Proceedings of the 2007 Winter Simulation Conference
S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds.

MODELING THE PERFORMANCE OF LOW LATENCY QUEUEING FOR EMERGENCY TELECOMMUNICATIONS

Denise M. Bevilacqua Masi, PhD
Martin J. Fischer, PhD
David A. Garbin

Noblis
3150 Fairview Park Drive South
Falls Church, VA 22042-4519, U.S.A.

ABSTRACT

Event simulation and analytic modeling are used to evaluate the performance of Low Latency Queueing (LLQ), a queueing discipline available in some Internet packet switching routers for integrated services performance assurance. LLQ combines priority queueing with Class-Based Weighted Fair Queueing (CBWFQ). Priority queueing is used to ensure satisfying tight delay constraints for real-time traffic, whereas CBWFQ is used to ensure acceptable throughput for traffic classes that are less sensitive to delay. Simulations are developed both using a commercial product, OPNET Modeler, and also custom simulators that we developed. Our custom simulators model two different approaches to CBWFQ; and comparisons between the approaches and that of the commercial simulator are conducted. Our computational experiences (central processing unit [CPU] times for model execution and post-processing) in using the simulators are described. This work is an important first step in the ability to model a proposed enhancement to LLQ which may be beneficial to Emergency Telecommunications Services.

1 INTRODUCTION

The ability to communicate during emergencies is essential for government personnel. The mission of the National Communications System (NCS) includes planning for and provisioning National Security/Emergency Preparedness (NS/EP) communications for the federal government under all circumstances, including crisis or emergency, attack, recovery, and reconstitution. Modeling telecommunications for current and future programs providing emergency service is essential. Industry is moving from circuit switched to Internet Protocol (IP) technology for all telecommunications applications, including voice, and NCS is investigating the need to evolve toward IP capability to ensure continuity of priority traffic during emergencies. We are conducting various modeling and simulation activities to support this effort.

There are several different mechanisms available that attempt to provide quality of service (QoS) in the Internet (Semeria 2001 and Cisco IOS Quality of Service Solutions Configuration Guide). Packet traffic on the Internet could simply be handled on a First Come, First Served (FCFS) basis. With a high enough bandwidth and under normal traffic conditions, this could be sufficient (Davie 2002; Fischer and Masi 2007). However, during emergency situations, historically disasters have been documented to produce demand of up to 10 times normal traffic in the public switched telephone network. In priority queueing (PQ), higher priority real-time traffic (e.g., Voice over IP or VoIP traffic) is transmitted before lower priority traffic (e.g., data traffic), with separate buffers for each class of traffic. Weighted Fair Queueing (WFQ) allocates the bandwidth fairly to network data traffic using weights to determine the amount of bandwidth allowed for different flows of traffic. Class-Based Weighted Fair Queueing (CBWFQ) extends weighted fair queueing to multiple user-defined traffic classes, rather than individual flows of traffic. Under CBWFQ, there are buffers for each class of traffic, but a certain portion of the bandwidth is set aside for each of the classes; when one class of traffic is not utilizing its bandwidth, the other class is allowed to overflow and use the bandwidth. In addition, Low Latency Queueing (LLQ), which combines priority queueing and class-based weighted fair queueing, is being used frequently on the Internet with these multiple classes. It is apparent there are many options available; modeling is essential to determine which QoS mechanism is most appropriate in advance, rather than using a trial and error approach on a real network.

Under the assumption of Poisson packet arrivals, analytic queueing results are available for FCFS and PQ, but not for CBWFQ or LLQ (see Fischer and Masi (2005b) for a summary of available results). In practice, there would likely be greater than one class of data traffic in CBWFQ (e.g., one or more Assured Forwarding [AF] classes, and a Best Effort [BE] class). LLQ combines CBWFQ with priority queueing, and typically voice traffic is assigned to the higher priority queue (the Expedited Forwarding [EF] traffic class) than the data classes which use CBWFQ. In the future there may be greater than one class of EF traffic. Requests to the Internet Assigned Numbers Authority (IANA) for additional EF classes, which could be used for disaster response VoIP or video traffic, have been submitted to the Internet Engineering Task Force (IETF) (Baker 2006; Baker 2007). This research focuses on the modeling of LLQ; it is one discipline being studied to determine if it can ensure QoS requirements will be met. Figure 1 depicts the relationship between the priority queue(s) and the class-based weighted fair queue in a router configured with the LLQ discipline.

The remainder of this paper will discuss the LLQ and CBWFQ scheduling algorithms (Section 2), describe the simulation tools used in this analysis (Section 3), review the assumptions (Section 4), provide numerical examples from our simulation models (Section 5) both under normal and emergency traffic conditions, and summarize our conclusions and next steps in this research (Section 6).

2 LOW LATENCY QUEUEING AND CASE-BASED WEIGHTED FAIR QUEUEING SCHEDULING ALGORITHMS

Modeling the priority queueing portion of LLQ, that is, the modeling of the EF class having priority over the CBWFQ or data classes, is well understood. Analytic queueing models exist for priority queues (see the standard non-preemptive priority queueing model in Gross and Harris, 1998; Cohen, 1969). These models can also handle the case of more than two priority classes, which arise if additional EF classes for disaster response VoIP or video traffic are used. Cohen (1969) gives the first and second moments

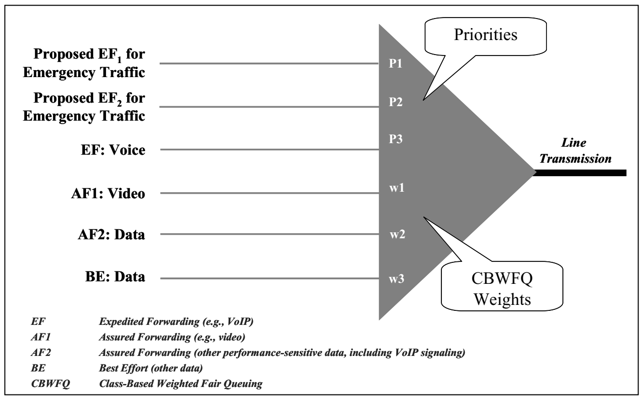
Proposed EF_1 for Emergency Traffic
Proposed EF_2 for Emergency Traffic
EF: Voice
AF1: Video
AF2: Data
BE: Data
P1
P2
P3
w1
w2
w3
Priorities
CBWFQ Weights
Line Transmission
EF Expedited Forwarding (e.g., VoIP)
AF1 Assured Forwarding (e.g., video)
AF2 Assured Forwarding (other performance-sensitive data, including VoIP signaling)
BE Best Effort (other data)
CBWFQ Class-Based Weighted Fair Queuing

Figure 1: Low Latency Queueing Router Depiction

for the queue waiting time; these analytic results for the higher priority class apply directly to the EF (voice) class in the router configuration that we are studying. Because we understand the interaction between the higher priority EF queue and the class-based weighted fair queues, and how to model that interaction, our focus in this modeling is within the CBWFQ portion of the router, which is less well understood.

Under CBWFQ, weights are specified for each class of traffic. Each class of traffic is guaranteed to obtain the portion of the bandwidth in accordance with the weight, if needed. When one class of traffic is not utilizing its bandwidth, other classes are allowed to overflow and use the bandwidth.

The precise rules used by class-based weighted fair queues to select which class of packet to transmit when the router is available are difficult to determine. We were able to obtain some information on the CBWFQ scheduling algorithms used in practice. We initiated a dialogue with a major router vendor to determine how these router schedulers select packets under CBWFQ. Router vendors do not usually publicize this detailed and sometimes proprietary information. The vendor informed us that these CBWFQ scheduling rules vary by class of router, but that most platforms implement CBWFQ using a calendar queue or timing wheel. To associate each packet with the correct slot or sequence number in the calendar queue, a "next time" value for each packet is computed. As the details on the computation of the next time value were not provided, this information was not detailed enough for us to build a CBWFQ model. We also spoke with *OPNET Technologies*, the creator of the OPNET Modeler, a popular discrete-event simulation package which is designed specifically to model telecommunications networks. OPNET Modeler includes models for CBWFQ. Personnel at *OPNET Technologies* told us that their implementation of CBWFQ is based on an academic implementation of CBWFQ. In this implementation, each packet is assigned a "virtual finish time," which is computed as the packet transmission time divided by the class weight, plus an arrival time. The packet having the smallest virtual finish time is selected for transmission when the router is free. The details of what the "arrival time" is were not provided.

We also researched CBWFQ algorithms discussed in the literature. Golestani (1994) describes a CBWFQ algorithm in which a virtual finish time is computed for each packet. This virtual finish time is the sum of the packet transmission time divided by the class weight, plus the maximum of two times which are the virtual finish time of the previous packet of the same class, and a class-independent time which is the tag of the packet in service at the arrival time of the current packet of interest. Specifically, the virtual finish tag $\hat{F}_k^i$, or service tag, of each packet is computed as follows (Golestani 1994):

$$\hat{F}_k^i = \frac{1}{r_k} L_k^i + \max(\hat{F}_k^{i-1}, \hat{v}(a_k^i))$$

Where i is the packet index, k indicates the class membership of packet i, r_k is the weight of class k, L_k^i is the transmission time of packet i, $\hat{v}(a_k^i)$ is a class-independent virtual finish tag of the packet in service at the arrival time of packet i, and $\hat{F}_k^0 = 0$.

In addition, we hypothesized that a CBWFQ scheduler that selects the packet class to transmit randomly, based on the class weights, would preserve the fairness of the algorithm as far as allocating to each class the desired share of the bandwidth, if required by the packet traffic. For the class selected by the random approach, the packet at the head of the queue would be transmitted. This approach to CBWFQ scheduling would be simple to implement in a simulation model, and also the random nature of the packet selection may be amenable to development of analytic queueing models later on.

In summary, there are three main approaches to CBWFQ scheduling that we have selected for further investigation and comparison:

- Random selection of the class for transmission based on the weights
- Golestani's virtual finish time approach
- OPNET Modeler's implementation of CBWFQ

We noted also that the Golestani algorithm and the OPNET algorithm both compute a virtual finish time containing a term that is the packet transmission time divided by the class weight. In both cases, the packet with the smallest value of the virtual finish time is selected for transmission when the router is free, and, thus, smaller packets (which have smaller transmission times) are given a preference for selection if all else is equal. There are similarities between these CBWFQ scheduling rules and the shortest processing time (SPT) rule in the queueing literature (Gross and Harris 1998; Schrage and Miller 1966), in which priority is given to the group of customers which on average has the faster service rate. Such a rule reduces the average wait in queue.

3 SIMULATION TOOLS USED IN ANALYSIS

Several simulation tools were used in this analysis. It is desirable that each of the simulation tools have the ability to compute several measures of effectiveness: latency (mean queue wait), jitter (assumed to be the 99.9 percentile of the queue wait), and packet loss. The OPNET Modeler has capabilities to model many of the QoS mechanisms described above, and we used it for our initial modeling of LLQ. However, we decided to develop our own custom simulators of the CBWFQ portion of the LLQ for several reasons:

1. In order to determine if our interpretations of CBWFQ were consistent with OPNET Modeler's implementation
2. To compare the sensitivity of the performance of CBWFQ to possible differing interpretations of how it works
3. To have a simulator that could be run quickly for comparisons with analytic results as we develop analytic algorithms for CBWFQ, because OPNET Modeler runs take some time in execution and post-processing of results
4. To have a flexible simulator that can be modified in the future to examine different system configurations such as additional priority queues for traffic

With respect to item 4., the OPNET Modeler router model does not support implementing two priority queues (e.g., EF plus an EF_1 for emergency traffic) as well as CBWFQ. To overcome this obstacle for future models with the additional emergency traffic EF queues, in the simulation model two routers would have to be chained together. The first router would provide CBWFQ for the AF1, AF2, and BE traffic, and the second router would be configured with multiple priority queues. The output of the CBWFQ router would be connected as the input to the lowest priority queue of the first router. Thus, for multiple reasons, a custom simulator of CBWFQ is desirable.

As previously mentioned, three main approaches to CBWFQ scheduling were selected for further investigation and comparison. The custom simulation models were implemented multiple times by analysts working independently, in order to verify that the implementations were done correctly. The specific models are denoted as follows:

- *Random_weights_1:* Random selection of the class for transmission based on the weights, coded in Microsoft Access Visual Basic for Applications (MS VBA)
- *Random_weights_2:* Random selection of the class for transmission based on the weights, coded in MS VBA by a different analyst independent of Random_ weights_1
- *Golestani_1:* Golestani's virtual finish time approach, coded in MS VBA
- *Golestani_2:* Golestani's virtual finish time approach, coded in object-oriented RealBasic by a different analyst independent of Golestani_1
- OPNET Modeler's implementation of CBWFQ

For the Random_weights_1 and Random_weights_2 custom simulators that we developed, we are assuming that when the router is free, the class of the next packet for transmission is chosen randomly according to the normalized weights of the classes that are present. If the mean packet sizes differ between classes, the weights are first adjusted by dividing by the mean packet size and then normalized. The first packet in the queue of the selected class is transmitted.

Random_weights_1 generates a series of packet arrivals for each of three classes of packets. When the router is free, the class of the packet to be transmitted next is determined by randomly selecting according to the normalized class weights. It assumes finite buffers for each class, and tracks the number of packets in the system at each arrival point and departure point. At each arrival point, it determines changes in the system that have occurred since the last arrival due to packet service completions and departures from the system. The number of packets in the system at arrival points is used to determine if the arrivals are blocked due to full buffers or not. Queue wait times are tracked for each packet. At the end of the simulation, Random_weights_1 collects statistics on the mean queue wait (latency) by class, packet loss, and the 99.9 percent quantile of queue wait (jitter) by traffic class.

In order to verify we were simulating our random selection interpretation of CBWFQ correctly, a second simulator, Random_weights_2, was developed by an independent analyst. Random_weights_2 simulates the system behavior at successive departure points. The number of packet arrivals by traffic class occurring between successive departure points is estimated. The class of the packet transmitted at each departure point is determined by randomly selecting according to the class weights, as mentioned above. Random_weights_2 looks at the system at packet departure points, and keeps track of the type of packet departing and the number of packets of each type left behind. It assumes infinite buffers, so is not suitable as written for heavy traffic situations where packet buffer limitations would be reached in practice. A packet's transmission time is selected by first determining the packet size based on our assumed packet size distribution, and thus the packet transmission time. The total number of arrivals independent of class during that transmission time is estimated; the binomial distribution is then used to determine number of packets of each class arriving during the transmission time. In this way, the number in the system of each class at same-class departure points is tracked. Little's formula is then used to get the expected buffer delay for each class using the number by class at departure points. That is, for the class 1 delay, only use the number of class 1 customers in the system at class 1 departure points.

Random_weights_1 computes mean queue delay, jitter, and packet loss, and, thus, is preferred over Random_weights_2, which only computes mean queue delay. However, we found it useful to have both simulators of the random weights approach, which were developed independently by different developers with different approaches, in order to validate the queue delay results generated by the two simulators.

Golestani_1 implements the Golestani CBWFQ approach (1994) in MS VBA. Golestani_2 is an implementation of the Golestani algorithm in object-oriented RealBasic, developed by a second analyst.

Lastly, OPNET Modeler was also used. The model consisted of a router with CBWFQ, interfaced with three sources of packet traffic and a transmission line. OPNET Modeler's raw packet generator was used to generate packets according to our desired packet size distribution, which is described in Section 4.

4 ASSUMPTIONS AND CASES

The assumptions and scenarios for our modeling will be described. The assumptions below were used in all simulators described in Section 3, so any significant differences in the results of the three simulators would reflect differences in the CBWFQ scheduling algorithms that are assumed. We used three traffic classes, which will be referred to as Class 1, Class 2, and Class 3 (rather than AF1, AF2, and BE). The voice (EF) traffic is not modeled. Poisson arrivals for all three traffic classes were assumed (Cao et al. 2002). Packet sizes for data are assumed to have three packet sizes; previous research shows that three packet sizes are dominant in IP traffic (Masi and Fischer 2005). This packet size distribution is 40 bytes (50 percent), 750 bytes (10 percent), and 1500 bytes (40 percent), with a mean packet size of 695 bytes.

Scenarios assumed a total line speed of T1 (1,544 kbps) for all traffic including voice; the remaining bandwidth for the data traffic is 1,059 kbps. This line speed was chosen because it is viewed as a popular network access speed; our experience with a particular large federal agency indicates that 75 percent of its locations have access speeds of T1 or less. This speed of 1,059 kbps is used for all cases.

Table 1 summarizes the input parameters for all cases. These scenarios are designed to have realistic baseline and overload traffic levels, and also investigate the variation in the model results for different methods of implementing CBWFQ as the traffic is varied and depending on the sufficiency of the bandwidth allocation to the traffic classes.

Case 1—Baseline—has a total line utilization of 74 percent. Case 1, Class 2's allocation of the bandwidth is much more than required, while Class 1's allocation of the bandwidth is close to what is required and Class 3's allocation of the bandwidth is much less than required. Case 2—Baseline with Equal Weights—adjusts the weights used in Case 1 so that the allocated bandwidth for Class 1 is much less than required. Cases 3 and 4 deviate from the total utilization of .74 by plus or minus 20 percent. Cases 5 and 6 further increase the traffic; Case 5 has a total utilization

Table 1: Model Parameters

Case Number/ Scenario Number	Case Name	Class 1				Class 2				Class 3				Total Utilization, Rho
		Lambda1 (ppms)	Buffer Size 1 (k1)	Weight 1 (w1)	Rho1	Lambda2 (ppms)	Buffer Size 2 (k2)	Weight 2 (w2)	Rho2	Lambda3 (ppms)	Buffer Size 3 (k3)	Weight 3 (w3)	Rho3	
1	Baseline	0.08	20	0.4	0.42	0.015	512	0.45	0.08	0.045	512	0.15	0.24	0.74
2	Baseline with Equal Weights	0.08	20	0.33	0.42	0.015	512	0.33	0.08	0.045	512	0.33	0.24	0.74
3	Baseline Plus 20% Traffic	0.096	20	0.4	0.50	0.018	512	0.45	0.09	0.054	512	0.15	0.28	0.88
4	Baseline Minus 20% Traffic	0.064	20	0.4	0.34	0.012	512	0.45	0.06	0.036	512	0.15	0.19	0.59
5	Heavy Traffic	0.1056	20	0.4	0.55	0.0198	512	0.45	0.10	0.0594	512	0.15	0.31	0.97
6	10x Overload	0.8	20	0.4	4.20	0.015	512	0.45	0.08	0.45	512	0.15	2.36	6.64

of 97 percent. However, disaster response traffic can be up to 10 times the normal load. Case 6 represents a disaster situation, and increases the Case 1 traffic levels for Classes 1 and 3 by a factor of ten. Class 2 is assumed to be a controlled class for business applications and is not subject to the traffic increase in Case 6.

5 NUMERICAL RESULTS

Figure 2 compares the mean queue waits under Case 1 for the five simulators. Figure 3 shows the 99.9 percentile of the queue wait (often referred to as Jitter) for this scenario.

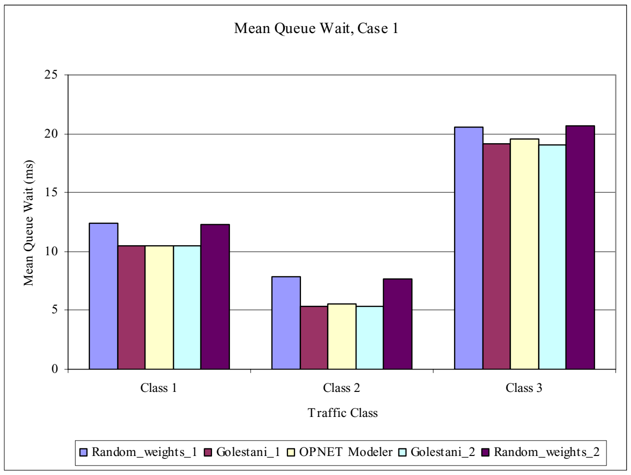

Figure 2: Mean Queue Wait, Case 1

As is seen in Figure 2, the Random_weights_1 and Random_weights_2 simulators are in agreement which verifies our implementation of this CBWFQ scheduling approach; hence, we will report only results from Random_weights_1 for subsequent examples. Also in Figures 2 and 3, the Golestani_1 and Golestani_2 simulators are in agreement which verifies our implementation of that CBWFQ scheduling approach, so we will report only results from Golestani_1 for subsequent cases. The more interesting comparisons are between the Random Weights, Golestani, and OPNET Modeler results. The Random Weights approach gives higher mean queue waits (between 6 percent and 31 percent higher, depending on the class) and a higher 99.9 percentile of the queue wait (between 3 percent and 38 percent higher) for all three traffic classes than the Golestani or OPNET Modeler approaches. We had suspected that the Golestani approach would yield lower queue waits based on its relationship to the shortest processing time rule. In addition, the variance of the queue waits were compared (Figure 4), and the Random Weights approach has a higher variance than Golestani or OPNET Modeler. For Case 1, packet loss was zero for all models. Also, the OPNET Modeler and Golestani results are close; differences are less than 3 percent for mean queue waits and less than 7 percent for the jitter. As we suspected, the OPNET Modeler is likely using the Golestani approach to model CBWFQ.

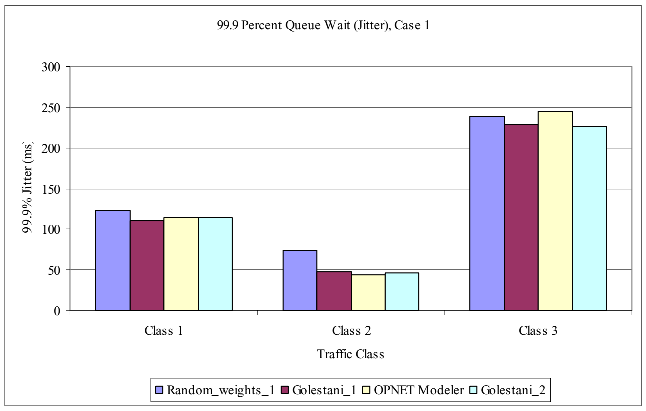

Figure 3: 99.9 Percent Queue Wait (Jitter), Case 1

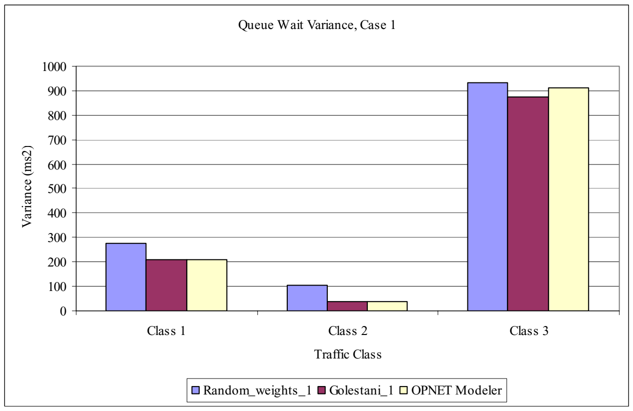

Figure 4: Queue Wait Variance, Case 1

When the CBWFQ weights are equal for all cases (Case 2), that is, the weights decrease for Classes 1 and 2 and increase for Class 3, queue waits increase significantly for Classes 1 and 2 and decrease for Class 3 for both the Random Weights and Golestani approaches (Table 2). Thus, the importance of selecting weights that are appropriate for the performance needs of the various classes of traffic are seen. The Random_weights_1 mean queue waits remain higher than the Golestani_1 mean queue waits for this case (between 8 percent and 21 percent higher), as was also seen in the Case 1. Similar increases in jitter versus the Case 1 are also seen, with Random_weights_1 giving higher jitter values than Golestani_1 for this case (Table 3).

Table 4 summarizes the mean queue waits for the various cases having unequal class weights, and Table 5 shows the 99.9 percent jitter for the same cases. Table 6 gives the packet loss probabilities, which are non-zero for the heavier traffic cases. Percent differences are also given in all tables. The decreasing percent differences between the two approaches as the total load increases is evident. Figures 5 and 6 show these results in graphical form, only for the cases with total offered load less than one, and the fairly close tracking of the Random_weights_1 and Golestani_1 results are seen.

Run speeds are significantly less for our custom simulators versus the OPNET Modeler, especially when the OPNET Modeler post-processing that is required to summarize the measures of interest is considered. Table 7 gives the CPU time required for the Random_weights_1 custom simulator a per-packet basis due to the differing numbers of packets used in runs from the two simulators. CPU times are longer for the OPNET Modeler than our own simulator written in VBA (e.g., about 20 seconds for our simulator versus 322 seconds for the OPNET Modeler, with similar numbers of packets). After the OPNET Models are run, the results must be exported to a text file for each traffic class and statistic type (queueing delay by packet and traffic dropped over time). The data is then imported into Microsoft Access due to the large number of records in each of the exported text files and the limitations in the number of records in a Microsoft Excel spreadsheet. Queries must then be run in Microsoft Access to summarize the mean queue wait, extract the 99.9 percentile of queue wait, and obtain the mean packet loss. This whole process takes about an hour. Our own simulators in VBA were easily programmed to compute all performance measures of interest within the CPU times given in Table 7, so additional post-processing is not required. The Random_weights_1 and Golestani_1 simulators have similar CPU times. The custom simulators were run on a Dell Latitude D810 laptop (with a 2.13 Ghz CPU and 1.0 GB of RAM), while the OPNET Models were run on a workstation with a 2.8 Ghz CPU and 3.5 GB of RAM.

6 SUMMARY AND NEXT STEPS

We infer from the results of the above simulations that there are some differences between our Random Selection based on the weights method of modeling CBWFQ and the other methods investigated. The Golestani approach and the OPNET Modeler's implementation of CBWFQ have lower estimated packet queue waits than our Random Selection based on the weights method of modeling CBWFQ. In the cases we investigated, our Random Selection method gave especially higher estimates than the Golestani approach for classes with more than enough allocated bandwidth to carry the offered traffic. The Golestani approach to CBWFQ scheduling appears to be similar to the OPNET Modeler's implementation. Under emergency conditions with traffic up to 10 times the normal load, and for classes whose weights do not provide sufficient bandwidth to handle their traffic load, differences between the Random Selection based on weights method and the Golestani method decrease. Based on the scenarios we modeled, the OPNET Modeler appears to utilize the Golestani method for CBWFQ.

Our custom simulators provide more efficient analysis capabilities than the commercial simulator for several reasons. The custom simulator run times are faster than the commercial simulator, and they also can be programmed to quickly compute the desired measures of effectiveness to minimize or eliminate the need for post-processing of output data. The OPNET Modeler appears to use the Golestani approach for CBWFQ, and thus we suggest using the Golestani approach for custom simulators. The Random Selection based on the weights method gives similar results to the Golestani method in emergency-type overload traffic, but may have the advantage of being more amenable as a model for developing analytic queueing models of CBWFQ.

We plan to conduct additional work in several areas. We have begun researching more appropriate statistical distributions for modeling video traffic, using actual packet trace data from video equipment. Our simulation model will then be enhanced to utilize the selected video distributions. In addition, we are investigating several approaches to analytic approximations of CBWFQ, one of which is a Markov Chain approach. Our custom simulators are being used to calibrate the analytic results with greater speed and flexibility than the commercial CBWFQ models, and, thus provide needed capabilities for future work.

Table 2: Mean Queue Waits for Original Weights Versus Equal Weights Cases

Case	w1	w2	w3	Mean Queue Wait (ms)					
				Random_ Weights_1, Class 1	Random_ Weights_1, Class 2	Random_ Weights_1, Class 3	Golestani_ 1, Class 1	Golestani _1, Class 2	Golestani_ 1, Class 3
1	0.40	0.45	0.15	12.38	7.88	20.57	10.50	5.37	19.20
2	0.33	0.33	0.33	16.68	9.79	12.31	15.36	8.11	10.76

Table 3: 99.9 Percent Jitter for Original Weights Versus Equal Weights Cases

Case	w1	w2	w3	99.9 Percent Jitter (ms)					
				Random_ Weights_1, Class 1	Random_ Weights_1, Class 2	Random_ Weights_1, Class 3	Golestani_ 1, Class 1	Golestani_ 1, Class 2	Golestani_ 1, Class 3
1	0.40	0.45	0.15	122.82	73.72	238.77	110.48	47.20	228.02
2	0.33	0.33	0.33	167.70	101.02	127.39	161.97	69.01	108.12

Table 4: Mean Queue Waits

								Percent Differences		
Case	Total Rho	Random_ Weights_ 1, Class 1	Random_ Weights_ 1, Class 2	Random_ Weights_ 1, Class 3	Golestani_1, Class 1	Golestani_1, Class 2	Golestani_1, Class 3	Class 1	Class 2	Class 3
4	0.59	6.72	5.35	9.45	5.83	3.80	8.65	15.2	40.9	9.2
1	0.74	12.38	7.88	20.57	10.50	5.37	19.20	17.9	46.6	7.1
3	0.88	26.52	11.28	70.19	22.95	7.69	67.74	15.5	46.6	3.6
5	0.97	46.37	13.65	334.60	42.42	9.72	342.02	9.3	40.5	-2.2
6	6.64	154.67	14.18	10615.43	155.74	10.57	10703.80	-0.7	34.2	-0.8

Table 5: 99.9 Percent Jitter

								Percent Differences		
Case	Total Rho	Random_ Weights_ 1, Class 1	Random_ Weights_ 1, Class 2	Random_ Weights_ 1, Class 3	Golestani_1, Class 1	Golestani_1, Class 2	Golestani_1, Class 3	Class 1	Class 2	Class 3
4	0.59	73.65	57.98	123.21	66.46	35.58	115.42	10.8	63.0	6.7
1	0.74	122.82	73.72	238.77	110.48	47.20	228.02	11.2	56.2	4.7
3	0.88	189.82	87.77	726.60	183.44	57.94	660.46	3.5	51.5	10.0
5	0.97	227.65	96.31	2110.48	221.06	66.85	2141.63	3.0	44.1	-1.5
6	6.64	282.11	88.78	12283.85	274.51	62.11	12366.07	2.8	42.9	-0.7

Table 6: Packet Loss Probabilities

								Percent Differences		
Case	Total Rho	Random_ Weights_ 1, Class 1	Random_ Weights_ 1, Class 2	Random_ Weights_ 1, Class 3	Golestani_1, Class 1	Golestani_1, Class 2	Golestani_1, Class 3	Class 1	Class 2	Class 3
4	0.59	0.000	0.000	0.000	0.000	0.000	0.000	N/A	N/A	N/A
1	0.74	0.000	0.000	0.000	0.000	0.000	0.000	N/A	N/A	N/A
3	0.88	0.001	0.000	0.000	0.001	0.000	0.000	29.0	N/A	N/A
5	0.97	0.007	0.000	0.000	0.006	0.000	0.000	23.3	N/A	N/A
6	6.64	0.840	0.000	0.893	0.842	0.000	0.893	-0.1	N/A	-0.1

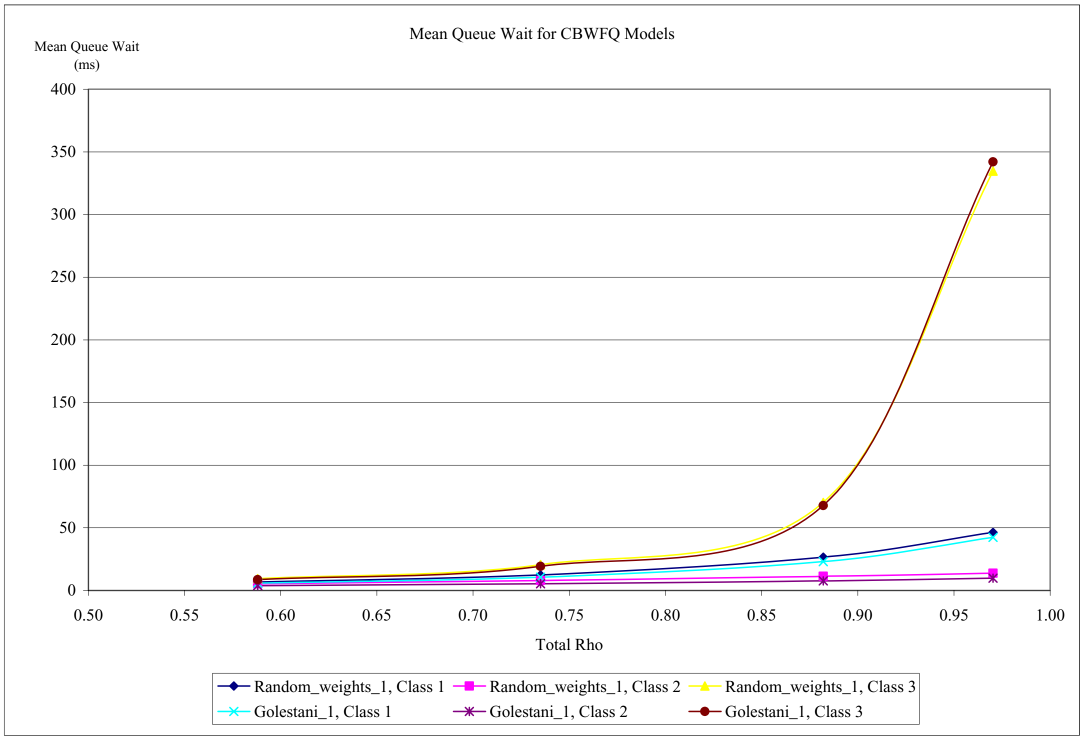

Figure 5: Mean Queue Wait Time

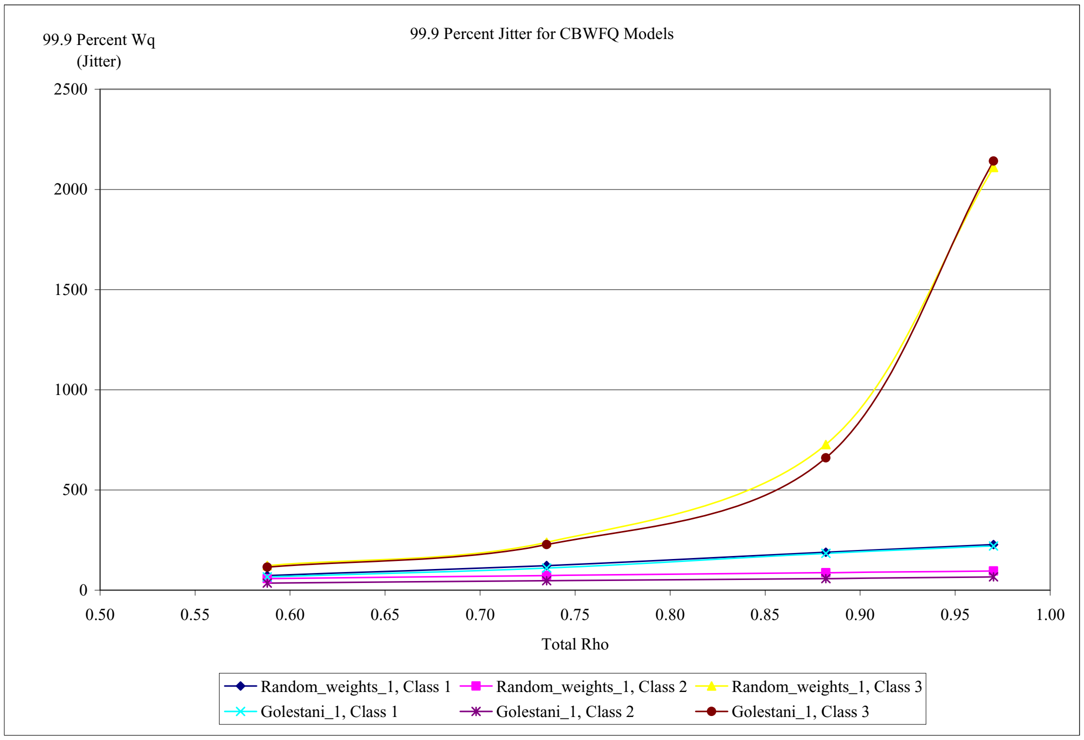

Figure 6: 99.9 Percent Queue Wait (Jitter)

Table 7: Simulator Execution and Processing Time Requirements

Random_Weights_1			OPNET Modeler			
Packets Simulated	CPU Time (sec)	CPU Seconds Per Packet	Packets Simulated	CPU Time (sec)	CPU Seconds Per Packet	Post Processing Time (hrs)
1,000,000	19.8	2.0E-05	999,986	322.0	3.2E-04	≈1 hour

ACKNOWLEDGMENTS

This work was partially funded by the National Communications System Contract Number NBCH-D-02-0039 (Task Order Number D0200390095). Research funding was also provided by the Center for Network-Based Systems, a research collaboration of Noblis and George Mason University. We would like to thank Dr. Pat McGregor of Nyquetek for reviewing our assumptions and giving us feedback.

REFERENCES

Baker, F., Polk, J., and M. Dolly, "An EF DSCP for Capacity-Admitted Traffic," draft-ietf-tsvwg-admitted-realtime-dscp-00 (work in progress), December 2006, *http://tools.ietf.org/id/draft-ietf-tsvwg-admitted-realtime-dscp-00.txt.*

Baker, F., J. Polk, and M. Dolly, "DSCPs for Capacity-Admitted Traffic," draft-ietf-tsvwg-admitted-realtime-dscp-01 (work in progress), March 2007; *http://www.ietf.org/internet-drafts/draft-ietf-tsvwg-admitted-realtime-dscp-01.txt.*

Cao, J., W. S. Cleveland, D. Lin, and D. X. Sun, "Internet Traffic Tends Toward Poisson and Independent as the Load Increases," *Nonlinear Estimation and Classification,* Editors: C. Holmes, D. Denison, M. Hansen, B. Yu, and B. Mallick, Springer, New York, NY, 2002; *http://cm.belllabs.com/who/dong/papers/lrd2poisson.pdf.*

Cisco IOS Quality of Service Solutions Configuration Guide, Release 12.2 (Congestion Management Overview chapter); *http://www.cisco.com/en/US/proucts/sw/iosswrel/ps1835/products_configuration_guide_chapter09186a00800b75a9.html.*

Cohen, J. W., "The Single Server Queue," North-Holland Publishing Company, New York, 1969.

Davie, B. "Is QoS Necessary? Quality of Service Mechanisms vs. Bandwidth Provisioning," Fall 2002 Seminar/Public Lecture Series, Stanford University School of Engineering, U.S. Asia Technology Management Center, 2002.

Fischer, M. J., and D. M. B. Masi, "Voice Packet Arrival Models and Their Effect on Packet Performance," *Applied Telecommunications Symposium*, San Diego, CA, April 3–7, 2005.

Fischer, M. J. and D. M. B. Masi, "A Quantitative Analysis of the Voice and Data Quality of Service Problem," *The Telecommunications Review 2007*, Noblis, Falls Church, VA.

Golestani, S. J., "A Self-Clocked Fair Queuing Scheme for Broadband Applications," *Proceedings of the IEEE INFOCOM,* 1994.

Gross, D. and C. M. Harris, "Fundamentals of Queueing Theory," Third Edition, John Wiley, New York, NY, 1998.

Masi, D. M. B. and M. J. Fischer, "Voice over Internet Protocol (VoIP) Performance Models—A Comprehensive Approach", International Conference on Telecommunication Systems–Modeling and Analysis (ICTSMA), Dallas, TX, November 17–20, 2005.

Schrage, L. E., and L. W. Miller, "The Queue M/G/1 with the Shortest Remaining Processing Time Discipline," *Operations Research 14*, pp. 670–684, 1966.

Semeria, C., "Supporting Differentiated Service Class: Queue Scheduling Disciplines," *Juniper White Paper,* 2001; *http://www.juniper.net/solutions/literature/white_papers/200020.pdf.*

AUTHOR BIOGRAPHIES

DENISE M. BEVILACQUA MASI is a senior principal engineer at Noblis. Her experience and research interests include queueing theory and simulation applied to telecommunications networks. She received her doctorate degree in information technology and engineering at George Mason University. Her email address is *dmasi@noblis.org.*

MARTIN J. FISCHER is a senior fellow at Noblis. His experience includes network design and performance analysis in telecommunications. He has published approximately 40+ articles in refereed journals. He received his doctorate degree in Operations Research from Southern Methodist University. His email address is *mfischer@noblis.org.*

DAVID A GARBIN is a senior fellow at Noblis. His 30+ years of experience is in the telecommunications and networking field, focusing on the design and economic analysis of large networks, both for carriers and their customers. His current duties include research into providing

quality of service in convergent IP networks and advising government agencies in the acquisition and implementation of VoIP. He holds advanced degrees from MIT and is the co-author of the best-selling New McGraw-Hill *Telecom Factbook*. His email address is *david.garbin@noblis.org*.