5. BEST EFFORT JITTER BUFFER MANAGEMENT ALGORITHM

5.1 BASICS OF JITTER BUFFER MANAGEMENT ALGORITHM

In this chapter we mainly focus on the fundamentals of jitter buffer, the proposed algorithm to enhance the performance of voice communication and the E model calculation which is used as a benchmark to judge the performance of voice communication.

5.1.1 Jitter

Jitter is defined as a difference in the delay of received packets. The sender transmits the packets in a continuous stream and spaces them uniformly apart. The jitter buffer places the incoming audio packets in the buffer for a short period of time and then plays them out. The delay between data packets vary as shown in Figure 5.2 and do not remain constant. The reason behind the variability of delay is the congestion in the network, improper queuing, and configuration errors. This variation in delay causes problems for the quality of audio played at the receiving end. Playback experiences a gaps while waiting for the variable delayed packets to arrive.

Figure 5.1 Contribution of Jitter Buffer in the VoIP Network
5.1.2 Play out Delay Buffer

Whenever the router receives a data voice, it must be able to compensate for the jitter that it detects. The play out delay buffer mechanism handles this function. Play out delay is the difference in time, between the time a voice packet reaches the jitter buffer and the time the packet is played out to the codec. The play out delay buffer must buffer these packets and then play them out in a steady stream to the processor as shown in fig 5.3. The system then converts the voice packets back into an analog audio signal. The play out delay buffer is also referred to as the dejitter buffer.
5.1.3 Dropped Packets

If the jitter is very high then the packets are received out of range. The packets which are received, out-of-range are discarded and the dropped packets degrades the quality of audio as shown in the Figure 5.4. For losses as small as one to 4 packets, the system addresses the problem of inaudibility through the Packet Loss Concealment (PLC) algorithm discussed in the previous chapter.

Figure 5.3 Play out Delay Buffer

Figure 5.4(a) Dropped Packets
5.1.4 Jitter Buffer for Voice Over IP

The packet delivery in IP network is based on the best-effort and thus, depends on the amount of traffic in network. If the packets arrive at the destination late, they may arrive out of order, or they may get lost. The packets that carry speech segment are time sensitive and correction of these packets (that is retransmission) cannot be done easily.

![Fig 5.4(b) Dropped Packets](image)

5.1.5 Jitter Buffer Operation

![Fig 5.5 Jitter Buffer Operation](image)

The jitter buffer receives voice packets from the IP network at irregular intervals. Occasionally, the voice packets are out of sequence. The packets are made to stay in the jitter buffer for period of time and then the packets are
transmitted at uniformly spaced intervals to the decoder. Algorithms in the network device determine the size and behaviour of the jitter buffer based on user configuration and current network jitter conditions. The system uses this information to increase the number of correctly received packets and minimize the time of delay.

The size of the jitter buffer and the amount of delay is configurable by the user with the play out-delay command. Proper configuration is critical. If the packets are made to stay in the jitter buffer for very short period of time, then the variations in delay may cause the buffer to become empty and creates gaps in speech. However, the packets that arrives when the buffer is full are dropped, and causes gaps in speech.

To improve voice quality, the speech gaps are hidden by effective packet loss concealment algorithm that synthesizes the packets to replace those that were lost or not received in time. Depending on the contiguous duration of the gaps, the missing voice frames are enlarged using the past frames (usually the last frame) and future frame followed by silence if the condition persists (for more than 30 to 50 ms).

The various jitter problems are discussed as follows:

- **Receive Delay**: The play out delay for compensating the jitter plus the average delay expected after the frame is available for playing out.

- **Gap Fill With**: These fields refer to the amount of concealment—or packet synthesizing—that took place in this call, to replace the voice packets that were lost or not received in time.

- **Lost Packets**: The actual number of packets that were lost; that is, the packets *not* received at the egress gateway. This is detected using the sequence number field in the RTP packets.

- **Early Packets**: The actual number of packets that arrived earlier than the expected time. These early packets are used in the network jitter calculation.
• **Late Packets**: The packets that arrive after the scheduled play out time. These late packets are dropped and the information in the late packets are discarded.

### 5.1.6 Classes Of Jitter Buffers

Delay and packet loss can significantly affect the perceived quality of voice transmitted over packet networks. Packets travelling in the network, suffer from delay variation, arrive out of order or even get lost. To compensate for delay variation, de-jitter buffers are used at the receive side of packet based systems. The jitter buffers are used to restore the proper order of packets so that the “slower” packets will also arrive [47].

There are two major classes of jitter buffers: static jitter buffers and dynamic jitter buffers.

**Static Jitter Buffer**

The static jitter buffers have fixed size packets and the packets leaving the jitter buffer maintain a constant delay (the delay is the time the packet is generated until the time when the packet is consumed), whereas the adaptive jitter buffers maintain variable size packets and variable delay.

![Figure 5.6 Static Jitter Buffer](image-url)
When you configure the play out-delay mode fixed option, you can specify the nominal delay value, which is the amount of play out delay applied at the beginning of a call by the jitter buffer. This will be the maximum size of the jitter buffer throughout the call connection.

If the play out delay mode is selected as fixed option, then the delay value is maintained throughout the jitter buffer. This is maintained as the maximum size of the jitter buffer throughout the call.

**Dynamic Jitter Buffer**

The play out-delay command allows you to select a jitter buffer mode (static or dynamic) and specify certain values that are used by the gateway to adjust the size of the jitter buffer. During a VoIP call, the algorithms will read the time stamps value in the RTP packet headers to decide the amount of delay that the jitter buffer will apply to an average packet; that is, as though there is no jitter at all in the network. This is called the average delay.

**Adaptive Jitter Buffer** algorithm is used for resolving network problems. It can work with arbitrary external coders and audio codec's, with arbitrary size of network packets, with external and internal PLC (packet loss concealment) algorithms. If internal PLC is used then it can dramatically
improve voice quality in networks with packets loss in respect to standards algorithms [47].

Different algorithms to estimate play out buffer have been studied in literature. These algorithms are grouped as 4 categories [48] as follows:

- Algorithms that establish the play out delay based on a continuous estimation of the network parameters.
- Statistics based algorithms.
- Algorithms that are maximizing the user satisfaction.
- Algorithms that are using various heuristics and monitor certain parameters (e.g. late packets fraction, buffer occupancy, etc.).

5.2 DRAWBACK OF THE EXISTING JITTER BUFFER MANAGEMENT ALGORITHM

The peculiarities of the VoIP traffic in the network and delays caused by it are described in [49,50]. Thus, for instance, a situation is described in [51] in which an elementary Jitter Buffer Management Algorithm is ineffective as it does not store any data on the prehistory of the packet flow, particularly the number of the last VoIP packet input to the codec. If the buffer is empty at the moment of the arrival of the next packet to the JB (Jitter Buffer), then the these packets are received and placed into the last block of the jitter buffer at the largest distance from the codec. This is not always correct, and lower voice transmission quality may be the result.

A graphic representation of this situation is depicted in Figure 5.9. The packet number 11 is the last one sent to the codec in time. The successive packets 12, 14, and 15, as well as the packet number 7, are delayed in the network, and the buffer has been emptied. The packet number 7 arrives first among the delayed packets. Since the system does not store data on the last packet accepted by the codec (the packet number 11 in our case), packet number 7 will be placed in the buffer (Figure. 5.9). As a result, packets 12, 14, and 15, which arrive afterwards, will be removed as there is no space
available for them in the Jitter Buffer (behind packet number 7). It would be more effective to not accept packet number 7 into the buffer, as the time of the reproduction of the audio information contained in it has passed and it will be rejected by the codec, but rather to accept packets 12, 14, and 15. A rather large number of such situations may arise during a communication session, and they affect the voice communication’s quality significantly.

Figure 5.8 The first case Specifies the operation of fixed JBMA

Figure 5.9 The second case illustrating the violation of the sequence order Packets.

Another situation presented in Figure 5.10 is also caused by possible violations of the sequence order of the VoIP packets transferred via the network. In this case, packet number 15 arrives ahead of packages 12 and 13, which was sent earlier. Despite the availability of free blocks in the Jitter Buffer, packet number 15 will be removed, considering that the available free blocks may only be occupied by packets 12 and 13, respectively.
5.3 PROPOSED JITER BUFFER MANAGEMENT ALGORITHMS FOR VOICE COMMUNICATION OVER THE IPTELEPHONY

The first elementary modification of the algorithm (we denote it as DB1) implies storing the number of the last packet that was input to the codec in time. In this case, the number of each packet arriving from the network is, first of all, compared to the number of the last packet that was input to the codec in time. If the number of the packet received from the network turns out to be less than the number of the last packet that was input to the codec and reproduced, then the received packet will be immediately removed by the DB1 and won't enter the jitter buffer. The packet generation time in the transmitting router, are included in the headers of the packets of this protocol in addition to the index number of the packet being transferred. The receiving and transmitting sides negotiate the type of codec and its parameters for the current session in the process of establishing the communication.

Algorithm 5.1: Best Effort Jitter Buffer Management Algorithm

//Algorithm : Network Based Jitter Buffer Management Algorithm
//This algorithms explains the working at the rear end of the jitter buffer queue

Input : Packets from the Network
Output : Packets for the Codec

REAR END:
if(Packet_Number_Newly_Received_Packet>Packet_Number_Last_packet_JitterBuffer then //indicates early packets
    if Jitter_Buffer is not full() then// checks if the buffer is not full
        Jitter_Buffer(rear+1) = Newly_received_Packet() //early packets has to be stored in the buffer
    else
        //else
Circular_Buffer(rear+1) = Newly_received_Packet

if network spike is true then

Decrease play out delay //the interval of transmission

else

Reset play out delay()

end if

end if

else

Drop Packet(); //late packets are discarded

Repeat the process

end if

Algorithm 5.2: Codec Side Jitter Buffer Management Algorithm

//Algorithm : Codec side Jitter Buffer Management Algorithm

// This algorithm explains the working at the front end of the jitter buffer queue

Input: Packets from the Jitter Buffer

Output: Packets for the codec

FRONT END:

IF Packet_Number_first_Packet_JitterBuffer>Packet_Number_first_CircularBuffer then

Move the first _ Circular_Buffer to Codec

DEQUEUE CB

ELSE

MOVE THE first_Packet_JitterBuffer to CODEC

DEQUEUE JB

END IF
The interval of the generation of the IP packets with audio information is 4ms and this remains constant during the whole session. N is the number of blocks in the Jitter Buffer and it is taken as 10 in this experimental setup. The packet number of the arriving packet(A) from the network is compared with the last received packet(R) in the codec. (1) If A>R then A is an early packet and so it is enqueued into the JB. (2) If A<R then it is an late packet and so it is dropped. (3) If the buffer is full and A>R then A, which is an early packet is enqueued into the CB (Circular Buffer).

The CB (Circular Buffer) helps to capture some more early packets and thus the probability of losing the early packets is reduced. If both the JB (Jitter Buffer) is full then the network condition is checked. If the mode is found to be spike then play out interval is reduced by \(\frac{1}{2}\).i.e normally the packets are played for every 4ms and during spiky mode the delay is reduced to 2ms. This helps to handle the bulk traffic generated and the dropping of packets due to high loaded network can be reduced. This change in the play out interval has to be synchronized with the codec.

When the network condition comes back to the normal condition the play out delay is reset to 4ms. This algorithm allows the packets into Jitter Buffer only if the packets contain audio information which is yet to be played in the codec. Thus unnecessary storing of late packets into buffer and discarding them during the play out time at the codec is avoided. The proposed algorithm makes efficient use of the buffer for handling only the needed packets.

The two main advantages of proposed algorithm is

1. The probability of packets drop due to early packets is reduced.
2. The order of packet is maintained which has an impact on the clarity.
5.4. COMPARATIVE ANALYSIS OF THE PERFORMANCE OF THE EXISTING AND PROPOSED JITTER BUFFER MANAGEMENT ALGORITHM

5.4.1 Experiment

To compare the performances of different Jitter Buffer Algorithms, it is necessary to use the available voice communication quality criteria and define new ones. Methods of the subjective analysis of the voice communication quality based on the analysis of the users’ assessments are presented in recommendation [15]. The suggested methods and criteria can’t be used in the present paper, as a simulation model rather than the actual system is used for the comparative analysis. A calculation algorithm for the so-called E-model for the assessment of the data communication quality is presented in [15]. The algorithm presented in [50] takes into account the room noise, distortions introduced by the codecs, and accidental packet losses as well.

This algorithm is also not suitable for the purposes of the present research, since the problem is to compare the performances of Jitter Buffer rather than the performances of codecs. The quality of any Buffer Management Algorithm is defined by the number of packets transferred to the codec for reproduction in time. We denote the number of packets transferred to the receiver codec in time during one communication session as PC (Packet Count). It is therefore assumed that all the packets generated at the transmitting side reach the far end receiving router.

This assumption does not affect the performance of the proposed algorithm, since all the algorithms are compared under similar network loadings; second, the lack of packet losses is the most stressful mode of the Jitter Buffer operation. With the above conditions taken into account, PC (Packet Count) may be determined as follows: \( PC = PG - PJB + PCB \), where PG (Packet generated) is the total number of voice IP packets generated by the transmitting side during the communication session, PJB is the number of packets removed by the Jitter Buffer during the same session, and PCB is
the number of packets that were dropped by the Circular Buffer at the buffer pool because these packets did not arrive the codec in time. Since the number of packets generated during one session was constant, the quality of the Jitter Buffer operation may also be assessed based on the total number of removed packets by jitter Buffer and Circular Buffer.

\[ PC = PG - PJB + PCB \]

and

\[ PC\% = (PC/PG) * 100 \text{ in the percentage form} \]

Where

PC = Number of packets transferred to the receiver codec in time

PG = Total number of voice IP packets generated by the transmitting side

PJB = Number of packets removed by the Jitter Buffer

PCB = Number of packets removed by the Circular Buffer

PD = PG – PC where PD is the packet drop

The algorithm will be more effective if the count of received packet is higher. The model of the data communication network are used to analyze of the performance of buffer management algorithm, is similar to the one described in [50]. Each generated voice IP packet may take one of three routes to reach the destination: fast, medium, or slow. Each route of the model simulates passing through four routers. The processing time of a packet at each router is simulated by the corresponding random value having an exponential distribution.

The probabilities of the packet distribution by routes for a weakly loaded network were as follows: 0.97, 0.025, and 0.005. In other words, the probability of the packet taking the fast route was 0.97; for the medium route, it was 0.025; and, for the slow route, it was 0.005. Given the medium network loading, the distribution of the packets by the routes was assumed to be the following: 0.60, 0.33, and 0.07.
Finally, given the strong network loading, the following distribution of the packets by routes was used: 0.40, 0.35, and 0.25. The network may be considered overloaded in this case. On average, it took 10ms, 44ms, and 144ms for the packets to go through the fast, medium, and slow routes, respectively. The result of Adaptive jitter buffer algorithm proposed by Broitman [50] is shown as A_JB which places the packet s in the particular block and the performance of the fixed jitter buffer is shown as F_JB. The performance of the proposed best effort jitter is shown as BF_JF.

The performance of the A_JB, F_JB and BF_JF jitter buffer algorithms are compared in the weak, medium and strong network loading and the comparison graphs are discussed below.

5.4.2 Comparison of Jitter Buffer Performance in Weak Network Loading

The percentage of packets transferred to the receiver codec under weak network loading are analyzed for the various buffer sizes ranging from 1 to 10 is presented in figure 5.11. The performance of the proposed algorithm BE_JB (Best Effort Jitter Buffer) is better than the existing algorithm with the difference around 1% for the buffer sizes starting with N=1. The performance fixed, adaptive and best effort algorithms are practically the same for the buffer sizes starting with N=5. The results of adaptive and proposed Best Effort Buffer management algorithm are the same for the buffer size starting with N=2.
The Packet Drop% under weak network loading is presented in Figure 5.10. The performance of fixed and adaptive jitter buffer algorithms (F_JB, A_JB) are nearly the same for buffer sizes starting with 6 (N = 6). The Best Effort algorithm are found to be more effective for N < 4; however, the difference is only 3–10% in comparison with A_JB (Adaptive Jitter buffer).

5.4.3 Comparison of Jitter Buffer Performance in Medium Network Loading

The performance of the buffer Management Algorithm in medium network loading is presented in Figure 5.11. The advantage of Best Effort algorithm in comparison with adaptive algorithms is, the percentage of removed packets being lowered than the maximum allowed level (15%) for N>3.
5.4.4 Comparison of Jitter Buffer Performance in Strong Network Loading

The performance of the proposed algorithm under the strong network loading condition shows improvement when compared with the Adaptive buffer algorithm. In this case the fixed jitter buffer is unable to handle the bulk network traffic and so almost most of the packets are dropped by the jitter buffer (%PD>90). The proposed algorithm provides an improvement in comparison with fixed and adaptive algorithms. The proposed algorithm provides an acceptable result even in the over loaded network and it is achieved at N=6.

The packet drop percentage using proposed Best Effort algorithm remains stable after N=6 whereas in adaptive buffer the acceptable packet drop percent is reached only when N=10. In the strong network the proposed algorithm reduces the packed drop at the expense of reduced play out delay gap. When bulk amount of packets are generated in the network, more
number of early packets will reach the buffer and it will be dropped in the existing algorithm.

\[ \text{Figure 5.12 Performance Comparison of proposed algorithm in strong network Loading} \]

The proposed algorithm solves this issue with an additional circular buffer of size 10 and reducing the play out interval if the network traffic mode=Spike. Thus the rate of packets sent to the codec increase which in turn reduces the probability of early packet drop. This play out buffer time is synchronized with the codec.
5.5 CONCLUSION

Packet drop cannot be avoided in real time VOIP application, mainly in the bursty network. The major problem with the network is when bulk amount packets are generated, more packets are dropped. The quality of the voice will be highly affected if the packet drop percent is more than the acceptable limit of 15%. The proposed algorithms reduces the packet drop ratio in the weak, medium and strong network traffic when compared with the existing algorithms.

Even with Best Effort jitter buffer management algorithm, a certain number of packets will arrive after their scheduled play out time. These lately arriving packets are dropped both in the circular buffer and in the jitter buffer. This loss of packets over the network creates disturbances in the quality of voice at the hearing side. To recover the lost information, the packet loss concealment algorithms are used at the codec. The Packet Loss Concealment (PLC) algorithm for the codec G.711 is discussed in the next chapter.