Using Optimization to Achieve Efficient Quality of Service in Voice over IP Networks

Michael Todd Gardner*, Victor S. Frost**, and David W. Petr**
**Information and Telecommunications Technology Center
Department of Electrical Engineering & Computer Science
University of Kansas, Lawrence, KS 66045,
E-mail: frost@eecs.ku.edu, petr@eecs.ku.edu
*Federal Aviation Administration, ACE-474, E-mail: todd.gardner@faa.gov

Abstract

For Internet Telephony to be a viable alternative to the Public Switch Telephone Network (PSTN), efficient and high quality communications are required. This paper proposes an optimization algorithm that selects parameters like coding scheme, packet loss bound, and maximum link utilization level in a Voice over IP (VoIP) network. The goal is to deliver guaranteed Quality of Service (for voice) while maximizing the number of users served. A VoIP architecture is also discussed that could use optimization algorithms to dynamically provision VoIP networks.

1. Introduction

As the Internet becomes a true multi-service medium, Voice-over-IP (VoIP), particularly in the form of Internet telephony, is gaining in importance. Increasing revenue, by maximizing the number of calls, is the primary reason to emphasize the efficient use of bandwidth. With an increasing number of high bandwidth applications (video on demand) competing for Internet bandwidth, how do we maximize the number of calls while maintaining high Quality of Service (QoS)?

There are several methods used to design VoIP networks that attempt to maintain the quality of voice across a network. Trial and error may be used. This may include setting up a "test" network using live traffic on a temporary basis then attempting to modify the network to provide for the necessary QoS. This is expensive and may or may not determine the optimal network configuration. Another approach is a "rule of thumb" approach. An example may be "voice delay must not exceed 300 ms". This approach may cause the network to be over-provisioned and used inefficiently.

Other methods are more analytical in nature. A designer may use the E-Model [1][2] in static design. In this approach, the designer feeds the current network design into the E-Model (delay, coder, etc...). The E-Model then returns the resultant predicted voice quality. The problem with this approach is that if the network characteristics change, the network design is no longer valid.

What is the most efficient way to deliver VoIP? To answer this question, we looked at how a VoIP system is organized and how transmission parameters are set in VoIP systems. Next we looked at the tradeoffs that affect voice quality, bandwidth use, and call volume. In [13], it was noted that there is a delay/buffer loss tradeoff that affects voice quality. Further, [13] also defined the relationship between buffer loss, delay, and voice quality as MOS(loss, delay). There is also a voice coder bit rate/voice quality tradeoff.

If information about the network, like delay and loss rates, is available, an optimization routine could make decisions about the coding scheme and routing that would maximize the amount of calls that could be placed and still “guarantee” a minimum level of voice quality. Section 2 of this paper describes a VoIP architecture that could be deployed using the optimization concepts presented in this paper to maximize efficiency in the VoIP network.

Section 3 proposes an optimization algorithm that uses the computations in the E-Model to select VoIP network parameters necessary to provision the network. The E-Model is a computational model developed by the European Telecommunications Standards Institute (ETSI) and standardized by the International Telecommunication Union (ITU) and the Telecommunications Industries Association (TIA) that uses voice and network transmission parameters to predict voice quality [1][3].

Section 4 describes three tests that were used to verify and validate the optimization algorithm. Section 5 concludes the paper.

2. VOIP System Optimization

This section describes an architecture that could be used to implement an E-Model optimization algorithm. In a simple VoIP network, there is a sender (encoder/ packetizer), access router (possibly combined with the packetizer), access link, and network router. On the receiving end, there is the network router, access link, access router, and receiver (de-packetizer/decoder). The sender and receiver are charged with choosing the coding scheme and parameters. The access router is responsible for routing and implementing queuing priority by tagging the packets.

The problem with this approach is that the network is constant changing and the statically assigned parameters may not be optimal. For example, if a given link has high delay and high loss, G.711 may be the only voice coding scheme that will work. But if the same link had low levels of loss and low delay, a more efficient scheme like G.723.1 may be adequate, allowing many more calls.

There are also problems related to setting the parameters associated with playout buffer based on a single tuning. In [13], it was shown that on low delay links, a single tuning was effective in controlling loss. But on high delay links with slowly varying delay and little overhead for the playout buffer, an adaptive playout buffer was necessary and effective. But, on other high delay links, it failed due to poor tuning of the parameters and the inability to accurately predict delay [13].

Figure 1 shows a VoIP connection that includes a VoIP broker that has the following functionality:

- Keeps a call database and their current parameters.
- Collects requests for call connects and disconnects.
- Collects delay, loss, and routing information from packets that have traversed the network.
- Calculates and distributes the optimum network configuration, including packet tagging information, coding parameters, and call acceptances/denials.

Ongoing research concerning VoIP could be extended to support this architecture. Session Initiation Protocol (SIP) along with Telephony Routing over IP (TRIP) have established that the SIP architecture can be used to support routing via TRIP[14]. There are studies that have been completed to look at the scalability issues surrounding TRIP [14]. In addition, the Session Description Protocol (SDP) provides a method to transmit coding parameters across the VoIP network.

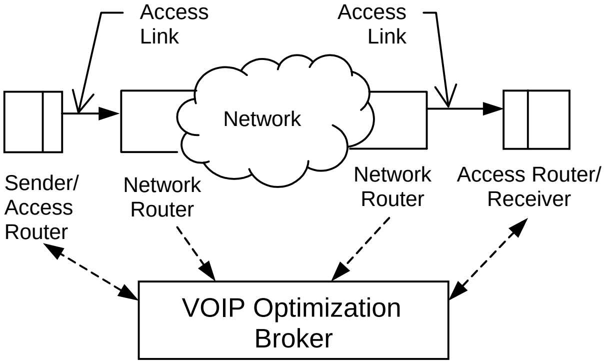

Figure 1. VoIP Connection with Optimization Enhancements

Scalability could be an issue with the architecture that is proposed in this paper. However, if SIP is the underlying architecture, it is logical for the optimization algorithm to run on the same server that is running the SIP Location Server (LS). Simulation studies are needed to verify that the optimization algorithm, described in Section 3 would scale well on the SIP/TRIP architecture.

3. Optimization Based on the E-Model

Voice quality is often measured by subjective opinion [4]. The traditional measurement for voice quality in telecommunications is the Mean Opinion Score (MOS). The MOS level 4.0, which is considered to be “good” quality of speech, has traditionally been considered “Toll Quality”. This was the quality that could be expected for a connection in the United States public switched telephone network (PSTN).

The E-model, which is based on the premise that “Psychological factors on the psychological scale are additive”, is used to gauge the quality of voice when several impairments are present [6]. These impairments include packet loss, coding scheme, delay, and echo. Comparing the MOS scale and E-model, as shown in Table 1, provides a reference for acceptability [3].

Table 1. E-Model vs. MOS Rating System [3]

User Satisfaction	E-model 'R'	MOS
Very Satisfied	90	4.3
Satisfied	80	4.0
Some Users Dissatisfied	70	3.6
Many Users Dissatisfied	60	3.1
Nearly All Users Dissatisfied	50	2.6
Not Recommended	0	1.0

3.1 E-Model Description

Using the E-Model, *R* is the parameter that represents voice quality. *R* is defined in [1][2][3][6] as:

$$R = Ro - Is - Id - Ie + A \quad (1)$$

Ro is the basic signal-to-noise ratio [1]. *Ro* is derived from send and receive loudness ratings, circuit noise, and room noise. *Is* represents impairments associated with voice signals, like incorrect loudness levels, quantization noise, and incorrect sidetone levels [1]. *Id* is the impairment associated with delay, including end-to-end delay and increased echo impairment due to delay [1]. *Ie* represents impairments associated with specific equipment [1], including coding schemes and packet loss levels. Finally, *A* is based on the "advantage of access" [1]. An example is a satellite phone in an area with no other access. For complete details about the E-model, the reader is referred to [1], [2], and [3].

Typical sources of delay in VoIP systems include encoding/packetization delay, switching and queuing delay, serialization delay, propagation delay, decoding delay and dejitter buffer. The E-Model impairment due to delay is the Delay Impairment factor (*Id*). *Id* is defined as [1]:

$$Id = Idte + Idle + Idd \quad (2)$$

Idte is impairment caused by talker echo. *Idle* is the impairment due to listener echo. *Idd* is the impairment due to absolute delay. The equation that describes *Idd* is [1]:

$$Idd = 25\left\{\left(1+X^6\right)^{1/6} - 3\left(1+\left[\frac{X}{3}\right]^6\right)^{1/6} + 2\right\} \quad (3)$$

$$X = \frac{\ln\left(\frac{Ta}{100}\right)}{\ln(2)}$$

Ta is the one way delay in an echo free environment in milliseconds. If *Ta* < 100 ms, *Idd* is assumed to be 0. Fig. 2 (which is an implementation of the E-Model [1]) shows the impact that delay has on voice quality.

Ie is the impairment that the E-Model uses to account for the degradation in the original signal due to speech coding schemes. ITU G.711, G.729, and G.723.1 compression standards use packet loss concealment (PLC) methods to deal with packet loss. Testing has been conducted on various coding methods (including varying levels of packet loss) to determine the amount of the impairment caused by packet loss using different coders. Table 2 shows the impairment values, *Ie* for three different coders as a function of packet loss [7].

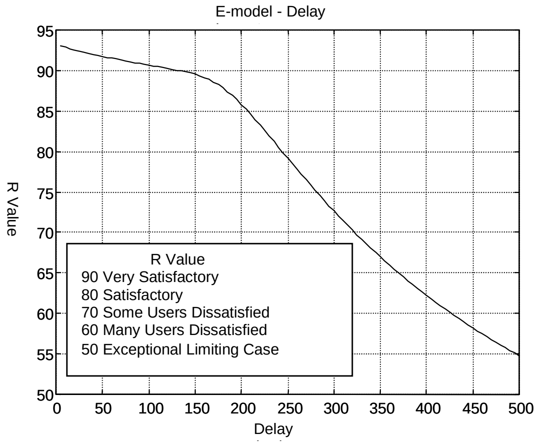

Fig. 2. E-model 'R' value as a function of delay.

Table 2. Coder Impairment with Packet Loss [7]

Packet Loss %	G.711 with PLC Random Packet Loss (*Ie*)	G.729A + VAD (*Ie*)	G.723.1+ VAD (6.3 kbits/s) (*Ie*)
0	0	11	15
0.5		11	15
1	5	15	19
1.5		17	22
2	7	19	24
3	10	23	27
4		26	32
5	15		
7	20		
8		36	41
10	25		
15	35		
16		49	55
20	45		

3.2 The E-Model Optimization Algorithm

The goal of the optimization algorithm is to place as many calls over a VoIP link as possible without the quality of voice degrading past a minimum quality level. Stated classically, using E-Model quality measures:

Maximize: Number of calls over link
Subject to: R(coding scheme, loss, delay, link bandwidth) >= 70

As mentioned in Section 1, several tradeoffs exist that effect voice quality. There is the loss/delay tradeoff as well as a voice coder bit rate/voice quality tradeoff.

To explore these relationships, we looked at the relevant inputs to the E-Model. These include: *T, Ta, Tr,* and *Ie. T* is the mean one way delay of the echo path, *Ta* is the absolute delay in echo free conditions, and *Tr* is the round trip delay in a 4-wire loop [1]. Since we assume that the echo cancellers are effectively employed, we can say that *T = Ta = (1/2)Tr*.

Since *Ie* is generally based on the type of coder and amount of packet loss, it is easy to see that if the packet loss is due to congestion, then *Ie* is related to delay. To further explore this relationship, the variables Packet Loss % (*PL*), Link Utilization (ρ), and Coder Type are introduced. Equation (4) shows the *Code Delay,* which applies to high link speeds (link speed >> coder rate) [3].

$$\text{Code Delay(ms)} = (N+1)\ \text{frame length(ms)} + \text{look-ahead(ms)} \qquad (4)$$

N is the number of frames per packet. The look-ahead delay is a code specific amount of time that the code must look forward prior to coding the current samples.

To address the delay caused by queuing, the M/M/1 queuing model was used to establish variable delay on a link as a function of utilization and packet loss. Other queuing models can easily be substituted. Packet loss is assumed to be the point on the tail of the delay distribution where packets are simply dropped. The following set of equations is used to relate packet loss to delay. In (5), *S(Td)* is the probability distribution of total delay of the M/M/1 queue, μ is the service rate, and ρ is the link utilization [11]. Equation (6) is derived from (5). *Td* represents delay for a given packet loss, utilization, and service rate. *PL%* is the packet loss expressed as the numerical value (example 3% is .03). The relationship between *S(Td)* and *PL%* is *PL% = 1-S(Td).*

$$S(Td) = 1 - e^{-\mu(1-\rho)Td} \qquad (5)$$

$$Td = \frac{\ln(PL\%)}{-\mu(1-\rho)} \qquad (6)$$

$$T = \text{Hop Count}*Td + \text{Code Delay} + \text{Propagation Delay} + \text{Misc. Delay} \qquad (7)$$

The total one way delay is represented as *T* in (7). For the cases considered here, we assumed a 5 hop system. We also assumed that the propagation delay is 25 ms and miscellaneous delays caused by switching and echo cancellers are 6 ms.

The approach used to set up this optimization is to define a "set" that includes the combinations of the items that are varied. For example, in Case 1, the set includes the coders. For Case 2, the set is the combination of coders and packet loss percentages. This allows the algorithm to search the "universe" of possibilities and define its working set based on the constraints. The optimization problem is set up as follows.

- The *Set* of system configurations is defined.
- The parameters are calculated. This includes all E-Model parameters with fixed inputs and variable inputs based on the combinations in the *Set*.
- Establish the objective, which is to maximize the number of calls on a link.
- Set the first constraint, (R>=70). Set the second constraint (Sum of *Portion = 1.0).*

This yields the following AMPL [10] optimization algorithm (shown for Case 1).

```
Set:        CODE;
Parameters:  T{CODE}, E-Model Parameters,
Ie{CODE}, MTU {CODE}, Fixed Data Delay, Calculation
of E-Model Parameters,
Variables:  Portion{CODE}, Code_Feas {CODE} binary
Objective:
            Maximize Calls: sum {i in CODE}:
            (Code_Feas[i]*portion[i]*LinkBW*util/Rate[i])
Subject to:
             Minimum R {i in CODE} : Ro – (Id[i] + Is[i] +
            Ie[i])*Code_Feas[i] + A   >= 70;
Subject to:
            Total Code: sum {i in CODE}portion [i] = 1;
```

Fig. 3. AMPL Simplified Optimization Algorithm (shown for Case 1)

Since the number of calls will be maximized with one of the *Set* combinations, this problem can be considered an "assignment" type optimization. The variable *Portion* is used to assign the calls to a particular combination in the *Set*. Strict assignment would require *Portion* to be a binary integer (1 or 0). To avoid this non-linearity, we relaxed the integer requirement and allowed the program to make fractional assignments. Despite allowing fractional values, the assignment theorem [12] ensures that the solution produced will always exhibit an assignment of 1 or 0 for every *Portion* variable.

Code_Feas is a binary variable that penalizes coders that do not meet the constraints and limits the working set to $R \geq 70$. If the algorithm is having difficulty meeting the minimum *R* constraint, the only variable that it has at its disposal is the variable *Code_Feas*. The algorithm simply switches *Code_Feas* for that coder from a 1 to a 0, which eliminates the impairment portion of the equation and satisfies the constraint. By setting *Code_Feas* to zero for that coder, it eliminates it from participating in the objective, which removes that coder from its working set.

This is an implementation of a penalty function as described in [10]. One pitfall of using the hard (binary) penalty function is that it is non-linear.

Being non-linear, the algorithm found the first coder that met the constraints and did not look for others that could produce a better objective function. This problem was solved by setting all *Code_Feas* variables to “1” during program initialization. For the algorithm to meet the $R \geq 70$ constraint, it MUST look at all *Code_Feas* variables and reverse them if necessary.

4. Optimization Results

Three cases were considered to verify and validate the E-Model optimization. The cases considered were:

- Optimization 1: Find the optimal voice coder given link bandwidth, packet loss level, and link utilization.
- Optimization 2: Find the optimal voice coder and the optimal packet loss level given link bandwidth and link utilization.
- Optimization 3: Find the optimal voice coder and the optimal link utilization level given link bandwidth and packet loss level.

The results of these cases are briefly described below. [8] contains a more complete description of the test results.

4.1 Case 1 - Optimizing for Coder Selection

Case 1 was run for two different link speeds: 256 kbps and 1.544 Mbps. When the link speed was 256 kbps, the objective returned was 4.3 calls. G.729A was the coder chosen by the optimization. For Case 1 with a link speed of 1.544 Mbps, G.723.1 was selected as the optimum coder by the optimization routine. The objective function found 37.1 calls. The results of this case are not surprising, as G.723.1 is a more efficient but lower quality of voice. In both cases, the algorithm selected the coder that produced the most calls, without allowing the voice quality to fall below $R = 70$.

4.2 Case 2 – Optimizing for Coder and Packet Loss

Test 2 required additional analysis because not all of the packet loss percentages were available (see Table 2). We found a polynomial fit for each coder.

For the 256 kbps link speed, the objective returned was 4.3. G.729A with packet loss of 2% was the combination chosen. For a link speed of 1.544 Mbps, the objective was 37.1. G.723.1 with packet loss of 1% was the combination chosen. The objective value(s) returned in Case 2 are always the same as Case 1 because packet loss only affects voice quality, not the number of calls.

The algorithm for Case 2 has a modification that allows it to favor higher packet losses by adding a small benefit to the objective function. This benefit consists of the packet loss level multiplied by the *portion* assigned to that coder. This was done to bias the optimal solution toward a minimum requirement for a link (rather than the most stringent requirements). This modification was demonstrated when the G.729 coder with 2% packet loss was chosen over the G.729 coder with 1% packet loss.

4.3 Case 3 – Optimizing for Coder and Link Utilization

When Case 3 was run with a link bandwidth of 256 kbps. The objective returned was 5.6. G.729A running with 60% link utilization was the combination chosen. For Case 3 with a link speed of 1.544 Mbps, the objective returned was 66.8. G.723.1 running with 90% link utilization was the combination chosen. Looking at Fig. 4, we can see that all three coders were in a feasible range until link utilization reached approximately 85%.

In Fig. 4, *R* remains constant for all coders until a point where *R* declines rapidly. This is important because it suggest that there is optimal link utilization where the system can be operated prior to the *R* value decline. The sudden decrease in *R* is due to the fact that as utilization values approach 1.0, the delay becomes unbounded, which negatively affects the *R* value.

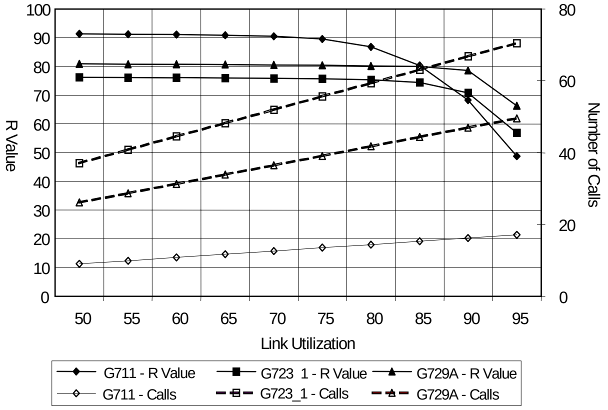

Fig. 4. Data Collected from Case 3 (1.544 Mbps) - R Value vs. Packet Loss for Three Coders

5. Conclusions

This research began with the goal of finding a way to efficiently transport Voice over IP across the Internet. To accomplish this goal, an optimization algorithm was developed and tested. This algorithm utilized the E-Model to provide predicted measurements of voice quality based on a variety of parameters. A VoIP architecture was proposed that could utilize an E-Model optimization algorithm by providing a broker that can be a central point for computations and communications with the senders and receivers. By developing the optimization algorithm and testing it on three problems, this research has demonstrated the following points.

The E-Model optimization algorithm worked correctly therefore verifying the algorithm. All three problems maximized the total number of calls on the sample voice over IP network while maintaining a voice quality level of R = 70. The three problems also demonstrated the inefficiencies associated with poor parameter choices. This validates the algorithm as viable.

Logical results from the optimization can be proved and limits of their use can be determined. For example, all three cases found that G.729A is a better coder with lower bandwidth links and G.723.1 is a better coder with higher bandwidth links. This is because the quality of speech is generally higher with G.729A. But G.729A uses more bandwidth than G.723.1. Algorithm sensitivity to changes in network parameters can also be determined. In Case 2, both G.729A and G.723.1 were sensitive to changes in packet loss, but G.711 was not as sensitive. In Case 3, voice quality was not sensitive to changes in the link load until the link load grew above approximately 80%.

There is a large body of research that remains to be completed in this area. First, the algorithm that was designed for this paper was relatively simple. If it were to be deployed across a wide area network with multiple senders and receivers the following needs to occur:

- The optimization algorithm needs to be extended to include multiple hop links. This is necessary because as a voice packet passes through the Internet, it may share links with many different calls.
- The algorithm needs to be extended to be able to handle existing calls (that cannot change their coding parameters).
- The algorithm needs to be tested with more variables being allowed to float (like coder selection, packet loss level, and link utilization).
- More work needs to be done with the architecture of a VoIP broker. This may entail extending SIP and/or TRIP to include the optimization algorithm.

Although an optimization algorithm was designed and tested successfully, much more work is necessary to implement this algorithm across a VoIP network.

6. References

[1] ITU-T Recommendation G.107, “The E-model, a Computational Model for use in Transmission Planning”, Pre-published Recommendation, May, 2000

[2] ITU-T Recommendation G.108, “Application of the E-model: A planning guide”, September, 1999

[3] Telecommunications Industries Association (TIA), “Voice Quality Recommendations for IP Telephony”, EIA/TIA/TSB-116.

[4] ITU-T Recommendation P.80, “Methods for Subjective Determination of Transmission Quality”, March, 1993

[5] Olivier Hersent, David Gurle, Jean-Pierre Petit, “IP Telephony, Packet-Based Multimedia Communications Systems”, Addison Wesley, 2000.

[6] “Transmission and Multiplexing; Speech Communications Quality from Mouth to Ear for 3, 1kHz Handset Telephony across Networks”, ETSI Technical Report ETR 250, July, 1996.

[7] ITU-T Recommendation G.113, "Transmission impairments due to speech processing", Pre-Published Recommendation, February, 2001.

[8] Michael Todd Gardner, "Analyzing Mission Critical Voice over IP Networks", Masters Thesis, University of Kansas, 2001.

[9] Robert Fourer, David M. Gay, Brian W. Kernighan, “AMPL A Modeling Language for Mathematical Programming”, The Scientific Press Series, 1993

[10] Robert Fourer, David M. Gay, Brian W. Kernighan, “AMPL A Modeling Language for Mathematical Programming with AMPL Plus Student Edition for Microsoft Windows”, Duxbury Press, 1997

[11] Leornard Kleinrock, “Queueing Systems, Volume 1: Theory”, John Wiley & Sons, 1975.

[12] David G. Luenberger, "Linear and Nonlinear Programming, Second Edition", Addison-Wesley, May 1989.

[13] Athina P. Markopoulou, Fouad A. Tobagi, Mansour J. Karam, “Assessment of VoIP Quality over Internet Backbones”, IEEE Infocom, 2002

[14] Matthew C. Schlesener, "Performance Evaluation of Telephony Routing over IP (TRIP)", Masters Thesis, University of Kansas, 2002.