# How to Transfer Flows Efficiently via the Internet?

Sándor Molnár$^{\dagger *}$, Zoltán Móczár$^{\dagger *}$, Balázs Sonkoly$^{\dagger}$
$^{\dagger}$Dept. of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Budapest, Hungary
$^{*}$Inter-University Centre for Telecommunications and Informatics, Kassai út 26., 4028 Debrecen, Hungary
E-mail: {molnar, moczar, sonkoly}@tmit.bme.hu

***Abstract*—In this paper we present a performance evaluation study of different transport mechanisms carried out in a testbed environment to reveal their efficiency regarding the transfer of different flows. The current versions of the *Transmission Control Protocol* (TCP) are compared to the *Digital Fountain based Communication Protocol* (DFCP), which is our newly developed transport protocol where congestion control is not applied, but Raptor code based erasure coding scheme is used to recover lost packets. Our results demonstrate that both short-lived and long-lived flows can be transferred more efficiently by DFCP than by TCPs in various network conditions.**

## I. Introduction

From the early days of the Internet when congestion collapse occurred [1] till today congestion control is used to regulate traffic and avoid situations where increasing network load results in a decrease in the useful work done by the network. This functionality is mostly performed by the *Transmission Control Protocol* (TCP), which was continuously developed and tuned over the previous decades. The development of TCP was unavoidable due to the emerging challenges of the next generation networks like high speed communication, communication over different media, etc. [2], [3]. TCP is a connection-oriented unicast transport protocol that offers reliable data transfer as well as flow and congestion control. Basically, TCP maintains a congestion window that controls the number of outstanding unacknowledged data packets in the network.

The limitations of TCP and the need of up-to-date tuning of its underlying mechanisms may result in TCP versions, which cannot be optimal for all environments and are becoming more and more complex with emerging drawbacks. This was the reason to rethink the concept of this transport protocol and design it from scratch with a brave step towards omitting its congestion control functionality. The idea was first presented by GENI (Global Environment for Network Innovations), which advocated a Future Internet without congestion control [4] by suggesting efficient erasure coding schemes to recover lost packets. We did not find any realization or further refinement of this concept, thus we made our own design and implementation resulting in a transport protocol called *Digital Fountain based Communication Protocol* (DFCP). The protocol was presented in [5] together with the first analytical, simulation and testbed results. Furthermore, the paper gives a discussion about the design principles and highlights the main benefits of the new data transfer paradigm. The idea related to DFCP is that end hosts can send their data at maximal rates while *fair schedulers* deployed in the network nodes are responsible for providing fairness among competing flows. We note that several implementations approximating the ideal fair scheduling, such as Deficit Round Robin (DRR) [6], are already available and can be configured easily. The eligibility of this solution is confirmed by the fact that per-flow fair queueing was proven to be scalable and feasible [7]. If a packet loss is detected (it is very likely since no congestion control is applied), efficient digital fountain based (rateless) codes are used to recover lost packets. We have designed DFCP with Raptor codes [8] and implemented in Linux [9].

The proper evaluation of transport protocols is important and requires thorough investigations. The performance must be analyzed carefully, and it is crucial whether congestion control is applied or not and how efficiently it can work. It is known that the performance of the implemented transport protocols (e.g. TCP versions) in the Internet differ from theory due to the interactions between TCP and middleboxes along the network path [10]. For example, Performance Enhancing Proxies (PEPs) break single TCP connections into two connections potentially changing the end-to-end behavior. In order to evaluate the performance of different techniques like congestion control based TCPs or methods not applying congestion control like DFCP, right metrics must be chosen. Besides the broadly investigated *throughput*, the *Flow Completion Time* (FCT) also serves as an important metric [11] since most of the applications use flow transfers and users' main interest is to download their flows as fast as possible. FCT is the time elapsed from when the first packet of a flow is sent until the last packet is received. Flows transmitted via the Internet have very complex characteristics [12] and the mechanisms of different transport protocols can handle them differently. For example, it is known that TCP enters the congestion avoidance phase after slow-start, which takes many round-trip times (RTT), but the majority of short-lived flows never leave slow-start resulting in high FCTs. In case of long-lived flows the additive increase of the congestion avoidance phase limits the transfer speed, and the fact that TCP fills the bottleneck buffer also contributes to the increase of FCT and it is far from being optimal. Therefore, it is of high interest how different transport protocols are able to cope with different flows in the Internet, which was the motivation of this research.

In this paper we investigate the flow transfer efficiency of two reliable data transfer mechanisms in a testbed environment by simulating different packet loss rates and round-trip times in the network. We compare the congestion control based TCP to the digital fountain based DFCP which is the first implementation of the concept of transport protocol without congestion control [5]. In Section II an overview of related work is given. A short description of the investigated TCP versions and DFCP is provided in Section III. In Section IV we present our performance measurement results focusing on the transient behavior and flow transfer efficiency of DFCP and TCPs. Finally, Section V concludes the paper.

## II. Related Work

Regarding TCP-type transport protocols a huge volume of literature exists since TCP and its different versions (HSTCP, CUBIC, FAST, Compound, Westwood, etc.) determined the mainstream of this research. For a comprehensive tutorial see [13]. The performance of these versions was also investigated and compared, for instance in [3], [13]. Additionally, there have been intensive research efforts on other protocols like eXplicit Control Protocol (XCP), Rate Control Protocol (RCP), etc. to overcome the limitations of TCP [11]. In contrast, no relevant research has been focused on the design, development and analysis of a reliable transport protocol not applying congestion control since the presentation of the idea in GENI [4]. In a broader scope a discussion about some related work can be found in [5].

Surprisingly, the issue of how to design a transport protocol, which optimizes FCT was addressed only in a few papers. It is known that for a single link the *Shortest Remaining Processing Time* (SRPT) scheduling discipline minimizes FCT [14]. The practical implementation of SRPT is limited in the Internet due to many problems, for example, it requires the flow size information for the end hosts, which is not available when a flow starts. There are many suggestions trying to approximate SRPT, and a practical approach is to consider Processor Sharing (PS) policy [11]. PS is eligible for approximating SRPT and it has the advantage that it does not require flow size information in advance. Most of the TCP versions can approximate PS behavior for long-lived flows, but they fail to achieve it for short-lived flows. Since most Internet flows fall into the latter category, it can be a serious limiting factor for many applications regarding flow transfer efficiency. Web traffic serves as a prominent example due to its significant contribution to the generation of short-lived flows, and the underlying transport mechanism plays a key role in the optimization of web performance [15]. Therefore, over the last decades many researchers have focused on the improvement of the slow-start algorithm of TCP to make it more efficient in high speed networks (e.g. [16]) and they introduced various techniques to speed up its operation for short-lived flows as well [11].

## III. Overview of the Transport Protocols

### A. TCP Versions

TCP is a connection-oriented transport protocol that provides reliable data transfer in end-to-end communication. It means that lost packets are retransmitted, and therefore, each sent packet will be delivered to the destination. The most important feature of TCP is its congestion control mechanism, which is used to avoid congestion collapse by determining the proper sending rate and to achieve high performance. TCP maintains a congestion window that controls the number of outstanding unacknowledged data packets in the network. Over the years, many versions of TCP have been developed in order to fit the ever-changing requirements of communication networks. In this paper we investigate two popular and widely used TCP variants in comparison to DFCP, namely TCP Cubic [17] which is the default congestion control algorithm in the Linux kernel and TCP NewReno [18] with SACK option (for brevity it is referred to as TCP Reno in the next sections).

### B. Digital Fountain based Communication Protocol

DFCP is also a connection-oriented, reliable transport protocol, which can be found in the transport layer of the TCP/IP stack [5]. However, unlike TCP our newly developed DFCP protocol does not use any congestion control. Instead, it uses efficient erasure coding based on Raptor codes [8], which are an extension of LT codes offering linear time encoding and decoding complexity. Basically, DFCP sends the encoded data towards the receiver at maximal rate making possible to carry out a very efficient operation. In this case, efficient means that available resources in the network can be fully and quickly utilized without experiencing performance degradation. For further details of DFCP please see [9].

## IV. Performance Evaluation

In this section we present and discuss our measurement results comparing DFCP and two TCP variants in a testbed environment for different network topologies and test scenarios. Our purpose was twofold: (1) to investigate the transient behavior of the transport protocols, and (2) to reveal their flow transfer efficiency in various network conditions regarding both short-lived and long-lived flows.

TABLE I
Hardware components of test computers

| Component | Type and parameters |
|---|---|
| Processor | Intel® Core$^{\text{TM}}$2 Duo E8400 @ 3 GHz |
| Memory | 2 GB DDR2 RAM |
| Network adapter | TP-Link TG-3468 Gigabit PCI-E |
| Operating system | Debian Lenny with modified kernel |

(a) Hardware components of senders and receivers

| Component | Type and parameters |
|---|---|
| Processor | Intel® Core$^{\text{TM}}$ i3-530 @ 2.93 GHz |
| Memory | 2 GB DDR2 RAM |
| Network adapter | TP-Link TG-3468 Gigabit PCI-E |
| Operating system | FreeBSD 8.2 |

(b) Hardware components of the network emulator

The testbed environment was composed of senders, receivers and a Dummynet network emulator. We were able to tune different network parameters by Dummynet such as packet loss probability, delay, queue length and bandwidth [19]. The main hardware specification of the test computers are given in Table I.

### A. Transient Behavior

It is of high importance to investigate the transient behavior of different transport protocols since a huge number of applications download short-lived flows (e.g. web objects) that are performed mostly or fully in the transient phases of these protocols.

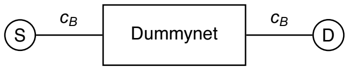
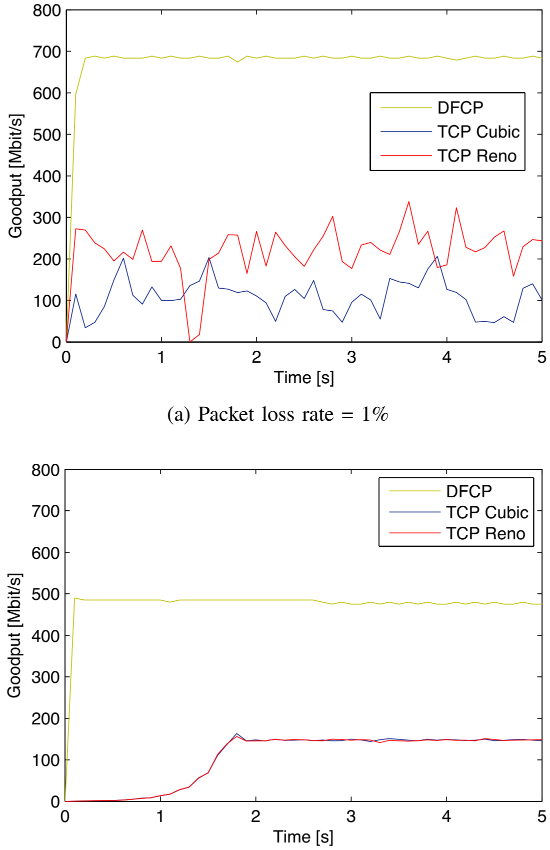


Fig. 1. Dumbbell topology with one source-destination pair

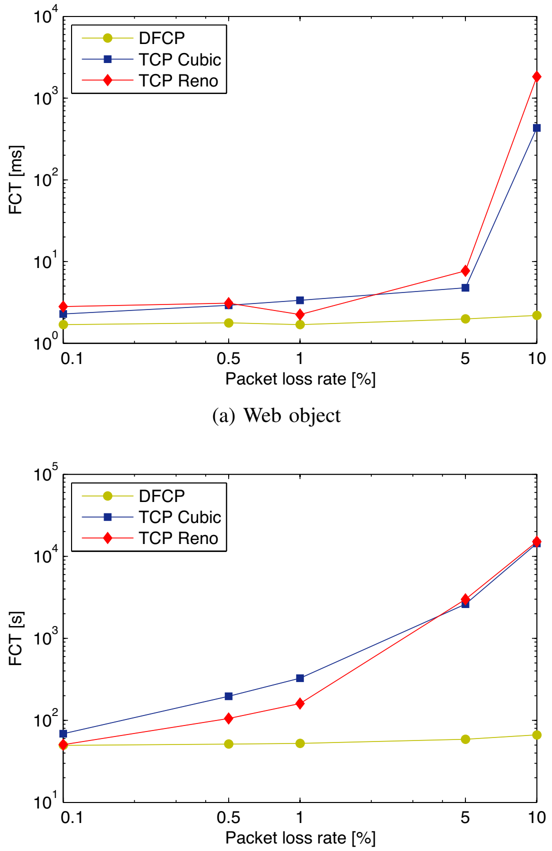
The experiments were performed on a simple dumbbell topology with one source and destination as shown in Figure 1.



(a) Packet loss rate = 1%

(b) Round-trip time = 100 ms

Fig. 2. Transient behavior of the investigated transport protocols



(a) Web object

(b) DVD

Fig. 3. Flow completion time for different packet loss rates

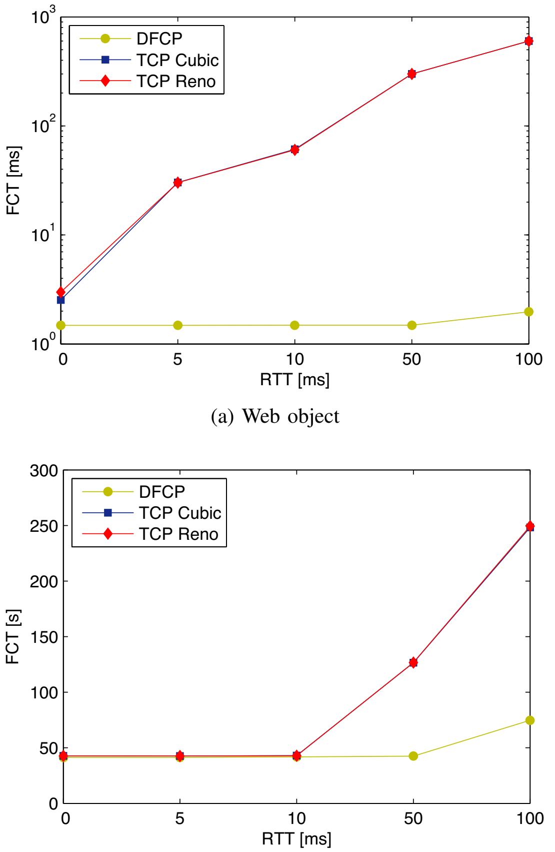
The measurement duration was 60 seconds for each test, and the flows were started separately. Regarding the network parameters only the packet loss rate and the round-trip time were varied. The buffer size was set to a high value in order to exclude it from the limiting factors, and the bottleneck link had a capacity $c_B = 1$ Gbps. In these scenarios we used the goodput (i.e. the number of useful bytes transferred per second) as the performance metric.

In Figure 2 the goodput is depicted for the first 5 seconds of the measurement simulating different network conditions. Figure 2a shows the case when the packet loss rate was fixed at 1%, and the redundancy parameter of DFCP was set to an optimal value. Optimal redundancy is the minimum coding overhead assuming a given loss rate that is necessary for successful data transmission and decoding at the receiver side. The figure clearly indicates that DFCP significantly outperforms both TCP versions in terms of goodput in a lossy environment, and unlike TCP the goodput of DFCP does not fluctuate over time. In other words, DFCP is much less sensitive to packet loss than TCP, which is an outstanding result since one of the most well-known drawbacks of TCP is that its performance degrades very quickly for increasing packet loss probability. Our analysis results also pointed out that, as we increase the packet loss rate, the difference between DFCP and TCPs becomes even more dramatic regarding the goodput. Figure 2b demonstrates the performance of the investigated protocols when the round-trip time was set to 100 ms. We can see that, while DFCP immediately achieves its full speed, the transfer rates of the TCP variants increase much more slowly, and the steady-state goodput is considerably lower compared to DFCP.

### B. *Flow Transfer Efficiency*

As we mentioned in Section I, flow completion time is one of the most important performance metrics from the user's point of view because of the fact that users want to download web pages, softwares, movies and many other contents as fast as possible. Accordingly, we investigated two different categories: (1) web object (150 kB, the mean size is about 100–200 kB [20]) and (2) DVD (4.7 GB), which represent short and long data transfers, respectively.

Figure 3 illustrates how the flow completion time depends on packet loss rate. The flow completion times longer than 60 seconds were calculated by using the steady-state goodput for each figure of this subsection. One can see that in both cases DFCP provides the fastest download indicating its potential in case of web traffic as well as heavy data transfers, however, the benefit is more significant in the latter case. By transferring a typical web object, the most considerable performance gain can be experienced for high packet loss rates (see Figure 3a). However, if we transfer a full DVD, the advantage of DFCP is pronounced in the whole range of packet loss rate (see Figure 3b). Moreover, with optimal redundancy parameters, DFCP becomes almost insensitive to packet loss in these practically relevant scenarios.



(a) Web object

(b) DVD

Fig. 4. Flow completion time for different round-trip times

Investigating the impact of round-trip time we can also find significant differences in the performance of DFCP and TCPs as shown in Figure 4. Specifically, in case of a web object there are several orders of magnitude between the download time of DFCP and TCP for increasing round-trip time (see Figure 4a). Considering the category of DVD it can be stated that, for low RTT values, the difference in download time is negligible, however, for high RTT values it gets more and more significant (see Figure 4b).

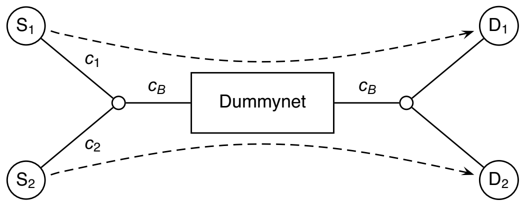
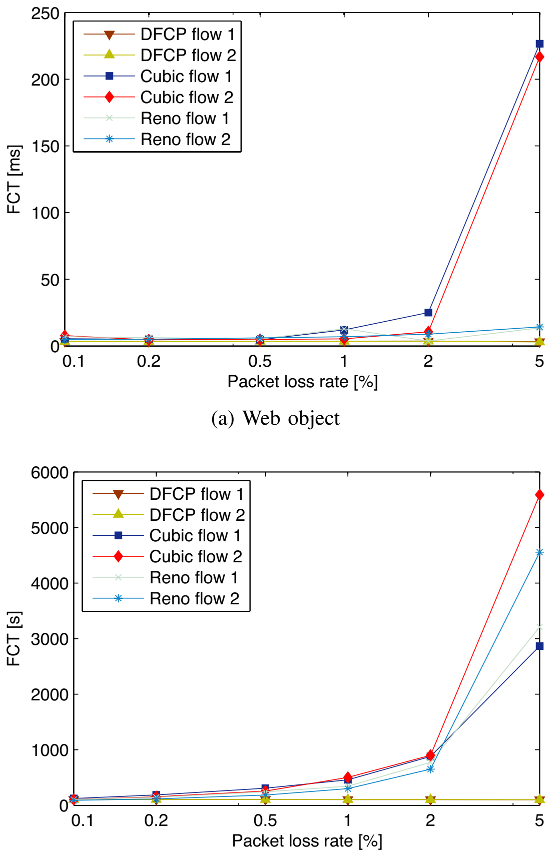


Fig. 5. Dumbbell topology with two source-destination pairs

We also performed experiments with two competing flows of the same type to study how the transport protocols share the bottleneck bandwidth. The second measurement setup can be seen in Figure 5 where all parameters were set similarly as described at the first dumbbell topology complemented by the condition $c_1 = c_2 = 1$ Gbps. The flows were started together and we used WFQ (Weighted Fair Queueing) as the scheduling method with equal weights (i.e. 50-50%).



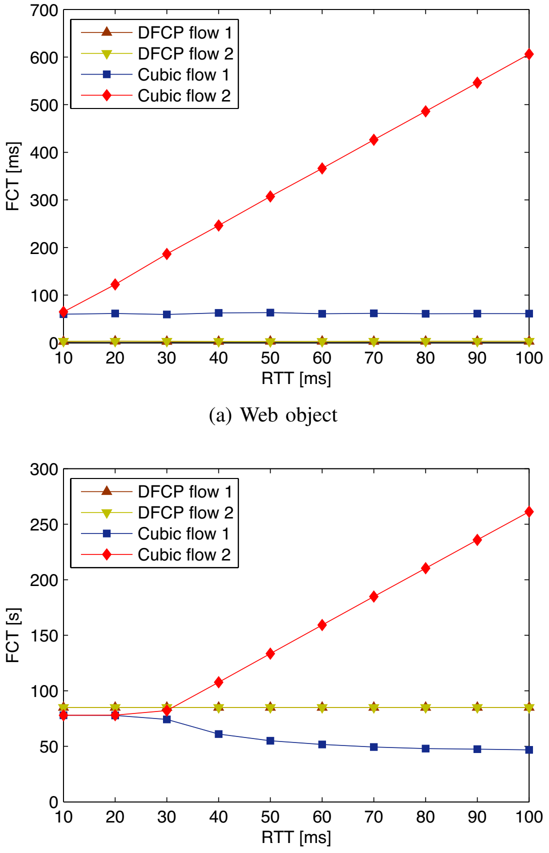
(a) Web object

(b) DVD

Fig. 6. Flow completion time for two competing flows with equal loss rate

In the followings we highlight two practically interesting cases. On the one hand, the case when the packet loss rate is equal for each flow, and on the other hand, the case when the round-trip time has different values. The former is depicted in Figure 6 where the redundancy parameter of DFCP was adjusted to 5% of packet loss. We can observe that the difference between the flow completion times of the two flows of the same type for a web object is quite small. Therefore, each transport protocol behaves in a quasi-fair way, but equal bandwidth sharing is only experienced in case of DFCP (see Figure 6a). It is also important to note that the download time of DFCP is independent of the packet loss rate. Considering the category of DVD, the two TCP variants become unfair at high loss rates in case of long data transfers (see Figure 6b).

Figure 7 shows the flow completion time for two competing DFCP and TCP Cubic flows where the first flow has a fixed RTT of 10 ms and the delay of the second flow is varied between 10 and 100 ms. We observed that the results for TCP Reno were quite the same as in case of TCP Cubic, hence only the latter was depicted. Looking at Figure 7a one can see that in case of a web object DFCP produces excellent results. It does not only achieve 20 times faster download than TCP even in the worst case, but also provides equal shares of the available bandwidth for the competing flows, thus both DFCP flows have nearly the same download time. If we transfer a full DVD, the two TCP flows behave in a fair way, but only for RTT values less than 20 ms (see Figure 7b). In contrast, DFCP flows attain equal download time in the whole range



(a) Web object

(b) DVD

Fig. 7. Flow completion time for two competing flows with the one having a fixed RTT of 10 ms and the other one having an RTT varied between 10 and 100 ms

since DFCP protocol is insensitive to high RTTs compared to TCP. We note that the difference in the flow completion times of DFCP and TCP flows for RTT values less than 20 ms is due to the redundancy used in DFCP.

## V. Conclusion

The important issue of efficiently transferring flows via the Internet regarding different transport protocols has been addressed in this paper. We carried out a performance comparison study of recent TCP versions and our newly designed and implemented DFCP protocol. The analysis focused on the transient behavior and the flow completion times of short-lived and long-lived flow transfers. The results demonstrated the outstanding performance of DFCP compared to TCPs in various network conditions. We pointed out that many applications can benefit from the transfer mechanism of DFCP, which can also provide an alternative way for improving web performance, just to mention one of the most important examples. The analysis also highlighted the main drawbacks of the currently used TCP versions and it motivates research on transport protocols for Future Internet based on different principles. Our future plans include the further development and evaluation of such new concepts, especially the DFCP protocol.

## Acknowledgment

This work was partially supported by the European Union and the European Social Fund through project FIRST (grant no. TÁMOP-4.2.2.C-11/1/KONV-2012-0001) organized by ETIK Debrecen. The research also received support from the High Speed Networks Laboratory (HSNLab) at Budapest University of Technology and Economics.

## References

[1] S. Floyd, K. Fall, "Promoting the Use of End-to-End Congestion Control in the Internet", *IEEE/ACM Transactions on Networking*, vol. 7, no. 4, pp. 458–472, 1999.

[2] 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.

[3] 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.

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

[5] 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.

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

[7] 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.

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

[9] S. Molnár, Z. Móczár, B. Sonkoly, Sz. Solymos, T. Csicsics, "Design and Performance Evaluation of the Digital Fountain based Communication Protocol", *Technical Report*, 2012.
http://hsnlab.tmit.bme.hu/∼molnar/files/DFCPTechReport.pdf

[10] A. Medina, M. Allman, S. Floyd, "Measuring the Evolution of Transport Protocols in the Internet", *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 2, pp. 37–52, 2005.

[11] N. Dukkipati, N. McKeown, "Why Flow-Completion Time is the Right Metric for Congestion Control", *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 1, pp. 59–62, 2006.

[12] S. Molnár, Z. Móczár, "Three-dimensional Characterization of Internet Flows", *Proceedings of the 2011 IEEE International Conference on Communications*, pp. 1–6, Kyoto, Japan, 2011.

[13] 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.

[14] L. Schrage, "A Proof of the Optimality of the Shortest Remaining Processing Time Discipline", *Operations Research*, vol. 16, no. 3, pp. 687–690, 1968.

[15] S. Sundaresan, N. Magharei, N. Feamster, R. Teixeira, S. Crawford, "Web Performance Bottlenecks in Broadband Access Networks", *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, no. 1, pp. 383–384, 2013.

[16] D. Cavendish, K. Kumazoe, M. Tsuru, Y. Oie, M. Gerla, "CapStart: An Adaptive TCP Slow Start for High Speed Networks", *Proceedings of the 1st International Conference on Evolving Internet"*, pp. 15–20, Cannes, France, 2009.

[17] 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.

[18] S. Floyd, T. Henderson, A. Gurtov, "The NewReno Modification to TCP's Fast Recovery Algorithm", *RFC 3782, IETF*, 2004.

[19] Dummynet Network Emulator, http://info.iet.unipi.it/∼luigi/dummynet/

[20] HTTP Archive, http://www.httparchive.org/interesting.php