Delaying Transmissions in Data Communication Networks to Improve Transport-Layer Performance

[1]Pangambam Uttam Kumar, [2]Jyoti Gupta

[1, 2] Department of Electronics and Communication Engineering, M.M. University, Mullana, Ambala, India

Abstract

The performance improvement of the network with small buffers needs to be focused. The main factor that reduces the performance of the network is the packet loss in the network layer in the event of data transmission. Packet loss occurred due to bit errors in physical layer and congestion in network layer degrades the performance in the transport layer, in terms of delay and bandwidth. Existing approaches improving the performance of the transport layer requires either much delay or bandwidth. So, Queue Length Based Pacing (QLBP) is presented, which is a novel pacing algorithm that decreases the burstiness of network traffic by delaying packets based on the length of the local packet buffer. Stability control mechanism is applied as contribution which further reduces the packet loss. Simulation is performed over the network of dumbbell topology with QLBP technique using Network simulator 2 (NS 2). Performance metric such as throughput, delay, packets drops over buffer size is analyzed.

Keywords: Dumbbell Topology, Packet Drop, Propagation Delay, Small Buffer Network, TCP, Throughput, Traffic Pacing and Transport Layer.

"1.Introduction"

TCP mechanisms

TCP is a sliding window-based protocol. The effective window used by a TCP sender is the minimum of the congestion window (cwnd – set to match the bandwidth of the network) and the receiver's buffer size. Since the window is the number of bytes the sender is allowed to send without an acknowledgment, the average rate at which traffic enters the network is governed by the window size divided by the round trip time (RTT). The sender uses the incoming acknowledgments to determine when to send new data, a mechanism referred to as ACK-clocking. The congestion window adjustment algorithm has two phases. In the slow start phase, the sender increases the congestion window rapidly in order to quickly identify the bottleneck rate while at the same time theoretically establishing a stream of well-spaced acknowledgments. The sender typically starts with a window of one packet; each acknowledgment increases the window by 1, effectively doubling the window every round trip time. Assuming the sender is not limited by the receiver's buffer space, the sender increases its congestion window until it detects that a packet loss has occurred; the loss is taken as a signal that the sender is transmitting packets faster than the network can handle. At this point the window is cut in half, and the sender enters the congestion avoidance phase. The sender then increases the window by 1/cwnd on every acknowledgment, effectively increasing it by 1 packet every round trip time. Again, the sender increases the window until it detects a packet loss; on a loss, the window is cut by half, and the sender resumes increasing the window by 1 packet per round trip. As a result, the sender's congestion window (controlling the rate at which packets are sent) at first increases along an exponential curve during slow start, then over time repeats a saw-tooth pattern of a factor of two decrease and a subsequent slow, linear increase. Since the receipt of acknowledgments governs the rate of increase in the congestion window, connections with longer round trip times have windows that grow at a slower rate (both slow start and the saw-tooth have a proportionately lower slope).

Each TCP acknowledgment reports the highest-numbered packet (more precisely, byte) that had been received along with all earlier packets. Assuming there are no losses, the information in each acknowledgment therefore makes all earlier acknowledgments redundant. As an optimization, TCP receivers try to reduce the number of acknowledgments by delaying returning an acknowledgment for a single received packet (assuming all prior packets had been correctly received) until either another packet arrives or a timer fires. Because half as many acknowledgments are received, and since acknowledgments trigger increases in the congestion window, the principal effect of delayed acknowledgments is to slow the rate at the congestion window is increased: during slow start from a factor of 2 to a factor of 1.5 per round trip and during congestion avoidance from 1 packet to 0.5 packets per round trip [21]. During slow start, every successfully acknowledged packet increases the window size by one packet. Thus, the sender transmits two packets for every new acknowledgment. Queuing theorists are used to thinking of sizing buffers so as to prevent them from overflowing and losing packets. But TCP's "saw

tooth" congestion control algorithm is designed to fill any buffer, and deliberately causes occasional loss to provide feedback to the sender. No matter how big we make the buffers at a bottleneck link, TCP will cause the buffer to overflow.

Goal of pacing

The goal of pacing is to evenly spread the transmission of a window of packets across the entire duration of the round trip time [21]. This can be implemented either by the sender or the receiver. At the sender, instead of transmitting packets immediately upon receipt of an acknowledgment, the sender can delay transmitting packets to spread them out at the rate defined by the congestion control algorithm – the window size divided by the estimated round-trip time. Alternatively, a receiver can delay acknowledgments to spread them across the round trip time, so that when they arrive at the sender, they will trigger spaced data packets. Of course, receiver pacing is less effective, since as we discussed earlier, acknowledgments arriving at the sender can trigger multiple data sends; with receiver pacing, these packets will be sent in a burst. Further, receiver pacing is susceptible to ACK compression. Therefore, only sender-based pacing is simulated. As a traditional window based protocol, TCP uses a window to determine the number of packets that can be sent and uses the receipt of acknowledgments to trigger the sending of packets [10]. Pure rate based schemes, on the other hand, use rates to determine both how much and when to send. Pacing is a hybrid between these two approaches – it uses the TCP window to determine how much to send but uses rates instead of acknowledgments to determine when to send.

Uses of pacing

One way to understand the impact of pacing is to consider the router from a queuing theory perspective. With bursty traffic, packets arrive all at once. As a result, queuing delay grows linearly with load, even when the load is below capacity. TCP, due to its burstiness, can be thought of as being close to this worst case curve. With pacing, traffic is evenly spaced out; so there is minimal queuing until the load matches the bottleneck capacity [29]. The queuing delay increases linearly once the bottleneck is saturated. This represents the best possible time curve for a queuing system and is theoretically even better than the curve for a random traffic source. TCP uses feedback from the network to detect congestion and adjust to it. With pacing, this feedback is delayed until the network is saturated, making it difficult for senders to "avoid" overwhelming the network.

Queue length based pacing algorithm

The general ideal of Queue Length Based Pacing (QLBP) is to dynamically adjust the sending rate of a queue according to the queue length, rather than to send packets at a constant rate [8]. Traffic pacing is based on TCP, but uses traffic conditioning techniques in the network to reduce traffic bursts. By delaying some packet transmissions, less packet losses occur and thus less retransmission are needed. Traffic pacing incurs a small additional delay, but uses less bandwidth than TCP since fewer retransmissions are necessary. Pacing is deployed on several (but not necessarily all) nodes in the network. This pacing process can be implemented on the outgoing interfaces of routers. These routers have sufficiently large buffers that allow moderate traffic bursts to be absorbed and paced without packet loss. At the network edge, routers with pacing capabilities reduce the burstiness of traffic before it enters the small-buffer network core. The inter-packet pacing delay is calculated based on the packet arrival curve and the packet deadline curve within the same pacing interval. QLBP, determine this delay based on some very simple rules: (a) if the pacing queue increases due to a higher input traffic rate, QLBP intentionally lowers the introduced pacing delay. This rule ensures that the link can be fully utilized under heavy load. (b) For packets that arrive at a rate lower than μmin, they do not get delayed. This rule ensures that pacing is only activated when packets arrive at a certain high rate.

Problems

One of the most problematic events for data transmissions in the network layer is a packet loss. The two main causes for packet loss in networks are:

• Bit errors in physical layer: Bit errors in the physical layer most commonly occur in wireless transmissions due to interference, but can also occur in wired links. These bit errors causes checksums in the data link layer to fail, triggering a packet drop.

• Congestion in network layer: Statistical multiplexing of network traffic implies that there are no guarantees about the available bandwidth on any given link. Thus, network traffic can congest the outgoing port of a router and cause transmission buffers to fill up. If a packet arrives at such a transmission queue when no more buffer space is available, then it is dropped. While these causes of packet loss are fundamentally different, their effects result in the same performance degradation in the transport layer.

"2. Literature Survey"

A. Vishwanath et al. [1] proposed that internet traffic is expected to grow phenomenally over the next five to ten years, and to cope with such large traffic volumes, core networks are expected to scale to capacities of terabits-per-second and beyond. Increasing the role of optics for switching and transmission inside the core network seems to be the most promising way forward to accomplish this capacity scaling. Unfortunately, unlike electronic memory, it remains a formidable challenge to build even a few packets of integrated all-optical buffers. In the context of envisioning a bufferless (or near-zero buffer) core network, their contributions are three fold: First, a novel edge-to-edge based packet-level forward error correction (FEC) framework as a means of combating high core losses was proposed and investigated via analysis and simulation the appropriate FEC strength for a single core link. Secondly, considering a realistic multi-hop network and develop an optimization framework that adjusts the FEC strength on a per-flow basis to ensure fairness between single and multi-hop flows. Third, studying the efficacy of FEC for various system parameters such as relative mixes of short-lived and long-lived TCP flows, and average offered link loads. Their study is the first to show that packet-level FEC, when tuned properly, can be very effective in mitigating high core losses, thus opening the doors to a buffer less core network in the future.

M. Shifrin et al. [2] proposed that because of TCP dynamics, Internet backbone routers hold large packet buffers, which significantly increase their power consumption and design time. Recent models of large-buffer networks have suggested that these large buffers could be replaced with much smaller ones. Unfortunately, it turns out that these large-buffer network models are not valid anymore in small-buffer networks, and therefore cannot predict how these small-buffer networks will behave. In this paper, a new model that provides a complete statistical description of small-buffer Internet networks was introduced. Novel models of the distributions of several network components, such as the line occupancies of each flow, the instantaneous arrival rates to the bottleneck queues, and the bottleneck queue sizes was presented. Later, all these models are combined in a single fixed-point algorithm that forms the key to a global statistical small-buffer network model. In particular, given some QoS requirements, also showed how this new model can be used to precisely size small buffers in backbone router designs.

M. Aron et al. [3] proposed that for the optical packet-switching routers to be widely deployed in the Internet, the size of packet buffers on routers has to be significantly small. Such small-buffer networks rely on traffic with low levels of burstiness to avoid buffer overflows and packet losses. The proposed work presented a pacing system that proactively shapes traffic in the edge network to reduce burstiness. The queue length based pacing uses an adaptive pacing on a single queue and paces traffic indiscriminately where deployed. In this work, they showed through analysis and simulation that this pacing approach introduces a bounded delay and that it effectively reduces traffic burstiness. The work also showed that it can achieve higher throughput than end-system based pacing.

A. Lakshmikantha et al. [4] suggested that traditionally, it had been assumed that the efficiency requirements of TCP dictate that the buffer size at the router must be of the order of the bandwidth-delay ($C \times$ RTT) product. Recently, this assumption was questioned in a number of papers, and the rule was shown to be conservative for certain traffic models. In particular, by appealing to statistical multiplexing, it was shown that on a router with N long-lived connections, buffers of size $O(C \times \text{RTT}/\sqrt{N})$ or even O (1) are sufficient. In this paper, the buffer-size requirements of core routers when flows arrive and depart are examined. The conclusion is as follows: If the core-to-access-speed ratio is large, then O (1) buffers are sufficient at the core routers; otherwise, larger buffer sizes do improve the flow-level performance of the users. From a modeling point of view, our analysis offers two new insights. First, it may not be appropriate to derive buffer-sizing rules by studying a network with a fixed number of users. In fact, depending upon the core-to-access-speed ratio, the buffer size itself may affect the number of flows in the system, so these two parameters (buffer size and number of flows in the system) should not be treated as independent quantities. Second, in the regime where the core-to-access-speed ratio is large, we note that the O (1) buffer sizes are sufficient for good performance and that no loss of utilization results, as previously believed.

Y. Huang et al. [6] proposed that Queuing analysis is important in providing guiding principles for packet network analysis. Stochastic fluid queuing models have been widely used as burst scale models for high speed communication networks. In this paper, the authors proposed a novel two-level Markov on-off source model to model the burstiness of a packet stream at different time scales. Analytical results are obtained to reveal the impact of traffic burstiness at two levels on the queue lengths in a tandem queue system. The method combines the modeling power of the Poisson processes with that of stochastic differential equations to handle the complex interactions between the packet arrivals and the queue content. Results for the tandem queuing network could be used to further justify the packet spacing scheme in helping deploying small buffer routers.

Y. Gu et al.[7] proposed that there is growing interest in designing high speed routers with small buffers that store only tens of packets. Recent studies suggest that TCP New Reno, with the addition of a pacing mechanism, can interact with such routers without sacrificing link utilization. The work showed in this paper, as workload requirements grow and connection bandwidths increase, the interaction between the congestion control protocol and small buffer routers produce link utilizations that tend to zero. This is a simple consequence of the inverse square root dependence of TCP throughput on loss probability. In this paper they presented a new congestion controller that avoids this problem by allowing a TCP connection to achieve arbitrarily large bandwidths without demanding the loss probability go to zero. Authors showed that this controller produces stable behaviour and, through simulation, they showed its performance to be superior to TCP New Reno in a variety of environments. Lastly, because of its advantages in high bandwidth environments, they compared their controller's performance to some of the recently proposed high performance versions of TCP including HSTCP, STCP, and FAST. Simulations illustrate the superior performance of the proposed controller in a small buffer environment.

M. C. Weigle et al. [8] proposed that when doing simulations, authors were confronted with the problem of choosing a good workload model. Often, realistic workload models are difficult to come up with. So, this paper proposed a tool, called Tmix, that allows to automatically extract communication workloads from packet traces and then to replay those workloads in the ns simulator or in test beds.

V. Sivaraman et al.[9] proposed that in the absence of a cost-effective technology for storing optical signals, emerging optical packet switched (OPS) networks are expected to have severely limited buffering capability. To mitigate the performance degradation resulting from small buffers, this paper proposes that optical edge nodes "pace" the injection of traffic into the OPS core. The contributions relating to pacing in OPS networks are three-fold: first, real-time pacing algorithms of poly-logarithmic complexity that are feasible for practical implementation in emerging high-speed OPS networks was developed. Second, an analytical quantification of the benefits of pacing in reducing traffic burstiness and traffic loss at a link with very small buffers was proposed. Third, authors showed via simulations of realistic network topologies that pacing can significantly reduce network losses at the expense of a small and bounded increase in end-to-end delay for real-time traffic flows. It was argued that the loss-delay trade off mechanism provided by pacing can be instrumental in overcoming the performance hurdle arising from the scarcity of buffers in OPS networks.

M. Enachescu et al. [10] proposed that internet routers require buffers to hold packets during times of congestion. The buffers need to be fast, and so ideally they should be small enough to use fast memory technologies such as SRAM or all-optical buffering. Unfortunately, a widely used rule-of-thumb says we need a bandwidth-delay product of buffering at each router so as not to lose link utilization. In this paper, they explored how buffers in the backbone can be significantly reduced even more, to as little as a few dozen packets, if they are willing to sacrifice a small amount of link capacity. Authors argued that if the TCP sources are not overly bursty, then fewer than twenty packet buffers are sufficient for high throughput. Specifically, authors argued that O (logW) buffers are sufficient, where W is the window size of each flow. The proposed worked supported the claim with analysis and a variety of simulations. The change they need to make to TCP is minimal-each sender just needs to pace packet injections from its window. Moreover, there is some evidence that such small buffers are sufficient even if we don't modify the TCP sources so long as the access network is much slower than the backbone, which is true today and likely to remain true in the future. The proposed work concluded that buffers can be made small enough for all-optical routers with small integrated optical buffers.

J. Naor et al. [11] considered the problem of scheduling a sequence of packets over a linear network, where every packet has a source and a target, as well as a release time and a deadline by which it must arrive at its target. The model considered is bufferless, where packets are not allowed to be buffered in nodes along their paths other than at their source. This model applies to optical networks where opto-electronic conversion is costly, and packets mostly travel through bufferless hops. The offline version of this problem was previously studied in M. Adler et al. (2002). Authors studied the online version of the problem, where they are required to schedule the packets without knowledge of future packet arrivals. Competitive analysis to evaluate the performance of our algorithms was used. The first deterministic online algorithms for several versions of the problem were presented. For the problem of throughput maximization, where all packets have uniform weights, they gave an algorithm with a logarithmic competitive ratio, and present some lower bounds. For other weight functions, algorithms that achieve optimal competitive ratios were showed.

G. Raina et al. [12] described how control theory has been used to address the question of how to size the buffers in core Internet routers. Control theory aims to predict whether the network is stable, i.e. whether TCP flows are desynchronized. If flows are desynchronized then small buffers are sufficient. The theory here shows that small buffers actually promoted resynchronization—a virtuous circle.

J. DeHart et al. [13] proposed that the Open Network Laboratory (ONL) is a remotely accessible network test bed designed to enable networking faculty, students and researchers to conduct experiments using high performance routers and applications. The system is built around a set of extensible, high-performance routers and has a graphical interface that enables users to easily configure and run experiments remotely. ONL's Remote Laboratory Interface (RLI) allows users to easily configure a network topology, configure routes and packet filters in the routers, assign flows or flow aggregates to separate queues with configurable QoS and attach hardware monitoring points to real-time charts. The remote visualization features of the RLI make it easy to directly view the effects of traffic as it moves through a router, allowing the user to gain better insight into system behaviour and create compelling demonstrations. Each port of the router is equipped with an embedded processor that provides a simple environment for software plugins allowing users to extend the system's functionality. This paper described the general facilities and some networking experiments that can be carried out.
M. Adler et al. [15] proposed that the time-constrained packet routing problem is to schedule a set of packets to be transmitted through a multi-node network, where every packet has a source and a destination (as in traditional packet routing problems) as well as a release time and deadline. The objective is to schedule the maximum number of packets subject to deadline constraints. In this paper authors extended the results in two directions. First, the more general network topologies of trees and 2-dimensional meshes were considered. Secondly, authors associated with each packet a measure of utility, called a weight, and study the problem of maximizing the total weight of the packets that are scheduled subject to their timing constraints. For the bufferless case, constant factor approximation algorithms for the time-constrained scheduling problem with weighted packets on trees and mashes were provided. Provisions of logarithmic approximations for the same problems in the buffered case were made. These results are complemented by new lower bounds, which demonstrate that they cannot hope to achieve the same results for general network topologies.

C.V.Hollot et al. [16] proposed that routers handle data packets from sources unresponsive to TCP's congestion avoidance feedback. Interest was in the impact these sources have on active queue management (AQM) control of long-lived TCP traffic. In these paper models of TCP/AQM dynamics was combined with models of unresponsive traffic to analyze the effects on AQM performance.

Y. Wu et al. [17] studied a queue where the rapidly varying component of input traffic is treated as a noise. The queue length statistics are compared between the cases with and without noise smoothing. This relates to abstract simulation and traffic modeling. The system load and traffic burstiness at large timescales have critical impacts on the evaluation errors caused by such smoothing.

M. Adler et al. [18] studied the problem of centrally scheduling multiple messages in a linear network, when each message has both a release time and a deadline. The proposed work showed that the problem of transmitting optimally many messages is NP-hard, both when messages may be buffered in transit and when they may not be; for either case, efficient algorithms that produce approximately optimal schedules were presented. In particular, the bufferless scheduling algorithm achieves throughput that is within a factor of two of optimal. It was showed that buffering can improve throughput in general by a logarithmic factor (but no more), but that in several significant special cases, such as when all messages can be released immediately, buffering can help by only a small constant factor. Finally, the proposed work showed how to convert their centralized, offline bufferless schedules to equally productive fully distributed online buffered ones. Most of the results extend readily to ring-structured networks.
R. Ahlswede et al. [20] introduced a new class of problems called network information flow which is inspired by computer network applications. Consider a point-to-point communication network on which a number of information sources are to be multicast to certain sets of destinations. Authors assumed that the information sources are mutually independent. The problem is to characterize the admissible coding rate region. This model subsumes all previously studied models along the same line. In this paper, authors studied the problem with one information source, and obtained a simple characterization of the admissible coding rate region. The result can be regarded as the Max-flow Min-cut Theorem for network information flow. Contrary to one's intuition, their work reveals that it is in general not optimal to regard the information to be multicast as a "fluid" which can simply be routed or replicated. Rather, by employing coding at the nodes, which was referred to as network coding, bandwidth can in general be saved. This finding may have significant impact on future design of switching systems.
R. W. Brockett et al. [22] suggested that recent study for congestion control in high speed networks indicates that the derivative information for the congestion at the common buffer for multiple sources could be useful in achieving efficient and fair allocation of the bandwidth. In this paper they presented an algorithm that estimates such derivatives for multiple on-off sources. The algorithm has its root in the infinitesimal perturbation analysis (IPA) for the classical queuing systems. Although the traditional IPA algorithm does not give unbiased derivative estimates for multi-class arrivals, the proposed work was able

to prove the unbiasedness in the case of multi-class ON-OFF sources. The results in this paper may motivate a new look at the end-to-end congestion control issue.

J. D. Salehi et al. [23] proposed that VBR compressed video is known to exhibit significant, multiple-time-scale bit rate variability. In this paper, the transmission of stored video from a server to a client across a high speed network were considered, and explore how the client buffer space can be used most effectively toward reducing the variability of the transmitted bit rate. Two basic results were presented. First, an optimal smoothing algorithm for achieving the greatest possible reduction in rate variability when transmitting stored video to a client with given buffer size was presented. A formal proof of optimality was presented, and demonstrated the performance of the algorithm on a set of long MPEG-1 encoded video traces. Second, the impact of optimal smoothing on the network resources needed for video transport, under two network service models: Deterministic Guaranteed service and Renegotiated CBR (RCBR) service was evaluated. Under both models, the impact of optimal smoothing was found to be dramatic.

M. Aron et al. [24] studied the start up dynamics of TCP on both high as well as low bandwidth delay network paths and proposed a set of enhancements that improve both the latency as well as throughput of relatively short TCP transfers. Authors also suggested that numerous studies have shown that the timer and congestion control mechanisms in TCP can have a limiting effect on performance in the start up phase. Based on the results of the study, mechanisms for adapting TCP in order to yield increased performance were proposed. First, a framework for the management of timing in TCP was proposed. Second, the work showed how TCP can utilize the proposed timer framework to reduce the overly conservative delay associated with a retransmission timeout. Third, the use of packet pacing in the initial slow-start to improve the performance of relatively short transfers that characterize the web traffic was proposed. Finally, the importance of estimating the initial slow-start threshold in TCP, mainly on high bandwidth-delay paths was quantified.

V. Visweswaraiah et al. [25] proposed that TCP congestion avoidance mechanisms are based on adjustments to the congestion-window size, triggered by the ACK clock. These mechanisms are not well matched to large but intermittent bursts of traffic, such as responses from a HTTP/1.1-based web server. Idle periods between bursts (web page replies) stop the ACK clock and hence disrupt even data flow. When restarting data flow after an idle period, current implementations either enforce slow start (SSR) or use the prior congestion window (NSSR). The former approach, while conservative, leads to low effective throughput in cases like P-HTTP. The latter case optimistically sends a large burst of back-to-back packets, risking router buffer overflow and subsequent packet loss. This paper proposed a third alternative: pacing some packets at a certain rate until the ACK clock can be restarted. The motivation and implementation of this third alternative was described and present simulation results which show that it achieves the elapsed-time performance comparable to NSSR and loss behaviour of SSR.

W. Willinger et al. [26] proposed that a number of empirical studies of traffic measurements from a variety of working packet networks have demonstrated that actual network traffic is self-similar or long-range dependent in nature-in sharp contrast to commonly made traffic modeling assumptions. A plausible physical explanation for the occurrence of self-similarity in local-area network (LAN) traffic was provided. The explanation is based on convergence results for processes that exhibit high variability and is supported by detailed statistical analyzes of real-time traffic measurements from Ethernet LANs at the level of individual sources. Here the mathematical results concerning the superposition of strictly alternating ON/OFF sources was developed. The key mathematical result states that the superposition of many ON/OFF sources (also known as packet-trains) with strictly alternating ON- and OFF-periods and whose ON-periods or OFF-periods exhibit the Noah effect produces aggregate network traffic that exhibits the Joseph effect. There is, moreover, a simple relation between the parameters describing the intensities of the Noah effect (high variability) and the Joseph effect (self-similarity). An extensive statistical analysis of high time-resolution Ethernet LAN traffic traces confirms that the data at the level of individual sources or source-destination pairs are consistent with the Noah effect. Implications of this simple physical explanation for the presence of self-similar traffic patterns in modern high-speed network traffic were discussed.

M. Mathis et al. [27] proposed that TCP may experience poor performance when multiple packets are lost from one window of data. With the limited information available from cumulative acknowledgments, a TCP sender can only learn about single lost packet per round trip time. An aggressive sender could choose to retransmit packets early, but such retransmitted segments may have already been successfully received. A Selective Acknowledgment (SACK) mechanism, combined with a selective repeat retransmission policy, can help to overcome these limitations. The receiving TCP sends back SACK packets to the sender informing the sender of data that has been received. The sender can then retransmit only the missing data segments. This memo proposes an implementation of SACK and discusses its performance and related issues.

I.Nikolaidis et al. [28] evaluated the statistical multiplexing gain in ATM networks for bursty as well as variable bit rate (VBR) traffic using a fluid-flow approximate model. The required bandwidth per source in a finite buffer multiplexer in order to achieve a given Grade of Service (GOS), expressed by the cell loss probability was obtained. For both bursty and VBR traffic sources, they performed a sensitivity analysis of significant parameters in the homogeneous case (all traffic sources are of the same type). The required bandwidth for bursty sources is shown to depend on burst and buffer length through their ratio. Finally, the mixing of bursty traffic and variable bit rate is considered. The results obtained through simulation with approximations proposed in the literature were compared.

L. Zhang et al. [29] used simulation to study the dynamics of the congestion control algorithm embedded in the BSD 4.3-Tahoe TCP implementation. The simple case of a few TCP connections, originating and terminating at the same pair of hosts, using a single bottleneck link were investigated. In this paper the dynamics that results from two-way traffic (in which there are data sources on both hosts) was also investigated. The one-way traffic clustering and loss-synchronization a phenomenon was discussed. In addition, there are two new phenomena: (1) ACK-compression, which is due to the interaction of data and ACK packets and gives rise to rapid fluctuations in queue length, and (2) an out-of-phase queue-synchronization mode which keeps link utilization less than optimal even in the limit of very large buffers. These phenomena are helpful in understanding results from an earlier study of network oscillations.

D. D. Clark et al. [31] proposed that Bulk data transmission is now finding more and more application in various fields. The major performance concern of a bulk data transfer protocol is high throughput. Theoretically, a packet switched network allows any single user an unlimited share of the network resources. In the absence of other traffic, therefore, a user should be able to transmit data at the raw bandwidth of a network channel. In reality, achievable end-to-end throughputs over high bandwidth channels are often an order of magnitude lower than the provided bandwidth. Experience shows that the throughput is often limited by the transport protocol and its flow control mechanism. It is especially difficult to achieve high throughput, reliable data transmissions across long delay, unreliable network paths. In this paper they introduced a new transport protocol, NETBLT, which was designed for high throughput, bulk data transmission applications. They first analyze the impact of network unreliability and delay on the end-to-end transport protocol; they then summarize previous experience; next they show the design and implementation of NETBLT, followed by the initial experience. Generally speaking, errors and variable delays are two barriers to high performance for all transport protocols. The NETBLT design and experience explores general principles for overcoming these barriers.

H. Heffes et al. [32] studied the performance of a statistical multiplexer whose inputs consist of a superposition of packetized voice sources and data. The performance analysis predicts voice packet delay distributions, which usually have a stringent requirement, as well as data packet delay distributions. The superposition is approximated by a correlated Markov modulated Poisson process (MMPP), which is chosen such that several of its statistical characteristics identically match those of the superposition. Matrix analytic methods are then used to evaluate system performance measures. In particular, moments of voice and data delay distributions and queue length distributions were obtained. Laplace-Satieties transform of the voice and data packet delay distributions, which are numerically inverted to evaluate tails of delay distributions, was also obtained. It is shown how the matrix analytic methodology can incorporate practical system considerations such as finite buffers and a class of overload control mechanisms discussed in the literature. Comparisons with simulation show the methods to be accurate. The numerical results for the tails of the voice packet delay distribution show the dramatic effect of traffic variability and correlations on performance.

"3.RELATED WORK"

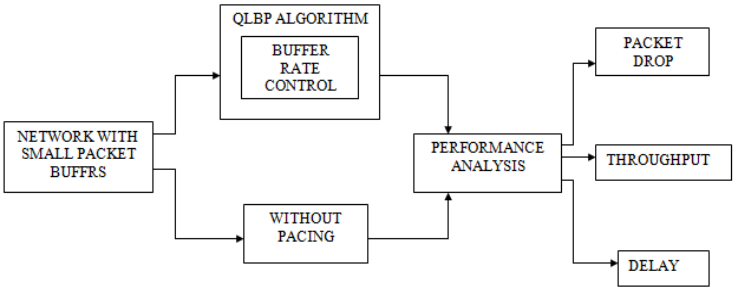
There are several possible approaches to addressing the problem of reducing the impact of packet loss on the delay in transport layer communication. The main techniques noted in this figure are:

• Lossy transmission: Using lossy transmission protocols (e.g., User Datagram Protocol (UDP)) places the bandwidth needs and delay close to the ideal lower bounds. Marginal amounts of additional bandwidth are necessary for packet headers and additional delay is incurred due to the packetized transmission of data.

• Reliable transmission: The baseline protocol for reliable transmission is the Transmission Control Protocol (TCP). Compared to UDP, TCP requires more bandwidth since some packets need to be retransmitted. It also incurs additional delay due to these retransmissions.

• Network coding: There are several coding techniques to reduce packet loss in networks. To reduce bit errors, error correction coding can be used. To avoid packet losses, transmission information can be spread across multiple paths in the network using network coding. These techniques require additional bandwidth since they rely on redundant transmission of information. These also exhibit increased delay over a lossy transmission due to the need for data reconstruction at the receiver. However, these techniques incur less delay than TCP.

In this work, the performance of the network is to be checked and a comparison is made with the proposed technique and the technique of non paced one. Firstly the data packets are allowed to send using the non paced technique and its performance in terms of throughput, packet drop and delays are analyzed. To make a comparison, the data packets are again sent using the proposed technique and various performances in terms of throughput, packet drop and delays are considered. Finally, it is found that the technique of Queue Length Based Pacing yields a better performance compared to the non paced technique, because eventhough some delays are intentionally made, all the data packets are successfully sent and received at the same time.So, there is no point of losing any data while transmitting. The general ideal of Queue Length Based Pacing (QLBP) is to dynamically adjust the sending rate of a queue according to the queue length, rather than to send packets at a constant rate.



"Figure 1. Block diagram of proposed work"

An extensive simulation model having scenario of 14 (user defined) wired nodes connected with duplex link is used to study inter-layer interactions and their performance implications. The other parameters used in this model are as under:

"Table 1. Simulation model specification"

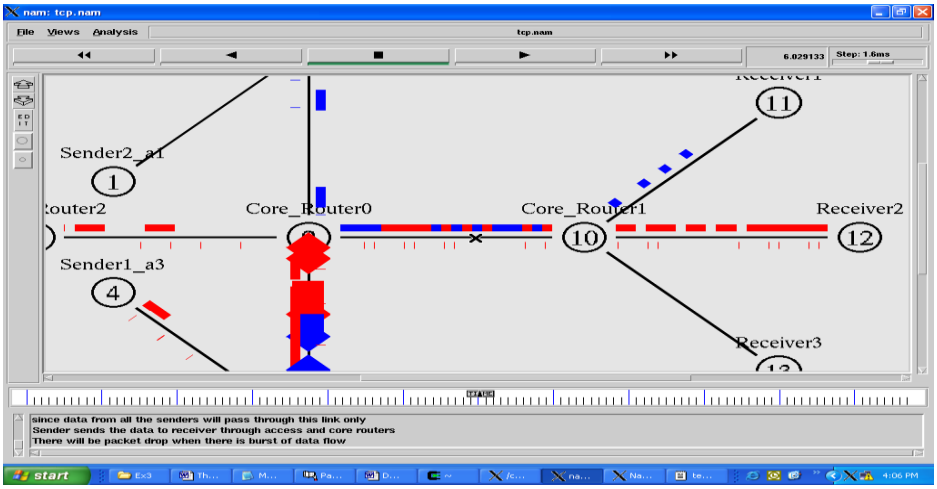
Software for Simulation	Network Simulator 2
Channel	Wired
Simulation run time	30 seconds
Area in which node move	600×600
Packet size	522 bytes
Application type	FTP/CBR
Transport agent	TCP
Link Type	Duplex Link
Queue Type	Drop Tail, FQ
Q- length	50-100Packets (variable)
Routing protocol	rlm
Propagation delay	variable

The type of link used for the proposed model is the Duplex link. The routing protocol used is the rlm (reactive layer multicast) protocol, the protocol used for wired routing. The queue type used in the proposed method is the Drop tail and the Fair Queue (FQ). After selecting the requirements for data transmission, the results are taken after simulation using the Network Simulator.

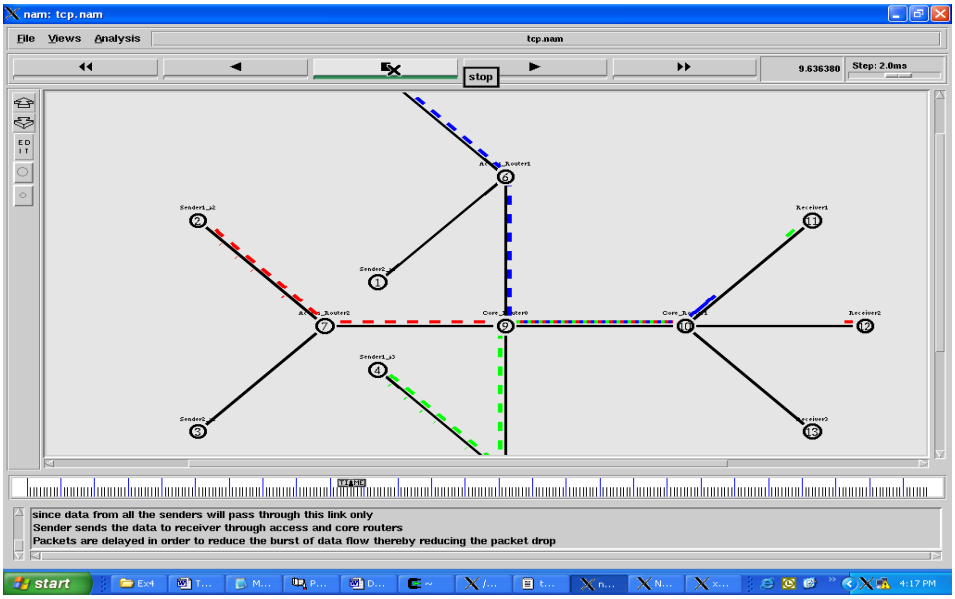
"4. Results and discussions"

The results are based on the simulations done on the non paced and paced techniques. Firstly, the simulations are performed on non paced techniques and finally on the paced techniques. The results are compared to verify the performances. The performance differentials are analyzed using throughput, delay, packet drops and buffer size.

As shown in the figure 2, due to the increased number of data packets being sent by the sending nodes packet drop occurs at the bottle link between core_Router0 and Core_Router1 due to the buffer over flow. The data packets are not yet received at the receivers but the sending process is not terminated but is continuing and due to this the receiver is not in a position to accumulate them all at their buffers. This result the packets to get dropped instead of being stored at the buffer. Thus, losing packets during transmission is like sending incomplete and meaningless data.



"Figure 2. Packets drop due to bursty traffic when applying non paced technique"



"Figure 3. Reducing bursty traffic by applying pacing technique"

As seen in the figure 3, all the data are received at the destinations, even though there is a burst of traffic in the network. So, there are no dropping of packets and due to this, a meaningful transmission is achieved just by delaying the packets during transmission.

Output comparison between existing and proposed technique

Propagation delay of the packet over the link is set as 50ms in the network in which no pacing technique is applied. Propagation delay of the packet over the link set as 100ms in the network in which pacing technique is applied since we intentionally delay the packet to improve the network performance.

Throughput of 4.35156e+09 bits/s is achieved in the network of dumbbell topology without pacing technique in 11.9 seconds with packet drop of 145 packets. But the same throughput is achieved over the same network with pacing technique in 26.6 seconds with no packet drop. So some delay is incurred using the proposed technique but the bursty traffic can be reduced thereby reducing the packet drop.

Propagation delay with pacing = 100ms

Propagation delay without pacing = 50ms

Buffer size = 100

"Table 2. Delay versus throughput and packet drop"

Delay(s)	Without pacing Throughput(bits/sec)	Without pacing Packet drop	With pacing Throughput (bits/sec)	With pacing Packet drop
11.998752	4.35156e+09	145	2090004480	No
26.67936	-	145	4.54045e+09	No

Propagation delay with pacing = 100ms

Propagation delay without pacing = 50ms

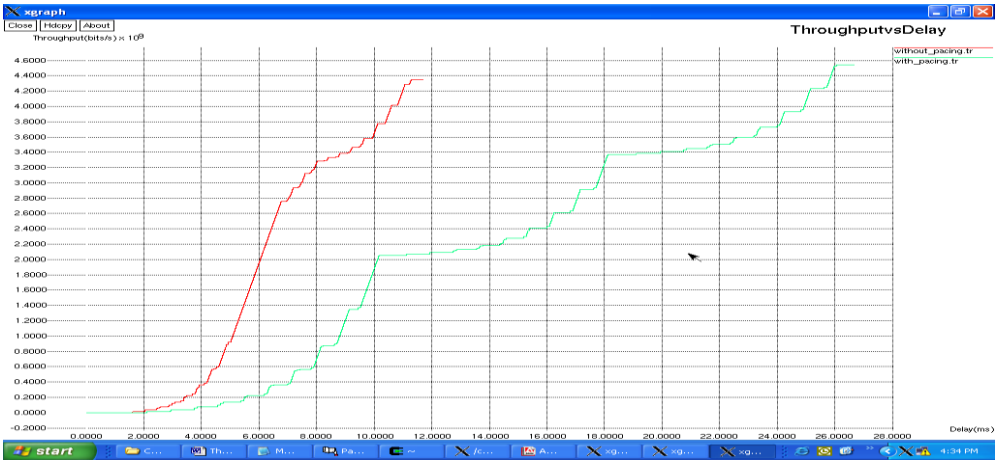
Buffer size = 100 and 50

"Table 3. Buffer size versus throughput and packet drop."

Buffer size (Packets)	Without pacing Throughput(bits/sec)	Without pacing Packet drop	With pacing Throughput (bits/sec)	With pacing Packet drop
100	4.35156e+09	50	4.54045e+09	No
50	3.24154e+09	145	4.54045e+09	No

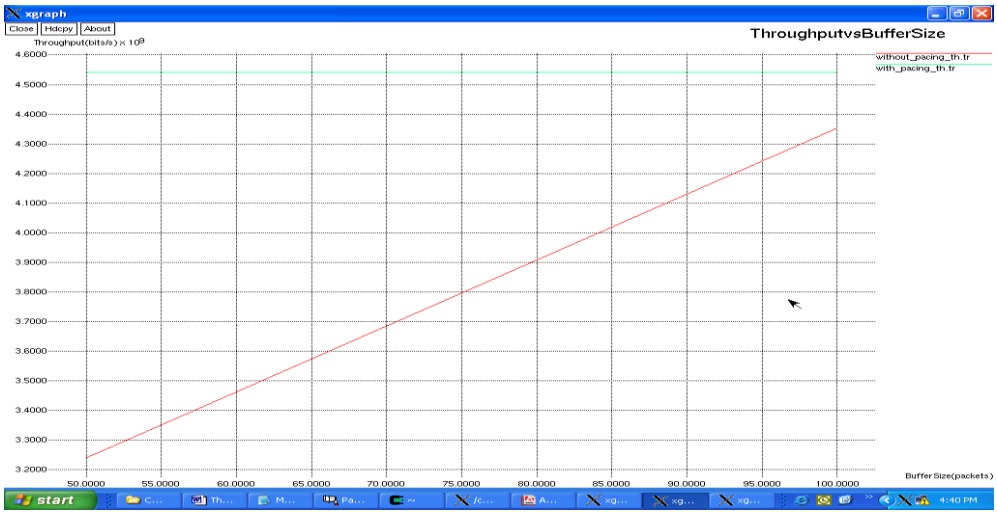
The network without pacing technique produces the lesser throughput when buffer size is decreased. The network with pacing technique produces the stable throughput even when the buffer size is decreased.

Comparative graph of transport layer performance with and without pacing technique



"Figure 4. Delay versus Throughput of non paced and pacing technique"

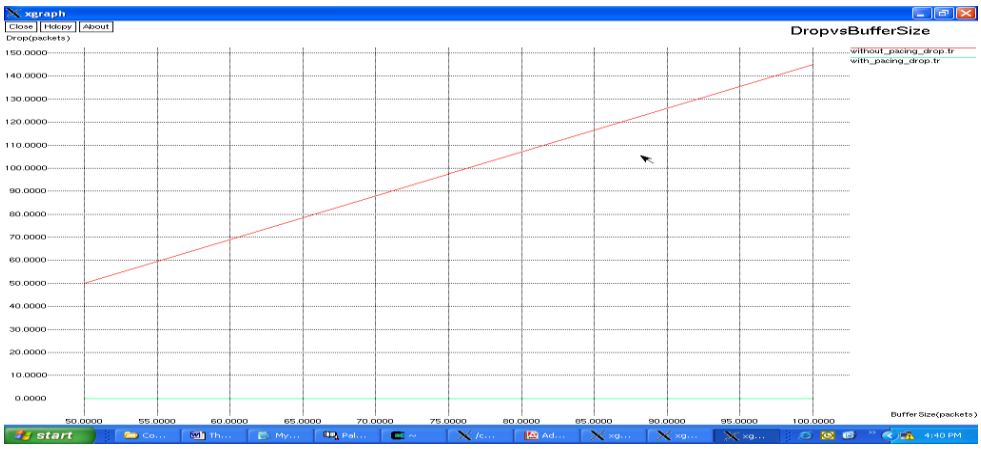
Network without pacing technique produces the higher throughput in 11 seconds. But the network with pacing technique produces the same throughput with some intentional delay but with no packet drop. Thus, achieving the same throughput and with no packet drop means a reliable and secure transmission since all the required data are received to the destination. So, this proposed technique of pacing is useful compared to the non paced method, because data packets are lost in the non paced one.



"Figure 4. Throughput versus buffer size"

Network without pacing technique produces the higher throughput when buffer size is increased. But the network with pacing technique produces the stable throughput even when buffer size is increased. Thus, in case of paced technique there is no effect of buffer size to the throughput. In the case of non paced technique, the throughput and buffer size increases linearly. Once the buffer size is increased the throughput also increases but not up to the paced technique, since in the paced technique the buffer size increases but the throughput remains at the same level.

As shown in figure 5, network without pacing technique produces the higher packet drop when buffer size is increased. But the network with pacing technique produces no packet drop when buffer size is increased. Thus, the pacing techniques is quite useful compared to the non paced one, since there is no packet lost at all.



"Figure 5. Packet drop versus buffer size"

Conclusion

This work presents a novel view on the tradeoff between link bandwidth and packet delay. Instead of using an error correction or network coding approach where more bandwidth is used to avoid packet losses, and proposed to delay packet transmissions to reduce the burstiness of traffic and thus reduce packet losses in small-buffer networks. This present Queue Length Based Pacing, which is a pacing technique that uses a single pacing queue on router ports and adapts its sending rate based on the amount of traffic that is buffered at that port. The analysis shows that pacing delay due to QLBP is bounded and that the variance of the instantaneous traffic rate is reduced. This shows that the effectiveness of QLBP through a prototype implementation and simulation. Specifically, the TCP connections in a small-buffer network with QLBP pacing achieve higher link utilization than in non-paced networks. Therefore, QLBP is an effective approach in improving the operation of networks and improving the effective bandwidth of connections at the cost of only small amounts of additional delay. Stability control mechanism is applied along with proposed technique using Fair Queue that helped to improve the network performance further.

References

[1] A. Vishwanath, V. Sivaraman, M. Thottan, and C. Dovrolis. Enabling a buffer less core network using edge-to-edge packet-level fec. In Proc. IEEE INFOCOM 10, San Diego, CA, March 2010.

[2] M. Shifrin and I. Keslassy. Small-buffer networks. Computer Networks, Volume 53, No. 14, pages 2552–2565, September 2009.

[3] Y. CAI, S. Hanay, and T. Wolf. Practical packet pacing in small-buffer networks. In ICC '09, Dresden, Germany, June 2009.

[4] A. Lakshmikantha, R. Srikant, and C. Beck. Impact of File Arrivals and Departures on Buffer Sizing in Core Router. In Proc. IEEE INFOCOM, 2008.

[5] O. Alparslan, S. Arakawa, and M. Murata. Node pacing for optical packet switching. In Proc. Photonics in Switching, Sapporo, August 2008.

[6] Y. Huang, Y. Liu, W. Gong, and D. Towsley. Two-level Stochastic Fluid Tandem Queuing Model for Burst Impact Analysis. In IEEE CDC, December 2007.

[7] Y. Gu, D. Towsely, C. V. Hollot, and H. Zhang. Congestion control for small buffer high speed networks. In Proc. IEEE INFOCOM 07, Anchorage, Alaska, pages 1037–1045, May 2007,.

[8] M. C. Weigle, P. Adurthi, F. H.-C. K. Jeffay, and F. D. Smith. Tmix: A tool for generating realistic TCP application workloads in ns-2. SIGCOMM Computer Communication Review, Volume 36, No. 3, pages 67–76, 2006.

[9] V. Sivaraman, H. Elgindy, D. Moreland, and D. Ostry. Packet pacing in short buffer optical packet switched networks. In Proc. IEEE INFOCOM 06, Spain, April 2006.

[10] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Rough garden. Routers with very small buffers. In Proc. Twenty-fifth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2006), Barcelona, Spain, April 2006.

[11] J. Naor, A. Rosen, and G. Scalosub. Online time-constrained scheduling in linear networks. In Proc. IEEE INFOCOM 05, Miami, FL, March 2005

[12] G. Raina, D. Towsely, and D. Wischik. Part II: Control theory for buffer sizing. ACM SIG COMM Computer Communication Rev, pages 79–82, July 2005.
[13] J. DeHart, F. Kuhns, J. Parwatikar, J. Turner, C. Wiseman, and K. Wong. The open network laboratory: a resource for networking research and education. ACM SIGCOMM Computer Communication Review, Volume 35, No. 5, pages 75–78, October 2005.
[14] D. Wischik and N. McKeon. Part I: Buffer sizes for core routers. ACM SIGCOMM Computer Communication Rev, pages 75–78, July 2005.
[15] M. Adler, S. Khanna, R. Rajaraman, and A. Rosen. Time-constrained scheduling of weighted packets on trees and meshes. Algorithmic, Volume 36, No. 2, pages 123–152, 2003
[16] C. V. Hollot, Y. Liu, V. Misra, and D. Towsley. Unresponsive Flows and AQM Performance. In Proc. IEEE INFOCOM, April 2003.
[17] Y. Wu, W. Gong, and D. Towsley. Analysis of abstract simulation via stochastic differential equation models. In IEEE CDC '03, December 2003.
[18] M. Adler, A. L. Rosenberg, R. K. Sitaram, and W. Unger. Scheduling time-constrained communication in linear networks. Theoretical Comp. Sc, Volume. 35, No. 6, pages 559–623, 2002.
[19] A. Razdan, A. Nandan, R. Wang, M. Y. Sanadidi and M. Gerla. Enhancing TCP Performance in Networks with Small Buffers. Network Research Laboratory Computer Science Department University of California, Los Angeles, CA 90095-1596, 0-7803-7533-X/02/$17.00 (C) IEEE 2002.
[20] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. Network information flow. IEEE Trans. Inf. Theory, Vol. 46, No. 4, pages 1204–1216, July 2000.
[21] A. Aggarwal, S. Savage, and T. Anderson. Understanding the performance of TCP pacing. In Proc. IEEE INFOCOM 2000, Tel Aviv, Israel, pages 157–1165, Mach 2000.
[22] R. W. Brockett, W. Gong, and Y. Guo. Stochastic analysis for fluid queueing systems. In IEEE CDC, December 1999.
[23] J. D. Salehi, Z. Zhang, J. Kurose, and D. Towsley. Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing. IEEE/ACM Trans. Netw. Volume 6, No. 4, pages 397–410, 1998
[24] M. Aron and P. Druschel. TCP: Improving start up dynamics by adaptive timers and congestion control. Rice University, Technical Report TR98-318, 1998.
[25] V. Visweswaraiah and J. Heidermann. Improving restart of idle TCP connections. University of Southern California, Technical Report TR97-661, 1997.
[26] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson. Self similarity through high-variability: Statistical analysis of Ethernet LAN traffic at the source level. IEEE/ACM Trans. Netw., Volume. 5, pages 71–86, 1997.
[27] M. Mathis, J. Mahdavi, S. Floyd, and A.] J. Hoe. Start-up dynamics of TCP's congestion control and avoidance schemes. Masterthesis, MIT, June 1995.
[28] I. Nikolaidis and I. Akyildiz. Source characterization and statistical multiplexing in ATM networks. Georgia Tech., Technical Report GITCC 92-24, 1992.
[29] L. Zhang, S. Shenker, and D. D. Clark. Observations on the dynamics of a congestion control algorithm: the effects of two way traffic. In Proc. ACM SIGCOMM 91, Zurich, Switzerland, pages 133–147 September 1991.
[30] Van Jacobson. Congestion avoidance and Control. In Proceedings of the ACM SIGCOMM '88 Conference on Communications Architectures and Protocols, pages 314–329, August 1988.
[31] D. D. Clark, M. M. Lambert, and L. Zhang. NETBLT: A high throughput transport protocol. ACM SIGCOMM Comp. Comm. Rev., Volume 17, pages 353–359, August 1987.
[32] H. Heffes and D.Lucantoni. A markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance. In IEEE J. Sel. Areas Communication. Pages 856–868 September 1986.
[33] J. Postel. User Datagram Protocol. Information Sciences Institute, RFC 768, August 1980.
[34] Vilas Bagad, Iresh Dhotre. Computer Communication Networks. Technical Publications Pune, First edition: October 2005, Chapter 6, pages 1-24, reprint: August 2006.