

July, 1976

Report ESL-R-673

PROTOCOL PROBLEMS ASSOCIATED WITH  
SIMPLE COMMUNICATION NETWORKS

by

Roger J. Camrass

This report is based on the unaltered thesis of Roger J. Camrass, submitted in partial fulfillment of the requirements for the degree of Master of Science at the Massachusetts Institute of Technology in February, 1976. This research was conducted at the M.I.T. Electronic Systems Laboratory with partial support provided by the National Science Foundation under Grant NSF-ENG75-14103.

Electronic Systems Laboratory  
Department of Electrical Engineering and Computer Science  
Massachusetts Institute of Technology  
Cambridge, Massachusetts 02139

PROTOCOL PROBLEMS ASSOCIATED WITH  
SIMPLE COMMUNICATION NETWORKS

by

Roger Jeremy Camrass

B.A., Clare College, University of Cambridge

(1972)

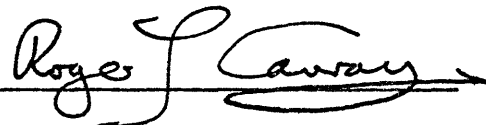
SUBMITTED IN PARTIAL FULFILMENT OF THE  
REQUIREMENTS FOR THE DEGREE OF  
MASTER OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February, 1976

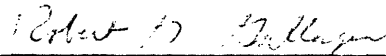
Signature of the Author



Department of Electrical Engineering and Computer

Science, January 20, 1976

Certified by



Thesis Supervisor

Accepted by

Chairman, Department Committee on Graduate Students

PROTOCOL PROBLEMS ASSOCIATED WITH SIMPLE  
COMMUNICATION NETWORKS

by

Roger Jeremy Camrass

Submitted to the Department of Electrical Engineering  
and Computer Science on January 21, 1976, in partial  
fulfilment of the requirements for the Degree of  
Master of Science

ABSTRACT

Control or protocol information must generally accompany messages in a communications network in order to keep track of the beginning, end and destination of each message. Such additional data constitutes a network overhead, and occupies valuable network resources. For economic reasons it is important to keep this overhead to a necessary minimum. An efficient method for encoding protocol information, based on source coding, is applied to the coding of the beginning, end and destination of a message, and the results are compared to existing schemes.

The relationship between protocol information for specifying the beginning and end of a message and its length is illustrated by a single source/receiver network. A Huffman encoding of message length is devised and compared to fixed packet and terminal flag strategies.

A communication link with identical sources is used to demonstrate how start-stop protocols for messages are sufficient to convey destination information in addition to the beginning and end of each message.

The protocol strategies developed from a source coding approach come close to meeting the lower bounds proposed recently for such information.

THESIS SUPERVISOR: Robert.G.Gallager

TITLE: Professor of Electrical Engineering

ACKNOWLEDGEMENTS

The Author wishes to acknowledge his sincere appreciation to his thesis supervisor, Professor Robert Gallager, for his initial theoretical studies on Network Protocols, and for his constant interest and advice during the production of this thesis.

The research was funded by a National Science Foundation Grant NSF - ENG 75 - 14103;

The Plessey Telecommunications Corporation (U.K.), and the personal support of Richard and Maureen Camrass, to whom the author is especially indebted.

Table of Contents

	<u>page</u>
Abstract	2
Acknowledgements	3
Table of Contents	4
List of Figures	6
List of Graphs	6
List of Tables	7
List of Appendices	7
List of Symbols and Abbreviations	8

SECTION

TITLE

1	<u>The Introduction</u>	10
2	<u>A source coding approach to protocol information</u>	14
3	<u>Start-Stop Protocols</u>	17
4	<u>Three Start-Stop Protocols:</u>	23
4.2	Fixed length packet strategy	23
4.5	Huffman encoding of length strategy	28
4.9	Terminating Flag strategy	35
5	<u>Conclusions for Stop Protocols:</u>	58
5.6	Conclusions	65

<u>SECTION</u>	<u>TITLE</u>	<u>page</u>
6	<u>Protocol for a single link with identical sources</u>	68
6.3	Huffman encoding of Start-Stop Protocols	71
6.5	Universal encoding of Start-Stop Protocols	84
6.7	Conclusions on Strategies for encoding protocol	87
6.8	System design implications	88
7	<u>The Conclusion</u>	94
7.2	The design of efficient network protocols	94
7.3	Start-Stop Protocols	95
7.4	Problems of Future Interest	96
	References	98

List of Figures

	<u>page</u>
3-1 A Single Source/Receiver Network	18
4-1 Fixed Packet Strategy	25
4-2 Coding trees for message content and length	30
4-3 Huffman length encoding strategy	32
4-4 Terminating Flag strategy	36
6-1 Link with Identical Sources	70
6-2 Markov Information Source	70
6-3 Coding tree for transitions of sources from idle to busy state	74
6-4 Distance between two idle sources in transition to busy state	76
6-5 Length encoding of distance between transitions at two different time instants	78
6-7 A three source/receiver network	89

List of Graphs

5-1 First moment of message block length	60
5-2 Second moment of message block length	63
6-1 Redundancy of protocol coding for a link with identical sources	83

List of Tables

	<u>page</u>
4-1 Packet length for the fixed packet strategy	46
5-1 Mean value of message protocol data	59
5-2 Second moment of message block length	62

List of Appendices

4a Probability distribution of the number of packets, N	42
4b Optimal packet length for the fixed packet strategy	43
4c Second moment of the sum of two dependent random variables; Message length, M, and number of packets, N.	47
4d Second moment of the sum of two dependent random variables; Message length, M, and number of insertions, I.	50
6 Encoding distance between sources in idle lists at different time instants, j and j+1	90



List of Symbols and Abbreviations

- B Length of message and protocol data
- I Number of insertions in a message
- K Number of sources
- L Length of a packet
- M Length of a message
- N Number of packets containing a message
- Q Customers in a queue
- R Redundancy of final packet
- r+1 Length of terminal flag
- S Length of protocol data for a message
- W Waiting time in service and queue
  
- $\lambda$  Arrival rate of customers in queue
- $\rho$  Traffic intensity for source queue
- $1/\epsilon$  Mean length of a message
- $1/\delta$  Mean length of an idle period associated with a message

To

[Richard+Maureen Camrass]

Section 1

Introduction

1.1 What is a network protocol?

The function of a communications network is to provide a temporary link between an information source (human voice, data terminal, computer) and the appropriate destination. A set of rules are necessary in order to establish and terminate such a connection, and are called network protocols. These rules generally necessitate control information to be transmitted through the network in addition to message data. The control information may be considered as a network overhead, and is called protocol information.

Examples of protocol information include the beginning, the end and the destination of a message, all of which are discussed in this work. Other protocols are associated with network operation, and include routing and supervision.

1.2 Why are network protocols important?

A communications network has finite resources including channel bandwidth and buffer storage, with which to service potential users. Overheads, including protocol data, have an associated cost relating to the bandwidth they occupy and the transmission delays that messages incur due to the accompanying control data. In order to allocate network resources efficiently, overheads must be kept to a necessary

minimum, and this research is directed towards such an end.

In addition to reducing overheads in existing networks, the theoretical study of network protocols can offer some insight on how to improve system design, as will be illustrated in the discussion on addressing information.

### 1.3 Data communication networks

Data communication networks will be used to illustrate the design and evaluation of network protocols. These will be assumed to consist of a finite collection of nodes, to which computers are attached, interconnected by two way noiseless channels of fixed capacity. The nodes are store and forward centers for messages passing through the network.

Messages originate at the computers with random arrival times and data lengths. The network capacity will be assumed sufficient to ensure that despite heavy loading, messages will not incur delays in excess of a specified time during transmission.

In order to concentrate on specific protocols concerned with message lengths and destinations, it is necessary to subdivide the protocols in a network into hierarchical levels including:

- (i) Process to process<sup>1</sup> (programs within computers)
- (ii) Host to host<sup>2</sup> (computer locations)
- (iii) Interconnecting networks<sup>3</sup>
- (iv) Subnetworks or line protocols<sup>4</sup>

Interest in this thesis will be directed towards the fourth

category which includes the transmission of messages between nodes.

#### 1.4 Background to network protocols

The theoretical study of network protocols can be separated into two components; the derivation of lower bounds for protocol information, and the construction of coding schemes to achieve these bounds. Recent work has concentrated on the problem of constructing lower bounds, and has used information theory to represent protocol information by a source code<sup>4</sup>. Based on the results of this work, protocol encoding schemes will be presented which are close to the lower bounds.

The practical development of distributed computer networks originated in the 1960's with the ARPA NETWORK<sup>1</sup>. This system uses a packet switching approach, in which messages are subdivided into packets, each containing address and length information. The packets are independently routed through the network and assembled at the destination. A more recent system replaces the packet by a statistical multiplexor technique (as used by Codex). This allocates to each source sharing a common channel, a separate variable length time slot for its contents.

These two approaches will be used to illustrate how an understanding of the nature of protocol information can suggest systems which have practical application in existing and future networks.

## 1.5 Outline of research

The following section will indicate which results from information theory have application to the understanding of protocol information. Sections 3,4 and 5 present a detailed discussion of start-stop protocols for a single source and receiver communicating over a channel. This information allows the receiver to identify the beginning and end of a message in a continuous stream of binary data.

Three protocol strategies for conveying start-stop information are described and compared in section 4. These include modified examples of existing schemes including fixed and variable length packets, and terminal flags.

The concept of address information is introduced in section 6, when many source/receiver pairs communicate over a single channel. Two coding strategies for start-stop and destination information are described and compared, including a Huffman and Universal coding scheme.

Finally in sections 6.8 and 7, mention is made of the application of the protocol strategies to practical networks, together with some comments on the success of the source coding approach.

## Section 2

### A source coding approach to protocol information

#### 2.1 Introduction

The necessity for protocol information in a communications network is essentially to resolve the statistical uncertainties associated with incoming messages; including arrival times, message length and destination. Information theory helps derive a lower bound on such information, and suggests in some cases an encoding scheme which achieves that bound<sup>4,5</sup>.

Two concepts from information theory which include source codes and source entropy will be discussed briefly, before passing onto practical coding schemes for encoding protocol information.

#### 2.2 A source code

Consider the protocol information which describes message length. If the length is a random variable, then each element,  $a_k$ , belonging to the set of all possible lengths,  $X$ , can be described by its probability of occurrence  $P_X(a_k)$ . The probability function,  $P_X$ , forms a complete statistical characterization of the information source,  $X$ . The protocol information describing message length is, to the information theorist, a source.

For any information source,  $X$ , there is a quantity called the source entropy or self information,  $H(X)$ . The entropy represents a lower bound for the average number of binary digits,  $\bar{n}$ , required to encode each

source letter,  $a_k$ . For a source with  $K$  elements,

$$H(X) = \sum_{k=1}^K P_X(a_k) \log \frac{1}{P_X(a_k)} \quad (2.2.1)$$

The lower bound is expressed in a source coding theorem which states that for a source,  $X$ , it is possible to assign prefix codewords to the source letters,  $a_k$ , in such a way that the average length of a codeword,  $\bar{n}$ , satisfies

$$\bar{n} < H(X) + 1 \quad (2.2.2)$$

and for a uniquely decodable set of codewords

$$\bar{n} \geq H(X) \quad (2.2.3)$$

The theoretical limit of (2.2.3) can be approached by employing efficient coding techniques including the Huffman code<sup>6,7</sup>.

### 2.3 Start-stop information

A source of binary data can exist in either of two states; the idle state during which it generates idle characters, and the busy state during which it generates messages. The start-stop information need only express the lengths of each consecutive state, i.e., the idle and busy periods. To appreciate this, consider a receiver which is informed of the initial state of the source, and the lengths of all subsequent states. It will then be able to reconstruct from the incoming data stream the idle and busy periods.

Coding of start-stop information involves two independent information sources; one belonging to the idle period, and one to the busy



period. The source elements are the different lengths of the idle or busy states. According to the inter arrival time and message length statistics, the entropy of the sources can be calculated using (2.2.1), and an efficient coding scheme designed to meet bounds (2.2.2&3).

Poisson arrivals and geometric length statistics will be assumed in the discussion of start-stop information which follows.

## Section 3

### Start-Stop Protocols

#### 3.1 Single source/receiver model

The model proposed to investigate start-stop protocols consists of a single data source communicating with a receiver through a fixed capacity (one digit/second) channel. Between the source and the channel is placed a node (data processor) acting as a buffer for incoming messages, and able to generate protocol information necessary for communication (Fig. 3.1).

The source generates messages with interarrival times modelled by the poisson process, and lengths described by a geometric probability distribution. Each message, upon arrival from the source, joins a queue at the source node.

#### 3.2 Encoding start information

To appreciate the significance of start-stop information, consider the above system in an idle state (i.e., the source node contains no messages). After a random interval of time, a message is generated by the source and joins the empty queue, awaiting transmission. The node must communicate the change of state to the receiver before transmitting data.

It is not possible to predetermine the length of the idle period until the next arrival occurs, so the source must send frequent state information to the receiver. Any attempt to encode this information into

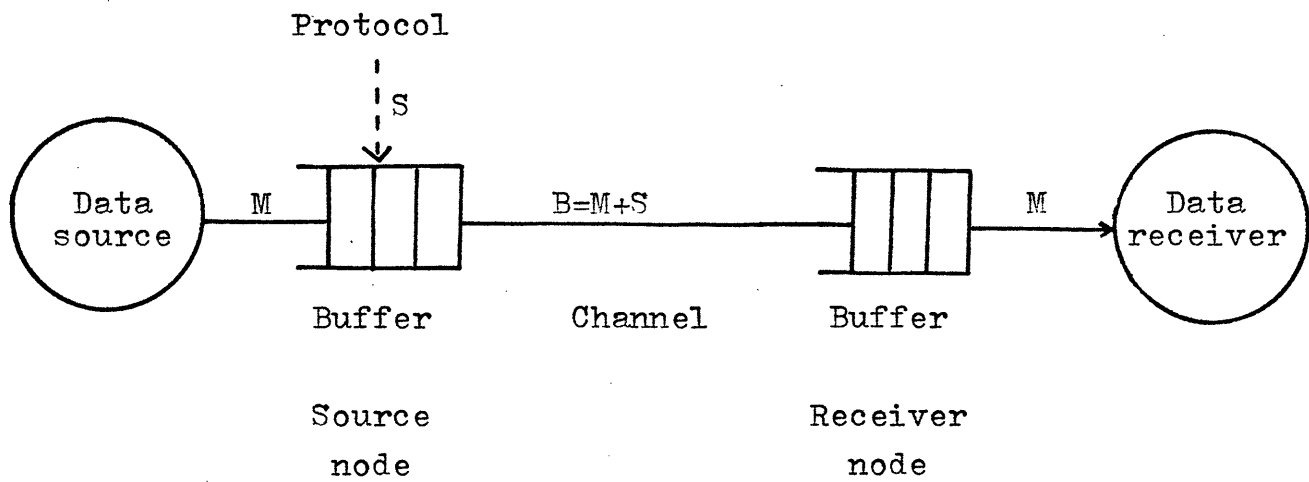


Figure 3.1

Single source/receiver link

a reduced form will result in probable message delays.

For example, consider sending only one idle character (say a 0) for every L seconds spent in the idle state. Should a message arrive during the intervening period, a delay of up to L seconds will be incurred before the receiver is informed of the change of state.

To avoid such a delay, the following strategy is an obvious choice. During the idle period, idle characters (for example, binary zeros) are transmitted every second. On arrival of a message to the source node, a busy character is transmitted (say a binary one). This strategy ensures minimum delay for message data at the expense of a less efficient encoding of idle characters.

Accepting the idle characters as a necessary cost for avoiding delays, the single start bit (indicating the transition to the busy state) must be included as part of the start protocol information accompanying each message.

### 3.3 Encoding stop information

Once the source node enters the busy state, it remains there for at least one message. Protocol information must be sent to the receiver to indicate the end of the message. It is sufficient to send an encoding of the message length itself as stop information. An efficient coding scheme exists for this purpose, and is discussed as a possible optimal strategy for stop protocol information.

Two other approaches exist in practical networks, and are compared in efficiency with the proposed optimal scheme. The first employs

packets of fixed length, and the second places a unique flag at the end of the message data. The method of comparison is based on the estimation of mean message delays at the source node. An M/G/1 queueing process is used to analyze the model on account of the special source statistics.

On completed transmission of a message, the receiver awaits state information from the source before accepting a following message.

### 3.4 M/G/1 Queues

Messages arriving at the source node form a Poisson input queue with mean arrival rate of  $\lambda$ . An addition of S bits of protocol information is made to each message of length M, generating a combined block length of B bits.

$$B = M + S \quad (3.4.1)$$

The channel, considered as a server, takes B seconds to transmit each message in the queue, and has an arbitrary service rate of  $E(B)^{-1}$  seconds<sup>-1</sup>, dependent on the protocol strategy used to generate S. The queueing process is therefore described by an M/G/1 model; poisson input and general service time.

The Pollaczek-Khintchine formula provides a value for expected system size,  $E(Q)$ , that is the number of messages in the queue and in service, in terms of traffic intensity,  $\rho$ , arrival rate,  $\lambda$ , and variance of service time,  $\text{var}(B)$ .<sup>8</sup>

$$\rho = \lambda E(B) \leq 1 \quad (3.4.2)$$

$$E(Q) = \rho + \frac{\rho^2 + \lambda^2 \text{var}(B)}{2(1-\rho)} \quad (3.4.3)$$

The expected system size,  $E(Q)$ , may be expressed in terms of the first and second moments of service time, by expanding the variance of (B).

$$E(Q) = \lambda E(B) + \frac{\lambda^2 E(B^2)}{2(1-\lambda E(B))} \quad (3.4.4)$$

The expected waiting time,  $E(W)$ , in the queue and in service, can be obtained by Little's formula.

$$E(W) = E(Q)/\lambda \quad (3.4.5)$$

The dependence of waiting time on first and second moments of block length carries some implications for an efficient protocol strategy. It should employ a minimal average number of bits,  $E(S)$ , and have a small variance associated with this mean. Mean waiting time is given by:

$$E(W) = E(B) + \frac{\lambda E(B^2)}{2(1-\lambda E(B))} \quad (3.4.6)$$

### 3.5 Minimizing protocol information

Associated with all protocol strategies discussed herein, there appears an independent parameter,  $L$ , associated with the function,  $S$ . For example,  $L$  is the length of a packet in the fixed packet strategy. Block length is dependent on  $L$  through variable  $S$ ,

$$B(L) = M + S(L) \quad (3.5.1)$$

To ensure a meaningful comparison between queue delays in different protocol strategies, it is necessary to optimize the value of  $L$ . A suitable criterion for optimization, is to choose  $L$  to minimize expected block length, or equivalently mean protocol information,  $S(L)$ .

$$\begin{aligned} \left. \frac{dE(B)}{dL} \right|_{L=L^0} &= 0 \\ &= \left. \frac{dE(M)}{dL} \right|_{L=L^0} + \left. \frac{dE(S(L))}{dL} \right|_{L=L^0} \end{aligned} \quad (3.5.2)$$

The moments of service time, B, can then be calculated in terms of  $L^0$ , allowing a meaningful comparison of E(W) between strategies.

### 3.6 Summary of the analytical technique

It is proposed to compare three protocol strategies for transmitting start-stop information across a single channel, by calculating mean waiting time of messages arriving at the source node. Waiting time is of practical significance in data networks and is a useful indication of the efficiency of a protocol scheme.

In order to calculate waiting times, the first and second moments of block length, B, must be derived (block length, B, includes both message and protocol data).

The three protocol strategies of interest are the fixed packet strategy in which messages are subdivided into fixed length sections, the terminal flag strategy, and a scheme based on Huffman encoding. The last strategy corresponds to a variable length packet approach, and has similarities to schemes implemented in packet switched networks.

## Section 4

### Three start-stop protocol strategies

#### 4.1 Introduction

In the previous section, a single source/receiver model was proposed on which to evaluate different start-stop protocols, together with a criterion for assessing their relative efficiencies in terms of queueing delay. This section will examine three different strategies all of which have practical counterparts, although in somewhat modified forms. One strategy, the Huffman encoding of length, is based on source coding ideas.

The comparison between the three strategies is intended to illustrate the performance of schemes devised as practical solutions in operating networks against a theoretical solution advanced in the thesis.

#### 4.2 Fixed length packet strategy

Each message is transmitted in a sequence of fixed length packets, of L bits. For a message of M bits, the number of packets employed, N, is given by:

$$N = \left\lceil \frac{M}{L} \right\rceil * \quad (4.2.1)$$

The message length, a random variable, is not usually an integer multiple of L, causing redundancy in the last fixed packet of R bits. A length specifier, placed at the end of the message, encodes the useful

---

\* The integer value greater or equal to (M/L)



number of message bits,  $(L-R)$ , in the last packet. If  $L$  is an integer power of 2, then a fixed codeword of length  $\log_2 L$  is sufficient to encode these bits. Otherwise a set of variable length words, some shorter and some longer than  $\log_2 L$  but having the same mean value, must be employed.

Each packet is preceded by a busy bit, or binary one, to indicate the arrival of another packet. A binary zero is placed inbetween the last packet and length specifier to indicate the end of the packet sequence. The receiver can then identify the following variable length codeword, and subsequently return to the idle state (Fig. 4.1).

#### 4.3 Block length statistics

Define the sequence of data which includes both the message and protocol data as a single block, of length  $B$  bits. Some statistics of  $B$  must be derived in order to determine queueing delays (see equation 3.4.6).

The number of packets,  $N$ , has a geometric probability distribution (see Appendix 4a). The protocol information in each block is coded into  $S$  bits which include  $N$  busy bits, a length specifier with a zero preceding it, and  $R$  bits of redundant data in the final packet

$$S(L) = N + \log_2 L + 1 + R \quad (4.3.1)$$

where  $R = NL - M \quad (4.3.2)$

From (3.5.1)

$$B(L) = (L+1)N + \log_2 L + 1 \quad (4.3.3)$$

The first moment of  $B$  is obtained from (4.3.3), with  $L$  as a fixed parameter

Message (m bits)



Transmitted block (message and protocol)

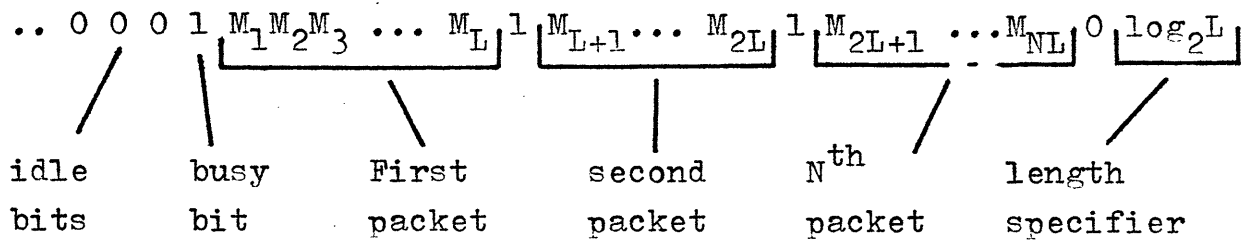


Figure 4.1

Fixed packet strategy

$$E(B) = (L+1)E(N) + \log_2 L + 1 \quad (4.3.4)$$

In Appendix 4b, the parameter L is chosen to minimize the expected block length, E(B), according to (3.5.2). The optimum packet length,  $L^{\circ}$ , and number of packets, E(N), were found to be;

$$L^{\circ} = (2E(M))^{\frac{1}{2}} - 1.782 + O(E(M)^{-\frac{1}{2}}) \quad (4.3.5)$$

$$\begin{aligned} E(N) &= \frac{1}{1-a^L} \Big|_{L=L^{\circ}} \quad (\text{Appendix 4a}) \\ &= \left(\frac{1}{2}E(M)\right)^{\frac{1}{2}} + 1.39 + O(E(M)^{-\frac{1}{2}}) \end{aligned} \quad (4.3.6)$$

In data networks, typical message lengths are confined to the range  $10 < E(M) < 10^5$ . It is thus reasonable to neglect terms of order  $E(M)^{-\frac{1}{2}}$  and below. Table (1) confirms the following approximations to be acceptable for  $L^{\circ}$  and E(N) (as given in (4.3.5), (4.3.6)), in the range  $10 < E(M) < 10^5$ ,

$$L^{\circ} \approx (2E(M))^{\frac{1}{2}} - 1.782 \quad (4.3.7)$$

$$E(N) \approx \left(\frac{1}{2}E(M)\right)^{\frac{1}{2}} + 1.39 \quad (4.3.8)$$

Values for the first and second moments of B can now be obtained as functions of E(M) alone. Consider the square of block length, B, as given in (4.3.3). The expected value of this expression is the second moment;

$$\begin{aligned} E(B^2) &= (L+1)^2 E(N^2) + 2(L+1)E(N)(\log_2 L + 1) \\ &\quad + (\log_2 L + 1)^2 \end{aligned} \quad (4.3.9)$$

From Appendix 4a,  $E(N^2) = 2E(N)^2 - E(N)$

Substituting values of  $L^0$  and  $E(N)$  found above, into the moments,  $E(B)$  and  $E(B^2)$  given in (4.3.4) and (4.3.9);

$$E(B) \approx E(M) + 1.43E(M)^{\frac{1}{2}} + \log_2(2E(M))^{\frac{1}{2}} \quad (4.3.10)$$

$$\begin{aligned} E(B^2) \approx & 2E(M)^2 + E(M) \left( 4.3E(M)^{\frac{1}{2}} + 0.47 \right. \\ & + 2 \log_2 \left( (2E(M))^{\frac{1}{2}} - 1.8 \right) \\ & + E(M)^{\frac{1}{2}} \left( 2.86 \log_2 \left( (2E(M))^{\frac{1}{2}} - 1.78 \right) - 0.74 \right) \\ & - 2.2 \log_2 \left( (2E(M))^{\frac{1}{2}} - 1.78 \right) - 2.2 \\ & \left. + (1 + \log_2 \left( (2E(M))^{\frac{1}{2}} - 1.78 \right))^2 \right) \quad (4.3.11) \end{aligned}$$

Although it has been necessary to approximate some of the coefficients in the above expressions, the functional relationships have been preserved sufficiently well to illustrate later that this strategy has considerably larger moments,  $E(B)$  and  $E(B^2)$ , than the other strategies.

#### 4.4 Concluding remarks for the fixed length strategy

The major inefficiency in this strategy is the redundancy in the last packet. By removing the necessity for fixed packet length on the final packet, and rearranging the length specifier, this redundancy could be avoided. Such an observation suggested a strategy which corresponds to the Huffman encoding of message length, as will be seen in the next paragraphs.

#### 4.5 Huffman length encoding strategy

By employing a direct encoding of message length,  $M$ , to supply the stop information for each message block,  $B$ , an efficient protocol strategy can be achieved. The integer,  $M$ , is assumed to have a geometric probability distribution and can be encoded by a Huffman coding procedure<sup>7</sup>, to give an average word length,  $\bar{n}_s$ , which exceeds the source entropy,  $H(S)$ , by an average of 0.03 bits.

The entropy of message content and length,  $H(S)$ , may be expressed as a function of the mean length of a busy period,  $1/\epsilon$ .

$$H(S) = 1/\epsilon + 1/\epsilon \mathcal{H}(\epsilon) \quad \text{bits/message} \quad (4.5.1)$$

where  $\mathcal{H}(x) = -x \log(x) - (1-x) \log(1-x)$

The first term contains the entropy of message content, and the second the entropy of message length or stop information. The Huffman encoding of length achieves an average redundancy of 0.03 bits above the source entropy. Average codeword length of stop information encoded by the Huffman scheme is  $\bar{n}_s$ , where

$$\bar{n}_s = 1/\epsilon \mathcal{H}(\epsilon) + 0.03 \quad \text{bits/message} \quad (4.5.2)$$

The binary encoding of message data achieves on average  $E(M)$  bits/message, by definition, and thus

$$E(M) = 1/\epsilon = \frac{1}{1-a} \quad \text{bits/message} \quad (4.5.3)$$

The start information is coded in the same manner as the fixed packet strategy, with binary zeros transmitted during the idle period, and a binary one transmitted to indicate the beginning of a busy period.

As no information about the length of the idle period is available in advance, like the length of messages (which arrive instantaneously), no economy in coding can be achieved.

Although the separate encoding of length and message information closely approaches the sum of the first two terms in (4.5.1), an improvement can be made by using a joint encoding scheme. The construction of a Huffman code for the joint alphabet is particularly difficult, and has not been performed because of the insignificant saving of a fraction of a bit, i.e., under the present scheme of separate encoding, only 0.03 bits are wasted on average. The separate and joint source trees are illustrated in (Fig. 4.2).

The codeword specifying message length is constructed as follows.

Let  $L$  be the integer which satisfies the inequality:

$$a^L + a^{L+1} < 1 < a^L + a^{L-1} \quad (4.5.4)$$

The mean message length,  $E(M)$ , falls in the range  $10 < E(M) < 10^5$  in most networks. An approximate value for  $a^L$  may be found under this assumption; where  $a$  is defined in (4.5.3),

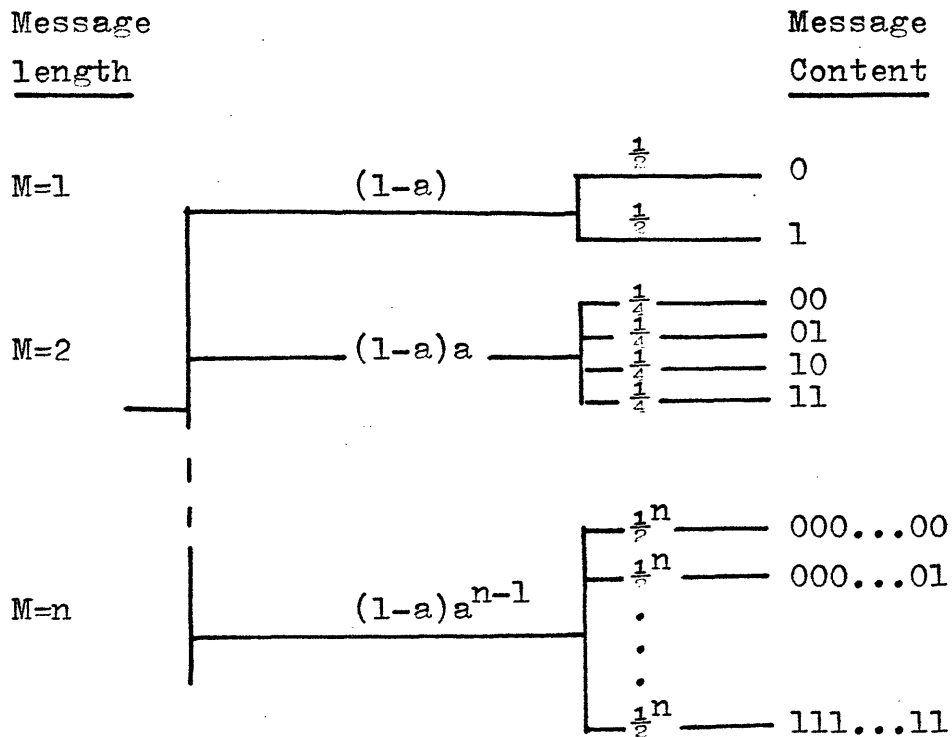
$$a^L = \frac{1}{2} + O(E(M)^{-1}) \approx \frac{1}{2} \quad (4.5.5)$$

The integer,  $M$ , may be represented by the expression:

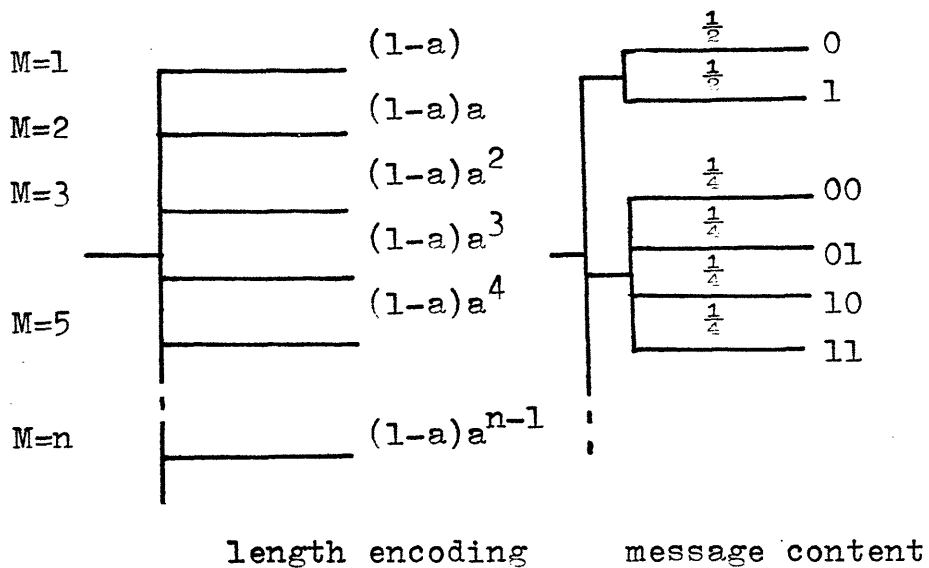
$$M = (N-1)L + [M] \bmod(L) \quad (4.5.6)$$

The integer,  $N$ , is defined by equation (4.2.1). The length encoding becomes the concatenation of a unary code of  $N-1$  binary ones followed by a zero, and a variable length codeword, of length  $\log_2 L$ ,

Figure 4.2



Joint encoding tree for message length and content



Separate coding trees

which encodes  $[M] \bmod(L)$ .

#### 4.6 Implementing the length encoding strategy

The similarity between the fixed length strategy and the one described above becomes apparent when the Huffman code is implemented in the following way.

A message of length  $M$  bits is decomposed into  $N-1$  packets of length  $L$ , and a final packet of length less than  $L$ . A busy bit is transmitted in front of each of the first  $N-1$  packets (a binary one, corresponding to the unary code in the Huffman scheme). A binary zero is placed after these packets to indicate the arrival of the length specifier, which encodes the number of bits in the final packet. The remaining message bits,  $[M] \bmod(L)$ , follow the length specifier (see Fig. 4.3). The set of  $N-1$  busy bits and the length specifier are equivalent to the two words in the Huffman scheme, although they are placed apart.

A start bit is placed in front of the first busy bit in order to avoid confusion when only one packet is transmitted in a message (i.e.,  $M < L$ ), and no busy bit is included before the length specifier.

The protocol information,  $S(L)$ , includes a start bit,  $N-1$  busy bits ('1's), a '0' placed before the length specifier, and the specifier, length  $\log_2(L)$

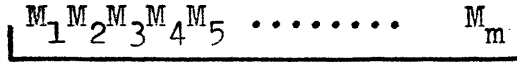
$$S(L) = 1 + (N-1) + 1 + \log_2(L) \quad (4.6.1)$$

The total length of the message and protocol information,  $B$ , becomes

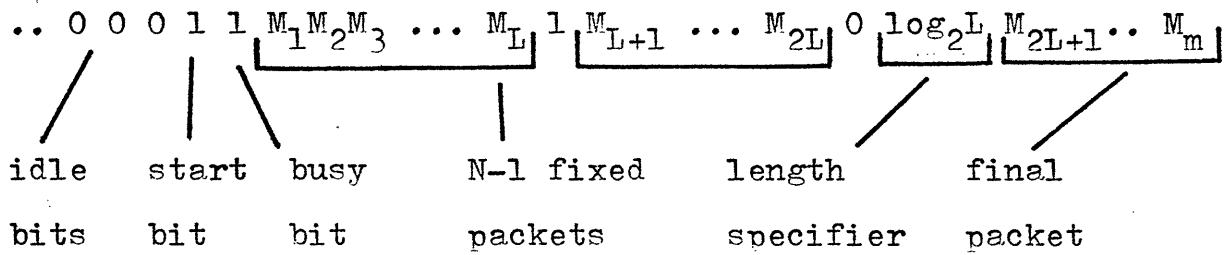
$$B = M + S = M + N + 1 + \log_2(L) \quad (4.6.2)$$



Message (m bits)



Transmitted block (message and protocol)



(N=3)

Figure 4.3

Huffman length encoding

The expected value of integer,  $N$ , is obtained from the approximation, (4.5.5) and equation (A4) from Appendix (4a).

$$E(N) = 1/(1-a^L) = 2.0 + O(E(M)^{-1}) \approx 2.0 \quad (4.6.3)$$

The expected block length,  $E(B)$  becomes

$$E(B) = E(M) + E(N) + 1 + \log_2(L) \quad (4.6.4)$$

The optimum value for  $L$  is obtained from the constraint imposed upon  $a^L$ , (4.5.4). From (4.5.3) and (4.5.5),

$$L^0 = \ln(2)E(M) + O(E(M)^{-1}) \quad (4.6.5)$$

The expected block length,  $E(B)$ , given in (4.6.4) can be expressed in terms of  $E(M)$

$$\begin{aligned} E(B) &= E(M) + 3.0 + \log_2(E(M)\ln 2) + O(E(M)^{-1}) \\ &\approx E(M) + 3.0 + \log_2(E(M)\ln 2) \end{aligned} \quad (4.6.6)$$

#### 4.7 Optimality of the Huffman scheme

The Huffman encoding of length information provides a set of code words whose average length is close to the source entropy (0.03 bits larger than  $H(S)$ ). From information theory, the Huffman coding is more efficient in this sense than other coding schemes which can be devised. If the objective of a protocol strategy is to minimize the average overhead in a network, the Huffman scheme will satisfy this condition.

A more practical measure of protocol coding efficiency in a network is the transmission delay incurred by messages operating under a specific protocol. Transmission delay in data communication networks is

related to queueing time at nodes, which have been shown to depend both on first and second moments of block length, B (Eq. 3.4.6).

#### 4.8 Second moment of block length, $E(B^2)$

Squaring the value for B, given in (4.6.2), and taking the mean,

$$E(B^2) = E(M+N)^2 + 2E(M+N) (\log_2 L + 1) + (\log_2 L + 1)^2 \quad (4.8.1)$$

In order to evaluate this expression, the joint moments of (M+N) must be derived. This has been done in Appendix (4c) in terms of a new random variable, C, where

$$C = M + N \quad (4.8.2)$$

The moment generating function of C allows the moments  $E(C)$  and  $E(C^2)$  to be derived as follows (see Appendix 4c):

$$E(C) = E(M+N) = E(M) + E(N) \quad (C5)$$

$$E(C^2) = E(M+N)^2 = 2E^2(M) + 2E^2(N) + 2E(M)E(N) - E(M) - E(N) + 2L \cdot \text{var}(N) \quad (C13)$$

Choosing the parameter L, as in (4.6.5), and the corresponding value of  $E(N)$ , as in (4.6.3), the above moments may be expressed in terms of  $E(M)$  alone.

$$\text{For } L^0 = E(M) \ln 2 \text{ and } E(N) = 2.0$$

$$\text{then } E(M+N) = E(M) + 2.0 \quad (4.8.3)$$

$$\text{and } E(M+N)^2 \approx 2E^2(M) + 5.7724E(M) + 6.0 \quad (4.8.4)$$

The second moment,  $E(B^2)$ , can now be expressed as a function of

$E(M)$ ,

$$\begin{aligned} E(B^2) \approx & 2E^2(M) + E(M)(7.7724 + 2 \log_2(E(M)\ln 2)) \\ & + 4 \log_2(E(M)\ln 2) + (\log_2(E(M)\ln 2) + 1)^2 + 10.0 \end{aligned} \quad (4.8.5)$$

#### 4.9 Terminating flag strategy

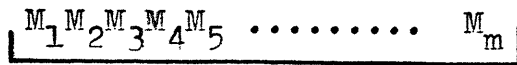
A unique bit pattern of  $r+1$  digits (a flag) is used to indicate the end of a message. When the receiver identifies the flag, it assumes that transmission of the current data is complete, and awaits either a new message or idle bits. To prevent premature terminations, the source must recognize and modify any  $r$  bit pattern which is identical to the first  $r$  bits of the flag. The encoding consists of an insertion of a single bit after the pattern, which is complementary to the  $r+1$  flag bit (see Fig. 4.4).

The receiver is constantly looking for the flag pattern. On receiving the first  $r$  of these bits, it inspects the following bit. If it is identical to the final flag bit, the receiver terminates the message. If the two are different, the receiver deletes the bit from the message, and continues to accept the incoming message.

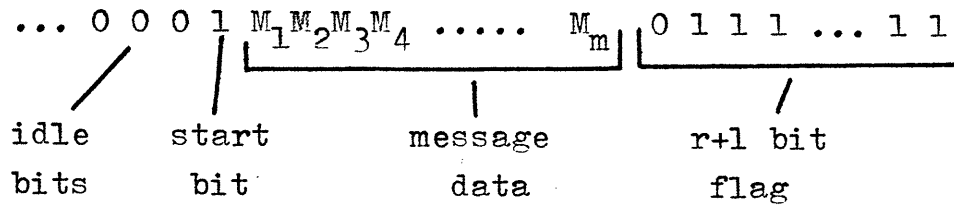
The first  $r$  bits of the flag are referred to as a recurrence pattern. The insertions caused by the occurrence of this pattern in the message constitute a component of the length protocol information.

To illustrate this strategy, consider the flag of a zero followed by  $r$  ones. If the source identifies a zero followed by  $r-1$  ones in the message, it inserts a zero after the pattern. When the receiver iden-

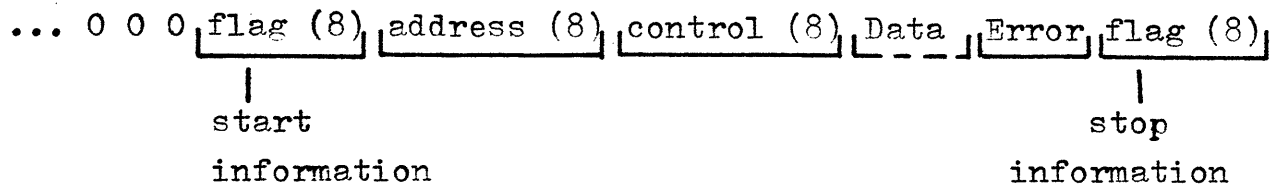
Message (m bits)



Transmitted block (message with protocol)



IBM Message Format (synchronous data link control)



The Flag strategy

Figure 4.4

tifies the same  $r$  bit pattern, it inspects the next bit. If this is a zero, it assumes an insertion which can be deleted. If it is a one, it assumes the end of a message.

Flag pattern: 01111 ..... 11

Message data:  $M_i M_j$  01111 ..... 10  $M_k M_L$

with insertion.

#### 4.10 Choosing an optimal flag pattern

A flag pattern must be found which minimizes the mean number of protocol bits,  $E(S)$ , whilst conforming to the strategy described earlier. The protocol data is related to the number of insertions,  $I$ , whose mean depends on the likelihood of the  $r$  bit recurrence pattern. The flag may be constructed from two classes of recurrence pattern, each with a different probability of occurrence.

(1) Identification of the recurrence pattern does not depend on preceding message bits. An example of such a pattern is the earlier flag pattern (4.9) of 0111...11. The probability of occurrence,  $p_1$ , of the first  $r$  digits of this pattern within the message is

$$p_1 = \left(\frac{1}{2}\right)^r \quad (4.10.1)$$

$p_1$  is also the probability of an insertion.

(2) Identification of the recurrence pattern does depend on preceding message data. Consider the success run of  $r$  ones occurring at the  $M_{j+r}$  bit in a message:

$$M_i M_j \text{ 111 } \dots \text{ 111 } M_{j+r+1} \dots$$

The event is conditioned on the value of  $M_j$ . It will occur only if  $M_j$  is a zero, or itself the last bit of an earlier  $r$  bit success run. The probability of a success run at the  $M_{j+r}$  bit is  $p_2$ , where, in the limit as  $j \rightarrow \infty$ ,

$$p_2 = \frac{\left(\frac{1}{2}\right)^{r+1}}{1 - \left(\frac{1}{2}\right)^r} \quad (4.10.2)$$

Under the flag strategy outlined in section 4.9, both the flag pattern of  $r+1$  bits and the recurrence pattern must be uniquely distinguishable to the receiver. Any such patterns which depend on preceding message data are unsuitable candidates for this strategy. For example consider a recurrence pattern of  $r$  binary ones, and a flag of  $r$  binary ones followed by a zero,  $1111\dots10$ . If the final bit of message data is a binary one, the receiver will falsely recognize an insertion in the second to last bit of the flag, i.e., it will count  $r$  binary ones followed by another one, indicating an insertion.

Although pattern (2) above is less likely than pattern (1) and would thus have a lower average number of insertions associated with the recurrence pattern, it does not give unique decoding, as illustrated in the previous example. Only patterns of the first category are suitable for the flag strategy, with a probability of insertion  $p_1$ .

The flag strategy is currently adopted by IBM in their Synchronous Data Link Control<sup>9</sup>. The SDLC line protocol places both messages and control data in similar blocks, or frames, whose format is shown in Fig. 4.4. The flag (of 8 bits) consists of a  $01111110$  pattern, and an

insertion is made into the message stream if the source finds five consecutive binary ones in the outgoing data. The insertion is a binary zero. If the receiver finds five consecutive ones followed by a zero, it deletes the last bit. If however it finds six consecutive ones, then it recognizes the flag pattern.

For example, a message contains data  $M_i M_j 011111 M_k M_L$ . An insertion is made of a binary zero in bit position  $M_k$ . The receiver counts five binary ones, i.e.,  $M_i M_j 011111 0 M_k$ , and deletes the next zero. If  $M_L$  is the last bit of a message, the source sends a flag after  $M_L$ , i.e.,  $M_L 011111 1 0$ . The receiver then counts six consecutive ones, and terminates the message. The last binary zero in the flag is quite unnecessary because the 011111 pattern is uniquely identifiable alone. Mention will be made in the next section about how large an optimal flag pattern must be.

#### 4.11 First moment of block length, $E(B)$

The protocol data,  $S$ , for the flag strategy consists of a flag of  $r+1$  bits, a start bit, and  $I$  insertions, where

$$S = r+1 + 1 + I \quad (4.11.1)$$

and the expected block length,  $E(B) = E(M) + E(S)$ , is

$$E(B) = E(M) + E(I) + r+2 \quad (4.11.2)$$

An insertion is made in the message data when an  $r$  bit pattern occurs, which is identical to the first  $r$  bits of the flag pattern. The probability of such an event,  $P(I)$ , is from (4.10.1),



$$P(I) = \left(\frac{1}{2}\right)^r \quad (4.11.3)$$

A message of length  $E(M)$  bits can only have insertions in  $E(M)-r$  positions, each with probability  $P(I)$ . The mean number of insertions,  $E(I)$ , is the sum of the expectations of an insertion at each possible location, and is given by

$$E(I) = \frac{E(M)-r}{2^r} \quad (4.11.4)$$

To obtain a minimum block length whilst employing the flag strategy, the value of  $r+1$  may be chosen as

$$r+1 = \left\lceil \log_2(E(M) \ln 2) \right\rceil \quad (4.11.5)$$

This result is derived by differentiating  $E(B)$  with respect to  $r$ , and equating to zero. The resemblance of the flag strategy to the Huffman encoding becomes apparent when the flag is compared to the length specifier.

A further constraint is made upon  $E(M) \ln 2$  in order to simplify comparison between strategies. For  $E(M) \ln 2$  an integer power of 2, the expected insertions,  $E(I)$ , becomes

$$E(I) = \frac{2}{\ln 2} - \frac{\log_2(E(M) \ln 2)}{E(M) \ln 2} + \frac{1}{E(M) \ln 2} \quad (4.11.6)$$

Neglecting ends effects which are negligible for  $E(M) \gg r$

$$E(I) \approx 2.886 ; E(M) > r \quad (4.11.7)$$

$$E(B) \approx E(M) + \log_2(E(M) \ln 2) + 3.886 \quad (4.11.8)$$

The terms omitted in (4.11.8) are of order  $\log(E(M) \ln 2)/E(M)$  and

$1/E(M)$ . For large values of  $E(M)$ , these do not contribute significantly to the mean value,  $E(B)$ .

#### 4.12 Second moment of block length, $E(B^2)$

The second moment of  $B$  is complicated by the presence of  $M+I$  terms whose statistics are not independent. In Appendix 4d the moment generating function of this sum is derived, and  $E(M+I)$ ,  $E(M+I)^2$  is calculated.

$$E(M+I) = E(M) + E(I) \quad (4.12.1)$$

$$E(M+I)^2 = 2E^2(M) - E(M) + 2E^2(I) - E(I) \\ + 2(r-1)E(I) + 4E(M)E(I) \quad (4.12.2)$$

The moments of  $I$  are also obtained in Appendix 4d, where

$$E(I) \approx 2.886 ; E(M) \gg r \quad (4.12.3)$$

$$E(I^2) = E(I) + \left(\frac{1}{2}\right)^{r-1} (1 - 1/E(M)^r) \\ = 2.886 \left(1 + \frac{4}{\ln 2} (1 - 0(\log E(M)/E(M)))\right)$$

$$E(I^2) = 19.542 + 0(\log E(M)/E(M)) \quad (4.12.4)$$

The second moment of block length can now be evaluated as a function of  $E(M)$ ; from (4.11.2)

$$E(B^2) = E(M+I)^2 + 2E(M+I)(r+2) + (r+2)^2 \quad (4.12.5)$$

Inserting the moments of  $(M+I)$  given in (4.12.1) and (4.12.2),

$$E(B^2) = 2E^2(M) + E(M)(3 + 2r + 4E(I)) \\ + 2E^2(I) + E(I)(1 + 4r) + (r+2)^2 \quad (4.12.6)$$

where  $E(I)$  is given in (4.12.3).

Appendix 4a

Probability distribution of N

The number of packets required to transmit a message of length M bits, is given by the random variable, N;

$$N = \left\lceil \frac{M}{L} \right\rceil$$

The probability distribution of variable M is geometric,

$$P_M(m) = (1-a)a^{m-1} ; m \geq 1 \quad (A1)$$

With a mean value, E(M);

$$E(M) = (1-a)^{-1} \quad (A2)$$

The definition of N in (A1) may be rewritten as

$$(N-1)L < M \leq NL$$

which gives a probability mass function

$$\begin{aligned} P_N(n) &= \Pr((n-1)L < M \leq nL) \\ P_N(n) &= (1-a^L)a^{(n-1)L} ; n \geq 1 \end{aligned} \quad (A3)$$

The moments of N are simply calculated from (A3);

$$E(N) = (1-a^L)^{-1} \quad (A4)$$

$$E(N^2) = 2E^2(N) - E(N) \quad (A5)$$

and 
$$\text{Var}(N) = E^2(N) - E(N) \quad (A6)$$

Appendix 4b

Optimal packet length,  $L^0$ , for the  
fixed packet strategy

An expression was obtained in section (4.3) for the expected block length of a message together with its protocol information. The block length was found to be functionally related to expected message length,  $E(M)$ , through variable  $a$ , and also to packet length,  $L$ . It is possible to minimize block length with respect to  $L$ , as described in section (3.5). An exact relationship between  $L$  and  $E(M)$  is difficult to obtain; however a useful approximation can be made and tested.

The block length of a single message is given by equation (4.3.4). The expected number of packets,  $E(N)$ , employed in one message is derived in Appendix (4a). Taking the first derivative of  $E(B)$ , and equating to zero to find the minimum value:

$$\frac{dE(B)}{dL} = 0 = \frac{(1+L)a^L \ln(a)}{(1-a^L)^2} + \frac{1}{(1-a^L)} + \frac{1}{L \ln(2)} \quad (B1)$$

Defining an additional variable,  $x$ , to obtain a parametric equation pair between  $L$  and  $a$ ; from Appendix (4a)

$$a = 1 - 1/E(M) \quad (B2)$$

$$x = -L \cdot \ln(a) \quad \text{or} \quad a^L = e^{-x} \quad (B3)$$

At the minimum of  $E(B)$ , as given in (B1),

$$0 = \frac{(\ln(a)-x)e^{-x}}{(1-e^{-x})^2} + \frac{1}{1-e^{-x}} + \frac{\ln(a)}{x\ln(2)}$$

Multiplying the last expression by  $e^x(1-e^{-x})^2$

$$0 = \ln(a) - x + e^x - 1 - \frac{e^x \ln(a) (1-e^{-x})^2}{x\ln(2)}$$

which simplifies into an expression for  $-\ln(a)$ ;

$$-\ln(a) = \frac{e^x - 1 - x}{1 - (e^x + e^{-x} - 2)/x\ln(2)} \quad (B4)$$

By expanding terms in  $e^x$  and  $e^{-x}$ , it is possible to obtain an approximate result for  $-\ln(a)$  as a series of decreasing terms in  $x^n$ ;  $x$  being less than unity.

$$-\ln(a) = \frac{x^2}{2} + x^3 \left[ \frac{1}{6} + \frac{1}{2\ln(2)} \right] + x^4 \left[ \frac{1}{24} + \frac{1}{6\ln(2)} \right] + O(x^5)$$

Ignoring terms in  $x^4$ ,  $x^5$  and higher powers, the following approximation can be made for  $\ln(a)$ :

$$-\ln(a) \approx \frac{x^2}{2} + x^3 \left[ \frac{1}{6} + \frac{1}{2\ln(2)} \right] \quad (B5)$$

This approximate expression gives a value for  $x$  which may be substituted back into (B3) to obtain  $L$ .

$$x = \left[ \frac{-2\ln(a)}{1+x(1/3+1/\ln(2))} \right]^{\frac{1}{2}} \\ \approx \sqrt{-2\ln(a)} \left[ 1 - \sqrt{-2\ln(a)} (1/6+1/2\ln(2)) \right] \quad (B6)$$

$$L^{\circ} = \frac{x}{-\ln(a)}$$
$$\approx (2E(M))^{\frac{1}{2}} - 1.782 + O(E(M)^{-\frac{1}{2}}) \quad (B7)$$

Table (1) evaluates (B4) directly to gain an accurate numerical correspondence between parameter,  $x$ , and  $L^{\circ}$ . The approximate expression for  $L^{\circ}$  is also evaluated for the same values of  $x$ , and listed beside the results achieved without approximation. In the range  $10 < E(M) < 10^5$ , the correspondence is close, especially as  $E(M)$  grows larger. Beyond this range, the higher powers of  $x$  are negligible, and improve the correspondence further.

In practice the values of  $L_1$  and  $L_2$  listed in table (1) are integer valued and so there is no real difference between the approximation,  $L_2$ , and the exact value,  $L_1$ .

Table 4-1

Fixed Packet Length, L

<u>x</u>	<u>E(M)</u>	<u>L<sup>o</sup></u>	<u>E(N<sup>o</sup>)</u>	<u>L'</u>	<u>E(N')</u>	<u>Δ</u>
0.3	11.8	3.4	3.846	4	3.353	0.077
0.2	33.7	6.64	5.516	7	5.260	0.012
0.1	166.0	16.55	10.508	17	10.244	0.01
0.09	208.9	18.76	11.616	19	11.476	0.006
0.08	269.6	21.53	13.006	22	12.739	0.008
0.07	358.9	25.09	14.790	26	14.291	0.04
0.06	497.8	29.84	17.171	30	17.082	0.001
0.05	730.5	36.40	20.559	37	20.234	0.008
0.04	1.2x10 <sup>3</sup>	46.49	26.304	47	26.025	0.05
0.03	2.1x10 <sup>3</sup>	63.14	33.754	64	33.307	0.02
0.02	4.8x10 <sup>3</sup>	96.47	50.253	97	49.981	0.05
0.01	1.96x10 <sup>4</sup>	196.5	100.244	197	99.991	0.001
0.005	7.93x10 <sup>4</sup>	396.5	200.500	397	200.250	0.014

Notes:

E(M) Message length

L<sup>o</sup> Packet length calculated from (B3) and (B4); the optimum value.

E(N<sup>o</sup>) The number of mean packets associated with packet length, L<sup>o</sup>

L' Packet length calculated from approximation (4.3.7), and integer rounded for practical purposes.

E(N') The mean number of packets associated with packet length, L'

Δ Numerical difference between block lengths derived from (4.3.4), using L<sup>o</sup>, E(N<sup>o</sup>) for E(B<sup>o</sup>) and L', E(N') for E(B'),

$$\Delta = E(B') - E(B^o)$$

Appendix 4c

Second moment of the sum of two dependent  
random variables, M and N

A message of length M bits is transmitted in N packets where both M and N have a geometric probability distribution (see Appendix 4a). The sum of the two variables forms a new random variable, C, whose probability mass function is  $P_C(c)$  and characteristic function,  $C(s)$ .

$$C = M + N \quad (C1)$$

$$C(s) = \sum_{c=2}^{\infty} P_C(c) s^c \quad (C2)$$

The probability mass function,  $P_C(c)$ , may be obtained by considering (A1) and (C1), and gives an expression for  $C(s)$  as follows:

$$C(s) = \sum_{i=1}^{\infty} \sum_{m=(i-1)L+1}^{iL} P_M^{(m)} s^{m+i} \quad (C3)$$

Performing the double summation in (C3), the characteristic function may be evaluated into the expression:

$$C(s) = \frac{(1-a)(1-(as)^L)s^2}{(1-as)(1-a^L s^{L+1})} \quad (C4)$$

The first moment of C is obtained by taking the first derivative of  $C(s)$ , and putting  $(s=1)$

$$E(C) = \left. \frac{dC(s)}{ds} \right|_{s=1} = \frac{1}{(1-a)} + \frac{1}{(1-a^L)}$$



$$E(C) = E(M) + E(N) \quad (C5)$$

The second moment of C is obtained by taking the second derivative of the characteristic function, C(s)

$$\left. \frac{d^2 C(s)}{ds^2} \right|_{s=1} = E(C^2) - E(C) \quad (C6)$$

Consider the factorization of C(s) into two components

$$C(s) = Q(s)s^2 \quad (C7)$$

where the derivatives of Q(s) may be easily calculated

$$Q(s) = \frac{(1-a)(1-(as)^L)}{(1-as)(1-a^L s^{L+1})} \quad (C8)$$

$$\left. \frac{dQ(s)}{ds} \right|_{s=1} = \frac{a}{(1-a)} + \frac{a^L}{(1-a^L)} \quad (C9)$$

$$\begin{aligned} \left. \frac{d^2 Q(s)}{ds^2} \right|_{s=1} &= \frac{2a^2}{(1-a)^2} + \frac{2a^{2L}}{(1-a^L)^2} \\ &+ \frac{2La^L}{(1-a^L)^2} + \frac{(2a)}{(1-a)} \frac{(a^L)}{(1-a^L)} \end{aligned} \quad (C10)$$

The second derivative in (C6) becomes

$$\begin{aligned} \left. \frac{d^2 C(s)}{ds^2} \right|_{s=1} &= \left. \frac{d^2 Q(s)}{ds^2} \right|_{s=1} + \left. \frac{4dQ(s)}{ds} \right|_{s=1} + 2Q(1) \\ &= \frac{2a^2}{(1-a)^2} + \frac{2a^{2L}}{(1-a^L)^2} + \frac{(2a)(a^L)}{(1-a)(1-a^L)} + \end{aligned}$$

$$+ \frac{2L(a^L)}{(1-a^L)^2} + \frac{4a}{(1-a)} + \frac{4a^L}{(1-a^L)} + 2 \quad (C11)$$

The second moment,  $E(C^2)$ , may be obtained from (C11) in terms of  $E(M)$  and  $E(N)$ , where the following identities are required:

$$\begin{aligned} E(M) &= (1-a)^{-1} \\ E(N) &= (1-a^L)^{-1} \\ E(M^2) &= E^2(M) - E(M) \\ E(N^2) &= E^2(N) - E(N) \quad (\text{as in App. 3a}) \end{aligned}$$

The second derivative at  $(s=1)$  becomes

$$\begin{aligned} \left. \frac{d^2 C(s)}{ds^2} \right|_{s=1} &= 2E^2(M) + 2E^2(N) + 2E(M)E(N) \\ &+ 2L\text{var}(N) - 2E(M) - 2E(N) \end{aligned} \quad (C12)$$

Combining (C12) with (C6), the second moment becomes:

$$E(C^2) = 2E^2(M) + 2E^2(N) + 2E(M)E(N) - E(M) - E(N) + 2L\text{var}(N) \quad (C13)$$

No approximations have been made in deriving this result.

Appendix 4d

The second moment of the sum of two  
dependent random variables, M, I

The results associated with the theory of recurrent events<sup>10</sup> may be applied to the flag strategy described in section (4.9). The recurrent event is taken here to be the possible replication of the flag pattern within the message data. Upon each replication, an insertion of an extra bit is made into the message data. The total number of insertions,  $I_m$ , in a message of  $m$  bits will thus correspond to the number of recurrent events,  $N_m$ .

Specifically, by observing a source which produces one bit of data per instant of time, when busy, a recurrent event,  $E$ , is defined to occur at the  $(j+r)^{\text{th}}$  instant if the message sequence between the  $j^{\text{th}}$  and  $(j+r)^{\text{th}}$  instants corresponds to the first  $r$  bits of the flag pattern.

Before considering the sum of message length,  $M$ , and number of insertions,  $I$ , it is necessary to derive the probability mass function of  $I_m$ . The probability distribution of message length is assumed to be geometric, with mean value  $(1-a)^{-1}$ . The distribution function for  $I$  is related to the conditional distribution for  $I_m$  by

$$P_I(i) = \sum_{m=1}^{\infty} P_{I_m}(i|m)P_M(m) \quad (D1)$$



$$u_n = f_1 u_{n-1} + f_2 u_{n-2} + f_3 u_{n-3} + \dots + f_n u_0$$

which transforms into the s domain easily, by considering the right hand side of the expression as a convolution.

$$F(s) = \frac{U(s)-1}{U(s)} \quad (D7)$$

It is known from the flag strategy that the probability of an insertion is  $(\frac{1}{2})^r$ , which is the same as  $u_n$  for  $n \geq r$ ;

$$u_n = \begin{cases} (\frac{1}{2})^r & ; \quad n \geq r \\ 0 & ; \quad 0 < n < r \end{cases} \quad (D8)$$

The characteristic function,  $U(s)$  may be evaluated using (D4, 6, 8);

$$U(s) = \frac{1-s+(\frac{1}{2})^r s^r}{(1-s)} \quad (D9)$$

The characteristic function of  $F(s)$  can now be evaluated from (D9, 7);

$$F(s) = \frac{(\frac{1}{2})^r s^r}{1-s+(\frac{1}{2})^r s^r} \quad (D10)$$

There is a simple relationship between waiting time,  $T_1$ , and probability of a first event,  $f_n$ ;

$$\Pr(T_1 = n) = f_n \quad (D11)$$

If a recurrent event,  $E$ , occurs for the second time at the  $n^{\text{th}}$  instant, a probability,  $f_n^{(2)}$ , is assigned, where from (D2);

$$\Pr(T_1 + T_2 = T^{(2)} = n) = f_n^{(2)} \quad (D12)$$

Similarly, if the  $q^{\text{th}}$  event occurs at the  $n^{\text{th}}$  instant

$$\Pr(T^{(q)} = n) = f_n^{(q)} \quad (D13)$$

The probability assignment,  $f_n^{(2)}$ , is the convolution of  $f_n$  with itself; this suggests that its characteristic function is  $F^2(s)$ . The result extends to the  $q^{\text{th}}$  case above:

$$f_n^{(2)} = f_1 f_{n-1} + f_2 f_{n-2} + \dots + f_{n-1} f_1$$

$$\text{Then } F^{(2)}(s) = F^2(s)$$

$$q^{\text{th}} \text{ case: } f_n^{(q)} = f_n * f_n * \dots * f_n \quad (D14)$$

$$\text{and } F^{(q)}(s) = F^q(s) = \sum_{p=1}^{\infty} f_p^{(q)} s^p \quad (D15)$$

These results may now be used to obtain  $P_{I_m}(i)$ .

The probability mass function may be written as

$$P_{I_m}(i) = \Pr(T^{(i+1)} > m) - \Pr(T^{(i)} > m) \quad (D16)$$

Using the relationship between  $T^{(i)}$  and  $f_m^{(i)}$  in (D13), (D16)

becomes

$$P_{I_m}(i) = \sum_{p=1}^m f_p^{(i)} - \sum_{p=1}^m f_p^{(i+1)} \quad (D17)$$

The probability mass function,  $P_{I_m}(i)$ , as defined in (D1) can be evaluated using (D17) and a geometric distribution for  $m$ , and also (D15);

$$P_{I_m}(i) = \sum_{m=1}^{\infty} P_{I_m}(i) (1-a) a^{m-1}$$

$$P_I(i) = \begin{cases} \frac{F^i(a)(1-F(a))}{a} & ; i \geq 1 \\ 1 - \frac{F(a)}{a} & ; i = 0 \end{cases} \quad (D18)$$

Moments of I, E(I), E(I<sup>2</sup>)

Having derived the probability mass function of I, a characteristic function may be found, using (D18);

$$I(s) = \sum_{i=0}^{\infty} P_I(i) s^i$$

$$I(s) = 1 + \frac{F(a)(s-1)}{a(1-F(a)s)} \quad (D19)$$

The first moment is obtained from the first derivative of I(s), setting s = 1

$$E(I) = \left. \frac{dI(s)}{ds} \right|_{s=1} = \frac{F(a)}{a(1-F(a))}$$

$$E(I) = \frac{\left(\frac{1}{2}\right)^r a^{r-1}}{(1-a)} \quad (D20)$$

An approximation of the first moment, E(I), can be made using the substitution suggested in section (4.11) for r

$$(r + 1) = \log_2(E(M) \ln 2)$$

where  $E(M) = (1-a)^{-1}$

then  $E(I) = \frac{E(M)}{2^r} (1-1/E(M))^{r-1} \approx \frac{2 \cdot E(M) (1-r/E(M))}{E(M) \ln 2}$

$$\approx \frac{2}{\ln 2} - \frac{2 \log_2(\ln 2 E(M))}{\ln 2 E(M)} \quad (D21)$$

Terms of order  $1/E(M)$  and lower magnitudes are ignored. The second moment of  $I$  is obtained from the second derivative of  $I(s)$ ,

$$\left. \frac{d^2 I(s)}{ds} \right|_{s=1} = E(I^2) - E^2(I) \quad (D22)$$

$$E(I^2) = \frac{F(a)(1+F(a))}{a(1-F(a))^2}$$

The sum of two dependent variables,  $M+I$

Having obtained  $P_{I_m}(i)$  and knowing the distribution of  $M$ ,  $P_M(m)$ , the characteristic function of a new random variable,  $C$ , defined as the sum of  $M$  and  $I$ , can be found.

$$C = M + I$$

$$P_C(c) = \sum_{m=1}^{\infty} P_{I_m}(c-m) P_M(m) \quad ; \quad c \geq 1 \quad (D23)$$

The characteristic function of  $C$ ,  $C(s)$ , is defined as

$$C(s) = \sum_{c=1}^{\infty} P_C(c) s^c \quad (D24)$$

Substituting  $P_C(c)$ , as given in (D23), into (D24) and rearranging the summations,

$$C(s) = \sum_{m=1}^{\infty} P_M(m) s^m \sum_{n=0}^{\infty} P_{I_m}(n) s^n \quad (D25)$$

Using the expression for  $P_{I_m}(i)$  given in (D17) and using (D15),



$$F^i(as) = \sum_{n=0}^{\infty} f_n^{(i)} (as)^n \quad (D26)$$

The summation in (D25) may be simplified to

$$C(s) = \sum_{m=1}^{\infty} P_M^{(m)} s^m \sum_{n=1}^{\infty} (s^n - 1) \left[ \sum_{p=1}^m f_p^{(n)} - \sum_{p=1}^m f_p^{(n+1)} \right] + M(s)$$

$$C(s) = \frac{(1-a)F(as)(s-1)}{(1-as)a(1-F(as)s)} + M(s) \quad (D27)$$

where the characteristic function of M being M(s). The first moment of C, E(C), is found by taking the first derivative of C(s)

$$E(C) = \left. \frac{dC(s)}{ds} \right|_{s=1} = \frac{F(a)}{a(1-F(a))} + E(M)$$

Using (D20),

$$E(c) = E(I) + E(M) \quad (D28)$$

The second derivative of C(s) gives the second moment,

$$\left. \frac{d^2 C(s)}{ds^2} \right|_{s=1} = E(C^2) - E(C) \quad (D29)$$

Using (D27)

$$= \frac{2}{a} \left[ \frac{(F(a))^2}{(1-F(a))^2} + \frac{aF(a)}{(1-a)(1-F(a))} + \frac{aF'(a)}{(1-F(a))^2} \right] + M''(s) \quad (D30)$$

From (D10), the first derivative, F'(a), can be found.

Then

$$\frac{F'(a)}{(1-F(a))^2} = \frac{(r-1) \left(\frac{1}{2}\right)^r a^{r-1}}{(1-a)} + \frac{\left(\frac{1}{2}\right)^r a^r}{(1-a)^2}$$

$$= (r-1)E(I) + E(I)E(M) \tag{D31}$$

where we have used (D20).

Substituting (D31) back into (D30) and rearranging to obtain  $E(C^2)$ , and using the mean value of  $E(C)$  in (D28),

$$E(C^2) = 2E^2(M) + 2E^2(I) - E(M) - E(I) + 2(r-1)E(I) + 4E(I)E(M)$$

$$\frac{-2E^2(I)}{E(M)} \tag{D32}$$

No approximations have been made in obtaining the result (D32). However in section 4, the final term in  $E^2(I)/E(M)$  will be neglected when using this result because it is of order less than unity in the range  $10^2 < E(M) < 10^5$ .

## Section 5

### Conclusions for Stop Protocols

#### 5.1 Introduction

A criterion for evaluating protocol strategies in terms of waiting time,  $W$ , in the source node queue and in transmission was discussed in section (3), where

$$E(W) = E(B) + \frac{\lambda E(B^2)}{2(1-\lambda E(B))} \quad (5.1.1)$$

In the fourth section first and second moments of block length,  $B$ , were derived for three protocol strategies; the fixed packet, flag and Huffman encoding of length schemes. To prove that the waiting time,  $W_i$ , for the  $i^{\text{th}}$  strategy is shorter than that of the  $j^{\text{th}}$  strategy,  $W_j$ , it is sufficient to show that the two conditions are met:

$$E(B_i) < E(B_j) \quad \text{and} \quad E(B_i^2) < E(B_j^2) \quad (5.1.2)$$

so that from (5.1.1)

$$W_i < W_j \quad (5.1.3)$$

In comparing the three strategies analyzed in section 4 condition (5.1.2) may be employed to give a simple ordering of efficiencies.

#### 5.2 Comparing the first moments, $E(B)$

The first moment is especially important in the analysis of protocol information, because it relates directly to the entropy of the

information (section 4.5). In comparing expected block length between strategies, it is only necessary to compare average protocol data,  $E(S)$ , because  $E(M)$  is common to all schemes.

$$E(B) = E(M) + E(S) \quad (5.2.1)$$

The first moments of block length were found in section 4 to have the values: (4.3.10), (4.11.8), (4.6.6),

$$\text{Fixed Packet: } E(B_1) = E(M) + 1.43E^{\frac{1}{2}}(M) + \log_2(2E(M))^{\frac{1}{2}}$$

$$\text{Flag: } E(B_2) = E(M) + \log_2(E(M)\ln 2) + 3.886$$

$$\text{Huffman: } E(B_3) = E(M) + \log_2(E(M)\ln 2) + 3.0$$

For purpose of comparison, it is convenient to introduce the common parameter,  $r+1$ , from the flag strategy (4.11.5), providing the parametric equation for all schemes

$$E(S_i) = a_i E^{\frac{1}{2}}(M) + b_i r + c_i \quad (5.2.2)$$

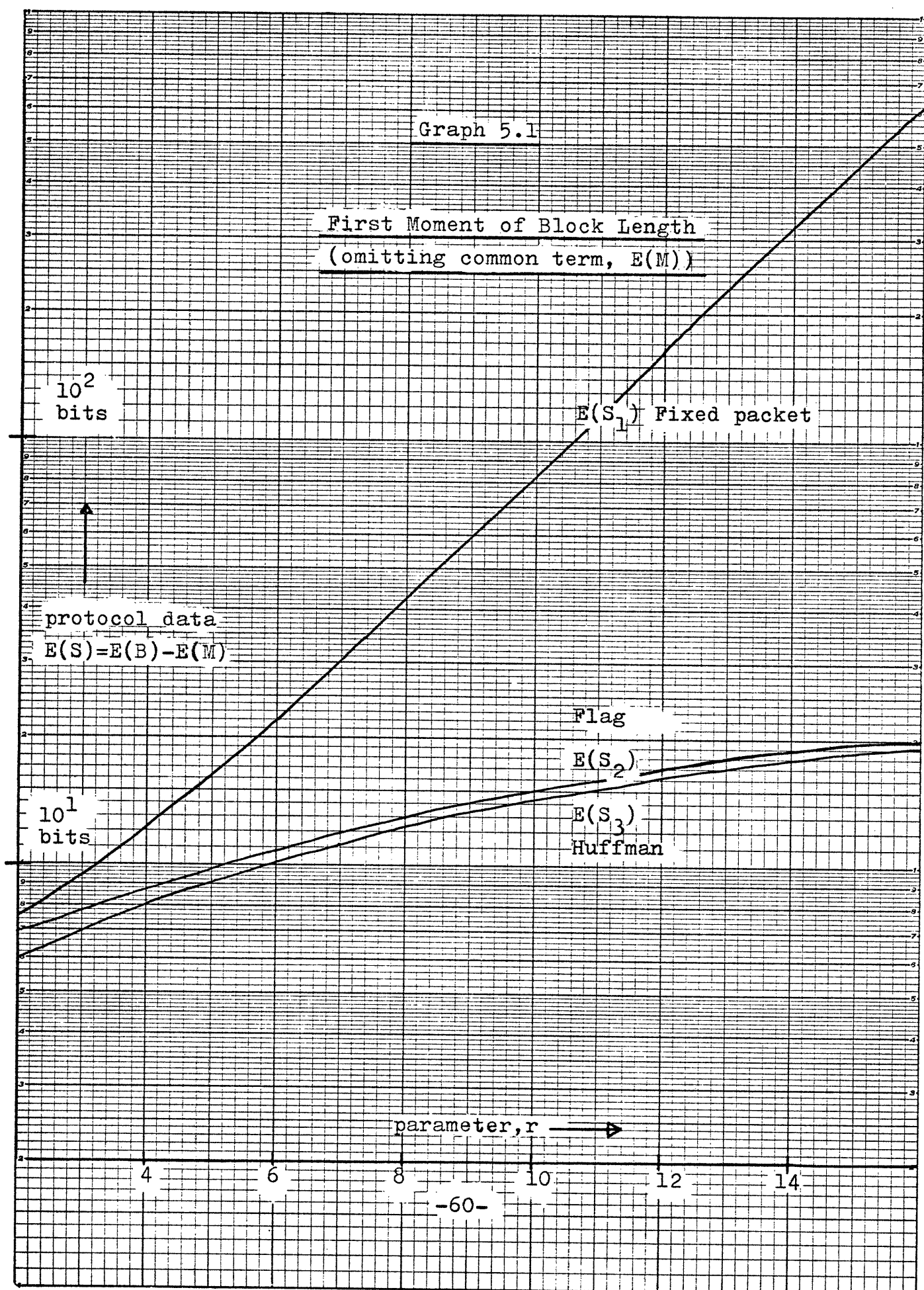
where  $r+1 = \log_2(E(M)\ln 2) \quad (5.2.3)$

and  $E(M) = \frac{2^{r+1}}{\ln 2} \quad (5.2.4)$

Table 5.1 below lists the coefficients for the three strategies, and graph 5.1 plots  $E(S_i)$  over the range  $3 \leq r \leq 13$ , or  $23 \leq E(M) \leq 2.10^4$ .

Table 5.1 Mean value of protocol data

Fixed packet:	$E(S_1) = 1.4E^{\frac{1}{2}}(M) + 0.5r + 1.26$
Flag:	$E(S_2) = 0 + r + 4.89$
Huffman:	$E(S_3) = 0 + r + 4.00$



Protocol data,  $E(S)$ , has been derived from  $E(B) - E(M)$ , in the equations of section 4. Substitution has been made for  $E(M)$  according to Equation (5.2.4).

From the graph, and table it can be seen that

$$E(S_1) > E(S_2) > E(S_3) \quad (5.2.5)$$

$$E(B_1) > E(B_2) > E(B_3) \quad (5.2.6)$$

### 5.3 Discussion of first moments

The fixed packet strategy contains a non zero  $E^{\frac{1}{2}}(M)$  coefficient, which dominates  $E(S_1)$  for large values of  $E(M)$ . This term is associated with the redundancy of the final packet, which is eliminated in the Huffman scheme by relocating the length specifier. Similarly in the flag strategy there is no equivalent redundancy.

The Huffman length encoding and flag strategies are remarkably close, and there is a simple explanation for this similarity. The Huffman scheme employs two concatenated code words: one a unary encoding of packets,  $N$ , and the second a variable length codeword expressing final packet redundancy. The second codeword has the same expected value as the flag length (5.2.3). Each insertion has probability  $(\frac{1}{2})^x$ , which is equivalent to an insertion per  $2^x$  bits, on average. This corresponds to the busy bit preceding each packet in the Huffman code (the unary code-word contains  $N+1$  bits).

The Huffman code is marginally more efficient than the flag strategy (by 0.89 bits), but both schemes vary widely from the fixed

packet strategy.

#### 5.4 Comparing second moments, $E(B^2)$

The second moment gives a measure of the dispersion of block length about its mean value, and influences the waiting time of messages in the source queue (see 5.1.1).

Employing the parameter  $r+1$ , a general equation may be written for the second moment,  $E(B^2)$ ;

$$E(B_i^2) = 2E^2(M) + a_i E^{3/2}(M) + (b_i r + c_i) E(M) + (d_i r + e_i) E^{\frac{1}{2}}(M) + f_i r^2 + g_i r + h_i \quad (5.4.1)$$

Each strategy has a second moment defined by the set of coefficients (a, b, c, d, e, f, g, h), as listed in Table 5.2 below. Graph 5.2 plots  $E(B_i^2)$  over the range of  $r$ ,  $3 \leq r \leq 13$ . The term in  $E^2(M)$  is omitted in the graphical values, being common to each strategy.

Table 5.2 Second moments of block length

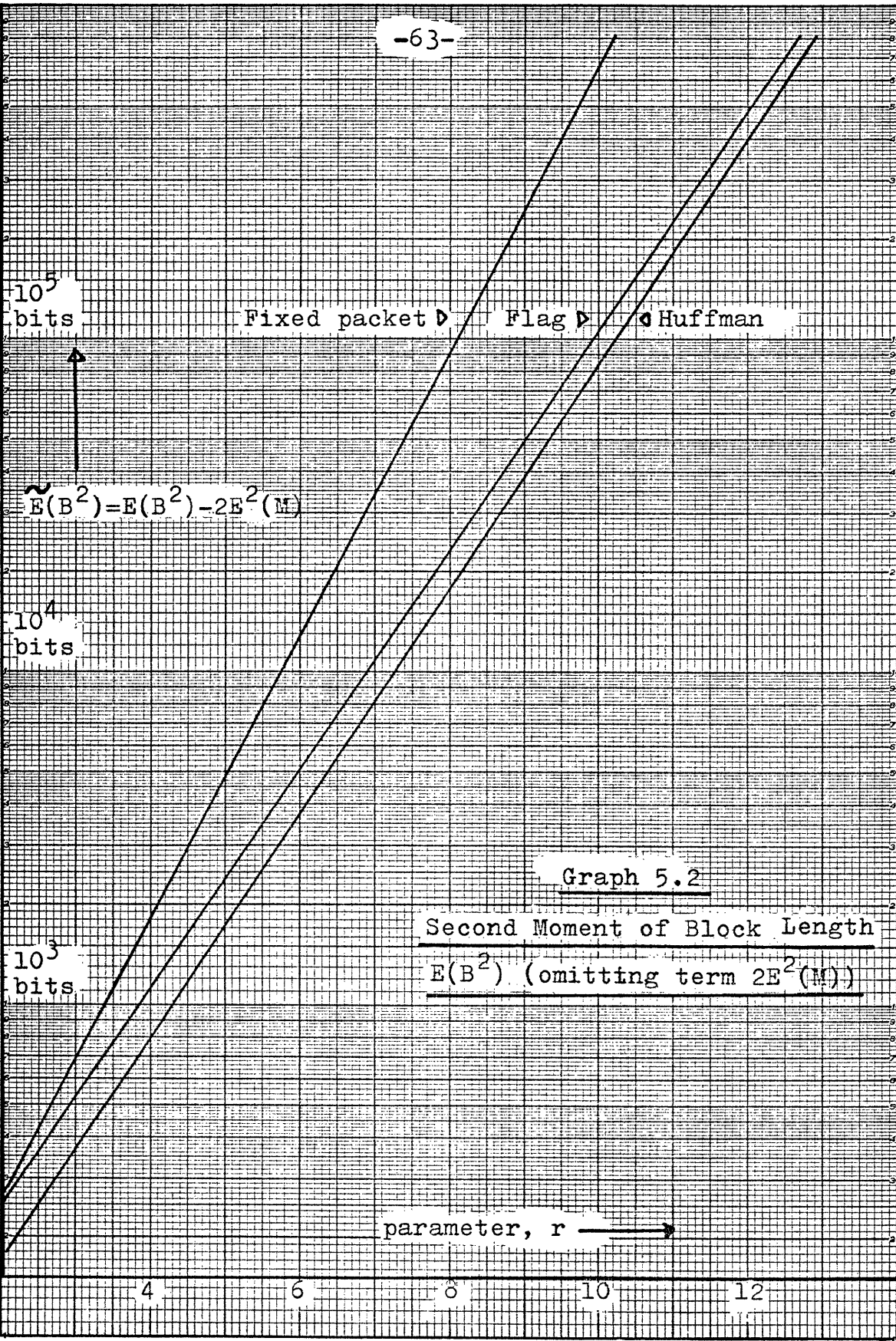
Omitting the common term in  $2E^2(M)$  from  $E(B_i^2)$ , the second moments for the fixed packet, flag and Huffman schemes are:

$$\tilde{E}(B_1^2) = 4.3E^{3/2}(M) + (r+3)E(M) + (1.43r+2.88)E^{\frac{1}{2}}(M) + 0.25r^2 + 1.16r + 0.15$$

$$\tilde{E}(B_2^2) = 0 + (2r+14.54)E(M) + 0 + r^2 + 15.54r + 23.55$$

$$\tilde{E}(B_3^2) = 0 + (2r+9.77)E(M) + 0 + r^2 + 8.0r + 18.0$$

The results above were taken from (4.3.11), (4.12.6) and (4.8.5)



Graph 5.2

Second Moment of Block Length  
 $E(B^2)$  (omitting term  $2E^2(M)$ )

$10^5$   
bits

$10^4$   
bits

$10^3$   
bits

$E(B^2) = E(B^2) - 2E^2(M)$

Fixed packet

Flag

Huffman

parameter, r



respectively, where  $\log_2 E(M)$  is replaced by  $f(r)$  defined in (5.2.3).

From the graph, and coefficients above, it can be seen that

$$E(B_1^2) > E(B_2^2) > E(B_3^2) \quad (5.4.2)$$

The magnitude of first moments have a similar ordering (5.2.6), such that condition (5.1.2) is applicable to the three strategies under discussion. The order of magnitude of waiting times becomes

$$W_1 > W_2 > W_3 \quad (5.4.3)$$

where the waiting time of the fixed packet strategy is  $W_1$ , that of the flag strategy is  $W_2$ , and the Huffman length encoding strategy is  $W_3$ .

### 5.5 Discussion of second moments

The redundancy in the final packet of the fixed packet strategy contributes a term in  $E^{3/2}(M)$  which is not present in the other strategies. This term causes  $E(B_1^2)$  to greatly exceed the other moments, for larger values of  $E(M)$ .

The flag and Huffman encoding strategies contain terms of similar order, but with different coefficient values. The difference between  $E(B_2^2)$  and  $E(B_3^2)$ , for large values of  $E(M)$ , becomes  $4.77E(M)$  which increases as an exponent of  $r$ . This is a more significant difference than 0.89 bits in the first moment, and illustrates the greater uncertainty of the number of protocol bits in the flag strategy.

## 5.6 Conclusion

In the context of a single source/receiver link, three protocol strategies for transmitting start-stop information were analyzed for messages with geometrically distributed lengths. The mean waiting time of messages in the source node queue and transmission was taken as a performance measure under which the strategies could be compared. Queueing and service time in store and forward networks is directly related to transmission delay, which is of practical importance in any network.

Three strategies were taken from existing networks, including fixed packet, flag and variable length packet strategies. The latter was based on ideas from information theory, and was found to be the most efficient in terms of queueing delays, and average codeword length for the protocol data.

The queueing problem was simplified by assuming geometric message length statistics, which although not generally equivalent to practical cases, do exhibit an extremal property. Such statistics maximize the amount of protocol information required to encode message length, and the most efficient encoding for this case satisfies the minimax condition, i.e., the most economic coding under the worst source statistics.

The queueing problem was analyzed according to an M/G/1 process, where the waiting and service time was found to depend only on first and second moments of block length (message and protocol data). The first moment,  $E(B)$ , also has significance from a source coding viewpoint. An efficient coding scheme, in an information theoretic sense, is one that

has an average codeword length close to the source entropy. The Huffman scheme has a mean redundancy above the source entropy of 0.03 bits/message.

The first moments of the three strategies are presented in table 5.1. Both the Huffman and flag strategies achieve a coding redundancy of less than one bit/message, and have some close similarities. For instance, the flag closely resembles the length specifier and the insertions (occurring every  $2^r$  bits, on average) resemble the busy bits placed before each packet of length  $L$ . The fixed packet strategy is less efficient due to the redundancy in the final packet, which is eliminated in the Huffman scheme.

The second moments are listed in table 5.2. The variance of protocol data for each scheme is directly related to second moments. The block length of message and protocol data in the Huffman case has a lower variance than under the flag strategy. This may be understood by considering the appearance of insertions in the flag strategy in contrast with the busy bits of the Huffman scheme. The former are subject to greater statistical uncertainty, and contribute to the higher overall variance of the flag strategy. The redundancy in the final packet of the fixed packet strategy makes the second moment considerably larger than the other schemes.

The main theoretical result to emerge from the study is the close relationship between stop information and message length. This was used to advantage by devising a protocol strategy using an efficient encoding

of message length. The source coding approach was shown to be efficient in a practical sense by minimizing queueing and transmission delays in data networks, as well as reducing average protocol data to a minimum level.

Some practical results from the analysis include optimum packet and flag lengths for the appropriate strategies. For example, in the fixed packet strategy, packet length which minimizes protocol data was found to be approximately  $(2E(M))^{1/2}$ , where expected message length is  $E(M)$ . In the Huffman scheme, packet size was  $E(M)\ln 2$ . In the flag strategy, optimum flag length was  $\lceil \log_2 E(M)\ln 2 \rceil$ .

Applying these results to the IBM line protocol described in section 4.10, assuming an average message length of  $10^3$  bits, flag length would be ten bits. Also flag structure would be modified by omitting the final binary zero, i.e., 0111111111.

Having completed the discussion on start-stop protocol for a single source/receiver pair, attention will be given to devising a source code for start-stop information for many sources and receivers sharing a single channel.

## Section 6

### Protocol for a single link with identical sources

#### 6.1 Introduction

Previous sections have been concerned with protocol for a single link with one source and receiver. Protocol was necessary to specify the beginning and end of each message. This required start-stop information to be transmitted together with the message data. Discussion is extended here to a single link with identical sources and receivers. Naively one would expect that in addition to start-stop information there would need to be additional data conveying destination information to the receiver node. It will be shown that start-stop protocols are sufficient to express destination as well as the beginning and end of messages.

A simple model of a link with identical sources will be developed. Two coding strategies for protocol information will be presented and compared: the Huffman and universal<sup>12</sup> coding schemes. Redundancy of the coding schemes over source entropy will be considered as a performance measure for making comparisons, and estimating efficiency. One consequence of this section will be to demonstrate that addressing information in a data network can be avoided by selecting the appropriate encoding of protocol information.

## 6.2 Model of a single link with identical sources

A set of  $K$  identical independent synchronous sources share a binary channel to a corresponding number of receivers. Each source,  $k$ , ( $1 \leq k \leq K$ ) communicates only with its receiver,  $k$ . The source node resembles a concentrator which serves all  $K$  sources by inspecting their contents at each instant of time (see Figure 6.1.)

Each source can exist in either one of two states: the idle and busy states. Transitions between states take place in a synchronous manner with changing time instants. Each source is represented by a markov process (see Figure 6.2), where the probability of being idle is

$$\text{Prob. (idle)} = \frac{\epsilon}{\epsilon + \delta}$$

and the probability of being busy is

$$\text{Prob. (busy)} = \frac{\delta}{\epsilon + \delta}$$

When idle, the source delivers idle characters,  $i$ , and when busy it delivers binary 0's and 1's, corresponding to the message data.

Information relating to state and message data in a two state markov source may be quantified by the source entropy,  $H(S)$ , where

$$H(S) = \frac{\delta}{\epsilon + \delta} + \frac{\delta}{\epsilon + \delta} H(\epsilon) + \frac{\epsilon}{\epsilon + \delta} H(\delta) \quad \text{bits/unit time}$$

and  $H(x) = -x \log(x) - (1-x) \log(1-x)$  (6.2.1)

The entropy of the source provides a lower bound on the average length of codewords required to transmit all information relating to

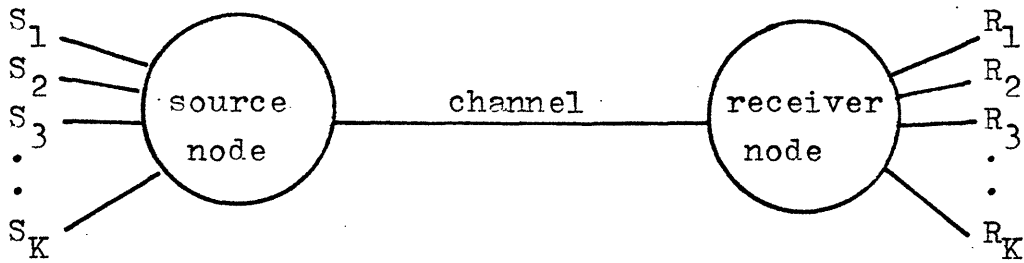


Figure 6.1

Identical synchronous source/receiver pairs

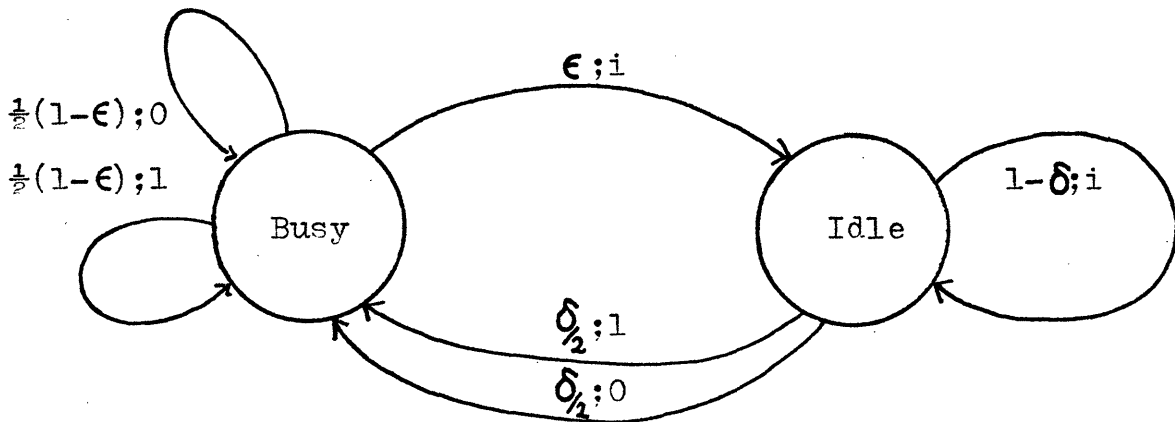


Figure 6.2

Markov information source

that source, including state and message data (see source coding theorem, section 3). For  $K$  identical independent sources, each with entropy  $H(S)$ , the minimum capacity of the binary channel must be greater than the total entropy of the combined sources,  $KH(S)$ , in order that reliable transmission can take place.

The entropy for the combined sources,  $KH(S)$ , contains three terms (see equation (6.2.1)). The first,  $K \frac{\delta}{\epsilon + \delta}$  is the average message data per time instant. The second,  $K \frac{\delta}{\delta + \epsilon} H(\epsilon)$ , is the stop information per unit time for all the sources, and the third is the start information,  $K \frac{\epsilon}{\delta + \epsilon} H(\delta)$ .

A meaningful performance measure which is concerned with minimizing overhead data is the coding redundancy of a protocol over the source entropy. Huffman encoding of stop information has been shown to be efficient in this sense, and will be used again for start-stop protocols. A second scheme involving universal coding<sup>12</sup> will also be discussed as an alternative approach.

### 6.3 Huffman encoding of start-stop information

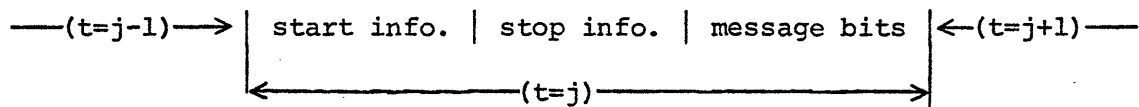
Let  $K_I(j)$  and  $K_B(j)$  denote the number of sources in the idle and busy states respectively, during the  $j^{\text{th}}$  time instant. The total number of sources is  $K$ . As the system enters the  $j+1$  time instant, a finite number of sources,  $q$ , change from the idle to busy state ( $0 \leq q \leq K_I(j)$ ), and a finite number,  $p$ , become idle after completing a message, ( $0 \leq p \leq K_B(j)$ ). The new state contains  $K_I(j+1)$  idle and



$K_B(j+1)$  busy sources.

If initially all sources are assumed idle, and both the source and receiver nodes maintain a list of idle and busy sources at all time instants, only information conveying those sources in transition need be transmitted at any one time instant to update the lists. The start (and stop) information is the location of the sources in the idle (and busy) list which become active (inactive). This information is sufficient to allow the receiver to update lists, and allocate the received message bits to the appropriate receivers corresponding to active sources.

For a time instant,  $j$ , a possible format for transmitted data could be



Before being able to devise a coding scheme for the start-stop protocol information, it is necessary to identify the structure of the information source. This may be done for the start information by considering those idle sources which become active during one time instant,  $j$ . They are chosen randomly out of the list of idle sources,  $K_I(j-1)$ , where the probability of  $q$  transition is given by the binomial distribution

$$\Pr(q) = \binom{K_I(j-1)}{q} (\delta)^q (1-\delta)^{K_I(j-1)-q} \quad (6.3.1)$$

\*  $\binom{a}{b} = \frac{a!}{(a-b)!b!}$

and  $0 \leq q \leq K_I(j-1)$

A similar distribution applies to busy sources which become idle (p of them) where

$$\Pr(p) = \binom{K_B(j-1)}{p} (\epsilon)^p (1-\epsilon)^{K_B(j-1)-p} \quad (6.3.2)$$

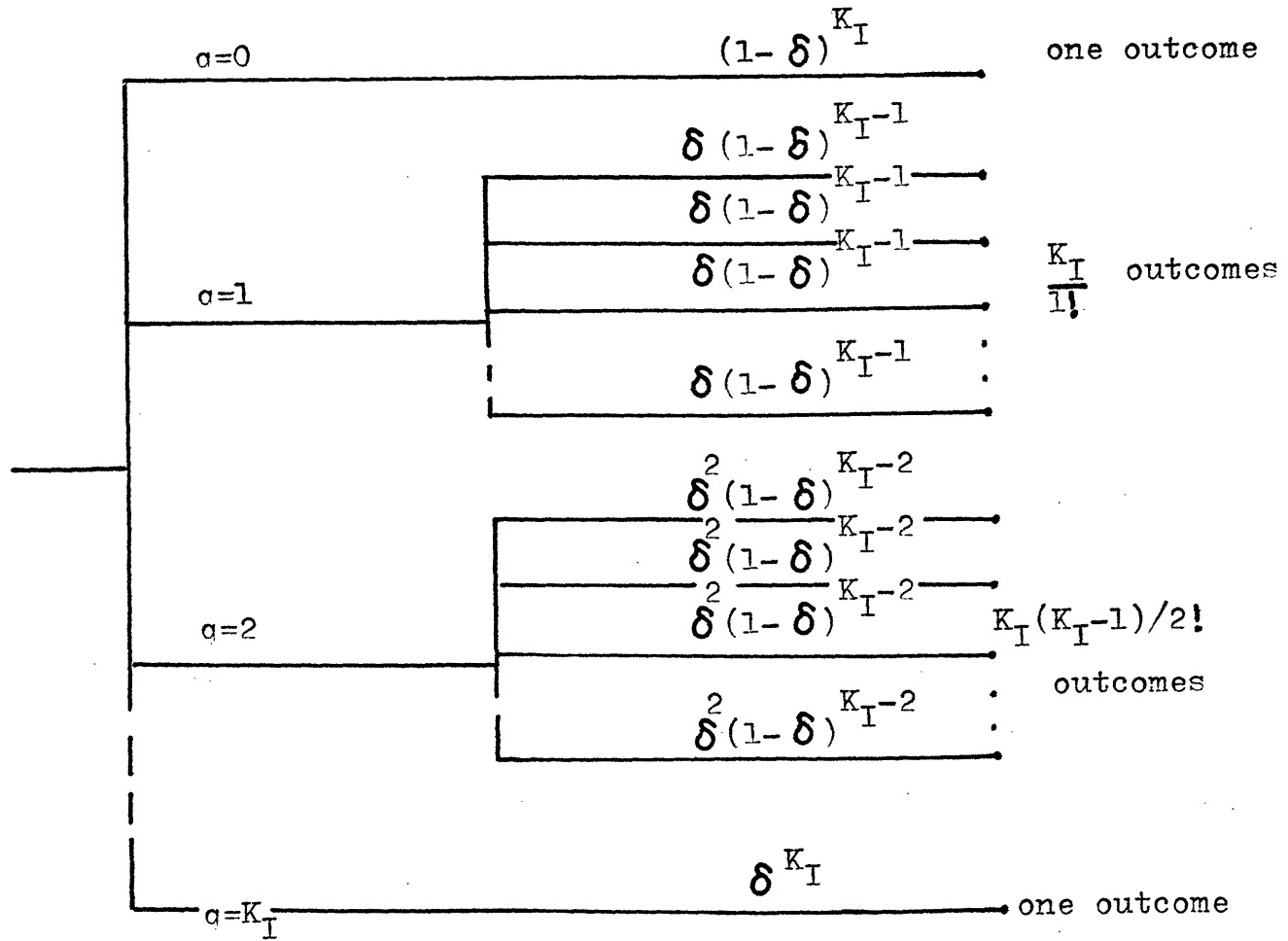
and  $0 \leq p \leq K_B(j-1)$

The set of all possible outcomes for transitions in either direction may be enumerated with the help of a coding tree, where each terminal node represents a unique outcome (Figure 6.3). To construct a Huffman code for such a finite tree would require an accurate knowledge of  $\epsilon$ ,  $\delta$ , together with much computation.<sup>6</sup> The coding problem may be simplified by considering the distances between sources in transition (i.e., run length coding), in the idle and busy lists.

For sources in the idle state, the probability of a transition is  $\delta$ . The probability distribution of the distance,  $d$ , between two transitions is geometric, providing that the list is infinitely long, where

$$\Pr(d) = \delta(1-\delta)^{d-1} \quad \text{and} \quad 1 \leq d \leq \infty \quad (6.3.3)$$

The condition of infinite length may be realized by concatenating the lists of idle sources at different time instants into one infinitely long list. In such a case, the distance between transitions is described exactly by (6.3.3). The position of time markers corresponding to the division between lists at different times  $j-1$ ,  $j$ , etc., can be



<u>transitions</u>	<u>probability</u>	<u>outcomes</u>
(q)	$\delta^q(1-\delta)^{K_I-q}$	$\binom{K_I}{q}$

Figure 6.3

Coding tree for transitions to the busy state ( $K_I$  sources)

indicated to the receiver node by an extra bit in the length encoding of those distances which include the markers (see Figure 6.4).

To demonstrate the encoding of a distance between sources in transition, by the Huffman coding scheme, consider a typical distance,  $d$ . Define a fixed parameter  $L$  which depends on mean distance between transitions; in the idle list,  $1/\delta$ . From (4.6.5)

$$L = (1/\delta) \ln(2) \quad (6.3.4)$$

Define a second integer variable,  $N$ , such that

$$N = \left\lceil \frac{d}{L} \right\rceil$$

The codeword for  $d$  is constructed in two parts. The first is a unary code of  $N-1$  binary ones followed by a binary zero. The second is a variable length encoding of  $[d] \text{ Mod}(L)$ , whose mean length is  $\log_2(L)$ , resembling the coding employed in section 4.6. The distance,  $d$ , may be expressed as

$$d = (N-1)L + [d] \text{ Mod}(L) \quad (6.3.6)$$

The coding of distance between transitions in two different lists, i.e., at times  $j$ ,  $j+1$ , introduces an undetermined future event. For example, when coding start information at time  $j$ , no information is yet available about transitions at time  $j+1$ .

If the last source to become active in the list of idle sources at time  $j$  is  $s_k$  ( $0 \leq k \leq K$ ), which is at distance  $d_k^j$  from the end of the list, the source node can only indicate to the receiver node that the

Distance between idle sources in transition to busy state

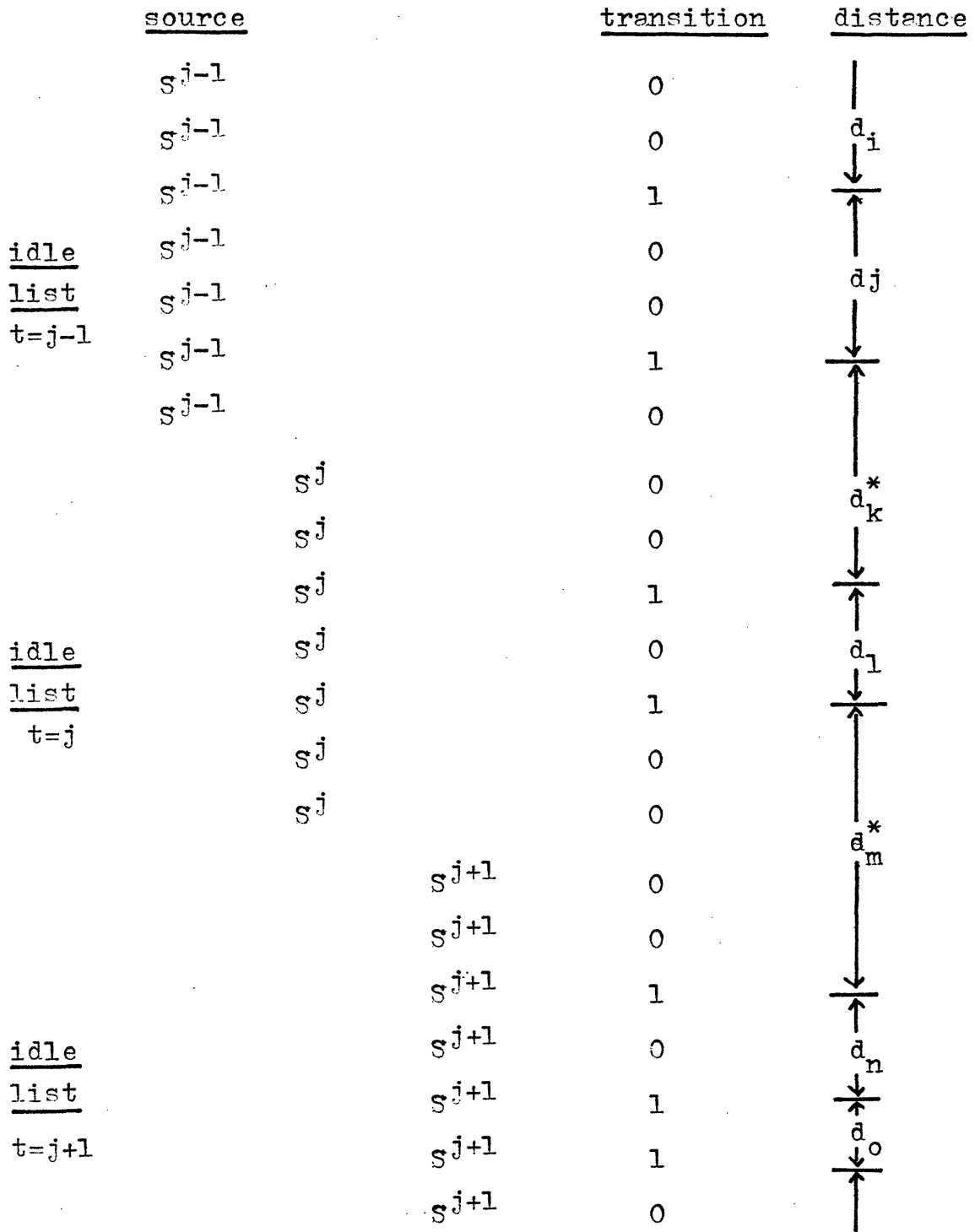


Figure 6.4

\* includes time marker;

next source in transition is beyond the last source in the idle list,  $s_k$ .

The final codeword in the start information field for the  $j^{\text{th}}$  instant of time will consist of  $N = \left\lceil \frac{d_k}{L} \right\rceil$  binary ones. The receiver will count  $NL$  source positions down its own idle list, which will take it beyond the final element. The receiver will assume that all start information for time  $j$  is complete, and will look for stop information.

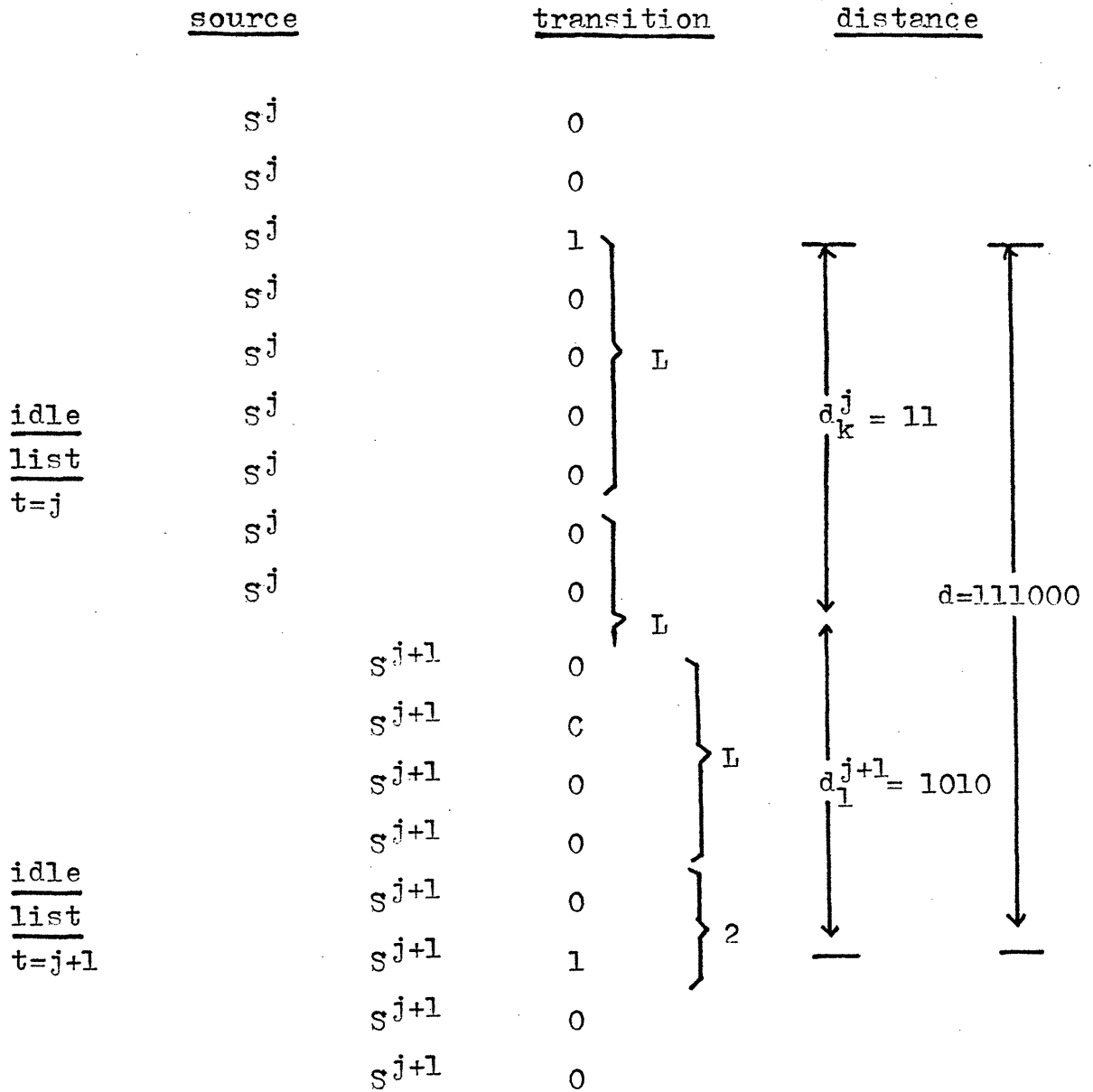
The first codeword for the start (or stop) information in the  $j+1^{\text{th}}$  time instant records the position,  $d_1^{j+1}$ , of the first source in transition from the top of the idle (or busy) list (see Figure 6.5). The redundancy incurred by specifying the distance between the last source at time  $j$  and the first source at time  $j+1$  as two codewords instead of one is found in Appendix 6 to be 0.614 bits. The distances  $d_1^{j+1}$  and  $d_k^j$  still have a geometric distribution owing to the special property of the source statistics, i.e., regardless of where one starts to count to the next source in transition, the distance,  $d$ , remains geometrically distributed.

The encoding of transitions in the busy source list proceeds in the same manner, except that parameter  $L$  is redefined as

$$L = (1/\epsilon) \ln(2) \quad (6.3.6)$$

On receiving all start-stop information, the receiver can compute the number of message bits originating at the active source at the particular instant of time. These bits are transmitted in the same order

Length encoding between lists,  $t=j, j+1$



Coding table ( $L=4; \log_2 L=2$ )

$[x] \bmod(L)$	codeword
0	00
1	01
2	10
3	11

Figure 6.5

as the sources appear in the busy list so that they may be routed to the appropriate receiver pairs, in corresponding positions in the receiver node list.

In the case of no idle (or busy) transitions in one time instant, an appropriate number of binary ones are sent to indicate that the distance between sources in transition exceeds the idle (or busy) list size.

#### 6.4 Performance of the Huffman coding scheme

In order to reduce cost in a network, i.e., minimize channel capacity (allocated on a unit cost basis), it is necessary to design protocol strategies with small coding redundancy. A further objective of practical importance is to design protocols which minimize transmission delays, including queueing time at nodes. The single source/receiver link illustrated how queueing delays are related to the expected value of overhead data as well as the second moment.

The arrival of message bits at the source node of a link with  $K$  identical sources does present a queueing problem if the number of sources is small, and the channel capacity only sufficient to transmit an average message load. In order to eliminate second moments of protocol data from this discussion, it has been assumed that the value of  $K$  is large enough to allow the statistical law of large numbers to operate, ensuring that the load at all times corresponds with channel capacity. This assumption implies that a protocol strategy which



minimizes expected overhead data is to be considered the most efficient scheme for conveying protocol information.

It is convenient to analyze the expected overhead or protocol data associated with a single message delivered by one of the  $K$  identical sources. This will allow us to compare the coding redundancy of protocol information for the single source/receiver model with the link with identical sources model.

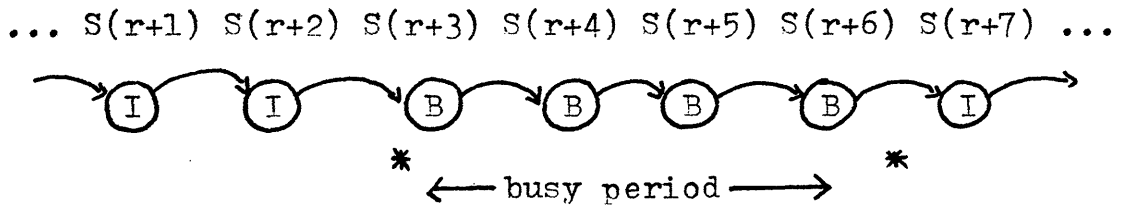
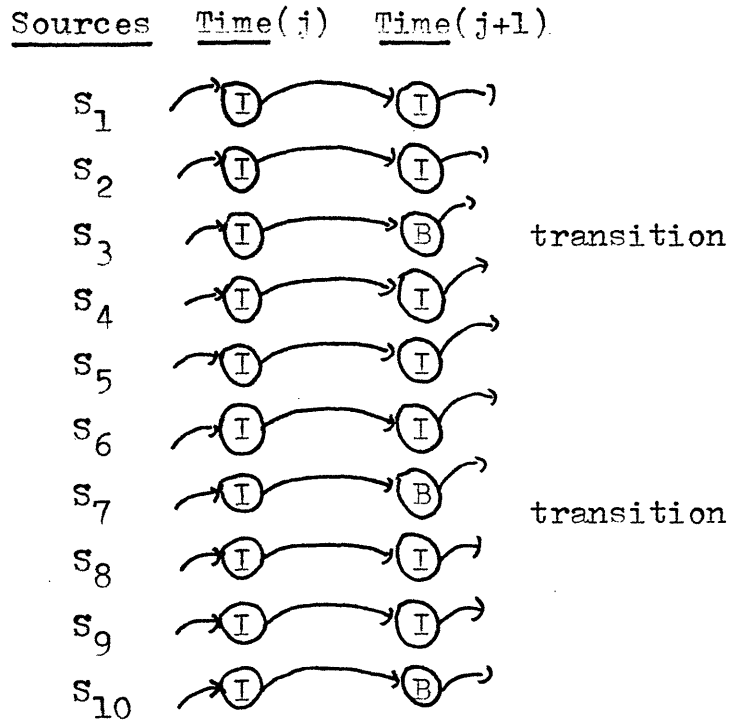
The entropy of a single message from a markov source, with mean length  $1/\epsilon$ , and idle period  $1/\delta$ , is  $H(S)$ ,

$$H(S) = 1/\epsilon H(\epsilon) + 1/\delta H(\delta) + 1/\epsilon \quad (6.4.1)$$

The first term represents message length, or stop information, the second represents idle or start information, and the third is message content. One bit of message data is delivered by the source, when busy, each instant of time.

The statistics of a single source over many time instants are identical to those of a chain of markov sources at one time instant. For example, consider the list of busy sources at time  $j$ . The length between two sources in transition in the list has the same geometric distribution as the busy period of a single markov source over many instants of time. The mean distance between transitions, and busy period length is  $1/\epsilon$ . The Huffman coding of stop information in earlier sections was for a single source over many time instants. Here the encoding is performed for sources in transition in the busy list at one time instant (see Figure 6.6).

List of identical sources at time j, j+1



Single source over many time instants

Figure 6.6

The coding described in Section 6.3 achieves a redundancy of mean value about 0.03 bits (this fluctuates between 0.02 and 0.04 bits depending on expected message length) above the source entropy, for each encoded distance between transitions<sup>7</sup>, excluding the extra 0.614 bits required to indicate the time marker (see Appendix 6).

The redundancy incurred by specifying the end of the busy and idle lists amounts to 1.288 bits (on average) per instant of time. Over a long time period, the average number of messages per instant averages out to be  $\epsilon\delta K/(\epsilon + \delta)$ , where the length of a message and preceding idle period is  $1/\epsilon + 1/\delta$ . The weak law of large numbers gives an average bit redundancy per message,  $R$ , of

$$R = 0.06 + 1.288 \frac{(\epsilon + \delta)}{\epsilon\delta K} \quad (6.4.2)$$

Define  $\eta$  as the ratio of the coding redundancy,  $R$ , to the length of a message,  $1/\epsilon$ . Then

$$\eta = \epsilon R = 0.06\epsilon + \frac{1.288}{K} + \frac{1.288}{K} \frac{\epsilon}{\delta} \quad (6.4.3)$$

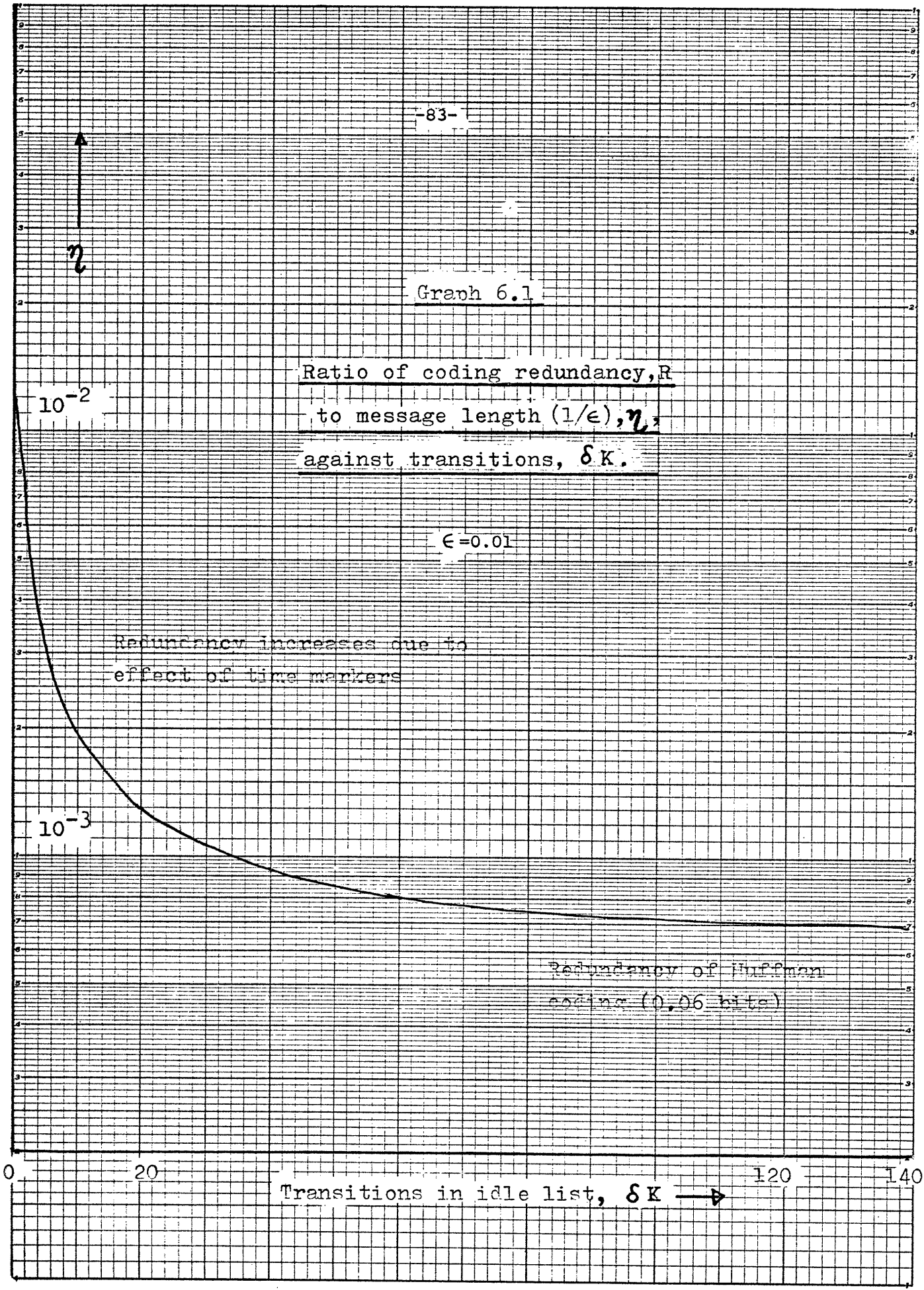
Consider the case for large values of  $K$ , and infrequent message arrivals, such that the idle period is much greater than the busy period,  $1/\delta \gg 1/\epsilon$ . In the range  $1 \leq \delta K \leq 100$ , the approximation for  $\eta$  can be made, where

$$\eta \approx \epsilon \left( \frac{1.288}{\delta K} + 0.06 \right) \quad (6.4.4)$$

and  $\delta K_I \approx \delta K$

Graph 6.1 illustrates how the coding redundancy ratio depends on

Graph 6.1



message length,  $1/\epsilon$ , and number of transitions, per time instant for both start and stop protocols. The effect of the time marker on a message becomes negligible in the upper range of  $k$  transitions. In such a case, the redundancy of protocol encoding corresponds to that of the single source/receiver model, i.e., 0.06 bits on average per message.

### 6.5 A universal coding approach

A second approach to the coding of start-stop information can be made using a universal coding scheme<sup>12</sup>. Again lists of idle and busy sources are maintained at each node, and updated each instant of time. The coding of source transitions in the two lists may best be explained by an example.

Consider a list of eleven idle sources at time  $j-1$ . Two sources become active before the  $j^{\text{th}}$  time instant, say at the second and fifth positions in the list. A binary word is constructed to represent this change, where sources which remain idle are represented by a binary zero, and those which become active by a binary one. The information for this event would then be 0100100000 where the first bit of the word refers to the first source in the list, etc.

The universal code proceeds by transmitting the number of transitions, in this case two, as a run length code word, i.e., 110. Having conveyed the number of transitions, the possible number of eleven bit binary words are reduced from  $2^{11}$  to  $\binom{11}{2} = 55$ . Each outcome is equally likely, and may be encoded by a fixed length word of  $\lceil \log_2(55) \rceil = 6$  bits.

The receiver has a decoding table for each of the possible transitions. Listed below are some possible codewords for two transitions in an eleven source list:

List	Codeword (6 bits)
11000000000	000000
10100000000	000001
10010000000	000010
10001000000	000011 etc.
. . . . .	. . . . .
01001000000	001110

The codeword for 01001000 is the concatenation of 110 (two transitions), and 001110 (location of the transitions).

The universal scheme is most efficient for small numbers of transitions. It has one advantage over the Huffman scheme in that it is constructed for a finite coding tree, and does not need to specify time markers between lists. Inspecting the coding tree in Figure 6.3, it is seen that the universal scheme, by specifying the number of transitions, reduces the tree to one branch with  $\binom{K_I}{q}$  equally likely outcomes.  $K_I$  is the number of sources in the list, and  $q$  the number of transitions.

### 6.6 Performance of the universal code

Assuming that the lists,  $K_I$  and  $K_B$ , are very long, and the number of transitions are large, sterlings approximation may be employed to give an estimate of the average codeword length for start and stop

information,  $\bar{n}_I, \bar{n}_B$ . The distribution of transitions in the lists has been given in (6.3.1) and (6.3.2). The length of the run length codeword conveying the number of transitions is  $q + 1$ . The codeword conveying position is  $\log_2 \left[ \binom{K_I}{q} \right]$ .

$$\bar{n}_I = E(q+1) + E \left( \left[ \log_2 \binom{K_I}{q} \right] \right) \quad (6.5.1)$$

Sterling's approximation gives

$$\log_2 \binom{K_I}{q} \approx K_I H(q/K_I) + \frac{1}{2} \log \left( \frac{K_I}{2nq(K_I - q)} \right)$$

For sufficiently large  $K_I$ ,  $q/K_I \approx \delta$ , so that

$$\bar{n}_I \approx \delta K_I + 1 + K_I H(\delta) \quad (6.5.2)$$

To evaluate the number of protocol bits per message required to transmit start-stop information,  $\bar{n}_I$  represents the jointly encoded start information for  $\delta K_I$  transitions. For each transition,

$$\left( \frac{\bar{n}_I}{\delta K_I} \right) \approx 1 + 1/\delta H(\delta) \text{ (bits/message); } \delta K_I \gg 1 \quad (6.5.3)$$

The second term is the entropy of the start information, inferring that one bit of redundant code per message is required for start information. Similarly, one bit of redundancy occurs in the stop information giving a total of two bits of redundant protocol data per message in the universal coding scheme.

### 6.7 Conclusions on protocol strategies

The protocol information necessary to communicate independent messages over a binary channel between K source/receiver pairs includes the beginning, the end and the destination of the messages. By considering the source node as a simple concentrator serving K sources, it is sufficient to convey only start-stop information from the source to receiver node, in addition to message data.

The coding of start-stop information was performed by Huffman and universal coding schemes, independently, to illustrate two separate and efficient source codings. The performance of each scheme was measured by comparing average codeword lengths to source entropy (the lower bound, according to the source coding theorem of Section 3). Efficient source coding of protocol information ensures that average overhead data is kept to a necessary minimum.

The coding redundancy of the Huffman scheme, R, has three components,

$$R = 0.06 + \frac{0.614}{\delta K_I} + \frac{0.614}{\epsilon K_B} \quad (6.7.1)$$

The constant first term is associated with the coding of distances between transitions, i.e., idle and busy period lengths (0.03 bits according to Ref. 6). The other terms occur because an additional bit is generally required each time instant in both the start and stop information fields to indicate the end of each set of codewords. This bit is averaged out over all length encodings per time instant, and becomes



negligible for large values of  $K$  (see Graph 6.1).

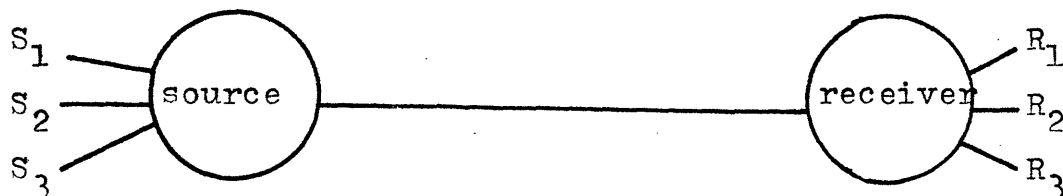
The universal scheme is less efficient when several transitions take place per time instant, with a bit redundancy per message of two. It is also considerably less practical due to the large number of decoding tables which must be kept at the receiver node to identify the position of source transitions. The Huffman scheme, a run length coding technique, is simple to implement, and efficient in terms of average length of codewords.

#### 6.8 System design implications

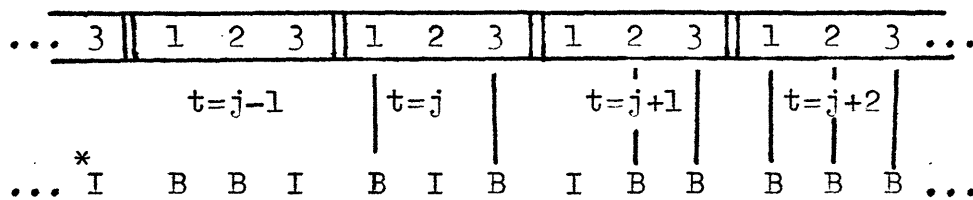
A concentrator, as incorporated into our simple  $K$  source/receiver network, is an efficient means of allocating channel bandwidth to many users. It avoids the need for address information, as do time division multiplexing systems, but also allocates channel space dynamically thus ensuring no empty time slots.

Simple time division multiplexing systems provide regular time slots for all sources sharing a link. If one or more sources are idle, the time slot will be empty. The concentrator keeps a list of all active sources, and allocates one time slot to each of these per time instant. No channel space is allocated to idle sources. The statistical allocation of bandwidth by a concentrator is as efficient as transmitting messages in a single queue, but in addition eliminates address information which would be necessary under a single server approach (see Figure 6.7).

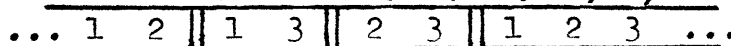
Three source/receiver network



A. Time division multiplexor (all S/Rs allocated a slot)

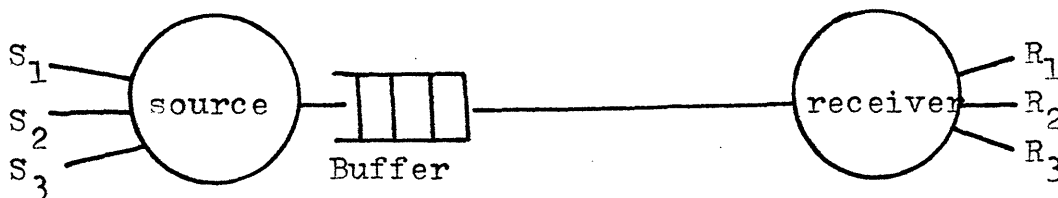


B. Concentrator

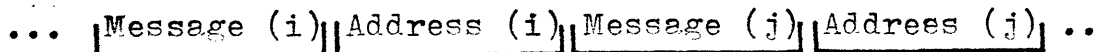


(busy S/Rs only allocated a slot)

C. Single server



Transmitted data:



\* I = idle state  
B = busy state

Figure 6.7

Appendix 6

Encoding distance between sources at time instants  $j, j+1$

Consider the start information for time instant  $j$ . In the list of idle sources, the last source to become active is distance  $d_k^j$  from the end of the list. There may be no transition at time  $j$  in which case,  $d_k^j$  is the total number of sources in the list. For positive integers  $N_1 = \left\lceil \frac{d_k^j}{L} \right\rceil$  and  $R_1$  ( $0 \leq R_1 < L$ ), the coding of  $d_k^j$  is performed according to the procedure of Section 6.3 where  $L = 1/\delta \ln 2$  and

$$d_k^j = (N_1 - 1)L + R_1 \quad (6a)$$

The codeword for  $d_k^j$  consists of  $N_1$  binary ones, which indicates to the receiver that the next transition is distance  $(N_1)L$  sources from the previous one, which will be beyond the final element of the idle list at time instant  $j$ . In the next time instant,  $j+1$ , the first codeword of the start information will give the position of the first source in transition from the top of the idle list,  $d_1^{j+1}$ . If there are no sources in transition in this list, then a similar procedure used for the final codeword at time  $j$  is adopted. For positive integers  $N_2$  and  $R_2$ , where  $0 \leq R_2 < L$ , distance

$$d_1^{j+1} = (N_2 - 1)L + R_2 \quad (6b)$$

The coding of  $d_1^{j+1}$  will consist of  $N_2 - 1$  binary ones followed by a zero, and a variable length encoding of  $R_2$ , average length  $\log_2(L)$ .

In the case of no transitions,  $N_2$  ones are sent alone. The number of sources between the two sources in transition (in lists  $t=j, j+1$ ) is  $d$ ,

$$d = d_k^j + d_1^{j+1} = (N_1 + N_2 - 2)L + R_1 + R_2 \quad (6c)$$

(see Figure 6.5).

The mean lengths of the codewords for  $d_k^j$  and  $d_1^{j+1}$  together exceed the mean length of  $d$ , as coded directly by the Huffman scheme, by a fraction of a bit,  $x$ . This fraction is the redundancy caused by the additional specification of a new list at time  $j+1$  in addition to the distance to the next source in transition,  $d$ :

Having defined the procedure for encoding distances  $d_k^j$  and  $d_1^{j+1}$ , it is now possible to calculate  $x$  by comparing the mean lengths of the two codewords to that of  $d$  (coded as an integer by the Huffman scheme). Consider the event A which occurs when  $R_1 + R_2 < L$ ; the codeword for  $d$  has length  $n_d$ , where

$$n_d = N_1 + N_2 - 1 + \log_2 L \quad (6d)$$

A second event, B, occurs when  $R_1 + R_2 \geq L$ , such that the length of the codeword for  $d$  becomes

$$n_d = N_1 + N_2 + \log_2 L \quad (6e)$$

The combined lengths of the two codewords for  $d_k^j$  and  $d_1^{j+1}$  are  $n_d^1$  and  $n_d^2$ , where

$$n_d^1 + n_d^2 = (N_1) + (N_2 + \log_2 L) \quad (6f)$$

Only during event A is the coding of  $d$  performed more efficiently

by one codeword rather than two, comparing (6f) with (6d, 6e). In event A, a single bit must be included in the start information to indicate the time division between lists. Thus the mean value of the bit redundancy is  $x$ , where

$$\begin{aligned}x &= 1 \cdot \text{Pr}(\text{event A}) + 0 \cdot \text{Pr}(\text{event B}) \\ &= \text{Pr}(\text{event A}) \\ &= \text{Pr}(R_1 + R_2 < L)\end{aligned}\tag{6g}$$

The integers  $R_1$  and  $R_2$  are random variables with distributions of the form

$$P_R(r) = \frac{\delta(1-\delta)^{r-1}}{1-(1-\delta)^L}\tag{6h}$$

where

$$0 \leq r < L$$

The two variables are statistically independent. As

$$\begin{aligned}\text{Pr}(A) &= 1 - \text{Pr}(B) \\ &= 1 - \sum_{r_2=0}^{L-1} P_{R_1}(r_1 \geq L - r_2) P_{R_2}(r_2).\end{aligned}$$

Evaluating the previous expression,

$$\Pr(A) = 1 - \frac{\delta L (1-\delta)^{L-2}}{(1-(1-\delta)^L)^2} + \frac{(1-\delta)^{L-1}}{1-(1-\delta)^L} \quad (6i)$$

For large values of L, ie: long idle periods, the approximation below can be made,

$$(1-\delta)^L \approx 1/2$$

where  $L = (1/\delta) \ln(2)$

The probability of event A, for long idle periods ( $1/\delta \gg 1$ ), becomes

$$x = \Pr(A) = 0.614 \text{ bits} \quad (6j)$$

In the case where no transitions occur at time j, the analysis remains unchanged, although  $d_k^j$  covers the entire list of  $K_I(j)$  elements rather than the final part.

## Section 7

### The Conclusion

#### 7.1 Summarizing the network protocol problem

Three categories of protocol information have been discussed at length here in order to illustrate a design procedure based on source coding. These include the beginning, end, and destination of messages. The objective in each case was to find a protocol strategy which minimizes the average control data which accompanies messages during transmission. In so doing, channel bandwidth requirements and transmission delays are minimized, leading to a more efficient useage of network resources.

#### 7.2 The design of efficient network protocols

Protocol information is necessary in a communications network to resolve the statistical uncertainties associated with incoming messages, including arrival time, length and destination. These uncertainties may be modelled by separate information sources from those supplying message data. The design approach adopted here was to find efficient source encodings which met the lower bounds already constructed for some protocols<sup>4</sup>.

The availability of reliable statistics of network users is essential in order to achieve efficient source codes. For purposes of illustration, some standard distributions have been assumed including geometrically distributed message lengths and poisson arrivals.

### 7.3 Start-stop protocols

The start-stop protocols associated with message arrival and length information are found to be related to the lengths of idle and busy periods of the information source. A Huffman coding scheme is available to encode efficiently the geometrically distributed integers corresponding to these periods. The scheme is discussed both for single source and receiver links (Sections 3,4,5) and many identical source/receiver pairs sharing a single link (Section 6).

For the single source/receiver model, the condition variable length packets was found necessary to achieve a small coding redundancy of message length, or stop information. Such a condition may be met by using either a terminating flag character or a Huffman length encoding (with slight advantage to the latter scheme). Section 5.6 summarizes the relative performance of a fixed packet, terminal flag and Huffman length encoding approach to stop information for single source/receiver models.

By employing an information concentrator in a network with identical sources sharing a single link, the need for separate address information was eliminated. It was found sufficient to provide the receiver node with start-stop information for each source/receiver pair individually in order to communicate messages to the appropriate destinations. Again a Huffman length encoding scheme efficiently conveyed the start-stop information, which contained the lengths of the idle and busy



periods. Section 6.7 summarizes the efficiency of this approach, and section 6.8 comments on design implications.

#### 7.4 Problems of future interest

The problem of transmission delay is of major practical importance in communication networks. A source coding approach to protocol information is able to reduce network overheads to a minimum, but sometimes at the expense of increased delay. For example, it is more efficient to perform joint encodings of message lengths and arrivals than to send individual protocol information for each message to be transmitted. However, joint encoding assumes that one waits for several messages to arrive in a queue before commencing transmission. The relationship between delay and efficient protocol coding is yet to be explored from a practical standpoint.

The second major issue in network protocols is the routing and supervisory information. No bounds yet exist for such information with which to evaluate existing protocols, and devise more efficient ones. The problem is complicated by the intimate relationship between effective control and state information in a network. By supplying additional information on traffic flow conditions, it is generally possible to improve routing of messages, and thus utilize network capacity more efficiently. However, the control information itself reduces network capacity, and message flow. This field will require a joint control and communications approach.

The rapid expansion of data networks in the near future should provide the economic incentives to improve the efficiency of overhead information. The study of protocol structure and implementation could offer significant savings in system overheads.

References

- [1] Wesley W.Chu, Advances in Computer Networks, ARTECH House, 1974
- [2] S.Carr,S.D.Crocker and V.G.Cerf "Host-host Communication Protocol in the ARPA Network" AFIPS Spring Conference 1970, vol 36
- [3] V.G.Cerf and R.E.Kahn "A Protocol for Packet Network Inter-communications", IEEE Trans. Comp. vol Com-22 No.5, May 1974
- [4] Robert G.Gallager "Basic Limits on Protocol Information in Data Communication Networks" M.I.T. publication, August 1974
- [5] Robert G.Gallager, Information Thoery and Reliable Communications, John Wiley and Sons, N.Y., 1968
- [6] D.Huffman "A Method for the Construction of Minimum Redundancy Codes" Proc. IRE, vol 40, 1962
- [7] Robert G.Gallager and D.C.Van Voorhis "Optimal Source Codes for Geometrically Distributed Integer Alphabets" IEEE Transactions Inform. The., vol IT-21 March 1975
- [8] D.Gross and C.M.Harris, Fundamentals of Queueing Thoery John Wiley and Sons, 1974
- [9] J.R.Kersey (IBM) "Synchronous Data Link Control" from Data Communications, McGraw Hill, 1974
- [10] William Feller, Introduction to Probability Theory and Its Applications, Volume 1, John Wiley, 1968