Voice Conferencing over IP Networks

Paxton J. Smith

Department of Electrical & Computer Engineering
McGill University
Montreal, Canada

January 2002

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Engineering.

2002/01/18

Abstract

Traditional telephone conferencing has been accomplished by way of a centralized conference bridge. An Internet Protocol (IP)-based conference bridge is subject to speech distortions and substantial computational demands due to the tandem arrangement of high compression speech codecs. Decentralized architectures avoid the speech distortions and delay, but lack strong control and have a key dependence on silence suppression for endpoint scalability. One solution is to use centralized speaker selection and forwarding, and decentralized decoding and mixing. This approach eliminates the problem of tandem encodings and maintains tight control, thereby improving the speech quality and scalability of the conference. This thesis considers design options and solutions for this model, and evaluates performance through live conferences with real conferees. Conferees found the speaker selection of the new conference model to be transparent, and strongly preferred the resulting speech quality to that of a centralized IP-based conference bridge.

Sommaire

Les conférences téléphoniques traditionnelles sont rendues possibles par l'utilisation de ponts centralisés. Un pont de téléconférence IP ("Internet Protocol") est soumis à une distorsion des signaux de la parole et une demande de calculs importante due à la configuration en "tandem" des méthodes de codage de la parole à haut taux de compression. Les systèmes décentralisés échappent aux problèmes de distorsions et de délais, mais ne permettent pas un bon contrôle et dépendent de la suppression des silences pour une bonne extensibilité. Une des solutions est de centraliser la sélection de l'interlocuteur et de l'acheminement et de décentraliser le décodage et le mélange des signaux de parole. Cette approche élimine le problème de l'encodage en "tandem" et conserve un contrôle précis permettant une meilleure qualité des signaux de parole et une bonne extensibilité des conférences. Ce mémoire examine les différentes options de design et évalue leurs caractéristiques par le biais de vraies conférences avec plusieurs interlocuteurs. Les participants ont grandement préféré la qualité des signaux de parole provenant du système proposé par rapport à ceux provenant d'un pont IP traditionnel et ils ont apprécié la transparence de l'algorithme de sélection de l'interlocuteur du modèle proposé.

Acknowledgments

I greatly appreciate the financial support given by the Wireless Speech and Data Processing group of Nortel Networks. I would like to express my sincere thanks to my advisor and co-advisor, Professors Maier Blostein and Peter Kabal, for their guidance and support during my graduate studies at McGill University. I am also thankful to Mr. Rafi Rabipour for providing the idea and motivation for the research, as well as for his insight and enthusiasm throughout the project. The involvement of all three of these individuals has greatly benefited both the research and myself.

My colleagues of the Telecommunications and Signal Processing Laboratory are owed several thanks for their technical assistance and friendship. I am especially thankful to Tarun for his proofreading efforts, not to mention Mark and Alex. Benoît's French translation of the abstract is much appreciated. To all of those whose time was sacrificed repeatedly by the conferencing experiments, thank you. You are: Chris, Jen, Joe, Wesley, C. C., Issac, Peter, and Andre Stedenko.

My respect and regards goes out to the RAT development team at University College London for developing, sharing, and supporting such fine conferencing tools and libraries.

I am grateful to my family for all of their love and support over these last two years, as well as my good friends (you know who you are) for their patience and understanding.

Contents

1 Introduction **1**
1.1 PSTN Conferencing 2
1.2 VoIP Conference Bridge 3
1.3 Problems with VoIP Conference Bridges 5
1.4 Previous Related Work 7
1.4.1 Origins of Tandem-Free Operation Conferencing 7
1.4.2 New Approaches to Centralized Voice Conferencing 8
1.5 Thesis Contribution 9
1.6 Thesis Organization 11

2 Principles of VoIP Conferencing **12**
2.1 Introduction to IP 13
2.1.1 Delivery Mechanisms 13
2.1.2 Transport Mechanisms 14
2.2 Real Time Protocol 14
2.2.1 Real Time Control Protocol 16
2.2.2 Intermediate RTP Systems—Mixers and Translators 16
2.2.3 RTP Header Compression 17
2.3 VoIP Transmission Systems 17
2.3.1 Speech Codecs 19
2.3.2 Voice Activity Detection 21
2.3.3 Silence Suppression/DTX 23
2.3.4 Comfort Noise Generation 23
2.3.5 Jitter Buffers/Playout Scheduling 24

2.3.6 Packet Loss Concealment . . . 27
2.3.7 Audio Clock Skew Compensation . . . 28
2.3.8 Audio I/O . . . 29
2.3.9 Echo Cancellation . . . 30
2.4 IP Telephony Architecture . . . 31
2.4.1 VoIP Protocol Suite Evolution . . . 31
2.4.2 Media Gateway Controllers . . . 33
2.4.3 Media Gateways . . . 33
2.4.4 Signalling Gateways . . . 34
2.5 Centralized Conferencing . . . 35
2.5.1 Audio Bridge Fundamentals . . . 36
2.5.2 Speaker Selection Bridging . . . 37
2.5.3 Signal Level Equalization and Fading . . . 41
2.5.4 Audio Mixing Options . . . 41
2.5.5 Commercial Conference Bridges . . . 43
2.5.6 End System Mixing—A Centralized Conferencing Variant . . . 44
2.6 Decentralized Conferencing . . . 45
2.6.1 Full Mesh Conferencing . . . 45
2.6.2 Multicast Conferencing . . . 46
2.6.3 Drawbacks of the Models . . . 47
2.7 Conversational Dynamics of Voice Conferencing . . . 48
2.7.1 Two-Way Conversations . . . 49
2.7.2 *N*-Way Conversations . . . 51
2.8 Chapter Summary . . . 52

3 Design Solutions in VoIP Conferencing 53
3.1 Tandem-Free Operation Conferencing . . . 54
3.1.1 TFO Bridge Overview . . . 54
3.1.2 TFO Endpoint Overview . . . 56
3.1.3 TFO Advantages . . . 57
3.1.4 TFO Disadvantages . . . 58
3.2 TFB VAD Design . . . 59
3.2.1 Requirements . . . 59

3.2.2 TFO VAD Options 60
3.2.3 TFB VAD Algorithm 62
3.3 TFB Speaker Selection Design 63
3.3.1 Motivation for a New Algorithm 63
3.3.2 Requirements 64
3.3.3 Tandem-Free Speaker Selection Algorithm 66
3.3.4 Other Algorithms 70
3.3.5 Performance Criteria 71
3.3.6 Simulations 72
3.3.7 Algorithm Comparison 73
3.4 RTP/RTCP Design 77
3.4.1 Payload Requirements 77
3.4.2 VAD Field Use Scenarios 79
3.4.3 Payload Format Options 80
3.4.4 Payload Format Comparison 83
3.5 Chapter Summary 84

4 System Evaluations and Results **85**
4.1 Overview of Conferencing Models and Parameters 86
4.2 Complexity Analysis 87
4.2.1 Method of Analysis 87
4.2.2 Bridge Processing Requirements 88
4.2.3 Endpoint Processing Requirements 89
4.2.4 System Bandwidth Utilization 90
4.3 Voice Quality Evaluations 91
4.3.1 Conferencing Facility 92
4.3.2 Experimental Scope 93
4.3.3 Experimental Method 94
4.3.4 Data Collection 95
4.3.5 Evaluation Results 96
4.3.6 Summary of Evaluations 98

5 Conclusion **99**
5.1 Summary and Discussion of Results 99
5.2 Future Work 103
5.2.1 TFSS Algorithm Enhancements 103
5.2.2 Stream and TFSS State Synchronization 104
5.2.3 Support for Silence Suppression/DTX 104
5.2.4 TFB Output Rate Control 104
5.2.5 Improved TFB VAD 105
5.2.6 Gain Equalization 105
5.2.7 Experimental Design 105

A Conferencing Facility **107**
A.1 System Implementation 108
A.1.1 RAT—the TFO Endpoint 108
A.1.2 RTP Connections and Translations 110
A.1.3 TFB Implementation 112
A.1.4 Conference Hardware 113
A.2 RAT End-to-End Delay 114
A.2.1 Total Round-Trip Delay 115
A.2.2 Network and Jitter Buffer Delay 116
A.2.3 Audio I/O Delay 116
A.2.4 Discussion of Delay Measurements 117
A.2.5 Summary of System Delay 119

B Processing Delay and Complexity **120**

List of Figures

1.1 A conference bridge forming a unique sum for each conferee 3
1.2 A generic VoIP conference bridge 4

2.1 A generic VoIP transmission system 18
2.2 Example operation of a playout buffer adapting per-talkspurt 25
2.3 Simplified VoIP protocol stack 31
2.4 A generic IP telephony network 33
2.5 Media connections in a centralized VoIP conference 36
2.6 A generic audio bridge 38
2.7 Endpoint Mixing 45
2.8 Decentralized conferencing models 46
2.9 Media connections in a full-mesh VoIP conference 47
2.10 Four-state model of a two-way conversation 50

3.1 TFO conferencing model 55
3.2 Components and data flow of the TFB 57
3.3 TFB VAD options 60
3.4 State distribution of a four-way conversation 64
3.5 TFSS algorithm state transition diagram 69
3.6 Selected speech using FCFS 75
3.7 Selected speech using LT 75
3.8 Selected speech using NT 76
3.9 Selected speech using TFSS 76
3.10 A basic TFO data frame 79
3.11 TFO payload using stacked format 80

3.12 TFO payload using blocked format 81
3.13 A TFO data frame with F-bit 82

4.1 Complexity reduction factor of the TFO bridge algorithm 88

A.1 Round trip audio delay on a multitasking workstation during web-browsing 117

List of Tables

2.1 Commonly used speech codecs 20
2.2 Summary of front-end clipping performance 22
2.3 Summary of mid-speech burst clipping performance 27
2.4 Summary of commercial VoIP conference bridges 44

3.1 Average durations of talkspurts and pauses after speaker selection 73
3.2 Speech clipping due to speaker selection 74

4.1 Order of computational complexity for conferencing models 87
4.2 Worst-case CPU utilization for a conference endpoint 89
4.3 Order of worst-case bandwidth for conferencing models 90
4.4 Worst-case bandwidth of conferencing models 91
4.5 Summary of supported conferencing scenarios 93
4.6 Schedule of conferencing system comparisons 94
4.7 Summary of conferee's opinions and system rankings 98

A.1 Workstation configurations 114
A.2 Summary of one-way delay between two RATs 118

B.1 G.711 and G.729A processing delay and CPU utilization 120
B.2 TFSS and mixing processing delay and CPU utilization 120

List of Terms

ACELP	Algebraic-Code-Excited Linear-Prediction
AGC	Automatic Gain Control
ATM	Asynchronous Transfer Mode
AVP	RTP Profile for Audio and Video Conferences with Minimum Control
BEC	Back-End Clipping
CELP	Code-Excited Linear-Prediction
CNG	Comfort Noise Generation
cRTP	RTP Header Compression
DSP	Digital Signal Processor
DSVD	Digital Simultaneous Voice and Data
DTX	Discontinuous Transmission
FCFS	First-Come-First-Served
FEC	Front-End Clipping
IETF	Internet Engineering Task Force
IP	Internet Protocol
ITU-T	International Telecommunications Union
LAN	Local Area Network
LP	Linear Prediction
LPC	Linear-Predictive Coding
LT	Loudest Talker
MBONE	Multicast Backbone
MEGACO	Media Gateway Control
MG	Media Gateway
MGC	Media Gateway Controller
MGCP	Media Gateway Control Protocol

MOS	Mean Opinion Score
MSC	Mid-speech Burst Clipping
NT	Northern Telecom
NTP	Network Time Protocol
PBX	Private Branch Exchange
PCM	Pulse Code Modulation
PLC	Packet Loss Concealment
POTS	Plain Old Telephone Service
PSTN	Public Switched Telephone Network
QoS	Quality of Service
RAT	Robust Audio Tool
RTP	Real Time Protocol
RTCP	Real Time Control Protocol
RTSP	Real Time Streaming Protocol
RTT	Round-Trip Time
SCP	Signal Control Point
SDP	Session Description Protocol
SG	Signalling Gateway
SI	Speaker/Interrupter
SID	Silence Insertion Descriptor
SIP	Session Initiation Protocol
SNR	Signal-to-Noise Ratio
SS7	Signalling System Number 7
STP	Signal Transfer Point
TCP	Transmission Control Protocol
TDM	Time Division Multiplexing
TFB	Tandem-Free Operation Bridge
TFO	Tandem-Free Operation
TFSS	Tandem-Free Speaker Selection
UDP	User Datagram Protocol
VAD	Voice Activity Detector
VoIP	Voice over IP
WAN	Wide Area Network

Chapter 1

Introduction

Conferencing capability is an integral part of a carrier-grade voice communication network. This service facilitates group collaboration in geographically dispersed organizations, such as corporate, military, and government institutions. For the enterprise user, modern conferencing has been enriched with video images and data sharing, while residential and mobile telephony customers are typically offered a three-way calling service. Speech remains at the core of conferencing, and connectivity continues to be provided by digital conference bridges residing in the Public Switched Telephone Network (PSTN) network.

Current trends indicate that the circuit-switched telephone networks will migrate to packet-based Internet Protocol (IP) networks in the short-term. In addition to providing new services, Voice over IP (VoIP) networks must preserve the rich feature set and quality of service of the trusted PSTN. Telecommunication equipment vendors have chosen to meet these goals by "re-wrapping" or co-operating their legacy solutions with IP ones. This approach positions logical services such as toll-free calling and third-party billing for rapid deployment, but creates problems for media-based services such as voice conferencing.

Conventional VoIP conference bridges yield a reduction in speech quality relative to their PSTN predecessors. Faced with an explosive increase in multimedia conferencing users over the next few years, service providers will demand an improved approach. Fortunately, the connectionless nature of IP networks can be leveraged to increase the speech quality above the current norm. This is accomplished by a rearrangement of processing roles from the traditional model. With this rearrangement comes various new problems that must be resolved.

1.1 PSTN Conferencing

Conferencing in the PSTN has been historically accomplished with a conference bridge. Conference terminals, or endpoints, connect to the bridge in star configuration. The purpose of the bridge is to sum up the input signals of each conferee and subsequently supply the summed signal(s) back to each conferee.

Bridges are "four-wire" devices, meaning that the transmit and receive ports are separated and a seemingly full-duplex connection is actually two half-duplex connections. The first conference bridges were analog, and connectivity between conferees was accomplished with simple analog circuitry. Here, the conferees' signals were superimposed, forming a *conference sum* [1, 2]. When digital transmission lines were deployed in the late 1970s the same analog bridges were still used, albeit A/D conversion was performed prior to mixing, followed by D/A prior to transmission [1]. Digital bridges began to replace their analog counterparts in the early 1980s.

Modern digital bridges are typically implemented in hardware (i.e., with Digital Signal Processors (DSPs)), and "hang off" a Central Office (CO) switch or a Private Branch Exchange (PBX) [3]. A bridge may also be part of a network services complex, which hosts various modules for providing complimentary voice (and data) services to customers, such as recorded greetings and data conferencing [4]. The bridge is used for pre-arranged "meet-me" conferences in which conferees "dial-in" to the bridge (or the bridge dials-out) at a predetermined time.

Conference bridges are connected to their respective terminals by way of the usual call processing procedures, which are provided by the network switches. Once a voice circuit is "nailed-up" between the bridge and each conference terminal, the bridge may send and receive voice signals on its various ports. Hence, a generic conference bridge consists of a (speech) data interface to network switching system and an audio bridge.

The network interface handles all necessary A/D and D/A conversions, Time Division Multiplexing (TDM) and demultiplexing, and the μ-law Pulse Code Modulation (PCM) speech codecs, thereby decoupling the conferencing processing from the details of terminal connections[1]. For example, a conference bridge may operate on 2 ms (16 samples) frames, requiring the parallelization and decoding of the appropriate TDM-PCM channel.

[1] A PSTN data network interface is described by the BORCHST (Battery feed, Overvoltage protection, Ringing, Supervision, Codec, Hybrid, and Testing) circuitry, which consists of what is necessary to interface to a particular type of connection [1]. The circuitry differs for each type of connection (i.e., line or trunk).

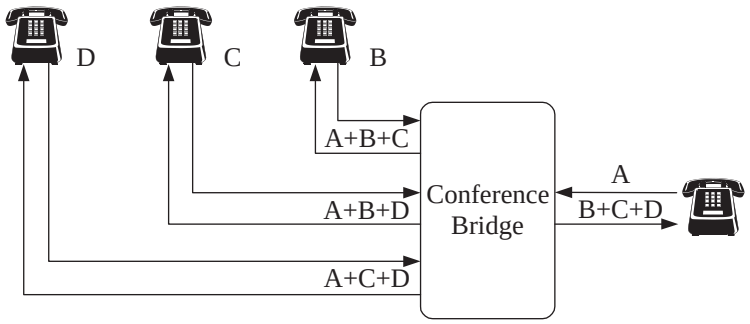


Fig. 1.1 A conference bridge forming a unique sum for each conferee [1].

1.2 VoIP Conference Bridge

The centralized approach is a convenient choice for VoIP conferencing since the design principles and caveats of conference bridges are well-known. Both traditional (e.g., Nortel Networks, Lucent) and new (e.g., Convedia, IP Unity, Radvision) telecommunication equipment vendors have released IP-based conference bridges within the past few years. These products are generally referred to as conference/audio servers [5, 6], mixers [7, 8], or multipoint processors [9, 10].

Current trends in VoIP view conferencing as one of many applications which are built upon powerful DSP-based media processing platforms. Such *media servers* are used to provide application-specific media processing functions such as speech recognition, text-to-speech, voice recording/storage, and conferencing [11]. This strategy concentrates all of the processing power in one centralized location. Call servers, or *softswitches*, direct voice streams to said media servers on a per-need basis, thereby keeping the call processing and service logic light[2].

The speech processing functions of a VoIP conference bridge are essentially unchanged from the PSTN case. Thus, much of the core technology developed for the legacy bridges is reused, although wrapped inside of an IP interface [11]. For example, voice activity detection, speaker selection, gain equalization, and mixing are common functions found on PSTN conference bridges [4, 12], and can be applied as usual providing that the received packet streams are first converted to (linear) PCM. Thus, the fundamental difference between a legacy and VoIP-based bridge is in the network interface.

In VoIP, compressed speech signals are encapsulated in packets and prepended with a

[2]In the context of this thesis, only the conferencing function of such media servers are considered.

Real Time Protocol (RTP) [7] header primarily containing a timestamp, sequence number, and source identifier. Packet networks notoriously inject variable transit delay into the voice streams, requiring the bridge receiver to absorb said variable delay by means of a jitter buffer [13–16]. Jitter buffers are allocated per-stream such that a continuous stream of samples is produced and fed into the mixing module. In this way, the core audio bridge may produce a conference sum for each conferee as usual. Each composite signal is encoded and then encapsulated in RTP packets before being returned to the conferees. An example of a generic VoIP conference bridge is shown in Fig. 1.2.

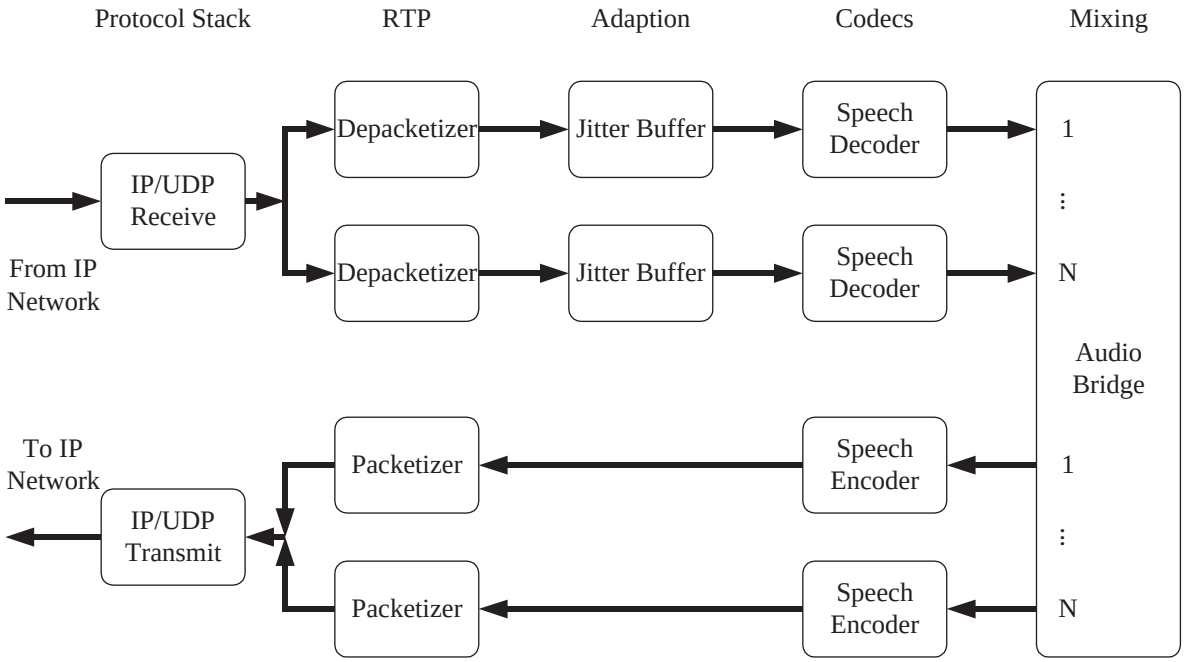


Fig. 1.2 A generic VoIP conference bridge. The speech/audio processing is essentially the same as the legacy PCM bridges. The major changes are the RTP packetizer, jitter buffers, and speech codecs.

Another major difference between the PSTN and VoIP network lies in the speech coding technology. The PSTN is inflexible insofar as the format and transmission of speech signals, i.e., the TDM-PCM format is mandatory. In contrast, the payload of an RTP packet can be used to transport speech in many different formats [17]. In carrier-class VoIP, toll quality is typically provided by waveform coders (G.711, G.726) or low-bitrate coders based on Algebraic-Code-Excited Linear-Prediction (ACELP) (e.g.,G.729(A), G.723.1) [18–20].

Ideally, the VoIP conference bridge is equipped to handle *all* codecs, meaning that

conferees do not have to share a common codec in order to participate in a conference. This is known as *endpoint heterogeneity*. The advantage of this inherent flexibility is that conference endpoints may choose a speech codec based on their desired speech quality or the available bandwidth of their connection. Again, the function of the conference processing algorithm is independent of speech codecs since the network interface is responsible for translating between the given codec and linear PCM.

1.3 Problems with VoIP Conference Bridges

Several problems exist when the centralized conference bridge is migrated to a VoIP network. For instance, the use of high compression codecs such as G.729(A) leads to speech quality degradations and scalability problems. Specifically, speech quality is reduced by the tandem connection of speech codecs, i.e., the serial connection of speech codecs used to compress and decompress the signal. Additionally, the input signals to each coder must be buffered and processed, which increases the overall end-to-end delay. The tandeming limits the scalability of the bridge since a large amount of computational power is required, mainly due to the operation of the speech codecs. The remainder of this section expands on these problems:

1. **Tandem Encoding:** A centralized conference bridge must decode all received signals prior to mixing, and then re-encode the conference sum prior to transmission. Successive encodings degrade the speech quality. When the conference processing load is distributed over multiple bridges, *three* encodings (two tandem encodings) are required, since partial sums are exchanged between bridges.

2. **Encoding a Multi-Speaker Signal:** Code-Excited Linear-Prediction (CELP) technology uses Linear-Predictive Coding (LPC) at the core of the compression algorithm. With LPC, speech is modelled as the output of a linear system representing the vocal tract and an excitation source [21]. The technique is optimized for encoding a single voice, yet a conference sum is often composed of *multiple* voices. This said, the final

encoding phase of the bridge is subject to distortion[3].

3. **Jitter Buffer Delay:** A conference bridge uses a per-stream jitter buffer to smooth out the variable delay between successive inbound packets. The length of the jitter buffer usually adapts to the short-term jitter statistics. For example, in a carrier-grade network this delay may range between 20–100 ms, where in the public Internet it may range from 100–2000 ms [14, 18]. This increase in delay may easily yield a *total* end-to-end delay greater than the 150 ms target for "good" service, or worse, exceed the 400 ms threshold that represents a noticeable reduction in interactivity [24]. Moreover, the jitter buffer adds an increased memory requirement to the bridge that is proportional to the product of the number of conferees, the time duration of the jitter buffer, and the bitrate of the buffered speech data.

4. **Codec Processing Delay:** Speech codecs operate on a collection of samples, called *frames*, which are typically 10–30 ms in duration. Thus, there is always a one-frame delay before processing can begin. Some coders use a *look-ahead* in order to enhance the processing of the current frame which adds additional delay equal to the length of the look-ahead. In addition, it is generally assumed that the duration of the encode-decode operation is equal to the length of the analysis frame, in order to make efficient use of processor resources [24]. However, the total contribution from look-ahead and codec processing is counted once for each codec between the source and destination. Hence, the conference bridge codec processing increases the end-to-end delay by the sum of the look-ahead time and codec processing time. For example, G.729 has a frame duration of 10 ms and a look-ahead of 5 ms, yielding an additional delay of 15 ms at the conference bridge.

5. **Speech Coding and Mixing Complexity:** The main processing components of a centralized conference bridge are the speech codecs and the mixer. In the worst-

[3]Specifically, the Linear Prediction (LP) analysis filter will fail to accurately represent the higher frequency formants, since more speech energy is concentrated in the lower portion of the spectrum—this leads to a low-passed, "muffled" sound. For example, in G.729, ten LP coefficients are used, which is enough to accurately represent the first four formants (8 coefficients) and the finer, peaky details elsewhere in the spectrum (2 coefficients) [21, 22]. However, the presence of two or more superimposed formant structures with potentially different centre frequencies leads to more LP coefficients allocated to the lower frequencies. Further, pitch prediction is compromised due to the overlapping pitch periods from the individual voices. Synthesis of speech coded in this manner leads to decreased intelligibility and a reduced ability to distinguish between different talkers [23].

case the bridge must execute N decode, mix, and encode operations per-frame. If speaker selection or silence suppression is used, the number of mixing operations may be reduced, since only $M+1$ sums need be formed (see Section 2.5.4). Although the number of mix-encode operations is fixed, the N decode operations prohibit scalability to large conferences.

1.4 Previous Related Work

The root of the aforementioned problems is the tandem encoding of speech signals at the conference bridge. Tandem operation of speech codecs is a well-known problem, and is not limited to conference bridges. For example, interworking between different wireless networks, as well as mobile-to-mobile calls also result in tandem encodings since coded speech is transcoded between different formats by intermediary systems. For wireless mobile-to-mobile calls, research has focussed on a tandem-bypass mechanism, known as Tandem-Free Operation (TFO) [25], which is used to forward codec data between the mobiles instead of decoding and re-encoding at the base stations [26]. Of course, this only works if both mobiles use the same speech codec. To deal with transcoding between two different codecs, bitstream mappings have been proposed [27].

1.4.1 Origins of Tandem-Free Operation Conferencing

Similar work has been performed in the context of conferencing. In 1979, Forgie studied the human factors that contribute to the perceived quality of a conferencing system under a range of conditions and speech codecs [23]. In particular, Forgie found that the use of coded speech should *not* be used in conference bridges that sum the input signals due to severe distortion caused by the tandem operation of codecs, and the encoding of a multi-speaker signal. Forgie developed an alternative technique he called *signal selection*, where the "bridge" used voice-activated switching to select a primary speaker from N input streams. The primary speaker's compressed signal was re-routed back to the $N-1$ endpoints without undergoing the usual mixing and re-encoding process, thereby preserving speech quality[4].

[4]Forgie's conferencing system consisted of 10–12 regular telephones which were connected to a conference "bridge" through a private digital PCM network. The bridge *emulated* centralized conferencing in tandem operation by running the PCM signals through a codec before and after the summation (i.e., the original encoding was performed at the bridge, not at the endpoint). For the signal selection case, the speech of the primary speaker was run through a codec before being returned to the conferees.

The most successful speaker selection algorithm developed by Forgie was called Speaker/Interrupter (SI) conferencing. The algorithm monitored the speech activity of each received stream with a Voice Activity Detector (VAD), and assigned talking privileges based on the First-Come-First-Served (FCFS) criterion. SI was strictly a single-talker algorithm, since only one downstream channel existed between each conferee and the bridge. However, the current talker received the voice of the second-highest priority speaker, giving the system "interruption" capability. In other words, only the primary talker heard the secondary talker, while the rest of the conferees only heard the primary talker. Forgie also reported that a single-talker, level-based speaker selection algorithm fragmented the speech in an unnatural or rapid manner.

Interest in tandem-free conferencing systems was relatively quiet in the 1980s, presumably due to lack of network and terminal support for low-bitrate speech coding. Instead, the focus was on the design of suitable switched-loss and speaker selection algorithms for controlling noise and echo in PCM conference bridges [2, 4, 28–33]. However, the problem resurfaced in the 1990s when the use of speech coding became more widespread.

1.4.2 New Approaches to Centralized Voice Conferencing

To date, the tandem encoding problem for *packet* voice conference bridges has been addressed through industrial research. VoIP conferencing standards, from both the International Telecommunications Union (ITU-T) and Internet Engineering Task Force (IETF) exist [5, 9], yet fail to recognize the problem. A few solutions have been patented throughout the 1990s and are reviewed in the following sections. These ideas are based on either codec-domain processing or speaker (signal) selection and forwarding, the latter being an extension of Forgie's work in [23].

Bitstream Mapping (Champion)

In 1991, Champion [34] proposed a conferencing system specifically for military applications; the system was tied to a Sinusoidal Transform Coding (STC)-based speech coding scheme at 4.8 kbps. The bridge selected two current talkers (i.e., primary and secondary speakers) using the FCFS criterion. Speech activity was determined at the endpoint; the resultant VAD decision was transmitted to the bridge in each upstream packet. During periods of single-talk, the selected stream was broadcast back to the endpoints. However,

during periods of double-talk, the bridge transcoded each of the upstream channels to half-rate by mapping the coded parameters to a codebook with half the original resolution. The endpoint performed STC-domain mixing, followed by regular decoding. Meanwhile, the primary channel received the secondary channel at full-rate, and vice-versa.

Single-Talker Tandem By-Pass (Nahumi)

In 1995, Nahumi proposed a conference bridge which *forwarded* codec data during periods of single talk, but executed the normal decode-mix-encode procedure during multi-talk [35]. Speech activity was determined either by a codec-specific VAD which extracted features by way of a *partial* decode, or by a conventional VAD, following a full decode. Here, tandeming was not eliminated, but only *restricted* to less frequent periods of multi-talk.

Select-and-Forward (Simard and Rabipour)

The latest contribution to the problem of tandem operation in centralized VoIP conferencing may be viewed as a combination of the work of Forgie and Champion. Two inventors, Simard et. al. [36] and Rabipour [37] (both of Nortel Networks), have independently and simultaneously proposed a completely tandem-free conference bridge. Similar to Forgie's SI system, the bridge selects M talkers (usually a primary and a secondary talker) and forwards their streams back to the endpoints. As usual, the primary talker hears only the secondary talker, and vice versa. The endpoints are responsible for simultaneously receiving, decoding, and mixing the two selected streams. Similar to Champion's system, the parameters used to effect speaker selection, such as a VAD bit, or measure of signal power, are computed at the endpoint and included as additional fields in each upstream packet. This technique has the advantage over the formerly described systems since tandem encodings are avoided altogether, and the system is codec-independent.

1.5 Thesis Contribution

This thesis investigates the design options and performance of the conferencing model proposed by Simard and Rabipour[5]. The primary goal of the work was to evaluate the difference in speech quality between the conventional centralized and TFO conferencing

[5]As per-Rabipour, the model will be referred to as the Tandem-Free Operation (TFO) model herein.

arrangements, using a popular VoIP codec, G.729A [38]. This required that a flexible, real-time conferencing facility be built which could support or emulate a variety of conferencing scenarios. The first phase of the work concentrated on the selection and modification of suitable technologies on which meaningful experimentation could be performed. Since Simard and Rabipour present only a generic view of the TFO model, design solutions were required for the speaker selection and RTP modules. The second phase of the work concentrated on these issues.

In terms of speaker selection, Simard makes specific reference to an order-based algorithm in [12] intended for a three-way calling feature on high-end PSTN phones. Rabipour specifies no explicit algorithm, but suggests that events such as speech-onset, mid-speech, and speech-offset be used to effect speaker selection. Therefore, a novel speaker selection algorithm was developed, namely the Tandem-Free Speaker Selection (TFSS) algorithm, based on the Rabipour approach. Speaker activity is divided into six states, and talking privileges are assigned based on the average speech amplitude and a hysteresis. The states were derived in part from a well-cited AT&T algorithm described in [4]. TFSS combines the benefits of traditional FCFS and Loudest Talker (LT) selection algorithms in order to provide smoother switching between active talkers. With this improvement, the Tandem-Free Operation Bridge (TFB) minimizes the perceived speech clipping, and allows strong interactivity, making a two-talker system sound like an N-talker system.

Both Simard and Rabipour suggest that each endpoint append speech "features"—such as signal power or a VAD-bit—to every packet that is sent to the bridge. These features, or "TFO data", are then used to effect speaker selection. Rabipour outlines a generic "TFO packet", which includes the speaker identifier, timestamp, codec identifier, codec data, and the TFO data fields. Aside from the TFO data, these fields map directly to an RTP packet [7]. However, a new RTP payload type is required in order to carry both TFO and speech data within a single packet. This thesis suggests a framework for such a payload type.

In the end, a PC-based conferencing facility was implemented, consisting of a custom-built TFB and modified version of the Robust Audio Tool (RAT) [39]. At the core of the TFB was the new speaker selection algorithm; other algorithms were built-in, so as to facilitate comparisons. Emulations of conventional centralized and decentralized conferences could be configured through a combination of TFB and RAT settings. The performance of each model was measured subjectively through comparative tests carried out by actual

conferees.

1.6 Thesis Organization

The fundamentals of VoIP and conferencing are outlined in Chapter 2. First, the components and performance attributes of a VoIP transmission system are discussed, followed by an architectural view of carrier-grade IP telephony networks. This moves the thesis into the realm of conferencing. Additional details of centralized conference bridges are discussed, with emphasis on audio processing and speaker selection. It is then shown how conferencing can be achieved in a decentralized fashion. The chapter concludes by familiarizing the reader with the human conversational dynamics of teleconferencing systems. Chapter 3 enters the design phase of the thesis. First, the concept of Simard and Rabipour's TFO conferencing model is re-introduced. Design solutions are presented for the model's speech processing and RTP issues, resulting in a novel speaker selection algorithm and a new RTP payload type. The performance of the scheme is considered in Chapter 4, where the system is evaluated comparatively against conventional centralized and decentralized conferencing systems. The chapter concludes with a summary of conferees' opinions, collected during live conferences with the TFO conferencing system built for this thesis. Conclusions are presented in Chapter 5, along with suggestions for future work. Appendix A provides the implementation details of the conferencing platform including the selected endpoint, the RTP connection model, and the total end-to-end delay performance.

Chapter 2

Principles of VoIP Conferencing

As seen in Chapter 1, centralized VoIP conferencing is prone to various problems. Some of these problems are inherent to all VoIP systems, the most prevalent being due to speech compression, delay, and packet loss. VoIP transmission systems are equipped with particular technologies to help cope with these issues. These technologies work together to implement an end-to-end transmission system. However, in order to create services such as conferencing, the network must have intelligence which allows these transmission links to be established between end systems. As is customary in telephone systems, these signalling and control functions are separated from the underlying transport system. A conferencing application uses said transport and signalling systems as building blocks to establish connectivity between conferees. The connections may be provided by way of a conference bridge, i.e., a centralized conference, or directly between end systems, i.e., a decentralized conference. In the former case, audio mixing is performed at the conference bridge, while in the latter the mixing is performed by each endpoint. However, mixing multiple speech signals together presents its own set of problems, hence various techniques have been developed which aim to maximize the sound quality of the conference sum.

The chapter is laid out as follows. First, the reader is acquainted with the essential components of VoIP transmission systems. Second, the architecture of carrier-grade IP telephony networks are introduced, providing an overview of how connections are established. Third, further details of centralized conference bridges are provided, both legacy and IP-based. Then, alternatives to the centralized conference bridge are discussed. The chapter concludes by overviewing some interesting properties of conversational speech.

2.1 Introduction to IP

The Internet protocol suite is divided into several layers, based on the seven-layer Open Systems Interconnection (OSI) reference model. Layers 1–4 (Physical, Link, Network, Transport) are well-defined by standards, whereas layers 5–7 (Session, Presentation, Application) are generally "rolled" together into a single application. IP resides at layer 3, which is responsible for moving data between endpoints, regardless of physical location. One major benefit of VoIP is that new products may be developed on top of IP without concern for the underlying transmission medium.

IP is a connectionless protocol, meaning that an IP packet may be sent to another endpoint without first "nailing up" a connection, as in circuit switched networks. An IP host is identified by its IP address, which consists of both a network and host address. The network address identifies the subnetwork (or subnet), while the host address identifies the actual machine on said subnetwork. Routers use the destination network address portion to move packets between subnetworks, and ultimately to the target machine.

2.1.1 Delivery Mechanisms

IP networks offer three different modes of transmission: (1) unicast, (2) broadcast, and (3) multicast. Unicast is used for point-to-point communications between two hosts. Broadcast enables a host to send a packet to *all* other hosts on the immediate subnetwork (and adjacent subnetworks if the gateway allows it). Finally, multicast allows a host to send a packet to a *group* of hosts.

Multicasting is a one-to-many operation, requiring the sender to generate only one output stream for the entire multicast group. The network copies and forwards the packets at each fork in the multicast tree. Bandwidth is saved in this way because packets are not duplicated on any one link. This mode of distribution is convenient for large, wide area conferences, and webcasts. However, widespread support for multicast does not exist, hence the alternative is to use the so-called "multi-unicast" approach. Here, a host (or packet reflector) generates an output stream for each member of the group (i.e., $N - 1$ output streams).

2.1.2 Transport Mechanisms

IP is a connectionless delivery service. Hence, the Transport layer (optionally) ensures the reliable transport of IP packets between hosts. This layer carries source and destination port numbers which identify the process or application that each packet came from or is going to. The traffic generated by common Internet application protocols is characterized by the destination port number (e.g., HTTP uses port 80). The two common Transport Layer protocols are the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP).

In particular, TCP is used to provide reliable delivery of IP packets. TCP sets up a connection between two hosts before any data is exchanged—data integrity is ensured by way of packet acknowledgments and flow-control. For example, in the event of a missed acknowledgment, a retransmit occurs and the transmission rate is dynamically adjusted to accommodate adverse link conditions (e.g., loss due to temporary congestion). This behaviour results in unpredictable end-to-end delays which render TCP undesirable for the real-time transmission of voice data.

In contrast, UDP is a connectionless protocol which does not provide sequencing or acknowledgments. It is used when applications do not need the stringent reliability mechanisms imposed by TCP. Instead, an application may provide its own reliability mechanism or none at all. Such is the case for VoIP, where small amounts of packet loss are generally tolerated such that lower end-to-end delay may be achieved. For this reason, UDP is used for the transport of interactive VoIP *media* streams.

2.2 Real Time Protocol

The Real Time Protocol (RTP) [7] provides the mechanisms required for end-to-end delivery of real-time data such as voice, video, Dual-Tone Multi-Frequency (DTMF), comfort noise etc. Each RTP packet consists of a header and media payload; the packets are customarily transported over IP/UDP. The RTP header contains a timestamp, sequence number, Synchronization Source (SSRC) identifier, and payload type identification. This information allows the receiver to recreate the stream in the presence of variable delays, loss, and out-of-order packets.

RTP does not explicitly provide a mechanism to provide Quality of Service (QoS), but

instead relies on underlying and peer mechanisms to do so. It does *not* provide timely or guaranteed delivery of packets, and neither does it prevent misordering of packets, nor does it assume that mechanisms to effect such services exist. Instead, it is expected that receivers use the timestamp and sequence numbers to reconstruct the stream in order, and to detect packet loss, variable delay, and clock skew. The mechanisms to carry out these tasks are defined by the *application* and not by RTP. Often, the protocol is tightly coupled to the application (Layers 5–7), instead of the underlying protocol stack (Layers 1–4).

RTP is intentionally generic and relies on companion documents to define specific uses of the protocol. A *profile* specification broadly defines how a class of payloads (e.g., audio and video) are carried by RTP. The profile specifies the interpretation of the header fields, as well as media-centric attributes such as default payload durations and permissible sampling rates. A *payload* format addresses the nuances of how a particular payload is to be carried[1]. Most speech codecs operate under the RTP Profile for Audio and Video Conferences with Minimum Control (AVP) [17]. A few key points of RTP, and the AVP are summarized as follows:

- **RTP Session:** An RTP session is defined by the destination transport address (network address and port number) to which a media stream is sent; a session may consist of multiple inbound streams, and only one outbound stream. Each stream is uniquely identified by its SSRC, which is assigned by the respective senders. For example, each voice stream emitted from a conference bridge is carried in its own RTP session.

- **Timestamp:** The RTP timestamp is based on the *sampling* clock of the media source, representing the number of sampling instants which have elapsed relative to its initial value. For example, when transporting an 8 kHz speech signal via RTP, with a 20 ms payload, the timestamp increases by $8000 \times 0.02 = 160$ "ticks" per-packet. The initial RTP timestamp is random, i.e., it is not derived from the true "wall clock" time.

- **Marker bit:** In the case that silence suppression is used (see Section 2.3.3), the marker bit is set at the beginning of each talkspurt, otherwise, it is set to zero.

- **Packetization:** An RTP packet may contain one or more codec data frames. Applications typically signal the payload size during the session initialization or as-

[1] This will become important in Chapter 3, when a new payload format is described.

sume the default value defined by the AVP. For constant-bitrate codecs (e.g.,G.711, G.729(A)), the boundaries of the codec frames can be found without explicit indication of their offsets. However, for variable-bitrate codecs (e.g., Enhanced Variable Rate Codec (EVRC)) the RTP payload includes special Table-of-Contents (ToC) fields used for frame indexing.

2.2.1 Real Time Control Protocol

The Real Time Control Protocol (RTCP) provides a QoS *feedback* mechanism for an RTP session. Primarily, RTCP provides statistics on the quality of the RTP session, such as the number of packets/bytes sent to and received from a given participant. Using these statistics, endpoints can dynamically adapt to varying conditions on the link. RTCP also provides an enhanced textual description of a source's identity. This information is intermittently exchanged between the session participants through RTCP reports. The reporting interval is constrained such that the RTCP packets only use a small portion (e.g., 5%) of the total *session bandwidth* (the session bandwidth is the sum of the nominal bandwidths of the expected number of active senders).

2.2.2 Intermediate RTP Systems—Mixers and Translators

Intermediate RTP systems fall into one of two categories: (1) translator, or (2) mixer. To summarize, translators typically forward packets to the receivers intact, but may change the packetization interval or the media encoding. In addition, the SSRC of the packet remains intact such that the receiver can identify the original source of the packet. In contrast, mixers form a new output stream that is comprised of some combination of multiple input streams. The receiver does not have access to the original streams, but may learn the originator's SSRC through the Contributing Source (CSRC) field. An example of an RTP translator and mixer is a packet reflector and a conventional VoIP conference bridge, respectively.

2.2.3 RTP Header Compression

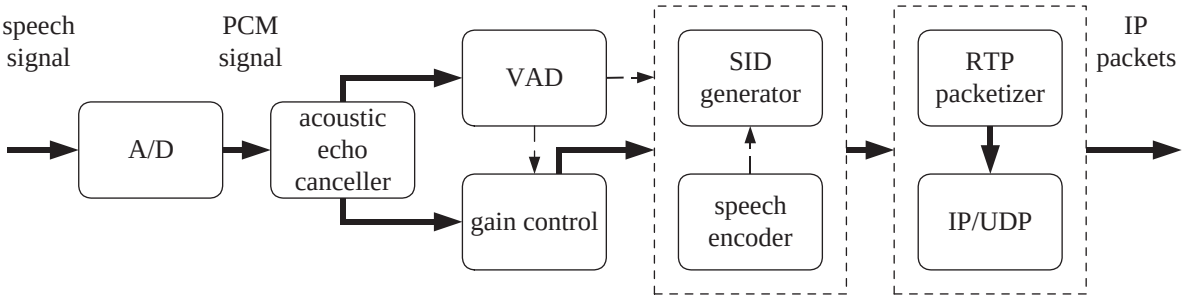
The size of the IP/UDP/RTP header imposes a significant reduction in transmission efficiency when short payload lengths (e.g., 10–30 ms) are used[2]. When the Link Layer header overhead is considered, the combined bitrate of the sender and receiver streams may exceed the nominal bandwidth of the channel (e.g., 56 kbps modem, cellular wireless). In these cases, RTP Header Compression (cRTP) [40] may be used to remove redundancies from the IP/UDP/RTP header, reducing it from 40 bytes to 2–4 bytes. This is possible since about 50% of the header values do not change over the course of the connection, or their first-order differences are constant (e.g., RTP timestamp always increases by 160 for 8 kHz speech and 20 ms payloads). Only *changes* in the first-order differences are transmitted. However, it is expected that 98% of the time only the compressed packet will be sent [18].

2.3 VoIP Transmission Systems

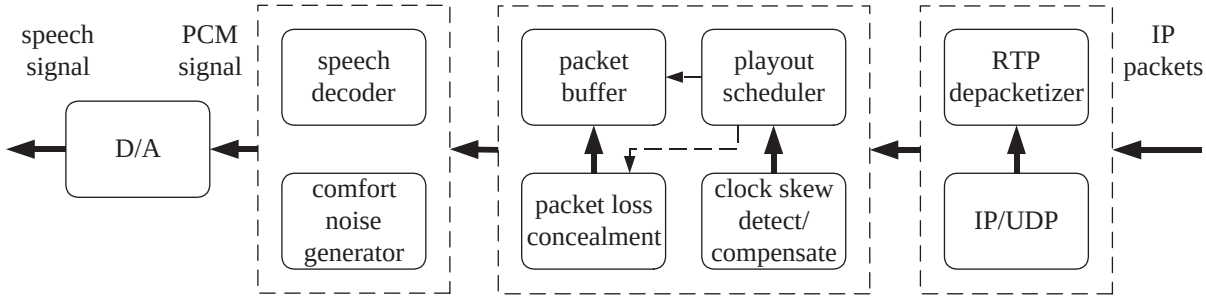
In legacy telephone networks, speech is digitized and transmitted as μ-law PCM signals over TDM trunks. The voice trunks are physically separated from the signalling links and connections are established only when circuits are available through the network. First and foremost the PSTN is designed to carry voice, hence the QoS remains high. Data transmissions, such as fax or modem, are carried with the same precedence as voice at the cost of a low bitrate.

Conversely, the subjective quality of voice communications is compromised when speech signals are carried over a network designed for data, as is the case for VoIP. Problems mainly stem from unpredictable network performance, which cause variable delays and packet loss. To compensate, speech packets travel through various mechanisms before being transmitted or played by an application. These mechanisms help maintain the integrity and quality of the transmission. The reader is referred to Fig. 2.1 for a depiction of a generic VoIP transmission system.

[2] For example, transmission of G.729(A) over RTP with a payload length of 20 ms is only 33.3% efficient, since the header and payload bitrates are 16 kbps and 8 kbps, respectively.



(a) A VoIP transmitter.



(b) A VoIP receiver.

Fig. 2.1 A generic VoIP transmission system.

2.3.1 Speech Codecs

In telecommunications, speech coding is used to achieve efficient transmission of speech signals over the communication network. Conventional telephone speech is sampled at 8 kHz and then bandpass filtered from 300–3400 Hz, allowing representation of the first four formants for most phonemes [21]. The signal is initially sampled with at least 11–12 bits/sample (i.e., 88–96 kbps), and then quantized with the well-known log PCM technique in order to reduce the bitrate to 64 kbps. This scheme results in "toll quality" speech, allowing for a high degree of intelligibility, and adequate speaker identifiability. However, speech coding techniques can reduce the bitrate by exploiting redundancies inherent to the speech signal.

Speech coders divide the input signal into frames. The encoder represents each frame with a reduced number of bits, and the decoder attempts to render the original waveform using the received set of bits. Note that speech compression is *lossy*, meaning the original waveform is never perfectly recovered. Coders typically fall into one of three categories: (1) waveform coders, which attempt to code and reconstruct the speech waveform (e.g., PCM, ADPCM), (2) parametric/source/voice coders (vocoders), which represent speech as the output of a linear system consisting of a vocal tract shape and excitation (e.g., LPC), and (3) hybrid coders, which combine aspects of both waveform coders and vocoders (e.g., CELP).

Speech coder performance is typically measured over a set of attributes such as bitrate, delay, computational complexity, and subjective quality. For carrier-grade VoIP, one desires the highest quality for the lowest bitrate without excessive algorithmic delay and complexity. Two of the most popular codecs for carrier-grade VoIP are ITU-T G.711 and G.729. The G.729A codec trades off a slight reduction in quality relative to G.729 for a 50% reduction in complexity. G.729A is bitstream-compatible with G.729, and is intended for Digital Simultaneous Voice and Data (DSVD) applications such as voice conferencing. These and other popular codecs are summarized in Table 2.1.

The G.711 codec is the traditional 64 kbps log PCM voice encoding technique used in the PSTN. The codec is stateless, and for VoIP the default RTP packetization interval is 20 ms [17]. G.711 is an attractive codec for high bandwidth scenarios and is resilient to tandem encodings. Inter-networking between the PSTN and an VoIP network is simplified in that transcoding is not required, since the speech is already in the proper log PCM format.

The stateless nature of the codec implies that packet loss does not affect encoder/decoder synchronization.

Many modern toll quality, low-bitrate speech codecs such as G.729(A)[3] are generally based on CELP technology. Currently, G.729 is poised to be the codec of choice for carrier-grade VoIP. Although the codec frame length is 10 ms (with a 5 ms lookahead), the default RTP packetization interval is 20 ms, yet transmission of single 10 ms frames is allowed [17]. G.729 is heavily state-dependent, thus the encoder and decoder must remain synchronized in order to prevent degradations in speech quality. Multiple encodings of G.729(A) result in significant degradations in perceived quality. For example, two encodings of G.729 and G.729A result in a degradation of 0.65 and 0.48 Mean Opinion Score (MOS)[4], respectively; three encodings degrade the speech quality by 1.24 and 1.1 MOS, respectively.

Table 2.1 Commonly used speech codecs. Note that the MIPS values are approximate, and were taken from [21], while MOS values were taken from [41]. For consistency, values not available in these two references are not shown.

Standard	Method	Bitrate (kbps)	Delay (ms)	Complexity (MIPS)	Quality (MOS)
G.711	LOG PCM	64	0.125	0.01	4.1
G.726	ADPCM	32	0.125	2	3.85
G.728	LD-CELP	16	0.625	30	3.61
G.729	CS-ACELP	8	15	20	3.92
G.729A	CS-ACELP	8	15	10.5	
G.723.1	MP-MLQ	6.3	37.5	14.6	
	ACELP	5.3	37.5	16	
IS-54	VSELP	7.95	20	14	3.54
GSM-FR	RPE-LTP	13	20	6	3.5
GSM-EFR	ACELP	12.2	20		
GSM-HR	VSELP	5.6	20		

[3]For the remainder of this thesis, the notation, G.729(A) will be used when the context applies to both G.729 *and* G.729A.

[4]MOS is a subjective speech quality measure which ranks speech quality on a five-point scale from 5 (Excellent) to 1 (Unsatisfactory). The MOS is simply the average score over a number of listeners [21].

2.3.2 Voice Activity Detection

In short, the purpose of a Voice Activity Detector (VAD) is to classify a speech signal into speech and silence regions. Among other things, VADs are commonly used to implement silence suppression/Discontinuous Transmission (DTX) in VoIP transmitters (Section 2.3.3), as well as to detect speaker activity in conference bridges (Section 2.5.2).

A VAD, much like a speech coder, operates on a frame-by-frame basis. It produces a single binary *VAD decision* (i.e., VAD = speech, $\overline{\text{VAD}}$ = silence/noise) based on features extracted from the signal. The analysis frame length is usually 10–30 ms, as is typical in speech processing since speech is wide-sense stationary over short periods of time. As the VAD processes the signal, a series of 1s and 0s are formed, creating "on-off" patterns which are known as talkspurts and pauses [42]. The final VAD decision is usually a combination of an initial VAD decision followed by hangover. Hangover "smoothes" the speech-to-silence transitions by extending the initial VAD decision, filling in the natural pauses between words and sentences which helps in preventing audible speech clipping. Traditional VADs used in telecommunications employ a hangover of 150–200 ms [42, 43].

At the core of most VADs is a comparison of the average signal energy (i.e., power) to a noise threshold (noise floor). Speech is declared if the signal power exceeds the noise threshold. Simple VADs use a fixed noise floor, E_N, set to anywhere between -35–50 dBoV [42, 44]. The modern approach is to compare the running estimate (exponential average) of the signal power, $\bar{E}_f$, to both the running estimate of the noise power, $\bar{E}_n$, and the absolute noise floor, E_N [12]. $\bar{E}_f$ is updated for each frame; time constants/update rates are chosen for $\bar{E}_n$ such that it adapts quickly during periods of noise and slowly during periods of speech[5]. Speech is declared if $\bar{E}_f > \bar{E}_n + E_{th}$, where $\bar{E}_n > E_N$, while E_{th} provides hysteresis [45].

The averaging/comparison is often performed on a per-band basis such that thresholds and update rates for each band may be finely tuned. For example, it is common to compute the average low-band energy, $\bar{E}_l$, from 0–1 kHz, and the high-band energy, $\bar{E}_h$, from 1–4 kHz [46, 47], while another algorithm uses sixteen bands [48]. The final VAD decision may be enhanced by inspecting the signal for periodicity, i.e., voicing [49]. In contrast, G.729B

[5]For example, $\bar{E}_n$ can be updated on a per-frame basis, provided that the time constant of the exponential averaging is longer than the one used to update $\bar{E}_f$ [45]; a reasonable value is ≈ 100 ms. Otherwise, $\bar{E}_n$ is updated on every frame during silent periods, and intermittently during speech periods (e.g., once every 350–400 ms) so that it's value remains consistent with temporal variations in the signal energy.

uses four spectral features, including $\bar{E}_l$ and $\bar{E}_h$, in a pattern-matching approach. The VAD declares speech when these parameters occupy an empirically identified region of a four-dimensional Euclidean space. Further, the hangover is dynamic and is based on the recent history of the VAD features [46].

VAD performance is typically evaluated by comparing the VAD output to a "hand-marked" segment of speech. The manual classification divides the test signal into regions of speech and silence. Further classification may involve determining activity during voiced, unvoiced, silence, and transition regions [46]. The VAD is typically tuned to error in favour of speech such that audible speech clipping is minimized. Speech clipping performance relies on four factors, related by

$$P = FL/\alpha, \tag{2.1}$$

where P is the proportion of speech clipped, F is the frequency of clipping, L is the clipping duration, and α is the speech activity ($0 < \alpha < 1$) [43]. Therefore, to maintain a given P, one can increase the *length* of the clips in order to reduce their *frequency*, or vice versa. Silence-to-speech transitions are typically the most difficult to detect and may result in Front-End Clipping (FEC) occurrences. In general, FECs > 50 ms are perceivable and have a negative effect on subjective quality. FEC performance is summarized in Table 2.2.

Table 2.2 Summary of front-end clipping performance [43].

Front-End Clipping Duration	Proportion of Speech Clipped	Frequency of Clip Occurrence Range	Grade-of-Service	
L (ms)	P (%)	F (FECs/min)	MOS	Remark
≤ 15	≤ 0.5	≤ 18	> 4	Good-Excellent
≤ 50	≤ 0.5	≤ 5	≈ 4	Good
≤ 50	≤ 1.6	≤ 17.6	≈ 3.5	Fair-Good
> 50	> 0.04	≤ 0.4		

2.3.3 Silence Suppression/DTX

One of the advantages of packet-based voice communications is that packets need only be sent when the user is talking. This leads to a dual-rate transmission technique known as silence suppression, or Discontinuous Transmission (DTX). The savings in bandwidth are critical for Digital Simultaneous Voice and Data (DSVD) and other bitrate-sensitive applications, and allow for statistical multiplexing of voice traffic over the packet network. Considering a two-party conversation, the bandwidth savings are in the range of 50–60% since one person usually listens while the other one talks (see Section 2.7.1).

During periods of silence, the silence suppression scheme may either turn off the transmitter, or reduce the transmission rate. The former scheme is ad-hoc, and is used by non-commercial, workstation-based conferencing tools, such as RAT. It is the latter approach that is referred to as DTX. DTX schemes typically send one Silence Insertion Descriptor (SID) packet at the beginning of, and intermittently throughout, the silence period. The SID frames are smaller than the speech codec data frames and usually contain the parameters of the background noise. Thus, DTX is better than simply turning off the transmitter since the SID frames provide explicit in-band signalling of the silence period, and help to maintain synchronization between the encoder and decoder states.

2.3.4 Comfort Noise Generation

The use of silence suppression implies that a receiver will have no packets to play out to the audio device during a silence period. Simple receivers play out absolute silence either by writing zeros to the audio device or nothing at all [39]. This is known to confuse listeners, leading them to believe that the line has gone "dead". The solution is to introduce a Comfort Noise Generation (CNG) scheme, where low-level noise is artificially generated and played out to the listener during the silent intervals. Simple CNG algorithms produce comfort noise by generating a frame of perceptually pleasing Gaussian white noise and running it through a colouration filter such as

$$H(z) = \frac{0.325}{1 - 0.675z^{-1}}, \tag{2.2}$$

which gives the noise a low-pass characteristic similar to room (Hoth) noise [45].

In more modern CNG schemes, background noise is generated based on intermittent

parameter updates from the transmitter. This allows the synthesized background noise to have the same nature as the actual instantaneous background noise at the transmitter, leading to a more natural sound. For example, the G.729B CNG scheme transmits a 15-bit SID packet using 5-bits for gain and 10-bits for spectrum representation. An SID packet is always sent for the first frame of silence following a talkspurt. However, a minimum spacing of two frames is imposed between the transmission of consecutive SID frames [46].

In G.729B, the comfort noise is generated by filtering an excitation signal through the LP synthesis filter. The excitation signal is a linear combination of three signals, two of which are derived from the G.729 excitation sources, and the third being Gaussian white noise. This provides a perceptually favourable background noise representation, without creating audible switching between the true and synthetic background noises on a speech-to-silence transition.

2.3.5 Jitter Buffers/Playout Scheduling

The problem of jitter arises from the statistical multiplexing nature of IP networks. VoIP transmitters generate packets on fixed intervals and inject them into the IP network for transmission. During their flight through the network the packets may experience variable delay (i.e., jitter) due to different levels of congestion, or if they are routed over different paths. The result is that packets may arrive at the receiver at irregular intervals, possibly misordered. Jitter is combatted at the network level by a combination of resource reservation and preferential treatment of voice packets. It is also customary to compensate for jitter at the receiver by way of a jitter (playout) buffer.

The purpose of a jitter buffer is to absorb the variable delay of the packet arrival process such that a continuous stream of speech may be delivered to the audio device at the receiver. This is accomplished by buffering the received packets and scheduling them for playout at regular intervals in the future. A packet is discarded if it arrives after its scheduled playout time. Hence, the packet loss rate due to late arrivals can be traded off against the buffering delay.

The length of the jitter buffer can be fixed or made adaptive and is controlled by a playout scheduling algorithm. Fixed approaches assume knowledge of the network delay distribution, and compute the playout delay such that no more than a given fraction of packets are lost [50]. This approach is generally undesirable since punctual packets are

needlessly delayed. Instead, adaptive approaches attempt to estimate the actual end-to-end delay and adjust the playout delay accordingly[6].

A common approach to low complexity adaptive playout delay algorithms is based on a technique first advocated in [13]. The idea is to adapt the playout delay *per-talkspurt*, effectively lengthening or shortening periods of silence. First, the playout point is chosen for the first packet of the talkspurt as

$$p_1^k = t_1^k + D_k, \tag{2.3}$$

where p_1^k and t_1^k are the respective playout and production times of the first packet of the kth talkspurt, and D_k is the playout delay. Second, the playout times of remaining packets of the talkspurt are calculated as an offset from p_1^k, where the offset is equal to the difference in production times of the first and ith packets of the kth talkspurt, i.e.,

$$p_i^k = p_1^k + (t_i^k - t_1^k). \tag{2.4}$$

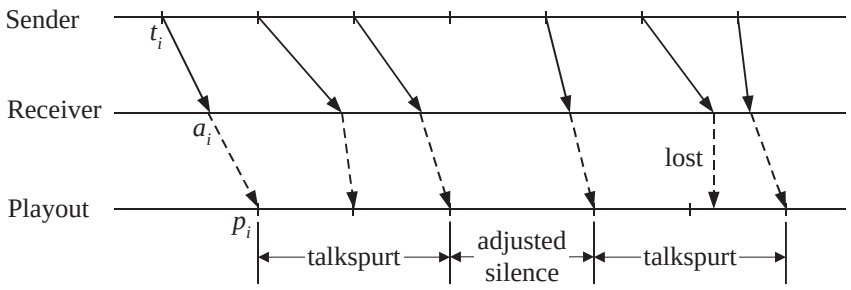


Fig. 2.2 Example operation of a playout buffer adapting per-talkspurt.

Methods based on the above tend to differ in the way D_k is obtained. One well-known

[6]Note that the *network* delay is the difference between the arrival time and production time of a packet, $n_i = a_i - t_i$, while the *playout* delay is the difference between the actual playout time and the production time of a packet, $D_i = p_i - t_i$. The playout delay is also the *total end-to-end delay*, disregarding audio I/O. Sender and receiver clocks are usually not synchronized since they are based on the RTP timestamp. Thus, there is an inherent clock offset in both D_i and n_i which masks of the actual network delay, but not its variance.

method is to set D_k based on the mean and standard deviation of the network delay,

$$D_k = \hat{d}_i^k + \gamma \hat{v}_i^k, \tag{2.5}$$

where $\hat{d}_i^k$ and $\hat{v}_i^k$ are the respective running estimates of the aforementioned quantities. Here, γ is chosen to be the number of standard deviations from $\hat{d}_i^k$ that places p_1^k "far enough" beyond the delay estimate such that only a small portion of arriving packets will be late [14].

Another common approach used in [15, 51, 52] is to track the short-term distribution of network delay (e.g., over 200 s) and then choose D_k such that only a qth fraction of packets are lost. The network delay of each packet is recorded in a histogram, and D_k is obtained as the smallest value satisfying

$$q \leq \sum_{d_n=D_k}^{d_N} h(d_n) / \sum_{d_n=d_1}^{d_N} h(d_n), \tag{2.6}$$

where q is the desired fraction of lost packets, $h(d_n)$ is the delay histogram, and d_1, d_N are the delays which index $h(d_n)$ at the first and last bins, respectively. Note that $h(d_n)$ is updated for each packet arrival.

One disadvantage of using these methods is that they do not adapt quickly to *spikes* in network delay, where a spike is a sharp rise and fall in network delay of several hundred milliseconds. Hence, they operate in one of two modes: "normal" and "spike" mode [14, 15, 51]. While operating in spike mode, the playout delay, D_k, of a *new* talkspurt is set to the network delay of the first packet of that talkspurt, instead of relying on past delay statistics.

A recent approach to adaptive playout scheduling uses the histogram method of Eq. (2.6), but varies the playout delay *per-packet* instead of per-talkspurt [16]. This allows the algorithm to inherently adapt to delay spikes. Upon arrival of the ith packet, the delay and playout time of the $(i+1)$th packet is predicted, and the ith packet is then time-scaled such that it finishes playing its last sample at p_{i+1}. If the delay of the $(i+1)$th packet was predicted correctly, the $(i+1)$th packet will arrive before p_{i+1}.

2.3.6 Packet Loss Concealment

Packet Loss Concealment (PLC) repairs gaps in the voice stream due to lost or late packets. Loss occurs in VoIP networks when congested routers and gateways discard packets or when a packet fails to arrive at a receiver before its playout deadline. In both cases, the most likely root cause of the packet loss is congestion. This implies that packet loss is highly correlated, i.e., it occurs in bursts [53]. Intelligibility is compromised if packets contain more than 50 ms of speech since burst errors of 80–100 ms in duration may lead to loss of entire phonemes [54]. Performance requirements for Mid-speech Burst Clipping (MSC), which is caused by loss, are shown in Table 2.3.

Table 2.3 Summary of mid-speech burst clipping performance [43]. Note that the requirements here are somewhat stringent, and assume no PLC.

Mid-Speech Burst Clipping Duration L (ms)	Proportion of Speech Clipped P (%)	Frequency of Clip Occurrence Range F (MSCs/min)	Grade-of-Service	
			MOS	Remark
≤ 50	≤ 0.2	$\geq 2.2 \geq$	≥ 4	Good-Excellent
≤ 50	≤ 0.5	$\geq 5.4 \geq$	3.5	Fair-Good

Low-cost PLC such as silence or noise substitution can provide adequate intelligibility for low loss rates, although packet repetition is preferable [55]. Modern speech codecs for VoIP are equipped with built-in PLC mechanisms and are typically rated for 3–5% burst frame errors. For G.729[7], burst frame errors of 3 and 5% result in a reduction of 0.5 and 0.97 MOS, respectively [22]; for G.729A, another study showed the reduction in MOS to be 0.58 and 0.92, respectively [38]. Packet loss may have long-term effects on state-dependent coders. For example, the G.729A decoder requires up to 300 ms to fully re-synchronize with the encoder following a period of packet loss, although the audible distortion does not last as long [56]. PLC can also be used to reduce the length of the jitter buffer by trading off a higher loss rate for less delay [16].

[7] In G.729(A), a missing frame is regenerated by exciting the last known LP synthesis filter with either a purely periodic excitation with decreasing lag, or random, non-periodic excitation, depending on whether the last good frame was periodic or not. In both cases, the excitation gain is exponentially damped for each consecutive frames [22].

A media stream may be further protected against loss by a variety of transmission schemes. Schemes supported by RTP have been summarized in [57]. Forward error correction schemes add redundancy to the media stream with either media-independent or media-dependent encodings. The idea is to transmit the k previous codec frames in addition to frame i, allowing the receiver to recover frame i by some combination of the following packets and the redundant stream. Media-*independent* schemes output one redundant frame as an error correcting function of k prior frames [58]. Media-*dependent* streams encode each of the k prior frames at a much lower bitrate and append them to the current packet [59]. The drawback of either scheme is that there is a delay of at least one packetization interval before recovery can be performed. If delay is not an issue (e.g., non-interactive sessions), interleaving can be used. In this approach, multiple non-adjacent codec frames are bundled into a single packet such that packet loss does not result in burst frame errors. This allows the missing frames to be recovered more robustly, using speech processing techniques which take advantage of past and future frames [60].

2.3.7 Audio Clock Skew Compensation

Audio clock synchronization between a sender and receiver is a challenging problem in conventional VoIP networks. The *actual* sampling rate of the audio device will deviate (slightly) from the *expected* rate. In PC-based systems, the difference between sender and receiver clock frequencies, i.e., the skew, may be pronounced and can reach as high as $\pm 0.5\%$ [61]. If a receiver consumes audio at a faster/slower rate than it is produced by the sender, the jitter buffer will eventually underflow/overflow. In either case, the continuity of the audio stream is abruptly interrupted and the audio quality is compromised[8].

A simple mechanism for skew detection and compensation has been suggested in [61]. The skew detector estimates the divergence in samples (i.e., RTP timestamps), δ, between the sender and receiver sampling clocks. Once δ reaches a threshold, the skew correction algorithm runs. To estimate δ, a running average of the one-way network delay, $\hat{n}_i$, is maintained and periodically stored as $\hat{n}_{ref}$. If $\hat{n}_i$ diverges from $\hat{n}_{ref}$ by some threshold, the skew compensator inserts/deletes δ samples to/from the received stream, bringing it back

[8]The Network Time Protocol (NTP) is used to synchronize the *system* clocks of IP hosts, providing accuracy in the order of milliseconds. However, the accuracy is not guaranteed, and not all hosts support it [62]. Note that NTP does not solve the audio skew problem, since the audio clock frequency is independent of the system clock.

into alignment with the local sampling clock. The insertion/deletion is based on low-cost waveform substitution techniques.

One drawback of the above scheme is that short-term fluctuations in the network delay are indistinguishable from clock skew, hence the compensator may run needlessly. Nevertheless, the algorithm has been shown to maintain jitter buffer occupancy, and thus synchronism between endpoints. More robust skew detection techniques may be used if the long-term packet arrival process is considered [15].

2.3.8 Audio I/O

Simply put, a VoIP application uses the audio device to capture/playback audio signals from/to the user of the terminal. Real-time, dedicated voice terminals such as wireless handsets or IP phones are optimized to provide low latency access to the audio device. Yet, softphones and conferencing tools running on multi-purpose desktop workstations are often plagued by poor audio I/O latency[9]. Audio drop-outs may occur if the latency is too long. Typically, kernel modifications are necessary in order to provide the real-time capabilities, but this not always an acceptable solution for the casual user. The three main factors associated with the audio latency are (1) the audio latency in the audio subsystem, (2) the scheduler latency, and (3) the A/D and D/A converter latency.

Audio latency is the delay between the time an application writes/reads audio data to/from the audio device, and the time the request is completed. On Windows, any output audio stream must travel through the Windows Kernel Mode Mixer (KMixer), the system software mixer, no matter if it is generated through the Waveform Audio or DirectSound API. The KMixer is known to add at least 30 ms of buffering delay to an output stream [63], although this delay can grow when KMixer adapts after periods of starvation [64]. Lower latencies can be acheived on Windows XP, which allows the KMixer to be bypassed. On Linux, audio I/O is accomplished with either the Open Sound System (OSS) or Advanced Linux Sound Architecture (ALSA) API. "Untuned" Linux platforms can yield audio latencies $\leq$5 ms [65].

Scheduling anomalies may cause the audio device to underflow. This results in a period of silence (drop-out) equal to the duration of the starvation *minus* the audio already stored in the DMA buffers. In the case of a continuous stream (no silence suppression),

[9] A detailed description of the delay budget for a two-way call using RAT, on both Windows and Linux platforms is provided in Appendix A.

the silent period adds a fixed, irreversible delay to the output stream. At any time during the conference, the additional audio output delay will be equal to the longest audio drop out. Softphones deal with this problem indirectly by using silence suppression on the transmit stream, allowing the audio buffer at the receiver to drain during silence periods. In addition, a "cushion" of audio may be fed to the audio device, to cover for periods of scheduler-induced starvation [66]. The length of the cushion corresponds to some user-defined percentile of the short-term starvation length distribution, similar to the way playout delay is calculated in [15, 51, 52].

A/D conversion delay is the least severe audio-related delay. PC soundcards support a standard set of sampling rates ranging from 8 kHz to 48 kHz. However, some cards sample at the highest possible rate and then up/down-sample to meet the requested rate. The anti-aliasing filters in the A/D and D/A units impose a fixed delay of approximately 1–1.5 ms [63].

2.3.9 Echo Cancellation

Echo has a detrimental effect on the perceived quality of a communication link. It occurs because of transmitted signals being coupled into a return path and fed back to their respective sources. Echo path delays of about 25 ms become annoying and can break the cadence in a conversation [18], yet longer delays imply more severe impairment [43]. To compensate, echo cancellers are used to estimate and subtract the echo from the affected signal. There are two types of echoes in a voice communication network: hybrid and acoustic.

Hybrid echo is common for Plain Old Telephone Service (POTS) connections, resulting from an impedance mismatch at a four-wire to two-wire hybrid. Mismatches in the hybrid cause signals from the incoming branch of the four-wire circuit to get fed back into the outgoing branch towards the source [1]. In VoIP, this type of echo occurs for PSTN-to-IP connections (or vice-versa). Echo cancellation is performed by a gateway on the four-wire side of the hybrid [53]. Note that hybrid echo is not present in pure IP-to-IP calls since the connection is digital from end-to-end.

Acoustic echo occurs at a VoIP endpoint when some of the signal from the ear-piece (or speakers) re-enters the mouthpiece and is sent back towards the source. This is a common problem for inexpensive hands-free speaker-phones and headsets [55]. In this case, the echo

canceller is built into the endpoint, close to the source of the echo.

2.4 IP Telephony Architecture

VoIP started as the enabling technology for computer-based multimedia conferencing over Local Area Networks (LANs) and the Internet. Shortly thereafter, this technology gained popularity by providing a long-distance "toll-bypass" system for enterprise telephony. Now the concentration is on the design and deployment of large-scale VoIP networks that co-exist with the PSTN. Retirement of the PSTN is still far off, and conventional carrier-grade IP telephony networks highly resemble the PSTN, yet both media and signalling traffic are multiplexed on an IP switching fabric.

2.4.1 VoIP Protocol Suite Evolution

The VoIP protocol suite consists of a collection of network, transport, and call processing protocols. Most are carried over IP/UDP, since flow control is usually inherent to the higher level protocol. The major call processing protocols are H.323 and the Session Initiation Protocol (SIP) for establishing multimedia sessions, as well as Media Gateway Control Protocol (MGCP) and Media Gateway Control (MEGACO) for controlling Media Gateways (MGs). Each of these protocols (and more) are being used today, to some extent. One important commonality between these protocols is that RTP is used for media transport.

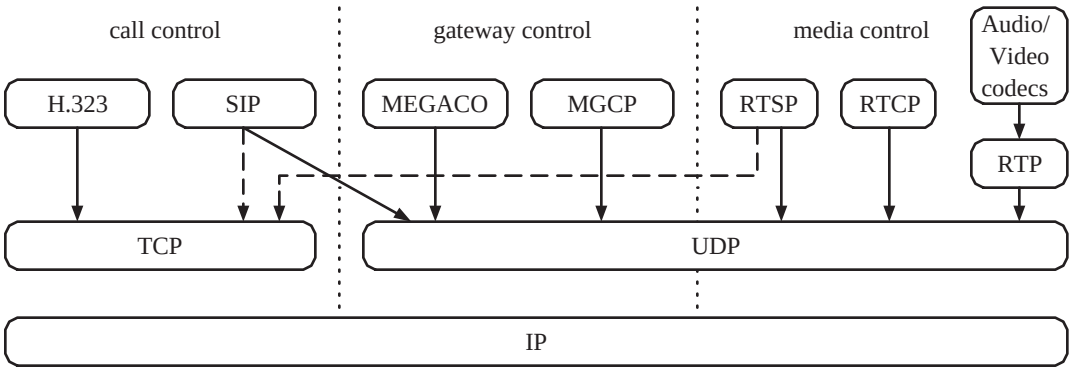


Fig. 2.3 Simplified VoIP protocol stack. Note that RTSP is used for the acquisition of an on-demand media stream, such as a pre-recorded network announcement. The actual streaming is accomplished with RTP. Further, note that SDP (not shown here) is used by SIP, MEGACO, and MGCP for describing media sessions.

ITU-T H.323 [9] was originally intended and used for LAN-based multimedia conferencing, relying heavily upon existing ITU-T multimedia protocols. It is an umbrella specification, i.e., it is comprised of a number of other specifications which attempt to define all aspects of audio, video, and data communication over the IP network. From the very beginning, H.323 focussed on interoperability with the PSTN by defining the IP-to-PSTN *gateway*. However, the original H.323 gateway was not scalable as it encompassed call control, signalling and media conversion on one platform [67]. Further, H.323 had no provision for Signalling System Number 7 (SS7) connectivity. New solutions were sought as it became clear that the H.323 architecture was not suitable for large carrier-grade networks.

The idea of decomposed gateways emerged in the form of the MGCP—the MEGACO [68] standard was soon to follow. The emergence of the Media Gateway Controller (MGC) and MG separated call control from media handling, thereby alleviating the scalability problems imposed by the H.323 gateway. This separation followed the traditional view of telephone switches, where centralized intelligence was separated from the peripherals on which the actual media traffic terminates [69]. While MGCP/MEGACO cleared up the immediate issues of PSTN interoperability, there remained a need for higher level mechanisms to tie MGCs together and to provide enhanced SS7-like services. SIP has been proposed to fill this role.

The Session Initiation Protocol (SIP) [70] is used to establish, modify, or terminate multimedia sessions such as an Internet phone call or a multimedia conference. SIP provides a signalling framework in an Internet context for implementing services such as call forwarding, conferencing, or unified messaging [6], yet the protocol itself is independent of the details of the underlying service or media. Like RTP, the protocol relies on companion documents to outline specific use-cases. As mentioned above, SIP has found potential use in IP telephony as a communication mechanism between MGCs or between MGCs and intelligent endpoints such as softphones. Moreover, it provides a suitable mechanism for mapping or tunnelling SS7 messages to the IP domain [8, 71].

To summarize, the current view of carrier-grade IP telephony is that of a gateway-driven model. The network can support IP-to-IP calls, IP-to-PSTN calls, or PSTN-to-PSTN calls, where in this last case the IP network is used as an intermediate leg in the connection. Further details of the gateway model are given in the following sections.

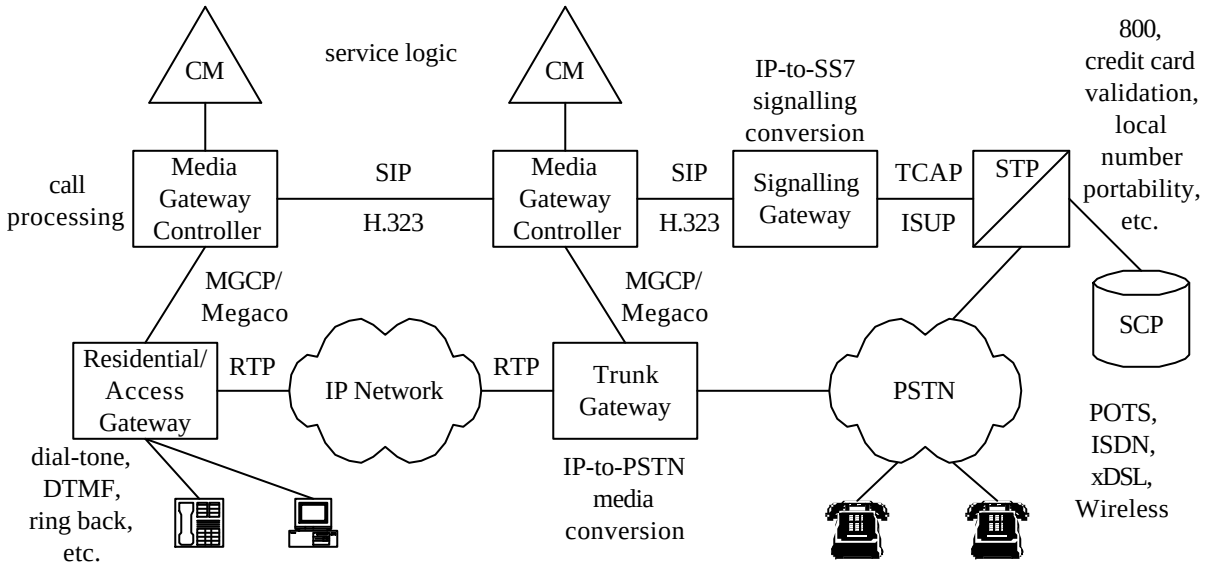


Fig. 2.4 A generic IP telephony network (based in part on [8]).

2.4.2 Media Gateway Controllers

The Media Gateway Controller (MGC)[10] is the overall controller of the system, implementing the call control plane of the IP telephony architecture [18]. Typically, the MGC controls one or more MGs and houses the core service logic which is used to set-up, maintain, and tear-down calls between endpoints. The MGC typically includes another higher-level module, i.e., the "Central Module" (CM), which is responsible for call routing, resource management, call logging etc. To complete an IP-to-PSTN call, the MGC initiates/terminates SS7 signalling with the PSTN through a Signalling Gateway (SG) to a Signal Transfer Point (STP).

2.4.3 Media Gateways

Media Gateways (MGs) are edge nodes which provide the media mapping and/or transcoding functions between IP and other networks [67]. The MG includes standard components of a packet transmitter/receiver such as a bank of speech codecs, adaptive jitter buffers, PLC mechanisms, echo cancellers etc. Since media transformations take place over the

[10]Media Gateway Controller is a term used by the MEGACO standard. The same entity may be known as a Call Agent (MGCP), Softswitch, or simply Gateway Controller. Esoteric terms include Virtual Switch Controller (Cisco Systems) and Communication Server (Nortel Networks).

duration of the call, MGs are typically laden with processing power drawn from a DSP platform. Depending on the need, the MGs may serve different roles at different locations in the network. Referring to Fig. 2.4, the MG is used as both a residential/access gateway and also a trunking gateway.

In the residential case, the MG may provide the hardware/software interconnect to support calls originating from POTS, cable, xDSL, IP, or ISDN connections. Further to media conversion, the MG may perform DTMF detection/generation, call progress tone generation/detection (e.g., dial-tone, busy, fast busy, and ring-back), or analog line supervision (e.g., on-hook, off-hook, flash, and ring). Other connection types and capabilities are added to the MG as the need arises [68].

In the trunking gateway case, the MG compresses/decompresses large volumes of speech traffic as it travels between the IP network and some other network. Here, the role of the MG may be as an intermediate system between two carrier offices such as when the IP network is used for long haul transport of long distance calls [8]. Although mainly responsible for packet-to-PCM conversion, other mappings may be required such as IP-to-ATM. The MG may connect to a wireless network, providing opportunity for TFO where the far-end supports the source codec [25, 26].

Instead of residing in the network, the MGs may also reside directly on an intelligent endpoint such as softphone or an IP phone. In this way, the endpoint may communicate directly with an MGC—the endpoint terminates the media connection itself, forgoing the need for an intermediate MG. In essence, this allows the MGC to control a plethora of MG-phones instead of controlling an MG, which in turn is connected to a PBX or end-office [71].

2.4.4 Signalling Gateways

Just as the MG maps media formats between the IP network and others, the Signalling Gateway (SG) is responsible for translating *signalling* messages to and from the IP domain. The SG sits between an MGC and an STP—SS7 messages are converted to their equivalent SIP/H.323 messages, or tunnelled the directly to the MGC [71]. The importance of the SG is two-fold: it allows IP-to-PSTN calls to be realized, and it provides the IP network with access to the mature services and features offered by the PSTN.

The latter point is imperative for the success of IP telephony. Other than voice quality,

a major barrier to full-scale deployment of VoIP has been the migration of traditional SS7 services such as 800 (toll-free) calling, credit card authentication, local number portability, etc. In the PSTN, these services are provided by a combination of STP signalling and Signal Control Point (SCP) transactions. Instead of re-implementing these services from scratch, the IP network simply accesses them from the PSTN via the SG.

2.5 Centralized Conferencing

As discussed in Chapter 1, centralized conferencing via a "dial-in" conference bridge is the most prevalent way to establish medium-to-large conferences in a carrier-grade network. Recall that the bridge may be comprised of a network interface module and an audio mixing module. Thus, the "ugly" details of the communication medium may be hidden from the core audio bridge, which operates on linear PCM samples. The details of the network interface are those components discussed in Fig. 2.1, although the A/D and D/A units are removed and the internal PCM signals are effectively coupled to an audio bridge. A simplified view of this has also been shown in Fig. 1.2.

The media connections of an exemplary centralized VoIP conference are illustrated in Fig. 2.5. Each endpoint sends one voice stream to the bridge and receives one in return. The role of the bridge is to form N composite output signals based on M inputs, where $M \leq N$. Customarily, each conferee receives a tailored audio mix consisting of the voices of all other conferees except his/her own. Note that M is some function of the number of *active talkers*. For example, if the conference terminals are using silence suppression/DTX, then M may be the number of active transmitters, or the bridge may perform speaker selection, where it chooses M-out-of-N conferees as the current talkers and partially or completely attenuates the signals of the $N - M$ other conferees. The latter is the way of traditional bridges, as DTX was not available. Some modern digital bridges even go as far as to provide additional speech processing functionality such as stereo imaging, 3-D audio rendering, subconferencing, whispering, and muting [72, 73].

Most standards and public documents which address conference bridging leave the details of the audio processing algorithm to the implementer. These algorithms are relatively simple, but have been "fine-tuned" since the 1970s leading to a certain degree of industry "know-how". This information is generally captured in patent disclosures. The following three sections attempt to bring to light some of the traditional mixing techniques used in

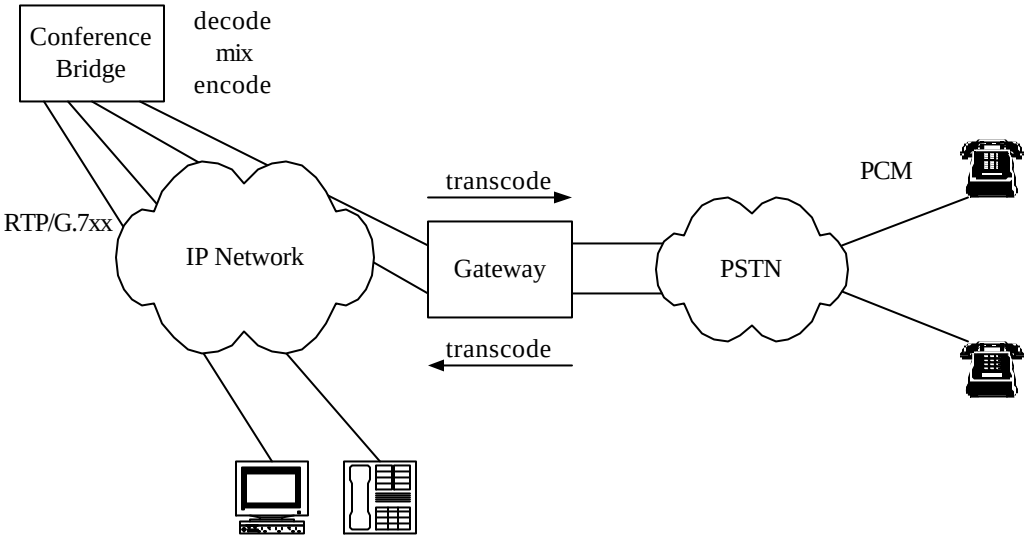


Fig. 2.5 Media connections in a centralized VoIP conference. For simplicity, the signalling connections are not shown. This example shows the internetworking of two VoIP endpoints and two PSTN telephones. Each line represents one full-duplex connection. Note that additional speech quality reduction will occur if the gateway transcodes between two different high compression codecs, as is the case for transoceanic links.

audio bridges, followed by a discussion of some of the possible variations made available by IP (not including the tandem-bypass techniques highlighted in Section 1.4).

2.5.1 Audio Bridge Fundamentals

Other than to provide communication between N conferees, one of the goals of the bridge is to make an N-party connection sound as good as a typical two-party connection. However, conferees usually connect to the bridge with different terminal equipment and over different transmission facilities, making it difficult to achieve the same quality as a two-party call. For example, it is common for the analog leg (i.e., the subscriber loop) of the transmission facilities to introduce low level noise and echo. Singing[11] may occur if the accumulated noise and echo becomes too large [4]. This is a common problem with conference bridges

[11] *Talker* echo occurs when the talker's signal is reflected back to the ear of the talker; *listener* echo occurs when the talker's signal is received more than once at the listener's ear due to multiple reflections at the hybrid. *Singing* occurs when echo is repeatedly reflected back and forth between two hybrids. The superposition of the reflections produce oscillations, which may render the circuit unstable at the frequency of oscillation, producing a high pitched tone [1, 43]

since the signal power of the conference sum is often greater than that of a typical two-party call, resulting in a larger reflection at the four-wire to two-wire hybrid [2].

Originally, simple conference bridges attenuated the conference sum before transmission at the cost of providing lower signal power to the conferees. This approach affected intelligibility. However, the longstanding approach has been to minimize echo and singing by limiting the number of signals that are included in the conference sum. There are two basic operations that are common across all bridge implementations [33]:

1. The input signal of an active conferee is not included in the composite signal that is returned to the same active conferee (Fig. 1).

2. The input signals of inactive conferees are either partially or completely attenuated.

The first point eliminates howling and perceived talker echo at each conference terminal, as well as the unnecessary use of network echo cancellers. The second point avoids the problem of low-level noise build-up from the inactive conferees which contributes to echo and singing. The ITU-T has included these two requirements in their voice conferencing recommendations, for both the PSTN [33] and IP networks [9]. Specifically in [33], the bridge is required to insert at least 15 dB of loss (attenuation) into the inactive conferees' input signals. When conferees become active, the same loss is switched to their listening path such that they do not hear their own voice [33]. Here, "activity" is determined by a VAD—in the packet world, speech activity can be implied by the arrival of a non-SID frame.

2.5.2 Speaker Selection Bridging

Even when a bridge employs switched-loss, problems still remain when all N conferees talk at once, specifically saturation and an increase in the worst-case echo. As mentioned above, early bridges simply attenuated the entire conference sum prior to transmission. The use of speaker selection techniques allowed higher signal powers to be provided to the conferees [1].

Speaker selection consists of selecting M-out-of-N ($M \leq N$) received signals for output. Often, M is set to one or two, which limits the ability of the conferees to interact. Some conferees will try to speak but will not be heard, so the algorithm must make an M-talker system sound and behave like an N-talker system (without noise). This is achievable since

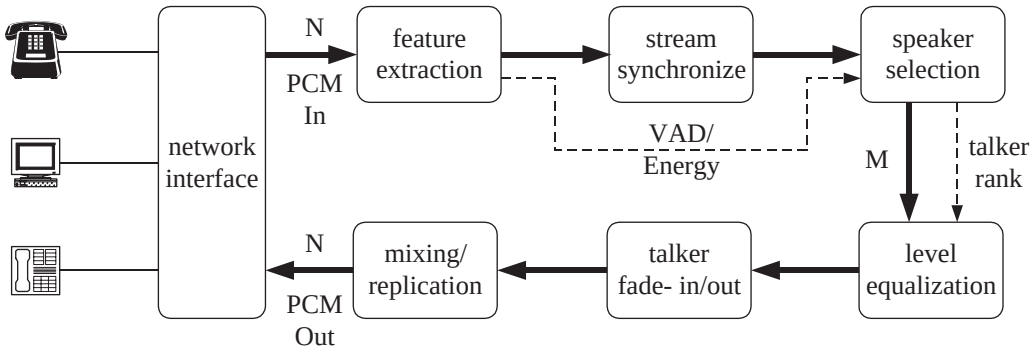


Fig. 2.6 A generic audio bridge. The network interface hides the details of the transmission facilities from the audio processing program. Note that early audio bridges often included an echo canceller. Today, robust echo cancellation is provided by the network—in VoIP, specifically by the Media Gateway.

periods of multi-talk are short and consist of overlaps at the beginnings/ends of talkspurts, short utterances, interruptions, and noise [23, 44]. Section 2.7 expands on these points in more detail.

Speaker selection algorithms make comparisons between conferees based on some criteria; the input streams are generally cross-synchronized before the algorithm is run. Typically, selection features are extracted from the input signals, the features being an indication of activity (i.e., VAD bit) or average speech level (or energy). The former implies a time-, or order-based approach, while the latter implies a level-based approach.

Time-Based Criterion

In general, time-based algorithms assign talking privileges based on the First-Come-First-Served (FCFS) criterion. The algorithm is supported by a VAD which determines if a conferee's current speech frame is active or inactive. A prioritized list of active conferees is maintained—the order of the list represents the order that the conferees became active. For each frame, the algorithm chooses the first M priority slots as selected conferees. Once a conferee has assumed one of the first M priority slots, speaking privileges are not lost until the conferee stops talking. When a selected conferee becomes inactive, all lower priority active conferees are effectively moved up one position in priority.

A problem occurs when two conferees become active at the same time. The solution is to maintain a dual list of inactive conferees (i.e., the "interrupters"), prioritized according

to some heuristic. For example, the last conferee to become inactive could be assigned the highest priority, giving preference to conferees that most recently participated in the conference. This has been shown to maintain the focus and context of the conference [23].

An inherent problem with the FCFS approach is that conferees are often selected in mid-talkspurt, resulting in a front-end clipping occurrence for the newly selected conferee. Intelligibility may be compromised if the front-end clipping is greater than 50 ms [43]. In addition, the switch is highly noticeable since there is an abrupt change in volume as control is passed to the new conferee[12]. A solution is to delay the switch until the newly selected speaker stops and re-starts talking [34], although this slows the pace of the conference.

Overall, the main advantage of the FCFS approach is that it gives both quiet and loud talkers equal opportunity to participate in the conference. However, the method's weak point is that it has no notion of *preemption*, thereby depending on the good manners of the current talker to relinquish control of the conference. Therefore, FCFS selection is suitable for small three-member conferences [12].

Level-Based Criterion

Level-based algorithms assign talking privileges based on the average speech level or energy of the conferees. Unlike the former approach, a list of priorities is usually not maintained since the order of participation is not considered. One of the first approaches, known as the "instant speaker" algorithm, was operated on a sample-by-sample basis. This algorithm chose the sample with greatest magnitude and transmitted it to the other $N-1$ conferees [28]. Other similar approaches were based on the average absolute level per-frame, the frame duration being 2–6 ms therein [31, 74]. Another system only switched when a conferee's speech level exceeded that of the current talker for 24 *consecutive* samples [29].

Multi-speaker level-based algorithms are a natural extension to those methods outlined above, except the M loudest conferees are given talking privileges. This exposes these systems to the previously discussed echo problem. To compensate, a varying degree of attenuation is applied to the M selected signals based on their relative loudness. The loudest talker is attenuated the least (0–3 dB), while the Mth talker is attenuated the most (3–20 dB) [30, 32].

The main advantage of level-based algorithms over time-based algorithms is that they

[12] The speech signal power drops several dB before the VAD declares inactivity, or silence.

provide a mechanism for interrupting the current talker. However, level-based speaker selection is prone to frequent switching when two or more conferees have approximately equal speech level averages. This results in voice break-up and is disturbing to the listeners. Sudden short bursts of noise such as paper rustling or coughing will also cause spurious switching if $M > 1$.

Time-Level Criterion

The problems with purely time- or level-based algorithms can be partially controlled by using time- *and* level-based measures in unison to smooth out the selection process. One conferencing system that pioneered this concept is described in [4]. This algorithm tracks the state of each conferee's talkspurt with a six-state model. The model breaks each utterance into a periods of speech onset and mid-speech, each with a respective hangover state. In addition, two inactive states are used to discern between recently active and passive conferees. Priority is (re)assigned every 2–3 ms, and the combination of state, time spent in state, and VAD decision are used to derive an "adjusted priority level" (APL) constant for each conferee. The M conferees with the greatest APL's are assigned talking privileges. The state priorities have a hierarchical structure that guarantee that a talker in a high priority state cannot be interrupted by one in a lower priority state, much like the FCFS approach. However, once two conferees are in the same state, priority is assigned based on relative loudness.

A less complicated two-talker algorithm protects against frequent switching but not against noise [30]. This algorithm operates on a sample-by-sample basis and operates directly on the PCM codeword. The algorithm behaves as an envelope detector, as it precisely tracks increases in speech level, but prolongs decreases in level by way of an exponential decay with a time constant $\tau = 46.24$ ms. Thus, as the speaker's level suddenly drops at a pause between phrases, the algorithm smoothes out the dip, providing some protection against spurious switching.

These algorithms have a distinct advantage over FCFS techniques in that they avoid switching during mid-speech and prevent short bursts of noise from being heard. In addition, they prevent voice break-up from rapid switching between conferees as in the level-based approaches.

2.5.3 Signal Level Equalization and Fading

In addition to the problem of noise accumulation, the conference audio quality may suffer due to large speech level contrasts between conferees. These contrasts may arise due to the transmission facilities (e.g., long distance vs. local call), the conferee's proximity to the microphone or mouthpiece, and/or naturally quiet or loud talkers. To remedy this, Automatic Gain Control (AGC) is used to equalize gain levels between conferees and thereby minimize speech level contrast. The AGC unit requires the co-existence of a VAD so that gain control is not performed during silence periods. The basic idea is to amplify quiet signals and attenuate loud signals with respect to the recent history of the speech level [4]. Typically, only the signals of the M current talkers undergo level equalization.

Another form of contrast is due to speaker selection, specifically when the signal level of the newly selected speaker is several dB above that of the previous speaker. For this reason, an increasing/decreasing amount of gain is applied to newly selected/unselected talkers until a unity/zero gain condition is reached. This technique helps to eliminate abrupt and annoying contrasts in volume when a new conferee is selected for output [4, 12].

In both of the above cases, gain-scaling may be applied to the speech signals as they pass through each respective module or it may be deferred to coincide with the mixing operation. Since carrier-grade conference bridges are typically built on DSP platforms, the standard multiply-accumulate operation can be used to combine the gain-scaling and mixing into one step. Here, the frame length can derived from either the speaker selection module, or from the speech codec frame length.

2.5.4 Audio Mixing Options

The final task of the audio bridge is to sum (i.e., mix) the input signals together, forming the conference sum. The mixing operations are tightly coupled to the output of the speaker selection module in order to optimize the use of the mixing resource. The mixer must form N output signals based on M input signals as identified by the speaker selection module. In some cases, all N input signals are routed to the mixer and summed, but the $N - M$ inactive signals are highly attenuated before the time of the mix, as is the case in switched-loss algorithms. A more efficient means is to consider the M selected signals for mixing and discard the $N - M$ inactive ones. These issues are discussed in the following paragraphs.

In general, there are two methods that legacy bridges use to form a gain-adjusted sum

for a given conferee [3]. The first way to generate the conference sum is to hold the gain coefficients constant for all conferees, k. Then the sample received by the kth conferee is:

$$R_k = \sum_{i=1}^{N} g_i T_i - g_k T_k, \quad k = 1 \ldots N, \tag{2.7}$$

where T_i and T_k are the mouth[13] samples on ports i and k, g_i is the gain coefficient for the ith port, and g_k is the gain coefficient from port k to itself. Here, the total gain-scaled conference sum is formed first, followed by the subtraction of the receiving port's gain-scaled ear sample, requiring a total of $2N$ operations. Note that a given sample's contribution to the sum may be eliminated by setting $g_i = g_k$, when $i = k$.

The second method is more flexible although it requires considerably more operations. Instead of accumulating the entire conference sum and then subtracting each port's mouth sample, the sample received by each conferee is built up individually according to the desired gain between the input and output ports. The kth port's receive sample is calculated as

$$R_k = \sum_{\substack{i=1 \\ i \neq k}}^{N} g_{ik} T_i, \quad k = 1 \ldots N, \tag{2.8}$$

where g_{ik} is the desired inter-port gain between ports i and k, and T_i is the mouth sample for port i. Here, each R_k is formed by summing up the gain-scaled mouth samples from the $N - 1$ other conferees, requiring $N^2 - N$ operations.

Although the second method is more computationally taxing, it allows the bridge more freedom in assigning inter-port gains. The need arises, for example, when the bridge resides within a PBX and hosts conferees from within the PBX and from a Central Office (CO). In this case, conferees residing within the PBX require 6 dB switched loss between them, while connections through the CO require 0 dB loss, since loss is already built into the network [3]. Another use is on advanced digital bridges where each conferee is allowed to control attributes of the conference sum, such as the received signal level of a *particular* conferee, or the 3-D audio positioning [73]. However, the flexibility of Eq. (2.8) is not needed

[13] In digital conferencing, one frequently refers to *mouth* samples and *ear* samples. The bridge receives a mouth sample from a conferee on one of its N input ports. An ear sample is destined for the ear of another conferee, prior to gain scaling and mixing. In short, a mouth sample becomes an ear sample once it is placed in another conferee's listening path.

in all cases. If it is known in advance that the bridge will serve homogeneous endpoints (e.g., Microsoft NetMeeting terminals on an enterprise LAN) then Eq. (2.7) can be used without compromising functionality. The algorithm may be enhanced with additional logic which allows further gain-scaling per-sample as needed.

If the bridge uses speaker selection, or if the endpoints use silence suppression/DTX, the mixing algorithms described above are highly inefficient. In these two cases, the mixer need only consider the M current talkers. This reduces the number of *unique* output signals to be $M+1$. In this case, unique sums for the M current talkers may be formed by running the summations in Eq. (2.7) and Eq. (2.8) over the indices corresponding to said conferees. Then, the sum of *all* M selected talkers is formed once, followed by replication and distribution to the $N-M$ inactive conferees. Of course, if further inter-port gain scaling is required it can be applied on a per-need basis.

In the case where the bridge forms $M+1$ sums it has been suggested in [9, 75] that the bridge distribute the last sum via a single *multicast* transmit, as opposed to the usual replicate-transmit approach. This implies that each endpoint transmit/receive to/from both a unicast and multicast address over the course of the conference. Upon receipt of the multicast packet, the endpoint inspects the CSRC field contained in the RTP header to see if its own audio is contained in the mix—if not, it is played out. The endpoint receives on its unicast port as usual. The advantage of this technique is that bandwidth is conserved since the number of output streams (at the bridge) is reduced from N to $M+1$. The main disadvantage is that the peak bandwidth requirement at the endpoint is doubled, since the current talkers needlessly receive the multicast stream in addition to the unicast stream.

2.5.5 Commercial Conference Bridges

As described in Chapter 1, the conference bridge may be implemented as a stand-alone server, or as an application on an multi-faceted audio processing platform. Early VoIP conference bridges were based on the H.323-defined Multipoint Processor (MP). However, the growing popularity of MGCP, MEGACO, and SIP have led to new products with control interfaces based on these standards. In fact, conferencing functionality is inherent to the MGCP/MEGACO specifications, but not to SIP, although SIP-based conferencing models have been outlined in [5]. A summary of new and old conference bridge products is shown in Table 2.4. Note that some commercial VoIP conferencing *solutions* may simply

involve a traditional PCM bridge interfaced to the IP network through a Media Gateway.

Table 2.4 Summary of commercial VoIP conference bridges, based in part on [76]. This list will be quickly outdated due to rapid industry changes. Note that all of these products are actually multi-purpose *media servers* which offer a conferencing application (except the Radvision MP Platform).

Vendor	Product	Total Ports	Conference Capacity	Speech Codecs	Control Interface
Convedia	CMS-6000 Media Server	≤18,000	300	G.711 G.723.1 G.729(A)	MGCP Megaco SIP
IP Unity	Harmony 6000 Media Server	≤16,000	1000, or 5000 3-way calls	G.711 G.726 G.723.1 G.729(A) G.728	MGCP Megaco/H.248
Lucent	Media Server			G.711	MGCP Megaco
Nortel Networks	Universal Audio Server	240 480	61	G.711 G.729(A)	Megaco/H.248
Radvision	MP Platform		24 20 20	G.711 G.723.1 G.729(A)	Radvision MP API

2.5.6 End System Mixing—A Centralized Conferencing Variant

End system mixing is suitable for small, ad-hoc conferences [5]. In this model, one of the endpoints assumes the mixing and distribution duties normally performed by the conference bridge. Each endpoint sends its voice stream to the endpoint-bridge, where normal bridging functions take place. The only difference between this and the typical centralized case is that the endpoint-bridge now adds its locally generated speech to each outgoing stream. The scheme is depicted in Fig. 2.7.

Another notable difference between this model and the bridged approach is the way the

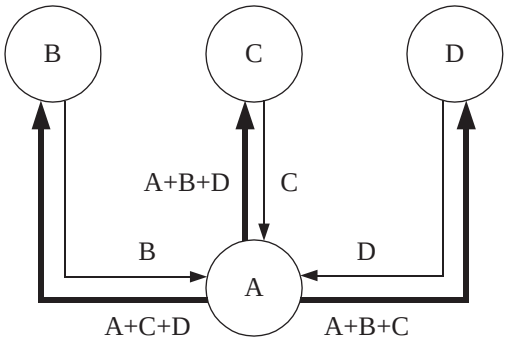


Fig. 2.7 Endpoint Mixing. Here, endpoint A assumes the role of the bridge.

conference is created. It begins as a normal two-way call, after which the conferees decide to "conference in" a third member; the endpoint that initiates the invitation assumes mixing duties. This model is a derivative of high-end digital PSTN phones that support three-way conferencing in an identical fashion [12].

2.6 Decentralized Conferencing

Up until now, only centralized bridged conferences have been considered. However, the flexibility of the IP switching fabric allows media connections to be established directly between endpoints, leading to the *decentralized* class of conferencing models. Here, each endpoint must have the ability to receive and mix $N-1$ streams.

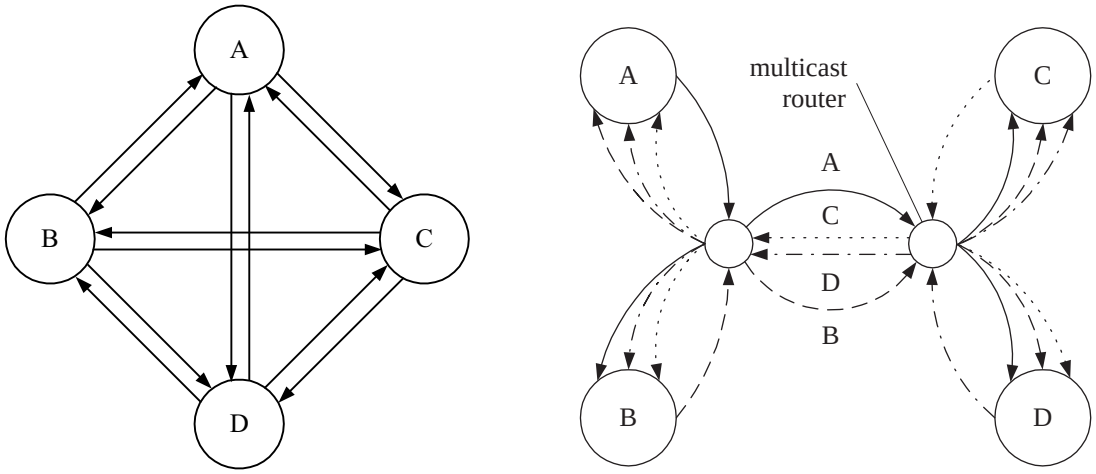
In decentralized conferences, the absence of the voice bridge automatically yields opportunity for increased speech quality. Most importantly, the tandem encoding-related speech distortions are avoided since each voice stream only travels through one encode-decode cycle. In addition, the end-to-end delay is shorter since the jitter buffers and audio bridge are also eliminated. Finally, since the $N-1$ streams arrive at each endpoint unmixed, each endpoint may *independently* provide enhanced audio processing such as 3-D audio rendering or stereo imaging. Such features yield improved intelligibility over the monaural sound of (basic) bridged conferences [77].

2.6.1 Full Mesh Conferencing

A full mesh conference earns its name from the media distribution model. Each endpoint establishes a one-to-one media connection with the $N-1$ other endpoints and distributes

copies of its speech data via multi-unicasting. The model is depicted in Fig. 2.8(a). While media is exchanged directly between conferees, conference control may or may not be provided by a centralized server [5, 9]. The latter is preferable, as will be seen in Section 2.6.3. Such an architecture is unsuitable for the PSTN since one-to-many connections are generally not supported.

To participate in a decentralized VoIP conference, each pair of endpoints must share a common codec. This is usually guaranteed since current VoIP standards require that conference endpoints be equipped with G.711. However, all media connections are one-to-one, allowing any *pair* of endpoints to communicate with the codec of their choice without affecting the other conferees. For example, if a slow connection exists between two endpoints they can switch to a lower bitrate codec in order to reduce packet loss.



(a) Full Mesh (b) Multicast

Fig. 2.8 Decentralized conferencing models. Thin or textured arrows represent single voice streams, while thick arrows represent mixed composite voice streams. In the full mesh model, endpoints send their audio three (i.e., $N-1$) times, while in the multicast model the endpoints only send their audio once.

2.6.2 Multicast Conferencing

In a multicast conference, each endpoint transmits a *single* copy of its audio to the conference multicast address and receives $N-1$ streams in return. Unlike the full mesh model,

the endpoints need not generate $N-1$ identical output streams since the multicast network appropriately replicates and forwards each stream at the nodes of the multicast tree. Thus, the model is highly bandwidth efficient and scalable to large conferences[14]. Fig. 2.8(b) exemplifies the media connections in a generic multicast conference. This model has been used for large-scale Multicast Backbone (MBONE) conferences such as the well-known NASA mission webcasts.

2.6.3 Drawbacks of the Models

To date, the decentralized conferencing model is not widely used, mainly due to the one-to-one connection model of the PSTN and a lack of widespread support for network-layer multicast in IP networks. Therefore, neither full mesh nor multicast conferencing is typically a candidate for carrier-grade conferencing systems. Nevertheless, these models avoid many of the problems related to centralized conference bridges, but are subject to many of their own:

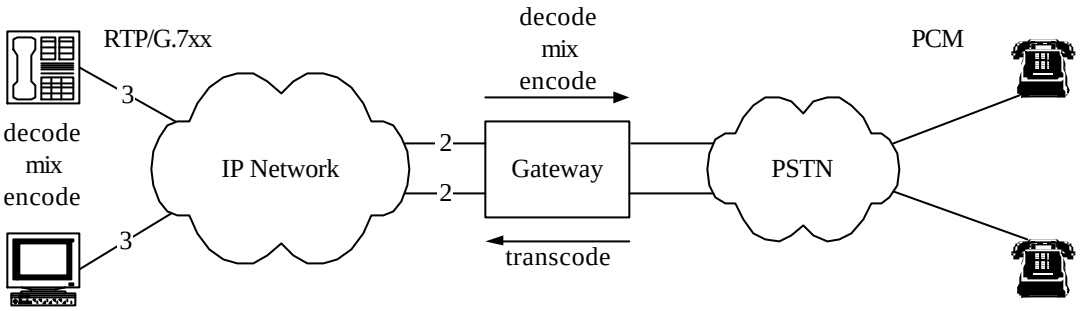


Fig. 2.9 Media connections in a full-mesh VoIP conference. This example shows the inter-networking of two VoIP endpoints and two PSTN telephones. Each line represents one full-duplex connection unless otherwise labelled. Contrary to the centralized case, there is no tandem coding with a high compression codec.

1. **Autonomous Speaker Selection:** As in the centralized case, it is desirable to attenuate or eliminate the signals of inactive conferees from the conference sum in order to reduce the background noise before playout, and to reduce the number of mix

[14] Further reduction in bandwidth is possible using a technique proposed by Forgie in [78]. Here, a novel "self-selecting" endpoint was used, which only transmitted if no packets had been received in the past 0.5 s. The technique could also be used in centralized conferences.

operations. However, unless the network delay/jitter is relatively equal between pairs of endpoints, the order of arrival for each stream will be different at each endpoint—this can lead to a different choice of current talkers between endpoints, i.e., each endpoint could hear a different version of the conference. This is obviously undesirable since it would undoubtedly confuse the conferees and inhibit the productivity of the conference.

2. **Bandwidth Consumption:** In a full mesh conference, each endpoint must be provisioned with enough bandwidth to accommodate $N-1$ output streams and $N-1$ input streams. Multicast conference endpoints require enough bandwidth for N streams. If this bandwidth is not available, periods of N-talk will suffer from packet loss. This leads to lowered intelligibility and a reduction in the pace of the conference. For wireless networks, full mesh conferencing leads to higher probabilities of call blocking and increased hand-off latencies between cells [79]. Overall, the bandwidth constraint prevents both models from being scalable to large interactive conferences.

3. **Endpoint Computational Load:** In a decentralized conference, each endpoint must have the ability to decode and mix $N-1$ incoming streams, while simultaneously encoding its own speech for output. Clearly this scheme is not scalable to large conferences, with the limiting factor being the complexity of the decode operation. In some cases, POTS phones will participate in the conference through a gateway as in Fig. 2.9. If k conferees connect through the same gateway, the computational requirements of the gateway are increased k-fold.

4. **Endpoint Homogeneity:** If the multicast model is used, *all* the endpoints must share a common codec. This will often force the endpoints to communicate with G.711, implying a high bandwidth requirement at each endpoint.

2.7 Conversational Dynamics of Voice Conferencing

The previous sections have laid a foundation for understanding the mechanics of VoIP conferencing systems. A significant portion of these discussions has focussed on voice-operated devices such as VADs and speaker selection algorithms. Design of such devices is not an exact science, and knowledge of the temporal characteristics of conversational

speech benefits their design. For example, speaker selection restricts the number of current talkers to be one or two, since it is generally assumed that most of the time a small number of conferees will actually talk. Yet, approximately how often, and how long do periods of multi-talk last? This is useful information when one wishes to know, for example, the effect of selecting two talkers instead of three. The following sections answer this question and summarize some of the relevant properties of conversational speech that have been observed during two- and multi-party teleconferences.

2.7.1 Two-Way Conversations

To study the fine structure of a two- or multi-party conversation, one typically must have the ability to recreate, by way of recordings, the flow of conversation that has occurred between conferees. In a classic study, offline processing of speech on-off patterns consisted of filling in gaps and rejecting spurts [42]. Here, a *gap* was a consecutive string of 0s, while a *spurt* was a consecutive string of 1s, both of which were produced from the *initial* VAD trace. The VAD used a fixed-threshold algorithm with 5 ms analysis frames. Next, all spurts $\leq$ 15 ms were erased and all gaps $\leq$ 200 ms were filled in, creating a new on-off pattern consisting of *talkspurts* and *pauses*. Note that this approach required a 200 ms lookahead, making it unreasonable for real-time use. However, this method produced talkspurt/pause length distributions which are a better model of human speech than a VAD with fixed hangover since the latter technique extends *every* spurt by the hangover time. It was found that these talkspurt and pause lengths were approximately exponential in nature[15].

In a follow-up study to the one described above, conversations were categorized into a set of ten events: (1) Talkspurt (defined above), (2) Pause (defined above), (3) Double talk, (4) Mutual silence, (5) Alternation silence (mutual silence between the end of one talker's talkspurt, and the beginning of a different talker's talkspurt), (6) Pause in isolation (mutual silence between the end of a talker's talkspurt, and the beginning of the *same* talker's next talkspurt), (7) Solitary talkspurt (a talkspurt that occurs entirely within the silence of the other talker), (8) Interruption, (9) Speech after interruption, and (10) Speech before interruption [81]. The study measured the durations of each of these events and presented

[15]However, subsequent studies have shown that the exponential model does not always hold—in fact, it is the VAD algorithm that changes the distributions [80]. This is obvious, since talkspurts and pauses are a function of the VAD used to produce them.

their distribution functions and corresponding means and medians. In a final effort, the author simulated the behaviour of the conversations with a six-state model tuned with empirically derived parameters [82]. This model was modified to yield a simpler four-state model which has been adopted by the ITU-T [83].

The four-state model defines its states as follows: (1) Single talk (talker A), (2) Single talk (talker B), (3) Double Talk, and (4) Mutual silence. The state holding times are assumed to be exponentially distributed while transition probabilities are fixed. The model is a steady state approximation to a real two-way conversation, which of course does not have fixed transition probabilities between states. The model assumes that the conversation, on average, will spend 0.854, 0.226, and 0.456 s in the single-talk, double-talk, and mutual silence states, respectively. The state diagram is shown in Fig. 2.10.

When this model is used to generate an artificial conversation, the resultant steady state distribution of the number of simultaneous talkers reveals that single-talk, double-talk, and mutual silence account for about 70.7, 11.2, and 18.1% of the total conversation time respectively. The model can accurately predict the talkspurt/pause distributions, but does a poor job of modelling double talk, which is generally thought to account for about 6.58% of total conversation time.

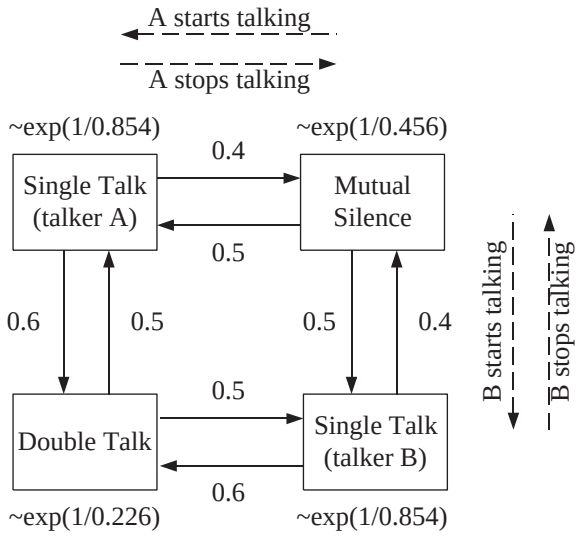


Fig. 2.10 Four-state model of a two-way conversation [83].

2.7.2 N-Way Conversations

An N-way conversation is essentially a conference with N participants ($N > 2$). The characterization and study of vocal interactions between conferees was active in the late 1970s, which was about the same time that conference bridges and speaker selection algorithms were being first developed. To date, there is no accepted model of an N-way conversation. Thus, this section merely summarizes some of the relevant attributes of such conversations.

In general, the total conference time of an N-way conversation can be divided into $N+1$ states: (1) Silence, (2) Single-talk, ..., N-talk. In terms of conference bridge design, it is advantageous to be aware of the distribution of time spent in each of these states. For example, it has been shown formally the conference can be adequately represented with two simultaneous talkers [44]. In general, periods of multi-talk account for 5–11% of total conference time. Interestingly, this percentage tends to remain fixed as the size of the conference is increased. This is because conferees tend to exhibit increased variability in speaking time as the size of the conference grows due to the dominance of a few key members. However, this distribution of talker activity may vary between conferences.

In [23], Forgie divided the periods of multi-talk into five useful categories:

1. **Reinforcement:** The intent of the speaker is to provide feedback to the primary speaker. The category is made up of short exclamations such as "yes", "no", "really", or non-speech sounds such as chuckles or groans.

2. **Collision:** This situation occurs when two or more speakers attempt to talk at the same time. After a collision, one speaker may continue talking or all colliders may stop talking and try again after a short pause. The presence of simultaneous, multiple voices is likely to affect intelligibility.

3. **Overlap:** An overlap occurs when another speaker starts talking just before the previous speaker finishes. The new speaker senses that the previous one is finishing from the context and intonation of the utterance. Overlap should not have an adverse effect on intelligibility.

4. **Interruption:** An interruption occurs when a new speaker makes a deliberate attempt to cut-off the current speaker. Intelligibility will be lost if both speakers persist. Interruptions, on average, last about 0.8 s.

5. **Noise:** Noise are events such as coughs, the rustling of paper, pencil taps etc. which are not intended to be heard by the conference.

The events can be evaluated in terms of their overall effect on the conference. In terms of conference quality, reinforcement and overlap are positive events. Reinforcement allows the current speakers to adjust to their audience, and overlap allows the conference to proceed more quickly. Interruption is another desirable feature of conferences since it prevents a chairperson or designated conferee to have absolute control over the floor. In essence, it allows the exchange of information to be accomplished naturally, as in a free-air conference. On the other hand, collisions and noise are detrimental to the overall conference quality. Successive collisions may disrupt the flow of the conversation for several seconds, while noise, as always, should be attenuated or eliminated.

2.8 Chapter Summary

This chapter has reviewed the fundamentals of VoIP conferencing systems from a transmission and speech processing perspective. Endpoints in a VoIP conference, such as a computer terminal or an MG, are based on these technologies, as are centralized conference bridges. Carrier grade conferencing service is still usually offered by way of centralized bridges. Decentralized conferencing schemes such as full mesh and multicast are possible, but scalability and control remain the limiting factors. The next chapter presents an alternative conferencing model which combines aspects of both the centralized and decentralized architectures.

Chapter 3

Design Solutions in VoIP Conferencing

This chapter investigates the design options and solutions for the Tandem-Free Operation (TFO) conferencing model proposed by Simard and Rabipour. Recall that the need for the model is motivated by the speech quality impairments inherent to VoIP conference bridges, due to the tandem arrangement of speech codecs. Full mesh and multicast conferencing models eliminate this problem, but lack strong control and synchronization, and impose potentially high requirements on the endpoints in terms of bandwidth and computational power.

The remainder of the chapter is structured as follows. Section 3.1 re-introduces the TFO conferencing model in terms of a high-level description of its media connection model. Thus, Section 3.1.1 continues with a description of the TFB and the TFO endpoint. Section 3.2 briefly outlines the requirements of the TFB VAD, which is used as a front-end to the speaker selection module. Section 3.3 concentrates on the motivation, design, and test of the Tandem-Free Speaker Selection (TFSS) algorithm. The algorithm is compared against other selection algorithms which have been proposed for both packet and PCM bridges. Drawing on the findings of the previous sections, Section 3.4 outlines an RTP payload format for transporting the side-information, i.e., the TFO data frames, from the endpoints to the bridge. The details of the TFB's RTP connection model and header/report processing are deferred to Section A.1.2.

3.1 Tandem-Free Operation Conferencing

In short, the TFO model uses centralized control (speaker selection) and decentralized decoding and mixing. For a conference with N participants, each endpoint transmits its stream to the conference bridge, but receives M-out-of-N streams in return. The bridge performs speaker selection on a per-packet basis, selectively forwarding the packets of the M selected talkers to the endpoints, while discarding the remaining $N - M$ packets. Each endpoint appends side-information containing speech activity information to each outbound packet. As in a decentralized conference, each endpoint must be capable of simultaneously receiving, decoding, and mixing multiple streams of speech. In Simard's TFO model, M is always set to two, while Rabipour suggests two *or* three. For the remainder of this thesis, only the primary and secondary talkers will be considered, i.e., $M = 2$. A high-level connection and flow diagram is shown in Fig. 3.1.

Fig. 3.1 shows that the endpoints connect to the bridge in a star configuration, which is advantageous for control and synchronization. The model does not look much different than Forgie's SI model, other than increasing the number of simultaneous speakers from one to two. However, the fundamental difference is the inclusion of side-information in each upstream packet for the purpose of speaker selection (the side-information will be defined in a later section). Partial- or full-decoding is not required for feature parameter extraction since such information is explicitly carried as side-information. Now the bridge is not only tandem-free, but is also codec-independent.

3.1.1 TFO Bridge Overview

A functional depiction of the Tandem-Free Operation Bridge (TFB) is shown in Fig. 3.2. The receive/synchronize module is responsible for the basic duties such as RTP packet reception and endpoint-bridge synchronization as outlined in [13–15]. The packet generation time is mapped to the local TFB clock, accounting for the transit delay and network jitter. Next, the packet and its corresponding local time is passed to the state updater.

The speaker selection unit maintains a state for each conferee, which essentially consists of a priority rank and algorithm-specific metrics such as average speech energy, talkspurt state, and/or a VAD decision. Upon receipt of a packet, the speaker selection module updates the state of the corresponding conferee, and then performs speaker selection against the states of the other conferees for the same time interval. This module marks the packet

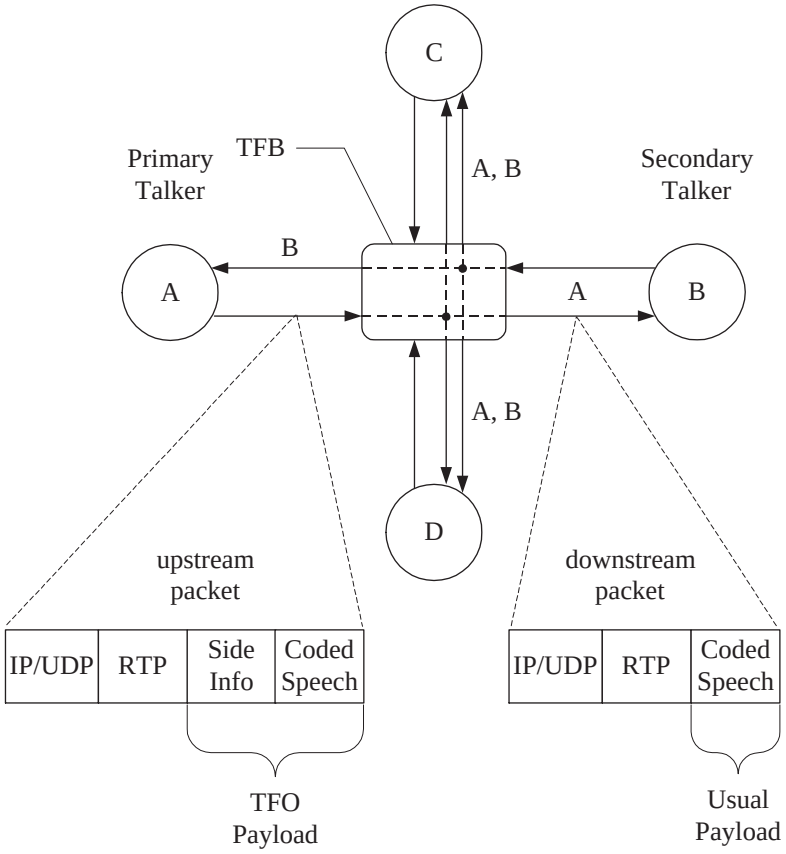


Fig. 3.1 TFO conferencing model with two simultaneous talkers (i.e., $M = 2$). Conferees A–D are connected to the TFB, but only conferees A and B have been granted talking privileges. Therefore, A receives packets from B, B receives packets from A, while C and D receive packets from both A and B. Packets from C and D are discarded at the TFB. Side-information is sent to TFB in each packet, where it is used by the speaker selection algorithm. The TFB may remove this field from the packets before they are redistributed to the endpoints.

as either selected or not-selected and passes it (and the selection decision) to the rate controller.

As will often happen due to jitter, the state of the current conferee will "lead" or "lag" those of the other conferees by some amount of time. This is because the corresponding state of another conferee is unknown since a packet had not yet been received for that interval, or because the conferee stopped talking. In such a case, the last known state is re-used; the updated state and the packet is then passed to the speaker selection module.

The job of the rate controller is to ensure that, at most, one packet per-selected talker leaves the TFB for every time interval. The TFB processes packets without delay or de-jittering. Hence, the case may occur where the packet of a loud, interrupting conferee arrives and is selected *after* the packets of the two current talkers have already been received and forwarded (for the same time interval). In such a case, the rate-controller discards the packet.

The RTP state is updated to reflect the number of packets actually transmitted as opposed to the number received. At this point, the sequence number and marker bit of the RTP header need to be updated. If the received packet was an RTCP packet, then the Sender Report (SR) or Receiver Report (RR) records must also be modified. Once this is complete, a copy of the selected packet is sent to the other $N-1$ endpoints. As indicated in Fig. 3.1, the side-information may be removed before the packet is forwarded. These translations are described further in Section A.1.2.

3.1.2 TFO Endpoint Overview

To function in a TFO conferencing arrangement, the endpoint must satisfy two mandatory requirements. As mentioned above, the endpoint must have the capability of simultaneously receiving, decoding, and mixing M streams. Most workstation-based conferencing tools, such as RAT, already have this capability. The second requirement is that each endpoint support transmission of TFO payload, while receiving a (typical) speech codec payload. In the case where an endpoint is connecting through a VoIP gateway, the gateway must support the behaviour of TFO compliant endpoints.

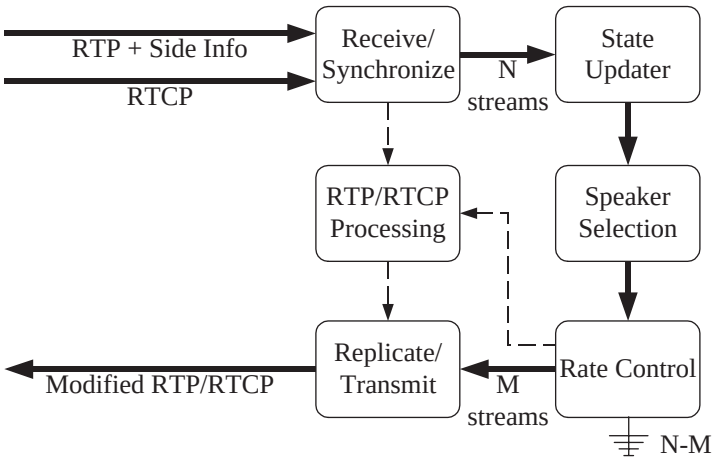


Fig. 3.2 Components and data flow of the TFB. RTP streams arrive at the bridge and the usual synchronization takes place. If necessary, conferee states are predicted before the speaker selection algorithm is executed. Selected packets are forwarded to the rate controller, where it is ensured that at most two packets leave the bridge per packet interval. The RTP/RTCP states are adjusted according to the output of the selection algorithm and the rate control, and then used to update the RTP packet headers and the RTCP SR and RR records.

3.1.3 TFO Advantages

Some of the advantages of the proposed TFO system has already been mentioned in the preceding sections. Before proceeding with the design options and details, it is favourable to summarize some of the obvious advantages and disadvantages of the system, with respect to conventional VoIP conferencing models:

1. **Speech Quality:** By design, the TFO system provides superior speech quality compared to the centralized VoIP conferencing bridge by avoiding tandem coding of a multi-speaker signal. This is a problem that has been traditionally addressed with ad-hoc, system-specific solutions.

2. **Bandwidth Scalability:** Since only a maximum fixed number of packets, M, are forwarded to the endpoints per packet interval, the worst-case bandwidth consumption per-link is independent of the size of the conference, and is $O(M + 1)$, where typically $M = 2$.

3. **Codec Independent:** The TFB executes the select-forward operation based solely

on the contents of the TFO payload, i.e., no speech codecs are needed on-board the TFB.

4. **Reduced Bridge Complexity:** Since the TFB replaces the computationally expensive speech codecs and mixing module with a comparatively low-complexity speaker selection algorithm, the bridge can be built independent of DSPs. Large conferences no longer have to be distributed across multiple bridges, eliminating the need for "server farms".

5. **Reduced Delay:** Since no mixing is performed on the TFB, the jitter buffer and codec latencies are eliminated, reducing the end-to-end delay. This is in contrast to a conventional VoIP bridge from Radvision which adds 165–200 ms of latency to the overall end-to-end delay when compressed speech is used [10].

6. **Protocol Compliance:** Integration with existing protocols is simple. A new RTP payload type will be defined for transport of the TFO payload, which is entirely within the framework of RTP. The use of "TFO mode" can be signalled during conference initiation.

7. **Endpoint Support:** Many workstation-based conferencing tools such as RAT support multiple-stream decoding and mixing.

8. **Centralized Control:** Although centralized control schemes are always subject to being a single-point of failure[1], they are more attractive than their decentralized counterpart in terms of management. The TFB conforms to this traditional practice.

3.1.4 TFO Disadvantages

The disadvantages of the TFO scheme are related to the *endpoints*, and manifest themselves in terms of integration with existing VoIP protocols and practice:

1. **Common Codecs:** As in a multicast conference, *all* TFO endpoints must share a common codec. This may result in the use of G.711, in which case the tandem-bypass is not essential to maintain high speech quality. Nevertheless, TFO conferencing with

[1] Traditional PSTN switching equipment addresses this issue by running a back-up switch in tandem with the live switch. In the event of a failure, a Switch-of-Activity (SWACT) occurs, and the back-up system takes over.

G.711 gives advantages in terms of reduced delay and complexity over centralized conferencing, and bandwidth scalability over decentralized conferencing.

2. **Endpoint Support:** In the near future, the main use of VoIP will be to carry regular PSTN traffic originating from the local loop. This implies that the burden of the final decode-mix will be placed on the VoIP gateway, and not directly on the user terminal. This requires that gateway vendors support the TFO model. In addition, standard issue IP and mobile phones only have the ability to receive a single stream.

3. **Increased Endpoint Complexity:** The computational complexity of a conference endpoint is raised by the additional decode-mix operation.

4. **Increased Bandwidth:** The bitrate of the upstream link is (slightly) raised due to the inclusion of the side-information. On the downstream link, the bitrate is raised due by the presence of one additional RTP stream.

3.2 TFB VAD Design

A VAD is an important part of a conference bridge, as it identifies those conferees that are speaking (or transmitting). In legacy conference bridges, VADs are typically used as front-ends to the speaker selection module. For this reason, a VAD is not explicitly shown in Fig. 3.2. This section outlines the VAD options for the three possible transmission scenarios which may occur in conventional VoIP systems. Note that this thesis did not develop a VAD from scratch—instead, requirements were identified which allowed selection of an existing VAD.

3.2.1 Requirements

Since the VAD features are sent to the TFB as side-information, the type of VAD must be chosen such that the bitrate of the side-information is small, and the features are easily computed at the endpoint (i.e., low complexity). On the other hand, if the conference endpoints are using silence suppression, then the VAD is not required since the arrival of a packet is an implicit indication of the voice activity. Further, if the endpoints are using a DTX/CNG scheme such as G.729B, the first frame of the silence period is explicitly signalled with an SID frame. This implies two modes of operation shown in Fig. 3.3. Note

that when endpoints do not use silence suppression/DTX, VAD features *must* be sent to the TFB since the RTP marker bit will always be zero.

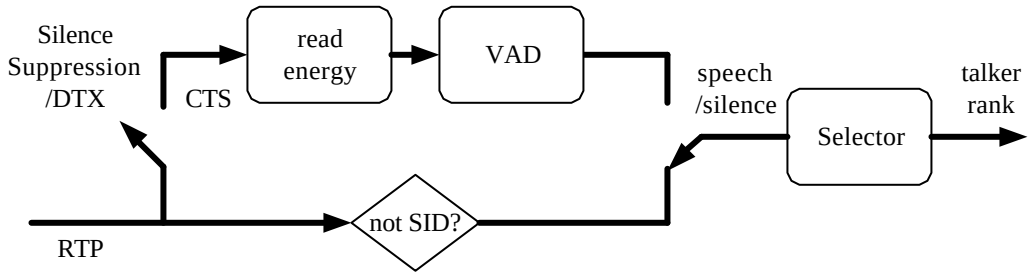


Fig. 3.3 TFB VAD options. If the endpoints are using silence suppression/DTX, the VAD is not required. Otherwise the VAD is required to identify active talkers.

3.2.2 TFO VAD Options

This section presents three options for obtaining a VAD decision at the TFB. In essence, the options depend on the capabilities of the endpoint, i.e., whether or not a VAD, or silence suppression/DTX, is supported. Thus, the options are derived from the transmission scenario, described below.

Scenario 1: Transmission of VAD features

When silence suppression/DTX is not used, endpoints transmit RTP packets to the TFB according to the packetization interval—this can be 10–80 ms. The side-information must contain features that are sufficient for both the VAD and speaker selection algorithms. If an energy-based VAD is used (see Section 2.3.2), then the same side-information can be used for both the VAD and speaker selection modules. Energy-based VADs perform reasonably well in high Signal-to-Noise Ratio (SNR) conditions and are often found in conferencing arrangements since conferees are usually seated in quiet rooms or offices.

In terms of speaker selection, the robustness of the VAD is dependent on whether the selection algorithm is time- or level-based. In the time-based case, the VAD must faithfully discern between speech and silent periods, since current talkers are selected solely on the basis of VAD decision. This requirement is best illustrated with an example of a single-talker FCFS system. In this case, traditional VADs with static thresholds—as were used

on old PSTN bridges—will constantly select a conferee with high background noise. The FCFS nature of the algorithm disallows other conferees from being heard.

Speaker selection with the time-level criterion does not depend as much on the robustness of the front-end VAD, since conferees are selected when they speak louder than another *active* conferee. Going back to the example in the previous paragraph, the conferee with the high background noise can be interrupted by another, louder conferee. Hence, unlike the time-based approach, an adaptive energy-based VAD is sufficient for the TFB.

An energy-based VAD is also an attractive alternative given the ease with which the energy can be extracted and coded. The well-known definition of the *normalized* full-band speech frame energy is

$$\bar{E}_f = \frac{1}{L}\sum_{n=1}^{L} s(n)^2, \tag{3.1}$$

where s is the speech signal, and L is the frame length (in samples). $\bar{E}_f$ can be coded with few bits, yielding little distortion even at low SNRs ($\approx$ 10 dB). An alternative is to enhance the VAD decision by inspecting both the low- and high-band energies [46, 47]. In this case, a low complexity calculation of the (normalized) band energies, $\bar{E}_l$ and $\bar{E}_h$, can be accomplished with a single filtering operation, instead of two. For example, $\bar{E}_l$ is first obtained as

$$\bar{E}_l = \frac{1}{L}\sum_{n=1}^{L} \left(s(n) * h(n)\right)^2, \tag{3.2}$$

where h_n is a low-pass filter typically in the range of 0–1 kHz. Next, the high-band energy, $\bar{E}_h$, can be approximated as

$$\bar{E}_h \approx \max\left(\bar{E}_f - \bar{E}_l, 0\right). \tag{3.3}$$

The max operator is used to compensate for the pass-band ripples of h_n, which may result in $\bar{E}_l > \bar{E}_f$ during periods of voiced speech. Note that a high complexity, multi-feature VAD such as G.729B cannot be adopted by the TFO model without a substantial increase of the bitrate on the upstream link.

Scenario 2: Transmission of the VAD Decision

Still another option is to host the VAD on the endpoint and include the binary VAD decision as part of the side-information. In this case, a "V-bit" is sent as part of the TFO side-information. The clear advantage here is that the choice of VAD is not limited by the

size of its feature set, since it is only the resultant VAD decision that is transmitted. This allows for a better VAD to be used, such as G.729B.

Scenario 3: Silence Suppression/DTX Mode

The use of silence suppression or a DTX/CNG scheme foregoes the need for a VAD feature to be present in the TFO payload. Since packets are only transmitted to the TFB during periods of speech (as determined by the local VAD), the TFB has an implicit indication of speech activity whenever a packet arrives from a given conferee. The problem with this scenario is that the TFB does not have an explicit indication of *silences*, since no packets will arrive until the beginning of the *next* talkspurt. In this case, the TFB sets a wait deadline based on the short-term transit jitter characteristics associated with the given conferee, and marks the conferee as inactive once this deadline expires. However, this presents a problem when trying to distinguish between silent periods and late packets.

The above problem disappears when a DTX/CNG scheme such as G.729B is used. In this (and equivalent) schemes, an SID frame is *always* sent at the beginning of the silence, providing an explicit indication of the end of the talkspurt. Hence, the arrival of an SID frame allows the TFB to mark the given conferee as inactive.

3.2.3 TFB VAD Algorithm

To summarize thus far, an energy-based VAD is an adequate choice for the TFB VAD for the following reasons:

1. For high SNR conditions typically found in conferencing scenarios, the performance of energy-based VADs is comparable to higher complexity VADs.

2. Energy is easily calculated and coded, thus easily adopted by existing VoIP end-points/clients.

3. The same energy parameter(s) can be used as input to the ensuing speaker selection module. Thus, by choosing the energy as the side-information, all three of the aforementioned scenarios can be accomplished, *plus* speaker selection.

It was discerned that the RAT VAD met the above requirements. However, it was modified before being integrated into the TFB. The original RAT VAD relied on a 60 ms

pre-hang, or look-ahead, in order to make better decisions. This pre-hang was reset to 20 ms, meaning that VAD decisions were not delayed when a 20 ms packetization interval was used. This allowed short bursts of noise to trigger the VAD, creating a 200 ms talkspurt. Therefore, the hangover mechanism was changed to be equal to the length of the talkspurt with a minimum duration of 20 ms, and a maximum of 200 ms—a similar technique was used in [12]. Therefore, in the case of very short talkspurts, or bursts of noise, a very short hangover was applied.

3.3 TFB Speaker Selection Design

As described in Chapter 2, speaker selection has long been performed in centralized conferencing arrangements. In essence, a speaker selection algorithm is simply a two-step process comprised of feature extraction, and a comparison and ranking between conferees. Example features are VAD decisions and/or speech energy. The algorithm then uses these features to update the state of the current conferee, after which a comparison and ranking (selection) is performed. The success of the technique relies on the fact that periods of multi-talk account for a small portion of total conference time. This section outlines the motivation and design of the TFSS algorithm developed for this thesis. The benefits of the algorithm are investigated through relative comparisons against traditional time- and level-based algorithms.

3.3.1 Motivation for a New Algorithm

It is generally assumed that periods of double- or multi-talk will account for 5–11% of the total conference time [23, 44]. However, empirical evidence with four conferees gathered during the testing phase of this thesis (see Chapter 4) indicates that this assumption does not always hold. In fact, using two different VADs to identify talkspurts and pauses, it was found that about 40% of the conference time was spent in a multi-talk state. The amount of time spent in a triple- or quadruple-talk state was found to account for $< 10\%$ of total conference time. The complete state distribution, obtained with both the G.729B and TFB VADs, is shown in Fig. 3.4.

The high occupancy rate of state two indicates that $M = 2$ is a good choice for the number of selected talkers. This claim is supported by findings presented in [44]. Considering the G.729B distribution to be ideal, it is clear from state 0 that the energy-based TFB

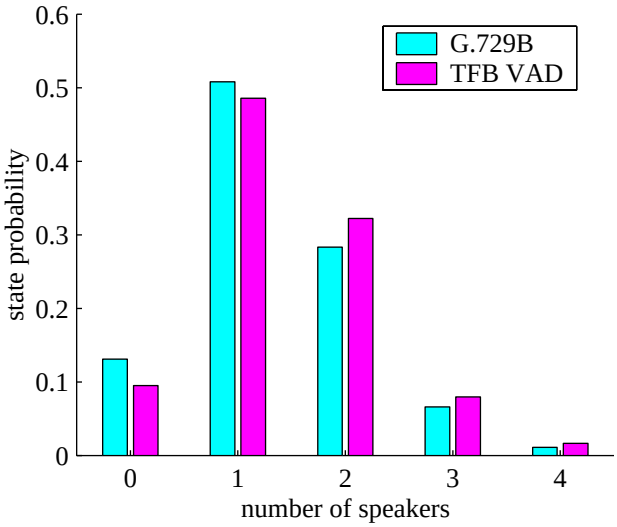


Fig. 3.4 State distribution of a four-way conversation, as indicated by the G.729B and TFB VADs. The mean speech activity factor of the four conferees was 26.4, which implies that they spoke as often as they would in a normal, two-way conversation (conversation speech has an activity factor of 27.6 [83]).

VAD is not as accurate when detecting silences, mainly due to its longer hangover. This shows the need for the speaker selection algorithm to use *both* $\bar{E}_f$ and the VAD decision to determine the current talkers.

3.3.2 Requirements

In the end, the speech quality of the system is the strongest factor in determining the system's overall performance [23]. Thus, much of the success of the TFO model is dependent on the speaker selection algorithm. For the model to be a viable alternative, the speaker selection algorithm must satisfy the following requirements:

1. Number of Speakers: No more than two-of-N streams ($M = 2$) will be switched back to the endpoints. This implies that talking privileges will be assigned to a *primary* and *secondary* speaker. Empirical data and practice have shown $M = 2$ to be a satisfactory number of current talkers.

2. Audible Switching: The speaker selection process should not introduce any audible artifacts, such as speech clipping, into the composite output signal generated at the endpoints.

3. Interactivity: Further to the previous point, the speaker selection process should appear seamless to the conferees. Events such as reinforcement, interruption, and overlap should be preserved, while detrimental events such as collisions should be controlled[2].

4. Noise Protection: The algorithm should possess some degree of immunity to low-level, or spurious noise. The subjective effect of differences in the nature of the background noise *between* conferees is beyond the scope of this thesis.

5. Generic Operation: The algorithm should not depend on the session parameters (e.g., speech codec, silence suppression/DTX).

Given these requirements, a high-level description of the speaker selection module can be envisioned. Given that it is the most flexible, the time-level criterion is adopted for use in the TFB. The following briefly lists some of the implications of this approach in terms of the switching requirements:

- Speaker selection will be accomplished with a time-level approach. This implies that the selection module will be comprised of a VAD and energy comparison. The VAD provides additional resilience against noise, and reduces the number of comparisons by the selector. Ultimately, conferees are selected on the basis of "loudness".

- Being a derivative of level-based selection, the time-level approach permits the occurrence of natural conversational events such as overlap and interruption. It also exposes the algorithm to detrimental effects such as collisions. Therefore, the interruption capability needs to be controlled. This can be accomplished by way of a simple "barge-in" threshold which adds a hysteresis to the system, preventing rapid switching between conferees.

- The front-end VAD provides some degree of noise immunity; however, more energetic bursts of noise occur occasionally, and will be classified as speech by the VAD. Therefore, the selection algorithm requires a mechanism that will allow short bursts of noise energy to be deemphasized.

[2]These events were defined in Section 2.7.2.

Within the scope of these requirements, there are still several degrees of freedom. For example, what features should be chosen as side-information in order to perform both the VAD and selection functions? How does one prevent rapid switching between conferees while using a level-based approach? These questions are addressed in the following sections.

3.3.3 Tandem-Free Speaker Selection Algorithm

The Tandem-Free Speaker Selection (TFSS) algorithm is a novel speaker selection technique developed for this thesis. The algorithm assigns talking privileges based on a running estimate of the average speech energy of each conferee, plus a barge-in threshold. The running average is updated at one of two rates, depending on talkspurt state. Recalling the conversational dynamics reviewed in Section 2.7.2, the speech on-off patterns are divided into one of six states which have the following meanings:

1. **Idle:** The conferee is silent.
2. **Entry:** The conferee has started to talk following a period of silence.
3. **Short Hangover:** After starting to talk, the conferee immediately stopped talking. Collision, Reinforcement, Interruption, or Noise has occurred.
4. **Bridged:** The conferee has talked continuously for at least 0.86 seconds.
5. **Long Hangover:** The conferee has paused while talking. Either an articulation pause has occurred, or the conferee has been interrupted.
6. **Short Entry:** The conferee has resumed talking after a brief pause (< 1.56 s).

The states are partially based on a legacy AT&T speaker selection algorithm previously described in Section 2.5.2, as well as [4]. The AT&T algorithm, by inspection, was determined to use an ad-hoc method for assigning speaker priority, presumably due to lack of computing power in the early 1980s. Some of the drawbacks are outlined below.

Old Algorithm

One downfall of the AT&T algorithm is that priority is dominated by state, with only a minor influence from the speech level. This was due to the layout of the 16-bit Adjusted

Priority Level (APL) constant, whose value was used to assign talking privileges. Since the state weights formed the upper 8-bits of the APL, a conferee in a lower priority state could not interrupt a conferee in a higher priority state based on his/her relative loudness. Therefore, speech level only influenced speaker selection for conferees that were in the *same* state.

Further, in the case where $M = 1$, a situation could easily occur where a new conferee wishes to interrupt the current talker who was holding the highest priority state. The new conferee was required to continue talking for 0.5–1 s before they had a chance to interrupt. If the interruption was successful, the beginning of his/her talkspurt would be heavily clipped, leading to a loss in intelligibility.

Abrupt switching as in FCFS could also occur when the speech level of a high priority conferee tapered off at the end of a talkspurt, *before* the VAD declared silence. Following an active-to-inactive transition, an active speaker in a lower priority state could suddenly become selected, resulting in an undesirable and abrupt change in speech level.

TFSS Algorithm

The state hierarchy of the AT&T algorithm limited barge-in capability, and created the potential for abrupt speech level contrasts following a switch of current talkers. Due to these problems, the original assignment of priority was abandoned in favour of a new approach described below.

The running estimate of the average speech energy, or the speech envelope, ε_n, of each conferee is updated once per-frame with the $\bar{E}_f$. The envelope calculation is essentially a smoothing of the energy track accomplished with one of two exponential first-order IIR filters. One filter is used for front-end talkspurt smoothing while the other is used for periods of mid-speech and hangover. The basic form of the filter is

$$\varepsilon_n = \beta\varepsilon_{n-1} + (1-\beta)\bar{E}_f, \tag{3.4}$$

where ε_n is the energy "envelope", and $\bar{E}_f$ is the speech energy of the nth frame. The filter parameter, β, is obtained as usual from

$$\beta = e^{-t/\tau}, \tag{3.5}$$

where t is the speech frame duration, and τ is the time constant of the exponential averaging.

Priority is assigned via a combination of order-of-activity and ε_n. Following a period of silence, the first two conferees to become active are assigned the two highest priority slots and are given talking privileges. A new, interrupting conferee is moved up in priority given that his/her ε_n exceeds the ε_n of a higher priority conferee by a barge-in threshold, B_{th}, of 3.3 dB. Then, the interrupter is assigned a priority ahead of the interrupted conferee, effectively demoting all lower priority conferees by one position. At the end of each evaluation interval, the two conferees with the greatest priority are assigned talking privileges.

The current state of the algorithm determines the amount of smoothing. The front-ends of talkspurts are smoothed less than periods of mid-speech and hangover, by choosing τ to be 0.04 and 0.08 s, respectively. This allows ε_n to have a fast rise and slow decay characteristic. The slower decay at speech offset provides enough hangover to smooth out articulation pauses. Note that for periods of hangover, the second term in Eq. (3.4) vanishes since $\bar{E}_f = 0$.

As previously stated, TFSS more-or-less retains the states and transitions of the original AT&T algorithm, but discards the use of state-assigned priority. Rather, the state machine is used to detect and classify the beginning, middle, and ends of each talkspurt such that the desired τ could be used to update ε_n. Since a fast rise was desired following speech onset, an additional state, Short Entry (ES), was added between the original Long Hangover (HL)-to-Bridged (BA) transition so that the short time constant could be used. The holding time of ES is equal to the length of time spent in the Long Hangover (HL) state. Finally, the state holding times were changed to fit the observed distributions of talkspurts and pauses produced by the modified TFB VAD. The means and medians of these distributions are shown in Table 3.1. The remainder of the section describes the TFSS state transition diagram shown in Fig. 3.5.

TFSS State Transitions

As in the original algorithm, inactive conferees remain in the Idle (II) state until the VAD detects speech, at which time they are moved to the Entry (EA) state and ε_n is updated with $\tau = 0.08$ s. If the conferee remains active for 0.86 s (T_E), they are moved to the Bridged (BA) state otherwise they move to the Short Hangover (HS) state. Note that

T_E corresponds to the mean talkspurt length of 0.865 s from Table 3.1. While in Short Hangover (HS), ε_n is decayed exponentially with $\tau = 0.08$ s, and the holding time, t_s, is equal to the amount of time the conferee spends in the Entry (EA) state. Therefore, conferees that emit very short bursts of speech are timed out of HS equally as fast so that they do not compete with the other conferees for talking privileges.

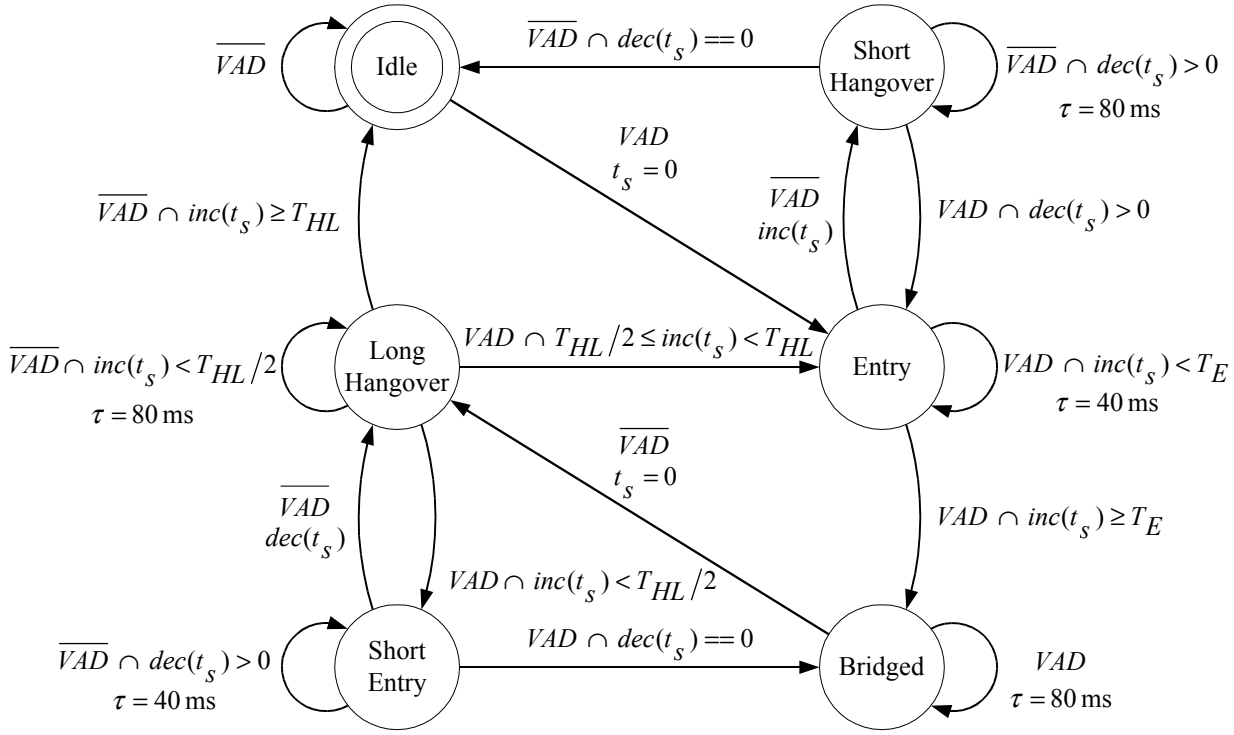


Fig. 3.5 TFSS algorithm state transition diagram, derived from [4]. The variable t_s denotes the time spent in a given state, and is incremented or decremented by the speech frame duration.

The BA state lasts as long as the conferee remains active. Once speech offset is detected, the conferee is moved to the HL state which provides hangover by decaying ε_n exponentially with $\tau = 0.08$ s. The maximum holding time of HL is 1.56 s (T_{HL}), which was derived from the mean pause length of 1.555 s shown in Table 3.1. However, if the conferee begins talking within $T_{HL}/2$ s of assuming HL, he/she is moved to the new ES state and ε_n is updated with $\tau = 0.04$ s. The conferees that remain active for the duration of ES are moved back to BA, otherwise they return to HL. Thus, the combination of the HL and EA states bridge articulation pauses. In the case where a conferee becomes active while in

HL after $T_{HL}/2$ s, he/she is moved back to the EA state. Conferees are considered to be finished talking once their state time reaches or exceeds T_{HL}.

3.3.4 Other Algorithms

This section outlines three existing speaker selection algorithms which may be found in both PCM and packet bridges. The chosen algorithms are a representative set of time-, level-, and time-level based techniques which have been previously described in Section 2.5.2. The algorithms are not *exact* implementations of formerly used algorithms, but are generalizations of a particular class.

Within a class of algorithm, differences usually exist in terms of the analysis frame length (e.g., 0.125–20 ms), signal power computation (e.g., average sum-of-squares vs. average absolute amplitude), or the front-end VAD. Such parameters have been fixed across each of the following algorithms. For example, the TFB VAD was used to provide a VAD decision to the selection algorithm, and the number of current talkers was set to two. The switching interval was fixed at 20 ms in order to coincide with common speech coder frame lengths and RTP packetization intervals.

Algorithm 1: First-Come-First-Served

The First-Come-First-Served (FCFS) technique is a popular choice for packet-based systems, such as [23, 34–36, 84]. The algorithm presented here was mainly derived from [12, 23]. As is customary, FCFS assigns talking privileges based on the order that the conferees talk or become active. The first two conferees that become active following a period of silence are assigned talking privileges. Active conferees remain bridged until they become inactive, at which point talking privileges are reassigned to the next active conferee, even if he/she is mid-talkspurt. The algorithm sets the most recently active conferee as the primary interrupter.

Algorithm 2: Loudest Talker

The Loudest Talker (LT) technique is the traditional choice for PCM-based systems, some of which have been described in [28, 29, 31]. Notably, a single-talker LT algorithm is proposed for an ATM packet-bridge in [74]. In this thesis, the LT algorithm is a pure level-based approach, which selects the speech frames of the two loudest talkers for output every

20 ms. Loudness is the average absolute speech level per-frame. If more than two conferees had the same speech level, then the most recently selected conferees were reselected.

Algorithm 3: Northern Telecom

The so-called Northern Telecom (NT) algorithm is the second of two time-level selection algorithms previously described in Section 2.5.2, and in [30]. Intended for PCM bridges, the original algorithm tracked the speech envelope in the μ-law domain. In this thesis, the algorithm tracks the envelope, $\hat{s}$, of the average absolute signal level, $\bar{s}$, using

$$\begin{aligned} &\text{if} \quad (\bar{s} \geq \hat{s}) \\ &\qquad \hat{s} = \bar{s} \\ &\text{else} \\ &\qquad \hat{s} = \hat{s}e^{t/\tau}, \end{aligned}$$

where t is the frame duration (20 ms), and $\tau = 46.24$ ms as specified in [30]. At the end of each evaluation interval, the two conferees with the greatest $\hat{s}$ are assigned talking privileges. In the case of a tie, the most recently selected conferee is reselected.

3.3.5 Performance Criteria

To date, there is a scant number of methodologies used to objectively characterize the performance of a speaker selection algorithm. Forgie presented measures such as time between interruptions or time spent in various states, but also concluded that the utility of the data was questionable [23]. Another study measured the transparency of an algorithm as the proportion of time a given conferee was "serviced", or heard [44]—this measure is only useful when applied to speech data collected from a live evaluation of the system in question, since the system itself influences the way in which the conferees interact. Since there was no well-known technique for evaluating differences between two talker systems, speech clipping metrics were used.

Speaker selection inevitably clips the speech of each conferee. However, in this context the subjective impact of clipping is not as severe as shown in Table 2.2 and Table 2.3, since clipped regions of speech from one conferee are usually "replaced" with the speech from another conferee. That is, there is no period of silence due to the clip, since the only time

a change of talkers occurs is when an interrupter is already active.

Subjective testing has shown that FEC and Back-End Clipping (BEC)[3] is less noticeable than Mid-speech Burst Clipping (MSC) [43]. Therefore, in the context of speaker selection, it is desirable for a selection algorithm to reduce MSC as much as possible. Of course, since clipping is unavoidable, a reduction in MSC implies an increase in FEC. The goal is to balance the two such that neither has too much of an adverse affect on the speech quality. In addition, there is a trade-off between the length of the clip and the clipping frequency. According to Eq. (2.1), frequent clipping implies shorter clips, while less frequent switching implies longer clips. From a speaker selection standpoint, it is more desirable to have less frequent, longer MSC and more frequent, shorter FEC.

3.3.6 Simulations

Four-member conferences with speaker selection were simulated using the new TFSS algorithm and the three other algorithms. Since there is no accepted model of a conference between more than two persons, the conferencing facility was used to collect speech traces which could be used as input for the simulations. To collect the data, a small feature was added to RAT that saved the microphone input of each station to a file during a conference. The use of raw speech data allowed for both quantitative and qualitative performance analysis of the selection algorithm under test.

Effect on Talkspurt and Pause Distributions

The means and medians of the talkspurts and pauses produced by the TFB VAD are presented in the first row of Table 3.1. These measurements were taken on individual speech traces *before* any sort of speaker selection was performed. Since speaker selection only permits two conferees to be heard at any time, portions of active speech will be clipped. Thus, it is interesting to investigate how well a selection algorithm preserves the original talkspurts and pauses of an individual conferee. If a speaker selection algorithm comes close to preserving the original distribution, it is probably doing a good job.

The resulting means and medians of the talkspurts and pauses due to speaker selection are shown in rows 2–5 of Table 3.1. FCFS results in an increase in the mean and median

[3]This thesis makes a distinction between MSC and BEC. Note that a BEC occurrence starts after the beginning of a talkspurt and continues until its completion, where a MSC occurrence both starts and stops somewhere in the middle of the talkspurt.

because of its no barge-in policy; once a conferee has acquired the bridge, they cannot be pre-empted and short bursts of speech from the would-be interrupters are eliminated. However, LT clearly "chops up" the speech by a shortening both the talkspurts and pauses by about 30 and 50%, respectively.

Table 3.1 Averages of means and medians of durations of talkspurts and pauses (in seconds) after speaker selection.

Method	Pauses				Talkspurts			
	Mean	σ	Median	σ	Mean	σ	Median	σ
Original	1.555	0.37	0.693	0.142	0.865	0.065	0.608	0.060
FCFS	1.843	0.49	0.923	0.135	0.897	0.105	0.633	0.07
LT	1.073	0.133	0.255	0.076	0.463	0.116	0.258	0.045
NT	1.537	0.322	0.653	0.119	0.753	0.046	0.54	0.067
TFSS	1.639	0.357	0.738	0.132	0.8	0.063	0.56	0.057

Effect of Speech Clipping

Table 3.2 shows the clipping performance for each of the four selection algorithms. The nature of FCFS and LT lead to expected results: FCFS causes relatively long FEC in comparison to the other algorithms. Note that no other clipping occurs since FCFS does not allow interruptions. Although LT results in shorter clips, the frequencies of MSC and BEC are intolerable. Results for NT and TFSS are close in most categories, especially in terms of FEC performance: NT yields less clipping by nearly one frame (19 ms) and gives a 5% reduction in clip occurrence. However, the 38% reduction in MSC occurrence given by TFSS over NT is desirable since it indicates less switching back-and-forth between speakers. In comparison to NT, TFSS only decreases the occurrence of BEC by 12%.

3.3.7 Algorithm Comparison

The previous section highlighted certain advantages and disadvantages of the four algorithms tested. For example, the lack of barge-in of FCFS is apparent due to the long durations of FEC, while the lack of history in LT is demonstrated by the high rates of MSC and BEC. Clearly, the two time-level based algorithms, namely NT and TFSS, consistently outperform the two other approaches.

Table 3.2 Speech clipping due to speaker selection. The results were obtained by comparing the selected speech after speaker selection to the original speech on-off patterns for each individual conferee. Clipping performance was computed on an individual basis and then averaged across all conferees.

Region	Method	Speech Clipping Duration L (s)	Proportion of Speech Clipped P (%)	Frequency of Clip Occurrence F (clips/min)
Front	FCFS	0.300	5.4	3.81
	LT	0.064	0.5	1.79
	NT	0.075	0.8	2.30
	TFSS	0.094	1.1	2.43
Middle	LT	0.076	5.2	15.36
	NT	0.126	1.7	2.95
	TFSS	0.224	1.8	1.81
Back	LT	0.167	10.9	14.50
	NT	0.198	4.4	4.71
	TFSS	0.180	3.5	4.14

In terms of FEC, both NT and TFSS are nearly equivalent. However, the small increase in FEC is negligible when compared to TFSS's reduction in MSC and BEC occurrences. This reduced rate of clipping implies that the custom algorithm does not spuriously switch between current talkers, but instead keeps the key conferees enabled until the end of their talkspurt or until a very loud, interrupting conferee breaks into the conference.

Fig. 3.6–Fig. 3.9 show graphical examples of how each of the algorithms perform during 15 seconds of conversation between four conferees. The speech signal from each conferee is shown on a separate axis and the portions of selected speech have been indicated with a bounding box. The third conferee is the primary speaker for the majority of the talkspurt, while the other conferees tend to produce short utterances which are reinforcements or responses to the primary speaker. It is clear that LT and FCFS do not have any notion of the third conferee's importance. Note that FCFS clearly clips major portions of the third and fourth conferee's signals between the 5–6 second mark, while LT introduces rapid switching between conferees two, three, and four around the 9 second mark. In contrast, NT eliminates much of the spurious clipping seen by conferee three; however there are still

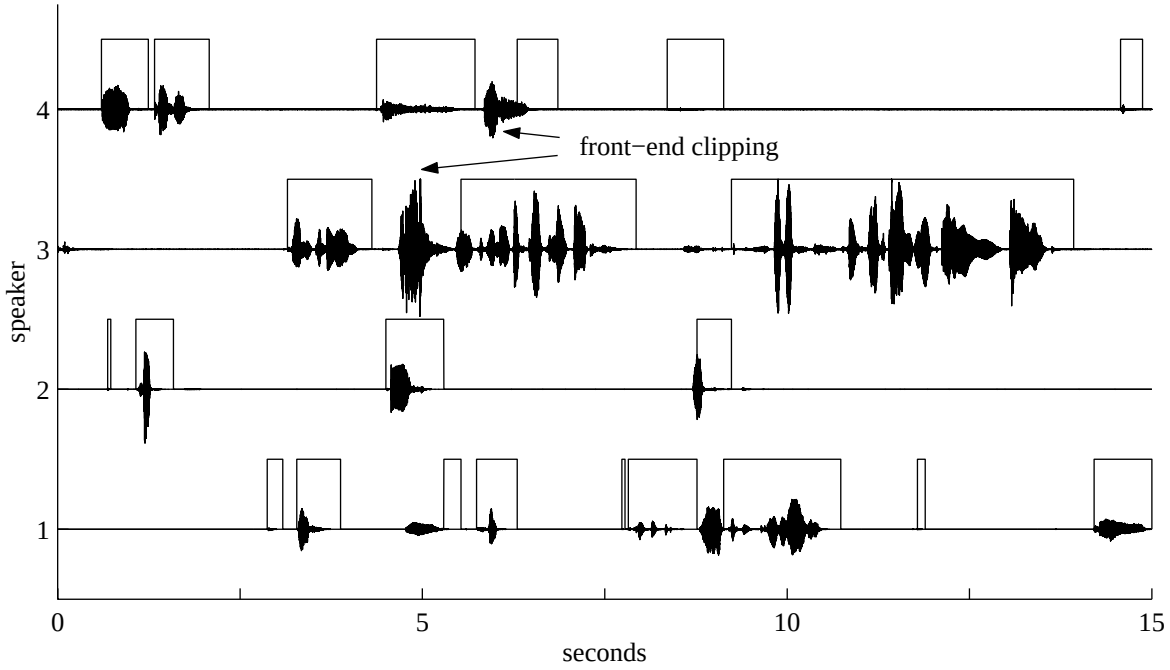


Fig. 3.6 Selected speech using FCFS.

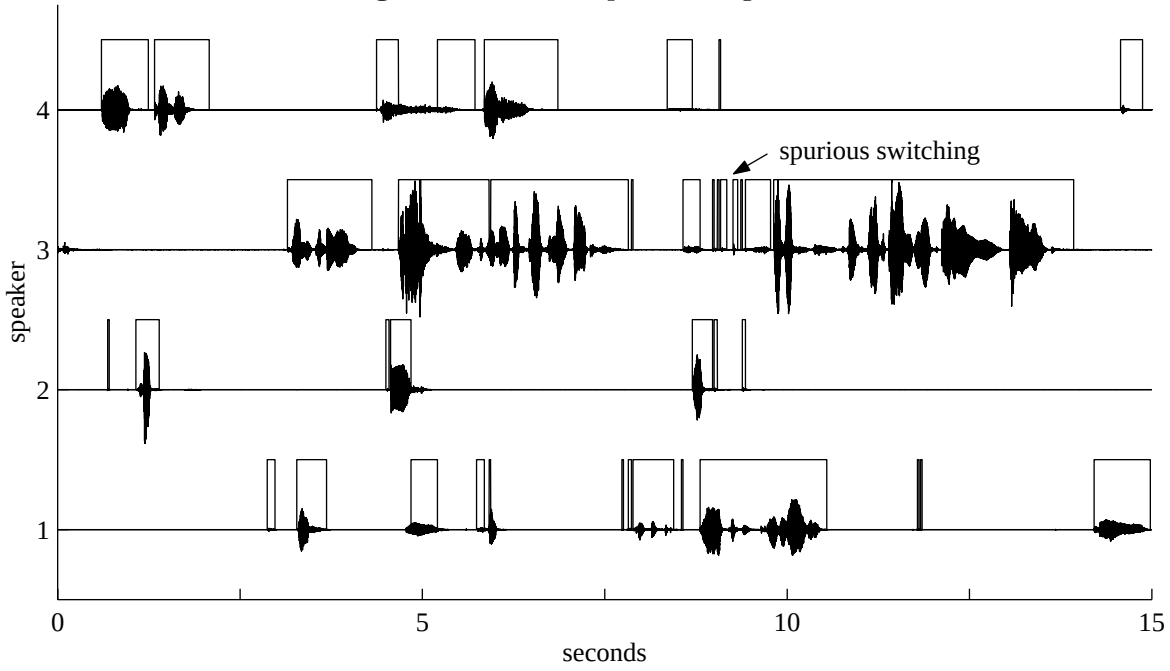


Fig. 3.7 Selected speech using LT.

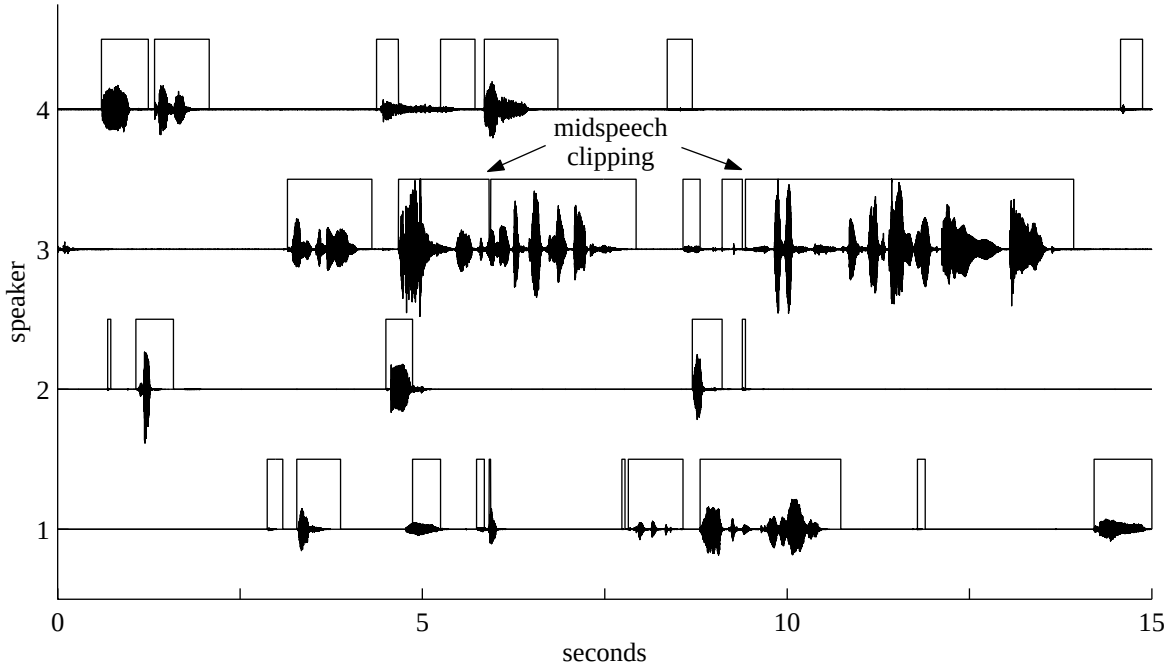


Fig. 3.8 Selected speech using NT.

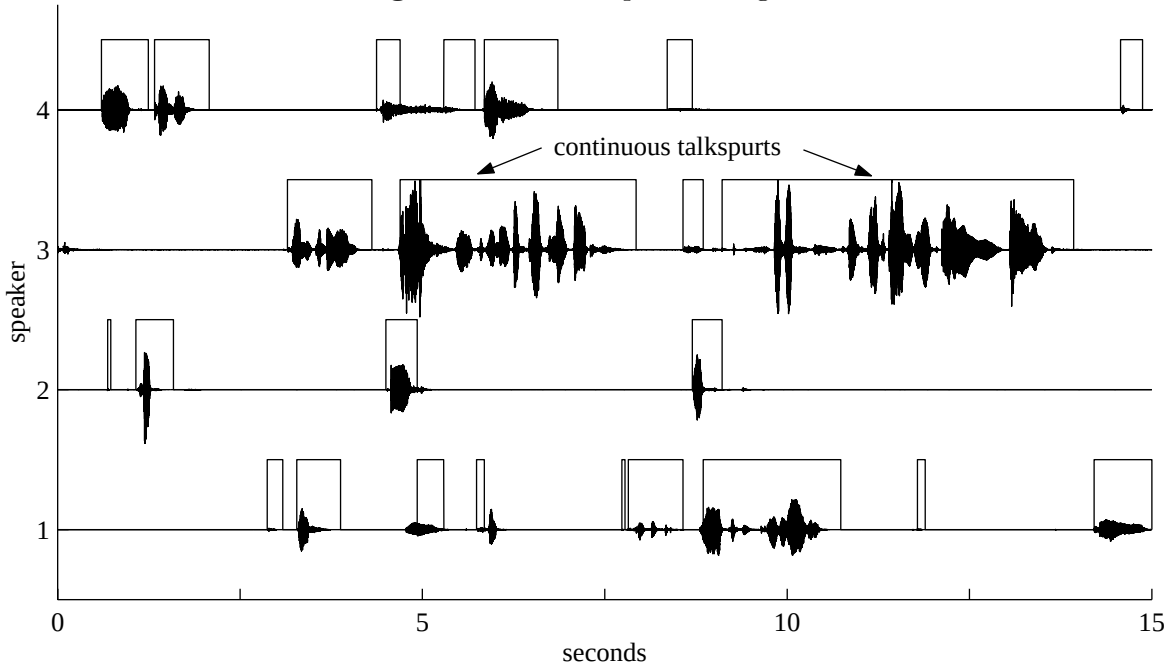


Fig. 3.9 Selected speech using TFSS.

two very short MSC occurrences in the third conferee's second and fourth talkspurt. The hysteresis of TFSS bridges these two short regions and provides a continuous stream of speech from this conferee.

Overall, the TFSS algorithm provides an improved two-talker speaker selection algorithm for use with 20 ms speech frames. The algorithm is thus suitable for use in the TFB, and provides an alternative to the traditional use of FCFS in packet-based conference bridges.

3.4 RTP/RTCP Design

RTP is used to carry real-time voice data over the IP network. As was explained in Section 2.2, the AVP (a companion document to the RTP standard) defines the RTP payload format for common speech and video codecs. However, a new payload type definition is required when a new data type needs to be carried in RTP, but does not fit into existing profiles or payload type specifications[4]. In the case of the TFO conferencing model, a new RTP payload type is required for carrying the side-information used for speaker selection at the TFB. This section outlines the requirements and framework for such a payload type, which is referred to the TFO payload type herein. To be clear, the TFO payload is the combination of the codec data frames, and the additional side-information, or TFO *data frames*. Section 3.4.1 states the requirements of the payload, and Section 3.4.2 presents typical use scenarios. Section 3.4.3 presents options for carrying the TFO data frames in RTP. Finally, Section 3.4.4 selects the best option.

3.4.1 Payload Requirements

It was shown in the previous section that a VAD decision and the signal energy, $\bar{E}_f$, are sufficient speech "features" for robust speaker selection using TFSS algorithm. These features, or TFO data, need to be calculated at the endpoint and transmitted to the TFB in the payload section of the RTP packet. The following points summarize the desired attributes of the TFO payload format:

[4]Payload types are constantly being defined and updated for a range of media. For example, a format has been defined for conveying the position of whiteboard pointers in a multimedia conference, while another has defined the payload format for carrying the EVRC [85].

1. Indication of TFO Payload: It is assumed that the use of the TFO payload is signalled out-of-band during the session (conference) initiation. These signalling exchanges also include negotiation of the speech coder, use of silence suppression/DTX etc.

2. RTP Header: All existing RTP header fields are interpreted as specified by the AVP.

3. Bundling: The sender must generate and transmit one TFO data frame for every speech codec data frame. Corresponding TFO data and speech codec data are always carried in the same packet.

4. Transparency: The TFO payload must be independent of, and must not prevent or hinder parsing of, any fields in the speech codec data frames, or any codec/payload-specific fields of the RTP header or payload.

5. Overhead: It is desirable that the number of overhead bits, such as a length indicator, be kept as small as possible.

6. Byte Alignment: Packet payloads are always zero-padded to an integer number of bytes. Thus, the TFO data should be an integer number of bytes in length, so needless padding bits are not transmitted.

7. VAD Indication: The TFO data frames should contain a VAD field. Since various VAD options exist (see Section 3.2), use of the field is optional and can be negotiated on a per-connection basis.

8. Energy Indication: The TFO data frame should contain an energy field which is populated with the normalized frame energy calculated in Eq. (3.1).

The above requirements lead to a certain intuition regarding the format of the payload. The energy and VAD fields are coded into TFO data frames—each data frame corresponds to one speech codec frame. Both the TFO and codec data frames are transmitted in the same packet. Clearly, the VAD and energy fields can be coded with 1- and 7-bits, respectively, yielding a one-byte TFO data frame. This strategy instantly satisfies requirements 7–8. An exemplary TFO data frame is shown in Fig. 3.10. The following section presents specific use scenarios which will enlighten the reader's understanding of how the fields in the TFO data frame may be used.

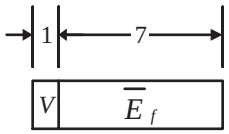


Fig. 3.10 A basic TFO data frame, consisting of a VAD decision, V, and the normalized frame energy, $\bar{E}_f$.

3.4.2 VAD Field Use Scenarios

Section 3.2 showed that there are actually three options for obtaining a VAD decision at the TFB: (1) explicit transmission of the V-bit from the endpoint to the TFB, (2) local generation, based on energy field in the TFO data frame, and (3) SID frame detection. Depending on the case, the V-bit may or may not be used. As is customary, the V-bit is interpreted as speech = 1 and silence = 0. The following two options present themselves as scenarios for V-bit usage.

Scenario 1: Continuous Transmission

When silence suppression/DTX is not used, either the TFB or the endpoints must produce a VAD decision which is presented to the speaker selection module. If it determined during the session initiation that the TFB is equipped with a better VAD than the endpoint, then the TFB VAD should be used, and the endpoint should set the VAD field to zero. If the converse is true, then the endpoint generates the VAD decision and transmits it to the TFB. However, the latter case is rare, since it is likely that silence suppression/DTX would be used if the endpoint is equipped with a VAD.

Scenario 2: Silence Suppression/DTX

When SID frames are supported, the endpoint always sets the VAD field to zero. Recall that in this case the TFB can discern between speech and silent periods by monitoring the arrival of the SID frames. However, in low-cost systems where silence suppression is accomplished by simply turning off the transmitter, the endpoint *could* populate the VAD field in order to help the TFB determine speech and silence regions.

3.4.3 Payload Format Options

Having presented requirements and use cases for the TFO data frames, the question that remains is, what are the options for carrying the TFO data frames in an RTP packet? This section answers the question, and generalizes to the case of a k-frame RTP payload, where $k \geq 1$. If k codec frames are bundled in one RTP packet, then k TFO data frames are also required, i.e., one per-codec frame. This allows the TFB to select individual frames for output, instead of the entire bundled payload. Since the RTP packet header contains no information regarding the number of codec frames carried in its payload, such as a length field, TFO data can be carried in the payload section of the packet without affecting the header. Transport of the TFO and codec data frames can be accomplished with either a stacked or blocked format.

Option 1: Stacked Format

The stacked format payload is illustrated in Fig. 3.11. In this case, each TFO data frame precedes its corresponding codec data frame. Thus, the TFO data becomes a "mini-header" for *each* frame of codec data.

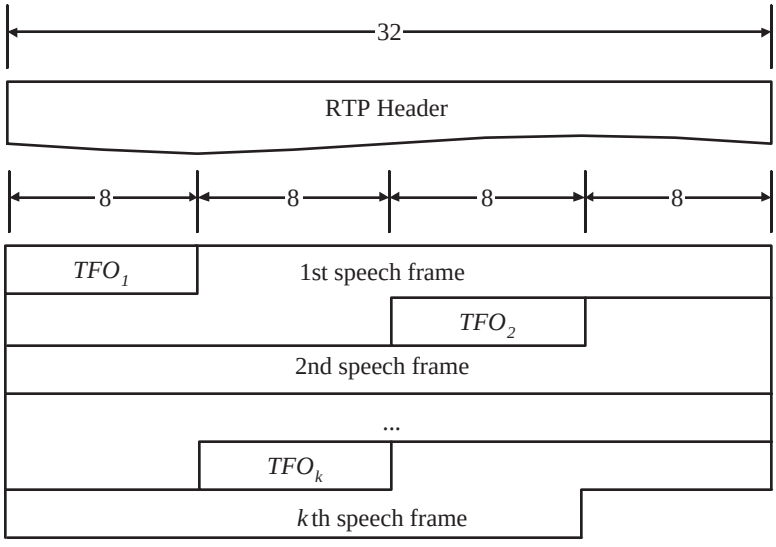


Fig. 3.11 TFO payload using stacked format.

This option makes the association between the TFO data and its corresponding codec frame straightforward, since they are adjacent to each other. Since the TFO data frames

are the same length, and most speech coders emit fixed length frames, the length of the RTP payload is an implicit indication of the number of frames, k, it carries. However, the stacked format presents a problem when RTP carries bundled variable rate codec data. In these cases, the length of the payload is not indicative of k because multiple combinations of variable length frames can produce the same payload length. These payloads can be delineated by transmitting a table, or ToC fields, that describe the contents of the payload. These fields are transmitted immediately following the RTP header[5]. If TFO data frames are interleaved within the packet, the meanings of the ToC fields are lost. Of course, the ToC fields could be modified to reflect the presence of the TFO data, but this would make the scheme payload specific, which is undesirable.

Option 2: Blocked Format

Packing of the blocked format payload is illustrated in Fig. 3.12. In this case, the TFO data frames are laid out one-after-another, following the RTP header. The order in which the TFO data frames appear is the same as the order in which their corresponding codec frames appear.

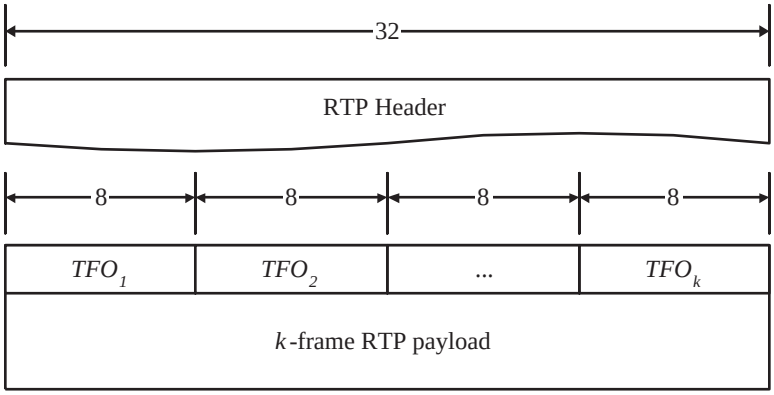


Fig. 3.12 TFO payload using blocked format.

This option has the advantage of separating the TFO data from the regular RTP payload. Hence, once the depacketizer has finished processing the k TFO data frames, the remainder of the payload may be processed in the usual way. Another minor advantage is that the speech data can be "block copied" from the input buffer. However, in the case that

[5]Such is the case for the Type 2 RTP payload for EVRC speech [85].

the packet carries bundled variable rate codec data, the number of frames, k, is derived from the ToC fields. Since the ToC follows the k TFO data frames, they cannot be found without first knowing k. Therefore, the end of the TFO data must be signalled.

Two methods can be used to indicate the number of TFO data frames carried in the packet: either an explicit length field, which immediately follows the header, or an implicit indication of length, using a "Further Entry Indication" field in each TFO data frame. The latter method is used by the EVRC ToC fields [85]. A length field imposes a fixed overhead to each packet, and is wasteful when the packet only contains one frame, i.e., $k = 1$. On the other hand, the Further Entry Indication field creates an overhead of 1-bit per-frame, and has the advantage that the TFO data frames remain self-contained. Further, the TFO payload is intended for interactive conferences where payload lengths are necessarily short. Hence, the k-bit overhead is optimal, and the Further Entry Indication is a better choice (the author admits the number of bits saved/lost by either scheme is very small).

The Further Entry Indication, F, is allocated the first bit of each TFO data frame. The last TFO data frame is indicated when $F = 0$, otherwise $F = 1$. The addition of the F-bit implies that the energy field must be reduced to 6 bits. This is reasonable, since energy can adequately be represented with as little as 5 bits [46]. The layout of the new TFO data frame is shown in Fig. 3.13.

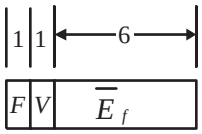


Fig. 3.13 A TFO data frame, consisting of a Further Entry Indication, F, a VAD decision, V, and the normalized frame energy, $\bar{E}_f$.

Option 3: Header Extension

The two options considered above provide mechanisms for carrying the TFO data in the payload section of packet. However, RTP supports a header extension mechanism which may be used to transport application-specific data, without effect on the regular payload. Use of the header extension is primarily intended for limited, *experimental* use. Yet, it does present an interesting option.

The header extension contains two mandatory 16-bit fields, the Type and Length fields, and an optional Data field. The Type field is user-defined, and the Length field indicates

the number of 32-bit words carried in the Data field. The Data field has no structure, and is populated with application-specific data, such as the TFO data frames. Note that the Data field must be padded to an integer multiple of 32-bits. Hence, k TFO data frames (with F-bits) can be laid out in the extension, and padded out if necessary. This approach has the advantage that the header extension already exists in RTP, and is independent of the payload section. Thus, an intermediate system, such as an RTP translator, may process the packets between the endpoint and TFB without having to support the TFO payload.

3.4.4 Payload Format Comparison

Clearly, Options 2 and 3 are the superior payload packing scheme, since they offer the most transparent operation with existing payload types, and satisfy all of the requirements. In addition, the TFO data is really additional header information, so it makes sense to place it directly after the header (which both options do).

Using Option 2, the increase in bitrate due to the TFO data frames is proportional to the *codec* frame rate, since the TFO data is fixed at 8 bits/frame. The TFO data bitrate is simply

$$\epsilon = 8/t_f \quad \text{kbps}, \tag{3.6}$$

where t_f is the duration of the codec analysis frame in ms. Considering a typical 20 ms payload of G.729(A) (i.e., 2 frames/packet), $\epsilon = 0.8$ kbps. Option 3 does not fare as well due to the 32-bit overhead of the Type and Length fields, and the padded 32-bit Data field. Considering the same example, Option 3 yields $\epsilon = 3.2$ kbps.

Although the RTP header extension is flexible and convenient, it is inefficient. Further, it is intended for experimental use. Therefore, the blocked format of Option 2 is the best choice for transporting the TFO data in RTP.

However, during the TFO conferencing experiments described in Chapter 4, the header extension option was used, and the TFO data frame was simplified since the number of frames per-packet was always fixed. Implementation and performance details of the conferencing facility are provided in Appendix A.

3.5 Chapter Summary

This chapter has presented design options and solutions for a new conferencing model suitable for use in current and future VoIP networks. The analysis considered the core technologies and methods required for realizing the model, mainly the speaker selection and RTP issues. In particular, it was shown that an enhanced legacy speaker selection technique yields superior results compared to current methods. The key idea behind the TFO model is that additional speech features are sent to the bridge in each packet such that speaker selection can be performed independently of the speech codec. These features were defined to be the average signal energy and a VAD-bit; a new RTP payload type was designed around these features. The methods developed and selected throughout this chapter set the basis for an implementation and performance analysis of the TFO conferencing model.

Chapter 4

System Evaluations and Results

The field of teleconferencing has little established practice regarding performance evaluation. It was decided at the outset of this project that the subjective performance evaluation would be conducted on a conferencing testbed capable of supporting or emulating various conferencing scenarios. The testbed facilitated comparisons between systems allowing two primary goals to be achieved. The first goal was to evaluate "how bad" the speech quality actually was when using a modern speech coder such as G.729A in a conventional VoIP conference bridge. G.729A (and other) speech coders are typically rated for two or three encodings of a *single* voice, yet the effect of multiple voices has not been formally evaluated since Forgie did so in 1979 with relatively low-quality speech coding techniques (LPC at 2.4 kbps, CVSD at 32 kbps) [23]. The second goal was to evaluate the speech quality of the Tandem-Free Operation (TFO) conferencing arrangement using the TFSS algorithm for voice switching. Both the G.729A and G.711 speech codecs were used in the evaluations, although G.711 was typically used to set a speech quality baseline. These evaluations are described in Section 4.3.5.

The chapter is divided into three sections. Section 4.1 presents the models and parameters considered in the remainder of the chapter. As a precursor to the live testing, Section 4.2 characterizes the systems in terms of computational complexity and bandwidth utilization. Section 4.3 presents the experimental set-up, method, and results of the real-time subjective comparisons between five different conferencing systems of interest.

4.1 Overview of Conferencing Models and Parameters

Section 2.5 and Section 2.6 presented the centralized and decentralized conferencing classes. In VoIP, the most common embodiments of these two conferencing paradigms are a conventional IP conference bridge and a wide-area multicast conference, respectively. For the remainder of this chapter, these two models set the basis for comparison against the TFO conferencing model. The three systems considered are:

- TFO: Each endpoint transmits one stream to the Tandem-Free Operation Bridge (TFB) and receives M streams in return. The streams are mixed at the endpoints.
- Conventional Bridge (CB): Each endpoint transmits its stream to the bridge and receives one mixed, composite signal in return.
- Multicast (MCAST): Each endpoint transmits one stream to the conference mulitcast address and receives $N - 1$ streams in return from the same address. Again, the streams are mixed at the endpoints.

In addition to these models, certain parameters were fixed across all offline and subjective performance evaluations. These parameters are:

- Speech Codecs: Only G.711 and G.729A are considered.
- Payload Duration: The RTP packet payload duration was always set to 20 ms. Each RTP packet contained two G.729A codec data frames, or one G.711 codec data frame.
- Speaker Selection: If used, the speaker selection algorithm selected two talkers, i.e., $M = 2$. Selection was performed on whole RTP packets.
- No Silence Suppression/DTX: All packet streams were assumed to be continuous, i.e., silence suppression/DTX was not used. This permits a *worst-case* analysis in terms of computational requirements and bandwidth[1].

[1] As stated in Section 2.7.2, there is no accepted model for an N-way conversation ($N > 2$), and it was not the intent of this thesis to provide one. In any case, worst-case analysis is preferable since it shows the bounds of the system.

4.2 Complexity Analysis

This section characterizes the processing and bandwidth requirements of both the TFB and a TFO endpoint relative to the equivalent entities in the CB and MCAST conferencing models. Note that the analysis in this section was accomplished through simulation as opposed to benchmarking real systems. Only the core bridging and endpoint algorithms are considered, and not the overhead due to specific platform or implementation issues. For example, the effect of the operating system scheduler, packet or audio I/O, or list/queue management etc. are not studied.

4.2.1 Method of Analysis

CPU utilization (CPU%) was used to obtain relative complexity metrics for the bridge and endpoints of the TFO, CB, and MCAST conferencing models. Conference algorithms were broken down into four "atomic" operations: (1) speech encoding, (2) speech decoding, (3) speaker selection, and (4) mixing, where one mixing operation is assumed to be the vector addition of two decoded speech frames: $z_n = \sum_{n=1}^{L}(x_n + y_n)$. The number of these operations for the bridge and endpoint of a given system is shown in Table 4.1. The estimated CPU% for a bridge or endpoint was determined by scaling each operation by its empirically measured CPU%, and then summing the operations in a column-wise fashion. Measurements were taken on "slow" (266 MHz/PII), "moderate" (400 MHz/Celeron), and "fast" (800 MHz/PIII) PCs. The actual CPU% of each operation are shown in Appendix B.

Table 4.1 Order of computational complexity for conferencing models, where N is the number of conferees, and M is the number of selected speakers. The breakdown of CPU utilization for each event is shown in Table B.1 and Table B.2.

Model	Bridge				Endpoint			
	Encode	Decode	Select	Mix	Encode	Decode	Select	Mix
TFO	—	—	N	—	1	2	—	1
CB	$M+1$	N	N	$2(M-1)$	1	1	—	—
MCAST	—	—	—	—	1	$N-1$	$N-1$	$M-1$

Table 4.1 shows the mixing operations to be $O(M)$ and not $O(N)$ because it was assumed that the CB and MCAST models use speaker selection to fix the number of mixes. In CB,

the bridge is assumed to use the $2N$ algorithm of Section 2.5.4, hence it executes $2(M-1)$ mixes; the MCAST endpoint executes $M-1$ mixes. Note that TFSS is always used for speaker selection, and the effect of multiple simultaneous conferences is not considered.

Table 4.1 can be used to make some initial observations. For example, the scalability of CB is limited by the N decode operations. The complexities of the TFO and CB endpoints are independent of conference size, implying that they are both scalable to large conferences. This is not the case with an MCAST endpoint, which is limited by the $N-1$ decodes.

4.2.2 Bridge Processing Requirements

The TFO conference model reduces the complexity of the bridge by eliminating the speech coders and mixing modules from the bridging algorithm. For large conferences, the *reduction* in complexity is proportional to the N decode operations, since the mixing and encoding operations are held constant by speaker selection. The gain is higher for small conferences because the processing requirements of the traditional algorithm are dominated by the $M+1$ encode operations. This is illustrated in Fig. 4.1, which shows the relative reduction in complexity of the TFO bridge in comparison to a traditional bridge when G.711 or G.729A are used.

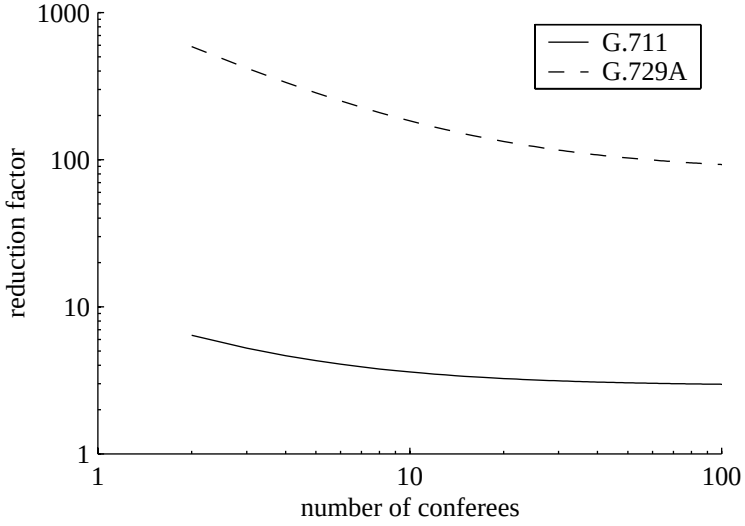


Fig. 4.1 Complexity reduction factor of the TFO bridge algorithm when compared to a conventional VoIP conference bridge. The two algorithms were benchmarked on a Windows 2000 machine with a 800 MHz/PIII processor.

Clearly, the TFO model is an attractive alternative if a conferencing provider wishes to use a high complexity codec such as G.729A. Even in the case of G.711, which is no more complex than a table look-up, the TFO bridge gives a three-fold savings in computational power.

The above analysis disregards the effect of the underlying computing platform on which a conference bridge runs. Nevertheless, the results indicate that the speaker selection algorithm should not be the bottleneck in a real-time implementation. For example, a very recent paper demonstrated that a software implementation of a traditional bridge (running on a 360 MHz Sun SPARC Ultra 10), had a maximum capacity of 80 conferees—all using G.711—before the speech quality was degraded [6]. Interestingly, only 10–15% of the CPU was being used at this operating point, implying that the decode-mix-encode sequence was not the performance bottleneck.

4.2.3 Endpoint Processing Requirements

The complexity of a TFO endpoint is increased by two decodes and one mix. Such an increase does not present a problem for even a "slow" processor. For example, a G.729A decoder on a 266 MHz/PII processor requires about 2.2% of the CPU; therefore, two decodes can easily be managed. To compare with the CB and MCAST endpoints, the estimated processing requirements with respect to G.711 and G.729A have been summarized in Table 4.2[2].

Table 4.2 Worst-case CPU utilization (CPU%) for an endpoint in a TFO, CB, and MCAST conferencing scenario. N is the number of conferees. All measurements were taken on Win2k PCs.

CPU	TFO		CB		MCAST	
(MHz/Class)	G.711	G.729A	G.711	G.729A	G.711	G.729A
266/PII	0.08	13.19	0.05	11.01	$0.03N$	$2.18N$+6.67
400/Celeron	0.07	8.75	0.04	7.29	$0.02N$	$1.47N$+4.37
800/PIII	0.03	4.28	0.02	3.58	$0.01N$	$0.70N$+2.18

A TFO endpoint using G.729A requires a 20% increase in complexity over a CB endpoint. For PC-based conferencing, this is generally not a problem since processors are

[2]Note that the calculation of the TFO data has *not* been included in the following analysis, since it is a minor factor in comparison to the speech coder.

rarely used to their full potential. However, for tightly-engineered cell and/or IP phones, processor speed would have to be increased in order to support TFO conferencing. In comparison to the tandem-free, yet non-scalable MCAST conference endpoint, this is a modest increase in complexity.

4.2.4 System Bandwidth Utilization

This section compares the TFO, CB, and MCAST conferencing models in terms of worst-case bandwidth of the RTP streams. The worst-case can occur when (1) all endpoints are transmitting at full rate, i.e., silence suppression/DTX is not used, or (2) silence suppression/DTX *is* being used, and all the conferees talk at the same time. The analysis of the three models is straightforward, and is summarized in Table 4.3.

Table 4.3 Order of worst-case bandwidth (kbps) for conferencing models, where N is the number of conferees, B is the bandwidth (kbps) of the RTP stream, and ϵ is the bandwidth (kbps) of the TFO payload from Eq. (3.6).

Model	Bridge		Endpoint	
	In	Out	In	Out
TFO	$N(B+\epsilon)$	$2(N-1)B$	$2B$	$B+\epsilon$
CB	NB	NB	B	B
MCAST	—	—	$(N-1)B$	B

Each TFO endpoint includes one byte of TFO data for each codec frame packed in the RTP packet (Section 3.4), yielding an increase in bandwidth of ϵ kbps on the upstream (Eq. (3.6)). At the bridge, this translates to an increased input bandwidth of $N\epsilon$ kbps. On the downstream, the TFO bridge selects and forwards the packets of the two current talkers back to the endpoints—the current talkers only receive the other's packets. Since the 1-byte TFO data frame is removed at the bridge, the total output rate is $2(N-1)B$ kbps. Therefore, the bandwidth requirement at the TFB is slightly higher than CB scenario, although they both grow with N.

It is instructive to add numbers to Table 4.3 so that real comparisons can be made. Table 4.4 is simply Table 4.3 with B replaced by the bitrate of the RTP stream, i.e., the sum of the 16 kbps IP/UDP/RTP header and the 64, or 8 kbps payload for G.711 and G.729(A), respectively.

Table 4.4 Worst-case bandwidth (kbps) of conferencing models, where N is the number of conferees ($N > 1$). The analysis assumes that each RTP packet carries 20 ms of coded speech (i.e., 50 packets/s). For the TFO model, there is 1-byte of TFO data per-codec frame. Both RTP header compression and silence suppression/DTX are off.

Model	Bridge				Endpoint			
	G.711		G.729(A)		G.711		G.729(A)	
	In	Out	In	Out	In	Out	In	Out
TFO	$80.4N$	$160(N-1)$	$24.8N$	$48(N-1)$	160	80.4	48	24.8
CB	$80.0N$	$80N$	$24.0N$	$24N$	80	80	24	24
MCAST	—	—	—	—	$80(N-1)$	80	$24(N-1)$	24

Immediately, the minimal impact of the side-information is apparent: a mere 0.4 and 0.8 kbps increase on the upstream links for G.711 and G.729(A) respectively. However, in an overall comparison to CB conferencing, the channel between the bridge and endpoint is subject to 50.3 and 51.7% increase for the same two codecs.

The original idea behind the TFO model was to allow compressed speech to be used in centralized, carrier-grade conferencing arrangements without sacrificing voice quality. Thus, what is really of interest is how TFO conferencing with G.729(A) compares to CB conferencing with G.711. In this case, TFO/G.729(A) gives a 54.5% reduction in bandwidth over a CB/G.711 conference—if RTP header compression is used[3], then the bandwidth requirements are reduced by $\approx 79\%$.

4.3 Voice Quality Evaluations

Until now, the performance of the TFO conferencing system has been characterized. However, the success of the technique, as always, depends on the resultant speech quality. This section presents the methodology, and results of five different system comparisons, performed over a live conferencing system.

[3]Note that to support TFO conferencing over low-speed links, such as a 56 kbps modem, RTP header compression is required, since the downstream channel operates at twice the normal bandwidth. In the case of TFO conferencing with G.729(A), the bandwidth would be 27.2–29.6 kbps with cRTP, or 72 kbps without.

4.3.1 Conferencing Facility

The recent stabilization, popularity, and availability of computer-based conferencing software has made the PC-based conferencing facility an obvious choice for experimentation. The facility consisted of a custom TFB and four conference terminals, all implemented entirely in software. The TFB ran on a Windows 2000 platform, while each conference station consisted of a Linux PC running the Robust Audio Tool (RAT) [39]. The PCs were connected via a 100 Mbps switched Ethernet at the McGill University Telecommunications and Signal Processing Laboratory (TSP Lab). The stations were arranged such that conferees were unable to see or hear one another during the conference. Two of the stations were in their own rooms, while the remaining two stations were placed in cubicles at opposite ends of a third room.

The TFO conferencing system was constructed according to the design solutions presented in Chapter 3. However, the system could emulate TFO, CB, and MCAST conferencing by a combination of endpoint and TFB parameters. The live evaluations were run using the same assumptions as presented in Section 4.1, except that speaker selection was not used in the CB and MCAST scenarios to detect inactive conferees.

RAT underwent various modifications and enhancements in order to support the live evaluations. For example, G.729A was added since RAT only supports low-quality, freely-available speech coding algorithms. Since full support of the proposed blocked format TFO payload was deemed to have no experimental value, the RTP header extension was used to transport the TFO data from endpoint to bridge; only $\bar{E}_f$ was transmitted. In addition, tighter control was maintained over RAT's playout buffer by changing the interface to allow a lower maximum playout delay to be set, as well as an optional bypass of the adaptive cushion algorithm (the reader is encouraged to review the dissection of the end-to-end delay between two RATs in Section A.2). Note that the EVRC noise-suppression block VAD was also added, but was not used in the experiments. A detailed description of these modifications is provided in Section A.1.1.

Each endpoint transmitted one stream to the TFB, and received M streams in return. The number of speakers to select, M, was configurable, as was the speaker selection strategy and any relevant parameters. When $M < N$, the TFB selectively forwarded packets based on the output of the speaker selection algorithm. The CB scenario was emulated by yet another modification to RAT, which passed the conference sum through another encode-

decode cycle prior to playout. Since the test LAN filtered multicast packets, MCAST conferencing was emulated by setting $M = N$, which essentially bypassed the selection process and caused the bridge to operate as a packet reflector. The available conferencing scenarios are summarized in Table 4.5.

Table 4.5 Summary of supported conferencing scenarios. The TFB used the First-Come-First-Served (FCFS), Loudest Talker (LT), and Tandem-Free Speaker Selection (TFSS) algorithms for speaker selection.

Conference Model	Speaker Selection	Speech Codec	
		G.711	G.729A
CB		X	X
MCAST		X	X
TFO	FCFS	X	X
	LT	X	X
	TFSS	X	X

For simplicity, session initiation tools were ruled out in favour of manual conference configuration. Initially, the Session Directory (SDR) tool was used to advertise sessions, but was found to serve no useful purpose since the nature of the conferences and the capabilities of each station were well known in advance.

Prior to each session, RAT was manually invoked on each station, providing the address of the TFB as input from the command line. The TFB was then invoked on a fifth PC; the switching parameters were set using a single file which was loaded at startup. For example, the number of selected speakers could be set between one and four, and the switching parameters could be set to the desired speaker selection algorithm.

Speech codecs could be changed using the RAT interface at anytime during the conference. Either G.711 or G.729A was used. The bridging software performed speaker selection and forwarding regardless of the speech codec being used by RAT.

4.3.2 Experimental Scope

In the speaker selection scenarios, the number of selected conferees, M, was set to two, and there were a total of three selection algorithms. This limited the number of possible scenarios to ten. Some scenarios were eliminated since they were discerned to give no

real useful information. For example, NT speaker selection was not added to the system since it was not expected to differ greatly from TFSS. FCFS was only tested with G.711 since at the time of the test G.729A had not been integrated into RAT. The LT and TFSS algorithms were only tested with G.729A since the speaker selection technique is not warranted when G.711 conferencing is available. The final schedule of comparisons is summarized in Table 4.6.

Table 4.6 Schedule of conferencing system comparisons. Note that all conferences were held with four conferees. For each system the number of selected simultaneous talkers is M-of-N, where $M \leq 4$, and $N = 4$. For example, Comparison 5 reads as CB with 4-of-4 selected talkers vs. TFSS with 2-of-4 selected talkers. The speech codec used in the comparison is marked with X.

Comparison	System 1			System 2		Speech Codec	
	Model	M-of-N		Model	M-of-N	G.711	G.729A
1	CB	(4-of-4)	vs.	MCAST	(4-of-4)	X	X
2	MCAST	(4-of-4)	vs.	FCFS	(2-of-4)	X	
3	MCAST	(4-of-4)	vs.	TFSS	(2-of-4)		X
4	LT	(2-of-4)	vs.	TFSS	(2-of-4)		X
5	CB	(4-of-4)	vs.	TFSS	(2-of-4)		X

4.3.3 Experimental Method

To encourage interaction among conferees during live tests, a simple game was developed based on the TV game show *Family Feud*. A similar technique was used in [44], while more complicated problem solving tasks were designed in [23]. The Family Feud option was chosen since its rules are well-known and/or easily learned. In addition, it encourages competition between conferees, resulting in speech overlaps which test the performance of the speaker selection algorithms. A list of questions and "correct" answers were gathered from an online version of Family Feud and from the actual television show.

During the game, the four conferees worked together to find as many suitable answers to the questions as possible. Once the group had decided on a final list, the true answers were revealed to the conferees. During each round, one conferee selected a question to read to the others, but also participated in the problem solving process. Organization of the game, such as designating a "reader", moving to the next question, and/or score keeping was

left up to the conferees. This created a "real" problem solving scenario for the conferees, since they were forced to organize themselves in order to complete the questionnaire. Each experimental scenario lasted approximately 10–15 minutes over which time several different questions were solved.

Prior to each conference, the conferees were instructed on how to play the game and the question sheets were handed out. At the beginning of each conference, the experimenter used one of the stations to discuss the overall operation and expected quality of the system with the conferees. This briefing session usually lasted a few minutes and allowed the participants to grow accustomed to the overall sound of the system. These sessions were conducted with G.711 and without speaker selection so that the conferees could form a baseline against which the speech quality of ensuing scenarios could be compared.

Informal observations indicated that novice PC-telephony users were not accustomed to the "sound" of the system. This was attributed to the following factors: (1) the speech was not telephone bandpass filtered, (2) each conferee wore headsets as opposed to the customary single ear-piece telephone handsets and (3) there were slight variations in gain levels between conference stations. Formal study of these effects can be found in [86].

Over the course of the project, twelve subjects in total used the system. Eight of the twelve were repeat subjects, and became familiar with the equipment and proceedings. Some groups, mainly consisting of colleagues, were used to perform integrity tests on each of the systems. These tests consisted of listening for switching anomalies in FCFS, LT, and TFSS during periods of multi-talk. For example, the naturalness of barge-in for TFSS was tested by having two people read a passage of text at a normal level while a third shouted. The fourth conferee listened to make sure that the third person was heard and recorded whether or not the interruption sounded natural.

4.3.4 Data Collection

A post-conference interview was held, with the conferees' as a group, following each experimental scenario. Subjective ratings and comments were collected during this phase. In particular, the subjects were asked to rate the system in terms of system effects and speech quality. System effects were defined for the conferees as factors such as noise level and speech clipping, while speech quality was defined in terms of intelligibility. System effects were directly related to the speaker selection algorithm (if present), while speech quality

was determined by the arrangement and choice of codecs.

4.3.5 Evaluation Results

This section summarizes the conferee's attitudes regarding the system and the speech quality. In all cases the conferees were asked to choose the system that they preferred.

Comparison 1: Conventional Bridge vs. Multicast

During this experiment the centralized and decentralized conferencing scenarios were tested with both G.711 and G.729A. Two groups of three and four participants were used over four different sessions. None of the conferees perceived a difference between the CB and MCAST scenarios for G.711, and felt that the speech quality was high. For MCAST with G.729A, conferees commented that fricatives such /f/ and /s/ seemed distorted, which was later verified to be an inherent property of the G.729A coder. However, all conferees felt strongly that the combination of CB and G.729A seriously degraded speech quality and intelligibility. Subjects often asked the primary talker to repeat him/herself because the meaning of entire sentences was sometimes lost. This was confirmed in later trials by an experimenter either participating or listening in on the conference. The majority of subjects felt that CB sounded so poor they believed there was something wrong with the conferencing system.

Comparison 2: Multicast vs. FCFS

This scenario was tested once with a group of four conferees, after which the algorithm was abandoned. However, the initial purpose of the test was to discern whether or not the speech clipping and lack of barge-in of FCFS would have a noticeable effect on the perceived quality. G.711 was used to provide higher speech quality so that clipping anomalies could be perceived more easily. In general, the conferees did not find any significant differences between the two systems. Comments made after the conference indicated that the participants were aware that some type of speaker switching was being performed for FCFS and not MCAST because of the abruptness of the switching. However, the switching did not seem to impair the conferees' ability to use the system or to *want* to use the system.

Comparison 3: Multicast vs. TFSS

Three groups of four were used to evaluate this scenario. The game was only used on two occasions; the third occasion consisted of testing the limits of the TFSS system with an experienced group. The purpose of each trial was to see if the conferees were aware of the presence of a switching algorithm, and to see if there was a noticeable difference in speech quality between MCAST and TFSS.

The first two groups showed no preference between the two systems, while the third, most experienced group described TFSS as "marginally" better. Integrity tests with the third group revealed that TFSS introduced minor side effects such as clicks and pops. These effects were also heard in the simulations and determined to be a product of late switching decisions (1–2 frames) and G.729A decoder state loss. TFSS handled barge-in with no perceivable voice breakups. There was also an indication that TFSS provided a reduction in low-level noise over MCAST. This was expected since RAT was not modified to suppress the signals of the inactive conferees.

Comparison 4: LT vs. TFSS

Three groups of four were used to evaluate this scenario. The game was only used on two occasions; the third trial involved testing the limits of the two selection algorithms by purposely creating long durations of double and triple talk. It was expected that the TFSS algorithm would smooth out speaker transitions as opposed to LT.

As expected, conferees showed a strong preference for TFSS over LT. The LT algorithm resulted in frequent voice breakups, and a low, choppy sound was heard during periods of silence and single-talk. Both were due to spurious switching between conferees that had near-equivalent energy levels, mainly during times of speech-onset and offset or silence. In general, conferees found LT difficult and unpleasant to use. The background noise of the TFSS system was smooth and consistent, and there were no voice break-ups, resulting in an overall "cleaner" sound.

Comparison 5: Conventional Bridge vs. TFSS

This test was performed with two groups of four conferees using G.729A. The game was used as an outline for the tests, yet much of the conversations were centred around the speech quality difference between the two systems. Conferees unanimously agreed that not

only did the TFSS system provide superior speech quality, but that the CB system was very poor and unpleasant to use. As in earlier trials with CB, the conferees reported reduced intelligibility and muffled-sounding speech. Again, conferees attributed TFSS's occasional short clicks and pops to overall system effects and not to the switching algorithm.

4.3.6 Summary of Evaluations

Table 4.7 ranks the conferencing systems based on the opinions of conferees collected during the evaluations. A notable outcome of the experiments is that the conferees did not detect the presence of the TFSS speaker selection algorithm, even though the number of output streams was limited to two. In the case of LT, voice break-up was often so severe it was perceived as a system malfunction and not a selection algorithm. In contrast TFSS was definitely preferred over most systems, although only marginally over MCAST. Speech quality was perceived to be high even in the face of minor clicks and pops due to occasional switching errors and G.729A state loss. In general, TFSS was found to provide equivalent or better speech quality than all other systems.

In contrast CB was consistently rated the worst by the conferees. The system was unpleasant and difficult to use due to severe distortion caused by tandeming a multi-speaker signal with G.729A. However, conferees found CB to be indistinguishable from MCAST when using G.711.

Table 4.7 Summary of conferee's opinions and system rankings. FCFS is not included in the rankings because it was not directly compared to the other selection algorithms.

Rank	System	M-of-N	Speech Codec	Speech Quality	Comments
1	CB MCAST	(4-of-4)	G.711	Good	
2	TFSS	(2-of-4)	G.729A	Good	Occasional Pops/Clicks
3	MCAST	(4-of-4)	G.729A	Good	More Background Noise
4	CB	(4-of-4)	G.729A	Poor	Muffled, Less Intelligibility
5	LT	(2-of-4)	G.711 G.729A	Very Poor	Voice Break-ups

Chapter 5

Conclusion

A framework has been described for supporting the Tandem-Free Operation (TFO) conferencing model in the context of Voice over IP (VoIP). Much of the work has concentrated on design options and solutions for the Voice Activity Detector (VAD) and speaker selection modules of the Tandem-Free Operation Bridge (TFB), as well as a Real Time Protocol (RTP) payload format for carrying TFO data from the endpoints to the TFB. These works laid the foundation for a software-based implementation of the TFB, which was subsequently used to assess the performance of the TFO conferencing model through live four-member conferences. This last chapter provides a summary of the work, starting with a general discussion and conclusions in Section 5.1, followed by an exposition of potential future work in Section 5.2.

5.1 Summary and Discussion of Results

Chapter 1 presented the problem of moving the traditional conference bridge into an Internet Protocol (IP) network with highly compressed speech. Tandem encodings and vocoder representation of a multi-speaker signal result in a significant reduction in speech intelligibility, increased end-to-end delay, and a need for massive computing power. One can appreciate the advantages of the TFO conferencing model prior to formal experimentation. Clearly, the elimination of the tandem encodings at the bridge has a positive effect on the resultant speech quality and complexity of the system. Subjective testing affirmed this assumption: the resultant speech quality of a conventional IP-bridge was in fact *worse* than expected. Hence the traditional conferencing service is compromised when G.729A

and equivalent codecs are used. Other codecs with better resilience to tandem encodings *may* be used, but need to be tested on a per-codec basis.

Multicast conferencing tools have been providing bridge-less, peer-to-peer conferencing capability since the mid 1990s. However, the utility of multicast conferencing is limited due to the lack of widespread support for multicast routing, especially in modern IP telephony networks. As VoIP networks mature and evolve, it is likely that wide-area multicast conferences will supplant traditional conference bridges. Until this occurs, the TFO conferencing model is well-suited as a transition technology. For example, carriers could reduce costs by implementing their three-way calling line-option with the TFO model instead of using expensive audio bridge resources. Moreover, service providers could offer "classes" of conferencing service to their users: "executive" or "business" class could use a traditional bridge using G.711 (or better), while "basic" service could be provided with the TFO model using G.729(A) (or equivalent).

Deploying such TFO conferencing services is inexpensive since the TFB requires limited computational power. It can easily run on a modern multi-purpose workstation or could be implemented as additional service logic within a router. Much like the endsystem mixing model of Section 2.5.6, the TFB could be hosted on one of the endpoints. The size of the conference would be limited by the packet I/O and scheduling capabilities of the underlying computing platform of the TFB, as opposed to the bridging algorithm. The additional load on the TFO endpoint is really a non-issue since VoIP endpoints are predominantly high-powered Media Gateways or workstations, both of which have ample processing power. Since the TFB restricts the number of talkers to M (i.e., 2), endpoints are scalable to *any* size of conference. In essence, the TFO conferencing model leverages the computing power already available in the network instead of adding more complexity to it.

The TFO conferencing model raises the bandwidth requirements over that of a traditional centralized conference by the bitrate of the upstream TFO payload and the extra stream in the return path. However, this is manageable for high-bandwidth terrestrial VoIP networks. The worst-case bandwidth requirement per-link remains fixed at $M + 1$ streams so that network planning and provisioning tasks are simplified. Therefore, even if wide-area multicast conferencing is available, TFO conferencing has the distinct advantage of limiting the endpoint bandwidth requirements. In this way, the TFB serves as a content-aware congestion control mechanism, intelligently forwarding the packets of the primary and secondary talkers while dropping the packets of the unselected conferees. In

a multicast conference, unpredictable packet dropping would occur since it is unlikely that the worst-case bandwidth would be allocated per-link. Additional bandwidth can be saved if the TFB distributed the selected streams back to the endpoints (or a capable subset thereof) via multicasting.

Subjective testing described in Chapter 4 showed that the TFO/TFSS configuration is perceived as slightly better than the multicast conferencing scenario due to a reduction in background noise. In practice this would likely be a moot point since silence suppression/DTX would be used; hence the silent periods would not be transmitted. However, it is not clear what the current practice is in terms of endpoint mixing. For example, RAT does not follow the customary mixing practices described in Section 2.5.4, where silence periods are not included in the conference sum. Thus TFO is advantageous since background noise reduction is inherent to the system.

A drawback of both TFO and multicast conferences is that they require all endpoints to transmit and receive using the same speech codec. This is not a concern for commercial VoIP networks. Most endpoints connect through a gateway, and it is reasonable to assume that most gateways in a given carrier network will support the same speech coding standards. Problems do arise however, when the endpoints make direct connections to the network (e.g., LAN, or Internet conferencing) where different soft- or IP-phones support different codecs. Still, the use of G.729(A) is so widespread that this would present a problem in a small number of cases. An enhancement is possible where a subset of endpoints has more than one codec in common. These endpoints can bundle different encodings of the same speech frame in the upstream packets; the TFB is responsible for distributing the proper encoding type back to a given endpoint.

The results of Section 3.3.6 indicated that a time-level algorithm tuned for switching 20 ms frames provided good conference quality. As expected, Loudest Talker (LT) resulted in voice break-ups due to rapid switching, and First-Come-First-Served (FCFS) led to abrupt and unnatural talker sequencing due to its dogmatic order-based approach. In contrast, the legacy Northern Telecom (NT) and new Tandem-Free Speaker Selection (TFSS) algorithms performed equally as well (and notably better than FCFS) in terms of identifying the "true" current talkers. These time-level algorithms were not plagued with erroneous switching decisions as in FCFS since talking privileges were assigned primarily on the basis of average speech energy, which compensated for the VAD's false triggers. However, TFSS provides better speech quality because it switched between conferees less often than NT—the *rate*

of clipping has a stronger effect on perceived quality than the duration, at least when the clipping durations are long [43].

Another attractive quality of TFSS's less frequent switching is that synchronization between the speech encoder and decoder is maintained over a longer period of time. This is important for heavily state-dependent speech coders such as G.729(A), since distortion occurs every time the first frame following a clip is decoded. Since a switch of current talkers is equivalent to introducing burst frame errors into the unselected stream, it may be desirable to allow the decoder to invoke its built-in Packet Loss Concealment (PLC) scheme (but discard the recovered frame) in order to partially recover the decoder state. For clipping durations exceeding 80–100 ms, it is better to reset the decoder rather than to rely on the built-in repair mechanism since whole phonemes may be lost, rendering the decoder state meaningless. Distortion due to Front-End Clipping (FEC) and Back-End Clipping (BEC) on a particular stream can be mitigated by resetting the decoder prior to synthesizing the first received frame following the clip.

Overall, the TFSS performed well but has some shortcomings. Increases in signal energy were better tracked with NT, yet loud events such as coughing and/or microphone taps caused unwanted preemptions of the current talkers. Conversely, the exponential averaging of TFSS sometimes adapted too slowly and was observed to lag 1–2 frames behind where it ideally should have been, resulting in late switching decisions. Using NT's envelope detector with TFSS's state machine and tuning the barge-in threshold per-state, or on a sliding scale, would likely give better switching decisions. Another option is to simply raise the number of selected speakers to three instead of two. However, selection of three current talkers raises the endpoint complexity. Of course, not all endpoints have to receive the same number of streams. Instead, the TFB could select two streams by default, but could select-and-forward additional streams to a particular endpoint if so requested. Potential solutions for improving TFSS based on $M = 2$ are discussed in Section 5.2.1.

A final comment can be made with respect to performing voice quality testing on a PC-based conferencing test-bed, as was done in Section 4.3. RAT and other freely available software-based conferencing tools are convenient platforms for experimentation, but suffer from poor speech quality since they do not follow advances in speech processing. In particular, the speech coding and silence suppression techniques used by RAT are outdated. These may have a stronger impact on the subjective quality than the speaker selection technique. Many of these problems were improved by adding G.729A to RAT and running

the conferences without silence suppression. However, this increases the end-to-end delay. Therefore, conferencing tools such as RAT should not be used blindly when performing voice quality testing. Workstation-based conferencing platforms do not provide the same grade-of-service as conventional telephony, hence the shortcomings of such platforms need to be overcome before meaningful voice quality testing can be achieved. Appendix A provides the details of the conferencing testbed and the delay performance of the PC-based conferencing facility used in this thesis—it serves as guide for those who may wish to repeat or replicate the experiments outlined in Chapter 4.

5.2 Future Work

This section covers the remaining issues to be solved in terms of the mechanisms needed to make the TFB robust to real-world conditions. Many of the ideas have been touched upon in the body of the thesis, as well as in the previous discussion. Primarily, the recommended work is focussed on speech/signal processing issues, such as speaker selection enhancements and VAD improvements, as well as networking issues such as jitter compensation and TFB support for silence suppression. The last subsection outlines some ideas which could benefit future live subjective testing.

5.2.1 TFSS Algorithm Enhancements

Section 5.1 mentioned that the TFSS algorithm was sometimes prone to late switching decisions. This may be remedied by giving the algorithm a notion of look-ahead. Visual inspection of the conferees' energy signals revealed that if TFSS accounted for where the energy of a talker *might* be in 1–3 frames, the accuracy of the current switching decision could be improved. If the future "energy" is estimated, it, along with the current and past energies, could be combined in a weighted sum which would be then used to establish priority.

During the live conferences (not the simulations) some distortion was heard on the secondary channel when the secondary talker was silent. This was attributed to occasional switching between the silent conferees due to spurious noise such as breaths. Note that this problem was likely due to TFB VAD false alarms. The TFSS algorithm can be improved by using another threshold under which switching never takes place. The "switching threshold" could be based on running estimates of each conferee's noise floor. In the current TFSS

algorithm, no such thresholds were used since it was assumed that each conferee's ε_n acted as such.

5.2.2 Stream and TFSS State Synchronization

The current implementation of the TFB was sufficient for carrying out the live tests between conferees. Since the experiments were carried out over the local LAN, provisions for coping with real-world problems such as jitter and packet loss were not added to the TFB. The TFB (correctly) assumed that packets arrived in order, hence speaker selection updates were triggered by each packet arrival. Stream synchronization mechanisms need to be added to the TFB such that its performance can be studied in adverse network conditions. Relative synchronization needs to be performed on a per-stream basis, as well as cross synchronization between conferees. The former can use a standard playout delay algorithm as outlined in Section 2.3.5. Cross-stream synchronization should be achieved by synchronizing to the primary talker, as suggested in [36]. Note here that state synchronization does not imply that packets will be buffered and delayed by appropriate amounts in order to present the selection module with a set of N packets at the end of each evaluation interval. Rather, it is intended for synchronization of the *states* between conferees, so that current values are not compared to future or past states of the other conferees, leading to erroneous switching decisions. Some of the questions to be answered are: What happens if a packet arrives early or late? What happens when a packet does not arrive at all?

5.2.3 Support for Silence Suppression/DTX

Currently, the TFB expects a full-rate, continuous packet stream from each endpoint. The TFB VAD operates on the average energy field contained in the TFO data frames. However, activity detection can be accomplished by monitoring the arrival of SID frames when a DTX scheme is used. The solution for this problem is dependent on Section 5.2.2. An interesting question to be solved here is, should the TFB drop or forward SID frames to the unselected endpoints? If so, how does this affect the speech decoder?

5.2.4 TFB Output Rate Control

As outlined in Section 3.1.1, a rate controller is required to ensure that only M packets are forwarded during each evaluation interval. The problem stems from the instantaneous

nature of the switching algorithm. Consider the case where the number of simultaneous talkers is limited to one ($M = 1$). A case may arise where an unselected conferee's packet, upon its arrival, will be selected as the current talker and forwarded to the conference endpoints. However, when the packet from the previously selected talker arrives (during the same evaluation interval), the selection algorithm may re-select this conferee as the current talker, and subsequently transmit the packet to the endpoints. Therefore, *two* packets have been sent instead of one during the same evaluation period.

5.2.5 Improved TFB VAD

Currently, the TFB VAD is a slightly enhanced version of the RAT VAD. This VAD has a high rate of false alarms ($\approx 30\%$). Luckily the TFSS algorithm is somewhat resilient to such a condition. However, poor VAD decisions can lead to poor switching decisions. A robust energy-based VAD, which has comparable performance to a modern VAD such as G.729B, needs to be developed for the TFB. It is recommended that features such as $\bar{E}_l$ and $\bar{E}_h$ be used, as suggested in [47]. This would require a quantization scheme for these values, as well as an update to the TFO data frame proposed in Section 3.4.3. The overall TFO payload format would likely remain the same.

5.2.6 Gain Equalization

Modern digital conference bridges perform automatic gain control on each received stream in order to equalize speech level contrasts between conferees. Speech level contrasts—whether due to source equipment (e.g., microphone gain), or naturally loud or quiet talkers—also pose a problem to a level-based speaker selection algorithm such as TFSS. Then, in the context of this thesis, gain equalization needs to be performed on the $\bar{E}_f$ of the TFO data frame, prior to speaker selection. A typical solution is to choose a global reference value, and normalize each energy relative to this value. Scaling factors used in the normalization process are derived based on the recent history of the observed energy [4].

5.2.7 Experimental Design

Evaluation results drawn from the experiments are somewhat preliminary due to the small sample size used in the live scenarios. However, subjective opinions were consistent for each given system, and the debriefing periods revealed that conferees usually perceived

broad differences between each of the two systems under comparison. Further testing with a larger subject pool, coupled with more precise data collection in the form of well-planned questionnaires, could produce a ranking system over a range of values, revealing the magnitude of difference between systems. This would be particularity useful for Comparison 2, where only a slight preference for TFSS was found. It is expected that further testing with four-person conferences would not change the rankings of Table 4.7.

Another limiting factor in the experiments was the size restriction of four conferees. Larger conferences have the effect of producing a higher rate of interruptions and increasing the low-level noise. These two attributes are particularity interesting for further investigating performance bounds of TFSS. In the context of TFSS with G.729A, short and frequent interruptions *could* sufficiently desynchronize the decoder state to the point of a distinct reduction in speech quality. On the other hand, the reduction of background noise from idle conferees would be more apparent with the use of TFSS, assuming silence suppression/DTX is not used. It is suspected that speech output and interaction between conferees for a 8–12 person conference would be adequate for studying a conferencing system designed to carry a much larger number of conferees, such as 50 [23]. Of course, factors such as background noise from 32 additional stations would have to be artificially inserted to each output stream.

Appendix A

Conferencing Facility

Many of the challenges and encountered during this thesis stemmed from the actual implementation and configuration of the Tandem-Free Operation Bridge (TFB) and the conferencing facility. The TFB was custom-built for this thesis, while a modified third-party conferencing tool, Robust Audio Tool (RAT), was used as the endpoint. Initial trials with RAT in point-to-point mode revealed that the platform was not suitable for voice quality testing, primarily due to a lack of high quality codecs, a poorly configured Voice Activity Detector (VAD), and a lack of Comfort Noise Generation (CNG). Running RAT without silence suppression fixed the latter two problems but led to higher end-to-end delays. Therefore, much of the early work was devoted to "fine-tuning" the conferencing software and selecting suitable platforms on which to run the TFB and RAT. This appendix provides a detailed summary of the mechanical aspects of the work[1].

The appendix is divided into two sections. Section A.1 is informational in nature, and presents the implementation details of the conferencing testbed. This includes a review of the modified RAT and the construction of the TFB. Section A.2 analyzes the end-to-end delay between two RATs on Windows 2000 (Win2k) and Linux platforms.

[1] An excellent resource for Internet conferencing research is at the homepage of the Network and Multimedia Research Group at University College London (UCL)—the collected works include software, human factors studies, platform issues, as well as audio quality and transmission issues.

A.1 System Implementation

The primary function of the testbed was to facilitate the performance evaluation between different conferencing systems of interest. Issues such as protocol compliance and robustness to high degrees of jitter were deemed less important than the ability to compare the speech quality of the Tandem-Free Operation (TFO), Conventional Bridge (CB), and Multicast (MCAST) conferencing models of Section 4.1. The "best case" performance was desired such that extraneous effects did not interfere with the evaluations. The work involved RAT enchancements, Real Time Protocol (RTP) modelling and TFB construction, and workstation hardware.

A.1.1 RAT—the TFO Endpoint

Recent efforts by several institutions have produced a variety of software for desktop conferencing. Three of the most sophisticated tools from a research perspective are the Video Audio Tool (VAT), Network Voice Terminal (NeVot), and the RAT. Each of these tools were developed for lightweight Multicast Backbone (MBONE) conferencing and experimentation. At the time of writing this thesis, RAT was the state-of-the-art in its field, and so was adopted as the endpoint for the conferencing testbed.

RAT provides a number of desirable features for conferencing research. It is compliant with the RTP standards [7, 17], and can participate in point-to-point or multicast conferences. No special hardware is required aside from the network interface and soundcard. The tool implements a novel algorithm for improving the output audio quality in the event of workstation scheduler anomalies [66]. Packet playout delay is adapted per-talkspurt using the well-known algorithm from [14], but strict lower and upper bounds can be set by the user. The tool provides the option of using an energy-based silence-suppression scheme that operates independently of the speech codec. RAT provides a standard assortment of publicly available speech coders such as μ-law or a-law (64 kbps), GSM-FR (13 kbps), and G.726 (16–40 kbps).

It was decided that RAT version 4.2.10 was to be used as the endpoint in the conferencing experiments. However, a few modifications were necessary in order to support the live evaluations.

Supporting G.729A

G.729A was added to RAT to ensure that the system evaluations for speech quality were not compromised by a poor speech codec. Furthermore, the G.729A interface was modified to accept speech frames of 20 ms instead of 10 ms, so that the packet departure rate was equal to RAT's other codecs. Note that equivalent RTP functionality could have been achieved by setting the RTP payload length directly from the RAT interface; however, this would not have been obvious to the untrained conferee, and therefore it was ruled out in favour of the former scheme.

In general, RAT does not perform well when transmitting a single 10 ms speech frame per RTP packet. This is because RAT sets the soundcard fragment size to 20 ms (320 bytes), resulting in two back-to-back RTP transmits with 10 ms payloads. This makes every packet received at the far-end RAT appear to have 10 ms of jitter, causing unnecessary playout delay.

Controlling Playout Delay

Informal testing with RAT revealed that workstation-induced jitter often caused the adaptive playout buffer to rapidly approach 200 ms or more. Since RAT was always used in continuous rate mode, there were no periods of silence during which the algorithm could adjust the playout delay. The delay persisted unless the RAT receiver was reset by toggling the "listen" button. To compensate, the RAT interface was modified so that the maximum playout delay could be set between 0–2 seconds, instead of 1–2 seconds. With this change, the maximum playout delay was typically set to 60 ms. In addition, a flag was added that allowed for the adaptive cushion algorithm to be bypassed; the parameter was set once per session by reading its value from the RAT configuration file.

Conventional VoIP-Bridge Emulation

To compare the conference quality of a conventional VoIP bridge to that of the TFB, it was necessary to build or emulate the former. Ultimately, the build option was ruled out in favour of an emulation. An additional encode-decode step was added to each RAT following the mixing operation. The codec used in the tandem operation was the same as in the normal transmit path. A single button, labelled "tandem mode" was added to the

RAT interface so that the emulation could easily be turned "on" or "off" while a session was in progress. This allowed conferees to easily compare the difference in speech quality.

RTP Payload Emulation

The final modification to RAT involved the calculation and transport of the speech level estimate, $\bar{E}_f$, per-packet. Instead of adding support for the proposed blocked format payload type recommended in Section 3.4.4, an unquantized $\bar{E}_f$ was transported in the RTP header extension. The use of the header extension is ideal for experimental purposes since it allows new RTP payloads and/or features to be tested independently of normal processing. Neither the F- nor V-bits were necessary because the length of the payload was always 20 ms, and the TFB was used for speech detection.

A.1.2 RTP Connections and Translations

A traditional VoIP conference bridge establishes a one-to-one signalling and media connections between itself and each endpoint. Therefore, the bridge becomes the media signal source, and each endpoint need only keep one receive path. In the TFO model each endpoint must maintain a receive path for all other $N-1$ conferees so that the incoming packets can be demultiplexed to the correct jitter buffer and speech decoder, for example. Multicast conferencing tools such as RAT have this exact ability, and typically demultiplex incoming streams according to the RTP Synchronization Source (SSRC). To leverage this ability of RAT, the TFB was modelled as an RTP reflector, thereby emulating the connections in a multicast conference.

In the reflector/translator model, the speech and control packets flowed through TFB, but the logical RTP connections are established directly between conference endpoints. The TFB replicates and transmits the selected M streams back to the endpoints. The TFB translates certain fields in the RTP header and in the RTCP reports. In essence, the original packet stream generated by a selected talker is passed on to the receiver. This approach keeps the TFB light, since RTP sessions are not explicitly managed. Relatively simple translation duties require the TFB keep state per-conferee.

When an RTP packet is received by the TFB, the header information is nearly identical to what is replicated and returned to the conferees. However, some fields must be translated so they can be interpreted correctly at the receiver. Essentially, the TFB can modify the

received packet in place, and copy the resultant packet to the output buffer for transmission. The RTP translations required by the TFB are:

1. Sequence Number Translation: An RTP transmitter increments the packet sequence number by one for every packet that is transmitted. Again, since the TFB drops packets, gaps will appear in the sequence numbers. Therefore, the TFB must re-assign sequence numbers to the selected, outgoing packets such that it appears that no selective dropping has occurred. For each conferee, this is accomplished by replacing the original sequence number with a locally generated one, which is incremented for each successive transmission.

2. Marker bit Translation: Normally, when an RTP transmitter uses silence suppression/DTX, the first packet of a talkspurt has its marker bit set. This is to indicate that the time gap seen between itself and the previous packet was intentional, rather than due to loss. Therefore, when a new conferee is assigned talking privileges, the TFB *must* set the marker bit of the first packet of the ensuing packet stream. This applies for both continuous and silence suppression/DTX modes of communication.

3. Payload Type Translation: During session initialization, an 8-bit payload type will be associated with the upstream channel, identifying the TFO payload type. This number will be used in the payload type field of all upstream packets. However, a different payload type will be associated with the downstream channel, corresponding to the underlying speech codec. The TFB must translate payload type field of outgoing packets to reflect the payload of the source speech codec.

4. RTCP Translation: Measures for translating RTCP reports are outlined in [7]. To summarize, the translations involve proper translation of the Sender Report (SR) "sender's byte" count, and "sender packet count" fields, as well as the "extended highest sequence number received" field of the report blocks. The corrected values of these fields can be derived in a straightforward fashion with respect to the RTP translations outlined above.

Of these four translations, only the sequence number and marker-bit translations were implemented—the others were omitted since they did not affect the performance of the overall system and hence had no experimental value. For example, the RTP payload type

field was always set to 0 (G.711) or 18 (G.729A) since the TFO data was carried in the header extension. RTCP translations were not supported since RAT uses RTCP reports for display purposes only.

A.1.3 TFB Implementation

In comparison to a traditional VoIP conference bridge, the TFB was straightforward to realize in software. As described in Chapter 3, the TFB was implemented with a minimal amount of RTP/RTCP handling, a VAD, and a speaker selection function; no playout buffers or speech codecs were required. The TFB did not support SIP or H.323 call signalling, but rather followed RAT's strategy and added or deleted conferees based on RTP/RTCP messages.

The TFB was implemented in C and used parts of the RTP library from UCL. The functionality of the TFB was divided into one of four objects: Session, Receive, Selector, and Transmit. The role of the Session was to provide a global structure that tied everything together. The Receive module implemented a normal UDP/IP socket interface and cast the received packets as a typical RTP/RTCP data structure. State information for each conferee was stored in a Host list maintained by the Selector module. Each Host object included an RTP/RTCP state, a VAD and several speaker selection parameters, as well as other conferee-specific "bookkeeping" data. The Transmit module received the selected packets from the Selector and reflected them back to each conferee except the source. A single loop implemented a receive-select/drop-replicate/transmit pipeline which was executed on every packet arrival.

TFB Data Flow

Received RTP packets were stored in a packet queue owned by the Selector. Conferees were demultiplexed according to their SSRC identifier; if a packet arrived with an unknown SSRC, a new conferee object was created and saved in a list with the others. Reception of an RTCP BYE packet triggered the deletion of an active conferee.

Next, control moved to the Selector, where a given selection algorithm was executed once per packet. The selection algorithm used the current packet to update the state of the corresponding conferee, then prioritized the conferee relative to the *last* known state of the other conferees. In this way, jitter buffers were not needed since packets were immediately

dropped or forwarded, decreasing the overall end-to-end delay.

Speaker selection was accomplished with one of the First-Come-First-Served (FCFS), Loudest Talker (LT), or Tandem-Free Speaker Selection (TFSS) techniques, which were outlined in Section 3.3. Each algorithm produced a sorted priority list of Host objects; if the SSRC of the current packet was found in one of the top M priority positions, the packet was immediately forwarded, otherwise it was dropped. As outlined in Section A.1.2, the sequence number and optionally the marker bit of the RTP packet was updated before transmission to the other conferees. This was to prevent the endpoints from unnecessarily invoking packet loss recovery mechanisms or extending the playout delay.

Following the selection and sequence number processing, each selected packet was passed to the Transmit module, which sent a copy of the packet to each of the other conferees. Once this was done, control was passed back to the main processing loop where the process was repeated on the next packet arrival.

A.1.4 Conference Hardware

The selection of hardware is critical when constructing a PC-based conferencing facility. In general, low-cost PC microphones, earphones, and speakers are geared towards the casual user and are not suitable for speech quality evaluations. Speech quality at every station can be compromised even if one soundcard has a low SNR. In addition, some soundcards introduce additional delay due to large, fixed fragment sizes.

Microphones and Headphones

In general, it was observed that hand-held and boom microphones were not suitable for PC-based conferencing. With such equipment, it was difficult for some conferees to maintain a constant distance between the microphone and their mouths, resulting in rustling and frequent fluctuations in speech volume. Therefore, each of the conference stations was equipped with a headset, two of which were off-the shelf products from Plantronics, while the other two were custom-made. The custom-made headsets were built by attaching a lavalier microphone to a flexible boom, which in turn was fastened to a pair of closed-back AudioTechnica studiophones. Informal tests showed that the new headsets had a positive effect on the overall conference quality.

Soundcards

Each PC was equipped with a 16-bit full-duplex SoundBlaster card from Creative Labs. There were two PCI64 cards, one PCI128, and one LIVE!; the SNRs for the PCI and LIVE! cards are rated at 90 and 96 dB, respectively.

Workstations

The conference facility was composed of the five workstations shown in Table A.1. The TFB ran on Windows 2000 (Win2k) while the RAT endpoints ran on Linux (kernel 2.2.16). Each Linux station had no difficulty supporting RAT and the G.729A codec, even when the tandeming feature was used. A former facility was based on Windows 95 (Win95) machines with 200 MHz Pentium II processers and 32 MB RAM, but was replaced due to lack of real-time capability. The following section highlights some of the practical problems encountered with the conferencing facility.

Table A.1 Workstation configurations for conferencing experiments.

Role	Platform (OS)		CPU (MHz/Class)	RAM (MB)	Sound Card (SoundBlaster)	Headset
TFB	Windows	2k	800/PIII	256	—	—
RAT	Mandrake	7.1	400/Celeron	164	PCI64	Custom
	Mandrake	7.1	400/Celeron	128	PCI64	Plantronics HS-1
	Red Hat	6.2	550/PII	128	PCI128	Plantronics HS-1
	Mandrake	7.1	266/PII	128	LIVE!	Custom

A.2 RAT End-to-End Delay

In general, the total round-trip delay of a full-duplex voice communication system is considered good for most user applications if it is <300 ms, and acceptable if in the range of 300–800 ms. Round-trip delays above 800 ms are unacceptable [24]. However, even in a "perfect network", such as the LAN testbed, there is a significant delay imposed by the Operating System (OS) of a multitasking workstation. Lack of real-time support in the Win2k and Linux operating systems resulted in additional delay due to interrupt latency and the scheduler. For example, interrupt latency may add jitter to the transmit packet

stream since a departure can only occur when a frame of input audio has been received from the soundcard. Interrupt latency may also affect the transfer of packets from the network interface card (NIC) to the application. The scheduler adds latency due to unpredictable timing, which may delay the delivery of audio from the application to the soundcard. As a result, this may cause the device to run dry, followed by a backlog of audio due to the period of starvation stealing real-time from the output stream. When taken together, these events may easily yield a total round-trip delay beyond the 300 ms threshold.

Long round-trip delays $\geq$1 s were observed during initial trials with a Windows-based conferencing system. It was found that the delay tended to grow as the conference wore on, and that the performance was generally independent of CPU speed. The creeping delay was due to audio backlog in the workstation audio driver since RAT was run without silence suppression. RAT stations were then replaced with the Linux-based facility summarized in Table A.1, giving a noticeable improvement. It was found that shutting down non-essential applications, and careful handling the RAT user interface could yield reasonable performance on this system. Measurements were then taken on identical Win2k and Linux PCs in order to fully characterize each component of the system delay for each OS.

A.2.1 Total Round-Trip Delay

The round-trip delay measurements were obtained using a well-known technique used for benchmarking the latency of a full-duplex voice communication system. The test required the use of two conference stations and a recording device, such as another PC with a soundcard. One station was used as a source, while the other acted as a loopback. The general idea was to transmit a reference signal from the source to the loopback such that a delayed version of the reference signal was received at the source. The delay was calculated by measuring the time shift between the original and received signals, and *adding* the A/D and D/A converter delay at the source station.

A point-to-point connection between two RATs was established using two 400 MHz/Celeron machines shown in Table A.1. At the source station, the audio loopback feature of the soundcard was enabled such that the microphone input was both transmitted to the loopback station and coupled to the speaker output. On the loopback station, a stereo cable was used to physically connect the speaker output to the line-in, causing the received audio to be sent back to the source. The speaker output of the source was connected to the line-in

of a third computer for recording. A series of clicks were presented to the microphone of the source station, and the resultant combination of original and delayed clicks was recorded on the third computer. The tests were conducted without silence suppression and using the GSM-FR codec (G.729A had not yet been added to RAT). In the end, the total round-trip delay for Windows and Linux was found to be 433 and 289 ms, respectively.

A.2.2 Network and Jitter Buffer Delay

While the clicks was being recorded, the Round-Trip Time (RTT), jitter buffer delay, and jitter statistics were recorded from the user interfaces of each RAT. Note that the RTT represents the round-trip *packet* time between the two RATs, not the round-trip audio delay. On the LAN testbed, the RTT accounts for the (negligible) propagation delay and the RTP/UDP/IP processing overhead. RAT obtains the RTT measurements from the RTCP reports as outlined in [7], while the jitter and playout delay is calculated as in [14]. The one-way network delay, shown in Table A.2 was simply calculated as $\overline{\text{RTT}}/2$. This was acceptable since there was no difference in the transmit and receive paths, and the background traffic was essentially zero.

The average jitter was found to be 22 and 9 ms for the Win2k and Linux platforms, respectively. This was interesting since there was essentially no background traffic, and the switched Ethernet implied the workstations did not share bandwidth. Therefore, the network jitter should have been non-existent. Since the only other components of the test bed that created timing variability were the workstations themselves, it was clear that workstation jitter added significant variable delay to packet arrivals and departures. This is not surprising given that the OS for a multitasking workstation was not designed for real-time voice communication.

A.2.3 Audio I/O Delay

The round-trip audio I/O delay was measured using a technique similar to that used to measure the total round-trip delay. An audio loopback program was written in C for both Windows and Linux that facilitated measurement of the round trip audio delay between the soundcard and an application. The tool was based on the RAT audio drivers so that fair comparison could be made between each OS.

The software loopback read 20 ms frames of audio from the soundcard and immediately

wrote them back out. To measure the round trip audio delay, a series of clicks were presented to a microphone such that they travelled through the software loopback and the hardware loopback on-board the soundcard. The delay was calculated by measuring the distance between the two superimposed waveforms and subtracting the frame length. The average delay was found to be approximately 39.8 and 11.4 ms for Win2k and Linux platforms, respectively. However, the delay was observed to be very sensitive to the use of other applications such as a web browser, as shown in Fig. A.1.

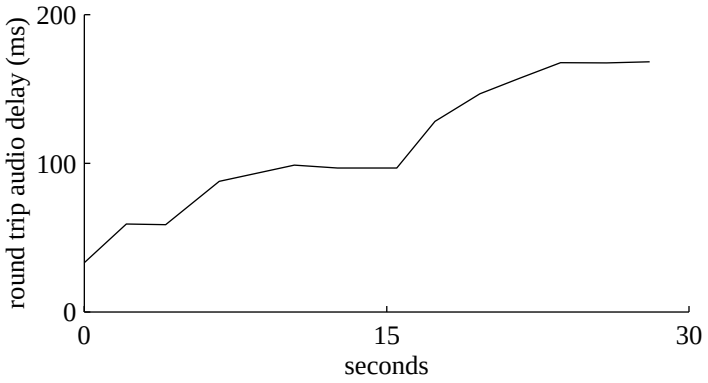


Fig. A.1 Round trip audio delay on a multitasking workstation during web-browsing. This specific result was generated using a Win2k machine. Note that the browsing was "busy", in that a new site was visited immediately after the current site had finished loading.

A.2.4 Discussion of Delay Measurements

Table A.2 provides a summary of the delay measurements. This table shows both the sum of the individually measured delays and the total end-to-end delay. The Windows system exceeds the 150 ms ITU-T threshold for "good" voice quality by 65.5 ms, while the Linux system is a mere 5.5 ms below it. It is important to note here that the measurements shown in Table A.2 were taken under ideal operating conditions, in that the systems were not running any applications other than RAT or the loopback program. Therefore, they represent an approximation to the *best case* end-to-end delay achievable with RAT running on a multitasking workstation.

According to Table A.2, there was a 33% reduction in delay by running RAT on Linux instead of Win2k. The major differences in delay were between the playout buffer and the

Table A.2 Summary of one-way delay between two RATs. All values are reported in ms.

Delay Source	Operating System	
	Windows	Linux
Playout Buffer	117.0	85.0
Audio I/O	39.8	11.4
Frame Duration	20.0	20.0
Network	8.7	7.5
Encode/Decode	0.4	0.4
A/D & D/A	1.0	1.0
Total	186.9	125.3
Actual (Measured)	216.5	144.5

audio I/O. However, since the playout delay was a function of the network jitter, which in turn was function of the workstation jitter, the implication is that this difference was due to the audio I/O. Note that the sum of individually measured delays is slightly less than the actual end-to-end delay. This difference is 29.6 and 19.2 ms for Win2k and Linux platforms, respectively, and it represents the additional delay incurred due to dependencies between components, plus some error due to the approximations in the measurements. This difference implies that there was more workstation jitter on Win2k than on Linux. This claim is supported by the difference in jitter between the two OSs shown in Section A.2.2.

Effect of Audio Input Latency

The Windows and Linux versions of RAT differ in the way that audio input is performed; the Linux version uses polling while the Windows version uses a threaded callback function. The Windows method is subject to additional delay since it takes time for the host scheduler to wake up the thread in order to deliver the audio, i.e., the additional context switch adds latency to the system. However, input audio latency, or audio interrupt latency, was the main culprit responsible for adding jitter to the transmit packet stream. Since the packet departure process of RAT was tied to the audio input function, packets were sent only when a complete frame of audio was received from the soundcard. Observations of RAT on a Win2k platform confirmed this; packets often arrived at the receiver in groups of two

or three.

Effect of Windows KMixer

In Windows, all audio output must travel through the Kernel Mode Mixer (KMixer), the audio subsystem software mixer. It is a well-known fact that the KMixer adds about 30 ms of additional delay to an output stream [63]. However this delay can grow when the KMixer adapts after periods of starvation, yielding output latencies of 50–75 ms [64, 87]. This inherent latency of the Win2k audio subsystem accounts for the large discrepancy in round-trip audio delays between the two OSs.

A.2.5 Summary of System Delay

In short, the system was limited due to the PC platforms and RAT. The total delay of each system seems long, while in reality it was much lower than in other popular PC telephony applications. For example, NetMeeting and SpeakFreely has been reported to yield end-to-end delays of 285 and 1200 ms, respectively [88]. Since the field of PC telephony is not fully developed, real-time support is not a standard feature on multitasking workstations. Until this is changed, software-based, PC-conferencing tools will barely achieve toll-quality performance in terms of end-to-end delay.

Workstation jitter was measured implicitly in terms of network jitter, and was shown to be more severe on Win2k platforms than on Linux platforms. In addition, more delay was unaccounted for on Windows, implying that this OS created more variability in interrupt and scheduler latencies when RAT ran as a whole. For these reasons, Linux was chosen as the platform for the real time trials. Proposals for improved real-time support are currently underway for Windows, while a kernel patch for Linux already restricts audio interrupt latencies to be $\leq$2 ms [65].

Appendix B

Processing Delay and Complexity

This appendix provides the CPU% measurements used in Section 4.2 for estimating the bridge and endpoint complexity of the Tandem-Free Operation (TFO), Conventional Bridge (CB), and Multicast (MCAST) conferencing systems. In each case, the CPU% was determined by dividing the running time of each component by the total number of frames processed, and the frame duration. Below, processing delay corresponds to 20 ms packets.

Table B.1 G.711 and G.729A processing delay and CPU utilization.

CPU	G.711				G.729A			
	Encoder		Decoder		Encoder		Decoder	
(MHz/Class)	(ms)	(CPU%)	(ms)	(CPU%)	(ms)	(CPU%)	(ms)	(CPU%)
266/PII	0.006	0.026	0.004	0.019	1.768	8.839	0.434	2.169
400/Celeron	0.005	0.023	0.003	0.016	1.165	5.827	0.291	1.457
800/PIII	0.002	0.009	0.002	0.008	0.576	2.882	0.140	0.699

Table B.2 TFSS and mixing processing delay and CPU utilization.

CPU	TFSS		Mix	
(MHz/Class)	(ms)	(CPU%)	(ms)	(CPU%)
266/PII	0.0110	0.0138	0.0031	0.0155
400/Celeron	0.0074	0.0092	0.0016	0.0078
800/PIII	0.0034	0.0043	0.0008	0.0039

References

[1] J. Bellamy, *Digital Telephony*. New York, NY: John Wiley & Sons, Inc., second ed., 1991.

[2] M. A. Hashemi and G. P. Pucci, "Telephone conference system with active analog conference." United States Patent 4,139,731, Feb. 1979.

[3] L. A. Baxter, P. R. Berkowitz, C. A. Buzzard, J. J. Horenkamp, and F. E. Wyatt, "System 75: Communications and control architecture," *AT&T Technical Journal*, vol. 64, pp. 153–173, Jan. 1985.

[4] M. A. Marouf and P. W. Vancil, "Method and apparatus for controlling signal level in a digital conference arrangement." United States Patent 4,499,578, Feb. 1985.

[5] J. Rosenberg and H. Schulzrinne, "Models for multi-party conferencing in SIP." Internet Draft, Internet Engineering Task Force—Work in Progress, Nov. 2000.

[6] K. Singh, G. Nair, and H. Schulzrinne, "Centralized conferencing using SIP," *Proc. 2nd IP-Telephony Workshop (IPTel2001)* (New York, NY), Apr. 2001.

[7] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications." RFC 1889, Internet Engineering Task Force, Jan. 1996.

[8] H. Schulzrinne and J. Rosenberg, "The Session Initiation Protocol: Internet-centric signaling," *IEEE Communications Magazine*, vol. 38, pp. 134–141, Oct. 2000.

[9] ITU-T Recommendation H.323, "Packet-Based Multimedia Communication Systems (prepublication version)," Nov. 2000.

[10] Radvision, "MP Platform Product Overview." [online] http://www.radvision.com/f_products/f3_mp.php3?prod=MP+Platform, May 2001.

[11] B. Michael, "Network based media servers: The next generation." [online] http://www.computertelephony.com/article/CTM20010326S0007/7, Apr. 2001.

[12] P. Smeulders and D. Dal Farra, "Method of providing conferencing in telephony." Canadian Patent Application 2,224,541, July 1998.

[13] W. A. Montgomery, "Techniques for packet voice synchronization," *IEEE J. Selected Areas Communications*, vol. SAC-1, pp. 1022–1028, Dec. 1983.

[14] R. Ramjee, J. Kurose, D. Towsley, and H. Schulzrinne, "Adaptive playout mechanisms for packetized audio applications in wide area networks," *Proc. of the Conf. on Computer Communications (IEEE-Infocom)* (Toronto, Canada), vol. 2, pp. 680–688, June 1994.

[15] S. B. Moon, J. Kurose, and D. Towsley, "Packet audio playout delay adjustment: performance bounds and algorithms," *ACM/Springer Multimedia Systems*, vol. 5, pp. 17–28, Jan. 1998.

[16] Y. J. Liang, N. Färber, and B. Girod, "Adaptive playout scheduling using time-scale modification in packet voice communications," *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (Salt Lake, UT), vol. 3, pp. 1445–1448, May 2001.

[17] H. Schulzrinne, "RTP profile for audio and video conferences with minimal control." RFC 1890, Internet Engineering Task Force, Jan. 1996.

[18] J. Davidson and J. Peters, *Voice Over IP Fundamentals*. Indianapolis, IN: Cisco Press, 2000.

[19] U. Black, *Voice Over IP*. Upper Saddle River, NJ: Prentice Hall, 2000.

[20] R. V. Cox and P. Kroon, "Low bit-rate speech coders for multimedia communication," *IEEE Communications Magazine*, vol. 34, pp. 34–41, Dec. 1996.

[21] D. O'Shaughnessy, *Speech Communications: Human and Machine*. New York, NY: IEEE Press, second ed., 2000.

[22] R. Salami, C. Laflamme, J. P. Adoul, A. Katoaka, S. Hayashi, T. Moriya, C. Lamblin, D. Massaloux, S. Proust, P. Kroon, and Y. Shoham, "Design and description of CS-ACELP: A toll quality 8 kb/s speech coder," *IEEE Trans. Speech and Audio Processing*, vol. 6, pp. 116–130, Mar. 1998.

[23] J. Forgie, C. Feehrer, and P. Weene, "Voice Conferencing Technology Final Report," Tech. Rep. DDC AD-A074498, M.I.T. Lincoln Lab., Lexington, MA, Mar. 1979.

[24] ITU-T Recommendation G.114, "One-way Transmission Time," May 2000.

[25] ETSI, "Digital Cellular Telecommunications System (Phase 2+): Inband Tandem-Free Operation (TFO) of Speech Codecs; (GSM 08.62 version 7.0.0 Release 1998)."

[26] H. Pon, R. Rabipour, and C. C. Chu, "System for TDMA mobile-to-mobile VSELP CODEC bypass." United States Patent 6,185,424, Feb. 2001.

[27] H. G. Kang, H. K. Kim, and R. V. Cox, "Improving transcoding capability of speech coders in clean and frame erasured channel environments," *Proc. IEEE Workshop on Speech Coding* (Delevan, WI), pp. 78–80, Sept. 2000.

[28] S. G. Pitroda, "Conferencing arrangement for use in a PCM system." United States Patent 4,031,328, June 1977.

[29] J. F. Regan and B. A. Mehta, "Method and means for digital conferencing." United States Patent 4,267,593, May 1981.

[30] E. A. Munter, "Digital conference circuit and method." United States Patent 4,387,457, June 1983.

[31] B. M. Comstock, "Digital teleconferencing control device, system and method." United States Patent 4,541,087, Sept. 1985.

[32] T. R. Hsing, "Framed digital voice summing for teleconferencing." United States Patent 4,658,398, Apr. 1987.

[33] ITU-T Recommendation G.172, "Transmission Plan Aspects of International Conference Calls," Nov. 1988.

[34] T. G. Champion, "Multi-speaker conferencing over narrowband channels," *Proc. IEEE Military Communications Conf.* (Washington, D.C.), pp. 1220–1223, Nov. 1991.

[35] D. Nahumi, "Conferencing arrangement for compressed information signals." United States Patent 5,390,177, Feb. 1995.

[36] N. K. Burns, P. K. Edholm, and F. F. Simard, "Apparatus and method for packet-based media communications." Canadian Patent Application 2,319,655, June 2001.

[37] R. Rabipour and P. Coverdale, "Tandem-free VoX conferencing." Internal memo, Nortel Networks, Montreal, Canada, Aug. 1999.

[38] R. Salami, C. Laflamme, B. Bessette, and J. P. Adoul, "ITU-T annex A: Reduced complexity 8 kb/s CS-ACELP codec for digital simultaneous voice and data," *IEEE-COMMAG*, vol. 37, pp. 56–63, Sept. 1997.

[39] O. Hodson and C. Perkins, "Robust Audio Tool (RAT) version 4." [online] http://www-mice.cs.ucl.ac.uk/multimedia/software/rat, Nov. 2000.

[40] S. Casner and V. Jacobson, "Compressing IP/UDP/RTP headers for low-speed serial links." RFC 2508, Internet Engineering Task Force, Feb. 1999.

[41] M. E. Perkins, K. Evans, D. Pascal, and L. A. Thorpe, "Characterizing the subjective performance of the ITU-T 8kb/s speech coding algorithm — ITU-T G.729," *IEEE Communications Magazine*, vol. 37, pp. 74–81, Sept. 1997.

[42] P. T. Brady, "A technique for investigating on-off patterns of speech," *Bell System Technical Journal*, vol. 44, pp. 1–22, Jan. 1965.

[43] J. G. Gruber and N. H. Le, "Performance requirements for integrated voice/data networks," *IEEE J. Selected Areas Communications*, vol. SAC-1, pp. 981–1005, Dec. 1983.

[44] J. D. Tardelli, P. D. Gatewood, E. W. Kreamer, and P. A. La Follette, "The benefits of multi-speaker conferencing and the design of conference bridge control algorithms," *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (Minneapolis,USA), vol. 2, pp. 435–438, Apr. 1993.

[45] J. F. Lynch Jr., J. G. Josenhans, and R. E. Crochiere, "Speech/silence segmentation for real-time coding via rule based adaptive endpoint detection," *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (Dallas, Texas), pp. 31.7.1–31.7.4, Apr. 1987.

[46] A. Benyassine, E. Shlomot, H. Y. Su, D. Massaloux, C. Lamblin, and J. P. Petit, "ITU-T recommendation G.729 annex B: A silence suppression scheme for use with G.729 optimized for V.70 digital simultaneous voice and data applications," *IEEE Communications Magazine*, vol. 37, pp. 64–73, Sept. 1997.

[47] N. Avadhanam, V. M. Vijayachandran, M. Mathew, and Y. Yoganandam, "A low complexity voice activity detector and comfort noise generator," *Proc. Int. Conf. on Signal Processing Applications and Technology* (Dallas, TX), Oct. 2000.

[48] TIA/EIA, TR45 (1996), "Enhanced variable rate codec, speech service option 3 for wideband spread spectrum digital systems." PN-3292—to be published as IS-127, Jan. 1996.

[49] ETSI, "Digital cellular telecommunications system (Phase 2+); Voice Activity Detector (VAD) for Adaptive Multi-Rate (AMR) speech traffic channels; General Description; (GSM 06.94 version 7.1.0 Release 1998)."

[50] F. Alvarez-Cuevas, M. Bertran, F. Oller, and J. Selga, "Voice synchronization in packet switching networks," *IEEE Network*, pp. 20–25, Sept. 1993.

[51] J. Pinto and K. J. Christensen, "An algorithm for playout of packet voice based on adaptive adjustment of talkspurt silence periods," *Proc. IEEE Conf. Local Computer Networks* (Lowell, MA), pp. 224–229, Oct. 1999.

[52] N. Figueira and J. Pasquale, "VirtualQueue: A technique for packet voice stream reconstruction," *Proc. IEEE Int. Conf. on Multimedia Computing and Systems* (Florence, Italy), vol. 2, pp. 312–316, June 1999.

[53] T. J. Kostas, M. S. Borella, I. Sidhu, G. M. Schuster, J. Grabiec, and J. Mahler, "Real-time voice over packet-switched networks," *IEEE Network*, vol. 12, pp. 18–27, Jan.-Feb 1998.

[54] C. J. Weinstein and J. W. Forgie, "Experience with speech communication in packet networks," *IEEE J. Selected Areas Communications*, vol. SAC-1, pp. 963–980, Dec. 1983.

[55] A. Watson and M. A. Sasse, "Evaluating audio and video quality in low-cost multimedia conferencing systems," *Interacting with Computers*, vol. 8, no. 3, pp. 255–275, 1996.

[56] C. Montminy and T. Aboulnasr, "Improving the performance of ITU-T G.729A for VoIP," *Proc. IEEE Int. Conf. on Multimedia and Expo* (New York, NY), pp. 433–436, July 2000.

[57] C. Perkins and O. Hodson, "Options for repair of streaming media." RFC 2354, Internet Engineering Task Force, June 1998.

[58] J. Rosenberg and H. Schulzrinne, "An RTP payload format for generic forward error correction." RFC 2733, Internet Engineering Task Force, Dec. 1999.

[59] C. Perkins, I. Kouvelas, O. Hodson, V. Hardman, M. Handley, J. Bolot, A. Vega-Garcia, and S. Fosse-Parisis, "RTP payload for redundant audio data." RFC 2198, Internet Engineering Task Force, Sept. 1997.

[60] E. Mahfuz, "Packet loss concealment for voice transmission over IP networks," Master's thesis, McGill University, Montreal, Canada, Sept. 2001.

[61] O. Hodson, C. Perkins, and V. Hardman, "Skew detection and compensation for Internet audio applications," *Proc. IEEE Int. Conf. on Multimedia and Expo* (New York, NY), July 2000.

[62] S. B. Moon, P. Skelly, and D. Towsley, "Estimation and removal of clock skew from network delay measurements," *Proc. of the Conf. on Computer Communications (IEEE-Infocom)* (New York, New York), pp. 227–234, Mar. 1999.

[63] R. Kuper, "Audio I/O, Today and Tomorrow." [online] http://www.cakewalk.com/DevXchange/audio_i.htm.

[64] Open System Resources, Inc., "KMixer latency." [online] http://www.osr.com/ddk/ddk.htm, Feb. 2001.

[65] D. Phillips, "Low Latency in the Linux Kernel." [online] http://linux.oreillynet.com/pub/a/linux/2000/11/17/low_latency.html, Nov. 2000.

[66] I. Kouvelas and V. Hardman, "Overcoming workstation scheduling problems in a real-time audio tool," *Proc. USENIX Annual Tech. Conf.* (Anaheim, CA), pp. 235–242, Jan. 1997.

[67] H. Liu and P. Mouchtaris, "Voice over IP signaling: H.323 and beyond," *IEEE Communications Magazine*, vol. 38, pp. 142–148, Oct. 2000.

[68] F. Cuervo, N. Greene, A. Rayhan, C. Huitema, B. Rosen, and J. Segers, "Megaco protocol version 1.0." RFC 3015, Internet Engineering Task Force, Nov. 2000.

[69] T. Taylor, "Megaco/H.248: A new standard for media gateway control," *IEEE Communications Magazine*, vol. 38, pp. 124–132, Oct. 2000.

[70] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, "SIP: Session initiation protocol." RFC 2543, Internet Engineering Task Force, Mar. 1999.

[71] U. Black, *Internet Telephony Call Processing Protocols.* Upper Saddle River, NJ: Prentice Hall, 2001.

[72] V. Krishnaswamy and S. R. Thuel, "Multimedia bridging: A structured view," *Proc. Conf. Emerging Technologies and Applications in Communications* (Portland, Oregon), pp. 144–147, May 1996.

[73] D. N. Horn, "Teleconferencing audio bridge." United States Patent 6,178,237, Jan. 2001.

[74] T. W. Anderson, N. R. Tiedemann, and P. W. Vancil, "Conference bridge for packetized speech-signal networks." United States Patent 5,436,896, July 1995.

[75] J. E. O'Niel and G. W. Kajos, "Method and apparatus for centralized multipoint conferencing in a packet network." United States Patent 5,963,547, Oct. 1999.

[76] IP Telephony Magazine, "Media switch technologies." [online] http://www.iptelephony.org/GIP/vendors/mediaservers, May 2001.

[77] V. Hardman, M. A. Sasse, and I. Kouvelas, "Successful multiparty audio communication over the Internet," *Communications of the ACM*, vol. 41, pp. 74–80, May 1998.

[78] J. W. Forgie, "Voice conferencing in packet networks," *Proc. IEEE Int. Conf. on Communications* (Seattle, Washington), pp. 21.3.1–21.3.4, June 1980.

[79] G. Meempat and A. McDonald, "Mobile teleconferencing: Design and performance of handoff management and session multicasting schemes," *Proc. Int. Conf. on Universal Personal Communications* (Florence, Italy), vol. 2, pp. 1351–1357, Oct. 1998.

[80] W. Jiang and H. Schulzrinne, "Analysis of on-off patterns in VoIP and their effect on voice traffic aggregation," *Proc. of the Int. Conf. on Computer Communications and Networks* (Las Vegas, Nevada), pp. 82–87, Oct. 2000.

[81] P. T. Brady, "A statistical analysis of on-off patterns in 16 conversations," *Bell System Technical Journal*, vol. 47, pp. 73–91, Jan. 1968.

[82] P. T. Brady, "A model for generating on-off speech patterns in two-way conversation," *Bell System Technical Journal*, vol. 48, pp. 2445–2471, Sept. 1969.

[83] ITU-T Recommendation P.59, "Artificial Conversational Speech," Mar. 1993.

[84] R. E. Holm, "Transform based scalable audio compression algorithms and low cost audio multi-point conferencing systems." United States Patent 5,570,363, Oct. 1996.

[85] A. H. Li, "An RTP payload format for EVRC speech." Internet Draft, Internet Engineering Task Force—Work in Progress, Apr. 2001.

[86] A. Watson and M. A. Sasse, "The good, the bad, and the muffled: The impact of different degradations on Internet speech," *Proc. ACM Multimedia* (Los Angeles, CA), pp. 269–276, Oct. 2000.

[87] J. D. Mars, "Better Late than Never. A Long Overdue Discussion of Audio Latency Issues." [online] http://www.digitalprosound.com/Htm/Articles/April/Audio_Latency.htm, Apr. 2001.

[88] K. Weiner, "Telephony Audio Latency." [online] http://www.dw.com/twpaper.htm, 2000.