Data Transfer Paradigms for Future Networks: Fountain Coding or Congestion Control?

Sándor Molnár[†*], Zoltán Móczár[†*], András Temesváry[†], Balázs Sonkoly[†], Szilárd Solymos[†], Tamás Csicsics[†]
[†]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}@tmit.bme.hu

***Abstract*—The research history of congestion control protocols has unveiled that it is difficult to find an optimal solution, which meets all the challenges of the evolving Internet. As an alternative paradigm recent studies suggest replacing congestion control with erasure coding techniques. In this paper we present a performance study of this approach comparing it with TCP based solutions. In order to investigate the new paradigm in details and also in practice, we have developed and implemented a novel transport protocol (Digital Fountain based Communication Protocol, DFCP) where an efficient digital fountain based erasure coding scheme plays the role of correcting packet loss. We present some analytical and simulation results together with our first performance comparisons in a testbed environment.**

I. Introduction

Since the first efficient reaction to the phenomenon of congestion collapse in the early Internet, congestion control, mostly performed by the Transmission Control Protocol (TCP), has played an important role in communication networks [1]. Several TCP versions have been developed in order to fit the ever-changing requirements of communication networks [2], [3]. Although, current high speed TCP variants provide efficient solutions for many network environments, they all fail to act as a universal mechanism considering heterogeneous and changing network conditions.

Concerning the limitations of TCP there is a significant justification to rethink the concept of this transport protocol and design it from scratch, omitting the main TCP-related features, most interestingly its congestion control mechanism. Some ideas have already been proposed and investigated where congestion control was not employed at all. One of these ideas was outlined by GENI (Global Environment for Network Innovations), which advocates a Future Internet without congestion control [4] by suggesting efficient erasure coding schemes to recover lost packets.

The idea to use coding in general for efficient networking is not new, and many approaches have already been proposed. There is no doubt, coding has a very high potential in many areas of data communication. For example, a mechanism called TCP/NC that incorporates network coding into TCP with only minor changes to the protocol stack is presented in [5]. According to this method the source transmits random linear combinations of packets currently found in the congestion window. Coding essentially masks losses from the congestion control algorithm and allows TCP/NC to react smoothly to them providing an effective solution for congestion control in lossy environments such as wireless networks. However, the idea of replacing congestion control with erasure coding schemes is a different approach and investigated only in a few papers as we briefly overview in the followings.

A decongestion controller was proposed by Raghavan and Snoeren who studied its benefits [6]. Bonald et al. analyzed the network behavior in the absence of congestion control [7]. Their surprising result is the confutation of the common belief that operating a network without congestion control necessarily leads to congestion collapse. López et al. investigated a fountain based protocol using game theory [8]. They showed that a Nash equilibrium can be achieved, and at this equilibrium, the performance of the network is similar to the performance obtained when all hosts comply with TCP. Botos et al. presented a transport protocol based on the modification of TCP for high loss rate environment using rateless erasure codes [9]. In their proposal the well-known slow-start and congestion avoidance algorithms of TCP are used, but some modifications are suggested to avoid the dramatic decrease of the sending rate in case of high packet loss. Kumar et al. proposed a transport protocol based on fountain codes for wireless networks and pointed out that regarding performance this approach can be beneficial in such environments [10]. As for now it seems that these studies have shown realistic potential of omitting congestion control functionality and applying erasure coding schemes in future networks. However, no detailed analysis has already been performed to make reliable conclusions yet and this is the motivation of our study.

In this paper we take the first practical steps towards a network architecture based on erasure codes instead of congestion control. To the best of our knowledge, this is the first time to implement this new concept while revealing feasibility issues and answering practical questions. We present both analytical and simulation results focusing on the performance of such a system. However, to deeply understand whether Future Internet can operate without congestion control and rely on rateless codes, a comprehensive analysis is needed not only in theory but also in practice, hence we have developed and implemented a novel transport protocol called Digital Fountain based Communication Protocol (DFCP) for this purpose. We also present our first performance comparison study of DFCP and TCP versions carried out in a testbed environment.

The paper is organized as follows. First, we give a brief overview of a possible Future Internet concept, which is built on DFCP in Section II. After that two loss models are investigated in Section III with emphasis on keeping QoS requirements assuming that network entities operate according to the mechanism of DFCP. Section IV provides interesting analytical results for modeling interrupted data transfer scenarios using realistic ON-OFF generated sources. Section V gives a summary about the design and implementation of DFCP. Our initial testbed measurement results including comparisons to the performance of two TCP versions for different network scenarios are presented and discussed in Section VI. Finally, Section VII concludes the paper.

II. Digital Fountain Based Architecture for Future Internet

Omitting congestion control from the transport protocol together with enabling maximal rate sending for all network entities may easily result in enormous packet loss due to the constant overload of finite network resources. We note, however, that if no congestion is experienced during a maximal rate sending scenario, then this concept yields the optimal solution as no other method could utilize the network better as it can. To cope with packet loss in case of potentially heavy congestion, we propose to employ efficient digital fountain based (rateless) codes [11] in end-to-end communication. As opposed to traditional erasure coding, rateless codes do not have a fixed coding rate. Instead, they can produce a potentially infinite stream of codeword, as long as it is necessary for the actual transmission. The decoder can recover the sent information from any subset of the encoded stream, which is only slightly larger than the original message. Accordingly, when the loss experienced in the network is inconsequential, receiver hosts only have to collect any fragments of the encoded stream until they obtain the proper amount of it. The first practical realization of universal rateless codes were the Luby Transform (LT) codes [11], but they failed to provide low complexity encoding and decoding operations. That is why we propose the use of Raptor codes [12] for the forward error correction mechanism. Being an extension of LT codes, they offer linear time encoding and decoding complexity. Specifically, for a message consisting of k symbols and any real $\varepsilon > 0$ redundancy parameter, a Raptor code can generate a potentially infinite encoded stream, from which any subset of size $\lceil (1+\varepsilon)k \rceil$ is sufficient to provide high probability decoding of the original message.

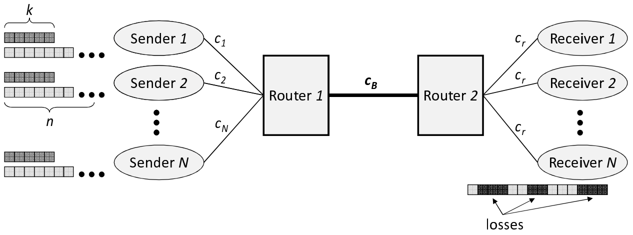


Fig. 1. The network architecture with N sender-receiver pairs

Figure 1 depicts the network architecture exploiting *digital fountain based error correction* in a simple dumbbell topology. The bottleneck link with capacity c_B is fed by flows connecting through the access links having capacities $c_1, c_2, \ldots, c_N$. According to our concept congestion control is omitted from the network, and end hosts send their data at maximal rates, which may easily result in high extent of loss during data transmission. However, this solution relying on Raptor codes is senseless to arbitrary extent of loss even in case of dynamically changing and/or burst loss characteristics. Sender processes produce a potentially infinite stream of encoded symbols from the original message of size k. Once any subset of size $\lceil (1+\varepsilon)k \rceil$ encoded symbols arrive to the receiver, successful decoding can be performed with high probability. If decoding fails, the receiver only has to wait to receive a bit more encoded symbols and try decoding again. Furthermore, additional encoded symbols increase the probability of successful decoding.

The use of the maximal rate sending mechanism arises the question of fairness. Competing flows could have different access rates to a shared bottleneck link, which asks for preventing eager hosts to starve less active ones. To solve the share allocation problem among flows sending at maximal rates we suggest to use *fair schedulers*. The implementation of an ideal fair scheduler can be a really difficult task, but we do not want to achieve ideal fairness. Instead, we advocate to use efficient approximate fair schedulers such as Deficit Round Robin (DRR), which are appropriate for this job, see [13]. The approximate fair sharing can be practically realized by such schedulers, because per-flow fair queueing is scalable and feasible [14], [15].

Since a larger set of encoded symbols needs to be received than the size of the original message, it is worth to introduce a goodput definition to measure the performance. Goodput is a well-known and widely used performance metric, which gives the number of useful data bytes (without any overhead) transferred per second. In case of employing an ideal fair scheduling mechanism and applying redundancy parameter ε during the transmission we can derive the following formula for the *goodput of flow* i using the notations of Figure 1:

$$G_i = \begin{cases} \frac{c_B}{(1+\varepsilon_i)N} & \forall j : c_j \geq \frac{c_B}{N} \\ \frac{c_i}{1+\varepsilon_i} & c_i < \frac{c_B}{N} \\ \dfrac{c_B - \sum_{k=1}^{N} I_{\{c_k < \frac{c_B}{N}\}} c_k}{(1+\varepsilon_i) \sum_{k=1}^{N} I_{\{c_k \geq \frac{c_B}{N}\}}} & \exists j : c_j < \frac{c_B}{N} \text{ and } c_i \geq \frac{c_B}{N} \end{cases}$$

where c_B denotes the capacity of the bottleneck link and N is the number of competing flows. In practice a digital fountain based protocol using Raptor code can easily be implemented with $\varepsilon \approx 0.05$ [16].

III. Analysis in Lossy Environment

Considering the fact that every arriving packet is useful for the decoding process in a fountain coding scheme, we analyze the transmission task in basic loss environments and give tight bounds on the number of required packets to be sent, keeping specific reliability requirements, simultaneously.

A. Independent-Loss Channels

In this section a lossy channel with independently and identically distributed random loss (Bernoulli loss) characteristics is investigated with emphasis on keeping a predefined success criterion during data transfer. Consider that a sender process would like k data packets to be delivered to a receiver via an imperfect link where each packet is dropped independently of the others with probability p and transmitted successfully with probability $1-p$. Let $X_{k,1-p}$ denote the number of packets required to be sent by the sender to fulfill this task. The random variable $X_{k,1-p}$ follows the negative binomial distribution with mean $k/(1-p)$ and standard deviation $\sqrt{kp}/(1-p)$. In case of a specific success criterion for the transmission task, the minimal value of sent packets, n should be determined, which satisfies the following reliability requirement:

$$\mathbf{P}(X_{k,1-p} \leq n^*) \geq \mathbf{P}_{sc}.$$

The value of n^* determines the *minimal number of packets* required to be sent in order to effectively deliver k packets to the receiver with $\mathbf{P}_{sc}$ success probability. Using the central limit theorem, for sufficiently large data transfers the success criterion can be formulated as follows:

$$\mathbf{P}\left(\frac{X_{k,1-p} - \frac{k}{1-p}}{\frac{\sqrt{kp}}{1-p}} \leq \frac{n^* - \frac{k}{1-p}}{\frac{\sqrt{kp}}{1-p}}\right) \geq \mathbf{P}_{sc}$$

where the transformed random variable is approximately a standard normal random variable, and by introducing $R = \mathbf{\Phi}^{-1}(\mathbf{P}_{sc})$ as a reliability factor according to the inverse of the standard normal distribution function, the minimal value of n is

$$n^* = \left\lceil \frac{k}{1-p} + \frac{R\sqrt{kp}}{1-p} \right\rceil.$$

In the above formula, the mean and the standard deviation of the negative binomial random variable $X_{k,1-p}$, in conjunction with the reliability factor R, determine a good approximation for the minimal number of required packets to be transmitted when k is sufficiently large.

Note that this data transmission task in the context of Raptor codes would only scale parameter k of the random variable $X_{k,1-p}$ to $k(\varepsilon) = \lceil (1+\varepsilon)k \rceil$ resulting in an other random variable following the negative binomial distribution with parameters $(k(\varepsilon), 1-p)$.

B. Burst-Loss Channels

Consider again the task when k number of packets should be delivered by a sender process to a receiver via an imperfect link, but now let the packet loss be characterized by the Gilbert model [17]. The Gilbert model has been commonly used to describe burst-loss channels often found in the Internet. The aim is to give a good bound on minimal packet number (n^*) that is required to be sent in order to fulfill a predefined success criterion on transmitting k packets effectively.

As Figure 2 shows, the Gilbert model has two states where residence times in each state follow geometric distributions with parameters p and q.

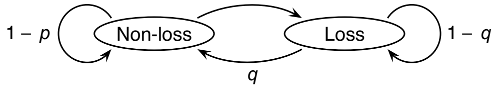


Fig. 2. The Gilbert model

Let $X[k]$ denote the total amount of lost packets until k packets successfully arrive to the receiver. In order to give a proper bound on the number of packets required to be sent, the probability distribution of $X[k]$ should be determined. In the presence of a success criterion ($\mathbf{P}_{sc}$) the aim is to find n^* such that

$$\mathbf{P}(k + X[k] \leq n^*) \geq \mathbf{P}_{sc}.$$

Note that $X[k]$ can be interpreted as the sum of k independently and identically distributed random variables $L_1, L_2, \ldots, L_k$ where L_i denotes the random size of a loss period between the successful delivery of the $(i-1)^{th}$ and i^{th} packets. The probability distribution of L_i can be expressed in the following form:

$$\mathbf{P}(L_i = l) = \begin{cases} 1-p & \text{if } l = 0 \\ p(1-q)^{l-1}q & \text{if } l > 0 \end{cases}.$$

The distribution of total loss $X[k]$ is then the convolution of the random variables $L_1, L_2, \ldots, L_k$ where L_i has mean p/q and standard deviation $\sqrt{p(2-p-q)}/q$. To cope with the problem of giving a good bound on n^* avoiding the calculation of the convolution of $\{L_i\}$ variables, the central limit theorem can be used, and for sufficiently large value of k we get that

$$\frac{\frac{L_1+L_2+\cdots+L_k}{k} - \frac{p}{q}}{\frac{\sqrt{p(2-p-q)}}{\sqrt{k}q}} = \frac{X[k] - \frac{kp}{q}}{\frac{\sqrt{kp(2-p-q)}}{q}} \approx N(0,1).$$

Therefore, for sufficiently large value of k, the success criterion with success probability $\mathbf{P}_{sc}$ can be written as follows:

$$\mathbf{P}\left(\frac{X[k] - \frac{kp}{q}}{\frac{\sqrt{kp(2-p-q)}}{q}} \leq \frac{n^{**} - \frac{kp}{q}}{\frac{\sqrt{kp(2-p-q)}}{q}}\right) \geq \mathbf{P}_{sc}.$$

According to the definition of $X[k]$, n^{**} denotes the necessary number of packets required to be sent beyond the k packets, which must be delivered to the receiver. Hence, $n^* = n^{**} + k$ stands, and introducing $R = \mathbf{\Phi}^{-1}(\mathbf{P}_{sc})$ we get the following:

$$\begin{aligned} n^{**} &= \left\lceil \frac{kp}{q} + \frac{R\sqrt{kp(2-p-q)}}{q} \right\rceil \\ n^* = n^{**} + k &= \left\lceil \frac{k(p+q)}{q} + \frac{R\sqrt{kp(2-p-q)}}{q} \right\rceil. \end{aligned}$$

Considering the Gilbert model parameters in terms of burstiness, in case of $p < 1-q$ the model describes bursty loss, and scattered loss for $p > 1-q$. Note that in case of $p = 1-q$ the Gilbert model is equivalent to the Bernoulli model. The value of n^* can be compared to the one obtained under Bernoulli loss. The case $p < 1-q$ results in higher and the case $p > 1-q$ results in lower values of required packets compared to the one obtained when $p = 1-q$. The difference (n_d^*), which is determined by the bias in the expected values and standard deviations can be given by

$$n_d^* = \left\lceil kp\left(\frac{1}{1-p} - \frac{1}{q}\right) + R\left(\frac{\sqrt{kp}}{1-p} - \frac{\sqrt{kp(2-p-q)}}{q}\right) \right\rceil$$

The results presented in this section give tight bounds on the number of required packets to be sent in case of two basic loss models. They also indicate that the proportion of the uncertainty expressed by the standard deviation relating to the expected value of the random process, becomes insignificant for large message sizes. Specifically, with increasing message size k, both $(R\sqrt{kp}/(1-p))/(k/(1-p))$ and $(R\sqrt{kp(2-p-q)}/q)/(k(p+q)/q)$ tend to 0.

IV. Towards a Realistic Traffic Model

In this section we extend our goodput analysis given only for an uninterrupted traffic in Section II by investigating a more realistic, ON-OFF traffic model. Consider again the scenario shown in Figure 1. If individual sources send according to an ON-OFF process with existing expected values μ_{ON} and μ_{OFF}, and if we denote the number of transmitting flows at time t by S_t, and the transmission indicator of a flow at time t by I_t, then using the notations of Figure 1 we can derive the following formula for the long-term goodput:

$$\mathbf{E}\left(\lim_{t\to\infty} G(t)\right) = \frac{c_B}{\mathbf{E}\left(\lim_{t\to\infty} S(t)\right)(1+\varepsilon)}\mathbf{E}\left(\lim_{t\to\infty} I_t\right) =$$

$$= \frac{c_B}{N\frac{\mu_{ON}}{\mu_{ON}+\mu_{OFF}}(1+\varepsilon)}\frac{\mu_{ON}}{\mu_{ON}+\mu_{OFF}} = \frac{c_B}{N(1+\varepsilon)}.$$

This result indicates that the expected behavior of the network in case of ON-OFF modulated sources is asymptotically equivalent to the one when each end host sends uninterruptedly.

It is also interesting to investigate the goodput in comparison to different TCP versions (Cubic and Compound) for both traffic types, so we carried out a simulation study for the topology of Figure 1 using 5 flows to evaluate these scenarios. Results for different TCP versions were obtained through ns-2 simulations [18], while results for the maximal rate sending method were obtained through a simple simulation tool. This tool could take into account the performance of individual maximal rate sending transfers calculated by $G = \frac{c_B}{(1+\varepsilon)N}$ according to the same ON-OFF processes, which drove the ns-2 simulations. In case of TCP congestion window values were retained in OFF periods for initializing their values in the consecutive ON periods. In each scenario all link delays were set to 10 ms, the simulation time was 2100 seconds and we used a redundancy parameter $\varepsilon = 10\%$.

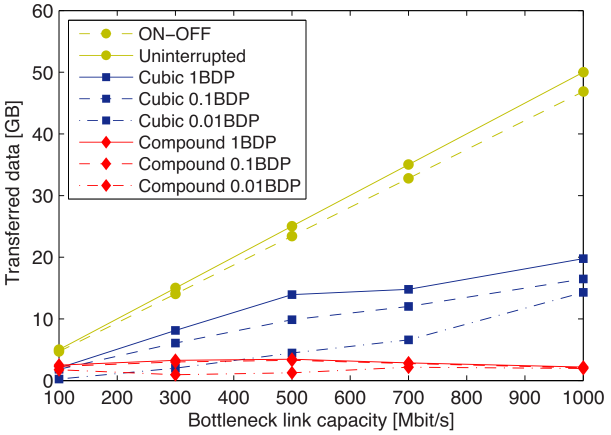


Fig. 3. Transferred data in case of ON-OFF generated transmissions

Figure 3 shows our results obtained when transmissions were generated according to exponential ON-OFF processes consisting of 100 ON and OFF intervals. The parameter of the exponential processes is 0.1, consequently, the expected value for both ON and OFF intervals is 10 seconds. The curve "ON-OFF" belongs to the case of maximal rate sending when transmissions are generated according to ON-OFF processes, and the curve "Uninterrupted" belongs to all-time sending cases when flows transfer uninterruptedly. The figure reveals that both TCP versions suffer if transmission is interrupted by idle intervals, and even in the best case the performance of TCP cannot approach the performance of our solution. This result confirms that the proposal with the idea of maximal rate sending has high potential in the cases when rate variation would appear for TCP. Figure 3 also gives visual confirmation to the asymptotic equivalency of the ON-OFF interrupted and uninterrupted transfers since ON-OFF transfer tends to the all-time transfer case. We can find out that, for the dynamic traffic presented in this section, the proposed scheme significantly surpasses the performance of TCP.

V. Digital Fountain based Communication Protocol

In this section a brief description of our new protocol called Digital Fountain based Communication Protocol (DFCP) is given including the main design principles, the operating mechanism and some implementation details. For a comprehensive discussion of DFCP, please see [19].

A. Overview

DFCP is a connection-oriented transport protocol, which can be found in the transport layer of the TCP/IP stack, and similar to TCP it ensures reliable end-to-end communication between hosts. The operation of the protocol consists of three main steps, namely connection establishment, data transfer and connection termination. However, unlike TCP our protocol does not use any congestion control algorithm, but just encodes the data using Raptor codes and 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. Although, coding needs an extra overhead, it will be shown in the following section that this approach has many advantages and can eliminate several drawbacks of TCP. DFCP has been implemented in the Linux kernel version 2.6.26-2 and it has been tested under the Debian Lenny distribution.

B. Connection Establishment and Termination

DFCP connection establishment is based on a three-way handshake procedure as in the case of TCP [20]. The handshaking mechanism is designed so that the sender can negotiate all the parameters necessary for decoding with the receiver before transmitting application data. When the data are successfully received by the destination host, the connection is released similarly to TCP.

C. Coding and Data Transfer

Once a connection is successfully established the protocol is ready to send application layer data. First, data are divided into blocks and each of them is stored in a kernel buffer until free space is available. After that DFCP performs encoding for the waiting blocks sequentially.

As shown in Figure 4, Raptor coding involves two phases: precoding and LT coding [12]. In our implementation precoding is realized by LDPC (Low-Density Parity-Check) coding [21], which adds some redundant bytes to the original message symbols. LT coder uses the result of the LDPC coding phase as input and produces a potentially infinite stream of encoded bytes.

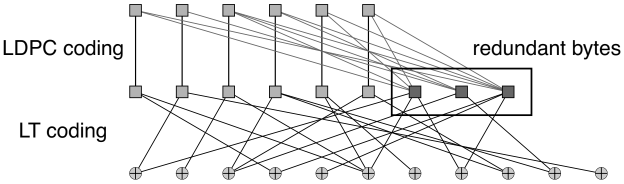


Fig. 4. Encoding phases of message blocks

When the encoding process is finished on a block, a specific protocol header is appended to the encoded block and it is immediately sent. The protocol header includes all necessary information for identifying and decoding the given message block.

D. Flow Control

To determine if the next block can be sent, a sliding window mechanism is applied at the sender. The window size gives the maximum number of unacknowledged blocks in the network. It can be set to an arbitrary value in DFCP, and the main purpose is to control the burstiness of data transfer. Once a block is received by the destination host, it returns an acknowledgement to the source host. The sliding window shifts by one unit, and the next block is sent. To ensure in-order delivery DFCP assigns a continuously increasing unique identifier to each block in the protocol header, hence the receiver can recover the original order of blocks automatically. Finally, received blocks can be decoded with high probability.

VI. Testbed Results

To investigate the behavior of our new protocol, a performance analysis was carried out in a testbed environment for different network topologies and test scenarios. The main target was to reveal how the goodput depends on different loss and delay parameters of the network. In order to focus on the effects of these parameters and to avoid the impact of the Raptor codec implementation, its delay was excluded from the results. In the following subsections our initial testbed measurement results are presented and discussed.

A. Testbed Environment

The measurement setup consisted of senders, receivers and a Dummynet network emulator, which was used for simulating various network parameters such as queue length, bandwidth, delay and packet loss probability [22]. Each test computer was equipped with the same hardware components according to Table I.

B. Experiments with Individual 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 first experiments were performed on a simple dumbbell topology with one sender and receiver by transferring 1 GB of data. In this scenario, our purpose was to investigate the steady-state goodput of the transport protocols. However, we note that a detailed performance evaluation regarding the transfer of different flow sizes was also carried out and can be found in [19]. The measurement setup is shown below in Figure 5.

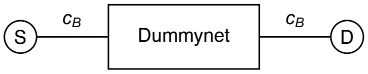


Fig. 5. Dumbbell topology with one source-destination pair

During these tests only the delay (round-trip time, RTT) and the packet loss rate 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 all cases we used the goodput (i.e. the number of useful bytes transferred per second) as the performance metric. To emphasize the beneficial properties of DFCP we compared it to different TCP versions, namely TCP Cubic which is the default congestion control algorithm in the Linux kernel and TCP NewReno with SACK option.

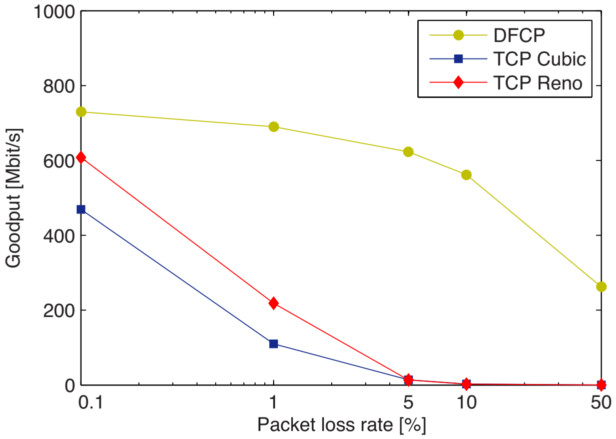


Fig. 6. Goodput for increasing packet loss rate using optimal redundancy parameters in DFCP

In Figure 6 the measured goodput is depicted for increasing packet loss rate. In case of DFCP an optimal redundancy parameter ε_{opt} was determined and set for each loss rate 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. These optimal values were found manually, but it is a possible option to perform this task by an adaptive algorithm of the protocol, so it will be part of our future research. The figure clearly indicates that DFCP significantly outperforms both TCP versions in terms of goodput in a lossy environment, and for increasing loss rate the difference becomes higher. 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. This scenario also demonstrates that in contrast to TCP our protocol can function even in the case of extremely high packet loss.

A practical result can be seen in Figure 7 where the goodput is examined only in the interval 0.1–1%. The figure illustrates that if the coding overhead is adjusted to a certain rate of packet loss, DFCP will become insensitive to packet loss when its rate is less than or equal to this value. Therefore, the protocol can guarantee a predictable performance within a given interval of loss rate based on the redundancy parameter, and as a result, the rate variation experienced in case of TCP can be avoided. Moreover, the goodput performance of DFCP is significantly better compared to both TCP versions in the whole range.

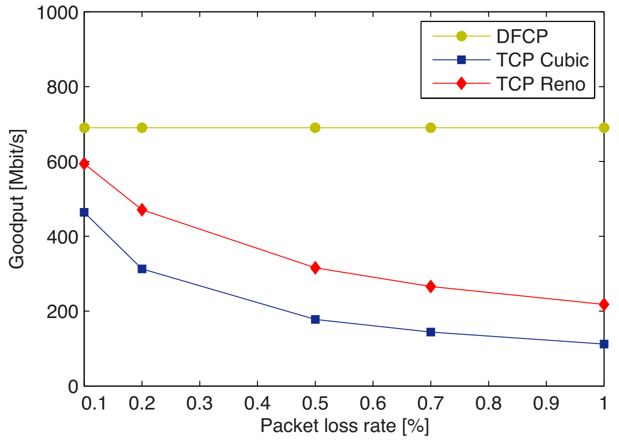


Fig. 7. Goodput for increasing packet loss rate using a fixed redundancy parameter in DFCP adjusted to packet loss of 1%

Another advantageous property of DFCP can be observed in Figure 8. Namely, our protocol can achieve good performance in a network with high delay as well since its goodput drops more slowly than in case of TCP for increasing round-trip time. It is a very important feature because of the fact that in real networks the most significant fraction of RTT values exceeds 10 ms [23]. We note that the goodput degradation of DFCP for large RTTs is due to the flow control limitation determined by the window size.

The previous figures showed the efficiency of DFCP in environments with high loss rate and delay, respectively. Going one step beyond, it is also interesting to investigate how the protocol behaves in a network where both packet loss rate and round-trip time are greater than zero. Figure 9 depicts the change of goodput for increasing round-trip time and different packet loss rates using optimal redundancy in each point. Although, TCP Cubic and Reno already suffer if packet loss rate or end-to-end delay has a high value, these conditions together have a greater impact on their performance making them almost impossible to operate. As an example, for packet loss rate of 0.1% and round-trip time of 5 ms, the performance of TCP Cubic and Reno is reduced to 8% compared to the ideal case when these parameters are equal to zero. At the same time, DFCP can work at 94% of the ideal goodput in such conditions indicating its robustness to various network parameters. Subsequently, our protocol can achieve high goodput values in many environments irrespective of which network parameter is changed.

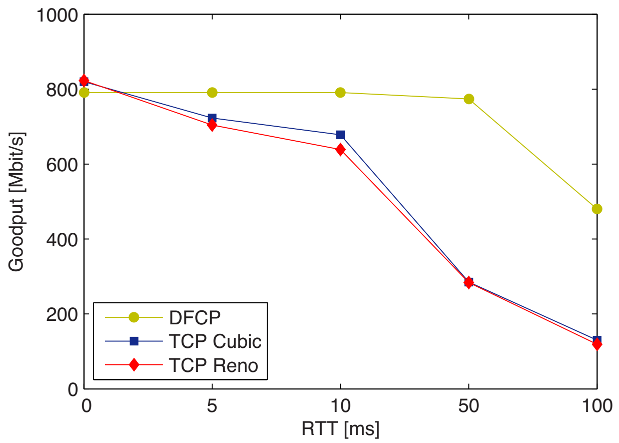


Fig. 8. Goodput for increasing round-trip time

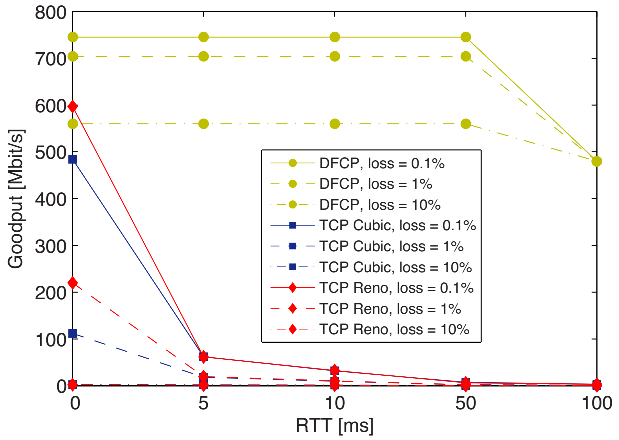


Fig. 9. Goodput for increasing round-trip time and different packet loss rates

TABLE II
TRANSFER DURATIONS OF A TYPICAL WEB OBJECT

Protocol	**Transfer duration [ms]**		
	RTT = 10 ms	*RTT = 50 ms*	*RTT = 100 ms*
DFCP	10	10	10
TCP Cubic	59	300	600
TCP Reno	61	301	600

Due to practical reasons it is important to reveal the transient behavior of a transport protocol. Although, the transient phase is relatively short in time, it has a high impact on small data transfers. 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. [24], [25]). They introduced various techniques to speed up its operation for short-lived flows as well, which is essential since recent studies showed that most flows are small carrying only several kilobytes of data and short lasting less than a few seconds [26]. Web traffic serves as a great example for this flow type because of the fact that the mean web object size is about 100–200 kB, consequently, the download time can be very short [27]. Table II gives the transfer durations of a typical web object (150 kB) for different round-trip times using DFCP, TCP Cubic and Reno. The table highlights that in this practically interesting case it matters how the transport protocol behaves in the transient phase. Namely, DFCP can achieve a 60 times shorter download time compared to TCP for round-trip time of 100 ms, however, the steady-state goodput ratio between DFCP and TCP is only about 4 in this case according to Figure 8. It means that for a usual value of round-trip time, DFCP can provide nearly 15 times faster download in the transient phase than on average showing its potential in case of web traffic. Additionally, for such a small flow size, the download time is independent of the round-trip delay.

C. Experiments with Competing Flows

The second measurement setup can be seen in Figure 10 where all parameters were set similarly as described at the first dumbbell topology complemented by the condition $c_1 = c_2 = 1$ Gbps. The next scenarios present the results for two competing flows of the same type. The measurement duration was 60 seconds for each test, and the first 15 seconds were excluded from the results neglecting the behavior of the examined protocols in the transient phase. The flows were started together and we used WFQ (Weighted Fair Queueing) as the scheduling method with equal weights (i.e. 50-50%) [28].

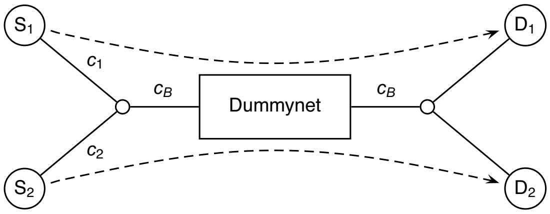


Fig. 10. Dumbbell topology with two source-destination pairs

It is widely known that, when several TCP flows with different round-trip times share the same bottleneck link, they will not gain access to an equal portion of the available bandwidth [29], [30]. This property is also called as RTT unfairness and is caused by the AIMD (Additive Increase Multiplicative Decrease) algorithm of TCP in the congestion avoidance phase. TCP increases the congestion window by one for each round-trip time and decreases it by half when a drop is detected [31]. Therefore, flows with smaller round-trip times increase their congestion windows more rapidly, and hence they are able to reach higher sending rates. Because of the issue mentioned above, it is essential to reveal and analyze the fairness properties of a new transport protocol.

Figure 11 shows the goodput 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 note that the results for TCP Reno were quite the same as in case of TCP Cubic, thus for perspicuity, only the latter was depicted. We can see in the figure that the bottleneck link capacity is equally shared by the two TCP flows for RTT values less than 20 ms. However, for RTTs greater than 20 ms the goodput of the second flow starts to decrease, and as a result, the first flow having lower RTT can attain a greater portion of the available bandwidth, indicating the unfairness behavior of TCP. In contrast, DFCP flows share the bottleneck capacity in a fair way since our protocol is much less sensitive to high delays 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.

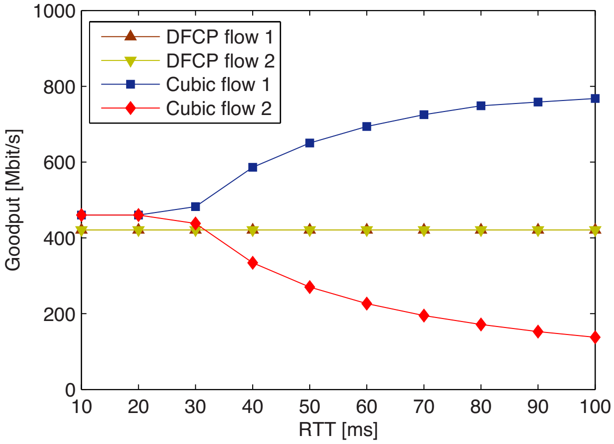


Fig. 11. Two competing flows with the one having a fixed RTT of 10 ms and the other one having an RTT varied between 10 and 100 ms

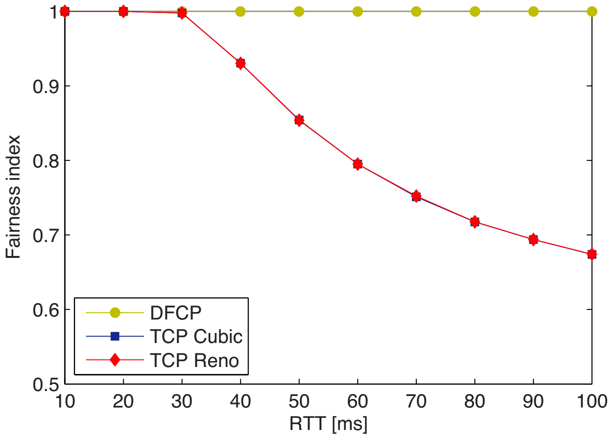


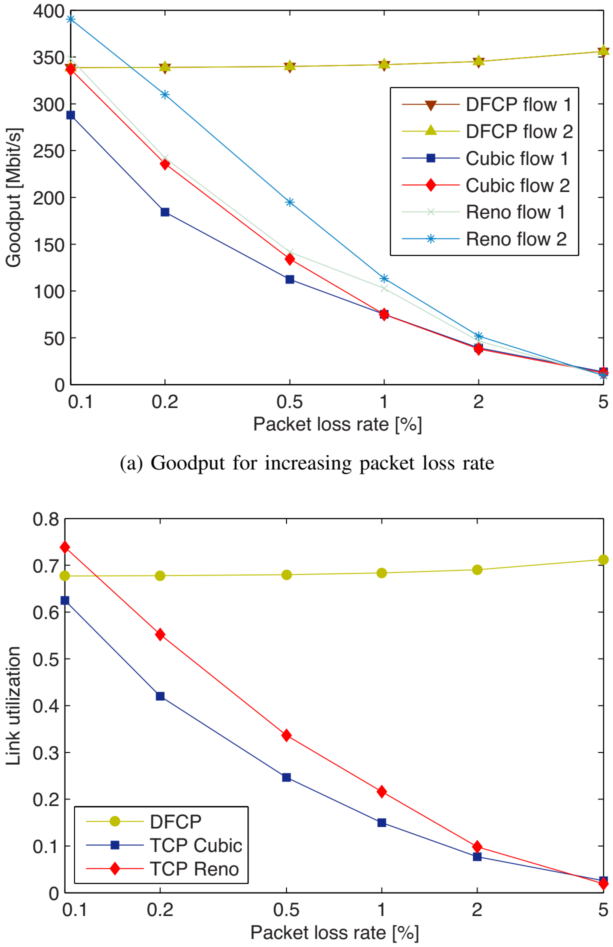
Fig. 12. Intra-protocol fairness of DFCP and TCP variants

In Figure 12 the intra-protocol fairness behavior of DFCP and TCP variants is shown for increasing round-trip time. 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 [31]. 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. The figure confirms the observations taken before (see Figure 11), namely, while different TCP versions become more and more unfair for increasing round-trip time, DFCP provides perfect fairness and it prefers none of the two flows independently of their RTT values.

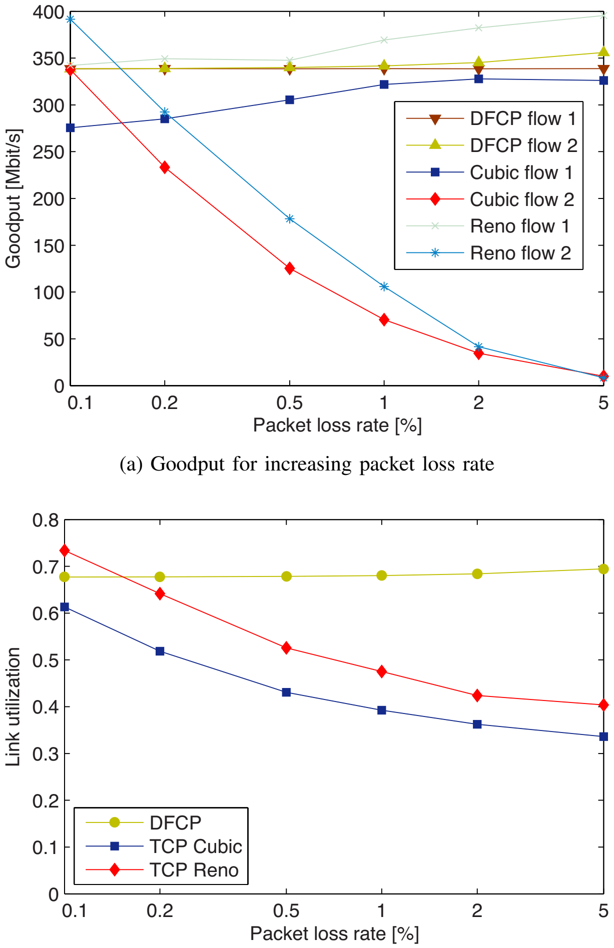


(a) Goodput for increasing packet loss rate

(b) Link utilization for increasing packet loss rate

Fig. 13. Goodput and link utilization for two competing flows with equal loss rate

Figures 13 and 14 illustrate the impact of packet loss rate on the goodput and link utilization for two competing flows. Figure 13 shows the case when packet loss rates are equal for both flows and changed according to the horizontal axis, and Figure 14 shows the case when they experienced different loss rates. In the latter case, the first flow has a fixed loss rate set to 0.1%, and the second one having loss rate varied between 0.1 and 5% as shown in Figure 14. We note that in these scenarios the redundancy parameter of DFCP was adjusted to packet loss of 5%. In Figure 13a it can be observed that both TCP Cubic and Reno flows do not share the available bandwidth equally for lower values of loss rate, but the difference reduces for increasing packet loss rate. Unlike different TCP variants DFCP results in a fair allocation. On the one hand, each DFCP flow achieves nearly the same goodput value, and it is almost independent of the packet loss rate. Furthermore, looking at Figure 13b one can see that both TCP versions can reach only a poor link utilization in contrast to DFCP. Figure 14 shows that, while DFCP behaves similarly in the cases of equal and different loss rates for the two flows, respectively, TCP Cubic and Reno share the bottleneck link capacity in an unfair way. We can conclude that, if optimal redundancy is properly chosen, it is irrelevant to DFCP that loss rates are equal or different for the competing flows, and what values they have.



(a) Goodput for increasing packet loss rate

(b) Link utilization for increasing packet loss rate

Fig. 14. Goodput and link utilization for two competing flows with different loss rates

VII. Conclusion

In this paper we investigated a recently proposed paradigm for Future Internet architecture where an efficient erasure coding scheme is applied instead of congestion control. In order to carry out a detailed and also practical analysis a novel protocol (DFCP) has been developed and implemented. We derived tight bounds on packets required to be sent using our rateless coding scheme with large message sizes in case of two loss models and predefined QoS requirements. Analytical investigations of related scenarios with realistic ON-OFF sources were also provided with simulation comparisons to TCP versions. Furthermore, we presented our first testbed performance measurement results by comparing DFCP to current TCP versions. We showed that the goodput performance of our protocol is significantly better than in case of different TCP versions in a wide range of packet loss rates and round-trip delays. Moreover, it turned out that DFCP can work even in environments with high loss rate where TCP is unable to operate. Another important observation is that, with appropriate settings, the performance of DFCP is independent of the loss and delay parameters of the network. The results demonstrate the great potential of the new paradigm and our future research will address the upcoming challenges such as scalability and the detailed performance evaluation of DFCP on more complex network topologies.

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 OTKA-KTIA grant no. CNK77802 and the High Speed Networks Laboratory (HSNLab) at Budapest University of Technology and Economics.

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] 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] J. K. Sundararajan, D. Shah, M. Médard, S. Jakubczak, M. Mitzenmacher, J. Barros, “Network Coding Meets TCP: Theory and Implementation”, *Proceedings of the IEEE*, vol. 99, no. 3, pp. 490–512, 2011.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] D. Kumar, T. Chahed, E. Altman, “Analysis of a Fountain Codes Based Transport in an 802.11 WLAN Cell”, *Proceedings of the 21st International Teletraffic Congress*, pp. 1–8, Paris, France, 2009.

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

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

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

[14] 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.

[15] A. Kortebi, L. Muscariello, S. Oueslati, J. Roberts, “Evaluating the Number of Active Flows in a Scheduler Realizing Fair Statistical Bandwidth Sharing”, *Proceedings of the ACM SIGMETRICS 2005 International Conference on Measurement and Modeling of Computer Systems*, pp. 217–228, Banff, AB, Canada, 2005.

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

[17] G. Haßlinger, O. Hohlfeld, “The Gilbert-Elliott Model for Packet Loss in Real Time Services on the Internet”, *Proceedings of the 14th GI/ITG Conference on Measurement, Modeling and Evaluation of Computer and Communication Systems*, pp. 269–286, Dortmund, Germany, 2008.

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

[19] 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

[20] J. Postel, “Transmission Control Protocol”, *RFC 793, IETF*, 1981.

[21] A. Shokrollahi, “LDPC Codes: An Introduction”, *Technical Report, Digital Fountain Inc.*, 2003.

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

[23] 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.

[24] N. Hu, P. Steenkiste, “Improving TCP Startup Performance using Active Measurements: Algorithm and Evaluation”, *Proceedings of the 11th IEEE International Conference on Network Protocols*, pp. 107–118, Atlanta, GA, USA, 2003.

[25] 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.

[26] 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.

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

[28] H. Zhang, “Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks”, *Proceedings of the IEEE*, vol. 83, no. 10, pp. 1374–1396, 1995.

[29] 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.

[30] T. H. Henderson, E. Sahouria, S. McCanne, R. H. Katz, “On Improving the Fairness of TCP Congestion Avoidance”, *Proceedings of the IEEE GLOBECOM 1998 International Conference*, vol. 1, pp. 539–544, Sydney, Australia, 1998.

[31] 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.