INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Estelle Verification of

ATM Available Bit Rate (ABR) Control Protocol

Weiyuan Huang

A Major Report
in
The Department
of
Computer Science

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Computer Science at
Concordia University
Montreal, Quebec, Canada

March 2001

Abstract

Estelle Verification of ATM Available Bit Rate (ABR) Control Protocol

Weiyuan Huang

This major report specifies and verifies the ATM Available Bit Rate (ABR) control protocol using a standard network specification language, Estelle. This verification is based on the English description of ABR protocol in the ATM Traffic Management Specification Version 4.0, and the published PEFSM model of ABR protocol.

The Estelle simulation model that we used to verify the ATM ABR protocol consists of two identical module instances of network station module. Connections between the two station nodes make a closed-loop feedback control system. We have defined Estelle observers to observe the behaivor of the ABR protocol.

Seven test cases have been designed and carried out. The results of all these test cases have shown that the protocol meets the functional specification for ABR protocol outlined in the ATM Traffic Management Specification.

Acknowledgments

I would like to take this opportunity to express my sincerest thanks to my supervisor, Dr. J. W. Atwood, for his advice, encouragement, and practical assistance during the course of this major report.

My special thanks go to Ms. Halina Monkiewicz, the secretary of Graduate Office in the Department of Computer Science, for her kindly assistance through out my study period at Concordia.

I also wish to thank all faculty members and staffs in the Department of Computer Science at Concordia University, for their many forms of help I received during my study.

Table of Contents

List of Figures

Chapter 1

Introduction

The ATM ABR protocol is one of the five service categories that an ATM layer provides. It tries to make unused network bandwidth available to end users, and guarantees a minimum cell loss rate and a minimum cell transmission rate.

The ABR protocol is first specified using an English description in the main body of the draft Traffic Management Specification 4.0 [1]. Lee *et. al.* have specified the protocol in Parameterized Extended Finite State Machines (PEFSM) in more detail [2]. Based on the available PEFSM model, we specified and verified the ABR protocol using a standard network specification language, Estelle.

In section 2, we give a brief introduction of concepts of ABR protocol and the Parameterized Extended Finite State Machines (PEFSM). Section 3 discusses detailed design and implementation concerns of ABR in Estelle specification language. Test cases and results are introduced in section 4. We give our conclusion in section 5.

Chapter 2

ABR Protocol

2.1 ATM and ABR

Fundamentally, ATM technology simultaneously transmits data, voice, and video traffic over high bandwidth circuits. ATM hardware and software platforms form a communications architecture based on the switching and relaying of small units of data, called cells. The primary distinction between ATM-based services and other existing data communications services is that ATM is the first technology and protocol structure to effectively integrate voice, data, and video over the same communications channel at any speed. This is possible due to the fact that, with ATM technology the network establishes a separate traffic contract with the user of each ATM Virtual Path Connection (VPC) or Virtual Channel Connection (VCC). This traffic contract is an agreement between a user and a network regarding the Quality of Service (QoS) that the network has guaranteed to provide.

The ATM Forum Traffic Management 4.0 specification defines ATM layer service categories as following:

- CBR Constant Bit Rate
- rt-VBR real-time Variable Bit Rate
- nrt-VBR non-real-time Variable Bit Rate
- UBR Unspecified Bit Rate
- ABR Available Bit Rate

The ABR service category defines a closed loop flow controlled service that uses feedback to achieve a specified loss objective if the user conforms to a traffic parameter defined by the Allowed Cell Rate (ACR). That is, if sources conform to the rules of ABR, then the network guarantees a Minimum Cell Rate (MCR) with minimum cell loss. The goal for the network is to make unused bandwidth available to cooperating end users in a fair, timely manner.

2.2 ABR Closed-Loop Flow Control

There are several variants to the possible congestion control algorithms that may be adopted in the network. The ATM Traffic Management Specification Version 4.0 (which we use in this verification) includes the basic two ABR modes that ATM uses to achieve fairly arbitrated congestion control [3].

ABR's binary mode is like the green/red lights at the entrance to congested freeways. ATM switches indicate congestion using the EFCI bit in the ATM cell header in the forward direction. The destination end system employs a Congestion

Indication (CI) field in a Resource Management (RM) cell to communicate the presence of congestion back to the source, which makes green/red light type decisions regarding the source's current transmission rate. The recipient of the CI bit does not stop sending, but instead reduces its transmission rate by a fraction of the currently available cell rate.

The Explicit Rate (ER) mode of ABR operation adds another degree of complexity. Similar to the manner in which air traffic controllers control the speed of multiple airplanes (cells) converging on a crowded airport, ATM switches along the path of the end-to-end connection communicate an explicit rate for each source. In this way, controller (RM cells) throttle back fast planes (user cells) during periods of congestion to yield a regular arrival pattern at the congested airport (interface).

2.3 The Specification and the Parameterized Extended Finite State Machines

The specification for the ABR service is primarily that of the source and destination policies, while the specification of the operation of the intermediate nodes (switches) has been specified somewhat more loosely. In broad terms, the ABR scheme has sources transmitting data cells at a specified rate, derived using a feedback control mechanism by which the network communicates its state. Periodically sources transmit an RM cell that serves the function of a probe into

the network to detect the state of the network (congested or otherwise). The RM cell also serves as a cell in which the network may communicate an ER (explicit rate) back to the source to indicate the rate at which the source is permitted to transmit. RM cells sent by a source are received by the remote destination and have to be turned around and sent back. This function is performed by the destination part of the ABR specification.

The source and destination policies have been specified using an English description in the main body of the ATM Traffic Management Specification version 4.0 [1]. It is impossible to conduct an automatic verification/testing based on this English specification for such a complicated protocol. Furthermore, while considerable energy has been spent in providing a reasonably precise specification, an English description may sometimes lead to implementations that do not meet the letter and/or the spirit of the specification.

Thanks to several researchers from Bell and AT&T, they have provided us the Parameterized Extended Finite State Machines (PEFSM) of the ABR protocol based on the specification. Figure 1 shows an overview of the protocol state machines and cell flow between source and destination stations [2]. As in a Finite State Machine there are states and transitions between states. In addition, there are variables and predicates (events). A transition is executable if the associated predicate is TRUE with the current variable values and in this case the action updates the variable values and machine moves to the next state. On the other

hand, there are parameters associated with the input/output cells (messages) to/from the machines and they affect the execution of the transitions and the variable values.

Our simulation model is basically an Estelle implementation of these PEFSM machines.

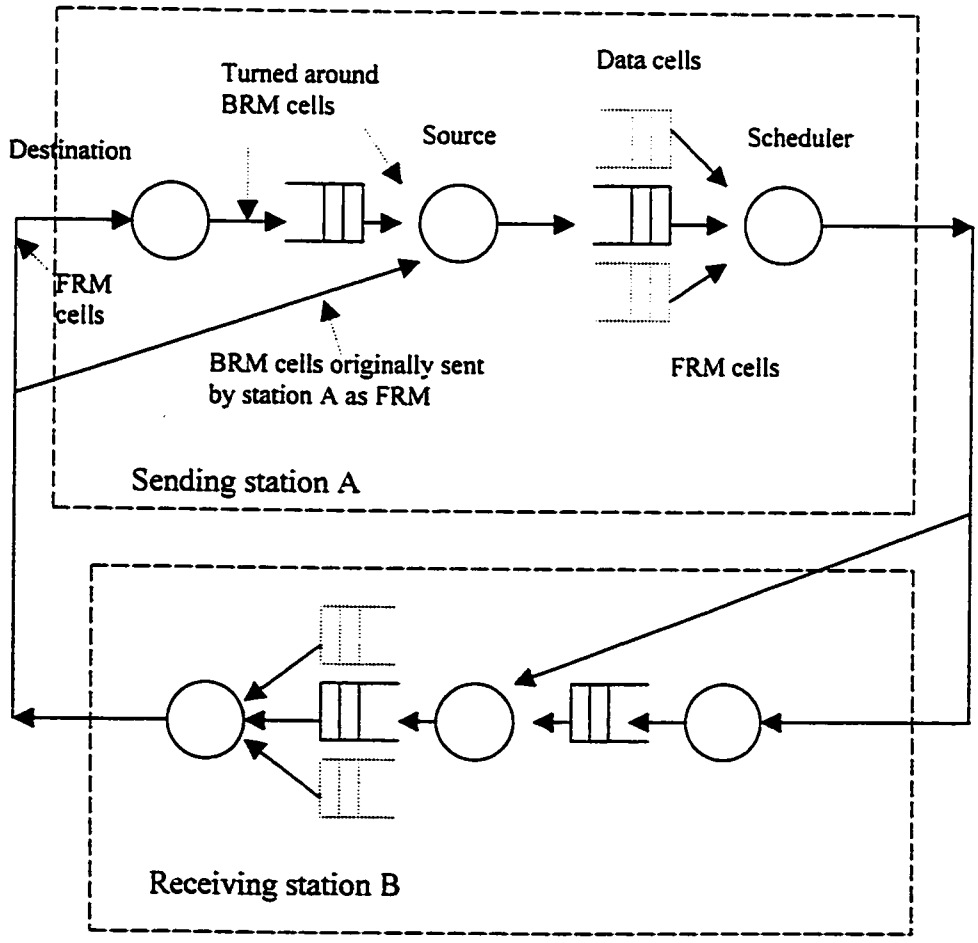


Figure 1: Overview of ABR PEFSM machines and cell flow[2]

Chapter 3

The Simulation Model

3.1 Basic Estelle Concepts

Estelle is an ISO standard specification language for communication protocols [4]. It defines a set of modules for network entities or encapsulated tasks, and a set of bi-directional channels as the communication media between these modules. Two modules use one or more channels to communicate with each other. The interface between a module and the channel is called an Interaction Point (IP). An interaction is the message one module sends to the other through a specified IP. A module defines a set of variables, a set of states and a set of transitions. At a given time, the module is in one particular state. It goes into another state when a set of pre-defined events occurs.

The definition of a module contains two major parts: the header and the body. The header of a module defines some public information about the module, which is needed for other modules to communicate with it. It is responsible for declaring

the module's type, the IP's, the rules of each IP for a specific channel (i.e., input or output), and a set of shared variables. The body of a module defines the private part of the module. It contains the definition of states, transitions, and variables. The body may also contain the definition of another module(s). In this case, the nested module is considered to be a child module. A child module is usually an encapsulated task of its parent module, and it may share its parent IP to communicate with the rest of the world.

3.2 The ABR Module

Our ABR module consists of two identical module instances of the Station module, each responsible for sending and receiving data and management cells to/from the other one. Connections (Channels) between the two station nodes make a closed-loop feedback control system.

The functionality of a station has been broken down to three sub-machines: the source machine, the destination machine, and the scheduler machine. Two children modules of the station module take responsibility of destination and scheduler machines respectively, and the station module itself implements the functionality of the source machine. The reason is that, in the PEFSM of ABR there are several variables shared either between the source machine and the destination machine, or between the source machine and the scheduler machine.

In Estelle, parent and child module relationship is the only way to share a common variable between two different modules.

An implementation issue has to be addressed before we go into details of the simulation model. It is concerned with the in-rate and out-of-rate cells. In the ATM Traffic Management Specification Version 4.0 [1], under some situations RM cells may be sent out-of-rate. One use of out-of-rate RM-cells is to enable a rate increase for a connection that has an ACR of zero. The source would use the out-of-rate cells to learn when it may increase its rate. However, these issues are implementation dependent, which have been elected to ignore by the designers of the PEFSM machine. So we are primarily concerned with simulating the actions that are specified when transmitting cells "in-rate". This means that any time a cell is ready for transmission, it is allowed to be transmitted only after a minimum time of (1/ACR) has elapsed since the transmission of the previous cell from the same network node (station).

3.2.1 Channels

Messages are exchanged through the use of channels between each module. Even between the parent module and its child module, an internal channel has to be defined before any messages can be passed to/from each other. The definition of a channel consists of the name of the channel, two opposite roles that act like

message input and output respectively, and a set of predefined types of messages that are going to be sent out by the specific role which follows the "by" clause.

Four channels are defined for the ABR module. Channel FD and B are defined so that FRM, Data, and BRM cells can be exchanged between two station modules. Channel D_S is defined to pass Turned-around BRM (TBRM) cells from Destination sub-machine to its parent Station module. Channel S_S is defined to pass Turned-around BRM (TBRM) cells from Station module to its Scheduler sub-machine.

CHANNEL FD(FDOUT, FDIN);

by FDOUT:

FRM(*DIR: integer; BN: integer; CI: integer; NI: integer; ER: real; CCR: real; MCR: real*);

Data(*EFCI: integer*);

CHANNEL B(BOUT, BIN);

by BOUT:

BRM(*DIR: integer; BN: integer; CI: integer; NI: integer; ER: real; CCR: real; MCR: real*);

CHANNEL D_S(TBOUT, TBIN);

by TBOUT:

TBRM(*DIR: integer; BN: integer; CI: integer; NI: integer; ER: real; CCR: real; MCR: real*);

CHANNEL S_S(TBOUT, TBIN);

by TBOUT:

TBRM(*DIR: integer; BN: integer; CI: integer; NI: integer; ER: real; CCR: real; MCR: real*);

In FRM, BRM and TBRM cells, the following parameters are of interest to the ABR protocol:

- DIR: Direction of the cell, 0 if forward, 1 if backward.
- BN: 0 if the cell is source generated, 1 if destination or switch generated.
- CI: Congestion Indication, 0 if no congestion, 1 otherwise.
- NI: No (rate) Increase. 0 if cell rate can be increased, 1 otherwise.
- ER: Explicit (cell) Rate.
- CCR: Current Cell Rate.
- MCR: Minimum Cell Rate.

The only interesting parameter to ABR in a Data cell is EFCI.

- EFCI: Explicit Forward Congestion Indication. 0 if no congestion, 1 otherwise.

3.2.2. Destination Machine

According to the ABR protocol, when a data cell is received, its EFCI value is saved as the EFCI value of the connection. On receiving a forward RM cell (FRM), the destination station should set parameter values of this RM cell (we call it TBRM cell, as it is now being turned to be a BRM cell), and to return this RM cell to the source station as backward RM cell (BRM).

In our ABR module, one station consists of a Station module and two sub-modules: a Destination and a Scheduler. The basic task for the Destination module is to set EFCI value of the connection upon receiving a data cell, and to set parameter values of the TBRM cell reflecting the connection condition when a FRM cell is received. If an internal congestion within the station occurs, the Destination gets this information (ici=1), and can generate a TBRM cell without getting a FRM cell. After the parameters have been set properly, the Destination then sends the TBRM cell (either source generated or destination generated) to its parent module, the Station module. The Station module sends the TBRM cell to its Scheduler, and the Scheduler sends the TBRM cell to the source station as a BRM cell.

In the ATM specification it is stated that, when a new FRM cell is being turned around while there are old TBRM cells queued in the Scheduler (from Station, which in turn is from Destination machine), there are two alternatives: (1) to overwrite the contents of all old TBRM cells with the new information in the new TBRM cell, and queue the new TBRM cell in the Scheduler. Or (2) to discard all old TBRM cells previously queued in the Scheduler, and queue this TBRM cell for transmission. Instead of keeping track of individual RM cells queued to the Scheduler, the ABR PEFSM designers decided to maintain a simple counter of the number of TBRM cells queued to the Station (which in turn queued to the Scheduler) by the Destination machine, and a single copy of the contents of the TBRM cell queued. This is because all the TBRM cells queued at the Scheduler are either dropped or updated with the latest FRM (the new TBRM) cell information received by the Destination machine.

Two interaction points have to be defined for the Destination module. One is "din" which acts as the receiver's role FDIN of channel FD to receive FRM and Data cells from another source station. The other one is "dout" which acts as the sender role TBOUT of the channel D_S, to send the TBRM cell to its Station module.

The Destination module keeps track of the number of TBRM cells turned around and queued to the Scheduler (through its parent Station module). Since this

variable is shared with its parent Station module, we must declare it as an "export" variable.

The Destination machine can be in two different states:

- S0: in this state, the Destination is getting FRM, Data cells, or receiving Internal Congestion Indication (ici). Upon receiving a FRM cell or getting internal congestion indication, the Destination machine goes to S1 state.
- S1: in this state, the Destination machine is able to send out a TBRM cell to its parent Station module. After the TBRM cell is sent, the Destination machine goes back to state S0.

Fourteen transitions have been defined for the Destination machine.

Transition 1 (implements the Destination rule 1 of the ABR protocol defined in ATM Traffic Management Specification Version 4.0): in S0, the Destination gets a Data cell, and saves the EFCI value of the data cell as the EFCI value of the connection.

Transition 2 (implements Destination rule 2a): in S0, the Destination gets a FRM cell when there is no internal congestion and the saved EFCI is 0. It sets the next TBRM cell to be a source generated, backward RM cell by indicating "bn=0" and "dir=1". It also tells the source station that cell rate can be increased by setting "ci=0" and "ni=0". The Destination machine then goes into S1 state where it

generates the TBRM cell using these variable values, and sends the TBRM cell to its parent Station module.

Transition 3 (implements Destination rule2a): in S0, the Destination gets a FRM cell when there is no internal congestion, but congestion occurred in the connection (saved EFCI=1). It lets the source station be aware of the occurrence of congestion by setting "ci=1" for the next TBRM cell. Then it goes to S1.

Transition 4 (implements Destination rule 2b for ER mode): in S0. the Destination gets an FRM cell when there is internal congestion, but there is no congestion in the connection. It reduces the ER (Explicit Rate) by a certain amount. This transition is only for ABR run in ER mode.

Transition 5 (implements Destination rule 2b for non-ER mode): in S0, the Destination gets an FRM cell when there is internal congestion, but there is no congestion in the connection. It prevents the source station from increasing the cell rate by setting "ni=1" for the next TBRM cell.

Transition 6 (implements Destination rule 2b for non-ER mode): in S0, the Destination gets a FRM cell when there is internal congestion, but there is no congestion in the connection. It lets the source station be aware of the occurrence of congestion by setting "ci=1" for the next TBRM cell.

Transition 7 (implements Destination 2b for ER mode): in S0, the Destination gets a FRM cell when there is internal congestion, but there is no congestion in the connection. It lets the source station be aware of the occurrence of congestion by setting "ci=1" for the next TBRM cell. In addition to that, the Destination also reduces the ER value by a certain amount. This transition is only for ER mode.

Transition 8 (implements Destination rule 2b for non-ER mode): in S0, the Destination gets a FRM cell when congestion has occurred both internal and in the connection. It lets the source station be aware of the occurrence of congestion by setting "ci=1" for the next TBRM cell.

Transition 9 (implements Destination rule 5): in S0, when there is internal congestion indication without getting an FRM cell, the Destination voluntarily starts to generate a BRM cell to inform the source station of the occurrence of congestion by setting "ci=1" and "bn=1".

Transition 10 (implements Destination rule 5): in S0, when there is internal congestion indication without getting an FRM cell, the Destination voluntarily starts to generate a BRM cell to prevent the source from increasing the cell rate, by setting "ni=1" and "bn=1".

Transition 11 (implements Destination rule 3a, implicit for BRM rewriting): in S1, the Destination sends out the TBRM cell to Station module, and lets the

values of the parameters in this TBRM cell over write the values of TBRM cells previously queued in the Scheduler module.

Transition 12 (implements Destination rule 3b, implicit for BRM dropping): in S1, the Destination sends out the TBRM cell to Station module, and drops all other TBRM cells previously queued in Scheduler module. The dropping is done by setting the value of N be 1. (N is the number of TBRM cells queued to Scheduler, a shared variable with Station module).

Transition 13 (implements Destination rule 5): in S1, the Destination sends out the voluntarily generated TBRM cell to the Station, with all the values of parameters reflecting the condition of the connection.

Transition 14 (Destination implicit): in S1, when there no previously queued TBRM cells in the Scheduler (indicated by N=0), the Destination simply sends out the TBRM cell to Station.

The following figure shows a summary of transitions and states of the Destination module.

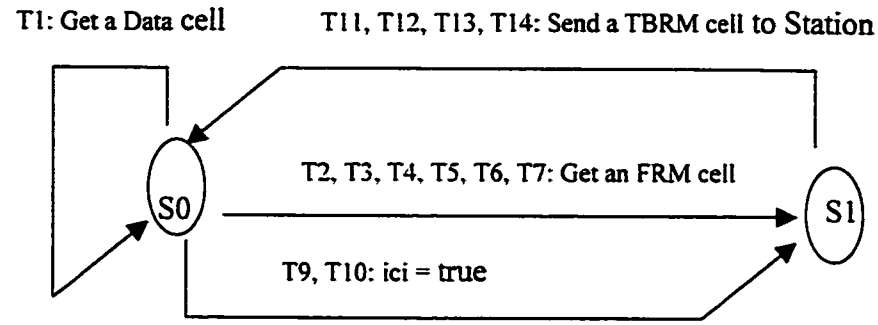


Figure 2: Destination State Machine

3.2.3 Scheduler Machine

The Scheduler machine is another child module of the Station module. Its basic task is to act as an output interface to other peer stations. All cells (FRM, BRM, and Data cells) are sent through the Scheduler machine in a timing constraint.

Three types of cells are sent through the Scheduler machine. The Scheduler receives TBRM cells from its parent Station module, and sends them out as BRM cells. FRM cells are generated by the Scheduler when it is needed according to the timing constraint defined in the ATM ABR specification. Since the actual user data in the Data cells are not critical to our ABR simulation model, the Data cells generated by the Scheduler only contain one parameter EFCI, which is used to indicate the occurrence of congestion in the connection. A variable "d_ready" has

been defined to inform the Scheduler whether there are data cells waiting to be sent. Under the timing constraint which is visualized by the variable "Allowed Cell Rate (ACR)", the Scheduler generates and sends a Data cell out whenever the "d_ready" is set.

The Scheduler machine needs three interaction points to be defined. One is "sin" which acts as the receiver's role TBIN of channel S_S, to receive TBRM cells from Station module. The other two, "sfdout" and "sbout", are output roles of FD and B channels respectively. "sfdout" sends out FRM and Data cells, and the "sbout" sends out BRM cells, to another station.

Four variables are shared with the Station module:

- X: the number of BRM cells queued to Scheduler by Station.
- Y: the number of FRM cells sent by the Scheduler since last BRM cell with BN=0 received by Station
- ACR: Allowed Cell Rate.
- d_ready: indication of data cells waiting to be sent.

Among these, ACR is the most interesting variable to our ABR model. It is this variable that determines the cell rate at which all in-rate cells should be sent out from this station. There are two constant variables, PCR (Peak Cell Rate) and MCR (Minimum Cell Rate), which are set when the connection is established. The value of ACR is being adjusted between PCR and MCR, according to the

internal and external condition of the connection. After ACR has been changed, the new value is set to CCR (Current Cell Rate), which is a parameter of the RM cells, and to be passed through the connection to other stations.

The Scheduler has three different states: S0, S1, and S2.

- S0: the state in which the Scheduler is initialized. The machine goes to S1 after the first FRM cell is sent out.
- S1: active state in which the Scheduler is able to receive and send out cells. If the value of ACR has to be adjusted, then the machine goes to state S2 before sends the cell out.
- S2: the state in which rate maybe changed.

Eleven transitions have been defined for the Scheduler machine.

Transition 1 (initial setup, implements the Source rule 2 in the ATM ABR specification): the Scheduler sends out the first FRM cell, with the value of ACR being set to the value of ICR (Initial Cell Rate, a constant variable set at the time when the connection is established). After the first FRM cell is sent out, the Scheduler goes to S1.

Transition 2 and 3 (implements Source rule 3a and 5): in S1, the Scheduler sends out a FRM cell, when either "at least Mrm in-rate cells have been sent and at least Trm time has elapsed" or "Nrm-1 in-rate cells have been sent". If ACR<ICR and

the time elapsed since the last in-rate FRM cell was sent is NOT greater than the ADFT, which is the case of Transition 2, the Scheduler sets the value of CCR to be the value of ACR before sending the FRM cell. Otherwise the Scheduler first sets ACR to be the value of ICR before sending the FRM cell, which is the case of Transition 3.

Transition 4 and 5 (implements Source rule 3a, 5, and 6): in S2, the Scheduler adjusts the value of ACR before sending the FRM cell. The adjustment is based on whether or not there have been CRM (Missing RM cell count, a constant value defined when the connection is established) in-rate FRM cells sent out since the last BRM cell with BN=0 was received.

Transition 6 (implements Source rule 3b): in S1, the Scheduler determines that it is time to send out a BRM cell, since all the following requirements are met: (1) it is not a time to send a FRM cell, (2) there is no Data cell waiting to be sent, and (3) sending the BRM cell will not exceed the cell rate constraint defined by ACR.

Transition 7 (implements Source rule 3c): in S1, the Scheduler determines that it is time to send next Data cell.

Transition 8 (implicit): simply increase the timer for the Scheduler if there is no cell to be sent.

Transition 9, 10, and 11 (implicit): in any state, if the Scheduler receives a TBRM cell, it records the values of the parameters of the TBRM cell. These values then can be used to set the values of parameters of next FRM or BRM cell, reflecting the change of condition of the connection.

Figure 3 shows the transitions and states of the Scheduler machine.

3.2.4 Station Module

The Station module is the parent module of both Destination and Scheduler sub-machines. It acts as one node (or switch) in the network connection in our ABR simulation model.

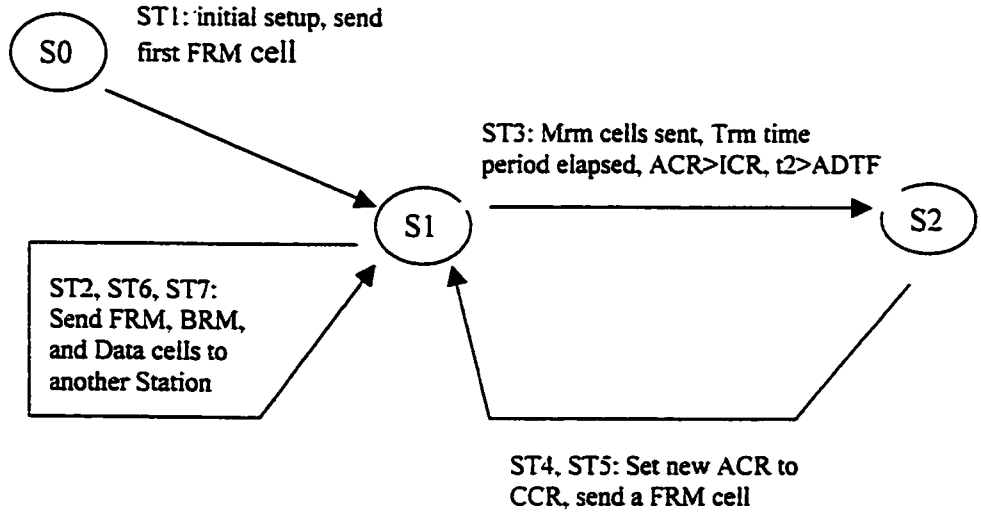


FIGURE 3: Scheduler State Machine

There are four external and two internal interaction points for the Station module.

- infd: the interaction point that receives FRM and Data cells from other stations (Nodes) in the network, and passes these cells directly to its Destination sub-machine.
- inb: the interaction point that receives BRM cells from other stations. The Station machine reads the information in the BRM cell, and adjusts the value of ACR variable of the Scheduler sub-machine. The Station module can alter the value of ACR because it is a shared variable between these two modules.
- outfd: the interaction point that the Station used to send FRM and Data cells to other stations (Nodes) in the network. This IP is directly attached to the "sfdout" IP of the Scheduler sub-machine.
- outb: the interaction point that the Station used to send BRM cell to other stations. This is also directly attached to the "sbout" IP of the Scheduler sub-machine.
- intb: an internal interaction point that the Station module used to connect to its child module, the Destination machine, to receive the TBRM cells from the later.
- outtb: an internal interaction point that the Station module used to connect to its child module, the Scheduler machine, to pass the TBRM cells to the later.

There are four states of the Station module.

- S0: the initial state in that the variables are initialized. When there is an indication that data cells are waiting to be sent, the Station goes to S1.
- S1: active state in that the Station receives BRM and TBRM cells.
- S2: when a BRM cell is received in S1, the Station goes into S2 and adjusts the value of the shared variable ACR. The new ACR then determines the cell rate at which the Scheduler machine can send cells out.
- S3: when a TBRM cell is received in S1, the Station goes into the S3, and adjusts the queued BRM cell in the Scheduler by dropping or rewriting those BRM cells accordingly.

We also need to define two module variables, "desti" and "schdu", for the Destination and the Scheduler machine respectively.

For the Station module, there are twelve transitions among those four states.

Transition 1 (implements source rule 2 of the ATM ABR specification): the Station sets ACR to the value of ICR (Initial Cell Rate), and goes to the active state S1.

Transition 2 and 3 (source rule 8): in S1, the Station module receives a BRM cell from another station (network node), and records the values of the parameters of the BRM cell. If the BRM cell is actually an FRM cell that was previously sent

out by this station (indicated by BN=0, which is the situation of transition 3), then the Station re-sets the value of Y to be 0 (which is the number of FRM cells sent out by the Scheduler since the last BRM cell with BN=0 was received).

Transition 4, 5, 6, and 7 (source rule 8 and 9): after a BRM cell has been received and the information has been recorded is S1, the Station now is in state S2. The Station adjusts the value of ACR based on the information from the BRM cell in the following ways: (1) if CI=1, then the ACR shall be reduced by at least ACR*RDF, which is the situation of Transition 4; (2) if both CI=0 and NI=0, then the ACR can be increased by no more than RIF*PCR to a rate not greater than PCR, which is the situation of Transition 5; (3) if CI=0 and ACR>=ER, then the ACR shall be set to the same as ER, which is the situation of Transition 7; (4) if CI=0, NI=1, and ACR<ER, then nothing needs to be changed, which is the situation of Transition 6. However, in all the above transitions, the final value of ACR should be not greater than ER and not smaller than MCR.

Transition 8 (source rule implicit): in S1, the Station receives a TBRM cell from its Destination sub-machine, and records the information in the cell to local variables. The Station module then goes into state S3 and queue the TBRM cell to its Scheduler sub-machine.

Transition 9 (source rule 3): in S3, when there are no other TBRM cells queued to the Scheduler machine (indicated by X=0), then the Station simply queues this TBRM cell to the Scheduler.

Transition 10 and 11 (source rule 3): in S3, when the new arrival TBRM cell is source generated (indicated by BN=0) and there are other TBRM cells queued to the Scheduler, then the Station module has two alternatives chosen randomly. (1) with transition 10, the Station drops all old TBRM cells previously queued to the Scheduler, and sends this new TBRM cell to the Scheduler. (2) with transition 11, the Station overwrites all the TBRM cells previously queued to the Scheduler with new values obtained from this new TBRM cell, and queues this new TBRM cell to the Scheduler.

Transition 12 (source rule 3): in S3, when the new arrival TBRM cell is not source generated (indicated by BN=1), meaning that this Station voluntarily generated this TBRM cell to report occurrence of congestion to another station (a network node), the Station queues this TBRM cell to the Scheduler machine.

Figure 4 shows the summary of transitions and states of the Station module.

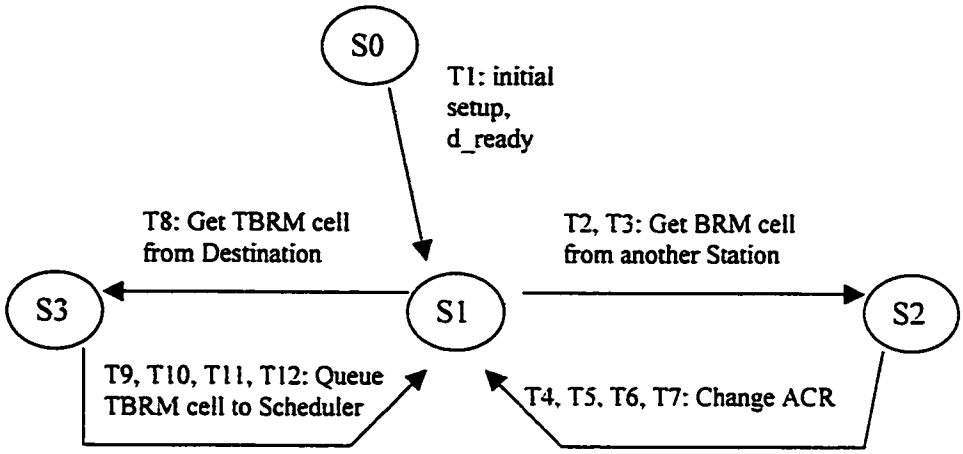
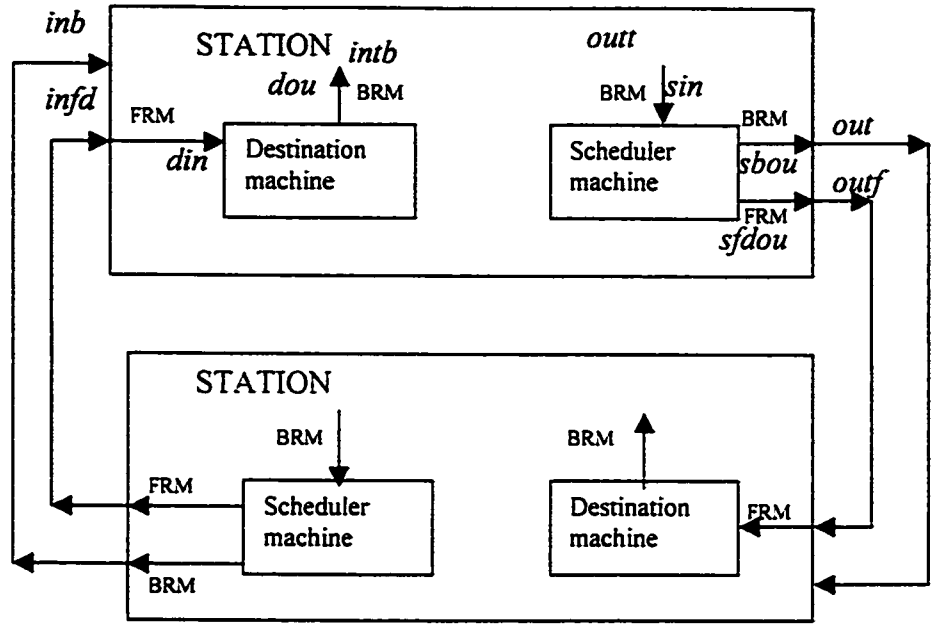


FIGURE 4: Station State Machine

3.2.5 The ABR Simulation Module

After all necessary components have been defined, we declare two network stations using the header definition Station_h module, "node_a" and "node_b", initialize these nodes with the body definition Station_b, and make connections between these to nodes.

Figure 5 shows the structure of the ABR simulation model.



infd ------- interaction point FRM ------- interaction

FIGURE 5: ABR Simulation Model

Chapter 4

Test Cases and Results

As we have pointed, the basic purpose of ABR is to make unused bandwidth available to cooperating end users in a fair, timely manner. This goal is achieved by changing the value of ACR for a particular network source node, and only allowing the source node to send cells at the rate not greater than the ACR. Since the ACR is adjusted within the range between PCR and MCR, if sources conform to the rules of ABR, then the network guarantees a Minimum Cell Rate (MCR) with minimum cell loss.

Based on this assumption, our test cases are designed to make sure if the goal can be achieved by the ABR simulation model. To do that, we have to observe the execution of the model in two different ways: run and observe the ABR model as a black box and as a white box. To observe the model as a black box means that, we only focus on whether the expected results (for the ABR, it is the change of value of ACR) can be obtained by execution of the model. Observing the model as a white box means that, we want to make sure that the model achieves the expected goal not in an arbitrary way but only in the specific way that we have modeled it.

The value of ACR is adjusted to reflect the occurrence of congestion. If there is no congestion and the ACR is smaller than the PCR, then the value of ACR can be increased gradually, allowing the node to send cells at a higher speed. If there is congestion, then the ACR will be reduced, forcing the node to slow down sending cells. To test these, we have the following two test cases:

- TC1: Title ---Execution of ABR in non-congestion condition

  Objective ---Observe the increase of ACR
- TC2: Title ---Execution of ABR in congestion condition

  Objective ---Observe the decrease of ACR

For white box test cases, we focus on the internal mechanism of adjusting the value of ACR. Examining the ATM ABR specification, we observe that a network source node periodically sends out a FRM cell to probe the network condition, and adjusts the value of ACR in one or in a combination of the following ways, according to the information received from the BRM cell and/or the EFCI value of a Data cell. (1) Reduces the ACR to ICR; (2) reduces the ACR by ACR*CDF; (3) reduces the ACR by ACR*RDF; (4) increases the ACR by PCR*RIF. These situations are covered by the following test cases:

- TC3: Title ---Sending a FRM cell

  Objective ---Observe the situation in which a FRM cell is allowed to be sent out.
- TC4: Title ---Reduce ACR to ICR

Objective --- Observe the situation in which ACR must be reduced to the value of ICR.

- TC5: Title ---Reduce ACR by ACR*CDF

  Objective ---Observe the situation in which ACR is reduced by ACR*CDF.

- TC6: Title ---Increase ACR by PCR*RIF

  Objective ---Observe the situation in which ACR is increased by PCR*RIF.

- TC7: Title ---Reduce ACR by ACR*RDF

  Objective ---Observe the situation in which ACR is reduced by ACR*RDF.

4.1 TC1: Execution of ABR in non-congestion condition

In our simulation model, we have defined the variable "ici" (Internal Congestion Indication) to allow us control the occurrence of congestion in the model. The variable is within the Destination sub-machine of each network node (the Station). When the "ici" is set to true (or false), it means there is (there is not) internal congestion occurring in this station.

We set our first test case by simply setting the "ici" in both stations to be false, and run the following commands:

> so { d 4->ACR}

> $fs:=50

> continue

The first command "so {d 4->ACR}" sets a simple edb observer to display the content of ACR at each time a transition is completed. Please note that the digit value 4 is the module instance number of the Scheduler sub-machine of one Station module in the ABR model, the other Scheduler sub-machine is numbered by 6. The second command sets the number of transitions to be executed be 50. Thus, each time we invoke the "continue" command, a total of 50 transitions will be executed.

The test case is passed if ACR is increased gradually and finally reaches (but does not exceed) the maximum value that equals the value of PCR. It fails otherwise.

4.2 TC2: Execution of ABR in congestion condition

Test case 2 is similar to TC1, but with a true value of "ici" to simulate the situation when congestion does occur. We can either set the "ici" in one or both of the station instances. To set the value while the edb is running, we can invoke the command "6->ici:=true", that sets the ici of module number 6 to be true.

The following commands invokes TC2 in that we only make congestion occur in the receiving station:

> so { d 4->ACR}

> 6->ici:=true

> $fs:=50

> continue

The test case is passed if ACR is decreased gradually, and finally reaches (but does not go below) the minimum value that equals the value of MCR. It fails otherwise.

It is also possible to combine TC1 and TC2 to manipulate the value of ici and to observe the change of direction in which ACR is being adjusted.

4.3 TC3: Sending an FRM cell

FRM (which stands for forward RM) cells are used to probe the network condition by a source network node. An FRM cell previously sent out is received by the same network node as a BRM cell that carries information about the network. The ATM ABR specification defines the situation in which a source node is allowed to send out an FRM cell as the following.

The next in-rate cell shall be a forward RM-cell if and only if, since the last in-rate forward RM-cell was sent, either:

(i) at least Mrm in-rate cells have been sent and at least Trm time has elapsed, or

(ii) (ii) Nrm-1 in-rate cells have been sent.

To verify that our ABR model does act in the way specified in the ATM ABR specification, we need keep track of two variable values. One is the number of cells that were sent out since the last FRM cell was sent, the other is the time elapsed since the last FRM cell was sent.

In the Scheduler sub-machine, we have defined X1 be the number of BRM cells transmitted since the last FRM cell, and X2 be the number of Data cells transmitted since the last FRM cell. Thus, X1 + X2 will give us the total number of cells sent since the last FRM cell was sent. We have also defined t2 be the time units elapsed since the last FRM cell was sent. After the first cell (which is an FRM cell) has been sent, three transitions may send out FRM cells. So, what TC3 basically does is to show us both the values of X1+X2 and t2 whenever one of these three transitions is executed.

We define the following edb observer "case3" to do the above task:

```
d 4->X1; d 4->X2; d 4->t1; d 4->t2; \
```

```
d 4->ACR ; d 4->X ; d 4->Y ; \
if ($nbuse(4->1) = #epdst2) then \
print "One FRM cell was sent out by ST2"; \
break \
        else if ($nbuse(4->3)=#epdst4) then \
        print "One FRM cell was sent out by ST4"; \
        break \
                else if ($nbuse(4->4) = #epdst5) then \
                print "One FRM cell was sent out by ST5"; \
                break fi \
        fi \
fi
```

The command line "if ($nbuse(4->1) = #epdst2) then" states "if the total times of execution of transition 4->1 (which is ST2 in the Scheduler) equals the expected times of execution of ST2, then". The edb numbering system starts from #0, and 4->1 stands for module instance 4 transition #1 that is ST2 in our ABR model. "epdst2" is a user defined variable that I use to indicate the "expected times of execution of ST2".

To invoke TC3 and observe the situation in that one FRM cell is sent out when X1+X2>Mrm and t2>Trm, we type the following commands.

> so { case3()}

> #epdst2:=1

> #epdst4:=1

> #epdst5:=1

> $fs:=10

> continue

In order to be able observe the situation in that an FRM cell is sent out when X1+X2>Nrm-1, we must enlarge the value of Trm. We can do this by modifying Trm be 2.0, and then do the same as above.

4.4 TC4: Reduce ACR to ICR

Test case 4 is defined to make sure that ACR is reduced to ICR "before sending a forward in-rate RM cell. if ACR>ICR and the time T that has elapsed since the last in-rate forward RM-cell was sent is greater than ADTF" (defined as Source Behavior #5 in the ATM ABR specification). To do this we need to display ACR and t2 (which is the time elapsed since last FRM cell was sent) at the moment just before an FRM cell is going to be sent. This situation is implemented in transition ST3 of the Scheduler sub-machine.

We define the following edb observer "case45" for both case 4 and case 5.

```
d 4->ACR ; d 4->X1 ; d 4->X2 ; d 4->t1 ; d 4->t2 ; \
d 4->X ; d 4->Y ; \
if ($nbuse(4->2) = #epdst3) then \
print "ACR is reduced to ICR by ST3"; \
break \
        else if ($nbuse(4->4) = #epdst5) then \
        print "ACR is reduced by ACR*CDF by ST5"; \
        break fi \
fi
```

We invoke TC4 by executing the follow edb commands.

```
> so { case45()}
> #epdst3 := 1
> #epdst5 := 1
> $fs := 50
> continue
```

Note that the value of ACR will first be increased to the level that is greater than ICR, and then be reduced to ICR.

4.5 TC5: Reduce ACR by ACR*CDF

The Source Behavior #6 of ATM ABR specification defines that, "Before sending an in-rate forward RM-cell, and after following behavior #5 above, if at least Crm in-rate forward RM-cells have been sent since the last backward RM-cell with BN=0 was received, then ACR shall be reduced by at least ACR*CDF, unless that reduction would result in a rate below MCR, in which case ACR shall be set to MCR." This is what the test case 5 is going to verify. TC5 must be invoked right after TC4, as the specification has declared.

The rationale of Source Behavior #6 is that, if a certain number of FRM cells are lost in the network, then it is an indication that there must be a transmission problem occurring in the network. Thus, we must have some FRM cells lost in the network before the above situation can be met and TC5 can be triggered. There is a simple way to change the environment and invoke this error condition. Since we have a variable "Y" in the Scheduler sub-machine, which is the number of FRM cells sent out since last BRM cell with BN=0 was received, we can manually modify the value of this variable so that Y is greater than Crm, indicating network error has occurred. This modification can be done by the command " 4->Y := 5", if 5 is the desired value of Y.

Right after the message "ACR is reduced to ICR by ST3" has been displayed in TC4, we can invoke TC5 by the following edb command lines.

> #epdst3 := 2

> 4->Y := 5

> continue

When the message "ACR is reduced by ACR*CDF by ST5" is printed on the screen, observe the new value of ACR and verify that the new value is indeed reduced by the amount of ACR*CDF.

4.6 TC6: Increase ACR by PCR*RIF

The Source Behavior #8 adjusts ACR in two different ways. "If a backward RM-cell is received with CI=1, then ACR shall be reduced by at least ACR*RDF, unless that reduction would result in a rate below MCR, in which case ACR shall be set to MCR. If the backward RM-cell has both CI=0 and NI=0, then the ACR may be increased by no more than RIF*PCR, to a rate not greater than PCR." Transition T4 and T5 of the Station module in our ABR simulation model implement behavior #8.

Test case 6 is defined to make sure our ABR model correctly implements the mechanism to increase the ACR by PCR*RIF, and test case 7 is for decreasing the ACR by ACR*RDF.

Since both TC6 and TC7 involve transitions of the Destination sub-machine, we put them together in the edb observer "case67" as following.

```
d 4->ACR ; d 2->stci ; d 2->stni ; \
if ($nbuse(2->3) = #epdt4) then \
print "ACR is reduced to ACR*RDF by T4"; \
break \
        else if ($nbuse(2->4) = #epdt5) then \
        print "ACR is increased by PCR*RIF by T5"; \
        break fi \
fi
```

To invoke TC6, we execute the following edb command lines.

```
> so { case67()}
> #epdt4 := 1
> #epdt5 := 1
> $fs := 50
> continue
```

When the message "ACR is increased by PCR*RIF by T5" is printed on screen, observe the value of ACR, ci, and ni.

4.7 TC7: Reduce ACR by ACR*RDF

To invoke TC7, we have to let internal congestion occur in the receiving network station so that a BRM cell with CI=1 can be produced. This can be done by modifying the "ici" value of the receiving station, using the edb command line "6->ici := true". Here "6" is the index number of the Destination sub-machine of the receiving Station.

Right after the message "ACR is increased by PCR*RIF by T5" is printed on screen in TC6, we continue to execute the following.

> #epdt5 :=2

> 6->ici := true

> continue

Repeating the execution of "continue" command, we shall get the message "ACR is reduced by ACR*RDF by T4". Observe the value of ACR, ci, and ni.

4.8 Test Results

The test cases are executed under the following constants and variable environment.

ADTF =2;

ICR =200;

MCR =100;

Mrm =2;

Nrm =16;

PCR = 1000;

RDF =0.1;

RIF =0.1;

Trm =0.2;

Crm = 3;

CDF =0.1;

ervalue =1000;

elapse =0.0;

delapse =0.02;

* Before testing for (X1+X2)>=Nrm-1 in TC3 second part, set Trm=2.0. Set back Trm=0.2 after this test is finished.

** Before executing test cases 4 and 5, set ADTF=0.18, and set it back to ADTF=2.0 after these two cases are finished.

Test Case	Expected Result	Test Result	Status
TC1	ACR increases from 200 to 1000	As expected	Passed
TC2	ACR decreases from 1000 to 100	As expected	Passed
TC3*	A FRM cell is sent out when ((X1+X2)>=2 & t2>=0.2) or (X1+X2)>=15	A FRM cell is sent out when ((X1+X2)=2 & t2=0.24) or (X1+X2)=15	Passed
TC4**	ACR is reduced to ICR when (t2>0.18 & ACR>200)	ACR is reduced to 200 when (t2=0.24 & ACR=300)	Passed
TC5**	ACR is reduced by ACR*0.1 when Y>3	ACR is reduced by ACR*0.1 when Y=5	Passed
TC6	ACR is increased by PCR*RIF when both stci=0 and stni=0	ACR is increased by 1000*0.1 when both stci=0 and stni=0	Passed
TC7	ACR is decreased by ACR*RDF when stci=1	ACR is decreased by ACR*0.1 when stci=1	Passed

Chapter 5

Conclusion

The ATM ABR protocol is designed to make unused bandwidth available to cooperating end users in a fair, timely manner. This goal is achieved by adjusting value of Allowed Cell Rate (ACR) which controls the transmission speed of a network station for a specific connection.

The ABR protocol is first specified using an English description in the main body of the draft Traffic Management specification [ATM]. Some researchers have specified the protocol in Parameterized Extended Finite State Machines (PEFSM) in more detail.

Based on the available PEFSM model, we implemented the ABR protocol using a standard network specification language Estelle, and verified the protocol as both a black-box and a white-box. Through the black-box testing, we observed that the value of ACR has been correctly adjusted according to the occurrence of network congestion. Using white-box test cases, we observed that the ABR model correctly adjusted ACR in the exact way the protocol has been specified.

During the process of specification and verification for ABR protocol, we also found that the Estelle Development Toolset (EDT) is a great tool that has provided us with a flexible environment for verifying networking protocols. With the benefit of using "observers", we have successfully controlled, observed, and traced the execution of the ABR simulation model.

References

[1] ATM Forum Technical Committee, "Traffic Management Specification Version 4.0".

[2] David Lee, K. K. Ramakrishnan, W. Melody Moh, A. Udaya Shankar, "Protocol Specification Using Parameterized Communicating Extended Finite State Machines: A Case Study of the ATM ABR Rate Control Scheme", IEEE, 1996.

[3] David McDysan, Darren Spohn, "ATM Theory and Applications", McGraw-Hill, 1998.

[4] S. Budkowski, et al, "Estelle, Simulator/Debugger (EDB), User Reference Manual", Institute National des Telecommunications, Department systems et Reseaux, 9, rue charles Fourier, 91011 Evry, France.

[5] ESTELLE DEVELOPMENT TOOLKIT: http://www-lor.int-evry.fr/edt

Appendix

The Estelle Implementation of the Scheduler Module

```
{========== BEGIN: DEFINITION OF MODULE Scheduler ============}
module Scheduler_h activity;
      ip  sin: S_S(TBIN);
          sfdout: FD(FDOUT);
          sbout: B(BOUT);
      export X, Y: integer;
          ACR: real;
            { X: number of BRM cells queued to Scheduler by Station}
            { Y: number of FRM cells sent by Scheduler since last BRM
cell with BN=0 received by Station }
            {ACR: Allowed Cell Rate }
          d_ready: boolean;  {indicator data cells waiting to be sent }

end;

BODY Scheduler_b for Scheduler_h;
      STATE S0, S1, S2;

      VAR
            X1: integer;
                  {number of BRM cells transmitted by Scheduler since
                  last FRM cell transmitted by Scheduler }
            X2: integer;
                  {number of data cells transmitted by Scheduler since
                  last FRM cell transmitted by Scheduler }
            t1: real; {time elapsed since last cell sent by Scheduler}
            t2: real;
                  {time elapsed since last FRM cell sent by Scheduler}

            { the following are vars for FRM cell }
            sdir: integer;
            sbn: integer;
            sci: integer;
            sni: integer;
            ser: real;
            sccr: real;

            { the following are vars for BRM cell }
            bdir: integer;
            bbn: integer;
            bci: integer;
            bni: integer;
            ber: real;
            bccr: real;

            sefci: integer;   { for Data cell }
```

```
      temp: real;      { temp value for calculation }

Initialize to S0
begin
      X :=0;
      Y :=0;
      ACR:=ICR;
      X1 :=0;
      X2 :=0;
      t1 :=0;
      t2 :=0;
      d_ready := false;

      sdir :=0;
      sbn :=0;
      sci:=0;
      sni:=0;
      ser:=ervalue;
      sccr:=ICR;

      bdir:=0;
      bbn:=0;
      bci:=0;
      bni:=0;
      ber:=ervalue;
      bccr:=ICR;

      sefci:=0;
      temp :=0.0;
end;

{========== BEGIN: TRANSITIONS OF SCHEDULER ==============}
TRANS

from S0 to S1  {initial set up, send the first FRM cell}
name ST1: begin
      t1 :=t1+elapse;
      t2 :=t2+elapse;
      X1 :=0;
      X2 :=0;
      output sfdout.FRM(sdir, sbn, sci, sni, ser, sccr, MCR);
            {send first FRM cell}
      Y:= 1;
      t1 :=0;
      t2 :=0;
      d_ready :=true; { data allways ready to send}
end;

from S1 to S1      { Source rule 3a and 5}
provided (( (X1+X2>=Mrm)and(t2>=Trm) ) or (X1+X2= Nrm-1) )
            and ( (ACR<=ICR)or (t2<=ADTF) )
            and (t1>=1/ACR)
name ST2: begin
      t1:=t1+elapse;
      t2:=t2+elapse;
      sccr := ACR;
      output sfdout.FRM(sdir, sbn, sci, sni, ser, sccr, MCR);
```

```
            Y := Y+1;
            X1 :=0;
            X2 :=0;
            t1 :=0;
            t2 :=0;
      end;

      from S1 to S2      { Source rule 3a and 5}
      provided ( ( (X1+X2>=Mrm)and(t2>=Trm) ) or (X1+X2=Nrm-1) )
                  and (ACR>ICR) and (t2>ADTF)
      name ST3: begin
            t1:=t1+elapse;
            t2:=t2+elapse;
            ACR := ICR;
      end;

      from S2 to S1      {Source rule 3a, 5, and 6}
      provided (Y<Crm)
      name ST4: begin
            t1:=t1+elapse;
            t2:=t2+elapse;
            sccr := ACR;
            output sfdout.FRM(sdir, sbn, sci, sni, ser, sccr, MCR);
            Y := Y+1;
            X1 := 0;
            X2 := 0;
            t1 := 0;
            t2 := 0;
      end;

      from S2 to S1    {Source rule 6 in the context of Source rule 5
and 7}
      provided (Y>=Crm)
      name ST5: begin
            t1:=t1+elapse;
            t2:=t2+elapse;
            temp := ACR*(1-CDF);
            if (MCR>temp) then
                  ACR:= MCR
            else
                  ACR:= temp;


            sccr := ACR;
            output sfdout.FRM(sdir, sbn, sci, sni, ser, sccr, MCR);
            Y := Y+1;
            X1 := 0;
            X2 := 0;
            t1 := 0;
            t2 := 0;
      end;


      from S1      { Source rule 3b send a BRM cell to network}
      provided (X>0) and ((X1-X2<Mrm) or (t2<Trm))
            and (X1+X2 < Nrm-1) and
            ((X1=0) or not d_ready) and (t1>= 1/ACR)
```

```
name ST6: begin
      t1:=t1+elapse;
      t2:=t2+elapse;
      X := X-1;
      X1 := X1 +1;
      output sbout.BRM(bdir, bbn, bci, bni, ber, bccr, MCR);
      t1 :=0;
end;

from S1            { Source rule 3c send a data cell to network}
provided d_ready and ((X=0) or (X1>0)) and
      ((X1+X2<Mrm) or (t2<Trm)) and
      (X1+X2<Nrm-1) and (t1>=1/ACR)
name ST7: begin
      t1:=t1+delapse;
      t2:=t2+delapse;
      X2 := X2+1;
      output sfdout.Data(sefci);
      t1 := 0;
end;

from S1     {increase timer, and check if data cell is waiting }
name ST9: begin
      t1 := t1+delapse;
      t2 := t2+delapse;
            d_ready :=true;
end;

{the following record the value for TBRM cell when getting it}
from S0
when sin.TBRM(DIR, BN, CI, NI, ER, CCR, MCR)
name ST9: begin
      t1:=t1+elapse;
      t2:=t2+elapse;
      bdir := DIR;
      bbn := BN;
      bci := CI;
      bni := ni;
      ber := ER;
      bccr :=CCR;
end;

from S1
when sin.TBRM(DIR, BN, CI, NI, ER, CCR, MCR)
name ST10: begin
      t1:=t1+elapse;
      t2:=t2+elapse;
      bdir := DIR;
      bbn := BN;
      bci := CI;
      bni := ni;
      ber := ER;
      bccr :=CCR;
end;

from S2
when sin.TBRM(DIR, BN, CI, NI, ER, CCR, MCR)
```

```
      name ST11: begin
            t1:=t1+elapse;
            t2:=t2+elapse;
            bdir := DIR;
            bbn := BN;
            bci := CI;
            bni := ni;
            ber := ER;
            bccr :=CCR;
      end;

      {============END: TRANSITIONS OF SCHEDULER============};
end;

{=============== END: BODY OF SCHEDULER ====================};
```