

A Framework for Modeling, Analysis and Optimization of Robust Header Compression

CHO CHIA YUAN

(B.Eng. (First Class Hons), NUS)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2005

ACKNOWLEDGEMENTS

The author would like to thank his supervisors Dr Winston Seah Khoon Guan and Dr Chew Yong Huat for introducing him into the exciting world of research, and especially for investing much of their time in the many discussions and repeated reviews towards the improvement of the work leading to this thesis.

The author has also learnt much on the field of header compression through prior work and discussions with Mr Sukanta Kumar Hazra and Mr Wang Haiguang.

Many others have contributed to making the author's candidature at the Institute for Infocomm Research a satisfying and enlightening experience. The author thanks Professor Tjhung Tjeng Thiang for giving him the chance to help out with the administration of the International Journal on Wireless and Optical Communications (IJWOC), Dr Yeo Boon Sain for offering numerous helpful advices and Dr Kong Peng Yong for his time in discussions.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS		i
SUMMARY		iv
LIST OF FIGURES		v
LIST OF TABLES		vi
LIST OF SYMBOLS		vii
LIST OF ABBREVIATIONS		ix
Chapter 1	Introduction	1
	1.1 Motivation	1
	1.2 Contributions	4
	1.3 Thesis Layout and Notation	6
Chapter 2	Background and Problem Definition	7
	2.1 Overview of Robust Header Compression	7
	2.2 Redundancy in Packet Headers	9
	2.3 Encoding Methods	10
	2.3.1 Delta Encoding	11
	2.3.2 Least Significant Bit Encoding	11
	2.3.3 Intermediate Encoding	13
	2.4 The Ingredients of Robustness	14
	2.5 Problem Definition	16
	2.6 Channel Model	18
Chapter 3	A Framework for Modeling Robust Header Compression	21
	3.1 Overview of Modeling Framework	21
	3.2 The Source Process	22
	3.3 The Channel Processes	24
	3.4 The Compressor Process	26
	3.5 The Decompressor Process	31
	3.6 Performance metrics in New Perspectives	33
	3.6.1 Compression Efficiency	33
	3.6.2 Robustness	35
	3.6.3 Compression Transparency	36
	3.7 The Optimization of a Scheme	37

	3.7.1 The Goal of Optimization	37
	3.7.2 The Optimization Procedure	38
	3.8 Modeling Different Source and Deployment Scenarios	39
Chapter 4	The IPID Source Model	42
	4.1 Structure of Source Model	42
	4.2 Validation of Model	47
	4.3 Constructing a Real-World Source Model	49
	4.3.1 Truncating the Number of States	50
	4.3.2 Two-flow Assumption	51
	4.3.3 Resultant Real-World Source Model	52
Chapter 5	Results and Discussions	60
Chapter 6	Conclusion and Future Work	70
	6.1 Conclusion	70
	6.2 Future Work	71
	REFERENCES	72
APPENDIX A	Derivation of Eq. (4.13)	76
APPENDIX B	Derivation of Eq. (4.15)	78

SUMMARY

The Robust Header Compression (ROHC) is a technique which compresses protocol headers robustly over wireless channels to improve bandwidth efficiency and its specifications are being developed by the Internet Engineering Task Force (IETF). Traditionally, header compression schemes are designed based on qualitative descriptions of source headers. This is inadequate because qualitative descriptions do not precisely describe the effect of different source and deployment scenarios, and it is difficult to perform optimization using this methodology. In addition, due to the use of qualitative descriptions, most studies on header compression performance do not take into account the tradeoff between performance metrics such as robustness and compression efficiency. In this thesis, we present a modeling framework for header compression. For the first time, a source model is developed to study header compression. Modeling the way packets are generated from a source with multiple concurrent flows, the source model captures the real-world behavior of the IP Identification header field. By varying the parameters in the source and channel models of our framework, different source and deployment scenarios can be modeled. We use the framework to define and establish the relationship between performance metrics, offering new perspectives to their current definitions. We then introduce the objective of scheme design and the notion of optimal schemes. Based on this new paradigm, we present a novel way to study the tradeoff dependencies between performance metrics. We demonstrate how a scheme can be designed to optimize tradeoffs based on the desired level of performance.

LIST OF FIGURES

Fig. 1	Pictorial overview of Robust Header Compression system	8
Fig. 2	A Typical TCP/IP Header	10
Fig. 3	Markov model of channel state process	18
Fig. 4	Header compression deployment in a general network scenario	22
Fig. 5	Observed sequences for different flows of observations	23
Fig. 6	Header compression deployment over the last hop	40
Fig. 7	IPID Markov Model for 3 concurrent connections	43
Fig. 8	$P(\Delta = \delta)$ for 4 concurrent flows, generated by FTP file downloads	48
Fig. 9	$P(\Delta = \delta)$ for 2 concurrent flows, generated by HTTP file download ACKs	48
Fig. 10	Illustration of model truncation	50
Fig. 11	Estimates of delta probability ratios obtained from trace	53
Fig. 12	The estimate of the n th order probability distribution γ_n from packet trace	55
Fig. 13	Comparison of IPID delta distribution between model and trace	56
Fig. 14	Comparison of IPID delta cumulative distributions between trace and model	57
Fig. 15	Number of concurrent flows generated	58
Fig. 16	Distribution from 10-flow model compared to a heavy-tailed flow in trace	59
Fig. 17	Encoding performance of ROHC-TCP IPID codebook	63
Fig. 18	CT in wireless Channel B versus context window size	64
Fig. 19	Asymptotic CE of optimized codebooks for direct WLSB encoding	65
Fig. 20	IPID delta cumulative distribution at high orders	66
Fig. 21	Variation of CT_{min} with context window size or robustness, w	67
Fig. 22	CE at various context window size, w and context refresh periods, r	68
Fig. 23	Variation of optimum CE with the desired CT_{min}	69

LIST OF TABLES

Table 1	Variations in Controlled Environments	47
Table 2	Markov Model Parameters for state (f,j)	55
Table 3	Channel Model Parameters	61
Table 4	Current ROHC-TCP Specifications	61

LIST OF SYMBOLS

A	Channel A packet error process
B	Channel B packet error process
b	Bit Parameter of (W)LSB code
BER_g	Bit Error Rate on condition that the channel state is good
BER_b	Bit Error Rate on condition that the channel state is bad
C	Compressor process
\bar{C}_j	Mean compression success probability for the j th code in the codebook
D	Decompressor process
f	A generic flow of packets
f_o	Flow of observation
g	Number of complicated CHANGING fields in the header
h_f	State truncation threshold; number of states in flow f of truncated model
K	Number of codes in the codebook Ψ
m	Length of the field in bits
n_A	Number of packets transmitted by the source S to the compressor C
n_B	Number of packets received by the compressor C from the source S
o_f	Offset parameter of (W)LSB code
$q_{(f,j)}^{(f',j')}$	Source model state transition probability, from state (f,j) to (f',j')
r	Context refresh period
S	Source process
U_j	Probability of using the j th code in the codebook
w	Size of context window; measure of robustness
X	Channel state process
Y	Bit error process
Z	Packet error process

β_j	Position of the first bit of the j th packet in a series of packets
Δ	Source delta process
ε	Error due to truncation in Markov model
γ_n	n th order probability distribution
η	Overhead incurred in ‘discriminator bits’ to signal code used in codebook
λ_j	Length of the j th packet
ζ	Set of (w, r) pairs satisfying the desired compression transparency criterion
Ψ^K	Codebook of $K-1$ (W)LSB codes with 1 fallback uncompressed code

LIST OF ABBREVIATIONS

3GPP	3rd Generation Partnership Project
CE	Compression Efficiency
CRC	Cyclic Redundancy Check
CT	Compression Transparency
FO	First Order (state of compressor)
IETF	Internet Engineering Task Force
IPHC	IP Header Compression
IPID	IP Identification
IR	Initialization & Refresh
LSB	Least Significant Bit encoding
MPLS	Multi-Protocol Layer Switching
MSN	Master Sequence Number
ROHC	Robust Header Compression
RTP	Real Time Protocol
SO	Second Order (state of compressor)
TAROC	TCP-Aware Robust Header Compression
UMTS	Universal Mobile Telephone System
VJHC	Van Jacobson Header Compression
WLSB	Window-based Least Significant Bit encoding

Chapter 1 Introduction

1.1 Motivation

Header compression improves the bandwidth efficiency over bandwidth scarce channels and is especially attractive in the presence of small packet payloads, which is often the case in practice. Interactive real-time applications like IP telephony, multi-player network gaming and online chats all generate disproportionately small payloads in comparison to headers. In addition, non real-time applications like web browsing predominantly carry payloads no more than a few hundred bytes.

The adoption of early header compression schemes over wireless links failed because early schemes like Van Jacobson Header Compression (VJHC) [1] were designed to operate over reliable wired links. Each loss of a compressed packet caused the compressor-decompressor context synchronization to be lost, generating a series of packets discards due to corrupted packets from decompression failures. The error condition persisted till packet retransmission initiated by higher layers (e.g. TCP) restored context synchronization. Over wireless links where high error rates and long round trip times are common, this caused header compression performance to deteriorate unacceptably. To deal with this, a number of schemes like IP Header Compression (IPHC) [10] and TCP-Aware Robust Header Compression (TAROC) [9] were proposed to offer robustness against packet loss in wireless channels. The ROHC is currently the state-of-the-art header compression technique. A robust and extensible scheme, the ROHC is being developed by the IETF [2], and is an integral part of the 3rd Generation Partnership Project-Universal Mobile Telephone System (3GPP-UMTS) specification [3].

The deployment scenarios for header compression have increased over the years. Early header compression schemes like VJHC were first used over wired serial IP lines [1]. Current efforts mainly focus on developing header compression over ‘last hop’ wireless links and cellular links like UMTS [2]. Some of the most recent proposals explore header compression over multiple hops in a mobile ad hoc network [6], and even for high-speed backbone networks [14].

With the expected deployment of ROHC in increasingly diverse types of networks, the evaluation of Robust Header Compression performance in different scenarios becomes crucial. A number of tools and studies related to header compression performance can be found in the literature. The effect of ROHC on the subjective and objective quality of video was evaluated in [12] from a test-bed. Other studies evaluate header compression performance by simulation. Specialized ROHC simulators like the Acticom ROHC Performance Evaluation Tool [8], the Effnet HC-Sim [7], and the ROHC simulator-visualizer [13] have been developed for this purpose, though they are not readily available in public domain. Most studies in literature focus on various issues in header compression. An early study investigated the effect of inter-leaving at the packet source on RTP header compression [11]. A proposal on header compression over Multi-Protocol Layer Switching (MPLS) in high-speed networks investigated the tradeoff between compression gains and implementation cost [18]. The cost and performance due to the context establishment has been studied using an analytical model in [15] and the handover aspect was analyzed in [16]. The notion of adaptive header compression was introduced in [17], where it was suggested that scheme parameters like the context window size and packet refresh rate be made adaptive to link conditions and packet sizes. However, the issue of *how* these parameters can be made adaptive was not addressed in the same thesis.

While progress in several key aspects has been made in the above studies, we note that the above studies on header compression performance typically assume some particular network deployment scenario, i.e. over ‘last-hop’ wireless links. Moreover, we find that with the exception of few studies [7], [11], the operating environment influencing the *content* and *sequence* of headers arriving at the compressor has not adequately addressed. The common setup used involves two nodes – the compressor and decompressor, separated by a wireless channel (simulated or real) in between. Indeed, this is a setup used in ROHC interoperability tests [8], [12]. In most studies, the performance is evaluated by generating packets at the compressor (sometimes with real application payloads) for performing header compression. We note that the header contents generated in experimental conditions may be different from those in real operating environments. Because most studies do not ensure their headers are generated based on real-world behavior, they inadvertently assume idealized operating environments (e.g. handling non-concurrent flows) at the source. Moreover, the effect of packet loss between the source and compressor has not been studied in any existing work. Due to these shortcomings, packet headers produced under experimental conditions may become easily compressed at high efficiencies. Because this seems easily achieved, the interaction and tradeoffs between ROHC performance metrics like robustness and compression efficiency are often not examined in existing work.

The second issue deals with the design methodology of header compression schemes. Since the proposal of the first TCP/IP header compression scheme, VJHC [1] in 1990, it has been more or less a tradition for scheme design to be based on rules-of-thumb and qualitative descriptions of source headers [2], [4]. Without a formal approach, the effects of different source and deployment scenarios cannot be precisely described, and optimization is difficult. As such, the notion of optimized schemes does not exist.

1.2 Contributions

To deal with the issues highlighted in the previous section, our prior work started with the quantification and analysis of TCP/IP inter-flow field behavior based on a database of 2 million packet headers captured from *real traffic*. The details on the behaviour of all TCP/IP header fields can be found in [22]. Based on this, we have developed an approach to optimize inter-flow header compression (termed “context replication” in ROHC terminology). In the same paper, we have shown that inter-flow header compression gains can be improved by using a design methodology based on the quantitative description of real-world field behaviour [21].

Our first contribution in this thesis is to propose a framework for modeling Robust Header Compression in general. The framework has five stochastic processes as its main components: the source, the source-compressor channel, the compressor, the compressor-decompressor channel, and finally the decompressor. By including the source process and source-compressor channel in the framework, a more complete picture of the main components affecting the performance is obtained. The framework is designed to be flexible enough to allow different scenarios to be modeled. For example, different deployment scenarios can be modeled by tuning the parameters of the channel models.

The ROHC has qualitatively defined three metrics for ascertaining the performance of an ROHC scheme: compression efficiency, robustness and compression transparency. We show that our modeling framework offers new perspectives to the definition and understanding of header compression performance metrics, using which we present a novel way to study the tradeoff dependencies between performance metrics.

Moving on from qualitative descriptions of header behavior to mathematical models, we present a real-world source model for studying header compression. This is the first time a source model is used for studying header compression. Built on a Markov model of the packet source, our source model captures the *real-world* behavior of the IP Identification header field in TCP flows. The effect of multiple *concurrent flows* on field behavior is modeled using a chain of Markov states for each packet flow. Using real traffic, we have built a real-world IPID source model for the average source. Interestingly, the source model may have wider applications because it also models the way packets are generated from a source with multiple concurrent flows. We also obtain the models for a busy source and the non-concurrent source in idealized operating environment. By plugging the desired source model into our modeling framework, the effect of different source scenarios on the performance outcome is investigated. Our results in Chapter 5 verify our intuition that the idealized operating environment of non-concurrency coupled with a perfect source-compressor channel leads to unrealistically high compression efficiencies almost independent of the robustness configuration.

Using our framework, we formally introduce the notion of optimized schemes. Presenting a tradeoff optimization procedure, we show, for the first time, that the parameters of a ROHC scheme can be tradeoff optimized based on the desired level of performance. This opens up the possibility of *adaptively optimizing* the entire set of parameters in a ROHC scheme, instead of adapting two parameters as suggested in [17] without optimization.

A short description of the work done based on the above key ideas can be found in [23]. Important expansions and elaborations on the key ideas as well as new results are

found in an extended version [24] as well as in the remaining of this thesis.

1.3 Thesis Layout and Notation

This thesis is organized in the following structure. In the next chapter, we present the background and problem definition. Our framework for modeling header compression will be developed in Chapter 3. In Chapter 4, we present the source model for studying header compression. This is followed by our results and discussion from the performance and tradeoff study in Chapter 5. We end this thesis with the significance of our contributions in conclusion and discussion of future work.

The notation adopted in this thesis is as follows: random variables are in upper case whilst values are in lower case. Vectors are assumed to be row vectors, and both vectors and matrices are denoted in bold, while the former is in lower case and the latter is in upper case. $(\cdot)^T$ is used to denote the transpose of a matrix or vector.

Chapter 2 Background and Problem Definition

2.1 Overview of Robust Header Compression

Fig. 1 gives a pictorial overview of the ROHC system over a wireless channel. In general, a number of packet flows pass through the system simultaneously. The compressor compresses each packet by referring to previous headers of the same flow. This is done by maintaining a window of w contexts per flow, where each n th context stores the n th previous header. As will be elaborated upon, the window of w contexts are required for robustness. The decompressor is only required to maintain a single context per flow. This context stores the latest header which has been verified to be successfully decompressed through passing the Cyclic Redundancy Check (CRC). The decompressor may feedback the compressor upon verification success or failure. To facilitate feedbacks, each packet is uniquely identified with a Sequence Number. In ROHC-TCP, this is called the Master Sequence Number (MSN), which is maintained as a flow-specific counter [5]. The MSN is part of the ROHC header in compressed packets and is added by the Compressor.

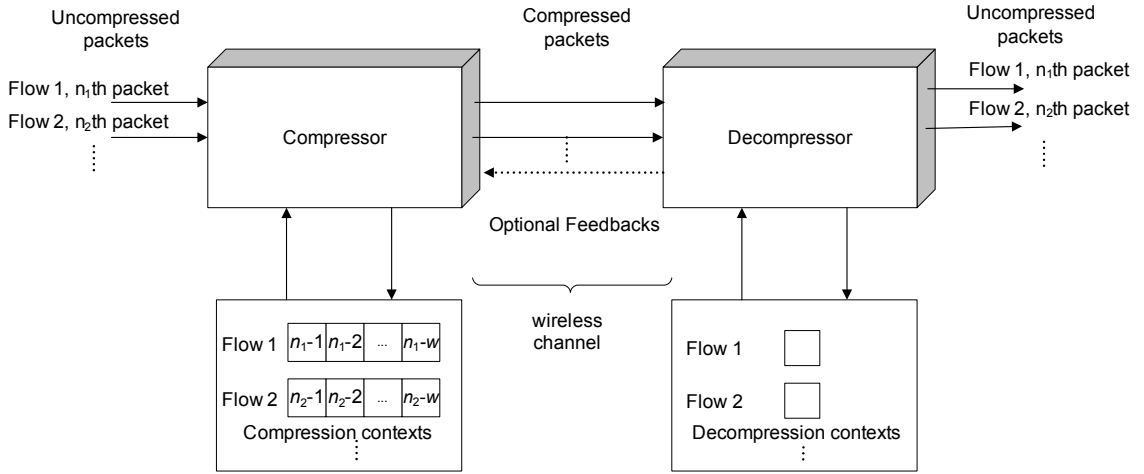


Fig. 1: Pictorial overview of Robust Header Compression system

The actions performed by the compressor and decompressor are state-dependent, controlled by the compressor and decompressor state-machines respectively. Three compressor states are defined in the ROHC framework: Initialization and Refresh (IR) state, First Order (FO) state, and Second Order (SO) state [2]; the three states are reduced to two in ROHC-TCP: IR state and Compressed (CO) state, for which the latter state is synonymous to the FO state [5]. The name of the state is indicative of the operation in that state: In IR state, the full header is sent uncompressed; In FO (SO) state, the first (second) order differences between packets are used to perform compression. Naturally, header compression is the most efficient in the SO state.

For the purpose of clarity, we will implicitly adopt the two-state compressor state machine used in ROHC-TCP for our problem definition and analysis. However, it is not too difficult to extend our analysis using the same approach for the three-state case.

2.2 Redundancy in Packet Headers

Most header fields either do not change throughout a flow, or typically increase with small deltas between consecutive packets of a flow. Header compression capitalizes on the behavioral patterns of header fields and exploits the redundancy between header fields of different packets belonging to the *same packet flow*. For ease of reference, the header fields found in a typical TCP/IP header is shown in Fig. 2.

All header fields can fit into either one of the following general categories: INFERRED, STATIC, STATIC-KNOWN and CHANGING [9]. These category names indicate the behavioral pattern of that particular type of fields. Correspondingly, fields in each category are encoded in a way unique to that category. INFERRED fields can be inferred without requiring the sending of that field. An example is the IP Packet Length field. STATIC fields like the IP Source and Destination Addresses do not change throughout the entire packet flow. These fields need to be communicated only at the beginning of each flow. STATIC-KNOWN fields are well-known values which do not change throughout the entire connection, and thus need not be sent at all. Last of all, CHANGING fields vary dynamically throughout a flow. Most CHANGING fields share the common characteristic of small delta increases between packet headers. Examples of CHANGING fields include the IP Identification (IPID), TCP Sequence Number and TCP Acknowledgement Number.

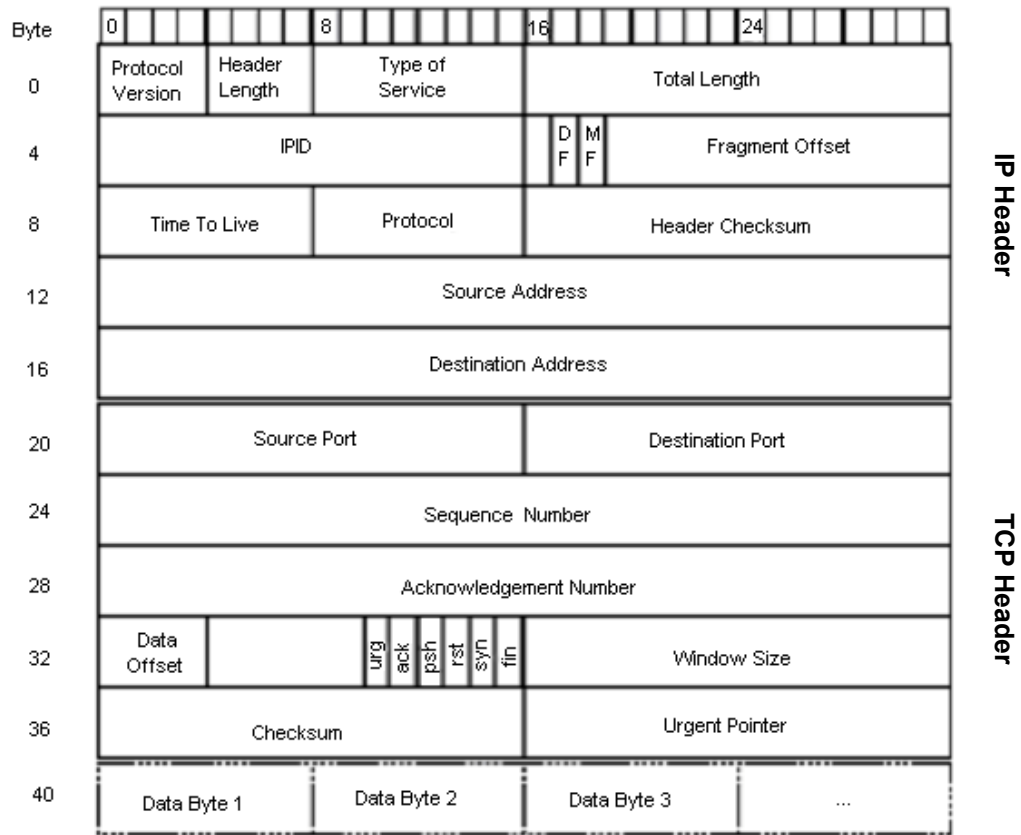


Fig. 2: A Typical TCP/IP Header

2.3 Encoding Methods

Most of the complexities required in header compression schemes are attributed to a relatively small number of CHANGING fields. The type of encoding used for these deltas makes the difference between a good and poor scheme. In this section, we will introduce the two main ways of encoding CHANGING fields – delta encoding and Least Significant Bit (LSB) encoding. We also briefly discuss the use of intermediate encoding to further improve header compression gains.

2.3.1 Delta Encoding

Delta encoding is a straightforward approach to reduce the redundancy between headers. Due to the fact that many CHANGING header fields increase with small deltas between consecutive headers, delta encoding simply encodes a field as the *difference* in its value between two consecutive headers. For example, if the TCP Sequence Numbers in two consecutive headers are 2900000 and 2900360, then the field in the second header can be encoded into its delta, 360 instead. To facilitate encoding (decoding), the previous packet header is stored in the context at the compressor (decompressor).

Though this approach is simple, the decompression of each header requires the previous header to be received correctly. A single packet loss induces a series of further packet discards due to decompression errors as the compressor-decompressor context synchronization is lost. This phenomenon is known as damage propagation. The avalanche of packet discards continues till higher layer (e.g. TCP) retransmission mechanisms are activated. This approach is acceptable over wired channels due to low residual error rates and short round-trip delay. Over error-prone wireless channels, this solution is unsatisfactory because the higher layer recovery is achieved only after long delay and high packet loss ratio. Thus over wireless channels delta encoding results in extremely poor performance and is unsuitable.

2.3.2 Least Significant Bit Encoding

The Least Significant Bit encoding (LSB) is proposed in ROHC as an alternative to delta encoding. A LSB code is defined by two parameters, (b, o) . Instead of compressing fields into deltas, it requires the b least significant bits of the field to be

sent over the channel. A LSB encoded field can be decoded unambiguously if the *difference* of the original value with respect to the reference value is *within* the *interpretation interval* $[-o_f, 2^b - 1 - o_f]$.

Using the previous example where we encode the value 2900360 using 2900000, suppose we first define a LSB code (10, 0) known to both the compressor and decompressor. With knowledge of the previous value, 2900000, and receiving only the 10 least significant bits, i.e. 0110001000 in binary, the decompressor simply locates the binary sequence in the range $[2900000, 2900000 + 2^{10} - 1]$ and thus is able to uniquely identify the next value as 2900360.

Note that the field can be encoded *only if* the delta is within the interval. Using the same example, if the LSB code (4, 0) is used instead, then only values in between 2900000 to 2900015 inclusive can be encoded without decoding ambiguity. In this case, the LSB code defined by (4, 0) has failed to encode the field and the compressor has to decide on other alternatives. Note that since the size of the interpretation interval is 2^b , only b bits are required to identify the position within the interval, and thus only b bits are communicated in encoded form. The position of the interpretation interval (with respect to the reference field) can be shifted through the pre-defined offset o_f . The ROHC recommends defining o_f based on field behavior [2], i.e. if the field value only increases, then o_f should be -1. If the field value is non-decreasing, then o_f should be 0. If it is strictly decreasing, then it should be 2^b .

Note that LSB encoding by itself is not superior to delta encoding in the sense that it just as vulnerable to damage propagation. However, the concept of LSB encoding enables its enhanced form, Window-based LSB (WLSB) to be used. This is the key robustness ingredient in ROHC, as will be elaborated in Section 2.4.

2.3.3 Intermediate Encoding

The purpose of intermediate encoding is to improve header compression gains by leveraging on the redundancy between header fields *within the same header*. In most cases, such redundancy is limited between header fields. However, a degree of inference is possible when two header fields are sufficiently similar. In fact, if a header field can be *completely described* by another field within the same header, then it is categorized as INFERRED and it need not even be sent at all (see Section 2.2). Otherwise, if there is still *significant redundancy* with another field, then a form of intermediate encoding can be performed before using LSB encoding.

The most common form of intermediate encoding comes from the ‘INFERRED-OFFSET’ encoding method defined by ROHC. Given that there are two header fields within the same header which are sufficiently similar, this encoding method simply replaces one of the field by subtracting one field from the other to form a new field. The new field then becomes the input to LSB encoding. Intuitively, intermediate encoding causes the delta differences between consecutive headers to be reduced, thus allowing higher gains.

In ROHC-TCP, the IPID field shares a similar characteristic to the Master Sequence Number (MSN) field. The MSN is a ROHC header field introduced in Section 2.1. In the ROHC-TCP specification, the IPID is specified to be encoded with respect to the MSN via the ‘INFERRED-OFFSET’ encoding method before using LSB on the resultant field [5]. Results on the improvement due to this intermediate encoding are presented in this thesis.

2.4 The Ingredients of Robustness

The ROHC is designed to operate over wireless error-prone channels because it has mechanisms to *prevent* damage propagation and *quickly recover* from damage propagation. Damage prevention is achieved by Window-based LSB encoding (WLSB); fast recovery is achieved by either periodic context refreshes or feedback-initiated context refreshes.

Unlike delta and LSB encoding, WLSB encoding does not require exact context synchronization between the compressor and decompressor. This means that the decompressor need not refer to the same context used by the compressor when decompressing a packet.

In WLSB, the compressor keeps a *sliding window* of the *last w contexts*, but the decompressor maintains only the *last successfully decompressed context* (see Fig. 1). Thus, the LSB is in fact a specific case of WLSB with $w = 1$. For each packet, *the compressor ensures that the compressed packet can be decompressed using any context within its sliding window*. Thus, the decompressor's context is valid as long as it is identical to *any one context* inside the sliding window used at the compressor. We can see that robustness is achieved: only one out of w contexts at the compressor need to be synchronized with that at the decompressor and in the worst case, the scheme can tolerate up to $(w - 1)$ consecutive packet drops without damage propagation.

We now explain *how* “the compressor ensures the compressed packet can be decompressed using any context within its sliding window”. Recall from our explanation on the LSB code (b, o_f) that an encoded field can be decoded

unambiguously if the *difference* of the original value with respect to the reference value is *within* the *interpretation interval* $[-o_f, 2^b - 1 - o_f]$. We now extend this reasoning to the WLSB code (b, o_f) where there is a window of w contexts (and thus a window of w reference values). If the compressor wants to ensure that the encoded field can be decoded using any context within its sliding window, then it encodes the field *only if* this condition is satisfied: The *difference* of the original value with respect to *each* reference value in the window of contexts is *within* the *interpretation interval* $[-o_f, 2^b - 1 - o_f]$.

We illustrate the concept of robustness using the sequence of three values: 2899700, 2900000, and 2900360. Suppose the WLSB code $(10, 0)$ is used and a sliding window of size $w = 2$ is maintained at the compressor. We focus on the WLSB encoding of the third value. In the same way as that in LSB encoding, the compressor transmits only the 10 least significant bits of 2900360, which is 0110001000 in binary. The decompressor is able to locate this binary sequence uniquely within the range of both intervals $[2899700, 2899700 + 2^{10} - 1]$ and $[2900000, 2900000 + 2^{10} - 1]$ (note that both intervals have a size of 2^{10}). Therefore, the decompressor requires the apriori error-free reception of only *either* 2899700 *or* 2900000 to identify the third value as 2900360. This means that with $w = 2$, a single packet loss is tolerated without causing damage propagation. The penalty to pay for robustness is the stronger condition for encoding success: the value to be encoded *must* be within the interpretation interval of *all* the previous w values. It is easy to see that this condition is satisfied in the above example. Changing the WLSB code to $(9, 0)$ in the above example, the encoding attempt now fails because $2900360 \in [2900000, 2900000 + 2^9 - 1]$ but $2900360 \notin [2899700, 2899700 + 2^9 - 1]$.

WLSB achieves robustness by *preventing* damage propagation. The ROHC uses 2 other mechanisms for *fast recovery* from damage propagation: periodic context refreshes and optional feedbacks from the decompressor to compressor. Periodic context refreshes simply means sending uncompressed packets at periodic intervals of r packets. With this mechanism in place, damage propagation can no longer extend beyond r packets easily. Feedbacks from the decompressor further accelerate recovery through explicit retransmission requests at the header compression layer. However, this is an optional feature and we assume the absence of feedbacks in this thesis.

2.5 Problem Definition

As mentioned in the previous sections on LSB and WLSB encoding, if the delta lies outside the interpretation interval of a particular (W)LSB code (b, o_f) , then this code has failed to encode the field and the compressor has to decide on other alternatives. In general, the alternatives involve either using another (W)LSB code, or sending the field uncompressed.

When a WLSB code fails to encode a field, a series of WLSB codes are often used as further attempts to WLSB encode the field. In ROHC specifications, the offset parameters for a series of WLSB codes are held constant or relatively constant for a given field. When all specified WLSB codes fail, the compressor falls back to sending the field uncompressed. The latter can be seen as the final ‘code’.

The assumptions made leading to our problem definition are as follows:

- The compressor uses the two-state compressor state machine in ROHC-TCP.

- The set of specified codes (codebook) is shared apriori between the compressor and decompressor.
- The offset parameter o_f is held constant for a given field in all WLSB codes.

A set of $K-1$ WLSB codes and one uncompressed ‘code’ constitutes the codebook for a single field, and can be defined as $\Psi^K = \left\{ \left\{ (b_i, o_f) \right\}_{i=1}^{K-1}, m \right\}$, where $b_j > b_i$ if $j > i$, and m is the length of the field in bits. The compressor shares the same codebook with the decompressor, and the compressor signals the code used with up to $\lceil \log_2 K \rceil$ ‘discriminator bits’, which are incurred as overhead. In ROHC-TCP, the overhead may be less than $\lceil \log_2 K \rceil$ bits due to the use of Huffman coding, and this is usually necessary when K is large.

Finally, note that for a single field, we define the set of parameters denoting a Robust Header Compression scheme as $\{\Psi^K, w, r\}$, where Ψ^K is the codebook such that $\Psi^K = \left\{ \left\{ (b_i, o_f) \right\}_{i=1}^{K-1}, m \right\}$. Extending this to include all CHANGING fields in the header, the entire set of parameters defining a Robust Header Compression scheme becomes $\{\Psi_1^{K_1}, \Psi_2^{K_2}, \dots, \Psi_g^{K_g}, w, r\}$ where g is the number of complicated CHANGING fields.

At this stage, the objective of scheme design is to determine *suitable values* for these parameters given *external conditions* over which the designer has no control of. We will see that our modeling framework enables a more formal and meaningful definition of the objective of scheme design.

2.6 Channel Model

We are interested in a model for the packet loss/survival process over a channel. A variety of models attempting to model the packet loss/survival process exists in the literature. We select a particular model which has been found to be relatively accurate [19], and which can be easily adjusted to model wireless channels in high and low speed mobility [20]. We outline an adapted version of this model and the interested reader is referred to [19] for details.

Let $\{Z(j): j = 1, 2, \dots\}$ be the packet loss/survival stochastic process over the channel, with event space $\{1,0\}$ denoting the events of packet survival and packet corruption respectively. $\{Z(j)\}$ can be defined from the lower-level bit error/error-free process.

The bit error/error-free process is modeled by the well-known Gilbert-Elliot model. Let $X(i) \in \{\text{good}, \text{bad}\}$ denote the channel state during the transmission of bit i . The channel state process $\{X(i): i = 1, 2, \dots\}$ is modeled by a two-state Markov chain as shown in Fig. 3. In each state, a state-dependent bit error rate exists in the duration of that bit transmission, denoted by BER_g and BER_b , where $BER_g < BER_b$. The effects of Forward Error Correction (FEC) can be taken into consideration when defining the bit error rates. Practical values of state transition probabilities and state-dependent bit error rates due to different mobility speeds have been found in [20] and will be used in this thesis.

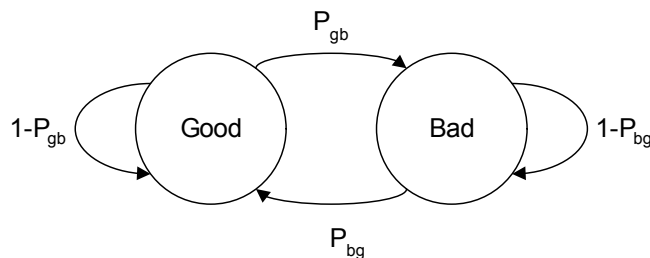


Fig. 3: Markov model of channel state process

Let $\{Y(i): i = 1, 2, \dots\}$ be the bit error/error-free process with event space $\{1,0\}$ denoting the events of bit error-free and bit error respectively. Then the outcome of bit i conditioned on the channel state is given by

$$\begin{aligned} P(Y(i) = 0 | X(i) = good) &= BER_g, \\ P(Y(i) = 0 | X(i) = bad) &= BER_b, \\ P(Y(i) = 1 | X(i) = good) &= 1 - BER_g, \\ P(Y(i) = 1 | X(i) = bad) &= 1 - BER_b. \end{aligned} \quad (2.1)$$

Let the j th packet start at bit β_j and end at bit $\beta_j + \lambda_j - 1$. The packet remains uncorrupted when all bits in the range $[\beta_j, \beta_j+1, \dots, \beta_j+\lambda_j-1]$ are error-free. Therefore, at each j , the packet loss probability of $Z(j)$ can be defined as

$$P(Z(j) = 1) = P\left(\bigcap_{i=\beta_j}^{\beta_j+\lambda_j-1} Y(i) = 1\right). \quad (2.2)$$

As a result of the above expression, the *error-free* probability of the j th packet given its initial channel state can be evaluated as:

$$\begin{aligned} P(Z(j) = 1 | X(\beta_j) = X_{\beta_j}) &\triangleq P_{Z_j | X_{\beta_j}}^{1|x_{\beta_j}} = \mathbf{x}_j \mathbf{M}^{\lambda_j-1} \mathbf{e}^T, \\ \text{where} \\ \mathbf{x}_j &= \begin{cases} [1 \ 0], x_{\beta_j} = good \\ [0 \ 1], x_{\beta_j} = bad \end{cases} \\ \mathbf{M} &= \begin{bmatrix} p_{gg}(1 - BER_g) & p_{gb}(1 - BER_g) \\ p_{bg}(1 - BER_b) & p_{bb}(1 - BER_b) \end{bmatrix} \\ \mathbf{e} &= [(1 - BER_g) \ (1 - BER_b)]. \end{aligned} \quad (2.3)$$

and $P(Z(j) = 1 | X(\beta_j) = x_{\beta_j}) \triangleq P_{Z_j | X_{\beta_j}}^{1|x_{\beta_j}}$ is defined for compactness. For analyzing a group of consecutive packets, a useful auxiliary result is the probability that the j th packet is error-free *and* the $(j+1)$ th packet starts at a particular channel state, *given* the initial channel state at the start of the j th packet:

$$P\left(Z(j)=1, X(\beta_{j+1})=x_{\beta_{j+1}} \mid X(\beta_j)=x_{\beta_j}\right) \triangleq P_{Z_j, X_{\beta_{j+1}} \mid X_{\beta_j}}^{1, x_{\beta_{j+1}} \mid x_{\beta_j}} = \mathbf{x}_j \mathbf{M}^{\lambda_j - 1} \begin{pmatrix} \tau \\ \tau \end{pmatrix},$$

where

$$\mathbf{e}' = \begin{cases} \begin{bmatrix} p_{gg}(1 - BER_g) & p_{bg}(1 - BER_b) \end{bmatrix}, x_{\beta_{j+1}} = \text{good} \\ \begin{bmatrix} p_{gb}(1 - BER_g) & p_{bb}(1 - BER_b) \end{bmatrix}, x_{\beta_{j+1}} = \text{bad}. \end{cases}$$

Finally, it is not too difficult to derive similar expressions for variations of other packet *error* probabilities given the initial channel state:

$$\begin{aligned} P_{Z_j \mid X_{\beta_j}}^{0 \mid x_{\beta_j}} &= 1 - P_{Z_j \mid X_{\beta_j}}^{1 \mid x_{\beta_j}}, \\ P_{Z_j, X_{\beta_{j+1}} \mid X_{\beta_j}}^{0, x_{\beta_{j+1}} \mid x_{\beta_j}} &= P_{X_{\beta_{j+1}} \mid X_{\beta_j}}^{x_{\beta_{j+1}} \mid x_{\beta_j}} - P_{Z_j, X_{\beta_{j+1}} \mid X_{\beta_j}}^{1, x_{\beta_{j+1}} \mid x_{\beta_j}}. \end{aligned}$$

Chapter 3 A Framework for Modeling Robust Header Compression

We begin this chapter with a high level presentation of our modeling framework in terms of five stochastic processes - a source process, two channel processes, a compression process and a decompression process. We give the initial broad definition for the source process, and describe the channel processes based on existing channel models. We then show how the compression and decompression processes can be defined based on the source process and channel processes. Extending current definitions, we offer new perspectives to the three performance metrics. We end this chapter by showing how the framework can be used to model different source and deployment scenarios.

3.1 Overview of Modeling Framework

Fig. 4 shows header compression deployment in a general scenario. The source node is the generator of packet headers, transmitting packets to the remote compressor through Channel A. The compressor compresses the packet headers and transmits compressed packets through Channel B to the decompressor. The passage beyond the decompressor has no effect on the header compression system. As illustrated in Fig. 4, this can be modeled with five stochastic processes. Trivial and starkly simple, we will show that this representation allows different source and deployment scenarios to be modeled.

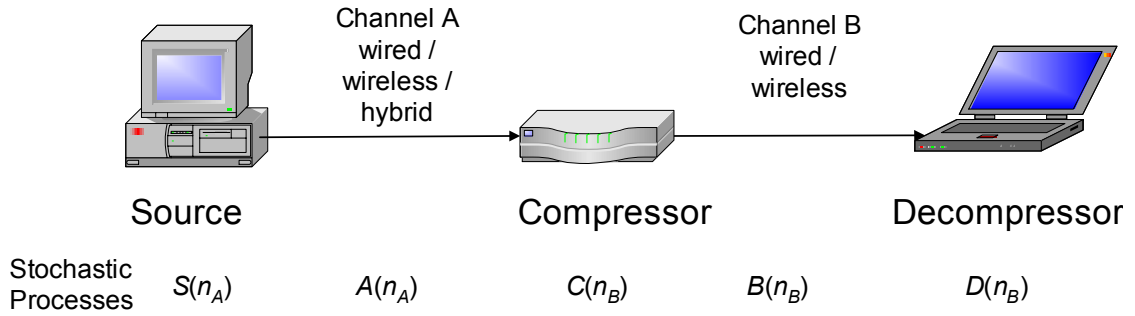


Fig. 4: Header compression deployment in a general network scenario

3.2 The Source Process

Different protocol fields possess different and often independent behavior, making it difficult to have a single source model describing all the fields in the entire header. As a start, with the assumption of independence, the approach is to develop source models for each field individually. To analyze different fields, we simply use different source models.

For each field, the source of the flow under observation, f_o , is a discrete-time stochastic process, $\{S_{f_o}(n_A): n_A = 0, 1, \dots\}$, which takes values in the discrete space $[0, 2^m - 1]$ for a field of m bits long. A convenient way of simplifying the process $\{S_{f_o}(n_A)\}$ is to view it as having an initial value $s_{f_o}(0)$ and being generated from a ‘delta process’, $\{\Delta_{f_o}(n_A): n_A = 0, 1, \dots\}$, such that

$$\begin{aligned}
 \Delta_{f_o}(n_A) &= S_{f_o}(n_A) - S_{f_o}(n_A - 1), \\
 \delta_{f_o}(n_A) &= s_{f_o}(n_A) - s_{f_o}(n_A - 1).
 \end{aligned}
 \tag{3.1}$$

Thus, in general, the source of a flow of observation can be modeled by its delta process $\{\Delta_{f_o}(n_A)\}$ and an initial value $s_{f_o}(0)$. Conversely, we can express the difference between $s_{f_o}(n_A)$ and $s_{f_o}(n_A - h)$ in terms of deltas:

$$\begin{aligned}
s_{f_o}(n_A) - s_{f_o}(n_A - h) &= \sum_{i=n_A-h+1}^{n_A} s_{f_o}(i) - s_{f_o}(i-1) \\
&= \sum_{i=n_A-h+1}^{n_A} \delta_{f_o}(i).
\end{aligned} \tag{3.2}$$

Note that during header compression operation, the compressor transmits a number of least significant bits defined by b_i due to its choice of code (b_i, o_f) in its codebook Ψ^k . The above delta process is a means to model the source process, and so should not be confused as the output of the compressor.

Let us apply the concept of source and delta processes to the IPID field. The IPID is a unique identifier for packets from a source. A predominant implementation is to use a common counter shared by all flows, incremented for every packet sent from the source. For a single flow, consecutive packets carry values with small sequential offsets. This is known to account for the small IPID deltas characterization in literature [4]. As an example, consider a source transmitting 2 flows (flows 1 and 2) concurrently. Fig. 5 illustrates the sequence of values observed when either flow is chosen to be the flow of observation.

IPID value:	1	2	3	4	5	6	7	8	9	10	
Flow:	1	2	2	1	1	1	2	1	2	1	
$f_o=1$	$s_1(n_A)$:	1	-	-	4	5	6	-	8	-	10
	$\delta_1(n_A)$:	-	-	-	+3	+1	+1	-	+2	-	+2
$f_o=2$	$s_2(n_A)$:	-	2	3	-	-	-	7	-	9	-
	$\delta_2(n_A)$:	-	-	+1	-	-	-	+4	-	+2	-

Fig. 5: Observed sequences for different flows of observations

For the rest of this thesis, the notion of the chosen flow of observation is assumed to apply for the source process, and the flow subscript f_o will be dropped from $S_{f_o}(n_A)$ and $\Delta_{f_o}(n_A)$.

3.3 The Channel Processes

Let Channel A be the channel between the source and compressor. To model the packet loss in the passage through Channel A, we define a packet survival/loss discrete-time stochastic process $\{A(n_A): n_A = 1, 2, \dots\}$, such that

$$A(n_A) = \begin{cases} 1, & n_A \text{th packet is not corrupted in Channel A} \\ 0, & n_A \text{th packet is corrupted in Channel A.} \end{cases} \quad (3.3)$$

Existing channel models like that presented in Section 2.6 can be used to model this process. We also derive another stochastic process from $\{A(n_A)\}$, called the Channel A *loss run process*, $\{L_A(j_A): j_A = 1, 2, \dots\}$, where for a particular index j_A , the number of packets lost between two nearest loss-free packets is a random variable $L_A(j_A)$. For example, in the following sample sequence of $\{A(n_A)\}$: 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, ..., the sequence of loss run lengths would be: 0, 1, 0, 0, 0, 2, 3, The probability of a loss run, $P(L_A(j_A) = l_A)$, can be expressed as:

$$P(L_A(j_A) = l_A) = P(A(n_A - 1) = 0, A(n_A - 2) = 0, \dots, A(n_A - l_A) = 0, A(n_A - l_A - 1) = 1 | A(n_A) = 1), \quad (3.4)$$

where the j_A th run is defined in terms of the Channel A process as $j_A = \left(\sum_{i=1}^{n_A} a(i) \right) - 1$.

Let Channel B be the channel between the compressor and decompressor. In the same way,

$$B(n_B) = \begin{cases} 1, & n_B \text{th packet is not corrupted in Channel B} \\ 0, & n_B \text{th packet is corrupted in Channel B,} \end{cases} \quad (3.5)$$

where $n_B = \left(\sum_{i=1}^{n_A} a(i) \right)$ is the number of packets from the source successfully received

by the compressor and

$$P(L_B(j_B) = l_B) = P(B(n_B - 1) = 0, B(n_B - 2) = 0, \dots, B(n_B - l_B) = 0, B(n_B - l_B - 1) = 1 | B(n_B) = 1), \quad (3.6)$$

where the j_B th run in Channel B is defined by $j_B = \left(\sum_{i=1}^{n_B} b(i) \right) - 1$.

Note that the stochastic process $\{B(n_B): n_B = 1, 2, \dots\}$ maintains a separate counter n_B due to the fact that $B(n_B)$ is defined only in the event $A(n_A) = 1$. We make the assumption that $\{A(n_A)\}$ and $\{B(n_B)\}$ are independent, which is reasonable in practice. Then $\{B(n_B)\}$ is another channel process independently defined from the packet loss/survival model in Section 2.6.

As $n_A \rightarrow \infty$, it is well-known that the Gilbert-Elliot channel state model converges to a steady-state distribution:

$$\begin{aligned} P_X^g &\triangleq P(X(n_A) = g) \Big|_{n_A \rightarrow \infty} = \frac{P_{bg}}{P_{gb} + P_{bg}}, \\ P_X^b &\triangleq P(X(n_A) = b) \Big|_{n_A \rightarrow \infty} = 1 - P_X^g. \end{aligned} \quad (3.7)$$

This allows the steady-state probability of the packet loss process and loss run process to be formulated. Focusing on Channel A with results equally applicable to Channel B, we know from Eq. (3.4) as $n_A \rightarrow \infty, j_A \rightarrow \infty$ that

$$P(L_A = l_A) = \frac{P(A_0 = 1, A_{-1} = 0, \dots, A_{-l_A} = 0, A_{-l_A-1} = 1)}{P(A_0 = 1)}, \quad (3.8)$$

where $A_{-l} \triangleq A(n_A - l) \Big|_{n_A \rightarrow \infty}$. We assume that all packets passing through Channel A are of the same length, so that we need not be concerned with differences in packet lengths at steady state. The denominator in Eq. (3.8) is the probability of packet survival in Channel A. Using results in Section 2.3, the steady-state probability of the packet loss

process can be evaluated as

$$P(A_0 = 1) = P_{A|X}^{1|g} P_X^g + P_{A|X}^{1|b} P_X^b. \quad (3.9)$$

The numerator in Eq. (3.8) can be derived by considering, for each packet, the probability for all combinations between (i) the initial channel state of the current packet, (ii) the current packet loss/survival, and (iii) the channel state after the current packet. These combinations were introduced in Eq.s (2.3) to (2.5), and the result can be expressed in matrix form as:

$$P(A_0 = 1, A_{-1} = 0, \dots, A_{-l_A} = 0, A_{-l_A-1} = 1) = \begin{bmatrix} P_X^g & P_X^b \end{bmatrix} \begin{bmatrix} P_{A,X|X_{-1}}^{1,g|g} & P_{A,X|X_{-1}}^{1,b|g} \\ P_{A,X|X_{-1}}^{1,g|b} & P_{A,X|X_{-1}}^{1,b|b} \end{bmatrix} \begin{bmatrix} P_{A,X|X_{-1}}^{0,g|g} & P_{A,X|X_{-1}}^{0,b|g} \\ P_{A,X|X_{-1}}^{0,g|b} & P_{A,X|X_{-1}}^{0,b|b} \end{bmatrix}^{l_A} \begin{bmatrix} P_{A|X}^{1|g} \\ P_{A|X}^{1|b} \end{bmatrix}. \quad (3.10)$$

We will validate our steady-state analysis by comparison with simulation results in Chapter 5.

3.4 The Compressor Process

Similar to $\{B(n_B)\}$, the compressor process $\{C(n_B): n_B = 1, 2, \dots\}$ is defined only in the events $\{A(n_A) = 1 \forall n_A = 1, 2, \dots\}$. We have found it convenient to represent the compressor process with the following event space:

$$C(n_B) = \begin{cases} 1, & s(n_A) \text{ can be compressed} \\ 0, & s(n_A) \text{ cannot be compressed.} \end{cases} \quad (3.11)$$

Recall from Section 2.2 that the set of parameters defining a ROHC scheme for a single field under study is $\{\Psi^K, w, r\}$ where $\Psi^K = \left\{ \{(b_i, o_f)\}_{i=1}^{K-1}, m \right\}$. The objective is to define

the compressor process $C(n_B)$ given $\{\Psi^K, w, r\}$.

The compressibility of a field depends on a number of issues: (i) the size of the context window, w , (ii) the parameters in the codebook Ψ^K , (iii) presence of packet losses in Channel A, and (iv) the use of intermediate encoding.

To focus on the effect of context window size, we first make the assumption of a perfect Channel A and deal with the case of the simplest codebook ($K = 2$) without any complicating intermediate encoding. The effect of the context window size on the compressibility of a field can be seen from the requirements of robustness in Section 2.2. Note in particular that the compressor encodes the field *only if* this condition is satisfied: The *difference* of the original value with respect to *each* reference value in the window of contexts is *within the interpretation interval* $[-o_f, 2^b - 1 - o_f]$. Given that the compressor is using a single fixed WLSB code (b, o_f) and denoting the interpretation interval $[-o_f, 2^b - 1 - o_f]$ as $V(o_f, b)$, the event of compression success occurs when the difference between $s(n_A)$ and *each* h th previous context lies within the interpretation interval:

$$\begin{aligned} & (s(n_A) - s(n_A - h)) \in V(o_f, b), \quad \forall h = 1, 2, \dots, w \\ \therefore & \left(\sum_{i=n_A-h+1}^{n_A} \delta(i) \right) \in V(o_f, b), \quad \forall h = 1, 2, \dots, w. \end{aligned} \quad (3.12)$$

In practice, header field values increase monotonically in the duration of a flow except when the occasional wraparound occurs. This means that the largest difference between $s(n_A)$ and any $s(n_A - h)$, $h = 1, 2, \dots, w$ is given by $s(n_A) - s(n_A - w)$. Thus, the compression success probability can be evaluated as

$$P(C(n_B)=1|o_f, b, w) = P\left(\sum_{i=n_A-w+1}^{n_A} \Delta(i) \in V(o_f, b)\right). \quad (3.13)$$

Note that the compression success probability is not conditioned on r though it is part of the set of parameters $\{\Psi^K, w, r\}$ in the scheme because it has no effect on the compressibility of a field. However, r determines the scheduling of uncompressed packets. This is to be conditioned in the decompression process.

The significance of Eq. (3.13) is that the probability of compression success depends on up to the $(w-1)$ th order probabilities of the source delta process. As we will later demonstrate, Eq. (3.13) can be used for studying and validating the high order probabilities of a source model. Also, Eq. (3.13) requires a source model to remain reasonably accurate at high orders.

We now discuss the effect of the issue (ii): the parameters in the codebook Ψ^K . This can be done by extending the above case of a single code to the codebook of K codes:

$\Psi^K = \left\{ \left\{ (b_i, o_f) \right\}_{i=1}^{K-1}, m \right\}$ where $K = 2$. Due to the presence of $K-1$ WLSB codes in the codebook, a given field value is considered compressible if *at least one* of the $K-1$ codes encodes the field successfully, i.e. the event of compression success using the codebook Ψ^K is the *union* of compression success events for each WLSB code in the codebook $\Psi^K = \left\{ \left\{ (b_i, o_f) \right\}_{i=1}^{K-1}, m \right\}$ with the following result:

Lemma 1. Given the WLSB codebook $\Psi^K = \left\{ \left\{ (b_i, o_f) \right\}_{i=1}^{K-1}, m \right\}$ in the set of parameters

$\{\Psi^K, w, r\}$ in the scheme, where $b_j > b_i$ if $j > i$, the compression success probability of the codebook is the same as that of the $(K-1)$ th WLSB code (b_{K-1}, o_f) , i.e.

$$P(C(n_B) = 1 | \Psi^K, w) = P(C(n_B) = 1 | o_f, b_{K-1}, w).$$

Proof.

The event $C(n_B) = 1$ occurs when *any* of the $K-1$ WLSB codes in the codebook encodes $s(n_A)$ successfully, i.e.

$$P(C(n_B) = 1 | \Psi^K, w) = P\left(\bigcup_{i=1}^{K-1} C(n_B) = 1 | o_f, b_i, w\right).$$

From Eq. (3.13), we know that the compression success probability of a WLSB code (b, o_f) depends on the size of its interpretation interval and we can modify the above expression as:

$$P(C(n_B) = 1 | \Psi^K, w) = P\left(\sum_{i=n_A-w+1}^{n_A} \Delta(i) \in \bigcup_{i=1}^{K-1} V(o_f, b_i)\right)$$

Due to the property that $b_i < b_j$ if $i < j$, and recalling that $V(o_f, b) \triangleq [-o_f, 2^b - 1 - o_f]$, then it is straightforward to see that $V(o_f, b_i) \subset V(o_f, b_j)$ if $i < j$. It follows by induction that that $V(o_f, b_i) \subset V(o_f, b_{K-1}) \forall i \in \mathbb{Z}^+$ and $i < K - 1$. Therefore, the probability of compression success of the codebook is the same as that of the last WLSB code (b_{K-1}, o_f) . \square

Note that Lemma 1 makes no assumptions on Channel A and its result is equally applicable to non-ideal Channel A conditions. We now consider the effect of packet losses in Channel A by illustration. Suppose for $w = 3$, we have the following sequence of values: $\dots, s(n_A - 7), \text{\textcancel{ $s(n_A - 6)$ }}, \text{\textcancel{ $s(n_A - 5)$ }}, s(n_A - 4), \text{\textcancel{ $s(n_A - 3)$ }}, s(n_A - 2), \text{\textcancel{ $s(n_A - 1)$ }}, s(n_A), \dots$, where the current value, $s(n_A)$, is to be compressed and values marked with double strikethrough were corrupted in Channel A. We find that Eq. (3.13) is no longer

correct since the w th previous value is $s(n_A - 7)$ instead of $s(n_A - 3)$. The number of packets lost between two correctly received packets in the above example is given by values from the loss run process, i.e. $L_A(j_A - 2) = 2$, $L_A(j_A - 1) = 1$ and $L_A(j_A) = 1$.

If we define l_{AT} as the total number of packets lost *since the w th previous context*, then the following result is evident from the above example:

$$P\left(C(n_B) = 1 \mid \Psi^K, w\right) = \sum_{l_{AT}=0}^{n_B - n_A} P\left(\sum_{i=n_A - l_{AT} - w + 1}^{n_A} \Delta(i) \in V(o_f, b_{K-1})\right) \times P\left(\left(\sum_{i'=j_A - w + 1}^{j_A} L_A(i')\right) = l_{AT}\right). \quad (3.14)$$

Note that the difference $(n_B - n_A)$ is used in Eq. (3.14) because for some particular n_A and n_B , $(n_B - n_A)$ is the total number of packets lost in Channel A.

Eq. (3.14) can be used readily if the compressor attempts to use the codebook Ψ^K on the field directly. In ROHC specifications, it is common to find some fields being processed with intermediate encoding before using the WLSB codebook to encode the field into its final compressed form [5]. The motivation for using intermediate encoding is to obtain better gains. Though intuitive, this is yet unproven in open literature. Here, we show how intermediate encoding on the IP Identification (IPID) field can be studied through some simple modification in our compressor process.

The Master Sequence Number (MSN) is a ROHC field which increments for every ROHC packet transmitted to the decompressor within a flow (see Sections 2.1 and 2.3.3). This characteristic is similar to the IPID field. As intermediate encoding it is specified that the MSN field be subtracted from the IPID field (this is in fact performed using an encoding method called ‘INFERRED-OFFSET’). Due to this intermediate

step, the modified input to the WLSB codebook at the compressor becomes $s'(n_A) = s(n_A) - msn(n_A)$. The result of this is that

$$s'(n_A) - s'(n_A - (l_{AT} + w)) = [s(n_A) - s(n_A - (l_{AT} + w))] - w$$

$$\text{or } \sum_{i=n_A-l_{AT}-w+1}^{n_A} \delta'(i) = \left(\sum_{i=n_A-l_{AT}-w+1}^{n_A} \delta(i) \right) - w \quad (3.15)$$

Note that we have used the result $msn(n_A) - msn(n_A - (l_{AT} + w)) = w$ to obtain the above expression. Recall from Section 2.1 that the MSN is a ROHC field *added at the compressor*, and that l_{AT} is the total number of packets lost in Channel A since the w th previous context. Then due to the fact that the MSN field increments for each packet of the same flow received (and transmitted) by the compressor, we yield the above expression.

3.5 The Decompressor Process

The decompressor process $\{D(n_B): n_B = 1, 2, \dots\}$ is defined only in the events $\{A(n_A) = 1 \forall n_A = 1, 2, \dots\}$. We define it as a discrete-time stochastic process with the following event space:

$$D(n_B) = \begin{cases} 1, & s(n_A) \text{ is decompressed successfully} \\ 0, & s(n_A) \text{ is not decompressed successfully.} \end{cases} \quad (3.16)$$

Note that decompression failure arises when a packet fails its CRC integrity check. The reason for this can be attributed to channel errors (corruptions), and/or when the decompression context is invalid. In either case, the packet decodes erroneously. Thus, failure occurs once a packet is corrupted in Channel B, i.e. $D(n_B) = 0$ if $B(n_B) = 0$, but not the converse.

Two key factors determine whether $s(n_A)$ can be decompressed: (i) whether it has been compressed, and if so, (ii) the number of packets between the current packet and the last successfully decompressed packet.

We first consider the effect of whether $s(n_A)$ has been compressed. In the event that $s(n_A)$ was uncompressed, then decompression fails *only if* the packet was corrupted in Channel B. $s(n_A)$ can be uncompressed if $C(n_B)=0$, or if the compressor schedules the n_B th packet to be uncompressed as part of the context refresh procedure. The scheduling of periodic context refresh packets is deterministic. The second factor arises from the fact that a WLSB encoded field can be decoded only if the number of consecutive decompression failures (since the last decompression success) is less than w . We summarize all the above considerations as:

$$P\left(D(n_B)=1 \mid \Psi^K, w, r\right) = \begin{cases} P(B(n_B)=1), n_B \in \{1, r+1, 2r+1, \dots\} \\ P(B(n_B)=1) \left[P(C(n_B)=0) + P(C(n_B)=1) \sum_{l_B=0}^{w-1} P(L_B(j_B)=l_B, D(n_B-l_B-1)=1) \right] \\ \text{, otherwise.} \end{cases} \quad (3.17)$$

We note that the expression $P(L_B(j_B)=l_B, D(n_B-l_B-1)=1) = P(L_B(j_B)=l_B) \times P(D(n_B-l_B-1)=1 \mid B(n_B-l_B-1)=1)$ holds because the probability of decompression success at the (n_B-l_B-1) th packet is independent of all future packet errors over Channel B. Therefore, expressing Eq. (3.17) in conditional probability with further simplification, we have a recursive definition

$$P_{D \mid \Psi^K, w, r, B}^1(n_B) \triangleq P\left(D(n_B)=1 \mid \Psi^K, w, r, B(n_B)=1\right) = \begin{cases} 1, n_B \in \{1, r+1, 2r+1, \dots\} \\ 1 - P(C(n_B)=1) \left(1 - \sum_{l_B=0}^{w-1} P(L_B(j_B)=l_B) \times P_{D \mid \Psi^K, w, r, B}^1(n_B-l_B-1) \right), \text{ otherwise.} \end{cases} \quad (3.18)$$

3.6 Performance metrics in New Perspectives

We show that the modeling framework offers new perspectives on the three performance metrics defined in ROHC [2].

3.6.1 Compression Efficiency

The compression efficiency, or CE , is determined by how much the header sizes are reduced by the compression scheme. We show how it can be defined from our modeling framework.

For the scheme defined by the set of parameters $\{\Psi^K, w, r\}$, the mean compression success probability of the j th WLSB code (b_j, o_j) is the time average of its compression success probability:

$$\bar{C}_j = \begin{cases} \lim_{n_B \rightarrow \infty} \frac{1}{n_B} \sum_{i=1}^{n_B} P(C(i) = 1 | o_j, b_j, w), & j = 1, 2, \dots, K-1 \\ 1, & j = K. \end{cases} \quad (3.19)$$

where from Lemma 1, $\bar{C}_{K-1} = \bar{C}$ is also the mean probability of compression success for the codebook Ψ^K , and $\bar{C}_K = 1$ due to the fact there are no conditions for leaving a field uncompressed.

Note that we have defined the mean compression success probability in the form as shown in Eq. (3.19) for flexibility. In simulation, Eq. (3.19) can be evaluated by using a large number of samples ($n_A \rightarrow \infty$) and each i th term becomes an event with binary outcome (1 or 0). If both source and channel processes are ergodic in the limit $n_A \rightarrow \infty$, then Eq. (3.19) can also be evaluated from steady-state analytical expressions. It has

already been shown that the channel process converges to steady-state probability, and is thus ergodic in the limit $n_A \rightarrow \infty$. We will present in Section 4 an IPID source model which also converges to ergodicity.

Due to the property that $V(o_f, b_i) \subset V(o_f, b_j)$ if $i < j$, The compression success event of each j th code is a superset of preceding codes. We thus obtain the probability of *using* the j th code as:

$$U_j = \begin{cases} \bar{C}_j, & j=1 \\ \bar{C}_j - \bar{C}_{j-1}, & 2 \leq j \leq K. \end{cases} \quad (3.20)$$

In the absence of context refreshes, the mean compressed size can be expressed as

$$\bar{b}' = \sum_{i=1}^{K-1} U_i b_i + U_K m + \eta \quad (3.21)$$

where η is the overhead incurred in discriminator bits. These are sometimes Huffman coded as part of packet format discriminators in ROHC-TCP, and η can be approximated by the entropy of discriminator bits, i.e. $\eta \cong -\sum_{i=1}^K U_i \log_2 U_i$. In other cases, these bits are simply sent as is, and so $\eta = \lceil \log_2 K \rceil$. We assume that Huffman coding is used in all cases.

Factoring in the overhead incurred in the presence of context refreshes of period r , the mean compressed outcome becomes

$$\bar{b} = \frac{r-1}{r} \bar{b}' + \frac{1}{r} m \quad (3.22)$$

and we can then define the compression efficiency of a scheme as

$$\begin{aligned}
CE(\Psi^K, w, r) &= m / \bar{b} \\
&= mr \left[m + (r-1) \left(\sum_{i=1}^{K-1} U_i b_i + U_K m + \eta \right) \right]^{-1}
\end{aligned} \tag{3.23}$$

which takes on values in the range $CE \geq 1$. We can see both from intuition and Eq. (3.23) that $CE(\Psi^K, w, r)$ is a monotonically increasing function of r .

In the limit $r \rightarrow \infty$, the asymptotic upper limit of CE is expressed as

$$\begin{aligned}
CE_\infty(\Psi^K, w) &= \lim_{r \rightarrow \infty} CE(\Psi^K, w, r) \\
&= m \left[\sum_{i=1}^{K-1} U_i b_i + U_K m + \eta \right]^{-1}.
\end{aligned} \tag{3.24}$$

The above result plays an important role in the optimization of the codebook $\Psi^K = \left\{ \{(b_i, o_f)\}_{i=1}^{K-1}, m \right\}$ in a scheme, as will be elaborated upon in Section 3.7. Finally, the definition of CE is non-unique. For example, another definition would be the ratio of the entropy to the mean compressed size.

3.6.2 Robustness

A robust scheme tolerates loss and residual errors on the link over which header compression takes place without losing additional packets or introducing additional errors in decompressed headers [2].

Because WLSB is the only mechanism in ROHC *preventing* damage propagation, instead of *helping recovery* from it, the robustness of a scheme can be seen from its w : a larger w indicates a more robust scheme.

3.6.3 Compression Transparency

The compression transparency, CT , is the extent to which a scheme prevents extra packet loss due to header compression, i.e. packet discards caused by invalid contexts. This can be obtained from the time average of conditional decompression success probability:

$$CT(\Psi^K, w, r) = \lim_{n_B \rightarrow \infty} \frac{1}{n_B} \sum_{i=1}^{n_B} P(D(i) = 1 | \Psi^K, w, r, B(i) = 1). \quad (3.25)$$

One disadvantage of the above expression and Eq. (3.18) is that it is dependent on the probability of compression success, which is in turn dependent on the source model. We can study a less complex relationship by finding the *minimum* compression transparency CT_{min} . Let the compression success probability be 1, i.e. $P(C(n_B) = 1) = 1$. The conditional decompression success probability is no longer dependent on the codebook, and Eq. (3.18) reduces to

$$P_{D|w,r,B}^1(n_B) \triangleq P(D(n_B) = 1 | w, r, B(n_B) = 1) \begin{cases} 1, n_B \in \{1, r+1, 2r+1, \dots\} \\ \sum_{l_B=0}^{w-1} P(L_B(j_B) = l_B) P_{D|w,r,B}^1(n_B - l_B - 1), \text{ otherwise.} \end{cases} \quad (3.26)$$

A result of this is that

$$P_{D|w,r,B}^1(n_B) \leq P_{D|\Psi^K,w,r,B}^1(n_B) \quad (3.27) \\ \therefore CT_{min}(w, r) \leq CT(\Psi^K, w, r).$$

Note that CT_{min} becomes a close approximation of CT if the codebook is designed such that *high compression success probability* is achieved, i.e. $P(C(n_B) = 1) \approx 1 \forall n_B = 1, 2, \dots$. In this case, the compression transparency can be approximated independently of the source, Channel A and compressor processes. In most applications, a high CT is

important because the cost of each extra packet loss is high. A high CE and low CT is in fact detrimental to overall performance because it does not make sense to compress packets into tiny sizes but lose most of them due to invalid contexts. We use CT_{min} to guarantee the *desired level of performance* in our scheme optimization procedure.

3.7 The Optimization of a Scheme

3.7.1 The Goal of Optimization

Given the source process $\{S(n_A)\}$ and channel processes $\{A(n_A)\}$, $\{B(n_B)\}$, the goal of optimization is to find the set of parameters $\{\Psi_{opt}^{K_{opt}}, w_{opt}, r_{opt}\}$ used by the compressor and decompressor processes $\{C(n_B)\}$, $\{D(n_B)\}$ achieving *at least* the desired level of compression transparency, CT_{des} , such that the compression efficiency is maximized.

We have used the notation $\Psi_{opt}^{K_{opt}}$ to denote the optimum codebook at the optimum size of K_{opt} codes. For *any* codebook of K codes in general, where $\Psi^K = \left\{ \left\{ (b_i, o_f) \right\}_{i=1}^{K-1}, m \right\}$ and $K - 1 \leq m$, there can be a total of

$\theta_K = \frac{m!}{(K-1)!(m-K+1)!}$ different codebooks each with a unique combination of

parameters. This is due to the fact that to have any compression gains, i.e. CE is in the range $CE \geq 1$, all b_i 's must be in the range $0 \leq b_i \leq m - 1$. Furthermore, all b_i 's are unique in the codebook. For any particular fixed K , we denote the entire set codebooks of K codes by $\{\Psi_j^K : j = 1, 2, \dots, \theta_K\}$.

3.7.2 The Optimization Procedure

We first derive a result which will be used as part of our optimization procedure:

Lemma 2. *For any fixed w and r , the optimum codebook $\Psi_{opt}^{K_{opt}}(w)$ maximizing the asymptotic Compression Efficiency, $CE_{\infty}(\Psi^K, w)$ also maximizes the Compression Efficiency, $CE(\Psi^K, w, r)$.*

Proof.

The optimum codebook at a fixed w , $\Psi_{opt}^{K_{opt}}(w)$, maximizes the asymptotic Compression Efficiency, i.e. $CE_{\infty}(\Psi_{opt}^{K_{opt}}(w), w) = \max_{j,K} \{CE_{\infty}(\Psi_j^K, w)\}$. We know from Eq. (3.22), Eq.

(3.23) and Eq. (3.24) that $\frac{1}{CE(\Psi^K, w, r)} = \left(\frac{r-1}{r}\right) \frac{1}{CE_{\infty}(\Psi^K, w)} + \frac{m}{r}$. This mapping from

$CE_{\infty}^{-1}(\Psi^K, w)$ to $CE^{-1}(\Psi^K, w, r)$ is affine when r is fixed and thus

$$CE(\Psi_{opt}^{K_{opt}}(w), w, r) = \left[\left(\frac{r-1}{r}\right) \frac{1}{\max_{j,K} \{CE_{\infty}(\Psi_j^K, w)\}} + \frac{m}{r} \right]^{-1} = \max_{j,K} \{CE(\Psi_j^K, w, r)\}. \square$$

We now explain the idea behind our optimization procedure. For each fixed w , we can find the optimum codebook $\Psi_{opt}^{K_{opt}}(w)$ which maximizes $CE_{\infty}(\Psi^K, w)$. From Lemma 2, we know that the resultant codebook at each w is also optimum for $CE(\Psi^K, w, r)$. Next, if we know the set of (w, r) pairs ζ such that the condition $CT_{min}(w, r) \geq CT_{des}$ is satisfied, then regardless of the codebook used, the compression transparency criterion in the goal of optimization will be satisfied in ζ , i.e. $CT(\Psi^K, w, r) \geq CT_{min}(w, r) \geq CT_{des} \quad \forall (w, r) \in \zeta$. Finally, by finding the optimum pair (w_{opt}, r_{opt}) in ζ which maximizes $CE(\Psi_{opt}^{K_{opt}}(w), w, r)$, we will have achieved the goal of

optimization since $\Psi_{opt}^{K_{opt}}(w_{opt})$, w_{opt} and r_{opt} are found. We therefore present a three-step approach to achieve scheme optimization:

Step 1. For *each* given w , find the optimum codebook, $\Psi_{opt}^{K_{opt}}(w) = \{(b_i, o_f), m\}_{i=1}^{K_{opt}-1}$

such that $CE_{\infty}(\Psi_{opt}^{K_{opt}}(w), w) = \max_{j, K} \{CE_{\infty}(\Psi_j^K, w)\}$.

Step 2. Add all parameter pairs (w, r) that satisfies the condition $CT_{min}(w, r) \geq CT_{des}$ to the set ζ , i.e. $(w, r) \in \zeta$ if $CT_{min}(w, r) \geq CT_{des}$.

Step 3. Find the optimum parameter pair (w_{opt}, r_{opt}) such that

$$CE(\Psi_{opt}^{K_{opt}}(w_{opt}), w_{opt}, r_{opt}) = \max_{(w, r) \in \zeta} \{CE(\Psi_{opt}^{K_{opt}}(w), w, r)\}.$$

Note that we have presented a conceptual optimization *procedure* rather than an *algorithm*. An efficient algorithm can be developed to achieve the same conceptual outcome of our procedure. The design of an efficient algorithm is however outside the scope of this thesis. Note also that we have substantially reduced the parameter space by using a constant offset in our codebook $\Psi^K = \{(b_i, o_f)\}_{i=1}^{K-1}, m\}$. Thus, the optimum codebook obtained in Step 1 is optimum only for a particular fixed offset.

3.8 Modeling Different Source and Deployment Scenarios

Different deployment scenarios can be modeled by varying the channel processes $\{A(n_A)\}$, $\{B(n_B)\}$. Note that (i) our modeling framework models the data flow in a single direction, (ii) Channel A is the channel between the source and compressor and (iii) Channel B is the channel between the compressor and decompressor.

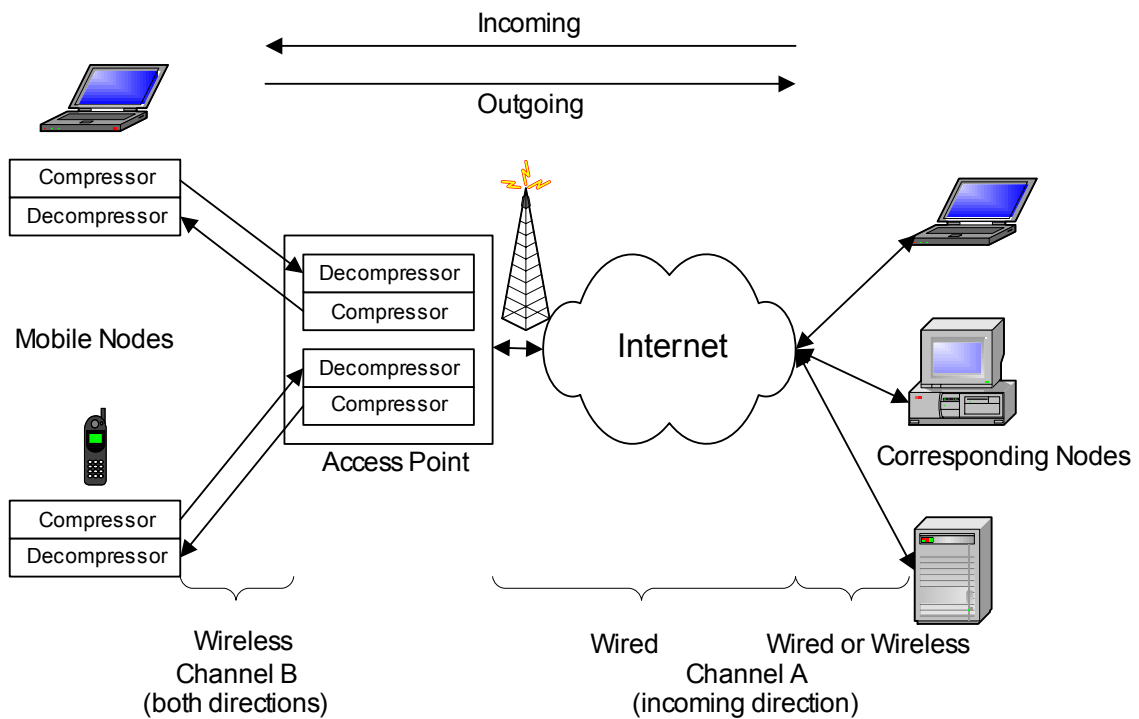


Fig. 6: Header compression deployment over the last hop

Fig. 6 illustrates the common ‘last hop’ deployment of header compression. We define ‘incoming’ as the direction of packet flow towards the clients and ‘outgoing’ to be in the opposite direction. Our modeling framework is easily adapted to this deployment scenario. Regardless of the direction of flow, Channel B is wireless. In the ‘outgoing’ direction, Channel A is a perfect channel (because the source and compressor are co-located); in the ‘incoming’ direction, we have a wired/wireless/hybrid Channel A.

In another scenario where both the corresponding nodes are mobile, or when header compression is deployed over an intermediate wireless hop, then both Channels A and B are simply wireless channels.

Since wireless channels are affected by the effects of fading due to mobility, the

effect of mobility speed can be studied by tuning the channel models for different speeds. Different mobility speeds arise due to different deployment scenarios (e.g. header compression nodes are on a moving vehicle versus a walking pedestrian).

Finally, different source scenarios can be modeled by tuning the model for the source process, $\{S(n_A)\}$ or $\{\Delta(n_A)\}$. We present such a model for IPID in the next section.

Chapter 4 The IPID Source Model

The IPID field has been chosen to illustrate the concept of a source model, because it is one of the few header fields with complicated behavior in header compression schemes, and the *only* commonly used field with complicated behavior in the IP protocol.

We first present the structure of our source model in detail. We then show that it can be built to model real-world traffic traces and validate its accuracy.

4.1 Structure of Source Model

Our approach is to develop a model for the discrete stochastic process $\{\Delta(n_A)\}$ generating IPID deltas $\{\delta(n_A)\}$.

Consider a source generating N flows concurrently. Let each flow f be represented by a Markov chain $\{(f,1), (f,2), \dots, (f,j), \dots\}$ where j is the number of *consecutive packets* in flow f sent by the source *without switching to another flow*. In a state (f,j) , the source is in the process of transmitting the j th consecutive packet from flow f . To transmit the next packet, it either makes a transition to the next state of the same flow $(f,j+1)$ or to the first state of another flow $(f',1)$, such that $f' \in [1, N], f' \neq f$. At each transition, the shared IPID is incremented. For the case of three flows, i.e. $N = 3$, we can visualize the IPID model as shown in Fig. 7.

Let (f,j) and (f',j') be any two states in the model and $q_{(f,j)}^{(f',j')}$ denote the transition probability from state (f,j) to (f',j') . We can make the following general characterizations for any outward transitions from (f,j) :

$$q_{(f',j')}^{(f,j)} = \begin{cases} q_{(f,j)}^{(f,j+1)} & \text{if } f' = f, \text{ where } q_{(f,j)}^{(f,j+1)} \in [0,1) \\ q_{(f',1)}^{(f',1)} & \text{if } f' \neq f, \text{ where } q_{(f',1)}^{(f',1)} \in [0,1] \\ 0, & \text{otherwise} \end{cases} \quad (4.1)$$

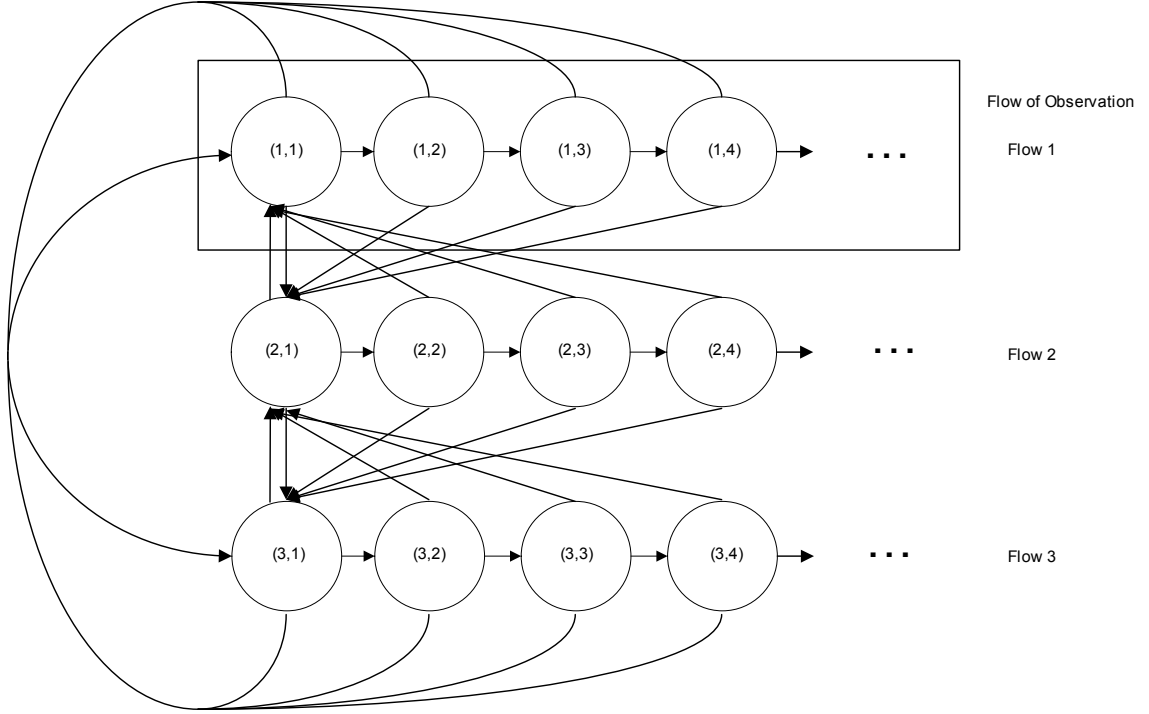


Fig. 7: IPID Markov Model for 3 concurrent connections ($N=3$) and flow 1 is the flow of observation ($f_o=1$)

The notion of N concurrent flows advocates that the transition to the next state of the same flow cannot be made with certainty. Therefore, $q_{(f,j)}^{(f,j+1)}$ is in the range $[0,1)$ as indicated in Eq. (4.1). We also adopt the following notations for compactness:

$$Q(f, j) \triangleq \begin{cases} 1, & j = 0 \\ \prod_{i=1}^j q_{(f,i)}^{(f,i+1)}, & \text{otherwise.} \end{cases} \quad (4.2)$$

It can be seen from the structure of the model that it is indecomposable (there is only one essential state set) and aperiodic. Thus, given sufficient time, it converges to a stationary distribution.

Define $p(f,j)$ as the stationary probability of being in state (f,j) . The stationary probability of any state (f,j) which is not the head of its flow can be expressed as:

$$p(f, j) = p(f, 1)Q(f, j-1), j > 1 \quad (4.3)$$

Thus, we only need to know each $p(f,1) \forall f=1,2,\dots,N$. $p(f,1)$ can be obtained from the balance equation:

$$p(f, 1) = \sum_{\substack{k=1 \\ k \neq f}}^N \sum_{j=1}^{\infty} p(k, j)q_{(k,j)}^{(f,1)}, \forall f \in [1, N] \quad (4.4)$$

Also, we know that all state probabilities must sum up to 1, which can be simply expressed as:

$$\sum_{k=1}^N \sum_{j=1}^{\infty} p(k, j) = 1 \quad (4.5)$$

Using Eq.s (4.3) - (4.5), we can easily solve the stationary state probabilities.

We are interested in deriving $P(\Delta(n_A) = \delta(n_A))$ in terms of Markov state probabilities. Upon convergence, as $n_A \rightarrow \infty$, the stationary state probabilities are fixed and the problem reduces to determining the expression for $P(\Delta = \delta)$ in terms of the stationary state probabilities.

Defining \mathcal{S} as the current state within the flow of observation, we can express $P(\Delta = \delta)$ in terms of its conditional probabilities:

$$\begin{aligned} P(\Delta = \delta) &= \frac{1}{p(f_o)} \sum_{i=1}^{\infty} P(\Delta = \delta | \mathcal{S} = (f_o, i)) p(f_o, i) \\ &= \frac{p(f_o, 1)}{p(f_o)} \left[\sum_{i=1}^{\infty} Q(f_o, i-1) P(\Delta = \delta | \mathcal{S} = (f_o, i)) \right], \end{aligned} \quad (4.6)$$

where $p(f_o) = \sum_{j=1}^{\infty} p(f_o, j)$ is the probability that a packet belongs to the flow of observation f_o . The complexity of the problem lies in the expression $P(\Delta = \delta | \mathcal{S} = (f_o, i))$.

Note that δ is actually the *number of transitions* made between two *observable states*. To the observer, an observation, δ is made *only when* there is a transition *into* any state in f_o . Furthermore, the first $(\delta - 1)$ transitions *since the last observation* must occur *outside* f_o . The remaining transition must be back to f_o (so as to make the observation δ).

The case of $\delta = 1$ is a special case because there is no transition out of f_o , i.e. the next transition must be to the next state of the same flow. If (f_o, i) is the current state, the next state must be $(f_o, i+1)$. Therefore, we can see that

$$P(\Delta = \delta | \mathcal{S} = (f_o, i)) = q_{(f_o, i)}^{(f_o, i+1)}, \text{ if } \delta = 1. \quad (4.7)$$

For $\delta \geq 2$, we only know that the first transition is *out* of f_o , and the last transition is *back* to $(f_o, 1)$. At high N and δ the number of possible paths increases tremendously and it is difficult to obtain a general expression for $P(\Delta = \delta | \mathcal{S} = (f_o, i))$ in closed form. Instead, we can evaluate $P(\Delta = \delta | \mathcal{S} = (f_o, i))$ in the range $\delta \geq 2$ with the help of a recursive function.

We define the recursive function $F((f, i), T, f_o)$, described as “the probability of making $T-1$ transitions *outside* f_o before making the final transition back to $(f_o, 1)$, given the current state is (f, i) ”. This function can thus be defined as:

$$F((f, i), T, f_o) = \begin{cases} q_{(f, i)}^{(f_o, 1)}, & \text{if } T = 1 \\ \left[\sum_{\substack{f'=1 \\ f' \neq f_o, f}}^N q_{(f, i)}^{(f', 1)} F((f', 1), T-1, f_o) + q_{(f, i)}^{(f, i+1)} F((f, i+1), T-1, f_o) \right] \\ \text{, otherwise} \end{cases} \quad (4.8)$$

$P(\Delta = \delta | \mathcal{S} = (f_o, i))$ can be obtained in terms of F as follows:

$$P(\Delta = \delta | \mathcal{S} = (f_o, i)) = \sum_{\substack{f=1 \\ f \neq f_o}}^N q_{(f_o, i)}^{(f, 1)} F((f, 1), \delta-1, f_o), \delta \geq 2 \quad (4.9)$$

Summarizing the above results, we can express $P(\Delta = \delta)$ as:

$$P(\Delta = \delta) = \begin{cases} \frac{p(f_o, 1)}{p(f_o)} \left[\sum_{i=1}^{\infty} Q(f_o, i) \right], & \delta = 1 \\ \frac{p(f_o, 1)}{p(f_o)} \left[\sum_{i=1}^{\infty} Q(f_o, i-1) \sum_{\substack{f=1 \\ f \neq f_o}}^N q_{(f_o, i)}^{(f, 1)} F((f, 1), \delta-1, f_o) \right], & \delta \geq 2 \end{cases} \quad (4.10)$$

Eq. (4.10) can be plugged into the modeling framework at Eq. (3.14) for the analysis of the performance of a scheme designed to compress the IPID. Since it also models the way packets are generated from a source handling multiple concurrent flows, it can also be used in other applications.

Note that different source scenarios can arise causing IPID source behavior to be different. A busy source generating a large number of flows can be intuitively modeled using a high N model. Conversely, a naive source generating only a single, non-concurrent flow is easily modeled with $N = 1$. In Section 4.3, we build a 2-flow model for an average source and a 10-flow model for the busy source.

4.2 Validation of Model

Our model validation approach is to generate from a source terminal a large number of packets from a known number of concurrent flows in controlled environments and capture them right at the source. This ensures that a large number of outgoing packet samples is available for obtaining the Markov transition parameters of the model. Using packet traces, we also verify our assumption in Section 3.4 that the occurrence of wraparound is rare. Having built the model from the packet traces, the distribution $P(\Delta = \delta)$ is then obtained analytically from the model for comparison with the histogram of IPID deltas found in the traces.

Using source terminals running Microsoft Windows, different controlled environments were explored, and our parameter space includes the number of concurrent flows, type of channels, type of applications, type of protocol headers, and the nature of payloads. These variations are tabulated in Table 1.

Table 1
Variations in Controlled Environments

Parameter	Parameter Space
No. of concurrent flows	2 to 4
Type of channel	wired (802.3) or wireless (802.11b)
User application	web browser, remote terminal, LAN gaming, file downloads, file sharing
Protocol headers	{HTTP ¹ , SSH ² , proprietary, FTP ³ , NetBIOS ⁴ } over TCP/IP
Nature of payload	Data or Acknowledgments
¹ HyperText Transfer Protocol	² Secure SHell
³ File Transfer Protocol	⁴ Network Basic Input Output System

After converting the IPID values in the traces into network byte order, we find that the distributions obtained from our IPID model nicely track the IPID distributions obtained from traces in all of the above experiments. We show the distribution comparisons for the case of FTP file download over wireless Ethernet in Fig. 8 and the case of HTTP file download acknowledgments over wired Ethernet in Fig. 9.

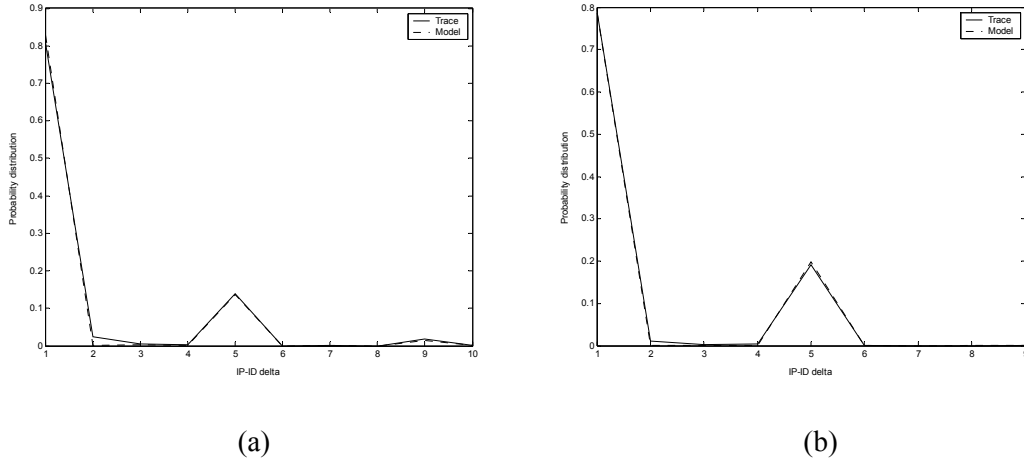


Fig. 8: $P(\Delta = \delta)$ for 4 concurrent flows, generated by FTP file downloads over wireless Ethernet. Flows of observation $f_o = 1$ (a) and $f_o = 2$ (b) are data flows. Distributions from control flows are not obtained due to small numbers of control packets.

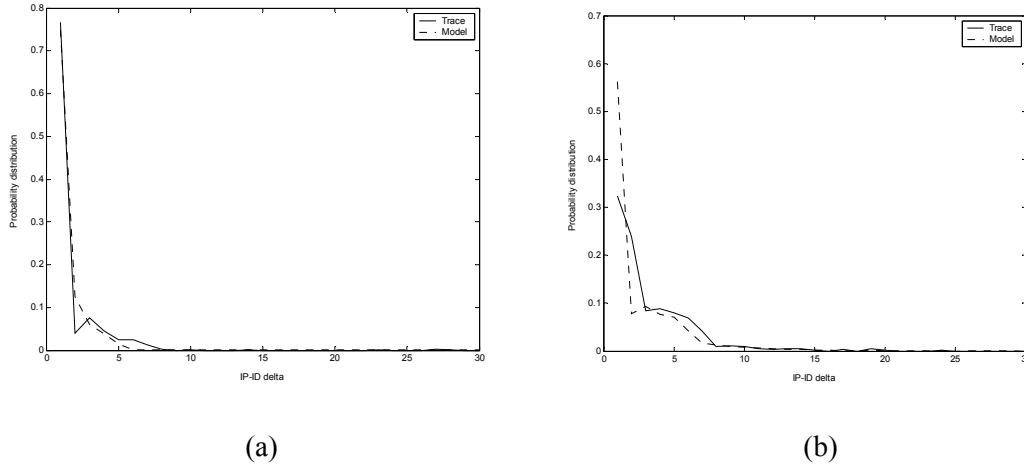


Fig. 9: $P(\Delta = \delta)$ for 2 concurrent flows, generated by HTTP file download ACKs over wired Ethernet. Flows of observation are $f_o = 1$ (a) and $f_o = 2$ (b).

We note from Fig. 8 that the distribution obtained from our model is almost identical to that in the trace. The same level of resemblance is not found in Fig. 9. Examining packets in that trace, we find that for HTTP especially, two consecutive packets generated from the source *may not* carry an increment of +1. Furthermore, packets exhibiting such behavior characteristically show unusually long timestamp lags. This suggests that such packets were interrupted before they could be generated completely. In spite of this, the assumption of +1 increment per packet is usually true and Fig. 9

shows that the distribution obtained with this assumption closely tracks that from traces.

4.3 Constructing a Real-World Source Model

In the previous section we have only verified the concept of a source model in controlled environments. We desire to build a model for the average source in the real world. Based on the IPID delta probability distributions from packets in real traffic, our objective is to find a model which approximates the source at the zeroth as well as higher orders. To do this, we have obtained a trace, TCP080903out which captures 266831 outgoing TCP packet headers from the gateway router of the Institute of Infocomm Research LAN over a half-hour interval.

Essentially, we are trying to obtain the Markov transition parameters from the probability distribution, i.e. performing the reverse of what we did in the previous section. We can see from Eq. (4.10) that this is a non-trivial task due to the infinite number of states and the unknown (and furthermore non-constant) number of concurrent flows, N . Naturally, this requires some assumptions and approximations to be made. We first reduce the search space by truncating the chain of states in each flow. We further assume that the average source can be modeled with 2 concurrent flows, i.e. $N = 2$. We then show how a real-world source model can be built.

4.3.1 Truncating the Number of States

To reduce the complexity of the model, our aim is to truncate the number of states in each flow to a small finite number with little sacrifice in accuracy. To avoid deriving all our earlier source model equations again, we achieve this in two steps: We first approximate the original infinite-states model with a simpler infinite-states model. The approximated infinite-states model is then mapped to an equivalent finite-states model. Using a single flow as an example, this is illustrated in Fig. 10.

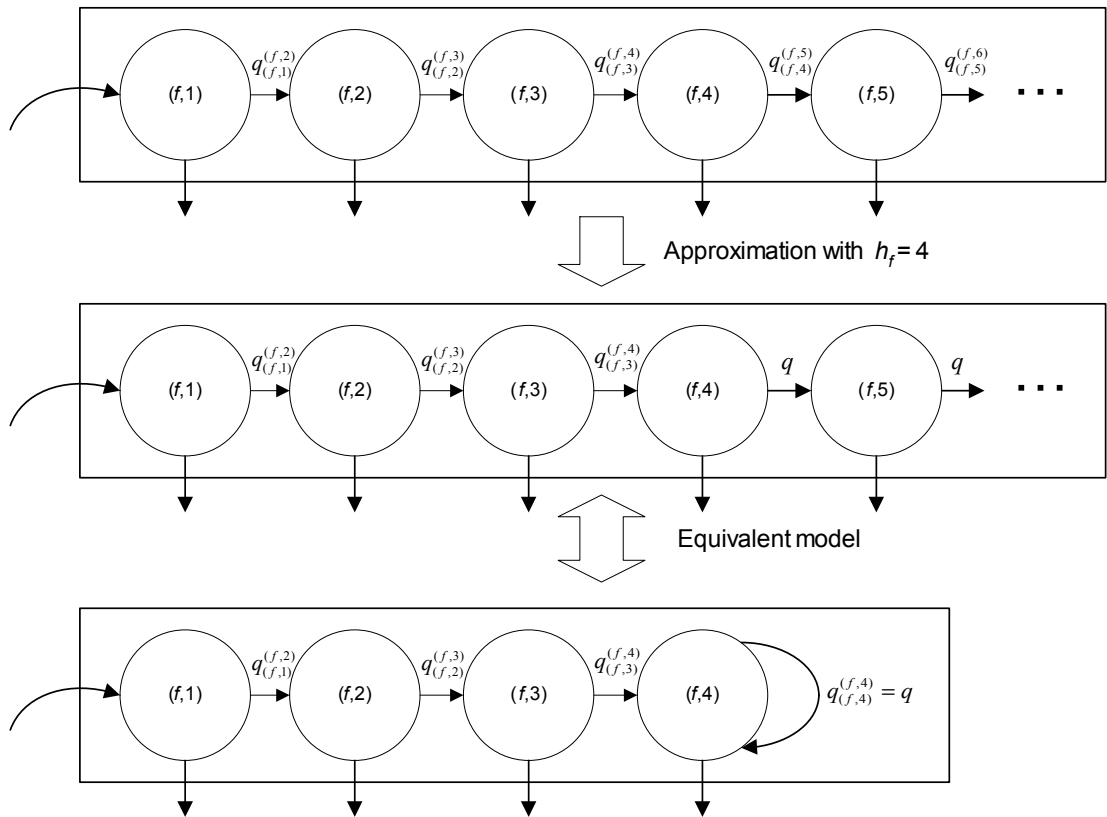


Fig. 10: Illustration of model truncation

For each chain of states representing a flow f , we first approximate the remaining transition probabilities beyond a certain threshold state, h_f , as an averaged constant, q . Then the approximation of $Q(f, j)$ defined in Eq. (4.2) is

$$\hat{Q}(f, j) = \begin{cases} Q(f, j), & j < h_f \\ q^{j-h_f+1} Q(f, h_f - 1), & j \geq h_f \end{cases} \quad (4.11)$$

and the approximation error can be defined as:

$$\varepsilon(f, j) = \left| Q(f, j) - \hat{Q}(f, j) \right|. \quad (4.12)$$

It can be easily shown that $\lim_{j \rightarrow \infty} \varepsilon(f, j) = 0 \forall h_f$, due to the fact that $Q(f, j)$ is in the range $[0, 1) \forall f = 1, 2, \dots, N$ (see Eq. (4.1)). This means that the approximation error decreases to zero if the approximated state is far from the head of its flow, regardless of the threshold value h_f . However, in general h_f remains a tradeoff between model complexity and approximation error.

As illustrated in Fig. 10, the approximated model of infinite states in Eq. (4.11) can be mapped to an equivalent model with h_f states in each flow f , such that the tail state of the flow transits to itself with probability $q_{(f, h_f)}^{(f, h_f)} = q$. To continue using our earlier results, the following corollary is useful for mapping between the approximated model of infinite states and its equivalent model of finite states:

Corollary 1. If the approximated model of infinite states has an equivalent model of

finite states, the following mapping holds:
$$\sum_{j=h_f}^{\infty} \hat{Q}(f, j) = \frac{Q(f, h_f - 1)q_{(f, h_f)}^{(f, h_f)}}{1 - q_{(f, h_f)}^{(f, h_f)}}.$$

Proof.

This result follows from Eq. (4.11) and can be proved easily.

4.3.2 Two-flow Assumption

For the case of 2 concurrent flows, i.e. $N = 2$, Eq. (4.10) can be reduced into a simpler closed form. This is due to the fact that there are only 2 paths out of each state

(see Eq. (4.1)). Furthermore, given any $\delta \geq 2$ there is only a single deterministic path to follow. As such, there is no need to express $P(\Delta = \delta | \mathcal{S} = (f_o, i))$ in terms of the recursive function F . Let the two flows be f_o and f_1 where f_o is the flow of observation. We obtain a simpler form of Eq. (4.10):

$$P(\Delta = \delta) = \frac{p(f_o, 1)Q(f_1, \delta - 2)(1 - q_{(f_1, \delta - 1)}^{(f_1, \delta)})}{p(f_o)}, \delta \geq 2 \quad (4.13)$$

The derivation of Eq. (4.13) is shown in Appendix A.

4.3.3 Resultant Real-World Source Model

We now apply our two-flow assumption and truncation technique and show how the model parameters can be obtained from the probability distribution in the trace. Let the unknown number of states in each flow be h_o and h_1 respectively. Applying the truncation approximation to the two-flow result in Eq. (4.13), we can isolate flow f_1 transition probabilities by obtaining the ratios of consecutive delta probabilities:

$$\frac{\hat{P}(\Delta = \delta + 1)}{\hat{P}(\Delta = \delta)} = \begin{cases} \frac{q_{(f_1, \delta - 1)}^{(f_1, \delta)} (1 - q_{(f_1, \delta)}^{(f_1, \delta + 1)})}{1 - q_{(f_1, \delta - 1)}^{(f_1, \delta)}}, & 2 \leq \delta \leq h_1 \\ q_{(f_1, h_1)}^{(f_1, h_1)}, & \delta \geq h_1 + 1 \end{cases} \quad (4.14)$$

Estimating delta probability ratios from the trace, all flow f_1 transition probabilities can be obtained by solving the set of equations in Eq. (4.14) recursively.

Fig. 11 shows the delta probability ratios obtained from the trace, as well as the number of packets available at each delta value. We observe that the probability ratio gets increasingly jittery as delta increases. The reason for that is attributed to the

decreasing number of packet samples available as delta increases, as shown in the bottom portion of Fig. 11. Fig. 11 also reveals that the probability ratio increases quickly at small deltas and remains relatively constant thereafter. This agrees well with Eq. (4.14) where the probability ratio remains constant at $q_{(f_1, h_1)}^{(f_1, h_1)}$ for $\delta \geq h_1 + 1$. We choose $h_1 = 19$ and obtain $q_{(f_1, h_1)}^{(f_1, h_1)}$ from the ratio mean for $\delta \geq h_1 + 1$. Solving Eq. (4.14) recursively, we then obtain the entire set of transition probabilities in flow f_1 . The values obtained are shown in Table 2.

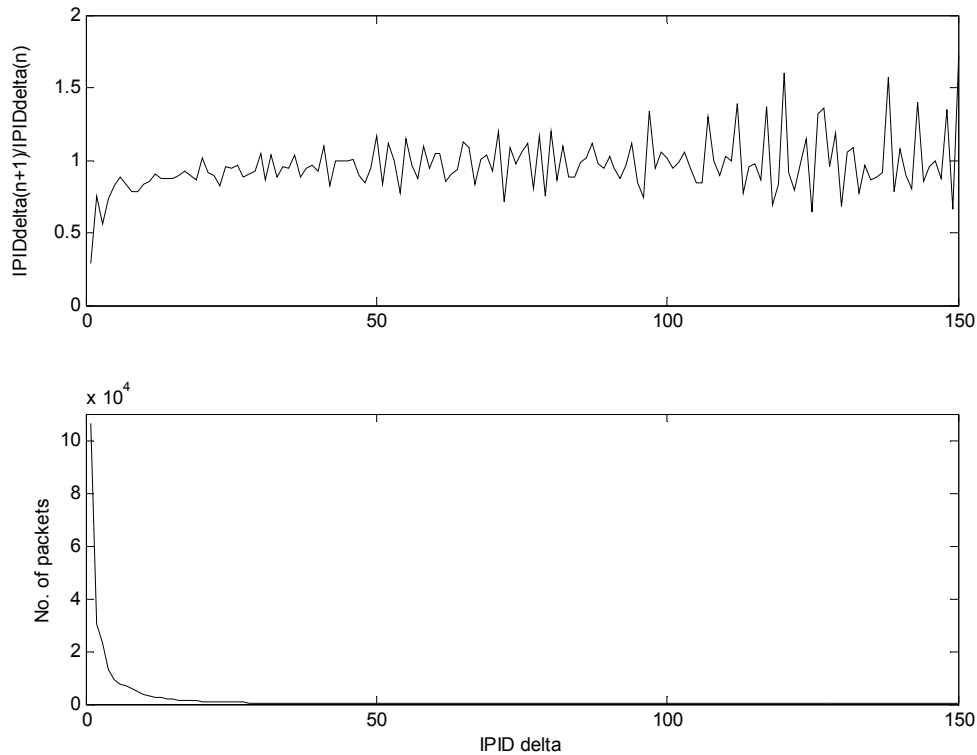


Fig. 11: Estimates of delta probability ratios obtained from trace (top) and corresponding number of packet samples (bottom).

We also exploit another statistical characteristic of the trace to obtain flow f_o transition parameters. Recall from the comments on Eq. (3.13) that a good source model should also approximate the high-order behavior of real sources, as this affects

the compressibility. Therefore, our approach is to obtain flow f_o transition parameters from the high order probability distributions in the trace.

At steady state, the n th order probability distribution can be defined and expressed in the following form:

$$\begin{aligned} \gamma_n &= \hat{P}(\Delta(n_A) = 1 | \Delta(n_A - n) = 1, \dots, \Delta(n_A - 1) = 1) \\ &= \frac{\alpha_n}{1 - q_{(f_o, h_o)}^{(f_o, h_o)} + \alpha_n}, \text{ where } n \geq 1 \text{ and} \\ \alpha_n &= \begin{cases} q_{(f_o, n+1)}^{(f_o, n+2)} \\ \times \left[(1 - q_{(f_o, h_o)}^{(f_o, h_o)}) \left(1 + \sum_{j=n+2}^{h_o-1} \prod_{i=n+2}^j q_{(f_o, i)}^{(f_o, i+1)} \right) + q_{(f_o, h_o)}^{(f_o, h_o)} \prod_{i=n+2}^{h_o-1} q_{(f_o, i)}^{(f_o, i+1)} \right], & h_o \geq 4 \text{ and } 0 \leq n \leq h_o - 3 \\ q_{(f_o, n+1)}^{(f_o, n+2)}, & n = h_o - 2 \\ q_{(f_o, h_o)}^{(f_o, h_o)}, & n \geq h_o - 1 \end{cases} \end{aligned}$$

The derivation of Eq. (4.15) is shown in Appendix B. By estimating each γ_n from the trace and solving Eq. (4.15) recursively, we can obtain all $q_{(f_o, i)}^{(f_o, i+1)}$ in the range $2 \leq i \leq h_o - 1$. The remaining transition probability at $i = 1$, $q_{(f_o, 1)}^{(f_o, 2)}$, can be found using the following relationship:

$$\alpha_0 = (1 - q_{(f_o, h_o)}^{(f_o, h_o)}) \left(\frac{1 - q_{(f_1, 1)}^{(f_1, 2)}}{\hat{P}(\Delta = 2)} - 1 \right) \quad (4.16)$$

The result $q_{(f_o, h_o)}^{(f_o, h_o)} = \gamma_n$, $n \geq h_o - 1$ in Eq. (4.15) requires high order statistics beyond the n th order in the trace to be constant. From Fig. 12, we verify that this is indeed the case as $n \rightarrow 20^-$ and $n \geq 21$. In the region $n \leq 20$, the probability that the next packet generated comes from the same flow (given that the last n packets also came from the same flow) increases asymptotically to 1. The lack of an uninterrupted series of 22 or more packets of the same flow causes the abrupt drop to a constant zero for $m \geq 21$.

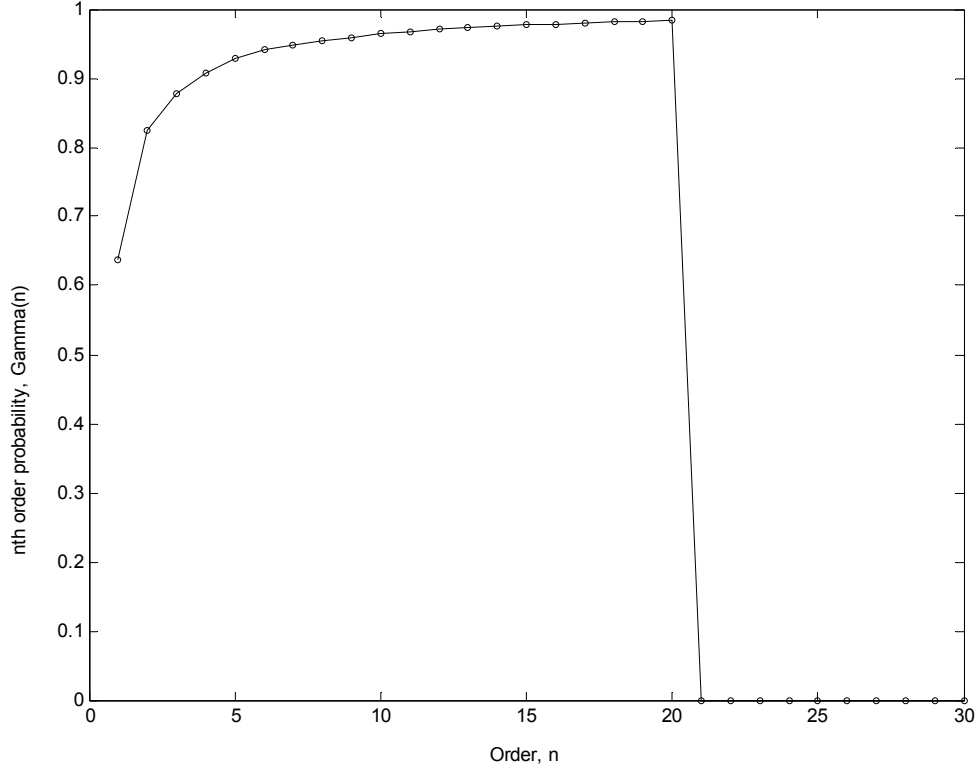


Fig. 12: The estimate of the n th order probability distribution γ_n obtained from packet trace.

Based on the above characteristic, we can either model flow f_o with around 10 states with $q_{(f_o, h_o)}^{(f_o, h_o)} \approx 1$, or obtain an exact 22-state model for flow f_o where $q_{(f_o, h_o)}^{(f_o, h_o)} = 0$. However, we adopt the 10-states model because we find that the abrupt drop to zero in the 22-state model coupled with the two-flow ($N = 2$) assumption produces a worse fit to the trace distribution compared to the 10-state model.

Table 2 shows the resultant parameters of the entire model with the transition probabilities in f_o and f_1 combined. The zeroth order probability distribution obtained from this model is almost identical to the trace distribution as shown in Fig. 13.

Table 2
Markov Model Parameters for state (f_j)

	$j = 1$	$j = 2$	$j = 3$	$j = 4$	$j = 5$	$j = 6$	$j = 7$	$j = 8$	$j = 9$	$j = 10$
$f = f_o$	0.263	0.3082	0.5783	0.6554	0.7124	0.7665	0.8209	0.8397	0.8297	0.96
$f = f_1$	0.7912	0.8005	0.8606	0.8813	0.8891	0.8893	0.8958	0.9086	0.9208	0.9281

	$j = 11$	$j = 12$	$j = 13$	$j = 14$	$j = 15$	$j = 16$	$j = 17$	$j = 18$	$j = 19$	$j = 20$
$f = f_0$	-	-	-	-	-	-	-	-	-	-
$f = f_1$	0.9338	0.9361	0.9403	0.9446	0.9486	0.9518	0.9532	0.956	0.96	-

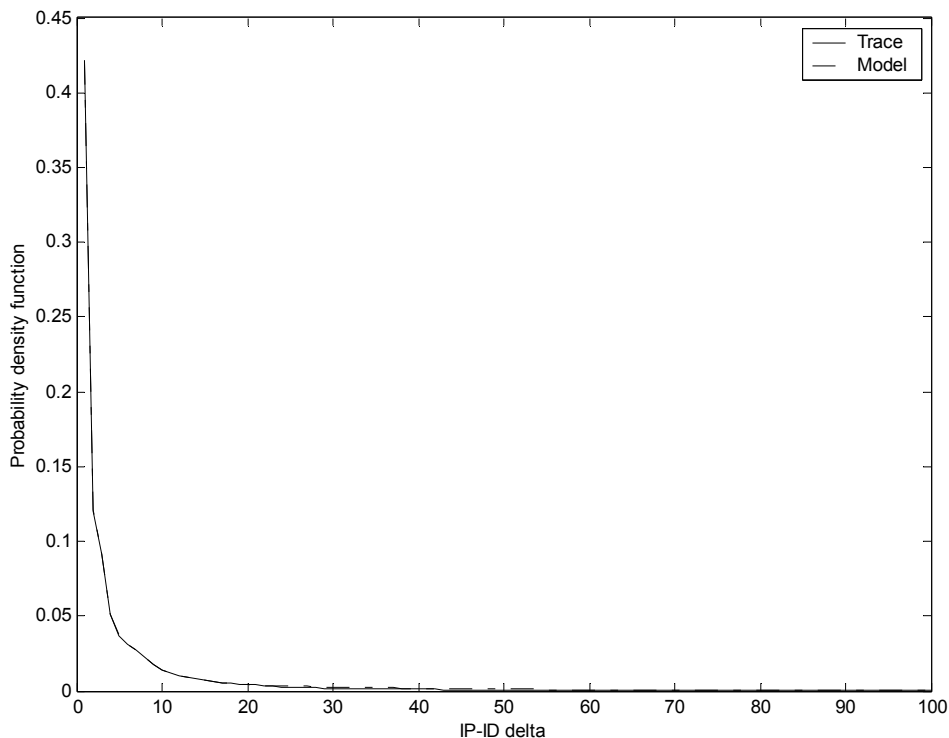


Fig. 13: Comparison of IPID delta distribution between model and trace. The trace distribution is heavy tailed and is truncated to show the fit at small deltas

Fig. 13 shows that the trace distribution is essentially heavy-tailed and has been truncated to show the fit at small deltas. To observe the closeness-of-fit at high deltas, we show the cumulative probabilities on a log scale in Fig. 14. The same figure also shows the distribution comparison at high orders using Eq. (3.13) and Eq. (3.19) (setting $\sigma_f = -1$, since IPID is increasing). Note that this is possible because the n th order cumulative probability of compression success can be obtained using a window size of $(w-1)$ at the compressor with perfect Channel A. Fig. 14 shows that the model distributions track the high order distributions and heavy tail characteristic of the trace rather well. In fact, the probability density function of the source model remains reasonably accurate even at high orders with deviations no more than 0.12. However, the trace distribution is slightly more heavy-tailed. This limit is imposed by our 2-flow

assumption and truncation approximation. However, given the model simplicity, we consider this a good tradeoff.

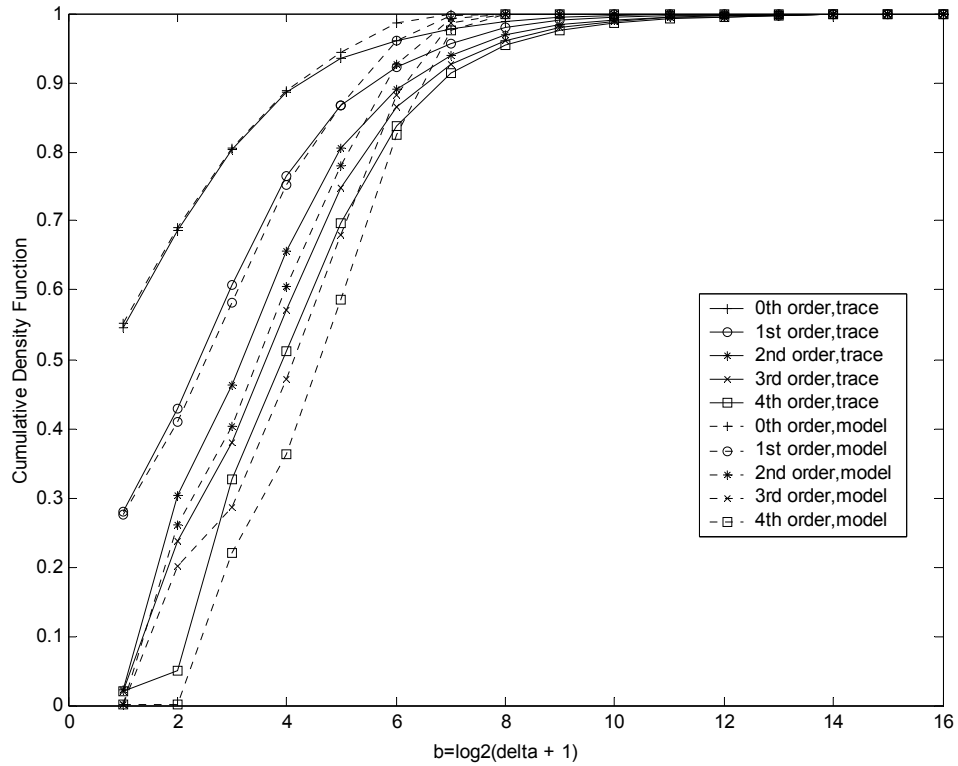


Fig. 14: Comparison of IPID delta cumulative distributions between trace and model on log scale from the zeroth order to the fourth order.

Fig. 15 shows the number of instantaneous concurrent flows generated from a real source. Interestingly, the case of 2 concurrent flows seems to be rather common. We note also that a source may experience short periods of extreme business, seen in the form of spikes, where large numbers of flows are generated.

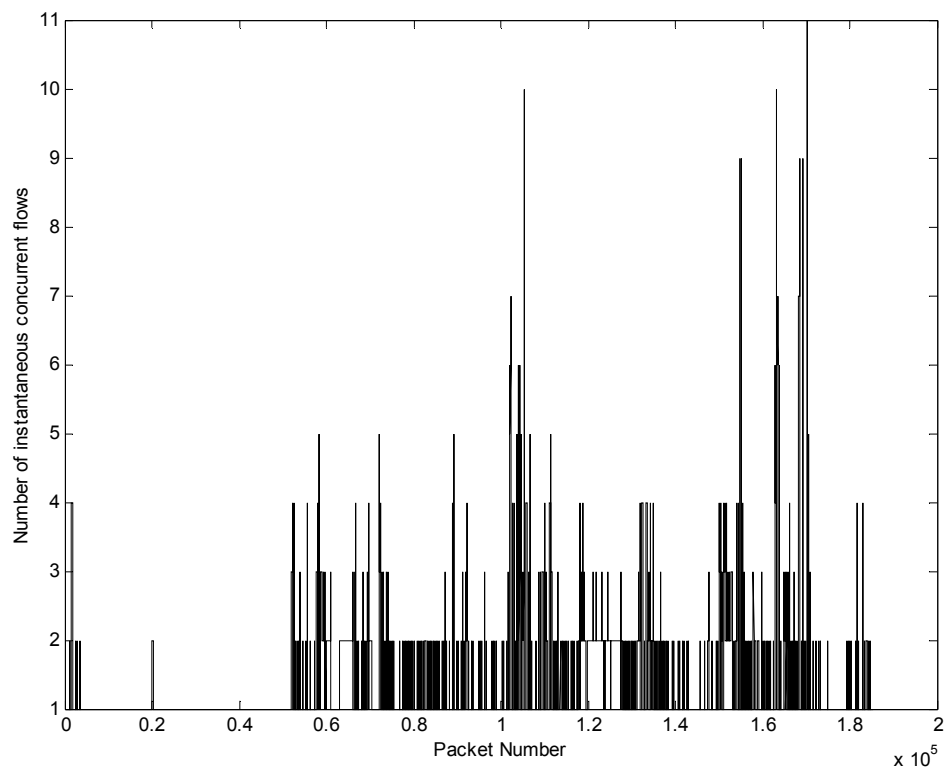


Fig. 15: Number of concurrent flows generated

Our 2-flow model cannot match the heavy tail of the trace distribution exactly mainly because the number of concurrent flows at the source in real life is a *non-constant*. The heavy tail portion is therefore contributed by busy sources generating large numbers of concurrent flows. To verify that this concept is sound, we extended our 2-flow model to a 10-flow model. This is achieved by duplicating 5 copies of each chain of Markov states in the 2-flow model and retaining the transition probabilities down each chain. In Fig. 16, we compare the distribution from the 10-flow model with a particularly heavy-tailed flow in the trace. Note that small spikes are observed from the trace distribution due to limited resolution from small number of packet samples in a single flow. Regardless of this, both distributions share the characteristics of a single large spike at $\delta = 1$ which diminishes to near zero at $\delta = 2$, and an extremely heavy tail. This suggests that the heavy-tail portion unaccounted for by the 2-flow model is in fact contributed by

busy sources with large numbers of concurrent connections. However, the accuracy of the 10-flow model is only weakly justified.

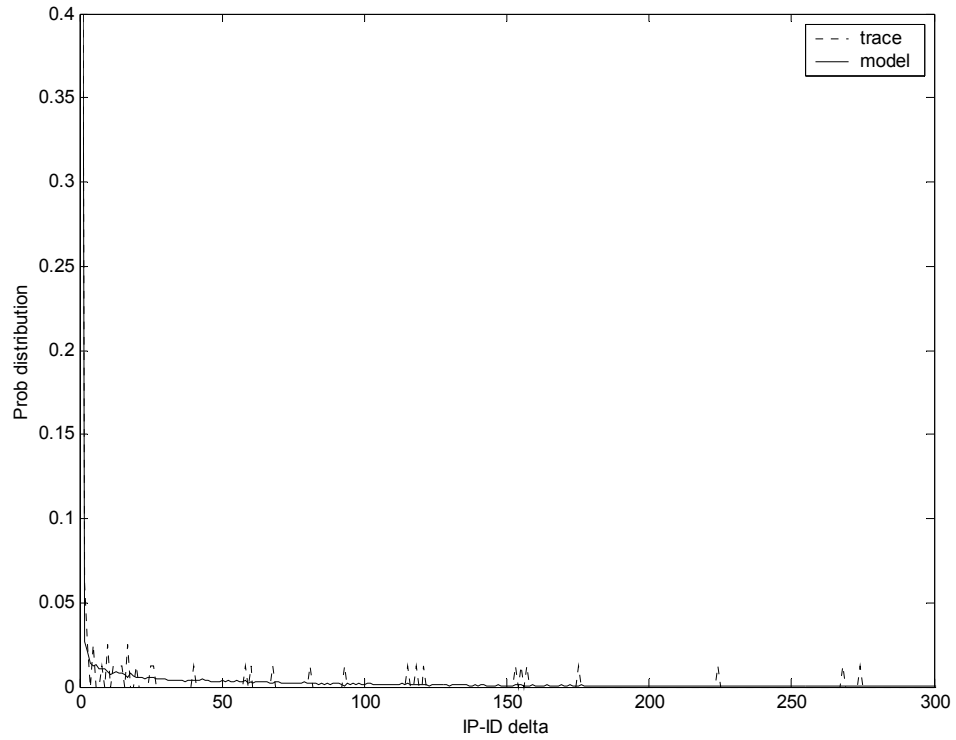


Fig. 16: Comparison of delta distribution from 10-flow model with a heavy-tailed flow in trace. Spikes are seen in the trace distribution due to limited resolution from in limited number of samples for a single flow.

Chapter 5 Results and Discussions

We now use our source model in our modeling framework to study the performance of Robust Header Compression schemes defined by a set of parameters $\{\Psi^K, w, r\}$. Using both analysis and simulation, we evaluate the performance of current ROHC-TCP specifications under different source and deployment scenarios. We then compare the performance of optimized codebooks with and without intermediate encoding, benchmarked against the (unoptimized) ROHC performance. Using numerical examples, we illustrate the concept of tradeoff optimization at the desired level of performance. The optimized schemes obtained are then used in the performance tradeoff study. To illustrate the usefulness and flexibility of our framework and source model, our study involves *different* source scenarios and *different* deployment scenarios.

Three source scenarios are studied: the non-concurrent source (1-flow model) representing an idealized source, an average source (2-flow model) or a busy source (10-flow model). We also study different deployment scenarios by varying the Channel A and Channel B models, examining the scenario of source-compressor co-location (perfect Channel A) or non co-location (wireless Channel A). Channel B deployment scenarios are varied at high speed or low speed mobility in wireless Channel B. We study encoding variations based on whether the WLSB codebook is directly applied on uncompressed fields, or with INFERRED-OFFSET as a prior intermediate step.

Using results from literature [20], two sets of parameters are used to model different mobility speeds as shown in Table 3. We set the uncompressed packet size to 300 bytes (in Channel A) and compressed packet size to 265 bytes (in Channel B). This comes

from the fact that the mean size of data packets is near 300 bytes in the trace, and that header compression typically reduces the 40 bytes TCP/IP header to 5 bytes[‡].

Table 3
Channel Model Parameters

	Low speed (2 km/h)	High speed (50 km/h)
P_{gb}	1.67×10^{-5}	5.35×10^{-5}
P_{bg}	1.66×10^{-3}	4.96×10^{-4}
BER_g	10^{-5}	10^{-5}
BER_b	0.1	0.1

Table 4 shows an extract from the current ROHC specifications for compressing TCP/IP headers (ROHC-TCP). Note that the WLSB offset parameter, o_f , changes from 0 to 1 for the third and fourth code in the codebook. This change is made to accommodate the case of single packet re-ordering in either Channel A or Channel B. Since this change has almost no effect on the interpretation intervals, and our channel processes do not model packet re-ordering, we use a constant $o_f = 0$ in our study. Note also that the context window size and context refresh period are left as implementation parameters with unspecified values in ROHC. We will illustrate the process of obtaining tradeoff optimized values for these parameters.

Table 4
Current ROHC-TCP Specifications

Parameter	Specification
IPID WLSB Codebook, Ψ^K	$\{(0,0), (6,0), (8,0), (11,1), (12,1), 16\}$, with intermediate encoding using MSN
Context Window Size, w	Up to Implementation
Context Refresh Period, r	Up to Implementation

Using our modeling framework and source models, we obtain numerical values for the three performance metrics. We validate our framework by comparing the results from the average source model with those from trace-based simulation. To do this, we obtained a separate trace, TCP180903out, consisting of 285571 packets captured over a

[‡] These are based on data payloads. Per packet gains from real-time applications like Voice over IP (VoIP) are typically much higher due to smaller payloads.

half-hour interval on a *different* day from the previous trace.

Fig. 17 depicts the encoding performance of ROHC-TCP specifications under the three source types (non-concurrent, average and busy) and the two Channel A types (perfect or wireless). This is measured using the inverse of the asymptotic ($r \rightarrow \infty$) Compression Efficiency, $1/CE_\infty$, which gives the ratio of the asymptotic mean compressed length over the uncompressed length (see Eq. (3.24)). Naturally, smaller ratios are indicative of higher compression efficiency and better performance. We first note that the performance obtained from trace-based simulation agree well with analytical results from the average source model, which was built from a different trace. Second, we observe that the compression efficiency is dependent on the nature of the source scenario and Channel A. A busier source deteriorates the compression efficiency, and packet drops in Channel A cause further degradation of performance, though to a lesser extent. Note in particular that the idealized operating environment of non-concurrent source with perfect Channel A gives unrealistically good and constant performance regardless of the context window size, which is far from actual cases found in real world. Thus, over-optimistic conclusions drawn from performance evaluations making this assumption might inadvertently prompt implementations to use large context window sizes for high robustness without clear understanding of its tradeoffs. Third, with the exception of the aforementioned case, the compression efficiency generally decreases with increasing context window size (robustness). This agrees with intuition that there is tradeoff involved between compression efficiency and robustness.

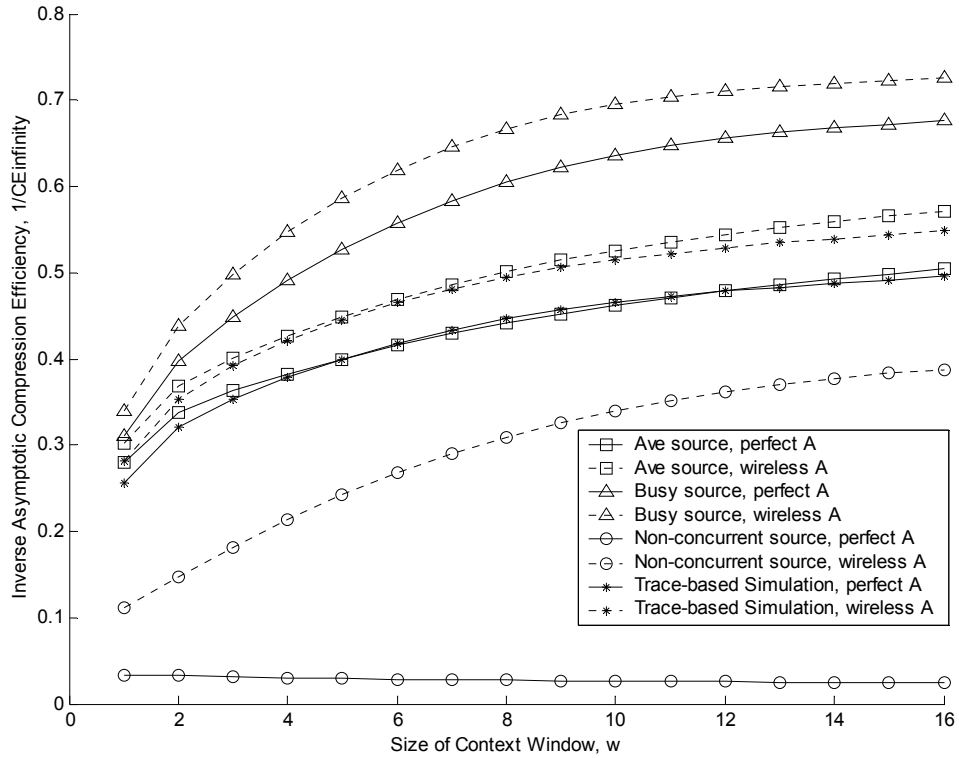


Fig. 17: The encoding performance of ROHC-TCP IPID codebook in the face of various source and Channel A types. The plot shows the inverse of the Asymptotic Compression Efficiency, $1/CE_{\infty}$ versus the context window size, w . $1/CE_{\infty}$ is a ratio of the asymptotic mean compressed length over the uncompressed length. The context window size is also a measure of robustness.

Examine next Fig. 18, which shows the compression transparencies, CT , over wireless Channel B at two different extents of mobility, after compression using the ROHC codebook. Each single curve denotes a fixed context refresh period, r . We observe that for each fixed r , the CT curve obtained from trace-based simulation is mostly close to and above the analytical CT_{min} bound. It is clear from Fig. 18 that CT can be increased by increasing the context window size to improve the robustness of the scheme. However, the choice of (w, r) parameters depends greatly on the extent of mobility in wireless Channel B. Evidently, high speed mobility requires greater robustness in terms of higher w values and/or more frequent context refreshes (lower r values) to achieve the same compression transparency. The significance of this is that these two parameters should be made adaptive if the extent of node mobility is not

expected to be fixed in deployment scenarios.

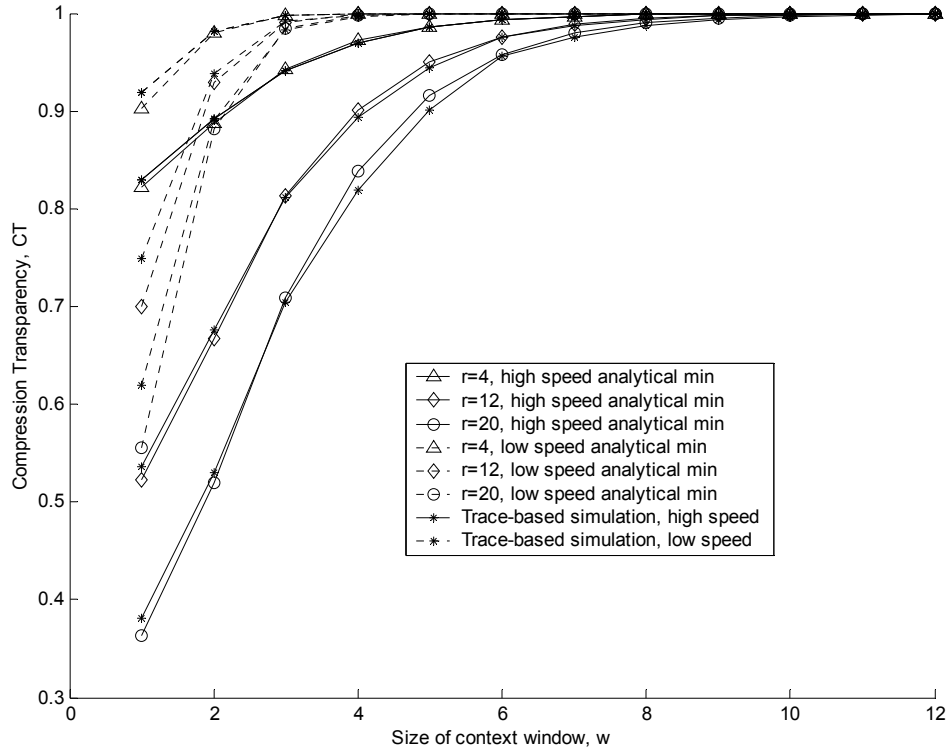


Fig. 18: Compression Transparency in wireless Channel B versus context window size at high speed or low speed mobility. The Compression Transparency, CT , is almost identical to the minimum Compression Transparency, CT_{min} .

The following results demonstrate the 3 steps involved in our optimization procedure in Section 3.7.2. Applying Step 1 of our optimization procedure, we examine the asymptotic compression efficiencies of optimized codebooks $\Psi_{opt}^{K_{opt}}(w)$ using (i) direct WLSB encoding and (ii) with intermediate encoding using the MSN field as described in Sections 2.3.3 and 3.4. Both data sets are benchmarked against the (unoptimized) ROHC codebook in Fig. 19. The case of an average source and perfect Channel A is assumed. As expected, the use of prior offset against the MSN field, as prescribed in ROHC-TCP, increases the compression efficiency. We note that the optimized codebook is non-constant and may change incrementally as w increases. On the other hand, the ROHC codebook remains constant for all context window size w , and it can

be seen that its asymptotic compression efficiencies fall below that from the optimum codebook at all w . In the range $w \geq 3$, the ROHC codebook is not far from optimum. Therefore, the ROHC codebook may be used as a reasonable approximation to $\Psi_{opt}^{K_{opt}}$, which is constant for all w . It will be demonstrated later how the remaining two parameters w_{opt} and r_{opt} in the optimized scheme $\{\Psi_{opt}^{K_{opt}}(w_{opt}), w_{opt}, r_{opt}\}$ may be found.

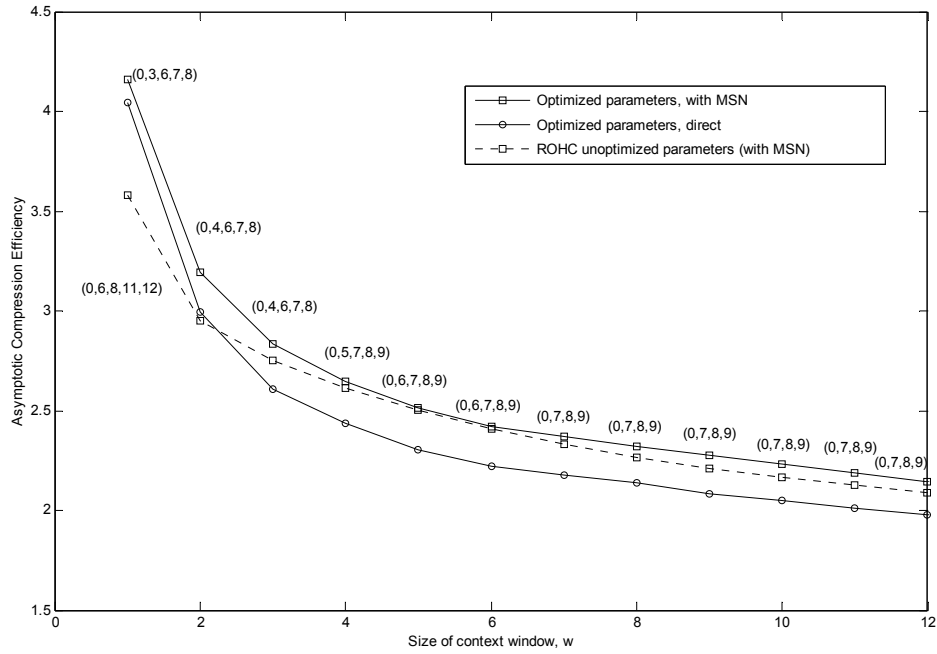


Fig. 19: Asymptotic Compression Efficiencies of optimized codebooks for direct WLSB encoding compared to WLSB with prior offset against the MSN. This is benchmarked against the ROHC codebook. The case of the average source with perfect Channel A is assumed. The set of b-parameters $\{b_i\}_{i=1}^{K-1}$ is shown as a vector for each point.

We examine the reason for the compression efficiency improvement due to intermediate encoding in Fig. 20, which compares the IPID delta cumulative distributions for the two encoding variations on a log scale. Notice the key difference that the set of distributions with intermediate encoding starts from non-zero probabilities at the vertical axis where $b = 0$. This allows WLSB encoded fields to be efficiently encoded into *zero bits* (i.e. which is also known as STATIC encoding) with significant probabilities, lowering the mean encoded size. The choice of the MSN field

as the offset base field achieves this remarkably well as it shares the incrementing characteristic with the IPID field.

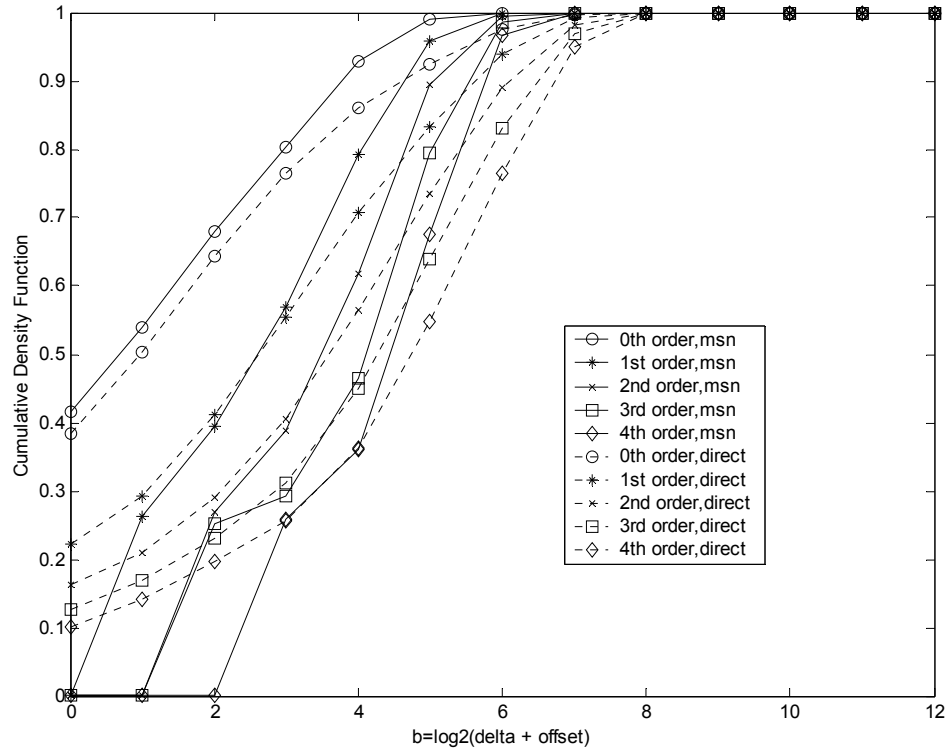


Fig. 20: IPID delta cumulative distribution at high orders, without intermediate encoding (direct), compared to that with intermediate encoding (MSN).

We now illustrate the concept of meeting the desired level of compression transparency guarantee CT_{des} using CT_{min} . Fig. 21 shows two sets of CT_{min} curves, at high speed and low speed mobility in wireless Channel B. The context refresh period, r is fixed in each single curve. We observe that higher context refresh periods result in lower CT_{min} curves. A horizontal line is drawn at $desired CT_{min} = CT_{des} = 0.96$ as an example of indicating the desired level of (minimum compression transparency) performance. Given the extent of Channel B mobility, the points above the form the set of (w,r) combinations guaranteeing that level of compression transparency. In Step 2 of our optimization procedure, all combination pairs satisfying this criterion are put into the set ζ .

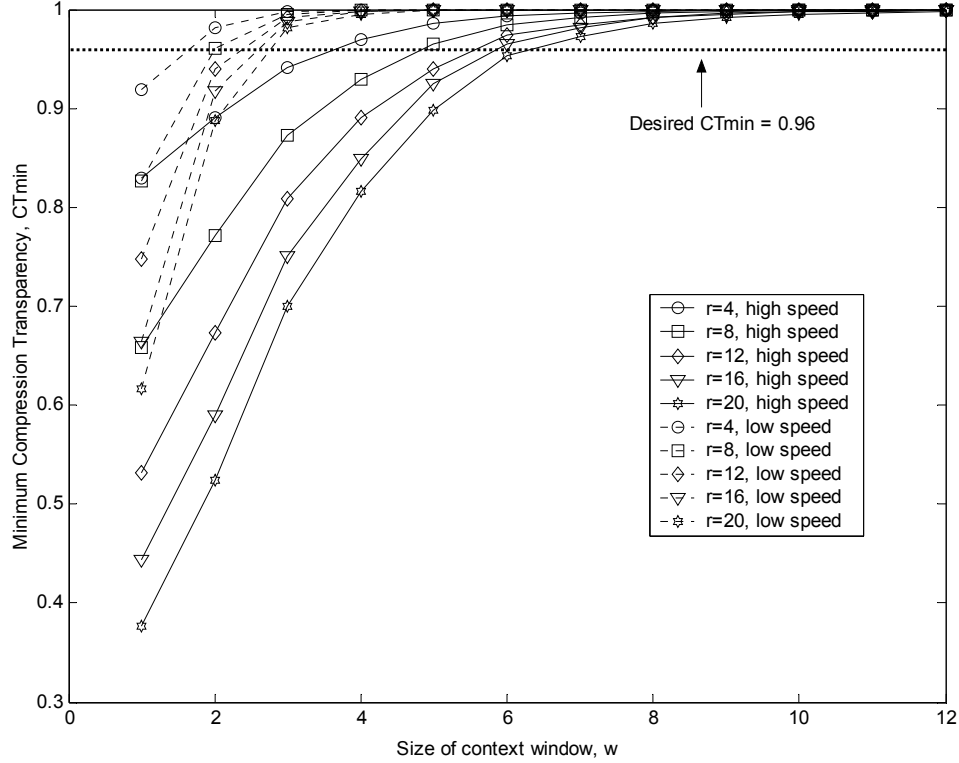


Fig. 21: Variation of minimum Compression Transparency, CT_{min} with context window size or robustness, w .

Having obtained the set of (w, r) combination pairs providing transparency guarantee (for a given channel mobility), the next question arises on *which* is the *best pair* to use. This is determined by choosing the pair with the highest compression efficiency in Step 3 of the optimization procedure. We now consider the spectrum of CE curves for the case of average source and perfect Channel A in Fig. 22. We note that the asymptotic compression efficiency, CE_{∞} is the highest curve at which $r \rightarrow \infty$. A horizontal *desired* CT_{min} line in Fig. 21 indicating the desired level of transparency transforms into a dotted curve in Fig. 22. Agreeing with intuition, curves at higher transparencies suffer slides in compression efficiencies. Another result is that the *same desired level of transparency* is achieved at different compression efficiencies depending on the extent of mobility in wireless Channel B. Finally, the optimum set of parameters $\{\Psi_{opt}^{K_{opt}}(w_{opt}), w_{opt}, r_{opt}\}$ at a desired level of transparency CT_{min} can be found at the

maxima of the *desired* CT_{min} dotted curve in Fig. 22. For example, suppose $CT_{des} = 0.96$ in high speed mobility. In this case, the optimum (w,r) pair, $(w_{opt}, r_{opt}) = (7, 48)$, can be found at the maxima of the $CT_{min} = 0.96$ curve for high speed mobility as shown in Fig. 22. Since w_{opt} is known, the optimum codebook $\Psi_{opt}^{K_{opt}}(w_{opt})$ for this scenario is obtained by referring to Fig. 19, i.e. $\Psi_{opt}^{K_{opt}}(w_{opt}) = \{(0,0), (7,0), (8,0), (9,0), 16\}$.

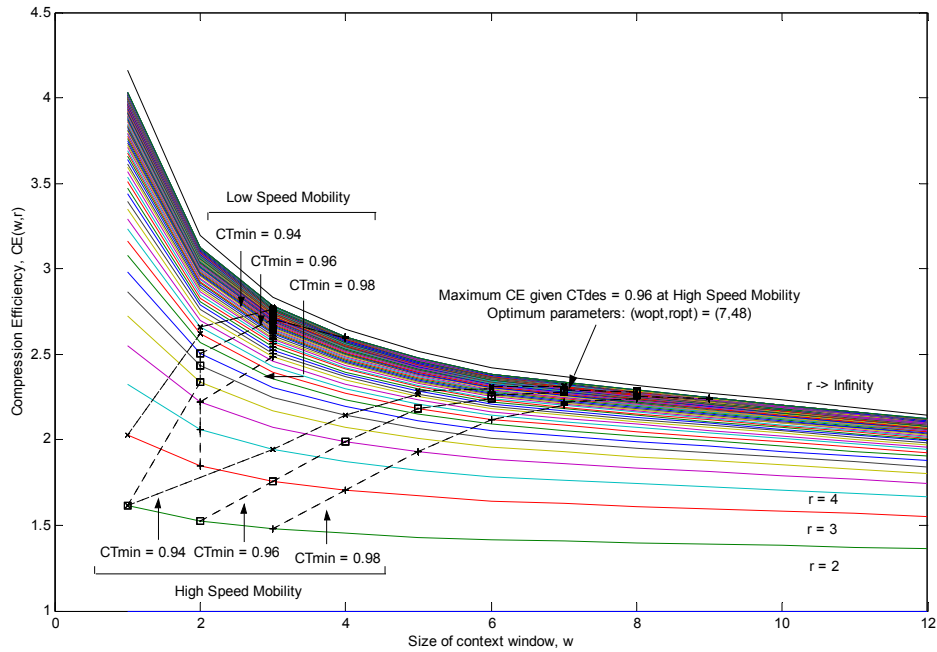


Fig. 22: Compression Efficiencies, CE at various context window size, w and context refresh periods, r for the average source and perfect Channel A.

Fig. 23 shows the tradeoff involved between the optimum compression efficiencies (from optimized schemes) and the desired levels of minimum compression transparency. Note that we have not shown the non-concurrent source results in Fig. 23 because it is simply a horizontal line at $CE_{\infty} = 40$, which is out of scale. Otherwise, due to relatively gentle slopes, we note that the compression transparency improves greatly with relatively little sacrifice in compression efficiency. This is good news for optimized robust schemes. We can see that the tradeoff curve obtained is heavily dependent on the nature of the source and extent of mobility in Channel B. Thus, the

effects of different source and deployment scenarios on header compression are too significant to be ignored.

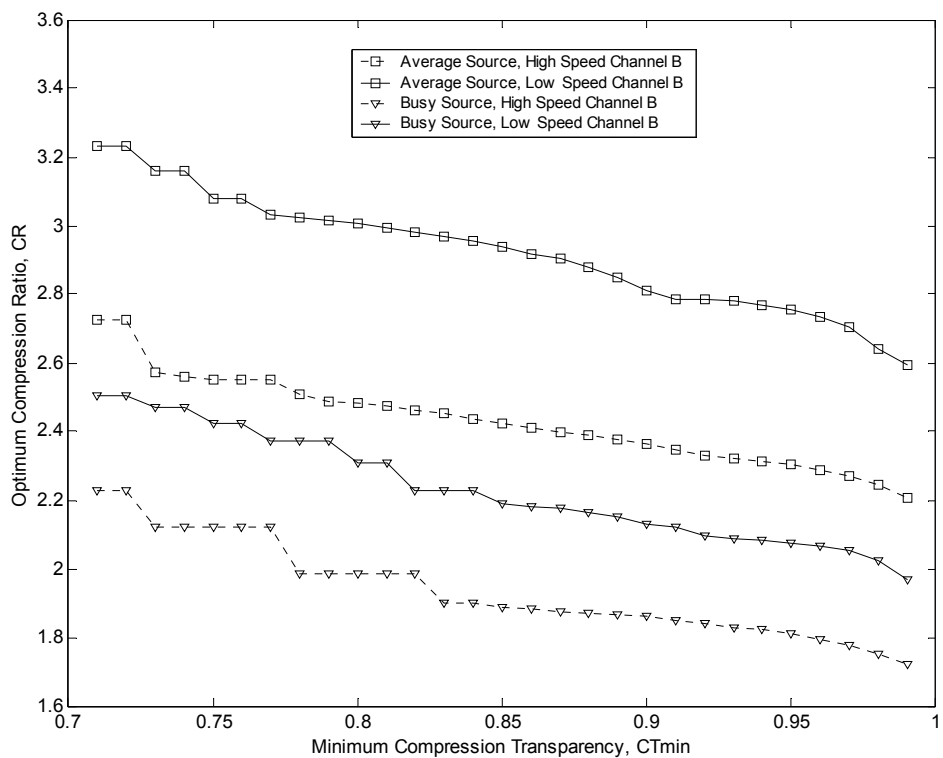


Fig. 23: Variation of optimum Compression Efficiencies with the desired minimum Compression Transparency, showing the tradeoffs involved.

Chapter 6 Conclusion and Future Work

6.1 Conclusion

We have presented some novel contributions in this thesis. For the first time, a source model has been developed for studying header compression. We have shown that the probability density function of our average source model matches that from real-world traces and remains reasonably accurate even at high orders with deviations no more than 0.12. Since it also models the way packets are generated from a source handling multiple concurrent flows, the source model can also be used for more general applications. We have presented a modeling framework allowing the study of header compression performance in different scenarios. Analytical results obtained using the modeling framework agreed well with that obtained via trace-based simulation using a different trace, for the entire practical range of context window sizes. Using our framework, we have offered new perspectives to the definition of performance metrics and studied tradeoffs in a novel way. We have shown, for the first time that header compression schemes can be optimized at desired levels of performance. Our results reveal that the common assumption of non-concurrent sources and perfect Channel A leads to unrealistic asymptotic compression efficiencies which remain high even when the context window size is increased. We have also shown that achieved performance and tradeoffs are heavily dependent on the source and deployment scenarios, and these should not be ignored in both scheme design and performance evaluation.

6.2 Future Work

By proposing our source model, we have taken the first step towards the modeling of real-world operating environments. We acknowledge that the current source model requires further work for improvement. We have seen from Section 4.3.3 that the number of concurrent flows generated by real sources is non-constant and stochastic in nature, posing limitations to the accuracy of our source model. Moreover, we have seen from Section 2.2 that the IPID is not the only CHANGING header field in a TCP/IP header. Future work involves extending the source model to cover multiple CHANGING fields with inter-dependency, and the development of a more accurate model with a non-constant number of concurrent flows.

We have also opened up the possibility of performing adaptive scheme optimization. Though we have demonstrated scheme optimization using the source model in this thesis, the source model is not mandatory for this purpose. A header compression system in deployment can also use its trace sequence to compute performance metrics online, based on which it can adaptively optimize its parameters following the principles developed in this thesis.

REFERENCES

- [1] V. Jacobson, "Compressing TCP/IP Headers for Low-Speed Serial Links", RFC 1144, 1990.
- [2] C. Bormann, et al, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", RFC 3095, July 2001.
- [3] ROHC Charter, <http://www.ietf.org/html.charters/rohc-charter.html>.
- [4] West, M. and S. McCann, "TCP/IP Field Behavior", Internet Draft (work in progress), <draft-ietf-rohc-tcp-field-behavior-02.txt>, March 2003.
- [5] Pelletier, G., Zhang, Q., Jonsson, L-E., Liao, H., West, M., "RObust Header Compression (ROHC): TCP/IP Profile (ROHC-TCP)", Internet Draft (work in progress), <draft-ietf-rohc-tcp-07.txt>, 2004.
- [6] R. Sridharan, R. Sridhar, S. Mishra, "A Robust Header Compression Technique for Wireless Ad Hoc Networks", ACM SIGMOBILE'03, vol. 7, issue 3, pp. 23 – 24, 2003.
- [7] Effnet HC-SIM, http://www.effnet.com/sites/effnet/pages/uk/products_hc-sim.asp
- [8] Acticom ROHC evaluation tool, <http://www.acticom.de/1412.html>
- [9] H. Liao, Q. Zhang, W. Zhu, and Y.-Q. Zhang, "A robust TCP/IP header compression scheme for wireless networks", IEEE International Conference on 3G wireless and Beyond (3Gwireless'01), June, 2001, San Francisco, CA, USA.

- [10] Mikael Degermark, Björn Nordgren, Stephen Pink, "IP Header Compression", RFC 2507, February 1999.
- [11] C. Perkins, J. Crowcroft, "Effects of Interleaving on RTP Header Compression", Proceedings of IEEE Infocom 2000, Tel Aviv, Israel, March 2000.
- [12] P. Seeling, M. Reisslein, F.H.P. Fitzek, S. Hendrata, "Video Quality Evaluation for Wireless Transmission with Robust Header Compression", Proceedings of the IEEE Fourth International Conference on Information, Communications & Signal Processing and Fourth IEEE Pacific-Rim Conference On Multimedia (ICICS-PCM 03), pp 1346-1350, Singapore.
- [13] Gu, X., Hartenstein, H., Fischer, S., "A robust header compression simulator & visualizer", Proc. Int. Conf. on Architectures of Computing Systems, Springer Lecture Notes in Computer Science, Vol. 2299, April 2002, pp. 274-286.
- [14] J. Ash, B. Goode, J. Hand, R. Zhang, "Requirements for Header Compression over MPLS", Internet Draft (work in progress), <draft-ietf-avt-hc-mpls-reqs-02.txt>, June 2004
- [15] H. Wang and K.G.Seah, "An Analytical Model for the ROHC RTP Profile", Proceedings of the IEEE Wireless Communications and Networking Conference 2004, WCNC 2004
- [16] Cédric Westphal, Rajeev Koodli, "IP Header compression: A study of context establishment", WCNC 2003 - IEEE Wireless Communications and Networking Conference, vol. 4, no. 1, Mar 2003 pp. 1025-1031.

- [17] C. Jiao, L. Schwiebert, G. Richard, "Adaptive Header Compression for Wireless Networks", 26th Annual IEEE Conference on Local Computer Networks (LCN'01), pp. 377, November 2001.
- [18] M. Menth, O. Rose, "Performance Tradeoffs for Header Compression in MPLS Networks", Technical Report 291, University of Würzburg Institute of Computer Science Research Report Series, Nov 2001.
- [19] C. Jiao, L. Schwiebert, B. Xu, "On Modeling the Packet Error Statistics in Bursty Channels", 27th Annual IEEE Conference on Local Computer Networks (LCN'02), pp. 534-541, November 2002.
- [20] P. Lettieri, C. Fragouli, M. B. Srivastava, "Low power error control for wireless links," Proceedings of ACM/IEEE MobiCom'97, pp. 139–150, 1997.
- [21] Chia Yuan Cho, Sukanta Kumar Hazra, Winston Khoon Guan Seah, "Exploiting Inter-flow Redundancy: Context Replication in ROHC-TCP", Proceedings of the IEEE Global Telecommunications Conference, GLOBECOM 2004.
- [22] Chia Yuan Cho, Sukanta Kumar Hazra, "Statistical Inter-flow Field Behaviour for Context Replication in ROHC-TCP", *Internet-draft*, <draft-cho-rohc-tcp-interflow-behaviour-00.txt>, Internet Engineering Task Force (IETF), 2004.
- [23] Chia Yuan Cho, Yong Huat Chew, Winston Khoon Guan Seah, "Modeling and Analysis of Robust Header Compression Performance", Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WOWMOM 2005).
- [24] Chia Yuan Cho, Winston Khoon Guan Seah, Yong Huat Chew, "A Framework and Source Model for Design and Evaluation of Robust Header Compression Performance",

accepted by Computer Networks: The International Journal of Computer and
Telecommunications Networking.

APPENDIX A Derivati on of Eq. (4.13)

When $N = 2$, we know from Eq. (4.1) that there are only 2 paths out of each state. For any δ in $P(\Delta = \delta | \mathcal{S} = (f_o, i))$, the first $\delta - 1$ transitions must occur to a state outside the flow of observation, f_o ; the last transition is back to f_o . Because there is only a single flow outside f_o , there is only a single deterministic path to follow. For $\delta = 1$, the result of Eq. (4.10) remains unchanged. For $\delta \geq 2$, using the same logic for evaluating $P(\Delta = \delta | \mathcal{S} = (f_o, i))$ in Section 4.1, it is straightforward to obtain the following expression by following the deterministic path:

$$P(\Delta = \delta | \mathcal{S} = (f_o, i)) = (1 - q_{(f_o, i)}^{(f_o, i+1)}) Q(f_1, \delta - 2) (1 - q_{(f_1, \delta-1)}^{(f_1, \delta)}) \quad (\text{A.1})$$

Substituting into Eq. (4.6), we have for $N = 2$ and $\delta \geq 2$:

$$\begin{aligned} P(\Delta = \delta) &= \left\{ \frac{p(f_o, 1)}{p(f_o)} \left[\sum_{i=1}^{\infty} Q(f_o, i-1) (1 - q_{(f_o, i)}^{(f_o, i+1)}) \right] \times Q(f_1, \delta - 2) (1 - q_{(f_1, \delta-1)}^{(f_1, \delta)}) \right\}, \delta \geq 2 \\ &= \frac{p(f_o, 1) Q(f, \delta - 2) (1 - q_{(f_1, \delta-1)}^{(f_1, \delta)})}{p(f_o)}, \delta \geq 2 \end{aligned} \quad (\text{A.2})$$

The expression inside the square parentheses of the above expression is shown to be 1:

$$\left[\sum_{i=1}^{\infty} Q(f_o, i-1) (1 - q_{(f_o, i)}^{(f_o, i+1)}) \right] = 1 \quad (\text{A.3})$$

Proof.

$$\begin{aligned} & \sum_{i=1}^{\infty} Q(f_o, i-1) (1 - q_{(f_o, i)}^{(f_o, i+1)}) \\ &= \sum_{i=1}^{\infty} Q(f_o, i-1) - \sum_{i=1}^{\infty} Q(f_o, i) \\ &= 1 + \sum_{i=2}^{\infty} Q(f_o, i-1) - \sum_{i=1}^{\infty} Q(f_o, i) \end{aligned}$$

$$= 1 + \sum_{i'=1}^{\infty} Q(f_o, i') - \sum_{i=1}^{\infty} Q(f_o, i), \text{ where } i' = i-1$$
$$= 1 \quad \square$$

APPENDIX B Derivati on of Eq. (4.15)

The n th order delta probability at steady state can be expressed as a fraction of n th and $(n-1)$ th order joint probabilities:

$$\gamma_m = \frac{\hat{P}(\Delta(n_A) = 1, \dots, \Delta(n_A - n) = 1)}{\hat{P}(\Delta(n_A) = 1, \dots, \Delta(n_A - (n-1)) = 1)} \quad (\text{B.1})$$

Consider first the approximated model with infinite states per flow. We can then find the expression for *any* n th order joint probability as:

$$\begin{aligned} & \hat{P}(\Delta(n_A) = 1, \dots, \Delta(n_A - n) = 1) \\ &= \frac{1}{P(f_o)} \left[P(f_o, 1) \prod_{i=1}^{n+1} q_{(f_o, i)}^{(f_o, i+1)} + P(f_o, 2) \prod_{i=2}^{n+2} q_{(f_o, i)}^{(f_o, i+1)} + \dots \right] \\ &= \frac{P(f_o, 1)}{P(f_o)} \left[\sum_{j=1}^{\infty} \prod_{i=1}^{n+j} q_{(f_o, i)}^{(f_o, i+1)} \right] \end{aligned} \quad (\text{B.2})$$

A fraction of the n th and $(n-1)$ th order joint probabilities would yield

$$\begin{aligned} \frac{\hat{P}(\Delta(n_A) = 1, \dots, \Delta(n_A - n) = 1)}{\hat{P}(\Delta(n_A) = 1, \dots, \Delta(n_A - (n-1)) = 1)} &= \frac{\sum_{j=1}^{\infty} \prod_{i=1}^{n+j} q_{(f_o, i)}^{(f_o, i+1)}}{\sum_{j=1}^{\infty} \prod_{i=1}^{n+j-1} q_{(f_o, i)}^{(f_o, i+1)}} \\ &= \frac{\sum_{j=n+1}^{\infty} \prod_{i=n+1}^j q_{(f_o, i)}^{(f_o, i+1)}}{1 + \sum_{j=n+1}^{\infty} \prod_{i=n+1}^j q_{(f_o, i)}^{(f_o, i+1)}} \end{aligned} \quad (\text{B.3})$$

We now adapt Corollary 1 to a more general scenario required in Eq. (B.3) where mapping occurs from state $(f, n+1)$ onwards:

$$\sum_{j=n+1}^{\infty} \prod_{i=n+1}^j q_{(f,i)}^{(f,i+1)} = \begin{cases} \sum_{j=n+1}^{h_f-1} \prod_{i=n+1}^j q_{(f,i)}^{(f,i+1)} + \frac{q_{(f,h_f)}^{(f,h_f)}}{1 - q_{(f,h_f)}^{(f,h_f)}} \prod_{i=n+1}^{h_f-1} q_{(f,i)}^{(f,i+1)}, & 0 \leq n \leq h_f - 2 \\ \frac{q_{(f,h_f)}^{(f,h_f)}}{1 - q_{(f,h_f)}^{(f,h_f)}}, & n \geq h_f - 1 \end{cases} \quad (\text{B.4})$$

Substituting Eq. (B.4) into Eq. (B.3) in the same form as Eq. (4.15) and noting that:

$$\alpha_n = \begin{cases} \left(1 - q_{(f_o,h_o)}^{(f_o,h_o)}\right) \sum_{j=n+1}^{h_o-1} \prod_{i=n+1}^j q_{(f_o,i)}^{(f_o,i+1)} + q_{(f_o,h_o)}^{(f_o,h_o)} \prod_{i=n+1}^{h_o-1} q_{(f_o,i)}^{(f_o,i+1)}, & 0 \leq n \leq h_o - 2 \\ q_{(f_o,h_o)}^{(f_o,h_o)}, & n \geq h_o - 1 \end{cases} \quad (\text{B.5})$$

we obtain Eq. (4.15).