Effective Control of Traffic Flow in ATM Networks Using Fuzzy Explicit Rate Marking (FERM)

Andreas Pitsillides, Y. Ahmet Şekercioğlu, and Gopalakrishnan Ramamurthy

Abstract— **In this paper, we describe the fuzzy explicit rate marking (FERM) traffic flow control algorithm for a class of best effort service, known as available bit rate (ABR), proposed by the ATM Forum. FERM is an explicit rate marking scheme in which an explicit rate is calculated at the asynchronous transfer mode (ATM) switch and sent back to the ABR traffic sources encapsulated within resource management (RM) cells. The flow rate is calculated by the fuzzy congestion control (FCC) module by monitoring the average ABR queue length and its rate of change, then by using a set of linguistic rules. We use simulation to compare the steady-state and transient performance of FERM with EPRCA (a current favorite by the ATM Forum) in the presence of high priority variable bit rate (VBR) video and constant bit rate (CBR) in both a local-area network (LAN) and a wide-area network (WAN) environment. Our experiments show that FERM exhibits a robust behavior, even under extreme network loading conditions, and ensures fair share of the bandwidth for all virtual channels (VC's) regardless of the number of hops they traverse. Additionally, FERM controls congestion substantially better than EPRCA, offers faster transient response, leads to lower end-to-end delay and better network utilization.**

I. Introduction

A MAJOR DEVELOPMENT in high-speed networking is the emergence of broadband integrated service digital networks (B-ISDN's) and asynchronous transfer mode (ATM). ATM has been designed to support various classes of multimedia traffic with different bit rates and quality-of-service (QoS) requirements. Due to the unpredictable fluctuations and burstiness of traffic flow within multimedia networks, congestion can occur frequently. Therefore, it is necessary to design appropriate congestion control mechanisms to ensure the promised QoS is met. Meanwhile, due to the high speed transfer rate (in the range of hundreds to thousands of Mb/s) and rather short cell length (53 bytes), the ratio of propagation delay to cell transmission time and the ratio of processing time to cell transmission time of ATM networks are significantly higher than that of existing networks. This leads to a shift in the network's performance bottleneck from channel transmission speed (in most existing networks) to propagation delay of the channel and the processing speed at the network switching nodes [2]. Therefore, an important issue in the flow and congestion control of ATM networks is how to handle the conditions of a large number of cells being in transit between two ATM switching nodes. As a result, many congestion control algorithms of existing packet switching networks do not operate efficiently in ATM networks. Designing effective congestion control techniques and developing traffic models for ATM networks have been difficult because of the nature of the multimedia traffic and future networking scenarios. In the networking literature several congestion control schemes have been proposed and their performance reported. The control policy is often heuristically motivated, based on simple on–off models, for example, as in control methods based on forward or backward explicit congestion notification [3]–[6].

The ATM Forum has defined a new service class for data applications called available bit rate (ABR) [7]. While this service does not provide any strict guarantees, it attempts to minimize the cell loss at the expense of delay. Using appropriate distributed flow controls, competing users of this service dynamically share the available bandwidth (left over by the other services). The ATM Forum has accepted a rate-based flow control proposal whose main goal is to define a flexible framework, or family of rate-based closed-loop schemes for congestion control. Within this framework, we have designed an effective explicit rate (ER) based control strategy—fuzzy explicit rate marking (FERM)—for ABR traffic. ER based schemes have been advocated by a number of researchers due to their reported advantages over single bit feedback schemes (e.g., see Charny *et al*. [8], Roberts [9], Jain *et al*. [10], Pitsillides *et al*. [1], Bonomi and Fendick [11]). The reported advantages of ER based schemes include: the use of more information from the switches, such as the current queue length and its growth rate; the reaction times are substantially better; faster startup in most cases; policing can be straightforward since the entry switches can monitor returning resource management (RM) cells and use the rate directly in their policing algorithms; and robustness against loss of RM cells, as the next correct RM cell will bring the rate to its correct value. Our scheme is substantially different from the ones proposed in the literature, in the sense that we use formal computational intelligence techniques (fuzzy control) in the design of the control algorithm. We therefore draw upon the vast experience, in both theoretical as well as practical terms, of computational intelligence control. Additionally, in the calculation of the ER, we monitor both the current queue length and its growth rate. The queue length captures the current state of the queue, and the rate of change of the

Manuscript received February 1996; revised July 1, 1996. This paper was presented at the GLOBECOM'95 Conference.

A. Pitsillides is with the Department of Computer Science, University of Cyprus, Nicosia, Cyprus.

Y. A. Şekercioğlu is with the School of Computer Science and Software Engineering, Swinburne University of Technology, Hawthorn, Victoria, 3122, Australia.

G. Ramamurthy is with C & C Research Laboratories, NEC USA Inc., Princeton, NJ 08540 USA.

Publisher Item Identifier S 0733-8716(97)00475-7.

queue length provides some form of prediction for the future queue behavior. Thus, our scheme is expected to be more effective than schemes using feedback based on the queue length threshold, queue length, or the rate of change of the queue length alone. Additionally, our model is more powerful, better able to reflect the system behavior (captured by simple linguistic information—see later discussion), as compared with some of the simple models used, as for example in forward or backward congestion notification schemes (on–off model). Unlike some other ER schemes [9] where the sources are given an ER only after congestion occurs, our scheme provides the ER to all the active virtual channels (VC's) at all the time so that congestion and undesired resulting behavior can be avoided. Note that, not many researchers have adopted the reported strength of fuzzy logic in solving telecommunication problems; some notable examples include telecommunications network control and management [12]–[14] and traffic routing [15].

Nowadays, we are faced with increasingly complex control problems, for which different modeling representations (e.g., piecewise linear models, radial-basis function models) may be difficult to obtain. This difficulty has stimulated the development of alternative modeling and control techniques which include fuzzy logic based ones. Fuzzy modeling methods may lead to models that describe the behavior of systems sufficiently well for their application in fuzzy control (a complement to the linguistic fuzzy modeling method is the fuzzy model based control). Thus due to the demand for inexpensive but reliable models, the fuzzy modeling approach may turn out to be a useful complement to traditional modeling and control approaches when both the complexity and uncertainty about the system increases. This is of great practical significance, since modeling is usually the bottleneck for the application of effective (model-based) control.

When designing the new scheme, our major motivation was to exploit the advantages of fuzzy logic control: ability to quickly express the control structure of a system using *a priori* knowledge; less dependence on the availability of a precise model of the controlled process; and easy handling of the inherent nonlinearities. We expect that this strategy will be robust with respect to traffic modeling uncertainties and system nonlinearities, yet provide tight control (and as a result, maintain QoS to the user). Also that, it will have applicability in the presence of high priority variable bit rate (VBR) video and constant bit rate (CBR) traffic in both local and wide area environments.

The rest of the paper is organized in five sections: Section II provides a very brief introduction to fuzzy logic based control; Section III describes the proposed FERM flow control scheme and provides guidelines for its design; Section IV discusses performance issues and evaluates performance using simulation of a three node local-area network/wide-area network LAN/WAN (Fig. 1) and compares the results of experiments; Sections V and VI present properties of the scheme and our conclusions; finally, Appendix A summarizes the EPRCA algorithm we have used in our comparison study, and Appendix B summarizes the operational principles of fuzzy logic controllers.

II. Fuzzy Logic and Fuzzy Control Systems

Fuzzy logic is a method for representing information in a way that resembles natural human communication, and for handling this information in a way that is similar to human reasoning. A major factor behind using fuzzy logic in control engineering is to provide us a method of blending qualitative linguistic expressions favored by human experts in the structure of control systems or, as stated by Zadeh, "computing with words."

Concepts of fuzzy sets, fuzzy logic, and fuzzy logic control have been introduced and developed by Zadeh in a series of articles spanning a few years—for an introduction, the reader is referred to [16]–[21].

A fuzzy logic controller (FLC) can be conceived as a nonlinear controller of which the input–output relationship can be expressed by using a small number of linguistic rules or relational expressions. As an example, one of the rules in our fuzzy congestion controller (FCC) is: "If the buffer is *full* and the rate of change is *increasing_fast*, then flow rate should be *very_little*." In this relational expression "buffer," "rate of change" and "flow rate" are called *linguistic variables* which accept values among the words of a natural or synthetic language such as "full," "increasing_fast" or "very_little." Possible values of the linguistic variables are called *linguistic values* and they are modeled by fuzzy sets [16]. In most cases, this representation leads to compact descriptions of systems and natural handling of any inherent nonlinearities in the control loops. Otherwise implementation of a controller could be difficult, if not impossible.

There are a few possible ways for obtaining the linguistic rules of a FLC. These rules can be expressed by a human expert or can be determined by using system identification techniques. In the first commercial application of FLC's (cement kiln controller), design engineers used the directives tabulated in the training manual for cement kiln operators.

Today, FLC's are embedded in a wide variety of applications as diverse as sake brewing [22] to attitude control of satellites [23]. Operational principles of FLC's are summarized in Appendix B, and a comprehensive discussion of FLC's can be found in [24]–[26].

III. Fuzzy Explicit Rate Marking (FERM) Congestion Control Algorithm

In this section, we describe the operation of our FERM scheme. FERM uses five parameters (Table I) and a set of linguistic rules (Fig. 19) which can be implemented as a lookup table for real-time operation. Source end system (SES) and destination end system (DES) behavior is similar to EPRCA (see Appendix A) and is compliant with ATM Forum Traffic Management Specification, Version 4. Cell rates of data sources are adjusted by ER information carried by resource management cells. Calculation of ER is performed by the FCC of each ATM switch.

A. Fuzzy Congestion Controller

Designing a FLC involves selection of suitable mathematical representations for t-norm, s-norm, defuzzification

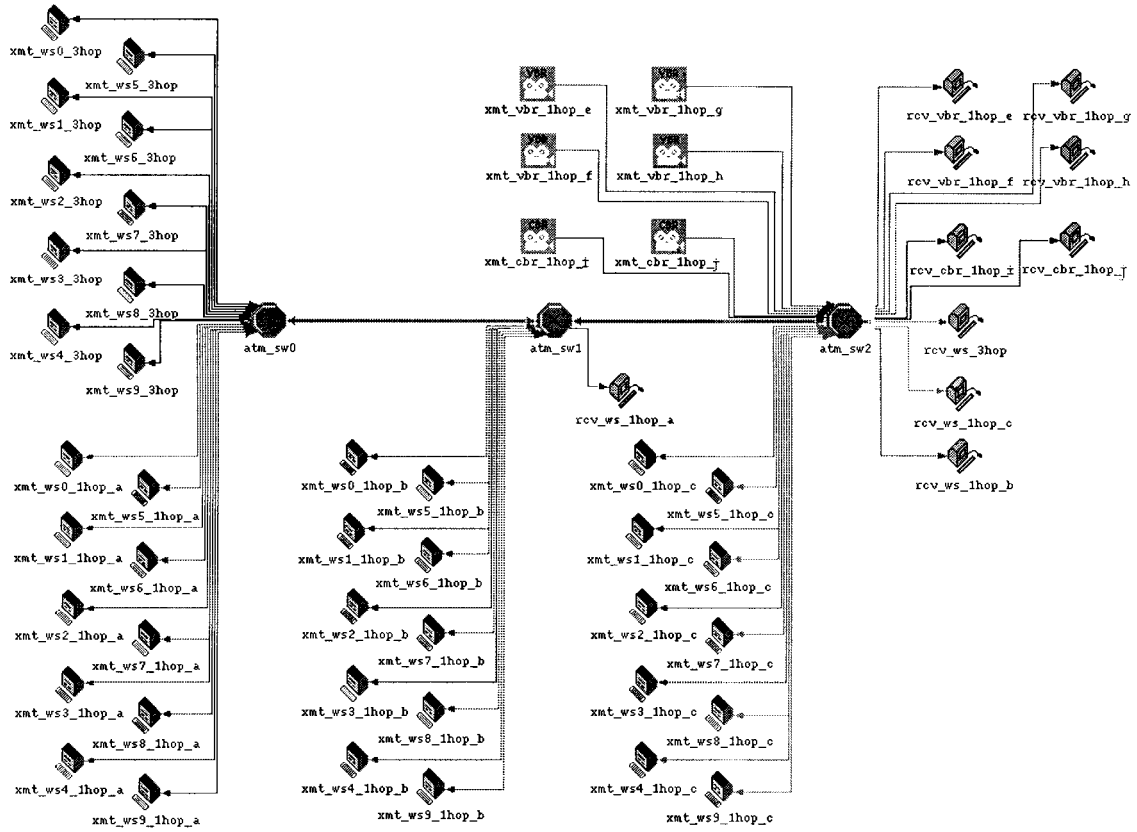

Fig. 1. ATM network model used during the simulations. Same network configuration is used for the simulation of ATM WAN and ATM LAN except that interswitch distances for the ATM WAN case are assumed to be 1500 km, and 10 km for the ATM LAN simulations. All traffic (except 1-hop b traffic) leaving ATM switch 2 travels to a fourth ATM switch via a 155 Mb/s link and distributed. Since no cell buffering occurs at this ATM switch, it is not included in the simulation model.

TABLE I
FERM PARAMETERS

PARAMETER	DEFINITION	VALUE
PCR	Peak cell rate	149.76 Mbits/sec
ICR	Initial cell rate	= PCR
AIR	Additive increase rate	= PCR
MCR	Minimum cell rate	2 Mbits/sec
N_{fp}	Control interval	50 cell service periods

operators, fuzzy implication functions, and shapes of membership functions among a rich set of candidates. Particular selection of these operators and functions alter the nonlinear input–output relationship, or in other words, the behavior of a FLC. But, research has shown that same effects can be achieved by proper modification of the rule base [26]. Therefore, in practical applications, usually computationally lighter and well studied operators and functions are selected, and desired behavior of a FLC is obtained by altering the rules.

The fuzzy congestion controller (FCC) uses two input variables to calculate an explicit flow rate: ABR queue length and its growth rate. By monitoring the rate of change in queue length in addition to the length of the ABR queue, we are able to provide a measure of future queue state, and by using explicit rate signals we can make sources more responsive to sudden changes in the network traffic volume. As can be observed from the control surface of FCC (Fig. 18), it is a nonlinear controller. For a certain queue length, it calculates different flow rate limits depending on the rate at which queue length varies.

At the end of the each filter period of N_{fp} cell times (control interval), two numerical values showing the average length of the ABR queue and the difference of the ABR queue length from the previous control interval (i.e., queue growth rate) are calculated and fed to FCC. Based on this data and the linguistic information stored in the rule base, FCC computes the fractional flow rate (FFR $\in [0, 1]$) and an explicit rate (ER = FFR $\times$ link cell rate) for the sources feeding the ATM switch. If, within the current control interval, the ATM switch receives an RM cell traveling to the upstream nodes along the relevant VC, it examines the ER field of the cell and if this rate is greater than the calculated flow rate, it modifies the ER field with the calculated value and retransmits the RM cell. Otherwise, it generates an RM cell and places the calculated flow rate in the ER field. This arrangement is intended to limit volume of control traffic and to enhance the scheme's robustness against lost or discarded RM cells.

B. Rule Base Design

The selection of rule base is based on our experience and beliefs on how the system should behave. Design of a rule base is two-fold: first, the linguistic rules ("surface structure") are set; afterwards, membership functions of the linguistic values ("deep structure") are determined.

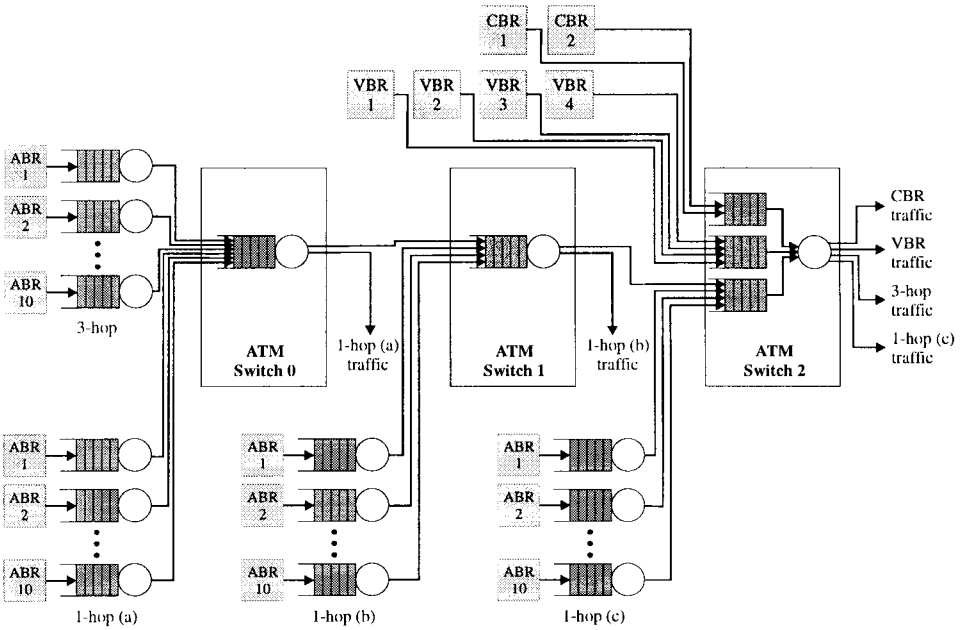

Fig. 2. Queuing model used during the simulations. ABR, VBR, and CBR queues in the ATM switches are 1024, 128, and 128 cells long, respectively. A simple priority mechanism is adopted for servicing CBR, VBR, and ABR queues; whenever the server is ready for transmitting a cell checks the CBR queue first, if there are any cells waiting, CBR queue is serviced, otherwise VBR queue is serviced. If there are no cells present in CBR and VBR queues, ABR queue is serviced.

The tradeoff involving the design of the rule base is to have a set of minimum number of linguistic rules representing the control surface with sufficient accuracy to achieve an acceptable performance. Recently, in the fuzzy control literature, some formal techniques for obtaining a rule base by using artificial neural networks or genetic algorithms have appeared. Nevertheless, we have used the conventional trial and error approach under the guidance of some design rules of thumb (see [24] for a discussion of these).

Usually, to define the linguistic rules of a fuzzy variable, Gaussian, triangular or trapezoidal shaped membership functions are used. Since triangular and trapezoidal shaped functions offer more computational simplicity, we have selected them for our rule base.

Then, the rule base is fine tuned by observing the progress of simulation, such as cell loss occurrences and demand versus throughput curves. The tuning can be done with different objectives in mind. For example, any gain in throughput must be traded off by a possible increase in the delay experienced at the terminal queues. However, since the tuning of the fuzzy rules is intuitive and can be related in simple linguistic terms with user's experience, it should be a straightforward matter to achieve an appropriate balance between a tolerable end-to-end delay, and the increase in throughput. Alternatively, an adaptive fuzzy logic control method [27] can be used which can tune the parameters of the fuzzy logic controller on line, using measurements from the system. The tuning objective can be based on a desired optimization criterion, for example, a tradeoff between maximization of throughput with minimization of end-to-end delay experienced by the users.

We have developed a compiler and an embeddeble fuzzy logic inference engine (C-FLIE) for flexible definition of the rule base of FCC and for conveniently interfacing to OPNET simulation tool [28]. In Fig. 19, source code which is parsed to build the definitions of linguistic variables (Fig. 16) and "knowledge" of FCC is shown. When FCC first starts, it reads and parses this source file to initialize the inference engine. The development environment provides the FCC designer with a simple interface which allows the rule base to be tuned using observation of behavior of the simulated network, such as cell loss occurrences.

IV. Performance Evaluation

In this section, we concentrate on simulation to evaluate the performance of FERM and also to compare with EPRCA. Note that the inherent robustness of fuzzy control has been extensively discussed in the literature [29].

A. Steady-State Response

1) Simulation Model: Our ATM network model is shown in Fig. 1 and corresponding simulation model in Fig. 2. It consists of three ATM switches. This reference model has been designed to capture: the interference between traffic traveling a different number of hops; the interference from real-time (guaranteed) traffic competing for the server resources; the effect of propagation delay on the effectiveness of the control scheme; and the fairness (or lack of it) among traffic traveling a different number of hops. For the sake of computational simplicity, we assume all of the queuing occurs at the output buffers of the switches and there is no internal blocking. In each ATM switch, there are three separate logical buffers (per output port) collecting CBR, VBR, and ABR cells. We assume that CBR and VBR buffers can accommodate 128 cells and the ABR buffer can accommodate 1024 cells. A simple priority mechanism is adopted for servicing CBR, VBR and ABR queues: whenever the server is ready for transmitting a cell, it checks the CBR queue first, if there are any cells waiting

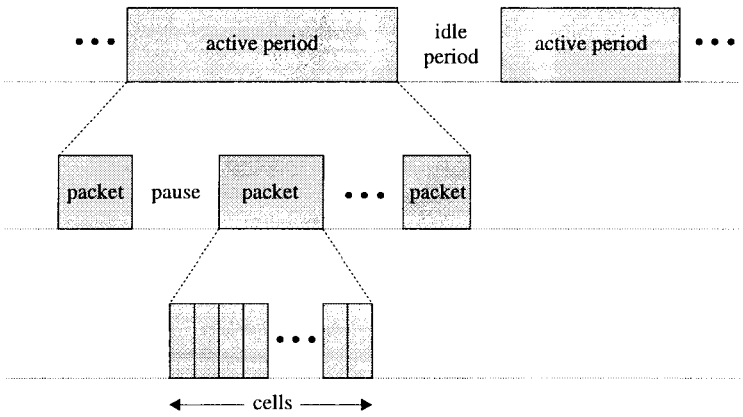

Fig. 3. ABR traffic source model. The model parameters selected for simulations can be seen in Table II.

CBR queue is serviced, otherwise VBR queue is serviced if it contains any cells. If there are no cells present in the CBR and VBR queues, ABR queue is serviced.

We use the same network model for the simulation of ATM LAN and ATM WAN, but the distances between the switches are changed to reflect the different geographic spans of the two network types. In the ATM WAN case, inter-switch distances are assumed to be 1500-km long and for the ATM LAN case 10-km long. Lengths of the access links connecting the terminals to the ATM switches are taken as 2-km long in both models. All of the links are assumed to have 155 Mb/s transmission speed. For the ABR traffic, we consider 40 connections at the edge of the network (20 are connected directly to ATM switch 0, and ten in each of ATM switches 1 and 2), which can have their transmission rate controlled by the network. Three of the ABR flow paths are 1-hop paths, and one is a 3-hop path. Also four VBR and two CBR sources are directly connected to ATM switch 2 (1 hop-path). Each ABR terminal generates traffic based on a three-state model (see Fig. 3 for model and Table II for the selected parameters). In the idle state no traffic is generated. The idle period is generated from a geometric distribution with a mean period chosen to adjust the offered load on the link. In the active state the source generates a series of packets or bursts which are interspersed by short pauses. The period of each pause is drawn from a negative exponential distribution. The packet size and the number of packets generated during an active period are geometrically distributed. Each VBR source is simulated by using the autoregressive model proposed by Maglaris *et al.* [30] (only difference is that we have assumed a video source having 480 000 pixels/frame). The CBR source generates 25 Mb/s and paces the cells into the network uniformly. In case of cell losses, which occur during the periods of congestion, we use a simple retransmission protocol. A packet is presumed to be lost even if a single cell is lost. Packets that are received by the receive terminal with missing cells are retransmitted by the source until successful delivery. The "useful network throughput" represents the actual throughput of packets (in Mb/s) that are eventually delivered to the destination without cell loss (after retransmission if necessary).

We have used Opnet simulation tool for our experiments. For the purpose of gathering the admission delay statistics, we have considered the ABR source terminal buffers as having infinite capacity.

TABLE II
ABR TRAFFIC SOURCE MODEL PARAMETERS

PARAMETER	DISTRIBUTION	MEAN VALUE
idle period	geometric	chosen to adjust network load
number of generated packets in an active period	geometric	10
packet size	geometric	8 Kbytes
pause period	exponential	0.5 msec.

2) Simulation Results: Using the simulation model we compare the performance of ATM LAN and ATM WAN for the cases of no control, control using FERM, and control using EPRCA. Figs. 4–9 show the plots of 1-hop a, 1-hop b, 1-hop c, and 3-hop links' useful throughput (at the packet level), for each of the above cases, as a function of end-to-end delay. The numbers appended next to the curves indicate the load factor on each link. For example a load factor of 1.5 means that the average offered load on a link is 1.5 times its maximum capacity (an overload in this case).

Fig. 4 considers the case of an ATM LAN without any congestion control. As the offered link load increases the packet level throughput at the congested link (1-hop c and 3-hop) decreases and the end-to-end delay increases (that is we experience a throughput collapse under overload). However, the 1-hop a and 1-hop b throughput remain high with low delay, since their corresponding switches are not as congested as the switch shared by the 1-hop c and 3-hop links. Note that the end-to-end delays do not increase substantially as the source is not controlled (cells are not queued at the terminal), That is there is no admission delay. Fig. 5 considers the case of an ATM WAN without applying any control. The results are similar to the ATM LAN without any control. Again, we experience a throughput collapse under overload, and due to the propagation delay we also observe an increase in the end-to-end delay (about 15 ms for the 3-hop traffic).

Fig. 6 shows the case of simulated ATM LAN under FERM control, which shows that FERM provides effective control.

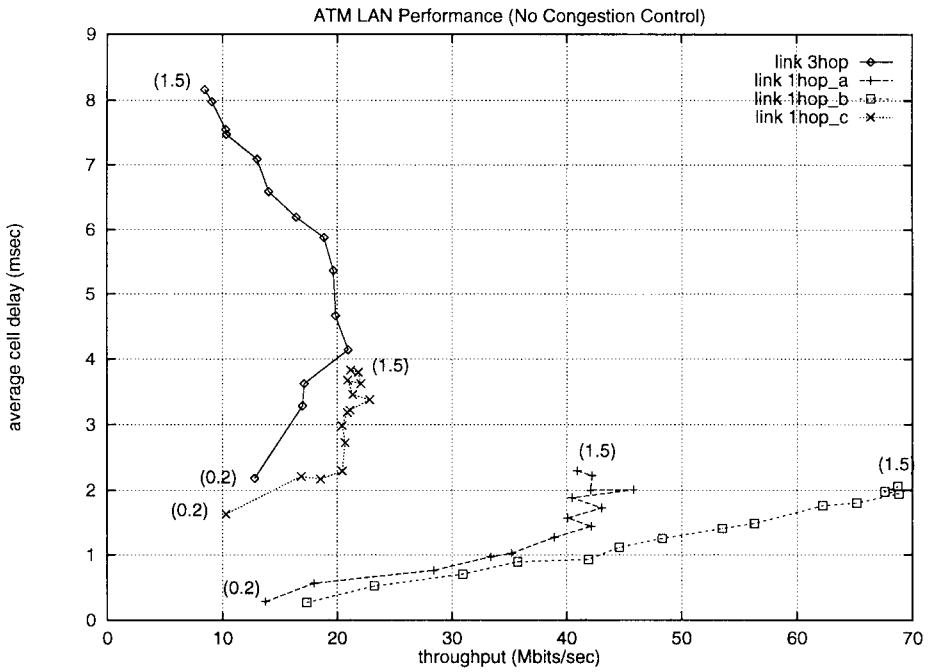

Fig. 4. Plot of average end-to-end cell delay versus useful throughput of simulated ATM LAN without applying any congestion control. The graph was produced by varying the offered link loads generated by ABR traffic sources from 20–150% of the link capacities.

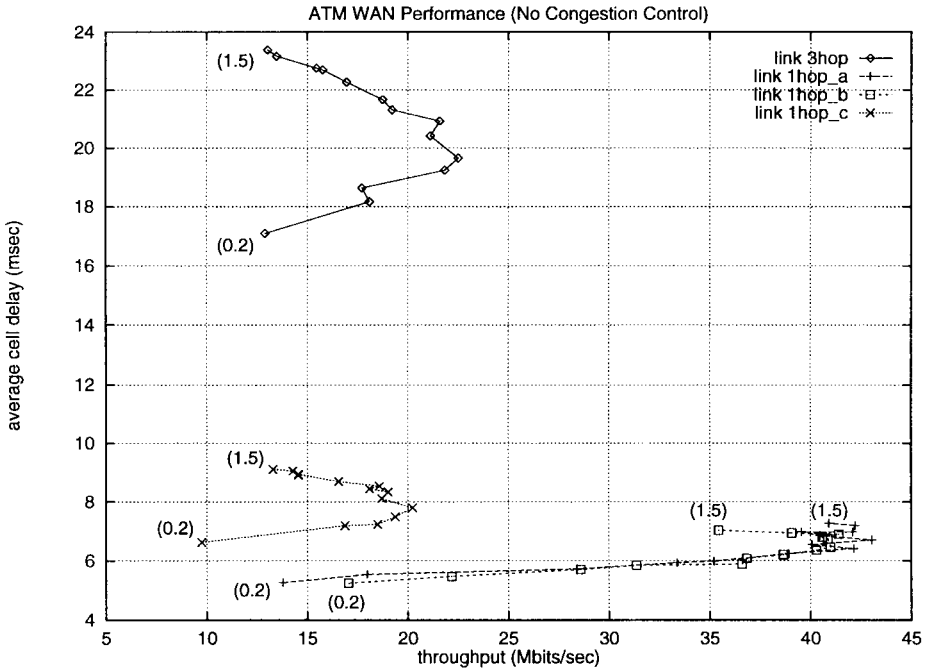

Fig. 5. Plot of average end-to-end cell delay versus useful throughput of simulated ATM WAN without applying any congestion control. The graph was produced by varying the offered link loads generated by ABR traffic sources from 20–150% of the link capacities.

That is, there is no evidence of any congestion collapse, even for offered link loads exceeding the link capacity. For the case of an uncongested link, 1-hop a and 1-hop b show only a marginal increase in the end-to-end delay as the offered link load increases, whilst the throughput increases monotonically. Even in the case of the congested link (ATM switch 2), 3-hop and 1-hop c exhibit no congestion collapse. As expected, at link overload the end-to-end delay increases considerable (for 3-hop it is about 240 ms at a link overload factor of 1.5). This is due to the increase in the admission delay because of the actions taken by the FERM control; an acceptable tradeoff for network stability (i.e., no congestion collapse), as well as an increase in the network throughput, in comparison to the case without any control. For example, at a link load factor of 1.5 the total network throughput for the case of no control is 139 Mb/s, as compared to 295 for FERM; a 112% improvement.[1] Fig. 7 shows the case of simulated ATM WAN under FERM

[1]Total network throughput does not include the VBR and CBR sources of 80 Mb/s Observe that the maximum total network throughput for ABR sources is 300 Mb/s.

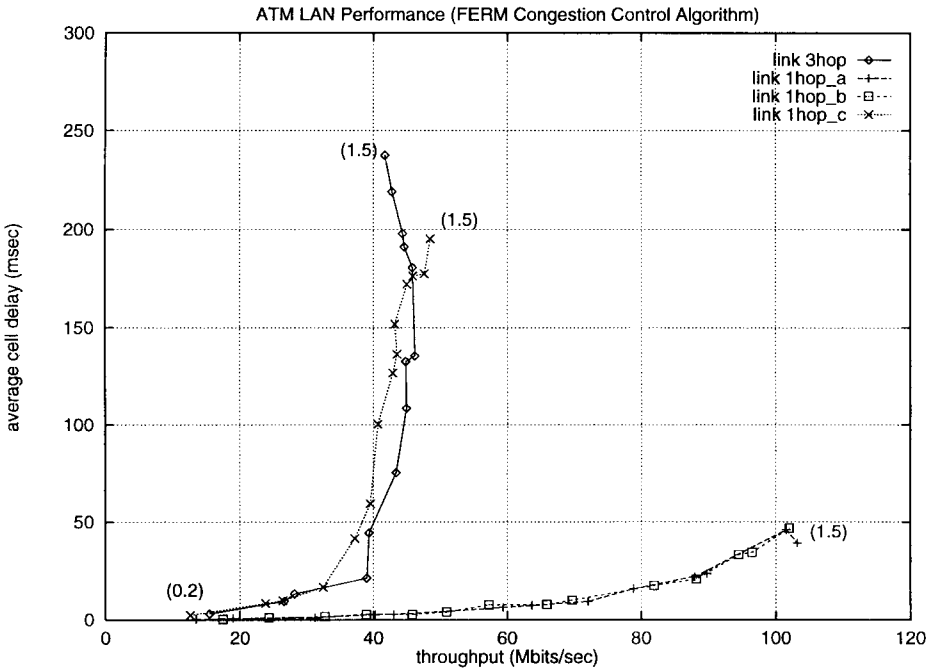

Fig. 6. Plot of average end-to-end cell delay versus useful throughput of simulated ATM LAN under FERM congestion control. The graph was produced by varying the offered link loads generated by ABR traffic sources from 20–150% of the link capacities.

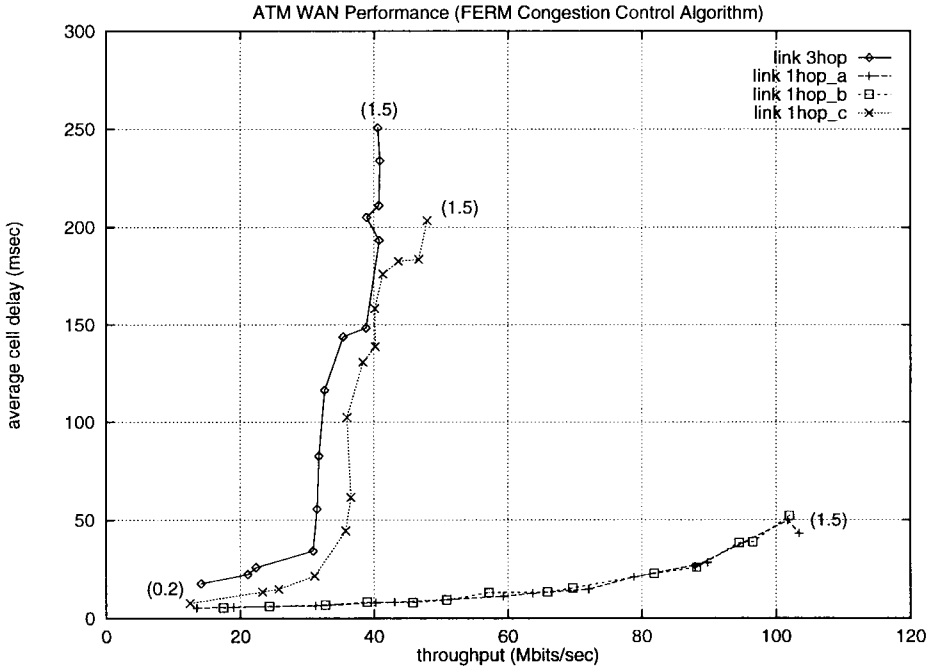

Fig. 7. Plot of average end-to-end cell delay versus useful throughput of simulated ATM WAN under FERM congestion control. The graph was produced by varying the offered link loads generated by ABR traffic sources from 20–150% of the link capacities.

control. Similar observations can be made as for the ATM LAN case (now, for a link load factor of 1.5, the total network throughput under FERM control is 294 Mb/s, in comparison to 103 Mb/s for the case of no control; a 186% improvement offered by FERM). Only difference being that that all end-to-end delays have increased marginally (by about 15 ms) due to the longer propagation delays. Note that there is no drop in total network throughput, even though there is a large delay in the feedback caused by the propagation delay. It is worth pointing out the fairness exhibited by FERM for both ATM LAN and WAN. From Figs. 6 and 7, we observe that the throughput versus delay characteristics of the 3-hop and 1-hop c traffic (which share the same congested ATM switch 2 and compete with other real time traffic for the limited bandwidth) remain virtually the same. This is despite the large propagation delay for the 3-hop traffic (it travels an extra 3000 km before it reaches ATM switch 2, as compared to 1-hop c which only travels 2 km). Fairness is further discussed in Section IV-C.

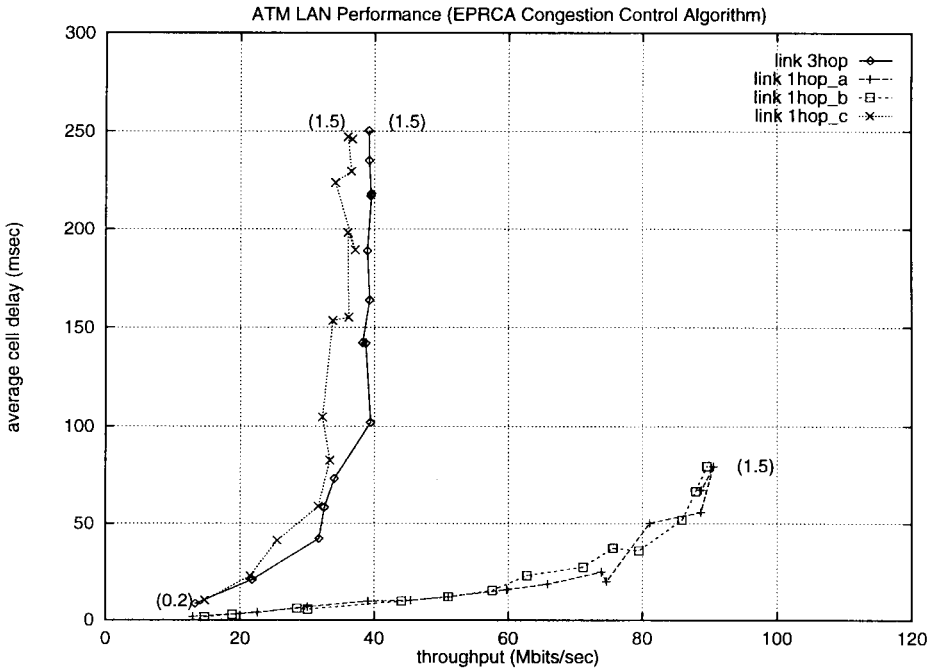

Fig. 8. Plot of average end-to-end cell delay versus useful throughput of simulated ATM LAN under EPRCA congestion control. The graph was produced by varying the offered link loads generated by ABR traffic sources from 20–150% of the link capacities.

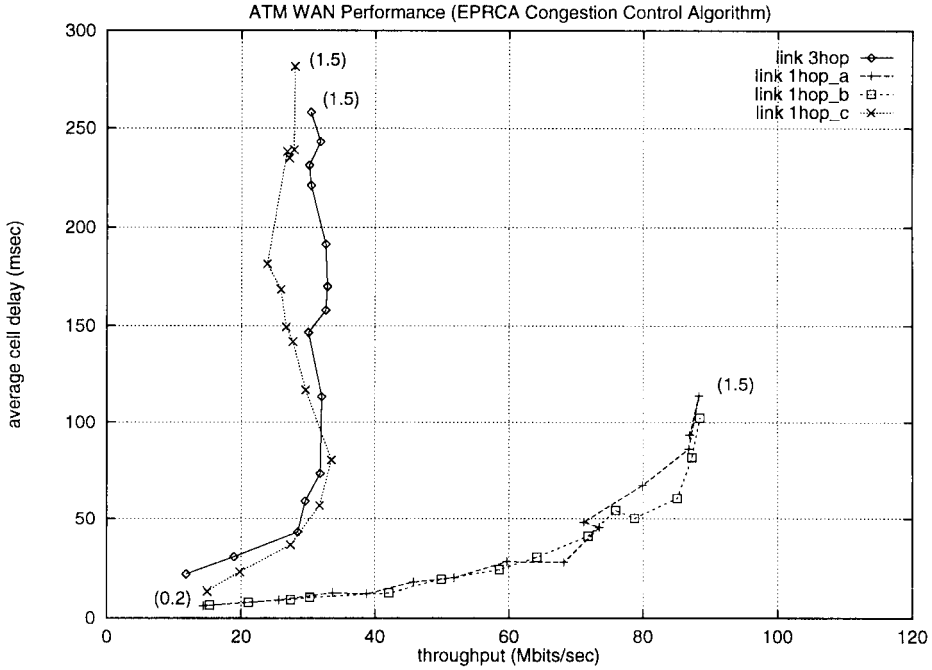

Fig. 9. Plot of average end-to-end cell delay versus useful throughput of simulated ATM WAN under EPRCA congestion control. The graph was produced by varying the offered link loads generated by ABR traffic sources from 20–150% of the link capacities.

Figs. 8 and 9 consider the case of ATM LAN and WAN under EPRCA congestion control. The behavior of EPRCA is similar to FERM, but with the following important differences: Firstly, there is a significant deterioration in the end-to-end delay performance of EPRCA. For example, FERM offers a 67% improvement in the total delay of the network at a link load factor of 0.5 (43 ms for FERM and 132.5 ms for EPRCA) in the case of an ATM LAN, and 48% (72 ms for FERM and 138 ms for EPRCA) in the case of an ATM WAN, and still a substantial 20% (LAN) and 25% (WAN) improvement for a link load factor of 1.5. Secondly, FERM is more efficient. For example, at a link load factor of 0.5, FERM offers 14% more total network throughput (151 Mb/s for FERM and 133 Mb/s for EPRCA) for the case of ATM LAN and 8% for the case of ATM WAN (141 Mb/s for FERM and 130 Mb/s for EPRCA). The superiority of FERM over EPRCA, is more pronounced at higher link loads. For example, at a link load factor of 1.5, FERM offers 16% more total

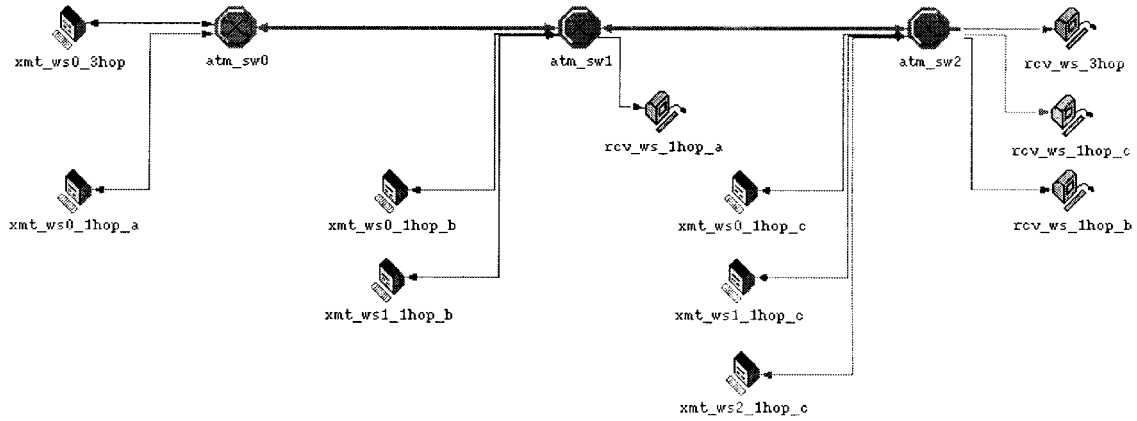

Fig. 10. ATM network model used for transient response simulations.

network throughput (294 Mb/s for FERM and 254 Mb/s for EPRCA) in the case of an ATM LAN, and 26% (294 Mb/s for FERM and 234 Mb/s for EPRCA) for the case of an ATM WAN. Note that the superiority of FERM over EPRCA will be even more pronounced in the case of random connections and disconnections of sources (due to better transient response by FERM; see next section).

B. Transient Response

In this section, we study the transient performance. We consider the response of the control, when the load on the network changes suddenly. We are interested in the rise time, the time to reach stable steady state (if at all), and the overshoots and undershoots from the steady-state values.

1) Simulation Model: For the transient response, we use the simulation model shown in Fig. 10. In order to allow for easier interpretation of the transient behavior, we consider ABR sources only (i.e., VBR and CBR sources are turned off). The sources are persistent (with infinite backlog). The 3-hop and 1-hop a links have one source, 1-hop b has two sources, and 1-hop c has three sources. The 3-hop source begins transmitting cells at time $t = 0$, 1-hop a source starts at time $t = 0.2$ s, 1-hop b and 1-hop c sources become active at time $t = 0.4$ s and $t = 0.6$ s respectively.

2) Simulation Results: A comparison of the transient responses of ATM LAN under EPRCA and FERM control is shown in Figs. 11 and 12. FERM offers good transient behavior with good rise time, good settling time (with no, or small overshoots or undershoots), and insignificant, if any, oscillations. Its transient behavior is much better than EPRCA, in the sense that FERM attains steady state much faster and that it offers "smooth" control (no or negligible oscillations present). The rise time for FERM is less than 0.005 s, compared with 0.05 s for EPRCA. The settling time is about 0.02 s for FERM and infinite for EPRCA (cycles do not appear to die out). EPRCA exhibits constant oscillations, of mostly significantly large amplitudes (e.g., 1-hop a source during the period 0.6–0.8 s exhibits peak-to-peak variations of about 20% of link rate). It is worth noting that after each new connection both schemes settle to the same steady-state value (average for EPRCA). See next section on fairness for a discussion.

The transient responses for the ATM WAN case are shown in Figs. 13 and 14. FERM, again, as in the case of ATM LAN, offers good transient behavior; only difference being that the settling time has increased somewhat (lasting about 0.08 s for the ATM WAN case in comparison to about 0.02 s in the case of an ATM LAN) and rise time to about 0.01 s (compared to 0.005 s). Considering the large propagation delay, we believe that FERM offers good transient behavior. This can be contrasted with the very poor transient behavior of EPRCA. The EPRCA scheme, as in the case of an ATM LAN, does not reach steady-state. Rather it oscillates around the steady-state value, with very large oscillation amplitudes (about 50% of link rate in some cases). Its settling time is again infinite, and its rise time difficult to establish, but in some cases, it can be over 0.1 s (compared with 0.01 for FERM).

C. Fairness

We use the same results obtained in the transient case (see Figs. 11–14) and infer fairness from them. For fairness it is important to achieve "fair" sharing of the available bandwidth, in the presence of competing users of possibly different geographic locations. Several definitions of fairness exist [31]; the max–min definition of fairness is the most popular one. FERM achieves fairness, in a max–min sense. Note that max–min fairness is achieved without keeping track of the bottleneck rates of individual sources.

This can be observed, by considering the responses for both ATM LAN and ATM WAN. For example, consider the ATM WAN case under FERM control (see Figs. 12 and 14). At time $t = 0$ source 3-hop starts. As there are no other sources competing, at any part of the network, full rate is allocated to it. At time $t = 0.2$ s source 1-hop a starts. Since there are two sources now competing for the resources at ATM Switch 0, equal allocation is made for both sources (see period 0.2 to 0.4 s). At time $t = 0.4$ s two 1-hop b sources are connected. Now at ATM switch 1, there are three sources (one 3-hop and two 1-hop b sources) competing for the resource, therefore the rate is equally divided between them. That is 3-hop now is reduced to 1/3 of the link rate. Since 3-hop is reduced to 1/3 of link rate, 1-hop a (the source at ATM switch 0) increases its cell rate to 2/3 of the link rate (i.e., it picks up the spare

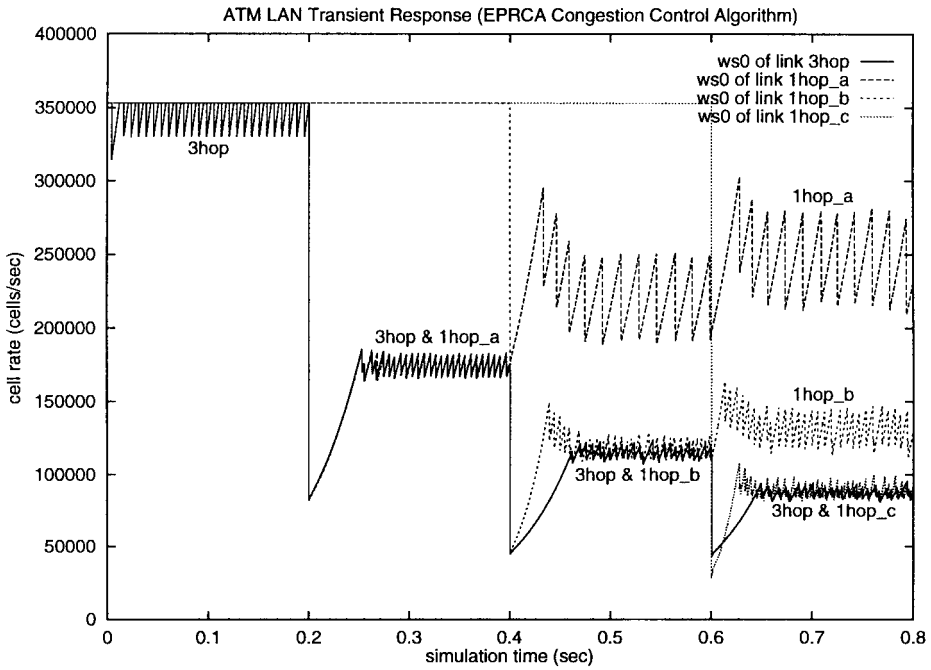

Fig. 11. Source allowed cell rate transient response of simulated ATM LAN under EPRCA congestion control. At time $t = 0$, 3-hop source, $t = 0.2$ s 1-hop a source, $t = 0.4$ s 1-hop b sources, and $t = 0.6$ s 1-hop c sources start transmitting cells.

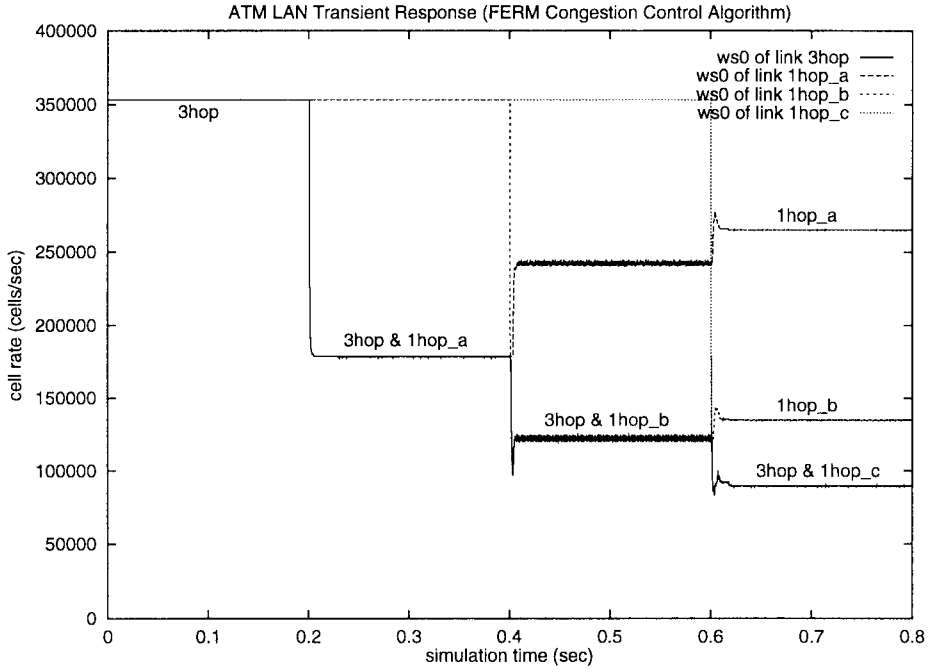

Fig. 12. Source allowed cell rate transient response of simulated ATM LAN under FERM congestion control. At time $t = 0$, 3-hop source, $t = 0.2$ s 1-hop a source, $t = 0.4$ s 1-hop b sources, and $t = 0.6$ s 1-hop c sources start transmitting cells.

slack). Similarly, when the three sources at ATM switch 2 are connected at time $t = 0.6$, we see a redistribution of the allocated rate in a max–min sense. That is the rate of 3-hop is reduced to 1/4 of the link rate, 1-hop a therefore increases its rate to 3/4 of the link rate, each 1-hop b source increases its rate to 3/8 of link rate, and each of the three 1-hop c sources is allocated its fair share of 1/4 of the link rate.

Similar results are observed for EPRCA. However, it must be pointed out that even though EPRCA exhibits max-min fairness, it does that only in an "average" sense.

V. Properties of FERM

Any suggested control scheme must be effective and offer certain control features which will make it attractive for implementation purposes. FERM is effective and compares favorably with the following desirable features (some of these have been discussed by a number of researchers, e.g., [11], [31], and [32]).

1) Robustness: Our simulative experience shows no collapse in throughput, even in the presence of severe overloads.

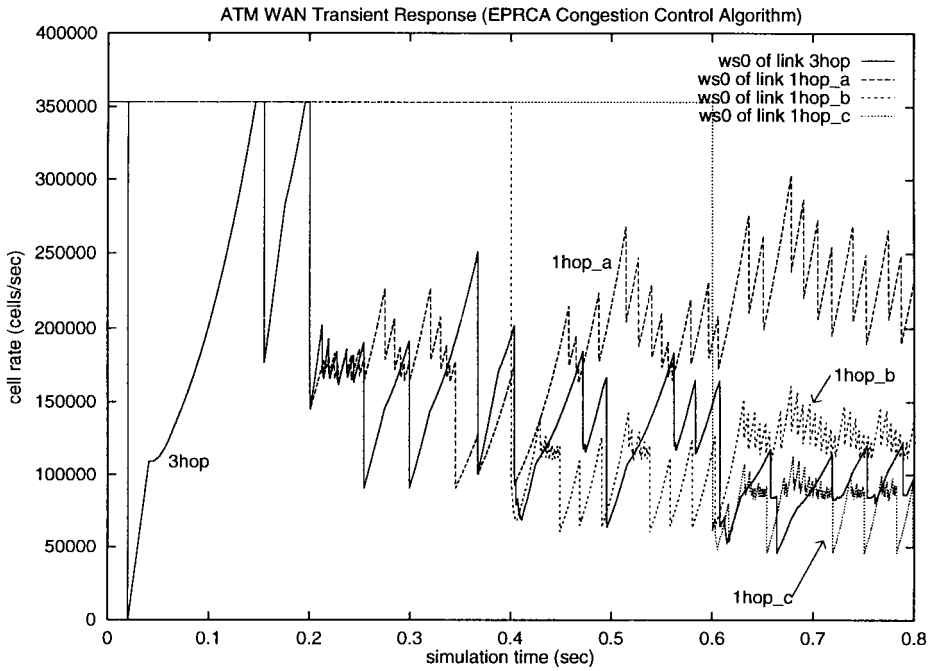

Fig. 13. Source allowed cell rate transient response of simulated ATM WAN under EPRCA congestion control. At time $t = 0$, 3-hop source, $t = 0.2$ s 1-hop a source, $t = 0.4$ s 1-hop b sources, and $t = 0.6$ s 1-hop c sources start transmitting cells.

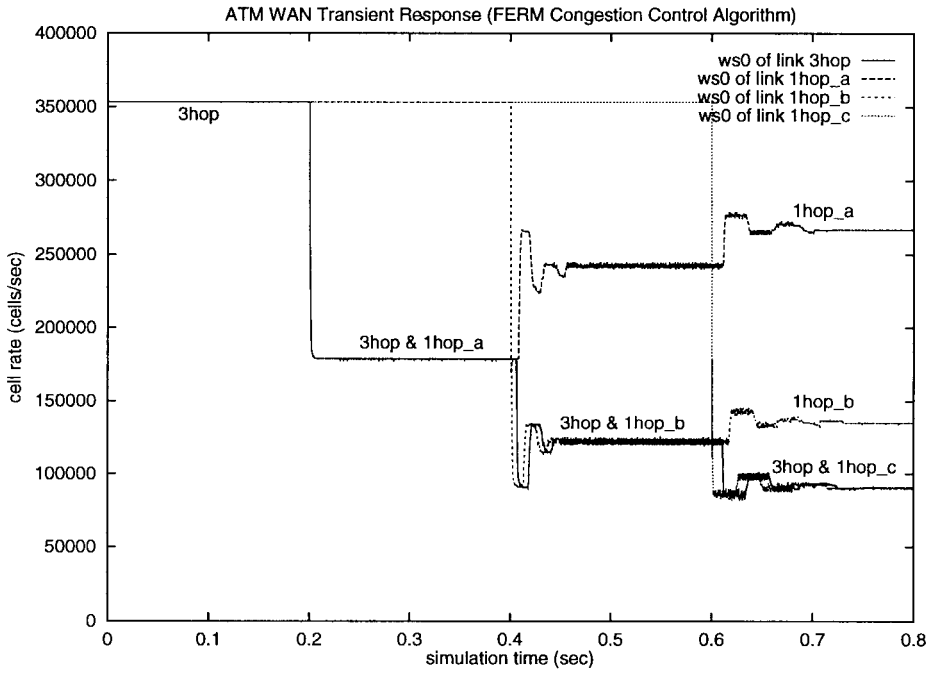

Fig. 14. Source allowed cell rate transient response of simulated ATM WAN under FERM congestion control. At time $t = 0$, 3-hop source, $t = 0.2$ s 1-hop a source, $t = 0.4$ s 1-hop b sources, and $t = 0.6$ s 1-hop c sources start transmitting cells.

To the contrary, at a link load factor of 1.5 the total network throughput is 295 Mb/s (out of a maximum possible of 300 Mb/s, i.e., almost unity utilization). Its inherent robustness is also supported by the vast literature on fuzzy control. The ultimate test for any method of control is successful applications in industry. Fuzzy control is a prime example of the success of the theory in solving difficult control problems in real practice.

2) Efficiency: Our simulative results (see Section IV) demonstrate the efficiency of FERM (e.g., unity at link load factor of 1.5), and show that is more efficient than EPRCA.

3) Behavior Under Both Steady-State and Transient Network Conditions: The excellent performance of FERM in both steady-state and transient (unlike EPRCA whose transient behavior is very poor) is supported by the simulation results.

4) Implementation Complexity: There is no need to place a fuzzy inference engine (which is the most computationally intensive part of any fuzzy control scheme) in a real switch. After selecting appropriate fuzzy sets and rule base, and tuning the rules with a simulator, the control surface is known (see

Fig. 18) and can be stored as a lookup table for fast execution [33]. FERM can be implemented by this way, requiring a few Kbytes of ROM space and a simple interpolation algorithm. That is, it offers ease of implementation with a very fast response.

5) Fairness: The fairness of FERM (at least in a max–min sense) is demonstrated in Section IV-C. Note that fairness is achieved without the need to keep track of any other than local information. (e.g., no need to track the bottle-neck rates of individual sources, or congestion state of other switches.)

6) Ease of Tuning: As discussed earlier, since *the* control information is given in terms of linguistic variables (see Fig. 16), then it is reasonably easy to relate this to desired control performance (see earlier discussion on rule base design). This must be contrasted with the tuning of other schemes, as for example the EPRCA scheme (see Tables III and IV), where a number of parameters bearing no easy association to control performance must be tuned.

7) Scalability: As shown in the performance evaluation section, FERM scheme is capable of operating under ATM LAN or ATM WAN without the need to modify the control parameters. This is also the case for links with higher bit rates. (Note that we have not investigated the performance of the scheme when a mixture of different link rates are present in the network, but we expect that it will scale well, as the control objective is clearly stated in the linguistic variables, and it is independent of link speeds.)

8) Interworking with Other Schemes: Our scheme can be implemented by using explicit down switches (EDS's). Since the algorithm is fully decentralized, with the ER calculated at each switch based on local measurements, it can coexist with any of the other algorithms (in the same fashion as EPRCA).

9) Policing of Connections: Since the ER is placed in an RM cell, this can be provided to an appropriate policing unit at the edge of the network which will ensure that the maximum allowed rate is not exceeded by the source.

10) Minimum Cell Rate (MCR): This supports the implementation of MCR which ensures that a certain minimum portion of the network bandwidth is always provided to the connection.

VI. Conclusion

In this paper, we present and evaluate the performance (both transient and steady state in the presence of VBR and CBR traffic) and discuss some of the properties of FERM flow control algorithm for ABR traffic. We also compare its performance with the case without applying any control and the case with EPRCA control.

We show that the FERM scheme is well suited for implementation in ATM flow control in both an ATM LAN as well as an ATM WAN environment, without the need for changing or re-tuning the control law. As is evident from the simulations the same control algorithm achieves excellent delay and throughput performance in steady-state (much better than the case of no control or the case with EPRCA control) in both the ATM LAN and the ATM WAN case. Its transient behavior, in contrast to EPRCA, is excellent with low latency, quick rise time and quick settling time. FERM also achieves fairness in a max–min sense without the need for any information from either the sources or other switches in the network.

Future extensions to the scheme will include the formulation of an integrated connection admission controller (CAC), and the application of adaptive fuzzy control strategies, such as [27], to ensure that FERM remains optimally tuned under all network conditions.

Appendix A
Enhanced Proportional Rate Control Algorithm (EPRCA)

The basic features of EPRCA can be summarized as follows.

The node at which the ATM cells are generated is called SES, and the receiver of the cells is called DES. An SES periodically sends an RM cell after transmitting N_{rm} number of data cells. Before transmitting each RM cell, the SES initializes the ER field to its PCR, CCR field to its current ACR, and clears the congestion indication bit (CI). The SES reduces its cell rate if the number of forward RM cells sent since the receipt of the last RM cell exceeds a threshold (CRM)

$$\text{ACR} \leftarrow \max(\text{MCR}, \text{ACR} - \text{ACR} \times \text{CDF}).$$

When a source receives a returned RM cell, it updates its cell rate based on the values contained in the RM cell by using the algorithm shown at the bottom of the page.[2]

The DES reflects back the RM cells to the SES. But, before retransmitting them, it sets the CI bit in the received RM cell if the last received data cell had EFCI bit set.

Three types of switch architectures with different functions are described in [9] for EPRCA scheme: EFCI bit setting switches (EFCI), binary enhanced switches (BES), and explicit down switches (EDS). An EDS type switch is capable of setting an explicit rate to regulate the cell rate of the sources. To do this, the switch maintains a control parameter called mean allowed cell rate (MACR) that should ideally represent the average of the ACR's of all active connections. When the cell rates of all active connections are equal to MACR, the available bandwidth is shared fairly. MACR is computed continuously by monitoring the CCR fields of the RM cells traveling in forward direction by using exponential weighted averaging

$$\text{MACR} \leftarrow (1 - \alpha) \times \text{MACR} + \alpha \times \text{CCR}.$$

An EDS switch is considered congested if the number of cells in ABR queue is greater than SW_HT and stays in the congested state until the number of cells drops below SW_LT. If an EDS switch is congested, it sets the ER fields of the RM cells traveling in backward direction if they have a larger CCR than MACR. The value of the ER field is set to MACR $\times$ ERF (explicit reduction factor). If the switch becomes very congested (where queue length becomes greater

[2]Note that in the recently approved ATM Forum Traffic Management Specification Document [7], some of the parameter acronyms and their meanings have changed. In this particular case, AIR is replaced by RIF

$$\begin{aligned}\text{if CI} = 0 \text{ then} \quad & \text{ACR} \leftarrow \max[\text{MCR}, \min(\text{ACR} + \text{RIF} \\ & \times \text{PCR}, \text{ER}, \text{PCR})] \\ \text{else} \quad & \text{ACR} \leftarrow \max(\text{MCR}, \text{ACR} - \text{ACR} \times \text{RDF}).\end{aligned}$$

The relation between AIR and RIF can be written as RIF = AIR/PCR.

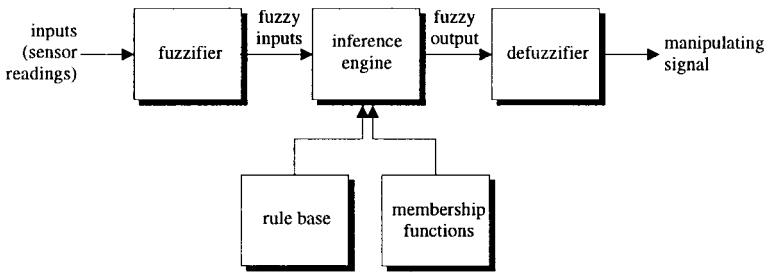

Fig. 15. Overall diagram of a fuzzy logic controller.

TABLE III
EPRCA PARAMETERS FOR ABR TRAFFIC SOURCES

PARAMETER	DEFINITION	VALUE
AIR	Additive increase rate	0.212 Mbits/sec
CDF	Cutoff decrease factor	2^{-4}
CRM	Missing RM cell count	5
ICR	Initial cell rate	= PCR
MCR	Minimum cell rate	2 Mbits/sec
PCR	Peak cell rate	149.76 Mbits/sec
RDF	Rate decrease factor	2^{-8}
N_{rm}	RM cell interval	32

TABLE IV
EPRCA PARAMETERS FOR EDS SWITCHES

PARAMETER	DEFINITION	VALUE
SW_LT	ABR queue low threshold	45
SW_HT	ABR queue high threshold	50
SW_DQT	ABR queue very congested threshold	100
SW_IMR	Initial rate for MACR	PCR/100
SW_AV	Exponential averaging factor	2^{-4}
SW_AV (LAN)	Major reduction factor for LAN	2^{-2}
SW_AV (WAN)	Major reduction factor for WAN	2^{-1}
SW_DPF	Down pressure factor	$1-2^{-3}$
SW_ERF	Explicit reduction factor	$1-2^{-4}$

than SW_DQT), the ER field of all RM cells are reduced by MACR $\times$ MRF (major reduction factor) to achieve a fast congestion relief. Furthermore, the ER field is updated only if the contents of the ER field is greater than the calculated value in order not to lose the prior switch congestion information.

For comparison purposes with the FERM scheme, we have considered only EDS type switches in our simulated ATM network. The parameters of EPRCA are shown in Tables III and IV. They are taken from [31] and [34]. We have done simulations also by using the values for parameters SW_LT = 100, SW_HT = 100 and SW_DQT = 500 as suggested in [35] and found that cell loss performance of the scheme is much worse with these values.

APPENDIX B
GENERAL STRUCTURE OF FUZZY LOGIC CONTROLLERS

Fig. 15 shows the general structure of an FLC, and Fig. 17 depicts an example of its operation which can be used to enhance one's understanding of the theory. In a multiple-input–single-output (MISO) FLC, its *rule base* contains a set of conditional statements in the form of if–then rules

$$\begin{aligned}
R^{(1)}: &\quad \text{if } X_1 \text{ is } \tilde{A}_1^{(1)} \text{ and } \cdots \text{ and } X_n \text{ is } \tilde{A}_n^{(1)} \text{ then } Y \text{ is } \tilde{B}^{(1)}\\
R^{(2)}: &\quad \text{if } X_1 \text{ is } \tilde{A}_1^{(2)} \text{ and } \cdots \text{ and } X_n \text{ is } \tilde{A}_n^{(2)} \text{ then } Y \text{ is } \tilde{B}^{(2)}\\
&\vdots \qquad\qquad \vdots \qquad\qquad \vdots\\
R^{(m)}: &\quad \text{if } X_1 \text{ is } \tilde{A}_1^{(m)} \text{ and } \cdots \text{ and } X_n \text{ is } \tilde{A}_n^{(m)} \text{ then } Y \text{ is } \tilde{B}^{(m)}
\end{aligned}$$

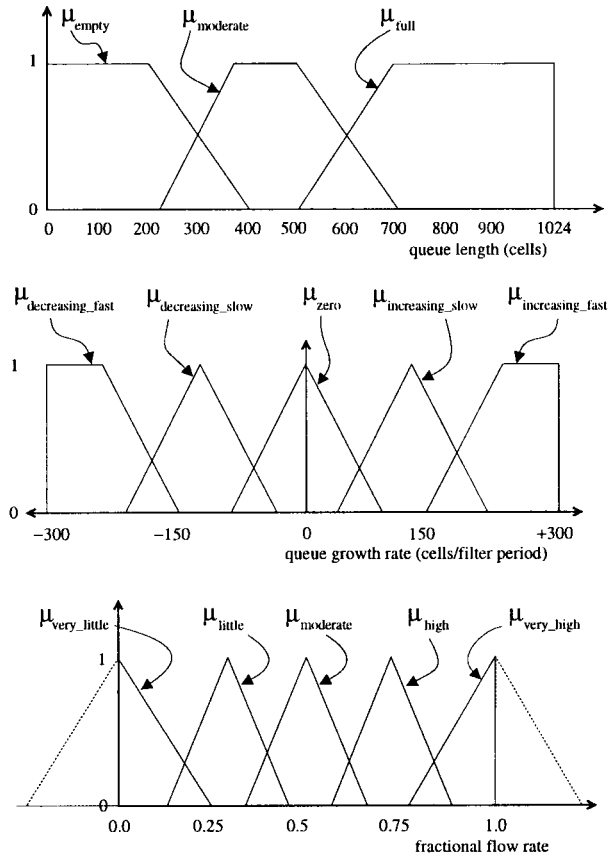

Fig. 16. Membership functions of the linguistic values for representing the linguistic variables "queue length," "queue growth rate," and "flow rate."

here, $X_1, X_2, \cdots, X_n$ and Y are linguistic variables, and $\tilde{A}_1^{(l)}, \tilde{A}_2^{(l)}, \cdots, \tilde{A}_n^{(l)}$ and $\tilde{B}^{(l)}$ $(l = 1, \cdots, m)$ are the linguistic values of inputs and output. Each linguistic value $\tilde{A}_k^{(l)}$ and $\tilde{B}^{(l)}$ $(k = 1, \cdots, n)$ is a fuzzy set and characterized by its membership function $\mu_{\tilde{A}_k^{(l)}}(x)$ and $\mu_{\tilde{B}^{(l)}}(y)$, respectively (see Fig. 16), x and y being the elements of universal sets U and V. Each rule defines a relation between the linguistic variables $\tilde{A}_1^{(l)}, \tilde{A}_2^{(l)}, \cdots, \tilde{A}_n^{(l)}$ and $\tilde{B}^{(l)}$. This suggests that a rule be defined as a fuzzy implication

$$\tilde{R}^{(l)}: \quad \tilde{A}_1^{(l)} \times \cdots \times \tilde{A}_n^{(l)} \rightarrow \tilde{B}^{(l)}.$$

In other words, it performs a mapping from fuzzy input state-space to a fuzzy output value.

The *inference engine* operates by using the dual concepts of generalized modus ponens and compositional rule of inference [21]. To emphasize the contribution of fuzzy logic to robust

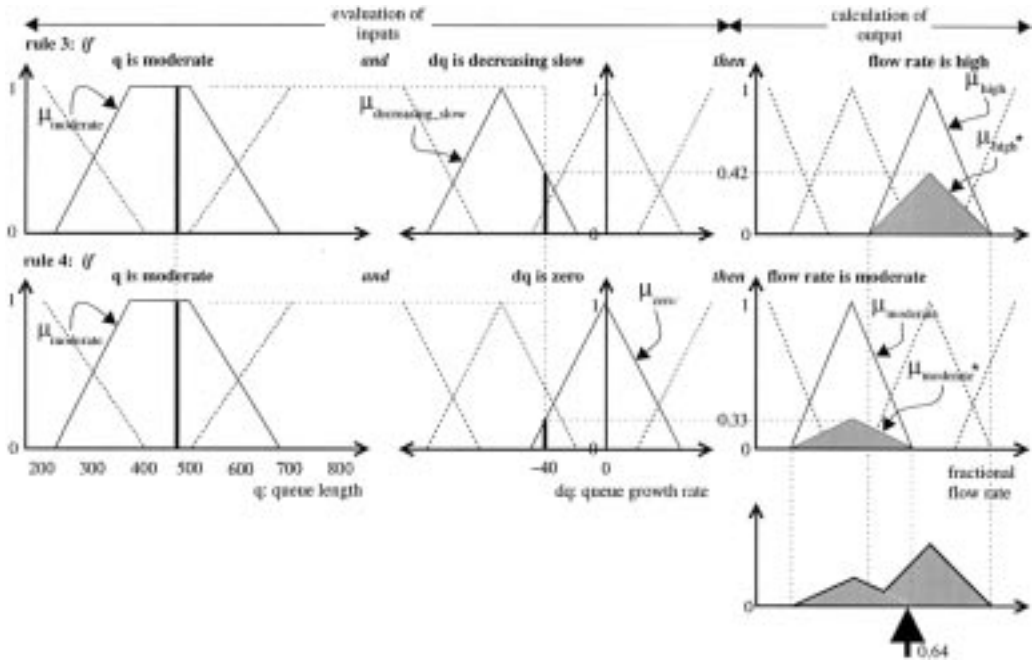

Fig. 17. An example of computing the fractional flow rate: assume that at the end of two successive control intervals (past and current) average ABR queue lengths are read as 515 and 475 cells, respectively; therefore, queue rate growth is -40 and the current length is 475. Note that the current length is a member of the fuzzy set "moderate" to a full extent and the change of queue length is a member of the fuzzy sets decreasing slow" and "zero" with membership values of 0.42 and 0.33. Their membership of any other fuzzy sets is zero. Each rule (see Fig. 19) is visited, and the minimum of membership values of the inputs to the corresponding linguistic values are found (3) and then the corresponding (for each rule) output linguistic value is scaled accordingly (6). That is, with the numerical value of input variables used in this example, only rule 3 and rule 4 (Fig. 19) yield nonzero scaling factors (0.42 and 0.33) and thus contribute in the calculation of the output. Considering rule 3, the output (flow rate) is "high," scaled by the min(1, 0.42) and the output due to rule 4 is "moderate" scaled by the min(1, 0.33); see the shaded regions in the top two curves of the "calculation of the output" section of the figure. For the calculation of the output, we take the union of all scaled output fuzzy sets (4). Then by using the center of gravity (CoG) defuzzification method (7), a numerical value for the output variable (fractional flow rate) is calculated as 0.64 in this example.

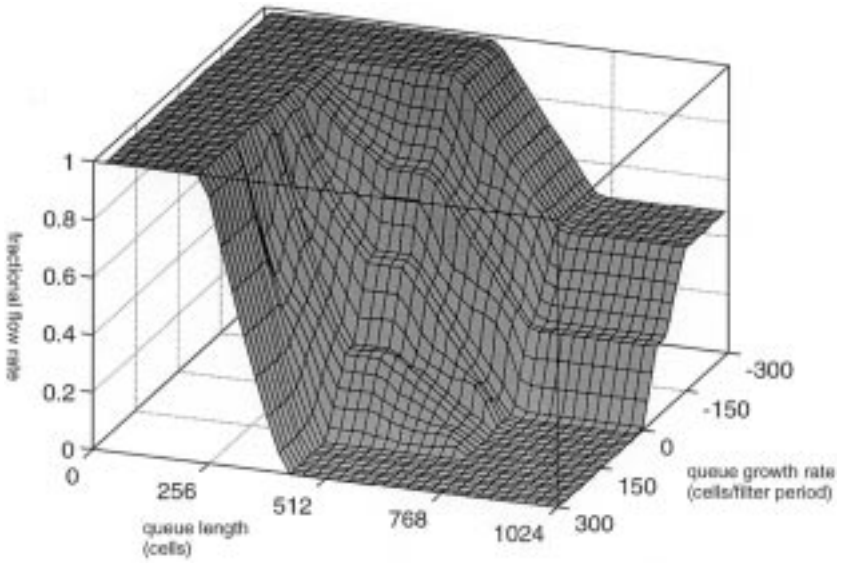

Fig. 18. Control surface of the FCC. The control surface is shaped by the rule base and the linguistic values of the linguistic variables. By observing the progress of simulation, and modifying the rules and definitions of the linguistic values, we can tune the FCC to achieve better server utilization, and lower cell loss coupled with minimal end-to-end cell delay.

reasoning, it is worthwhile to touch upon the operation of modus ponens, or direct reasoning of binary logic first. Modus ponens allows us to draw a conclusion from two premises; assume that we have the proposition p: "x is A" and the implication $p \rightarrow q$: "if x is A then y is B" as true, we can immediately conclude that the proposition q: "y is B" has to be true. In the context of a control system, implication $p \rightarrow q$ represents our knowledge about the operation of the system: "if p happens q should happen." Here, the problem is that reasoning collapses if something happens which does not exactly match our existing knowledge. Generalized modus ponens (GMP) of fuzzy logic offers a solution: If something

```
/* input variable: queue length */
lingvar q on [0,1024] initially 0.0 with
     lingvalue empty     is    0   0  200  400
     lingvalue moderate  is  220 380  580  820
     lingvalue full      is  600 830 1024 1024
end

/* input variable: rate of change in queue length */
lingvar dq on [-300,300] initially 0.0 with
     lingvalue decreasing_fast is -500 -500 -200 -85
     lingvalue decreasing_slow is -158  -78  -78 -12.5
     lingvalue zero            is  -60    0    0  60
     lingvalue increasing_slow is  12.5  78   78 158
     lingvalue increasing_fast is   85  200  500 500
end

/* output variable: flow rate limit for ABR source */
lingvar flow_rate on [-0.20,1.20] initially 1.0 with
     lingvalue very_little is -0.20 0    0    0.20
     lingvalue little      is  0.05 0.25 0.25 0.45
     lingvalue moderate    is  0.30 0.5  0.5  0.70
     lingvalue high        is  0.55 0.75 0.75 0.95
     lingvalue very_high   is  0.80 1    1    1.2
end

/* linguistic rules */
if q is empty then flow_rate is very_high;                                  /* r1 */
if q is moderate and dq is decreasing_fast then flow_rate is very_high;     /* r2 */
if q is moderate and dq is decreasing_slow then flow_rate is high;          /* r3 */
if q is moderate and dq is zero then flow_rate is moderate;                 /* r4 */
if q is moderate and dq is increasing_slow then flow_rate is little;        /* r5 */
if q is moderate and dq is increasing_fast then flow_rate is very_little; /* r6 */
if q is full and dq is decreasing_fast then flow_rate is moderate;          /* r7 */
if q is full and dq is decreasing_slow then flow_rate is little;            /* r8 */
if q is full and dq is zero then flow_rate is very_little;                  /* r9 */
if q is full and dq is increasing_slow then flow_rate is very_little;       /* r10*/
if q is full and dq is increasing_fast then flow_rate is very_little;       /* r11*/
```

Fig. 19. C-FLIE source file for defining the rules and linguistic variables of FCC. Linguistic values of a linguistic variable are defined as trapezoidals (Fig. 16) by identifying their vertices.

happens which does not match, but is somewhat similar to the existing knowledge, we can still draw a conclusion. Assume that we have the propositions $\tilde{p}$: "X is $\tilde{A}$" and $\tilde{q}$: "Y is $\tilde{B}$" where X, Y are linguistic variables and $\tilde{A}$, $\tilde{B}$ are their linguistic values respectively. If we have the implication $\tilde{p} \rightarrow \tilde{q}$: "if X is $\tilde{A}$ then Y is $\tilde{B}$" and proposition $\tilde{p}^*$: "X is $\tilde{A}^*$" where $\tilde{A}^*$ is not necessarily the same as $\tilde{A}$, we can use GMP to perform the inferencing

$$\frac{\begin{array}{ll}\text{if} & X \text{ is } \tilde{A} \text{ then } \quad Y \text{ is } \tilde{B} \\ & X \text{ is } \tilde{A}^*\end{array}}{\therefore \qquad\qquad Y \text{ is } \tilde{B}^*}.$$

The membership function of $\tilde{B}^*$ is calculated by using the sup-$\star$ compositional rule of inference

$$\begin{aligned}\mu_{\tilde{B}^*}(y) &= \mu_{\tilde{A}^* \circ \tilde{R}}(y) \\ &= \sup_x [\mu_{\tilde{A}^*}(x) \star \mu_{\tilde{A} \rightarrow \tilde{B}}(x, y)]. \qquad \text{(A.1)}\end{aligned}$$

The symbol "$\star$" denotes the t-norm operator [p. 45, 37] and $\mu_{\tilde{A} \rightarrow \tilde{B}}(x, y)$ is the membership function of fuzzy set corresponding to implication $\tilde{p} \rightarrow \tilde{q}$. Several different definitions of fuzzy implication have been reported and compared in the literature [38], [39]. In FLC's, usually Larsen's product operation rule of fuzzy implication is used to represent a linguistic rule

$$\begin{aligned}\mu_{\tilde{R}^{(l)}} &= \mu_{\tilde{A}^{(l)} \rightarrow \tilde{B}^{(l)}}(x, y) \\ &= \mu_{\tilde{A}^{(l)}}(x) \mu_{\tilde{B}^{(l)}}(y).\end{aligned}$$

Since, most engineering applications involve multiple input variables, application of GMP can easily be extended to these cases by interpreting the fuzzy set $\tilde{A}^{(l)}$ as the product of fuzzy sets $\tilde{A}_1^{(l)}, \cdots, \tilde{A}_n^{(l)}$ where its membership function is defined as

$$\begin{aligned}&\mu_{\tilde{A}_1^{(l)} \times \cdots \times \tilde{A}_n^{(l)}}(x_1, \cdots, x_n) = \\ &\quad \mu_{\tilde{A}_1^{(l)}}(x_1) \star \mu_{\tilde{A}_2^{(l)}}(x_2) \star \cdots \star \mu_{\tilde{A}_n^{(l)}}(x_n). \qquad \text{(A.2)}\end{aligned}$$

Again, "$\star$" represents the t-norm operator. T-norms are a family of functions, and in this case, used for defining conjunctions in approximate reasoning and corresponds to the connective "and" in the rules of the rule base. Usual selection

of FLC designers is the intersection definition of t-norm: $u \star w = \min(u, w)$. By applying this definition, we rewrite (2) as

$$\mu_{\tilde{A}_1^{(l)} \times \cdots \times \tilde{A}_n^{(l)}}(x_1, \cdots, x_n) = \min[\mu_{\tilde{A}_1^{(l)}}(x_1)\mu_{\tilde{A}_2^{(l)}}(x_2), \cdots, \mu_{\tilde{A}_n^{(l)}}(x_n). \quad \text{(A.3)}$$

A rule base contains more than one rule. We can evaluate each rule independently from the others (i.e., calculate $\mu_{\tilde{B}^{(l)*}}(y)$'s by using (1) if we model implications by using a conjunction (such as Larsen's product operation)). Then, overall decision of the system can be obtained by taking the union of fuzzy sets $\tilde{B}^{(l)*}$ $(l = 1, \cdots, m)$

$$\mu_{\tilde{B}^{(1)*} \cup \cdots \cup \tilde{B}^{(m)*}}(y) = \mu_{\tilde{B}^{(1)*}}(y) \dot{+} \cdots \dot{+} \mu_{\tilde{B}^{(m)*}}(y) \quad \text{(A.4)}$$

where "$\dot{+}$" represents s-norm operator [37] (p. 49). S-norms are a family of functions for defining disjunctions in approximate reasoning. In practical FLC designs, usually union definition of s-norm is used: $u \dot{+} w = \max(u, w)$.

A fuzzy inference engine maps fuzzy sets to fuzzy sets. In control engineering applications we almost always deal with numerical values. *Fuzzifier* and *defuzzifier* modules act as interfaces between linguistic world of the fuzzy inference engine and numerical world. Fuzzifier module takes a numerical value x_{in} then maps it to a fuzzy set $\tilde{A}^*$. If there is no uncertainty in the numerical reading, this fuzzy set $\tilde{A}^*$ is expressed as a singleton

$$\mu_{\tilde{A}^*}(x) = \begin{cases} 1, & \text{if } x = x_{in} \\ 0, & \text{if } x \neq x_{in} \end{cases}$$

If, for some reason, there is uncertainty in the numerical reading and we would like to express the uncertainty while we carry out fuzzification, mapping can be done to a fuzzy set. In this case, membership values of the elements of $\tilde{A}^*$ can be selected such that, $\mu_{\tilde{A}^*}(x)$ is taken as 1 if $x = x_{in}$ and $\mu_{\tilde{A}}(x)$ decreases from one as x moves away from x_{in}. Note that, if singleton fuzzification is selected, since $\tilde{A}^*$ will contain only a single element (with membership value equal to 1) at $x = x_{in}$, the supreme operation in (1) disappears and calculation of $\mu_{\tilde{B}^*}(y)$ becomes simpler:

$$\mu_{\tilde{B}^*}(y) = \mu_{\tilde{A} \to \tilde{B}}(x_{in}, y)$$

furthermore, if we choose Larsen's product operation rule of implication,

$$\mu_{\tilde{B}^*}(y) = \mu_{\tilde{A}}(x_{in})\mu_{\tilde{B}}(y) \quad \text{(A.5)}$$

and in the case of multiple input variables, we substitute (3) to (5) and obtain

$$\mu_{\tilde{B}^*}(y) = \min[\mu_{\tilde{A}_1}(x_{1_{in}}), \cdots, \mu_{\tilde{A}_n}(x_{n_{in}})]\mu_{\tilde{B}}(y) \quad \text{(A.6)}$$

to compute the decision of the one rule of the inference engine. Finally, we find the overall decision of the inference engine by aggregating the calculated $\tilde{B}^{(l)*}$'s by applying (4).

Since the decision of the inference engine is a fuzzy set, in order to be able to use it as a control signal in a physical system, it has to be mapped to a numerical value in $\mathbb{R}$. *Defuzzifier* module produces a nonfuzzy output whose objective is to represent the possibility distribution of the inference. There is no single method for performing the defuzzification. In fuzzy control applications, because of the ability of generating smoother control surface, usually the CoG method is used. This method divides the first moment of area under the membership function calculated by (6) into half, and the defuzzified value y_{out} marks the dividing line. Mathematically, CoG can be expressed as

$$y_{out} = \frac{\int y\mu_{\tilde{B}^*}(y)\,dy}{\int \mu_{\tilde{B}^*}(y)\,dy}. \quad \text{(A.7)}$$

Operation of an FLC is illustrated in Fig. 17 using the FCC as an example.

ACKNOWLEDGMENT

The authors thank Prof. J. Lambert of the Swinburne University of Technology for useful discussions.

REFERENCES

[1] A. Pitsillides, Y. A. Şekercioğlu, and G. Ramamurthy, "Fuzzy backward congestion notification (FBCN) congestion control in asynchronous transfer mode (ATM)," in *Proc. IEEE Conf. GLOBECOM'95*, 1995, pp. 280–285.
[2] J. Bae and T. Suda, "Survey of traffic control schemes and protocols in ATM networks," *Proc. IEEE*, Feb. 1991.
[3] O. Aboul-Magd and H. Gilbert, "Incorporating congestion feedback in B-ISDN traffic management strategy," in *Proc. ISS'92 Int. Switching Symp.*, Osaka, Japan, Oct. 1992, pp. 12–16.
[4] C. Ikeda and H. Suzuki, "Adaptive congestion control schemes for ATMLAN's," in *Proc. IEEE INFOCOM'94 Conf.*, 1994, pp. 829–838.
[5] P. Newman, "Backward explicit congestion notification for ATM local area networks," in *Proc. GLOBECOM'93*, Houston, TX, 1993, pp. 719–723.
[6] ——, "Traffic management for ATM local area networks," *IEEE Commun. Mag.*, Aug. 1994.
[7] ATM Forum, "Traffic management specification version 4.0," Tech. Rep. AF-TM-0056.000, Apr. 1996.
[8] A. Charny, D. Clark, and R. Jain, "Congestion control with explicit rate indication," Tech. Rep. AF-TM 94-0692, July 1994, (updated version published in Proc. ICC'95, June 1995).
[9] L. Roberts, "Enhanced PRCA (proportional rate-control algorithm)," Tech. Rep. AF-TM 94-0735R1, Aug. 1994.
[10] R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, and F. Lu, "ERICA+: Extensions to the ERICA switch algorithm," Tech. Rep. ATM Forum Contribution AF-TM 95-1145R1, Oct. 1995.
[11] F. Bonomi and K. W. Fendick, "The rate-based flow control framework for the available bit rate ATM service," *IEEE Network*, pp. 25–39, 1995.
[12] C. Chang and R. Cheng, "Traffic control in an ATM network using fuzzy set theory," in *Proc. IEEE INFOCOM'94 Conf.*, vol. 40, pp. 1200–1207.
[13] D. Jensen, "B-ISDN network management by a fuzzy logic controller," in *Proc. IEEE GLOBECOM'94*, San Francisco, CA, 1994, pp. 799–804.
[14] J. A. Smith and M. Fry, "Artificial intelligence in network management," *Austr. J. Intel. Inform. Process. Syst.*, pp. 53–62, Autumn 1995.
[15] P. Chemouil, J. Khalfet, and M. Lebourges, "A fuzzy control approach for adaptive traffic routing," *IEEE Commun. Mag.*, pp. 70–76, July 1995.
[16] L. A. Zadeh, "Fuzzy sets," *Inform. Cont.*,, vol. 8, pp. 338–353, 1965.
[17] ——, "Fuzzy algorithms," *Inform. Cont.*, vol. 12, pp. 94–102, 1968.
[18] ——, "A rationale for fuzzy control," *J. Dynamic Sys., Measurement, Cont.*, vol. 94, no. series G, pp. 3–4, 1972.
[19] ——, "Outline of a new approach to the analysis of complex systems and decision processes," *IEEE Trans. Sys., Man, Cybern.*, vol. SMC-3, no. 1, pp. 28–44, Jan. 1973.
[20] ——, "A fuzzy-algorithmic approach to the definition of complex or imprecise concepts," *Int. J. Man-Machine Studies*, vol. 8, pp. 249–291, 1976.
[21] ——, "Fuzzy logic," *IEEE Comput.*, pp. 83–93, Apr. 1988.
[22] K. Oishi, M. Tominaga, A. Kawato, Y. Abe, S. Imayasu, and A. Nanba, "Application of fuzzy control theory to the sake brewing process," *J. Fermentation Bioeng.*, vol. 72, no. 2, pp. 115–121, 1991.

[23] S. Daley and K. F. Gill, "Comparison of a fuzzy logic controller with a $P + D$ control law," *Trans. ASME,* vol. 111, pp. 128–137, June 1989.
[24] D. Driankov, H. Hellendoorn, and M. Reinfrank, *An Introduction to Fuzzy Control.* New York: Springer-Verlag, 1993.
[25] W. Pedrycz, *Fuzzy Sets Engineering.* Boca Raton, FL: CRC Press, 1995.
[26] R. Jager, "Fuzzy logic in control," Ph.D. dissertation, Technische Universiteit Delft, The Netherlands, 1995.
[27] Y. A. Şekercioğlu, A. Pitsillides, and G. K. Egan, "Study of an adaptive fuzzy controller based on the adaptation of relative rule weights," in *Proc. ANZIIS'94*, Brisbane, Queensland, Australia, Nov. 1994, pp. 204–208.
[28] Y. A. Şekercioğlu, "Fuzzy logic control techniques for asynchronous transfer mode (ATM) based multimedia networks," Ph.D. dissertation, Swinburne University of Technology, 1996.
[29] E. H. Mamdani, "Twenty years of fuzzy control: Experiences gained and lessons learned," in *Proc. Second IEEE Int. Conf. Fuzzy Systems*, San Francisco, CA, Mar. 1993, pp. 339–344.
[30] B. Maglaris, D. Anastassiou, P. Sen, G. Karlsson, and J. D. Robbins, "Performance models of statistical multiplexing in packet video communications," *IEEE Trans. Commun.*, vol. 36, no. 7, pp. 834–843, July 1988.
[31] R. Jain, "Congestion control and traffic management in ATM networks: Recent advances and a survey," *Comp. Networks and ISDN Systems*, 1995.
[32] C. E. Rohrs, R. A. Berry, and S. J. O'Halek, "A control engineer's look at ATM congestion avoidance," in *Proc. IEEE Conf. GLOBECOM'95*, 1995.
[33] P. P. Bonissone, "A compiler for fuzzy logic controllers," in *Fuzzy Eng. Toward Human Friendly Sys. IFES'91* IOS Press, 1992.
[34] R. Jain, S. Kalyanaraman, and R. Viswanathan, "Transient performance of EPRCA and EPRCA++," ATM Forum Contribution 94-1173, Tech. Rep., Nov. 1994.
[35] H. Ohsaki, M. Murata, H. Suzuki, C. Ikeda, and H. Miyahara, "Analysis of rate-based congestion control algorithms for ATM networks—Part I: Steady state analysis," in *Proc. IEEE Conf. GLOBECOM'95*, vol. 40.
[36] C. C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller—Part I," *IEEE Trans. Sys., Man, Cybern.*, vol. 20, no. 2, pp. 404–418, Mar./Apr. 1990.
[37] C. J. Harris, C. G. Moore, and M. Brown, *Intelligent Control: Aspects of Fuzzy Logic and Neural Nets, World Scientific Series in Robotics and Automated Systems.* Singapore: World Scientific, 1993, vol. 6.
[38] M. Mizumoto and H. Zimmermann, "Comparison of fuzzy reasoning methods," *Fuzzy Sets Syst.*, vol. 8, pp. 253–283, 1982.
[39] D. Dubois and H. Prade, "Fuzzy sets in approximate reasoning, part 1: Inference with possibility distributions," *Fuzzy Sets Syst.*, vol. 40, pp. 143–202, 1991.
[40] IEEE Communications Society, *Proceedings of the IEEE Global Telecommunications Conference GLOBECOM'95*, Singapore, Nov. 1995.
[41] ——, *Proc. IEEE INFOCOM'94 Conference*, Toronto, Canada, 1994.

Andreas Pitsillides received the Undergraduate degree in electrical and electronic engineering from the University of Manchester Institute of Science and Technology, Manchester, U.K., in 1980, and the Ph.D. degree in high speed multimedia networks from the Swinburne University of Technology, Melbourne, Australia, in 1993.

He has worked in the industry for six years (from 1980 to 1986), and from 1987 to 1994 he taught at the Swinburne University of Technology (Lecturer, Senior Lecturer 1990–1994, and Foundation Director of the Swinburne Laboratory for Telecommunications Research, 1992–1994). In 1992, he spent six months as an Academic Visitor at the Telstra (Australia) Telecom Research Labs (TRL). Since January 1995, he has been an Assistant Professor with the Department of Computer Science, University of Cyprus. His research interests include high speed multiservice and multimedia networks, control structures and techniques, intelligent control and signal processing (e.g., fuzzy, neural, and adaptive control, and higher order spectra), integrated control, and their application to solve network problems, network survivability, optimal resource allocation, switching fabrics for ATM networks, and nonstationary (transient) queuing systems theory.

Y. Ahmet Şekercioğlu received the B.Sc. and M.Sc. degrees from the Department of Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey, in 1982 and 1985, respectively.

He is working as a Lecturer at the School of Computer Science and Software Engineering, Swinburne University of Technology, Melbourne, Australia. His research interests include high speed multiservice and multimedia networks, intelligent control, fuzzy logic, and genetic programming.

Gopalakrishnan Ramamurthy received the Masters degree in electrical communication engineering from the Indian Institute of Science, New Delhi, India, and the Ph.D. in electrical engineering from the University of Aston, Birmingham, U.K.

He was with the Indian Telephone Industries, Bangalore, India, from 1974 to 1980. He was a Member of the Technical Staff at AT&T Bell Laboratories, NJ, from 1984 to 1991. He is currently a Senior Research Staff Member with the Systems Architecture Department, C&C Research Laboratories, NEC, Princeton, NJ. His current research interests are in broad-band network architecture and control, multiclass resource and QoS management, and transport for multimedia communication.