Some Thoughts on the Packet Network Architecture

Lixia Zhang

Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

Performance control in packet switching networks is one of the remaining areas that are yet to be explored. In this paper we look into the performance problems with two existing network architectures, datagram and virtual circuit. The major problem with the datagram network is lack of effective traffic control; and with the virtual circuit network, the improperly defined network control semantics and mechanisms. We consider that a new network architecture should be investigated in order to provide performance support for diverse applications.

1. Introduction

In packet switching computer communication networks, the network layer in the architecture plays an important role. Its functionality is to deliver user data bits with specified performance requirements. Above it is the end-to-end transport layer residing in users' hosts. It is the network layer, together with the layers below it, that physically carries out data transmissions. Its performance determines how good the network transmission services will be.

Traffic congestion has been a major obstacle in achieving good network performance. Data from all sources are statistically multiplexed on shared network resources; the moment-to-moment bandwidth requirement often varies dramatically. The network is designed to tolerate modest traffic fluctuations, but not without limitation. Whenever a packet surge exceeds the limit, it may congest one or more network switches and bring about disastrous consequences: creating long transmission delays, causing data losses, or even blocking up the network, breaking down end users's connections. For example, the experience of running the ARPA Internet shows that congestion has been the major cause of network vulnerability. Among the problems remaining unsolved in packet switching networks, congestion control is perhaps one of the least understood [11].

Meanwhile, more and more new applications are being introduced into packet switching networks, such as real time voice, image data, and teleconferencing. These new applications

raise stringent and diverse performance requirements. They require that the network not only prevent congestion, but also provide a guarantee of various service characteristics, such as low transmission delays or high throughputs. New applications bring another impetus to the research for an effective network performance control. Their successes, or the future success of the packet switching, will depend on how well the network can meet users' diverse performance requirements.

In this paper, we examine problems in the two existing networks architectures, namely datagram and virtual circuit, then identify new directions to solve the problems. We believe that a new network architecture is needed to provide effective network traffic control and to facilitate network resource management[1].

2. The Problem

This section describes the two network approaches, datagram and virtual circuit (VC), in turn, and pinpoints their problems. In particular, we consider how well they perform network resource management, and how well they can meet users's diverse performance requirements.

Our focus will be on the network *internal* architecture and protocols, rather than the network *interface* to host machines. We are interested in resource management in geographically distributed, packet switching networks. Processing power, communication channels, and buffer space are the three major components in the network, with the first two as the driving forces to accomplish data transmissions. We call the processing power and communication channels together as the network *data forwarding resources*, which exclude buffer space. The asynchronous resource sharing feature of packet switching requires data buffering inside the network, in order to resolve temporary contentions on the forwarding resource . Buffer space itself, however, does not directly contribute to data forwarding tasks.

2.1. Datagram Networks

Datagram networks offer a simple and flexible data transmission service, the so called "best effort" delivery. They deliver data packets as independent entities. Each packet, called a *datagram*, carries its own destination address, which is used by the switch node to dispatch it

[1]To make the study easier, we temporarily ignore the network routing consideration in this paper, leaving it for future studies.

onto the next hop. No information about user data flows is kept inside the datagram network. Along the way packets may be damaged, duplicated, reordered, or lost. Example datagram networks are the ARPA Internet, Cyclades network, and DECNET [13, 15, 21].

The "stateless" feature of datagram networks is considered highly desirable for several reasons [5]. The most important one probably is to provide robust services in the face of network component failures: a switch node crash does not lose any information concerning the control of data transmissions; packets can freely change route to get around failed areas. Another reason is to simplify switch node implementations, and to facilitate interconnections between heterogeneous networks. Still another one is to offer network users the flexibility of building their own desired data transmission properties on top of provided network services[2].

Unfortunately, the "stateless" feature precludes datagram networks from having an effective traffic control, and network congestion has been a serious problem. Considerable effort was spent adding a congestion control part into the datagram network, but without much success. The first proposed congestion control algorithm, isarithmic algorithm [8], never satisfiably solved the problems of how to distribute data transmission credits, and how to recover from credit losses. Other proposed or implemented algorithms are, in one way or another, almost all around the idea of managing packet buffers at switch nodes in certain topology-dependent ways[3], hence prone to unfairness in services.

The root of the problem is that the datagram network sees individual packets only. Because any attempted algorithm has to work on a packet-by-packet basis, it can be very expensive; examples are the *source quench* in IP [14] and the *choke message* in Cyclades network [15]. Because the network recognizes individual packets, instead of end users, it is difficult or even impossible to guarantee the fairness in service. Because packets are considered independent entities, and because the network is unable to manage the data forwarding resources on a per packet basis, attention naturally turns to various buffer management schemes to decide the

[2]As pointed out in [5], a mistaken assumption often associated with the datagram is that the motivation for datagrams is a better match to some high level applications which require a datagram service. In fact, this is seldom the case; even transaction-like applications would like a more sophisticated transport model than the simple datagram (as shown in [3], for example). Whether one takes the datagram approach is a network *architecture* decision, rather than an issue of providing a specific kind of services.

[3]Examples are the favoring transit packets over input packets scheme, and the buffer allocation according to packet "ages" (see [10]).

acceptance of each incoming packet on a buffer availability basis.

Two common ways the datagram network reacts to congestion are either to block the host-network interface by the link layer protocol, or to drop excessive packets inside the network. Blocking the network interface locks out all data flows, both the connections that caused the congestion and others that do not; it may also cause the higher layer protocols to time-out and pump more duplicates to the interface. While dropped packets rely on the end-to-end level protocols to recover, substantial losses not only are expensive to recover, but also make high performance unattainable.

The best resource management the datagram network can do is a dynamic routing, which tries to evenly spread-out data traffic. This may delay, but does not prevent, network congestion. Congestion prevention requires that the network be able to control the traffic volume, i.e. to adjust source hosts data generation rates when needed. The datagram network is disabled from effectively doing so by its traffic model of "independent entities". Due to lack of control, the performance of the datagram network is up to the offered input. Only when the network is lightly loaded will the performance be satisfactory[4]. As soon as the network load increases to a moderate level, the performance will start degrading drastically, as has been observed in the ARPA Internet in the recent years.

We claim that packets in the datagram network are not independent entities. Most application processes generate a sequence of packets to accomplish one task, and packets in the net certainly have effects on each other through competing for network resources. Dynamic bandwidth sharing is the unique feature of packet switching; without control, however, this feature leads to data traffic interference. Shared resources require control to enforce an intelligent use. To perform the control, in turn, requires information about individual users' data transmissions. Because of the "statelessness", the datagram network surrenders its control, resulting in vulnerability under the circumstances of heavy traffic or user malfunction. A typical example of this sort is given in [12]: an implementation bug caused an ARPANET host to retransmit one datagram as fast as the network could accept, congesting a nearby gateway; the other networks attached to the same gateway were effectively disconnected for several hours.

[4]In fact, this has been the case in many networks. Most host protocol implementations are complex and slow; the tight end-to-end flow controls, and the limited types of current network applications do not strengthen the network throughout either.

until the malfunctioning host crashed.

2.2. Virtual Circuit Networks

VC networks[5] attempt to offer a reliable data delivery. Upon user request, a logical connection will be established hop-by-hop through the network. The VC network maintains the connection state at each switch node for two purposes: to check and remedy any "data integrity" damage, namely bit error, duplication or loss, and reordering, inside the network; and to control data flow on individual connections. Example VC networks are Transpac and Tymnet [7, 16].

Commercial networks, most of which adopted the VC approach, have enjoyed a rapid growth and a great success over the last decade. That is not a sufficient proof, however, for the VC network being the correct architectural model of packet switching. In fact, the success was achieved on a rather limited functionality basis. Contrary to the datagram network's goal of offering a flexible data delivery service on a network substrate (which is possibly composed of a variety of communication media), most VC networks existing today, especially the commercial ones, have been built with narrower ranges of constructing components and applications in mind. Namely, they employ low to medium speed telephone lines to connect up packet switches, and offer only bi-directional, reliable data delivery service, mainly aiming at supporting remote login applications. Such a homogeneous environment makes the traffic pattern rather predictable and flow control easier. Telephone lines give some unique communication channel features: low propagation delay, low error rate, and identical channel bandwidths across the network. The dominant remote login applications mean both a low throughput requirement and an undemanding performance expectation -- when the network load becomes heavy and transmission delays increase, end user's operations are consequently slowed down as well.

Known VC networks have all adopted a window flow control mechanism, either entry-to-exit or hop-by-hop. Some of them make buffer reservations at each switch node along with the virtual connection setup. Therefore compared with their datagram counterparts, VC networks have more control over data flows on virtual connections; when buffers are allocated to individual connections, traffic interference is also reduced: no single connection can transmit

[5]Here we consider networks that build virtual connections internally, not those that only have a virtual circuit interface (such as Datapac [20], which has a virtual circuit interface but employs datagram inside).

wildly since one's throughput is restricted by the allocated buffer space, which usually is very small.

Having a control mechanism does not necessarily imply an effective or efficient control, however. Small window sizes or small buffer allocations can severely constrain users' throughput in high bandwidth or long delay networks. Moreover, the control over data flows relies on the back-pressure effect to stop source hosts when congestion develops, i.e. data flows do not stop until filling up the paths with data; consequently. there are repeated packet losses and retransmissions between switches, albeit within the network boundary and hidden from the end hosts. Therefore a VC connection indeed does not appear as losing many packets, as the datagram net does, in the face of congestion; but congestion inescapably shows its effect at the end hosts, as intolerable transmission delays[6].

Network congestion is caused by inadequate data forwarding resources to deliver offered data. Window or buffer space allocation does not match the data forwarding resources allocations. The former is measured in *space*, the latter, in *rate* -- the switch node CPU processes a certain number of packets per second, and the communication channels drain out a certain number of data bits per second. Window by itself does not control the data flow rate; with a given window size, the actual transmission rate is determined by the transmission delay and the error rate (and hence the retransmission strategy), where the delay varies with the network load. A common practical approach is to use a network chosen window size, which is neither adjusted to the specific user's throughput nor adjusted according to the system load[7]. In fact, most VC networks's major tool of handling congestion is to rejects new call requests when congestion occurred; that is an aftermath, not a prevention.

Today's VC networks do not support diverse applications. They do not offer services with delivery time limits, because the hop-by-hop checking of error-free transmissions may cause a

---

[6]These conceptual arguments would be better supported by quantitative performance figures, unfortunately it is difficult to find such information from public literature; commercial nets do not seem to publish their detailed performance results.

[7]SNA pacing [1] is one of the few that tune the window size to adopt to the network load, but its effectiveness is limited. We see at least two problems with it: (1)The pacing uses an entry-exit window with the size proportional to the path length; when heavy load occurs, there usually exists some bottleneck point, and packets admitted by the large window of a long-path will get accumulated (if not dropped) at the bottleneck point, competing for the scarce resources. (2)Although congested nodes may set a flag to request window size reduction, the flag does not bring enough information as what a proper window size should be.

long time delay; they do not support specific throughput values, especially high throughputs of tens or hundreds Kbps; although some VC networks claim a selection of throughput classes, the throughput class parameters are actually used to decide the window sizes, or buffer allocations, or priorities, rather than to allocate needed data forwarding resources. A sad fact about the VC network is that it builds the rigid reliability mechanism into the network, and there is no way for a user to turn it off to achieve other desired features.

Even assuming reliable transmissions are the only goal of the network, [18] argues convincingly that an end-to-end error checking and recovery mechanism is absolutely necessary; performing error detection and recovery inside the network may have its gain and/or its loss in performance, as well as its cost in complexity. Therefore the network should leave users the option of whether employing the network's reliability checking mechanism, rather than forcing on them a wired-in feature. In addition, the VC network approach is also criticized for vulnerability [17], for cascading a connection state throughout intermediate switches leads to the connection being broken when any of the switches fails.

Let us take a couple of examples to see what goes wrong with the VC network control. A file transfer requires every bit be received correctly, but it is not much concerned with network delay, so long as the delay is within a reasonable boundary; while packet voice transmissions are more concerned with network transit delays than occasional bit errors, for packets become obsolete if not delivered within a certain time period. The different application requirements indicate that different state information is needed for managing the end connections and the network resources, though the latter are used to carry out data transmissions for the former. To a end user, the confirmation of a reliable file transfer requires correct reception of each data piece; such information is recorded in the *user connection state*. To carry out the transmission promptly, on the other hand, the network cannot commit resources to correct delivery of each piece; instead, it should allocate the needed data forwarding resources for the whole task. The network therefore needs to maintain information on the forwarding resources allocations, and keeps track with the start and termination of the transfer. Such information is considered the *network state*. Similar arguments apply to meeting delivery delay requirements as well: being a statistically multiplexed system, the packet switching network cannot guarantee a random data source a tight delivery time of every single packet; instead, the network concerns itself with the amount of data forwarding resources that should be allocated, in order to maintain the

probability of exceeding the delay limit within an acceptable region.

VC networks failed to recognize the above distinctions. They use the end connection state to control packet flows through the network, which, unfortunately, does not match well with the rhythm of the network resources.

2.3. New Applications

Up to now the major traffic in packet switching networks has been computer generated data transmissions from three applications: remote login, file transfer, and electronic mail. These applications typically tolerate narrow communication bandwidths and variant transmission delays. When serious problems occur inside the network, such as traffic congestion, they are usually reflected to the end-to-end layer protocols or human users to handle.

If we believe that datagram or VC networks have served us reasonably well in the past, they certainly will not be able to meet today's new challenge. With the advent of personal computers and workstations, the once dominant network application of remote login has started diminishing. At the same time, the demand to offering various types of service to meet new applications requirements is ever increasing. Transmitting real time voice requires a short and stable transit delay; delivering image data requires high throughput; a recently developed bulk data transmission protocol, NETBLT [6], expects the network to offer a relatively stable throughput to facilitate the end host for high performance. Neither datagram nor VC networks can fulfill the mission of ensuring needed transmission service characteristics. A new architecture for packet switching networks need be investigated.

2.4. Summary

The above discussions showed the problems with datagram and VC networks, respectively. The datagram network contains no information concerning user data flows, therefore it is unable to effectively control the traffic. The VC network uses the end connection state for network control purpose, which does not match well with rate-based network resources. Because they both do not directly manage the data forwarding resources, they do not prevent network congestion, nor provide data transmissions with all desired performance.

Correspondingly, we draw two conclusions. First, a network layer protocol must concern itself with data forwarding resources allocations, and must maintain its own state, the state of

the resources consumed by individual users, in order to provide fair service and to prevent congestion. Keeping such network state is also the key to meet the requirements of new applications, i.e. explicit declaring and securing the amount of resources for specified data transmissions.

Second, the conventional buffer management and window flow control approaches cannot be a substitute for direct monitoring on data forwarding resources. The measurement, allocation, and control of use of these resources should all be done in the unit of rate.

3. Identify New Directions

Two questions arise from the previous section. First, if neither the datagram network's best effort nor VC network's reliable delivery services is desirable, then what kind of transmission services should be considered mandatory for a network protocol to offer? Second, is it feasible to control data flows by rate in the packet switching network? We discuss these two questions below.

3.1. What Services the Network Should Support

To answer this question, we make the following two observations. First, we notice that some types of services (TOS) are in conflict with one another. For instance, guaranteed transmission reliability and finite delivery delay are not compatible, because the network is not perfectly reliable and error recoveries possibly take a long time. This being the case, the network should not take over the former as its only service. Conflicts like this need careful evaluation, in order to avoid the situation where network wired-in functions prevent users from achieving their own desired transmission characteristics on top of the offered network services.

Second, we perceive that, although it is desirable for users to build their needed TOSs at the end-to-end layer[8], not all TOSs can be achieved in this way. Data transit delay, for example, once acquired inside the network, can never be corrected by higher layer protocols. We consider that TOS requirements can be sorted into two categories:

1. those that can be realized with end-to-end protocols, and

2. those that can only be met by support within the network.

[8]For example, inadequacies in the degree of reliability or security offered by the network can be enhanced by end-to-end transport protocols.

It is clear that the network should offer the services belonging to the second category, or otherwise users by no means can have them. With our focus on network resource management, we consider the two important TOSs in the second category are delivery delay and throughput.

The delivery delay of a packet, P, is defined as the time period from its transmission to its reception. If P is lost, its delivery delay becomes infinitely long. The loss can be recovered by retransmissions, but then the delivery delay will cover the time period starting from the time of the first transmission. Delivery delay, like the entropy in thermodynamics, can never be reduced once acquired inside the system.

The throughput of a user data transmission is defined as the number of data units delivered per unit time. Inadequate throughput, as the case of the delivery delay, cannot be improved by higher layer protocols, e.g. splitting one transmission over multiple end-to-end connections does not increase the total throughput, if the network does not provide sufficient transmission bandwidths.

All data transmissions are carried out with certain delay, throughput, and reliability measurements. Since transmission systems are not perfect, enhanced transmission reliability is achieved by forward-error-correction coding and/or retransmissions: the former converts reduced effective throughput to reliability, the latter, a longer delivery delay. That is, transmission reliability above that of the physical system is achieved by converting from delay or throughput. Users can always enhance the reliability at the end-to-end layers. On the other hand, performing this conversion by the network protocol possibly gives a better performance tradeoff (e.g. a node-to-node retransmission takes a shorter time than an end-to-end one). But the conversion is one-direction only, unwanted rigidity of wiring it into the net should be avoided.

We conclude that a network protocol must offer data transmission services that can meet users's delay and throughput requirements; from the performance consideration, it should also offer users an option of a high transmission reliability, with relaxing on the delay or throughput requirements.

3.2. Statistical Multiplexing and Rate Control

Traffic in the packet switching network has been characterized as bursty, i.e. data sources do not generate constant and continuous flows of bits, the bandwidth demand varies from time to time. Nevertheless, packet traffic is not totally haphazard. Individual data transmissions, in most cases if not all, either are under control, or behave with some inherent or identifiable characteristics. An example of the former case is bulk data shipments: when a user requests a transmission, it is the data transport protocol that determines how the shipment, which may require sending thousands of packets, is carried out. An example of the latter is packet voice: a conversation generates packets randomly, due to random talk spurts, but with well known average and upper bound rates.

Given this viewpoint, we consider that network application designers, whenever possible, should design and implement protocols that generate controllable or predictable data flows. The network can then coordinate with data sources on resource management and traffic control, based on these regularities and controllabilities. The better the network knows the demand, the easier and more effective the control will be, the higher the achievable performance, and the higher the achievable resource utilization. On the other hand, the network will not be able to make any performance promise if it does not have any knowledge of the traffic.

We further consider that the packet switching network should control traffic on an average rate basis. The dynamic resource sharing feature distinguishes packet switching from the traditional circuit switching approach, but does not change the nature of the packet switching network as being a transmission system. In a transmission system, the capacity should be measured in rate; the resources should be allocated in rate; the sharing among users should also be controlled by rate. Compared to circuit switching, packet switching requires a change of network resource sharing from the static bandwidth allocation to statistical multiplexing. The key to the rate control on packet traffic is applying knowledge of traffic statistics to estimating the demand average and to monitoring network capacity assignment.[9]

---

[9] An early example of applying rate-based statistical multiplexing is TASI -- time assignment speech interpolation The statistic pattern of telephone conversations has been well studied, showing that active speech signals in each direction occur no more than 40% of the time during a conversation. Applying this knowledge, a TASI system can double the number of conversations handled by the same bandwidths by dynamically assigning transmission bandwidths to those channels which are identified as having useful signals [9, 19].

Most of the previous efforts on network flow control have focused on the window approach, i.e. controlling the amount of outstanding packets. We argue that window is not a proper mechanism for the needed functionalities. For instance, being a data transmitter, what is the highest possible transmission rate that will not cause network congestion? Or can a required throughput be achieved? From the network side, how should the network regulate traffics to keep a low queueing delay inside the net? How should the network decide whether to accept a new transmission request? The answers to these questions cannot be directly found from the window sizes of the connections *alone*. They require that the network closely monitor the *rate* of all data flows. Studies exist that convert the rate flow control information to the window form [2], which simply show that the window mechanism, at the best, is an indirect way to control transmission rates.

The concept of rate control in packet switching has been previously considered. In discussing network flow control, Cerf pointed out that, "It is generally the case that flow control is enforced through the allocation of permits to send packets and the reservation of buffers to receive them. ... In fact, flow control should really be dealt with by metering the rate of flow of packets into the network bound for given destinations. But for asynchronous systems, the measurement and control of rate of flow is very difficult to implement. This is still very much a research topic." [4] With almost twenty years of experience with packet switching, it should be the time now to explore rate control in packet switching networks.

4. Conclusion

Performance control in packet switching networks is one of the remaining areas that are yet to be explored. In this paper we looked into the performance problems with two existing network architectures, datagram and virtual circuit. We conclude that the major problem with the datagram network is lack of effective traffic control; and with the virtual circuit network, the improperly defined network control semantics and mechanisms.

A new network architecture should be investigated. We imagine that the new architecture should maintain a system state of user traffic information and resource allocations at each switch node, to prevent congestion and to provide data transmissions with user specified performance requirements in terms of throughput and delay. The network resource management and user traffic control can both be performed on an average transmission rate basis.

Acknowledgment

This is a status report of on-going research. The Distributed System group at MIT-LCS provides a warm research environment; in particular, I would like to thank Dave Clark for many illuminating discussions.

References

[1] James Atkins.
Path Control - The Network Layer of System Network Architecture.
Computer Network Architecture and Protocols.
Plenum Press, 1982, pages 297-326, Chapter 11.

[2] Dimitri Bertsekas and Robert Gallager.
Flow Control.
Data Networks.
Prentice-Hall, 1986, pages 423-461, Chapter 6.

[3] Robert Braden.
Towards A Transport Service for Transaction Processing Applications.
Network Information Center RFC-955, SRI International.
September, 1985

[4] Vinton G. Cerf.
Packet Communication Technology.
Protocols and Techniques for Data Communication Networks.
Prentics-Hall, Inc., 1981, pages 1-34, Chapter 1.

[5] David Clark.
Some Thought on the DARPA Internet Architecture.
Paper in preparation.
1986

[6] David Clark, Mark Lambert, and Lixia Zhang.
NETBLT: A Bulk Data Transfer Protocol.
Network Information Center RFC-998, SRI International.
March, 1987

[7] A. Danet et al.
The French Publick Packet Switching Service: The Transpac Network.
In *Proceedings of the 3rd International Conference on Computer Comuunications.*
August, 1976.

[8] D. Davies.
The Control of Congestion in Packet Switching Networks.
IEEE Transactions on Communications COM-20(6):546-550, June, 1972.

[9] J. Fraser, D. Bullock, and N. Long.
Overall Characteristics of a TASI System.
Bell System Technical Journal :1439-1454, July, 1962.

[10] Mario Gerla and Leonard Kleinrock.
Flow Control: A Comparative Survey.
IEEE Transactions on Communications COM-28(4), April, 1980.

[11] L. Kleinrock.
A Decade of Network Development.
Journal of Telecommunication Networks , Spring, 1982.

[12] John Nagle.
Congestion Control in TCP/IP Internetworks.
ACM Computer Communications Review 14(4), October, 1984.

[13] J. Postel, Carl Sunshine, and Danny Cohen.
The Arpa Internet Protocol.
Computer Networks 5(1981):261-271, 1981.

[14] J. Postel.
Internet Control Message Protocol.
Network Information Center RFC-792, SRI International.
September, 1981

[15] L. Pouzin, editor.
The Cyclades Computer Network.
North-Holland Publishing Company, 1982.

[16] J. Rinde.
TYMNET: An Alternative to Packet Technology.
In *Proceedings of the 3rd International Conference on Computer Comuunications*, pages 268-273. August, 1976.

[17] Marshall T. Rose.
Comments on "Comments on 'Congestion Control in TCP/IP Internetworks'" or The Holy Wars Begin Again.
ACM Computer Communications Review 15(4), October, 1985.

[18] Jerome Saltzer, David Reed, David Clark.
End-to-End Arguments in System Design.
In *Proceedings of the Second International Conference on Distributed Computing Systems.* IEEE, 1981.

[19] David R. Smith.
Operational Evaluation of a Voice Concentrator Over AUTOVON Interswitch Trunks.
IEEE Transactions on Communications COM-30(4):792-802, April, 1982.

[20] D. Sproule and F. Mellor.
Routing, Flow, and Congestion Control in the Datapac Network.
IEEE Transactions on Communications COM-29(4), April, 1981.

[21] Stuart Wecker.
DNA - The Digital Network Architecture.
Computer Network Architecture and Protocols.
Plenum Press, 1982, pages 249-296, Chapter 10.