TCP Performance in ATM Networks: ABR Parameter Tuning and ABR/UBR Comparisons

Chien Fang
Sandia National Labs
P.O. Box 969, MS 9011
Livermore, CA 94551-0969
cfang@ca.sandia.gov

Arthur Lin
Cisco Systems, Inc.
170 West Tasman Dr.
San Jose, CA 95134-1706
alin@cisco.com

February 27, 1996

Abstract

This paper explores two issues on TCP performance over ATM networks: ABR parameter tuning and performance comparison of binary mode ABR with enhanced UBR services. Of the fifteen parameters defined for ABR, two parameters dominate binary mode ABR performance: Rate Increase Factor (RIF) and Rate Decrease Factor (RDF). Using simulations, we study the effects of these two parameters on TCP over ABR performance. We compare TCP performance with different ABR parameter settings in terms of throughputs and fairness. The effects of different buffer sizes and LAN/WAN distances are also examined. We then compare TCP performance with the best ABR parameter setting with corresponding UBR service enhanced with Early Packet Discard and also with a fair buffer allocation scheme. The results show that TCP performance over binary mode ABR is very sensitive to parameter value settings, and that a poor choice of parameters can result in ABR performance worse than that of the much less expensive UBR-EPD scheme.

Keywords: ATM flow control, TCP over ATM, Available Bit Rate (ABR) service, Early Packet Discard (EPD).

1 Introduction

ATM is an emerging technology designed to support a wide variety of services, including video, voice and data. The ATM Forum defines five service categories for different application requirements. For real-time applications, there are Constant Bit Rate (CBR) and Real-Time Variable Bit Rate (rt-VBR) service categories, which provide numerical guarantees in cell transfer rates, cell transfer delay and delay variations. For non-real-time applications, there are Non-Real-Time Variable Bit Rate (nrt-VBR), Available Bit Rate (ABR) and Unspecified Bit Rate (UBR) service categories. Nrt-VBR provides some degree of guarantee in cell transfer rates but no guarantee on delay bounds. Both ABR and UBR are designed to utilize the left-over bandwidth of CBR and VBR (i.e., guaranteed) services. To achieve this goal, ABR employs a feedback-based flow control mechanism to minimize cell loss. However, there is no quantitative assurance for ABR service. UBR is the true "best-effort" service which employs no flow control and, as ABR, provides no guarantee on cell loss ratios, transfer rates or transfer delay. Of the three non-real-time services, ABR and UBR are expected to be the primary services for carrying traditional data traffic, i.e., file transfer, email and telnet sessions etc.

The basic idea behind the ABR flow control scheme is using network feedback to adjust source rates. Central to the ABR flow control are information carried in Resource Management (RM) cells. ABR sources periodically probe the network state (such as bandwidth availability, state of congestion and impending congestion) with RM cells sent intermixed with data cells. The RM-cells are turned around at the destination and sent back to the source. Along the way, ATM switches can write congestion information (e.g., congestion flag, sustainable rates etc.) into these RM cells. Upon receiving a returned RM cell, a source can then increase, decrease or maintain its rate depending on the information contained therein.

The ABR flow control algorithm supports two modes of switch operation: binary mode and explicit rate mode. In the binary mode, when an ATM switch queueing point detects incipient congestion the switch may either set the Explicit Forward Congestion Indication (EFCI) state in the headers of outgoing data cells or set the congestion indication (CI) or No Increase (NI) bit in forward and/or backward RM-cells. In the former case, termed *EFCI marking*, the EFCI state informantion is saved at the destination and transfered to the CI bit of a turn-around RM cell. In the latter approach, termed *Relative Rate marking*, the congestion indication can be written directly into the backward RM-cell instead of relying on the destination end-system to turn it around. Relative Rate marking can greatly reduce feadback delays and hence deliver better performance than EFCI marking. Upon receiving a returned RM cell, a source can either perform an additive increase (AI) or multiplicative decrease (MD) in its rate, depending on whether the CI (or NI) bit has been set or not. The performance of ABR under this operating mode thus mainly depends on the magnitude of AI and MD.

In the explicit rate (ER) mode, an ATM switch employs a control algorithm for managing the bandwidth among the virtual circuits (VCs) traversing the switch. In addition to load monitoring and congestion detection, the ER switch need to perform an explicit rate calculation function. The switch algorithm continuously monitors and computes maximum allowed cell rate (ACR) for each of its active VCs. The maximum value of ACR for each VC, called ER, is then written into corresponding RM cells (containing a higher value of ER). When a source receives a returned RM cell, its rate is adjusted by the ER rate contained therein.

Switch algorithms are not standardizied by the ATM Forum, and are therefore implementation specific. A number of switch algorithms have been proposed recently [1, 7, 9, 15].

UBR service employs no flow control and provides no numerical guarantee; it is therefore the least expensive service to provide. However, because of its simplicity, UBR performs poorly for TCP applications in a congested network. Early Packet Discard (EPD) [5] has been proposed as an inexpensive enhancement to improve the performance of UBR services. EPD is based on the observation that one single cell loss will render the entire packet useless, and thus it is more efficient to discard a complete packet rather than individual cell from multiple packets/connections during congestion. The EPD scheme is relatively inexpensive to implement on ATM switches, compared with the ABR flow control scheme. Although UBR-EPD offers improved TCP performance over pure UBR service, it does not address the fairness issue. To improve upon fairness in UBR-EPD the Fair Buffer Allocation (FBA) sheme [6] was proposed. FBA uses per-VC accounting to improve VC fairness for an ATM switch with FIFO buffers.

This paper will examine TCP performance over ATM networks using ABR and UBR services. For ABR service we only consider binary mode switches since there exists a large number of ATM switches which only support EFCI marking. Also the performance of ER operation is almost completely determined by the switch algorithm, which is not standardized. A performance comparison of different switch algorithms proposed to date is beyond the scope of this paper.

In this paper we will study the effect of ABR parameter tuning (parameters corresponding to AI and MD) on TCP performance. We will also compare TCP performance under binary mode ABR with that of UBR enhanced by EPD and UBR enhanced by EPD and FBA.

The rest of the paper is organized as follows. Section 2 briefly describes ABR flow control, ABR parameters, an Early Packet Discard scheme, and a Fair Buffer Allocation scheme. In Section 3, we then present the network model and simulation tools used. We discuss simulation results and compare the performance of TCP over ABR and UBR services in Section 4 and conclude in Section 5.

2 Background

2.1 ABR Flow Control

The ABR flow control algorithm specified by the ATM Forum [14] consists of three parts: source behavior, destination behavior and switch behavior. Each part is made up of a set of rules, and a total of fifteen parameters are defined for the algorithm. For the purpose of this paper, the most relevant source rule to ABR performance under binary switch mode are summarized as follows:

Rule 1: An RM-cell is sent for every Nrm data cells sent.[1]

This rule generates periodic RM cells interspersed in data cell streams. Nrm determines the frequency of RM-cell generation and also the overhead incurred by the flow control scheme.

[1]This is a simplification of the actual source rule, which includes conditions for maintaining minimum forward and backward RM-cell flow rates.

Rule 2: When a backward RM-cell is received with CI=1, then
ACR = MAX(ACR (1−RDF), MCR);
If the backward RM-cell has CI=0, then
ACR = MAX(ACR + RIF × PCR, PCR)

where CI is set to 1 by a congested ATM switch or rate-limited destination end system; ACR is the allowed source sending rate; RDF is the rate decrease (or reduction) factor; MCR is the minimum cell rate, RIF is the rate increase factor; PCR is the peak allowed source cell rate. This rule determines the multiplicative decrease when congestion occurs and additive increase when congestion clears.

The destination behavior consists of saving the EFCI state of incoming data cells, turning around forward RM-cells, and copying of the saved EFCI state into the CI bit of backward RM-cells.

The switch behavior under the binary mode consists of setting congestion flags in the passing cells. Two operation modes are specified: EFCI marking and Relative Rate Marking. In EFCI marking, the EFCI state in the data cell header is set; in Relative Rate marking the CI bit in the forward and/or backward RM-cells is set during congestion; when CI bit in the backward RM-cells is set, it is generally referred to as the *BECN* mode.

In this paper, we assume the BECN mode, as it yields better performance. We also assume FIFO output buffers with a simple congestion threshold. Under this operating mode, when the buffer occupancy exceeds the threshold, a congestion flag is set and the CI field of backward RM-cells are marked. When the buffer occupancy drops below the threshold, the congestion flag is cleared. The BECN mode provides faster feedback of congestion information to the source, although at a slightly higher implementational complexity in switches.

2.2 Early Packet Discard

While short, fixed-length ATM cells facilitate high-speed hardware switching, the majority of data units for applications/end-systems are variable-length. The term "packet" will be used to denote these different higher level protocol data units for the rest of this paper. Early Packet Discard (EPD) is a mechanism designed to optimize the performance of packet data traffic (such as IP and 802.x LANs) over ATM. Recognizing that a single cell loss causes an entire packet to be corrupted, EPD selectively discards cells belonging to the same packets when congestion occurs. Thus when cell loss is deemed unavoidable, EPD attempts to confine cell loss to as few connections as possible. EPD has the effect of freeing up buffers which otherwise would have been used by cells belonging to corrupted packets. Specifically, the EPD mechanism implemented in this study works as follows. When the first cell of a packet arrives at a switch queueing point, (1) if it is not congested, this cell and all the remaining cells of the packet shall be allowed to enter, provided that there is enough buffer space; (2) if it is congested, the cell and all subsequent cells of the same packet will be dropped. By dropping a small number of packets instead of cells from a large number of packets, EPD maximizes the system goodput [5]. Note that EPD is designed to enhance the performance of any packet traffic over ATM, and can be used in conjunction with both UBR and ABR services.

2.3 Fair Buffer Allocation

While EPD improves the performance of data traffic in ATM networks by eliminating bandwidth wastage, it contains no provision for providing fair bandwidth allocation among contending VCs. Depending on the frequency of encountering a congested queue, contending VCs at a bottleneck node may experience different packet loss ratios. Fair (per-VC) queueing and its weighted version (WFQ) shall provide the ideal fairness [13], however they may be costly in some cases such as high speed switching ports/links (STS-12c and higher). An alternative to achieve fairness is to employ per-VC accounting on FIFO buffers. One such scheme, the Fair Buffer Allocation (FBA) [6] algorithm, was proposed recently. FBA maintains per-VC usage statistics on all active VCs and performs packet discard on those VCs which have exceeded their fair share of the buffer usage. The FBA examined in this paper is implemented as follows. During congestion the first cell of a packet will be dropped if the VC has more than its fair share of the buffer usage, otherwise the cell will be accepted. The decision to drop the first cell of a packet is determined by the following formula:

$$(X > R) \text{ and } W_i > Z\left(1 + \frac{K - X}{X - R}\right) \tag{1}$$

where X is the total number of cells in queue, R is the congestion threshold, K is the queue capacity, Z is a free parameter ($0.5 \leq Z \leq 1.0$), and W_i is a measure of VC(i)'s usage of buffers, computed by multiplying the number of VC(i)'s cells in queue by the number of active VCs[2] and divided by X, the total number of cells in queue.

The bulk of the implemention complexity for FBA is in the amount of computation required during every cell time, as indicated by equation (1). However, equation (1) can be simplified to minimize computational requirements, and some possible approaches using lookup tables are described in [6].

2.4 Related Works

Performance of TCP over ATM networks was first examined by Romanow et al. in [5]. They showed that TCP performs poorly over ATM using (plain) UBR service and proposed EPD to improve TCP performance. Fang et al. [2] presented results on TCP performance over ATM using UBR and a number of congestion control schemes including DEC's credit-based ABR scheme and two selective cell-drop schemes. Their results showed that per-VC buffering and selective cell-drop can lead to significant performance improvement. Li et al. [11] presented simulation results on TCP performance over ABR and UBR services. They compared TCP performance over ABR with ER switch mode and UBR with EPD, and concluded that ABR with ER switches outperforms UBR with EPD in a LAN environment. Kalampoukas et al. [8] compared TCP performance of UBR-EPD with FCVC, the credit-based flow control scheme proposed by Kung [10]. They claimed that the UBR-EPD scheme is able to remove some of the unfairness and provide performance close to that of a datagram network as long as the buffer sizes are not too small. The FCVC provides the best performance among the schemes studied.

[2] a VC is active if it has at least one cell in queue.

3 Network Simulation Model

The network configuration used in our simulation study is shown in figure 1. This configuration

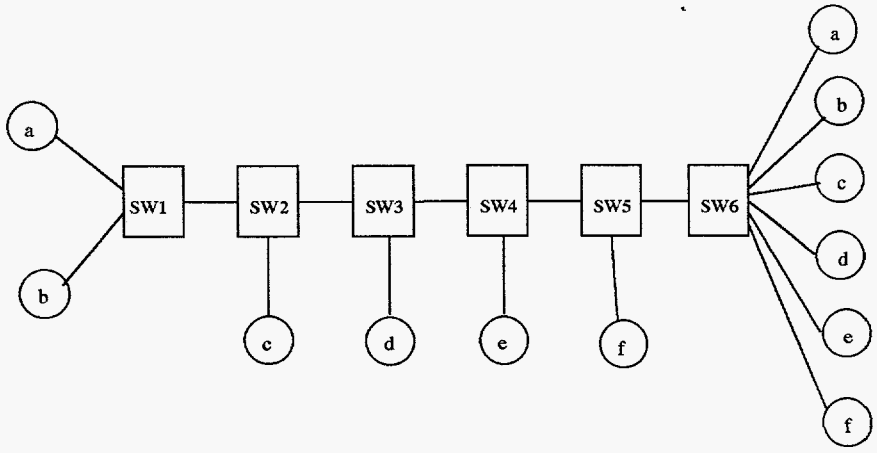

Figure 1: *Network Configuration*

consists of six ATM switches connected in series and six TCP sessions making two to six hops from source to destination. Two TCP connections originate from switch 1, one from each of switches 2 to 5, and all connections terminate at Switch 6, creating a bottleneck at the link connecting switches 5 and 6. All links are full duplex with link capacity of 151 Mbps, roughly corresponding to the effective capacity of STS-3c links (2.8 μs cell time slots are used). This configuration is interesting because we can study the behavior of TCP connections spanning different number of hops and distances. The ideal performance achievable is approximately $151/6 \times 48/53 = 22.8$ Mbps for each TCP connection.

The simulation tool used in this study is based on MIT's netsim [16]. We made extensive modifications to netsim to add ATM functionalities such as segmentation-and-reassembly (SAR) processing and ATM level flow control. The TCP module in netsim implements slow start, exponential back-off, and enhanced round-trip-time (RTT) estimation, but not fast-retransmit nor fast-recovery. A note on TCP's timer granularity is in order. Netsim uses a fine-grain clock for both RTT estimation and TCP time-out calculation. However, 500 *ms* clocks are common in the majority of today's kernel TCP implementations. While these coarse-grain timers are sufficient for traditional packet networks, they become grossly adequate in gigabit networks. At the same time, arbitrarily-fine-grained clocks (e.g., microseconds) would not be a good idea without some rethinking of the algorithms used to set the retransmit timers (note that the current TCP algorithms for measuring the RTT and for computing the retransmit timeout are coupled). While too fine a granularity can introduce unnecessary retransmission, a timer that is too coarse can result in under-utilized links. The optimal TCP timer granularity for high-speed ATM networks (both LAN and WAN) is still an open issue, and on-going discussion abounds in the research community. Our goal in this paper is to study the dynamics of TCP flow control algorithm in ATM networks using different services. With a fine-grain timer, our result is as independent as possible of any particular TCP implementation. Note that we have also performed numerous simulations using coarse-grain TCP timers, and although the

performance is poorer in all cases, the relative performance presented in this paper, and thus the conclusion, still holds [3, 4, 12].

The following parameters are used for all our simulation runs in this contribution.

Configuration parameters:

LAN: ES link = 0.5 km; BB link = 2.0 km; prog. delay = 5 μs/km
WAN: ES link = 0.5 km; BB link = 400.0 km

SES Parameters:

Nrm = 32; Trm = 100 ms; Mrm =2; TOFF = 1; CDF = 1/2;
Crm = 32; TDFF = 1/2; PNI =0;
PCR = 155 Mbps; MCR = 0.5 Mbps;

Switch Parameters:

Non-blocking output-buffered switch;
Simple FIFO w/adujstable Congestion Threshold;
Switching delay 4 μs;

TCP parameters:

TCP window = 64, 128 & 256 KB; MSS = 8192 byte;
Packet process delay 300 μs;

4 Simulation Results

4.1 ABR parameter tuning

The determining factors in binary mode ABR are the parameters RIF and RDF. RIF determines the amount of rate increase as a fraction of PCR when the network is not congested. RDF determines the fraction by which the current sending rate must be reduced when network feedback indicates congestion. Increases are additive and decreases are multiplicative. In ATM Forum TM 4.0 [14], both RIF and RDF are determined at connection setup time. A source sends its preferred RIF & RDF values in the signalling message. Each switch along the path will either accept the requested values or negotiate them downward. The returning signalling message contains the final RIF & RDF values supported by all the switches on its path, and these values are used by the source.

The values of RIF & RDF are powers of two and range from 1/32768 to 1.0. The smaller the values the more conservative is the rate increase/decrease. In this section we will examine TCP performance with different RIF & RDF values. The simulation runs in this section assume a switch queue size of 1024 and congestion threshold of 500 cells.

4.1.1 Aggressive Increase

In this section we set RIF to an aggressive value of 1/8, i.e., every receipt of an RM-cell with CI=0 would result in a rate increase of 1/8 of PCR. We vary RDF from an aggressive value of 1/2 to a moderate 1/16, to a conservative 1/64.

Figure 2-(a)&(b) show TCP performance when both increase and decrease factors are aggressively set, RIF=1/8 and RDF=1/2. As shown in Fig. 2-(a), the steady-state TCP throughputs show wide variation from 10 to 35 Mbps. The throughput achieved by each TCP is approximately inversely proportional to the number of hops it traverses. This phenomenon can be explained by the following observations. On start-up, each TCP sends out one packet. The acknowledgement of this packet determines the initial RTT estimate for each TCP. As TCP opens up its congestion window, switch buffer begins to build up at the bottleneck port and eventually congestion sets in. In response, the ABR source reduces its sending rate, in this case, aggressively. Although Fig. 2-(b) indicates that no cell is lost, a number of TCP connections begin to retransmit packets. This occurs because of the fine-grain timers assumed, i.e., TCP retransmits at the exact timer expiration time instead of multiples of some coarse granularity (e.g., 200-500 ms). Since the shorter the connection the smaller the initial RTT estimate is, the shorter connections have a greater chance of experiencing retransmission.

However, the artifact of fine-grain timer gradually disappears as the network reaches steady-state. The RTT estimate update algorithm in TCP eventually converged to an accurate steady-state value, which includes propagation delays and average queueing delay in both switch and the source's rate-control queue. At equilibrium, no retransmission was observed for any TCP connections.

In Fig. 2-(c)&(d), we reduced RDF to a more moderate value of 1/16. In this case the buffers consistently show both overflow and underflow (empty buffer). With a less aggressive RDF, the source rate did not decrease fast enough to prevent buffer overflow and cell loss. When cell lost occurs, TCP enters into slow-start, causing the links to be under-utilized. Observe that the fairness performance is also very poor in this case. In Fig. 2-(e)&(f), we repeated the simulation with a conservative RDF of 1/64. As the figures show, conservative RDF values work poorly with the aggressive RIF, causing constant fluctuations between buffer overflow and underflow.

The results from this section suggest that aggressive RIF must be matched by aggressive RDF to minimize cell loss probability. However, poor fairness performance seems to be an intrisic problem when aggressive parameters are used.

4.1.2 Moderate Increase

In this section we examine TCP performance with a moderate RIF of 1/64, with a range of RDF, ranging from aggressive to consevative. Fig. 3-(a)&(b) show the performance results for RIF=1/64 and RDF=1/2. With a large RDF, the maximum queue build-up was only about 600 cells. However, for the same reason, the queue became empty frequently, reducing link utilization and overall network throughputs.

Fig. 3-(c)&(d) show the result for a less aggressive RDF of 1/8. As expected, the maximum queue build-up was increased (900 cells), and fairness performance improved significantly. However, the link utilization is still less than optimal, since the bottleneck queue still became empty frequently. As RDF was reduced to a conservative 1/64 (Fig. 3-(e)&(f)), the source rate was not throttled fast enough to prevent queue build-up beyond buffer capacity. The resulting cell loss and TCP retransmission lead to poor performance as the bottleneck queue fluctuates between overflow and underflow.

4.1.3 Conservative Increase

In this section we examine TCP performance with a conservative RIF of 1/256 with different RDF settings. Fig. 4-(a)&(b) show the resluts for an aggressive RDF of 1/4. Similar to previous results on aggressive RDF, the maximum queue build-up was limited to approximately 600 cells, but large decreases resulted in under-utilized bottleneck links. Fig. 4-(c)&(d) plots the result for a moderate RDF of 1/16. Note that with this setting the queue fluctuation was much smaller than previous cases and the bottleneck queue never became empty after the initial transient period. Fig. 4-(c) indicates that the six TCP connections essentially achieved the optimal throughput performance. This combination of RIF and RDF produced the best TCP performance so far. The last combination to be examined is a conservative RIF with a conservative RDF. The plots in Fig. 4-(e)&(f) show the results for an RIF of 1/256 and an RDF of 1/64. As can be seen in Fig. 4-(f), an RDF of 1/64 is too small, even for a conservative RIF of 1/256, to avoid cell loss and TCP performance degradation.

4.1.4 Discussion on Parameter Tuning

The above results confirm our intuition about the settings of RIF and RDF–conservative increase and aggressive decrease produce stable system behavior. However, a stable system is not necessarily the system with the best performance. We found that under the conditions of our simulation, a conservative increase factor coupled with a moderate decrease factor produces by far the best TCP performance. Care should be taken in interpreting this result, however. In general, the optimal parameter values for RIF & RDF also depend on factors external to the ABR flow control algorithm, such as network configurations, RTT, buffer sizes and congestion thresholds etc. Our simulations assume one particular configuration with homogeneous ABR sources (identical RIF & RDF's); a different network configuration and/or heterogeneous ABR sources (different RIF & RDF's) may produce a slightly different pair of optimal values; however, the impact on TCP by tuning each parameter should remain similar. What we hope to demonstrate in our study are the effects each parameter has on TCP performance and also the sensitivity of binary ABR scheme with respect to the two parameters.

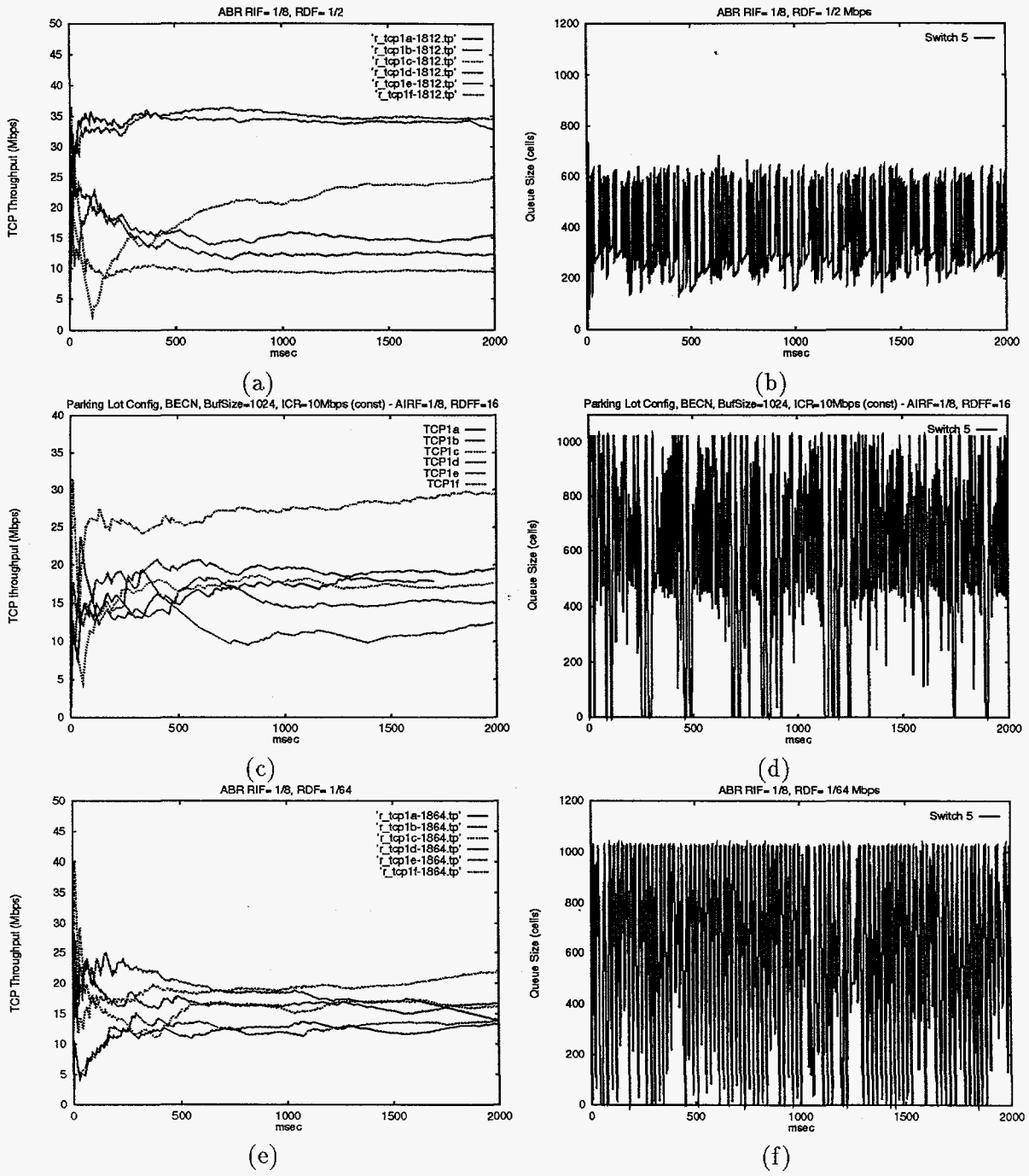

Figure 2: Aggressive Increase

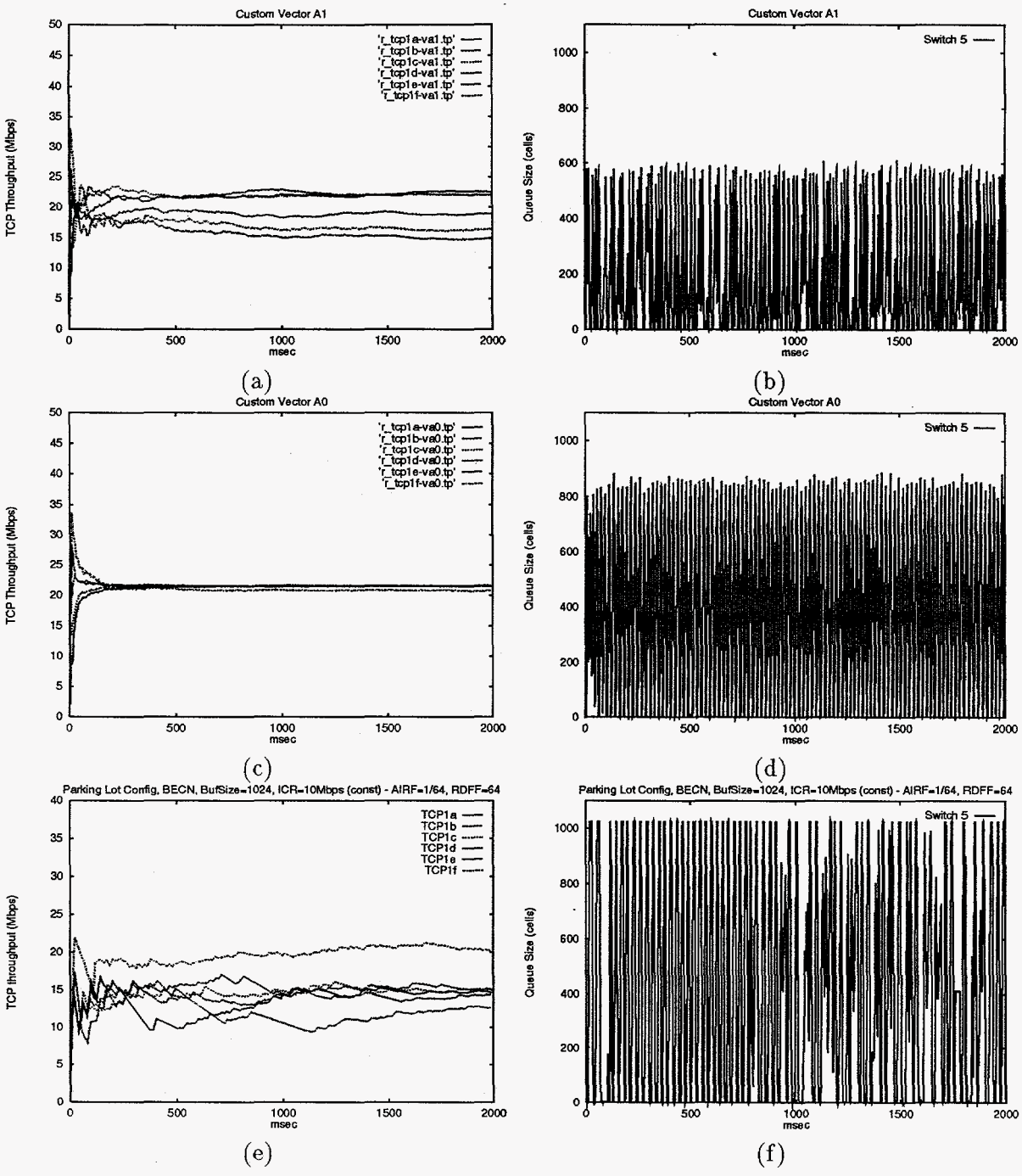

Figure 3: Moderate Increase

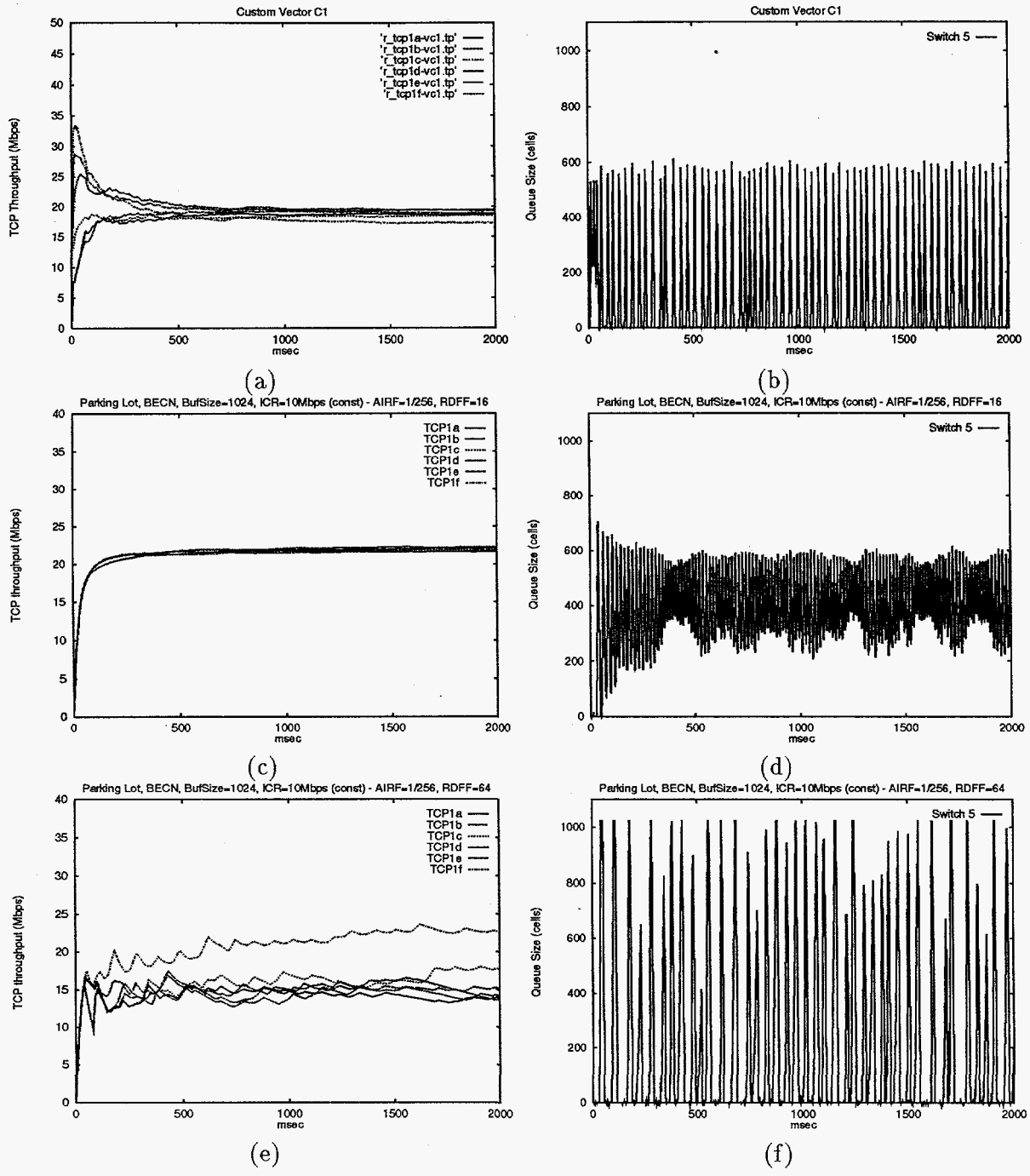

Figure 4: Conservative Increase

4.2 ABR and UBR Comparison

In this section we compare TCP performance over binary mode ABR with that of two enhanced UBR services: UBR-EPD and UBR-EPD-FBA. We compare performances assuming different buffer sizes and LAN and WAN distances. The RIF and RDF values used in the ABR simulations are taken from the best performing set shown in 4.1.3, i.e., RIF = 1/256 and RDF = 1/16.

Fig. 5 plots the throughputs and queue states of the bottleneck switch, for a LAN environment with 2K cell buffers. The congestion threshold for UBR services is 1500, and that for ABR is 1024 cells. As can be seen from Figs. 5-(a) & 5-(c), the fair buffer allocation scheme produced improved fairness performance for the six TCP connections. From the queue states plots in Figs. 5-(b) & 5-(d), we can see that the FBA scheme makes better use of the "headroom", i.e., the amount of buffer available to capture cells during congestion; EPD drops any incoming packet which sees a congested queue, while EPD-FBA only drops those that have used more than their fair share of the buffers during congestion. Note that the bottleneck link is fully utilized in both cases. The ABR throughputs show excellent fairness performance, although the bottleneck link is not fully utilized, as indicated by the bottleneck queue periodically emptying out. Note that the aggregate throughput for all three cases are approximately the same.

The above simulations were repeated with twice the amount of buffers, i.e., 4K cell buffers, with thresholds of 3072 for UBR and 2048 cells for ABR. The results are ploted in Fig. 6. Basically, the results are similar to those obtained with 2K buffers in Fig. 5. Comparing EPD and EPD-FBA, the fairness improvement produced by FBA is more pronounced than that in Fig. 5: link utilization is also improved by FBA, as the queue state never reaches zero during the entire simulation run. The ABR performance plotted in Fig. 6-(e)&(f) compared somewhat unfavorably with both UBR services. The ABR performed only slightly better in fairness, but the ABR aggregate throughput and the bottleneck link utilization are both less than its UBR counterparts.

By increasing the inter-switch link distance from 2 to 400 km, we examined TCP performance in a WAN environment. Because of the longer RTT, the TCP window was increased from 64 KB to 128 and 256 KB so as to be able to completely fill up the pipe. Fig. 7 plots the results for a WAN environment with 4K cell buffers. Fig. 7-(a) shows that simple EPD produced extremely poor fairness performance, with the shortest TCP connection (1f) grabbing an unfairly large share of the bottleneck bandwidth. In the WAN case, the fairness improvement by FBA is quite significant, as the range of throughputs is narrowed from 10 – 43 Mbps down to 16 – 29 Mbps. Link utilization is also improved slightly by FBA, judging from the fact that the FBA queue states empties out less often than the corresponding EPD plot. The ABR performance is slightly better in terms of fairness, but falls short in both aggregate throughput and link utilization.

The above WAN simulations were repeated with 8K buffers, with thresholds of 7168 for UBR and 4096 cells for ABR. Results are plotted in Fig. 8. Comparing Fig. 8-(a) with Fig. 7-(a), larger buffers has the effect of improved fairness performance for the EPD scheme; however, the link utilization is still less than 100%. FBA also significantly improved fairness performance in this case, in addition to also improving the link utilization. The corresponding ABR perfor-

mance paralleled that of Fig. 7, and compared somewhat unfavorably with both UBR services.

It should be noted that the ABR parameters used in these comparisons were chosen from Section 4.1 which used a buffer size of 1024 cells and a threshold of 500 cells. Further fine-tuning of RIF and RDF using different buffer sizes, thresholds and WAN distances may yield better results. However, such fine-tuning may be impractical in real networks without employing some kind of global optimization schemes. Unfortunately, such schemes are conspicuously lacking. Therefore, the ABR performance results used for these comparisons may not be optimal, but they should not be too far from typical ABR performance.

5 Summary and Discussion

In this paper we have presented simulation results on parameter tuning in binary mode ABR service in ATM networks. We examined the effects of binary mode ABR parameters on TCP performance. We have shown that TCP performance is highly sensitive to the values of two parameters, RIF and RDF. We observed that a conservative RIF coupled with a moderate RDF resulted in the best TCP performance under our test configuration. In the second part of the paper, we compared the performance of binary mode ABR and two enhanced UBR schemes in supporting TCP traffic. We found that UBR-EPD yielded poor fairness performance without fairness enhancement. FBA provided such a fairness enhancement with the addition of a per-VC accounting algorithm. In the absence of global optimization for ABR parameters, UBR-based services can provide comparable, if not indeed superior, performance to binary mode ABR.

The results presented in this paper raised a number of interesting questions for future work. One natural question to ask is whether the results will still hold when the bandwidth available to ABR service is a time-varying function, instead of a constant as assumed in this paper. When the available bandwidth changes with time, as is expected to be the case when there is CBR and VBR traffic, how well will ABR be able to fill up the left-over bandwidth? Is ABR performance as sensitive to RIF and RDF as the case when there is no CBR and VBR traffic? How will ABR and UBR comparisons be changed, if at all? Moreover, the bandwidth available to ABR can exhibit different degree of variability depending on the underlying VBR traffic characteristics. The performance of ABR with respect to different degree of variability in available bandwidth is another issue of interest. Is ABR more effective when the available bandwidth variability is low? Or, is there a threshold in variability beyond which ABR will cease to be effective?

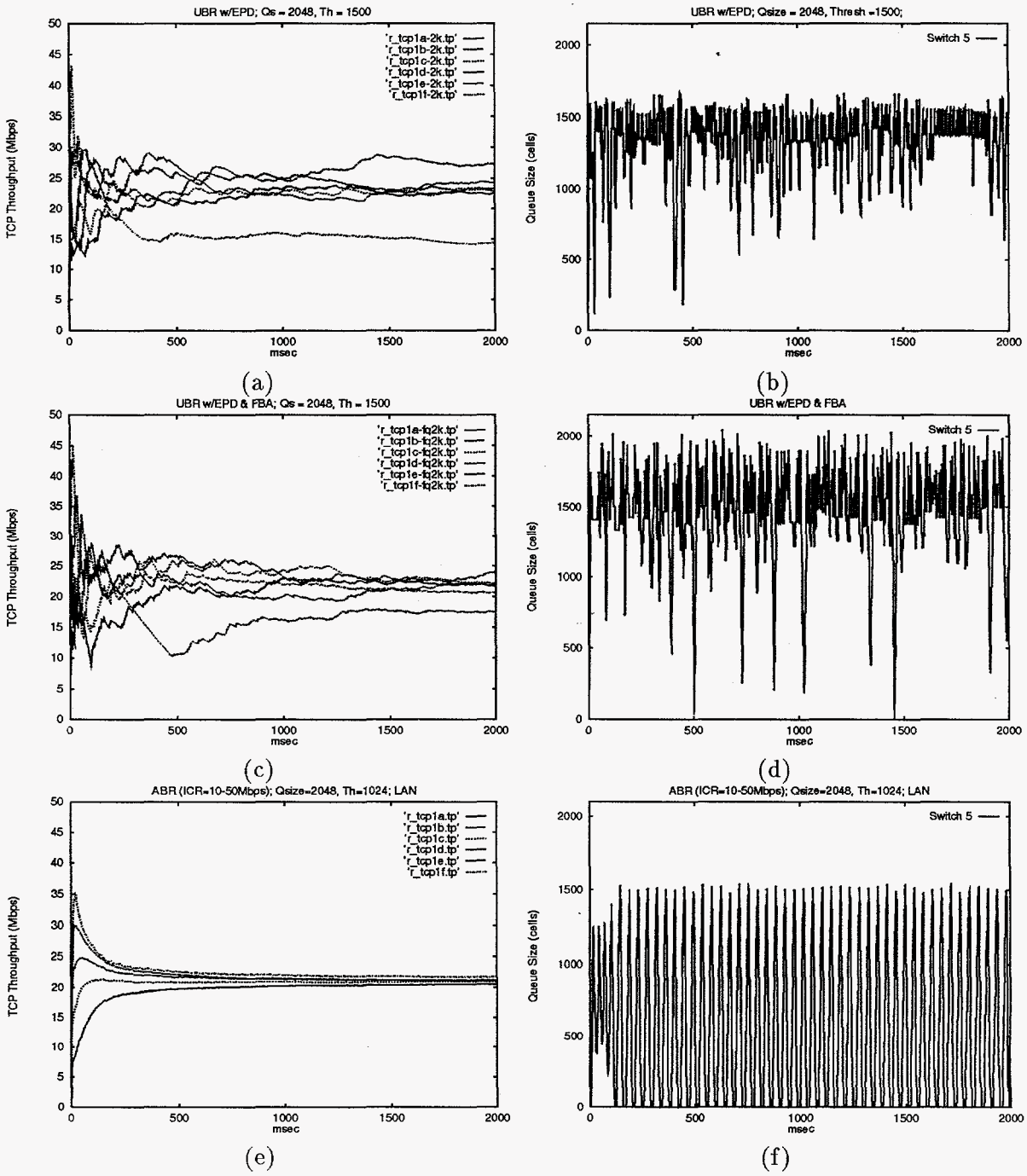

Figure 5: **UBR-EPD, UBR-EPD-FBA & ABR; 2K, LAN**

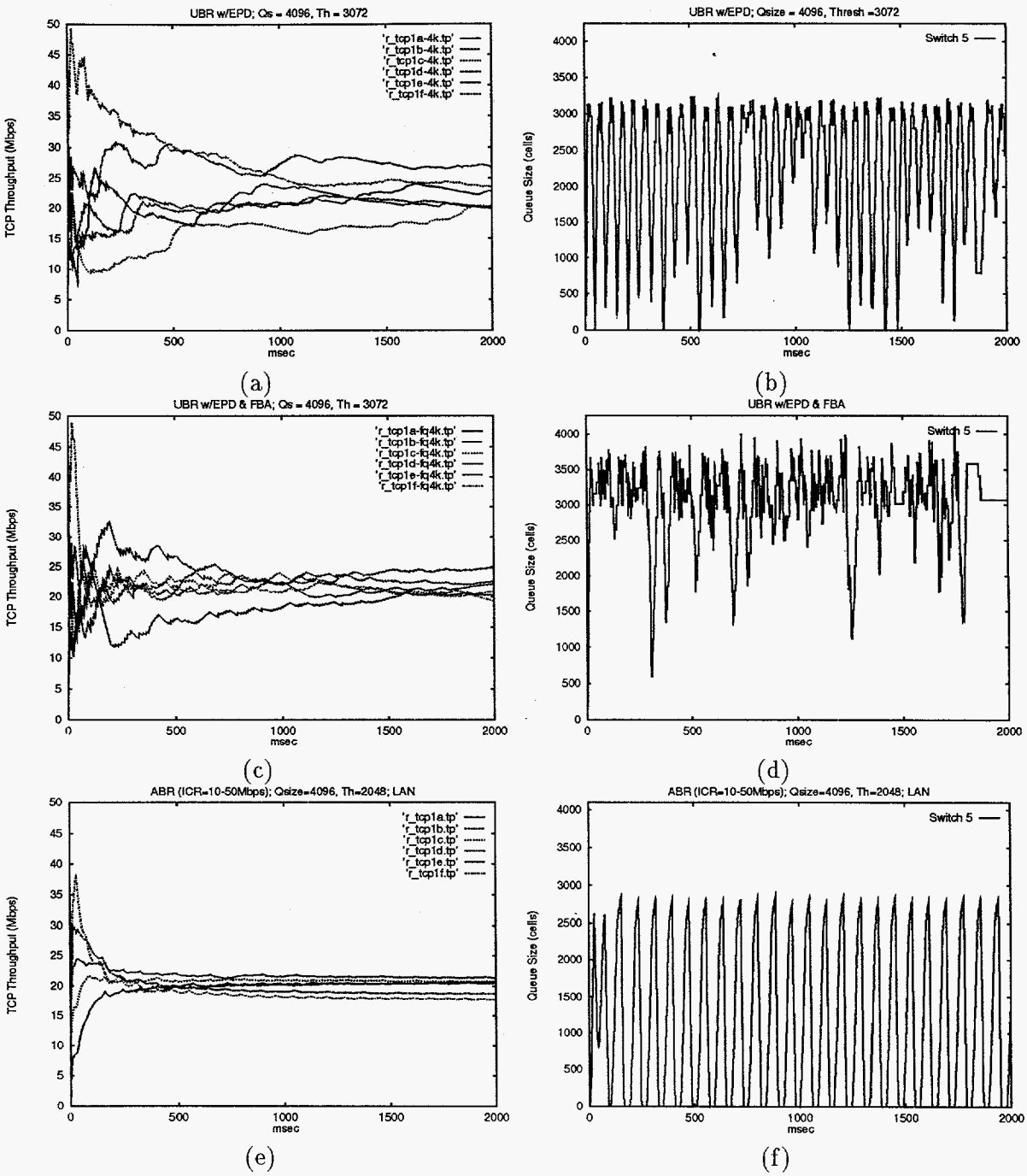

Figure 6: **UBR-EPD, UBR-EPD-FBA & ABR; 4K, LAN**

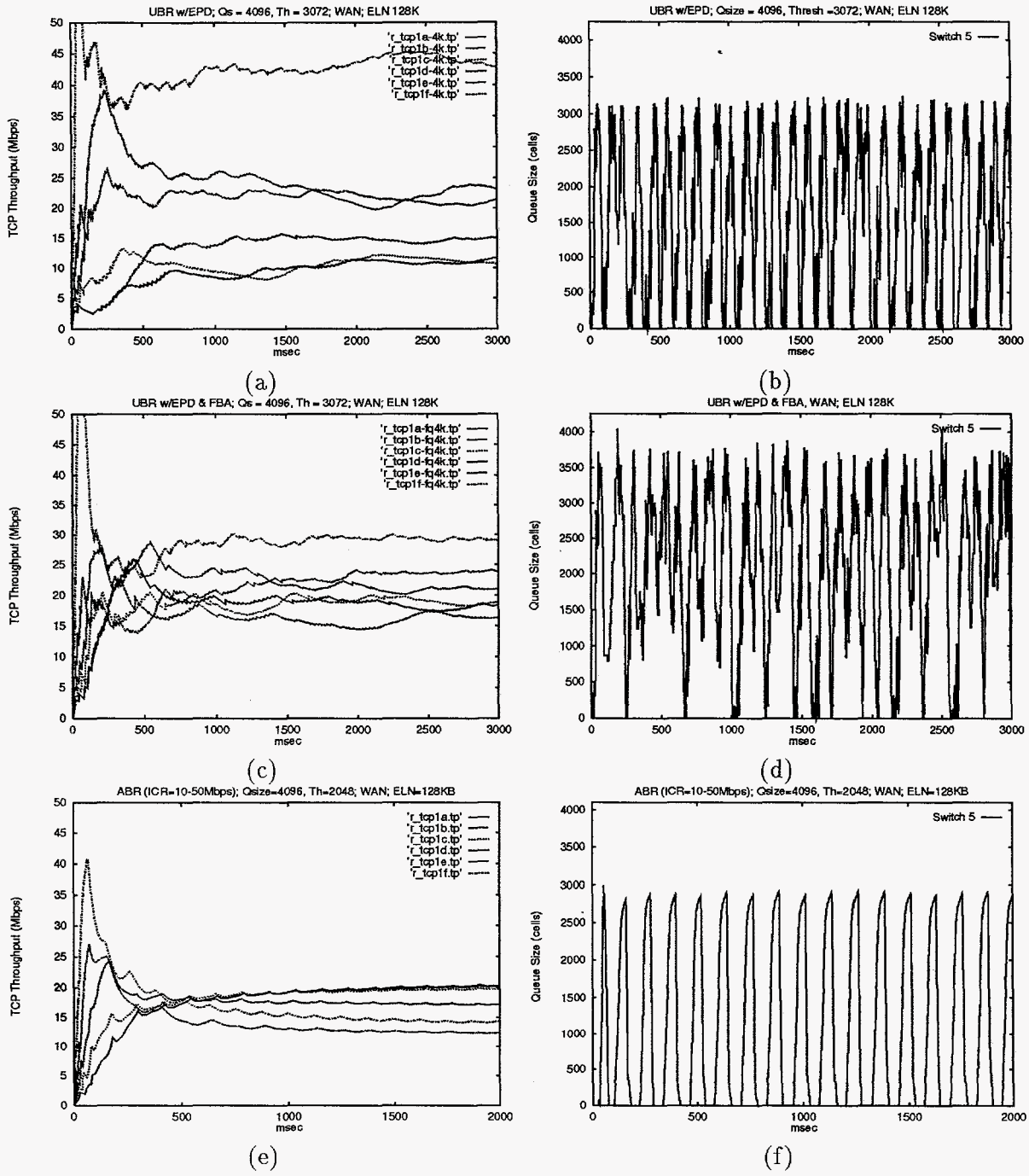

Figure 7: **UBR-EPD, UBR-EPD-FBA & ABR; 4K WAN**

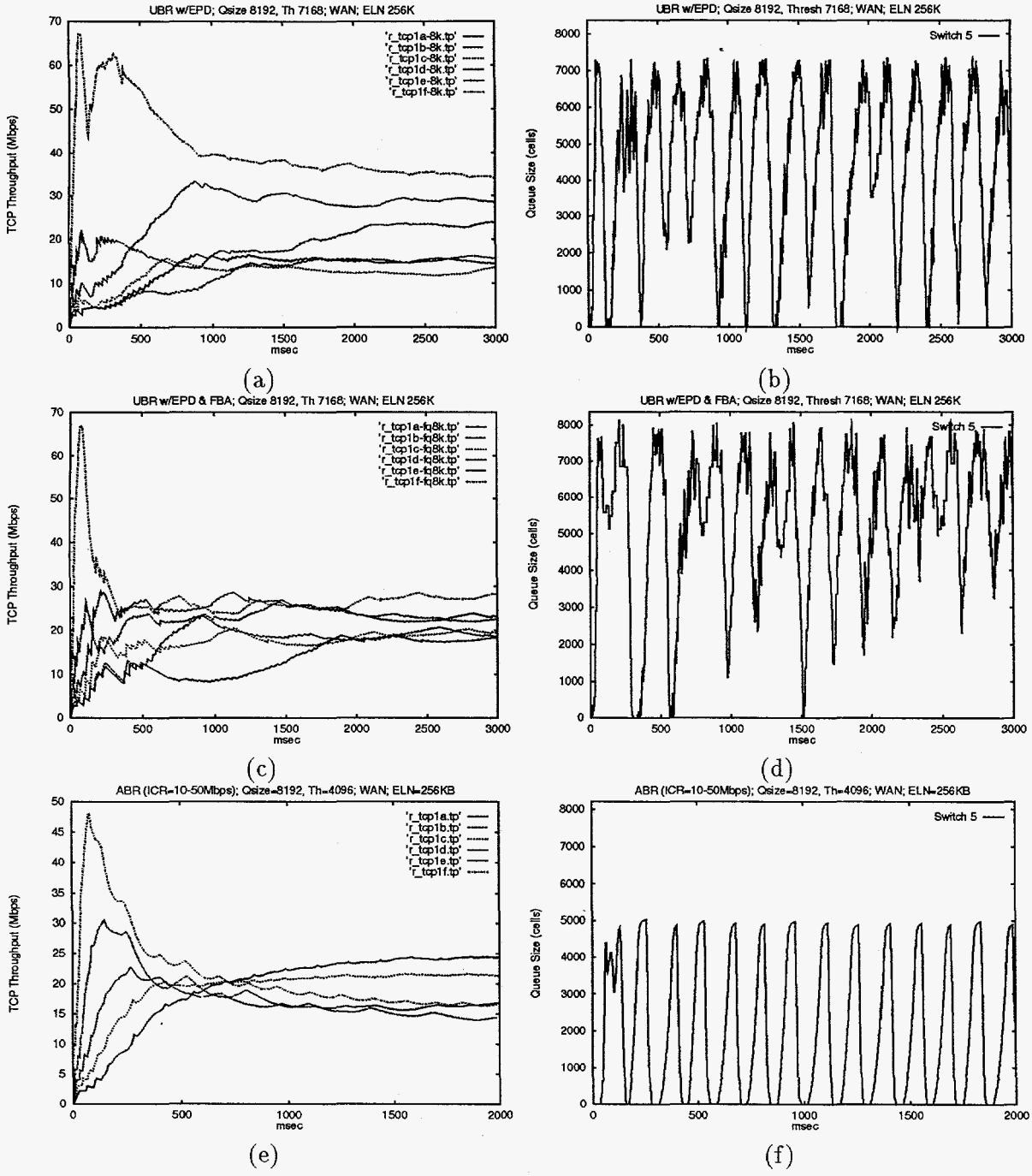

Figure 8: **UBR-EPD, UBR-EPD-FBA & ABR; 8K WAN**

References

[1] Y. Chang, N. Golmie, and D. Su, "A Rate Based Flow Control Switch Design for ABR Service in an ATM Network," in *Proceedings ICCC'95*, August 1995.

[2] C. Fang, H. Chen and J. Hutchins, "A Simulation Study of TCP Performance in ATM Networks," *Proceedings of IEEE GLOBECOM '94*, pp. 1217-1223, November, 1994.

[3] C. Fang and A. Lin, "A Simulation Study of ABR Robustness under Binary-Mode Switches: Part II," ATM Forum, AF-TM 95-1328R1, October 1995.

[4] C. Fang and A. Lin, "On TCP Performance of UBR with EPD and UBR-EPD with a Fair Buffer Allocation Scheme," ATM Forum, AF-TM 95-1645, December 1995.

[5] S. Floyd and A. Romanow, "Dynamics of TCP traffic over ATM Networks," in *Proceedings of ACM SIGCOMM'94*, pp. 79-88, Sept. 1994.

[6] J. Heinanen and K. Kilkki, "A Fair Buffer Allocation Scheme," submitted for publication.

[7] R. Jain, S. Kalyanaraman, R. Viswanathan and R. Goyal, "A Sample Switch Algorithm," ATM Forum, AF-TM 95-0178, February 1995.

[8] L. Kalampoukas and A. Varma, "Performance of TCP over Multi-Hop ATM Networks: A Comparative Study of ATM Layer Congestion Control Schemes," in *Proceedings of ICC'95*, June, 1995.

[9] L. Kalampoukas, A. Varma, and K.K. Ramakrishnan, "An efficient rate allocation algorithm for ATM networks providing max-min fairness," in *Proceedings of 6th IfIP International Conference on High Performance Networking, HPN'95*, September, 1995.

[10] H.T. Kung, R. Morris, T. Charuhas, and D. Lin, "Use of Link-by-link Flow Control in Maximizing ATM Network Performance: Simulation Results," in *Proceedings IEEE Hot Interconnects Symposium '93*, August, 1993.

[11] H. Li, K.-Y. Siu, H.-Y. Tzeng, C. Ikeda, and H. Suzuki, "TCP Performance over ABR and UBR Services in ATM," to appear in IPCCC'96, March 1996.

[12] A. Lin and C. Fang, "A Simulation Study of ABR Robustness under Binary Switch Modes," ATM Forum, AF-TM 95-1019, August 1995.

[13] B. Lyles and A. Lin, "Definition and Preliminary Simulation Results for a Rate-based Congestion Control Mechanism with Explicit Feedback of Bottleneck Rates," AF-TM 94-0708, May 1994.

[14] S.S. Sathaye, *ATM Forum Traffic Management Specification, Version 4.0,* ATM Forum, TM Subworking Group, January 1996. ATM Forum/95-0013R10.

[15] K.-Y. Siu and H.-Y. Tzeng, "Intelligent Congestion Control for ABR Service in ATM Networks," *Computer Communication Review*, Vol. 24, no. 5, pp. 81-106, Oct. 1994

[16] A. Heybey, "The Network Simulator," available from LCS, MIT, May 21, 1993 (available for anonymous ftp from allspice.lsc.mit.edu in /pub/netsim).