On the Dynamic Behavior of Digital Fountain Based Communication

Zoltán Móczár, Sándor Molnár
High Speed Networks Laboratory, Department of Telecommunications and Media Informatics
Budapest University of Technology and Economics, Budapest, Hungary
E-mail: {moczar, molnar}@tmit.bme.hu

***Abstract*—The volume of Internet traffic has been growing exponentially in the last years, and this trend is expected to intensify in the future. In typical backbone networks hundreds of thousands of flows, originated from different users and versatile applications, compete for the available resources. Such a heterogeneous mixture of traffic flows leads to a continuously changing environment, hence it is crucial to deeply understand the behavior of the underlying data transfer mechanisms regarding many features like stability, convergence and responsiveness. In this paper we study the characteristics of the recently proposed digital fountain based transport in dynamic network conditions, and carry out a comparison with the traditional approach relying on TCP's congestion control.**

I. Introduction

Since the early days of the Internet congestion control, introduced by the *Transmission Control Protocol (TCP)*, has played the key role in reliable host-to-host communication. In the last decades the characteristics of network traffic have changed considerably due to the evolving technologies and the diversity of applications. Today's Internet is a large-scale, highly dynamic network in which sudden variations are common due to topology and bandwidth changes. While a great portion of TCP evaluation studies deal with performance analysis solely in static environments, some researchers emphasize the importance of exploring how responsive a transport protocol is under rapidly changing conditions [1], [2]. In spite of the significant research efforts devoted to optimize the operation of TCP for a wide range of network environments, it seems that congestion control may not be able to cope with the increasing demands of future networks. In the recent years we have been working on an alternative data transfer paradigm, which omits congestion control and applies rateless erasure codes to recover the information lost during the transmission. The multi-platform performance evaluation of our experimental prototype implementation called *Digital Fountain based Communication Protocol (DFCP)* [3] revealed that this new approach has several potential benefits. In this paper we deeply investigate the dynamic behavior of our proposal and carry out a comparison with the traditional TCP-based solution of current Internet.

The paper is organized as follows. First, in Section II we discuss the main principles of the data transfer paradigms together with the core components including the *transport protocol* and the *scheduling mechanism*. In Section III we study the behavior of these approaches in dynamic environments through packet-level simulations focusing on the properties of stability, convergence and responsiveness. Finally, Section IV summarizes the paper and draws our conclusions.

II. Data Transfer Paradigms

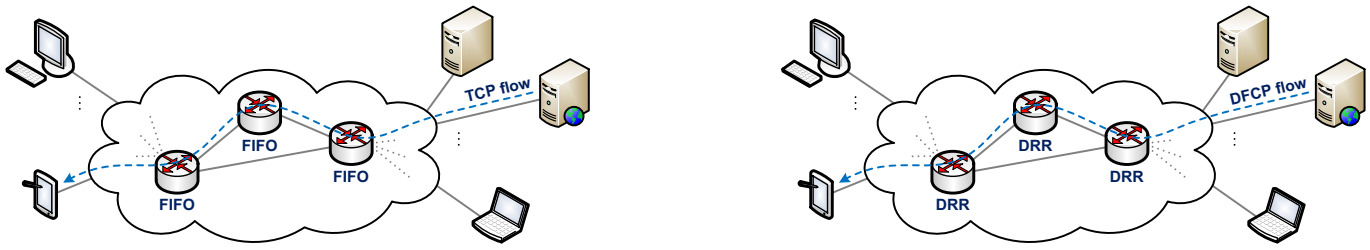
A. Congestion Control

In the history of the Internet closed-loop congestion control was the successful paradigm to avoid congestion collapse and the related performance degradation due to the overload of network resources. Congestion control is performed by the *Transmission Control Protocol (TCP)*, which transports more than 80% of Internet traffic. The basic idea behind the congestion control mechanism of TCP is that the source can determine the proper sending rate and reduce it before congestion could happen. The success of TCP was not even questioned until the fast development of networks, mobile devices and user applications led to heterogeneous and complex environments in the last decades. To cope with these changes many different TCP versions have been proposed [4], [5] and evaluated [6]. Although they introduced great ideas for further development of TCP, the vast majority of them have never been deployed in real networks. TCP Cubic [7] is one of the most widely used TCP versions, because it functions as the default congestion control algorithm of Linux operating systems.

Nowadays, most network routers apply a simple FIFO queue management algorithm to handle packet buffering. By using this method, when the queue becomes full, the newly arriving packets are dropped until the queue has enough room to accept incoming traffic. Due to the fact that network traffic consists of thousands of competing flows, a data transfer paradigm has to ensure fair bandwidth sharing. In case of the TCP-based architecture fairness is managed at the host side, but different TCP versions realize different types of fairness [8]. In this paper the dynamic behavior of TCP Cubic with FIFO queue management is investigated (see Figure 1a), and we call this approach as *Congestion Control based Architecture (CCA)*.

B. Fountain Coding

Over the last decade, the issues of TCP motivated researchers to find alternative ways for data transfer beside the traditional congestion control based approach. One of these ideas recommends the omission of congestion control from the transport layer and the application of erasure coding schemes instead to handle congestion [9]. This proposal raises new questions both in theory and practice, but many research works prove that it can be very promising for future networks. The most surprising result has been presented by Bonald et al. [10] who claim that the absence of congestion control does not necessarily lead to congestion collapse. Erasure codes have



(a) Congestion Control based Architecture

(b) Digital Fountain based Architecture

Fig. 1. Network architectures relying on different transport mechanisms

also been extensively investigated regarding their applicability in data transport, and these studies have revealed the potential benefits (see, e.g. [11], [12]).

The concept of reliable data transport without congestion control has not been specified in detail, and in the lack of implementation no comprehensive performance evaluation has been carried out so far by the research community. To take a step forward, we first introduced our prototype called *Digital Fountain based Communication Protocol (DFCP)* in [3] with some preliminary analytical, simulation and testbed results. In our vision of the future network architecture relying on DFCP each host is allowed to send at its maximum transmission rate. The original data bytes received from the application layer are organized into message blocks, then DFCP applies a Raptor coding scheme to these blocks sequentially. Raptor codes [13], being the most efficient fountain codes [14], can provide linear encoding and decoding complexity involving two phases: LDPC (Low-Density Parity-Check) [15] and LT (Luby Transform) [16] coding. This method enables the senders to generate a theoretically infinite stream of encoded bytes from the original message of size k by adding a redundancy of $\epsilon > 0$. When any subset of size $\lceil (1+\varepsilon)k \rceil$ encoded symbols arrive to the receiver, high probability decoding becomes possible, and fountain coding ensures that each received packet at the destination increases the probability of successful decoding. This approach makes it possible to leave the network congested resulting in fully utilized links. To provide equal bandwidth sharing among competing flows *fair schedulers*, such as Deficit Round Robin (DRR) [17], can be used in the network nodes because per-flow fair queuing has proven to be feasible and scalable [18]. We note that maximal rate sending does not mean the total utilization of the transmission capacity available at the sender side in all cases since it would lead to the so-called dead packet phenomenon. It happens when a source transmits at a higher speed than its fair share of the bottleneck link needlessly wasting the bandwidth on the whole path from concurrent flows. However, there are many possible ways to avoid this undesirable behavior. We are currently working on a solution, which can exploit the benefits of software-defined networks (SDN) where the controllers could provide information about the link utilization to the senders like in the OpenTCP framework [19] for rate control purposes. In order to prevent buffer overflows at the receiver end, a *flow control* mechanism is used in DFCP where the window comprises a certain number of encoded blocks. Since the Raptor coding scheme can generate an infinite stream of encoded bytes, in theory it is plausible to choose the window size as high as possible, however, the use of a larger window leads to a more bursty traffic. In general, it is practical to limit the window size at the point where further increasing does not improve performance.

Our transport protocol has been implemented in the Linux kernel, and validated on independent platforms [20] including simulation environments and real testbeds. We also carried out a comprehensive performance evaluation study both on simple topologies and in multi-bottleneck networks. The research highlighted the potential benefits of our digital fountain based approach, including the high loss and delay tolerance, fair bandwidth allocation, low buffer space demand and fast completion of traffic flows. In this paper the dynamic behavior of DFCP with DRR scheduling is investigated (see Figure 1b), and we refer to this concept as *Digital Fountain based Architecture (DFA)*.

III. Dynamic Behavior Analysis

In this section we reveal how the two data transfer paradigms, DFA and CCA, can handle dynamic traffic situations commonly seen in real networks. We study several important properties in different scenarios including stability, convergence, responsiveness and saturation time [5], [21].

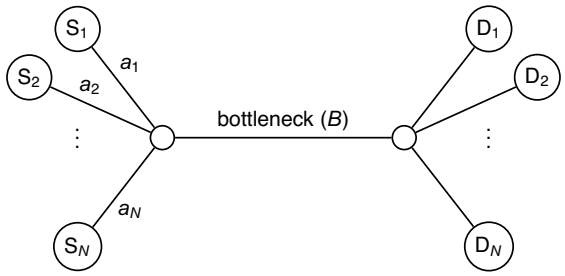
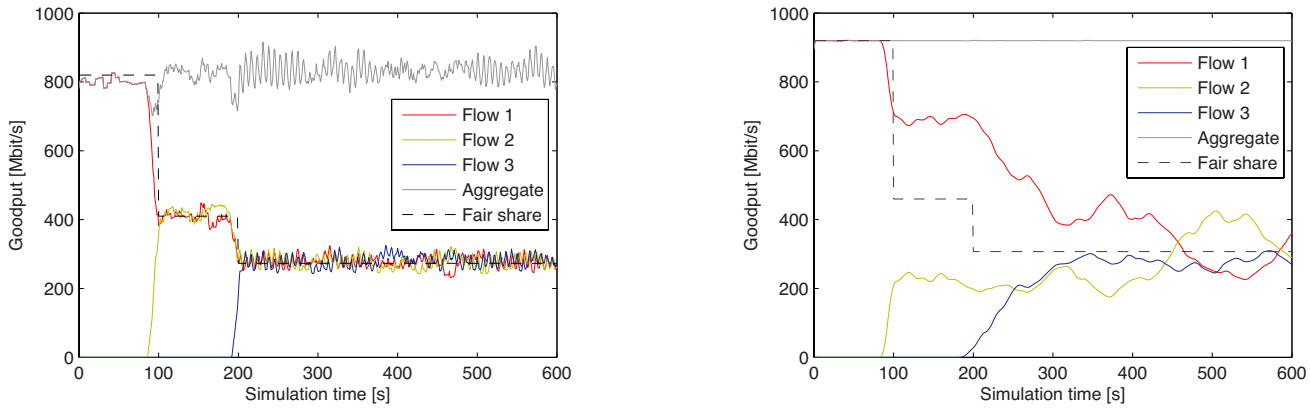


Fig. 2. Simulation topology with N senders and receivers

The performance evaluation was carried out by using the ns-2 packet-level network simulator [22] with the Network Simulation Cradle (NSC) extension [23]. This integrated simulation framework allows to test the kernel implementations of protocols and algorithms directly through the network stack of various operating systems. However, NSC only enables the simulation of TCP versions and new TCP-like transport mechanisms by default, hence some protocol-specific modifications have been made to integrate the source code of DFCP into the framework.

The experiments were performed on a dumbbell topology with N senders and receivers as shown in Figure 2. In the



(a) Digital Fountain based Architecture

(b) Congestion Control based Architecture

Fig. 3. Dynamics of concurrent flows started with different delays and their convergence to the fair share

measurement scenarios we examined real-world situations, which typically occur in a dynamic environment by varying the link capacity and delay, the buffer size and the number of competing flows. Simulations lasted for 600 seconds, and if not mentioned otherwise, the bottleneck link capacity was set to 1 Gbps, the round-trip time was fixed at 50 ms and the buffer size (denoted by b) was equal to the bandwidth-delay product (BDP). In DFCP the redundancy and the window size parameters were adjusted to $\epsilon = 0.05$ and 1000 blocks, respectively.

A. Stability and Convergence

Network traffic is generated by heterogeneous applications that results in many concurrent flows traversing different network paths with multiple bottlenecks from source to destination. The transmission rates of these flows fluctuate rapidly since the currently used congestion control based transfer mechanism could not adapt to changing conditions as fast as needed. Stability is an important property from both traffic engineering and user experience point of views [21]. Rate variations often lead to the oscillation of queue length that can eventually cause buffer overflows. Such an undesirable behavior can result in the loss of synchronization among competing flows, periodic underutilization of link capacity and degraded quality of service. It is also crucial regarding efficiency how fast a flow can obtain its equilibrium rate or converge to the fair share in a dynamic environment.

In Figure 3 the dynamics of three concurrent flows is illustrated when they were started with different delays, namely at 0, 100 and 200 seconds. The goodput gives the current useful data transmission speed in one second resolution, and the curves were smoothed by using a 10 seconds long moving window. We can observe that, for CCA, flows converge slowly to the fair share and then their goodput highly fluctuates around it. In case of DFA the convergence time is very low whereas the fluctuation around the fair share remains moderate. However, the transfer mechanism of DFA leads to a more bursty transmission than that of CCA, which is due to the trade-off between the window size and the burstiness of traffic.

To describe the transient fairness of transport mechanisms, we measured the goodput ratio of two competing flows started with unequal shares of the bottleneck bandwidth for DFA and CCA, respectively. Figure 4 shows a scenario when the flows were launched at 0 and 100 seconds, hence *flow 1* had been utilizing the total available bandwidth at the time of starting the second flow. The y-axis gives the goodput ratio of *flow 2* to *flow 1*. We can see that, for CCA, transmission rates of competing flows converge slowly, but they remain stable at different time scales. In contrast, for DFA, while the goodput convergence is fast providing high degree of long-term fairness, a slight oscillation around the equal share (i.e. the goodput ratio of 1) can be observed at small time scales.

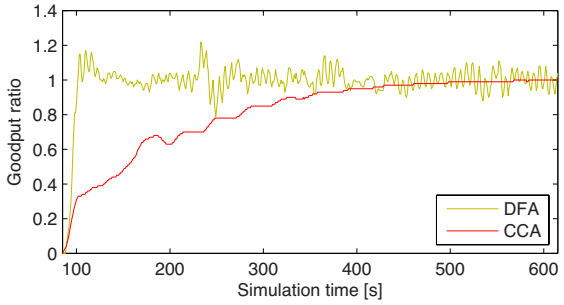


Fig. 4. Goodput ratio in the function of time for two delayed flows

TABLE I
CONVERGENCE TIME TO THE FAIR SHARE

Paradigm	**δ-fair convergence time**		
	$\delta = 0.1$	$\delta = 0.3$	$\delta = 0.5$
DFA	2 sec	1 sec	1 sec
CCA	180 sec	60 sec	5 sec

A frequently used metric to quantify the convergence speed is δ-fair convergence time [1], which can be given as the time taken by two flows to obtain a bandwidth allocation of $(\frac{1+\delta}{2}B, \frac{1-\delta}{2}B)$ starting from $(B - B_0, B_0)$ where $B >> B_0$. Using the settings of $B = 1000$ and $B_0 = 0$, the average δ-fair convergence times provided by DFA and CCA are summarized in Table I. These results have great significance in case of short downloads and suggest that CCA cannot guarantee reasonable fairness for many typical Internet applications since about half of network flows last less than a few seconds.

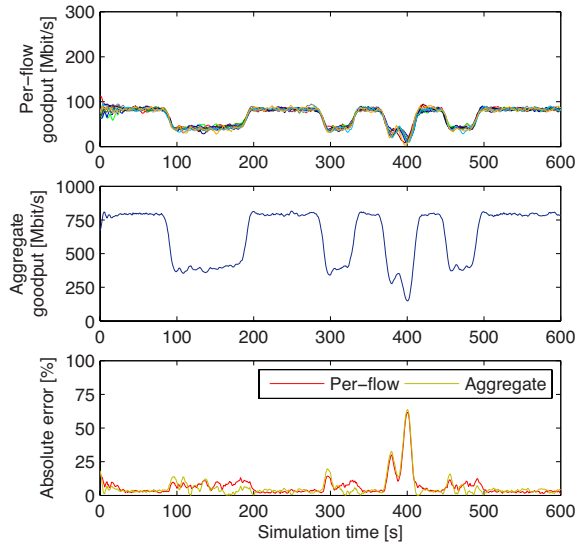


Fig. 5. Responsiveness of per-flow *(top)* and aggregate *(middle)* traffic, and the adaptation error *(bottom)* for DFA with a buffer size of 100 packets

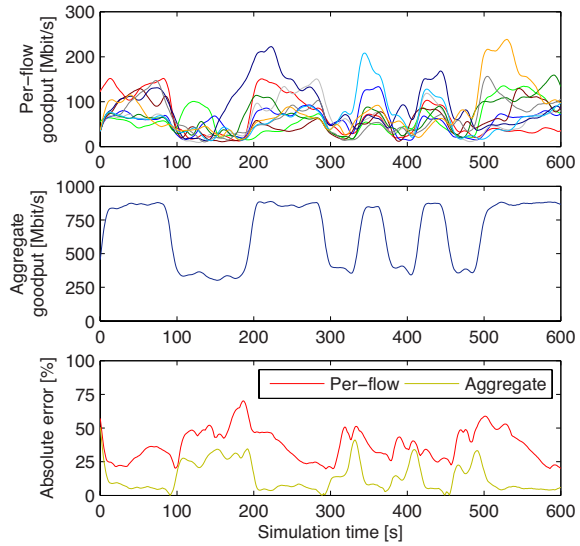


Fig. 7. Responsiveness of per-flow *(top)* and aggregate *(middle)* traffic, and the adaptation error *(bottom)* for CCA with a buffer size of 100 packets

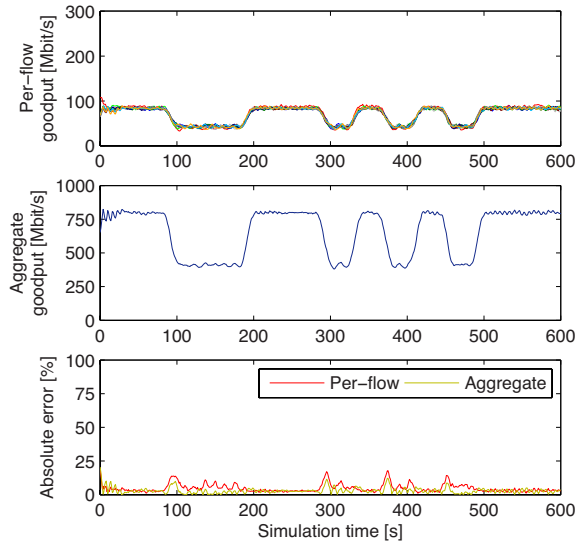


Fig. 6. Responsiveness of per-flow *(top)* and aggregate *(middle)* traffic, and the adaptation error *(bottom)* for DFA with a buffer size of 5000 packets

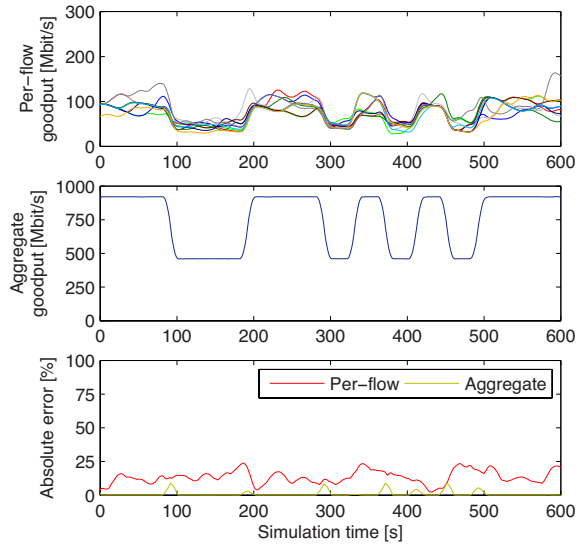


Fig. 8. Responsiveness of per-flow *(top)* and aggregate *(middle)* traffic, and the adaptation error *(bottom)* for CCA with a buffer size of 5000 packets

B. Responsiveness

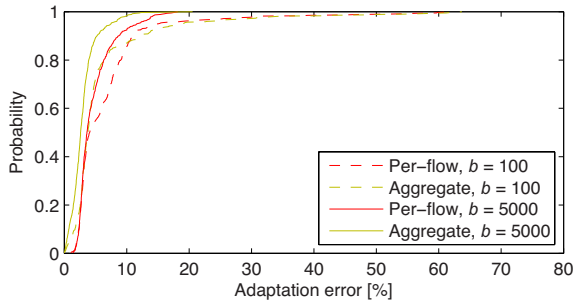
One of the key concerns in the design of transport protocols is the ability to handle abrupt change of network parameters and traffic conditions [21]. In a real network competing flows governed by different transfer mechanisms often face with quick variations mainly originated from routing and bandwidth changes, or sudden congestion. Reponsiveness is of high importance describing how fast and accurately a transport protocol can adapt to these environmental factors.

In this paper we focus on the change of the available bandwidth and quantify the responsiveness of per-flow and aggregate traffic for the two different data transfer paradigms. To this end, we defined and calculated a metric called *adaptation error* as follows. Let g_i be the goodput of flow i and f_i the ideal fair share for flow i taking into account the bandwidth change pattern. Using these notations the adaptation error of per-flow (e_p) and aggregate traffic (e_a) can be computed by the following formulas:

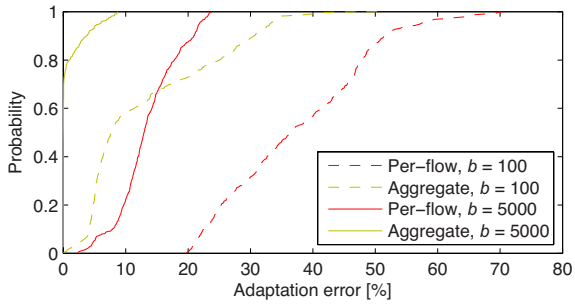
$$e_p = \frac{1}{n}\sum_{i=1}^{n}\frac{|g_i - f_i|}{f_i} \quad \text{and} \quad e_a = \frac{\sum_{i=1}^{n}|g_i - f_i|}{\sum_{i=1}^{n} f_i}$$

where n denotes the number of flows competing for the bottleneck bandwidth.

We analyzed the behavior of 10 competing flows by periodically halving the available bandwidth of the bottleneck link. Specifically, the bandwidth was reduced from 1000 Mbps to 500 Mbps in the interval of [100,200], [300,340], [380,420] and [460,500] as illustrated in Figure 10. In Figure 5–8 the responsiveness of per-flow and aggregate traffic, and the adaptation error are shown for DFA and CCA. Figure 9 depicts



(a) Digital Fountain based Architecture



(b) Congestion Control based Architecture

Fig. 9. CDF of adaptation error of per-flow and aggregate traffic for DFA and CCA

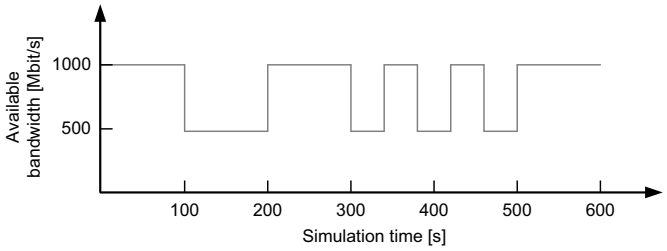


Fig. 10. The change of available bandwidth

the cumulative distribution function (CDF) of error whereas Table II summarizes the mean and standard deviation of values. To reveal the adaptation capability of transfer paradigms to different network environments we investigated both small ($b = 100$) and large ($b = 5000$) buffer sizes (measured in packets). The good operability with small buffers is a mandatory requirement for all-optical communication and also makes it possible to avoid the bufferbloat phenomenon experienced in current Internet mainly due to the use of over-sized router memories [24].

TABLE II
THE MEAN (LEFT) AND STANDARD DEVIATION (RIGHT) OF THE ADAPTATION ERROR IN PERCENTAGE

Source	DFA				CCA			
	$b = 100$		$b = 5000$		$b = 100$		$b = 5000$	
Single flow	2.7	2.6	2.3	1.9	26.1	20.7	2.4	4.5
10 flows, per-flow	6.7	9.2	4.8	4.3	37.5	23.7	13.5	10.4
10 flows, aggregate	6.1	8.4	3.0	2.1	13.2	10.4	0.9	1.9

Regarding the small buffer case ($b = 100$) we can see that the adaptation speed of CCA flows to bandwidth changes is very low (Figure 7). Ideally, each flow would receive an equal share of the bottleneck link, but in this case some flows react too aggresively and some too mildly to changing network conditions. This behavior leads to uneven bandwidth allocation especially during the periods after the available bandwidth is doubled. For example, in the interval of [200,300] the maximum perceived difference in goodput between individual flows exceeds 200 Mbps, which is two times more than the fair share. In spite of the high unresponsiveness of single flows the aggregate traffic roughly follow the change pattern. The per-flow adaptation error is significant and ranges between 20% and 70% with a mean of 38% (Table II) while in case of the aggregate more than half of the samples are below 10% (Figure 9) with a mean of 13%. For DFA, we can experience a moderate oscillation of per-flow goodput around the fair share (Figure 5), however, the stability of aggregate traffic does not show noticeable difference compared to that of CCA. Apart from some outlier values the error rate remains moderate with an average of 7% and 6%, hence it is only slightly higher for per-flow traffic. If large buffers with size close to the BDP ($b = 5000$) are used in CCA routers (Figure 8), individual flows can follow much more smoothly the bandwidth changes, which results in high responsiveness of the aggregate traffic. While the per-flow adaptation error does not exceed 25%, for the aggregate traffic, the error rate is negligible and can only be measured when changes occur. Although the use of large buffers also has a positive effect on the adaptation accuracy of DFA, the improvement is barely noticeable as can be seen in Figure 9. Our measurements indicate that CCA can provide better adaptivity than DFA only in case of the aggregate traffic and if sufficiently large buffers are applied.

C. Saturation Time

The operation of congestion control algorithms consists of two main transmission phases. In the initial phase TCP gradually increases the sending rate until the bottleneck buffer is filled. Then, it is followed by an equilibrium state when the protocol achieves the maximum transmission rate and tries to keep it stable. The length of the transient phase highly determines the download efficiency of short-lived flows, therefore it can affect the quality of experience (QoE) for many applications. In order to capture this behavior, we defined a performance metric called *saturation time* [5], which can be given for a loss-based protocol as the time elapsed from the starting of a flow until the first packet is dropped. Queue saturation time (QST) is a good indicator of how fast a transport protocol can obtain its steady-state performance.

Figure 11 shows the queue saturation time can be provided by the two data transmission paradigms for increasing number of flows. Similarly to responsiveness we investigated the impact of both small and large buffers. If we use a buffer of size greater than the bandwidth-delay product, the bottleneck queue is saturated in a very short time and independently of the number of concurrent flows for both transfer mechanisms. The figure also clearly demonstrates that, by using a buffer size of only 100 packets, QST becomes dramatically high for CCA and it decreases when many concurrent flows share the bandwidth of the bottleneck link. However, even if the results suggest that we can theoretically achieve low QST values for

hundreds of competing flows, CCA is unable to handle such amount of network traffic with small buffers in comparison to DFA.

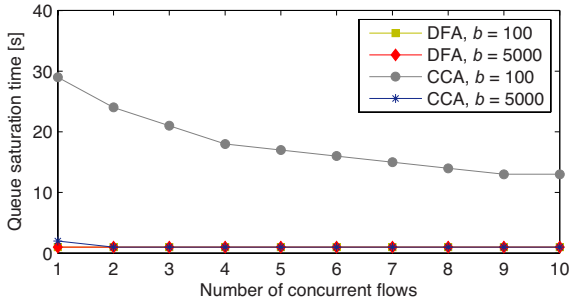


Fig. 11. Queue saturation time for increasing number of flows

In Figure 12 the queue saturation time is depicted for different round-trip time values. We can see that, using CCA, the round-trip time (RTT) considerably affects QST in case of a particular buffer size. With a buffer size of 100 packets QST is already noticeable on low-latency links and it increases for higher RTTs. The same tendency can be observed with a larger buffer of 5000 packets, but the increase in QST is less significant. In contrast, DFA is able to keep QST low even in high-delay environments.

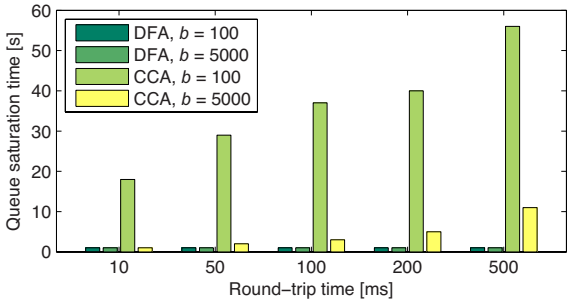


Fig. 12. Queue saturation time for different round-trip times

IV. Conclusion

In this paper we investigated the dynamic behavior of our DFCP-based data transfer paradigm (DFA) and compared it to the current Internet architecture (CCA) relying on the congestion control mechanism of TCP. The simulation results revealed that, while CCA can work with moderate goodput oscillation at small time scales, DFA is more stable in the long run and can guarantee fast covergence for competing traffic flows. We also found that DFA is able to cope with sudden change of network conditions regarding both per-flow and aggregate traffic independently of the buffer size. CCA shows better adaptivity only in the case of aggregate traffic and if sufficiently large buffers are used, with small router memories CCA is highly unresponsive. Furthermore, DFA provides low queue saturation time making QoE improvement possible for many applications.

References

[1] D. Bansal, H. Balakrishnan, S. Floyd, S. Shenker, "Dynamic Behavior of Slowly-Responsive Congestion Control Algorithms", *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 4, pp. 263–274, 2001.

[2] Y. R. Yang, M. S. Kim, S. S. Lam, "Transient Behaviors of TCP-friendly Congestion Control Protocols", *Proceedings of the 20th IEEE International Conference on Computer Communications*, vol. 3, pp. 1716–1725, Anchorage, AK, USA, 2001.

[3] S. Molnár, Z. Móczár, A. Temesváry, B. Sonkoly, Sz. Solymos, T. Csicsics, "Data Transfer Paradigms for Future Networks: Fountain Coding or Congestion Control?", *Proceedings of the IFIP Networking 2013 Conference*, pp. 1–9, New York, NY, USA, 2013.

[4] A. Afanasyev, N. Tilley, P. Reiher, L. Kleinrock, "Host-to-Host Congestion Control for TCP", *IEEE Communications Surveys and Tutorials*, vol. 12, no. 3, pp. 304–342, 2010.

[5] S. Molnár, B. Sonkoly, T. A. Trinh, "A Comprehensive TCP Fairness Analysis in High Speed Networks", *Computer Communications, Elsevier*, vol. 32, no. 13–14, pp. 1460–1484, 2009.

[6] Y.-T. Li, D. Leith, R. N. Shorten, "Experimental Evaluation of TCP Protocols for High-Speed Networks", *IEEE/ACM Transactions on Networking*, vol. 15, no. 5, pp. 1109–1122, 2007.

[7] I. Rhee, L. Xu, "CUBIC: A New TCP-Friendly High-Speed TCP Variant", *Proceedings of the 3rd International Workshop on Protocols for Fast Long-Distance Networks*, pp. 1–6, Lyon, France, 2005.

[8] J.-Y. L. Boudec, "Rate Adaptation, Congestion Control and Fairness: A Tutorial", *Technical Report, École Polytechnique Fédérale de Lausanne (EPFL)*, 2012.

[9] D. Clark, S. Shenker, A. Falk, "GENI Research Plan (Version 4.5)", April 23, 2007.

[10] T. Bonald, M. Feuillet, A. Proutiere, "Is the 'Law of the Jungle' Sustainable for the Internet?", *Proceedings of the 28th IEEE Conference on Computer Communications*, pp. 28–36, Rio de Janeiro, Brazil, 2009.

[11] A. Botos, Z. A. Polgar, V. Bota, "Analysis of a Transport Protocol Based on Rateless Erasure Correcting Codes", *Proceedings of the 2010 IEEE International Conference on Intelligent Computer Communication and Processing*, vol. 1, pp. 465–471, Cluj-Napoca, Romania, 2010.

[12] Y. Cui, X. Wang, H. Wang, G. Pan, Y. Wang, "FMTCP: A Fountain Code-Based Multipath Transmission Control Protocol", *Proceedings of the 32nd IEEE International Conference on Distributed Computing Systems*, pp. 366–375, Macau, China, 2012.

[13] A. Shokrollahi, "Raptor Codes", *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2551–2567, 2006.

[14] D. J. C. MacKay, "Fountain Codes", *IEE Proceedings – Communications*, vol. 152, no. 6, pp. 1062–1068, 2005.

[15] A. Shokrollahi, "LDPC Codes: An Introduction", *Technical Report, Digital Fountain Inc.*, 2003.

[16] M. Luby, "LT Codes", *Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science*, pp. 271–280, Vancouver, BC, Canada, 2002.

[17] M. Shreedhar, G. Varghese, "Efficient Fair Queuing Using Deficit Round-Robin", *IEEE/ACM Transactions on Networking*, vol. 4, no. 3, pp. 375–385, 1996.

[18] A. Kortebi, L. Muscariello, S. Oueslati, J. Roberts, "On the Scalability of Fair Queuing", *Proceedings of the 3rd ACM Workshop on Hot Topics in Networks*, pp. 1–6, San Diego, CA, USA, 2004.

[19] M. Ghobadi, S. H. Yeganeh, Y. Ganjali, "Rethinking End-to-End Congestion Control in Software-Defined Networks", *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, pp. 61–66, Redmond, WA, USA, 2012.

[20] Z. Móczár, S. Molnár, B. Sonkoly, "Multi-Platform Performance Evaluation of Digital Fountain Based Transport", *IEEE Science and Information Conference 2014*, pp. 690–697, London, UK, 2014.

[21] S. Floyd, "Metrics for the Evaluation of Congestion Control Mechanisms", *RFC 5166, IETF*, 2008.

[22] ns-2 Network Simulator, http://www.isi.edu/nsnam/ns/

[23] Network Simulation Cradle, http://www.wand.net.nz/~stj2/nsc/

[24] J. Gettys, K. Nichols, "Bufferbloat: Dark Buffers in the Internet", *Communications of the ACM*, vol. 55, no. 1, pp. 57–65, 2012.