Maximizing Throughput of Reliable Bulk Network Transmissions

by

John W. Byers

B.A. (Cornell University) 1991

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Computer Science

in the

GRADUATE DIVISION
of the
UNIVERSITY of CALIFORNIA at BERKELEY

Committee in charge:

Professor Christos Papadimitriou, Chair
Professor Richard M. Karp
Professor Michael Luby
Professor Dorit Hochbaum

Fall 1997

The dissertation of John W. Byers is approved:

Chair Date

Date

Date

Date

University of California at Berkeley

Fall 1997

Maximizing Throughput of Reliable Bulk Network Transmissions

by
John W. Byers

Abstract

Maximizing Throughput of Reliable Bulk Network Transmissions

by

John W. Byers
Doctor of Philosophy in Computer Science

University of California at Berkeley

Professor Christos Papadimitriou, Chair

We study combinatorial optimization and on-line scheduling problems which arise in the context of supporting applications which transmit bulk data over high-speed networks. One of our primary objectives in this thesis work is to formulate appropriate theoretical models in which to develop and analyze efficient algorithms for these problems – models which reflect both the experience of network architects, the design of network protocols, and contributions of theoretical research.

We first consider the optimization problem of maximizing the utilization of a shared resource, network bandwidth, across a set of point-to-point connections. A feasible solution to this allocation problem is an assignment of transmission rates to the connections which does not violate the capacity constraints of the network links. The connections and routers which are responsible for establishing this allocation must do so with incomplete information and limited communication capabilities. We develop a theoretical model which addresses these considerations and study the tradeoff between the quality of the solution we can obtain and the distributed running time. Our main theoretical result is a distributed algorithm for this problem which generates a feasible $(1+\epsilon)$-approximation to the optimal allocation in a polylogarithmic number of distributed rounds, improving on a logarithmic approximation ratio obtained in a logarithmic number of rounds due to [AA 94]. A sequential implementation of our distributed algorithm gives a simple, efficient approximation algorithm for general positive linear programming. Subsequent experience with an implementation of the algorithm indicates that it is well suited to future deployment in high-speed networks.

The next problem we consider is the following on-line scheduling problem, which the sender of a point-to-point bulk transmission must address: Given an on-line sequence of transmission times, determine which data item to transmit at each transmission time, so as to maximize effective throughput to the receiver at all points in time. For this application, we measure effective throughput as the length of the intact prefix of the message at the receiver. This problem is made difficult in practice by factors beyond the sender's control, such as packet loss and wide variance in packet round-trip times. Using the method of competitive analysis, we compare the performance of our algorithm to that of an omniscient algorithm. We prove that while all deterministic policies perform poorly in this model, a simple randomized policy delivers near-optimal performance at any given point in time with high probability. Moreover, our theoretical result ensures that typical performance does not degrade significantly – a claim which our empirical studies bear out.

Using the models and tools developed for these problems, we then consider analogous problems which arise for *multicast* bulk transmissions, transmissions targeted to multiple destinations. We show how to tune our bandwidth allocation policy to still deliver a $(1+\epsilon)$ approximation to the optimal allocation in a polylogarithmic number of distributed rounds. For the scheduling problem, we prove that no on-line scheduling policy can deliver high performance which scales with the number of receivers without using encoding. We then show that by using forward error correction coding techniques, a simple multicast policy delivers effective throughput within a constant factor of optimal independent of the number of receivers.

Professor Christos Papadimitriou
Dissertation Committee Chair

To Lisa, who was there every step of the way.

Contents

List of Figures

Chapter 1

Introduction

Integrated-service networks currently under development will support applications with a diverse set of performance constraints and traffic characteristics. One emerging theme in the design of integrated-service networks is the need for multiple classes of service to support the diverse requirements of these applications effectively. At present, there are at least three distinct classes of applications which are likely to compete for shared resources such as link bandwidth in these networks. The first class, *interactive applications*, are characterized by transmission of short point-to-point messages and typically involve multiple traffic sources. The primary performance requirement of interactive applications, exemplified by Internet telephony, is to minimize the end-to-end latency of these brief transmissions. Second, *streaming applications* such as real-time video broadcast, are characterized by continuous injection of data into the network by a single source. The requirements of such applications can be complex; it is often necessary to deliver the data to the receiver or receivers at a smooth rate while keeping the end-to-end latency of transmissions low. Finally, *bulk applications*, such as point-to-point file transfer, involve reliable dissemination of data from a single source in the network to one or more receivers. These applications, which currently comprise a large fraction of Internet traffic, have a relatively simple performance requirement to satisfy: deliver the data to the receivers as quickly as possible.

In this thesis, we focus on developing efficient algorithms for optimization and scheduling problems which arise in the context of providing support for bulk applications in integrated-service networks. Some of the algorithmic techniques we present also provide support for streaming applications. The first problem we study in this context regards the allocation of bandwidth to bulk applications in the network. Since the goal of these applications is to deliver data to the receivers as quickly as possible, if left unchecked, they can inject data into the network at rates which could overrun the capacity of the underlying links. We assume that sources inject data into the network using *packets* with a bounded maximum payload size. When packets arrive at a network router faster than they can be processed, queue buildup results, followed eventually by packet loss when the queues overflow.

One emerging technique used to address this problem, *rate-based flow control*, regulates the rates at which point-to-point applications may inject data along the fixed paths they use in the network. If packet sizes are fixed, these rates can be allocated in packets per second, but they are more commonly specified in bits per second. A feasible end-to-end allocation of rates to a static set of connections is one in which each connection may subsequently transmit data at their allocated rate without violating the capacity constraints of the network. Designing rate-based flow control algorithms is made difficult by the *distributed* nature of the computational resources in the network and by the *dynamic* arrival and departure of connections over time. We place substantial emphasis on developing a distributed model which captures the limited communication and computational capabilities of these network resources accurately. While one widely studied objective of this approach by researchers in the networking community is distributing the bandwidth *fairly* across the connections, we will motivate the study of a different objective function for this problem: maximizing network throughput.

The second problem we address regards a scheduling problem that the sender of any network transmission must solve: selecting what data to place in each packet of the transmission to make the most effective use of the available bandwidth. Complicating the issue is the fact that packets in the network may be lost, they may experience widely varying end-to-end network latency, and the rate at which a source can inject packets into the network may fluctuate dramatically over time. Moreover, these complications are brought on by factors beyond the sender's control. Specific applications may have to deal with additional considerations. For example, streaming applications must decide whether it is worthwhile to retransmit a lost packet which may not arrive in time to be of value to the receiver. Likewise, multicast applications must decide whether to retransmit a packet (to all multicast recipients) which has been lost by a subset of the multicast group. We formulate these problems more precisely after outlining our model of the network.

1.1 Networking Terminology

Throughout this thesis, we model the network as a capacitated, directed graph with nodes representing connection endpoints and network routers, edges representing links, and capacities representing link bandwidth. Since the router at the tail of an edge makes admission control decisions for that edge, we will frequently think of the routers, rather than the edges, as being capacitated. Often, a single physical router may reside at the tail of a constant number of network links; we model this phenomenon by using a separate virtual router for each outgoing edge. Each router may maintain a constant amount of state about each of the connections which utilize it. A source communicates by injecting *packets* into the network, each of which contains header information and a payload of bounded size. In many protocols, a packet arrival at a receiver triggers an *acknowledgment*, in the form of a packet initiated by the receiver and targeted to the sender. When acknowledgments are employed, we use the standard term *round-trip time* of a packet to denote the elapsed time between the sender's transmission of a packet and the sender's receipt of an acknowledgment for that packet. Throughout this thesis, we model the network as *connection-oriented*, in that for every application, all packets traveling from a source to a given destination use the same fixed route in the network. In this spirit, we often refer to the communication channel between a source and destination as a *connection* in the network, even though the connection is a virtual one. Our model is intended to reflect the reality of a trend in high-speed networks towards connection-oriented architectures with per-connection state inside the network [MSZ 96].

We separately consider unicast and multicast applications, applications which transmit data to a single destination and multiple destinations, respectively. In a connection-oriented network, a unicast application establishes a connection along a source-destination *path* in the network. Therefore, packets injected by a unicast source traverse the routers and edges along the fixed path to the destination. A multicast application communicates along a set of edges and routers that form a multicast *tree* in the network rooted at the sender. The leaves of the multicast tree correspond to the destinations of the transmission. In standard multicast implementations, such as IP Multicast, when a packet reaches a router corresponding to an internal node in this multicast tree, that router forwards a copy of the packet on *all* of its outgoing links which are edges of the multicast tree. We employ this model in our work as well, and describe it completely in Chapter 5.

1.2 Bandwidth Allocation

The most widely studied objective of rate-based flow control in the networking community is *max-min fairness* [Jaf 81]. A max-min fair equilibrium state has the following characterization: no connection can increase their rate without decreasing the rate of some other connection with equal or lower rate [BG 87]. Algorithmic techniques for efficiently converging on a max-min fair allocation for a set of bulk connections is described in the section on related work.

In many settings, fairness is not the most important consideration, and in fact, much of the theoretical work on rate-based allocation focuses on maximizing network *throughput.* There is extensive motivation for studying this objective function. In a pay-per-use network, network administrators are interested in generating allocations which maximize the throughput of the network, or maximize the revenue derived from the set of connections. In intranets where bulk connections can be prioritized, higher priority connections warrant receiving higher rate allocations. The objective in these scenarios can be expressed as maximizing the weighted throughput of the connections. A drawback of throughput maximization is that some low-profit connections may receive an allocated rate of zero. However, it is possible to combine approaches, first by providing a minimal rate, or fairness, guarantee to all connections, then maximizing the throughput on the residual network capacity.

We begin with a simplified formulation of the static bandwidth allocation problem where our objective is to maximize network throughput, measured as the sum of the allocated rates across n point-to-point connections. This problem is most easily expressed as the following simple linear program.

$$\begin{aligned} \max \sum_{j=1}^{n} y_j \text{ s.t.} \\ \forall i, \sum_{j} a_{ij} y_j \leq C_i \\ \forall j,\ y_j \geq 0 \end{aligned}$$

In the program we solve for the variables y_j, which correspond to the flow rates assigned to the connections. The 0/1 constants a_{ij} are set to 1 if the route of connection j utilizes router i and 0 otherwise. The constants C_i hold the value of the capacity of router

i. A feasible solution to this program corresponds to an end-to-end allocation to the set of connections.

There are numerous obstacles to developing an efficient algorithmic solution to this optimization problem which can be successfully deployed in a high-speed network. One fundamental obstacle is the *distributed* nature of the computational resources in our network. Although a centralized coordinator with complete information about the system, and thus the linear program, could compute an optimal allocation in polynomial time, no such coordination exists in a distributed network. In the system we study, connections are autonomous, meaning that they have no prior knowledge of other connections in the system, nor does the network have any prior knowledge of which connections will be present. We model this behavior by viewing connections and routers as distributed agents. Although these agents initially have very limited information, they are able to communicate in a natural, but restricted way. In particular, connections may communicate with routers through which they pass, and routers with connections which pass through them. Then with the incomplete information they obtain about the system, they must then decide upon a feasible allocation. We formalize this model, which derives from work of Papadimitriou and Yannakakis [PY 93] in Chapter 3.

In the setting of networking, the distributed running time is a crucial consideration, as algorithms with which run in polynomial time are often too inefficient to warrant practical implementation. It can therefore be important to sacrifice optimality for the sake of more efficient running time. We consider the trade-off between the distributed running time and the quality of the solution we obtain, measured in terms of the *approximation ratio*, the ratio between the value of the optimal solution and the value of the solution our algorithm obtains. Also, we strive to keep our algorithms as simple as possible, since they will be implemented at the connection endpoints and routers of the network, where processing time is a critical resource.

Another consideration is that of *dynamic* arrivals and departures of connections in the system. A solution to the static allocation problem is useful only while the set of connections involved in the allocation remain in the system. We consider how to smoothly and quickly adjust the allocation as connections arrive and depart. We address the following questions about this approach in this thesis:

Question 1 *What communication and computational model for the network is appropriate* *for designing distributed bandwidth allocation policies.*

Question 2 *What quality of solution and running time bounds can we obtain in this model?*

Question 3 *Are the algorithms we design sufficiently simple and efficient to warrant practical implementation?*

The next step is to extend these models and results to the setting in which multicast connections are also present in the network. We formalize a model in which to study allocations to multicast connections in Chapter 5. There, we explore the tradeoff between the optimality of the allocation and the distributed running time in this model for both multicast bulk applications and multicast streaming applications.

1.3 Data Selection Policies

The next problem we consider is the on-line scheduling problem which the sender of a unicast (single destination) bulk transmission faces: Given a set of fixed-size words $w_1, w_2, \ldots, w_M$ comprising a message and a sequence of events $\rho_1, \rho_2, \ldots, \rho_N$ which either request the sender to send a word, acknowledge the receipt of a word, or acknowledge that a word has been discarded, the sender must decide what data to transmit at each send request. In this on-line setting, the sender must determine which data to place into each packet of the transmission with imperfect information about the future. A primary objective of the sender is to minimize the completion time; we are interested in the more stringent requirement of making the most effective use of the available bandwidth at intermediate points of the transmission as well. We measure the performance of our algorithms at these intermediate points in time based on the length of the *prefix* of the message successfully delivered to the receiver.

There are at least three factors which contribute to the sender's uncertainty about the future, all of which the sender *cannot control* when best-effort protocols are employed, and network service is not guaranteed: Any packet which the sender transmits might be discarded. The round-trip time of packets in the network can vary widely over time. The rate at which a sender is allowed to inject packets into the network can increase or decrease based on external factors such as network congestion. Moreover, recent experience in modeling these phenomena indicate that they are extremely difficult to predict. Traces of packet loss patterns indicate bursty, self-similar behavior [Pax 96, YKT 96, LTWW 95]. Variance in packet latencies and packet interarrival times on the Internet is large [Jac 88, Pax 97]. These factors motivate studying the performance of transmission protocols on arbitrary transcripts of event sequences. In this worst-case model, the algorithm is asked to deliver high performance on any possible sequence of events relative to an optimal algorithm, and is graded based on its worst-case performance.

Although this worst-case specification may seem overly pessimistic, it provides a useful measure of assessment for data selection policies. Our main objective is to develop an algorithm which performs well with respect to the worst-case measure, but a secondary objective is to assess the performance of widely used data selection policies. If they fare poorly in the model, we are interested in understanding why, and consider how to improve them. A common complaint regarding competitive analysis of on-line algorithms is that optimizing worst-case behavior is an ill-advised strategy for practical scenarios which rarely exhibit worst-case behavior. Our response to this complaint is that for the problem we describe, the cost of tolerating worst-case behavior is minimal in terms of typical performance degradation. Moreover, existing algorithms which are not worst-case tolerant incur a large penalty when worst-case behavior arises. Our answers to the following questions will address these considerations:

Question 4 *What worst-case performance can an on-line unicast data selection policy deliver?*

Question 5 *To what extent does our data selection policy degrade performance over existing data selection policies on "typical" network behavior?*

Another topic we study is the interplay between bandwidth allocation policies and data selection policies. In general, given knowledge of the allocation policy, it may be possible to fine-tune a data selection policy to deliver higher performance for that particular allocation policy. This is precisely what is done in TCP; the protocol for packet retransmission is carefully intertwined with the policy for congestion control to deliver better performance. However, we prove that such fine-tuning can gain very little when the correct data selection policy is used. The unicast data selection policy we present provides *universally* high performance in conjunction with any allocation policy, and its performance can only be improved by an additive term. Therefore, bandwidth allocation policies and data selection policies for unicast connections can be viewed as "orthogonal". This observation leads to a better understanding of unicast transmission protocols, in that it gives provable grounds supporting the idea of explicitly separating the two policies. This modular separation has been suggested by other authors in the networking community, including [MM 96], but is not found in the most widely used protocols.

1.3.1 Data Selection Policies for Multicast Connections

For developing data selection policies for multicast connections, the main additional complication is *scalability*. Ideally, we would like the performance of our data selection policies to remain fixed as the number of receivers increases, all other things equal. However, this may not be possible, as a transmitted word which arrives successfully at some receivers may be discarded en route to other receivers, leading to a situation where retransmission of that word benefits some, but not all destination endpoints. In this setting, we prove that any data selection policy which only transmits words of the original message is inherently unscalable, as performance degrades when the number of users increases. Also, it is not the case that we can develop a multicast data selection policy which provides universally high performance in conjunction with any allocation policy, as we prove for unicast. On the positive side, we prove that as long as the allocation regime provides a minimal quality of service guarantee, one can define a *scalable* multicast data selection policy which has worst-case performance within a multiplicative constant factor of optimal. This policy uses *forward error correction (FEC)*, which transmits packets with redundant encoded data along with packets containing words of the original message to recover from packet loss in the network. For multicast connections, we address the following questions:

Question 6 *What level of performance and scalability are achieved with the use of FEC?*

Question 7 *How much overhead is required to use the encoding technology?*

To summarize, the topics described in this thesis focus on implementation of policies which optimize bandwidth utilization for both unicast and multicast bulk-data applications. The first is a network-wide allocation strategy which regulates the *rates* at which applications may inject packets into the system. The second is an application-level strategy which determines the *content* of those packets. These policies can either be used separately, or in conjunction with one another, and both guarantee bounds on worst-case performance.

1.4 Theory vs. Practice

Although our work is theoretical in nature, we place a particularly heavy emphasis on developing algorithms and models which address real problems in existing networks. To this end, the models we develop draw from experience gained from both practical and theoretical research, a fact which is reflected in this introduction and in the material we summarize in the chapter on previous work. We have also implemented and simulated all of the algorithms which we present and incorporate our findings into the chapters which follow.

1.5 Thesis Outline

The rest of the thesis is organized as follows. In Chapter 2, we discuss related work on mechanisms for allocating bandwidth to bulk connections in the network, data selection policies, and multicasting. In each case, we begin by summarizing the state of the art in existing networks, as this viewpoint helps form the basis for our theoretical models. We then discuss related theoretical work on each of the topics.

In the next two chapters, we begin a technical discussion of the bandwidth allocation and data selection policies we have developed in the simpler case in which all applications are unicast applications, i.e. targeted to a single receiver. In Chapter 3, we focus on bandwidth allocation policies maximizing weighted throughput. We show how to formulate the bandwidth allocation problem as a positive linear program and extend the model of Papadimitriou and Yannakakis [PY 93] for solving positive linear programs with distributed decision-makers. In our extension of their model, the distributed decision-makers are the connections and routers comprising the network. Using a natural set of communication primitives, we show how these agents can compute a feasible $(1+\epsilon)$-approximate solution to the global optimum in a polylogarithmic number of distributed communication rounds. This theoretical result appeared in preliminary form in [BBR 97] as joint work with Yair Bartal and Danny Raz. We then provide a discussion of the implications of this technique for solving the bandwidth allocation problem, in part focusing on experience with a serial implementation of the algorithm.

In Chapter 4, we focus on data selection policies. We measure the performance of a data selection policy according to the prefix measure described above, and in comparison to the offline optimum performance obtained by an optimal, omniscient algorithm. We then develop a formalism for analyzing policies according to this measure, based largely on existing network protocols. We prove that no deterministic data selection policy, such as that used in TCP, can deliver worst-case performance whose performance can be bounded by an additive term away from optimal. However, we show that with randomization, we can develop an algorithm which delivers very high performance, within an additive term from optimal at any point in time with high probability. Many of the theoretical results in this chapter appeared in preliminary form in [ABBLR 97] as joint work with Micah Adler, Yair Bartal, Michael Luby and Danny Raz. We bolster our results with comparative simulations of our algorithm against other data selection policies on network traces taken from the Internet and on simulated worst-case behavior.

In Chapter 5, we extend our results to networks which admit both multicast and unicast connections. We show how to incorporate reliable multicast connections into our positive linear programming formulation developed in Chapter 2. We then consider the interesting problem of choosing a data selection policy for multicast connections. As described earlier, it is quite easy to prove that the use of *encoding* is a prerequisite for a high-performance, scalable data selection policy in our model. We describe recent innovations in the area of forward error correction (FEC) which we then use to develop a scalable data selection policy for multicasting which delivers high performance. Again, we present empirical results from Internet trace data and on simulations we generated to demonstrate the high performance of the algorithm relative to existing policies.

We conclude with a discussion of the work described in this thesis, along with directions for future research work in the area in Chapter 6.