TRAFFIC AND CONGESTION CONTROL IN ATM NETWORKS

By Major Ehsan Ul Haq

CHAPTER 1

INTRODUCTION

ATM (Asynchronous Transfer Mode) is an evolving technology designed for the high-speed transfer of voice, video, and data through public and private networks in a cost-effective manner. **ATM** is based on the efforts of Study Group XVIII of the International Telecommunication Union Telecommunication Standardization Sector[1] (ITU-T, formerly the Consultative Committee for International Telegraph and Telephone [CCITT]) and the American National Standards Institute (ANSI) to apply very large-scale integration (VLSI) technology to the transfer of data within public networks. Before proceeding, it should be noted that specifications for **ATM** have been developed by both ITU-T standardization body and the **ATM** Forum[1]. ITU-T is primarily concerned with developing **ATM** standards as part of the broadband ISDN (B-ISDN) standardization effort, whereas the **ATM** Forum is interested in a broad range of **ATM** applications.

These days, most of the computing power resides on desktop PCs, and that power is constantly increasing [1]. The new classes of applications specially distributed applications require more bandwidth, and the Internet traffic is driving most LAN architectures to the limit.

Historically, LAN and WAN communications have remained different. For LANs, bandwidth is highly available. LANs are faster and carry usually data only. For WANs, bandwidth is costly and delay-sensitive traffic such as voice has remained separate from data. This is because voice requires predictable delay, which is usually not guaranteed in datagram networks. These conventions, however, are changing. The Internet has been the

[1] URL: http://www.itu.ch/ITU-T/

main source of multimedia to the desktop and mixes data, voice and video traffic thus breaking the rules. Such Internet applications as voice and real-time video require better, more predictable LAN and WAN performance. Moreover, since part of the Internet traffic travels over LANs and WANs, it is important that the WAN recognizes LAN traffic thus driving LAN/WAN integration.

ATM has emerged as one of the technologies for integrating LANs and WANs. **ATM** can support any traffic type in separate or mixed streams, delay-sensitive traffic, and non delay-sensitive traffic. **ATM** is a streamlined protocol with minimal error and flow control capabilities. This reduces the overhead of processing **ATM** cells and reduces the number of overhead bits required with each cell, thus enabling **ATM** to operate at high data rates. Further, the use of fixed-size cells simplifies the processing required at each **ATM** node, again supporting the use of **ATM** at high data rates. In sum, **ATM** is an efficient technology to integrate LANs and WANs as well as to combine multiple networks into one multi-service network.

1.1 Thesis Overview

The rest of this thesis is organized as follows. Chapter 2 provides a quick overview on **ATM**. Chapter 3 gives a fairly detailed description of the addressing scheme adopted in **ATM** technology and the related signaling and routing protocols used to setup a virtual circuit. In chapter 4, the **ATM** Adaptation Layer is discussed. Chapter 5 helps in understanding the basics of traffic and congestion control. Chapter 6 explains the recent work done to extend **ATM** capabilities in this field and examines the different promising proposal of rate based congestion control schemes and their limitation. Chapter 7 proposes a refined congestion control scheme and compares the simulation results with previous congestion control schemes. Moreover, a software has also been developed in order to examine the performance of new congestion control algorithm in comparison

[1] URL: http://www.atmforum.com/

with previous congestion control algorithm. This software also calculates the current cell rate and number of cell loss.

CHAPTER 2

ATM OVERVIEW

It is clear that **ATM** technology will play a central role in the evolution of current workgroup, campus and enterprise networks. **ATM** delivers important advantages over existing LAN and WAN technologies: the promise of scalable bandwidths and performance points and Quality of Service (QoS) guarantees, which facilitate new classes of applications such as multimedia including real-time audio and video as well as traditional data communication.

These benefits, however, come at a price. Contrary to common misconceptions, **ATM** is a very complex technology, perhaps the most complex ever developed by the networking industry. While the structure of **ATM** cells and cell switching do facilitate the development of hardware intensive, high performance **ATM** switches, the deployment of **ATM** networks requires the overlay of a highly complex, software intensive, protocol infrastructure. This infrastructure is required to allow both individual **ATM** switches to be linked into a network and for such networks to internetwork with the vast installed base of existing local and wide area networks.

2.1 ATM Network Operation

The basic component of an **ATM** network is a switch capable of transferring data at very high speeds[1]. Briefly, like an Ethernet NIC, a host interface board (equipped with circuitry to transmit and receive pulses of light) is plugged into the computer bus. A

[1] Currently up to 622 Mbps.

computer attaches to a switch through a pair of optical fibers[1] making data communication at high speeds possible. Then the **ATM** switches are interconnected by point-to-point **ATM** links or interfaces. An **ATM** switch supports two kinds of interfaces: user-network interface (UNI) and network-node[1] interface (NNI). A user network interface (UNI) connects **ATM** end-systems (like hosts, routers, servers, etc) to an **ATM** switch, while an NNI may be defined as an interface connecting two **ATM** switches or more precisely any physical or logical link across which two **ATM** switches exchange the NNI protocol. UNI and NNI have slightly different cell formats, as we will see later. To attached computers, the set of interconnected **ATM** switches appears to be a homogeneous network. So, like a bridged Ethernet, **ATM** hides the details of physical hardware and gives the abstraction of a single physical network as shown in Figure 1.

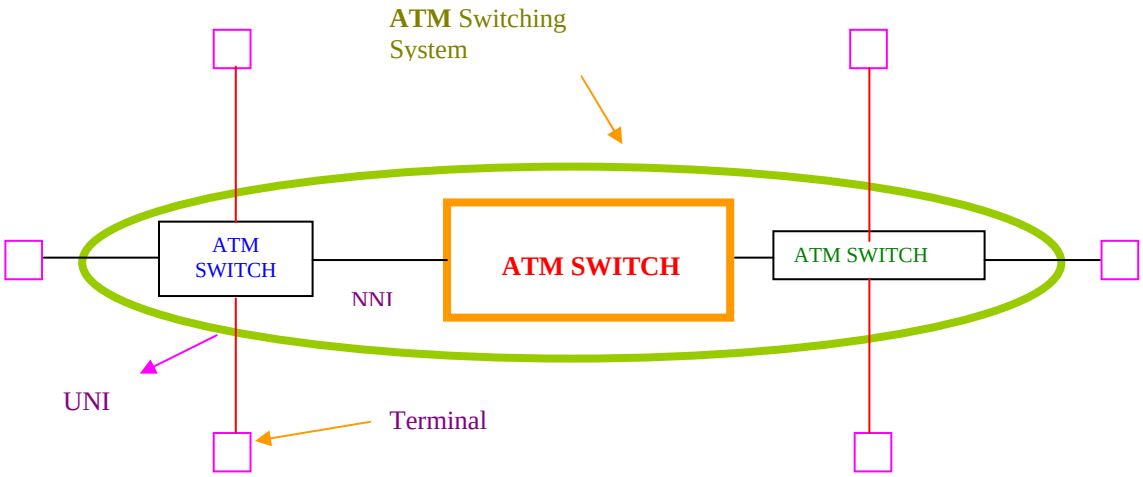


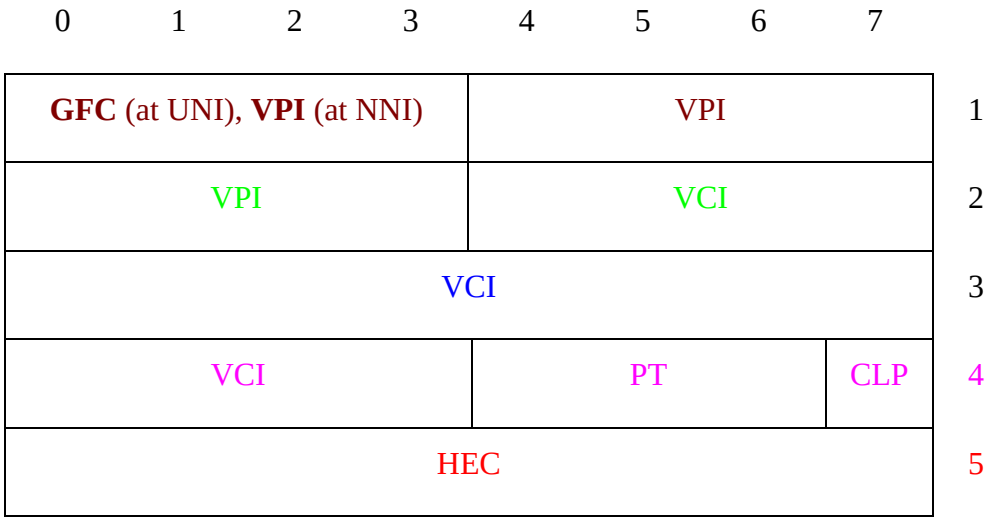
Figure 1. The logical view of a set of three ATM switches

At the lowest level, data is carried over **ATM** networks using fixed-size frames called *cells*. Each **ATM** cell is 53 octets long consisting of a 5-octet header and 48-octet data field (see Figure 2). **ATM** requires all cells to be of the same size because doing so makes it possible to build faster switching hardware [2]. In addition, the designers of

[1] Because a fiber can carry light in one direction only.

ATM chose to use small size format to minimize the processing time of each cell. Both fast switching and small processing time would lead to small delay and minimal delay variation, being two critical requirements to successfully transmit audio and video traffic[2] (see [3] for a good comparison).

ATM networks are fundamentally connection oriented. This implies that if two parties want to communicate, a virtual circuit[3] (VC) needs to be set up across the **ATM** network prior to any data transfer. The operation of setting up a VC is done through signaling protocols that will be covered in fair detail later.



GFC = Generic Flow Control PT = Payload Type

VPI = Virtual Path Identifier CLP = Cell Loss Priority

VCI = Virtual Channel Identifier HEC = Header Error Control

---

[1] Sometimes called network-network interface; the difference is unimportant.
[2] Note that this reasoning is not true for all types of data; namely it is inappropriate for traditional data traffic.
[3] As opposed to a circuit, because although, like a circuit, a fixed path is used all the time by a given VC, the capacity of each link is shared by the VC's using that link on a demand basis (statistical multiplexing) rather than by fixed allocations.

Figure 2. Format of the ATM cell header (note that the GFC is used only at the UNI)

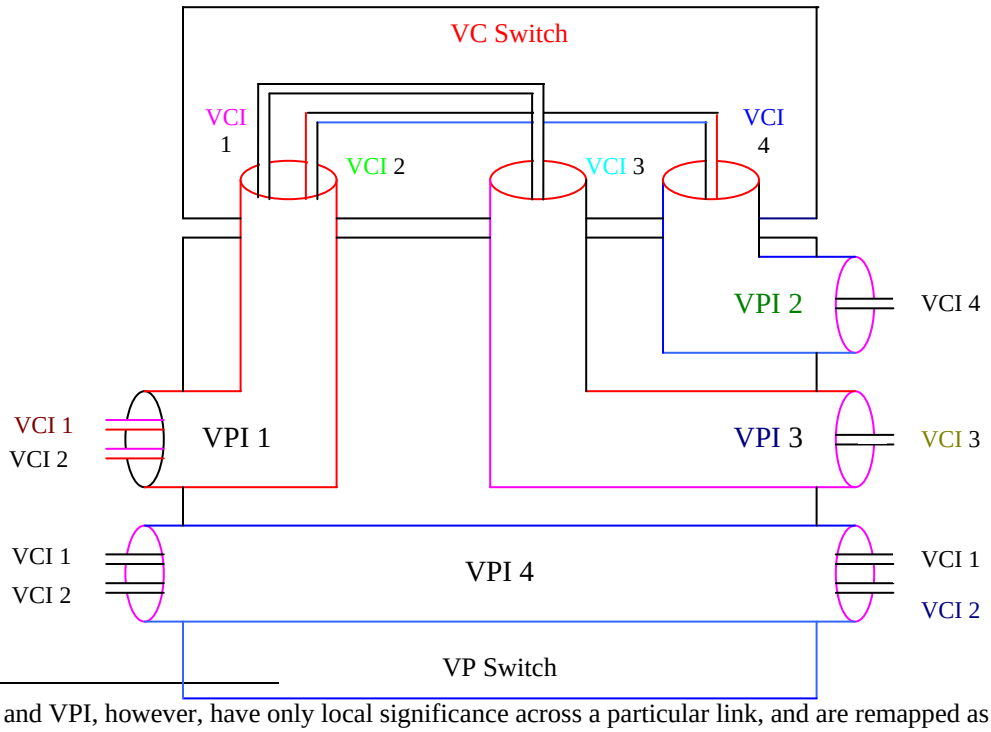
The way a VC is set up determines the two fundamental types of **ATM** connections, namely permanent virtual connections (PVC) and switched virtual connections (SVC). A PVC requires manual configuration usually through network management. An administrator specifies the source and destination of the connection, the quality of service the connection will receive, and the identifiers each host uses to access the connection. He also programs the switches between the source and destination end-systems with the appropriate identifiers to the connection. Although the use of SVCs is usually more convenient (as we will see), PVCs are especially important for network maintenance and debugging.

An SVC is similar in a way to a conventional telephone call. It is set up automatically through a signaling protocol where a host asks its local switch to establish a connection to a given destination. The switch then takes over and creates that connection. SVCs do not require any manual interaction, and thus are likely to be more widely used than PVCs. Actually, all higher layer protocols (like IP) operating over **ATM** primarily use SVCs.

ATM circuits are of two types: virtual paths (VPs), identified by virtual paths identifiers (VPI); and virtual channels (VCs), identified by the combination of a VPI and a virtual channel identifier (VCI). The idea of a virtual path (VP) is valuable in the overall design of **ATM** networks. A virtual path has a function similar to that of a postal sack. In the same way that a postal sack helps to ease the handling of all letters which share a similar destination, so a virtual path helps to ease the workload of the **ATM** network nodes by enabling them to handle bundled groups of virtual channels. Therefore, a virtual path is a bundle of virtual channels, all of which are switched transparently

across the **ATM** network based on the common VPI[1]. All virtual circuits in an **ATM** network (be they VCs or VPs) can be classified according to whether they are PVCs or SVCs[2].

A full **ATM** switch is the most complex and powerful of the elements making up an **ATM** network. It is capable not only of cross-connecting virtual paths, but also of sorting and switching their contents, the virtual channels. It is the only type of **ATM** nodes capable of interpreting and reacting upon user or network signaling for the establishment of new connections or the clearing of existing connections. The basic operation of an **ATM** switch, however, is very simple. It receives a cell on a link with a VCI/VPI pair value (stored in the header), looks up the value in a local translation table to determine the outgoing port(s) of the connection and the new VPI/VCI value of the connection on that link, and finally retransmits the cell on the outgoing link with new connection identifiers (see Figure 3).



[1] All VCI and VPI, however, have only local significance across a particular link, and are remapped as needed at each switch.

[2] For a full discussion and definition, please refer to [4].

Figure 3. VC and VP Switching [4]

ATM switches have "cross-connect" tables to translate the VPI/VCI connection (on a given link) identifiers in the incoming **ATM** cells to VPI/VCI combinations in the outgoing **ATM** cells. This translation can be done in hardware once a connection has been set up leading to a very fast switching operation. From the figure above, we can see that **ATM** switches translate either the virtual path identifier (VPI) only (in this case, it is called a VP-switch) or both the VPI and the VCI (called a virtual channel switch). Therefore, the VCI (virtual channel identifier) retains its value through a VP-switch but not through a VC-switch.

2.2 **How is a virtual circuit set up?**

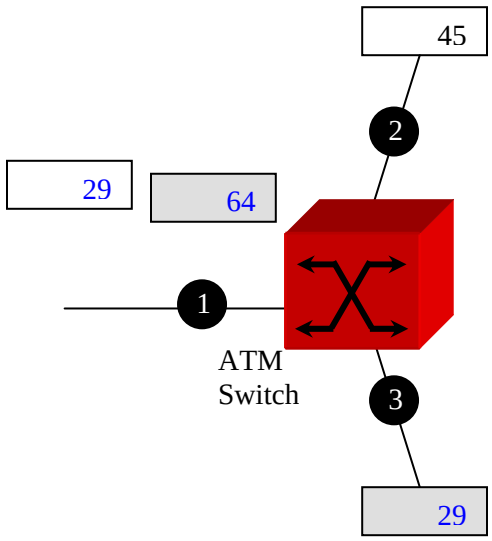
As it was mentioned above, most protocols running over **ATM** primarily use switched virtual circuits (or connections)[1]. It was also said that SVCs are set up through signaling mechanism. **ATM** signaling is initiated by an **ATM** end-system that desires to set up a connection through an **ATM** network. **ATM** signaling is allocated one or more *virtual channel connections* or *virtual path connections* within a given UNI or NNI link. These connections are then called *signaling virtual channels* (SVC)[2]. Information about all other virtual connections available between the user and the node at the UNI(user network interface) or between the two nodes at the NNI(network node interface)is carried by the signaling virtual channel. Thus unlike a telephone connection, where the dialed digits and other control information are transmitted over the channel itself (channel associated signaling), **ATM** uses a single, dedicated channel for all the signaling

[1] The difference is not relevant at this point.
[2] Not to be confused with SVC – *switched virtual connection.*

information (common channel signaling). On the UNI, signaling packets are sent over a well known[1] reserved virtual channel with VPI = 0 and VCI = 5.

To set up an SVC, a host communicates with its local **ATM** switch over the SVC (signaling VC) and specifies among others the complete **ATM** address[2] of a remote host and the quality of service. **ATM** signaling then takes over and establishes a path to the remote host. The signaling is routed through the network, from switch to switch, setting up the connection identifiers as it goes, until it reaches the destination end system. The latter can either accept and confirm the connection request, or can reject it, clearing the connection. Each **ATM** switch along the path examines the quality of service requested for the circuit. If it agrees to forward data, the switch records information about the circuit in its local translation table (see Figure 4) and sends the request to the next switch along the path. Such an agreement requires a commitment of hardware and software resources at each switch. When signaling completes, the local **ATM** switch reports success to both ends of the switched circuit (see Figure 5). The connection is set up along the path that the connection request follows; the data also flows along the same path.

Input		Output	
Port	VPI/VCI	Port	VPI/VCI
1	29	2	45
2	45	1	29
1	64	3	29
3	29	1	64



[1] ATM switches are pre-configured to receive any signaling packets sent across this connection and pass them to a signaling process.
[2] ATM address formats will be discussed briefly later.

Figure 4. An example of an ATM switch operation [5]

Since point-to-point virtual connections (the type of connections we discussed so far) are bi-directional, connection identifiers – with typically the same VPI/VCI values – are always allocated in each direction of the connection. However, the traffic parameters in each direction can be different; in particular, the bandwidth in one direction could be zero.

All aspects of the User-Network Interface, from the lowest physical details to the signaling messages exchanged above **ATM** level, are specified in the **ATM** Forum UNI 3.1 specification [6]. The signaling part of the UNI has been updated in UNI 4.0 specification [7].

ATM signaling uses the ‘one-pass’ method of connection set-up. That is, a connection request from the source end-system is propagated through the network, setting up the connection as it goes, until it reaches the final destination end-system. The routing

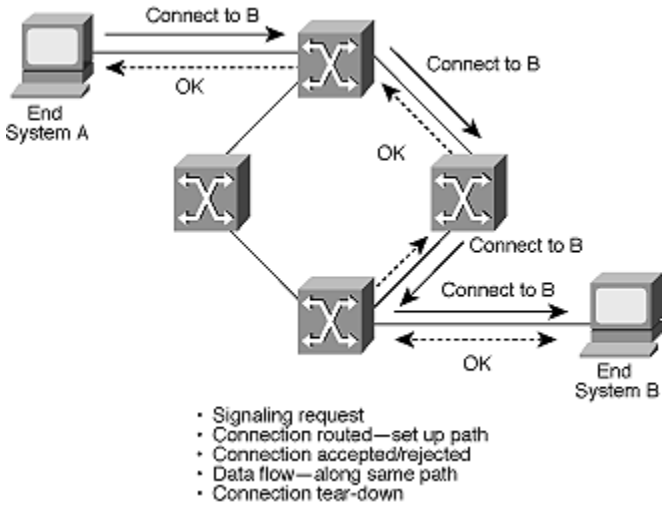


Figure 5. Connection setup through ATM signaling (SVC) [5]

of the connection request – and hence of any subsequent data flow – is governed by the **ATM** routing protocols discussed later. Such protocols route the connection request based upon both the destination address, the traffic, and the QoS parameters requested by the source end-system. The destination end-system may choose to accept or reject the connection request, but since the call routing is based purely on the parameters in the initial connection request message, the scope for negotiation of connection parameters between source and destination is limited.

The **ATM** signaling protocols run on top of a Service Specific Connection Oriented Protocol (SSCOP), defined by the ITU-T Recommendations Q.2100, Q.2110, and Q.2130. This is a data link protocol that guarantees delivery through the use of windows and retransmissions. In general, **ATM** does not offer an assured service – cells are not retransmitted by **ATM** devices upon loss, for instance, since it is assumed that higher layers (such as TCP[1]) will handle reliable delivery, if this is what the application requires. In fact, this makes **ATM** devices much simpler, faster, and cheaper. **ATM** signaling,

however, requires the assured delivery guarantees of SSCOP since it does not run on any standard higher layer protocol like TCP, and the signaling state machines can be made much simpler if assured delivery can be assumed.

A number of message types are defined in the UNI 3.1 specification, together with a number of state machines defining the operation of the protocol, cause error codes defining reasons for connection failure, and so forth. Data elements used in the signaling protocol – addresses, for instance – are carried within Information Elements (IE) within the signaling packets.

In overview, a source end-system wishing to set up a connection will compose and send into the network, across its UNI, a *Setup* message, containing the destination end-system address, desired traffic and QoS parameters, various IEs defining particular desired higher layer protocol bindings and so forth. This Setup message is sent to the first, ingress switch, across the UNI, which responds with a local *Call Proceeding* acknowledgment. The ingress switch will then invoke an **ATM** routing protocol to propagate the signaling request across the network, to the egress switch to which is attached the destination end-system.

This egress switch will then forward the Setup message to the end-system, across its UNI. The latter may choose to either accept or reject the connection request; in the former case, it returns a *Connect* message back through the network along the same path to the requesting end-system. Once the source end-system receives and acknowledges the Connect message, either node can then start sending data on the connection. If the destination end-system rejects the connection request, it returns a *Release* message, which is also sent back to the source end-system, clearing the connection (e.g. any allocated connection identifiers) as it proceeds. Release messages are also used by either of the end-systems or by the network to clear an already established connection. For more

---

[1] Transport Control Protocol.

detailed information on **ATM** signaling, I would strongly recommend reading **ATM** Forum UNI 3.1 specification [6].

In the **ATM** Forum UNI 4.0 specification, among other things, the notion of "Any-cast" service is introduced. "The ATM Anycast capability allows a user to request a point-to-point connection to a single ATM end system that is part of an ATM group" [7]. This means that a group of nodes can use a registration mechanism[1] to tell the network that they support a particular group address. Any node can then set up a point-to-point connection to any one of a group of **ATM** end-systems and the network will route the connection request to the *nearest* registered node. This feature will allow load sharing and redundant services without bothering **ATM** clients with connection failures.

[1] Actually, it is an extension of the Interim Local Management Interface (ILMI) address registration mechanism.

CHAPTER 3

ATM ADDRESSING AND ROUTING

3.1 ATM Addressing

Any signaling protocol requires an addressing scheme to allow the signaling protocol to identify the sources and destinations of connections. The ITU-T has long settled upon the use of telephone number-like E.164 addresses as the addressing structure for public **ATM** networks. Since E.164 addresses are a public (and expensive) resource, and cannot typically be used within private networks, the **ATM** Forum extended **ATM** addressing to include private networks.

3.1.1 ATM Model for Addressing

The **ATM** Forum chose what is called the overlay model for use with UNI 3.1 signaling. This model decouples the **ATM** layer from any existing protocol, defining for it an entirely new addressing structure. As a result, all existing protocols would still run over **ATM** networks. The overlay model requires the definition of both a new addressing structure and an associated routing protocol. All **ATM** systems would need to be assigned an **ATM** address in addition to any higher layer protocol (running on top of **ATM**) addresses. The **ATM** addressing space would be logically disjoint from the addressing space of whatever protocol is running over the **ATM** layer, and typically would not bear any relationship with it. Hence, all protocols operating over **ATM** would require an **ATM** address resolution mechanism to map higher layer addresses (such as IP addresses) to their corresponding **ATM** addresses.

3.1.2 ATM Addresses

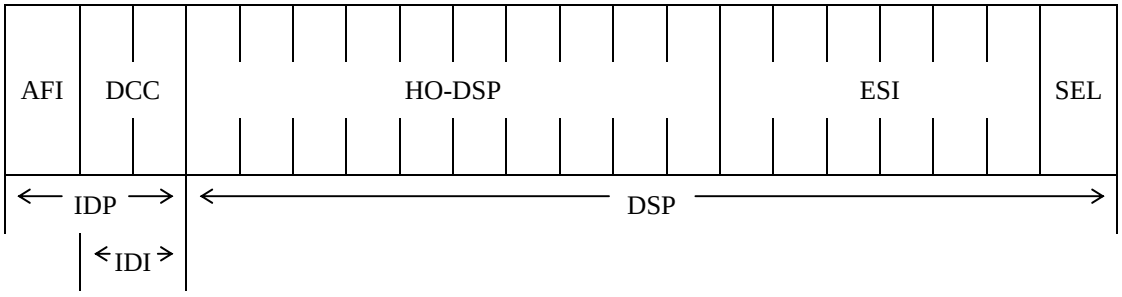
Given the choice of the overlay model, the **ATM** Forum defined an address format for private networks based on the syntax of an OSI Network Service Access Point (NSAP) address[1], as defined in ISO 8348 and ITU-T X.213 (see also IETF RFC 1629 [8]). The 20-octet NSAP format **ATM** addresses are designed for use within private **ATM** networks, while public networks typically use E.164 addresses defined by ITU-T. The **ATM** Forum did specify, however, an NSAP encoding for E.164 addresses. This will be used for encoding E.164 addresses within private networks but may also be used by some private networks.

All NSAP format **ATM** addresses consist of three components: an Authority and Format Identifier (AFI), which identifies the type and format of the Initial Domain Identifier (IDI); the IDI, which identifies the address allocation and administration authority; and the Domain Specific Part (DSP), which contains actual routing information.

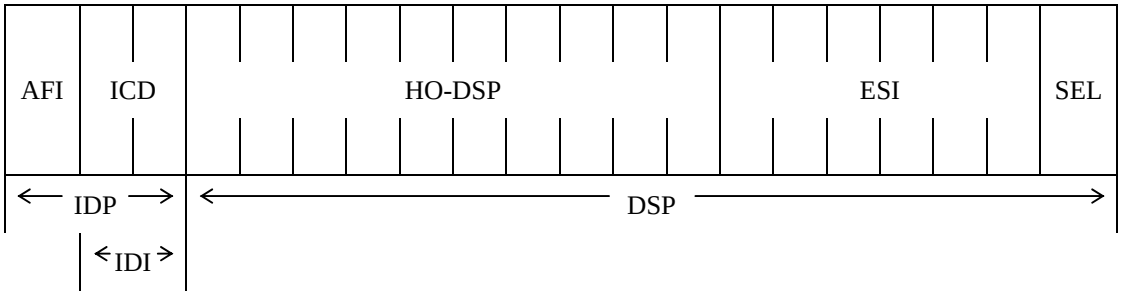
There are three formats of private **ATM** addressing that differ by the nature of the AFI and IDI (all three formats are, however, 20 octets long as specified by IETF RFC 1629, see Figure 6):

- *NSAP Encoded E.164 format:* where the IDI is an E.164 number.
- *DCC format:* where the IDI is a Data Country Code (DCC). This code identifies countries, as specified in ISO 3166. Such addresses are administered by the ISO National Member Body in each country.
- *ICD format:* where the IDI is an International Code Designator (ICD). It is allocated by the ISO 6523 registration authority (the British Standard Institute). ICD codes identify international organizations.

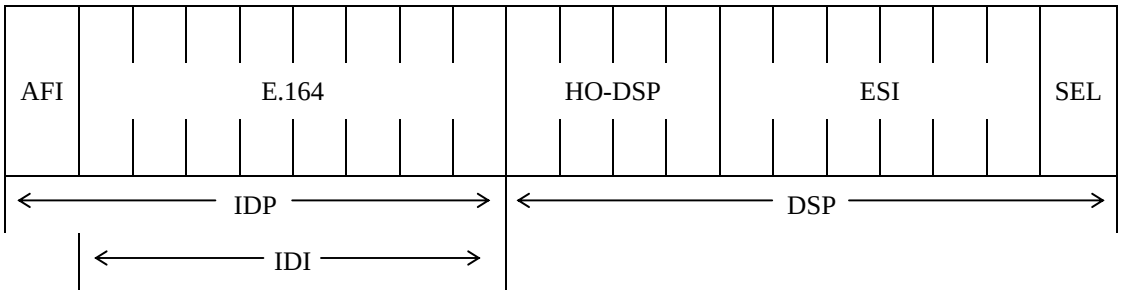
In real NSAPs, the DSP is typically subdivided into a fixed hierarchy that consists of a Routing Domain (RD), and Area identifier (AREA), and an End System Identifier (ESI). The **ATM** Forum, however, has combined the RD and AREA fields into a single High-Order DSP (HO-DSP) field, which is then used to support flexible, multi-level addressing hierarchies for prefix-based routing protocol.



DCC ATM Format



ICD ATM Format



E.164 ATM Format

Figure 6. ATM address formats

[1] Note, however, that an **ATM** address is not an NSAP despite the similar structure.

The ESI field is specified to be a 48-bit MAC address, as administered by the IEEE. This facilitates the support of both LAN equipment, which is typically hardwired with MAC addresses, and LAN protocols such as IPX, which rely on MAC addresses. The final, one octet, Selector (SEL) field[1] is used for local multiplexing within end-systems and has no network significance.

To facilitate the administration and configuration of **ATM** addresses into **ATM** end-systems across UNI, the **ATM** Forum defined an address registration mechanism using the ILMI[2]. This allows an **ATM** end-system to inform an **ATM** switch across the UNI, of its unique MAC address (called "user part"), and to receive the remainder of the node's full **ATM** address in return (called "network prefix") as shown in Figure 7.

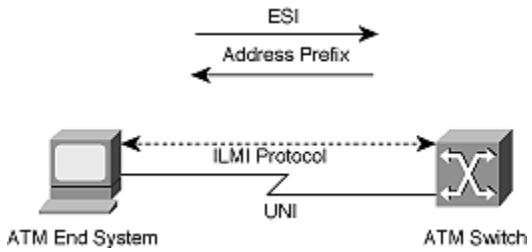


Figure 7. Address registration using the ILMI protocol [5]

This mechanism not only facilitates the autoconfiguration of a node's **ATM** addressing, but may also be extended in the future, for the autoconfiguration of other types of information (such as higher layer addresses and server addresses).

[1] It is similar, in a way, to the port number in TCP.
[2] Interim Local Management Interface

3.2 ATM Routing Protocols

Network Node Interface NNI protocols are used within **ATM** networks to route **ATM** signaling requests between **ATM** switches. Since **ATM** is connection oriented, a connection request needs to be routed from the requesting node through the **ATM** network and to the destination node. The **ATM** Forum has an ongoing effort to define a Private NNI (P-NNI) protocol. The current document [9] is a draft version of the Phase 1 P-NNI specification. The goal is to define NNI protocols for use within private **ATM** networks or, more specifically, within networks that use NSAP format **ATM** addresses. The P-NNI protocol consists of two components: the signaling protocol and the virtual circuit routing protocol. In the following two sections, a brief description of these protocols will be given. The last section describes the overall operation of the P-NNI Phase 1 protocol. If the reader is interested in greater detail, there are a number of good references on P-NNI routing protocols that he can consult, such as [4] and [9].

3.2.1 P-NNI Signaling Protocol

The first protocol is the P-NNI signaling protocol that is used for call/connection establishment and clearing over **ATM** networks, between the source and destination UNI. The UNI signaling request is mapped into NNI signaling at the source (ingress) switch. The NNI signaling is then remapped back into UNI signaling at the destination (egress) switch.

The P-NNI protocols operate between switching systems (which can represent either physical switches or entire networks operating as a single P-NNI entity), which are connected by P-NNI links. It is based on the **ATM** Forum UNI signaling, with mechanisms to support source routing, crankback, and alternate routing of call setup requests in case of connection setup failure. The P-NNI signaling protocol makes use of the information gathered by P-NNI routing protocol (described next). Specifically, it uses the calculations derived from the reachability, connectivity, and

resource information dynamically maintained by the P-NNI routing protocol. P-NNI signaling is carried across NNI links on the same virtual channel, VCI = 5, which is also used for signaling across the UNI, as mentioned earlier. The VPI value depends, however, on the NNI link type.

3.2.2 P-NNI Routing Protocol

The second component of the P-NNI protocol is the routing protocol, also called virtual circuit routing protocol. It is used to route the signaling request through the **ATM** network. The same path taken by the signaling request will be the route on which the **ATM** connection is set up, and along which the data will flow. This means that a path selected by P-NNI for establishment of a virtual connection or virtual path will remain in use for a potentially extended period of time. This, in turn, means that the consequences of an inefficient routing decision will affect a connection for as long as that connection remains open. Thus, it is critical that P-NNI routing protocol selects paths carefully. The operation of routing a signaling request through an **ATM** network, given the connection oriented nature of **ATM**, is superficially similar to that of routing connectionless packets within existing network layer protocols such as IP[1]. This is due to the fact that prior to connection setup, there is, of course, no connection for the signaling request to follow. As such, a VC routing protocol can use some of the concepts underlying many of the connectionless routing protocols that have been developed over the last few years. However, the P-NNI protocol is much more complex than any existing routing protocol. This complexity arises from two goals of the protocol: to allow for much greater scalability than what is possible with any existing protocol, and to support true QoS-based routing.

ATM allows a user to specify, when setting up a call, Quality of Service QoS and bandwidth parameter values that an **ATM** network must be able to guarantee

for that call. As mentioned before, call establishment consists of two operations: (1) the selection of a path, and (2) the setup of the connection state at each point (switch) along that path[2]. Path selection is done in such a way that the path chosen appears to be capable of supporting the QoS and bandwidth requested, based on currently available information. The processing of the call setup at each node along the path confirms that the resources requested are in fact available. If they are not, then crankback occurs, which causes a new path to be computed if possible. Therefore, the final outcome is either the establishment of a path satisfying the request, or refusal of the call.

3.2.3 P-NNI Phase 1 Protocol

The **ATM** Forum had set two main goals when designing the P-NNI protocol, namely, universal scalability and reachability, and true QoS support [9]. The P-NNI Phase 1 protocol is being designed to be capable of being applied both to small networks of few switches and to a possible future global **ATM** Internet comprising millions of switches. Such scalability is well beyond that of any single routing protocol today. By building upon the many years of experience gained in the development of current protocols[3], however, the **ATM** Forum hopes to build a single protocol that could perform at all levels within a network. Details of how this global scalability can be achieved are well documented in [9]. Since one of the great advantages of **ATM**, compared to other networking technologies, is its support for guaranteed QoS, the remaining of this section will be devoted to cover this aspect of the P-NNI Phase 1 protocol in detail.

A node requesting a connection setup can ask for a certain QoS from the network and can be assured that the network will deliver that QoS for the life of the

---

[1] Internet Protocol
[2] This latter operation is performed by the signaling protocol discussed in the previous section.
[3] Such as IGRP (Interior Gateway Routing Protocol) or OSPF (Open Shortest Path First).

connection. Such connections are categorized into various types of **ATM** QoS types: CBR, VBR, ABR, and UBR depending upon the nature of the QoS guarantee desired and the characteristics of the expected traffic types. Depending upon the type of **ATM** service requested, the network is expected to deliver guarantees on the particular mix of QoS elements that were specified at the time of connection setup (such as cell loss ratio, cell delay, and cell delay variation).

To deliver such QoS guarantees, **ATM** switches implement a function called *Connection Admission Control* CAC (see Figure 8). Whenever a connection request is received by the switch, the switch performs the CAC function. That is, based on the traffic parameters and requested QoS of the connection, the switch determines whether setting up the connection violates the QoS guarantees of established connections (for example, by excessive contention for switch buffering). The switch accepts the connection only if violations of current guarantees are not reported. CAC is a local switch function, and is dependent on the architecture of the switch and local decisions on the strictness of QoS guarantees.

The VC routing protocol must ensure that a connection request is routed along a path that leads to the destination and has a high probability of meeting the QoS requested in the connection setup – that is, of traversing switches whose local CAC will not reject the call. To do this, the protocol uses a topology state routing protocol in which nodes flood QoS and reachability information so that all nodes obtain knowledge about reachability information within the network and the available traffic resources. Such information is passed within P-NNI topology state packets (PTSP), which contain various type/length/value (TLV) encoded P-NNI topology state elements (PTSE). This is similar to current link state routing protocols such as OSPF used to advertise reachability information among different autonomous systems in the Internet. Unlike these, however, which only have rudimentary support for QoS, the P-NNI protocol supports a large number of

topology state parameters that are transmitted by nodes to indicate their current state at regular intervals, or when triggered by particular events.

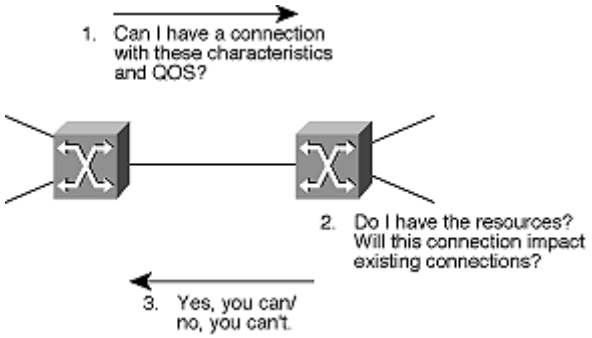


Figure 8. Connection admission control [5]

Topology state parameter is a generic term that refers to either a link state parameter or a nodal state parameter. Topology state parameters fall into two categories: topology metric (additive) and topology attribute (non-additive). A topology metric is a topology state parameter that requires the values of the state parameters of all links and nodes along a given path to be combined to determine whether the path is acceptable and/or desirable for carrying a given connection. A topology attribute is a topology state parameter that is considered individually to determine whether a given link or node is acceptable and/or desirable for carrying a given connection.

The current set of topology metrics includes the following:

- Cell Delay Variation (CDV) per traffic class specified in microseconds,
- Maximum Cell Transfer Delay (MCDT) per traffic class specified in microseconds,

- Administrative Weight (AW) is a value set by the network operator and is used to indicate the relative desirability of using a link or node. A higher value describes a link or node that is less desirable than a link with a lower value.

The current set of topology attributes includes the following:

- Cell Loss Ratio for CLP=0 (CLR_0)[1] is the maximum cell loss ratio objective for CLP=0 traffic. CLR is defined as the ratio of the number of cells that do not make it across a link or node to the total number of cells transmitted across that link or node,

- Cell Loss Ratio for CLP=0+1 (CLR_{0+1})[21] is the maximum cell loss ratio objective for CLP=0+1 traffic,

- Maximum Cell Rate (MCR) is the maximum capacity usable by connections belonging to the specified traffic class expressed in cells per second, required for ABR and UBR traffic,

- Available Cell Rate (ACR) is a measure of effective available capacity for CBR, VBR and ABR traffic, expressed in cells per second,

- Cell Rate Margin (CRM) is a measure of the difference between the effective bandwidth allocation per traffic class and the allocation for sustainable cell rate in cells per second; CRM is an indication of the safety margin allocated above the aggregate sustainable cell rate,

- Variance Factor (VF) is a relative measure of the square of the CRM normalized by the variance of the sum of the cell rates of all existing connections.

[1] Both CLR_0 and CLR_{0+1} are required topology attributes for CBR and VBR traffic.

All network nodes can obtain an estimate of the current state of the entire network through flooded PTSPs that contain such information as listed above. Typically, PTSPs include bi-directional information about the transit behavior of particular nodes based upon entry and exit port, and current internal state. This is particularly important in cases where the node represents an aggregated network (i.e. a peer group). In such a case, the node metrics must attempt to approximate the state of the entire aggregated network. This internal state is at least as important as that of the connecting links for QoS routing purposes. The need to aggregate network elements and their associated metric information also has important consequences on the accuracy of such information.

There are two basic routing techniques that have been used in networking: source routing and hop-by-hop routing. In source routing, the originating (or source) system selects the entire path to the final destination and other systems on the path obey the source's routing instructions. In hop-by-hop routing, each system independently selects the next hop for that path, which results in progress toward the destination provided that the decisions made at each hop are sufficiently consistent[1].

In principle, either routing technique can be applied to connection-oriented networks such as **ATM**. However, hop-by-hop routing has some disadvantages in this type of network. A major disadvantage is the need to eliminate the creation of routing loops. Routing loops might happen because of either inconsistency in routing decisions when switches use different routing algorithms or inconsistency in routing databases among the switches (typically due to changes in topology information that have not fully propagated yet). Another disadvantage of hop-by-hop routing is that it replicates the cost of path selection at each system. While this is less serious for connection-oriented networks (where the cost only occurs at connection setup), the QoS-sensitive path selection of P-NNI may be far more costly.

[1] Hop-by-hop routing is used by most current network layer protocols such as IP and IPX.

However, in source routing, only the source is responsible for selecting the path to the destination. It does this based on the requested QoS and its local knowledge of the network topology (gained from the PTSPs). It could then insert a full source routed path into the signaling request that would route it to the final destination. Ideally, intermediate nodes would only need to perform local CAC(connection admission control) before forwarding the request. Since only one database is involved, loops are not possible, nor will the paths be inefficient due to database inconsistency. Furthermore, since only the source selects the path, the algorithm used need not be the same in every system, and the cost of executing it is only incurred once. To avoid the difficulties of using hop-by-hop routing with connection-oriented service, and to gain the advantages of source routing, P-NNI uses source routing for all connection setup requests.

The above description of source routing is only ideal, however. In practice, the source-routed path can only be a best guess. This is because in any practical network, any node can have only an imperfect approximation of the true network state. This is due to the unavoidable latencies and periodicity in PTSP flooding. Furthermore, CAC is a local function as it was noted before. In particular, this means that the CAC algorithm performed by any given node is both system-dependent and open to vendor differentiation.

The P-NNI protocol tackles these problems by defining a Generic CAC (GCAC) algorithm. The idea of GCAC came from the realization that, since CAC is not to be standardized, it is not feasible to perform the actual CAC to determine if a switching system will admit a new connection. GCAC is a standard function needed in the path selection process to determine whether a link or node is likely to have enough resources available to support the proposed connection, based on that link/node’s advertised topology metrics (described above) and the requested QoS of the new connection request. GCAC should include a link/node if the switching system is likely to accept the proposed connection. Conversely, GCAC should exclude a link/node if the switching system is

likely to reject the new connection. In essence, GCAC predicts the outcome of the actual CAC performed at a switching system.

Using the GCAC, a node presented with a connection request (which passes its own CAC) processes the request as follows:

- All links that cannot provide the requested ACR and those whose CLR exceeds that of the requested connection are eliminated from the set of all possible paths using the GCAC.

- From this reduced set, along with the advertised reachability information, a shortest path computation is performed to determine a set of one or more possible paths to the destination.

- These possible paths are further pruned by using the (additive) topology metrics, such as delay, and possibly other constraints. One of the acceptable paths would then be chosen. If multiple paths are found, the node may optionally perform tasks such as load balancing.

- Once such a path is found[1], the node constructs a designated transit list (DTL) that describes the complete route to the destination and inserts this into the signaling request. The request is then forwarded along this path. For more information on the structure of DTLs, the reader may want to refer to [9].

This, however, is not the end of the story. Each node along this chosen path still performs its own CAC on the routed request because its own state may have changed since it last advertised it within the PTSP used for the GCAC at the source node. Its own CAC algorithm is also likely to be more accurate than the GCAC. Hence, there is always the possibility that a connection request may fail at some intermediate node. This

[1] Note that this is only an “acceptable” path to the destination, not the “best” path. The protocol does not attempt to be optimal.

becomes even more likely in large networks with many levels of hierarchy since QoS information cannot be accurately aggregated in such cases. To allow for such cases, without excessive connection failures and retries, the P-NNI protocol also supports the notion of *crankback* (see Figure 9).

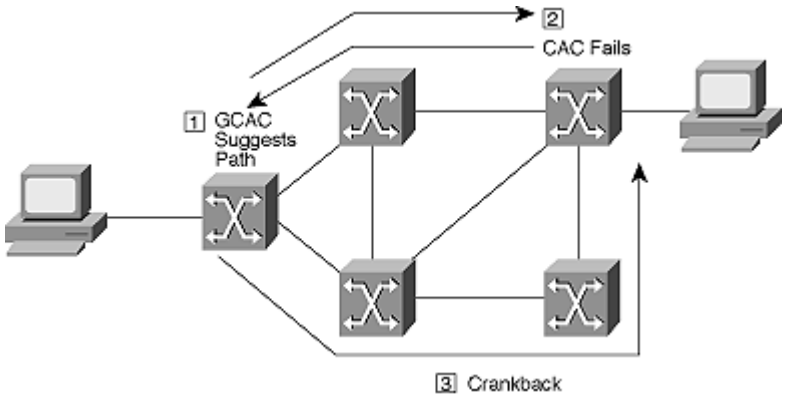


Figure 9. Operation of Crankback [5]

Crankback is where a connection, which is blocked along a selected path, is rolled back to an intermediate node, earlier in the path. This intermediate node[1] attempts to discover another path to the final destination, using the same procedure[2], based on newer, or hopefully more accurate network state. This is another mechanism that can be more easily supported in a source-routed protocol than in a hop-by-hop routing protocol. Crankback procedures are well explained in [9].

One of the concerns with P-NNI route generation is that most commonly used routing algorithms (such as Dijkstra calculations) were designed for single, cumulative metrics such as link weights or counts. Since P-NNI uses a number of complex link

[1] Only nodes that actually construct DTLs perform crankback.
[2] Described above and used by the source node.

parameters for link pruning, path selection may often not generate any acceptable path. In such cases, sophisticated algorithms may use a technique known as *fallback*, where particular attributes (such as delay) are selectively relaxed, and paths are recalculated in order to find a path that meets some minimal set of desired attributes.

At the beginning of this section, it was mentioned that the P-NNI Phase 1 specification is still a draft and that the **ATM** Forum is in the course of finalizing the specification. In the interim, the **ATM** Forum defined a P-NNI Phase 0 protocol, renamed later as Interim Inter-Switch Signaling Protocol (IISP) [10]. “It is built upon the ATM Forum UNI 3.1 Specification with optional support of the UNI 3.0, and allows for some base level capability at the P-NNI while the P-NNI Phase 1 Protocol Specification is being finalized” [10]. It should be noted, however, that as soon as the P-NNI Phase 1 specification is accepted, it would supercede the IISP specification [10].

There was a need for this IISP specification because, without a P-NNI protocol, there is no standard way for users to build interoperable multi-vendor **ATM** networks. The IISP, as the name suggests, is essentially a signaling protocol for inter-switch communication. Given the fact that the UNI 3.0/3.1 signaling procedures are symmetric, IISP uses UNI signaling for switch-to-switch communication.

Signaling requests are routed between switches using configured address prefix tables within each switch, which precludes the need for a VC routing protocol. These tables are configured with the address prefixes that are reachable through each port of the switch. When a signaling request is received by a switch, across either a UNI or an IISP[1] link, the switch checks the destination **ATM** address against the prefix table and notes the port with the longest prefix match. It then forwards the signaling request across that port using UNI procedures.

[1] An IISP link is a switch-to-switch link.

The IISP protocol is very simple and does not require any modification to UNI 3.0/3.1 signaling or any new VC routing protocol. The IISP, however, is not at all as scalable as the P-NNI Phase 1 protocol. For example, manually configuring prefix tables limits its applicability to networks with only a small number of nodes. For now, this should be adequate given that most **ATM** switches today are deployed in small test beds and not in large scale production networks. Users, however, need to upgrade their switches when the P-NNI Phase 1 protocol becomes available because it is not interoperable with IISP.

CHAPTER 4

ATM ADAPTATION LAYER

Although **ATM** switches small cells at the lowest level, application programs that transfer data over an **ATM** network do not read or write cells. Since The **ATM** layer provides a service-independent transport layer and it is a universal problem for **ATM** to adapt all information into the fixed-size 48-octet payloads, **ATM** designers defined the common protocol layer for mapping different types into the **ATM** layer. Hence, it was called the **ATM** adaptation layer (AAL). In addition, since different application classes have different networking requirements (such as error, delay and end-to-end timing), there is more than one type of AAL defined. Different AAL types were defined to meet more closely the needs of different types/classes of applications. The chief advantage of this approach is that whenever a new kind of traffic is recognized, a new AAL can be defined for it, while the **ATM** layer remains unchanged. The **ATM** adaptation layer is divided in two halves: one resides in the user plane and is called AAL; the second resides in the control plane and is called the signaling AAL or SAAL (see Figure 10).

It should be noted that the **ATM** protocols are organized into different protocol planes: user, control and management planes[1]. User plane represents user data transfer protocols while control plane represents signaling protocols for call/connection control. The reason there is a separate control plane is that **ATM** uses out-of-band signaling (as it was noted earlier). It is interesting, also, to note that the partition of the user and control planes occurs at the AAL and above but not at the **ATM** layer or physical layer. This is because the **ATM** layer does not care about the payload of the **ATM** cells whether it is control or user data.

[1] Note that the figure does not show the management plane; this is beyond the scope of this thesis.

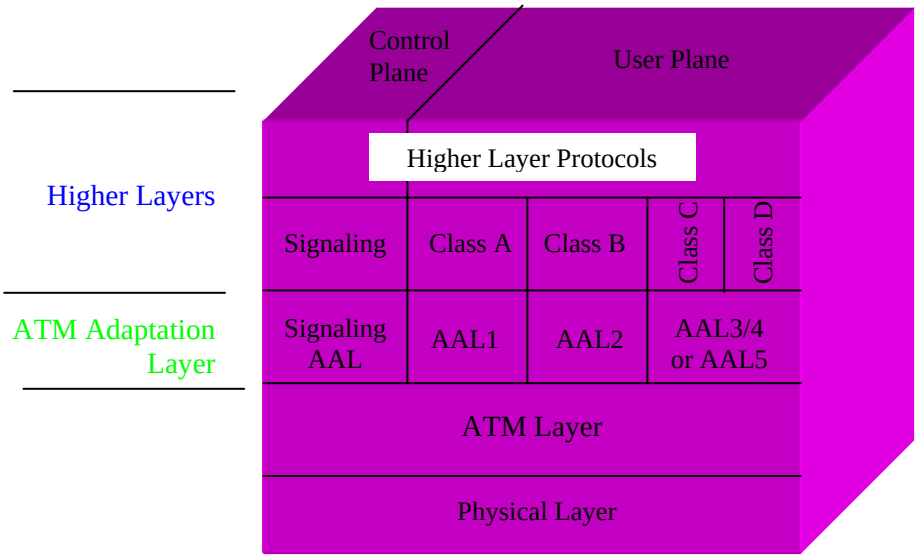


Figure 10. ATM Reference Model [11]

4.1 The ATM Adaptation Layer

The specifications of the **ATM** adaptation layer AAL are documented in the ITU-T I.362 [12]. It lists the following general examples of services provided by AAL:

- Handling of transmission errors,
- Segmentation and reassembly, to enable larger blocks of data to be carried in the information (payload) field of **ATM** cells,
- Handling of lost and misinserted cell conditions,
- Flow control and timing control.

The AAL layer is organized in two logical sublayers: the convergence sublayer (CS) and the segmentation and reassembly sublayer (SAR) [13] (see Figure 11). The convergence sublayer provides the functions needed to support specific applications using AAL. This sublayer is thus service dependent. The SAR sublayer is responsible for packing information received from CS into cells for transmission and unpacking the information at the other end.

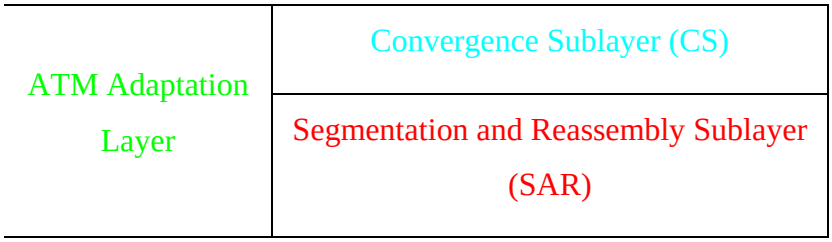


Figure 11. **ATM** adaptation sublayers (user plane)

To minimize the number of different AAL protocols that must be specified to meet a variety of needs, ITU-T has defined four classes of service that cover a broad range of requirements (see Table 1). The classification is based on whether a timing relationship must be maintained between source and destination, whether the application requires a constant bit rate, and whether the transfer is connection oriented or connectionless. ITU-T defined one protocol type for each class of service and each protocol type, actually, consists of two protocols, one at the CS sublayer and one at the SAR sublayer. The protocol types are: type 1, type 2, type 3/4 and type 5. Typically, an application would choose one specific AAL type[1] depending on the intended type of traffic and its constraints; the chosen AAL type will then be used all the time. Although each AAL is optimized for a specific type of traffic, there is no stipulation in the standards that AALs designed for one class of traffic cannot be used for another [14]. In all cases, a block of

[1] This is done at connection setup time and it cannot be changed.

data from a higher layer is encapsulated into a Protocol Data Unit (PDU) at the CS sublayer. This CS PDU is then passed to the SAR sublayer where it is broken up into payload blocks of size 48 octets ready to be carried by **ATM** cells.

	AAL Service Classes			
	Class A	Class B	Class C	Class D
AAL types	AAL1	AAL2[1]	AAL3/4 AAL5[2]	AAL3/4 AAL5
Bit rate	Constant (CBR)	Variable (VBR)	Variable (VBR)	Variable (VBR)
Timing Relation between source and destination	Required	Required	Not required	Not required
Connection Mode	Connection-oriented	Connection-oriented	Connection-oriented	Connectionless

Table 1. **ATM** classes of service [15]

In the next section, AAL5 will be covered in some detail since it is the most relevant AAL type to us in this thesis.

[1] AAL2 has not been fully developed yet.
[2] AAL5 is more often used for this class of service.

4.2 AAL5

AAL5 was developed by the computing industry to try to provide a more efficient AAL for data communications than AAL3/4 [16]. Although the AAL5 is the most recently standardized AAL type, it has become the most important AAL type because it supports all **ATM** switches and end-stations that implement SVCs. The reason is the **ATM** signaling protocols (i.e. UNI and NNI) have been standardized on using AAL5 (as we will see in the next section). The AAL5 designers had three major goals. They wanted a packet AAL with far less overhead than AAL3/4. The AAL was to minimize the computer's cost in handling cells (i.e. data is sent in large chunks). The AAL was also to behave as much as possible like existing data communications interfaces, like those for Ethernet and FDDI, so that existing data communications software could be easily ported to support **ATM**.

Using the AAL5 convergence sublayer, if the packet, including the 8-octet trailer, is an exact multiple of 48 octets, the division will produce completely full cells. If the packet is not an exact multiple of 48 octets, the final cell will not be full (see Figure 12). To accommodate arbitrary length packets, AAL5 allows the final cell to contain between 0 and 40 octets of data, followed by zero padding, followed by the 8-octet trailer. In other words, AAL5 places the trailer in the last 8 octets of the final cell, where it can be found and extracted without knowing the length of the packet.

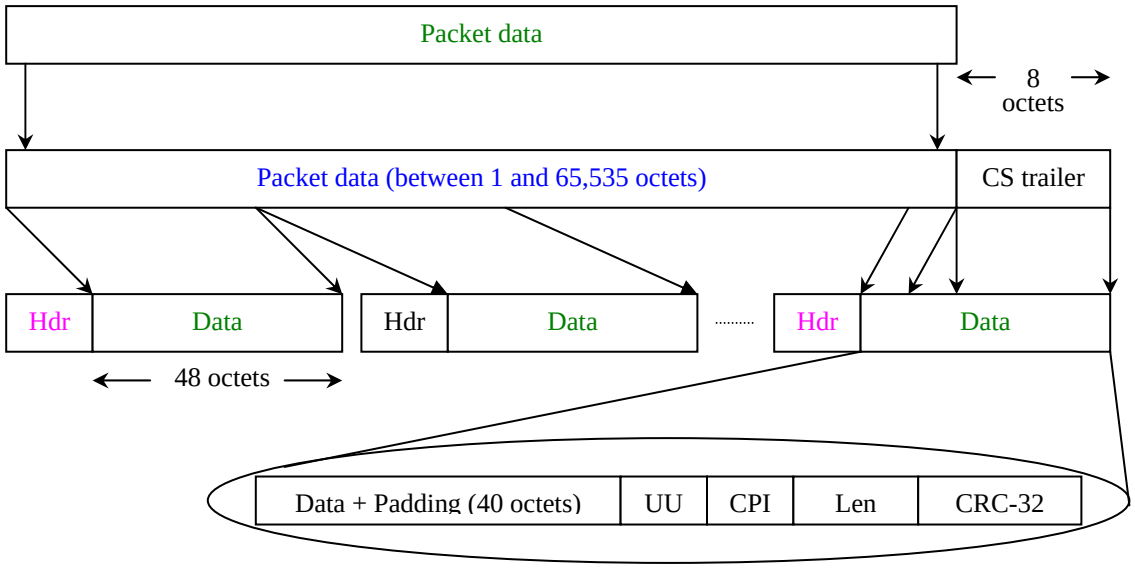


Figure 12. AAL5 Convergence and SAR formats

Unlike most network frames that place control information in a header, AAL5 places control information in an 8-octet trailer at the end of the packet. The trailer has four fields: a one-octet User-to-User Indication (UU) field, a one-octet Common Part Indicator (CPI), a two-octet length field, and a 32-bit cyclic redundancy check (CRC). The UU and CPI fields are currently unused and are set to 0. The length field is simply the number of octets of data in the packet (not including the padding).

The CRC is a 32-bit CRC over the entire CS packet, including the pad and the trailer. The CRC is extremely robust and can detect lost and misordered cells.

The SAR sublayer is encoded in the payload field of the **ATM** header. The sending AAL5 uses the low-order bit of the 3-bit payload type field of the **ATM** cell header to mark the final cell of a packet. Thus, the receiving AAL5 collects incoming cells until it finds one with the low-order bit of the PT field set. Then, AAL5 reassembles incoming cells into a packet, checks the CRC to ensure the packet has arrived intact, and passes the result to the host software.

One final worthy note is that an AAL5 frame is not self-identifying since the trailer does not contain a type field to tell which higher layer protocol (like IP) was used to create the packet. RFC 1483 [17] proposes a method to multiplex different protocols over a single **ATM** virtual circuit. Using this multiplexing, the higher layer protocol is identified by prefixing the packet by an IEEE 802.2 *Logical Link Control* (LLC) header followed by a *Sub Network Attachment Point* (SNAP) header. The 8-octet LLC/SNAP header is shown in Figure 13. Software on the sending host must prefix the LLC/SNAP header to each packet before sending it to AAL5, and software on the receiving host must examine the header to determine how to handle the packet. This scheme has the advantage of allowing all traffic to travel over the same circuit, but also has the disadvantage of requiring each packet to contain overhead that identifies the protocol

type. Another major disadvantage is that packets from all protocols travel with the same delay and priority.

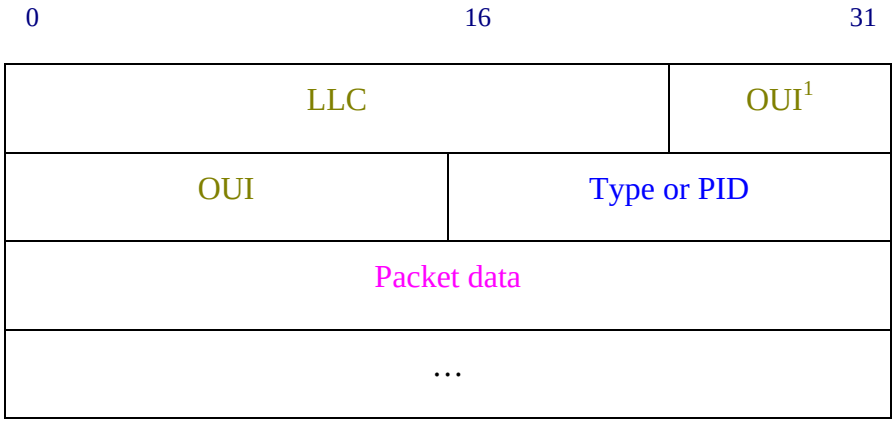


Figure 13. The LLC/SNAP header format [17]

4.3 Signaling AAL (SAAL)

The **ATM** Forum UNI signaling protocol is based on the ITU-T recommendation Q.2931. The Q.2931 (UNI) is supported over its own AAL, known as signaling AAL or SAAL. The purpose of the SAAL is to provide reliable transport of signaling messages between peer signaling entities (e.g. **ATM** switch and host) over the **ATM** layer [18]. The SAAL is composed of two parts: a common part and a service specific part (see Figure 14). The AAL5 common part has been adopted as the SAAL common part. The AAL5, however, provides an unassured information transfer mode. It does not guarantee reliability of data delivery; it only provides a mechanism for detecting corruption in the payload or Service Data Units (SDUs). To provide recovery of corrupted SDUs, a reliable protocol is used in the service specific convergence sublayer (SSCS) of the SAAL, that is, above the AAL5 common part. The SSCS of the SAAL also consists of two sublayers: the Service Specific Connection Oriented Protocol (SSCOP) and the Service Specific Coordination Function (SSCF). The SSCOP[2] is a reliable protocol that

[1] Organizationally Unique Identifier.
[2] It will be fully described later.

provides error recovery via retransmission and is specified in ITU-T recommendation Q.2110 [19]. An SSCF maps the service of SSCOP to the needs of the SSCF user. Different SSCFs may be defined to support the needs of different AAL users. Two SSCFs are currently defined: SSCF-UNI and SSCF-NNI. SSCF-UNI is the Service Specific Coordination Function that maps the particular requirements of the UNI layer 3 protocol to the SSCOP services. SSCF-UNI is specified in ITU-T recommendation Q.2130 [20]. SSCF-NNI is the Service Specific Coordination Function that maps the particular requirements of the NNI layer 3 protocol to the SSCOP services. SSCF-NNI is specified in ITU-T recommendation Q.2140 [21].

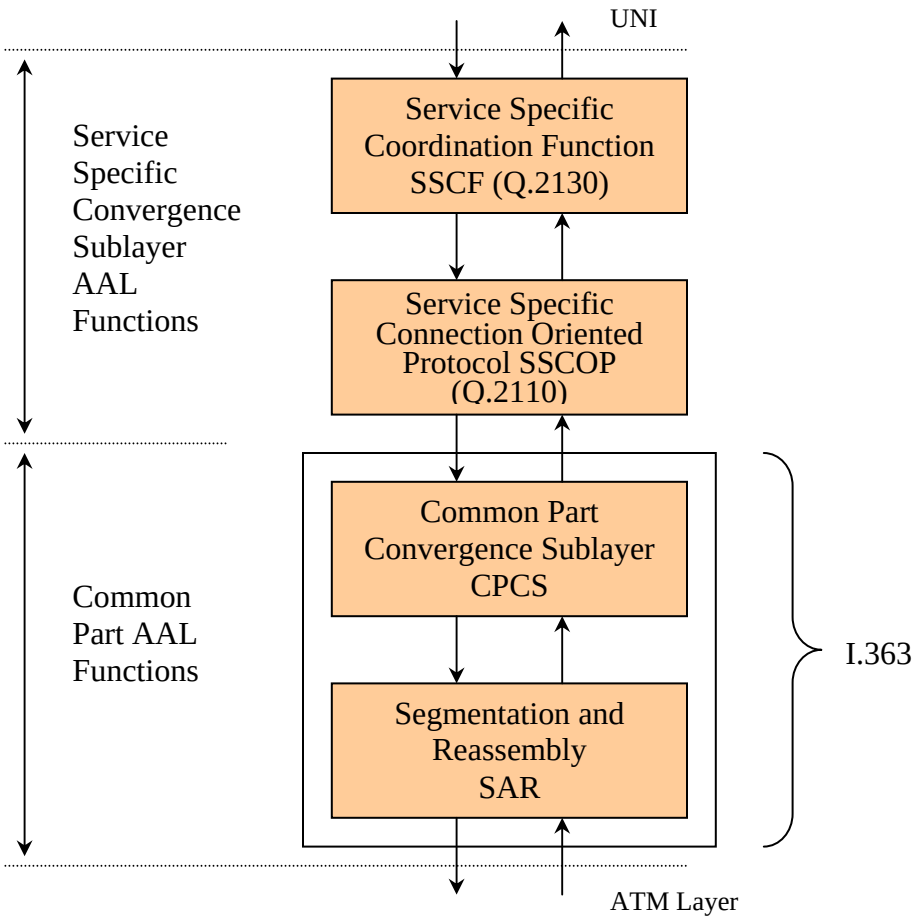


Figure 14. UNI SAAL Structure [15]

CHAPTER 5

TRAFFIC AND CONGESTION CONTROL IN ATM NETWORKS

5.1. Background

Congestion control has been a subject of continual interest to networking researchers and has become an increasingly important problem with the advent of gigabit networks. This is partly because the huge legacy of low-speed networks will continue to coexist with the current high-speed links for quite some time. This heterogeneity of networks and the resulting mismatch of link speeds are aggravating the congestion problem.

Among the available technologies, Asynchronous Transfer Mode (ATM) has emerged as a standard for supporting gigabit broadband integrated services digital networks (BISDN). Providing the desired quality of service of various traffic types (voice, video and data) is much more complex than the data networks of today. Proper traffic management helps ensure efficient and fair operation of networks in spite of constantly varying demand. Traffic management is concerned with ensuring that users get their desired quality of service. The problem is especially difficult during periods of heavy load particularly if the traffic demands cannot be predicted in advance. This is why congestion controlling, although only a part of the traffic management issues, is the most essential aspect of traffic management. New challenging issues arise in the congestion control of ATM networks. An important issue in the selection of congestion control scheme is the traffic control pattern. In ATM networks there are five traffic patterns or services:

5.1.1. Constant Bit Rate (CBR). This category is used of emulating circuit switching. The cell rate is constant. Cell loss ratio is specified for CLP=0 cells and may or may not be specified for CLP=1 cells. Examples of applications that can use CBR are telephone, video conferencing, and television (entertainment video).

5.1.2. Variable Bit Rate (VBR). This category allows users to send at a variable rate. Statistical multiplexing is used and so there may be a small nonzero random loss. Depending upon whether or not the application is sensitive to cell delay variation this category is subdivided into two categories: Real time VBR and Non real time VBR. For non real time VBR, only mean delay is specified, while for real time VBR, maximum delay and peak-to-peak CDV are specified. An example of real time VBR is interactive compressed video while that of non real time VBR is multimedia email.

5.1.3. Available Bit Rate (ABR). This category is designed for normal data traffic such as file transfer and email. Although, the standard does not require the cell transfer delay and cell loss ratio to be guaranteed or minimized, it is desirable for switches to minimize the delay and loss as much as possible. Depending upon the congestion state of the network, the source is required to control its rate. The users are allowed to declare a minimum cell rate, which is guaranteed to the VC by the network. Most VCs will ask for an MCR of zero. Those with higher MCR may be denied connection if sufficient bandwidth is not available.

5.1.4. Unspecified Bit Rate (UBR). This category is designed for those data applications that want to use any left-over capacity and are not sensitive to cell loss or delay. Such connections are not rejected on the basis of bandwidth shortage (no connection admission control) and not policed for their usage behaviour. During congestion, the cells are lost but the sources

are not expected to reduce their cell rate. In stead, these applications may have their own higher-level cell loss recovery and retransmission mechanisms. Examples of applications that can use this service are email, file transfer, news feed, etc. Of course, these same applications can use the ABR service, if desired.

These service categories relate traffic characteristics and quality of service (QoS) requirements to network behavior. The CBR service category is used by the connections that request a static amount of bandwidth that is continuously available during the connection lifetime. This amount of bandwidth is characterized by a Peak Cell Rate (PCR). The rt-VBR service category is intended for real time applications. rt-VBR connections are also characterized in terms of a PCR, and Sustainable Cell Rate (SCR), and Maximum Burst Size (MBS). The nrt-VI3R service category is intended for non real time applications, which have a bur4y traffic characteristics and which are characterized in terms of a PCR, SCR, and MBS. The UBR service category is intended for non-realtime applications. UBR service does not specific traffic related service guarantees. ABR is an ATM layer service category for which the limiting ATM layer characteristics provided by the network may change subsequent to connection establishment. A flow control mechanism is used which supports several types of feedback to control the source rate in response to changing ATM layer transfer characteristics. This feedback is conveyed to the source through specific control cells called Resource Management Cells (RM cells). It is expected that an end system that adapts the available bandwidth according to the feedback will experience a low cell loss ratio and obtain a fair share of available bandwidth according to a network specific allocation policy. On the establishment of an ABR connection, the end system shall specify to the network with a maximum required

bandwidth and a minimum usable bandwidth. These shall be designated as PCR, and Minimum Cell Rate (MCR).

While ATM was first conceived as a carrier of integrated traffic, the recent momentum on the rapid standardization of ATM has come from data networking applications. Most of these applications do not predict their own bandwidth requirements. Thus, an explicit guarantee of service cannot be provided. Since data application usually require a service that dynamically shares the available bandwidth among all active users, such a service is referred to as ABR. Due to highly bursty nature of data traffic, congestion control for ABR service poses more challenging problems than other services.

Several approaches to congestion control for ABR services in ATM networks have been considered in the past. One possible solution is to use sufficiently large buffers and the higher layer protocol handle congestion. However, simulation studies suggest that this approach does not achieve adequate performance. The loss mechanism simply discards arriving cells when the buffer is full, each discarded cell is likely to belong to a different packet in ATM networks. A significant portion of the bandwidth available to the ABR service is wasted since it is occupied by cells belonging to packets already corrupted by cell loss. Other loss mechanism has also been proposed, but they all yield inadequate performance. Among all proposed congestion control mechanisms for ABR service, closed-loop feedback control that allows the network to control the cell emission process at each source is now commonly agreed to be a standard mechanism for controlling congestion at ATM networks.

CHAPTER 6

CONGESTION CONTROL METHODS

6.1 Congestion

Congestion happens whenever the input rate is more than the available link capacity. More specific to ATM, congestion is defined as the condition where the offered load (demand) from the user to the network is approaching, or exceeds, the network design limits for guaranteeing the Quality of Service (QoS) specified in the traffic contract. Congestion control in ATM is different as compared to circuit switch networks or other packet switched networks due to following reasons :

6.1.1 Various B-ISDN VBR generates traffic at different rates.

6.1.2 A single source may generates different type of traffic.

6.1.3 ATM networks have to deal with cell delay variation, maximum delay.

6.1.4 Different services have different type of QoS requirements.

6.1.5 Traffic characteristics of various types of services are not well understood.

6.1.6 Due to large bandwidth propagation delay in B-ISDN a very large number of cells in transition, adds a new dimension to the problem.

7.1.7 High transmission speed limit the available time to do any processing at the intermediate level.

6.2 Quality of Service

Before we discuss the existing congestion control schemes, it is important to deal with the quality of service and its attributes. While setting up a connection on ATM network, users can specify the following parameters related to the input traffic characteristics and the desired quality of service:

6.2.1 **Peak Cell Rate (PCR).** The maximum instantaneous rate at with the user will transmit.

6.2.2 **Sustained Cell Rate (SCR).** This is the average rate as measured over a long interval.

6.2.3 **Cell Loss Ratio (CLR).** The percentage of cells that are lost in the network due to error and congestion and are not delivered to the destination.

$$\text{Cell Loss Ratio} = \frac{\text{Lost Cells}}{\text{Transmitted Cells}}.$$

Each ATM cell has a “Cell Loss Priority (CLP)” bit in the header. During congestion the network first drops cells that have CLP bit set. Since the loss of CLP=0 cell is more harmful to the operation of the application, CLR can be specified separately for cells with CLP=1 and for those with CLP=0.

6.2.4 **Cell Transfer Delay (CTD).** The delay experienced by a cell between network entry and exit points is called the cell transfer delay. It includes propagation delays, queuing delays at various intermediate switches, and service times at queuing points.

6.2.5 **Cell Delay Variation (CDV).** This is measure of variance of CTD. High variation implies larger buffering for delay sensitive traffic such as voice and video. There are multiple ways to measure CDV. One measure called "peak-to-peak" CDV consists of computing the difference between the (1 – α)- percentile and the minimum of the cell transfer delay for some small value of α.

6.2.6 **Cell Delay Variation Tolerance (CDVT) and Burst Tolerance (BT).** For sources transmitting at any given rate, a slight variation in the inter-cell time is allowed. For example, a source with a PCR of 10,000 cells per second should nominally transmits cells every 100 μs. A leaky bucket type algorithm called "Generalized Cell Rate Algorithm (GCRA)" is used to determine if the variation in the inter-cell times is acceptable. This algorithm has two parameters. This first parameter is the nominal inter-cell time (inverse of the rate) and the second parameter is the allowed variation in the inter-cell time. Thus, a GCRA (100μs, 10μs), will allow cells to arrive no more than 10μs earlier than their nominal scheduled time. The second parameter of the GCRA used to enforce PCR is called Cell Delay Variation Tolerance (CDVT) and of that used to enforce SCR is called Burst Tolerance (BT).

6.2.7 **Maximum Burst Size (MBS).** The maximum number of back-to-back cells that can be sent at the peak cell rate but without violating the sustained cell rate is called maximum burst size (MBS). It is related to the PCR, SCR, and BT as follow:

$$\text{Burst Tolerance} = (\text{MBS} - 1)\left(\frac{1}{\text{SCR}} - \frac{1}{\text{PCR}}\right)$$

Since MBS is more intuitive than BT, signaling messages use MBS. This means that during connection setup, a source is required to specify MBS. BT can be easily calculated from MBR, SCR, and PCR.

Note that PCR, SCR, CDVT, BT and MBS are input traffic characteristics and re enforced by the network at the network entry. CLR, CTD, and CDV are qualities of service provided by the network and are measured at the network exit point.

6.2.8 **Minimum Cell Rate (MCR).** This is the minimum rate desired by a user. Only the first six of the above parameters were specified in UNI. MCR has been added recently and will appear in the next traffic management scheme.

6.3 Existing congestion Control Schemes

Congestion control requires considerable exchange of information among the nodes in the network. The exchange of information occurs through the headers of data cells, and some through control cells (RM Cells). Congestion control schemes based on feed back mechanisms that have been proposed to the ATM forum fall into two categories: credit-based and rate-based. Rate-based schemes has been adopted by ATM Forum as a standard by ATM Forum for control of congestion in ATM networks .

6.3.1 Credit Based Schemes

Credit Schemes proposed in the ATM Forums adopt a link-by-link window flow control approach. In these schemes, control information (credit) on each link is returned from the receiving end to the sending end in order to ensure the receiving end’s ability to accept new data, thus avoiding congestion. More specifically, on each link a certain number of

cell buffers are reserved for each virtual connection (VC) at the receiving end of the link. A credit balanced is maintained at the sending end of each link and is deducted for every cell transmitted by the sending end. When a VC runs out of credit on a particular link, it will stop transmitting cells. As cells are removed from the buffer at the receiving-end, credits are returned on each VC to the sending end.

The per-VC link-by-link flow control mechanism adopted by existing credit schemes can prevent cell loss, which is important for data applications. Among other advantages such as maximal link utilization and fairness, the per-VC flow control also allows multiple VCs over the same physical link to operate at different speeds. However, per-VC link-by-link flow control requires per VC buffering at the switches, which result in considerable hardware complexity. In addition, the requirements imposed upon the switch architecture by credit schemes also limit flexibility in vendor implementation. In fact, many vendors participated in the ATM Forum have argued against the adoption of existing credit schemes as the ATM standards mainly because of these reasons.

6.3.2 **Rate Based Schemes**

Rate schemes use feedback information from the network to specify the maximum rate at which each source can emit cells into the network on every VC. The rate scheme proposed at ATM Forum by the rate based traffic management sub working group has undergone a number of variations and enhancements. In the following paragraphs we shall discuss several key developments of the rate schemes.

6.3.2.1 **FECN and BECN Rate Control Schemes**.

Forward Explicit Congestion Notification is one of the proposed rate scheme that are based on end to end control where the computational complexity resides mostly in the end systems. The feedback mechanism makes use of the explicit forward congestion indication (EFCI) state carried in the payload type identifier (PTl) field to convey congestion information in the forward direction. When a switch becomes congested it will mark on each VC the EFCI state of all cells being forwarded to the destination. Upon receiving the marked cells, the destination returns the congestion notification cells to the source of congested VCs to inform the source of the congestion status. The source uses this feedback information to increase or decrease the cell transmission rate accordingly on each VC.

Another scheme called Backward Explicit Congestion Notification (BECN) uses similar mechanism except that the congestion notification is returned directly from the point of congestion to the source. An obvious advantage of BECN over FECN is the faster response to congestion. Moreover, BECN is more robust against faulty or non-compliant end systems because the network itself generates the congestion notification. On the other hand, BECN requires more hardware in the switches to not only generate the BECN resource management (RM cells) but also filter the congestion information. This filtering process is

necessary in order to prevent the excessive RM cells being generated.

In both FECN and BECN schemes, a switch is considered congested when the length of the congested queue exceeds a certain threshold. Upon receiving a congestion notification cell, a source will decrease its cell rate for the corresponding VC. The current cell rate on a VC will increase if no congestion notification cell is received within a predetermined time period until reaches the peak cell rate. However, if the congestion notification cells returning to the source in the backward traffic experience extreme congestion, overall network congestion collapse may result. This is because every VC will attempt to reach the peak cell rate and overload the queue in the absence of returned congestion notification cells.

6.3.2.2 **PRCA (Proportional Rate Control Algorithm).**

As opposed to the negative feedback approach adopted by the FECN and BECN schemes, the proportional rate control algorithm is based on a positive feedback rate control algorithm. This shift in the rate control algorithm is intended to remedy the problem of network congestion collapse as described in preceding paragraphs. In PRCA, a source only increase it cell rate for a VC when it receives an explicit positive indication from its destination; otherwise, in the absence of such a positive indication, the source will continually decrease its cell rate. The increments and decrements in the cell rate of each VC are proportional to the current cell rate, thus eliminating the need for timers and timer value.

The feedback mechanism in PRCA also makes use of the EFCI state of the data cells. The source transmits all its cells with EFCI=1, except for the first cells and once every N cells that follow which are transmitted with EFCI=0. The parameter N is predetermined and will effect the response time to congestion and the backward link utilization. If a destination is not congested locally, for each unmarked cell (EFCI~0) it receives, it will generate and return a resource management (RM) cell to the source in the backward direction. A source will only increase its cell rate when it receives an RM cell from the destination. A switch experiencing congestion can therefore, change the EFCI state of the unmarked cells to EFC =1 or removes the RM cells in the backward direction. Since a source continually decreases its cell rate for every cell sent, each congested VC would then reduce its cell rate until the switch is not congested. The positive feedback mechanism incorporated in PRCA has resolved the problem of network congestion collapse caused by the negative feedback mechanisms of the FECN and BECN schemes. However, as in the early FECN and BECN proposals, PRCA still suffers from problems due to VC starvation. When a VC is going through more congested link its data cell will be marked more often then those of other VCs going through fewer congested links. Consequently, such a VC will have a lower allowed cell rate (ACR) then other. This undesirable effect of VC starvation is proportional to the number of congested links on which a VC transmits its cells, and is referred to as the ‘Beat Down Problem’ of PRCA.

6.3.2.3 **EPRCA (Enhanced Proportional Rate Control Algorithm)**

The enhanced proportional rate control algorithm (EPRCA) was proposed to perform intelligent marking in order to achieve fairness. In this algorithm all the connections are assumed to share a common queue with two congestion thresholds QT and DQT. For every forward RM cell received, the switch computes a Mean Allowed Cell Rate (MACR) using an exponential running average as follow:

$$MACR = (1 - \alpha)*MACR + \alpha*CCR$$

Where α is the averaging factor and is generally chosen to be 1/16. The switch estimates the average of the Current Cell Rate (CCR) values of connections that are not bottlenecked elsewhere by keeping track of the queue length. When the queue length exceeds the threshold QT, congestion is declared, and only those connections whose CCR values are above the fair share computed as DPF*MCCR are asked to reduce their rates, while the rates for the rest of connections are allowed to increase. The parameter Down Pressure Factor (DPF), typically set to 7/8. is known as the DPF, and is introduced to reduce the rates of those connections whose rates are very close to MACR, thus avoiding the potential oscillation. When the queue length exceeds the threshold DQT, the switch is said to be in a state of severe congestion. In this case, all the connections are asked to reduce their rates regardless of their CCR values. The switch uses the fair share to perform intelligent marking. In addition, if the value in the ER field of a backward

RM cell is above the fair share and the switch is in a congested state, the ER value is lowered to its fair share; otherwise, no adjustments need to be made.

The most obvious advantage of this algorithm is low implementation complexity. To make this scheme work effectively, in practice, however, CCR values and MACR must converge under all conditions. To accomplish this, the algorithm has used several multiplier factors to enforce the convergence. Unless these parameters are tuned properly, the algorithm could exhibit severe oscillations, and may not converge to the desired fair rate. Considerable unfairness could persist when the estimation of MACR is not close to the fair share to which the CCR values of the connections are supposed to converge. This situation could happen when some connections bottlenecked elsewhere in other switches cause underestimation of the fair share, when transient behaviors cause rate oscillations, and when the sources are not well behaved and/or insert incorrect CCR values.

6.3.2.4 **MMRCA (Maximum Minimum Rate Control Algorithm)**

One possible enhancement to the EPRCA is the MMRCA algorithm. In this algorithm, the switch maintains the minimum rate (MIN) and the maximum rate (MAX) of all the connections and their corresponding connection numbers, recorded as MAX-VC and MIN-VC, respectively, These rates, along with the two queue thresholds QT and DQT, are used to approximate the fair rate. When the queue length falls below the threshold QT, all the connections are allowed to increase their rates except the MAX-VC connection. Similar to the EPRCA, the switch is considered to

be in a state of congestion when the queue length exceeds the threshold QT. However, congestion is indicated to only those connections whose CCR values are above a value of IPF*MIN, while the rates for the rest of the connections are allowed to increase. The parameter IPF is known as Increase Pressure Factor, and serves a similar purpose to the DPF used in the EPRCA. If the difference between the MAX and MIN values is smaller than a predefined value, no connections are allowed to increase their rates. Similar to the EPRCA, all the connections are asked to decrease the rates if the queue length exceeds the. DQT threshold. It is shown that when the MAX and MIN rates do not quickly converge to the fair rate, the algorithm can potentially lead to unfairness. An algorithm to solve this problem has recently been proposed, and is presented below.

6.3.2.5 **DMRCA (Dynamic Maximum Rate Control Algorithm)**

Similar to the EPRCA, the dynamic max rate control algorithm (DMRCA) uses two queue length thresholds for congestion detection. Like the MMRCA, the switch monitors the maximum rate (MAX) of all connections arriving at the switch, and records the connection number corresponding to the MAX value as MAX-VC. In order to smooth out the oscillations due to dynamic changes of the MAX value, the switch maintains an exponential running average on the average maximum rate (AMAX) which is computed as follows:

$$AMAX = (1 - \alpha) * AMAX + \alpha * MAX.$$

Again, α is the averaging factor. It should be noted that the

above operation is only performed whenever a new MAX value has been detected.

When the queue length exceeds the threshold QT, the switch performs intelligent marking based on a marking threshold. This marking threshold, expressed below, is calculated as a function of the queue length and the AMAX.

$$MT = AMAX * Fn (Queue\ Length)$$

where the function Fn (Queue Length) is a discrete, non-decreasing function of the queue length ($0 < Fn < 1$).

When the queue length is somewhere between thresholds QT and DQT, the CCR field in the RM cell is compared against MT, and the outcome of this comparison is used to selectively indicate congestion. It was reported that to use the ER setting when the queue length stays in this region can sometimes result in poor performance. When the queue length is above the threshold DQT, the ER marking is used to reduce rates rapidly. The desired or Explicit Rate (ER), field in the RM cell is set equal to AMAX*MRF, where MRF stands for the major reduction factor.

The DMRCA algorithm has low implementation complexity and generally converges to the fair rate. The intelligent marking threshold defined to be a function of the queue length, results in better control of buffer usage. The algorithm is also proved to be insensitive if the CCR values are not set correctly. Special care has to be taken, however, to tune some of the parameters, such as RIF and RDF. The algorithm has been shown to exhibit very large rate oscillations if RIF is set too high, and

sometimes can lead to potential unfairness. For the algorithm to work effectively and correctly, it seems necessary to choose *a* very small RIF value.

6.3.2.6 **CAPC (Congestion Avoidance using Proportional Control)**

The CAPC algorithm proposed is a congestion avoidance scheme in which the rate of change of queue length is used as a load indicator. In this algorithm, the switch computes a load factor, LF, which is defined as a ratio of the actual input rate and a predetermined target rate for the link, and it is given as below:

LF = Input Rate/Target Rate

The input rate is measured over a fixed averaging interval and the target rate is set slightly below the link bandwidth (e.g. *85-90* percent). The load factor is used to update the fair share which is computed differently depending on the value of the load factor. Loads factor less than 1indicates that the queue is under loaded, whereas a load factor greater than 1 indicates that the queue is overloaded.

During under load, the fair share is increased according to

Fair Share = min (ERU, 1 + (1-LF)* Rup) *(Fair Share)

where Rup is a slope parameter between 0.025 and 0.1, and ERU determines the maximum allowed increase of the fair share. Accordingly, during overload, the fair share is decreased and updated as

Fair Share = max(ERF, 1– (LF-1)*Rdn) *(Fair Share)

Where Rdn is a slope parameter between 0.2 and 0.8, and ERF is the minimum allowed decrease and is chosen to be *0.5*. If the calculated fair share is lower than the ER value in the RM cell, then the ER field in the RM cell is set to the fair share. Similar to the FPRCA, the CAPC algorithm does not require' maintenance of any variables on a per connection basis. The parameters required to force convergence, however, must be set carefully; incorrect settings could cause large oscillations in rates and potential unfairness. Similar to other approximate algorithms, the CAPC is sensitive to the choice of RIF and RDF values.

6.3.2.7 **CCERI (Congestion Control with Explicit Rate Indication)**

The algorithm presented was an early proposal to compute explicit rate in a distributed manner, originally formulated in the context of packet switching. The development of this algorithm has had a significant influence on the fair rate allocation algorithms for ABR service. At the time of its development, much of the ABR specification did not exist. Thus, many of the features available now in RM cells were not utilized in its design.

In this algorithm, each switch monitors its traffic and calculates its available capacity per flow or per connection. This quantity is called the advertised rate. The switch keeps track of the number of bottlenecked connections at the switch and the last seen ER values. When an RM cell arrives at the switch, if its ER is less than the advertised rate, then the associated connection is assumed to be bottlenecked elsewhere. A bottleneck bit is marked and the

current rate of the connection is stored in a connection table. At each iteration, advertised rate is computed directly. If at any time a connection previously marked transmits at a rate larger khan the advertised rate, the corresponding bottleneck bit is then unmarked, and the advertised rate is recalculated.

The algorithm is proved to converge to the optimal max-min rate from any initial conditions, and the convergence time is upper-bounded by 4M round trip times, where M is the number of bottleneck links in the network. The steady state bandwidth utilization of this algorithm does not oscillate, and has a fast transient response. The algorithm however requires per connection information for each of the established connections at the time of advertised rate computation. Therefore, the algorithm has a computational complexity of 0(n), where n is the number of active connections.

The computation time would be significant when n grows larger. Moreover, this algorithm requires that all the switches along its path execute the same algorithm, thus preventing it from inter operating with any other switches that use a different fair rate allocation algorithm. This requirement is due to the fact that the advertised rate calculation is primarily based on the ER field in the RM cell without any added intelligence. Consequently, if a simple switch that does not mark the ER field exists in the network, the congestion information from the simple switch will be ignored.

This scheme requires that sources be allowed to send cells at a rate chosen from a discrete set of possible transmission rates. Thus, switches are only needed to store these discrete values, thus

reducing the complexity significantly. For this new scheme to work effectively, it is required that all the sources be aware of this discrete set of rates, and that all the switches in the network are implemented with the same algorithm.

6.3.2.8 **ERICA (Explicit Rate Indication for Congestion Avoidance)**

The ERICA algorithm proposed is another variation of congestion avoidance schemes. This algorithm calculates two quantities: fair share and this connection's (VC's) share. Similar to the CAPC, it computes a load factor based on the input and target rates. Utilizing the load factor LF, the VC share is calculated as

$$VC_{share} = CCR/LF$$

Where CCR is derived from the forward RM cell received to ensure that the most current information is used to provide fast feedback.

In order to achieve fairness, this algorithm allows connections to increase their rates to a fair share during under load. The fair share is calculated by the following formula:

$$\text{Fair share} = \text{Target Rate}/N$$

Where N is the number of active connections. Upon reception of a backward RM cell, its ER field is marked down as follows:

$$ER = \min\{ER, \text{rnax}(\text{Fair share}, VC_{share})\}$$

The ERICA algorithm can operate in a congestion avoidance state, is insensitive to parameter variations, and proves to be very robust. The rates converge very quickly and with little oscillation. This algorithm however, has some fundamental limitations in achieving desired fairness for all connections and buffer requirements. In some cases, a connection that gets started late gets its equal link share, but does not get the max min fair rate. Furthermore, during transient periods, and if the desired target utilization is set close to the full link rate, the queue grows rapidly and results in heavy cell loss. Although not discussed here, there exist many extensions and modifications of this algorithm. These extensions, with added complexity, try to eliminate some of the problems found in the basic ERICA algorithm.

6.3.2.9 **ERAA (Efficient Rate Allocation Algorithm)**

The algorithm presented, attempts to solve the computational complexity of the CCERJ algorithm. Similar to that algorithm, the switch maintains per connection information such as bottleneck state and bottleneck bandwidth of each connection. The switch calculates an equal share bandwidth (EQB) which is given by

EQB = Available bandwidth/Total number of connections

When a connection does not use the its equal share bandwidth, the connection is considered to be bottlenecked elsewhere. The sum of free bandwidth not used by the

bottlenecked connections is referred to as the free bandwidth (FB). A portion of the FB is allocated to the non bottlenecked connections in addition to their EQB, and this new rate is referred to as Amax.

The calculation of Amax occurs when a forward RM cell arrives. In order to perform the calculations in 0(1) time, only the change in the connection state is considered. It is important to note that the switches compute the AMAX based on the minimum of CCR and ER values received. This makes it possible for the algorithm to inter-operate with other algorithms. This algorithm in steady state converges to max-min fair rates quickly, while being efficient and stable.

The ERAA algorithm, however, assumes that the exact transmission rate is carried in the CCR field. If a connection is bursty or idle for long time, the CCR field in the RM cell may not carry the correct value. The inaccuracies in CCR value may present problems for ERAA to function efficiently. There are no mechanisms in ERAA to modify the rate allocations to reflect the CCR inaccuracies, and further study is required to address this problem.

6.3.2.10 **FMMRA (Fast Maximum Minimum Rate Allocation Algorithm)**

This algorithm is based on measurement of available capacity and exact calculation of fair rates. The goals of this algorithm are to reduce the computational complexity imposed by the CCERI algorithm discussed earlier, and at the same time

make the algorithm interoperable. An additional important feature of this algorithm is that it is not sensitive to inaccuracies in CCR values. The algorithm can be executed by any switch component experiencing congestion (input port, output port, etc.). Each ABR queue in the switch computes a rate it can support. This rate is referred to as the advertised rate. The advertised rate along with the ER fields in the RM cell are used to determine if the connection is bottlenecked elsewhere. If a connection cannot use the advertised rate, it is marked as bottlenecked elsewhere and its bottleneck bandwidth is recorded. The ER field in the RM cell is read and marked in both directions to speed up the rate allocation process. The bi-directional ER marking in this algorithm makes it possible for downstream switches to learn bottleneck bandwidth information of upstream switches, and upstream switches to learn bottleneck bandwidth information of downstream switches. Many of the proposed algorithms mark the ER field only in the backward direction. Because of the unidirectional ER marking, the switches closer to the source get more accurate ER information than those closer to the destination. As a result, this may result in slower response to congestion. This bi-directional updating of ER in the RM cell plays a significant role in drastically reducing the convergence time of the max min fair rate allocation process.

In contrast to the algorithm presented, which requires the switch to inspect the states of all the connections in the connection table when calculating the fair rate, the FMMRA algorithm only considers the changes in the per connection variables. In addition, it only requires knowledge of the

connections seen by the switch at the time of update. When a backward RM cell is received, the advertised rate is re-computed by incorporating the bottleneck state of the received connection. This feature reduces the computational complexity of this algorithm to be 0(1).This algorithm does not use the CCR carried in the RM cells; instead, it uses the load factor and the ER to compute an exponential running average of the maximum value of ER, denoted as ER max. This operation is shown below:

$$(ER_{max}) = (1 - \alpha)*ER_{max} + \alpha_{max} *[ER, ER_{max}/LF]$$

Where α is again an averaging factor and can be set to 1/8. The computation of ER_{max} is done in the backward direction, and reflects the advertised rate after taking the load into consideration. The load factor reflects how well the ABR bandwidth is utilized. For example, LF < 1 indicates that some of connections are sending cells at rates less than their allowed rates. This presents an opportunity for the non-bottlenecked connections in the link to increase their rates.

Based on the level of congestion, which is determined as a function of the queue length and the load factor, the ER field in the RM cell (both forward and backward) is updated according to

$$ER = \min\{ER, \max(\gamma, (1- \beta)*ER_{max})$$

where β is a single bit value indicating the connection is bottlenecked elsewhere. The *use* of the advertised rate along with ER_{max} helps the fair rate to converge faster. They also help to adjust the fair share according to load variations in the network, in order to achieve full link utilization. Moreover, the added

intelligence provided by the use of LF and ER_{max} makes it possible for this algorithm to inter-operate with other algorithms.

Updating the ER field in order to control the queue growth to prevent potential cell loss. If the queue length reaches a low threshold QT and LF> I, only the advertised rate is used in marking the ER field in the RM cell, as shown below:

$$ER = \min(ER, \gamma)$$

Under severe congestion, ER_{max} is set to the advertised rate. This ensures that whenever the potential for congestion is detected, even if some connections are idle, the non idle connections are not given any extra bandwidth, allowing the queue to drain. Furthermore, if the queue length exceeds a high threshold DQT (indicating heavy congestion), the target utilization factor is reduced by a target rate reduction factor (TRRF) until the queue length drops below the low threshold QT. This technique allows the queue to drain and operate at a desired queue length whenever the switch is heavily congested, for example, when many new connections are established.

6.3.2.11 **ICC (Intelligent Congestion Control)**

This scheme is a refinement of EPRCA. The key idea of this scheme is that each congested switch is to estimate the optimal cell rate on each VC with a small number of computation and without the need of per-VC queuing of accounting. Then, using simple feedback mechanisms, this estimated rate would be used to adaptively adjust the cell rates of the sources. A switch will estimate the Modified Current Cell Rate by

$$MCCR=MCCR + \beta * (CCR – MCCR)$$

When a congested switch receives an RM(CCR, ER) cell from the destination. If the current queue length is greater than DQT (a fixed threshold value), it replaces ER in the RM cell by min(ER, γ*MCCR); otherwise, if CCR>MCCR, it replaces ER in the RM cell by min(ER, MCCR), where y is the fast reduction ratio varies from 0 to I.

This scheme though converges to optimal cell rate quickly but suffers oscillations. If a connection is bursty or idle for long time, the CCR field in the RM cell may not carry the correct value. The inaccuracies in CCR value may present problems for ICC to function efficiently. There are no mechanisms in ICC to modify the rate allocations to reflect the CCR inaccuracies, and further study is required to address this problem.

CHAPTER 7

REFINEMENT OF PREVIOUS SCHEME

7.1 Undesirable Symptoms

To resolve the undesirable symptoms of existing rate schemes,. it is useful to first acquire a fundamental understanding of their limitations, summarized as below:

7.1.1 In most proposed rate schemes, a congested switch essentially makes no distinction among different VCs. In fact, it is this limitation that causes a VC going through more congested links to be 'beaten' more often than other VCs going through fewer congested links.

7.1.2 Lacking a mechanism to distinguish the cell rates between various sources, a switch experiencing congestion can only attempt to reduce the cell rates among all the VCs in the same fashion.

7.1.3 Per VC queuing thus enhancing cost and complexity.

7.1.4 Larger oscillations while converging to the optimal rates.

7.1.5 More time is required to converge to the optimal rate.

7.1.6 Large size of buffer thus increasing cost.

7.2 Refined Scheme

The scheme, which has been proposed, is a refinement of the 'Intelligent Congestion Control Scheme'. The key idea of our scheme is for each congested switch to estimate the optimal cell rate on each VC with a small number of computations and

without the need of per-VC queuing or accounting. Then, using simple feed back mechanism, this estimated rate would be used to adaptively adjust the cell rates of the sources. The schemes reduces the oscillations as compare to Intelligent scheme at the same time producing lesser occupancy of the buffer i.e. lesser cost of the switch.

More specifically, each VC periodically carries from the source to the destination an explicit RM cell containing its current allowed cell rate (CCR). Given the CCR of each VC, an intermediate switch (congested or non-congested) uses a simple filter to repeatedly estimate the optimal cell rate for each VC. To keep the memory requirement and computational minimal, a first order filter is used. Buffer occupancy at the switch is used it indicate the congestion. Three levels of threshold indicating mild, transient and sustained congestion are defined with gradually increasing values of ‘gamma’. The Fast Reduction Ratio’ are used for the computations of the desired cell rate or ‘explicit cell rate, ER; at each switch as opposed to Intelligent Scheme when only one level of threshold is defined, with a single value of gamma. A switch experiencing congestion will use its estimated cell rate and the CCR of a given VC to determine the appropriate new rate of that VC. This information from the switches will be conveyed to the sources with a positive feedback mechanism, as in Intelligent Congestion Control Scheme. Every time a source transmits a data cell, it will decrease its rate until it receives an RM cell from the destination. When the destination receives the RM cell, it will return in the backward direction another RM cell to the source. The purpose of this backward RM cell is to obtain the rate information computed at the switches and signal the source with new computed rate, which is smallest among all rates computed at the congested switches. Effectively, VCs with cell rates larger (respectively smaller) than the estimated cell rate of a congested switch will reduce (respectively increase) its rates.

7.3 Protocol of the Scheme

In the following, protocol of the scheme is outlined. To facilitate the description, a special type of RM cells defines: RM(CCR, ER). For each queue of a switch, a variable

MACR (Modified allowed Cell Rates) is defined and contains the value of the estimated optimal' cell rate.

7.3.1 A source transmits an RM (CCR, ER) for every N data cells transmuted, where ER is the maximum allowed cell rate of the source.

7.3.2 A source continually decrease its CCR

7.3.3 Upon receiving each RM (CCR,ER), a source increased its CCR by no greater than desired or Explicit Rate (ER).

7.3.4 A destination returns an RM (CCR, ER) cell upon receiving each RM (CCR)

7.3.5 Suppose a non-congested switch receives an RM (CCR, ER) cell from the source. It replaces MACR by MACR + (β* (CCR – MACR).

7.3.6 Suppose a congested switch receives an RM (CCR, ER) cell from the source. If CCR <MACR, it replaces MACR +(β* (CCR - MACR).

7.3.7 Suppose a congested switch receives an RM(CCR,ER) cell from the destination following actions are performed :

- If the current queue length is greater than DQT_1, it replaces ER in the RM cell by min

  (ER, γ_1 * MACR).

- If the current queue length is greater than DQT_2, it replaces ER in the RM cell by min

  (ER, γ_2 * MACR)

- If the current queue length is greater than DQT_3, it replaces ER in the RM cell by min

  (ER, γ_3 *MACR).

- Otherwise, if CCR>MACR, it replaces ER in the RM cell by mm

  (ER, MACR).

  Where $\gamma_3 > \gamma_2 > \gamma_1$

  $DQT_3 > DQT_2 > DQT_1$

  In this protocol if we increase one more threshold level i.e. DQT_4 and fast reduction ratio as γ_4, it will perform extremely fine because more checks better management.

7.4 Simulation Results

A switch is congested when the queue length of the switch increases over a period. The network configurations and the parameters used in simulations have been suggested by many of the simulation works presented at the ATM Forum and in the previous Congestion Control Schemes. In all of our simulations, we assume persistent sources, i.e. all sources will attempt to transmit cells at their highest possible speed. Although in practice most sources are not persistent, the fairness characteristics of a scheme can be better illustrated using persistent sources than using non-persistent sources. Moreover, by studying the transient behavior of a scheme for persistent sources, one can also understand its performance characteristics for non-persistent bursty sources.

Figures on the next pages illustrates the link utilization (including RM Cells) of VCs by applying intelligent congestion control scheme and shows the link utilization in case of proposed refined scheme. It is obvious, that our scheme converges to optimal cell

rate in the same convergence time without any cell loss and also displays much lesser oscillations as compared to Intelligent Congestion Control Scheme. Since an RM (CCR) cell is transmitted for every N data cells transmitted, the effective link utilization can be found through (l -1/N+1) or (CCR/Link Speed). Next graphical representations show the link utilization of individual VCs. A mark suppression of oscillations is also evident in case of proposed scheme.

7.5 **Software Analysis**.

The software developed in order to see the performance of both previous and proposed scheme has clearly proved the success of proposed scheme. Similar parameters were given to both the algorithms, their results showed current cell rate along with cell loss. Cell loss in case of proposed scheme was very less as compared to the previous scheme.

Conclusion

Traffic and congestion control is important in high-speed networks. Due to larger bandwidth-distance product, the amount of data lost due to simultaneous arrivals of bursts from multiple sources can be larger. For the success of ATM, it is important that it provides a good traffic management for both bursty and non-bursty sources.

This thesis proposes a congestion control scheme, which is a refinement of previous congestion control scheme. With the help of software, it has been proved that the refined scheme shows very less cell loss as compared to previous scheme. Simulation analysis further ensured the success of algorithm in the aspects such as, less oscillations, better traffic management, and convergence to optimal cell rate in the same convergence time.

REFERENCES

[1] Cisco Systems Inc. [1998], "Chapter 8: Designing ATM Internetworks", Internetwork Design Guide, Cisco Systems Inc.

http://www.cisco.com/univercd/cc/td/doc/cisintwk/idg4/index.htm

[2] Comer, D. E. [1995], *Internetworking with TCP/IP Volume I – Principles, Protocols, and Architecture*, 3rd edition, Prentice-Hall, Englewood Cliffs, New Jersey.

[3] Bertsekas, D. and R. Gallager [1992], *Data Networks*, 2nd edition, Prentice-Hall, Upper Saddle River, New Jersey.

[4] Clark, M. P. [1996], *ATM Networks: Principles and Use*, John Wiley, England.

[5] Alles, A. [May 1995], "ATM Internetworking", Cisco, San Jose, California.

[6] ATM Forum [June 1995], "ATM User-Network Interface Specification Version 3.1", Prentice-Hall, Englewood Cliffs, New Jersey.

[7] ATM Forum [July 1996], "ATM User-Network Interface Specification Version 4.0", ATM Forum: af-sig-0061.000.

[8] Colella, Callon, Gardner and Rekhter [May 1994], "Guidelines for OSI NSAP Allocation in the Internet", IETF Request for Comments: 1629.

[9] ATM Forum [March 1996], "Private Network-Network Interface Specification Version 1.0 (PNNI 1.0)", ATM Forum: af-pnni-0055.000.

[10] ATM Forum [December 1994], “Interim Inter-switch Signaling Protocol (IISP) Specification v1.0”, ATM Forum: af-pnni-0026.000.

[11] Chen, T. M. and S. S. Liu [1995], *ATM Switching Systems*, Artech House, Norwood, Massachusetts.

[12] ITU-T Recommendation I.362 [March 1993], “B-ISDN ATM Adaptation Layer (AAL) Functional Description”, Helsinki, Finland.

[13] Stallings, W. [1998], *High-Speed Networks: TCP/IP and ATM Design Principles*, Prentice-Hall, Upper Saddle River, New Jersey.

[14] Siu, K. and R. Jain [April 1995], “A Brief Overview of ATM: Protocol Layers, LAN Emulation, and Traffic Management”, *ACM SIGCOMM: Computer Communications Review (Special Issue on ATM)*, Vol. 25, No. 2.

[15] Kwok, T. [1998], *ATM: The New Paradigm for Internet, Intranet, and Residential Broadband Services and Applications*, Prentice-Hall, Upper Saddle River, New Jersey.

[16] Partridge, C. [1994], *Gigabit Networking*, Addison-Wesley, Reading, Massachusetts.

[17] Heinanen, J. [July 1993], “Multiprotocol Encapsulation over ATM Adaptation Layer 5”, IETF Request for Comments: 1483, Telecom Finland.

[18] ITU-T Recommendation Q.2100 [July 1994], “B-ISDN Signaling ATM Adaptation Layer (SAAL) Overview Description”, Geneva, Switzerland.

[19] ITU-T Recommendation Q.2110 [July 1994], “B-ISDN ATM Adaptation Layer – Service Specific Connection Oriented Protocol (SSCOP)”, Geneva, Switzerland.

[20] ITU-T Recommendation Q.2130 [July 1994], “B-ISDN Signaling ATM Adaptation Layer – Service Specific Coordination Function for Support of Signaling at the User Network Interface (SSCF at UNI)”, Geneva, Switzerland.

[21] The ATM Forum Technical Committee, “Traffic Management Specification Version 4.0” Apr 1996.

[22] C. Fang, H. Chen, and J. Hutchins, “A simulated Study of TCI~ Performance in ATM Networks” ATM Forum Contributions, Jan 1994.

[23] Aromanow and S. Floyed, “Dynamics of TCP traffic over ATM Networks”, presented to ATM Forum, Jan 1994.

[24] Kai-Yeungn Siu and Iiong-Yi Tzeng, “Intelligent Congestion Control for ABR Service in ATM Networks”,

[25] David E. MacDysan, Darren L. Spohn “ATM Theory and Application” McGraw-Hill International Editions.

[26] Andrew S. Tanenbaum” domputer Networks” Prentice Hall International Editions.

[27] Raif 0. Onvural “Asynchronous Transfer Mode Networks” Artech [louse, INC [8]

[28] Mark I. Williams “ATM - What does it Mean?”

[29] Fang Lu “ATM Congestion Control”

[30] Raj Jam “Congestion Control and Traffic Management in ATM Networks: Recent Advances and A Survey”

[31] A. Charny “An Algorithm for Rate Allocation in Cell-Switching Network with Feed back”

[32] Peter Newman, “Backward Explicit Congestion Notification for ATM Local Area Networks”, IEEE 1993.

[33] Raj Jam, ~Congestion Control and Traffic Management in ATM Networks, recent advances and a survey”, IEEE August 1996.

[34] Nmrwan Ansari, “Allocating Fare Rates for Available Bit Rate Service in ATM Networks”, IEEE November 1996.

[35] Flavio Bonomi and Kerry W. Fendick, “The Rate Based Flow Control Framework for the Available Bit Rate ATM Service”, IEEE Network, March/April 1995.

[36] Kerry W. Fendick, “Evaluation of Controls for Available Bit Rate Service”, IEEE Communications, November 1996.

[37] A. W. Barnhart “Baseline Model for Rate-Control Simulation”, ATM Forum Contribution May 1994.

[38] F. Bonomi, K.W. Fendick, and K. Meier-Iiellstern, “A Comparative Study of EPRCA Compatible Intelligent Congestion Indication Schemes for the Support of Fair ABR Service”, ATM Forum Contribution Sep1994.