TCP variants and transfer time predictability in very high speed networks

Romaric Guillier, Sébastien Soudan, Pascale Vicat-Blanc Primet
RESO Team, LIP Laboratory (UMR CNRS - ENS Lyon - UCB Lyon - INRIA 5668)
Lyon, France
Email: Pascale.Primet@ens-lyon.fr

Abstract—In high performance distributed computing applications, data movement have demanding performance requirements such as reliable and predictable delivery. Predicting the throughput of large transfers is very difficult in paths that are heavily loaded with just a few big flows. This paper explores how current high speed transport protocols behave and may improve transfer time predictability of gigabits of data among endpoints in a range of conditions. In a fully controlled long distance 10 Gbps network testbed, we compare several TCP variants behaviour in presence of diverse congestion level and reverse traffic situations. We show that these factors have a very strong impact on transfer time predictability of several transport protocols.

I. INTRODUCTION

In high performance distributed computing like experimental analysis of high-energy physics, climate modelling, and astronomy, massive datasets must be shared among different sites, and transferred across network for processing. The movement of data in these applications have demanding performance requirements such as reliable and predictable delivery [1].

Generally, distributed applications and high level communication libraries available on end systems use the socket API and TCP as transport protocol. TCP is a fully distributed congestion control protocol which statistically share available bandwidth among flows in a “fair” way. In Internet, where the endpoints’ access rates are generally much smaller (2 Mbps for DSL lines) than the backbone link’s capacity (2.5 Gbps for an OC48 link) these approaches used to be very efficient. It has also been shown that in such conditions, and particularly when the load is not too high and the degree of multiplexing in the bottleneck link is high, formula-based and history-based TCP throughput predictors give correct predictions [2]. But for high-end applications, the bandwidth demand of a single endpoint (1 Gbps, say) is comparable to the capacity of bottleneck link. In such a low multiplexing environment, high congestion level may be not rare and a transient burst of load on the forward or on the reverse path may cause active transfers to miss their deadlines. For example, this situation might occur when processes belonging to different applications are exchanging input and output files simultaneously.

The goal of this article is then to explore this issue and to examine how recent transport protocol enhancements could benefit to high-end applications in terms of data transfer efficiency and predictability in the absence of any access control and reservation mechanisms. It is centred on elephant-like bulk data transfers in very high-capacity (1 Gbps, 10 Gbps) networks these environments are supposed to benefit today and on new TCP variant protocols that are currently available on end nodes. The systematic evaluation of the protocols in a fully controlled and real testbed called Grid’5000 provides a set of measurements of transfer time in a broad range of conditions. We explore mainly three factors: synchronisation of start time, congestion level and reverse traffic.

The paper is organised as follows. In section II, several protocols enhancements proposed are briefly surveyed. Section III is dedicated to our experimental methodology and testbed. Experimental results are given and analysed in section IV. We study systematically three factors influencing the protocol behaviour and affecting the predictability of data transfers. Related works are reviewed in section V. Finally, we conclude in section VI and propose some perspectives for protocol and network service enhancement.

II. TRANSPORT PROTOCOL VARIANTS AND THEIR CHARACTERISATION

The enhancement of TCP/IP has been intensively pursued to tackle limits encounter in long bandwidth-delay product environment [3]. Different TCP variants have been proposed to improve the response function of AIMD congestion control algorithm in high bandwidth delay product networks. All these protocols are not equivalent and behave differently according the network and traffic conditions. In this paper we concentrate on the TCP variants available in all recent GNU/Linux kernel: High Speed TCP, Scalable TCP, Hamilton-TCP, BIC-TCP and CUBIC.

To analyse the data acquired, several metrics can be used to synthetically characterise the behaviour of different TCP variants [4]. These metrics are: fairness, throughput, delay, goodput distribution, variance of goodput, utilisation, efficiency, transfer time. This paper concentrates on transfer time metric which can be considered as a throughput metric. Indeed, throughput can be measured as a router-based metric of aggregate link utilisation, as a flow-based metric of per-connection transfer times, and as user-based metrics of utility functions or user waiting times. Throughput is distinguished from goodput, where throughput is the link utilisation or flow rate in bytes per second, and goodput, also measured in bytes per second, is the subset of throughput consisting of useful
traffic. We note that maximising throughput is of concern in a wide range of environments, from highly-congested networks to under-utilised ones, and from long-lived flows to very short ones. As an example, throughput has been evaluated in terms of the transfer times for connections with a range of transfer sizes for evaluating Quick-Start, a proposal to allow flows to start-up faster than slow-start [5].

III. METHODOLOGY

This article is associated with the Grid’5000 project, an experimental grid platform gathering 2500 processors over nine geographically distributed sites in France. It allows dynamic deployment of network stacks. The network infrastructure is an interconnection of LANs (i.e. grid sites) and an 10 Gbps lambda-based private network [6]. We are using iperf, GNU/Linux kernel version 2.6.16 with Web100 patch and CUBIC patch to perform our experiments.

Figure 2 presents the topology that was used for the experiments. It is a classical dumbbell, with N pairs of nodes that are able to send at 1 Gbps on each side. N is subdivided into two parts, according to the function assigned to the nodes. Nf refers to the number of nodes’ pairs used to send traffic on the forward path (A → B) and Nr the number of nodes’ pairs used to send traffic on the reverse path (B → A). One flow by nodes’ pair is used to perform a file transfer. The bottleneck is the L2 switch. Here the Grid’5000 backbone could be the 10 Gbps link between Rennes and Toulouse (19.8 ms RTT) or a 1 Gbps link between Rennes and Lyon (12.8 ms RTT).

The congestion factor is defined as the ratio between the Nf nodes’ nominal capacity and the bottleneck capacity. Similarly the reverse traffic factor is the ratio between the Nr nodes’ nominal capacity and the bottleneck capacity. The multiplexing level is equal to Nf.

We explore starting time, congestion level and reverse traffic level parameters. We are considering several metrics, along with those defined in [7]. The mean completion time is defined as: \[ T = \frac{1}{N_f} \sum_{i=1}^{N_f} T_i \] where Ti is the completion time of the ith Nf file transfer (typically 30 GB).

IV. RESULTS

A. Influence of starting time

The interval between each flow’s start is of importance as losses during slowstart lead to sthreshold moderation and may limit the achievable throughput during the whole transfer. Figure 3 illustrates the worst case: starting all flows simultaneously (within the same second) has the worst impact on the completion time of the flows and the best case: starting every flow outside the slow start phase of the others. The upper figure 3(a) exhibits a set of flows experiencing drops during their slow start phase. These were unable to obtain a correct share during the rest of the experiment. Other grabbed a large portion of the bandwidth and completed in a short time (300 s). Even though the mean completion time in the worst case is better in figure 3(b) (409 s vs 425 s), it has a much larger standard deviation (83 vs 28) than in the best case. We note that this parameter is especially important for the less aggressive TCP variants as they require a longer time to recover from these losses.

For the rest of our experiments, we choose to set the starting delay between transfers to 1 s to avoid potential harm from this parameter as in the best case, slow-start takes in the best case (log2 N − 1) * RTT, for a congestion window of N packets [8]. For a 19.7 ms RTT, N ≃ 1600 and slow-start lasts about 200 ms.

B. Congestion level

Figure 4 shows the impact of high congestion level on several TCP variants. For example, we observe that the predictability of a transfer time with Scalable is bad as there is more than 300 s between the first and the last completion time. Even though they are both completely filling the link, they have different behaviours. The bandwidth sharing with CUBIC is fair among the various transfers leading to a smaller variance in the completion time.

All protocols, except Scalable, behave similarly\(^1\). Scalable is somewhat remarkable as it is often displaying the shortest and the longest completion time for a given Nf as in figure 6 where we draw a comparison of the completion time distribution of CUBIC and Scalable. Even though both distributions are roughly gaussian-shaped, Scalable is more spread out (294 s vs 114 s for the 2.1 congestion level case) than CUBIC. It makes Scalable a poor choice if we need to wait for all transfers to complete. But if we can start computation on a limited dataset (like a DNA sequencing), we might be able to increase the usage of the computation nodes. It might not be the case in other applications like astronomy interferometry that will need

\(^1\)Our previous work [7] has shown that for the RTT used in our experiments here, most TCP variants tend to have similar performance.
full transfer of all images before the start of a computation phase. Figure 5 presents the impact of the congestion level on mean completion time for CUBIC and Scalable. The ideal TCP represented on the same figure corresponds to a TCP able to send continuously over 1 Gbps links, without slow-start phase, without losses or retransmissions and with equal sharing of the bottleneck link. We can see that the models are continuous but not differentiable when congestion appears. The completion time of transfers when there is no congestion is constant. Scalable is displaying an asymptotical behaviour. The fact that the slopes for CUBIC and the ideal TCP are very close (242 to 252) might indicate that for a greater number of transfers we may observe an asymptotic behaviour too. It may be linked to aspects of Altman’s modelling of TCP parallel transfers [9].

C. Reverse traffic impact

The impact of reverse traffic on transfer time for various reverse traffic congestion levels is then studied. For worst case analysis, reverse traffic is congesting and consists in similar 30 GB file transfers. In figure 7, we observe that reverse traffic has a huge impact, as aggregated goodput is nearly halved during reverse traffic presence and the latest completion time goes from 562 s to 875 s.

The multiplexing level appears to be an important parameter as for similar congestion and reverse traffic level, we observe that experiments at 10 Gbps (figure 8) using a more important number of nodes yield better results: about 617 s (30 % faster).
In the non-congesting reverse traffic case, the impact is very low, there is only a small hollow corresponding to the reverse path’s acks required bandwidth. When the reverse traffic is congesting the link, the aggregate goodput is no longer stable. CUBIC seems to be more sensible than Scalable to this parameter as we observe more than 2 Gbps goodput drop consecutive to synchronised losses. It might be problematic as we are clearly losing bandwidth in this case. If unexpectedly we start to have congesting reverse traffic on our path, it might be easy to miss a deadline.

Figure 9 presents the effect of different levels of reverse traffic on the mean completion time for CUBIC. We can see that for reverse traffic level lower than 1.0, its effect is limited on the mean completion time (about 2.5%). The fluctuations observed for 0.9 reverse traffic level are mainly due to the fact that we are close to the congestion gap and thus to a very unstable point. When the reverse traffic is congesting, we observe that the difference with the case without reverse traffic is much more important (about 10%).

Linear models seem to fit rather well (determination coefficient above 0.98). The slopes for 0.7, 1.1 and no reverse traffic level are very similar to each other, which indicates that reverse traffic’s impact could be seen as a reduction of the available bandwidth.

V. RELATED WORKS

High Speed transport protocol design and evaluation is a hot research topic [10]–[13]. Several papers have compared the protocol by simulations and real experiments [14], [15]. These works are general works and focus on analysing the behaviour of these protocols in high speed Internet context. Several methodologies and results have been proposed by [16]–[18] to identify characteristics, describes which aspect of evaluation scenario determine these characteristics and how they can affect the results of the experiments. These works helped us in defining our workloads and metrics. Our work focus on shared high speed networks dedicated to high performance distributed applications and on the transfer delay metric.

On transfer delay predictability, Gorinsky [19] has shown that to complete more tasks before their respective deadlines, sharing instantaneous bandwidth fairly among all active flows is not optimal. For example, it may be beneficial to
allow a connection with larger pending volume and earlier deadline to grab more bandwidth in a given period, as the Earliest Deadline First scheduling in real-time systems [20]. [21] introduces access control and flow scheduling in grid context. This harmonises network resource management with other resources management and serve the global optimisation objective.

To provide bulk data transfer with QoS as Agreement-Based service in Grids, Zhang et al. [22] evaluate the mechanisms of traffic prediction, rate-limiting and priority-based adaptation. In this way, agreements which guarantee that, within a certain confidence level, file transfer can be completed under a specified time are supported. Similarly, [23] also considers statistical guarantees.

[24] also proposes a study of the impact of reverse traffic on TCP variants, but it is only providing NS-2 simulations with a 250 Mbps bottleneck and a small number of nodes. He is only focusing on the impact on link utilisation, but our results are very similar (reduction of the global amount of bandwidth available for the application level). He is also considering a much larger range of RTT than us.

VI. CONCLUSION

This paper uses real experiments to examine the impact of a range of factors on transfer delay predictability in classical bandwidth sharing approach proposed by high speed TCP-like protocols. These factors are difficult to capture in classical analytic formulations. New models are then needed. We show that when bulk data transfers start simultaneously, transfer time efficiency and predictability are strongly affected. When the congestion level is high (> 120 %) transfer time efficiency and predictability depends on the chosen protocol. The most important factor this study reveals is the reverse traffic impact. It strongly affects all protocols. We conclude that flow scheduling service controlling the starting time and the congestion level in forward and reverse path is mandatory in these low multiplexing environments. Such service, combined with an adaptable and very responsive protocol which can fully exploit a dynamic and high capacity, could be a solution to provide a good transfer time predictability to high end applications. We plan to design, develop and experiment such a service in the Grid’5000 context.

VII. ACKNOWLEDGEMENT

This work has been funded by the French ministry of Education and Research via the IGTMD ANR grant, the ANR CIS HIPCAL project and the EC-GIN grant (IST 045256). Experiments presented in this paper were carried out using the Grid’5000 experimental testbed, an initiative from the French Ministry of Research through the ACI GRID incentive action, INRIA, CNRS and RENATER and other contributing partners (see http://www.grids5000.fr).

REFERENCES