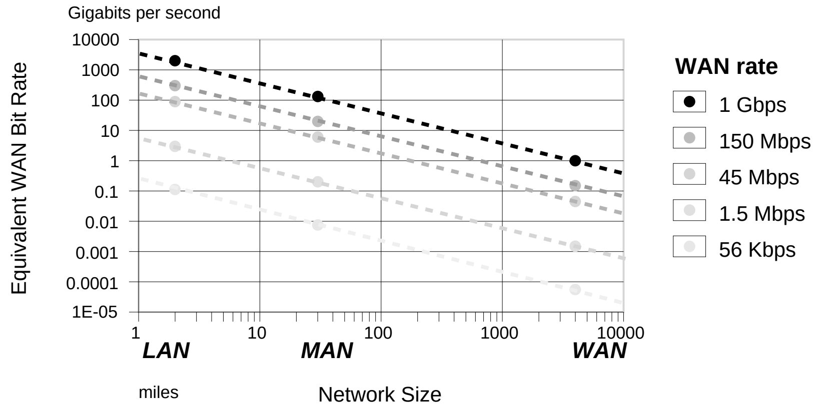
Reducing Latency in Communication



♦**Figure 1:** *Network rate and size equivalences*[1]

To the Editors:

We would like to address a few points in Prof. Kleinrock's "The Latency/Bandwidth Tradeoff in Gigabit Networks," in your April 1992 issue [1]. We appreciate his efforts to sensitize the readership to the main issues of gigabit networks, and would like to augment his discussion.

Prof. Kleinrock describes the well-known implications of the finite speed-of-light latency on communications. He also observes, peripherally but interestingly, that process parallelism may provide one solution to the latency-induced gaps in communication.

We developed a model which describes the effects of latency on communication, in which to evaluate such proposals [2]. Our work is based on the existing mechanism of having the sender model the receiver state. We added a mechanism of sender-based anticipation, in which *all possible next replies* are sent to the receiver before being requested. Using this latter mechanism, we can describe the conditions under which latency can be alleviated (and the extent thereto).

We feel that mechanisms for latency compensation are the most important issue in gigabit network protocol research. Although process parallelism was not proposed as a unique solution to latency, we would like to further examine it briefly, to demonstrate its effects and describe a more effective solution.

Prof. Kleinrock's discussion contains two issues of interest to us: that messages or files should be larger than the bandwidth-delay product to prevent latency-bound communication, and that parallelism can help abate latency-induced inefficiencies. Our research indicates that inefficiency in networks with high bit-latencies can be overcome with branching streams of messages [2].

In his discussion, first packet sizes, then message sizes, and finally file sizes are used to show the limits of a high bandwidth-delay product channel. We treat all these data streams as linear sequences, each with specific limitations on the duration (length) of the linearity. Most existing protocols are optimized for communicating linear streams of information, but their performance degrades sharply when the linearity length is short compared to the latency.

We claim that two things have changed that make gigabit wide area network protocols different from their slower or more proximal counterparts: an increase in the bandwidth-delay product, and a narrowing of the gap between WAN characteristics and workstation memory capacity. The increase in bandwidth-delay product reflects the effects of speed increases made to bring network implementations more up-to-date. Figure 1 indicates that gigabit LAN experiments do not scale to the WAN case; only terabit LANs have gigabit WAN implications, in terms of equivalent bandwidth-delay product. Gigabit LANs are important in developing protocol-independent hardware and interface issues, but not for the development of scalable protocols. *Protocol* and *control methods* do not scale as *latency* increases.

The second characteristic of gigabit WANs is the bandwidth-delay product compared to the average size of the network computers (nodes within the network, or user-access points). There is a phase change when the bandwidth-delay product exceeds the determinism of state and capability of deterministic messages to fill the pipe. In this case, the bandwidth-delay product nearly as large as the node size, but messages are always smaller than the node size (a node doesn't request a reply that is larger than its available buffer space), so messages are smaller than the bandwidth-delay product. Figure 2 shows that we are only now designing general networks with bit-latencies on the order of the entire buffer space of a node[2].

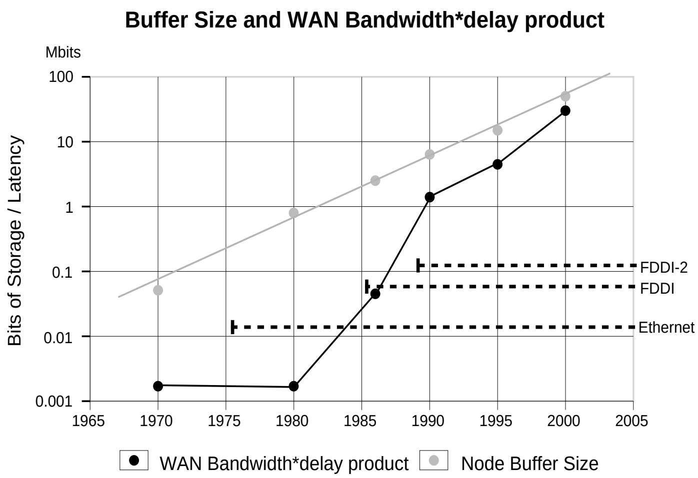
Prof. Kleinrock also indicates that closed-loop flow control is too sluggish, and that rate-based flow control is a more reactive method. Any feedback mechanism is limited by the round trip of feedback information; sluggishness is a property of this delayed feedback. Conventional flow control is coarse in its modeling of remote state, in that send and receive windows are moved *en masse*, rather than in fine increments according to their time-constrained behavior. Rate-based flow control includes this fine modeling of state, in-between the states of conventional flow control, to provide a smoother (but not faster-reacting) mechanism.

Prof. Kleinrock's paper states that we need to hide latency at the application layer, and that one possible mechanism for doing so uses process parallelism. Deterministic process parallelism reduces to channel sharing, which is admittedly user-equivalent to existing protocols and networks [2]. Nondeterministic process parallelism requires complex process group management (equivalent to a protocol) that governs the individual process protocols. We agree that 'set' information is the solution to latency issues, but we achieve it by modeling the remote state as a set of states, and explicitly identifying the need to control the set of states. In these domains we cannot rely on a global state to be maintained in a consistent and timely fashion [1]. When we relax this global state by maintaining sets of possible remote states, we can model more accurately the phenomenon induced by high bandwidth-delay product.

We prefer the notion of *communication parallelism* to process parallelism [3]. Communication parallelism results from the grouping of processes or modeling

[1]*Assuming no topological differences exist, or that they are important for routing, but not protocol operation.*

[2]*Satellite networks also have this characteristic, but combined with unique topological constraints.*



♦ **Figure 2:** *Node buffer size[3] (memory) vs. information separation (BW-delay product)*

branching streams of data. Getting around latency requires managing the indeterminism of state, whether via a set of processes in multitasking [1], or a set of states [2]. The key is the group dynamics of the set of states of a nondeterministic model of remote state.

According to our model for latency in communication, which we call *Mirage*, we model the stream of communication as a branching stream of messages [4, 2]. The branching stream is characterized by the length of the branch arms (expected stream linearities), and branch degree (degree of splitting of the stream, i.e., a measure of indeterminism induced by latency). Prof. Kleinrock's parameter '*a*' is similar to our branch arm length, and our branch arm degree is 2.

Using these parameters, we develop a model of potential stream utilization. This model uses guarded messages, or messages that account for the imprecision in remote state. A message is emitted with a condition, such that the receiver accepts the message only if its state matches that specified by the guard. In our method, the sender emits a set of all possible required messages, in anticipation of their request by the receiver.

There are several implications of this message set use [2]. First, anticipation is indicated as an effective method for alleviating latency effects. Second, anticipation compensates logarithmically for the imprecision induced by latency. Third, the sender needs an effective model of the receiver's actions in order to anticipate its needs and send an appropriate set of guarded messages.

The modeling of the remote state as accomplished by this set method is similar to the effects of managing a set of parallel processes, as presented in the paper. Anticipation, coupled with imprecision of state, is the solution in this case.

The modeling of the receiver state by the sender indicates that application layer state may be needed to effect anticipation. This indicates that layering is harmful, because it hides this state evolution information from the protocol. This information is required to effectively manage remote states in the presence of latency.

We summarize our observations by addressing Prof. Kleinrock's conclusions, paraphrased:

1) Gigabit rates force us to deal with finite latency

Only because node sizes don't similarly scale, and existing protocols weren't designed to accommodate state imprecision.

2) The user needs to pay attention to file sizes.

The user needs to pay attention to linearity limitations in his system state, whether due to sequential files of finite size, or other inhibitions on branch modeling.

3) One way to deal with latency at the application layer uses pipelining and parallelism.

The pipe is too large to do linear anticipation alone, and pipelining doesn't help initial latency effects, which are the problem. Parallelism helps only where it is communication parallelism, not process parallelism. Process parallelism utilization requires knowledge of the set of processes, resulting in effective communication parallelism. Our conclusions are that pipelining and parallelism cannot accommodate latency, and that sending sets of messages in anticipation might.

4) flow control/buffering/congestion control are central issues

We agree, but only because they are already 'broken', i.e., they model the receiver as a single state, rather than as a set of indeterminate states.

5) We cannot depend on a consistent global state to be maintained in a timely fashion...

We agree.

5 a)...because it affects flow control

State imprecision affects all mechanisms, including flow control. Application-layer information is required to deal with state imprecision, violating the principle of layering, and requiring new approaches to protocol design.

Joseph D. Touch,
Computer Scientist,
USC / Information Sciences Institute

David J. Farber,
Professor and Director,
Distributed Systems Lab,
University of Pennsylvania

[3]*Node buffer sizes are represented by Internet router mini/microcomputers. The rate of growth (4x every 3 years) and decrease in comparison to bandwidth-delay product is more important than the actual values.*

References

[1] Kleinrock, L., "The Latency/Bandwidth Tradeoff in Gigabit Networks," *IEEE Communications Magazine*, Vol. 30, No. 4, 36-40, April 1992.

[2] Touch, J.D., *Mirage: A Model for Latency in Communication*, Ph.D. dissertation, Dept of Computer and Information Science, University of Pennsylvania, Jan. 1992., Available as a Univ. of Penn. Dept. of Comp. and Info. Science Tech. Report, MS-CIS-92-42/DSL-11.

[3] Touch, J.D., "Parallel Communication," submitted to IEEE Infocom '93.

[4] Touch, J.D. and Farber, D.J., "Mirage: A Model for Ultra-High-Speed Protocol Analysis and Design," in *Protocols for High-Speed Networks*, Rudin, H. and Williamson, R., IFIP, North-Holland, 1989, pp. 115-134, Available as a Univ. of Penn. Dept. of Comp. and Info. Science Tech. Report M3S-CIS-89-79 / DSL-1.