The Use of Synchronised Time in Voice over Internet Protocol (VoIP) Applications

Hugh Melvin, B.E., M.Sc.

Department of Computer Science
Faculty of Science
University College Dublin
Belfield, Dublin 4
Ireland

Submitted in accordance with the requirements for the Degree of Doctor of Philosophy
April 2004

Research Supervisor: Liam Murphy
Head of Department: G.M.P. O’Hare

The candidate confirms that the work submitted is his own and that appropriate credit has been given where reference has been made to the work of others.
Acknowledgements
Abstract

From its origins in the 1960s, the Internet has continuously evolved to address the needs of its growing body of users. The development of the World Wide Web in the early 1990s signalled the start of its mass popularity and consequent congestion issues. Coupled with a growing demand for Internet-based multimedia, this has driven significant research on Quality-of-Service (QoS).

Despite the provision of a number of Internet protocols specifically designed for multimedia data delivery, the underlying best-effort network provides little support for interactive multimedia applications. The delivery of voice over the Internet- or VoIP- is one such application for which QoS is a critical concern. In particular the Mouth-to-Ear (M2E) delay and jitter of VoIP applications can far exceed that encountered in the circuit switched Plain Old Telephone System (POTS). Packet losses due to link problems or late arrival at the receiver can also degrade the QoS of a VoIP application. Therefore, much VoIP research has focused on building QoS measures into both sender and receiver endpoints that compensate for the limitations of the underlying network.

This thesis focuses primarily on receiver endpoint measures to improve VoIP QoS. It proposes and evaluates a hybrid playout strategy that utilises synchronised time, introduced into both sender and receiver endpoints. Synchronised time is implemented using the Network Time Protocol (NTP) and facilitated within the VoIP application using the RTP (Realtime Transport Protocol) Control Protocol, RTCP. This enables receivers to determine precise per-packet delay information which is used to implement an informed fixed playout delay whenever possible. A combination of live testing, simulation and emulation approaches is used to validate this approach and quantify performance gains over diverse networks. An analysis of third party delay studies further confirms its wider applicability.

The move from the circuit-switched POTS to VoIP has also introduced significant endpoint complications. Chief amongst these is the existence of multiple clocks contributing to a VoIP session in contrast to the single POTS clock. Relative differences in the default frequencies of such clocks, or skew can lead to distortion of both the receiver buffer performance and (where relevant) delay measurement. In this thesis, the cause and extent of such skew is assessed along with its effect on both the proposed hybrid playout strategy and other approaches. A high-level solution that quickly determines the extent of skew
between system and audio clocks at each end of a VoIP session is proposed and tested. This solution is based also on the combined use of NTP and RTCP and is easily integrated into the hybrid playout strategy.
# Contents

1 Introduction .......................... 11
   1.1 Internet Multimedia Support .......... 11
   1.2 Receiver Based Strategies ............ 12
   1.3 Research Motivation .................. 13
   1.4 Problem Statement .................... 14
   1.5 Solution Approach .................... 14
      1.5.1 Synchronised Time and Delay Measurement 14
      1.5.2 Hybrid Evaluation ................ 15
      1.5.3 Implementation Issues ............ 16
   1.6 Contributions of this Work .......... 16
   1.7 Thesis Outline ...................... 17

2 Evolution of Internet Multimedia and QoS .... 19
   2.1 Multimedia Content Delivery ......... 20
   2.2 Quality-of-Service ................... 21
      2.2.1 VoIP M2E Delay Components .......... 23
      2.2.2 QoS for VoIP ..................... 27

3 Receiver-based Playout Strategies .......... 32
   3.1 Fixed versus Adaptive Playout ........ 32
   3.2 Background .......................... 33
   3.3 Receiver Playout Strategies .......... 35
   3.4 Adaptive Approaches for Jitter Compensation 36
      3.4.1 Fixed versus Adaptive Playout Strategies 36
      3.4.2 Summary of Adaptive Playout Strategies 37
3.5 Adaptive Approaches for Jitter and Loss Compensation ................. 40
   3.5.1 Summary of Combined FEC-Playout Strategies ................. 41
3.6 Rationale for Proposed Hybrid Playout Strategy ......................... 42
   3.6.1 Hybrid Playout Strategy ................................ 43
   3.6.2 Hybrid Strategy Tuning ................................ 45
   3.6.3 Choice of Adaptive Playout Strategy within the Hybrid and for Comparative Testing .................................... 46
3.7 Summary ......................................................... 47

4 Synchronised Time and Delay Measurement Issues 49
   4.1 Clock Issues .................................................. 50
      4.1.1 Clock Terminology .................................. 51
   4.2 Delay Measurement Issues ................................... 52
      4.2.1 The value of RTT .................................... 55
   4.3 NTP Operation ................................................ 56
      4.3.1 Robustness Issues .................................... 58
   4.4 NTP Performance Tests ....................................... 61
   4.5 Results ......................................................... 63
      4.5.1 NTP Subnet Comparison ............................... 70
   4.6 Summary ......................................................... 70

5 Hybrid Strategy Evaluation: Feasibility and Applicability 74
   5.1 Hybrid Feasibility: Implementation Details ........................ 75
      5.1.1 Synchronised Time .................................... 76
   5.2 Subjective Testing, MOS and the ITU-T E-Model ..................... 77
      5.2.1 Application & Limitations of the E-Model .................. 79
      5.2.2 Application of the E-Model for Hybrid Playout Evaluation 82
   5.3 Hybrid Applicability .......................................... 83
      5.3.1 WAN Testbed Approach .................................. 83
      5.3.2 Simulator Approach .................................... 85
      5.3.3 Trace Delay Data ....................................... 87
      5.3.4 Internet Delay & Loss Modelling .......................... 88
      5.3.5 Emulator Approach ..................................... 93
   5.4 Summary ......................................................... 95
## 6 Hybrid Strategy Evaluation: Applicability Results

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 WAN Testbed Results</td>
<td>97</td>
</tr>
<tr>
<td>6.2 Simulation Results</td>
<td>103</td>
</tr>
<tr>
<td>6.2.1 Trace Data: Summary</td>
<td>103</td>
</tr>
<tr>
<td>6.2.2 Trace Data: Results</td>
<td>108</td>
</tr>
<tr>
<td>6.2.3 General Delay Models: Summary</td>
<td>117</td>
</tr>
<tr>
<td>6.2.4 General Delay Models: Results</td>
<td>122</td>
</tr>
<tr>
<td>6.3 Emulation</td>
<td>126</td>
</tr>
<tr>
<td>6.4 Internet Delay Studies</td>
<td>130</td>
</tr>
<tr>
<td>6.4.1 Importance of Current Data</td>
<td>130</td>
</tr>
<tr>
<td>6.5 Internet Delay Studies: Results</td>
<td>133</td>
</tr>
<tr>
<td>6.5.1 Paxson (1994/1995)</td>
<td>133</td>
</tr>
<tr>
<td>6.5.2 Tobagi et al. (2001)</td>
<td>134</td>
</tr>
<tr>
<td>6.5.3 Roychoudhuri et al. (2001/2002)</td>
<td>135</td>
</tr>
<tr>
<td>6.5.4 Active Measurement Project</td>
<td>136</td>
</tr>
<tr>
<td>6.6 Results Summary</td>
<td>139</td>
</tr>
<tr>
<td>6.6.1 Endpoint Complexities</td>
<td>140</td>
</tr>
<tr>
<td>6.7 Summary</td>
<td>145</td>
</tr>
</tbody>
</table>

## 7 Clock Skew

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1 Introduction to the Problem</td>
<td>147</td>
</tr>
<tr>
<td>7.2 Skew Detection for Accurate Delay Measurement</td>
<td>147</td>
</tr>
<tr>
<td>7.3 Multimedia Clock Skew</td>
<td>150</td>
</tr>
<tr>
<td>7.3.1 Delay Distortion</td>
<td>150</td>
</tr>
<tr>
<td>7.3.2 Buffer Distortion</td>
<td>153</td>
</tr>
<tr>
<td>7.4 Skew Detection &amp; Compensation for Realtime Multimedia</td>
<td>153</td>
</tr>
<tr>
<td>7.4.1 NTP/RTCP-based Skew Detection</td>
<td>154</td>
</tr>
<tr>
<td>7.4.2 Testbed Design</td>
<td>156</td>
</tr>
<tr>
<td>7.4.3 Results</td>
<td>157</td>
</tr>
<tr>
<td>7.4.4 Integrating Skew Detection &amp; Compensation into the Hybrid Playout</td>
<td>163</td>
</tr>
<tr>
<td>7.4.5 Relative Performance Comparison</td>
<td>165</td>
</tr>
<tr>
<td>7.4.6 Software &amp; Hardware Issues</td>
<td>166</td>
</tr>
</tbody>
</table>
8 Conclusions & Future Work

8.1 Main Contributions of This Work .............................................. 171
     8.1.1 Secondary Contributions ............................................. 172
8.2 Future Work ................................................................. 172
     8.2.1 Hybrid Playout Strategy Evaluation ................................. 173
     8.2.2 NTP Performance .................................................... 173
     8.2.3 Endpoint Issues .................................................... 174

A Adaptive Playout Approaches .............................................. 186
     A.1 Adaptive Approaches for Jitter Compensation ....................... 186
         A.1.1 Per-Talkspurt Reactive Approaches ............................ 186
         A.1.2 Per Talkspurt Predictive Approaches .......................... 189
         A.1.3 Per-Packet Based Predictive Approach ........................ 193
     A.2 Adaptive Approaches for Jitter and Loss Compensation .......... 196
         A.2.1 Network-Unaware FEC ........................................... 196
         A.2.2 Network-Aware FEC Strategies .................................. 198

B Hybrid Assessment Methodology ........................................... 201
     B.1 E-Model Description .................................................. 201
         B.1.1 Delay Impairment Id .............................................. 202
         B.1.2 Loss Impairment Ie .............................................. 202
     B.2 Voice Modelling ....................................................... 205

C Detailed Results .......................................................... 210
     C.1 Diurnal Delay Variation ............................................. 210
     C.2 Conditional CDF Results ............................................. 214
     C.3 Sample Trace Results .................................................. 217
     C.4 2 State Markov Delay Model Results ............................... 220
         C.4.1 LKN-Based Models .............................................. 220
         C.4.2 ICIR-Based Models .............................................. 231

D AMP Data Analysis ....................................................... 239
E Skew Detection & Compensation Techniques

E.1 Skew Detection for Delay Measurement ........................................... 254
  E.1.1 Paxson et al. (1996) ................................................................. 254
  E.1.2 Moon et al. (1999) ................................................................. 255
  E.1.3 Zhang et al. (2002) ................................................................. 256
E.2 Skew Detection & Compensation for Multimedia Applications ............ 256
  E.2.1 Akester et al. (2002) ................................................................. 256
  E.2.2 Hodson et al. (2001) ................................................................. 257

F Linux Divert Sockets versus NIST Net .................................................. 259

G Soundcard Driver Configuration Issues ................................................. 262

H Publications and Patents Arising from this Thesis ................................ 264
  H.1 International Refereed Conferences/Journals ................................. 264
  H.2 National Refereed Conferences ..................................................... 265
  H.3 Patent .......................................................................................... 265
List of Figures

2.1 ITU-T G.1010 ................................................. 20
2.2 Quality of Service Overview .................................. 22
2.3 Mouth to Ear Delay Components (M2E) ......................... 24
2.4 RTP Header .................................................. 28
2.5 RTCP SR Header .............................................. 28

4.1 Typical PC System Clock Accuracy .......................... 51
4.2 NTP Operational Model ...................................... 57
4.3 NTP Hierarchy ................................................ 59
4.4 NTP-Hybrid Testbed ........................................... 62
4.5 NTP Testbed .................................................. 63
4.6 Performance of the NULG NTP client ......................... 64
4.7 Performance of the DCU NTP client .......................... 65
4.8 Performance of Stratum 2 Linux Client ....................... 66
4.9 Performance of Stratum 2 Windows 2000 Client ............. 67
4.10 Diurnal Variation of Stratum 2 Linux Client ............... 68
4.11 Diurnal Variation of Stratum 2 Windows 2000 Client .... 69
4.12 Offsets Measured by ntp-sop.inria.fr to its Servers .......... 71
4.13 Offsets Measured by NULG Stratum 1 to its Servers ....... 72

5.1 Simplified yet Conservative E-Model Analysis ............. 84
5.2 Simulator ...................................................... 86
5.3 2-state Markov Delay Model ................................ 93
5.4 Testbed Emulator ............................................ 94

6.1 Typical test result ........................................... 98
6.2 Late Loss Reduction due to hybrid algorithm .................. 99
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.3 Increased delay due to hybrid algorithm</td>
<td>100</td>
</tr>
<tr>
<td>6.4 R-factor improvement due to the hybrid algorithm</td>
<td>101</td>
</tr>
<tr>
<td>6.5 Delay to NUI G NTP server seen from DCU</td>
<td>102</td>
</tr>
<tr>
<td>6.6 Mean Delay to Remote Servers (60 sec tests)</td>
<td>104</td>
</tr>
<tr>
<td>6.7 Packet Loss Rate to Remote Servers (60 sec tests)</td>
<td>105</td>
</tr>
<tr>
<td>6.8 Diurnal Variation in Delay to UCD</td>
<td>106</td>
</tr>
<tr>
<td>6.9 Conditional/Unconditional CDF: UCD</td>
<td>107</td>
</tr>
<tr>
<td>6.10 Sample Trace Performance: UCD</td>
<td>109</td>
</tr>
<tr>
<td>6.11 Hybrid Algorithm Performance: Self Tuning</td>
<td>110</td>
</tr>
<tr>
<td>6.12 Relative Algorithm Performance: UCD</td>
<td>111</td>
</tr>
<tr>
<td>6.13 Relative Algorithm Performance: UCL</td>
<td>112</td>
</tr>
<tr>
<td>6.14 Relative Algorithm Performance: LKN</td>
<td>113</td>
</tr>
<tr>
<td>6.15 Relative Algorithm Performance: ICIR</td>
<td>114</td>
</tr>
<tr>
<td>6.16 LKN: Bursty Delay Model</td>
<td>127</td>
</tr>
<tr>
<td>6.17 ICIR: Bursty Delay Model</td>
<td>128</td>
</tr>
<tr>
<td>6.18 Effect of 30 msec Fragment Size on Performance</td>
<td>143</td>
</tr>
<tr>
<td>7.1 Relationship between Audio and System Clocks</td>
<td>148</td>
</tr>
<tr>
<td>7.2 Effect of skew on Alg. A</td>
<td>152</td>
</tr>
<tr>
<td>7.3 Skew Detection Testbed</td>
<td>156</td>
</tr>
<tr>
<td>7.4 bibio audio seen by bibio2 sys clock</td>
<td>158</td>
</tr>
<tr>
<td>7.5 RTCP SR packets received by bibio2 (from bibio)</td>
<td>159</td>
</tr>
<tr>
<td>7.6 bibio2 audio seen by bibio sys clock</td>
<td>160</td>
</tr>
<tr>
<td>7.7 RTCP SR packets received by bibio (from bibio2)</td>
<td>161</td>
</tr>
<tr>
<td>7.8 Clock Skew Inter-relationships</td>
<td>162</td>
</tr>
<tr>
<td>7.9 Clock Skew Inter-relationships: bibio2 not synchronised</td>
<td>162</td>
</tr>
<tr>
<td>B.1 Delay Impairment</td>
<td>203</td>
</tr>
<tr>
<td>B.2 Loss Impairment</td>
<td>204</td>
</tr>
<tr>
<td>B.3 Mapping between MOS and R factor</td>
<td>206</td>
</tr>
<tr>
<td>B.4 6 State Model</td>
<td>207</td>
</tr>
<tr>
<td>B.5 2 State Model</td>
<td>208</td>
</tr>
<tr>
<td>C.1 Diurnal Variation in Delay to UCL</td>
<td>211</td>
</tr>
</tbody>
</table>
C.2 Diurnal Variation in Delay to LKN ........................................... 212
C.3 Diurnal Variation in Delay to ICIR (20 sec tests) ......................... 213
C.4 Conditional/Unconditional CDF: UCL ..................................... 214
C.5 Conditional/Unconditional CDF: LKN ..................................... 215
C.6 Conditional/Unconditional CDF: ICIR ..................................... 216
C.7 Sample Trace Performance: UCL .......................................... 217
C.8 Sample Trace Performance: LKN .......................................... 218
C.9 Sample Trace Performance: ICIR .......................................... 219
C.10 LKN: Delay Model A ......................................................... 220
C.11 LKN: Delay Model B ......................................................... 221
C.12 LKN: Delay Model C ......................................................... 222
C.13 LKN: Delay Model D ......................................................... 223
C.14 LKN: Delay Model E ......................................................... 224
C.15 LKN: Delay Model F ......................................................... 225
C.16 LKN: Delay Model I ......................................................... 226
C.17 LKN: Delay Model J ......................................................... 227
C.18 LKN: Delay Model K ......................................................... 228
C.19 LKN: Delay Model L ......................................................... 229
C.20 LKN: Delay Model M ......................................................... 230
C.21 ICIR: Delay Model A .......................................................... 231
C.22 ICIR: Delay Model B .......................................................... 232
C.23 ICIR: Delay Model C .......................................................... 233
C.24 ICIR: Delay Model D .......................................................... 234
C.25 ICIR: Delay Model E .......................................................... 235
C.26 ICIR: Delay Model F .......................................................... 236
C.27 ICIR: Delay Model G .......................................................... 237
C.28 ICIR: Delay Model H .......................................................... 238
D.1 amp-hean to amp-hutf ....................................................... 240
D.2 amp-hean to amp-eltm ....................................................... 241
D.3 amp-hean to amp-thor ....................................................... 242
D.4 amp-hean to amp-apantyo .................................................. 243
D.5 amp-thor to amp-columbia .................................................. 244
D.6 amp-thor to amp-fsu ....................................................... 245
D.7  amp-thor to amp-stanford ............................................. 246
D.8  amp-thor to amp-ucb .................................................. 247
D.9  amp-thor to amp-utexas ............................................... 248
D.10 amp-thor to amp-uwashington ...................................... 249
D.11 amp-uic to amp-usf .................................................. 250
D.12 amp-columbia to amp-alaska ....................................... 251
D.13 amp-harv to amp-ucb ................................................ 252
D.14 amp-alaska to amp-usf .............................................. 253
# List of Tables

3.1 Adaptive Algorithms for Jitter Compensation ........................................... 36  
3.2 Adaptive Algorithms for Jitter and Loss Compensation ............................. 41  
3.3 Comparative Testing: Choice of Algorithms ............................................. 47  
4.1 Serialisation Delay .................................................................................. 55  
5.1 NTP Server: Status Symbols ..................................................................... 76  
5.2 R values for Speech Quality Categories .................................................. 78  
5.3 R-MOS User Satisfaction Ratings ............................................................ 78  
6.1 Test Numbers and Corresponding Period .................................................. 103  
6.2 Average Results: Hybrid vs Alg.A ......................................................... 115  
6.3 Average Results: Hybrid vs Alg.B .......................................................... 115  
6.4 2 State Markov Parameters from Late Loss Patterns ................................ 116  
6.5 LKN-based Delay Models ....................................................................... 118  
6.6 ICIR-based Delay Models ........................................................................ 120  
6.7 Delay Model Applied to Actual Traces ..................................................... 121  
6.8 LKN-based Delay Model Results ............................................................. 122  
6.9 ICIR-based Delay Model Results ............................................................. 124  
6.10 Paths from Ireland to Mainland Europe & Toyko ................................... 132  
6.11 Transatlantic Paths ............................................................................... 132  
6.12 US Internal Paths ................................................................................... 132  
6.13 Path Classification According to Fixed and Variable Delay ...................... 134  
6.14 Delay and Loss Data for European Paths .............................................. 136  
6.15 Delay and Loss Data for Ireland to Toyko Path ...................................... 136  
6.16 Delay and Loss Data for Transatlantic Paths ......................................... 137  
6.17 Delay and Loss Data for Internal US Paths ............................................ 137
Chapter 1

Introduction

The Internet has continuously evolved from its ARPANET days though the rate at which it has done so has accelerated since the early 1990s. With the development and growing popularity of the World Wide Web came increased demands on its functionality as well as increased interest from commercial concerns. Internet-based multimedia applications started to appear in the early to mid 1990s and, coupled with the exponential demand for Internet connectivity, the weakness of the Internet’s best-effort service level was exposed and issues regarding congestion and more generally Quality-of-Service (QoS) began to arise.

1.1 Internet Multimedia Support

The development of multimedia support protocols such as the Realtime Transport Protocol (RTP) and RTP Control Protocol (RTCP) [1] enabled the effective reconstruction and synchronisation of multimedia streams at receivers. Furthermore, the Real Time Streaming Protocol (RTSP) facilitated user-controlled streaming [2]. However, the growing demand for interactive multimedia services with stringent timing requirements has particularly exposed the QoS limitations of the underlying network [3]. Voice over IP (VoIP) is one such application and is the focus of this thesis.

VoIP involves the digitalisation and packetisation of voice into Internet Protocol packets for transmission across IP networks. As such, it represents a significant departure from the traditional circuit-switched Plain Old Telephone System (POTS). The key attraction of VoIP is its use of the increasingly ubiquitous IP protocol and the potential for integrating advanced voice and data services. Although the emergence of VoIP in the late 1990s was
much heralded, its penetration so far has been limited. This is largely due to QoS problems such as non-determinism and packet loss in best-effort IP networks such as the Internet and thus its deployment has mostly been within private IP networks and Local Area Networks (LAN) where QoS within the intermediate network can be managed [4].

The Telecommunications division of the International Telecommunications Union (ITU-T) G.114 recommendation specifies that end-to-end delays should not exceed 150 msec [5] and the POTS generally presents delays much less than this (with the exception of satellite links). In a VoIP session, the end-to-end (or mouth-to-ear M2E) delay is composed of sender and receiver endpoint delays along with network delays. QoS strategies can thus be classified by whether they involve measures taken within sender or receiver endpoints, or within the intermediate network. Network-based measures seek to build on the default best-effort Internet service, by distinguishing between traffic flows and implementing appropriate QoS policies. Although such measures have been successfully implemented within islands of the Internet, the default best-effort service will remain dominant for the foreseeable future. As such both sender and receiver-based strategies that compensate for best-effort limitations will continue to play a critical role.

1.2 Receiver Based Strategies

In order to compensate for network jitter, receiver playout strategies delay delivery of incoming packets to the codec for playout in an attempt to smooth out jitter and thus reproduce the constant rate at which packets are generated. The extent to which packets are delayed represents a trade-off between additional delay and late packet loss i.e. by increasing the playout delay, the receiver adds to the total M2E delay but reduces the likelihood of packets arriving too late for playout. Fixed playout strategies that implement a fixed playout point for the duration of a VoIP session though simple to implement are unsuited to best-effort networks. Adaptive playout strategies attempt to optimise this trade-off by continuously monitoring and adapting to network conditions. Human speech can be seen as alternating between talkspurts and silence periods. Adaptive strategies take advantage of this and make adjustments either to inter-talkspurt silence periods and/or to actual packets within talkspurts, thus introducing some degree of temporal distortion of the original voice sample. They generally have no knowledge of actual delays, adapting instead to trends in delay estimates. Defining the rate at which they adapt to such trends is a crit-
ical design factor that presents a significant challenge. They must strike a balance between over-reaction, resulting in widely varying playout delays and under-reaction, resulting in increased late loss. Such tuning of adaptive approaches is often based on experimentation to capture network characteristics, that are however subject to change.

In addition to network jitter, strategies are also required to compensate for packet loss in best-effort networks. These include Forward Error Correction (FEC) and Packet Loss Concealment (PLC). The former requires the participation of the sender whereas the latter only involves the receiver. Recent research has examined the degree to which strategies for dealing with network jitter and loss can be coupled together. As such there is little point in implementing FEC within a sender if the receiver playout strategy operates independently without regard for the additional delay that FEC requires to reconstruct lost packets. On the other hand, a playout strategy that adds a fixed delay for FEC whether or not it is required is too conservative.

1.3 Research Motivation

Although conventional (delay-unaware) adaptive playout approaches are essential for coping with best-effort networks, this thesis suggests that they are often implemented unnecessarily. Because they operate without knowledge of actual delays, they may introduce unnecessary temporal distortion and late packet loss in situations where despite network jitter, overall M2E delays are well within the G.114 M2E limit. By implementing synchronised time within both sender and receiver endpoints, each receiver can determine precise per-packet delays and can make more informed decisions regarding playout strategy, implementing an informed and conservative fixed playout strategy whenever possible. By minimising the use of adaptive approaches, both inherent temporal distortion and late packet loss can be reduced, at the expense of slightly higher delays. This proposed hybrid strategy thus combines the benefits of both fixed and adaptive approaches. The ITU-T E-Model [6] and other studies have shown that users are more tolerant of increased delays than of increased packet loss which underpins the rationale for the proposed hybrid approach.

Although not the primary focus of this thesis, the availability of precise per-packet delays within the proposed hybrid approach greatly facilitates the coupling of strategies for dealing with network loss and jitter. With such information, both senders and receivers
can make more informed decisions regarding the use of FEC.

1.4 Problem Statement

In the context of the research motivation, outlined above, the following key questions emerge:

1. Can the level of synchronised time required for the operation of the proposed hybrid playout strategy be achieved and deployed effectively over Wide Area Networks (WAN)?

2. Can the proposed hybrid playout strategy be implemented within VoIP applications and if so, how widely can it be applied and what benefits will it bring?

1.5 Solution Approach

1.5.1 Synchronised Time and Delay Measurement

Implementing synchronised time to the degree required for the hybrid playout strategy is a non-trivial task. Problems relating to computer clock synchronisation include both underlying oscillator and higher level Operating System (OS) issues. The Network Time Protocol (NTP) [7] is the most commonly used mechanism for implementing distributed synchronised time though reported performance levels have been mixed. This thesis shows that effective NTP performance is however achievable with careful NTP subnet design and implementation. Two testbeds are developed to show that robustness issues can be addressed which will deliver the required performance levels. The effect on performance of local reference sources, diurnal traffic patterns and Operating System (OS) platforms are briefly examined.

Precise delay measurement issues are also addressed in this thesis. These include measurement uncertainties within endpoint hardware and software as well as broader issues relating to the usefulness of round trip time (RTT) delays for gauging one-way delays.
1.5.2 Hybrid Evaluation

In evaluating the hybrid playout strategy, two core objectives are identified and addressed. These are firstly to test the feasibility of the hybrid playout approach and secondly to assess its applicability within the Internet.

Regarding feasibility, a number of concerns arise. The first relates to the underlying requirement within the hybrid playout strategy for synchronised time, an issue that has been addressed above in section 1.5.1. A second concern relates to the technical feasibility of implementing the hybrid approach within a real VoIP application. A Wide Area Network (WAN) testbed is thus developed within the Irish academic and research network (HEAnet) where voice streams are delivered between hosts located at the National University of Ireland, Galway (NUI,G) and Dublin City University (DCU). The host clocks are synchronised using a locally strengthened NTP subnet and a VoIP application is developed which implements the hybrid strategy as well as other adaptive playout approaches. NTP along with the RTCP protocol are used to implement the mapping of RTP timestamps to system clock time thus facilitating delay measurement. This approach is used to test the feasibility of the hybrid, both in relation to the underlying requirement for synchronised time and its implementation within a real VoIP application. Regarding the second core objective, it also assesses its applicability in a limited network by comparing its performance with baseline adaptive approaches.

One of the great difficulties in evaluating the performance of any Internet application is the degree of network heterogeneity that the Internet presents. Significant research has been carried out in attempts to capture and model Internet characteristics and yet such research is always limited by the fact that not only is the Internet continuously growing but that the range and type of applications that utilise the Internet are also changing [8]. In this context, a second approach to applicability testing is simulator-driven whereby simulated voice streams are delivered across simulated networks and the relative performance of various playout strategies evaluated via simulation. Both trace-driven delay data and general delay models are used for network simulation. The trace data is gathered from tests to various locations in the UK, mainland Europe and the US.

A third approach involves the development of a LAN based emulator where packet delay data, derived either through trace testing or general delay models is simulated and applied to actual voice streams and a real VoIP application. This emulator approach is essentially similar to that of the WAN testbed except that more diverse network conditions
are simulated in a controlled andrepeatable LAN environment. This approach is primarily
used to confirm a subset of the simulator-derived results.

For all the above approaches, the ITU-T E-Model (developed originally by the European Telecommunications Standards Institute ETSI) is used to quantify relative performance, and shows that the hybrid playout strategy can result in significant gains. The degree to which the hybrid strategy can be deployed successfully is largely dependent on M2E Internet delay characteristics. As such the final approach taken is to analyse third party studies on Internet delays to assess its wider applicability. Results from this analysis indicate that although jitter remains a significant issue, delay characteristics are such that the hybrid playout strategy can deliver improved QoS across a wide range of Internet paths.

1.5.3 Implementation Issues

Although the primary advantage of the simulation approach described above is that it facilitates rapid and extensive testing, a further benefit is that it removes many endpoint complexities that testing in a real implementation environment present. Recent research such as [9] has shown that VoIP endpoints can often introduce significant delay and jitter uncertainty that dwarf the network component of M2E delay.

In particular, the move from POTS-based telephony to VoIP has resulted in the single POTS clock being replaced by a multitude of clocks within sender and receiver endpoints. The lack of synchronism between such clocks distorts delay measurements which (if not addressed) can seriously undermine the proposed hybrid playout strategy but which can also affect (though to a lesser degree) conventional adaptive approaches. Such skew can also distort the performance of receiver jitter buffers which is a matter of concern for any VoIP application.

In this thesis, a high level skew detection approach is proposed and tested. It integrates well with the hybrid playout strategy, utilising both the NTP and the RTCP protocols to quickly determine the extent of skew once a session is established. Alternative approaches aimed at resolving skew on the fly are reviewed [10] [11] and comparisons made.

1.6 Contributions of this Work

- The Network Time Protocol can deliver single figure msec level synchronisation across Wide Area Networks thus satisfying the requirements of the hybrid playout strategy.
To achieve these levels of synchronisation however, careful attention to NTP subnet design issues such as redundancy, path diversity and local reference sources is required. In particular, the proximity of local reference sources is essential. Both diurnal traffic patterns and OS platform all affect NTP performance, though in the context of the hybrid playout strategy requirements, not to a significant degree.

- The proposed hybrid playout strategy combines the useful characteristics of fixed and adaptive playout approaches, utilising synchronised time to implement an informed fixed delay playout whenever possible. A combination of live testing, simulation, emulation and an analysis of 3rd party Internet delay studies confirm the applicability of the hybrid strategy.

- Hardware and software issues within endpoints can introduce significant delay uncertainty within real VoIP applications. In particular, clock skew can affect both buffer performance and (where relevant) delay measurement. The impact of skew on conventional adaptive and the hybrid playout strategies is examined and a high level on the fly solution to skew detection that integrates seamlessly within the hybrid approach is proposed and tested.

- Although the E-Model has in very recent years been developed by researchers for VoIP, much of this work has been done without due regard for its limitations. In this thesis, the key limitations of the E-Model are identified. Through collaborative work with ETSI, the particular implementation of the E-Model used for assessing the hybrid playout strategy was approved.

1.7 Thesis Outline

Chapter 2 reviews the evolution of Internet multimedia applications and the emergence of VoIP and QoS issues. It examines the typical M2E delay of a VoIP session and summarises the various QoS initiatives aimed at improving voice quality.

Chapter 3 reviews the extensive literature on receiver-based playout strategies. It also describes a number of approaches that attempt to couple together strategies for dealing with both jitter and packet loss in order to optimise playout quality. From this review, the rationale for the hybrid playout strategy is developed and its operation described in some detail.
Chapter 4 addresses the issues relating to the implementation of synchronised time and precise delay measurement. It examines the underlying causes of clock errors and outlines the delay measurement uncertainties that can arise. It describes the Network Time Protocol focusing on requirements for effective performance. Testbeds are developed to assess NTP performance in a LAN and WAN environment and to examine the impact of issues such as proximity of local reference sources, OS dependency and diurnal traffic patterns. Results presented confirm effective NTP performance, in the context of the hybrid playout strategy requirements.

Chapter 5 examines feasibility issues regarding the implementation of the hybrid playout strategy within a real VoIP application and outlines the approaches taken to assess its applicability in the Internet. The ITU-T E-Model with which relative performance of various playout strategies is evaluated is described. Its limitations along with recent work on extending its use for VoIP are reviewed and the approach taken to its implementation in this thesis is outlined. To facilitate the simulator and emulator approaches, a review of both voice and network modelling is undertaken.

Chapter 6 presents results confirming the applicability of the hybrid playout strategy. A range of third party delay studies are identified and analysed. The chapter concludes by describing a range of implementation issues, including that of clock skew, that arose during the WAN testbed and emulator approaches to testing.

Chapter 7 focuses on the single issue of clock skew. From a multimedia perspective, its impact on both delay measurement and buffer performance are examined. A high level solution that integrates easily into the hybrid playout strategy is proposed and tested and compared with alternative approaches.

Chapter 8 concludes the thesis. It summarises the main contributions made to the research area and outlines a number of avenues for further research.
Chapter 2

Evolution of Internet Multimedia and QoS

In this chapter, a summary of the main developments that have taken place within the Internet community in recent years to facilitate multimedia content delivery is presented and the problems related to such delivery identified. The chapter commences by examining the varied requirements of differing multimedia applications and in so doing introduces QoS. The concept of QoS is reviewed in some detail from both a technical and user perspective in an attempt to add clarity to a term that in recent years has been somewhat overused. A detailed dissection of Mouth-to-Ear (M2E) delay for a typical VoIP session is then presented. In section 2.2.2, the RTP and RTCP protocols are reviewed, focusing on their features most relevant to this work. The chapter concludes with a broad view of the various QoS initiatives that have emerged in recent years. Such initiatives may reside in the sender, in the underlying network and in the receiver. It is important to note that as IP becomes increasingly pervasive, private IP networks where QoS can be managed are becoming more commonplace as opposed to the single best-effort model of the public Internet. There is thus often a distinction made between Voice on the Internet (VoNET) and Voice over IP (VoIP). No such distinction is made in this thesis and the use of the term VoIP refers to voice carried over any IP network.
2.1 Multimedia Content Delivery

The development of the WWW has heralded major changes in Internet usage. Unmanaged packet-based networks such as the Internet are ideally suited to deliver time-insensitive data such as Email, FTP and static web traffic. The growing requirement for delivery of Web-based multimedia data has however focused much attention on the limitations of the Internet core protocols. The public Internet’s best-effort service offers no end-to-end delay bounds, making it unsuited to the delivery of time-sensitive data. An important and further distinction needs to be made between interactive and streaming-type multimedia applications. ITU-T G.1010 recommendation reviews the spectrum of user applications in terms of their sensitivity to both packet loss and delay [3]. Fig. 2.1 summarises this work.

As a rule-of-thumb, for E-commerce type applications and E-mail, users will tolerate service-time delays of up to two seconds. A similar level of acceptable delay applies to messaging services though a certain degree of packet loss can also be tolerated. For streaming
applications, delays of up to 10 seconds (for buffering before playout) with a degree of loss can be tolerated; for FTP transfers, a similar delay is acceptable though obviously coupled with zero loss. For interactive applications such as remote login sessions, response times in the region of 100 msec are required whereas for VoIP, M2E delays should not exceed 150 msec though as with messaging and streaming, a degree of packet loss is acceptable.

The ITU-T G.114 recommendation specifies an acceptable one-way delay of 150 msec for telephony [5]. Such a delay budget is generally well within the reach of the conventional POTS (unless satellite links are involved) whereas the Internet presents varying delays where propagation time is often dwarfed by congestion delay. More detailed analysis by [12] suggests that the precise VoIP requirements are context dependent, in that business-type voice sessions demand stricter M2E budgets than social calls. From a loss perspective Fig. 2.1 illustrates that for multimedia applications, users will tolerate a certain degree of loss without significant loss in quality, particularly so if compensating mechanisms are employed within receiver endpoints. Although the term QoS existed in a POTS context and predates the popularisation of the Internet in the 1990s, its over-general application in the latter context has caused some confusion. In [13] the differing approaches of organisations such as the Internet Engineering Task Force (IETF), the European Telecommunications Standards Institute (ETSI) and the International Telecommunications Union (ITU) to the term is examined. In the next section this work is reviewed in order to bring some clarity to the term as applied in this thesis.

2.2 Quality-of-Service

In [13], a general model for QoS, proposed by [14] and shown in Fig. 2.2 is used to reconcile the various perspectives. This model introduces three notions of QoS, Intrinsic, Perceived and Assessed.

Intrinsic QoS is defined by technical performance issues such as delay, jitter, loss and bandwidth. The definition and application of these terms, particularly delay and jitter has also led to some confusion in recent years; RFCs 2679 [15] and 2680 [16] along with the more recent RFC 3393 [17] and 3357 [18] outline the IETF’s perspective on these issues which is adopted in this thesis. Regarding loss for multimedia applications, a distinction needs to be made between packets lost in the network (known as link loss in this thesis) and late packet loss which results from packets arriving at the receiver too late for playout.
### General model

<table>
<thead>
<tr>
<th>Assessed QoS</th>
</tr>
</thead>
<tbody>
<tr>
<td>QoS perceived by the customer</td>
</tr>
<tr>
<td>QoS achieved by the provider</td>
</tr>
</tbody>
</table>

### ITU/ETSI Approach

| QoS requirements of the customer |
| QoS offered by the provider |

### IETF Approach

| Quality of Service |

### Intrinsic QoS

| Network Performance (NP) |

---

**Figure 2.2: Quality of Service Overview**
and which are thus dropped. User perception therefore does not in any way influence the Intrinsic QoS rating.

Perceived QoS incorporates the user experience in that the same intrinsic QoS may be perceived differently by two users that have differing prior experience or expectations. Finally, Assessed QoS incorporates issues such as cost-effectiveness, customer support and overall user experience of a service provider.

The ITU and ETSI approaches to QoS-related terminology are similar and define QoS as *the collective effect of service performance which determine the degree of satisfaction of a user of a service*. As such, this definition reflects the notion of Perceived rather than Intrinsic QoS as evident from Fig. 2.2. Furthermore they define the notion of Network Performance (NP) which maps well to Intrinsic QoS. The IETF notion of QoS is different in that it focuses on Intrinsic QoS and does not deal with Perceived QoS. It defines QoS as *a set of service requirements to be met by the network while transporting a flow* and is quite similar to the ITU/ETSI notion of NP.

In some ways the Mean Opinion Score test [19], developed by the ITU-T maps well to the notion of Perceived QoS in that it is based on subjective testing. For this reason, the selection of candidates for MOS tests requires careful consideration to ensure the validity and repeatability of results. On the other hand the ITU-T E-Model [6], as applied in this thesis returns a QoS measure that maps well to Intrinsic QoS in that it returns a quality rating based on technical parameters such as delay and loss and thus will always return the same value for a given set of technical parameters. This distinction and indeed the E-Model are dealt with in greater detail in section 5.2. In this thesis, the IETF notion of QoS is adopted thus reflecting a technical measure similar to Intrinsic QoS.

### 2.2.1 VoIP M2E Delay Components

Before embarking on an in-depth analysis of QoS within a VoIP environment, it is useful to consider the various components that comprise the total M2E delay as experienced by typical VoIP applications. Delays can occur in the sender and receiver endpoints as well as in the intermediate network as outlined in Fig. 2.3.

- **Sender Endpoint**: Sender endpoint delay may include some or all of the following to lesser or greater degrees depending on whether the endpoint is PC-based or a dedicated IP-phone.
Mouth-to-Ear (M2E) Delay Components

- Packetisation Delay: Depends on the codec and the configured frame size. A large frame size will not only lead to higher packetisation delay but also to greater distortion if such frames are lost. For example, with gsm 20 msec packets, a frame size of four packets per frame results in a 80 msec packetisation delay. Considering that the average phoneme size (smallest unit of speech) is approximately 80 msec [20], frame loss of this or greater size can lead to significant loss of intelligibility [21]. Furthermore, large frames with multiple packets can consume a significant percentage of the G.114 150 msec budget, leaving little scope for further delays. In this context, the ITU-T G.108 recommendation outlines that packet size should not exceed 64 msec [22]. On a related note, [23] outlines that Forward Error Correction (FEC) results in improved MOS scores as frame size is increased though the corresponding negative effect of increased delay is also considered using the E-Model. Conversely the ability of Packet Loss Concealment (PLC) techniques to deal with loss is reduced with higher packet size; in any event FEC techniques are superior to PLC as [23] shows.
- Other Codec Delays: Some hybrid (combination of waveform and source codecs) codecs require look-ahead delay as part of the codec compression operation. For example, G.723 requires 7.5 msec whereas G.729 requires 5 msec leading to minimum frame sizes of 37.5 and 15 msec respectively [24]. Hybrid codecs also involve a certain amount of algorithmic processing which introduces some encoding delay.

- Sound Card Driver Delays: A mismatch between sound card driver and user-configured frame size can also lead to erratic timing behaviour or jitter. For example, the Linux OSS driver implements a fragment size that is a power-of-two bytes [25]. By selecting a frame size that matches this requirement eg. 32 msec g711 , the sound card codec will deliver packets smoothly to the sending application. Otherwise a significant step-like jitter is introduced resulting in additional delays. This topic is addressed in more detail in section 6.6.1 and Appendix G.

- Operating System (OS) Delays: PC-based endpoints have to compete for resources along with all other current processes and as such scheduling delays can arise. [26] devise an approach to incorporate a buffer within sound cards to avoid what is termed CPU starvation. Due to their dedicated OS, IP phones should generally provide better service. [9] compares some current IP phones to PC-based softphones and confirms this to a certain degree. More detail on this is provided in section 6.6.1.

- Transmission Delay: This is dependent on the interface speed and thus usually not significant. Where network delays are small however eg. single figure msec, transmission delay can be a significant component. Both Paxson in [27] and RFC 3393 [17] outline that for active delay measurements, it is important that packet sizes be similar to ensure similar transmission delays.

- Network: Network delay is comprised of a fixed and variable component. The fixed component is comprised of the propagation delay which is set by the physical medium and transmission delays which are encountered at every node and defined by interface speed. The variable component comes from queuing delays introduced at intermediate nodes such as routers and switches. Due to the lack of admission control or traffic differentiation, best-effort service can result in queuing delays that are signifi-
cantly higher than the fixed delays and that suffer from high variance, or jitter. The measurement and characterisation of network delay and jitter in the Internet is a primary objective of this thesis.

- **Receiver Endpoint**: Receiver endpoints introduce somewhat similar OS and software delays to that of the sender endpoints but in addition incorporate anti-jitter buffering delays. Jitter buffers store incoming packets for a period of time to ensure that the jitter in packet delays caused principally by varying queuing delays is absorbed, resulting in smooth playout. The extent to which packets are delayed in the jitter buffer is therefore determined by the extent of jitter in the network (rather than the fixed component of network delay). The design of optimum receiver playout strategies is the main focus of this thesis. In the next chapter, the significant body of research work in this area is reviewed and the contribution that this thesis makes to the research area outlined. Finally, as noted above under packetisation delay, strategies to compensate for packet loss such as FEC and to a lesser degree PLC add additional delay in the receiver. These are discussed in more detail in section 3.5.

As outlined above, [9] compared the performance of a number of VoIP softphones and IP phones using a black box approach. Results indicate that endpoints can often contribute very significant and varying delays that in themselves can reach or surpass the G.114 limit. This work also distinguishes between the contribution of sender and receiver endpoints and proposes some explanations for the results. Although chapter 6 includes a brief discussion on various endpoint issues, this thesis is primarily concerned with the design of receiver-based playout strategies to deal effectively with network delay and jitter. However in chapter 7, the issue of endpoint clock skew is addressed in detail, examining its effect on delay measurement and buffer performance. As detailed in chapter 7, clock skew can under certain conditions lead to large M2E delay increases.

In RFC 2681 [28], the concept of wire versus host time is introduced in the context of performing end-to-end delay measurement and accounting for measurement uncertainties within both sender and receiver endpoints. In a VoIP context the M2E delay described above maps well to host time (including uncertainties such as OS and application delay) whereas network delays described above map to wire time. However, M2E delay includes other delay components such as packetisation and jitter buffering that are explicitly built into a VoIP application. The issue of accurate delay measurement is dealt with in detail
in section 4.2.

### 2.2.2 QoS for VoIP

This section commences with a summary of the RTP and RTCP Multimedia support protocols developed by the IETF in recent years. The chapter concludes with a broad introduction to measures for improving VoIP QoS.

**Internet Multimedia Support Protocols**

In response to the need to deliver multimedia data across a non-deterministic network, the Internet protocols Realtime Transport Protocol (RTP) and its companion control protocol RTP Control Protocol (RTCP) were developed. Technical details are provided in RFC 1889 [1] whereas a more general discussion of their support for Internet multimedia is given in [29] and [30]. Fig. 2.4 outlines the RTP header. Of particular interest are the timestamp and sequence number fields that enable the accurate reconstruction of media streams at the destination host along with packet loss detection. The timestamp does not relate in any way to system time but rather is initialised randomly and increases at a rate depending on the sampling rate of the media stream. For audio, the rate is generally 8000 Hz, similar to the POTS rate. RTCP serves a number of functions via a number of different packet types (though it was noted during testing that some current products do not implement many RTCP features or do so incorrectly). Of particular interest is the RTCP SR (Sender Report) packets whose header is shown in Fig. 2.5. SR packets are generated periodically by receivers who also are senders eg. in a VoIP session, both end hosts generate SR packets. The NTP and RTP timestamp fields relate to the system clock time at which the RTCP SR packet was generated and the corresponding RTP timestamp respectively. One of the principal uses of RTCP SR packets is for lip-synchronisation whereby separate media streams such as audio and video can be related temporally. This is possible once the first RTCP SR packets are received for each media stream by the receiver as it can then relate the RTP timestamps (generated independently for each media stream and usually increasing at different rates) to a common NTP-formatted system timestamp. As described in detail in chapter 7, successive SR packets relating to a specific media stream can be used over time to determine the skew between a host’s system clock and audio card clock.

The other main benefit of RTCP is via its report blocks generated by both the SR and
RTP Header Format

<table>
<thead>
<tr>
<th>V</th>
<th>P</th>
<th>X</th>
<th>CC</th>
<th>M</th>
<th>PT</th>
<th>Sequence Number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RTP Timestamp</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Synchronisation Source Identifier (SSRC)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Contributing Source Identifiers (CSRC)..if present</td>
</tr>
</tbody>
</table>

Figure 2.4: RTP Header

RTCP SR (Sender Report) Header Format

Header (8 octets)

- NTP Timestamp, most significant word
- NTP Timestamp, least significant word
- RTP Timestamp
- Sender’s packet count
- Sender’s octet count

Report Block 1

Figure 2.5: RTCP SR Header
RR (Receiver Report) packets. They relate information back to each sender concerning the network conditions such as jitter and packets lost as determined by each receiver. They also enable each sender host to derive the round-trip-time (RTT) delay to each of its remote hosts. A new XR protocol is currently under development by the IETF [31]. This will provide more detailed information back to each sender such as the degree of burstiness of link loss, delay information, jitter buffer details including late loss rate, and estimates of quality as seen at each receiving host, quantified through use of the E-Model R factor and MOS values [32]. To minimise delay and protocol overhead at the expense of unrecovered packet loss, both RTP and RTCP typically utilise the unreliable transport layer User Datagram Protocol (UDP).

In summary, both RTP and RTCP provide significant support for multimedia data delivery. If however, data delivery is via a best-effort network, the underlying non-determinism and link loss can seriously degrade the quality of interactive applications such as VoIP. For this reason, streaming or messaging applications that do not have strict delay budgets can derive optimum benefit from RTP/RTCP. Finally, the Real Time Streaming Protocol (RTSP) can contribute to streaming application functionality by facilitating a degree of user control [2].

QoS Approaches

Various approaches have been proposed and taken to further improve the QoS for VoIP applications. These can be viewed in terms of whether they involve measures taken within the sender or receiver endpoints or indeed within the intermediate network.

- **Sender:** From the sender viewpoint, the use of more bandwidth-efficient codecs reduces the bandwidth requirements from the G.711 64kbps POTS standard to much lower bit-rates such as G.723 which operates at 6.3 kbps or lower. Lower bandwidth requirements will reduce the likelihood of congestion and thus delay but can often come at a quality cost. This is particularly the case for source codecs which utilise simple voice models but also for hybrid codecs. As such a pure waveform codec such as G.711 has the highest quality rating (MOS > 4) whereas the hybrid codecs G.723 and G.729 have lower maximum MOS values of 3.8 and 4 respectively [24]. In section 5.2, the loss impairment for various codecs as quantified by the E-Model is described and illustrated in Fig. B.2. It shows that even at zero % packet loss, the 0 – 100 R-factor is reduced by 5, 11 and 15 for GSM-EFR, G.729 and G.723
respectively. Nonetheless, Barberis et al. have devised a scheme whereby the bit rate of the sender codec reacts to feedback on congestion obtained via the RTCP packets from receivers [33].

Similar techniques are also applied in some video conferencing applications to give priority to the audio stream. During periods of significant congestion and loss, video transmission is temporarily suspended to release bandwidth and thus improve the quality of audio. In [34] it is argued that sending video over TCP rather than UDP can actually improve the audio quality due to TCP’s inbuilt congestion control. As detailed above, the use of FEC-techniques in the sender can enable a receiver to recover a certain degree of packet loss though as outlined in [23], the degree to which it succeeds is dependent on the packet size and temporal dependency or burstiness of the loss pattern.

- **Network:** A great deal of research has been done on network level solutions which aim to differentiate between data flows and offer appropriate QoS regimes as opposed to the single best-effort service currently offered by the public Internet. Such research is thus not specifically aimed at the VoIP market. In the Internet community, the early 1990s saw the development by the IETF of the IntServ architecture [35]. IntServ offers different levels of service ranging from the default best-effort to controlled-load and finally guaranteed. This is facilitated by the Resource Reservation Protocol (RSVP) which as the name suggests, allows for bandwidth to be reserved in advance [36]. For a variety of reasons, including IntServ scalability issues, the IETF have also developed the DiffServ architecture [37]. DiffServ uses the IP TOS field (renamed the DiffServ CodePoint or DSCP field) to differentiate between traffic flows. DiffServ-enabled routers then aggregate traffic according to the DSCP field. Other network based QoS initiatives include the Multi-Protocol Label Switching (MPLS) architecture [38]. MPLS uses a system of labels that (similar to DiffServ) differentiate between traffic flows and route traffic accordingly. MPLS can run over different protocols and not just IP and is a switching architecture rather than simply a QoS feature. In recent years, huge growth has taken place in managed IP networks where QoS is built into the network using a variety of approaches, including those mentioned here. Large scale deployment of QoS features within the public Internet will however require significant infrastructural investment.
On a separate but related note, there has been significant growth in integrated voice and data networks particularly within the Local Area Network (LAN) environment. This has been facilitated by QoS enabled high-speed switching technology leading to successful integration of voice and data networks. See [39] for a useful white paper on the topic. In many such cases the VoIP network is confined to the enterprise LAN and is connected to the POTS via a gateway. [40] describes such an approach whereby IP telephony services are integrated to an existing PBX and to the POTS.

- **Receiver:** As outlined above in section 2.2.1, both PLC and/or FEC techniques are used within receivers to compensate for lost packets. Details on PLC techniques can be found in [41] though in general they range from the very simple replay-last-packet to more complex interpolation methods such as [42]. Both FEC and PLC strategies are discussed in more detail in section 3.5.

A significant body of work has focused on how receiver playout strategies can be optimised to strike the correct balance between packet loss due to late arrival and overall M2E delay. By increasing the buffering delay, the receiver waits longer for delayed packets at the cost of increasing the overall end-to-end delay. Finally, more recent work has focused on coupling strategies to compensate for loss and delay to optimise playout quality. Such approaches may involve both sender and receiver based mechanisms. In this context, the work of [43] combines a variable rate codec mechanism, similar to [33], along with FEC and adaptive playout as part of an optimisation problem.

In summary, the sender and receiver approaches described above can be seen as attempting to compensate for the limitations of the Internet’s best-effort service whereas the focus of network approaches is to move away from the default best-effort network and offer QoS services appropriate to the requirements of the particular application. This thesis is primarily concerned with the design of receiver-based playout strategies for optimising VoIP QoS. In the next chapter, past and current work in this area is reviewed before describing the approach taken in this thesis, which utilises synchronised time within the receiver playout strategy.
Chapter 3

Receiver-based Playout Strategies

As outlined in the previous chapter, the demand for Internet-based multimedia has prompted much research aimed at improving QoS either through compensating mechanisms within sender/receiver endpoints and/or by improving on the default best-effort Internet service level. The latter approach has to date been implemented mostly in private IP networks as the costs associated with large scale deployment in the public Internet are very significant. Multimedia applications generate periodic streams of data and the non-deterministic nature of best-effort Internet packet delays results in jitter that if not smoothed out, seriously degrades playout quality.

3.1 Fixed versus Adaptive Playout

Receiver-based playout strategies enable receiver endpoints to control to a significant degree the playout quality by delaying playout in order to absorb network jitter. The degree to which playout is delayed is a critical factor; a longer delay will result in fewer late packets but adds to the overall M2E delay. Playout approaches may be fixed or adaptive. Differing definitions of fixed playout strategies exist however. In this thesis, a fixed size playout strategy is defined as one where the receiver buffers a fixed amount of data before commencing playout whereas a fixed delay playout strategy is where there is a defined delay between packet sendtime and playout time, thus requiring synchronised clocks. Finally a further category is fixed playout delay whereby for the first arrived packet, an estimate is derived of network jitter and from this a fixed playout delay value is added to the arrive time. Subsequent packets are played out at intervals defined by the packet size. Many
current VoIP applications employ either fixed playout delay or more-so fixed size playout strategies which are the simplest to implement. Fixed approaches are limited in that they set and maintain a parameter (either buffer size or delay) at the start of a session and fail to react to changing network conditions during a session. As such they are ill-suited to best-effort networks where conditions can vary greatly. In chapter 6, a review of third party Internet delay studies as well as significant delay measurements are carried out to assess the extent of such variation in the current Internet.

Adaptive playout strategies on the other hand monitor and adapt to network conditions. As outlined in section 1.2, speech is comprised of talkspurts and silence periods. Adaptive approaches generally operate by altering the silence periods whilst maintaining the integrity of talkspurts although other approaches involving packet scaling also exist. In this chapter, an extensive review of adaptive strategies is undertaken, starting with developments from the early 1980s when the idea of integrating voice and data over packet-based networks began to emerge. Various approaches are evaluated, and their relative strengths and weaknesses assessed. Recent research that attempts to integrate strategies for dealing with both network loss and jitter in order to optimise playout quality is also reviewed. The chapter concludes by introducing and describing the hybrid playout strategy proposed in this thesis.

3.2 Background

Before embarking on a detailed review of research into adaptive buffering strategies, it is useful to take a broader view of the area in order to set the context. As such, the work of Montgomery [44] is summarised, which though dating back to 1983 introduced a range of issues that are examined in greater detail not just in this section but throughout this thesis.

Montgomery reviewed the problem of reconstructing voice after delivery across a packet-based network. His work predated the development of the RTP/RTCP protocols though he outlined many of the requirements specific to voice that are now evident within these protocols. His main focus was on techniques for setting the playout point to optimise playout quality. From an interactive viewpoint, he outlined that an upper delay bound of 250 msec is acceptable which has since been revised by ITU-T G.114 to 150 msec. The techniques outlined are as follows:
- **Blind Delay:** This strategy involves a *worst case* assumption about jitter. This value is chosen to ensure that very few packets will arrive too late for playout. This approach is similar to the *fixed playout delay* approach defined above. He suggests that such a strategy is suitable only in LAN environments where upper delay and jitter bounds are generally known and low, relative to permitted values.

- **Roundtrip Measurement:** This strategy utilises RTT measurements which are sent by each end and which are used to set playout times. Due to possible asymmetric delays, Montgomery recommends that such a strategy, though better than *Blind Delay* is only useful in WAN environments where network jitter is low.

- **Absolute Timing:** This approach requires synchronised clocks. In a pre-GPS world, Montgomery outlined that synchronisation can be achieved by distributing absolute time from a master clock to local clocks over communication channels with known delays. The *best-effort* nature of packet networks makes the latter unachievable and in general, he maintained that the costs of deploying time synchronisation are prohibitive and only feasible in a LAN environment.

- **Added Variable Delay:** In this approach, network node elements record both the arrival and departure times of packets at the ingress/egress interfaces respectively, calculate the associated delay and add it to a variable contained with the packet header. This value thus accumulates as the packet traverses the network and gives the receiver a total *variable* delay. The mechanism could also add the known propagation and transmission delays due to network links and interface speeds in which case the value represents total delay.

- **Adaptive Strategies:** Although the emphasis of the above approaches is to estimate and implement a fixed playout strategy (based on estimated or actual delay information), Montgomery outlined an adaptive strategy mainly as a means of correcting for errors in the above approaches. He suggested that adaption can be through expansion/contraction of silence periods or through changing the rate at which speech is played out whilst preserving the pitch of the speech. Regarding the former, he claims that such silence period distortion is not noticeable in the reconstructed speech.

More generally he suggested that both the *Absolute Timing* and *Added Variable Delay* approaches can greatly assist in network management. In particular the *Added Variable
Delay approach could facilitate strategies whereby queuing priorities are changed when variable delays are approaching a critical value, or packets are dropped where the accumulated delay renders such packets useless (and thus to allow them to continue on their path only serves to add unnecessary congestion). To implement such an approach however would require significant overhead at the network layer which has not emerged. Finally Montgomery also touched upon the issue of clock skew, whereby frequency differences need to be taken into account when setting playout times. As discussed in section 4.1.1, there is significant overlap/confusion in the literature regarding clock terminology. Montgomery uses the term drift in place of what is now generally defined as skew; the former is now more usually defined as the rate of change of skew.

Montgomery’s work introduced a range of issues that are addressed throughout this work. In the next section the significant body of work on receiver playout strategies is reviewed. The limitations of fixed approaches are outlined, and various adaptive approaches are evaluated and compared. The proposed hybrid approach is then described which is based on synchronised time and thus somewhat similar to the Absolute Delay approach proposed by Montgomery. Results from chapter 4 show that synchronised time can now be implemented cost-effectively across WANs which as Montgomery outlined, was not the case back in the 1980s. In section 4.2, his concerns over the use of RTT as an estimate for one-way delay are discussed though its use to capture general Internet delay characteristics is justified. Finally the whole issue of clock skew and its effect on delay measurements and multimedia applications is dealt with in detail in chapter 7.

3.3 Receiver Playout Strategies

In this section, the extensive literature on receiver playout strategies is reviewed. The objectives are to summarise this body of work and assess the relative strengths and weaknesses of the various approaches, rather than detail the absolute benefits or demerits of one approach over another. The majority of these approaches are designed to compensate solely for network jitter and leave the issue of network or link packet loss to be addressed separately. As outlined briefly in section 2.1, Internet Multimedia applications need strategies to cope with both network jitter and loss. As such a review is also carried out of recent work that attempts to couple network loss and jitter strategies together in order to optimise playout quality.
<table>
<thead>
<tr>
<th>Researcher</th>
<th>Reference</th>
<th>Label</th>
<th>Main Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Per Talkspurt Reactive Approaches</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ramjee et al</td>
<td>[45]</td>
<td>A</td>
<td>$\alpha = 0.998002, \beta = 4$</td>
</tr>
<tr>
<td>Ramjee et al</td>
<td>[45]</td>
<td>B</td>
<td>Spike mode, $\alpha = 0.875, \beta = 4$</td>
</tr>
<tr>
<td>Narbutt et al</td>
<td>[46][47]</td>
<td>A’</td>
<td>$\alpha = fn(\hat{\beta})$</td>
</tr>
<tr>
<td>Per Talkspurt Predictive Approaches</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Moon et al</td>
<td>[48]</td>
<td>C</td>
<td>Spike mode, History size= 10,000 Pkt.</td>
</tr>
<tr>
<td>Pinto et al</td>
<td>[49]</td>
<td></td>
<td>History size= Previous T’spurt, Optimum gap</td>
</tr>
<tr>
<td>Sreenan et al</td>
<td>[50]</td>
<td>D1</td>
<td>NLMS predictor $\hat{n}_i$, History size= 18</td>
</tr>
<tr>
<td>Sreenan et al</td>
<td>[51][52]</td>
<td>D2</td>
<td>Concorde, History size= all packets (+ ageing)</td>
</tr>
<tr>
<td>Per Packet Predictive Approaches</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Liang et al</td>
<td>[53][42]</td>
<td>E</td>
<td>Spike mode, Scaling, History size= 35</td>
</tr>
</tbody>
</table>

Table 3.1: Adaptive Algorithms for Jitter Compensation

Following this review, the key strengths, limitations and issues from all approaches are summarised and in so doing, the rationale outlined for the development of the hybrid playout strategy. The latter is then described in some detail and adaptive approaches chosen for baseline comparisons.

### 3.4 Adaptive Approaches for Jitter Compensation

Table 3.1 lists the various approaches reviewed that adapt to network jitter only. The Label column indicates how each are referred to in this thesis. Details of each approach are provided in appendix A.1.

#### 3.4.1 Fixed versus Adaptive Playout Strategies

In [12], general comparisons are made between fixed and adaptive playout algorithms (Alg. B above). Extensive testing was carried out over US backbone paths. Unlike most of the above approaches, the ITU-T E-Model was used to quantify relative performance rather than delay-loss curves. The E-Model is described in detail in section 5.2. The main points of their analysis are as follows:
• Adaptive algorithms such as A & B that utilize the TCP-like formula such as equations A.1 and A.3 from appendix A tend to overestimate delays. Of a total of thirty three US Internet backbone paths whose delay/jitter and loss characteristics made them ideally suited to delivering VoIP, Alg. B performed poorly on all these 33 paths and only performed well on long distance loaded paths. More detailed analysis of this study is carried out in section 6.5.2.

• For reactive approaches, the selection of parameters such as α and β and spike threshold is a non-trivial matter and tuning is required to suit network characteristics. Such characteristics may however change with time. They outline that in tests where tuning parameters were adapted during calls, overall loss rates were reduced but that occasional periods of very high losses were reported. As outlined in appendix A.1.1, work by [46] also addresses this issue and good performance has been reported from actual measurements and simulation. No further details however are provided by [12].

• A fixed delay approach was implemented using GPS clocks for time synchronisation. Settings of both 100 and 150 msec were used and resulted in very different performance levels due to the particular delay characteristics. Their main finding is that fixed delay approaches perform well but only if settings are based on current delay characteristics.

### 3.4.2 Summary of Adaptive Playout Strategies

Considering the totality of playout strategies and tests outlined above and described in appendix A.1, the following are the main points that emerge from general and comparative analysis:

• Alg. A-E are designed to operate with some late loss although the degree to which they do so varies. The predictive approaches C-E give the receiver better control over end quality as they are based on extracting from recent data, a playout delay that is likely to meet the user-specified target, presuming that past trends remain relevant.

• Alg. A-C and D1 suffer from distortion of inter-talkspurt silence periods whereas Alg. E suffers from packet scaling (distortion of both silence periods and talkspurts). Scaling results in distortion but [53] outlines a significant net gain in MOS relative to per-talkspurt approaches. Although it is reported in [44] and subsequently referenced
by both [45] and [48] that silence period distortion does not impact noticeably on quality, little research exists to support this view. Undoubtedly, severe silence period distortion can affect the emphasis within speech which can be a serious issue.

- Although relative comparisons are presented in many of the above approaches, the conditions under which testing is carried out has a significant bearing on results. For example, Alg. E reacts to short spikes within talkspurts which explains its significant gains when tested over high jitter networks. Nonetheless Alg. E will result in lower loss rates for a given average playout delay though the WSOLA mechanism has a significant time complexity.

- Some of the playout strategies are implemented differently across the different tests which limits the extent to which an overall comparison of the various approaches can be made. For example, the original Alg. C proposed by [48] is implemented with a history of 10000 packets. In relative performance tests for Alg. E, a modified version of Alg. C with a history of 300 is used based on experimentation. In [54] introduced below in section 3.5, Alg. C is used though with a history of 1000, as 10000 was found to be too large and lacked responsiveness.

- The above point introduces what is undoubtedly the single most critical issue regarding adaptive playout strategies; the choice of tuning parameters. For reactive adaptive approaches such as Alg. A & B, the choice of $\alpha$, $\beta$ and spike threshold is critical and optimal values depend on network characteristics. [46] illustrates this by adapting the parameter $\alpha$ during a session and reports good overall performance though [12] report mixed results. All of the parameter values specified above are based on experimental results. Eg. spike threshold of $2 \times \hat{v}_i + 100$ in [45] is replaced by multipliers of 2 & 4 in [48]. The choice of $\beta$ reflects the tradeoff point between late loss and delay.

Similarly for the predictive approaches described i.e. Algorithms C, D1, D2 and E, all maintain a history of past delay values, though the extent of history is set at 10000, 18, all previous data, and 35 respectively. Ageing techniques are applied to Alg. D2 so that most recent delays are given a higher weighting and report that an ageing interval of 1000 packets has no noticeable effect, regardless of the ageing coefficient. This suggests that the 10000 value chosen by [48] though presumably appropriate
for their tests is generally too large. This is borne out by both [53] and [54] that significantly reduce the history value of Alg. C to 300 and 1000 respectively.

- At the start of this chapter, the distinction is made between fixed delay, fixed playout and fixed size strategies. A fixed delay strategy is the most difficult to implement, requiring time synchronisation. Both fixed playout and fixed size strategies on the other hand can be implemented by the receiver alone without any sender involvement and are used in many VoIP applications. Fixed delay strategies were tested in [42] and [12] with settings of 160 and 100/150 respectively whereas a fixed playout approach was used by [52] with a setting of 200 msec. In general, fixed strategies are inappropriate in best-effort networks where delay can vary significantly. Even if correctly chosen at the start of a session, a setting may quickly become inappropriate resulting in high late losses or unnecessarily high delays. For example, [52] report loss rates ranging from 0.2 to 94% depending on network conditions. On the other hand, fixed strategies maintain the integrity of sender speech both within and between talkspurts which is a significant benefit relative to adaptive approaches. Recall that Montgomery in [44] outlined that the blind-delay approach which is similar to fixed playout is suitable only in LAN environments where delays are usually small and bounded, and well within the G.114 150 msec limit. Considering the move to Fast and Gigabit Ethernet in recent years, coupled with QoS-enabled switches, this assertion is even more valid.

- Despite their shortcomings, adaptive approaches are far superior to fixed approaches. All of the adaptive algorithms described above do not require synchronised clocks and can be implemented without any sender involvement. Both reactive and predictive approaches monitor trends in network delay and as such utilise estimates rather than actual delays. Historically, receivers generally had not the capacity to determine actual one-way delays and adaptive algorithms work well in this context. If actual delays are known, and are within G.114 requirements, an informed fixed playout delay can avoid many of the problems associated with adaptive algorithms. Reactive adaptive approaches are tuned to track network conditions and thus even if actual delays are well within the G.114 limit, adaptive approaches are delay-unaware and will continue to track network characteristics resulting in unnecessary late losses. Similarly, predictive adaptive approaches explicitly target loss rates based on histori-
cal data though the actual loss rate can deviate from this. In any event both reactive and predictive approaches adapt by distorting sender speech (either through silence period distortion or packet scaling).

3.5 Adaptive Approaches for Jitter and Loss Compensation

In this section, recent research is reviewed that attempts to combine or couple together, strategies to deal with both network jitter and loss rather than considering them separately.

In all of the approaches considered to date, playout strategies are solely concerned with adapting to network jitter that results from best-effort service. In addition to jitter, link packet loss can be a serious problem. Many general Internet studies such as [55], [56] and [57] have attempted to characterise both delay and link loss. Other studies such as [58], [59], [23], [60], [21] and [12] have done so specifically in the context of Internet Multimedia and VoIP.

A robust VoIP application needs to address both issues. The impact of network or link loss can be compensated for through either Forward Error Correction (FEC) or Packet Loss Concealment (PLC) techniques. The former builds in redundancy into the transmitted audio to enable lost packets to be reconstructed at the receiver. FEC techniques are classified in [43] as either media independent or media specific. The media independent approach recovers an exact replica of the packet through bitwise redundancy. The media specific approach (also called signal processing FEC) is more common whereby redundancy is achieved by sending copies (perhaps multiple) of each packet encoded using lower bit rate codecs. As such the latter approach does not involve a correction mechanism and the term FEC is somewhat inappropriate. In fact, [23] distinguishes between and compares the two FEC approaches referring to the latter as Low Bitrate Redundancy (LBR) rather than FEC.

PLC techniques on the other hand involve no measures at the sender and can be considered as a second line of defense after FEC. With PLC, the receiver attempts to correct for lost packets using techniques ranging from the simple silence substitution or previous packet repetition to more advanced interpolation techniques. The packet scaling technique introduced above in [42] can be used both for adaptive playout and PLC.

Further details on both FEC and PLC techniques can be found in [41].
<table>
<thead>
<tr>
<th>Researcher</th>
<th>Reference</th>
<th>Algorithm</th>
<th>Main Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rosenberg et al</td>
<td>[54]</td>
<td>Virtual Delay (play first)</td>
<td>History= Previous T’spurt adjusts $\beta$</td>
</tr>
<tr>
<td>Rosenberg et al</td>
<td>[54]</td>
<td>Previous Optimal</td>
<td>History= Previous T’spurt Recursive filter $\rho = 0.25$</td>
</tr>
<tr>
<td><strong>Network Aware FEC</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Boutremans et al</td>
<td>[43][61]</td>
<td>N1</td>
<td>Sender selects FEC based on network info.</td>
</tr>
<tr>
<td>Boutremans et al</td>
<td>[43][61]</td>
<td>N2</td>
<td>Sender selects playout delay &amp; FEC based on network info.</td>
</tr>
</tbody>
</table>

Table 3.2: Adaptive Algorithms for Jitter and Loss Compensation

With FEC, the redundancy for packet $p_i$ is built into subsequent packets and thus in the event of $p_i$ being lost, the receiver must wait for delivery of subsequent packets before $p_i$ can be reconstructed. For this reason, recent studies have examined the need for adaptive playout and FEC strategies to be considered together (or coupled) rather than separately. Essentially, there is little point in implementing FEC if a playout strategy is not tuned to wait for such data in the event of packet loss. On the other hand, audio applications (particularly those with built in PLC) can tolerate a degree of packet loss without significant loss in intelligibility and thus to blindly implement an additional delay to facilitate FEC where overall loss rates are very low can be counter productive. A further complicating issue is that with multiple levels of redundancy (eg. where packet $p_i$ details are sent with $p_{i+1}, p_{i+2}, p_{i+3}$ etc), the additional delay required to reconstruct a lost packet presents another variable.

Table 3.2 lists the various approaches reviewed that attempt to combine network jitter and loss compensation strategies. Details of each are provided in appendix A.2.

### 3.5.1 Summary of Combined FEC-Playout Strategies

In summary, playout strategies that operate in tandem with FEC strategies will deliver better quality than if they operate in isolation. Furthermore, FEC strategies that have ac-
cess to network characteristics such as network delay and loss, and can adapt FEC strategy accordingly will further improve playout quality. To implement the latter however requires a mechanism to deliver this information. Currently, RTCP packets provide some loss and jitter information to senders though little additional information other than occasional estimates of RTT is available on delay.

### 3.6 Rationale for Proposed Hybrid Playout Strategy

The core points from sections 3.4.1, 3.4.2 and 3.5.1 are as follows:

- Adaptive playout strategies are most useful when receivers have no knowledge of actual network delay which is generally the case, or where delays are known and excessive.

- Both predictive and reactive approaches result in some degree of late loss. Many predictive approaches explicitly target a degree of loss whereas reactive approaches do so implicitly (eg. through parameters $\alpha$ and $\beta$). The performance of both approaches is influenced heavily by tuning parameters and the setting of such parameters is non-trivial.

- Adaptation is done through silence period distortion and/or packet scaling. Packet scaling from [53] has been shown to result in a non-negligible quality degradation. Although reported in the literature that silence period distortion has no noticeable quality implications, there is a lack of supporting qualitative research.

- Fixed strategies, though simple to implement (particularly in case of fixed size and fixed playout delay strategies), and avoid any silence period distortion or packet scaling, are unsuited to the changing characteristics of best-effort networks.

- Users are more tolerant of increased delays than increased losses.

- A combined FEC/playout strategy will optimise playout quality, particularly if the FEC scheme is delay/network aware.

The above points collectively outline the rationale for the hybrid playout strategy.
3.6.1 Hybrid Playout Strategy

The hybrid strategy operates as follows:

- Session commences implementing an adaptive playout algorithm.

- The availability of synchronised time enables one-way packet delays to be precisely determined. Each receiver builds up a histogram of delays and from this extracts a delay estimate value \( est \) to meet target loss requirements.

- In the following pseudocode, \( Pkt \) refers to the packetisation delay and \( Wf \) refers to a weight factor (0-1) applied in determining a fixed delay playout point. Recall that the G.114 limit is 150 msec.

\[
\text{If } (est < (150-Pkt)) \\
\quad \text{playout} = est + (150-Pkt-est) * Wf \\
\quad \text{Fixed-delay Playout Mode} = \text{TRUE} \quad \text{// switch to fixed mode}
\]

\[
\text{Else} \\
\quad \text{Maintain adaptive playout mode}
\]

- Maintain rolling histogram and at intervals, recalculate delay estimate \( est \) as follows:

\[
\text{If } (est < (150-Pkt)) \\
\quad \text{playout} = est + (150-Pkt-est) * Wf \\
\quad \text{Fixed-delay Playout Mode} = \text{TRUE} \quad \text{// switch to fixed mode}
\]

\[
\text{Else} \\
\quad \text{Maintain adaptive playout mode}
\]

43
If (est < (150-Pkt))
    playout = est + (150-Pkt-est) * Wf
If (Fixed-delay Playout Mode)
    If (conditions improved)
        Decrease fixed-delay playout point
    Elseif (conditions deteriorated)
        Increase fixed-delay playout point
Else
    Fixed-delay Playout Mode = TRUE  // switch to fixed mode

Else // high delays
    If (Fixed-delay Playout Mode)
        Fixed-delay Playout Mode = FALSE  // switch to adaptive mode
    Else
        Maintain adaptive playout mode

The hybrid strategy maintains a histogram of actual network delays and uses this to implement a fixed delay playout point whenever possible. As such it alternates between fixed delay and adaptive strategies but the fixed delay mode also tracks the network conditions. It thus avoids the inflexibility usually associated with fixed algorithms, be they fixed delay, fixed playout or fixed size. The weight factor $W_f$ provides an extra delay margin by positioning the fixed delay playout point between the extracted delay estimate $est$ and the G.114 limit. Increasing $W_f$ will result in greater reductions in late losses at the expense of a higher fixed delay playout point. As outlined previously and discussed in more detail in section 5.2, users are more sensitive to increased loss than to increased delay. This forms the basis for the development of the hybrid playout strategy in that a higher (though fixed) delay is imposed, resulting in lower late losses and overall better quality. Details of Hybrid tuning are provided in the next section.
3.6.2 Hybrid Strategy Tuning

- **Histogram Size**: As summarised in section 3.4.2, the choice of sample size to maintain the correct amount of history and thus responsiveness in predictive approaches varied greatly between those approaches described, ranging from 18 to 10,000. The latter value was deemed too large by a number of these approaches and thus a sample size of 200 was chosen for the hybrid, which equates to 6 – 12 seconds of data for 32 msec packets.

- **Extracted Delay Estimate**: The delay estimate est extracted from the histogram is based on that required to meet 98% of the sample.

- **Dynamic Wf**: Wf is initialised to 0.33 but subsequent values are a function of jitter/median. The use of median rather than mean eliminates outliers that otherwise distort the mechanism and is generally accepted as a more useful metric for such analysis. Regarding the term jitter, different definitions exist as outlined in RFC 3393 [17]. The interpretation applied here is that of RFC 1889 [1] for RTP.

  Floor and ceiling values are also applied to maintain Wf in the range 0.1 – 1. Essentially, this means that the safety margin applied is proportional to network jitter.

- **Dynamic Estimation Interval**: The frequency at which playout mode is re-evaluated is set dynamically according to network conditions. A baseline value of 500 packets is used (which equates to 16 sec with 32 msec packets) but subsequent intervals are a function of \((150 - Pkt)/median \) * 500. This ensures that where delays are running close to the G.114 limit, the operating mode is monitored more closely.

- In between checks, the late loss rate resulting from the imposed fixed playout delay is continuously monitored and if this rate exceeds a user specified threshold, the re-evaluation mechanism is initiated.

The core benefit of the hybrid strategy is that a fixed delay playout point is implemented whenever possible i.e. when M2E delays are within the G.114 limit, so that the integrity of speech both within and between talkspurts is maintained. Adaptive mode is thus used only in the initial start-up period and if M2E delays exceed the G.114 limit. A summarised version of the motivation behind the development of the hybrid playout strategy can be found in [62].
3.6.3 Choice of Adaptive Playout Strategy within the Hybrid and for Comparative Testing

To facilitate adaptive playout mode within the hybrid playout strategy and also to evaluate its overall effectiveness relative to other approaches, a choice of adaptive playout strategies had to be made. As evident from section 3.4.2, choosing the optimum adaptive strategy from the various approaches described that works best under all conditions is a non-trivial task. Each of the approaches have their relative strengths and weaknesses and results presented are based on tuning parameters derived from experimental testing. With regard to the hybrid playout strategy, the objective was not to carry out exhaustive testing to assess whether it outperformed all adaptive strategies under typical Internet conditions. Such an approach would present major challenges. Firstly it would require tuning of the various adaptive approaches for each network condition being tested rather than use the reported parameters to ensure a fair comparison. Secondly, as reported by [55] and [8] the degree of heterogeneity within the Internet and its constant state of flux are such that a typical Internet condition is impossible to define. Rather the objectives were twofold:

1. To assess the technical feasibility of implementing the hybrid playout strategy in a WAN environment.

2. To assess its applicability over a limited set of diverse Internet paths.

As such the choice of adaptive strategy both for integration within the hybrid and also for comparative testing was not driven by the need to quantify absolute performance gains.

Table 3.3 provides a summary of the approaches to comparative testing taken by the various researchers. From section 3.3 and appendix A, seven of the eight approaches subsequent to the first described [45] use Alg. A and/or B from [45] for comparative testing. Other research, not described here [63] [64] use Alg. A and Alg. C respectively. Only [53] includes neither A or B in comparative testing, choosing a fixed delay and Alg. C instead. As such, Alg. A was chosen as the adaptive approach within the hybrid strategy and Alg. A & B chosen for comparative testing.

Hybrid Playout Strategy Implications for Coupled FEC/Playout Strategy

With precise delay information available on a per-packet basis through use of the hybrid playout strategy, receivers can make more informed choices about FEC. The hybrid play-
<table>
<thead>
<tr>
<th>Research</th>
<th>Reference</th>
<th>Algorithm</th>
<th>Comparison</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moon et al.</td>
<td>[48]</td>
<td>C</td>
<td>A, B</td>
</tr>
<tr>
<td>Sreenan et al.</td>
<td>[50]</td>
<td>D1</td>
<td>A</td>
</tr>
<tr>
<td>Sreenan et al.</td>
<td>[51][52]</td>
<td>D2</td>
<td>B, Fixed</td>
</tr>
<tr>
<td>Liang et al.</td>
<td>[53][42]</td>
<td>E</td>
<td>C’, Fixed</td>
</tr>
<tr>
<td>Rosenberg et al.</td>
<td>[54]</td>
<td>Coupled FEC</td>
<td>A, B, C”</td>
</tr>
<tr>
<td>Boutremans et al.</td>
<td>[43][61]</td>
<td>Network Aware FEC</td>
<td>A</td>
</tr>
<tr>
<td>Roccetti et al.</td>
<td>[63][64]</td>
<td></td>
<td>A, C</td>
</tr>
<tr>
<td>Pinto et al.</td>
<td>[49]</td>
<td></td>
<td>A, B</td>
</tr>
<tr>
<td>Narbutt et al.</td>
<td>[46][47]</td>
<td>Variable α</td>
<td>A</td>
</tr>
</tbody>
</table>

Table 3.3: Comparative Testing: Choice of Algorithms

out strategy conservatively sets the fixed delay playout point thus the likelihood of FEC data being available for packet recovery is much greater. In any event, with a coupled approach, receivers can choose to wait for FEC data based on precise delay information. This represents a significant improvement of the coupled approach of [54] in that where packet loss has occurred, the receiver can make an informed decision regarding the net effect of waiting for subsequent packets to enable reconstruction, rather than blindly doing so. From the sender’s perspective, the combination of RTCP report blocks (which enable the sender to determine Round Trip Time) and precise one-way delays for incoming packet means that outbound delays can also be determined. This will facilitate the network-aware FEC approach described by [43] and shown to perform better than [54].

As such the hybrid playout strategy greatly facilitates the coupled approach to both network jitter and loss compensation. The focus of this thesis however is primarily on strategies for jitter compensation and thus the potential of the hybrid approach in the context of coupled strategies is not examined further.

3.7 Summary

In this chapter, the significant body of work relating to receiver-based playout strategies was reviewed. Since Montgomery’s early work, many different approaches have been proposed and tested. Most of the approaches are concerned solely with compensating for the effects
of network jitter though some recent work has attempted to integrate strategies for dealing with both network jitter and loss. From this review, the core issues were identified and from this emerged the rationale for the hybrid playout strategy. The two core objectives for evaluating the hybrid playout strategy were outlined. The first concerns the technical feasibility of implementing the hybrid whereas the second concerns its applicability within the Internet. Regarding the former, a critical requirement within the hybrid playout strategy is synchronised time. In [44], the Absolute Timing technique, based on synchronised clocks was outlined but disregarded at the time due to the significant problems relating to its implementation. The next chapter discusses in detail the problems relating to the implementation of cost-effective distributed synchronised time and outline how they can be overcome. In subsequent chapters, remaining issues regarding its feasibility and the other core objective of applicability are addressed.
Chapter 4

Synchronised Time and Delay Measurement Issues

The previous chapter reviewed and summarised much of the research in adaptive playout strategies before proposing and describing the hybrid playout strategy. The latter requires that receivers have precise delay information and thus requires time synchronisation between endpoints. Meeting these requirements is a non-trivial task and raises other issues such as clock skew and drift, clock resolution and Operating System (OS) jitter.

This chapter firstly examines the operation of computer clocks outlining the principal reasons for error and the extent of such errors. It moves then to a more general discussion on delay measurement and the uncertainties in measurement that can arise. It briefly describes the operation of NTP, focusing on robustness issues that need to be addressed in an NTP subnet to ensure effective synchronisation. Although NTP has been fine tuned over the years, reported performance levels have been mixed.

A number of testbeds are designed to evaluate the performance of NTP within both a Local Area Network (LAN) and Wide Area Network (WAN) environment. Issues examined include peer redundancy and path diversity as well as the proximity of local reference sources. Furthermore, the extent of Operating System (OS) dependency of NTP and the effect of diurnal traffic patterns are assessed. Results show that with careful attention to detail, NTP can provide the levels of time synchronisation across WANs required by the hybrid playout strategy.
4.1 Clock Issues

Computer clocks fundamentally consist of a quartz oscillator and a counter. The oscillator generates a consistent frequency or *tick* at which the counter is incremented. The counter value is then translated to a time convention or standard. The Universal Coordinated Time (UTC) standard is the most widely used standard. It evolved from Greenwich Mean Time (GMT) and though based on International Atomic Time (TAI), it incorporates adjustments in the form of leap seconds to keep it aligned with the earth’s rotation. A discussion on UTC, TAI and GPS (Global Positioning System) Time Standards is given in [68].

Conventional PCs have both a hardware and a software clock. The hardware clock, known as the Real Time Clock (RTC), uses a low frequency oscillator (typically 32,768 Hz) to minimise power consumption. On system startup, the software clock, hereafter referred to as the *system* clock, is initialised to the RTC but thereafter runs independently. The system clock operates as follows; processor interrupts are generated at regular intervals, determined by an oscillator (usually separate from the RTC oscillator: the 8254 timer chip operating at 1.193182 MHz is common), whereby a *tick* is added to a system variable representing the system clock time. In addition to the RTC and system clocks, and from a VoIP perspective, soundcards utilise further clocks to set the sampling rate for recording and playback. For clarity, such clocks will be referred to as *audio* clocks to distinguish them from both the RTC and system clocks.

Oscillator quality is generally described in terms of both *accuracy* and *stability*. Accuracy relates to how close the oscillators frequency is to its rated value. Stability relates to how the frequency varies as a function of parameters such as time and temperature. The principal reason for clock error is inaccuracy i.e. incorrect oscillator frequency. Oscillator frequency is largely determined by the manufacturing process and is thus subject to manufacturing tolerances. From a stability viewpoint, oscillator frequency is primarily affected by ageing and temperature. Other factors include oscillator noise, physical shock, magnetic and radiation effects. This thesis is largely concerned with oscillator accuracy which particularly in a VoIP context, is the more serious problem. Generally, system clock frequency errors are rarely above 100 microseconds per second (or ppm). However, some research has shown that audio clock errors can be much greater than this running into thousands of ppm.

The extent and effect of clock errors on VoIP performance is dealt with in detail in
chapter 7. In a survey by Mills [69] of 20,000 Internet clocks in 1999, a median frequency error of 78 ppm was reported. Fig. 4.1 shows results of previous work by Melvin et al [70] that reported an instance of system clock frequency error of 18 ppm.

4.1.1 Clock Terminology

To avoid confusion regarding clock-based terminology, this section outlines the definition of such terms as applied in this thesis. The approach taken draws both from Paxson in [27] and [71], and from Mills in [7]. These differ somewhat in the definition of some terms; for example, an accurate clock in Paxson’s work is one where its offset is zero at a particular instant whereas with Mills’ terminology, accuracy refers to general correctness.
involving skew and offset.

- Offset: Defined here as the difference between the time reported by the clock and true time as defined by the UTC standard. However, in a VoIP context, the time differences between sender and receiver clocks is of more relevance, in which case offset may refer to the offset of one clock relative to another, rather than to a true (UTC) clock. Although both terms are used in this thesis, the precise meaning will be clear from the context.

- Skew: Defined here as the difference in frequency between the clock and a true clock. Skew is normally measured in microseconds per second or parts per million (ppm). As above relative skew is often of greater interest, meaning the skew of one clock relative to another, rather than to a true (UTC) clock.

- Drift: Defined here as the rate of change of a clock’s frequency with respect to true time defined by the UTC standard. As detailed above clock drift can occur due to factors such as ageing or temperature changes. The extent and effect of drift are usually minimal in the context of a VoIP session where large temperature fluctuations are unlikely and the timescale does not approach that where ageing becomes an issue. The linear nature of Fig. 4.1 illustrates this point. Temperature effects are however important in that relative temperature differences between sender and receiver endpoints may vary significantly between VoIP sessions rather than within sessions and thus so will relative skew. The same argument applies to ageing though if similar crystal types are used they will have similar ageing characteristics, resulting in no change in relative skew.

The issue of skew is important for VoIP in that it can degrade both jitter buffer performance and accurate delay measurement. As such, this topic is dealt with in detail in chapter 7.

### 4.2 Delay Measurement Issues

Accurate one-way delay measurement within the Internet presents a significant challenge. As outlined in section 3.2, Montgomery in [44] evaluated a number of approaches to delay estimation including the use of Round Trip Times (RTT), distributed synchronized time and a novel approach involving a variable delay estimation mechanism within routers,
requiring a specific protocol format. The requirement for zero relative offset between sender and receiver host clocks is an obvious requirement for accurate one-way delay measurement but the issue is more complex than that. Where delay values are required over a period of time, the clocks must also be running at true clock frequencies i.e. have zero skew.

Both RFC 2330 [71] and RFC 2679 [15] describe in detail the problems that arise due to separate clocks at end-hosts. RFC 2679 in particular looks at the sources of error and uncertainty that arise in measuring end-to-end delays for Internet traffic. As outlined previously in section 2.3, RFC 2679 introduces the concept of host time versus wire time. It recommends that Global Positioning System (GPS) clocks be used in conjunction with high quality (i.e. low skew rate) system clocks to minimise clock-related uncertainty. In addition to offset and skew, other uncertainties relate to issues such as clock resolution, Operating System jitter and other hardware/software uncertainty. Some of these issues are addressed briefly below but dealt with in more detail in section 6.6.1 of chapter 6.

- Clock Resolution: As detailed above, computer clocks are implemented as counters that are incremented periodically, at intervals known as ticks. The ticksize thus determines the granularity of the counter or resolution of the clock. On some platforms, this is configurable such as the Unix real-time variant QNX where the ticksize utility can vary ticksize from 0.5 to 50 msec. Paxson in [72] develops an algorithm for determining clock resolution from measurements. He reports a granularity of 10 msec for IRIX, SunOS and Solaris and less than 1 msec for Digital Unix, though also reports some hardware dependencies. A resolution of 10 msec is still typical for Windows and many Unix variants whereas others such as DEC Alpha AXP systems have a ticksize of 7.8125 msec. In situations where delay measurements are required to single millisecond level or better, such resolutions can present a serious impediment.

- Operating System (OS) Issues: System time is made available to applications through specific system calls eg. gettimeofday() in Unix variants or GetSystemTimeAsFileTime() in Windows. OS scheduling in conventional multitasking systems differs from Real-Time OS systems in that no guarantees are given regarding service times and thus OS jitter can introduce errors to delay measurement. More fundamentally, if the system call is limited by the resolution issue described above then finer resolution is not possible. Paxson in [27] reports that some Unix variants implement what he terms monotonicity increments within a tick interval simply to enforce an increasing
clock.

In more recent years, solutions to resolution limitations have emerged. Many Unix
variants including Linux now provide a mechanism whereby the time elapsed between
the default 10 msec interrupt interval is interpolated based on the CPU cycle counter.
It is expected that Windows OS will also provide this facility though tests carried
out on Win2000 machines in 2003 at NUI,G showed that the system time remained
limited by 10 msec granularity [73]. More detailed work by [74] outlines developments
within Windows OS to improve timing performance. They describe the Multimedia
Timer facility that in theory allows the 10 msec interrupt frequency to be reduced to
1 msec. They compare the performance of the HALX86 and HALMPS approaches
and report very poor performance from the latter. Through tests, they show that
this is due to its use of the Real Time Clock (RTC) as source of interrupts rather
than the 8254 chip.

More generally they express concern over the use of conventional Operating Sys-
tems for precision timing. In this regard, Real Time Operating Systems (RTOS)
offer better bounds on processor scheduling as discussed by Melvin et al. in [70].
Section 6.6.1 revisits this topic in greater detail. An alternative approach is to use
specific hardware/software to reduce this uncertainty.

• In determining network delays rather than end-to-end delay, uncertainty can also
arise from the fact that the send and receive timestamps generated by system calls
are usually implemented at the application level. Where network delay measure-
ments are required to extreme levels of precision, the delays involved in traversing
the protocol stacks at either end may need to be considered. Furthermore, packet
departures or arrivals are not atomic events in that depending on the link or interface
speed, the transmission or serialisation delay may be relatively significant. In [75] a
measurement architecture is described that has a resolution of 60 nanoseconds. As
such, serialisation delays are often orders of magnitude higher than this as outlined
in Table 4.1 taken from [75]. It is thus important to clarify whether timestamps
refer to the start or finish bits of a packet. In RFC 3393 [17], delay is defined as the
difference between when the first bit of the packet is sent by the source and the last
bit is received at the destination.

For VoIP, the precision of delay measurement needs to be seen in the context of the
\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
\textbf{Data} & \textbf{Link} & \textbf{Serialisation Delay} \\
\hline
64 byte packet & 10BaseT & 51200 nanoseconds \\
64 byte packet & 100BaseTX & 5120 nanoseconds \\
ATM Cell 53 bytes & OC3c & 2720 nanoseconds \\
ATM Cell 53 bytes & OC48c & 168 nanoseconds \\
\hline
\end{tabular}
\caption{Serialisation Delay}
\end{table}

G.114 150 msec limit and thus that required for the hybrid algorithm is of the order of low single msec values. As such, OS scheduling, serialisation or protocol stack delay uncertainties are largely disregarded in this analysis though OS issues are revisited in section 6.6.1.

Regarding active delay measurement, RFC 2330 [71] outlines that a poisson-derived traffic model should be used as periodic test packet generation can lead to synchronisation issues or interference with network characteristics that are being measured. However, in many instances, delay measurement may not be the sole objective in that the impact of data flows such as multimedia streams on overall delay characteristics may also be significant. This is so in the case of research carried out by [45], [56], [59], [60] and others detailed in section 3.3 and is the approach taken here in evaluating the hybrid playout strategy. As such, the more recent RFC 3432 [76] which deals specifically with network measurements employing periodic streams (such as multimedia applications) is a more appropriate reference.

\subsection*{4.2.1 The value of RTT}

Although precise one-way delay measurement is obviously superior to estimating one-way delays from RTT measurements, it poses significant implementation difficulties. RTT on the other hand requires neither synchronised time or remote host cooperation, particularly if a utility such as ping is used. As detailed in [28], the main RTT uncertainty relates to delay asymmetry. This can be due to unequal load patterns (and thus different levels of congestion and queuing delay in either direction) and/or different routing paths on the outward and return journeys. A distinction thus exists between delay asymmetry and path asymmetry as the latter may not lead to the former. Paxson [72] in 1997 carried out
a comprehensive study of Internet routing and packet dynamics involving 1000 different paths. Path asymmetry was found in about half of all routes and significant delay asymmetries were also reported. However, the NPD measurement framework used was based on large file transfers, thus leading to unequal loading patterns with data transfer in the outward direction and much lower ACK traffic on the return path. As such, the test traffic characteristics of Paxson’s work is very different to that of a typical VoIP session where the bandwidth used is minimal and generally symmetric.

As described in section 5.3.3, the ping utility was used in this thesis to model multimedia traffic and capture RTT delay data to various remote hosts. The objective regarding the use of RTT was to capture the general characteristics of diverse network paths and thus RTT data was considered sufficient. Both [56] and [64] use round-trip-times, the former for general delay monitoring and the latter within a proposed playout strategy for VoIP. In [59], a technique presuming symmetric paths for estimating one-way delays from RTT by determining clock offsets is described.

Although RTT remains a simple yet unreliable delay estimation mechanism in today’s Internet, the potential for use of distributed synchronised time has greatly increased in recent years. This is due in part to the more widespread and careful deployment of the Network Time Protocol (NTP) but more importantly, to the increased availability of cheap yet highly accurate time sources such as GPS receivers.

4.3 NTP Operation

NTP has been in existence for more than 20 years but has evolved significantly in the interim and though version 4 is available and widely implemented, version 3 [7] remains the Internet standard. Fig. 4.2 illustrates the NTP operational model. Essentially a host running NTP operates by querying a number of peer NTP servers regarding their estimate of current time (NTP operates to the Universal Coordinated Time (UTC) standard). From the packet exchanges it also calculates Round-Trip-Delay (RTD) and estimates the relative offset to each peer presuming symmetric delays for the forward and reverse directions. Note that RTD is different from RTT in that it does not include the delay within each peer.

\[
RTD = RTT - \text{Delay}_{\text{at\_peer}}
\]  

(4.1)

Based on this information, it derives an estimate of true UTC time and adjusts its
NTP Operational Model

Figure 4.2: NTP Operational Model
system clock accordingly. Any asymmetry will result in incorrect offset estimates yet as outlined in the previous section, significant asymmetries are not uncommon. A range of algorithms are thus used in this process to implement filtering (to remove noise), intersection and clustering (to remove false-tickers and outliers, to detect asymmetries and to select strong candidates), and combining (to generate a UTC estimate for final adjustment). The adjustment process is step-like where large errors exist, otherwise the adjustment is more gradual.

It is important to note that NTP does not synchronise clocks to each other but rather to UTC. Although NTP was developed within the Unix community, it has recently been ported to Windows OS variants. NTP is hierarchical in design with servers classified by stratum as illustrated in Fig. 4.3. At the apex, stratum 1 NTP servers (or primary servers) have direct access to a local time reference. Such references can be based on radio signals or increasingly GPS clocks. Redundancy is a key requirement for effective NTP operation and thus although not shown in Fig. 4.3, stratum 1 servers should also peer with other stratum 1 servers to prevent against single points of failure (SPOF). A similar argument applies to the stratum 2 and 3 servers shown.

The focus of this section is not to review or comment on details of NTP operation; such information can be found in [7], [69] and [77] or at the NTP website [78]. Rather, the focus is on assessing the impact of a variety of factors such as proximity to local reference source, robustness of NTP subnet, OS platform and diurnal traffic patterns on NTP performance. As such it firstly reviews robustness issues that are critical in determining NTP performance before briefly outlining OS dependency issues. It then describes the testbeds developed to analyse NTP performance under both Linux and Windows platforms and present results.

### 4.3.1 Robustness Issues

For NTP to deliver stable and accurate performance, the following need to be addressed:

- **Redundancy:** An end host running NTP must include a minimum of three peers with which it can communicate.

- **Independence:** It is important that the various peers do not share common reference sources in order to minimise the risk of multiple peers being incorrect.

- **Geographical Location and Diversity:** The closer that peers are to the querying
Figure 4.3: NTP Hierarchy
host, the less the likelihood of asymmetries within round-trip-delays for the query-
response transactions. This results in more stable performance. Furthermore, it is
also important that peers do not share significant common paths as this increases the
likelihood of asymmetries being present in all peers, resulting in undetected errors.

Although NTP performance levels of low single millisecond level synchronisation on
LANs and marginally higher on WANs have been reported in the literature [69], poor NTP
performance has also been reported [27]. In a survey of NTP performance carried out in
1999 [79], many highly inaccurate time servers were found. Undoubtedly many of these
problems were caused by poor design and maintenance of NTP subnets. In addition, the
cost of primary reference sources was prohibitive in the past. Such local reference clocks
have however become more cost-effective in recent years. This has largely been driven by
the falling cost of GPS receivers, specifically those designed to deliver microsecond level
precision timing services. Local reference clocks greatly strengthen an NTP subnet but are
not a substitute for peer diversity. For this reason, well designed NTP subnets using local
stratum 1 servers attached to precision GPS clocks are increasingly being used for delay
measurement.

In [80] such an architecture, labelled SATURNE is developed to measure one-way delays
in order to monitor adherence to Service Level Agreements (SLA). Although other delay
measurement architectures rely solely on GPS clocks at either end, such an approach
presents single points of failure (SPOF) and the integration of GPS with NTP facilitating
redundancy through stratum 1 peers presents a more robust alternative. In application
domains such as power system control where better than microsecond level synchronisation
is required for fault detection or phasor-based power measurement, the sole use of GPS
clocks is required but multiple redundancy is employed to protect against SPOF [81].

Operating System Dependencies

NTP was initially designed for the Unix environment and has thus largely been limited to
Unix and its variants. In recent years however, it has been ported to NT, 2000 and XP.
Its operation under Unix is well described in [82], [83] and [84]. Offsets (under normal
non-step change operation) are corrected for by gradual adjustments every tick using the
adjtime() function, which accumulate to the required extent. A fixed amortisation rate is
applied over the adjustment interval after which the default frequency resumes. This leads
to a sawtooth-like pattern in offset error. The following example simplifies the issue. If
the amortisation rate is 4 microseconds every tick (10 msec), this amounts to a rate of 400 ppm as shown in eqn. 4.2.

\[
Rate = \frac{\text{tickadj}}{\text{tick}} = +/ - \left(\frac{4}{10000}\right) = +/ - 400 \text{ppm}
\]  
(4.2)

If the clock skew is +100 ppm, then within each second, the sawtooth is described by a period of 0.25 sec over which the effective rate is \(100 - 400 = -300\) ppm and a +100 ppm rate over the remaining 0.75 sec.

More recent NTP developments have implemented NTP in the OS kernel via the \texttt{ntp_adjtime()} call which significantly removes the above sawtooth-like adjustments. Details can be found in [77] and [85].

The operation of NTP under Windows NT/2000/XP is less transparent. However the timer resolution limitation of 10 msec outlined in section 4.2 has been overcome by NTP through use of a real time thread that interpolates between ticks using the 8254 timer-chip cycle counter, introduced in section 4.1. As such the solution is similar to that of Unix variants in improving the system clock resolution though the latter uses the main CPU cycle counter.

### 4.4 NTP Performance Tests

Fig. 4.4 illustrates the Wide Area Network (WAN) testbed developed for evaluating NTP performance and more generally, the hybrid playout strategy. In this section, the focus is solely on NTP performance. Results of the hybrid playout performance are presented in chapter 6. The end-hosts operating as stratum 2 servers, one at the National University of Ireland, Galway (NUI,G) and the other at Dublin City University (DCU) were 220 km apart and connected by the Higher Education Authority network (www.heanet.ie). Both end hosts were running on Linux platforms. HEAnet is Ireland’s academic and research network and is linked to similar networks in Europe and the US. HEAnet is a well-provisioned network and considering its limited geographical size, internal network delays are usually well within the G.114 limit.

A stratum 1 NTP server was set up at NUI,G to strengthen the local NTP infrastructure. A Trimble Acutime 2000 GPS clock provided its reference source [86]. Six other stratum 1 NTP servers, located in the UK, France, Germany and Switzerland were included in the NUI,G stratum 1 server’s configuration file. These were chosen to best meet
the requirements for robustness, described above. The NTP configuration file on the two end-host PCs was identical and included the NUI,G stratum 1 server as well as the other 6 stratum 1 servers. Both were thus operating as stratum 2 clients. As such the closest stratum 1 server (and thus the closest reference clock) for the DCU stratum 2 server was at NUI,G.

The scenario described above is to some degree best-case in that the end hosts operating at stratum 2 have access to the NUI,G GPS-based stratum 1 server and six other stratum 1 servers. In reality, stratum 2 servers should peer with other stratum 2 servers as otherwise the limited number of stratum 1 servers will become overloaded, affecting their performance. The hierarchical nature of NTP also requires such a configuration. As such separate tests were repeated across the NUI,G LAN on local stratum 2 clients with more realistic configurations. Details are provided in [87], [88] and [73]. Fig. 4.5 illustrates the second testbed. Clients running under Linux and Windows 2000 platforms were included to evaluate OS dependencies. The configuration files of the stratum 2 clients included the local stratum 1 server, but all other servers were operating at stratum 2 and were located in Dublin, Ireland, the UK, Germany, Croatia and Slovenia.

In the context of NTP robustness issues described above, it is obvious from both Figures 4.4 and 4.5 that the NTP subnet within Ireland was weak. Despite careful selection
of servers for the various configuration files, all stratum 1 & 2 servers (other than NUI,G stratum 1 and TCD stratum 2 server respectively) were located overseas in the UK and mainland Europe. This presented relatively long network paths, and more critically, common network paths with the consequent potential for delay asymmetries. Nonetheless, as results indicate, performance was still more than satisfactory.

4.5 Results

Fig. 4.6 & 4.7 respectively outline the performance of the NUI,G and DCU client NTP hosts from Fig. 4.4 over a 36 hr period, during which the hybrid evaluation tests were carried out. The legend outlines which of the seven servers were candidates at each sample instant plus the clock offset of each of the candidates relative to the client clock. The NUI,G client clock offset relative to the stratum 1 servers remained within a $+/-1$ msec band for 97% of the time whereas the DCU client clock offset remained within a $+/-7.5$ msec band for 95% of the time. As expected, the NUI,G client remained tightly coupled to the NUI,G stratum 1 server, selecting it as a candidate 93% of the time. The DCU client selected the NUI,G stratum 1 server as a candidate 60% of the time, disregarding it
whenever the DCU-NUIG link became congested, relative to the other servers. The lack of a local stratum 1 server at DCU is thus reflected both in the higher range of offset values and the looser coupling relative to the NUI,G stratum 1 server and confirms the need for local reference sources. Overall however, within a VoIP context, the performance of both the NUI,G and DCU NTP clients was very satisfactory.

The more realistic NTP scenario described in Fig. 4.5 whereby the LAN based stratum 2 clients communicated with stratum 2 servers in addition to the local NUI,G stratum 1 server also performed more than adequately. Figures 4.8 and 4.9 illustrate the performance of the Linux and Windows 2000 stratum 2 clients respectively, over a 240 hr period. The legend outlines which of the seven servers were candidates at each sample instant plus the clock offset of each of the candidates relative to the client clock. The Linux client offset
Figure 4.7: Performance of the DCU NTP client
remained within a $+/−1$ msec band for 96% of the time whereas the Windows 2000 client clock offset remained within a $+/−5$ msec band for 98% of the time. As expected, both clients remained tightly coupled to the NUI, G stratum 1 server, selecting it as a candidate 99% of the time.

The performance of both clients in Figures 4.8 and 4.9 suggests some diurnal variations, though to a greater degree for the Windows machine. Figures 4.10 and 4.11 clearly illustrate these variations by removing the clutter and outlining the offset of each client to the local stratum 1 server. More thorough testing would be required to identify the main cause of such variations and explain the differing range of variation between Linux and Windows platforms.
Figure 4.9: Performance of Stratum 2 Windows 2000 Client
Figure 4.10: Diurnal Variation of Stratum 2 Linux Client
Figure 4.11: Diurnal Variation of Stratum 2 Windows 2000 Client
4.5.1 NTP Subnet Comparison

As a final exercise, a comparison was made between the performance of the NUI,G stratum 1 server and one of the six stratum 1 servers in its configuration file. The French server ntp-sop.inria.fr was chosen for comparison, principally because it was selected most frequently by the NUI,G server as a synchronisation candidate. In addition to its primary GPS source, its configuration file mostly lists peer servers in France, Germany and Switzerland. Due to robustness issues described above, such as diverse paths and low delay to peered servers, it was expected that the performance of the French server would be better than the NUI,G server. In fact the opposite was the case though the margin was small. Fig. 4.12 illustrates the relative offsets of the French server to its candidate peer servers over a 300 hr period. The range of offsets is clearly larger than that of the NUI,G server shown in Fig. 4.13 though largely remained within a +/- 1 msec band. On the other hand, an analysis of both the delay values and network paths from the French server to its peers revealed that delays were lower (due to geographical proximity) and common network paths were fewer than those of the NUI,G server. Furthermore Fig. 4.13 for the NUI,G server also indicates the presence of some delay asymmetry albeit minimal. This is to be expected considering the common network paths to its stratum 1 peers and highlights the relative weakness of the Irish NTP subnet, despite its good performance in tests.

One of the dangers of utilising NTP to synchronise clocks for delay measurement that is highlighted in the literature [71] is that NTP performance can be affected by the same network conditions that it is attempting to measure! However, the use of local reference clocks and careful NTP subnet design (to separate the NTP subnet as much as possible from the network being measured) can address such concerns.

Finally, the ongoing roll-out of ADSL (Asymmetric Digital Subscriber Line) and GPRS (General Packet Radio Service) in both the terrestrial and mobile telephony markets respectively will lead to always-on Internet connection capability. This, coupled with the falling cost of precision GPS clocks will allow for much more widespread and effective deployment of NTP to domestic Internet users.

4.6 Summary

In this chapter the problem of clock errors along with the underlying issues of oscillator accuracy and stability were reviewed. Furthermore, the uncertainties relating to accurate
Figure 4.12: Offsets Measured by ntp-sop.inria.fr to its Servers
Figure 4.13: Offsets Measured by NUI,G Stratum 1 to its Servers
delay measurement and the limitations of RTT as a one-way delay estimator were identified. In the context of the hybrid playout strategy, one-way delay measurement is critical though only to single msec levels. Despite its limitations, the use of RTT is also acceptable where general delay characteristics are required. NTP operation was briefly reviewed along with robustness issues that impact on performance. The main contribution of this chapter was to show that with careful NTP subnet design, addressing issues of redundancy, diversity and independence, single millisecond level synchronisation can be attained across WANs. The proximity of local stratum 1 servers is a critical factor and though OS platform and traffic patterns affect its performance, synchronisation levels can be achieved that more than adequately meet the requirements of the hybrid playout strategy.

Having established the credentials of NTP in the context of the hybrid playout strategy, the next chapter addresses remaining feasibility issues and describes the various approaches employed to assess its applicability.
Chapter 5

Hybrid Strategy Evaluation: 
Feasibility and Applicability

In section 3.6.3, the core objectives regarding hybrid playout strategy evaluation were outlined as follows:

1. To assess the technical feasibility of implementing the hybrid strategy in a WAN environment.

2. To assess its applicability over diverse Internet paths.

The previous chapter examined and addressed the requirement for synchronised time within the hybrid playout strategy. It concluded that with careful design, the necessary performance levels over WANs can be achieved thus satisfying a core element of objective 1 above. In this chapter the remaining issues regarding the technical feasibility of the hybrid strategy are addressed and the methodology for meeting objective 2 outlined.

This chapter firstly describes how the hybrid playout strategy can be implemented within a real VoIP application and how synchronised time can be integrated within the hybrid code. It then outlines in detail a number of approaches that were adopted in order to assess its applicability. As described in section 3.6.3, applicability was assessed by comparing performance with the conventional adaptive playout strategies, referred to as Alg. A and B.

The first approach utilised the same WAN testbed introduced in section 4.4 for NTP performance evaluation. The objective this time was to evaluate the performance of the hybrid playout strategy within a real VoIP application across the limited Irish academic
and research network. The second approach was simulation based whereby voice streams, delays and playout strategies were all simulated. Regarding simulated voice streams, a brief review of the relevant literature on voice modelling is undertaken. Simulated delays were based both on captured trace-data and general delay models. Regarding the latter, the literature on Internet delay and loss modelling is summarised focusing primarily on delay and from this simple delay models are developed. The third approach was emulation-based which is a combination of simulation and live testing. This was similar to the WAN testbed approach described above except that it used a divert-sockets based application within a Local Area Network (LAN) testbed to simulate both trace-driven and source-driven delays. It used actual voice streams and a real VoIP application.

Although the WAN testbed and emulator approaches lend themselves to subjective testing, the ITU-T E-Model was used to quantify the relative performance of the various playout strategies. The E-Model is thus described in some detail, outlining its limitations and a review of recent work aimed at improving its effectiveness and use within the Internet is carried out. A simplified yet conservative E-Model analysis is developed that is used to quantify relative performance of the hybrid playout strategy. Note that the issue of subjective testing versus E-Model analysis is revisited in section 6.6.1 where endpoint hardware and software complexities are discussed.

### 5.1 Hybrid Feasibility: Implementation Details

Fig. 4.4 describes the WAN testbed already introduced in section 4.4. The testbed approach was useful for a number of reasons:

- **Real VoIP Application**: It enabled the hybrid and alternative strategies to be implemented and tested within a real VoIP application, thus assessing their technical feasibility.

- **It utilised actual voice streams delivered across a real network.**

- **It tested the integration of NTP within the hybrid playout strategy.**

The hybrid playout strategy along with Alg. A and B were implemented by rewriting relevant sections of the open source VoIP application *ohphone*, available from the openh323 project [89]. *ohphone* by default implements a *fixed-size* playout strategy. Although this can be configured at the start of a session, it is then fixed for the duration of the call.
<table>
<thead>
<tr>
<th>Status</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synchronisation source</td>
<td>*</td>
</tr>
<tr>
<td>Candidate in final selection</td>
<td>+</td>
</tr>
<tr>
<td>Falseticker</td>
<td>X</td>
</tr>
<tr>
<td>Discarded</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 5.1: NTP Server : Status Symbols

As detailed in section 3.6.3, Alg. A was chosen as the adaptive algorithm within the hybrid playout strategy. In order to facilitate the changeover from adaptive to fixed-delay (and vica-versa if necessary) within the hybrid, the modified application runs both Alg. A and the fixed-delay strategies in parallel. Alg. A was configured with the default settings of [45] with $\alpha=0.998002$ and $\beta=4$.

A Matlab-based simulator was also developed to confirm the correct operation of the modified ohphone code. Essentially trace data was extracted during tests such as packet arrival time, RTP Timestamp and Marker-bit (to indicate a talkspurt) and this was used to reproduce playout times according to the different strategies for comparison with those generated by the ohphone code in realtime.

### 5.1.1 Synchronised Time

As detailed in the previous chapter, NTP can deliver the level of synchronism required for the hybrid playout strategy across WANs. This section describes how synchronised time was integrated within the hybrid playout code.

At the connection setup phase, each end of a VoIP session needs to know whether NTP is performing adequately at the remote end. The approach taken to this was to use the NTP ntpq utility. An ntpq query is sent from each host to the remote host and the response is analysed. This response details the list of servers or peers that the remote host is configured with and classifies them according to performance details. Table 5.1 outlines the more common classifications and the associated symbols.

As such each server/peer is labelled as either synchronisation source, candidate for final selection, falseticker or discarded. For each server/peer the response presents details of the offset, delay and jitter characteristics. From this response the extent of offset and
Jitter are extracted, and NTP performance is deemed to be satisfactory if values are below a threshold. Threshold values for Max. Offset and Max. Jitter were set to 5 msec and 0.01 respectively, based on experimentation. Although not done here, this query/response interaction could be built into the existing signalling mechanisms that exist within the H.323 or SIP protocols.

Once NTP operation and performance is verified, a mechanism is required to enable one-way delays to be determined within the hybrid code. The RTCP SR packets, introduced in section 2.2.2 provide such a mechanism. As outlined, SR packets include a mapping between the system clock time in NTP format (relating to when the SR packet was generated) and the associated RTP timestamp. On receipt of the first incoming SR packet, each receiver extracts the two timestamps and because system clocks are synchronised through NTP, it can determine the one-way delay for that SR packet. More importantly, because it now has the mapping between RTP and NTP timestamps on the remote host, it can determine each subsequent incoming RTP packet one-way delay (The complicating issue of clock skew is addressed in chapter 7). Note that it was necessary to modify the implementation of RTCP SR packet generation within the default ohphone code as the mapping between NTP and RTP timestamps was done loosely.

Essentially this means that once the first RTCP SR packet is received, each receiver can begin to accumulate delay information for each subsequent incoming RTP packet. These delays are stored in a circular array (of size 200) and once full, a function is called that returns the delay estimate to meet a target late loss. Details are as indicated in section 3.6.2.

5.2 Subjective Testing, MOS and the ITU-T E-Model

The Mean Opinion Score (MOS) has been used for many years to quantify speech transmission quality. ITU-T recommendation P.800 [90] outlines details relating to its use. It distinguishes between MOS₇ which denotes MOS derived from subjective listening-only tests and MOS₉, derived from conversational-quality tests. The latter are obviously more difficult to carry out but subject to certain limits, MOS₇ results can be used to predict MOS₉.

For MOS₇, P.800 outlines a number of test methods, the most common one being the Absolute Category Rating (ACR) which returns a 1 to 5 rating, ranging from Bad
<table>
<thead>
<tr>
<th>R-value range</th>
<th>90-100</th>
<th>80-90</th>
<th>70-80</th>
<th>60-70</th>
<th>0-60</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speech quality</td>
<td>Best</td>
<td>High</td>
<td>Medium</td>
<td>Low</td>
<td>Very Poor</td>
</tr>
</tbody>
</table>

Table 5.2: R values for Speech Quality Categories

<table>
<thead>
<tr>
<th>R-value (lower limit)</th>
<th>MOSCQE (lower limit)</th>
<th>User Satisfaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>4.34</td>
<td>Very satisfied</td>
</tr>
<tr>
<td>80</td>
<td>4.03</td>
<td>Satisfied</td>
</tr>
<tr>
<td>70</td>
<td>3.6</td>
<td>Some users satisfied</td>
</tr>
<tr>
<td>60</td>
<td>3.1</td>
<td>Many users dissatisfied</td>
</tr>
<tr>
<td>50</td>
<td>2.58</td>
<td>Nearly all users dissatisfied</td>
</tr>
</tbody>
</table>

Table 5.3: R-MOS User Satisfaction Ratings

to Excellent quality. Although still widely used and the preferred means of evaluating speech transmission quality, it requires significant resources to make the results statistically significant and is susceptible to errors unless implemented in a controlled manner. More generally, in [91] MOS limitations such as the shortness of clips used in tests and the extent to which test candidates are interested in the test material (unrelated short and simple sentences are used) are discussed. Consequently a range of tests that continuously measure physiological responses such as heartrate and skin response were developed. Furthermore, to improve user involvement, candidates were required to answer questions after tests.

The E-Model is an ITU-T standardised planning tool for predicting how the average user rates the voice quality of a phone call with known transmission parameters. It returns a transmission rating factor R in the range 0-100. Technical details are specified in recommendation G.107 [6]. Developed originally by the European Telecommunications Standards Institute (ETSI) as ETR250, it was first published by the ITU-T in 1998 and since then has undergone continuous development. Associated publications include G.108 [22] which outlines a planning guide to applying the E-Model, G.109 [92] which defines speech transmission quality categories shown in Table. 5.2 and G.113 [93] [94] which examines the impact of varying bitrate codecs and packet loss. Table. 5.3 taken from G.107 outlines how user satisfaction rating can be translated into equivalent MOS and R values. It shows that transmission quality expressed as a MOS should normally be greater than 4.03 though
anything above a 3.6 rating or R-value of 70 is acceptable. Note that the subscript CQE refers to the fact that the conversational quality MOS score is derived from the E-Model R-factor and thus is an estimate. A glossary of MOS terminology can be found in ITU-T recommendation P.800.1 [19].

In appendix B.1, the operation of the E-Model is described in detail. In recent years it has been widely adopted and adapted by the research community for VoIP QoS evaluation. The following sections review this research and outline the limitations of the E-Model before describing the approach taken to its use for hybrid playout strategy evaluation.

5.2.1 Application & Limitations of the E-Model

Since its development, the E-Model has been increasingly used as a tool for assessing voice quality within packet based networks such as the Internet. This section summarises and reviews much of this research and concludes by outlining how the E-Model is applied in this thesis for assessing the performance of the hybrid playout strategy.

The work of [95] detailed in appendix B.1 utilise it to assess the potential for delivering PSTN-like quality over packet networks. More recently [23] use it to examine the trade-off between packet interval, FEC delay and overall M2E delay. [23] carry out subjective testing and reverse map MOS scores to R-factor in order to determine the effect on \( I_e \) of packet interval size with FEC (i.e. without considering increased delay). Their subjective testing shows that increased packet interval improves the performance of FEC and thus intrinsic speech quality at the receiver. However, increased packet interval coupled with FEC increases M2E delay and this tradeoff is also examined. On a related note, they confirm the increased distortion caused by bursty rather than random packet loss and report general agreement with G.113 [93].

[43], referenced earlier in section 3.5, develop it significantly in order to quantify the combined effect of three variables, namely, encoding rate of codec, M2E delay and overall packet loss rate. Equation 5.1 outlines their approach:

\[
f_i(r, d, plr) = 94.2 - I_{di}(d) - I_{cc}(r) - I_{el}(plr)
\]  

(5.1)

In the above \( I_{cc}(r) \) refers to the impairment due to the encoding rate of the codec (at zero loss). They extrapolate from [93] and use curve-fitting techniques in order to apply it to any codec rate. \( I_{el}(plr) \) refers to distortion due to overall packet loss rate \( plr \) and they develop equation 5.2 to quantify this loss for all codecs.
\[ I_{el}(p|\tau) = 34.3 \ln(1 + 12.8 p|\tau) \]  

Note that this curve is based on random packet loss rather than bursty loss and suggests that loss impairment is independent of codec. From [12] however, it is clear that packet size plays an important role in determining the slope of the loss curves for a given codec. For example, the loss impairment slope for G.723 30 msec packets is significantly less severe than for 120 msec packets.

Finally \( I_{di}(d) \) refers to delay impairment caused by M2E delay \( d \). They develop different impairment delay functions \( I_{di} \), depending on the nature of the conversation and user tolerance of echo loss. They outline that the E-Model fails to capture differing interactivity requirements and that a single curve is too general. For example, they model a conversation with strong interactivity requirements by a curve that returns a low impairment up to the 150 msec level, increases rapidly between 150 and 300 msec and levels out thereafter (as conversation has essentially become half duplex).

[12] examine this issue also, introducing the concept of Tasks originally developed by [96]. Task type 1 refers to conversation with very high interactivity requirements whereas task type 6 refers to a very relaxed conversation. They outline associated \( I_{d} \) curves for each task type and show that applying task type 1 rather than type 6 curves can result in a reduction in MOS of 0.5 – 1. Specifically for an echo loss value of 51 dB in the delay range 0 – 200 msec, they apply an \( I_{d} \) slope ranging from 8 (Task 6) to 15 (Task 1) units per 100 msec delay. Note that from Fig. B.1 in appendix B.1, the \( I_{d} \) impairment for similar echo loss values was 5 units per 100 msec up to 150 msec and 12 – 13 units per 100 msec thereafter.

More generally, [12] confirm the validity of, and expand the E-Model curves by considering other related work such as [97], [98] and [99].

**E-Model Limitations**

In the context of evaluating real voice sessions, the E-Model is limited in that it was designed to deliver an instantaneous rating based on singular loss and delay figures. Developed by ETSI as a planning tool, it was never intended for use as an evaluation tool, particularly in *best-effort* packet based networks such as the Internet where both delay and loss can vary significantly during voice calls. [12] outline a number of E-Model limitations and suggest some solutions as follows:
• Rating of Entire Calls: To evaluate a whole conversation lasting several minutes during which conditions can vary, they outline a number of approaches such as dividing the call up into equal intervals or preferably, into periods of low and high loss rates. This then requires a definition of low and high loss rates. In this regard, a criticism of recommendation G.113 [93] reported in both [43] and [23] is that the curves for bursty versus random loss do not define the extent of burstiness. In [100], the concept of gaps and bursts are introduced to bring clarity to this issue. A gap is defined as a period of low packet loss in that a minimum of $g_{\min}$ consecutive packets are received between two consecutive losses. A burst refers to a period of high packet loss in that there are less than $g_{\min}$ packets between consecutive losses. [12] uses this to break a conversation up into gaps and bursts. However the value of $g_{\min}$ varies from 16 in [100] to 64 in [12].

• Perceived versus Instantaneous Quality and Recency: Some confusion exists in the literature regarding the distinction between these two concepts. The interpretation taken in this thesis is that of [100] whereby recency refers to the fact that users tend to rate overall quality based on the most recent experience. For example, [100] reports that moving a 15 second period of high loss to the beginning rather than the end of a 60 sec voice call results in a MOS increase of 0.68! The notion of perceived versus instantaneous quality refers on the other hand to the fact that step changes in quality are reflected in more gradual changes in user perception. Clark applies time constants of 5 and 15 sec, representing transitions from GOOD to BAD and BAD to GOOD respectively. [12] apply this analysis in their implementation of the E-Model.

This work on fine tuning the E-Model is aimed at accurately estimating how a user will rate packet loss, based on how loss varied during the session. No corresponding work exists for determining how variation in playout delay during a session affects E-Model ratings i.e. for delay impairment, a single value, representing the average playout delay during a session is generally applied in determining $I_d$. For conventional adaptive algorithms, playout delay will vary continuously during a voice session, according to network conditions. As detailed in section 3.1, this results in elongation and contraction of silence periods or indeed scaling of packets though the effects of such distortion are reported as insignificant. Furthermore in [53], the authors claim that packet scaling resulted in negligible degradation in quality though qualify this by noting that scaling occurred infrequently during tests. Certainly,
silence period distortion can affect the emphasis within speech and thus more research is required in this area.

From ongoing dealings with the ETSI STQ (Speech and Transmission Quality) group [101] and the ITU-T work on QoS [102], concerns have been raised by both at how the work detailed above has modified and applied the E-Model. The main concerns, some of which are outlined in the most recent version of the G.107 Recommendation [6], are as follows:

- The E-Model is a planning tool rather than an evaluation tool.

- Regarding the delay impairment, it was developed on the basis of constant delay during a session.

- Regarding packet loss, insufficient subjective tests have been carried out to justify some generalisations that have been adopted by researchers.

In particular, they are concerned with attempts to derive absolute quality ratings through use of the E-Model as has been done by [43], [12] and [23]. In addition, the development in [43] of a single loss-impairment curve to represent all codecs is contrary to G.113 findings and other reported work [103]. Finally although the work of [100] on developing a loss impairment value that reflects both recency and subjective versus instantaneous quality is somewhat intuitive, insufficient testing has been carried out to validate its implementation within the E-Model.

5.2.2 Application of the E-Model for Hybrid Playout Evaluation

For evaluating the hybrid strategy, the same simplified yet conservative E-Model analysis first described in [104] was applied. As such, the analysis was different from that of [12], [23] or [43] in that it was aimed at extracting approximate relative values rather than detailed absolute values. This approach has been validated by ETSI STQ.

In considering $I_d$, a 50 dB loss value was assumed which corresponds to reasonable echo cancellation. Consequently the $I_d$ impairment can modelled by 10 units per 100 msec up to 150 msec. Note that this is conservative in that [23] uses 3 units/100 msec up to 170 msec and the analysis in appendix B.1 suggested an impairment of 5 units per 100 msec up to 150 msec and 12 – 13 units per 100 msec above this threshold. With regard to [12] and the effect of varying Task type on $I_d$, the 10 units per 100 msec maps to a medium interactivity requirement.
Note that [12] specify that task type 1 with strong interactive requirements equates to reading random numbers quickly. In addition to the reduced tolerance of such a conversation type to delay, and in the context of per-talkspurt adaptive playout strategies, it may also result in shorter and thus more frequent talkspurts than for type 6 tasks. As such, the responsiveness of the playout strategy to changing network conditions would be improved. Extensive testing would be required to assess the impact of this on performance.

Regarding the $I_e$ factor, the G.711 codec was utilised for all tests which introduces no distortion, unlike lower bitrate codecs such as GSM or G.723. The impairment value was determined on the basis of average loss values using bursty rather than random loss curves and with PLC. From appendix Fig. B.2 $I_e$ for bursty G.711, packet loss in the range $0 - 5\%$ is represented by a dual slope curve. This was approximated as shown in Fig. 5.1 by 10 units per % above 3% and at 3 units per % below 3%. Note also that $I_e$ values from [93] are based on 10 msec G.711 packets rather than the 32 msec packets used in tests. As such the impairment is underestimated as [98] illustrates for G.723.

5.3 Hybrid Applicability

Although the feasibility issues such as NTP performance over WANs and implementing the hybrid playout strategy within an actual VoIP application have been addressed in sections 4.5 and 5.1 respectively, the key question remains: will the hybrid playout strategy deliver improved playout quality relative to conventional adaptive approaches over a diverse range of Internet paths? In the following sections the approach taken to answering this key question is outlined.

5.3.1 WAN Testbed Approach

The first approach evaluated the hybrid strategy over the WAN testbed of Fig. 4.4. As detailed in section 4.4, the end hosts were 220 km apart and linked by HEAnet, the Irish academic and research network. As outlined in section 3.6.3 Algorithms A & B were used for comparative analysis. Both were tuned with the default settings from [45] with $\alpha=0.998002$ and $\beta=4$ for Alg. A and $\alpha=0.875$ and $\beta=4$ for Alg. B. Spike threshold values were however chosen based on network conditions and were varied from $10 - 30$ msec. Pre-recorded voice streams were delivered across the testbed and played out according to whichever strategy (either the hybrid strategy or comparative strategies A/B) was running. As outlined at
Impairment = 3 units per % below 3% and 10 units per % above 3%

Based on 10 msec G.711 Bursty Loss

Figure 5.1: Simplified yet Conservative E-Model Analysis
the start of this chapter, the E-Model was utilised to quantify the relative performance of the hybrid playout strategy rather than subjective testing. It was therefore necessary to extract trace data for analysis via Matlab-based scripts. As outlined in section 5.1 the correct operation of the code was confirmed via a Matlab-based simulator. For comparing the relative performance of the various playout strategies, this trace data was then used to determine average delay and late loss values, required for E-Model analysis.

Tests were carried out over a period of one week during which over 100 tests were carried out. In order to capture diurnal variations, tests were conducted from 09:00 to 21:00 hrs.

5.3.2 Simulator Approach

Fig. 5.2 outlines the simulator approach whereby voice input, delays and playout strategies were all simulated. Delay values were taken either from trace-generated data or general delay models. Delay values were loaded into a file and read by the simulator which implemented the various playout approaches and generated associated playout times. This simulation-based approach facilitated rapid testing and comparative analysis, and is similar to the approach adopted by many of the researchers reviewed in section 3.3. As with the testbed approach, comparisons were made with adaptive approaches A & B.

In appendix B.2 details are provided on voice modelling and Voice Activity Detection (VAD) schemes. The following section outlines the approach taken to voice modelling within the simulator. Literature related to network loss and delay modelling is then reviewed and the approach to delay modelling within the simulator outlined. Details of trace-generated delays are also provided.

Voice Simulation within the Simulator

For the purposes of voice simulation, none of the mathematical models described in appendix B.2 were used and instead a trace driven approach was used. As such, the distribution of talkspurts generated by the ohphone VoIP application with default VAD settings using actual recorded voice streams was extracted and used. Regarding the effect of differing task types, the speech pattern used for tests reflected a medium scale interactivity requirement. This reflected the E-Model implementation detailed in section 5.2.2 where the delay impairment $I_d$ curve was based on a medium scale task type.
Figure 5.2: Simulator
In effect the talkspurt distributions were stored as a sequence of binary bits where 1 represents the start of a talkspurt and 0 a packet within a talkspurt. These distributions were then applied to the packet delay data in order to simulate the conditions that a receiver playout strategy would encounter.

5.3.3 Trace Delay Data

Round Trip Time (RTT) delay data was gathered from tests to a number of remote Internet hosts, in Ireland, UK, Germany and the US. The destinations chosen were as follows:

- University College Dublin, Ireland (www.ucd.ie)
- University College London (www.cs.ucl.ac.uk)
- Munich University of Technology (www.lkn.ei.tum)
- ICIR, Berkeley, California (www.icir.org)

As discussed in section 4.2.1, the ping utility is an adequate tool where rough estimates of one-way delay (i.e. half the Round-Trip-Time (RTT)) are required, rather than precise delay information. Furthermore section 4.2 outlined that the recent RFC 3432 [76] is aimed specifically at delay measurement for multimedia applications and recommends the use of periodic streams. On Linux platforms, ping can be configured to mimic actual data streams by specifying packet size and interval. As such the following command generates a data stream of 1000 packets each of 256 bytes with an inter-packet interval of 32 msec.

```
ping -c 1000 -s 256 -i 0.032 destination
```

The \(-s\) switch generates a packet size \(s\) excluding the ICMP packet header of 8 bytes, representing a total packet size of \(s + 8\) bytes. As the UDP header is also 8 bytes, the above mimics G.711 data packets with a 32 msec sample size, similar to that used in the WAN testbed of Fig. 4.4.

In all, over 360 tests were carried out to the above destinations. Tests lasted between 20 seconds and three minutes and were undertaken during the following time periods, 09:00-10:00, 10:00-11:00, 13:00-14:00, 17:00-18:00 and 22:00-23:00 GMT in order to capture any diurnal variations.
In [72], Paxson reviewed past studies of Internet characteristics and classified them according to their range and focus. *Exhaustive* studies attempt to analyse the properties of a significant fraction of the entire Internet. For example Mills used *ping* to carry out such a study in 1983 [105]. The growth in size and heterogeneity of the Internet since then but particularly since the early 1990s make such a study now almost impossible. Paxson describes his work as an *end-to-end* study whereby the focus is on how Internet paths, comprising multiple links perform from an end-user perspective. The work of [56] and [12] fall into this category though [72] is more extensive than the others. In this context the trace data gathered here can be classified as a *limited end-to-end* study. The objective was to capture the general delay characteristics of VoIP data streams over a limited number of diverse paths in order to assess the applicability of the hybrid playout strategy. In section 6.4 a number of recent Internet delay studies are identified and analysed in order to assess the hybrid’s wider applicability.

5.3.4 Internet Delay & Loss Modelling

In [8], Floyd and Paxson outline the pitfalls of relying solely on trace-driven data for simulation or modelling. They argue that actual measurements, regardless of the diversity of paths tested, cannot be taken as capturing the Internet’s full diversity. The network characteristics captured in a given trace are very much dependent on aggregate conditions within the network *at that time*. For example, the impact of rate adaption caused by congestion control within the TCP protocol can be very significant and this *shaping* of traffic will be evident in the overall measured characteristics.

In sections 5.3.1 and 5.3.3 above, the focus has been on evaluating the hybrid playout strategy using measured network conditions. The approaches differ in that the former uses a real VoIP application whereas the latter incorporates captured traces within a simulator. Both approaches are typical of those used within the research summarised in section 3.3. In order to address the limitations of trace-driven evaluation outlined by [8] general delay models were also developed for use within the simulator and emulator approaches. Internet loss and to a lesser degree delay modelling has attracted much research in recent years. In the following section this work is summarised and from this, the approach taken to delay modelling within the simulator is developed.
Challenges for Internet Simulation and Modelling

A broad perspective on general Internet modelling is given in [8]. A number of key challenges are identified as follows:

- The huge success of the Internet is due to the degree to which its core IP protocol unifies diverse networks. However, IP brings uniform connectivity but not uniform behaviour due to the increasing scale of network diversity.

- In addition to the diversity of networks within, and the continued growth of the Internet, a further challenge lies in the changing nature of Internet applications. For example, the growth in Internet Multimedia has led to a increased rate of UDP traffic in recent years. In [106] the extent to which the lack of UDP congestion control is effecting TCP traffic is examined.

- Although the Internet uses a limited set of protocols, significant differences in implementation details often exist which can have significant behavioural implications. Reference is made to [107] where over 400 TCP variants are reported.

- Infrastructural changes such as Router Scheduling policies or Varying QoS regimes will also greatly effect Internet characteristics. For web traffic, web caching policies and the deployment of Content Distribution Networks (CDN) can have a significant impact on overall Internet traffic patterns.

Despite the above challenges, [8] outline a number of behavioural patterns that have been shown to consistently apply. Such invariants include diurnal patterns in traffic and poisson-driven session arrivals. The latter refers to high level events such as FTP or Web downloads (rather than the lower level packet flows) and when adjusted on a hourly basis to reflect diurnal patterns have been shown to accurately reflect traffic patterns. For timescales ranging from hundreds of milliseconds to minutes, and thus not of relevance for diurnal patterns, the degree to which self-similar or fractal processes apply is discussed.

On a related note, Willinger and Paxson in [108] distinguish between the traditional POTS traffic and Internet traffic. They conclude that for many years, teletraffic theory based on Poisson’s law has served network designers well and resulted in expertly engineered circuit switched voice networks. They caution however that data traffic is much more variable, with connection times and datarates exhibiting extreme variability. This
contrasts with traditional voice-based POTS where call arrivals follow a Poisson distribution and call durations follow an exponential distribution. As such, applying Poisson based models to Internet traffic fails and instead they show by examining Internet traffic characteristics over increasing timescales that measured traffic shows no tendency to smooth out as is characteristic of voice networks. Note from above that Poisson does describe high level user sessions whereas the analysis of [108] is at the lower packet level. [108] also describes the use of self-similar or fractal analysis to capture Internet traffic characteristics. As outlined previously, RFC 2330 recommends poisson sampling for general delay measurement whereas in a multimedia context, the recent RFC 3432 is more appropriate.

**Actual Delay and Loss Models**

The work of [8] and [108] described above is very broad in approach. Other research has attempted to derive actual models and assess their appropriateness by comparisons with actual trace data. Such work has examined both loss and to a lesser degree delay modelling. The following summarises the findings of a number of such approaches, particularly those that are aimed at Internet multimedia applications.

- In [60] and [59], both network delay and loss are shown by Jiang and Schulrinne to exhibit burstiness or temporal dependency. Regarding delays, appendix A.1.1 outlined that delay spikes are frequently reported in Internet measurements and many adaptive algorithms have in-built measures to react appropriately. Such spikes are a manifestation of delay burstiness. In [60] and [59], the conditional CDF defined in equation 5.3 is identified as a useful metric to capture delay burstiness.

\[
f(t) = P(d_i > t|d_{i-1} >= t), l = 1, 2, 3, ...
\]

(5.3)

In equation 5.3, \( l \) is the lag. For \( l = 1 \), \( f(t) \) defines the probability that packet \( i \) will have a delay \( >= t \) given that packet \( i - 1 \) had a delay \( >= t \).

Temporal dependency of delay has significant implications for multimedia applications. If a packet’s delay is high, resulting in a late loss, the next packet is likely to suffer the same fate resulting in bursty late loss which in turn will have implications for FEC and PLC strategies. Finally [59] also examines the degree to which different queuing models reproduce the conditional CDF phenomenon.
Regarding loss, [60] outlines that losses are often preceded by high delays which in a VoIP environment can lead to a final loss pattern (FLP) of large loss bursts, comprising late loss and link loss. In both [60] and [59] the performance of the 2-state Markov model (Gilbert) is compared with the extended Gilbert model in predicting burst link packet loss characteristics and though the former is more easily implemented, the latter performs best in predicting the extent of burstiness. In [57], a 2-state Markov chain was found to be truly accurate in only 10 out of 38 segments and higher orders were necessary to accurately model remaining segments. The correlation timescale for loss events was estimated at 1 second, beyond which packet losses were considered independent.

On a related note, analysis in [56] suggests that in the absence of congested networks, there is little temporal dependency of loss. Regarding delay, [56] confirms delay burstiness and models delay using a single server, 2 input stream queue where one stream represents periodic media packets and the other represents all other Internet traffic. They report good correlation between analytical and experimental results.

- In [109], a two-state Markov model is used to simulate packet loss in a wireless environment whereas a Markov Modulated Poisson Process (MMPP) model is used to model delay in a wireless channel with retransmissions. Such an environment results in zero-loss but high delay-variance.

- In [43], loss is modelled by the simple two-state Markov model, delay is modelled by an identical distribution with a simple upper bound and loss and delay are considered to be mutually independent. The latter assumption is simplistic as shown by a number of studies including [72] [60] and [110].

- Finally, in the recent RFC 3357 [18], the issue of characterising loss burstiness is discussed and two metrics, loss distance and loss period are introduced. The former metric refers to the distance or number of packets between loss bursts whereas the latter refers to the number and extent of such bursts.

**Simple Delay Model**

Many of the above approaches examine both network loss and delay. The focus here was to derive simplistic yet useful delay models with which to evaluate the hybrid playout
strategy and as such the delay modelling aspect of the above research was the main focus. In evaluating the hybrid strategy, a number of the above sources were used.

Regarding the simulator, the main objective was to model temporal dependency or burstiness of delay reported in much of the literature in order to assess its effect on algorithm performance. A further objective was to assess the degree to which the proximity of baseline delays to the G.114 limit of 150 msec would effect performance.

In justifying the use of *bursty* delay models the conditional CDF approach was firstly applied to determine the actual extent of delay burstiness within the traces from section 5.3.3. Furthermore the impact of such delay burstiness on the final loss pattern was assessed using the Markov 2-state model to determine the degree to which it translated into late loss bursts.

A range of 2-state Markov delay models were chosen to model burstiness, an approach similar to [109] and more usually applied to loss modelling. Fig. 5.3 illustrates its application to delay modelling. Note that [109] used a Poisson process (with different mean arrival rates) to model interarrival times for the GOOD and BAD states in a wireless environment where delays vary greatly due to reliable delivery in a lossy environment. For wired environments this is unsuitable and thus GOOD and BAD states were modelled differently. The GOOD state represented periods of low delay variance whereas the BAD state represented periods of high delay variance.

The following summarises the most relevant characteristics of the 2-state Markov delay models developed:

- BAD State Probability: This represents the percentage of packets that are affected by high delay variance.

- Average BAD State Burst Length: This determines how the BAD state packets are distributed. Much of the literature on loss modelling have reported that single losses are the most frequent i.e. loss burst of 1. Where strong temporal dependency of delay is present, this will result in clusters of BAD state packets resulting in BAD delay bursts spanning more than one packet. Longer BAD bursts will be reflected in higher values for $P_{BB}$ from Fig. 5.3.

- BAD/GOOD State Jitter Level: The delay model developed used different ranges of jitter to differentiate between GOOD and BAD states. Essentially, a GOOD state
jitter metric was set (as a % of base delays) and a multiplier was applied to represent the BAD state.

An additional requirement was to ensure that out-of-order packets could not arise: in reality such events are largely due to route changes and occur infrequently and thus it was important to reproduce this. For example, no out-of-order packets were detected during the trace delays gathered from testing. Although this issue does not generally arise for delay modelling, it is an important issue for multimedia data whereby packets are generated periodically with short inter-packet intervals and thus introducing significant delay jitter can inadvertently result in significant out-of-order packets. The NISTNet emulator described in appendix F uses the jitter and mean delay settings to implement out-of-order packets.

5.3.5 Emulator Approach

In [111], Emulation is defined as a semi-synthetic environment for running real code in that it combines Simulation and Live Testing. Emulation thus enables real VoIP applications to be tested using diverse network conditions in a LAN environment. The LAN-based testbed shown in Fig. 5.4 comprised two machines situated on the same network. Real voice streams
were delivered across the LAN using a VoIP application modified to implement the various playout algorithms. The testbed also incorporated a mechanism for simulating delays and indeed other network characteristics. This mechanism was based on Linux Divert Sockets. A detailed description of its operation and its use in testing applications over a variety of network models is given in [112] and [109] respectively. Essentially it enables a user to intercept media packets at the socket level and divert them to a user process which reinjects them according to user specified rules of loss or/and delay. This approach facilitated hybrid evaluation across diverse network conditions which were reproducible and controlled. Emulator design details are as follows:

- Both end hosts were situated on the same LAN with time synchronisation provided via NTP. The NTP performance of both end-hosts was thus better than that of the remote host in Fig. 4.4 due to the proximity to the local stratum 1 server.

- As with the simulator approach, packet delays were loaded into a file which was read by the divert sockets application. The delay values were either taken from the trace data or generated by the 2-state Markov delay models.
• Pre-recorded voice streams were delivered across the testbed. As with the simulator, an ON/OFF speech pattern reflecting a medium scale interactivity requirement was used.

In effect, the emulator approach was similar to the simulator approach in that it used the same delay data. As such it was used to verify a subset of the simulator-derived results. Its use of a real VoIP application, incorporating synchronised time added weight to the simulator-derived results.

5.4 Summary

This chapter commenced by addressing remaining feasibility questions relating to the hybrid playout strategy. It described how synchronised time can be integrated into the hybrid mechanism and in turn how the latter can be implemented within a VoIP application. The remainder of the chapter outlined the various approaches taken to addressing the second core objective, namely: To assess how widely applicable is the hybrid playout strategy within the current Internet? In order to quantify the relative performance of the hybrid strategy, the operation, application and limitations of the ITU-T E-Model were described. A number of approaches (WAN testbed/Simulator & Emulator) to comparative testing of the hybrid with conventional adaptive playout strategies were then outlined. In doing so, the issues of voice and delay modelling required for the simulator and emulator approaches was addressed. In the next chapter detailed results are presented from each of the above approaches.
Chapter 6

Hybrid Strategy Evaluation: Applicability Results

Chapter 5 outlined the various approaches taken to assess the applicability of the hybrid playout strategy. In this chapter, detailed results are presented from each of the outlined approaches. It firstly examines relative performance within the Irish research network HEAnet. It then presents extensive results from the simulator approach using both trace-driven and model-driven delays. A subset of the simulator-derived results are then confirmed via the emulator approach.

More generally, the degree to which the hybrid playout strategy can deliver performance gains is dependent on Internet delay characteristics. In order to assess its wider applicability, a number of recent Internet delay studies are identified and analysed.

The chapter concludes by outlining a range of endpoint hardware and software issues that arose in both the testbed and emulation approaches. Such issues introduce a degree of uncertainty in VoIP application performance and in this context the decision to quantify relative performance of the various playout strategies through use of the E-Model rather than subjective testing is revisited. A detailed analysis of such issues is beyond the scope of this work and as such a high level approach is taken except in the case of clock skew which is of particular importance and which is dealt with in detail in chapter 7.

A summarised version of results can be found in [118].
6.1 WAN Testbed Results

As outlined in section 5.1, over 100 tests were carried out over a one week period. Fig. 6.1 outlines a single test result and compares the performance of the hybrid algorithm with that of Alg. A. In all but 3 of the 100 tests, the hybrid strategy switched from adaptive to fixed playout mode at the 1st opportunity (after 1st RTCP Sender Report is received and delay histogram is generated and analysed) and remained in fixed playout mode thereafter. As detailed in section 3.6, a key advantage of the hybrid strategy is that the fixed-delay playout point adapts to changing network conditions. This is evident from Fig. 6.1 where an increase in congestion levels after 90 seconds (which equates to approximately 3000 packets) results in the hybrid strategy recalculating a higher fixed playout value of 60 msec. The increased late losses resulting from the conventional Alg. A is also very evident from Fig. 6.1. As detailed in section 3.3, adaptive strategies such as A & B react too slowly to sudden periods of increased network jitter but then via the TCP-like formulae settle at excessive playout values. Note that Fig. 6.1 illustrates the performance of the hybrid before the tuning features introduced in section 3.6.2 were introduced. Instead, a fixed $W_f$ value of 0.33 was applied along with a fixed playout reevaluation interval of 1000 packets. The performance improvements resulting from hybrid-tuning features are outlined briefly in section 6.2.2.

Figures 6.2, 6.3 and 6.4 summarise the results for one of the 5 days and are representative of the overall tests. As illustrated in Figures 6.2 and 6.3, the hybrid strategy resulted in a late loss reduction of up to 4 percent whilst incurring an increase in end-to-end delay of 30-40 msec. Note that Alg. B, though reacting well to spikes, performed poorly, resulting in high late losses when exiting spike mode. This is consistent with the findings of [48] and [12] detailed in section 3.3. By applying the approximate yet conservative E-Model analysis from section 5.2.2 to Figures 6.2 and 6.3, Fig. 6.4 shows that the net effect of the hybrid algorithm was an R-factor improvement of between 2-25 relative to Alg. A with an average gain of approximately 10. Although the range of improvement relative to Alg. B was similar, the average gain was higher at 15. The extent of gain correlated well with network traffic diurnal variation, with greatest gains in late morning, immediately after lunch and early evening periods. This reflects the findings in chapter 2 that adaptive approaches can result in unnecessary late losses during periods of significant jitter. Note that no out-of-order packets and very little link packet loss were observed during the tests.
Performance of Hybrid vs Adaptive algorithms

Hybrid commences in adaptive mode

Hybrid changeover to fixed playout

Subsequent c/o to higher fixed playout

Adaptive playout

1st RTCP packet received => analysis of actual delays commences

Figure 6.1: Typical test result
Figure 6.2: Late Loss Reduction due to hybrid algorithm
Figure 6.3: Increased delay due to hybrid algorithm
Figure 6.4: R-factor improvement due to the hybrid algorithm

Although Fig. 6.1 shows a high variance in network delays over a specific test period, the longer term variation in delay over the testbed link was less dramatic. Fig 6.5 shows the delay seen by the DCU NTP client host whilst querying the NUI,G server over a 36 hr period, during the 5 days of tests. Congestion was not a serious problem on this link and the round trip delay (RTD) remained around 16 msec though results are based on 15 minutes sample intervals. Considering HEAnet’s limited geographic size and well provisioned links, such results were not that surprising. In fact, analysis of the delays to the other six stratum 1 NTP servers, none of which were located in Ireland, indicates that the round trip delay pattern was quite similar although the baseline delay ranged from 30-60 msec depending on location.
Figure 6.5: Delay to NUI,G NTP server seen from DCU
<table>
<thead>
<tr>
<th>Test Number</th>
<th>Time Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-6</td>
<td>17:00-18:00 hrs</td>
</tr>
<tr>
<td>7-12</td>
<td>22:00-23:00 hrs</td>
</tr>
<tr>
<td>13-18</td>
<td>09:00-10:00 hrs</td>
</tr>
<tr>
<td>19-24</td>
<td>10:00-11:00 hrs</td>
</tr>
<tr>
<td>25-30</td>
<td>13:00-14:00 hrs</td>
</tr>
</tbody>
</table>

Table 6.1: Test Numbers and Corresponding Period

6.2 Simulation Results

As outlined in section 5.3.2 and shown in Fig. 5.2, the delay values used within the simulator were based on either measured trace data or general derived delay models. The following sections deal with these separately. Many of the results are detailed and illustrated in appendix C.

6.2.1 Trace Data: Summary

In this section, a representative subset of the total trace delay data captured is summarised. As outlined in section 5.3.3 round trip times (RTT) to four remote servers located in Dublin (www.ucd.ie), London (www.cs.ucl.ac.uk), Munich (www.lkn.ei.tum) and Berkeley, Calif. (www.icir.org) were recorded. Tests to each were carried out in blocks of 30 and Table 6.1 outlines how test numbers correspond to times of day (GMT).

As detailed in section 5.3.3, ping was configured to mimic 32 msec G.711 packets. Sample sizes were set at 600, 1500 and 5400 packets which equates to test durations of approximately 20, 50 and 180 seconds. Over 360 tests in total were carried out (4 locations x 3 test sample sizes x 30 tests).

In Fig. 6.6, the mean packet delay to the remote servers for 50 sec tests over a single day is shown. There was little evidence of significant diurnal variation other than a slight increase to Dublin, London and Munich for Test Numbers 1-6 corresponding to the evening peak.

The use of average delay in the above figures is a poor indicator of variance however and thus Fig. 6.8 shows the delay histograms for three tests (carried out in time windows 17:00-18:00, 22:00-23:00, 10:00-11:00) to UCD. Details for UCL, LKN and ICIR are shown
Figure 6.6: Mean Delay to Remote Servers (60 sec tests)
Figure 6.7: Packet Loss Rate to Remote Servers (60 sec tests)
in Figures C.1, C.2 and C.3 respectively in appendix C.1. The greatest delay spread was evident in the 17:00-18:00 window which agrees with Fig. 6.6. As outlined in section 5.3.4, diurnal variation in network traffic is described by [8] as a useful invariant that can be confidently modelled.

Fig. 6.9 shows a single test result for unconditional, along with lag 1 and lag 2 conditional CDF for UCD. Results for the other locations (Figures C.4, C.5 and C.6) are detailed in appendix C.2. Although there was evidence of temporal dependence of delay values, the degree of burstiness was limited and those shown are all in the 17:00-18:00 window GMT where network congestion (within Europe at least) was highest. Considering the extent of testing, it seems that the underlying links were largely uncongested.

Fig. 6.7 show the corresponding link packet loss rate for the series of delay tests shown
Figure 6.9: Conditional/Unconditional CDF: UCD
in Fig. 6.6. Link loss rates were generally very low, though interestingly were highest for the most local link to UCD and lowest to the most remote server (US). This contrasts with the more intuitive results presented by [66] and [58] that international links suffer from higher loss rates. Analysis in [56] suggests that in the absence of congested networks, there is little temporal dependency of loss and thus little evidence of conditional CDF for delay would also be expected. The absence of significant lag 1 or 2 CDF in the captured trace data except during Test Numbers 1-6 confirms this finding.

### 6.2.2 Trace Data: Results

Figures 6.10 illustrates for UCD, the relative performance of the various playout strategies for a single test. The corresponding figures for UCL, LKN and ICIR respectively are shown in Figures C.7, C.8 and C.9 in appendix C.3. The spike threshold for Alg. B was set to 10 msec. As evident from these figures, the hybrid resulted in higher playout delays but significantly lower late losses. Note that in contrast to Fig. 6.1, the additional tuning features of the hybrid playout strategy detailed in section 3.6.2 were implemented and can clearly be seen. For example, one aspect of tuning relates to the frequency at which the playout mode is reevaluated. As detailed in section 3.6.2, this is set at \((150 – Pkt)/median\) * 500 with the result that reevaluation is carried out more frequently as delays approach the G.114 limit. From Fig. C.8 and C.9, the reevaluation interval for the samples shown is approximately 3000 and 1000 reflecting the respective baseline delays of 26 and 88 msec. As such, the legend specifies that the hybrid performance refers to the Modified Hybrid.

Further results of the self-tuning characteristics of the hybrid are evident from Fig. 6.11. This shows the operation of the hybrid with and without the self-tuning features. The particular trace was to UCD which was characterised by low fixed delays and relatively low jitter. This results in a decrease in playout delay (due to the variable \(W_J\)) with almost no increase in late loss along with a significant reduction in the frequency of playout mode re-evaluation (every 10000 versus every 500 packets). All hybrid test results derived from the simulator approach are based on the tuned version.

Figures 6.12, 6.13, 6.14 and 6.15 outline the performance of the hybrid strategy along with adaptive algorithms Alg. A and B over a block of 30 tests (i.e. a single day) to UCD, UCL, LKN and ICIR respectively. Results are for the 1500 packet tests but are representative of overall results. The 30 tests are grouped together into ten groups of
Figure 6.10: Sample Trace Performance: UCD
Figure 6.11: Hybrid Algorithm Performance: Self Tuning
Figure 6.12: Relative Algorithm Performance: UCD
Figure 6.13: Relative Algorithm Performance: UCL
Figure 6.14: Relative Algorithm Performance: LKN
Figure 6.15: Relative Algorithm Performance: ICIR
<table>
<thead>
<tr>
<th>Location</th>
<th>Avg. Redn. in Loss Rate</th>
<th>Avg. Delay Increase</th>
<th>R-factor Gain %</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCD</td>
<td>2.8</td>
<td>17.2</td>
<td>9.8</td>
</tr>
<tr>
<td>UCL</td>
<td>2.2</td>
<td>15</td>
<td>6</td>
</tr>
<tr>
<td>LKN</td>
<td>2.3</td>
<td>13.7</td>
<td>7.6</td>
</tr>
<tr>
<td>ICIR</td>
<td>1.5</td>
<td>4.2</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 6.2: Average Results: Hybrid vs Alg.A

<table>
<thead>
<tr>
<th>Location</th>
<th>Avg. Redn. in Loss Rate</th>
<th>Avg. Delay Increase</th>
<th>R-factor Gain %</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCD</td>
<td>4.7</td>
<td>18.5</td>
<td>21</td>
</tr>
<tr>
<td>UCL</td>
<td>4.5</td>
<td>15.5</td>
<td>22</td>
</tr>
<tr>
<td>LKN</td>
<td>4.4</td>
<td>14</td>
<td>22</td>
</tr>
<tr>
<td>ICIR</td>
<td>3.9</td>
<td>4.7</td>
<td>17</td>
</tr>
</tbody>
</table>

Table 6.3: Average Results: Hybrid vs Alg.B

three and the plots within each figure show the late loss rate for all three algorithms, the additional average delay introduced by the hybrid relative to Alg.A/B, and the net effect in R-factor.

Of the 120 tests shown here, the hybrid strategy returned a net gain in all but one. Results comparing the performance of the hybrid relative to Alg. A and B are summarised in Tables 6.2 and 6.3 respectively. Gains relative to Alg. A were highest for UCD and peaked at 50% though the average gain over all tests was 7%. Note that a short period of extreme delays was encountered during tests 19-21 to UCD resulting in very high loss rates for the hybrid (10%). Gains relative to Alg. B were more significant with an average of 20%. Gains were highest in the test range 1-6 which corresponded to periods of highest jitter. This agrees with the findings from the WAN testbed approach and again confirms the fact, outlined in section 3.3 that adaptive algorithms tend to track network delays too closely resulting in unnecessary late losses when significant delay jitter occurs. To reinforce this point, Fig. 6.14 also includes the standard deviation of delays (outliers removed) and shows a close correlation with the R-factor gain.

As outlined in section 5.2.2, the application of the E-Model in quantifying relative performance was somewhat conservative and thus R-factor gains are understated. The delay
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Hybrid</th>
<th>Alg. A</th>
<th>Alg. B</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUI,G to UCD(<a href="http://www.ucd.ie">www.ucd.ie</a>)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAD State ($\pi_1$)</td>
<td>0.10</td>
<td>3.2</td>
<td>6.1</td>
</tr>
<tr>
<td>$p$</td>
<td>0.23</td>
<td>2.8</td>
<td>5</td>
</tr>
<tr>
<td>$1-q$</td>
<td>75</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>NUI,G to UCL(<a href="http://www.cs.ucl.ac.uk)">www.cs.ucl.ac.uk)</a></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAD State ($\pi_1$)</td>
<td>0.03</td>
<td>2.5</td>
<td>4.9</td>
</tr>
<tr>
<td>$p$</td>
<td>0.03</td>
<td>2.3</td>
<td>4</td>
</tr>
<tr>
<td>$1-q$</td>
<td>25</td>
<td>9</td>
<td>15</td>
</tr>
<tr>
<td>NUI,G to LKN(<a href="http://www.lkn.ei.tum">www.lkn.ei.tum</a>)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAD State ($\pi_1$)</td>
<td>0.04</td>
<td>2.5</td>
<td>3.9</td>
</tr>
<tr>
<td>$p$</td>
<td>0.02</td>
<td>2.3</td>
<td>3.5</td>
</tr>
<tr>
<td>$1-q$</td>
<td>40</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>NUI,G to ICIR(<a href="http://www.icir.org">www.icir.org</a>)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAD State ($\pi_1$)</td>
<td>0.42</td>
<td>2.2</td>
<td>5.1</td>
</tr>
<tr>
<td>$p$</td>
<td>0.35</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>$1-q$</td>
<td>17</td>
<td>11</td>
<td>14</td>
</tr>
</tbody>
</table>

Table 6.4: 2 State Markov Parameters from Late Loss Patterns

Impairment of 10 units per 100 msec up to 150 msec was significantly higher than that applied by [23] or [95]. Regarding packet loss, the bursty loss curves taken from [12] was applied. Although the conditional cdf curves showed limited delay burstiness, the appropriateness or otherwise of applying bursty loss curves is best assessed by examining the late loss packet distribution resulting from the application of the various playout strategies to the actual delay data.

Table 6.4 summarises this 2-state Markov analysis for the four destinations. It shows the BAD (i.e. loss) state probability $\pi_1$, the GOOD to BAD state change probability ($p$) and the conditional loss probability $clp$ ($1 - q$). Results presented are based on Test Numbers 1-9 where the greatest delay burstiness (and late loss) was experienced.

Results for all destinations clearly show that the conditional loss probability ($1 - q$) is consistently higher than the BAD (loss) state probability $\pi_1$. As expected this is clearly the case for the hybrid due to its informed fixed playout but is also true for late loss patterns.
resulting from both algorithms A & B. This justifies the use of bursty loss rather than random loss curves.

Note also that the loss curve was based on G.711 10 msec packets whereas actual packet size was 32 msec and thus the impairment would be greater. Finally, all tests were based on the G.711 codec. The ITU-T G.113 recommendation outlines that loss impairments per % packet loss are codec dependent with G.711 performing best.

6.2.3 General Delay Models: Summary

An extensive range of 2-state Markov delay models were developed and the performance of the hybrid strategy was compared with both Alg. A & B. Models were loosely based on both the LKN and ICIR traces regarding base delays (23 msec for the LKN models and 82 msec for ICIR models) but characteristics were varied significantly to extensively test the hybrid playout strategy.

LKN-Based Models

Table 6.5 outlines the characteristics of the LKN-based 2-state Markov models developed.

The sample size for most models was set at 100000 to ensure that the effects of different model characteristics could be seen. Smaller sample sizes resulted in too much random variation which tended to mask any trends due to the differing model characteristics.

Appendix C.4.1 contains Figures showing sample traces for all of the LKN based models except G,H and H2. Note that for clarity the figures show a 5000 packet session extract trace rather than the full 100000 packets.

The rationale for the range of LKN-based models shown in Table 6.5 was as follows:

- Models A-D differed only in the extent of temporal dependency or delay burstiness within the BAD (high jitter) state. As such the BAD state probability (0.2) and the extent of jitter within both the GOOD and BAD states were constant. Model A was configured to give an average BAD run of 10 packets (10 x 30 msec = 300) whereas Model D was configured with no burstiness (average BAD run of 1 packet). The consequent delay trend differences are evident from Figures C.10 to C.13: the former shows high jitter delays clustered together whereas the latter has the high jitter value packets more evenly spread out. These models were thus designed to assess the extent to which delay burstiness impacts on relative performance.
<table>
<thead>
<tr>
<th>Model</th>
<th>Number of Packets</th>
<th>Average BAD Burst (msec)</th>
<th>BAD State Probability</th>
<th>Jitter %</th>
<th>BAD State Multiplier</th>
<th>Spike Threshold (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>100000</td>
<td>300</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>100000</td>
<td>150</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>C</td>
<td>100000</td>
<td>90</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>D</td>
<td>100000</td>
<td>30</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>E</td>
<td>100000</td>
<td>300</td>
<td>0.2</td>
<td>40</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>F</td>
<td>5000</td>
<td>150</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>G</td>
<td>20000</td>
<td>300</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>H</td>
<td>20000</td>
<td>300</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>H2</td>
<td>20000</td>
<td>300</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>I</td>
<td>100000</td>
<td>300</td>
<td>0.2</td>
<td>5</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>J</td>
<td>100000</td>
<td>300</td>
<td>0.01</td>
<td>5</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>K</td>
<td>100000</td>
<td>300</td>
<td>0.4</td>
<td>20</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>L</td>
<td>100000</td>
<td>300</td>
<td>0.2</td>
<td>20</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>M</td>
<td>100000</td>
<td>60</td>
<td>0.05</td>
<td>20</td>
<td>3</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 6.5: LKN-based Delay Models
• Model E was designed to assess the impact of higher absolute jitter values present in the default (i.e. GOOD) state and thus (via the multiplier) also in the BAD state. As such Fig. C.14 shows much higher average jitter values than Fig. C.10.

• Models G, H and H2 were run for 20000 packets rather than 100000. Furthermore because the simulator consisted of a number of separate Matlab modules, it was possible to use the same delay data generated by the delay-generation module and thus the only variable was the spike threshold setting for Alg. B which was implemented in the algorithm-simulation module. They were thus designed to assess the impact of this setting on the performance of Alg. B.

Finally model F had similar settings to model H but had a sample size of 5000 rather than 20000. It was included simply to assess the impact of a relatively small sample size.

• Model I and J were designed to assess the impact of high (0.2) and very low (0.01) BAD state probabilities respectively with a low default jitter setting of 5%. In particular Fig. C.17 shows very little delay variation and is drawn with a smaller y-axis range.

• Models K was designed to assess the impact of very high BAD state probabilities whereas model L was designed to assess the impact of lower BAD state probability but very high absolute jitter values in the BAD state (via multiplier). Fig. C.18 thus shows more frequent but lower bursts for the BAD state than Fig. C.19.

• Finally Model M was designed to assess the impact of low burstiness (average 2 packets equating to 60 msec) and medium BAD state probability (0.05). Compared to Fig. C.13, Fig. C.20 thus shows much more infrequent bursts but the absolute value of jitter within bursts is similar.

ICIR-Based Models

Table 6.6 outlines the characteristics of the ICIR-based 2-state Markov models developed. Appendix C.4.2 contains figures with sample traces for all of the above ICIR based models. As with the LKN models, the figures show a 5000 packet session trace rather than the full 100000 packet trace.
<table>
<thead>
<tr>
<th>Model</th>
<th>Number of Packets</th>
<th>Average BAD Burst (msec)</th>
<th>BAD State Probability</th>
<th>Jitter %</th>
<th>BAD State Multiplier</th>
<th>Spike Threshold (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>100000</td>
<td>90</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>100000</td>
<td>90</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>C</td>
<td>100000</td>
<td>90</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>30</td>
</tr>
<tr>
<td>D</td>
<td>100000</td>
<td>30</td>
<td>0.2</td>
<td>20</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>E</td>
<td>100000</td>
<td>90</td>
<td>0.2</td>
<td>10</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>F</td>
<td>100000</td>
<td>90</td>
<td>0.05</td>
<td>20</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>G</td>
<td>100000</td>
<td>90</td>
<td>0.05</td>
<td>20</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>H</td>
<td>100000</td>
<td>90</td>
<td>0.05</td>
<td>5</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 6.6: ICIR-based Delay Models

A major difference between the models was the baseline network delays which were drawn from the actual traces (23 msec for the LKN models and 82 msec for the ICIR models). ICIR-based models were chosen for this reason as they facilitated relative testing of the various playout algorithms with delays much closer to the G.114 150 msec limit. As such the range of, and rationale for the various ICIR-based models developed was somewhat different:

- Models A-C were designed to assess the impact of the different spike threshold settings for Alg. B.
- Model D was similar to B except for a lower average BAD burst length (1 rather than 3 packets) and was thus designed to assess the impact of lower burstiness for a given BAD state probability (0.2). Fig. C.24 clearly shows that the high delay spikes are more evenly spread out than in Fig. C.22 where high delay spikes are more clustered.
- Model E was similar to A except for lower default absolute jitter values (compare Fig. C.25 with Fig. C.21) and was designed to assess the impact of this on performance given the proximity to the G.114 limit.
- Models F, G and H all had low BAD state probability of 0.05. F and G were similar other than having different BAD state multipliers (compare Fig. C.27 with Fig. C.26) whereas model H had a low default jitter level of 5% but a high multiplier.

120
### Table 6.7: Delay Model Applied to Actual Traces

<table>
<thead>
<tr>
<th>Destination</th>
<th>Filter No.</th>
<th>Jitter %</th>
<th>BAD State Multiplier</th>
<th>BAD State Probability</th>
<th>Average BAD Burst (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LKN</td>
<td>1</td>
<td>40</td>
<td>3</td>
<td>0.0023</td>
<td>60</td>
</tr>
<tr>
<td>LKN</td>
<td>2</td>
<td>5</td>
<td>3</td>
<td>0.32</td>
<td>90</td>
</tr>
<tr>
<td>ICIR</td>
<td>1</td>
<td>20</td>
<td>3</td>
<td>0.0006</td>
<td>90</td>
</tr>
<tr>
<td>ICIR</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>0.064</td>
<td>60</td>
</tr>
<tr>
<td>ICIR</td>
<td>3</td>
<td>10</td>
<td>5</td>
<td>0.009</td>
<td>60</td>
</tr>
</tbody>
</table>

In general, the contrast between sample measured traces to LKN and ICIR, Figures C.8 and C.9 respectively, and the corresponding model-generated traces (Figures C.10 to C.20 for LKN, and Figures C.21 to C.28) clearly illustrate the extent to which most of the delay models generated conditions significantly worse than those actually measured. In terms of overall jitter levels, LKN model M (Fig. C.20) and ICIR model H (Fig. C.28) resemble most the actual sample traces.

By applying the GOOD and BAD state jitter characteristics (i.e. default jitter and BAD State Multiplier) from some of the above models to the actual LKN and ICIR traces, the BAD state probability and average BAD run-length can be determined and thus more direct comparisons can be made. Table 6.7 outlines the results from applying various model characteristics (Filters) to the actual traces. Results are based on Test Numbers 1-6 where jitter was highest.

As evident from Table 6.7, by defining the BAD state by very high absolute levels of jitter eg. LKN Filter 1 and ICIR Filter 1, very low BAD state probabilities were returned, 0.0023 and 0.0006 for LKN and ICIR respectively. In contrast the BAD state probability set within the equivalent models was much higher than this (generally 20%), reflecting the more severe conditions being modelled.

On the other hand, the LKN Filter 2 and ICIR Filter 2 settings are similar to models M and H respectively, and so are the BAD state probabilities.
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>45.3</td>
<td>0</td>
<td>33.9</td>
<td>4.2</td>
<td>33</td>
<td>6.0</td>
<td>19</td>
</tr>
<tr>
<td>B</td>
<td>45.5</td>
<td>0</td>
<td>34</td>
<td>4.1</td>
<td>33.2</td>
<td>6.1</td>
<td>18</td>
</tr>
<tr>
<td>C</td>
<td>46.6</td>
<td>0</td>
<td>34</td>
<td>4.1</td>
<td>33.3</td>
<td>5.6</td>
<td>19</td>
</tr>
<tr>
<td>D</td>
<td>46.5</td>
<td>0</td>
<td>34</td>
<td>4.1</td>
<td>34</td>
<td>5.2</td>
<td>19</td>
</tr>
<tr>
<td>E</td>
<td>60</td>
<td>0.25</td>
<td>45</td>
<td>4.1</td>
<td>42</td>
<td>7.2</td>
<td>18</td>
</tr>
<tr>
<td>F</td>
<td>46</td>
<td>0</td>
<td>33</td>
<td>6.7</td>
<td>33</td>
<td>8.3</td>
<td>32</td>
</tr>
<tr>
<td>G</td>
<td>43</td>
<td>0</td>
<td>34</td>
<td>4.4</td>
<td>33</td>
<td>6.3</td>
<td>22</td>
</tr>
<tr>
<td>H</td>
<td>43</td>
<td>0</td>
<td>34</td>
<td>4.4</td>
<td>33</td>
<td>6.3</td>
<td>22</td>
</tr>
<tr>
<td>H2</td>
<td>43</td>
<td>0</td>
<td>33</td>
<td>4.4</td>
<td>32</td>
<td>6.4</td>
<td>22</td>
</tr>
<tr>
<td>I</td>
<td>39</td>
<td>0</td>
<td>26</td>
<td>4.1</td>
<td>25.4</td>
<td>6.4</td>
<td>19</td>
</tr>
<tr>
<td>J</td>
<td>38.5</td>
<td>0</td>
<td>25</td>
<td>0.4</td>
<td>25</td>
<td>0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>K</td>
<td>48</td>
<td>0</td>
<td>38</td>
<td>1.4</td>
<td>36</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>L</td>
<td>55</td>
<td>0.22</td>
<td>39</td>
<td>5.8</td>
<td>36</td>
<td>9.5</td>
<td>29</td>
</tr>
<tr>
<td>M</td>
<td>43</td>
<td>0</td>
<td>31</td>
<td>2.3</td>
<td>30</td>
<td>2.3</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 6.8: LKN-based Delay Model Results

### 6.2.4 General Delay Models: Results

**LKN-Based Models**

Results from the LKN-based models from Table 6.5 are summarised in Table 6.8. This shows the average delay and loss for each of the hybrid playout strategy, Alg. A, Alg. B as well as the R-factor gain of the hybrid relative to both Alg. A and B.

The main points arising from the above results are as follows:

- The hybrid playout strategy outperformed both Alg. A and B for all models, except model J. Model J was based on very low BAD state probability (0.01) and very low default levels of jitter. Other than model J and M where their performance was similar, Alg. A outperformed Alg. B.

- For models A-D, the impact of varying the average BAD state run length (from 10 packets to 1 packet) was insignificant, for a given BAD state probability. The E-
Model R factor is based on average delay and loss values during a session and thus the very obvious differences in delay patterns from model A to D (See Figures C.10, C.11, C.12 and C.13) are not reflected in overall R-factor values. The performance of B improved slightly as BAD state run length decreased. Note also that both the standard deviation and range of Alg. B playout delay reduced as BAD run length reduced which would have an impact on required buffer size.

- The higher default jitter levels of model E resulted in little change in Alg. A performance but a deterioration in Alg. B performance. Relative to model A, Alg. A suffered from no additional loss despite the increased default jitter levels. This was mainly due to the higher average playout delay and reflects the impact of the TCP-like formula for determining playout that over-reacts to sustained jitter levels as outlined in section 3.4.1.

- Results from models G, H and H2 showed that the impact of spike threshold setting for Alg. B had little effect on its overall performance. Shown in parenthesis alongside the R-factor gain is the number of spikes detected. This increased from 0 with a spike threshold setting of 20 msec to 8 with a setting of 10 msec and then to 242 with a threshold of 5 msec. The effect on late packet loss suggests that as the spike threshold reduces, resulting in more spikes, the late loss rate marginally increases! As suggested in section A.1.1, this is probably due to the tuning parameters of Alg. B. Note also that although the characteristics of model E were similar to model H, the sample size was a much reduced 5000 and results were quite different. At this size, repeated tests returned significantly different results due to the random variations in the delay generation model and confirms the need for large sample sizes.

- Both models I and J had low levels of default jitter but differed in the overall BAD state probability. Comparing results of model A with I, it is clear that with similar overall BAD state probabilities and average run length, the late loss rates of both Alg. A and B were somewhat similar despite the much lower absolute value of jitter for model I. In fact models A, E and I had default jitter levels of 20, 40 and 5 % and yet the loss rates of Alg. A and to a lesser extent B did not vary significantly.

On the other hand, model J had both very low absolute jitter values and very low BAD state probability resulting in very low loss rates for both Alg. A and B and thus a marginal loss in R-factor.
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>122</td>
<td>3.2</td>
<td>121</td>
<td>4.2</td>
<td>115</td>
<td>11.2</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>122</td>
<td>3.2</td>
<td>121</td>
<td>4.2</td>
<td>118</td>
<td>6.9</td>
<td>10</td>
</tr>
<tr>
<td>C</td>
<td>122</td>
<td>3.2</td>
<td>121</td>
<td>4.2</td>
<td>120</td>
<td>5.6</td>
<td>10</td>
</tr>
<tr>
<td>D</td>
<td>121</td>
<td>4</td>
<td>121</td>
<td>4.1</td>
<td>121</td>
<td>6.6</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>105</td>
<td>1.3</td>
<td>102</td>
<td>4.1</td>
<td>101</td>
<td>6.7</td>
<td>16</td>
</tr>
<tr>
<td>F</td>
<td>110</td>
<td>2.1</td>
<td>110</td>
<td>2.2</td>
<td>108</td>
<td>4.6</td>
<td>0.5</td>
</tr>
<tr>
<td>G</td>
<td>120</td>
<td>2.8</td>
<td>115</td>
<td>3.5</td>
<td>113</td>
<td>5.9</td>
<td>5</td>
</tr>
<tr>
<td>H</td>
<td>96</td>
<td>1.6</td>
<td>90</td>
<td>3</td>
<td>89</td>
<td>3.3</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 6.9: ICIR-based Delay Model Results

- Model K threw up some interesting results. Despite a much higher BAD state probability, albeit at the same absolute jitter values of model A, Alg. A performed significantly better than in model A. Comparing Fig. C.18 with C.10, it is clear that Alg. A had a higher average playout (37.9 for model K versus 33.9 for model A from Table 6.8) resulting in much lower late losses. There was no similar improvement in Alg. B.

- Model L produced the highest gains for the hybrid relative to both Alg. A and B. This model generated the highest absolute jitter values for the BAD state due to the high multiplier value. Alg. A was unable to cope with such sudden changes resulting in high late losses.

- Model M had a medium BAD state probability of 0.05, higher than model J but significantly less than the other models. It also had a higher default jitter level than model J and thus the BAD state spikes were higher resulting in higher loss.

**ICIR-Based Models**

Results from the ICIR-based models from Table 6.6 are summarised in Table 6.9. As with Table 6.8 it shows the average delay and loss for each of the hybrid playout strategy, Alg. A, Alg. B as well as the R-factor gain of the hybrid relative to both Alg. A and B.

The main points arising from the above results are as follows:
• The hybrid outperformed Alg. A and B for all models. In contrast to the LKN models, the hybrid playout strategy suffered from significant late loss, due to the proximity of the base delays to the G.114 limit. Note that the maximum playout delay of the hybrid was increased slightly to 135 msec from normal hybrid setting of 120 msec (i.e. G.114 limit minus packetisation delay of 30) to ensure that the hybrid stayed in fixed mode under the more severe model conditions. Due to the high fixed delay playout, the delay penalty due to the hybrid was minimal for the same reason. Overall, the extent of R-factor gain relative to both A and B was less than for the LKN models. Alg. B performed worst in all tests.

• As with the LKN models, the impact of spike threshold settings on overall performance was not that significant. The number of spikes detected is shown in parenthesis alongside the R-factor gain for models B and C and decreased from 1538 to 351 for a 10 msec increase in threshold. This difference can also be seen from Figures C.22 and C.23. The late loss rate for Alg. B actually increased as the spike threshold was reduced which confirms the problems relating to the tuning parameters (α and β) of Alg. B.

• Hybrid gains for model D were significantly less than for model B due to the higher hybrid late losses. The lower average BAD run length of model D resulted in a more even spread of high jitter as evident from Fig. C.24. As detailed in section 3.6.2, the hybrid playout strategy normally reevaluates playout mode at certain intervals determined by the average delay but a mechanism is also used whereby the late loss rate is continuously monitored (using a circular array) and if it exceeds a threshold (set to 2% for the simulator), the playout mode is also reevaluated. Where high delay jitter is more evenly spread out such as for model D, the latter mechanism is less likely to be invoked. In particular model B invoked this mechanism 200 times whereas model D invoked it 140 times. This distinction is evident by comparing Figures C.22 and C.24. In contrast, note that for the LKN-based models, BAD run lengths were shown to have no significant impact on performance. This was due to the fact that loss levels in hybrid mode were extremely low for all tests and thus the 2% threshold was never reached.

• Hybrid gains for model E were higher than for model A due to the lower default jitter settings. As with the LKN models A and E, lowering the default jitter did not
significantly impact on the performance of Alg. A but did result in lower late losses for the hybrid.

- Models F, G and H had low BAD state probability of 0.05 but resulted in marginal gains for the hybrid over Alg. A. Gains were higher for model G and H due to the higher BAD state multiplier which resulted in greater step changes in jitter within the BAD state. As with the LKN models, the BAD state multiplier had a greater impact than the default jitter levels.

Note also that from section 6.2.3, LKN model M and ICIR model H were shown to most resemble the actual LKN and ICIR measured traces. The R-factor gains for these models from Tables 6.8 and 6.9 also compare well with the average reported gains from section 6.2.2.

Finally, the effects of the self-tuning features within the hybrid are well illustrated by Figures. 6.16 and 6.17. Due to the higher baseline delays for the ICIR model, the reevaluation of playout mode is carried out more frequently in Fig. 6.17, with intervals varying from 500 to 2000 packets as opposed to a more steady interval of 3000 packets for Fig. 6.16.

6.3 Emulation

As outlined in section 5.3.5, the benefit of an emulation approach is that it facilitates playout evaluation using a real VoIP application across a diverse range of simulated networks in a controlled LAN environment. The WAN testbed approach in section 5.1 confirmed the feasibility of implementing the hybrid strategy within a real VoIP application and its applicability over the limited HEAnet network. The simulator approach on the other hand facilitated rapid testing of the different playout approaches under real and modelled network delay conditions. Regarding the latter, most of the delay models developed were tested with 100,000 packet sessions, equivalent to 1-1.5 hrs of speech if tested in a real application environment. Simulations are nonetheless abstractions of reality and thus the emulation approach was useful in that both the voice streams and VoIP applications are real.

Although both the testbed and more particularly the emulator approach facilitate subjective testing, performance results were based solely on the E-Model. The E-Model com-
Figure 6.16: LKN: Bursty Delay Model
ICIR: Performance over Simulated Bursty Network

Figure 6.17: ICIR: Bursty Delay Model
lements the rapid testing simulation approach and avoids the significant resource requirement of subjective testing.

Regarding the divert sockets approach to the emulator, an issue arose relating to timer granularity (previously introduced in section 4.2) and in particular, the implementation of the per-packet delays. The approach taken was via the `select()` function. This function can in theory implement microsecond level delays but in practice is limited by the OS process timeslice. For most Windows and many Unix variants this is limited to 10 msec which in turn is based on the underlying timer granularity. As such, the initial Linux based emulator suffered from a 10 msec granularity in its implementation of delays. This limitation was overcome by reprogramming the software clock (see section 4.1) to interrupt at a higher rate. Within the Linux OS, this can be done by altering the default frequency from 100 Hz to 1000 Hz and recomiling the kernel. This results in the OS timeslice being reset from 10 to 1 msec introducing an order-of-magnitude improvement in the performance of `select()`.

Overall the emulator performed well at a high level and was easily programmed to correctly filter out RTP data packets (eg. from RTCP packets) and read/implement packet delays from a file. As with the simulator approach, delay values were derived either from the trace data or general delay models. The main objective of the emulator approach was to confirm results derived from the simulator approach and thus a subset of the simulator tests was used. Results correlated very well with the R-factor performance gains reported from the simulator approach.

On a more general note, building a delay emulator on a non-RealTime OS platform such as Linux exposes it to OS scheduling uncertainties which can be significant if the system is heavily loaded. Furthermore, the issues relating to timer granularity encountered above illustrate the advantages of the simulator approach in that it avoids any complicating implementation issues. On the other hand, such issues have to be addressed for real applications. In this regard, the use of both the WAN testbed and emulator approaches uncovered a range of issues relating to endpoint hardware/software that can impact on delivered speech quality. As such, subjective testing is also susceptible to endpoint issues that are largely avoided through use of the E-Model. The extent and impact of endpoint issues both on VoIP applications and more specifically on subjective testing is discussed in greater detail in section 6.6.1.

Finally, the divert sockets emulator, though adequate for confirming simulator-derived
results is somewhat limited in functionality. In appendix F, the NISTNet emulator [111] is described and compared with the divert sockets approach.

6.4 Internet Delay Studies

The results so far confirm the applicability of the hybrid playout strategy via a combination of WAN testbed, simulator and emulator approaches. The WAN testbed approach evaluates the hybrid in the geographically limited Irish research network. For this reason, the simulator approach was used to test the wider applicability of the hybrid across diverse network paths, using a combination of trace-driven and general delay models. The emulator approach was used to confirm a subset of the simulator-derived results.

As discussed in section 3.3, adaptive approaches are most useful when endhosts have no knowledge of actual M2E delays or when actual delays are known but are well above the G.114 limit. Adaptive approaches track network conditions which though minimising delays, will result in late packet losses that is often unnecessary, given actual delays. The final approach taken therefore was to identify and analyse recent studies on Internet delay characteristics. The objective was not to evaluate the relative performance of the playout strategies from such studies but rather to examine both delay and jitter characteristics and thus the extent to which the hybrid playout strategy could effectively be applied.

6.4.1 Importance of Current Data

As outlined by a number of studies including [108], [55] and [8], Internet characteristics are continuously evolving. As such, in choosing Internet studies with which to assess the wider applicability of the hybrid playout strategy, it was critical that such studies were current and representative. For much of the research presented in section 3.3, the relative performance of the different strategies was tested using limited Internet paths or current (at that time) trace data. Much of that trace data is almost a decade old and thus not representative of current characteristics. Similarly, other studies such as Bolot [56] (1993) and Paxson [55] (1994/95) [72] (1996) are historical in Internet terms. Of these, the latter work of Paxson is the most extensive reporting on 37 sites and 1,000 Internet paths. He also captured two datasets spaced a year apart (1994/95) and noted significant differences which reinforces the importance of current data. Finally, many studies such as [66] focus
on loss more than delay characteristics whereas the primary focus of this thesis is on the latter.

Considering the above, the following Internet studies were identified (in chronological order) for analysis:

- Paxson’s work referenced above (1994/95). Approximately 20,000 TCP file transfers were carried out between 35/37 sites in nine countries. Paxson’s objectives were two-fold: a study of end-to-end routing behaviour (using 37 sites) and of packet dynamics (using 35 sites) such as bottleneck bandwidth, packet loss and delay. Although he used unsynchronised clocks, he devised a series of algorithms for estimating the one-way delay estimates of the TCP data flows and the corresponding reverse ACK flows. The focus here is solely on the delay measurement aspect of his work.

- The work of Tobagi et al. [12] referenced previously in sections 3.4.1 and 5.2.1. This work carried out extensive testing of forty three (43) US backbone paths linking five (5) cities and was carried out over a period of 2.5 days in 2001. Probes of 50 bytes each were sent every 10 msec to simulate voice data and delay/loss characteristics were measured. GPS clocks were used at each site to synchronise clocks and provide precise one-way delay measurements.

- More recent work by Roychoudhuri et al. [103] that studied delay/jitter/loss/out-of-order characteristics of fourteen (14) Internet paths (US & International). Tests were carried out over a period of one year in 2001/2002 with test durations of up to 2 hours. Each experiment involved a pre-recorded 5 minute voice streams being echoed back from each site using two different codecs (g.711 and g.728). This was repeated numerous time accumulating to over 2000 tests. Subjective testing was carried out on a sample of these to assess the impact of the different codecs. Clocks were unsynchronised and delay values are based on smoothed Round-Trip-Times.

- The ongoing Active Measurement Project (AMP) at the US National Laboratory for Applied Network Research (NLANR) [119]. The AMP project has a large and growing number of monitor nodes (currently 130) distributed across the world but mostly in the US that carry out continuous monitoring of network conditions that connect the monitors, amounting to over 14,000 paths. This is publicly available and includes information on Round Trip Times (RTT), measured every minute or
<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
<th>Dest. Location</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>amp − hean</td>
<td>amp − hutf</td>
<td>HUTF Finland</td>
<td>Irl1</td>
</tr>
<tr>
<td>amp − hean</td>
<td>amp − eltn</td>
<td>ELTENET Hungary</td>
<td>Irl2</td>
</tr>
<tr>
<td>amp − hean</td>
<td>amp − thor</td>
<td>Norwegian Univ. of Sc. and Tech.</td>
<td>Irl3</td>
</tr>
<tr>
<td>amp − hean</td>
<td>amp − apantyo</td>
<td>APAN Toyko</td>
<td>Irl4</td>
</tr>
</tbody>
</table>

Table 6.10: Paths from Ireland to Mainland Europe & Toyko

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
<th>Dest. Location</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>amp − thor</td>
<td>amp − columbia</td>
<td>Columbia University, NY</td>
<td>Nor1</td>
</tr>
<tr>
<td>amp − thor</td>
<td>amp − fsu</td>
<td>Florida State University</td>
<td>Nor2</td>
</tr>
<tr>
<td>amp − thor</td>
<td>amp − stanford</td>
<td>Stanford University</td>
<td>Nor3</td>
</tr>
<tr>
<td>amp − thor</td>
<td>amp − ucb</td>
<td>University of California, Berkeley</td>
<td>Nor4</td>
</tr>
<tr>
<td>amp − thor</td>
<td>amp − utexas</td>
<td>University of Texas</td>
<td>Nor5</td>
</tr>
<tr>
<td>amp − thor</td>
<td>amp − uwashington</td>
<td>University of Washington</td>
<td>Nor6</td>
</tr>
</tbody>
</table>

Table 6.11: Transatlantic Paths

so as well as packet loss and topology. A subset of 14 nodes was chosen, four in Europe, one in Toyko, and the remainder in the US. From these nodes a subset of 14 paths was selected, 4 from Ireland (amp − hean) to 3 locations in Europe and one to Toyko, 6 transatlantic from Norway (amp − thor) to the US (to both east and west coast) and the remainder within the US as outlined in Tables 6.10, 6.11 and 6.12 respectively.

One full week of data (from 8/12/03 to 14/12/03) was downloaded for each of the

<table>
<thead>
<tr>
<th>Source</th>
<th>Src. Location</th>
<th>Destination</th>
<th>Dest. Location</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>amp − uic</td>
<td>Univ. of Illinois, Chicago</td>
<td>amp − usf</td>
<td>Univ. of Sth. Florida</td>
<td>US1</td>
</tr>
<tr>
<td>amp − columbia</td>
<td>Columbia University, NY</td>
<td>amp − alaska</td>
<td>Univ. of Alaska</td>
<td>US2</td>
</tr>
<tr>
<td>amp − harv</td>
<td>Harvard University</td>
<td>amp − ucb</td>
<td>Univ. of Calif., Berkeley</td>
<td>US3</td>
</tr>
<tr>
<td>amp − alaska</td>
<td>Univ. of Alaska</td>
<td>amp − usf</td>
<td>Univ. of Sth. Florida</td>
<td>US4</td>
</tr>
</tbody>
</table>

Table 6.12: US Internal Paths

132
transatlantic and US internal paths and delay characteristics analysed.
The Irish amp — hean monitor node within HEAnet was only recently added and thus the data downloaded relates to the period 3/03/04 to 9/03/04.

Note that the AMP RTT data is only recorded every minute or so in contrast to the trace delay data used within the simulator and emulator approaches. The trace data measured RTT every 32 msec to simulate a VoIP data stream and thus could be used for comparative analysis of playout strategies. The AMP RTT data cannot therefore be used to simulate the delay and jitter characteristics that a VoIP application might encounter: nonetheless, delay data from the AMP project does provide very useful information for assessing the applicability of the hybrid playout approach.

6.5 Internet Delay Studies: Results

Of the four studies of Internet characteristics identified, two of these, [12] and [103] were specifically aimed at VoIP whereas [72] and [119] are more general studies. In particular Paxson’s work [72] examined a broad range of Internet characteristics. Each of these studies is analysed separately in the following sections.

6.5.1 Paxson (1994/1995)

The focus here is on Paxson’s delay analysis and specifically that of Round-Trip-Times (RTT) rather than his estimates of One-way Transit Times (OTT). His use of TCP file transfers rather than UDP probes or ping packets render his results less applicable to multimedia data than the other approaches for two reasons. Firstly, it resulted in asymmetric data flows (TCP data in the outgoing direction and the much smaller ACK data in incoming direction) and secondly TCP transfers are sent over a wide range of timescales compared to the constant-rate characteristic of the other approaches. A further limitation of Paxson’s work is that it does not provide summarised absolute delay data for all sites and paths but rather deals with ratios. Nonetheless, the more relevant findings were as follows:

- The use of RTT/2 as an estimate of one way delays should only be used for approximate analysis as asymmetries are often present. As such, the use of RTT measurements within this thesis is in line with this finding.
<table>
<thead>
<tr>
<th>Path Type</th>
<th>Number of Paths</th>
<th>Fixed Delay</th>
<th>Variable Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>11</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
<td>Low</td>
<td>Low with spikes</td>
</tr>
<tr>
<td>C</td>
<td>16</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>D</td>
<td>4</td>
<td>High</td>
<td>Low with spikes</td>
</tr>
<tr>
<td>E</td>
<td>10</td>
<td>High</td>
<td>High</td>
</tr>
</tbody>
</table>

Table 6.13: Path Classification According to Fixed and Variable Delay

- Although geographical distance largely determined the RTT delays, many anomalies were detected such as RTT delays that varied hugely between connections for no apparent reason. He reported a significant number of sites for which many connections had RTT values far in excess of the ITU G.114 limit for acceptable speech.

- The distribution of the ratio of maximum to minimum RTT showed great variation with a mean value of 3 but the upper 5% of measurements had a ratio of 6 and higher.

The extent of RTT variation reported is significant and indicates that many such paths might not be suitable for VoIP traffic. Paxson’s work had very broad objectives and as such his findings have limited value in the context of this thesis. In addition, results are based on measured data that is now almost a decade old.

6.5.2 Tobagi et al. (2001)

In [12] forty three (43) US backbone paths linking five (5) cities were studied over a period of 2.5 days in 2001. Probes of size 50 bytes were sent at 10 msec intervals in order to simulate VoIP traffic and one-way delays (using synchronised clocks) and loss rates were recorded. The 43 paths were further classified into five types (A-E) according to fixed and variable delay components, as shown in Table 6.13.

Types A & B had low fixed delay (all on the east coast) whereas types C, D & E were coast-to-coast with higher fixed delay. Furthermore types A & C had low variable i.e. queuing delay (indicating low congestion levels), types B & D had also low queuing except for occasional delay spikes and finally type E paths had high variable delay (indicating highly loaded paths).
For 39 of these paths (Types A, B, C and E) 98% of packets experienced one-way delays less than or equal to 120 msec. For the remaining 4 paths (type D), classified as long haul with significant delay burstiness, 98% of packets experienced delays less than or equal to 170 msec. Not surprisingly, paths of type A and C performed best from a VoIP perspective. For types B and D, a conservatively high fixed delay playout point performed better than the adaptive approach (Alg. B). The authors suggest that delay characteristics are such that jitter is a more serious issue than actual delay and that adaptive techniques perform poorly, relative to well informed fixed playout strategies. As such, they confirm that delay-aware playout strategies such as the hybrid are a viable alternative to conventional delay-unaware adaptive approaches.

6.5.3 Roychoudhuri et al. (2001/2002)

In [103], the RTT, jitter, and loss characteristics of 14 paths were examined. Seven of these were internal to the US and the others were from the US to locations spread out across Europe, Asia, Australia, South America and Egypt. Smoothed RTT values were presented using a weighting factor of \( \alpha = 0.9 \). Within the US, the maximum smoothed RTT delays did not exceed 140 msec except for one path and similar positive results were reported for the two European paths. The Australian, Egyptian and Asian data indicated maximum RTT values close to 400 msec and finally the Ecuadorian site, physically closer than many of the other International links returned the worst results with maximum RTT delays of over 600 msec. The principal finding of this work was that RTT delay values were generally within acceptable voice quality limits as derived by [95] from G.114 recommendation i.e. one way delay of 400 for G.711 and less so for G.728 presuming perfect echo cancellation and no loss. Periods of higher delay spikes however were reported, mostly on international links. On the other hand they report that jitter remains a significant problem, especially on the international links where differences between consecutive packet RTT delays regularly exceeded a 50 msec threshold. As illustrated in section 6.2.2, the hybrid strategy will outperform conventional adaptive strategies where delays are within G.114 and significant jitter is present.
<table>
<thead>
<tr>
<th>Path</th>
<th>RTT Delay (msec)</th>
<th>RTT Delay Range (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min.</td>
<td>Max.</td>
</tr>
<tr>
<td>Irl1</td>
<td>54</td>
<td>75</td>
</tr>
<tr>
<td>Irl2</td>
<td>48</td>
<td>116</td>
</tr>
<tr>
<td>Irl3</td>
<td>63</td>
<td>1415</td>
</tr>
</tbody>
</table>

Table 6.14: Delay and Loss Data for European Paths

<table>
<thead>
<tr>
<th>Path</th>
<th>RTT Delay (msec)</th>
<th>RTT Delay Range (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min.</td>
<td>Max.</td>
</tr>
<tr>
<td>Irl4</td>
<td>246</td>
<td>335</td>
</tr>
</tbody>
</table>

Table 6.15: Delay and Loss Data for Ireland to Toyko Path

6.5.4 Active Measurement Project

Finally a block of data from the AMP project was analysed. In total, a week's trace data for fourteen paths was downloaded and analysed. This included 4 from Ireland (Irl1 – 3) to 3 locations in Europe and one to Toyko (Irl4), 6 transatlantic from Norway to both east and west coast of the US (Nor1 – 6) and the remainder within the US (US1 – 4) as outlined in Tables 6.10, 6.11 and 6.12 respectively. Tables 6.14, 6.15, 6.16 and 6.17 summarise results for all paths. Note that the range of delays for the amp – hean to amp – apantyo path (Irl4) shown in Table 6.15 was similar to that used for the transatlantic data rather than for the other three European paths. The sample rate was approximately once per minute resulting in a sample size of approximately 10,080 over seven days for each path.

Looking firstly at the three European paths (Irl1 – 3) in Table 6.14, it is clear that RTT delays rarely exceeded 70 msec on any of the three paths. Although path Irl3 to amp – thor in Norway shows the highest delay spread, over 99% of packets still had delays under 70 msec. Despite the high maximum delay of 1415 msec, only 2 packets had delays greater than 300 msec. This path also suffered the highest link loss rate, albeit 0.4%.

Analysis of the transatlantic data shown in Table 6.16 shows that two of the six paths (Nor1 to amp – columbia & Nor5 to amp – utexas) had no RTT measurements greater than 300 msec, three had less than 1% (equivalent to 100 packets) above 300 msec and only path Nor3 (to Stanford) had a significantly higher rate above 300 msec (> 2 %). Note
<table>
<thead>
<tr>
<th>Path</th>
<th>RTT Delay (msec)</th>
<th>RTT Delay Range (msec)</th>
<th>Loss%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min.</td>
<td>Max.</td>
<td>Avg</td>
</tr>
<tr>
<td>Nor1</td>
<td>119</td>
<td>276</td>
<td>120</td>
</tr>
<tr>
<td>Nor2</td>
<td>135</td>
<td>593</td>
<td>153</td>
</tr>
<tr>
<td>Nor3</td>
<td>197</td>
<td>5215</td>
<td>289</td>
</tr>
<tr>
<td>Nor4</td>
<td>196</td>
<td>343</td>
<td>214</td>
</tr>
<tr>
<td>Nor5</td>
<td>153</td>
<td>298</td>
<td>177</td>
</tr>
<tr>
<td>Nor6</td>
<td>188</td>
<td>587</td>
<td>199</td>
</tr>
</tbody>
</table>

Table 6.16: Delay and Loss Data for Transatlantic Paths

<table>
<thead>
<tr>
<th>Path</th>
<th>RTT Delay (msec)</th>
<th>RTT Delay Range (msec)</th>
<th>Loss%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min.</td>
<td>Max.</td>
<td>Avg</td>
</tr>
<tr>
<td>US1</td>
<td>26</td>
<td>232</td>
<td>26.6</td>
</tr>
<tr>
<td>US2</td>
<td>115</td>
<td>691</td>
<td>115</td>
</tr>
<tr>
<td>US3</td>
<td>80</td>
<td>98</td>
<td>92</td>
</tr>
<tr>
<td>US4</td>
<td>112</td>
<td>122</td>
<td>112</td>
</tr>
</tbody>
</table>

Table 6.17: Delay and Loss Data for Internal US Paths
that *Nor3* also suffered from periods of high link loss, coupled with high delays of between
3-5 seconds which explains the high average value relative to the minimum.

Path *Ir14* from Dublin to Toyko, summarised in Table 6.15 showed the highest minimum
delay reflecting the physical distance. Nonetheless, over 99.9% of packets came in within
300 msec and link loss rates were very low at 0.02%.

Results for the internal US paths are summarised in Table 6.17. Both nodes for path
US1 are on the east coast which is reflected in the relatively low delay figures. Paths US2,
US3 and US4 are coast-to-coast and yet only one of over 30,000 packets had a RTT greater
than 300 msec. Interestingly loss rates were highest for the path US1 though still very low
at 0.2%.

More detailed delay analysis in histogram form is provided in appendix D. Figures D.1, D.2 and D.3 refer to the European paths and Figure D.4 to the Toyko path. Similarly, Figures D.5, D.6, D.7, D.8, D.9 and D.10 outline the delay spread for the transatlantic paths. Finally Figures D.11, D.12, D.13 and D.14 refer to the internal US paths. A bin size of 5 msec was chosen for all figures though the range of delays shown varies depending on the path type.

In general, the degree of jitter was surprisingly very low with a single bin accounting for
most of the measurements, regardless of the path type (European, Toyko, Transatlantic or
Internal US). Note that in the trace data presented in section 6.2.1, the delays from NUI,G
to Berkeley also showed very little jitter and link packet loss suggesting well provisioned
and uncongested transatlantic links.

In general the subset of results drawn from the AMP project indicate that delay character-
cistics confirm the applicability of the hybrid playout strategy. Only 2 of over 30,000
packets from the internal European paths and only 1 packet from the Toyko path had
delays greater than 300 msec with the European data generally within 70 msec. Of the
60,000 transatlantic measurements, only 0.6% had delays above 300 msec whereas within
the US, all but one measurement was less than 300 and most were within 150 msec. As
such an informed fixed delay playout point within the G.114 one-way limit of 150 msec will
work very well presuming symmetric delays. Nonetheless, the AMP results are limited for
a number of reasons:

- The AMP project nodes are generally connected by very well provisioned links, many
  of which are part of the Internet2 infrastructure. As such congestion problems
  are limited and thus they are not representative of the Internet. In [66], a similar
argument is made regarding the AMP data.

- The data is mostly US based though this situation is improving. Extensive transatlantic trace data for only one European node (amp – thor) was available which limits the value of the transatlantic results as the paths shared significant common links.

- RTT delay data is based on measurements every minute or so. As such jitter data relevant to the timescale of VoIP sessions is not available. For this reason, the AMP data was not considered for use within the simulator/emulator approaches.

- Notwithstanding the limitations of the data from a jitter perspective, the minimal degree of jitter present (albeit at 60 sec intervals or so) in the data for all the paths differs significantly from the results of [12] and [103] detailed above and reflects the extent to which the paths were uncongested. A principal advantage of the hybrid playout strategy is that it avoids the significant late losses that adaptive playout strategies suffer from due to significant network jitter. This suggests that the benefits of the hybrid playout strategy may be less significant for AMP paths due to the absence of significant jitter, though more detailed data would be necessary to examine this further. On the other hand, the AMP data is based on nodes that are directly connected to core network paths. In reality, VoIP endpoints using these paths will connect to them from edge networks which will thus add delay and more significant jitter to the overall M2E delays.

### 6.6 Results Summary

Overall, the results from the various approaches adopted indicate that the hybrid strategy can be implemented successfully within real VoIP applications and more importantly, that it can deliver significant benefits over both short and long distance Internet paths when compared with conventional adaptive strategies. The WAN testbed proved the feasibility of implementing the hybrid strategy within a real VoIP application and tested it across the limited HEAnet network. The simulator-derived results based on both captured trace delay data and derived delay models were more comprehensive but similarly positive and were confirmed by the emulator approach. Finally, analysis of the 3rd party delay studies confirm the hybrid strategy’s wide applicability.
Regarding the third party delay studies, an obvious but important point to note is that total M2E delay is the important parameter rather than the network delay which is the focus of these studies. As outlined in section 2.2.1, network delay is only one component of M2E delay and thus assessing network delay figures relative to a total delay budget of 150 msec specified by G.114 is somewhat simplistic. A further factor that needs to be considered is that the network paths studied by Tobagi et al. in [12] and taken from the AMP project were well provisioned backbone paths, particularly so in the latter case. Nonetheless the third party analysis is very useful in assessing general delay and jitter characteristics. The principal finding from our own and third party trace data is that although jitter is still evident over Internet links, network delays rarely exceed the G.114 limit and thus the delay-aware hybrid approach can be implemented successfully.

6.6.1 Endpoint Complexities

Although results both in this chapter and indeed NTP performance results from section 4.5 confirm the feasibility and applicability of the hybrid, a number of endpoint implementation issues arose during testing. Many of these are of concern to all VoIP applications and are not specific to the hybrid playout strategy. A detailed analysis of all the issues is beyond the scope of this work and thus a brief review is presented in the following section. Note that the issue of clock skew is of particular relevance to the hybrid strategy and is examined in detail in chapter 7.

Timer Granularity/OS impact on Delay Measurement and Playout Control

Irrespective of whether adaptive or hybrid playout strategies are employed, delay measurements, either estimated in the case of conventional adaptive, or actual in the case of the hybrid are required. As detailed in section 4.2 timestamping can be limited by timer granularity which affects the accuracy of delay measurement and thus playout performance. All experimental work presented in this thesis was conducted on Linux platforms with better than millisecond level timestamping so the issue did not arise. On the other hand, many Windows platforms only provide a default timestamping granularity of 10 msec.

A separate but related issue concerns the control of playout time i.e. once a playout time is determined from delay measurements (either estimated or actual), the OS may limit the degree to which playout can be implemented at the correct time. As outlined
in section 6.3, a common approach in the Unix environment to suspending processes is through use of the `select()` function. If within the receiver, a playout strategy determines that the playout time is in the future (otherwise packet is too late for playout), the process that passes packets from the receiver buffer to the codec `sleeps` for an appropriate period. If however the granularity is limited by the process timeslice size, the actual playout point will differ from that desired.

Furthermore, and more generally, OS scheduling uncertainty can present problems. Delay measurements are based on packet arrive and send times, the latter deduced from RTP timestamp which in turn refers to when the packet was generated. Any OS delays encountered whilst the packet traverses the protocol stack from application level onto the wire and vica-versa at the receiver will distort delay measurements. Kouvelas et al. in [26] outline the limitations of conventional non-realtime OS from a multimedia perspective and introduce the concept of a `cushion`, based on OS scheduling patterns to avoid buffer starvation i.e. whereby CPU is tied up with another process, leading to audio output buffer starvation. In [74], a study of latency issues within the Windows NT OS is carried out and some anomalies are highlighted. More generally, conventional Operating Systems are designed primarily to maximise throughput whilst achieving a reasonably fair sharing of resources which does not map well to applications requiring soft realtime performance. In [120], a hardware and software architecture for a packet telephony device is developed. It describes some of the problems associated with the use of standard PCs as telephony endpoints. It uses the VxWorks RealTime OS to minimise latencies within a custom built telephony device in order to address many of these issues.

As described in section 6.3, a solution (for the Linux OS) to both improving the process timeslice granularity and to a lesser extent OS scheduling latencies within standard PCs is to redefine the former. In fact some of the more recent Linux ports have a default software clock rate of 1000 Hz. This solves both the playout control problem of adaptive/hybrid playout strategies as well as improving significantly the performance of the emulator from section 6.3 above, presuming that `select()` is used in both cases. Note that the NISTNet emulator uses the Real Time Clock (RTC) with an interrupt rate of 8192 Hz, resulting in a 122 microsecond granularity. The RealTime Linux variant (RTLinux) described in [121] now includes a 1000 Hz interrupt frequency as default [122]. Section 4.2 outlined that for Windows OS variants, work on multimedia timers has attempted to resolve timer granularity problems with reasonable success, particularly for HALX86 implementations.

141
In [123], the authors argue that there is no reason why default \textit{ticksize} cannot be reduced as CPU speeds have increased over the years and outline the benefits of reducing the former from a realtime perspective. They also report good performance on a Linux PC, configured with a 20,000 Hz interrupt frequency. Redefining software clock interrupt rates does however introduce additional OS overhead and may have other undesirable effects but generally should presents little difficulty for modern processors.

Finally, in related work previously referenced in section 4.2, [75] outlines the growing difficulty of providing precision timestamping of packet arrivals as link speeds increase. A hardware driven approach using a 16.7 MHz clock with a high quality oscillator is used as part of their Dag Universal Clock Kit (DUCK), resulting in a 60 nanosecond resolution.

**Sound Card Driver**

As introduced in section 2.3, a mismatch between sound card driver and user-configured frame size can also lead to packets being delivered to and from the codec in a manner that introduces significant additional delay uncertainty. For example, the Linux OSS driver implements a \textit{fragment size} that is a power-of-two bytes [25] and thus a frame size that matches this requirement (such as a 32 msec g711 frame size equivalent to 256 bytes = $2^8$) results in good performance. Otherwise a significant step-like jitter can be introduced resulting in additional delays. Fig. 6.18 illustrates this problem clearly with additional delays of up to 30 msec; a possible explanation is provided in appendix G. On the other hand, the Windows 2000 driver resulted in a similar step-like pattern if configured with a 32 msec packet size but operated smoothly with 30 msec packets.

**Application Issue: Packet Size and Silence Suppression**

From testing the \textit{oiphone} VoIP application, it was noted that where frame size was configured with multiple packets (eg. 3 gsm packets of 20 msec yielding a 60 msec frame), the application occasionally generated frames with less than 3 packets. This occurred generally when a silence period was detected and resulted in a shorter inter-packet interval than usual. This needs to be taken into account within the hybrid mechanism (and indeed in any adaptive approach) so that a lower end-to-end delay is not presumed distorting playout performance.
Figure 6.18: Effect of 30 msec Fragment Size on Performance
Clock Skew

The move from POTS-based telephony to VoIP has resulted in the single POTS clock being replaced by a multitude of clocks within sender and receiver endpoints. As section 4.1 outlined, oscillators suffer from inaccuracy and instability thus leading to a lack of synchronism between such clocks. This lack of synchronism can seriously distort delay measurement and receiver buffer performance. Due to its potential impact on both conventional adaptive and hybrid playout performance, the next chapter deals with the issue of clock skew in some detail, and proposes a novel skew detection mechanism.

It is important to note that endpoint issues relating to OS scheduling, sound card driver configuration and buffer distortion through clock skew are general issues that are of concern to all VoIP applications and not just those implementing adaptive/hybrid receiver playout strategies.

Subjective Testing versus E-Model

Subjective testing is undoubtedly the best approach for evaluating the relative performance of the various playout strategies. However the bulk of the hybrid strategy applicability tests were based on the simulator approach, which facilitated extensive and rapid testing across diverse networks. The E-Model provides critical support for such an approach in that it enables rapid performance comparisons. Such extensive testing would be infeasible in a real implementation environment utilising a subjective testing approach.

Although subjective testing was possible for the limited testing carried out using the WAN testbed and emulator approaches, the E-Model was chosen primarily due to its ease of deployment and also due to the uncertainties that the above endpoint issues raise. Although the skew issue was resolved, and others such as sound card driver and timer granularity issues were identified, the use of the E-Model avoids many of these and other uncertainties. Recent work by [9] carried out black box testing of various VoIP endpoints, both PC-based and dedicated IP phones and highlighted the extent of endpoint problems. Measurements of M2E delay introduced solely by the endpoints i.e. across a LAN with negligible network delay showed that the combination of various hardware and software delays within endpoints can vary significantly during sessions and often accumulate to 150 msec or more. In the context of the hybrid playout strategy, where the objective is to maintain overall M2E delays below 150 msec, and indeed more generally for Internet-based
multimedia, such results are a serious cause for concern.

6.7 Summary

In this chapter, both the feasibility and applicability of the hybrid playout strategy were confirmed. The WAN testbed approach proved the feasibility of the approach as well as its applicability in a limited network. The bulk of the applicability testing was carried out using a simulator approach which proved that the hybrid playout strategy can yield significant performance gains. The emulator approach was useful in confirming simulator-derived results. Analysis of third party delay studies further confirmed the applicability of the hybrid playout approach.

Although both the testbed and emulator approaches lend themselves to subjective testing, the E-Model was chosen for all performance analysis. In this context, a range of endpoint issues were reviewed that can distort playout quality in real applications and some solutions were proposed. Of these, the issue of clock skew is perhaps the most significant. In the next chapter, the issue of clock skew is addressed in detail. Its impact on all VoIP applications but particularly the hybrid playout strategy is outlined as well as a novel solution to skew detection and compensation that can be easily integrated into the hybrid playout strategy.
Chapter 7

Clock Skew

The circuit switched POTS (Plain Old Telephone System) preserves the timing relationship between media samples from sender to receiver digital exchanges through use of a common clock. As described in section 4.1 a typical Multimedia PC has at least three separate clocks, namely the system clock, the real-time clock (RTC) and the audio clock. The RTC is used principally on boot time only and thus can be disregarded in these discussions. From a timing perspective, the existence of separate audio and system clocks on either end-host can however introduce significant complications. Although much work has taken place in recent years that addresses the issue of system clock skew and its effect on precise delay measurement, such analysis is primarily aimed at removing skew after the event. In a VoIP environment, system and audio clock skew can distort both delay measurement & playout control as well as lead to poor buffer performance. As such, solutions are needed that operate in realtime.

This chapter examines the extent and effect of clock skew on delay measurement and receiver buffer performance. Regarding delay measurement, it analyses the effect on conventional adaptive and the hybrid playout strategies, concluding that though the impact is greater for the latter, it can also seriously degrade the performance of the former. A review of existing approaches to realtime skew detection and compensation is carried out and a novel high level solution to skew detection proposed and tested. Similar to the hybrid playout strategy, this solution is based on the integration of the Network Time Protocol (NTP) with the RTP (Realtime Transport Protocol) Control Protocol or RTCP and is easily integrated into the hybrid strategy. The chapter concludes by examining some software and hardware issues that can distort the operation of the proposed skew detection
mechanism.
A summarised version of this research can be found in [124].

7.1 Introduction to the Problem

The trend towards Internet based multimedia in recent years has introduced a range of complexities resulting from the lack of a common clock. Fig. 7.1 illustrates the relationship between audio and system clocks for a unicast multimedia session. As such, there are four separate clocks contributing to the session, each with its unique frequency characteristics.

As detailed in section 4.1, oscillator crystals can have inherent frequency errors greater than a few hundred parts-per-million and the extent of such errors can change over time or due to temperature changes. For timescales typical of VoIP sessions, changes in clock skew or drift within such sessions can largely be ignored. However, these and other factors but particularly temperature changes can result in different levels of relative skew between VoIP sessions. As such an accurate relative skew value determined under certain conditions may not apply subsequently.

In the context of accurate delay measurement, RFC 2330 [71], reviewed in section 4.2 deals with measurement uncertainties, including clock skew. In recent years, significant research has taken place to address the skew problem. Many of these approaches however aim to remove skew through post-processing of data and thus the emphasis has been on accurate post-analysis rather than resolving skew during the data transfer. In a VoIP context, skew needs to be resolved both accurately and quickly during a session in order to make use of such information. For example, a relative skew of say 400 ppm between two clocks will accumulate to a 100 msec error in just over 4 minutes. In the context of the ITU-T G.114 limit of 150 msec for one way delay, the impact of skew can thus be very significant.

7.2 Skew Detection for Accurate Delay Measurement

This section reviews briefly, recent work in skew detection and compensation. All of the approaches reviewed are based on removing skew from system clock-derived delay measurements and are designed principally for post processing of delay data. They do not consider the additional complexity introduced in a multimedia context by audio clocks.
System vs Audio Card Clock Issues

![Diagram showing the relationship between Host bibio, Internet, Host bibio2, Motherboard (bibio) (incl. system clock), Motherboard (bibio2) (incl. system clock), Sound Card (bibio), Sound Card (bibio2), Voice Packets, Buffer 1, Buffer 2, To/From Speaker/Mic.]

Figure 7.1: Relationship between Audio and System Clocks
Further details of these approaches are provided in appendix E.1.

- Paxson et al. (1996): Paxson in [27] devised and tested a series of algorithms for detecting clock resolution, clock accuracy, clock adjustments and relative clock skew from delay measurements. These measurements were collected from a comprehensive test program involving file transfers via TCP in one direction and acknowledgements in the reverse direction. As such this asymmetry in data flows is atypical of multimedia flows but was necessary for Paxson’s work which had a broad scope. A critical issue with Paxson’s work is that the algorithms are based on both forward and reverse measurement data being available. This is an obvious limitation in the context of a VoIP session where skew needs to be determined in realtime and delay information is restricted largely to incoming packet delays.

With regard to clock skew, a statistical test based on cumulative minima is developed to detect skew. Once skew is detected, it is measured using a robust linear fitting technique using median slopes applied to the cumulative minima points. Although skew was detected in only 13% and 3% of traces from two separate sets of results, the extent of such skew was significant with median values of 230 and 110 ppm respectively. The largest skew detected was over 5,000,000 ppm where a clock ran five times faster than its trace pair!

- Moon et al. (1999): More recent work by Moon et al. [125] adopt a linear programming approach to the problem. This approach seeks to place a line under all of the data points with the condition that the sum of vertical distance from the line to the actual points in minimised. They show through actual measurements and simulation that their approach is superior to that of Paxson.

- Zhang et al. (2002): Finally, Zhang et al. outline a convex-hull approach to the problem [126]. This approach describes a hull whose lower boundary is composed of line segments with endpoints taken from the dataset. The idea is somewhat similar to that of Moon et al. [125] in that it aims to find a line such that all data points are above the line. The authors claim that their approach performs better than [125] in that it works well in the presence of clock adjustments and that it can also be used for realtime processing.
In summary, Paxson’s approach applies a statistical test to two way measurement data, based on cumulative minima along with certain heuristics to isolate skew from other network anomalies and then uses robust line fitting to determine the extent of skew. Moon et al. determine skew using a line fitting technique that minimises the vertical distance between the line and the data points. Zhang et al. adopt a similar approach though report better resilience against clock adjustments. All of the above approaches are similar in that their time complexity is linear \(O(N)\) with the number of data points. The degree to which they suffer from false positives/negatives varies with the latter approaches reporting better robustness. All require significant data before a statistically accurate skew estimate can be determined. This is particularly so where significant network noise is present leading to incorrect skew estimates. The extent therefore to which they can be deployed successfully in an environment where skew needs to be determined in realtime such as for short VoIP sessions is very much dependent on network characteristics.

The next section examines the particular effects of skew in the context of multimedia applications where skew detection and compensation measures are required in realtime to be of any use.

### 7.3 Multimedia Clock Skew

For VoIP, the precise effects of relative skew depend on the degree to which the various system and audio clocks are used within the application. In general however, such effects can relate to both delay measurement and receiver buffer performance.

#### 7.3.1 Delay Distortion

As outlined in section 3.1, some simple (and inflexible) VoIP applications implement fixed-size buffer strategies which do not require the use of system clocks and thus Fig. 7.1 is not appropriate. For these applications, the receiver endpoint delays playout until a certain amount of data has been buffered. On the other hand, section 3.3 outlined that conventional adaptive playout algorithms though delay-unaware, react to trends in delay estimates whereas the hybrid playout strategy requires knowledge of actual delays. In both of these scenarios, delay measurements are required and generally this is done through the system clock. From Fig. 7.1, packets are generated according to the sender sound card clock rate,
arrive at the receiver where they are timestamped by the receiver system clock to determine end-to-end delay (either actual or estimated) and are played out of the receiver buffer according to the receiver sound card clock rate. Note that NTP will synchronise system clocks but has no effect on audio clocks.

In common with many other adaptive playout strategies, the hybrid algorithm utilises the system clock to timestamp incoming packets though the rate at which packets arrive is determined by the sender audio clock. As such, skew between sender audio and receiver system clocks will result in an apparent and gradual increase or decrease in one-way delay. For delay-unaware algorithms such as A,B,C,D and E, this distortion of estimated delays will affect playout calculations. With Algorithms A & B for example, it might seem from equation A.1 in appendix A that in determining the playout time of packet i $p_i$, any gradual distortion of $\tilde{d}_i$ caused by skew will be fully offset by the opposite trend in $t_i$. This is not the case. The critical factor is $\alpha$ which determines the amount of history or memory that is taken into account in determining the current playout delay. Similarly, approaches such as Alg. C and D maintain a histogram of estimated delay values of past packets to set the current playout time. In such situations delay values that are being gradually skewed will, depending on the amount of history that is taken into account, and the skew-rate, distort the playout delay and thus playout quality.

A simulator was developed to test the degree to which skew and history distort playout values for Alg. A. For the simulation, zero jitter was incorporated into network delays to ensure that the effects of skew and memory could be isolated from other factors. As can be seen from Fig. 7.2, the degree of playout error depends on the extent of skew and the amount of history considered (compare $\alpha=0.998002$ with 0.99 at 500/1000 ppm). Such distortion can be quite significant where skew values are very high (which is not uncommon) and a large memory is employed. Note also that the higher the $\alpha$ value, the greater the error and also the longer it takes for the full extent of the error to emerge.

The effect of skew on the performance of the hybrid playout strategy is both more obvious and significant. As detailed above, a relative skew of 400 ppm will result in delay distortion of almost 100 msec over a 4 minute period. Considering the importance of accurate delay measurements within the hybrid algorithm, and the underlying G.114 150 msec limit, the need to detect and compensate for such skew is critical for effective hybrid performance.
Figure 7.2: Effect of skew on Alg. A
7.3.2 Buffer Distortion

Regardless of whether the playout algorithm is fixed, adaptive or hybrid, a relative skew between receiver and sender audio clock rates can seriously degrade buffer performance and playout quality. A faster sender audio clock can lead to packet accumulation in the buffer with consequent problems such as higher buffer residency delays and buffer overflow whereas a slower sender audio clock can result in buffer underfill with consequent audio discontinuities. For VoIP, the use of silence detection or voice activity detection (VAD) schemes reduce the impact of this problem as the stop-start nature of voice ensures that the buffer is frequently emptied during silence periods and so such buffer problems cannot accumulate to any great extent. Where VAD is suppressed or for situations where the extent of skew is high and talkspurts are prolonged, the issue needs to be addressed.

7.4 Skew Detection & Compensation for Realtime Multimedia

This section summarises two recent and related approaches to skew detection & compensation for multimedia applications. Details are provided in appendix E.2. It then introduces and describes the alternative approach to skew detection proposed in this thesis.

- Akester et al. (2002):

In [10], Akester et al determine the relative skew between sender and receiver audio clocks by timestamping the arrival of packets at the receiver using a low level interface to the audio clock and comparing the time between successive incoming packets with the time as indicated by the difference in packet RTP timestamps. This approach yields a ratio $e_o$ on a per packet basis but in order to smooth out the effect of network jitter, a filter is applied to yield a smoother ratio $e$:

$$e = e + (e_o - e)/16$$  \hspace{1cm} (7.1)

A mechanism for spike detection is also included. To compensate for skew, this ratio $e$ is applied to a low level rate converter, that matches the receiver audio clock rate to that of the sender.

- Hodson et al. (2001):
In [11], Hodson et al. utilise a similar low level mechanism to Akester but tuning details are different. Rather than maintaining a ratio, they timestamp incoming packets via the sound card buffer level and determine the estimated one-way delay $m_i$ by comparing this to incoming packet RTP timestamps.

To smooth out jitter, a linear filter mechanism is applied to estimated delays as follows:

$$
m_i = \alpha * m_{i-1} + (1 - \alpha) * m_i$$ (7.2)

$\alpha$, which determines the responsiveness of the mechanism, is set to 31/32. A spike detection mechanism is also employed. When the value of $\hat{m}_i$ deviates significantly from its starting value $m_{\text{active}}$, the compensation mechanism is invoked that adds or deletes samples from the receiver buffer.

The detection mechanism in both approaches is somewhat similar in that they use low-level system calls to monitor sound card buffer levels, though the approach to spike detection and the tuning parameters differ. Both approaches use a filtering mechanism that absorbs network jitter leading to a gradual convergence to a value that represents the relative audio skew. In the complete absence of jitter, a steady-state $e$ value will emerge from [10] resulting in a fixed rate conversion between receiver and sender whereas for [11], the intervals between adjustments (high/low watermark reached) will be constant. The tuning factors within each approach are designed to protect the mechanism from network jitter. Both approaches will however misinterpret a gradual and sustained change in actual delay as a change in skew. Akester’s approach in [10] will result in a different value for $e$ which will result in changing receiver playout rates relative to the sender. With [11], the result will be a gradual change in $\hat{m}_i$ with consequent insertion or deletion of samples. In general both approaches can be seen, not as robust skew estimators but rather as mechanisms for preventing the receiver buffer from either overflowing or emptying.

### 7.4.1 NTP/RTCP-based Skew Detection

As detailed in section 2.2.2, the RTP timestamp enables a receiver to accurately reconstruct incoming packets for playout. The timestamps are media specific and relate to the sample number generated by the codec. RTCP SR packets (if implemented correctly) include the
system clock timestamp (in NTP format) indicating when the SR packet was generated, along with the corresponding RTP timestamp which is set by the audio card rate.

Each sender periodically generates RTCP SR packets during the lifetime of a media session, and sends them to each receiving host. The rate at which they are generated depends on the number of participants and is described in detail in the associated RFC [1]. [127] outlines some problems with this mechanism when applied to very large multicast groups. In general total RTCP traffic should not exceed 5% of the available bandwidth and this dictates the rate at which RTCP packets are generated. For a unicast session, the interval between successive packets is of the order of seconds. If both system and audio clocks are running at exactly the same rate on a given host, the interval between successive RTCP SR packets as indicated by the increment in RTP and NTP timestamps over the period will be equal (eg. if interval is 10 sec according to the NTP timestamps, the RTP timestamp increment should be 80000, presuming that the sample interval is 125 microseconds). Any differences will indicate the skew between audio and system clock rates.

By accumulating the information from successive RTCP SR packets over a session, each receiver can precisely and quickly determine the relative skew value between the sender’s system and audio clocks. If additionally, system clocks are synchronised via NTP, this value also represents the relative skew between sender audio and the receiver system clock. With such information, receivers know precisely what correction needs to be applied to measured delays (actual or estimated) to avoid the gradual distortion described above in section 7.3.

Furthermore, by examining its own RTCP SR packets being generated for transmission, the receiver can determine the relative skew between its own audio and system clocks. As the relative skew of both audio clocks relative to the receiver system clock is now known, the receiver can deduce the relative skew between the sender audio clock and its own audio clock. It can thus correct for buffer overflow/underfill problems described above in 7.3.2.

In summary, using information from both sets of RTCP packets, each receiver can quickly generate a precise picture of all four clock rates and implement appropriate compensatory action. The following section outlines a testbed developed to test this approach and presents results.
7.4.2 Testbed Design

Fig. 7.3 describes the testbed developed for evaluating the skew detection approach. Essentially, it is similar to that developed for the emulator shown in Fig. 5.4 but with the delay emulator removed. To evaluate and prove the skew detection mechanism, a series of tests were carried out. Implementation details are as follows:

1. Firstly, incoming packet details (arrival time according to receiver system clock and RTP timestamp) were recorded. Secondly both incoming RTCP SR packets from the remote host along with outgoing RTCP SR packets from the local host were captured and NTP and RTP timestamps extracted. As outlined in section 5.1.1, the existing ohphone code for generating RTCP SR packets had to be rewritten to enforce more tightly, the mapping between the RTP and NTP timestamps.

With this data and with system clocks synchronised via NTP, it was possible to determine whether the skew determined from successive incoming RTCP packets using the above mechanism (sender system clock relative to sender audio clock) was
reproduced by timestamping incoming RTP data packets and thus determining the relative skew between sender audio and receiver system clock.

2. Tests were carried out across a lightly loaded LAN to ensure that the network component of delay suffered minimal jitter. This minimised any network noise that would otherwise have distorted the results from (1) above making validation more difficult.

3. The stratum 1 NTP server located on the same LAN provided a robust NTP subnet for system clocks. Tests were initially carried out with system clocks synchronised via NTP. A second series of tests were then carried out with the host bibio2 freewheeling i.e. with ntpd disabled. This was done to test the consistency of results.

7.4.3 Results

With NTP running on both end-hosts, numerous tests were carried out. A sample set are presented as follows:

Fig. 7.4 indicates the skew between the audio clock on bibio and the system clock on bibio2 as determined from RTP packets sent from bibio to bibio2. Fig. 7.5 indicates the skew between the audio and system clocks on bibio as determined from incoming RTCP SR packets received by bibio2.

Fig. 7.6 indicates the skew between the audio clock on bibio2 and the system clock on bibio as determined from RTP packets sent from bibio2 to bibio. Fig. 7.7 indicates the skew between the audio and system clocks on bibio2 as determined from incoming RTCP SR packets received by bibio.

As evident from the graphs, there is very strong agreement between Fig. 7.4 and Fig. 7.5, and between Fig. 7.6 and Fig. 7.7. Fig. 7.8 summarises these results for clarity. As such, a simple analysis of both incoming and outgoing RTCP SR packets will quickly yield a precise picture of all four clock rates.

Fig. 7.9 indicates results achieved when only bibio was synchronised via NTP and bibio2 system clock was running at its default (undisciplined) frequency. Although the respective positions of the audio clocks remain constant, the position of bibio2 system clock changes, reflecting its default frequency.
Incoming packets from bibio

Incoming traffic from bibio to bibio2
Skew = $-\frac{1900}{125}$
$= -15$ppm

Figure 7.4: bibio audio seen by bibio2 sys clock
Figure 7.5: RTCP SR packets received by bibio2 (from bibio)
Incoming packets from bibio2

![Graph showing time difference in milliseconds between system receive and user request times.]

- Incoming packets timestamped by bibio sys clock
- Skew $= \frac{-3600}{165} = -22$ ppm

**Figure 7.6:** bibio2 audio seen by bibio sys clock
Figure 7.7: RTCP SR packets received by bibio (from bibio2)
System vs Audio Clocks: Results

System clocks synchronised via NTP

Figure 7.8: Clock Skew Inter-relationships

System vs Audio Clocks: Results II

Figure 7.9: Clock Skew Inter-relationships: bibio2 not synchronised
7.4.4 Integrating Skew Detection & Compensation into the Hybrid Playout Strategy

The exact symptoms of skew and thus the appropriate compensatory action depend on the precise operation of the audio application. A simple fixed-size buffer application may over a period of time see overfill or underfill depending on the difference in audio clock rates. As outlined above in section 7.3.2, other factors such as the presence of VAD schemes or the extent of talkspurts and skew come into play. In section 7.4, two approaches to skew compensation were outlined, involving clock rate conversion and the more simple insertion/deletion of samples from the receiver buffer. The next section summarises how the skew related problems of delay distortion and buffer under/overfill are addressed within the hybrid playout strategy.

Delay Distortion

On receipt of successive incoming RTCP SR packets, the skew between sender audio and receiver system clock is determined and boolean variable AUD_SYS_SKEW RETD is set to TRUE. Subsequent incoming packet sendtimes can then be adjusted to the receiver’s clock rate. In the following pseudocode extract, the sendtime of incoming packets is adjusted according to the skew between sender audio and receiver system clocks. Init_sendtime is the sendtime of the first packet and delta_RTP is interval between first and current packet sendtimes as seen by the sender audio clock (converted to its equivalent UTC timescale i.e. typically a delta_RTP of 8000 equals 1 second).

```c
If (AUD_SYS_SKEW_RETD)
    Sendtime=Init_sendtime + delta_RTP( 1 + aud_sys_skew)
Else
    Sendtime=Init_sendtime + delta_RTP
```

Buffer Under/Overfill

The extent to which skew affects buffer performance is determined by factors such as the size of the buffer, the skew rate, and the duration of uninterrupted data i.e. a large buffer
will facilitate a large accumulation (presuming sender audio clock is faster than receiver audio clock) but this will add to the overall end-to-end delay. As such, the degree to which users will tolerate the effect of such skew can also be factored into the mechanism. This will then dictate the maximum period of uninterrupted data or limit before intervention is required. The following simplified pseudocode illustrates the mechanism as implemented within the hybrid strategy. Boolean variable AUD_AUD_SKEW_RET D is set to TRUE when both receiver and sender RTCP SR packets are received and processed yielding the relative skew between sender audio and receiver audio clocks:

If (AUD_AUD_SKEW_RET D)
    If (aud_aud_skew > 0) //Sender audio faster
        If (PACKET_NUM >= limit)
            Delete samples
            Reset PACKET_NUM
        Else
            PACKET_NUM++
    Else //Receiver audio faster
        If (PACKET_NUM >= limit)
            Add samples
            Reset PACKET_NUM
        Else
            PACKET_NUM++
    Else
        Maintain Uncorrected Mode

The strategy outlined above is similar to that of [11] that periodically inserts or deletes samples to help maintain the buffer at an optimum level rather than letting it move to either extreme. In any event, silence suppression (VAD) will if present, help prevent the problem building up to a significant degree, unless the relative skew rate is very high or the buffer is quite small. With VAD, the variable PACKET_NUM is reset to zero when a new talkspurt is detected. Due to the use of VAD and the relatively low level of audio-audio skew in the particular hardware employed, the problems associated with buffer over/underfill were not encountered during tests.
7.4.5 Relative Performance Comparison

In this section, the proposed NTP/RTCP skew detection approach is compared with the alternative approaches of [10] and [11] from section 7.4 and indeed more generally with the approaches aimed at accurate delay measurement, outlined in section 7.2. The proposed approach is based on integrating NTP with RTCP SR packets, similar to the hybrid playout strategy and as such, strict comparisons are difficult in that it is part of the overall package with the associated and significant benefits of synchronised time. The main points arising from such a comparison are as follows:

- The NTP/RTCP approach operates at a high level, integrating existing protocols. As such, there is no requirement for low level system calls. On the other hand, it requires that such protocols, particularly RTCP are implemented correctly.

- The NTP/RTCP approach is completely unaffected by network characteristics. It returns a skew value that unambiguously relates to sender and receiver clocks only. The approaches of [10] and [11] are not robust skew estimators in that they react to sustained trends in network characteristics. In addition they require tuning to filter out jitter which as described in section 3.3 is a non trivial problem. On the other hand, the approaches outlined above in section 7.2 are specifically designed to isolate skew from delay measurements with varying degrees of success. These post-processing approaches however all require significant data for robust operation, have similar time complexity of $O(N)$ and the degree to which they suffer from false positive/negatives varies with greater robustness reported from the more recent approaches such as [125] and [126]. Undoubtedly, they are best suited to post analysis rather than realtime analysis.

- The NTP/RTCP approach has a significant protocol overhead in that it requires NTP to be implemented effectively across end hosts. However in the context of multimedia applications, NTP brings the associated benefits of actual delay measurement data which none of the above approaches achieve. Finally and from a time complexity viewpoint, once protocols are implemented correctly, the actual NTP/RTCP skew detection mechanism is trivial in comparison to all of the other approaches.

Note that there is some evidence to indicate that the extent of skew (relative to UTC) of audio clocks can be much higher than that found in system clocks. Regarding system clocks,
Mills in [69] found an average skew of 78 ppm among 20,000 clocks and values determined in the course of this thesis were rarely above 50 ppm. Regarding audio clocks however, skew values greater than 10000 ppm were encountered during testing though results presented above in section 7.4.3 indicate that for the particular hardware employed, the extent of audio clock skew was similar in magnitude to that of the system clocks. Hodson in [11] report audio skew rates of +/-0.5% whereas [10] report values of 0.4%. Considering that 1% equates to 10,000 ppm, such results suggest that audio skew rates can be orders of magnitude higher than system clocks. As such the NTP/RTCP approach may well deliver useful data in the absence of NTP. A comprehensive survey of skew rates amongst audio and system clocks would be required to validate this suggestion and in any event, NTP provides significant other benefits such as precise delay information.

7.4.6 Software & Hardware Issues

In section 6.6.1 various endpoint issues that affect the performance of the hybrid playout strategy were outlined. Although both the hybrid playout strategy and the skew detection method utilize RTCP SR packets, the latter is more highly dependent on the precision of the RTCP SR packet generation process. The RTP and NTP timestamps must be generated simultaneously and must both be measured from an unbroken timeline. Although results presented show that the method performs well, it is affected by many of the software and hardware issues detailed in section 6.6.1 and in some ways is more sensitive to them. For example, a relative skew of 50 ppm between sender audio and receiver system clocks manifests itself in a 3 millisecond error per minute. In order to work effectively, the NTP/RTCP method needs to be able to detect trends of this magnitude. Outlined below therefore are a number of problems and issues that either were encountered during testing or were noted from other research.

- System Time: A limitation relating to system clock granularity such as the 10 msec Windows system clock call, GetSystemTimeAsFileTime() as detailed in section 4.2 will add significant noise to the accuracy of the system time returned. For this reason, the skew detection mechanism was not tested on a Windows platform.

- A similar problem applies regarding the sawtooth-like increase in delay caused by driver-application mismatch (described in section 6.6.1 and in greater detail in ap-
pendix G). Due to the use of an appropriate packet size (256 bytes), this problem was not present during the above testing.

- In theory, once the second RTCP SR packet is received, the slope of the line can be determined yielding the skew value. In practice the mechanism waits until three SR packets have been received and then calculates the inter-packet slopes yielding 3 values. If any outliers are present, then a problem exists and the process is repeated, otherwise an average slope is returned. From Figures 7.7 and 7.4, the benefit of using an average rather than a result based on the first two SR packets can be seen. As outlined in section 7.4.1, the interval between RTCP packets varies depending on the number of participants. For unicast sessions, the extra delay encountered in waiting for the third RTCP SR packet amounts to single figure seconds.

- On infrequent occasions, an outlier was detected amongst the three slopes. This was generally due to a large step in system clock time. This could be due to OS scheduling and resulted in a step increase in delay. As detailed in [26], conventional OS can cause problems for multimedia applications with realtime requirements.

- OS Dependency of NTP: As discussed in section 4.3.1, NTP was designed initially for Unix and has only recently been ported to Windows platforms. In addition, with certain operating systems such as the recent Linux versions, NTP is implemented within the kernel which largely eliminates the sawtooth effect that otherwise results from the correcting mechanism. Such a sawtooth pattern if present however would introduce errors in the system clock thus adding noise to the skew detection mechanism. The extent of such errors would depend on the magnitude of system clock skew and the amortisation rate. The effect of such errors would in turn depend on the relative ratio of system to audio clock skew and the timescale over which the skew detection mechanism operates. If relatively speaking, the audio skew is much larger than system clock skew as detailed above, the effect of the sawtooth pattern would be minimal. Furthermore, the extent of the error is fixed and thus diminishes in proportion to the duration over which the skew detection mechanism operates. As both test hosts were running Linux, such problems were not encountered.

- In Fig. 7.1, the existence of single audio clocks means that for simple applications that do not involve system clocks for measurement, the actual M2E delay in each
direction that users experience will change at equal but opposing rates due to relative audio clock skew and consequent buffer distortion. This fact is used by [55] as part of the heuristic checks for system clock skew. Although not encountered by the authors, work by [9] suggests that some audio cards may utilise separate clocks for playback and recording. This is based on results that show either same-sign skew in both directions or large differences in the magnitude of skew seen by either end. If separate audio clocks exist in some implementations, this will result in a total of six rather than four oscillators and further complications will arise, requiring changes to all the skew measurement approaches described in previous sections. Regarding the NTP/RTCP approach, recall that RTP timestamps within RTCP SR packets refer to data being sent and thus relate to the recording clock. However the degree to which a receiver buffer over or underfills depends on the difference between the rate at which packets are played out (set by the receiver audio playback clock) and the rate at which packets arrive (set by the sender audio recording clock). Similarly, both [10] and [11] estimate relative audio clock skew by examining the rate at which the receiver recording clock differs from the sender recording clock. Although the results presented by [9] could be attributed to application level or sound card driver issues, the suggestion of dual audio clocks needs to be researched in greater detail.

7.5 Summary

In this chapter, the single endpoint issue of clock skew was addressed. The effect of system clock skew on delay measurement has received significant attention over the years and a number of approaches aimed at post-processing of data were reviewed. With the move towards VoIP, the issue of clock skew becomes more complex as it involves both system and audio clocks and thus four (and possibly more) clocks. The effects of these include both delay measurement errors and poor playout buffer performance. A number of approaches aimed specifically at audio clock skew detection were reviewed before introducing and describing the NTP/RTCP-based method which resolves both system and audio clock skew at both ends. Although all adaptive playout approaches will be affected by clock skew, the issue is of greater consequence for the hybrid playout strategy due to its requirement for precise and actual delay measurement. On the other hand all VoIP and more generally multimedia-based applications can be affected by audio clock skew through poor buffer
performance.

Both the hybrid playout strategy and skew detection method utilise NTP and RTCP SR packets and thus can easily be integrated. Positive results were presented for the NTP/RTCP approach and its operation was compared with alternative approaches. The NTP/RTCP method operates at a high level by integrating existing protocols and does not require low level system calls. It is unaffected by network jitter and resolves all four clock skew rates in a matter of seconds.

The operation of the NTP/RTCP skew detection method is very much dependent on the precision of the RTCP SR packet generation process. As such issues such as sound card drivers, system time resolution and NTP OS dependency can undermine its performance. The second of these issues in particular relates to Windows OS and thus tests were carried out using the Linux OS. Many of these issues however are of general concern to all VoIP applications and not just to the operation of the skew detection method. Finally, the recent suggestion that audio cards use separate clocks for playout and recording, if true will add a further layer of complexity to the problem.
Chapter 8

Conclusions & Future Work

Ever since the idea of using packet-based networks for voice first emerged, concerns over Quality of Service (QoS) have arisen. Most commentators agree that VoIP will not achieve significant market penetration unless it can meet and surpass the QoS that users have become accustomed to with the traditional circuit switched POTS. Researchers have attempted to address QoS concerns through a variety of strategies implemented within sender/receiver endpoints and/or within the underlying network.

The primary objective of this thesis was to assess the benefits and issues arising from the use of synchronised time within Voice over IP (VoIP) applications. The hybrid playout strategy uses precise delay information to select between a fixed or adaptive playout approach, implementing the former whenever possible whilst combining the useful characteristics of both. Both the technical feasibility of implementing the hybrid playout strategy within a VoIP application and the extent to which such an approach can deliver QoS improvements were addressed. Regarding feasibility, results showed that with careful design of a synchronisation subnet, the required NTP performance can be delivered. Furthermore, E-Model based results using a variety of approaches showed that the QoS benefits of the hybrid approach are substantial and can be widely achieved. An analysis of third party Internet delay studies further confirmed the applicability of the hybrid approach.

The move towards Internet multimedia in general raises a number of issues relating to endpoint hardware and software. A number of such issues were examined but particularly that of clock skew, focusing on the implications for the hybrid playout strategy. A high level solution to skew detection was proposed and tested. This solution is based on the novel use of NTP and RTCP protocols and can be integrated easily within the hybrid
playout strategy. Many of these endpoint issues including skew are however of concern to all Internet multimedia applications.

8.1 Main Contributions of This Work

The key contributions of this thesis are as follows:

1. With careful NTP subnet design, synchronised time can be implemented cost effectively across WANs, delivering single figure msec synchronisation. This enables multimedia applications such as VoIP to determine precise delay information and thus make more informed decisions regarding playout strategy.

2. Through use of synchronised time, the proposed hybrid playout strategy has access to actual delay measurements. An analysis of both measured delay data and third party Internet delay studies suggest that although jitter remains a significant problem, delays are very often within the G.114 limit of 150 msec. The hybrid strategy implements an informed fixed delay playout approach whenever possible that in light of delay characteristics, can deliver significant benefits. Conventional adaptive playout strategies are most useful when receivers have no knowledge of end-to-end delays or when delays are excessive. However they can often result in unnecessary late loss and add inherent temporal distortion to speech.

3. Hardware and software issues within endpoints can have a significant impact on end-to-end QoS. As such, any advantages gained through QoS measures in the network or advanced receiver playout strategies may be undone by poor endpoint implementations. In this thesis a range of endpoint issues were reviewed. Chief amongst them was that of clock skew which can affect the buffer performance of all VoIP applications, and delay measurement required for both conventional adaptive playout and the hybrid playout approach.

The combined NTP/RTCP skew detection approach proposed and tested in this thesis offers a high level solution that integrates well with the hybrid playout strategy. Unlike alternative approaches, it is not affected by network noise and thus does not require tuning. It determines skew rapidly which is a critical requirement for VoIP applications.

171
4. Although a significant body of research exists relating to playout strategies, much of which is referenced in this thesis, the scope of such research generally does not extend to implementation issues such as those caused by endpoints. Similarly, a separate body of research has examined endpoint complexity issues and in particular that of clock skew. This thesis examines both issues and proposes a hybrid playout strategy with integrated skew detection, both of which are based on the combined use of NTP and RTCP protocols. As such, the availability of synchronised time facilitates the hybrid playout strategy but to implement such a strategy effectively requires the endpoint issue of clock skew to be addressed in parallel.

8.1.1 Secondary Contributions

In addition to the above, the following represent less significant contributions to the general area of VoIP:

- An additional benefit of the hybrid playout strategy is that through providing precise delay information, it facilitates the coupling together of strategies to deal with both network jitter and loss. Such coupled strategies have been shown elsewhere to deliver further QoS gains.

- Although the E-Model has in recent years been developed by researchers for VoIP, much of this work has been done without due regard for its limitations. In this thesis, the key limitations of the E-Model were identified and from collaborative work with ETSI, concerns outlined regarding its application. The particular implementation of the E-Model in assessing the hybrid playout strategy was approved by ETSI.

- For many of the adaptive approaches reviewed in chapter 2, actual trace data was used for relative performance comparisons. In [8], the dangers of relying solely on trace data for simulation or modelling are outlined. For this reason, a combination of trace data and general delay models was used to assess the hybrid playout strategy.

8.2 Future Work

Outlined below are a range of issues that though outside the scope of the thesis would nonetheless contribute to varying degrees to the research area.

172
8.2.1 Hybrid Playout Strategy Evaluation

- **Tuning Parameters:** An analysis of the effect of varying the hybrid tuning parameters such as histogram size, the weight factor $W_f$ and the intervals for playout evaluation might yield some useful insights.

- **Captured and Third Party Delay Data:** In general the network paths for which trace data was analysed (either captured or third party studies) were well provisioned links tending to reside in *backbone* rather than *edge* networks. This was particularly so for the AMP project data. Further Internet delay studies incorporating more *edge* networks would paint a broader picture.

- **Delay Modelling:** Although an extensive range of 2-state Markov delay models were developed for hybrid strategy evaluation, these models were relatively simplistic and thus more complex models would help further assess the applicability of the approach.

- **Voice Modelling and E-Model Analysis:** The voice model developed for the simulator approach, and the actual voice streams used in the WAN testbed and emulator approaches were based on a medium level of speech interactivity. It would be useful to model speech with differing interactivity requirements (Task 1 to 6) and see the effect of increased talkspurts on adaptive playout performance. Similarly, the E-Model delay impairment curve would need to reflect this change.

The E-Model analysis was based on a conservative delay impairment curve whereas the loss curve was based on bursty loss. Although the late loss pattern was indeed shown to be bursty, it would nonetheless be useful to test the sensitivity of this analysis to changes in such curves.

On a related note, collaborative work with ETSI is ongoing to further improve the E-Model and its usefulness in packet based networks.

8.2.2 NTP Performance

The performance of NTP across the HEAnet WAN was quite impressive. Nonetheless the Irish NTP subnet was relatively weak in that it relied on many overseas servers. Work is ongoing with HEAnet to remedy this situation by increasing the number of stratum 1 & 2 servers available within HEAnet. Furthermore, the NUI,G stratum 1 server is now
available as a public server on the NTP website. Finally more detailed analysis of OS dependency issues would be useful.

8.2.3 Endpoint Issues

Undoubtedly, this is an area requiring significant research. A range of issues were identified in the course of this work but other than skew, the analysis was not carried out in any great depth.

On a related note, both the WAN and emulator testbeds used to assess the hybrid playout strategy were implemented on Linux platforms. Although some of the relevant Windows limitations were identified, it would be useful to examine these in more detail.
Bibliography


176


178


179


183


185
Appendix A

Adaptive Playout Approaches

This appendix reviews in detail various adaptive playout approaches that have been proposed for VoIP. It first describes those approaches that compensate for jitter only. It then looks at approaches that attempt to couple together jitter and packet loss compensation.

A.1 Adaptive Approaches for Jitter Compensation

A.1.1 Per-Talkspurt Reactive Approaches

Algorithms A, A’& B

Ramjee et al. [45] compared four playout delay adjustment algorithms. A simulator was used to compare the performance of all four using captured delay data from diverse paths in the Internet. Two of these algorithms (denoted Alg. 1 and 4 in [45] and known as Alg. A and B respectively in this thesis) performed best and utilise linear recursive filters. They differ primarily in that Alg. B additionally employs a spike detection mode and also that the tuning characteristics are different. Spikes or compression events have been widely reported in the literature on delay studies [56], [48]. In particular Bolot [56] outlines a likely scenario resulting in such spikes, whereby a number of packets are delayed at a network node due to temporary congestion, resulting in a queue buildup.
Once congestion has eased, the packets are rapidly processed and dispatched resulting in the near-simultaneous arrival at the receiver.

In non-spike mode, both algorithms track network conditions using a mechanism similar to that used for the TCP retransmit timer [65] (see equations A.1 and A.2) and adjusts playout time accordingly at the start of a talkspurt as given by eqn.A.3.

\[
\hat{d}_i = \alpha \cdot \hat{d}_{i-1} + (1 - \alpha) \cdot n_i \tag{A.1}
\]
\[
\hat{v}_i = \alpha \cdot \hat{v}_{i-1} + (1 - \alpha) \cdot \text{abs}(\hat{d}_i - n_i) \tag{A.2}
\]
\[
p_i = t_i + \hat{d}_i + \beta \cdot \hat{v}_i \tag{A.3}
\]

In the above, \(i\) refers to packet \(i\), \(\hat{d}_i\) is the estimated end-to-end delay, \(\alpha\) is the filter gain, \(n_i\) is the measured delay, \(p_i\) is the playout time, \(t_i\) is the send time, \(\hat{v}_i\) is the estimated variation in delay and \(\beta\) is a multiplication factor. The choice of parameter \(\alpha\) dictates the extent to which the estimated delay is governed by past or recent values whereas the factor \(\beta\) imposes a safety margin above the delay estimate proportional to the extent of jitter in the network.

A spike is detected when the delay between consecutive packet arrivals exceeds a certain threshold. In spike mode, the delay estimate \(\hat{d}_i\) directly follows the characteristics of the spike as follows:

\[
\hat{d}_i = \hat{d}_{i-1} + n_i - n_{i-1} \tag{A.4}
\]

In spike mode therefore, the playout time (A.3) follows the spike thus improving the response of the playout algorithm which results in less late packet loss. Spike mode is exited when a variable \(\text{var}\) decays below a certain threshold according to equation A.5.

\[
\text{var} = \text{var}/2 + \text{abs}(2n_i - n_{i-1} - n_{i-2}) \tag{A.5}
\]

During a spike, due to the near simultaneous arrival of packets, \(n_i\), \(n_{i-1}\), and \(n_{i-2}\) will differ by an amount equal almost to the packet generation interval and thus the term \(\text{abs}(2n_i - n_{i-1} - n_{i-2})\) will be large. Once the spike has passed, the delays \(n_i\), \(n_{i-1}\), and \(n_{i-2}\) will tend to converge, presuming packet delays are roughly similar, and thus \(\text{abs}(2n_i - n_{i-1} - n_{i-2})\) will approach zero.

Results in [45] indicated that Alg. B will outperform A where significant jitter is present in the network. Note however that where a spike is completely contained within
a single talkspurt, the spike detection mechanism will fail to react. In appendix B.2 a review of the literature relating to voice modelling outlines that talkspurts with hangover time are reasonably well modelled by an exponential distribution with mean duration of approximately 1 second. Furthermore Moon et al. [48] report from network traces that short spikes are not uncommon and thus will often be contained within single talkspurts. In general however, under conditions of significant jitter, Alg. A reacts slowly resulting in higher late losses than Alg. B. On the other hand Alg. B can track the network too closely, particularly at the end of a spike resulting in unnecessary late losses. This is due both to equation A.4 and also to the lower $\alpha$ value which gives more weight to more recent delays.

More generally, it is clear to see that the choice of parameters such as $\alpha$, $\beta$, spike threshold and spike exit threshold will impact greatly on the performance of these algorithms. In [45], $\alpha$ was set to 0.998002 for Alg. A and 0.875 for Alg. B whereas $\beta$ was set to 4 for both. Regarding Alg. B, the spike threshold was set to greater than 100 msec (actually set to $2 \times \hat{v}_i + 100$) and spike exit was set to approximately 8 msec. The improvement resulting from Alg. B under significant jitter, reported by [45] was based on the correct spike threshold settings for the particular network characteristics. Note that adjusting on a per-talkspurt basis maintains the integrity of speech within talkspurts whilst altering the inter-talkspurt silence periods. Although [45] refer to Montgomery [44] in claiming that such distortion of silence periods does not impact noticeably on speech quality, the latter does not cite any specific work to back up the claim.

Very recent work by Narbutt et al. [46] examined the effect of tuning parameters such as $\alpha$ on the performance of Alg. A. This showed that there is no fixed value of $\alpha$ that works well under all conditions. For example, high values of $\alpha$ are appropriate only where delay values do not change significantly over the course of a session but jitter is significant. On the other hand, where jitter is minimal but delay varies significantly, low values of $\alpha$ (eg. $\alpha < 0.9$) performed best. This approach labelled Alg. A’ here recomputes $\alpha$ with each incoming packet using some function of the mean variation, as shown by equation A.6.

$$\alpha_i = fn(\hat{v}_i) \quad (A.6)$$

Performance comparisons were made with Alg. A, configured with a range of $\alpha$ val-
ues, fixed for the duration of each session. Both network simulation and actual delay measurements were used. Results indicated that the dynamic approach achieves a delay/late loss tradeoff that matches the best performance from the range of fixed $\alpha$ values under diverse network conditions. Work by Tobagi et al. [12] discussed later in this section briefly report from similar analysis that although average performance improved, periods of significant loss were reported. No other details are provided however which makes detailed comparisons impossible.

### A.1.2 Per Talkspurt Predictive Approaches

Algorithms A, A’ and B are reactive in that they track changing network conditions using an $\alpha$ factor to strike a balance between recent and historical data. In [52], Sreenan et al. make the distinction between such reactive approaches and predictive approaches. They outline that reactive approaches can result in widely varying buffer sizes and consequent delays. Predictive approaches on the other hand use past data to make predictions about future delays, enabling the user to better control the trade-off between late loss and delay. They outline that with loss recovery techniques such as FEC and PLC, many audio applications can cope with significant loss and thus the latter predictive approach can be used to minimise delays which is a critical requirement for interactive multimedia applications. A number of such predictive approaches are now reviewed.

#### Algorithm C

Moon, Kurose and Towsley proposed such an approach in [48] (denoted Alg. C in this thesis) whereby estimated delay percentile information for 10000 packets is maintained and updated with each received packet and used to dynamically predict and adjust playout delay, again on a per-talkspurt basis. The choice of 10000 was based on tests; values above 10000 were found to be of little benefit regarding improved delay-loss results whilst adding to the space complexity. Values below 10000 resulted in myopic performance. They also incorporated a spike detection mode though its implementation details are very different. In spike mode, the delay of the first packet in a talkspurt is used for playout delay, on the basis that subsequent packets will have a
lower playout delay. Regarding the detection of a spike, they employed a head and tail multiplier to detect the beginning and end of spike mode. They experimented with ranges of values for both to assess the algorithms sensitivity and chose values of 4 and 2. An approach similar to [45] was adopted in that a simulator was used to compare their approach with that of Alg. A and B. Delay traces, including some from [45] were used and performance was assessed by comparing average playout delay with late loss rates. In the case of Alg. A and B, the playout delay point was controlled by varying the value of $\beta$ in equation A.3 from 1 to 20. Their main findings were:

- Their approach Alg. C outperformed both A and B in many of the tests. Alg. A performed best on certain traces though little detail is provided other than high loss rates were encountered. They suggest that Alg. A is best suited to slowly changing conditions and reacts too slowly to sudden network variations.
- Alg. B rarely outperformed the other approaches and more generally for a given delay, resulted in higher loss rates.
- By focusing on the operating points that corresponding to a $\beta$ value of 4, and ignoring the high loss rate traces, useful comparisons with Ramjee et al. [45] can be made. Of the remaining four traces (Fig, 12,15,16,17 in [48]), the performance gains of Alg. C relative to Alg. A in particular were marginal in that for the same delay, loss rates were reduced by margins of less than 1%. For roughly the same delay, Alg. B generally resulted in 2% higher loss rate.

In general, the importance of algorithm tuning is again very evident. In addition to the spike thresholds, the choice of 10000 as the extent of history is interesting. Note that the packet size used was 10 msec and thus this equates to 200 seconds of data, presuming no silence detection. With a larger packet size and silence suppression, the requirement for 10000 packets could equate to 600 or more seconds. Considering that in other studies such as [66], call durations of between 3 and 7 minutes are considered typical, this requirement could be problematic. Furthermore, in comparison tests by Rosenberg et al. [54], detailed later in this section, a window size of 10000 was considered too large and a value of 1000 was instead chosen.

Moon et al. also investigated the sensitivity of Alg. A and B to the $\alpha$ value and found that values in the range 0.9 $\leq \alpha \leq 0.999$ did not greatly affect performance.
Values less than 0.90 were found however to degrade performance which, as suggested above, explains to some extent the relatively poor performance of Alg. B. However as shown by [46], low values may be appropriate under variable delay conditions. Finally Moon et al. refer to [44] in claiming that distortion of silence periods does not impact noticeably on speech quality.

In [49], an approach somewhat similar to [48] was proposed by Pinto et al. whereby a target loss rate is specified and playout delay is predicted based on past delay values. Essentially, in determining the playout delay for a given talkspurt $k$, a detailed analysis is carried out on the previous talkspurt $(k - 1)$ to determine what the playout delay should have been to meet the target late loss rate. The difference between this optimum value and the actual playout delay is known as the optimum gap ($Opt_{\text{gap}}^{k-1}$) which is incorporated into the buffering delay $\rho$ for the current talkspurt, as shown by equations A.7 and A.8.

$$\rho = \rho + Opt_{\text{gap}}^{k-1}$$  \hfill (A.7)

$$p_i^k = t_i^k + d_i^k + \rho$$  \hfill (A.8)

Comparisons were made via simulation of actual traces with Alg. A and B. As with [48], different late loss rates for Alg. A & B were generated by varying the value of $\beta$.

Results were mixed and were highly dependent on the characteristics of the traces used for simulation. Furthermore the mechanism was unable to reduce loss rates below 4\% approximately. The reliance on data from a single talkspurt to recalculate playout times according to a target loss rate is a possible explanation for this.

**Algorithm D1 & D2**

In [50], DeLeon and Sreenan used a normalised least-mean-square (NLMS) adaptive predictor labelled Alg. D1 here to predict delay $\hat{n}_i$ given the previous 18 packet delay estimates. This value was then used within the reactive approach of Alg. A in [45] in place of $\hat{d}_i$ as indicated by equation A.9.

$$p_i = t_i + \hat{n}_i + \beta \ast \hat{v}_i$$  \hfill (A.9)
Note that the definition of \( \hat{v}_i \) is similar to that in equation A.2.

A series of results were presented which compared the performance of Alg. D1 with that of Alg. A. The first set compared the average end-to-end delay or \( ted \) for roughly equivalent late loss rates. Alg. D1 generally resulted in a 10 msec lower average \( ted \) though late loss rates were slightly higher. The second set compared late loss rates for roughly equivalent average \( ted \). As with [48], delays for Alg.A were reduced by altering the \( \beta \) value (from 4 to 2.6). In this case, Alg. D1 resulted in marginally lower loss rates over most of the traces (but also had slightly lower average delay).

In further work by Agrawal, Chen and Sreenan [51] [52], the Concord algorithm labelled Alg. D2 here was presented and tested. This predictive approach involves constructing a Packet Delay Distribution or PDD for a media stream and defining two stream requirements, maximum acceptable delay or \( mad \) and maximum late packets or \( mlp \). The PDD is updated with each packet data and the overall requirement is that the \( ted \) is determined for each packet such that:

1. The chosen \( ted \) is less than \( mad \) for the stream, and
2. The chosen \( ted \) does not lead to more than \( mlp \) packets being lost.

The cumulative distribution function for the PDD is drawn and \( ted \) can be chosen to meet either \( mlp \) requirements or to minimise \( ted \).

A series of test results were presented which compared Alg. D2 (Concord) with Alg. B and a fixed playout approach. As with results in [42] detailed later, the inflexible nature of the fixed playout strategy resulted in loss rates varying from 0.2% to 94% depending on the trace characteristics. Comparing Alg. D2 with B, the former resulted in a much lower range of playout delays, reflecting the large sample space used to predict playout delay. Although the mean \( ted \) values for D2 and B were similar, Alg. B had much higher maximum, and much lower minimum delays. On the other hand, Alg. D2 suffered from higher loss rates (which is part of the design) but more significantly suffered from higher burst loss rates reflecting its longer term approach and insensitivity to very recent events. The lower range of \( ted \) values of Alg. D2 simplify the implementation of the receiver buffer.

In [52] the Concorde work was extended by examining the effect of applying different ageing techniques to the delay histograms. The idea was to ensure that as a media
session progresses, the playout ted was influenced more by recent delays than by previous delays. Three ageing techniques were applied to Alg. D2 and performance was compared to that of Alg. B and D2 (without ageing). The main findings were as follows:

- The lower the ageing coefficient (0 – 1), the less influence that older packets exert and the more reactive the algorithm becomes. This results in a greater range of ted values and lower average ted values.
- Similarly, the more frequent that ageing is applied, the quicker than older packets lose their influence. Interestingly, results indicate that increasing the ageing intervals beyond 100 packets had little additional impact regardless of the coefficient.

Summarising this work, Alg. D1 presented small performance gains over Alg. B. Alg. D2 (without ageing) resulted in a much lower range of ted values which simplifies buffer implementation. The average ted was however unchanged. Alg. D2 was designed to implement a minimal ted to satisfy acceptable delay requirement mad by allowing a certain late loss rate mlp. Its lack of sensitivity to recent network events can result in high burst loss rates where significant delay jitter occurs. Employing ageing techniques improved the responsiveness and lowered average ted values at the expense of a higher range of ted values. The ageing interval needs to be less than 100 packets to have a significant effect.

A.1.3 Per-Packet Based Predictive Approach

Algorithm E & C’

More recent work by Liang, Farber and Girod [53] [42], labelled Alg. E here, proposed yet another mechanism that differs significantly from the above. Similar to Algorithms C and D2, it maintains a continuously updated histogram of previous estimated packet delays to predict future playout delay. In contrast to Algorithms A, B, C and D1, the playout adjustment of Alg. E is made both by silence period adjustment between talkspurts and on a per-packet basis within talkspurts (Details of the adjustment
mechanism for Alg. D2 are not provided in [52]). [53] outlines and describes how per-talkspurt algorithms will fail to react to short spikes where such spikes are contained within a single talkspurt, an issue already discussed above in appendix A.1.1. As such, talkspurts can span multiple packets and thus adjusting on a per-talkspurt basis can result in higher late losses. Adapting on a per packet basis overcomes this limitation resulting in a more responsive algorithm.

For a given incoming packet \( p_i \), the mechanism estimates the delay encountered by \( p_i \) and predicts the delay for the next packet \( p_{i+1} \). It then scales the duration of \( p_i \) such that if the prediction is correct, \( p_{i+1} \) will arrive just as \( p_i \) has been played out. This scaling is achieved using a technique based on the Waveform Similarity Overlap-Add (WSOLA) algorithm. Essentially it selects a template segment in the sample to be scaled and then defines a search region where it looks for a similar segment. It uses weighting windows to blend the similar and template segments so as to achieve the modified length. A critical factor is that the mechanism preserves the pitch period (unlike mechanisms based on altering sampling rates to speed up or slow down audio such as [11]). Furthermore, regarding elongation, it ensures that the added pitch periods are not simple replications and insertions of other similar pitch periods but rather are based on interpolation which affects a number of periods in the elongated segment. Similarly with compression, the dropped pitch period is blended via interpolation into the shortened segment. In [42], the use of this technique for PLC in addition to playout scheduling is evaluated with positive results.

In predicting the playout time of \( p_{i+1} \), it maintains a histogram of previous packet delays similar to Alg. C and D2 and predicts delay based on a target loss rate. Unlike Alg. C, the histogram stores delay data for the previous 35 packets as opposed to 10000 packets. Note that Alg. D2 stores all previous delay values but only in statistical form thus simplifying the space complexity. Ageing techniques within D2 however reduce the weight of older packets. Recall also that Alg. D1 uses the previous 18 packets in predicting future delays. The value of 35 was chosen from experimentation with specific trace data. Hysteresis is built into the mechanism by ensuring that it responds more quickly to rising than to falling delays. Finally, a spike detection mechanism is also employed.

Comparisons were made with a fixed delay strategy and a per-talkspurt strategy sim-
ilar to Alg. C above [48]. Regarding the latter, the histogram size was set to 300 and not 10000, based on experimentation and thus is labelled as Alg. C’ here to distinguish it from Alg. C. The Network Time Protocol was used to determine actual delays for implementing the fixed delay strategy. Trace data from a series of tests within the US and between the US and China/Germany was gathered and used for comparison tests. Interestingly, delay jitter was very high on one internal US links and the US-Germany link (Traces 1&2 respectively) relative to the US-China link (Trace 4). Results are presented using both average buffering delay versus loss rate curves and controlled subjective testing. Their main findings were:

- The fixed delay algorithm performed worst of the three approaches.
- Alg. E outperformed Alg. C’ in all tests. For a given average buffering delay of 40 msec, it resulted in lower loss rates from 0% (Trace 4) to 10% (Trace 1). Similarly, for a given late loss rate of 5%, it resulted in lower average buffering delays of 4 to 30 msec.
- Results were also presented indicating the reduction in burst loss rate resulting from Alg. E. A burst loss is defined as two consecutive packets that arrive too late for playout. Improvements using burst loss metric were more spectacular.
- From subjective testing, they report that scaling of packets resulted in little degradation of audio quality. A 0.3 – 0.5 score using the DMOS (Degradation Mean Opinion Score) was reported using the DCR (Degradation Category Rating) Method [67] though they qualify this by noting that scaling occurred infrequently during the reported tests (17-24% of packets). Results from MOS testing indicated an improvement of 0.4 – 1 in MOS relative to Alg. C’.

Certainly, the results were impressive and suggest that scaling has many benefits with little degradation in quality. Other than Trace 3, the extent of jitter was however very significant and thus the benefits of Alg. E in responding within talkspurts was shown in the best possible light. The trace data gathered in this thesis and presented in section 6.2.1 and indeed the third party delay studies in section 6.5 suggest that such extreme jitter values may not be that common. The time complexity of the WSOLA method is also fairly significant and though data is presented for one scenario, it would be interesting to note the performance under more loaded OS conditions. The
choice of 35 and 300 as histogram size for Alg. D and C’ respectively were based on experimentation. Recall from [48] that a histogram size for Alg. C less than 10000 was deemed to be \textit{myopic} and performed poorly. No explanation is given for the order of magnitude of difference and in the absence of more information, this suggests the extreme importance of network tuning.

\section{A.2 Adaptive Approaches for Jitter and Loss Compensation}

\subsection{A.2.1 Network-Unaware FEC}

In [54], a range of algorithms were presented by Rosenberg at al that provide \textit{coupling} between the playout and FEC strategies. Of particular interest here are the \textit{Virtual Delay} and \textit{Previous Optimal} algorithms. Comparisons were made with uncoupled versions of Alg. A, B and C as described above. Note that with regard to Alg. C, they used a window size of 1000 as they report that a size of 10000 was found to be too large, with little consequent responsiveness. As such their implementation is labelled Alg. C” here to distinguish it from Alg.C’ described above.

Two \textit{uncoupled} versions were actually used for comparison: the first implemented Alg. A, B and C” without any additional delay for the FEC mechanism. The second added a fixed 80 msec delay to that determined from the playout strategy alone. Regarding the \textit{coupled} approach, the concept of \textit{Virtual Delay} or \(d_v\) for packet \(p_i\) was introduced by [54]. It is defined as the difference between the earlier of the arrive time or recovery time and the send time for packet \(p_i\). If out-of-sequence packets are ignored, and \(p_i\) is not lost in the network, the recovery time (from FEC) will always be after arrival time and thus \(d_v\) is the normal measured delay. Where \(p_i\) is lost but is recovered through FEC applied to subsequent packets, \(d_v\) is the difference between recovered time and send time. In the \textit{Virtual Delay} approach, Alg. A and B and C are implemented using virtual delays \(d_v\) rather than the measured estimates of delays. As such the \textit{Virtual Delay} approach will result in higher playout delays where packets are lost and recovered through FEC.
As outlined previously, an important point regarding multimedia applications is that a certain degree of link loss is acceptable. This remains the case even if FEC techniques cannot recover the lost data as PLC can still be used and in any event, a small residual loss may not significantly affect intelligibility. [54] outlines how the Virtual Delay approach can be used to target a certain overall (i.e. after FEC) loss rate $p_T$ for Algorithms A & B. Such a target may not be attainable due to very high link losses and thus the achievable overall loss rate $p_C$ is also introduced.

A running estimate of link loss rate $\hat{p}$ is maintained and at the end of each talkspurt, is used to compute $p_C$. The actual overall loss rate $p_L$ is also calculated and the $\beta$ multiplier from equation A.3 is varied accordingly as shown by the following code segment.

\[
\begin{align*}
&\text{if}(p_C < p_L - \theta) \\
&\quad \beta = \beta + \delta_{\text{inc}} \\
&\text{elseif}(p_C > p_L + \theta) \\
&\quad \beta = \beta - \delta_{\text{dec}} \\
&\text{else} \\
&\quad \beta = \beta
\end{align*}
\]

The above analysis is simplified from [54]. $\theta$ is a threshold which introduces hysteresis whereas $\delta_{\text{inc}}$ and $\delta_{\text{dec}}$ specify the rate at which the algorithm adapts to changing conditions. As seen with other approaches such as [53], this ensures that the algorithm responds more quickly to increasing that decreasing delays. Finally, floor and ceiling values are also applied to $\beta$.

The Previous Optimal approach is similar in that it also uses virtual delays. It differs in that it determines the optimal playout delay (based on target overall loss rate) for the previous talkspurt $D_{\text{opt}}$ and uses this to determine playout delay for the next talkspurt $D_w$ as follows:

\[
D_w = \rho D_{w-1} + (1 - \rho) D_{\text{opt}} \tag{A.10}
\]

In the above $\rho$ was set to 0.25.
For tests, *link* loss rates were varied by manipulating actual trace conditions and the three algorithms (third not shown here) were compared with the two uncoupled versions of Alg. A, B and C”.

Results are summarised as follows:

- As expected, the original Alg. A, B and C” with no added delays for FEC did not respond in anyway to increasing link loss rates and overall packet loss rates increased with link loss rate.

- The uncoupled versions that add a fixed 80 msec delay to that determined from playout algorithms alone resulted in much lower overall losses at the expense of much higher delays.

- The *Virtual Delay* approach resulted in much lower delays, particularly at the lower link loss rates than the uncoupled version (with 80 msec) whilst matching it in terms of overall loss rate.

- The *Previous Optimal* approach also performed quite well and indeed was better at reaching target loss rates.

In summary, [54] shows that by adapting a *coupled* approach to FEC and playout delay, overall loss rates equivalent to a *decoupled* approach with fixed FEC additional delay can be achieved, *without* the delay overhead, particularly at the lower link loss rates. Note however that though this approach provides coupling, the sender where FEC is implemented is unaware of network characteristics (loss and delay). The following section outlines a further development of this work.

### A.2.2 Network-Aware FEC Strategies

Boutremans et al. [43] further developed the work of Rosenberg et al.. The latter’s *coupled* approach results in a *play first strategy* at the receiver in that *virtual delays* use either the original packet or the recovered version. The FEC strategy implemented at the sender is unaware of network loss characteristics or the playout delay determined at the receiver. For very low link loss rates, this will result in unnecessary additional delay whilst the receiver waits for the FEC data for recovery. If overall M2E delays are already high, this may push them beyond the G.114 limit. A more effective approach is
to implement an FEC strategy that is aware of both playout delay being implemented at the receiver and network loss characteristics.

Two different solutions were presented and tested. The first, called \( N1 \) is similar to that just described whereby the playout delay is determined separately (for example using approaches outlined in [54]) but the sender is made aware of this information as well as network loss characteristics. This information is used to select the optimal FEC strategy to maximise playout quality (Eg. if loss rates are high but delays are low, then the sender can afford to implement multiple redundancy FEC).

The second approach, called \( N2 \) is more complex in that the selection of playout delay and FEC strategy are considered together at the sender in order to maximise playout quality. Comparisons were made with two combinations of existing FEC and playout delay strategies, known as \( O1 \) and \( O2 \). The following summarises the four approaches:

- \( O1 \): Decoupled approach whereby playout strategy was the original Alg. A and FEC strategy was delay aware.
- \( O2 \): Coupled approach whereby playout strategy was the virtualised Alg. A as described by [54] and sender FEC strategy was delay and network-unaware i.e. one of the approaches in [54].
- \( N1 \): Proposed method 1 similar to \( O2 \) except that sender FEC was aware of playout delay and network loss characteristics.
- \( N2 \): Proposed method 2 whereby both playout delay and FEC strategies were considered together at the source as parameters in an optimisation problem.

Although not of particular relevance to this section [43] also examined the effect of encoding rate in addition to network loss and delay on playout quality (i.e. lower codec rates tend to have lower quality). As such they also incorporated into testing a TCP-friendly rate control module, based on the use of RTCP receiver reports (RR). Testing was carried out via simulation across a bottleneck link with set bandwidth and transmission delay and the number of connections across the link was varied. To quantify the relative performance of the various approaches a complex expression for the ITU-T E-model R-factor was derived as a function of the encoding rate and packet loss rate. In assessing the impact of delay on R-factor, they used three different delay distortion functions corresponding to different modes of interactivity. This aspect of

199
their work is discussed in greater detail in section 5.2. A more thorough analysis of this work is provided in [61].

The most relevant results from tests were as follows:

- Method \(N2\) performed best by minimising delays and overall packet loss. This was particularly so for strict interactivity requirements where additional delays impact significantly on R-factor.

- Method \(N1\) performed next best and better than \(O2\). The only difference between them is that \(N1\) employs delay and network \textit{aware} FEC. As such the extent of redundancy within \(O2\) FEC is critical: a \(k = 1\) value indicating single redundancy performed very poorly (almost linearly) as expected when link loss rates increased.

- Decoupling within method \(O1\) meant that playout was determined in the absence of any consideration for FEC and thus FEC measures were largely wasted.
Appendix B

Hybrid Assessment Methodology

This appendix provides details on the ITU-T E-Model for voice transmission quality evaluation. It also reviews the areas of voice modelling and Voice Activity Detection (VAD) within codecs.

B.1 E-Model Description

The transmission rating factor $R$ is defined by equation B.1.

$$R = (R_o - I_s) - I_d - I_e + A$$  \hspace{1cm} (B.1)

The factor $R_o$ represents noise, both within the circuit and external to it. $I_s$ includes impairments such as quantisation distortion, loudness ratings and non-optimum sidetone. The $A$ or *Advantage* factor represents the quality penalty users will accept for service in remote areas or whilst on the move. Provisional $A$ factors range from 5 for mobile access in buildings to 20 for satellite connections to remote areas. All three are not considered in this analysis.

The factors of particular interest are thus $I_d$ (delay impairment) and $I_e$ (loss impairment). $I_e$ includes the distortion caused by low bit rate codec operation as well as the effect of packet loss (both link and late loss).
B.1.1 Delay Impairment Id

Impairment due to delay is given by three factors as per equation B.2.

\[ I_d = I_{dte} + I_{dle} + I_{dt} \]  \hspace{1cm} (B.2)

The first two terms capture the impairment due to talker and listener echo respectively. An echo loss value of 51 dB is considered effective echo cancellation. The third term captures the interactivity impairment that ITU-T recommendation G.114 refers to. Overall delay impairment \( I_d \) is strongly influenced by the echo loss values as Fig. B.1, taken from [95] illustrates. With perfect echo cancellation, there is no loss in quality for delays of up to 150 msec. On the other hand, for severe echo loss values of 20 dB, R factor drops below 70 at a M2E delay of 25 msec.

For reasonable echo cancellation such as 50 dB the slope of the impairment line is approximately 5 units per 100 msec delay for delays less than 150 msec and a more severe 12 – 13 units per 100 msec above this. As such, there is a much greater incentive to reduce delays to 150 msec rather than below 150 msec. This further reinforces the rationale for the hybrid playout strategy relative to conventional adaptive approaches in that the latter attempt to minimise delays at the expense of increased late losses regardless of actual delays whereas the former tends to increase delay towards the 150 msec limit and in so doing, minimise late losses.

B.1.2 Loss Impairment Ie

This impairment captures distortion due to the use of low bit rate codecs and packet losses (either link or late loss). Recommendation G.113 [93] [94] deals with \( I_e \), outlining provisional values for \( I_e \) under no loss conditions for a variety of codecs and separately outlines limited loss impairment information for a small number of codecs, under both random and (for G.711 only) bursty packet loss. In Fig. B.2, taken from [95], the experimental data from [93] [94] is extrapolated to estimate the impairment for various packet loss rates. These curves are based on specific packetisation intervals of 10 msec for G.711, 20 msec for G.729 and GSM and 30 msec for G.723.1. Note that only the G.711 curves originate at the origin; the others all intersect the Y-axis at non-zero impairment values. This reflects the underlying loss associated with each codec.
Figure B.1: Delay Impairment
Figure B.2: Loss Impairment

at zero packet loss rate. Obviously, the higher the impairment at zero loss, the smaller is the *impairment budget* that is available for packet loss and delay impairments, in order to avoid operating at at R-factor below the acceptable POTS level of 70.

The impact of PLC which is built into many low bitrate codecs is very evident from the two G.711 curves (with and without PLC). As noted in section 3.3, predictive type adaptive playout strategies can best take advantage of PLC techniques in that they can target a certain loss rate that is within the range of the PLC technique employed.

For G.711 with PLC and random packet loss, the slope of the curve can be approximated by $3 - 4$ units in R-factor for every % packet loss. For G.711 bursty loss however, the slope increases to approximately 10 units per % between 3% and 5%.
This is explained intuitively by the fact that with bursty loss, there is a greater chance of whole phonemes within speech being lost than with random loss. Both [12] and [23] use bursty loss curves in their analysis. In section 5.3.4, this issue is revisited with a review of research on Internet delay and loss modelling.

Finally [6] outlines a formula for converting from R-factor to MOS and vica versa which is illustrated by Fig B.3.

A critical factor for the hybrid algorithm confirmed by the E-Model is the lower sensitivity of \( I_d \) to increased delay than that of \( I_e \) to packet loss. From a practical viewpoint, this means that users will tolerate increased M2E delays before increased packet loss. As outlined in section 3.6, this justifies the use of the weight factor \( W_f \) in calculating the fixed playout delay from the delay estimate \( est \).

### B.2 Voice Modelling

Although voice modelling is a much researched topic in its own right, the objectives here were to develop a simple model of the human voice for use in the simulator.

Markov processes, with appropriate number of states are known to adequately describe speech. The accuracy of speech models is indicated by the degree to which they accurately predict the length of the ten speech events that can occur. The four most important events from [113] are as follows:

1. Talkspurt
2. Silence Period
3. Doubletalk
4. Mutual Silence

Fig. B.4 outlines a comprehensive 6-state diagram developed by [113] representing a two-way voice conversation. This 6-state diagram can model all of the ten events but comes at the expense of complexity. By eliminating the requirement to model mutual silences and doubletalk, a much simpler 2-state model, shown in Fig. B.5 can be developed. With this model, talkspurts are reasonably well modelled but the distribution of silence periods is not. However, it is much applied due to its simplicity.
Figure B.3: Mapping between MOS and R factor
Six-state Model for Speech

Figure B.4: 6 State Model
Two-state Model for Speech

Voice Activity Detection (VAD) or Silence Detection mechanisms within codecs take advantage of the existence of silence periods in speech. This saves bandwidth (enabling multiplexing) as well as facilitates per-talkspurt adaptive playout strategies, discussed in section 3.3. VAD schemes vary significantly between codecs and often have configurable thresholds resulting in varying distributions of talkspurts/silence periods. In particular codecs implement hangover techniques whereby the codec avoids clipping the end of talkspurts and bridges over very short speech gaps. The ITU-T P.59 [114] recommendation specifies that a model of conversational human speech should have mean talkspurt/silence periods of 227/596 msec respectively without hangover and 1.004/1.587 seconds with hangover. Other work [115] report a range of 200 – 400 / 500–700 msec for mean talkspurt/silence periods without hangover and approximately 1.2/1.8 seconds with hangover. Jiang et al. [116] in a recent study analysed speech characteristics (in both Chinese and English) to assess the extent to which speech deviates from the 2-state model. Their results confirm the sensitivity of talkspurt/silence period distribution to hangover and VAD settings and also that exponential distributions are particularly unsuited to modelling silence periods or gaps.

In the context of conversation Task types introduced in section 5.2.1, little research
exists on the effect of such types on speech models and in particular, the distribution of talkspurts and silence periods.

From a speech quality perspective, the use of VAD schemes, resulting in an ON/OFF packet distribution can seriously degrade receiver voice quality. This is principally due to the use of poor compensating techniques for comfort noise injection at the receiver to mask the effect of silence periods. An example is where at the sender, background music is playing at a level that interferes with the VAD mechanism and distorts the ON/OFF distribution. At the receiver, the actual silence periods within the reconstructed sender’s speech are thus replaced by segments of the background music (corresponding to those packets misinterpreted as sender speech) and comfort noise. This issue is dealt with in greater detail by Gierlich et al. in [117].
Appendix C

Detailed Results

This appendix includes detailed results relating to various aspects of testing from Chapter 6.

C.1 Diurnal Delay Variation

Figures C.1, C.2, and C.3 show diurnal delay patterns to UCL, LKN and ICIR respectively.
Figure C.1: Diurnal Variation in Delay to UCL
Figure C.2: Diurnal Variation in Delay to LKN
Figure C.3: Diurnal Variation in Delay to ICIR (20 sec tests)
C.2 Conditional CDF Results

Figures C.4, C.5, and C.6 show cdf results for UCL, LKN and ICIR respectively.
Figure C.5: Conditional/Unconditional CDF: LKN
Figure C.6: Conditional/Unconditional CDF: ICIR
C.3 Sample Trace Results

Figures C.7, C.8, and C.9 outline a sample test trace to each of UCL, LKN and ICIR.
Figure C.8: Sample Trace Performance: LKN
Figure C.9: Sample Trace Performance: ICIR
Figure C.10: LKN: Delay Model A

C.4 2 State Markov Delay Model Results

C.4.1 LKN-Based Models

Figures C.10 to C.20 show 5000 packet segments of simulated results from the various LKN-based 2-state Markov delay models developed.
Figure C.11: LKN: Delay Model B
Figure C.12: LKN: Delay Model C
Figure C.13: LKN: Delay Model D
Figure C.14: LKN: Delay Model E
Figure C.15: LKN: Delay Model F
Figure C.16: LKN: Delay Model I
Figure C.17: LKN: Delay Model J
Figure C.18: LKN: Delay Model K
Figure C.19: LKN: Delay Model L
LKN: Model M

Figure C.20: LKN: Delay Model M
C.4.2 ICIR-Based Models

Figures C.21 to C.28 show 5000 packet segments of simulated results from the various ICIR-based 2-state Markov delay models developed.
Figure C.22: ICIR: Delay Model B
Figure C.23: ICIR: Delay Model C
Figure C.24: ICIR: Delay Model D
Figure C.25: ICIR: Delay Model E
Figure C.26: ICIR: Delay Model F
Figure C.27: ICIR: Delay Model G
<table>
<thead>
<tr>
<th>Packet Number</th>
<th>Delay (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Arrival</td>
</tr>
<tr>
<td></td>
<td>Playout Modified Hybrid</td>
</tr>
<tr>
<td></td>
<td>Adaptive</td>
</tr>
<tr>
<td></td>
<td>Spike</td>
</tr>
</tbody>
</table>

Figure C.28: ICIR: Delay Model H
Appendix D

AMP Data Analysis

This appendix illustrates delay distributions analysed from data downloaded from the Active Measurement Data (AMP) project.

- Figures D.1, D.2, and D.3 outline the delay distributions for the three European paths from Ireland to Finland, Hungary and Norway respectively.
- Figure D.4 outlines the delay distribution from Ireland to Toyko.
- Figures D.5, D.6, D.7, D.8, D.9,and D.10 outline the delay distributions for the six transatlantic paths from Norway to Columbia University, Florida State University, Stanford, Berkeley, University of Texas and University of Washington respectively.
- Figures D.11, D.12, D.13, and D.14 outline the delay distributions for the four internal US paths from University of Illinois to University of South Florida, from Columbia University to University of Alaska, from Harvard to Berkeley and from University of Alaska to University of South Florida respectively.
amp-heanet to amp-hutf (Path lrl1)

Figure D.1: amp-hean to amp-hutf
Figure D.2: amp-hean to amp-eln
Figure D.3: amp-hean to amp-thor
Figure D.4: amp-hean to amp-apantyo
Figure D.5: amp-thor to amp-columbia
Figure D.6: amp-thor to amp-fsu
amp–thor to amp–stanford (Path Nor3)

Figure D.7: amp-thor to amp-stanford
amp–thor to amp–ucb (Path Nor4)

Figure D.8: amp-thor to amp-ucb
amp–thor to amp–utexas (Path Nor5)

Figure D.9: amp-thor to amp-utexas
Figure D.10: amp-thor to amp-uwashington
Figure D.11: amp-uic to amp-usf
Figure D.12: amp-columbia to amp-alaska
Figure D.13: amp-harv to amp-ucb
Figure D.14: amp-alaska to amp-usf
Appendix E

Skew Detection & Compensation Techniques

This appendix provides details on various approaches to skew detection & compensation, both for delay measurement and for realtime multimedia applications.

E.1 Skew Detection for Delay Measurement

E.1.1 Paxson et al. (1996)

Paxson’s statistical test for skew detection operates as follows.

Basically, if relative skew is present, this will manifest itself as an increasing delay trend in one direction and the opposite trend in the reverse direction. Given a sequence of $n$ packet delays $X_{ti}$, with $(1 \leq i \leq n)$, corresponding to ordered packets, Paxson defined an indicator $I_{tj}$, for identifying a decreasing trend as follows:

$$I_{tj} = 1, \quad \text{if} \ (X_{tj} < \min_{i<j} X_{ti} \quad \text{or} \ (j = 1)$$

$$\text{otherwise} \quad I_{tj} = 0$$

In other words, $I_{tj} = 1$ if $X_{tj}$ represents a new cumulative minimum relative to all packets up to $j$, otherwise $X_{tj} = 0$. 

254
If relative skew leading to a decreasing trend is present, then there will be many more cumulative minima as you examine delays from packet 1 to \( n \) than if there is no skew (delays independent).

When this test is carried out on both sets of data, a series of further heuristic checks are applied. The purpose of these is to isolate network effects, to check whether skew detection in one direction only is compelling enough, to check that the order of magnitude of skew in either direction is roughly similar and generally to add robustness to the mechanism. One particular problem that Paxson encountered was a much greater extent of network jitter and noise in the file transfer direction than in the ACK direction. In fact, in the final analysis, about three-quarters of skews were detected on the basis of reverse ACK measurement data alone.

On a separate issue, Paxson advises caution with regard to reliance on NTP to synchronise clocks. As shown in chapter 4 however, effective NTP performance can be achieved if sufficient consideration is given to subnet design and local reference sources are employed. The decreased cost and consequent increased availability of GPS clocks since Paxson’s work is a key factor in improved NTP performance.

### E.1.2 Moon et al. (1999)

This linear programming approach seeks to place a line under all of the data points such that the sum of vertical distance from the line to the actual points in minimised. They show through actual measurements and simulation that their approach is superior to that of Paxson. For comparison purposes, they modify Paxson’s approach for the situation where delays in one direction only are known. In general, they outline a number of criteria with which to assess any skew detection technique. These include space and time complexity and robustness i.e. the extent to which the error in skew estimate is independent of the magnitude of skew. They show that the linear programming approach outperforms the others. Regarding Paxson’s approach, they show through actual measurements and simulation that unlike their approach, the error in skew estimate increases with the magnitude of skew and also that the estimates derived from Paxson’s approach have a higher variance.
E.1.3 Zhang et al. (2002)

This approach is similar to that of Moon et al other than it describes a hull rather than a line whose lower boundary is composed of line segments with endpoints taken from the dataset. The convex hull approach is used to solve for the line under different conditions, or objective functions. These are to minimise the vertical distance between the line and the data points (which is similar to [125]), to minimise the area between the line and the data points (each two consecutive points treated as a line segment in calculating area) and finally to maximise the number of data points on the line. The significance of the hull is that the optimal line will be below the hull and touch it at some point. The authors claim that their approach performs better than [125] in that it works well in the presence of clock adjustments. The latter can be either step adjustments such as ntpdate or velocity adjustments in the case of NTP. They also outline that the approach works well online in that it builds up the lower hull quickly as measurement points accumulate. However, such an online mechanism will be subject to errors caused by temporary network congestion and thus its effectiveness for realtime applications such as short VoIP sessions will be limited.

E.2 Skew Detection & Compensation for Multimedia Applications

E.2.1 Akester et al. (2002)

In [10], Akester at al determine the relative skew between sender and receiver audio clocks by timestamping the arrival of packets at the receiver using the audio clock and comparing the time between successive incoming packets with the time as indicated by the difference in packet RTP timestamps, which are set by the sender audio clock. This approach yields a ratio:

$$e_o = (c - c_p) / (t - t_p)$$ (E.1)

The receiver audio clock values $c$ and $c_p$ are determined by low level system calls. They describe this mechanism for both a DMA(Direct Memory Access)/ISA (Integrated
Services Architecture) setup and the more common PCI (Peripheral Component Interconnect) Bus Mastering setup. The use of the audio clock for timestamping differs from the hybrid strategy, and indeed other adaptive playout approaches where delays are measured using the receiver system clocks. In reality, the audio clock value returned is not a timestamp as such but rather a reading of the sound card buffer level. This buffer which is completely separate from the anti-jitter receiver buffer is used to hold decoded audio data on the sound card. Essentially the mechanism calculates the change in the amount of data in the sound card buffer between successive incoming packets from the network (i.e. from the remote host) and compares this with the RTP timestamps $(t$ and $t_p)$ of the incoming packets. To smooth out the effect of network jitter, a filter is applied to yield a smoother ratio $e$:

$$e = e + (e_o - e)/16$$ (E.2)

The value 16 was obtained through experimentation and sets the responsiveness of the mechanism. To prevent the mechanism from reacting to sudden and large network disruption such as delay spikes, floor and ceiling values are applied as follows:

$$if((e_o/e) > 1.1$$
$$e_o = e * 1.1$$
$$elseif((e_o/e) < 0.9)$$
$$e_o = e * 0.9$$
$$endif$$

To compensate for skew, the ratio $e$ is applied to a low level rate converter, that matches the receiver audio clock rate to that of the sender. Note that in [42], a principal advantage of the packet scaling technique is that it preserves voice pitch unlike techniques that alter the playout speed.

### E.2.2 Hodson et al. (2001)

In [11], Hodson et al utilise a similar low level mechanism for detecting skew but the tuning details are different. Rather than maintaining a ratio, they timestamp incoming
packets via the sound card buffer level and determine the estimated one-way delay by comparing this to incoming packet RTP timestamps.

\[ m_i = a_i - s_i \]  

(E.3)

In the above \( m_i \) is the delay estimate, \( a_i \) is the arrive time, derived from sound card buffer level, and \( s_i \) is the packet RTP timestamp, relating to when the packet was sent. To smooth out jitter, a linear filter mechanism similar to that introduced in section 3.3 is applied as follows:

\[ \hat{m}_i = \alpha \cdot m_{i-1} + (1 - \alpha) \cdot m_i \]  

(E.4)

Based on experimentation the value of \( \alpha \), which determines the responsiveness of the mechanism, was set to 31/32. A spike detection mechanism similar to that outlined in 3.3 is used to prevent outliers from distorting results. When the value of \( \hat{m}_i \) deviates significantly from its starting value \( m_{\text{active}} \), the compensation mechanism is invoked. The level at which this occurs is set by high and low watermarks.

To compensate for skew, samples are added or deleted from the receiver buffer. The insertion/deletion points are chosen using an approach based on the temporal redundancy within audio. The value of \( m_{\text{active}} \) is updated and the process is repeated.

The detection mechanism in both approaches is somewhat similar in that they use low-level system calls to monitor sound card buffer levels, though the approach to spike detection and the tuning parameters differ. Both approaches use a filtering mechanism that absorbs network jitter leading to a gradual convergence to a value that represents the relative audio skew. In the complete absence of jitter, a steady-state \( e \) value will emerge from [10] resulting in a fixed rate conversion between receiver and sender whereas for [11], the intervals between adjustments (high/low watermark reached) will be constant. The tuning factors within each approach are designed to protect the mechanism from network jitter. Both approaches will however misinterpret a gradual and sustained change in actual delay as a change in skew. Akester’s approach in [10] will result in a different value for \( e \) which will result in changing receiver playout rates relative to the sender. With [11], the result will be a gradual change in \( \hat{m}_i \) with consequent insertion or deletion of samples. In general both approaches can be seen, not as robust skew estimators but rather as mechanisms for preventing the receiver buffer from either overflowing or emptying.
Appendix F

Linux Divert Sockets versus NIST Net

This appendix compares the divert sockets approach taken to emulation in this thesis which is based on the work of [109] with a more advanced alternative NISTNet.

A number of limitations and issues relating to the divert sockets approach emerged during its implementation. These include:

- Packet reordering: The user process which determines the treatment of diverted packets is a single process and acts on packets one at a time. As such it either drops, delays, corrupts or duplicates packets (to represent link loss, delay, noise and duplicates respectively). It cannot however model reordering of packets which though not required for this thesis, is a real phenomenon that can occur in the Internet, principally due to route changes. The study of Internet packet dynamics by [55] examines this along with other issues.

- Delay Modelling: In order to implement multimedia application delay modelling, it is necessary to have knowledge of the packet generation interval to correctly determine the reinject time. For example, if packets are being generated every 20 msec and the delay being modelled is typically 100 msec, a queue of 4-5 packets on average will build up whilst the current packet is being delayed and no record is kept of their arrival time. Their reinject time needs to be determined as outlined
by equation F.1.

\[ T_{inj}^{P_{kn}} = T_{arrive}^{P_{k1}} + (n - 1) \cdot \delta_{int} + \text{Delay}_{ktn} \]  

(F.1)

In the above \( T_{inj}^{P_{kn}} \) is the inject time for packet \( n \), \( T_{arrive}^{P_{k1}} \) is the arrive time for packet 1, \( \delta_{int} \) is the packet generation interval and \( \text{Delay}_{ktn} \) is the required delay to be imposed for packet \( n \).

- A related issue described in section 6.6.1 that required attention is where frames consist of multiple packets. In such situations, the interval between packets may not be constant if silence suppression is enabled as shorter packets may occasionally be sent (with consequent reduced interval).

On the other hand the LDS approach is simple to implement and is easily programmed to implement the required delay characteristics.

The NIST Net project is a much more advanced emulator developed by the National Institute of Standards and Technology. It enables application developers to model dynamics such as delay, jitter, bandwidth limitations, congestion, loss and duplication. A summary of its capabilities and underlying architecture is provided in [111]. The following describes those features most relevant to this thesis, outlining how they compare with those of the LDS approach.

- Delay Simulation Timer Issues: As outlined in section 6.3, NIST Net utilises the Real Time Clock (RTC) with which to set the interrupt frequency. Most RTCs use a 32,768 Hz crystal in order to minimise power consumption and NIST Net extracts a 8192 Hz frequency from this i.e. 32768/4. This equates to a 122 microsecond tick granularity. Note that the network software interrupt is driven off the software clock (typically the 1.193182 MHz 8254 timer chip) which is used to generate a 0.59659 MHz or 1.67619 microsecond tick.

- Scheduling: When each incoming packet is detected at the network interface, emulator entries are checked for a match (which can be based on IP address, port number, TCP/UDP protocol etc) and the appropriate policy implemented. Regarding delay modelling and thus scheduling of packets, NIST Net uses a much more complex mechanism that the LDS approach. It uses a variant of the Linux timer code running on the RTC rather than the software clock. Essentially five
levels of timers are used, all implemented as circular lists. The lowest level timer has \(2^8\) nodes in the list and its pointer advances by one node every ticksize (122 microseconds) and thus undergoes one revolution every \(2^8\) ticks (256 * 122 = 31.232msec). The next level has \(2^6\) nodes, and its pointer advances by a node every 31.232 msec and thus undergoes a revolution every \(2^6 \times 31.232 = 1.998\) sec. In effect the five levels implement timer events with a granularity of 122 microseconds looking 145 hours into the future! Every time a lower level list makes a revolution, the next batch of timers from the next level up are cascaded down into the lower level and sorted according to the finer level of granularity associated with the lower level. Finally the Linux timer code uses a default LIFO protocol to sort between events scheduled for the same tick. In NIST Net this is changed to FIFO to avoid any unnecessary reordering that might occur. As evident from the above description, this mechanism is much more advanced that the simple \textit{select}() function implemented in the LDS approach and does not require any knowledge of packet generation intervals.

- Delay Models: By default, NIST Net implements simple \textit{heavy-tailed} delay models whereby the user specifies parameters such as mean and standard deviation. It determines delays on a per packet basis by randomly choosing values from previously generated tables. By adjusting the relative size of the mean and standard deviation, the degree to which packet reordering occurs can be controlled. Though simple to implement the random selection of such models does not reflect delay correlation. More recent version of NIST Net enable more sophisticated methods to better model delays. For example, externally generated delays can be plugged into NIST Net based on multifractal techniques. Similar work by [108] in developing mathematical models of Internet traffic was quoted in section 5.3.4. This technique enables actual tracedata to be analysed and used in developing more realistic models.

Generally speaking, NIST Net is a more comprehensive modelling tool that is under continuous development. The LDS approach though adequate for the requirements of this thesis is limited in its functionality.
Appendix G

Soundcard Driver Configuration Issues

This appendix briefly outlines a possible high-level explanation for the mechanism by which additional delays are introduced by the OSS soundcard driver. From Fig. 6.18, the delay between the generation time of packets and their subsequent arrival at the receiver NIC is seen to increase in a sawtooth manner in steps of 2 msec up to 30 msec. This occurs when packet size is set at 30 msec (codec G.711) which equates to 240 bytes.

As outlined in section 6.6.1, the Linux OSS driver operates with fragment sizes that are a power-of-two ($2^n$) bytes and which by default is set to 256 ($n = 8$). If for example, the VoIP application is configured by the user to generate 30 msec G.711 packets, this equates to 240 bytes. When the first 32 msec of data arrives into the sender’s sound card buffer, the application sends a 30 msec packet packet onto the network leaving 2 msec of data in the soundcard buffer. This is insufficient for a packet so the application has to wait until the second fragment arrives 32 msec later. This results in a total of 34 msec of data, 30 msec of which is sent to the network leaving 4 msec of data. As such the interval between packets being sent onto the network is 32 msec despite the fact that packets are only 30 msec in size. This causes a 2 msec step increase in delay. Eventually, after 15 packets, the residual data in the soundcard buffer increases to 30 msec which means that two packets are send onto the network in quick succession.
This has the effect of reducing the delay by 30 msec. The process then repeats itself resulting in the sawtooth pattern evident from Fig. 6.18.
Appendix H

Publications and Patents Arising from this Thesis

H.1 International Refereed Conferences/Journals


H.2 National Refereed Conferences


H.3 Patent