slow start, i.e., the link can actually accommodate a much higher number of connections.

Single TCP connections downloading 5 MB across a TCP/NC tunnel into Niue achieved around 2–2.4 Mbps goodput with very low overhead, i.e., the entire link capacity would be exhausted by 3–4 such connections. Given the high existing link utilisation, this additional performance of even a single connection comes at the expense of conventional TCP goodput. However, if these connections are downloads, the higher goodput rate also shortens the flow. This poses the question as to whether short wideband flows with TCP/NC are better from a user perspective than long thin ones, given that the bulk of bandwidth is taken up by flows that download something.

In Niue, we also investigated the potential of H-TCP and Hybla compared to the standard Cubic TCP used in the Linux kernel. While there were considerable differences between them and Cubic at certain times, neither of the two presented a convincingly strong alternative on this narrowband path.

Funafuti: Conventional TCP on the link into Funafuti arrives at a SilverPeak NX-3700 WAN Optimiser [10], whose exact configuration could not be determined. The NX-3700 supports, among others, two functions that directly impact on our measurements: network memory and parity packets. Network memory is essentially a data compression technique that prevents previously seen data from transiting across the link again. Parity packets protect small sets of subsequent packets by parity bits, such that packets lost on the link can be recovered island-side (in principle, this is a very crude form of network coding). Our observations indicated that both features were active.

Our NC encoder/decoder in Funafuti sits in parallel to the NX-3700, i.e., the standard TCP in our measurements has the benefit of the NX-3700 whereas the TCP/NC traffic does not pass through the WAN optimiser. Figure 8 shows the goodput achieved by individual TCP and TCP/NC connections with 5 MB downloads into Funafuti and the packet loss experienced at the time. The TCP/NC

tunnel used $n = 30, \omega = 15$ throughout. Breaks in the curves indicate that the respective download failed. The packet losses into Funafuti are quite significant with up to 10 % and more seen on some weekdays – levels at which conventional TCP simply does not work. Closer inspection of log data reveals burst losses of many hundreds of packets. At some times, packet losses in generations exceed 15 and thus render entire generations undecodable, with a resulting irrecoverable TCP loss rate. The NX-3700 manages to persist in some of these cases, albeit at extremely low goodput rates. At times of moderate packet loss, e.g., in the mornings and evenings, TCP/NC provides significantly better goodput than TCP via the NX-3700. During the night, packet loss drops to negligible levels, and the lower overhead in the TCP connections via the NX-3700 results in better goodput here.

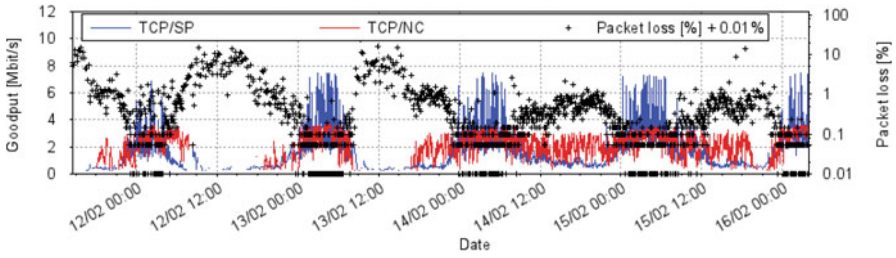

Fig. 8. Measured goodput of individual TCP and TCP/NC connections into Funafuti (Tuvalu) vs. packet loss.

Direct monitoring of the feed from the satellite link to the NX-3700 showed typical daytime link utilisation between 2 and 3 Mbps, far below the estimated 16 Mbps bandwidth. Again, this is a strong indicator for the presence of queue oscillation during these times. Our simulations show neither the low link utilisation observed here, nor the extreme burst error losses. A conceivable explanation for this is that rudimentary error correction such as that performed by the NX-3700 delays detection of packet loss onset by the TCP senders without ultimately being able to mask the packet loss, resulting in more radical TCP back-off.

Rarotonga: The first comparative multi-day measurement of traffic into Rarotonga at the end of January 2015 showed that both TCP and TCP/NC downloads of 20 MB could achieve high goodput of 20–25 Mbps. TCP/NC performed at this level almost continuously, whereas conventional TCP dropped to 5 Mbps and below during daytime peaks with high packet loss. Peak-time link utilisation was typically around 60 % of the available 160 Mbps. This corresponds well with the simulation above.

At the end of May 2015, inbound bandwidth had been increased to 332 Mbps. Telecom Cook Islands was now also able to provide us with hourly averages of the link utilisation as well as of the number of new connections per second. Typical daytime values for the latter were around 800–1000 per second, which

greatly assisted in modelling the link for the simulations above. Figure 9 shows that the goodput of our downloads did not benefit from the bandwidth increase at all: Peak-time goodput deteriorated significantly, despite hourly link utilisation peaking at only 150 Mbps – just 45 % of the available capacity. As predicted by the simulation, daytime packet loss (not shown) had also increased and frequently exceeded 1 % – more than sufficient to slow down TCP. Note that the periods of low goodput are once again characterised by high packet loss. They occur during times when link utilisation is comparatively high but still well below 50 %. The queue thus clearly oscillates here, too. Previously common goodput rates of 20 Mbps and more are now the exception and occur mostly during times of low and rising link utilisation. The rate at which new TCP connections commence on the link also follows a diurnal pattern, albeit with maxima both at the time of worst TCP goodput and of good TCP and TCP/NC goodput. Actual TCP goodput observed for our successful downloads was never below 0.6 Mbps, i.e., higher than the average of around 0.045 Mbps predicted by the simulation. A likely cause for this is TCP slow start: Our downloads were much longer than the average TCP flow and therefore had more opportunities to increase their congestion window over time.

TCP/NC gave consistently better goodput (on average 80 % more) during times of low conventional TCP goodput on the link. However, compared to the earlier data series, even this was still a fraction of the goodput at the time. This suggests that burst losses were large enough to damage entire generations on occasion.

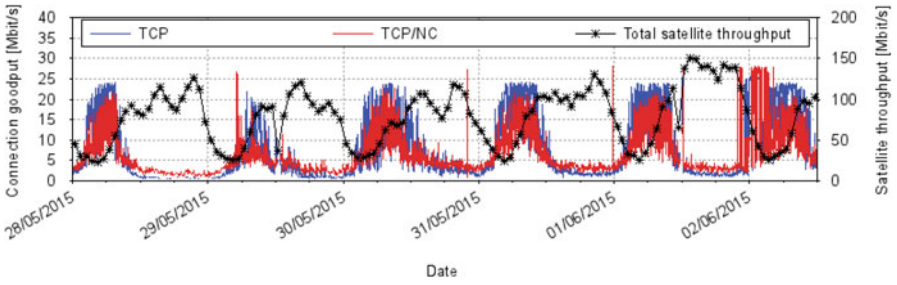

Fig. 9. Measured goodput of individual TCP and TCP/NC connections into Rarotonga and average hourly total satellite link utilisation on 332 Mbps inbound capacity.

7 Conclusions

Signs of TCP queue oscillation were present in all three links investigated. Observed link utilisation on the Niue link was high and short burst errors were observed, however our simulation of the link shows that the high link utilisation in this scenario does not prevent complete queue drain. Rather, the queue seems to refill again rapidly in this case. TCP/NC can achieve significantly higher

goodput than conventional TCP. However, since the link is already transporting almost exclusively goodput close to link capacity, the better performance of TCP/NC comes at the expense of the goodput of conventional TCP on this link.

The Tuvalu scenario is not as clear-cut. Observation of the actual link shows significant amounts of burst packet loss and very low link utilisation, as well as very low per-flow goodput for both TCP (via a SilverPeak NX-3700 WAN accelerator) and TCP/NC. All of these are symptoms of severe queue oscillation. However, while our attempts to simulate the link result in oscillation under many scenarios, we have not been able to replicate either the very low link utilisation or the extremely long burst packet losses. Further investigation will be required to establish whether extreme oscillation of this kind can be attributed to different traffic patterns or whether the WAN accelerator plays a role here. TCP/NC is able to provide significantly better goodput on the Tuvalu link at times of moderate packet loss, a condition met for at least several daytime hours on most days.

In the Rarotonga case, simulations of the 160 Mbps link agree well with the empirical data collected from the end of January 2015: Actual link utilisation and packet losses closely match those of a simulated queue with capacity for 50 packets. At the time, TCP/NC was able to provide consistently high goodput close to the best off-peak goodput from conventional TCP. During times of high daytime packet loss, TCP/NC sustains this goodput while conventional TCP goodput often shrinks to a fraction thereof.

Both our simulations and our empirical data from end of May/early June 2015 shows that additional bandwidth need not necessarily result in higher goodput for individual flows, and that the total number of concurrent flows and total goodput may remain virtually the same. The result in this case may simply be a less well utilised link. At the end of May/early June 2015, TCP/NC was again able to provide better goodput than conventional TCP during peak times, albeit at a fraction of the rates observed earlier in the year.

Obtaining a complete picture of the traffic on a particular link is inherently difficult. Information such as flow size distribution, flow creation rates, satellite gateway queue capacity, number of concurrent flows, or off-island latency distributions all have an influence on link behaviour, as do measures such as traffic shaping, rate limiting, or WAN optimisation. Operators in the islands are often not in a position to supply much of this information and data. Moreover, remote longitudinal data collection on site can seem like the communications equivalent of keyhole surgery: Not only does it load the link during experimentation, but also (in the reverse direction) during the retrieval of experimental data. This precludes in particular the retrieval of substantial packet trace files from island sites.

Another challenge is the fact that our TCP/NC experiments to date have not had exclusive access to a link for longer time periods: Our TCP/NC flows always had to share the link with a majority of concurrent conventional TCP flows. This exposes the TCP/NC flows to the burst packet losses experienced by conventional TCP. It is therefore difficult to assess how much of the TCP/NC

overhead is required only to cope with the presence of conventional TCP on the link. Access to an exclusive satellite transponder would therefore be desirable in any future investigations.

Acknowledgements. This research was supported by the Information Society Innovation Fund Asia through the Pacific Island Chapter of the Internet Society (PICISOC) and by Internet New Zealand. We would also like to thank the many Internet users and staff of Telecom Cook Islands, Internet Niue, and the Tuvalu Telecommunication Corporation for their patience during this study and for sharing their precious bandwidth with us. We would also like to thank Nevil Brownlee for letting us use his flow traces, which assisted us in modelling our flow size distribution.

References

1. O3b Networks, home page. http://www.o3bnetworks.com/
2. Postel, J.: Transmission Control Protocol, Internet RFC 793
3. Jacobson, V.: TCP Extensions for Long-Delay Paths, Internet RFC 1072
4. Jouanigot, J.M., Altaber, J., Barreira, G., Cannon, S., Carpenter, B. and others: CHEOPS Dataset Protocol: An efficient protcol for large disk based dataset transfer on the Olympus Satellite. CERN, Computing and Networks Division, CERN-CN-93-06 (1993)
5. Kim, J.H., Yeom, I.: Reducing queue oscillation at a congested link. IEEE Trans. Parallel Distrib. Syst. **19**(3), 394–407 (2008)
6. Caini, C., Firrincieli, R.: TCP hybla: a TCP enhancement for heterogeneous networks. Int. J. Satell. Commun. Netw. **22**(5), 547–566 (2004)
7. Leith, D.: H-TCP: TCP Congestion Control for High Bandwidth-Delay Product Paths. Internet Draft, IETF (2008). http://tools.ietf.org/html/draft-leith-tcp-htcp-06
8. Sundararajan, J.K., Shah, D., Médard, M., Jakubczak, S., Mitzenmacher, M., Barros, J.O.: Network coding meets TCP: theory and implementation. Proc. IEEE **99**(3), 490–512 (2011)
9. Hansen, J., Krigslund, J., Lucani, D.E., Fitzek, F.H.: Sub-transport layer coding: a simple network coding shim for IP traffic. In: 2014 IEEE 80th Vehicular Technology Conference (VTC Fall), pp. 1–5 (2014)
10. Silver Peak WAN Optimization Appliances, Appliance Manager Operators Guide, VXOA 6.2, December 2014. http://www.silver-peak.com/sites/default/files/userdocs/appliancemgr_operators_guide_r6-2-5_revn_december2014_0.pdf
11. ntop home page. http://www.ntop.org/