Multi-Platform Performance Evaluation of Digital Fountain Based Transport

Zoltán Móczár[†*], Sándor Molnár[†*], Balázs Sonkoly[†]
[†]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: {moczar, molnar, sonkoly}@tmit.bme.hu

***Abstract*—One of the key questions for future networks is how to efficiently transfer data traffic generated by versatile applications in heterogeneous and fast changing environments. The lesson learned from the history of the Internet is that congestion control performed by TCP can be a solution. However, due to the limitations of TCP versions a novel paradigm applying fountain code based transfer seems to be a promising alternative. In this paper we address this idea by investigating our recently developed transport protocol called Digital Fountain based Communication Protocol (DFCP). The operation of DFCP is validated on three different testing platforms including our laboratory testbed, the Emulab network emulation environment and the ns-2 network simulator. We present and discuss the results of a comprehensive performance evaluation study obtained on multiple platforms by comparing DFCP to widely used TCP versions, as well as using various network topologies and settings. The results point out that digital fountain based transport has many advantages over traditional TCP.**

I. Introduction

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 transfers more than 90% of Internet traffic. The success of TCP was not even questioned until the fast development of networks, mobile devices and user applications resulted in heterogeneous and complex environments with a huge amount of applications in the last decades. In order to fit these changes significant research was carried out to further develop TCP, and therefore, several different TCP versions have been proposed [1], [2], [3]. However, it turned out that it will be very difficult to modify TCP to work efficiently as a universal transport protocol.

In the recent years an alternative paradigm was suggested, which motivates to rethink the concept of future transport protocols. An interesting idea was presented by GENI (Global Environment for Network Innovations) [4], recommending the omission of congestion control and promoting erasure coding schemes instead to handle congestion and its consequences. Unfortunately, no realization or further refinement of this concept has been published so far with some exceptions that we will discuss in the followings. Raghavan and Snoeren suggested a decongestion controller and investigated its properties in [5]. Bonald et al. studied the network behavior in the absence of congestion control [6]. We emphasize their astonishing result, that is, operating a network without congestion control does not result in congestion collapse in many cases. López et al. presented a fountain based protocol using game theory [7]. They found that a Nash equilibrium can be obtained, and at this equilibrium, the performance of the network is close to the performance experienced when all hosts use TCP. Botos et al. proposed a modified TCP for high loss rate environment by utilizing rateless erasure codes [8].

In this paper we study the possibility of applying a fountain code [9] based transport protocol for data transfer instead of congestion control based TCP. Hence, we have designed and developed a new protocol called Digital Fountain based Communication Protocol (DFCP), which was introduced in [10] together with the first analytical, simulation and testbed results. This paper gives a detailed discussion about the novel data transfer paradigm and presents the thorough validation of DFCP on three different testing platforms including our laboratory testbed, the Emulab network emulation environment [11] and the ns-2 network simulator [12]. We believe that, to claim profound results, a new protocol has to be investigated on both simulation and testbed platforms. On the one hand, simulation environments often hide those effects that only appear in real-world measurements. On the other hand, the results obtained by testbed measurements may highly depend on the hardware components, which can lead to the loss of generality. DFCP has been implemented in the Linux kernel and we measured its performance in our laboratory testbed and in the Emulab environment. Additionally, we used the Network Simulation Cradle (NSC) [13] framework to wrap our existing DFCP kernel code into ns-2 for simulation analysis.

In this work we also carried out an extensive evaluation to understand the performance characteristics of DFCP in comparison with TCP. We investigated many important properties such as loss and delay tolerance, buffer size demand, fairness behavior in case of different schedulers, the performance in multi-bottleneck networks, as well as scalability. This study can unfold the advantages of DFCP over TCP and help us to find its applicability for future networks. In order to get sound results the evaluation was done on multiple platforms and different topologies (dumbbell and parking lot) by comparing DFCP to widely used TCP versions (TCP Cubic and TCP NewReno with SACK) in various network conditions.

The paper is organized as follows. First, in Section II we introduce the envisioned future network architecture based on our new transport protocol. Then Section III gives the main properties of DFCP with a thorough validation on three differ-

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.

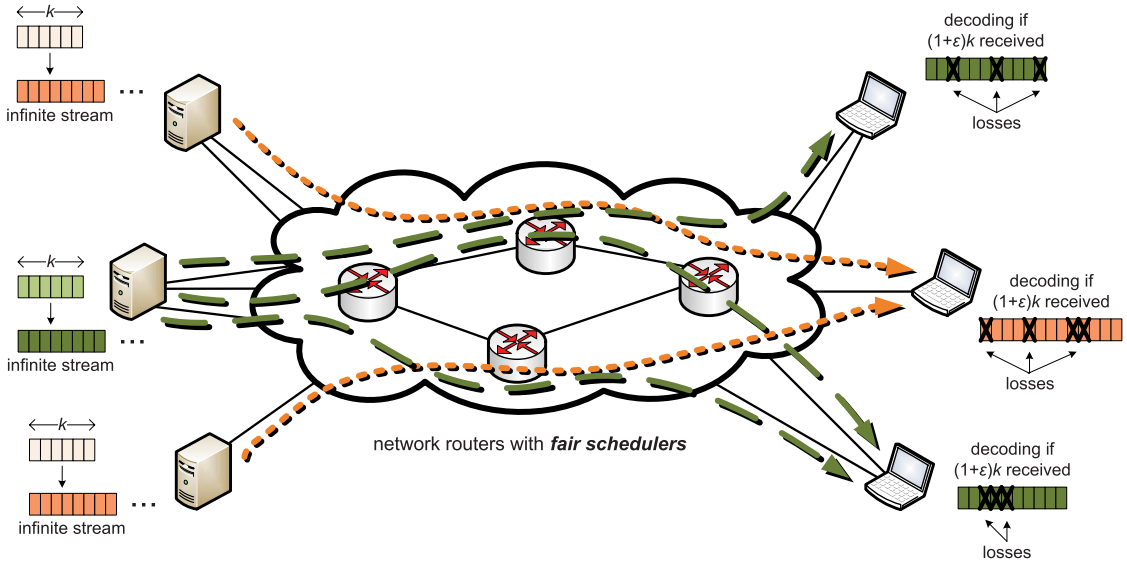


Fig. 1. The network architecture utilizing digital fountain based transfer

ent testing platforms. The comparative performance evaluation study of DFCP and different TCP versions can be found in Section IV. Finally, Section V concludes the paper.

II. Digital Fountain Based Data Transfer

This section is devoted to envision a network architecture relying on digital fountain based transfer and highlight the benefits of this approach. The main component of the architecture is a novel transport mechanism, which provides reliable transmission by efficient erasure coding and inherently makes it possible to get rid of congestion control and all related tasks at the transport layer. We have recently implemented this key component in the Linux kernel as a new transport protocol called Digital Fountain based Communication Protocol (DFCP). For detailed discussion about the design principles and the operating mechanism, please see [14], [10].

DFCP uses erasure coding schemes to recover lost packets instead of traditional retransmissions. Applying this approach yields hosts sending at "maximal rate", thus the network can easily be driven to a state with heavily congested, fully utilized links. However, preliminary works have shown that under some realistic assumptions this behavior does not cause problems, furthermore, the mechanism could exhibit beneficial properties with several respects. In DFCP, we propose the use of Raptor codes [15] to cope with packet losses as an efficient forward error correction mechanism, which is an extension of LT codes with linear time encoding and decoding complexity. The suggested network architecture relying on *digital fountain based error correction* is shown in Figure 1. We have multiple senders communicating with the corresponding receivers by producing a potentially infinite stream of encoded symbols from the original message of size k. Each received packet at the destination host increases the probability of successful decoding, and once any subset of size $\lceil(1+\varepsilon)k\rceil$ encoded symbols arrive to the receiver, decoding can be performed successfully with high probability (here $\epsilon > 0$ denotes the amount of redundancy added to the original message). We note that, in realistic scenarios, only slightly more packets are required than the original size of the message. Sending at "maximal rate" does not necessarily mean that the maximum transmission capacity is available at the sender side. A naive approach like that can lead to an operational state when a huge number of packets is steadily transferred via some parts of the network but reaching a bottleneck they are dropped. This unnecessary wasting of available bandwidth can be avoided in several ways. On the one hand, the sender could perform passive or active measurement on the currently available capacity along its network path. This monitoring can be done continuously during the connection and makes the sender capable of adapting to variable network states. It is worth noting, that we do not need sophisticated measurement tools and accurate results on available capacity. By these mechanisms, the enormous amount of packet losses can be controlled and held in a reasonable domain. On the other hand, network nodes can also assist in this operation. For example, network routers could inform the sender about the actual link utilization by different means. Another potential solution is based on SDN (Software Defined Networking), where the network domains have dedicated central controllers with central knowledge regarding the domains. Thus, they could provide information about the available bandwidth to the senders.

The envisioned architecture with our transport mechanism can leverage the redundancy in network topology using *multiple paths* simultaneously between the end hosts, which can connect to the network with multiple interfaces. By this approach, we can provide network resiliency, more efficient utilization and load balancing among redundant paths as well. Moreover, this framework supports not only unicast type traffic but inherently provides efficient solutions for *multicast* and *broadcast* services. Additionally, the more challenging n-to-1 and n-to-n communication patterns including multiple servers can also be realized in a straightforward manner due to the beneficial properties of the coding based approach as it does not matter which part of the message is received, and it

can be guaranteed that each received block provides extra information. Evidently, to fulfill this requirement, we need a careful protocol design with adequate coding scheme. Based on this architecture, many services such as multimedia, video streaming and broadcast services can be provisioned efficiently. The important issue of ensuring *fairness* among competing flows must also be solved. To this end, we suggest the use of *fair schedulers* in the network nodes. Several implementations approximating the ideal fair scheduling, such as Deficit Round Robin (DRR) [16], are available and can be configured easily in routing nodes. The feasibility of this solution is supported by the scalability of per-flow fair queueing [17]. An approach like this makes it possible to decouple fairness control from the transport protocol. Fairness can be treated in an orthogonal way by dedicated, changeable and easily configurable modules in network nodes. Invoking fair schedulers, gives the chance of using traditional TCP versions as well in the same network together with DFCP. It supports the deployment of the new transport mechanism as inter-protocol fairness can be controlled in several ways.

III. Validation on Multiple Platforms

Performance evaluation of a transport protocol in practice requires using different tools to get a clear picture about its behavior and specific properties, and to draw right conclusions. Even so, most researchers choose only one way to investigate their proposed protocols, namely simulation or testbed measurements. Especially for novel protocols and algorithms it can be misleading due to the unique nature of such environments. On the one hand, the main risk of relying only on simulation results is the fact that simulation environments are far from realistic in most cases, thus many real-world factors can easily be neglected [18], [19]. On the other hand, performing only testbed measurements can also lead to the loss of generality, because special hardware components can affect the results. In addition, building a network testbed is a time-consuming process, and measurements are very difficult to repeat as well [20], [21].

Since DFCP is based on a novel paradigm, it is crucial to ensure that our performance evaluation results are reliable and the conclusions are valid. In order to fit these requirements we carried out a validation study on multiple platforms including our laboratory testbed, the Emulab network emulation environment [11] and the ns-2 network simulator [12]. In this section the network topologies and scenarios are presented, and the description of these platforms is given focusing on the settings and parameters used in the test scenarios. Finally, we show that DFCP performs in a similar way in these environments providing a strong evidence for the operability of our protocol.

A. Network Topologies and Scenarios

The performance of DFCP was evaluated on different network topologies including the simple dumbbell topology and the more complex parking lot topology frequently used in the literature for experiments [22]. The dumbbell topology consisting of N source-destination pairs can be seen in Figure 2. First, we experimented with a single flow ($N = 1$) to reveal the ability of DFCP to resist against varying delay and packet loss rate parameters of the network. In this case the bottleneck link capacity (c_B) was set to 1 Gbps. Furthermore, we studied the fairness properties of DFCP by using two source nodes ($N = 2$). The main purpose was to observe how DFCP behaves in a situation when two concurrent flows compete for the available bandwidth determined by the bottleneck link. In this scenario both the access links (a_1, a_2) and the bottleneck link (B) had a capacity of 1 Gbps. Regarding scalability we investigated the performance and fairness stability of DFCP for increasing number of flows ($N = 10, 20, \ldots, 100$) and bottleneck bandwidth ($c_B = 0.1, 1, 10$ Gbps).

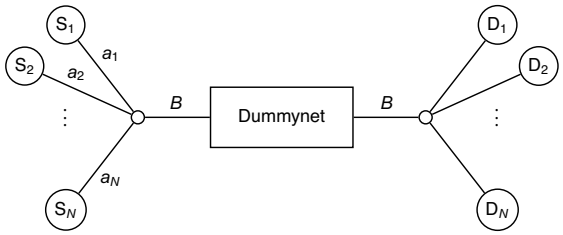


Fig. 2. Dumbbell topology with N source-destination pairs

The scenarios described above made possible to explore the fundamental features of DFCP and its scalability. Beyond these experiments DFCP was studied in a more realistic environment as well. Figure 3 depicts a parking lot topology with three sender and receiver nodes, which contains two bottleneck links. In a real network multiple bottlenecks are common, and therefore, it is indispensable to evaluate how a transport protocol performs in such conditions. In these tests the capacity was 1 Gbps for each access link (a_1, a_2, a_3), and the bottleneck link capacities (c_{B_1}, c_{B_2}) were set to different values as discussed in the following sections.

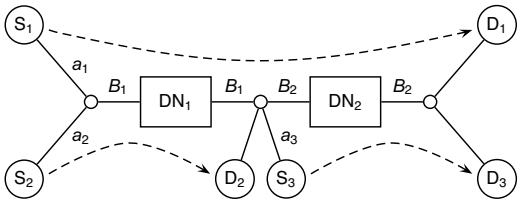
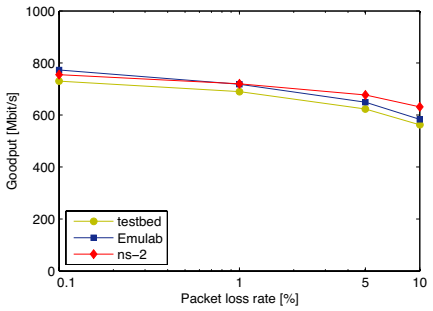


Fig. 3. Parking lot topology with three source-destination pairs

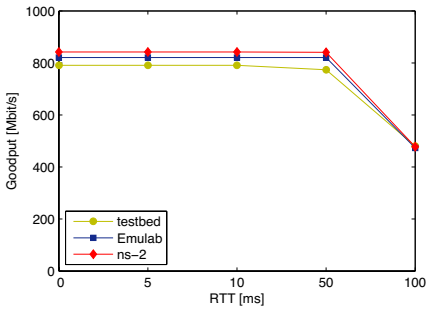
Measurements lasted for 60 seconds in the most scenarios (except if mentioned otherwise), and the results were obtained by excluding the first 15 seconds in order to ignore the impact of transient behavior of the investigated transport protocols. In case of multiple flows they were started at the same time, and for scheduling discipline WFQ (Weighted Fair Queueing) was applied by default with equal weights. However, we also experimented with other fair schedulers like SFQ (Stochastic Fair Queueing) and DRR (Deficit Round Robin), as well as with DropTail which is the simplest queue management mechanism available in today's network routers.

B. Test Environments

To validate the performance evaluation results the test scenarios were executed on the following three different platforms independently: (1) our laboratory testbed, (2) the Emulab network emulation environment and (3) the ns-2 network simulator.



(a) The impact of packet loss rate on the goodput performance



(b) The impact of round-trip time on the goodput performance

Fig. 4. The fundamental features of DFCP

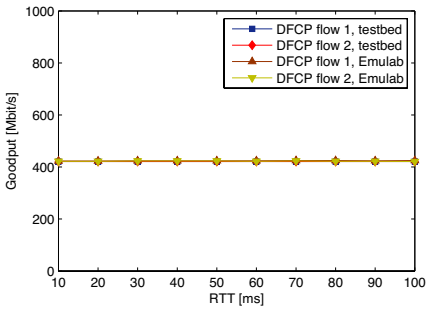


Fig. 5. Bandwidth sharing between two competing DFCP flows with different RTTs

The *laboratory testbed* consisted of senders, receivers and a Dummynet network emulator, which was used for simulating various network parameters such as buffer size, bandwidth, delay and packet loss probability [23]. Each test computer was equipped with the same hardware components [14]. The second platform we worked on, *Emulab*, is a network testbed giving researchers a wide range of environments in which to develop, debug and evaluate their systems [11]. The measurement setup was identical to the one used in our laboratory testbed for each test scenario. The type of the sender and receiver nodes was *pc3000* according to the Emulab label system, and the network emulators were run on *d710* type nodes [14]. Similarly to the testbed measurements our modified kernel including the implementation of DFCP was loaded into the test computers. The third tool was the *ns-2 network simulator* to validate DFCP, which is widely used by researchers to try out and evaluate their new methods [12]. Since the first prototype of DFCP has been implemented in the Linux kernel, we had to find a way to simulate our protocol directly through the network stack of Linux. In fact, there are some tools available for this purpose, but only a few of them can provide reasonable accuracy and efficiency, as well as support a wide range of operating systems and kernel versions [24]. Focusing on these requirements we chose Network Simulation Cradle (NSC), which is a framework for wrapping kernel code into simulators allowing the simulation of real-world behavior at little extra cost [13]. NSC supports the simulation of the network stacks of many operating systems such as FreeBSD, OpenBSD, lwIP and Linux. This tool has been validated by comparing situations using a test network with the same situations in the simulator, and it has been shown that NSC is able to produce extremely accurate results. Moreover, it has been ported to several network simulators including both ns-2 and ns-3. Although, NSC is an excellent tool for simulating different TCP versions and new TCP-like transport protocols, we had to carry out a challenging work to get NSC able to handle DFCP, which is based on a completely different paradigm compared to the principles applied by TCP.

C. Validation Results

In this part we present the validation results performed on the three different platforms discussed previously. The performance of DFCP was measured in terms of goodput, which is a well-known and widely used performance metric in networking, and it gives the number of useful data bytes successfully transferred per second. On the figures of the next sections the mean goodput is given calculated for the measurement period by excluding the transient phase as mentioned before.

Figure 4 illustrates the main features of DFCP introducing its high resistance to varying network conditions such as packet loss rate and round-trip time (RTT). These measurements were carried out on a dumbbell topology with one source-destination pair (see Figure 2). Figure 4a shows the impact of packet loss rate on the goodput performance of DFCP. It is important to investigate this aspect since TCP is very sensitive to packet loss resulting in a quick performance degradation for increasing loss rate. The figure clearly indicates that DFCP can operate efficiently even in high loss rate environments using optimal redundancy. 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. Figure 4b shows how the goodput performance of DFCP affected by the round-trip time parameter of the network. One can see that it achieves outstanding performance in a network with high delay, because goodput drops slowly as the round-trip time increases. Since the curves depicted in Figure 4 have very similar characteristics for the three platforms, we can state that these advantageous features of DFCP are validated.

Figure 5 presents how two DFCP flows having different round-trip times share the available bandwidth (see Figure 2), which is a common situation in real networks often referred to as RTT fairness problem in the context of transport protocols. It is an important property since traditional TCPs are unfair in the sense that the flow with lower RTT receives more bandwidth than the flow having higher RTT. In this scenario flow 1 had a fixed RTT of 10 ms, and the delay of flow 2 was increased from 10 to 100 ms. The figure shows that DFCP behaves in a fair way on different platforms since both flows get an equal share of the bottleneck bandwidth irrespective of their RTTs.

The results of Figure 6 were obtained on the parking lot topology illustrated in Figure 3. The scenario was designed to study the behavior of DFCP in a multi-bottleneck environment. The capacity of the first bottleneck link (B_1) was set to 1 Gbps, and the second bottleneck link (B_2) had a capacity of 500 Mbps. The figure depicts the goodput of the three DFCP flows as the function of the RTT experienced on B_2 while the RTT of B_1 is fixed at 10 ms. We can observe that flow 1 and flow 3 receive an equal portion of the bandwidth available

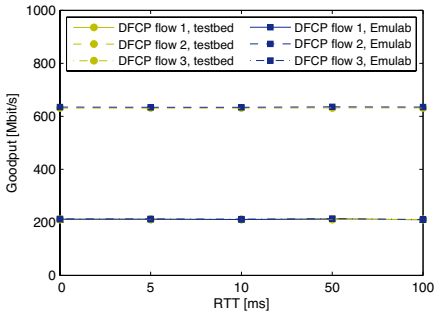
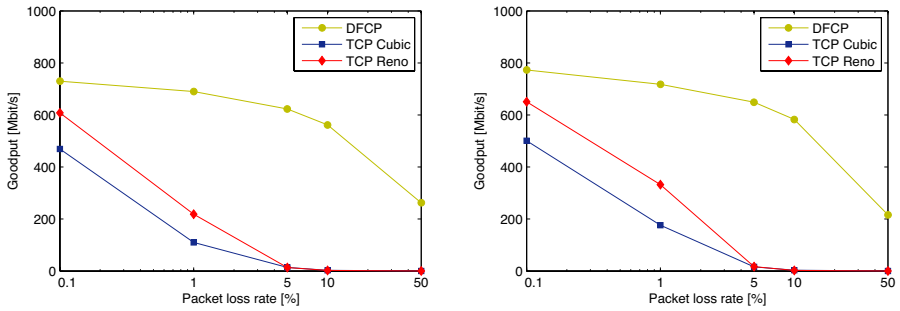


Fig. 6. The performance of DFCP in a network with multiple bottlenecks



(a) Testbed (b) Emulab

Fig. 7. The performance of DFCP and TCPs in a lossy environment

on B_2. Since the rate of flow 1 is limited by the capacity of B_2, flow 2 gets more bandwidth than flow 1 utilizing the available bandwidth of B_1. Therefore, each bottleneck link becomes fully utilized and is shared fairly by DFCP flows.

IV. Comparative Performance Evaluation

In this section we present a comprehensive performance analysis study carried out on three testing platforms by comparing DFCP to different TCP versions, namely TCP Cubic [25] which is the default congestion control algorithm in the Linux kernel and TCP NewReno [26] with SACK option.

A. Loss and Delay Tolerance

One of the main beneficial properties of DFCP can be seen in Figure 7. It demonstrates that DFCP is much more resistant to packet loss than TCP Cubic and Reno if optimal redundancy is used. The difference in goodput is already considerable for 0.1% of packet loss, but for increasing loss rate DFCP highly outperforms both TCP variants. For example, for 1% of packet loss the ratio between the goodput obtained by DFCP and TCP Reno is about 3, and this ratio is 6 for TCP Cubic. When the loss rate attains 10%, DFCP gets more than 250 times faster compared to TCPs, and it works efficiently even in the case of extremely high loss (50%) in contrast to TCPs, which are unable to operate under these network conditions. Note that the performance characteristics of the investigated transport protocols seem to be very similar in our laboratory testbed and in the Emulab environment, which can be considered as a validation of the results.

Figure 8 shows the performance comparison results of DFCP and TCPs for varying round-trip time. The figure illustrates that in the RTT interval 0–10 ms TCP versions perform better than DFCP in terms of goodput, but the difference is negligible and it is due to the coding overhead. Nevertheless, for delay values greater than 10 ms DFCP achieves significantly higher transfer rate compared to TCP Cubic and Reno. Since the typical value of round-trip time in a real network exceeds 10 ms [27], DFCP can function more efficiently than TCP in such conditions.

B. Buffer Size Demand

It is a well-known fact that the buffer size demand of TCP is at least of root order in the number of competing flows [28]. This requirement imposes a significant challenge in all-optical networks where only very small buffer sizes can be realized due to both economical and technological constraints [29]. Figure 9 demonstrates how the performance of DFCP and TCPs is affected by the buffer size. In this scenario the round-trip time was fixed at 10 ms and no packet loss was simulated. The buffer size is given in packets, and the vertical axis represents the performance utilization of the investigated transport protocols. Performance utilization is the ratio (expressed in percentage) between the goodput can be obtained with a particular buffer size and the maximum goodput can be achieved when the buffer size is set as high as to exclude it from the limiting factors. We can see that with a buffer size of 1000 packets each protocol is able to realize maximum performance utilization. However, by decreasing the buffer size the performance of TCP variants drops considerably. For example, with a small buffer of 50 packets TCP Cubic and TCP Reno can only work at a reduced transfer rate, 92% and 79% of the ideal case, respectively. In contrast, DFCP can bring out the maximum performance not only for large buffers, but also for small ones, and thanks to this property the transport mechanism of DFCP is closely aligned to the concept of all-optical networking.

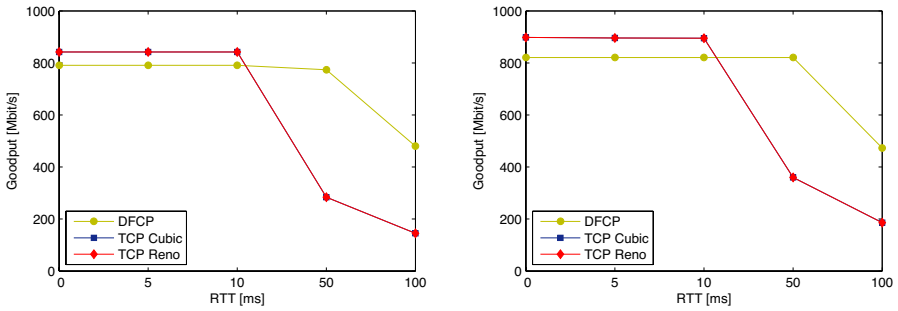
C. Bandwidth Sharing with Different Queueing Disciplines

Another important aspect that need to be revealed and investigated is how a transport protocol shares the available bandwidth of a bottleneck link among competing flows often called as fairness property. In our experiments we used the Jain's index as the fairness measure, which is one of the most popular and widely accepted fairness indices in the literature [30]. Jain's index can be calculated by the following formula:

$$JI = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2}$$

where x_i denotes the normalized throughput (or goodput) of flow i and n is the number of flows. It returns a value between 0 and 1 where a higher value indicates a higher degree of fairness. As widely known, standard TCP cannot provide an equal portion of the bottleneck bandwidth for competing flows with different round-trip times [31] due to its AIMD mechanism [30].

Figure 10 depicts the goodput for two competing DFCP and TCP Cubic flows. The delay of flow 1 was fixed at 10 ms, and for flow 2 we varied the delay parameter between 10 and 100 ms. Since the results for TCP Reno were quite the same as



(a) Testbed (b) Emulab

Fig. 8. The performance of DFCP and TCPs for varying RTT

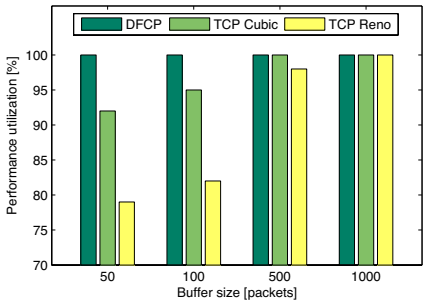
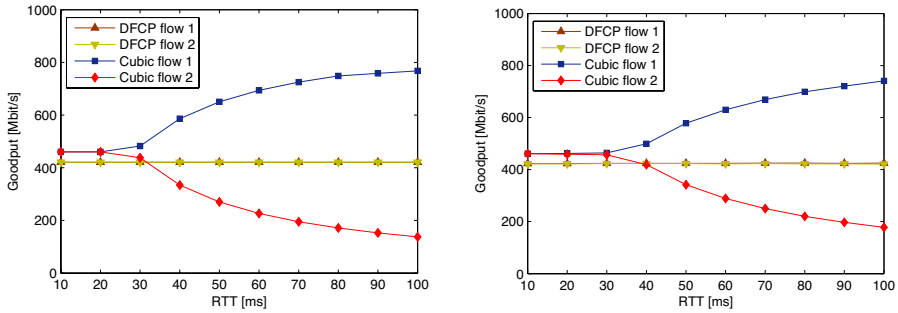


Fig. 9. The impact of buffer size on the performance of DFCP and TCPs



(a) Testbed (b) Emulab

Fig. 10. The performance of DFCP and TCP in case of two competing flows

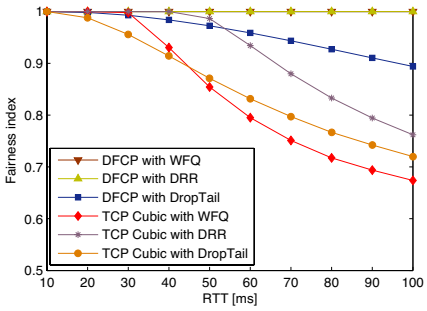


Fig. 11. Intra-protocol fairness with WFQ, DRR and DropTail queueing

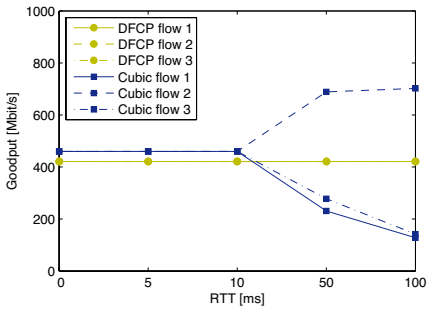
in case of TCP Cubic, only the latter was plotted. The figure shows that the bottleneck link capacity is equally shared by the two TCP flows for RTT values less than 20 ms regarding our testbed measurements (see Figure 10a). However, for RTTs greater than 20 ms the goodput of flow 2 starts to decrease, and as a result, flow 1 with lower RTT can gain access to a greater portion of the available bandwidth, indicating the unfairness behavior of TCP. In contrast, DFCP flows achieve perfect fairness as they share the bottleneck capacity equally and they are much less sensitive to the round-trip time compared to TCP. We note that the difference can be observed in the goodput of DFCP and TCP flows for RTT values less than 20 ms is due to the coding overhead. Comparing the results obtained on different platforms, we experienced that the behavior of DFCP in Emulab is as same as in our laboratory testbed while TCP Cubic achieves a slightly better fairness in Emulab.

In our proposed future network architecture the best solution to realize fairness is to use fair schedulers as we mentioned in Section II. In fact, unlike TCP the transfer mechanism of DFCP cannot guarantee fairness at the host side. Therefore, the only way is to perform this task by network routers. However, in this context there are some open questions to be answered. On the one hand, a plenty of fair scheduling algorithms have been worked out during the last two decades, but only a few are available in today's routers. So, the natural question is which one to choose? On the other hand, in most routers the DropTail queue management policy is applied by default as it is the simplest algorithm, but it does not eligible for providing fairness. How does DFCP perform in such conditions? To answer these questions we extend our fairness analysis by investigating other queueing mechanisms than WFQ in ns-2.

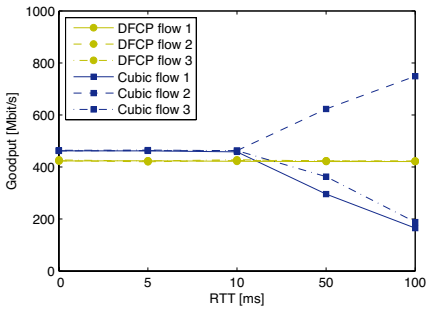
Figure 11 shows the fairness measure when different schedulers are used. The results clearly show that DFCP can guarantee perfect fairness for the two competing flows independently of their RTTs if fair schedulers are used. Moreover, DFCP achieves better fairness than TCP even with the much more simple DropTail algorithm. Concluding the observations we can say that in a realistic environment with typical network parameter values DFCP can obtain a higher degree of fairness compared to TCP for each queueing discipline.

D. Performance in Multi-Bottleneck Environment

Figure 12 presents the performance comparison of DFCP and TCP Cubic carried out on the parking lot topology illustrated in Figure 3 by starting three concurrent flows. In this test scenario the capacity was set to 1 Gbps for both bottleneck links denoted by B_1 and B_2. The round-trip time was fixed at 10 ms on B_1, but it was increased on B_2 from 0 to 100 ms. Looking at the figure we can make the following observations. Until the round-trip time experienced on B_2 attains 10 ms, both DFCP and TCP Cubic share the bottleneck bandwidth of B_1 and B_2 in a fair way. However, for higher delay values TCP Cubic gradually becomes unfair due to the fact pointed out in this section, namely, TCP is sensitive to round-trip time. As the goodput obtained by flow 1 and flow 3 drops for increasing RTT (since they go through B_2), flow 2 with lower RTT receives more and more bandwidth. Accordingly, TCP Cubic does not provide fairness between flow 1 and flow 2 having different RTTs. Moreover, in this case the available capacity of B_2 is also shared unequally, and hence, flow 1 and flow 3 achieve different goodput performance. As we mentioned earlier it is an undesirable behavior, and the results show that DFCP can solve this issue by providing perfect



(a) Testbed



(b) Emulab

Fig. 12. The behavior of DFCP and TCP in a multi-bottleneck network with varying delay

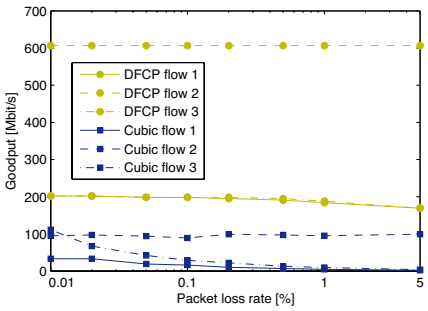


Fig. 13. The behavior of DFCP and TCP in a multi-bottleneck network with varying packet loss

fairness for each flow independently of their RTTs thanks to its robustness to varying network conditions.

Figure 13 depicts the results of a similar test scenario for varying packet loss rate performed on the same parking lot topology. In this case the capacity was set to 1 Gbps and 500 Mbps for the bottleneck links denoted by B_1 and B_2, respectively. The loss rate was fixed at 0.01% on B_1, but it was increased on B_2 from 0.01% to 5%. The round-trip delay was set to 10 ms on both links. We can see that DFCP provides fair shares for the flows competing for the available bandwidth of B_2, and their goodput drops very slowly as the packet loss increases, resulting in the excellent utilization of both bottleneck links. In contrast, TCP Cubic ensures fairness for flow 1 and flow 3 only for packet loss rate greater than 1% where both flows become almost unable to transfer data. The goodput of flow 1 starts from a lower value than the goodput of flow 3, because flow 1 goes through both B_1 and B_2, and hence experiences a higher rate of packet loss. The link utilization achieved by the TCP variants is quite poor due to their sensitivity to this network parameter.

E. Scalability

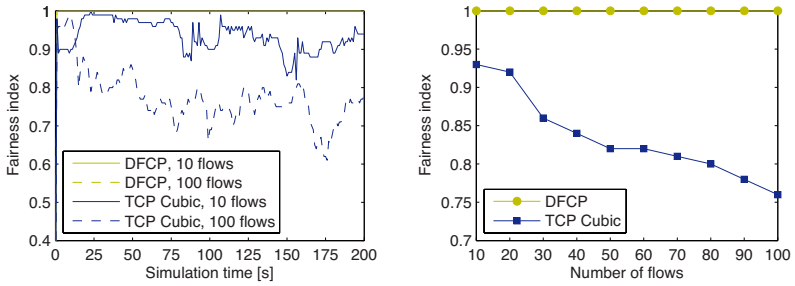
On a typical bottleneck link hundreds of flows compete for the available bandwidth, and the capacity of these links is continuously increasing due to the development of communication technologies. Good scalability is an important requirement for transport protocols meaning that they have to provide similar performance and fairness as the number of flows and the link capacity increase. The following simulations compare the scalability of two fundamentally different data transfer paradigms, TCP Cubic with DropTail queue management (current Internet) and DFCP with DRR scheduling (our concept). The results obtained for a 200 seconds long measurement period on the topology of Figure 2 with a 0.1 BDP buffer, and each flow experienced 100 ms of RTT.

Table I describes the performance scalability of the investigated transport protocols for different numbers of flows and link capacities. We computed the normalized aggregate goodput as the ratio of the aggregate goodput of concurrent flows and the maximum goodput can be achieved by a single flow. The normalized values are expressed in percentage and given for TCP Cubic and DFCP, respectively, separated by a slash mark. The results show that DFCP is able to gain the maximum performance irrespective of the number of flows and bottleneck bandwidth. In contrast, for TCP Cubic the normalized aggregate goodput increases with the number of flows, but decreases with the link capacity. For example, in case of a 100 Mbps link the maximum performance can be obtained by 50 competing flows, however, an increase in the link capacity by two orders of magnitude leads to a 5% performance degradation. Moreover, high capacity links cannot be fully utilized by a small number of flows since the round-trip time limits the transmission rate of individual flows. In this special case, 100 ms of RTT results in a goodput reduced to approx. 200 Mbps for each flow (see Figure 8), and hence the underutilization of the 10 Gbps link by 10 flows.

Figure 14 demonstrates the fairness scalability of DFCP and TCP Cubic in the function of time and with increasing number of flows. Figure 14a characterizes the fairness stability of the transport mechanisms for different numbers of flows. It is clearly shown that the Jain's index calculated for TCP Cubic significantly fluctuates over time, and an increase in the number of competing flows results in a higher degree of instability with lower mean fairness. Figure 14b illustrates the fairness measure for increasing number of flows. We emphasize that in this scenario each flow experienced the same delay to avoid the phenomenon of RTT unfairness. In spite of that the tendency is obvious for TCP Cubic: the larger the number of concurrent flows, the lower the fairness index. However, in contrast to all of these results DFCP can ensure fair bandwidth sharing on various time scales without suffering from stability issues and independently of the number of competing flows.

V. Conclusion

In this paper we investigated two alternative paradigms as possible data transport mechanisms for future networks: the digital fountain based DFCP and the congestion control based TCP. In order to draw solid conclusions we studied the operation of DFCP on various network topologies (dumbbell and parking lot) by multiple platforms including our laboratory testbed, the Emulab network emulation environment and the ns-2 network simulator. We also carried out a comparative performance evaluation study of DFCP and TCP on these platforms. We showed that the goodput performance of DFCP is significantly better than in case of the investigated TCP versions in a wide range of packet loss rates and round-trip times. The results also pointed out that DFCP is able to obtain maximum performance even with small buffers, which could make it attractive for all-optical networks. Moreover, DFCP provides fair bandwidth sharing among competing flows



(a) Fairness stability over time

(b) Fairness for increasing number of competing flows

Fig. 14. Fairness scalability

TABLE I
PERFORMANCE SCALABILITY

Bandwidth	Normalized aggregate goodput [%]		
	10 flows	*50 flows*	*100 flows*
0.1 Gbps	98 / 100	100 / 100	100 / 100
1 Gbps	96 / 100	98 / 100	99 / 100
10 Gbps	22 / 100	95 / 100	96 / 100

independently of their RTTs. Although, perfect fairness can only be achieved when fair schedulers (e.g. DRR) are used, DFCP can ensure better fairness than TCP even in the absence of any fair scheduler and if the simplest DropTail algorithm is applied. Finally, digital fountain based transport guarantees good scalability and stability as well, both in terms of performance and fairness for increasing number of flows and link capacity. The results suggest that it is a promising approach with numerous beneficial properties and a broad spectrum of possible applications.

REFERENCES

[1] 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.

[2] 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.

[3] B. Francis, V. Narasimhan, A. Nayak, I. Stojmenovic, "Techniques for Enhancing TCP Performance in Wireless Networks", *Proceedings of the 32nd International Conference on Distributed Computing Systems Workshops*, pp. 222–230, Macau, China, 2012.

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

[5] B. Raghavan, A. C. Snoeren, "Decongestion Control", *Proceedings of the 5th ACM Workshop on Hot Topics in Networks*, pp. 61–66, Irvine, CA, USA, 2006.

[6] 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.

[7] L. López, A. Fernández, V. Cholvi, "A Game Theoretic Comparison of TCP and Digital Fountain Based Protocols", *Computer Networks, Elsevier*, vol. 51, no. 12, pp. 3413–3426, 2007.

[8] 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.

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

[10] 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.

[11] Emulab Network Emulation Testbed, http://www.emulab.net/

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

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

[14] 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

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

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

[17] 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.

[18] K. Pawlikowski, H.-D. Joshua Jeong, J.-S. Ruth Lee, "On Credibility of Simulation Studies of Telecommunication Networks", *IEEE Communications Magazine*, vol. 40, no. 1, pp. 132–139, 2002.

[19] M. Bateman, S. Bhatti, "TCP Testing: How Well Does ns2 Match Reality?", *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications*, pp. 276–284, Perth, Australia, 2010.

[20] S. Floyd, E. Kohler, "Tools for the Evaluation of Simulation and Testbed Scenarios", *Technical Report, IETF*, 2006.

[21] M. P. Fernandez, S. Wahle, T. Magedanz, "A New Approach to NGN Evaluation Integrating Simulation and Testbed Methodology", *Proceedings of the 11th International Conference on Networks*, pp. 22–27, Saint-Gilles, Réunion Island, France, 2012.

[22] D. X. Wei, P. Cao, S. H. Low, "Time for a TCP Benchmark Suite?", *Technical Report, California Institute of Technology*, Pasadena, CA, USA, 2005.

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

[24] M. Lacage, "Experimentation Tools for Networking Research", *Ph.D. Thesis* (available at http://cutebugs.net/files/thesis.pdf), 2010.

[25] 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.

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

[27] S. Kaune, K. Pussep, C. Leng, A. Kovacevic, G. Tyson, R. Steinmetz, "Modelling the Internet Delay Space Based on Geographical Locations", *Proceedings of the 17th Euromicro International Conference on Parallel, Distributed and Network-based Processing*, pp. 301–310, Weimar, Germany, 2009.

[28] G. Appenzeller, I. Keslassy, N. McKeown, "Sizing Router Buffers", *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 281–292, 2004.

[29] H. Park, E. F. Burmeister, S. Bjorlin, J. E. Bowers, "40-Gb/s Optical Buffer Design and Simulation", *Proceedings of the 4th International Conference on Numerical Simulation of Optoelectronic Devices*, pp. 19–20, Santa Barbara, CA, USA, 2004.

[30] D.-M. Chiu, R. Jain, "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks", *Journal of Computer Networks and ISDN Systems*, vol. 17, no. 1, pp. 1–14, 1989.

[31] T. V. Lakshman, U. Madhow, "The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss", *IEEE/ACM Transactions on Networking*, vol. 5, no. 3, pp. 336–350, 1997.