PHILIPPINE ENGINEERING JOURNAL
PEJ 2005; **Vol. 26 No. 2**:57–74

Value-Based Utility Broker for Jitter Management of Voice on IP Networks†

Jhoanna Rhodette I. Pedrasa[1]*and Cedric Angelo M. Festin[2]

[1] *Department of Electrical and Electronics Engineering*
University of the Philippines Diliman
Quezon City 1101 PHILIPPINES

[2] *Department of Computer Science*
University of the Philippines Diliman
Quezon City 1101 PHILIPPINES

ABSTRACT

A variety of scheduling algorithms and management frameworks have been proposed to provide more than just best-effort service offered by the Internet Protocol. A new framework, Value-Based Utility (VBU), uses the perceived knowledge of the state and degree of user satisfaction in managing router resources and functions. VBU has been shown to effectively manage packet loss and delay in times of high demand and low resource availability. This paper applies VBU to jitter management by enhancing on two existing scheduling algorithms, namely Stop-and-Go and Jitter-EDD. Our results show that Stop-and-Go with Utility can achieve a tighter jitter bound of T, where T is the frame length, which is half of the original bound of 2T offered by Stop-and-Go alone. VBU has limited application however to Jitter-EDD, where the rigid environment restricts the re-allocation of resources according to demand.

1. INTRODUCTION

Voice over IP (VoIP) has been pushed as the next killer application in the market as early as the 1980's but it has not really taken off until in recent years. The biggest impediment to VoIP is the poor voice quality and call latency [1]. A packet-switched network may deliver packets containing voice signals at different times or in a different order. Packets may even be dropped when congestion occurs. As a result, voice quality easily degrades with changing network conditions.

Much research has been devoted to establishing mechanisms to guarantee a service quality over the IP network for applications that require more than just best-effort service. These mechanisms have all been in terms of quantitative requirements on network parameters. They fail to address the issue of varying user expectation and how much these expectations have

*Correspondence to: Department of Electrical and Electronics Engineering, University of the Philippines Diliman, Quezon City 1101 PHILIPPINES. email:
†This project is supported in part by the University of the Philippines Diliman Office of the Vice Chancellor for Research and Development under Grant No. 030305 TNSE-S.

Received April 24, 2007
Revised July 29, 2007
Accepted October 11, 2007

been met. Such are the motivations behind a new management framework, Value-Based Utility (VBU), which argues that knowing the state and degree of user satisfaction allows one to manage the network resources more effectively.

Our study applied the value-based utility model to two existing management schemes, Stop-and-Go and Jitter-EDD, to manage voice jitter. In particular, this research developed and implemented a value-based utility broker that handles voice calls on a single-hop IP network. Management decisions were targeted at optimizing jitter requirements for voice while still maintaining a balance in packet loss and delay and without penalizing non-voice traffic. Parts of our results have already been published in [2], [3].

The rest of this paper is organized as follows. Section 2 discusses the Value-Based Utility framework. In Section 3, we present the simulation environment used for experimentation and the evaluation process. Section 4 tackles the original Stop-and-Go algorithm and our enhancements to it using VBU and we show our results for Stop-and-Go in Section 5. Section 6 moves to the other algorithm, Jitter-EDD, and the corresponding results in Section 7. Finally we summarize our results and make our concluding statements in Section 8.

2. VALUE-BASED UTILITY

The term *utility* refers to the fields of business and economics. *Utility* is defined as the measure of the satisfaction gained from the consumption of a package of goods and services [4]. Although satisfaction is very subjective and differs for each individual, utility is used as an indicator of the market's willingness to own and pay for a particular good or service. In this study we define and use utility in a slightly different manner. We use utility to express a user's preference for a particular service but this preference is not necessarily linked to their willingness and capability to pay for it. The preference we deal with is based solely on need.

The Value-Based Utility (VBU) Model [5] is a new framework for resource management of computer networks. It is based on the assertion that it is not enough to achieve quantitative measures of QoS; there is a need to address the qualitative aspects of user predilection. It is important to know not just whether the expectations are met, but to what extent as well. This additional information can then be used to manage the resources more efficiently.

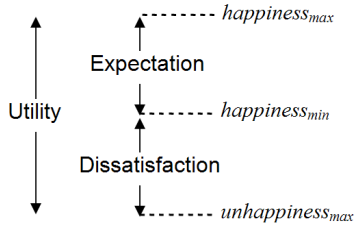

Figure 1. User Utility Range as Defined by the Value-Based Utility Model

The Value-Based Utility model defines an expectation range for each user and determines the user satisfaction in terms of how well these expectations are met (or how bad if they are not met) through a utility function. Figure 1 shows the utility range as some level of expectation or dissatisfaction [5]. The expectation range has at the extremes $happiness_{max}$,

the state of highest user satisfaction, and $happiness_{min}$, the state where only his minimum expectations are met. Falling below this threshold level will render the user dissatisfied, with $unhappiness_{max}$ as the state of extreme dissatisfaction.

The Value-Based Utility function, given in Equation 1, is an expression to represent both the state and degree of user satisfaction (or dissatisfaction).

$$U_{i,QoS,m,\Delta t(p,b)} = \frac{G}{N} - \frac{N-G}{N} \times \frac{p}{q} \tag{1}$$

where $U_{i,QoS,m,\Delta t(p,b)}$, flow i's utility for the specified QoS at point m during the time interval Δt

p, target percentage of packets that should meet QoS requirements,
b, target QoS bound,
G, number of packets meeting flow i's requirements,
N, total number of packets seen,
q, equal to $1 - p$

The result of the utility equation incorporates both the state and degree of user well being. A user is satisfied if $U \geq 0$, implying that user expectations were met. Otherwise a user is in a state of dissatisfaction if $U < 0$. The magnitude of the utility indicates the degree of satisfaction (dissatisfaction) of the user.

The three major points bounding the utility range in Figure 1 can be evaluated using Equation 1. The state of maximum happiness is achieved when all packets meet the QoS requirement (G=N), resulting to $happiness_{max}$ being equal to 1. The threshold value $happiness_{min}$ can be derived as the point where the fraction of good packets to the total number of packets is equal to the target percentage p. Evaluating Equation 1 by setting $p = G/N$ results to $happiness_{min} = 0$. Finally, the maximum state of dissatisfaction $unhappiness_{max}$ is experienced when utility is equal to $-p/q$.

The study [5] proved through simulations that Value-Based Utility could be used to manage router functions and services more efficiently. In particular, it was able to demonstrate the effectiveness of VBU at minimizing loss and delay through scheduling and buffer management. Our study builds on these results to investigate if this new framework can be used for jitter management as well. The novelty of this framework is that it attempts to capture the varying levels of user satisfaction (dissatisfaction) and use this data to allocate network resources more efficiently. This is particularly important for voice and video applications wherein perceived quality varies widely from person to person.

Our study arrived at a new architecture that uses a *utility broker* to manage the network. As with the bandwidth broker in the Differentiated Services architecture, the utility broker is an agent that is responsible for a particular domain and is tasked with efficiently allocating network resources. Management decisions are not based solely on organizational policies however. Instead, the utility broker makes decisions based on perceived user utilities. Note however that this utility broker does not have the power to allocate bandwidth from source to destination, as the focus of our study was to minimize jitter through buffer management.

3. EXPERIMENTAL ENVIRONMENT

Network simulation has proven to be an invaluable research tool allowing for quick evaluation of new algorithms and protocols. We test our assertions using the Network Simulator 2 (NS2).

The algorithms were implemented in C++ as a sub-class of the existing *Queue* class in NS2. Voice traffic sources and receivers were coded by adding to the *Application* class. The existing UDP module was also modified so that the queue manager at the routers will be able to distinguish voice data from other packets. OTcl scripts were written to set-up the network, configure the sources and run the simulations.

3.1. Simulation Environment

Voice is modeled as a 64kbps PCM-encoded data with no silence suppression. Each packet contains 20 ms worth of voice or 160 bytes of data and is encapsulated in 28 bytes of UDP and IP headers. This amounts to each voice source transmitting 188 bytes every 20 ms, or a data rate of 75.2kbps each. Non-voice traffic generators were also implemented aside from voice data. A non-voice source divides time into 20-ms frames and sends one UDP packet for each frame. The packet size is uniformly distributed between 40 bytes and 1040 bytes. The maximum data rate of non-voice traffic is 416 kbps while the average data rate is 216 kbps.

A single hop scenario is explored by having voice sources residing on a Local Area Network connect to a common gateway. The gateway is linked to the receiver over a 1Mbps dedicated line. Thirteen voice sources can be supported at an output bandwidth of 1Mbps, resulting to 97.7% utilization of the output link from the gateway. Our utility management scheme is then implemented as the scheduling mechanism of the gateway's output queue. Test cases with non-voice data maintained the same number of voice sources (thirteen) but additionally has a single non-voice traffic source connected to the gateway. The output bandwidth was correspondingly increased to 1.4Mbps during the mixed traffic test scenarios to accommodate the additional non-voice load.

3.2. Test Cases

Expectation levels reflect how discriminating the users are of their jitter requirements and directly impact the utility value calculation. Different user expectations were modeled by varying the value of p, the target percentage of packets that should meet jitter requirements, in Equation 1. The higher the value of p, the higher is the number of 'good' packets needed in order to satisfy the user. Three representative expectation levels were used, namely the *High Expectation Flows* (HEF), *Medium Expectation Flows* (MEF), and *Low Expectation Flows* (LEF). Table I lists the corresponding values of p assigned to each level.

Expectation Level	p
High Expectation Flow (HEF)	0.99
Medium Expectation Flow (MEF)	0.90
Low Expectation Flow (LEF)	0.80

Table I. Values of p for Each Expectation Level

Three test cases, one for each expectation level, make up the homogeneous expectation mix. Three other test cases explore the heterogeneous expectation mix, which has flows belonging to different expectation levels. The number of flows belonging to each expectation level for these different expectation mixes is summarized in Table II. The labels use the hexadecimal system to indicate the number of flows for each expectation level.

Each of the test scenarios was simulated over different jitter bounds to reflect a range of requirements. Stop-and-Go jitter measurements were in terms of inter-packet delay and the jitter bounds were set at 5 ms, 10 ms, 15 ms, 20 ms, and 30 ms for these experiments. Jitter-EDD measurements are in terms of end-to-end delay rather than inter-packet delay. Jitter bounds were adjusted accordingly to reflect the fact that Jitter-EDD treats the delay bound as a hard deadline, considering packets arriving D + j sec to be "'bad"' already. The jitter bounds were set at 2.5 ms, 5 ms, 7.5 ms, 10 ms, 15 ms, and 20 ms for Jitter-EDD. These bounds correspond to half the jitter bounds for Stop-and-Go experiments.

Label	**HEF**	**MEF**	**LEF**
$w00d$	0	0	13
$w0d0$	0	13	0
$wd00$	13	0	0
$w445$	4	4	5
$w805$	8	0	5
$w850$	8	5	0

Table II. Number of Flows for Each Expectation Mix

3.3. Evaluation Process

One connection per expectation flow is monitored during each simulation to limit the parameters being observed. The number of simulation runs (500 for each test case) is large enough that these representative flows can adequately capture the behavior of all the sources belonging to a particular expectation flow. We utilize the ratio of satisfied users as well as average utility values to evaluate our scheme. These are the same measurement parameters used in [6].

We assert in this study that value-based utility can be applied to the assessment and management of voice delay jitter. Although utility values can be used as a measure to determine the effectiveness of utilities, the ultimate question is whether more users can indeed be satisfied. Such information can be extracted by counting the number of experiments when a representative flow is happy. These *happiness plots* show the ratio of number of experiments with the user registering a happy utility over the total number of experiments. A value of one (1.0) means that the user is satisfied in all the simulations while a value of 0.40 means that the user is happy in only 40% of the simulations.

Some test scenarios have very strict jitter requirements that no user can be satisfied. In such cases, the average utility is the best means to evaluate if any improvement was achieved. This is represented as *utility plots* showing the average utility value of the representative flows across the experiments. Two average values are computed to differentiate the utility values of the happy flows from the unhappy ones. A negative utility represents the mean of the dissatisfied flows while a positive utility is the mean of the satisfied flows. These average values are plotted using a 95% confidence interval.

4. STOP-AND-GO

In this section we discuss the original Stop-and-Go algorithm and present our enhancements to it using VBU.

4.1. Stop-and-Go Algorithm

Stop-and-Go [7] or S&G, utilizes a framing strategy to achieve hard delay and jitter bounds. In such a scheme, time is divided into fixed frames of constant length T. S&G defines arriving packets to be (r, T)-smooth if during each frame of length T the arrived packets collectively have no more than $r \cdot T$ bits. Every connection k declares a transmission rate r_k and its packet arrival to the network is required to be (rk, T)-smooth. Any new packet that violates this limit is not admitted until the next frame starts.

Stop-and-Go queuing defines arriving and departing frames on each switch. A packet which arrives at a switch during frame f_n and is (r, T)-smooth becomes eligible for transmission only on the next time frame f_{n+1}. By holding the packets, they are guaranteed to experience an interpacket delay jitter d at each link that is bounded by the size of a frame. This is because any packet that arrives during frame f_n is guaranteed to be serviced before the following frame f_{n+1} expires. The end-to-end queuing delay jitter Q is thus bounded by

$$HT \leq Q \leq 2HT, \text{ where } H = \text{number of hops} \tag{2}$$

Any packet service policy may be applied for packets belonging to the same time frame in Stop-and-Go. The paper [7] suggests using FIFO queuing in order to simplify the implementation and to maintain packet sequence. We conjecture however that using a priority queue, with utility as the basis for determining priorities, can help some packets achieve tighter jitter bounds, albeit at the expense of the less demanding sources.

4.2. Delay - Bandwidth Granularity

The Stop-and-Go framing strategy introduces a coupling between the delay bound and packet allocation granularity [7], [8]. Lower delay bounds can be achieved by using smaller frame sizes. Small frame sizes however come at the cost of having to reserve a higher bandwidth.

Assume that only packets of size Γ are seen on the network. Then the incremental step of bandwidth allocation, Δr can be expressed as $\Delta r = \Gamma/T$ bits/sec. Substituting this into Equation 2 and re-arranging will yield

$$H\Gamma \leq \Delta r \cdot Q \leq 2H\Gamma \tag{3}$$

Equation 3 shows that for a fixed frame size and a fixed path traversed throughout the connection, queuing delay and bandwidth cannot be minimized simultaneously. Reducing the delay would mean a corresponding increase in bandwidth allocation granularity.

Let us take as an example a single-hop network using a packet size of 188 bytes for voice traffic. If the desired queuing delay bound is 20 ms and assuming worst-case scenario, then Δr becomes 150.4 kbps. This means that at a link capacity of 1Mbps, only six voice connections can be supported simultaneously. However, if we relax the delay bound to 40 ms, then Δr decreases to 75.2 kbps and 13 connections can now be supported. This shows why Stop-and-Go is unsuitable for low delay, low throughput applications such as voice.

A generalized algorithm using multiple frame sizes was presented as a solution to the delay-bandwidth coupling problem [7]. Packets belonging to the same frame observe the Stop-and-Go rule over these frames. An additional rule is observed for packets of different frame sizes. Packets that belong to the smaller-sized frame have a non-preemptive priority over packets on a larger frame size.

Multiple frame sizes lets connections with low delay bounds to be assigned to smaller frame sizes while allowing for finer bandwidth allocation over the larger-sized frames. However, this scheme does not completely solve the problem of delay-bandwidth coupling, as it still exists within each frame.

4.3. Stop-and-Go with Utility Algorithm

We assert that we can achieve tighter jitter bounds in Stop-and-Go by using utility in assigning priorities to the packets. The same can be achieved by using smaller frame lengths, but doing so will mean less number of connections can be supported simultaneously. On certain instances, our proposed scheme can even allow connections with varying jitter bounds to be aggregated into the same frame length, eliminating the need for multiple framing.

Our claim is made based on the following observations. Assume two levels of priority. High priority packets are enqueued at the head of the queue while low priority packets are enqueued at the end of the queue. Let n_H be the number of high priority connections and n_L the remaining connections of low priority. Denote the service time of each packet as s, which in our case is the time it takes to transmit the packet over the outgoing link. We can verify that the minimum inter-packet delay, $d_{H,min}$, occurs when the packet $p_{j,n}$ is transmitted last among the high priority packets on the previous frame and the succeeding packet $p_{j,n+1}$ gets transmitted first on the next frame. Thus $d_{H,min}$ can be calculated as $d_{H,min} = T-(n_H - 1)\, s$. The converse situation will yield the maximum inter-packet delay, $d_{H,max}$, as $d_{H,max} = T + (n_H - 1)\, s$. The inter-packet delay jitter is thus bounded by

$$j_H = d_{H,max} - d_{H,min} = 2\,(n_H - 1)\, s \tag{4}$$

Similarly, we can determine the best case and worst case scenarios for a connection of low priority, which we find to be similar to those previously discussed for a connection of high priority. The minimum and maximum inter-packet delays are given by $d_{L,min} = T-(n_L - 1)\, s$ and $d_{L,max} = T + (n_L - 1)\, s$, respectively. The effective delay jitter bound is thus

$$j_L = d_{L,max} - d_{L,min} = 2\,(n_L - 1)\, s \tag{5}$$

Note that Equations 4 and 5 are very similar to each other. We make the following observations:

- The jitter bound of the high priority connections is dependent only on the service time and on the number of active high priority flows. It does not matter how many low priority flows there are.
- The lesser number of flows belonging to the same priority level, the tighter the effective jitter bound for that level.

We modify the original S&G algorithm to keep a list of connections to be considered of high priority and use utility values to determine priorities. The receivers communicate their current states to the scheduler by sending an acknowledgement packet containing their jitter utility

value every second. The most dissatisfied flow, the one with the lowest negative utility value, is added to the list of high-priority connections. At most one connection is elevated as high priority in each round to minimize oscillations in the utility values.

5. STOP-AND-GO RESULTS

In this section we show our experimental results for Stop-and-Go with utility and compare it with the original Stop-and-Go.

5.1. Homogeneous Expectation Mix

We run simulations assuming all connections are synchronized and everyone starts transmission at the same time. We fix all sources at the same expectation level for each simulation using a frame of length 20 ms. Figure 2 shows the ratio of satisfied users for the homogeneous expectation test cases *w00d*, *w0d0* and *wd00* over different jitter bounds for S&G and S&G with utility. It can be seen that there is a marked increase in the number of happy flows with the application of utility. The jitter bound is decreased to 15 ms for both low and medium expectation flows from the original 30 ms. High expectation flows are happy 100% and about 97% of the time at jitter bounds of 30 ms and 20 ms, respectively.

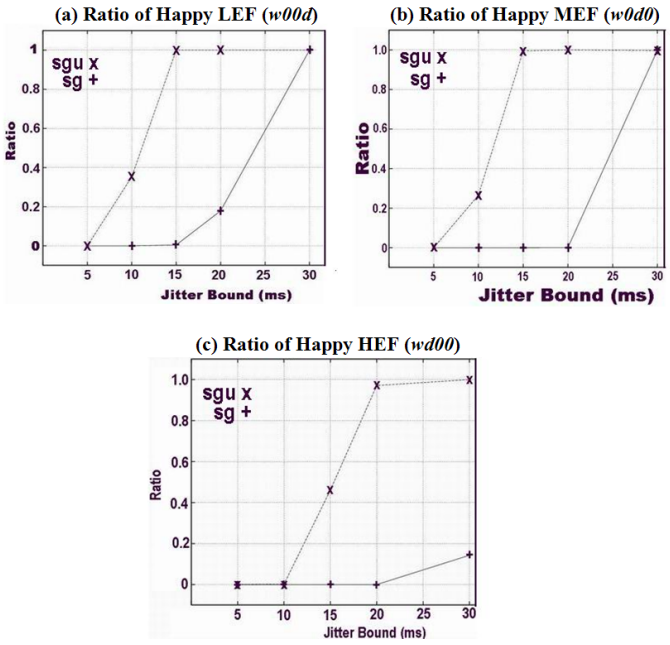

Figure 2. Ratio of happy flows over different jitter bounds for S&G (sg) and S&G with utility (sgu). (a) Low Expectation Flows (b) Medium Expectation Flows (c) High Expectation Flows. A marked increase in the number of satisfied flows can be observed with the application of utility.

Figure 3 shows the utility values of the homogeneous expectation flows. Positive and negative

utility values reflect the average utility for the happy and unhappy flows, respectively. Jitter bounds with two utility values indicate that some flows are satisfied while some flows are unhappy. We can see that the for all expectation levels, the average utility of the flows is higher with management than without management. This is true even for cases wherein no flows are satisfied, such as at a jitter bound of 5 ms.

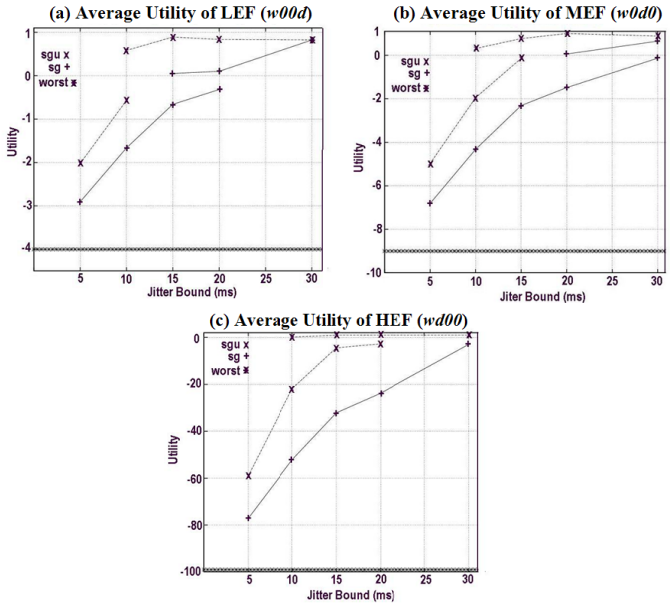

Figure 3. Average utility over different jitter bounds for S&G (sg) and S&G with utility (sgu). (a) Low Expectation Flows (b) Medium Expectation Flows (c) High Expectation Flows. The average utility of the flows is higher with management than without management for all expectation levels.

5.2. Heterogeneous Expectation Mix

The heterogeneous expectation mix covers the more general scenario of having different expectation flows at the same time. Figure 4 shows the ratio of satisfied users for the test case *w445*. It can be seen that for all the expectation flows, there is an increase in the number of happy users with the application of utility. Note too that we achieve the same bound on the ratio of satisfied users for each expectation flow alone. For instance, all flows are satisfied at jitter bounds of 15 ms and higher for low expectation and medium expectation flows with utility. This is also the bound at which we can satisfy all users for the corresponding homogeneous test cases *w00d* and *w0d0*. Similarly, we can satisfy about 97% of the high expectation flows at 20 ms, the same percentage for the homogenous test case *wd00*.

These same observations hold true for the two other heterogeneous test cases *w805* and *w850*. The results show that the heterogeneous expectation mix follows the trend observed in the homogeneous cases.

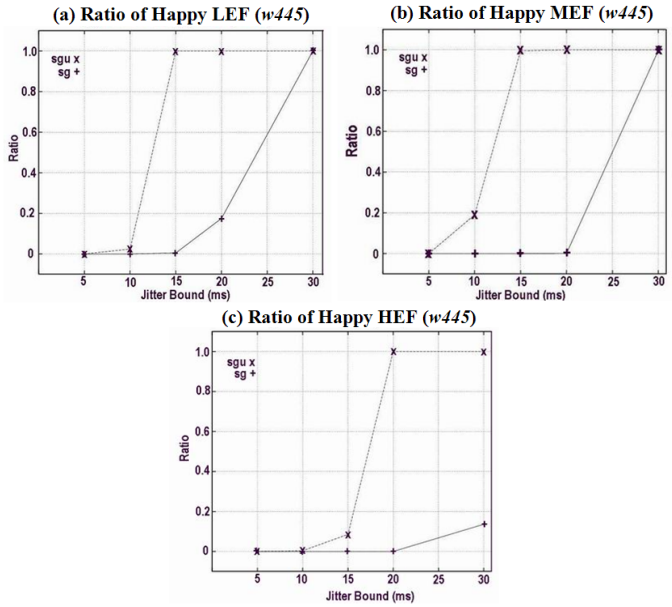

Figure 4. Ratio of happy flows over different jitter bounds for S&G (sg) and S&G with utility (sgu) for the heterogeneous test case w445. (a) Low Expectation Flows (b) Medium Expectation Flows (c) High Expectation Flows. We achieve the same bound on the ratio of satisfied users for each homogeneous expectation mix.

5.3. Scale-up

In order to determine if these results will be valid as we increase the number of users, we run experiments scaling up the output bandwidth to 2Mbps and 4Mbps. This effectively allows us to support 26 and 53 voice connections, respectively. Table III shows the ratio of happy users as we scale-up the bandwidth for the three homogeneous test cases. We can see that the ratio of satisfied users for Stop-and-Go is constant for each expectation level even as we add more users. It is evident that we gain improvements with the application of utility, as there is a significant increase in the ratio of satisfied flows. For Stop-and-Go with utility, we have a transition period (jitter bound of 15 ms) where there is a mix of happy and unhappy users. It is only in these transition periods that we observe slightly different values. In general, we can conclude that we achieve similar results as we increase the number of users.

5.4. Mixed Traffic

All the test cases previously mentioned are pure voice traffic mixes wherein all traffic flowing through the network are voice packets. Additional test cases explore the effect of non-voice traffic in the network. The mixed traffic test scenarios are the same as the pure voice test cases, with the addition of a non-voice source in the network sending one random-sized packet every frame. The additional traffic entails a corresponding increase in the output bandwidth to 1.4Mbps.

Figure 5 shows the ratio of happy users over different jitter bounds for Stop-and-Go and

(a) Ratio of Happy Low Expectation Flows (*w00d*)

Number of Users	Stop and Go					Stop and Go with Utility				
	Jitter Bound					Jitter Bound				
	5ms	10ms	15ms	20ms	30ms	5ms	10ms	15ms	20ms	30ms
13 users	0	0	0	0.18	1.0	0	0.35	1.0	1.0	1.0
26 users	0	0	0	0.28	1.0	0	0.30	1.0	1.0	1.0
53 users	0	0	0	0.18	1.0	0	0.30	1.0	1.0	1.0

(b) Ratio of Happy Medium Expectation Flows (*w0d0*)

Number of Users	Stop and Go					Stop and Go with Utility				
	Jitter Bound					Jitter Bound				
	5ms	10ms	15ms	20ms	30ms	5ms	10ms	15ms	20ms	30ms
13 users	0	0	0	0	1.0	0	0.25	0.97	1.0	1.0
26 users	0	0	0	0	0.98	0	0.04	0.92	1.0	1.0
53 users	0	0	0	0	0.96	0	0.04	0.84	1.0	1.0

(c) Ratio of Happy High Expectation Flows (*wd00*)

Number of Users	Stop and Go					Stop and Go with Utility				
	Jitter Bound					Jitter Bound				
	5ms	10ms	15ms	20ms	30ms	5ms	10ms	15ms	20ms	30ms
13 users	0	0	0	0	0.14	0	0	0.48	0.98	1.0
26 users	0	0	0	0	0.09	0	0	0.34	1.0	1.0
53 users	0	0	0	0	0.69	0	0	0.15	1.0	1.0

Table III. Ratio of happy users for S&G and S&G with Utility at a bandwidth of 1Mbps (13 users), 2Mbps (26 users), and 4Mbps (53 users). (a) LEF (b) MEF (c) HEF. The ratio of happy users is constant for each expectation level even with the addition of more users.

Stop-and-Go with Utility. We can easily observe that in general, more users are happy with our management scheme in place. Note too that as with the pure voice scenario, all users are satisfied at bounds of 15 ms, 20 ms, and 30 ms for both the low expectation and medium expectation flows. Around 97% of high expectation flows are satisfied at a bound of 20 ms. This is the same ratio achieved for the homogeneous high expectation test case at the same jitter bound. The ratio differs however at tighter jitter bounds. Our mixed traffic flows have more satisfied users than their corresponding pure voice counterparts.

6. JITTER EARLIEST DUE DATE

Jitter Earliest Due Date (Jitter-EDD) [9] is a deadline-based scheduling algorithm that extends Delay Earliest Due Date (Delay-EDD) [10] to provide hard end-to-end delay and jitter guarantees.

6.1. Channel Establishment

Jitter-EDD defines a real-time channel, k, as a simplex, fixed route connection with performance requirements from the network. In order for the network to guarantee real-time

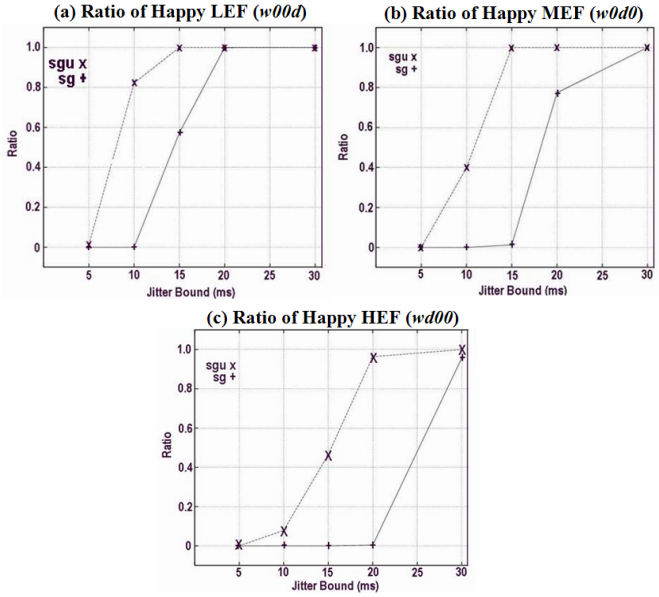

Figure 5. Ratio of happy users over different jitter bounds with non-voice traffic. (a) Low Expectation Flows (b) Medium Expectation Flows (c) High Expectation Flows. As with the pure voice test case, more users can be satisfied with management using utility.

service, clients must declare traffic characteristics such as packet size and minimum packet inter-arrival time. It also negotiates performance requirements with the network, specifically source to-destination delay bound, D, and source-to-destination jitter bound, J .

The goal of the establishment procedure is to break up the end-to-end delay and jitter bounds D_k and J_k into local delay bound $d_{n,k}$ and local jitter bound $j_{n,k}$ at each intermediate node n. As the channel establishment request moves from source to destination, each intermediate node offers a suggested delay bound and a suggested buffer bound. These local bounds are determined by performing tests at each intermediate node to determine if there are enough resources to satisfy this new request. One test in particular, the jitter bound test, ensures that existing channels do not violate their delay bounds with the addition of this new channel. The jitter bound test consists of verifying Equation 6 for all channels i in node n after the channel to be created has been added to the already established ones.

$$j_{n,i} \geq S + T, \text{ for all channels } i \text{ in node } n \tag{6}$$

where $j_{n,i}$, jitter bound of channel i
S, summation of service times of all channels
T , transmit time of largest packet traversing node n

The inequality ensures that the jitter bound of all real-time connections can be satisfied even in the worst case scenario, when connections are transmitting at their minimum inter-arrival rates and two flows have deadlines very close to each other.

6.2. Jitter and rate control

Jitter is controlled at each node by a set of regulators, one for each channel, which delays packets that arrive too early. Each node stamps each packet by a value *ahead*, the difference between the packet's deadline at that node and it's actual completion time. The regulator at the immediate succeeding node holds the packet by this value *ahead* so that it cannot be serviced too early. A bounded-delay server, shared by all channels, places a bound on the maximum delay experienced by each packet in the node. Figure 6 shows the model for each intermediate node.

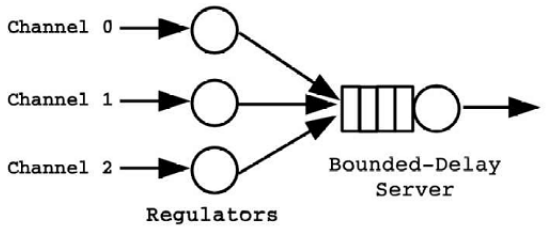

Figure 6. Model for Each Intermediate Note in Jitter-EDD

The bounded delay server adopts the scheduling policy of Delay-EDD [10], a variant of the EDD algorithm [11], which is calendar-based. The server calculates the time it takes to service each packet that arrives. Packets are sorted according to packet deadlines and are served in order of increasing deadlines. The scheme recommends a calendar queue implementation [12] to achieve *O(1)* time complexity when inserting packets in the queue.

7. JITTER-EDD RESULTS

Simulations for Jitter-EDD were run at a delay bound of 20 ms and with jitter bounds of 2.5 ms, 5 ms, 7.5 ms, 10 ms, 15 ms, and 20 ms. We vary the time voice sources start transmitting as well as the user's expectation levels. End-to-end delay jitter utility measured at the receiver provides the basis for analysis.

7.1. Jitter-EDD

We let the packet's start time be a randomly generated number between 0 ms and 20 ms. Figure 7 shows the ratio of satisfied high expectation users over time (rd00). It can be observed that all flows are happy at jitter bounds of 10 ms, 15 ms, and 20 ms. Only 97% can be satisfied at a bound of 7.5 ms, and this ratio decreases to about 74% at 5 ms and even less (24%) at 2.5 ms.

The percentage of happy users for high expectation flows (HEF) is the same for medium expectation flows (MEF) and low expectation flows (LEF). This indicates that the results we achieve are not dependent on the user's expectation level. We examine next their average utility values to determine if there are any patterns in our results.

Table IV summarizes the average utility of Low Expectation Flows over different jitter

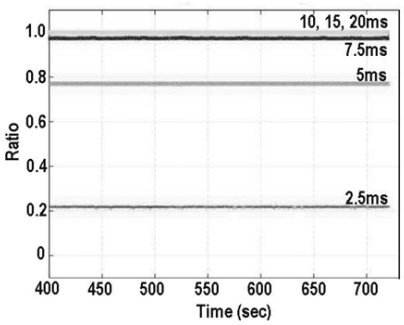

Figure 7. Ratio of happy High Expectation Flows End-to-End Jitter Utility (Random Case)

bounds. It can be observed that except for the jitter bound of 2.5 ms, the average utility of happy flows is 1.0. This means that all received packets met the jitter bound. The table also shows that the average utility of unhappy flows approach the worst-case value (-4), which means that only 2% of the received packets met the jitter bound. These observations of being either in the state of extreme satisfaction or extreme dissatisfaction holds true for the MEF and HEF as well. We can infer that once a connection starts transmitting packets, either all its packets will meet the jitter bound or all its packets will violate the jitter bound.

Jitter Bound	**Unhappy LEF**		**Happy LEF**	
	Mean Utility	Standard Deviation	Mean Utility	Standard Deviation
2.5ms	-3.94	0.40	0.97	0.13
5ms	-3.94	0.37	1.00	0.03
7ms	-4.00	0.00	1.00	0.02
10ms			1.00	0.00
15ms			1.00	0.00
20ms			1.00	0.00

Table IV. Average End-to-End Jitter Utility of Happy and Unhappy Low Expectation Flows (Random case)

These results can be explained by the fact that for Jitter-EDD, the jitter bound is effectively the window by which packets can be transmitted such that it will meet both its jitter bound and delay bound. The delay bound is the upper limit of our window, so a packet that meets its jitter bound will automatically satisfy its delay bound as well. The regulator holds the packet until $d_{n,k} - j_{n,k}$ ms, after which it is eligible for service at the bounded delay server. At this point, the packet must be transmitted within $j_{n,k}$ ms, otherwise it will not be able to meet its deadline. Such violation of the delay bound occurs when the summation of the service times of all the packets before it is greater than its jitter bound. This presents an impossible scheduling constraint that cannot be resolved on the bounded delay server. Such situation can be prevented by satisfying the connection establishment algorithm of Jitter-EDD, particularly the jitter bound test (Equation 6) .

In our simulations, each connection has the same service time (1.508 ms) and jitter requirement. When we apply the jitter bound test, we get that only one connection can be accommodated at a bound of 2.5 ms, three flows at a bound of 5 ms, and four, six, and ten connections at bounds of 7.5 ms, 10 ms, and 15 ms respectively. However, since our connections have their start times randomly generated, more test cases could actually be supported if the interval between their start times satisfies:

$$j_n + (n-1)\,\delta t \geq \sum_{i=1}^{n} t_i \tag{7}$$

where j_n, jitter bound (same for all connections)
n, number of connections
Δt, interval between the connection's start time
t_i, service time (1.508 ms)

If Equation 7 is not satisfied for specific connections, scheduling contention will occur for all of their packets and no packet will meet its deadline. This explains why, from our utility plot of Figure 7, satisfied flows have an average utility of 1.0 while dissatisfied flows have the worst-case average utility for their expectation level.

7.2. Jitter-EDD with utility

It is more important for voice packets to meet their delay bounds than their jitter bounds. Packets that arrive too late are discarded and considered lost. Thus, if we apply utility to such connections, it is more appropriate to set their delay bounds to be deterministic (p = 1.0) and let their jitter bounds be statistical. We can be able to accommodate more connections if we do not observe the eligibility time of some packets and simply allow them to be transmitted even if they are not yet eligible for service. Although these packets will not be able to satisfy their jitter bounds, all packets will meet their delay bounds. Such a scheme will work only if the summation of the statistical requirements of the connections in conflict is less than or equal to 1.0 (Equation 8):

$$\sum_{i=1}^{n} p_i \leq 1.0 \tag{8}$$

Let us take as an example two connections A and B. We relax the jitter requirement of these connections by declaring them to be statistical rather than deterministic channels. Statistical connections declare a probability of a packet meeting its network requirement at the node. This value is equivalent to the probability p in our utility calculations and Equation 8 above, the target percentage of packets that satisfy the QoS demand. Assume that connections A and B both have a jitter probability of $p = 0.5$, which means that the flows will be happy even if only half of their packets meet the jitter bound. We can use a simple round robin implementation to have every other packet from each connection meet their network requirement.

Connections must have very lax jitter requirements in order to satisfy Equation 8. All the expectation levels of our test case for instance did not meet this requirement. In case of a conflict, a low expectation flow ($p = 0.80$) can only be paired with a flow with a jitter probability of $p = 0.20$ or less. It is rare for a connection to be satisfied even if only 20% of

its packets meet its requirements. Medium expectation flows and high expectations flows can be paired with a flow with $p = 0.10$ and $p = 0.01$, respectively. Such lenient requirements are almost equivalent to not having a jitter bound at all.

We can conclude that utilities have limited application to our constant bit rate voice sources using Jitter-EDD. Even if we model them as exponential on-off sources, the problem still arises when active connections do not meet Equations 7 and 8. A voice application with silence suppression can be modeled as an exponential on-off source, wherein a source transmits at a constant bit rate (CBR) during its talk time and does not send any packets in periods of silence. QoS requirements must be met during those talk time however, so we can treat our (CBR) traffic as the worst-case scenario.

Enhancements can potentially be made if we allow packet loss to be probabilistic as well. That is, QoS requirements are specified as both packet loss and jitter probabilities. The value-based utility framework however does not currently have a formulation to support two or more QoS requirements simultaneously. We believe that this is a non-trivial task that should be addressed in future work.

8. Summary and Conclusion

We showed promising results for Stop-and-Go with VBU but were unable to attain any significant improvements for Jitter-EDD. We can conclude that the Value-Based Utility model can easily be applied to some scheduling mechanisms. There are instances however when the framework may not be able to perform very well.

8.1. Summary of Results

Stop-and-Go guarantees a jitter bound of $2T$ to each connection, where T is the frame length. Table V summarizes the achieved jitter bounds for the three expectation levels at $T = 20$ ms. The table indicates that utilities can be applied to the Stop-and-Go algorithm in order to achieve a tighter jitter bound (T) without the need to maintain multiple frames. We have also shown that these results scale up to an output bandwidth of 2Mbps and 4Mbps.

Expectation Level	**Stop-and-Go**	**S&G with VBU**
Low (LEF)	30 ms	15 ms
Medium (MEF)	30 ms	20 ms
High (HEF)	40 ms	20 ms

Table V. Achieved Jitter Bound for Stop-and-Go and Stop-and-Go with VBU

A characteristic of a framing scheme such as Stop-and-Go that makes it amenable to the use of VBU is the aggregation of connections in one frame. Such aggregation is not present with Jitter-EDD, which manages connections on a per-flow basis. The aggregation of flows is important to the VBU model as it relies on shared resources in order to satisfy the more discerning users. The VBU framework has its limitations in a rigid environment where resources cannot be re-allocated easily or according to demand.

8.2. VBU and VoIP

We have shown that Value-Based Utility can be used to improve on the jitter performance of Stop-and-Go for constant bit rate traffic. Although such a model leads to more stringent requirements from the network, a realistic voice model would take into account that commercial VoIP systems today utilize silence suppression. This entails changing our constant bit rate sources into exponential on-off voice models. Any variations in queuing delay will be caused by connections going on and off aside from different ordering of packets in the input queue of each node. We hypothesize that exponential on-off sources will allow us more flexibility to apply VBU to Stop-and-Go in order to satisfy more connections. This leads to an interesting issue of voice traffic violating the (r, T)-smooth property required by S&G, leading to increased delay and even packet loss. We recommend modifying our constant bit rate voice model to an exponential on-off voice source such as the ones used in [13], [14], [15] for further study.

8.3. Conclusion

In this paper, we assert that jitter enhancements can be achieved by considering specific user requirements. A new framework at management, the Value-Based Utility model, was applied to the Stop-and-Go algorithm. Stop-and-Go achieves hard delay and jitter bounds by dividing time into fixed frames of length T. We compared both algorithms through simulations using a single-hop network. We have shown that Stop-and-Go with Utility provides tighter jitter bounds than that of Stop-and-Go alone. Our results warrant extending the research, which we leave for further study, to more complex scenarios such as a multi-hop network and realistic traffic sources.

We can note that the Jitter-EDD and Stop-and-Go scheduling mechanisms fits with two emerging service models to meet the demand for QoS, namely the Integrated Services (IS) model and the Differentiated Services (DS) model [16]. The IS model requires per-flow state on each router in order to offer guaranteed service aside from best-effort service. The Jitter-EDD service discipline conforms to this model as it relies on individual flow information in order to provide delay and jitter bounds. The DS model, on the other hand, aggregates flows by defining various service classes according to delay, throughput, and loss rate requirements. Stop-and-Go can readily be incorporated in the DS model as the scheduler for the low-delay service class.

REFERENCES

1. L. Copeland. Packet-switched vs. circuit-switched networks, March 2000. http://www.computerworld .com/networkingtopics/networking/story/0,10801,41904,00.html.
2. J. Pedrasa and C. Festin. *An enhanced framing strategy for jitter management*. In Proceedings of IEEE Region Ten Conference, November 2005.
3. J. Pedrasa and C. Festin. *Value-Based Utility for Jitter Management*. National ECE Conference. December 2006, Quezon City, Philippines.
4. Utility definition, July 2007. http://en.wikipedia.org/wiki/Utility.
5. C. Festin and S. Sorensen. *Utility-based buffer management for networks*. Lecture Notes in Computer Science, **3420**: 518-526 (2005). As proceedings of IEEE ICN 2005, April 2005, Reunion Island.
6. C. Festin. Utility-Based Local Buffer Management and Scheduling for Networks. PhD thesis, University College London, September 2002.
7. S. Golestani. *A stop and go queuing framework for congestion management*. In Proceedings of ACM SIGCOMM, September 1990.

8. H. Zhang. *Service disciplines for guaranteed performance service in packet-switching network*. Proceedings of the IEEE, **83,10**:1374-1396, (October 1995).
9. D. Verma, H. Zhang, and D. Ferrari. *Delay jitter control for real-time communication in a packet switching network*. In Proceedings of Tricomm '91, April 1991.
10. D. Ferrari and D. Verma. *A scheme for real-time channel establishment in wide-area networks*. IEEE Journal on Selected Areas in Communications, April 1990.
11. C. Liu and J. Layland. *Scheduling algorithms for multi-programming in hard, real-time environment*. Journal of ACM, January 1973.
12. R. Brown. Calendar queues: *A fast O(1) priority queue implementation for the simulation event set problem*. Communications of the ACM, October 1988.
13. A. Watson and A. Sasse. *Measuring perceived quality of speech and video in multimedia conferencing applications*. In Proceedings of ACM Multimedia '98, September 1998.
14. M. Karam and F. Tobagi. *Analysis of the delay and jitter of voice over traffic over the internet*. In Proceedings of the Infocom, 2001.
15. A. Markopoulou, F. Tobagi, and M. Karam. *Assessment of VoIP quality over Internet backbones*. In Proceedings of IEEE Infocom, June 2002.
16. X. Xiao and L. Ni. *Internet QoS: A big picture*. IEEE Network, March/April 1999.

