Transport-level Protocol Coordination in Distributed Multimedia Applications

David E. Ott and Ketan Mayer-Patel (Advisor)
Department of Computer Science
University of North Carolina at Chapel Hill

Categories and Subject Descriptors

C.2.2 [**Computer Communication Networks**]: Network Protocols—*applications*.

General Terms

Design, Algorithms, Performance, Experimentation.

1. INTRODUCTION

Future Internet multimedia applications will become increasingly complex in their networking needs, requiring more and more streams to handle an ever-growing number of media types and modes of interactivity. Where the endpoints of communication were once single computing hosts, future endpoints will be collections of communication and computing devices. Examples of such applications include distributed sensor arrays, tele-immersion, computer-supported collaborative workspaces, ubiquitous computing environments, and complex multi-stream, multimedia presentations.

We are interested in a class of distributed multimedia applications called *Cluster-to-Cluster (C-to-C) applications*. In a C-to-C application, a collection of computing and communication devices communicates with a remote collection of computing and communication devices. *Endpoints* in the same cluster share a common gateway node known as the *Aggregation Point (AP)*. While few application flows share the same end-to-end path, all flows share a common intermediary path between clusters. Figure 1 illustrates this model.

C-to-C applications share several key characteristics.

- *Independent, but semantically related flows of data.* An application may prioritize streams in a particular way, or divide complex media objects into multiple streams with specific temporal or spatial relationships.
- *Transport-level heterogeneity.* UDP- or RTP-based protocols, for example, might be used for streaming media while TCP is used to reliably transport control data.
- *Changing network conditions along the shared data path.* While networks *within* a cluster can be provisioned to comfortably support an application's requirements, the forwarding path *between* clusters is shared with other Internet flows and typically cannot be provisioned end-to-end. Hence, it is the primary source of network latency and packet loss.

One example of a C-to-C application is *Office of the Future*, conceived by Fuchs et al.[5] In this immersive application, a set of video cameras and microphones are used to capture an office environment in order to construct a remote virtual environment using various computing and display devices. Complex spatial relationships exist between media streams, and the relative priority of these streams changes constantly as the user moves their

Copyright is held by the author/owner.
MM'03, November 2–8, 2003, Berkeley, California, USA..
ACM 1-58113-722-2/03/0011.

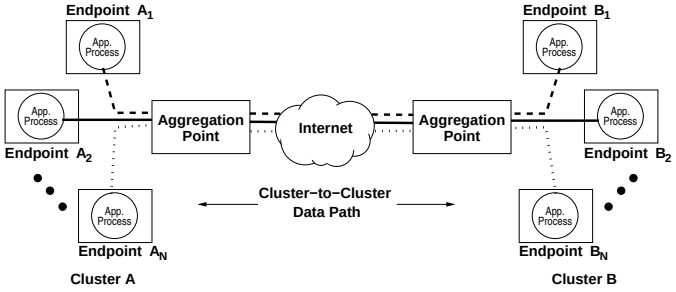

Figure 1: C-to-C application model.

region of interest. While few streams share a complete end-to-end path, all of the data streams (and control streams) share a common Internet path between media capture and reconstruction/display clusters.

A fundamental problem in the C-to-C application context is that of *flow coordination*. Application streams share a common intermediary path between clusters, and yet employ transport protocols that *operate in isolation from one another*. As a result, flows may compete with one another when network resources become limited instead of cooperating to use available bandwidth in application-controlled ways.

In *Office of the Future*, for example, media streams may compete with one another for bandwidth during periods of network congestion, unaware of inter-stream prioritization within the application or the effect on control message transmission. As a result, the performance of high priority streams may be unacceptable. Control flows using TCP reduce their window size in response to congestion. In the larger context of the C-to-C application, however, it makes more sense to have low priority media streams interrupt transmission to allow control flows *more* bandwidth to transmit crucial adaptation information.

In this dissertation, we argue that cluster-to-cluster applications like *Office of the Future* could significantly benefit from network mechanisms that facilitate transport-level protocol coordination of separate but semantically-related data flows along a shared intermediary transmission path. Such mechanisms could provide application endpoints with

- *Network awareness.* Endpoints, regardless of their transport-level protocol, are given a consistent view of network conditions, including round trip time, packet loss rates, and bandwidth available to the application as a whole.
- *Peer flow awareness.* Endpoints are given information on the number of peer flows transmitting across the shared data path, patterns of bandwidth usage, as well as aggregate bandwidth usage for the C-to-C application.

With this information, a C-to-C application can use smart transport-layer protocols to implement a coordination scheme uniquely suited to its needs. For example, *Office of the Future* might implement a *dynamic inter-stream prioritization* scheme to give media streams within a user's changing region of interest

a larger share of available bandwidth during periods of congestion, thus maintaining an acceptable frame rate and display resolution.

2. COORDINATION PROTOCOL

Our solution to the problem of flow coordination in C-to-C applications is called the *Coordination Protocol*, or *CP*. CP operates between the network layer (IP) and transport layer (TCP, UDP, etc.), making it transparent to IP routers on the C-to-C forwarding path and preserving the semantics of end-to-end transport-level protocols.

Application endpoints insert a CP header into each data packet before sending the packet to an endpoint on the remote cluster. The packet receives special handling at its local AP, and then is forwarded toward the remote cluster. After traversing the cluster-to-cluster data path using standard IP forwarding, the remote AP applies special handling to the packet before forwarding it to the destination endpoint. Endpoints use information in CP packet headers to make send rate adjustments that reflect application coordination strategies.

As a packet originates from an application endpoint, its CP header contains a cluster ID telling the AP to associate it with a particular C-to-C application. A flow ID likewise identifies the packet with a particular application flow. The AP keeps a table of bandwidth usage statistics on flows in the same C-to-C application, also tracking the number of flows and aggregate bandwidth usage by the C-to-C application as a whole.

Network probing works by using the CP header in each C-to-C data packet to piggyback probe information on the shared data path between APs. Each AP modifies the CP header from packets originating at its local cluster to add timestamp and sequence number information. As additional probe information from the remote AP is returned along the reverse cluster-to-cluster data path, the AP is able to obtain measurements of round trip time and loss rates across all aggregate C-to-C traffic.

Using round trip time, loss rate, and packet size information, a bandwidth availability estimate can be made at each AP using a throughput modeling equation. Our work has made use of the TFRC [2] equation, giving a throughput estimate that is both gradually responsive to network congestion and TCP-compatible. In addition, we have developed techniques to extend single-flow modeling equations to multiple flowshares, thus allowing m C-to-C application flows to receive the equivalent of m TCP-compatible flowshares.

Peer flow and network condition information is passed to each endpoint again using the CP header. An AP will write information into the header before an incoming packet from the remote cluster is forwarded to its local destination endpoint. A consistent view of network conditions across flows follows from the fact that the same information is shared among all endpoints.

Transport-level protocols at application endpoints are built on top of CP. Using information on aggregate bandwidth availability, loss rate, round trip time, number of application flows, etc., as well as various application configuration parameters, the transport protocol can choose an appropriate sending rate that reflects an application's global coordination strategies. Information on peer flows and network conditions is also made available to the application layer directly, allowing it to modify data encoding parameters or perform others types of adaptive behavior.

While CP provides the essential building blocks to enable C-to-C flow coordination, the implementation of a particular *flow coordination scheme* is left to the application. This is necessarily the case since it alone knows the nature and function of various data flows, the semantic relationships between them, and how best to use limited bandwidth during any given time interval.

Office of the Future, for example, may configure secondary streaming endpoints to reduce their sending rate, or stop sending altogether, in response to a drop in available bandwidth below a particular threshold value. At the same time, a primary stream endpoint may continue to send at its original rate, and a control endpoint may increase its sending rate somewhat in order to transmit essential control information telling the receive side how to respond to the change. Despite these differences in response behavior, the aggregate bandwidth usage drops appropriately to match a reduction in bandwidth availability.

3. RELATED WORK

Several approaches to congestion control for flow aggregates have been proposed (e.g., [3], [4]). The most relevant to this dissertation, however, is that of the *Congestion Manager (CM)*, proposed by Balakrishnan et al. in [1]. While the CM architecture proposes many useful concepts and mechanisms for managing congestion control in flow aggregates, we believe that it is not a good match for the C-to-C problem context. First, the CM sender module was designed to run on a single host using a system interface that tightly couples CM flow scheduling and application send events. Translating this interface to a message-passing interface between multiple endpoints and a gateway host is at best problematic. Second, CM's use of a flow scheduler to apportion bandwidth among flows works when flows can be well-characterized and maintain a relatively static priority level. (A situation more likely to occur when flows are not part of the same application.) A C-to-C application like *Office of the Future*, in contrast, changes flow priority and bandwidth usage needs in a complex and dynamic manner. It's not clear that a gateway scheduling node could be continually reconfigured on the fly for similar results.

4. CONTRIBUTIONS

- We identify an important class of Internet multimedia applications called *Cluster-to-Cluster (C-to-C) applications* and describe their unique networking requirements.
- We define the *flow coordination problem*. This problem is fundamental to C-to-C applications, but also of interest to multiflow Internet applications generally.
- We propose a novel architecture, called the *Coordination Protocol (CP)*, that collects information on C-to-C application flows and network conditions in order to make endpoint transport protocols *network-* and *peer flow-aware*.
- We show how CP can be used to implement application-specific coordination schemes that solve the flow coordination problem in flexible ways. These schemes often find it useful to decouple *individual flow behavior* and *aggregate congestion responsiveness*, a feature of our architecture.

5. REFERENCES

[1] H. Balakrishnan, H.S. Rahul, and S. Seshan. An Integrated Congestion Management Architecture for Internet Hosts. *Proc. of ACM SIGCOMM*, Sept. 1999.

[2] M. Handley, S. Floyd, J. Padhye, and J. Widmer. *RFC 3448: TCP Friendly Rate Control (TFRC): Protocol Specification*. IETF, Jan. 2003.

[3] H.T. Kung and S.Y. Wang. TCP Trunking: Design, Implementation and Performance. *Proc. of ICNP*, 1999.

[4] P. Pradhan, T. Chiueh, and A. Neogi. Aggregate TCP congestion control using multiple network probing. *Proc. of IEEE ICDCS*, 2000.

[5] R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, and H. Fuchs. The Office of the Future: A Unified Approach to Image-Based Modeling and Spatially Immersive Displays. *Proc. of ACM SIGGRAPH*, 1998.