Appeared in: Proceedings of the Joint Workshop on Parallel and Distributed Real-Time Systems,
Santa Barbara, California, April 24-26, 1995.

Real-Time Networking over HIPPI

Riccardo Bettati [*] *Anisoara Nica*

The Tenet Group
University of California at Berkeley
and
International Computer Science Institute

Abstract

HIPPI provides a very-high-speed communication medium, which is very well suited for a large number of bandwidth-demanding distributed applications. Unfortunately, its circuit-switched nature makes it very difficult to provide real-time guarantees when connections contend for network resources. We present a time-division-multiplex access scheme designed to give timing guarantees to high-speed connections. We describe the problem of scheduling the access to a HIPPI network, and show that, although the problem is very unlikely to be computationally tractable, very simple heuristics give high network utilizations for moderately-sized networks. We present the RMP/RMCP protocol, our implementation of the scheme described in this paper on the XUNET-West HIPPI testbed.

1 Introduction

A large number of applications in distributed control, distributed virtual reality, and remote laboratoring demand for hard delay guarantees in order to satisfy the timing requirements of their time-critical components. Such applications typically consist of long sequences of computation-intensive operations on large amounts of data, such as video, to be performed by special-purpose, remotely located, machines. The bandwidth requirements are very high because of the large amounts of data that are involved and the limited applicability of data compression because of its high computational cost.

A remote microscopy application, for example, may involve (1) an electrone microscope, which is equipped with a camera, (2) a supercomputer for image enhancement and focus control, and (3) a workstation with the human operator. The camera in the microscope captures the picture of the sample under observation and sends it as a stream of video images to the supercomputer. The latter performs the necessary image enhancements and forward the images to the workstation with the operator. It also computes the necessary control inputs to keep the sample in focus, and returns the results to the microscope. The operator observes the enhanced video sequence forwarded by the supercomputer, and guides the experiment by issuing control requests to the supercomputer (tuning of the image enhancement) and to the microscope (positioning of the sample). The underlying communication media must satisfy two criteria. First, the transmission of the video images requires very-high bandwidth connections. Second, the time-critical components of the system, such as the focus control loop, require guarantees by the network on transmission delay bounds.

The bandwidth requirements of such applications are expected to be in the hundreds of megabits per second, and therefore exceed the bandwidth offered by existing cell-switched technologies (e.g. 155Mb/s ATM). We have investigated the applicability of circuit-switched network technologies, in particular High-Performance Parallel Interface (HIPPI) [1, 2] for this class of applications. HIPPI is a simplex point-to-point interface for transferring data at very high rates (800Mb/s or 1600Mb/s) over limited distances (up to 25 meters). HIPPI physical layer switches can be used to interconnect multiple devices into a local area network [3]. Serial fiber extenders make it possible to connect devices over distances of up to 10 kilometers [4].

Once a HIPPI connection has been established, the full peak rate of 800 Mb/s is available end-to-end. In order to bound connection setup delays due to contention for network resources, however, proper con-

---

[*] Address: International Computer Science Institute, 1947 Center Street, Suite 600, Berkeley, CA 94704, USA. E-Mail: bettati@icsi.berkeley.edu. Tel: (510)643-9153. Fax: (510)643-7684

nection management strategies must be devised. The HIPPI standard does not address the issue of how to manage connection setup. Chlamtac et al. [5] analyze and compare connection management schemes for clusters of crossbar switches with respect to connection delay and throughput. The described approaches fail to guarantee either bounds for connection delay or bandwidth allocation.

In this paper we describe a connection establishment protocol for HIPPI networks that addresses the issues of time-constrained setup of guaranteed-bandwidth connections.

2 Scheduling of Real-Time Connections in Circuit-Switched Networks

A HIPPI network consists of a number of crossbar switches and hosts, connected by unidirectional links. Links connecting two switches are called *trunks*, whereas links connecting hosts to a switch are called *ports*. In order to setup a (simplex) *HIPPI connection*, the client at the source first acquires the port connecting the source to the network (the *source port*). Then it sequentially acquires the trunks along a path from the source port to the port connecting the network to the destination (the *destination port*). In the end the destination port must be acquired, and the connection is set up. A duplex connection consists of a pair of two independent connections in opposite directions. When all resources (i.e. ports and trunks) are available, the connection setup takes one round-trip delay between the source port and the destination port. If, however, a resource is used by another connection, the user can choose to either abort the setup process or to block until the resource becomes available, using the so-called *camp-on* feature of HIPPI.

This scheme can lead to deadlocks and to unbounded connection setup delays. The HIPPI standard does not provide a way to avoid these problems. Nevertheless, some recommendations are made about how to reduce connection setup delays [6]. In particular, these recommendations discourage from keeping connections too long (in order to reduce blocking times) and from camping on during connection setup. Although avoiding camping on eliminates the possibility of deadlocks, an additional protocol is necessary to avoid unbounded connection setup delays.

Channel acceptance protocols for circuit-switched networks must rely on a global scheduling scheme of the network resources (as opposed to local schemes used in packet-switched networks [7]). Whenever a connection is being set up, all necessary resources have to be available at that particular point in time for the setup to get through.

Typically, global resource access schemes rely on collision resolution, tokens, or time division multiplex. Traditional collision resolution schemes, such as tree-based resolution [8], are not applicable for HIPPI-style high-speed networks because of the high amount of bandwidth lost due to the resolution process.

Token-based approaches are not practicable as well because of two reasons: First, it can be shown that having tokens rotating in-band (i.e. competing for the same network resources as the data-transmission) would cause tokens to queue up and thus the scheme to degenerate into a single-token scenario and, hence, to a serialization of the access to the network. This would mean that only one connection could be set up at any particular point in time [1] . Second, token-based approaches do not, *per se*, guarantee timing constraints, even for very simple systems. Additional schemes, such as *timed tokens* in FDDI, must be used to give timing guarantees. Such schemes substantially reduce the maximum utilization of the network resources by real-time traffic, e.g. by two thirds for the case of timed tokens in a ring [9].

Time-division multiplexing (TDM), on the other hand, avoids resource contention by explicitely laying out the transmission schedule. This obviates the need for expensive collision resolution schemes and results in a high utilization of the network because it allows for high concurrency of access to the network resources. We will describe in the following how we apply this scheme for the particular case of real-time communication in a HIPPI network.

3 Time Division Multiplex: Allowing Multiple Real-Time Connections on HIPPI

We define the set of requested workload $\mathcal{C}$ to consist of the *real-time channels* $\{C_1, C_2, \ldots, C_N\}$. Each real-time channel C_i (in the following simply called *channel*) is a simplex connection with guaranteed quality of service and is characterized by the following tuple:

$$C_i = (S_i, D_i, p_i, \tau_i),$$

where the components are defined as follows:

---

[1] The serialization effect could be avoided by having the tokens rotate out-of-band, either by using a separate network for the token traffic or by allocating a portion of the HIPPI bandwidth for that purpose. The second solution would mean that one would have to multiplex the network between data traffic and token-passing. This would cause many of the same problems as in the TDM approach described below.

S_i: Source host of channel C_i.

D_i: Destination host of channel C_i.

p_i: Worst-case interarrival time of messages on channel C_i.

τ_i: Maximum length of messages on channel C_i.

We follow a guaranteed-bandwidth allocation model for quality of service (QOS) and do not take into consideration traffic burstiness because of two reasons: First, letting the application handle traffic bursts (for example with appropriate rate control) simplifies the mechanisms to give guarantees at network level considerably, in particular for circuit-switched architectures. Second, the class of applications considered here generates mainly continuous data streams with little or no compression and thus with small amounts of burstiness. Even if the underlying net