Computer Systems

G. Bell, D. Siewiorek, and S.H. Fuller, Editors

A Study of Line Overhead in the Arpanet

Leonard Kleinrock, William E. Naylor, and Holger Opderbeck
University of California at Los Angeles

The form, extent, and effect of the communication line overhead in the ARPANET are considered. The source of this overhead is separated into various levels of protocol hierarchy and the characteristics of each level are summarized. Then the line efficiency for various models of system use is studied. Some measurements of line efficiency for the ARPANET are presented and by extrapolation these measurements are used to anticipate overhead in a heavily loaded network. Similar results are derived for a recently proposed network protocol and compared with those for the current system.

Key Words and Phrases: ARPANET, computer communication networks, interprocess communication, measurement, packet switching, performance evaluation and efficiency, resource sharing

CR Categories: 3.81, 4.39, 4.6, 4.9

1. Introduction

The choice of control characters and control messages is one of the most important decisions in the design of computer communication networks. More elaborate control procedures or protocols should tend to make the exchange of data smoother and safer in a well-designed system. However, as with sophisticated operating systems, there is a limit to the complexity of the control procedures which is determined by the amount of overhead these procedures introduce. Beyond a certain point, the increase in overhead is too large to justify additional complexity for an intended improvement of service. Therefore it is important to analyze carefully the overhead characteristics of a given system.

It is a well-known fact that the resource allocation overhead in time-sharing systems depends critically on the behavior of the running programs. Similarly, the line overhead in communication networks depends critically on the characteristics of the exchange messages. A large number of small messages involves more overhead than a small number of large messages to transfer the same number of data bits. The line capacity and protocol must therefore be selected with respect to the expected characteristics of the message exchange.

In this study, we focus on the line overhead in packet-switched computer communications networks, using the ARPANET [1, 8–10, 20] as our basic model. We define line overhead as all those characters transmitted that are not exchanged between user processes in the attached computer systems (called HOSTs). A user process is here defined to be any process that makes use of the system calls provided for the Network Control Program (or NCP). With this definition of a user process we exclude from this study some higher level overhead. For example, a process that controls the transfer of files generates control messages that contribute to the total overhead. However, these higher level control messages cannot be recognized as such by only observing the traffic in the communications subnet. They have been excluded since our overhead study is based on subnet measurements at the UCLA Network Measurement Center (UCLA-NMC) only [11]. In what follows, then, we focus on the overhead induced by the subnet protocols and the HOST-to-HOST protocol. We will exclude from discussion the overhead due to the control messages of TELNET (terminal access) [23], FTP (file transfer) [7], and other higher level protocols [5].

A discussion of the effect of *flow control* on the delay experienced by user data packets is beyond the scope of this paper. (This, after all, is overhead as far as the user is concerned.) However, we show some effects that the HOST-to-HOST flow control mechanism has on line efficiency (and therefore throughput). In this study, we are principally concerned with throughput (i.e. efficiency) as opposed to delay. We derive a simple formula for the line efficiency as a function of the traffic characteristics. This allows us to do a best and worst case overhead analysis. Then a breakdown of the overhead for the current traffic characteristics is presented. The same data is calculated for a saturated net under the assumption that the traffic characteristics

Copyright © 1975, Association for Computing Machinery, Inc. General permission to republish, but not for profit, all or part of this material is granted provided that ACM's copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Association for Computing Machinery.

This research was supported by the Advanced Research Projects Agency of the Department of Defense under Contract No. DAHC-15-73-C-0368. An abbreviated version of this paper was presented at a conference on "Computer Communication Networks" in Baden, Austria, on October 22, 1974 (sponsored by the International Institute for Applied Systems Analysis, Laxenburg, Austria). Authors' addresses: L. Kleinrock and W. E. Naylor, Computer Science Department, University of California at Los Angeles, Los Angeles CA 90024; H. Opderbeck, Telenet Communications Corporation, 1666 K Street, N.W., Washington, D.C. 20006.

Table I. Line Overhead Classification.

Category	Name	No. Bits	Description
Level-0	SYN	16	2 hardware-generated SYN characters for clock synchronization
	DLE/STX, DLE/ETX	32	4 hardware-generated control characters for message delimiting
	H-checksum	24	Hardware-generated checksum
	ACK-header	16	Software-generated control word carrying acknowledgment bits
	S-checksum	16	Software-generated checksum
Level-1	Packet header	80	5 16-bit words of packet header in each noncontrol message
	Subnet control	64	4 words for each subnet control message (see Table II)
Level-2	HOST/HOST protocol	40	Per message overhead specified by the HOST/HOST protocol
	HOST/HOST control	Average 93.5	Messages of different lengths for control of HOST/HOST traffic (see Table III)
Background	Routing	1160	Routing message sent every 640 msec (includes level-0 and level-1 overhead)
	IHY	152	I-heard-you message sent every 640 msec to determine up/down status of line (includes level-0 and level-1 overhead)
	Status reports	1728 (+ 336 for 2 RFNMs)	Status reports (2 packets) sent every 52.4 sec to the Network Control Center (NCC) (includes level-0 and level-1 overhead)

do not change. Similar results are then derived for a new network protocol which has recently been suggested [2] and compared with those for the current system.

2. Levels of Overhead

In the following we give a detailed description of the line overhead in the ARPANET. This overhead will be classified according to the following four categories:

(1) *Level-0 overhead.* Control of packet transmission between adjacent IMPs (the communications processors in the ARPANET).

(2) *Level-1 overhead.* Message control in the subnet, i.e. transmission control between source IMP and destination IMP.

(3) *Level-2 overhead.* Message control between HOSTs.

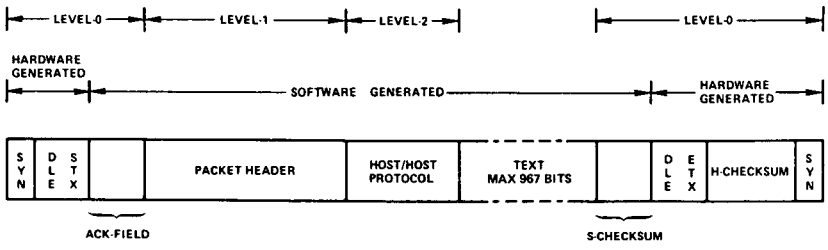
(4) *Background traffic overhead.* Routing messages, line status messages, status reports.

In the following discussion we will consider only that line overhead which is either constant or an increasing function of the network load. Thus we will ignore the following two kinds of line overhead that actually decrease with increasing user traffic: (1) "null packets" in which only the acknowledgment bits contain relevant information (these packets are only sent in the absence of other traffic), and (2) excess routing messages that are only sent if the line utilization is less than 80 percent.

Table I gives a detailed explanation of the line overhead on all three levels of communication and for the background traffic. The format of a single-packet user message is further illustrated in Figure 1. As Table I and Figure 1 show, there are nine hardware-generated characters for each transmitted packet. In addition, there are 16 bits for the IMP-to-IMP acknowledgment and 16 bits for the software checksum which contribute to the level-0 line overhead.

The level-1 line overhead consists of two parts: 80 bits for the packet header of each user message and 64

Fig. 1. Format of single-packet user message.



bits for each subnet control message. The 80 bits of the packet header are not exclusively used for message control in the subnet. There is, for instance, a field for message identification that is passed unmodified from the source HOST to the destination HOST and might therefore be considered level-2 line overhead. For simplicity, however, all 80 bits of the packet header are counted here as level-1 overhead.

The level-2 line overhead also consists of two parts: 40 bits for the extended leader of HOST-to-HOST protocol and an average number of 93.5 bits for each HOST-to-HOST control message as observed in the subnet by the packet tracing mechanism. (Padding bits are neglected.) We will assume that all messages adhere to the official HOST-to-HOST protocol. Although it is known that other private protocols are in use, they represent only a very small fraction of the total traffic.

The background traffic consists mainly of routing messages that are exchanged between any pair of adjacent IMPs at least every 640 msec (for 50 KBPS lines). The I-heard-you messages which are sent to test the status of the phone lines and the status reports which are sent by each IMP every 52.4 sec to the Network Control Center represent a much smaller fraction of the background traffic. Since the background traffic is assumed to be independent of the network load, the level-0 and level-1 overhead of the background messages is included in the line overhead for the background traffic. This distribution of line overhead will later facilitate our calculations.

3. Subnet Control Messages

A week-long measurement experiment in May 1974 showed that 49.15 percent of all packets transmitted in the ARPANET were subnet control messages. For a detailed description of the subnet control procedures the reader is referred to [9, 13, 21]. A list of all the subnet control messages, their frequency of occurrence, and their function is shown in Table II. The relative frequency of these subnet control messages was determined by means of a new measurement feature in the IMPs called "packet tracing" which was suggested by the UCLA-NMC and has recently been implemented by Bolt, Beranek and Newman, Inc. (BBN), the builder of the subnet. Using this packet tracing mechanism, about 75,000 subnet control messages were sampled from 35 different IMPs at different times of day and on different days of the week. The data of Table II represents the average over all these samples. Though not shown in the table, the deviation of the individual samples from the mean was remarkably small.

RFNMs for single-packet messages represent by far the largest fraction of subnet control messages. This is not surprising since some 96 percent of the messages entering the network are single-packet messages, as reported in [11] and further substantiated by the May 1974 experiment. It is interesting to note that we never observed an RFNM for a multipacket message that did not carry a "piggy-backed" ALLOCATE. This means that there is so much reassembly buffer space available that in almost all cases 8 packets can be allocated for the next transmission within 1 sec after the first packet of a multipacket message has been accepted by the destination HOST.

Table II shows that only about 2 to 3 percent of all messages are multipacket messages. This number is 1 to 2 percent smaller than the percent of multipacket messages reported in [11]. This discrepancy can be explained by the fact that the "incest traffic" [11] consists of proportionally more multipacket messages. Since the RFNMs generated by this incest traffic do not travel over any line, they have not been observed by means of our packet tracing method.

If our sampling technique were perfect, then the fraction of REQ-ALL messages, the fraction of ALL-M

Table II. Subnet Control Messages.

Name	% of total	Function
RFNM-S	88.77	Sent from destination IMP to source IMP to signal the correct receipt of a *single-packet message* (RFNM = *Request-for-Next-Message*)
RFNM-M	0.00	Sent from destination IMP to source IMP to signal the correct receipt of a *multipacket message*
ALL-S	3.98	Sent from destination IMP to source IMP to *signal the allocation* of one buffer for a *single-packet message*
REQ-ALL	1.09	Sent from source IMP to destination IMP to *request the allocation* of 8 buffers for a *multipacket message*
ALL-M	1.19	Sent from destination IMP to source IMP to *signal the allocation* of 8 buffers for a *multipacket message*
GIVEBACK	1.04	Sent from source IMP to destination IMP to *give back* an unsued buffer allocation that was received via a RFNM-ALL
RFNM-ALL	2.35	Combined effect of *RFNM-M* and *ALL-M*; the allocation for the next multipacket message is piggy-backed on the RFNM of the previous one
INCTRANS	1.13	An *incomplete transmission* message sent from destination IMP to source IMP for each message that could not be delivered correctly to its destination HOST
DESTDEAD	0.46	Sent from destination IMP to source IMP for each message that was sent to a *dead destination* HOST

messages, and the fraction of GIVEBACK messages would all be the same. This is because every request will sooner or later be granted and every allocation will, possibly after repeated use, be returned. The amount by which these fractions differ gives an indication of the accuracy of our sampling method. Table II shows that slightly more than 50 percent of all multipacket messages which enter the ARPANET do not need to request a buffer allocation at the destination IMP since such an allocation is already waiting at the source IMP to be used. Phrased differently, we can say that slightly less than 50 percent of all piggy-backed ALLOCATEs are not used by the source IMP and returned after a time-out of 125 msec to the destination IMP. This means that a much larger fraction of multipacket messages must wait for the necessary buffer allocation than one would have hoped in order to achieve a high throughput. There are two possible explanations for this behavior: (1) transfer of files which demand a long sequence of multipacket messages are relatively infrequent, (2) the time-out interval of 125 msec is too small compared to the HOST reaction time.

The large number of ALLOCATEs for single-packet messages is in agreement with the observations reported in [16]. Since the IMPs obviously are not short of reassembly buffers, all these ALLOCATEs are due to single-packet messages which arrive out of order at their destination IMP. The fraction of ALLOCATEs for single-packet messages has decreased lately since the IMP program has recently been modified in such a way that continued retransmission of single-packet messages is no longer possible.

A surprisingly large fraction of subnet control messages are INCTRANS which signal the source HOST that a message could not be delivered correctly to its destination HOST. The data of Table II indicate that, on the average, every hundredth message which enters the ARPANET will not reach its destination. The reason for this undesirable behavior is that many destination HOSTs are tardy in accepting messages. A HOST is declared down by its IMP if a message waits for more than 30 sec on the HOST output queue (from IMP to HOST) to be accepted. When this occurs, an INCTRANS control message is returned for every message that is waiting on the HOST output queue. (Future messages which reach the destination IMP after the HOST has already been declared down will generate a "destination dead" control message.) The frequent occurrence of incomplete transmissions is therefore not due to a failure of the subnet but is a result of the unresponsiveness of some of the attached HOST computers.[1]

Compared with the number of incomplete transmissions, the number of destination dead control messages is rather small. These DESTDEAD messages appear to be generated mainly in cases where one HOST wants to find out which other HOSTs are currently responding to net traffic, and thereby send a "probe" message to dead HOSTS.

4. HOST-to-HOST Control Messages

The packet tracing mechanism allows one to distinguish between HOST-to-HOST control packets and data packets (as well as subnet control packets). From the examination of several thousand samples it was determined that some 41 percent of all HOST-to-HOST packets that traverse the network are NCP control commands. Here, we examine the frequency with which each command type is sent, in order to determine what might be done to reduce this type of overhead.

In Table III we list the control commands, their length, and a short description together with bounds on their frequency of transmission. (For a more detailed description of NCP control commands, see [12].) The frequency of the HOST-to-HOST control commands was derived from the length of the corresponding control messages as observed in the subnet. Since this method does not allow us to uniquely identify each control command, we can present only an upper and lower bound for the frequency of their occurrence.

Most striking among the bounds in the table is the high frequency of the ALL (allocate) command. This phenomenon was reported, for instance, for the HARVARD-10 HOST in [22]. We now see that this is a network-wide characteristic. Let us consider the impact of this phenomenon.

The using up of network bandwidth has little effect since at the present time there is plenty to spare [11] (see Section 6). In Section 5 we show the effects of allocation size on available capacity. If a user message is required to wait in the sending HOST until an ALL arrives from the receiving HOST, the effect would surely be noticeable. This may in fact be the case, and as such would contribute to some excessive delay as seen by users but would not be attributable to network delay alone. The fact that there are only about 2 data messages per ALL command indicates that the allocations contained therein are very small. Note that the allocation size is a variable which depends on the NCP implementation.

Another consideration (possibly more important than the wasted network bandwidth and the extra waiting time for ALL control messages) is what portion of the HOST I/O and CPU bandwidth is spent in sending these overhead messages. The fewer messages sent and received by the NCP the smaller is the degradation to overall HOST performance. We noticed that even though the HARVARD-10 [22] sent a significant number of ALL type commands, its data buffer utilization

[1] For example, the "software halts" of the TENEX operating system, which are unusual conditions from which the system cannot recover without operator intervention, are a major source of HOST unresponsiveness [14].

Table III. HOST/HOST Control Commands.

Name	Length in bits	% of total	Function
RTS	80	3.2–7.5	Sent from *r*eceiving HOST *t*o *s*ending HOST to set up a connection
STR	80		Sent from *s*ending HOST *t*o *r*eceiving HOST to set a connection
CLS	72	3.2–7.5	Exchanged between receiving HOST and sending HOST to *close* a connection
ALL	64	63.8–79.0	Sent from receiving HOST to sending HOST to signal the *allocation* of message and bit space. The sending HOST is restricted from sending more messages or bits than have been allocated to him by the receiving HOST
GVB	32	0–10.0	Sent from receiving HOST as a request that the sending HOST *give back* all or part of its current allocation
RET	64	0–10.0	Sent from sending HOST to receiving HOST to *return* all or part of its allocation (response to give back)
INR	16	1.2–7.3	*Interrupt* command sent from the *r*eceiving HOST to the sending HOST
INS	16		*Interrupt* command sent from the *s*ending HOST to the receiving HOST
ECO	16		*Echo* command to determine if some other HOST is ready for a network conversation
ERP	16		*Echo reply* command returns data from the echo command to its sender
RST	8		*Reset* command for the reinitialization of NCP tables
RRP	8		*Reset reply* command (response to reset command)
ERR	max 96		*Error* command
NOP	8		*No operation*

was in the range of 3 to 4 percent, indicating that there is some excess capacity to store more data per connection. So it would appear that the HARVARD-10 in particular could send ALLs containing much larger allocations and thus send many fewer control messages. Perhaps not all the HOSTs have more storage to allocate to the NCP input buffers, but it would be well to examine these considerations in detail for each HOST.

Such an examination (i.e. the effect of allocation size) is often impossible without the aid of an instrumented NCP. In case of the ARPANET, there were several decisions to be made by NCP implementers. Among these were number of buffers and buffer size per connection, fixed or dynamically allocated buffers, maximum number of allowable connections, etc. In each case decisions such as these must be tested to ascertain their validity or to suggest improvements.

In conclusion, we note that most NCP control commands sent are of the ALL type. Therefore if one wants to reduce the overhead due to HOST-to-HOST control messages the most effective first step (within the current protocol) is to reduce the number of ALL type messages. Therefore, for those HOSTs which can afford to use larger and/or more buffers the answer is simply to send larger allocations!

5. Calculated Line Overhead

Let us now derive a simple formula for the line efficiency as a function of the traffic characteristics. The ARPANET HOST-to-HOST protocol provides for a connection-oriented message exchange. Thus, whenever one process decides to send data to another process, a connection must first be set up between these two processes (by means of the HOST-to-HOST control messages RTS and STR). When all the data has been sent, the connection is closed (by means of two CLS control messages). Furthermore, data can be transmitted only after storage has been allocated by the receiver by means of an ALL HOST-to-HOST control message.

Let N be a random variable representing the total number of bits that are to be transmitted and let A (also a random variable among HOSTs) be the number of bits that is allocated per ALL control message by the receiving HOST. Then the number of ALL control messages which must be sent from the receiver to the sender is[2]

$$a = \lceil N/A \rceil. \tag{1}$$

Define X to be the random number of data bits in a data message. Note that X must be less than or equal to: (a) N, the total number of transmitted bits, and (b) A, the number of allocated bits, and (c) 8023, the maximum number of data bits per message. Define

[2] $\lceil s \rceil$ is the usual ceiling function and is equal to the smallest integer greater than or equal to s.

Table IV. Line Overhead per Connection (in bits).

Message Type	No. Messages	Overhead per Message: Level-0 Overhead	Level-1 Overhead	Level-2 Overhead	RFNM Overhead	Total Overhead per Message
RTS	1	104	80	40 + 80	168	472
STR	1	104	80	40 + 80	168	472
ALL	$\bar{a}$	104	80	40 + 64	168	456
Data	$\bar{m}$	104Y	80Y	40	168	184Y + 208
CLS	2	104	80	40 + 72	168	464

Y to be the number of packets per message; we have

$$Y = \lceil (X + 40)/1008 \rceil. \quad (2)$$

(1008 is the maximum number of bits per packet in the ARPANET). Define m to be the total number of messages to be transmitted; we have

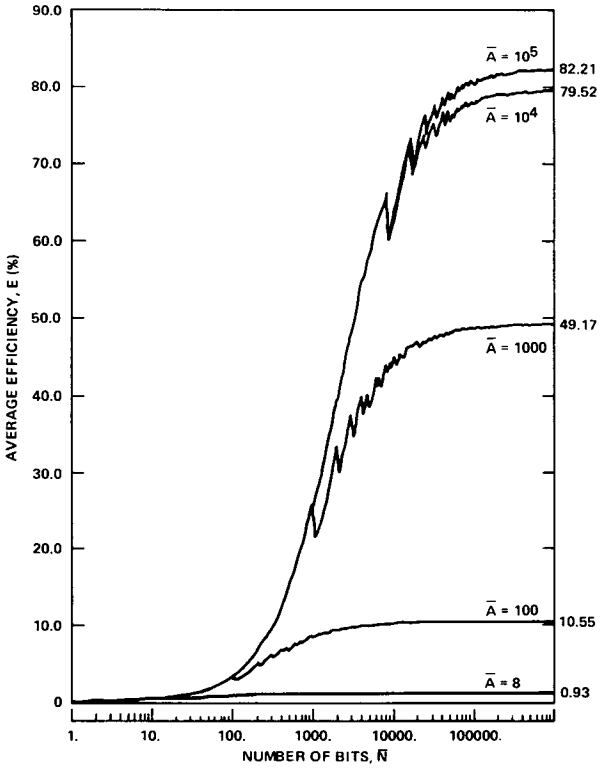
$$m = \lceil N/X \rceil. \quad (3)$$

We denote the mathematical expectation $E[\ ]$ by an overbar, thus defining $\bar{N}$, $\bar{A}$, $\bar{a}$, $\bar{Y}$, $\bar{X}$, and $\bar{m}$.

Note that we have ignored all the overhead bits used for message padding. The number of padding bits depends on the word length of the HOST computer. The exclusion of padding has only a small effect on our computations below; as a consequence, our results may be viewed as being slightly optimistic.

Table IV summarizes the line overhead involved in opening and closing a single connection and sending ALL control messages and data messages.

Fig. 2. Average line efficiency as a function of $\bar{N}$.



We assume that no HOST-to-HOST control messages are piggy-backed together, as for example sending the first ALL together with the RTS which is done by several HOSTs. Our measurement data shows that over 80 percent of all HOST-to-HOST control messages contain only one control command. If HOSTs maximize their message lengths (an assumption we shall make), we have

$$X = \min(N, A, 8023). \quad (4)$$

We define the average line efficiency E as the ratio of the total average number of data bits to the total average number of data plus overhead bits. Assuming that all the connections in the ARPANET can be described by the two variables $\bar{N}$ and $\bar{A}$, we make the following simple definition for the average line efficiency (see Table IV):

$$E = \bar{N}/[\bar{N} + 456\bar{a} + (184\bar{Y} + 208)\bar{m} + 1872].$$

Here, $456\bar{a}$ is the line overhead due to ALL commands, $(184\,\bar{Y} + 208)\bar{m}$ is the line overhead due to the overhead characters in data messages, and 1872 bits is the line overhead due to the opening and closing of a connection (one RTS one STR and two CLSs). Figure 2 shows the line efficiency E as a function of $\bar{N}$ for selected values of $\bar{A}$. The discontinuities in the curves are caused by message and packet boundaries. For $\bar{A} \geq \bar{N}$ only one ALL control message is necessary. Therefore the line efficiency is independent of $\bar{A}$ in this case. Note the low line efficiency for small values of $\bar{N}$. The line efficiency is only 0.32 percent if connections are used to transmit single characters ($\bar{N} = 8$). Even for large values of $\bar{N}$ the line efficiency is very low if the allocation size $\bar{A}$ is small. This shows what a drastic effect an NCP controlled parameter (A) can have on the efficiency of the communications subnet. A buffer shortage in the HOST computers can therefore directly lead to a decreased line utilization in the subnet. For the transfer of large quantities of data with a sufficiently large allocation size, the average line efficiency can be larger than 82 percent.

Since our definition of average line efficiency does not include the background traffic, we must subtract the average bandwidth for the background traffic from the given physical bandwidth before applying the cal-

Table V. Average Line Efficiency E_I (in Percentage).

$\bar{X}_{BITS}$ \ $\bar{A}_{BITS}$	8	100	1000	∞
8	0.93	1.83	1.98	2.00
40		6.51	8.88	9.26
100		10.55	18.60	20.33
200			29.27	33.78
500			44.64	56.05
1000			49.21	63.45
2000				72.46
5000				81.59
8023				82.69

culated percentages. In the ARPANET, the background traffic is 2.16 KBPS. The best possible bandwidth for process-to-process communication is therefore 82.69 percent of 47.84 KBPS or 39.56 KBPS. This corresponds to a 79.12-percent utilization of the 50 KBPS lines.

Let us now consider the case where connections are used for a long interactive use of a HOST computer. In this case the overhead for opening and closing connections can be neglected. The line efficiency is not determined by N, the total number of transmitted bits, but by the average size $\bar{X}$ of each interactive message. The formula for the average line efficiency can now be simplified:

$$E_I = 1/[1 + 456/\bar{A} + (184\bar{Y} + 208)/\bar{X}].$$

Table V shows the average line efficiency, E_I, for interactive use as a function of $\bar{X}$ and $\bar{A}$. Part of this table is empty since the average message size $\bar{X}$ can never be larger than the average allocation size $\bar{A}$. We again notice the decreased line efficiency for small values of $\bar{A}$. However, even for $\bar{A} = \infty$ the line efficiency is only 2.00 percent if the messages are sent one character at a time.

We expect the line efficiency of the ARPANET as it is now being used to be rather low for the following two reasons: (1) the character-at-a-time mode of communication with the TENEX systems, which represents a significant part of the total traffic, decreases the average number of data bits $\bar{X}$, and (2) the small buffer space in the TIPs decreases the average number of allocated bits $\bar{A}$. Section 6 presents measurement results on the line efficiency in the ARPANET.

6. Measured and Projected Line Overhead

As we have seen, the line efficiency in the ARPANET lies somewhere in the wide range between less than 1 percent and almost 79 percent. Let us now turn to measurement results that will allow us to calculate the current line efficiency. These results refer to the ARPANET as of May 1974 with 46 IMPs and 51 full duplex channels. To simplify matters, we make the following additional assumptions: (a) all lines have the same speed (50 KBPS); (b) all IMPs and lines are up; (c) the overhead for status reports can be equally allocated to all lines.

Table VI gives a breakdown of all the bits transmitted per second in the ARPANET according to the line overhead classification of Table I. These numbers represent an average over all 102 simplex lines. The contributions of the background traffic to the total traffic can be directly derived from Table I. For the status reports we assumed that they are, on the average, sent over 6.25 hops before they reach the Network Control Center (this number was computed for the topology of the ARPANET in May 1974). The average number of packets per second per channel was measured to be 4.27 pkt/sec (excluding status reports). From this we easily derive the level-0 line overhead. The fact that 49.15 percent of all transmitted packets represents subnet control messages allows us to determine the level-1 line overhead. The average number of bits per second per channel, excluding level-0 and level-1 line overhead and background traffic, was measured to be 454.28 bits/sec. 87.02 percent of all packets are the first packet of a message and therefore carry the additional 40 bits of HOST-to-HOST protocol overhead. As previously stated, 41 percent of all packets exchanged between HOSTs are HOST-to-HOST control messages with an average length of 93.5 bits (excluding the 40 bits of HOST-to-HOST overhead). These numbers allow us to determine the level-2 line overhead and from this we can determine the number of data bits exchanged between processes.

As can be seen from Table VI, about 64 percent of the traffic currently being carried by the ARPANET is background traffic. A large percentage of the background traffic is due to routing messages. The number of data bits per second is only about one half of one percent of the line capacity. The line utilization including all types of overhead is 6.73 percent. (As mentioned before, this does not include the extra routing messages that are sent when the line utilization is low.)

Because of the low line utilization, some of these numbers might be misleading. Therefore let us try to assess the effect of increasing the load on the subnet. While the background traffic is held constant, we will assume that the level-0, level-1, and level-2 line overhead as well as the data bits are increased proportionally until the line utilization is 100 percent. In this way we obtain an estimate for the overhead characteristics in a saturated net, *the traffic characteristics being unchanged.* The result of this traffic projection is displayed in Table VII.

It is interesting to note that about 35 percent of all transmitted characters are now due to IMP-to-IMP (level-0) transmission control. The best line efficiency (i.e. percentage of data bits) one can hope to achieve is about 20 percent (a conservative estimate of the 23.44 percent shown), because the delay increases indefinitely as the net saturates. This, of course, is an average

number. In particular cases one may get a far better line utilization. However, if the overall traffic characteristics remain constant, not more than roughly 10 of the 50 KBPS will, on the average, be available for process-to-process communication. Note also that the background traffic, which currently represents more than 64 percent of all the traffic, becomes almost negligible as the net saturates.

7. Evaluation of the Internetwork Protocol

Recently a new HOST-to-HOST protocol has been described by Cerf and Kahn [2]; we will call it the "internetwork protocol." A detailed specification of this protocol was worked out by Cerf, Dalal, and Sunshine [3]. The internetwork protocol tries to eliminate some of the difficulties which have been experienced with use of the ARPANET HOST-to-HOST protocol. It also provides for the exchange of messages between dissimilar computer communication networks. In this section we summarize the salient features of the internetwork protocol, and then evaluate its overhead characteristics and compare these with those for the ARPANET.

We wish to single out and discuss the following five stated features of the internetwork protocol: (a) no connection setup, (b) full-duplex connections, (c) flow control, (d) addressing structure, and (e) internetwork communication.

(a) Under the internetwork protocol it is not necessary to exchange special control commands before the sending of data messages can take place. The first message is simply marked with a bit in the message header which requests that a connection be set up. The receiving TCP (Transmission Control Program, equivalent to the NCP in the ARPANET) is, of course, free to reject this request by returning a small "negative" control message.

(b) In contrast to the ARPA HOST-to-HOST protocol, the internetwork protocol provides for full-duplex connections (called associations in [2]). Processes which communicate over these connections use unbounded but finite length messages. These messages can be subdivided into packets whose minimum size is equal to the message header.

(c) The internetwork protocol uses a flow control scheme which is tightly coupled with the reassembly, sequencing, and retransmission of messages and the duplicate detection. In the case of the ARPANET, most of these functions are provided in the communications subnet. Thus, the internetwork protocol was designed for less sophisticated subnet control procedures. This flow control and acknowledgment scheme is similar to that used by the French CYCLADES system [4, 18].

(d) The internetwork protocol provides for a richer addressing structure. The 8-bit HOST addresses of the ARPANET are replaced by 16-bit TCP addresses. It is possible that several TCPs reside in the same HOST (e.g. in several virtual machines). The 12-bit message identification of the ARPANET is replaced by a 24-bit process/port identification.

(e) The internetwork protocol was designed to allow an exchange of messages between networks of different characteristics. To deal with the problem of having different maximum packet lengths in different networks, the notions of a gateway, message splitting, and packet splitting were introduced [2]. The message header provides for a 16-bit source/destination network address.

Figure 3 shows the format of the header for the

Table VI. Line Overhead in the ARPANET (May 1974).

Category	Name	Line bandwidth Bits/sec	Sum	% of line capacity %	Sum
Level-0	SYN	68.32	444.08	0.14	0.89
	STX/ETX	136.64		0.27	
	H-checksum	102.48		0.20	
	ACK-header	68.32		0.14	
	S-checksum	68.32		0.14	
Level-1	Packet header	173.60	308.00	0.35	0.62
	Subnet control	134.40		0.27	
Level-2	HOST/HOST protocol	75.53	158.72	0.15	0.32
	HOST/HOST control	83.19		0.17	
Background	Routing messages	1812.50	2160.96	3.63	4.32
	IHY	237.50		0.48	
	Status reports	110.96		0.22	
Data	(Nonoverhead bits)	295.56	295.56	0.59	0.59
	Total Sum:	3367.32			6.73

Table VII. Projected Line Overhead.

Category	Name	Line bandwidth Bits/sec	Sum	% of line capacity %	Sum
Level-0	SYN	2709.28	17610.30	5.42	35.22
	STX/ETX	5418.55		10.84	
	H-checksum	4063.92		8.13	
	ACK-header	2709.28		5.42	
	S-checksum	2709.28		5.42	
Level-1	Packet header	6884.23	12213.95	13.77	24.43
	Subnet control	5329.72		10.66	
Level-2	HOST/HOST Protocol	2995.19	6294.15	5.99	12.59
	HOST/HOST Control	3298.96		6.60	
Background	Routing messages	1812.50	2160.96	3.63	4.32
	IHY	237.50		0.48	
	Status reports	110.96		0.22	
Data	(Nonoverhead bits)	11720.64	11720.64	23.44	23.44
Total Sum:		50000.00			100.00

Fig. 3. Suggested header for the internetwork protocol.

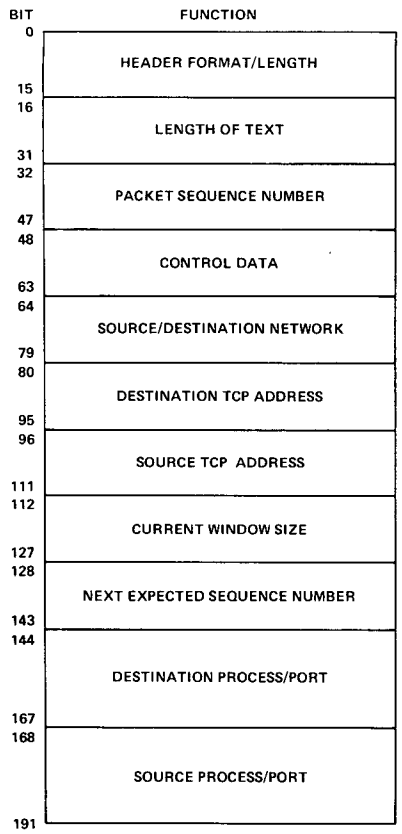
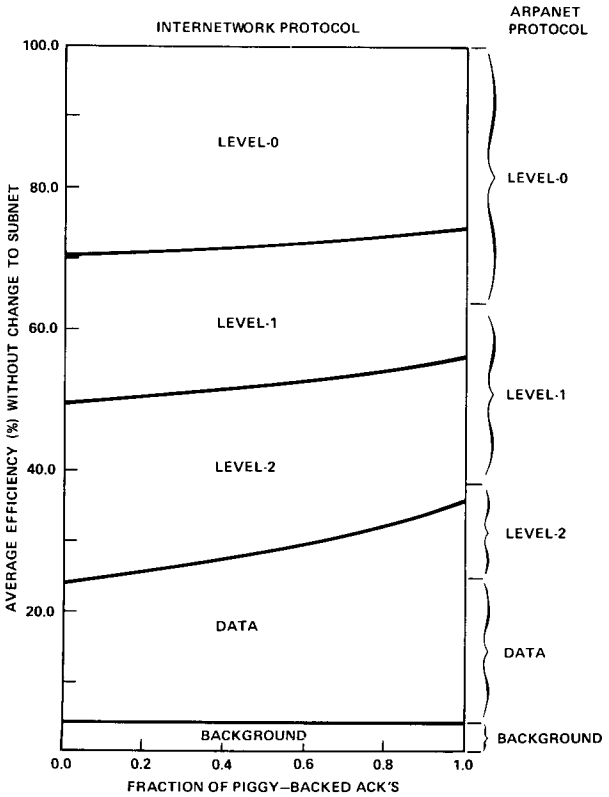


Fig. 4. Projected overhead characteristics of the internetwork protocol without change to subnet.



internetwork protocol as defined in [3]. The function of most of the fields in this header has already been mentioned or is self-explanatory. Bits 32 through 47 ("packet sequence number") and 112 through 143 ("current window size" and "next expected sequence number") contain information relating to HOST-to-HOST acknowledgment and flow control. The "control data" field is used for such purposes as opening, closing, and resetting of connections, process interruption, echoing, error indications, etc. The entire header is 192 bits long. We assume that these 192 bits replace the 40-bit HOST-to-HOST protocol overhead described in Section 2. We will, however, neglect the fact that this replacement increases the average number of packets per message. (Histograms of message length show that this increase is only about 1 percent.)

Let us now evaluate the effect that a replacement of the current ARPA HOST-to-HOST protocol by the internetwork protocol would have on the ARPANET line efficiency. Since the internetwork protocol incorporates some of the functions which are currently provided by the communications subnet, we will also consider the case where these functions have been removed from the subnet control procedures.

The internetwork protocol does not define separate control messages for HOST-to-HOST acknowledgment and flow control. Since full-duplex connections are used, this data can be piggy-backed on the traffic in the reverse direction. When, however, there is no data waiting to be sent in the reverse direction, then a special acknowledgment message must be transmitted which carries only control information. Thus we expect the overhead characteristics of the internetwork protocol to be a function of the fraction of piggy-backed acknowledgments. Only measurements will show how large this fraction is for different types of network use.

Figure 4 shows the extrapolated overhead characteristics of the internetwork protocol without changes to the subnet as a function of the fraction of piggy-backed acknowledgments. In order to compare these curves with the current ARPANET system, the values of Table VII are also indicated in Figure 4. The percentage of data traffic ranges between 19.7 percent (no piggy-backed acknowledgments) and 31.7 percent (all acknowledgments are piggy-backed). At least 42 percent of the acknowledgments must be piggy-backed to achieve a better average line efficiency than the current HOST-to-HOST protocol does. Note the large percentage of level-2 line overhead for the internetwork protocol which is due to the long header (compared with the current HOST-to-HOST protocol header).

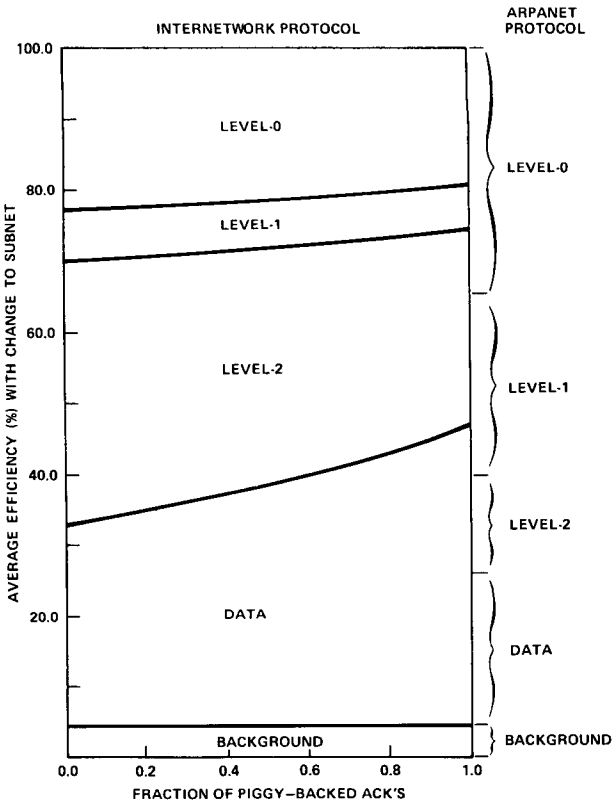
Let us now consider the possible changes to the ARPA communications subnet if the current ARPANET HOST-to-HOST protocol were replaced by the internetwork protocol. (It remains to be seen whether or not these changes are feasible in an operating environment.) Since the internetwork protocol provides for message acknowledgment on the HOST level of the

communications hierarchy, it is no longer necessary for the subnet to send RFNMs to the source HOSTs. If we now assume that flow control in the subnet does not depend on these RFNMs as, for instance, in the NPL network [6, 19], then the sending of RFMNs from the destination IMP to the source IMP becomes superfluous. Since message reassembly is done on the HOST level, the allocation of storage at the destination IMP becomes less critical. We will therefore also assume that no subnet control messages for buffer allocation need to be exchanged. However, "destination-dead" and "incomplete-transmission" control messages are still useful. Because of their low frequency they will be neglected. If there is no message control in the subnet, the size of the packet header can be decreased substantially. We will assume the current 80-bit packet header can be replaced by a 32-bit packet header.

Figure 5 shows the extrapolated overhead characteristics of the internetwork protocol with the above mentioned changes to the subnet. Note the large reduction of level-1 overhead and (to some extent) level-0 overhead. With the modified subnet the percentage of data traffic ranges between 28.6 percent (no piggy-backed acknowledgments) and 42.7 percent (all acknowledgments are piggy-backed). This means that in this case the internetwork protocol is superior, in terms of line efficiency, to the current HOST-to-HOST protocol (with only 23.4 percent of data traffic) for the observed traffic characteristics, independently of the fraction of piggy-backed acknowledgments.

We have evaluated only the line overhead characteristics of this proposed protocol under the current traffic pattern in the ARPANET. It must be stressed that line overhead is not the only consideration one should make in evaluating the operating characteristics of a protocol.

Fig. 5. Projected overhead characteristics of the internetwork protocol with change to subnet.



8. Conclusions

It has been argued that there should be no difference between process communication within a HOST and process communication over a network [15]. From a logical point of view this may be the right approach. As far as the efficient use of resources is concerned, such an approach may have disastrous results. In this sense the network is not transparent to interprocess communication. Rather the HOSTs must be aware of the fact that the allocation of network resources requires the same care as the allocation of any other resource. It appears that in some cases the freedom which the ARPANET protocols provide its implementers has been misused. In order to reduce the overhead, much more thought must be spent on the *efficient* implementation and use of net work protocols, rather than only on their feasibility.

Although the overhead can be decreased, the designers of computer networks must realize that a significant percentage of the line utilization will always be needed for control information. The exact amount of the overhead depends critically on the type of traffic (or traffic mix) the network is intended to carry. Only a careful study will reveal what part of the physical bandwidth is actually available for user-process to user-process communication. For the ARPANET we have shown that the average line efficiency can be as low as 1 percent for single-character traffic and as high as 79 percent for efficient file transfers. Assuming that the traffic characteristics remain unchanged, we were also able to show that the ARPANET is able to support little more than 10 KBPS user-to-user traffic on its 50 KBPS lines. Inefficiencies such as those discussed in this paper have already prompted a reexamination of some of the higher level protocols [17]. In view of these results we hope that in the future the design, implementation, and use of communication protocols take more account of the effect of overhead on the user-to-user throughput and thereby improve the network performance.

Acknowledgments. We gratefully acknowledge the contributions of the staff of the UCLA Network Measurement Center and the referees' constructive comments.

Received November 1974; revised February 1975

References

1. Carr, C.S., Crocker, S.D., and Cerf, V.G. HOST-HOST communication protocol in the ARPA network. Proc. AFIPS 1970 SJCC, Vol. 36, AFIPS Press, Montvale, N.J., pp. 589–597.
2. Cerf, V.G., and Kahn, R.E. A protocol for packet network intercommunication. *IEEE Trans. on Commun. COM-22*, 5 (May 1974), 637–648.
3. Cerf, V.G., Dalal, Y., and Sunshine, C. Specification of internet transmission control program. Internat. Network Working Group Note #72 (IFIP TC6.1 WG6.1), Dec. 1974.
4. Chambon, J.F., Elie, M., Le Bihan, J., LeLann, G., and Zimmerman, H. Functional specification of transmission station in the CYCLADES network. ST-ST protocol (in French). IRIA Tech. Rep., May 1973.
5. Crocker, S.D., Heafner, J.F., Metcalfe, R.M., and Postel, J.B. Function-oriented protocols for the ARPA computer network. Proc. AFIPS 1972 SJCC, Vol. 40, AFIPS Press, Montvale, N.J., pp. 271–279.
6. Davies, D.W. The control of congestion in packet switching networks. *IEEE Trans. on Commun. COM-20*, 3 (June 1972), 546–550.
7. File Transfer Protocol. NIC #17759, Network Information Center, Stanford Research Institute, Menlo Park, Calif., 1973.
8. Frank, H., Frisch, I.T., and Chou, W. Topological considerations in the design of the ARPA computer network. Proc. AFIPS 1970 SJCC, Vol. 36, AFIPS Press, Montvale, N.J., pp. 581–587.
9. Heart, F.E., Kahn, R.E. Ornstein, S.M., Crowther, W.R. and Walden, D.C. The interface message processor for the ARPA computer network. Proc. AFIPS 1970 SJCC, Vol. 36, AFIPS Press, Montvale, N.J., pp. 551–567.
10. Kleinrock, L. Analytic and simulation methods in computer network design. Proc. AFIPS 1970 SJCC, Vol. 36, AFIPS Press, Montvale, N.J., pp. 569–579.
11. Kleinrock, L., and Naylor, W.E. On measured behavior of the ARPA network. Proc. AFIPS 1974 NCC, Vol. 43, AFIPS Press, Montvale, N.J., pp. 767–780.
12. McKenzie, A.A. HOST/HOST protocol for the ARPA network. NIH #8246, Network Information Center, Stanford Research Institute, Menlo Park, Calif., Jan. 1972.
13. McQuillan, J.M., Crowther, W.R., Cosell, B.P., Walden, D.C., and Heart, F.E. Improvements in the design and performance of the ARPA network. Proc. AFIPS 1972 FJCC, Vol. 41, AFIPS Press, Montvale, N.J., pp. 741–754.
14. McQuillan, J.M. (personal communication, Oct. 1974).
15. Metcalfe, R.M. Strategies for operating systems in computer networks. Proc ACM Annual Conf., Aug. 1972, pp. 278–281.
16. Opderbeck, H. and Kleinrock, L. The influence of control procedures on the performance of packet-switched networks. National Telecommunications Conf., San Diego, Calif., Dec. 1974.
17. Postel, J. Announcing additional Telnet options. NWG/RFC #659, NIC #31177, Oct. 1974.
18. Pouzin, L. Presentation and major design aspects of the CYCLADES computer network. Proc. Third Data Communications Symp. St. Petersburg, Fla., Nov. 1973, pp. 80–87.
19. Price, W.L. Simulation of packet-switching networks controlled on isarithmic principles. Proc. Third Data Communications Symposium, St. Petersburg, Fla., Nov. 1973, pp. 44–49.
20. Roberts, L.G., and Wessler, B.D. Computer network development to achieve resource sharing. Proc. AFIPS 1970 SJCC, Vol. 36, AFIPS Press, Montvale, N.J., pp. 543–549.
21. Specifications for the interconnection of a HOST and an IMP. Rep. No. 1822 (rev.), Bolt, Beranek and Newman, Inc., Cambridge, Mass., Mar. 1974.
22. Taft, Edward. A few observations on NCP statistics. NWG/RFC #613, NIC #21989, Feb. 1974.
23. TELNET protocol specifications. NIC #15372, Network Information Center, Stanford Research Institute, Menlo Park, Calif. 1972.

Computer Systems

C. Bell, D. Sieworek and S.H. Fuller, Editors

An Anomaly in Disk Scheduling: A Comparison of FCFS and SSTF Seek Scheduling Using an Empirical Model for Disk Accesses

Neil C. Wilhelm
University of Rochester

A model for disk accesses based on published measurements is developed. The model is used to show that under highly probable conditions, FCFS seek scheduling is superior to SSTF scheduling in the sense of having a lower mean queue length. A simple example of an arrival sequence illustrating this anomaly is presented.

Key Words and Phrases: disks, disk scheduling, seek scheduling

CR Categories: 3.72, 4.35, 4.41, 4.6, 6.35

Introduction

Considerable effort has been expended in the investigation and analysis of seek scheduling strategies for moving-head disks, because the maximum seek time may be an order of magnitude greater than the expected rotational delay. A number of scheduling techniques have been devised by Denning [1], Frank [2], and Teorey and Pinkerton [3], to reduce the effects of the seek time on disk system performance. A good summary of seek scheduling rules may be found in [4]. Some consideration has also been given to reducing latency time effects

Copyright © 1976, Association for Computing Machinery, Inc. General permission to republish, but not for profit, all or part of this material is granted provided that ACM's copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Association for Computing Machinery.

Author's address: Department of Electrical Engineering, University of Rochester, College of Engineering and Applied Science, River Campus Station, Rochester, NY 14627.