International Institute for Applied Systems Analysis
IIASA www.iiasa.ac.at

The Impact of the X.25 on the Efficiency of the Communications Subnetwork

Butrimenko, A. and Sichra, U.

IIASA Working Paper

WP-80-027

February 1980

THE IMPACT OF THE X.25 ON THE EFFICIENCY OF THE COMMUNICATIONS SUBNETWORK

A. Butrimenko, U. Sichra

February 1980
WP-80-27

ABSTRACT

The acceptance of the X.25 international standards requires a virtual call procedure, which imposes some restriction on the usage of communication resources, namely a fixed route for the whole duration of the session. In an example of network simulation, datagram and virtual call methods are compared, producing the result that the datagram gives a better performance with regard to delivery time. The difference between these two methods is greater when the load, the number of packets per message, and the size of the network is increased.

CONTENTS

THE IMPACT OF THE X.25 ON THE EFFICIENCY OF THE COMMUNICATIONS SUBNETWORK

A. Butrimenko, U. Sichra

INTRODUCTION

It is a well-known fact that possibly one of the most significant breakthroughs in computer communication has been achieved as a result of applying the so-called store-and-forward technique. This technique relies explicitly upon the possibility of treating the pieces of a message as separate entities, which can be delivered in an arbitrary order and within different time intervals. This method allows a more effective usage of communication resources than, for example, the channel switching technique. The three most important features of the store-and-forward technique, by which the efficiency in a communication network can be increased are:

(a) by queueing the packets at every channel along the route, instead of having only one queue at the source, as is the case in a circuit connection;

(b) by using the capacity of a trunk as a single high-speed channel versus multiplexing it to a number of slow-speed channels;

(c) by using various routes for different pieces of information, depending upon the actual loading of the system, and by assembling the whole message at the destination only.

VIRTUAL CALL - DATAGRAM

The recently accepted international standard X.25 [1], based on the store-and-forward technique, requires a specific procedure, the virtual call, which provides a logical channel between source and destination for the duration of the whole session.

The accepted standard X.25 requires, strictly speaking, only a virtual call interface because the DCE (Data Communication Equipment) provides a connection between a pair of DTE (Data Terminal Equipment) similar to an electrical connection. One of the features of this kind of connection is that all the packets sent by the DTE will arrive at the corresponding DTE in the same order in which they were sent.*

The transport method in the data communication network itself can be organized in such a way that all the packets inside the data communication network are considered to be independent entities and routed separately. In this case it will be said that in the communication network (DGE) the datagram method (DG) is used. Figure 1 shows a typical datagram format [2]. The datagram packet, in addition to its data field, also contains addresses, and actually complete addresses, of both the communicating DTEs, as well as some additional control information. In this case, however, relatively complex functions are needed at the nodes communicating to the DTE to provide all the necessary features required by the X.25. One of those features is, in particular, the reordering of the incoming packets belonging to the same call.

*This is a basic feature of the X.25 Recommendation, as described in [2].

Another method which allows this function to be performed in a relatively easy way is to extend the virtual call philosophy to the communication network, i.e., to achieve the goal that all packets of the same session will be sent along the same route, which makes reordering unnecessary. As is done, for example, in the Transpac and Datapac networks [3], this can be achieved by switching the logical channels. The virtual call procedure implies that, before the virtual connection is established, a special packet called "CALL REQUEST" is sent from the source station to the destination station. Actually the physical route passed by this packet will be followed by all other packets of the same session. The format of the "CALL REQUEST" is shown in Figure 2. The switching of the logical channels is carried out in the communication nodes with the help of special matrices, which makes it further possible to route all subsequent packets of the same session over the same physical channel.

It is clear that, as soon as the logical channel, i.e., the virtual call, is established between the stations, there is no need for the full address in the subsequent packets of the same session, and only the number of the corresponding channels should be carried by the packets (see Figure 3)[1]. This logical channel number is naturally significantly shorter than the complete address of DTEs.

Later references in the text to virtual call (VC), will mean the above described procedure of setting up a virtual call.

One should, however, stress that the switching of logical channels does not monopolize physical channels, and that the same physical channel can be used for a number of virtual calls simultaneously.

Point (c) mentioned at the beginning of this paper is, however, no longer the source of increasing the efficiency, as soon as the route is fixed for the whole duration of the session. Namely the fixed route leads to some kind of "bursty" traffic on the same channels, and routes can no longer be optimized by the routing mechanism, as soon as a call has been established.

Table 1 indicates the differences between datagram and virtual call techniques with regard to the use of communication resources. There are obviously other significant differences that lead to the acceptance of the X.25 standard in general, but these problems are not considered here. This paper will concentrate only on the usage of communication resources. The aim is to estimate the price to be paid for a permanent route in a virtual call mode in terms of a possible decline in the efficient usage of communication resources.

THE SIMULATION MODEL

The objective of this paper is to study the delivery times of packets and calls for different types of routing (i.e., adaptive routing for every packet - DG, and adaptive routing only for the call - VC). For this purpose a very simple communication network was modelled, leaving out many real-life factors which do not influence significantly the variables measured. Besides this, quite a few assumptions had to be made which might be found in an actually functioning communication network.

The main assumptions underlying the simulation and the reasons for making them are:

- All links between nodes have the same speed. Reason: to study only the effects of routing and queueing, under the most homogeneous conditions;

- The time needed to handle a packet (choose a queue, accept the packet, finish a call) is equal to ϕ, as well as the time required for reassembling of the message. This assumption is true for networks with high-speed hosts and switching nodes and low or medium speed trunks;

- The links are error free. Reason: errors would influence both methods in a similar way; simplification and homogenization of the model;

- Within one routing method (VC or DG) all packets have the same length. Reason: this situation can be found in some packet-switching networks;

- All calls have the same number of packets. Reason: simplification of the simulation, and similar effects on both methods, if the number was different;

- The creation and acceptance frequency is fixed for every node and does not change along the simulation. Reason: simplification;

- Poisson arrivals of packet. Reason: any other distribution will increase "burstness" and therefore the difference between methods;

- The queuelength in each channel is not limited; the sum of all queuelengths is, however, finite, but large enough not to be of concern under the normal operation of the network. Reason: finite number of array elements in the simulation program that store the information.

There are many more simplifications but they do not seem to be as restrictive as the above and have therefore not been mentioned.

The flow chart in Figure 4 shows very roughly the sequence of events in the simulation program. A few events need some more exploration as they are rather important for the outcome of the comparison between virtual call and datagram.

Choose the Channel

When a packet arrives at a node it is checked to establish if that node is already the destination node. If not, the program selects the next channel that packet has to use in order to arrive at its destination. When simulating virtual calls, only the first packet of a call will get a "new" channel; all other packets of that call will travel along the route passed by the first packet. In the case of datagrams, all packets undergo the channel selection procedure. This procedure is closely related to the updating of the routing matrices, where a "relief method" has been used [4]. It works as follows.

The routing information is stored in each node in a matrix. The columns of the matrices correspond to the numbers of the outoing channels; the rows correspond to the node numbers. Each element r^m_{ij} of the matrix in node m is the estimated time a packet will take to reach the destination node i through channel j. The updating procedure is carried out with the help of a vector which has in each element the minimum of the corresponding row in the matrix, thus informing of the minimal delay from that node to each destination node. Each time there is a change in the queues of the outgoing channels of a node, it is checked to establish if the changes since the last update procedure are greater than a certain threshold, and if so, the vector is updated to meet the new situation. This new vector is then sent to all adjacent nodes where it is used to update the corresponding elements of the routing matrix. The vector in each adjacent node will probably also be updated (if the minimal elements have changed). This updating event always takes place if changes are great enough, and of course it can happen in every node. Therefore the whole net is constantly being updated to the new queues, and the routing is adaptive in that sense.

Creation of a New Call

Virtual Call

To establish a virtual call connection, two different methods could be used:

(a) The first packet of the message has a special calling function like a "call request". Until this packet has been delivered to its destination, no packet of the message will be generated and put into the network. The procedure for establishing a logical connection is actually more complex and requires a few shakehands between communicating DTEs. If the first packet is dropped, the message will merely require another attempt to establish a virtual call connection;

(b) The second possibility is that the generation process is not influenced by the establishing or non-establishing of a call. All subsequent packets just follow the first packet at specified creation rate intervals and on the same route. If the first packet is dropped, the generation process will be stopped and the call is considered as being lost. The rest of the packets will also be dropped - at the same point as the first one.

This second method is used in the simulation program in order to minimize delays in message delivery by the VC method.

Datagram

A datagram starts when the "event table" requires it, and at the same moment a packet is created. All other packets are produced at the times set by the generator, irrespective of the chance that a packet may be dropped. It is clear that, if there is no dropping, VC and DG have the same generation pattern.

As all packets of one datagram may travel along different routes, the drop-event only influences the "packets per call counter" and does not cause the dropping of the whole call, as in the VC connection.

It is conceivable that the packets of one call will arrive in disorder. The assumption then is that the reordering is done outside the communication network; thus the simulation does not account for the time taken by such actions.

Time of a Next Call or Packet

There is only <u>one</u> random number generator in the net, which outputs evenly distributed random numbers between Ø and 1. This generator is used on the one hand to choose the creation and destination node for every call. On the other hand, it produces negative exponentially distributed random numbers, i.e.:

$$F(x) = p(t < x) = \int_0^x \lambda e^{-\lambda t} \, dt = e^{-\lambda x} \quad ,$$

with $\frac{1}{\lambda}$ = mean inter-arrival time.

By generating random numbers y: $0 < y < 1$, and setting $y = e^{-\lambda x}$, one gets:

$$\ln y = -\lambda x \quad ,$$

or

$$- \frac{\ln y}{\lambda} = x \quad ,$$

and x is the time interval which lies between two events. In the simulation model, a mean packet inter-arrival time, and a mean call inter-arrival time are needed:

$$\mu_p = \frac{\text{shift}}{o_1}$$

$$\mu_c = \frac{\text{max}_p}{o_1}$$

o_1 = parameter which controls the overall load of the system

max_p = number of packets per call

shift = parameter to control the time interval between new packets in the system

The only random number generator used in the simulation outputs a variable r at request, and this is used to set the time for one of the following events:

1. create next call $\quad tim = -\ln r * \mu_c$

2. create new packet $\quad tim = -\ln r * \mu_p + zuber$

where zuber = length of the packet.

The creation time for a new packet is always increased by the length of the packet to avoid the possibility that two packets of the same call get created at shorter intervals than the time-span they need to arrive at the next node. In this way, it is not possible for congestion and delay to occur due to overlapping.

If shift is set equal to 1, there will be on the average only one call per time unit generating packets in the system, as the packet generation process with the mean interval time $\mu_p = \frac{1}{o_1}$ will stop after max_p packets have been generated, and after this time span a new call will be created:

$$\mu_c = \frac{max_p}{o_1} \; .$$

If shift is >1, the mean inter-arrival time of the packets is increased, whereas the call generator always produces at the same intervals:

$$\mu_c = \frac{max_p}{o_1} \; .$$

The consequence of this procedure is that there is on the average more than one packet generator in the net, and at the same time the overall packet creation for the whole net remains the same.

This feature can be used to avoid "burstness" at the nodes (because of frequent creation of packets when a call started), as this could be a source of delay in the virtual call procedure. An increase of the shift parameter will produce a much smoother generation process.

Confidence Interval

The simulation is carried on until the mean delay of the packets satisfies the condition:

$$P(|\bar{x} - \mu| < \varepsilon) = \alpha \quad ,$$

i.e., the probability that the measured and the actual mean delivery differ less than ε is α, α being 95% in the program, and ε being = 0.2. If the variables x_i, where $\sum_{i=1}^{N} x_i/N = \bar{x}$, are normally distributed with mean μ and variance σ^2, and the sample is big enough ($N > 30$), then from $P(|\bar{x} - \mu| < \varepsilon) = \alpha$, it follows that:

$$P\left(\frac{-\varepsilon\sqrt{N}}{s} < \frac{(\bar{x} - \mu)}{s}\sqrt{N} < \frac{\varepsilon\sqrt{N}}{s}\right) = \alpha \quad ,$$

and $T = \frac{(\bar{x} - \mu)\sqrt{N}}{s}$ is normally distributed with mean 0 and variance 1. $f(\frac{\varepsilon\sqrt{N}}{s})$ is the $1 - \alpha$ fractile of the (0,1) distribution and thus one can check if the new mean lies between the boundaries. This means: if $N < (\frac{z_{1-\alpha}}{\varepsilon})^2 \cdot s^2$, the simulation must go on, otherwise the accuracy is reached:

$$z_{1-\alpha} = 1 - \alpha \text{ fractile of } N(0,1) \quad ,$$

$$s^2 = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})^2 \quad .$$

Drop a Packet

As the routing procedure does not guarantee a loop-free route between origin and destination of a node, each packet carries a counter which is increased by one, after the packet has passed one link. If this counter reaches a maximum number, the packet is dropped. If the procedure simulated is a virtual call, the source node is informed about the dropping and does not generate more packets for that call. In the datagram mode, the source is informed that it should produce one more packet for that datagram, as one packet has been dropped. In a real network, time out parameters are used to find out that a packet has been lost. The above-mentioned procedure is used for the sake of simplicity of simulation.

SIMULATION RUNS

The main performance characteristic of this system is the delivery time of a message, i.e., of all packets generated during one session. It should, however, be noted that generation time and call duration time are not identical, as the call can only be closed when all the generated packets have arrived at the destination node. It must be stressed that the moment of generation of the next packet is counted as starting from the moment of creation of the previous packet (see Figure 5). If the inter-arrival intervals are allowed also to be shorter than the length of the packet, the packets of the same message could, therefore, overlap in time*. The delivery time of the message will, therefore, cover the period from the generation of the first packet of the message until the moment at which all packets of the same message have been delivered to their destination.

In order to attain the goal of this study, namely to find out the influence of the virtual call procedure on the usage of the communication resources, several runs were made. The most

*This possible source of delay, due to congestion at the origin, was accounted for later, as explained earlier.

important observed parameter was the delivery time of the message as a function of the overall network load, as well as the number of packets per call (message) and the generation time. Different lengths of packets were also considered. To analyze the observed differences for virtual call and datagram, some more detailed observations were required, i.e., to find out the influences of both the "burstness" of the traffic and the lack of routing adaptation in the virtual call handling. As a secondary goal, it was also intended to see the influence of both these methods on the amount of routing control information exchanged between communication nodes.

Two networks (Figures 6 and 7) of different size were simulated. Actual measurements were made for the delivery of each packet and message. Depending on the minimum distance between source and destination, all messages (and packets) were divided into classes in accordance with the minimal distance between source and destination. For example, in Figure 7, a message to be delivered from 1 to 4 belongs to the first class, and a message to be delivered from 2 to 12 belongs to the second class, independent of the actual number of transits that it will pass in the network. For every class of message, the delivery time of the first to the nth packet of the message was measured separately.

The first set of simulation runs was carried out in order to compare the delivery time of both methods with regard to the length of the messages and the number of packets per message. Table 2 shows that not only the delivery time for the virtual call is higher than the one for datagrams, but also that the delivery time for each packet of a virtual call is greater and increases with the length of the message. The load of the network was chosen in such a way that it remains constant, independent of the number of packets per call. Thus, Table 2 represents the results achieved by constant load, equal in this case to 6.6 packets per time unit, and equal packet length for both methods (virtual call and datagram).

Another group of runs was made for a number of loads, varying between 4.3 and 10 packets per time unit.

The most general results of these simulations are represented by two figures. Figure 8 shows the dependence of the delivery time on the load for both datagram and virtual call for three particular lengths of message, namely 3, 5 and 9 packets each. It is evident that, not only is the datagram faster, but also the difference in the delay between the datagram and the virtual call increases with the load. For all simulated loadings and lengths of messages, it is clear that the delivery time for a virtual call increases with its length, and the difference in the delay between these two methods also increases in accordance with the length of the message.

Figure 9 depicts the delivery time measured in time units depending on the message length, for both virtual call and datagram. The results of simulations with two different loadings are shown, and it can be seen that the relationship is more or less linear, although the rate of increase changes with the loading, as well as with virtual call handling, as compared to datagram handling.

The two figures, 9 and 10, demonstrate clearly that a datagram performs better in terms of one of the most important characteristics - delivery time, and furthermore, this is true for all lengths and loadings.

A more detailed analysis of the simulation results shows some interesting factors. The delivery time of every message consists of three components:

(a) Time length of the message, i.e., number of packets and time intervals between their creation points;

(b) Distance to be passed in the network;

(c) Delay caused by lining up every packet, routing it, eventually dropping a packet and rejecting it, plus regeneration of some packets when the network is loaded.

If one subtracts from the delivery time of a message, the time required to transmit it to the destination under ideal - unloaded network - conditions, one will get the delay caused by actions in point (c).

The ideal delivery time for virtual calls is characterized solely by the minimum distance between source and destination, and its length. The estimation of the delivery time is:

$$del_{vc} = pack_{vc} + cl - 1 \quad ,$$

because in the chosen VC creation process the packets can overlap. $pack_{vc}$ is the number of packets per message, and cl is the class (minimal distance) to which the call belongs.

The overall ideal delivery time in the VC mode then depends on the ratio of possible connections and on the ratio of messages generated for the different classes.

In the datagram delivery system, the ideal delay is dependent upon the topology. A "rough" calculation of the ideal delivery time, biased towards the least number of outgoing channels of the net in Figure 6, was made to draw the simulation results of Figure 10. The number of packets per message is plotted on the x-axis; the difference between actual and ideal delivery time is shown on the y-axis. Each pair of curves is for a specific class (1 or 2) and a fixed loading.

In class 1, virtual calls suffer less delay caused by the actions of point (c), i.e., routing, queueing, etc., than datagrams, but the delivery time is still better for datagrams than for virtual calls. In class 2, the difference of real and ideal delay is greater in virtual calls than in datagrams and increases when the net is more loaded.

As pointed out previously, the routing of the first packet in a virtual call connection is performed in the same way as the routing of all packets in the datagram transmission. Each node exchanges information with its neighbouring nodes about the situation of its queues. All nodes in the net therefore have complete, although delayed, information about the overall situation. When messages are transmitted through virtual calls, less information is exchanged between nodes than in the datagram treatment (see Table 3). This means that greater overload of control information is created in the datagram connections. In the simulation, this overload has no influence on the delivery time as the "update packets" are assumed to be transmitted along very fast channels different from those for the normal messages.

A short analysis of some simulation runs for the net in Figure 7 shows results similar to those for the small net (see Figure 6). In Figure 11 it can be seen that the delivery time of messages increases as the load increases and that it is higher in virtual calls than in datagrams. As the number of links to be passed increases, the delivery time rises as well, and in class 4 (minimal distance is 4 links) the difference becomes rather great.

One could object to the results of these simulation runs by argueing that it is highly unrealistic that the packets in the virtual call transmission have the same length as the packets in the datagram handling. The datagram mode allows different routes for different packets and thus more information about the source and destination node must be carried in the datagram-packet.

The case was thus looked into where packets in the datagram mode are 10% longer than the packets in the virtual call mode (within one mode all packets still have the same length). The results of a few simulation runs are also plotted in Figure 11. As one would expect, the delivery times of the larger packets in the DG mode are higher than the delivery times of the shorter ones, but in the moderate load range, these packets are still delivered earlier than the packets in the VC mode. Only in the overloaded situation do the packets in the DG transmission take longer than in the VC transmission.

The mean delivery time of packets and calls is the main parameter used up to now to measure the performance of the two types of data handling in the communication networks: virtual call and datagram. But obviously the overall mean delivery time of packets does not tell very much about the more detailed behaviour of the packets. Therefore one should look at different types of packets and also make separate measurements for them.

As has already been mentioned earlier, packets are subdivided into classes, depending on the minimum number of links they have to pass before arriving at their destination.

One can also divide the packets into groups, depending on their position within a call, i.e., first packets, second packets....last packets (in these simulation runs, fifth packets). It is, of course, also possible to measure the delivery times of the n-th packets in the j-th class.

If one looks then at the delivery times of the first, second....n-th packets in a specific transmission mode, one can see that the first packets are delivered faster than the last packets, in both VC and DG (Table 4, first column). One would expect this result in VC transmissions, as all packets of one call take the same route and might thus influence the behaviour of the subsequent ones. In the DG transmission, this is not obvious as packets of the same session might take different routes and not block each other from being sent in the shortest time.

The source of greater delay for the last packets as compared to the first packets is thus probably not the routing mechanism alone, but also the "burstness" of the creation process. As mentioned in the description of the simulation program, the variable "shift" influences the inter-arrival time of packets of one session. If the "shift" is increased, this inter-arrival time is increased as well, although the overall creation rate in the net - and thus the loading - remains the same.

In Table 4, the mean delivery times for 4 different mean packet creation intervals, from 1 to 9 are listed; this corresponds to 4 values of the shift (from 1 to 104). It can clearly be observed that in the DG mode the differences in the delays of the first and the last packets get smaller as the shift gets larger. This clearly shows that a good part of the delay of the packets is due to the "burstness" character of the creation process and that by increasing "shift", this source of delay can more or less be eliminated. In Figure 12, the curves of these differences are displayed. Nevertheless, there still remains a clear difference between the delivery times of the first and last packets in the different classes, when the packet handling is done in the virtual call mode. This increased delay thus stems from the poor adaptability of the route in the VC mode, as the optimal route will only be taken for the first packets, and not necessarily for the others.

The delivery time of the whole call, also listed in Table 4, is dependent, among other things, upon the delivery time of the last packet. The DG mode delivers the calls in a slightly shorter time than the VC mode, in all cases, although the datagrams are 10% longer than the virtual calls, and the network is thus more heavily loaded in the DG case.

Another interesting feature of the datagram mode, as compared with the virtual call mode is the different variances of the delivery times. It can be said that the flow of packets is much more homogeneous in the DG mode than in the VC mode. Figure 13 shows the behaviour of networks under different loads, but all with shift set equal to 1. A similar observation can be made when comparing the variances of the delays for different packet inter-arrival rates (changing shift). It is clear that less "bursty" traffic forces down the variance of the time delays and thus allows more homogeneous traffic.

CONCLUSIONS

The purpose of this paper was not to contribute to the discussion on the advantages or disadvantages of virtual call versus datagram services; neither is the analysis conducted complete enough to claim that virtual call is in any way in its interpretation used in this paper worse than datagram. It was the authors' intention merely to draw attention to the fact that, when introducing a standard, a most careful analysis should be conducted on how these standards influence characteristics which seem not to be involved directly. In particular, it has been shown in this paper that the X.25 hop-by-hop implementation leads to decreasing efficiency of the communication subnetwork in terms of delivery time, and this decrease in effectiveness could become significant for large and heavily loaded networks.

Table 1. Comparison of the datagram and virtual call in respect of the usage of communications resources.

	DATAGRAM	VIRTUAL CALL
Setting up of the session	none	delayed until the session is established or risk of misused resources if the first packet is lost or looped
Channel usage due to:	routing for every packet	routing only for setting of a call
Address	full address in each packet - longer packets	full address in the first packet - shorter packets

Table 2. Delivery times for a small network.

Packets/ message	PACKETS						MESSAGES					
	Overall delivery		Class 1 delivery		Class 2 delivery		Overall delivery		Class 1 delivery		Class 2 delivery	
	VC	DG	VC	DG	VC	DG	VC	DG	VC	DG	VC	DG
1	1.89	1.89	1.47	1.47	2.72	2.72	1.89	1.89	1.47	1.47	2.72	2.72
3	3.44	2.83	2.86	2.42	4.66	3.74	4.87	4.01	4.17	3.54	6.34	5.03
5	5.20	4.00	4.33	3.45	6.90	5.04	8.15	6.30	7.01	5.67	10.38	7.51
7	6.77	4.94	5.76	4.33	8.83	6.18	11.32	8.37	9.86	7.66	14.25	9.82
9	8.98	6.21	7.49	5.29	11.87	7.96	15.38	10.80	13.34	9.69	19.34	12.92
11	10.53	7.16	9.01	6.07	13.61	9.38	18.96	12.95	16.75	11.71	23.45	15.51

load: 6.6 packets/tu for net in Figure 1

Table 3. Update information.

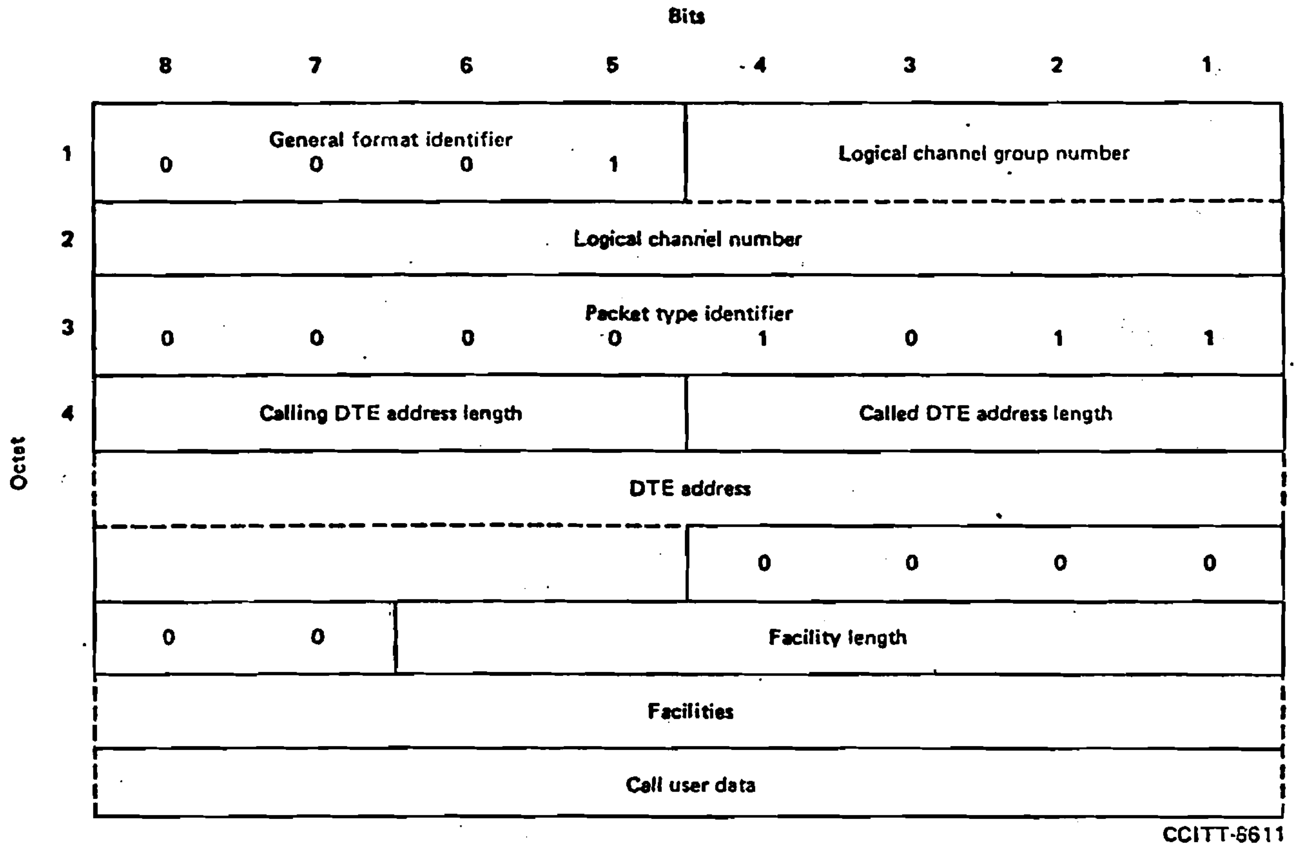
Packets/ message	Arrived update information VC	Created packets · 100 DG	Rejected update information VC	Created packets · 100 DG	∑ VC	∑ DG
1	40.93	40.93	4.78	4.78	45.71	45.71
3	44.30	52.48	12.09	13.70	56.39	66.18
5	42.00	55.13	14.68	20.42	56.68	75.55
7	40.34	54.85	16.34	23.30	56.68	77.15
9	37.22	55.17	17.08	24.62	54.30	79.79
11	31.63	53.50	14.88	25.24	46.51	78.74

Table 4. Packet delivery time as a function of its number in a message, and class of the message.

Transportation method	Class of packet	Interval between packets / The number of a packet in a message	1	3	5	9
Virtual call	K1	1	2.03	2.03	1.86	1.77
		5	4.37	3.39	2.69	2.33
	K4	1	6.83	6.31	6.27	6.35
		5	14.15	11.66	9.8	9.41
	Total delay of the message by VC		12.81	18.93	26.31	41.08
Datagram	K1	1	2.6	2.07	2.13	2.03
		5	3.71	2.48	2.38	2.17
	K4	1	10.53	7.93	8.13	7.66
		5	12.3	9.13	8.86	8.17
	Total delay of the message by DG		12.18	17.44	25.9	40.76

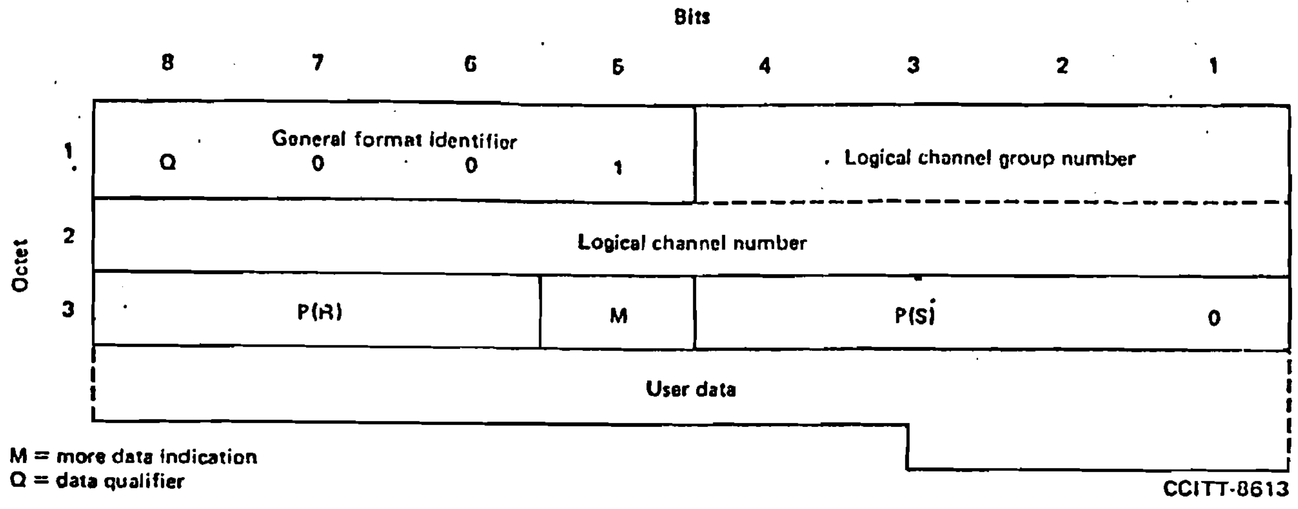
Source Address	Destination Address	Facilities	Identifier	Data

Figure 1. Datagram format.



Note. – The Figure assumes that a single address is present, consisting of an odd number of digits, and that the call user data field contains an integral number of octets.

Figure 2. Call request.



Note. – The Figure assumes that the user data field does not contain an integral number of octets.

Figure 3. DTE and DCE data packet format.

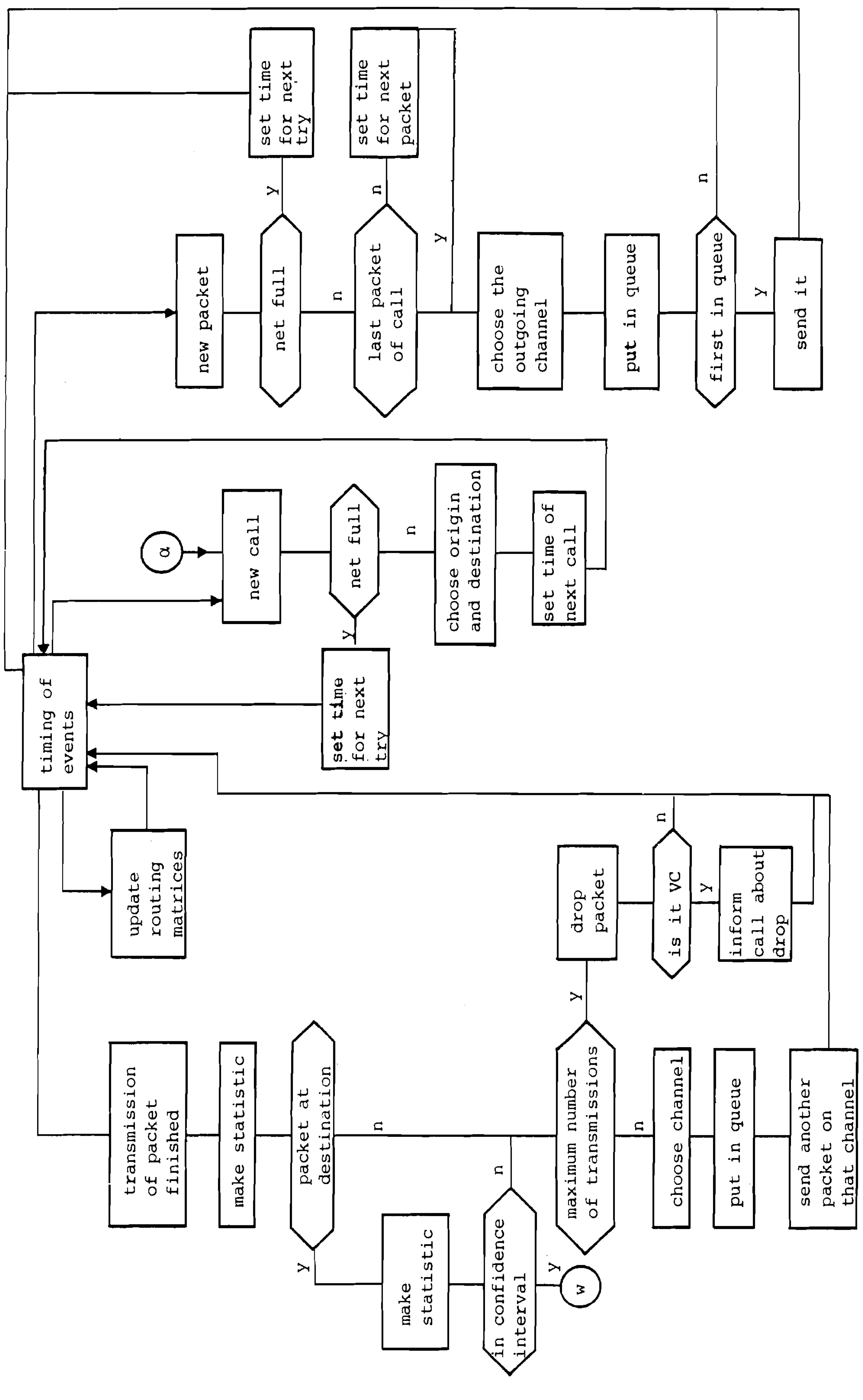
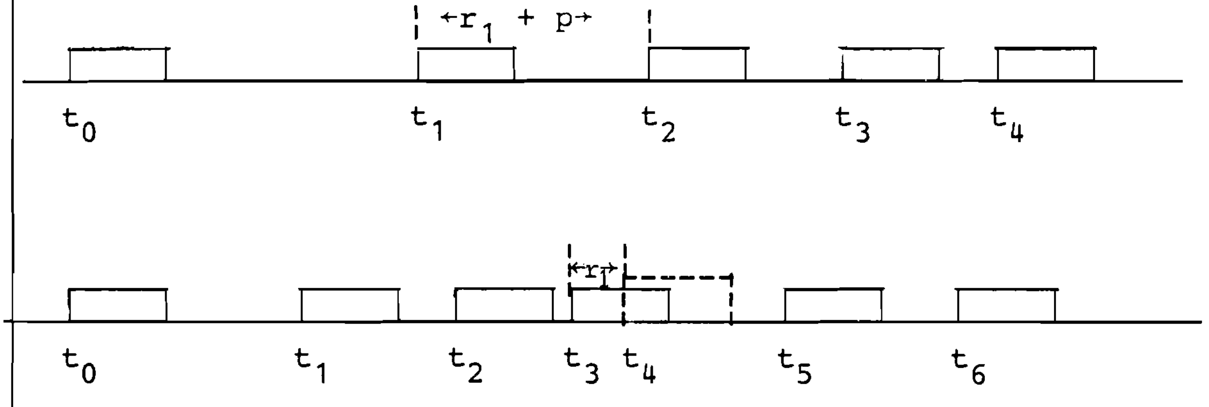


Figure 4. Flow chart of simulation.

creation intervals: $t_i - t_{i-1} = r_i + p \quad , \quad i = 1,\ldots,n \quad .$

packet size fixed: p

packets cannot overlap



$\leftarrow r_1 + p \rightarrow$
t_0
t_1
t_2
t_3
t_4
$\leftarrow r_1 \rightarrow$
t_0
t_1
t_2
t_3
t_4
t_5
t_6

creation intervals: $t_i - t_{i-1} = r_i \quad , \quad i = 1,\ldots,n \quad .$

fixed packet size: p

packets can overlap

Figure 5. Two creation patterns of packets.

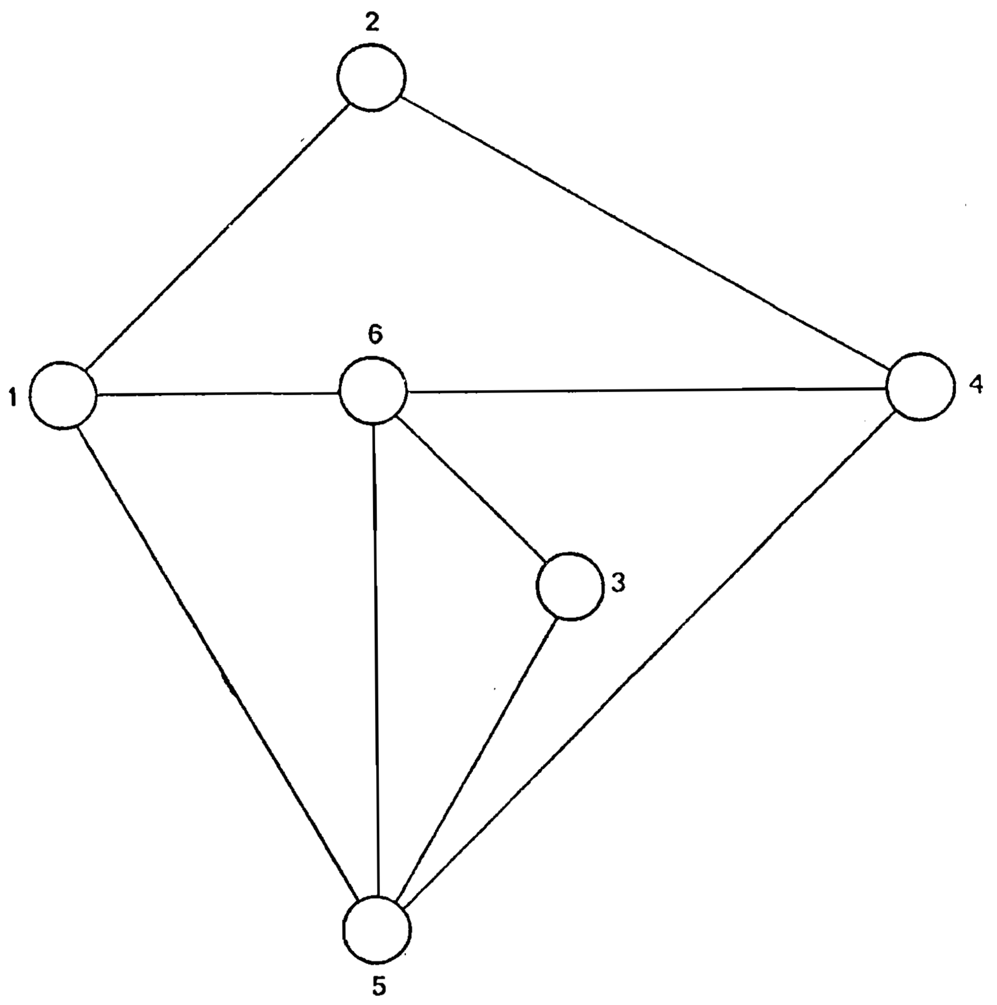


Figure 6. Small network.

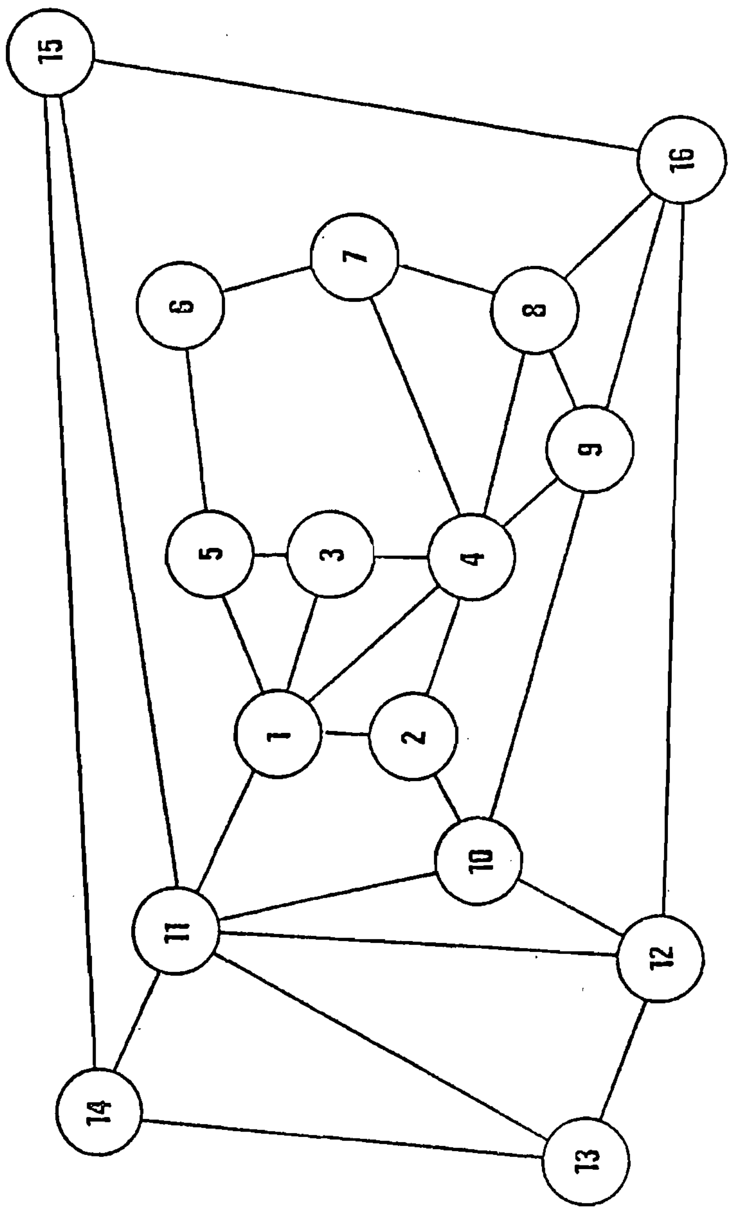


Figure 7. Large network.

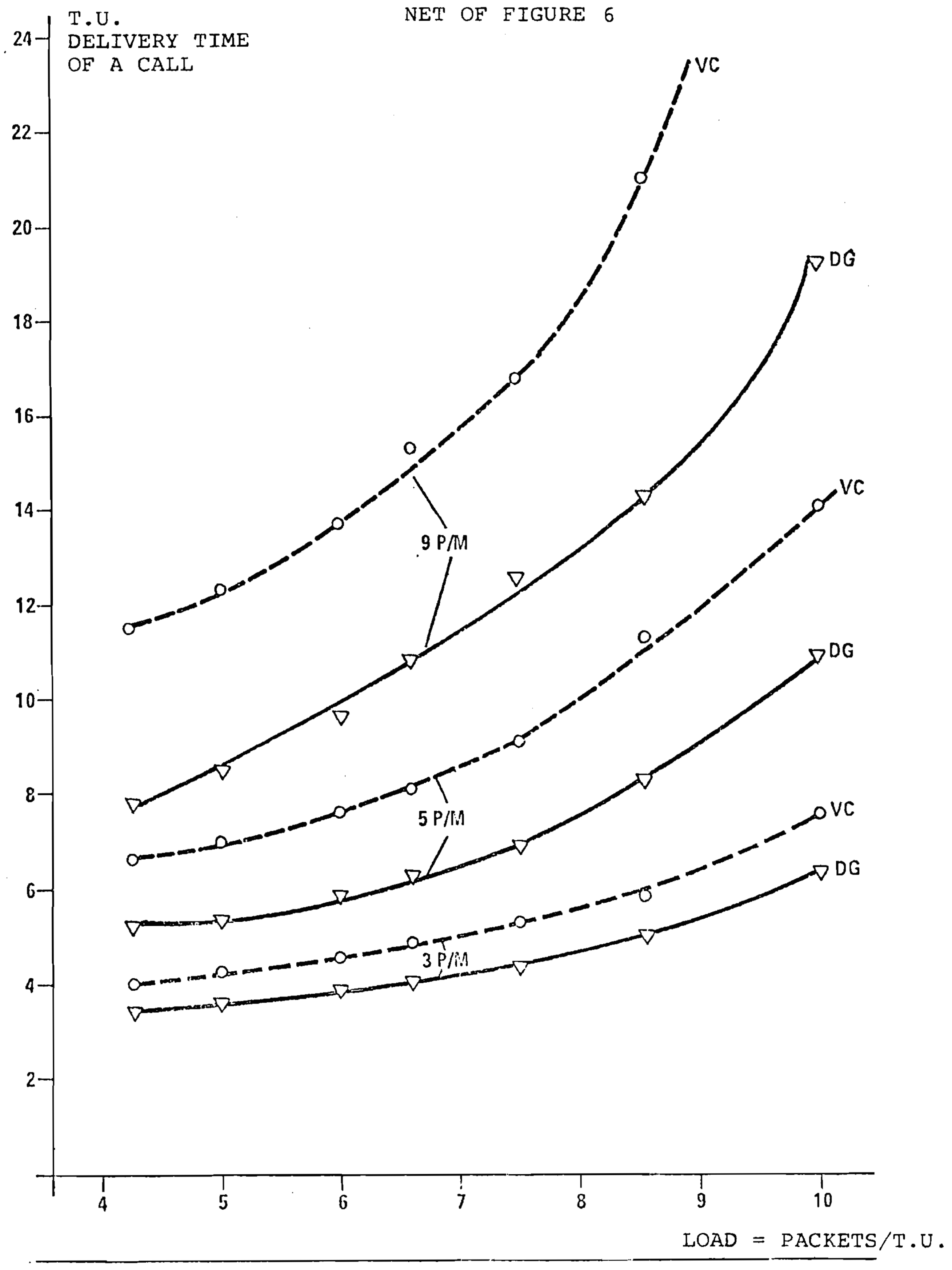


Figure 8. Dependence of delivery on load for 3 lengths.

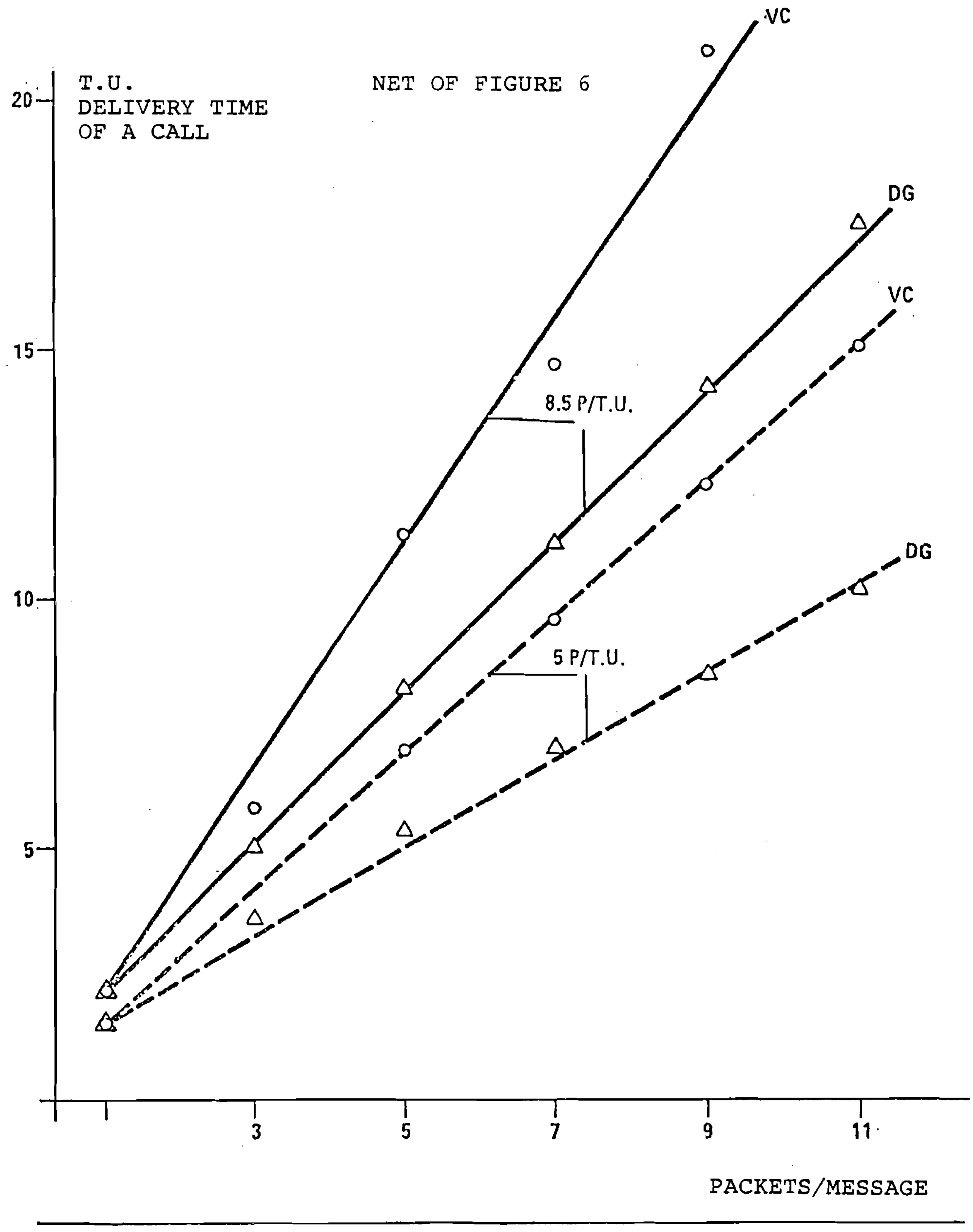


Figure 9. Delivery depending on length.

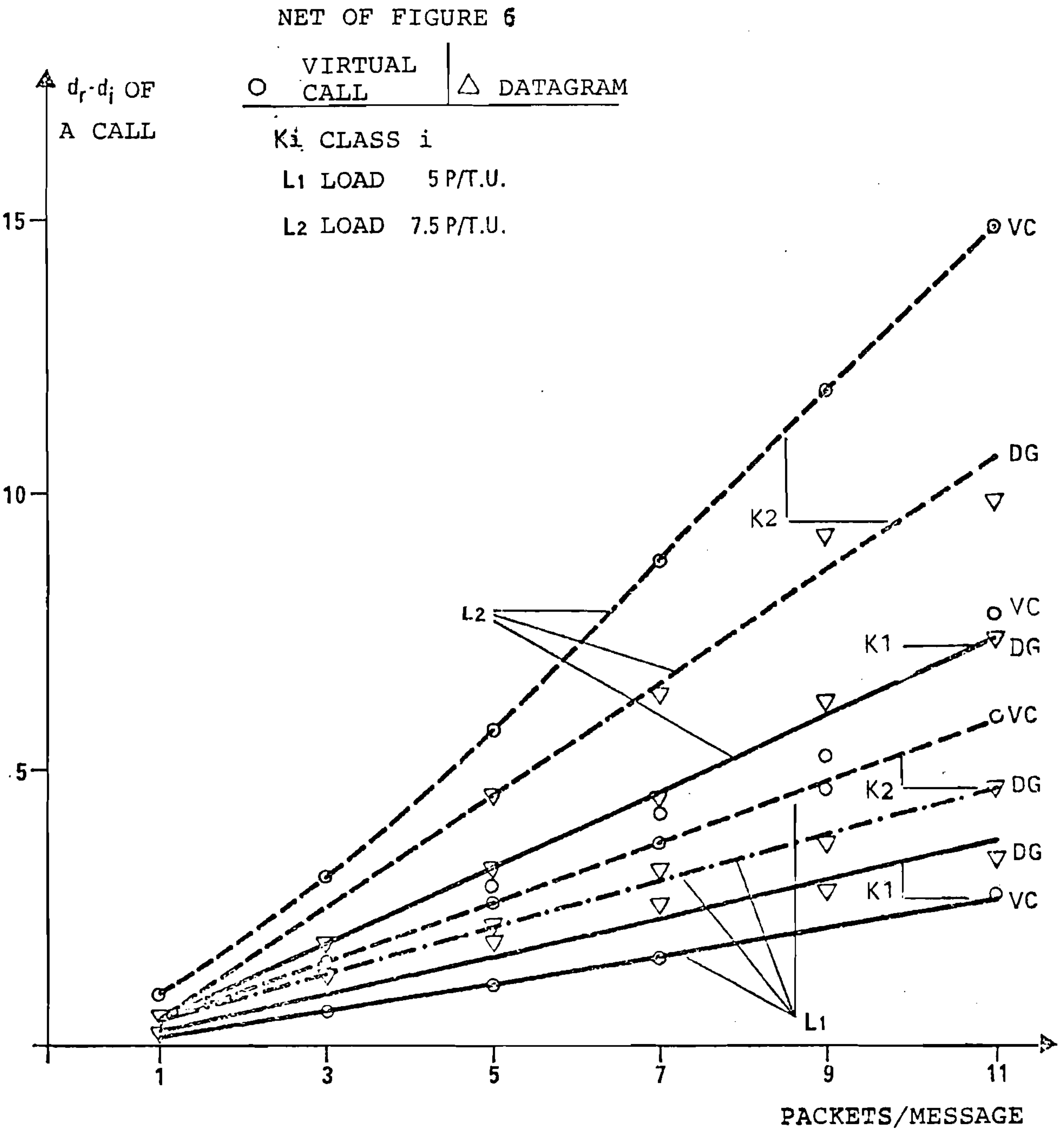


Figure 10. Difference between actual and ideal delivery time.

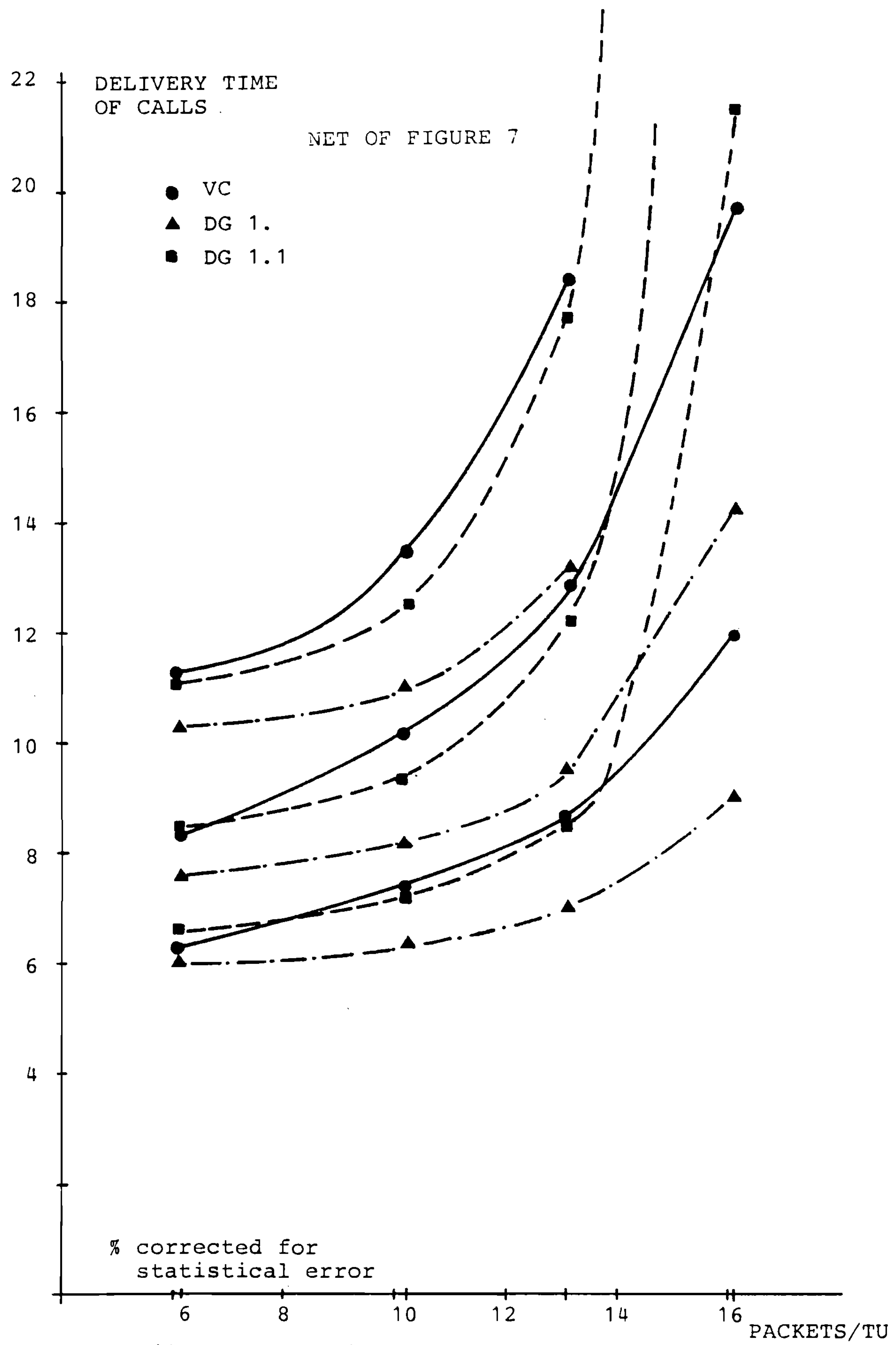


Figure 11. Delivery time of calls.

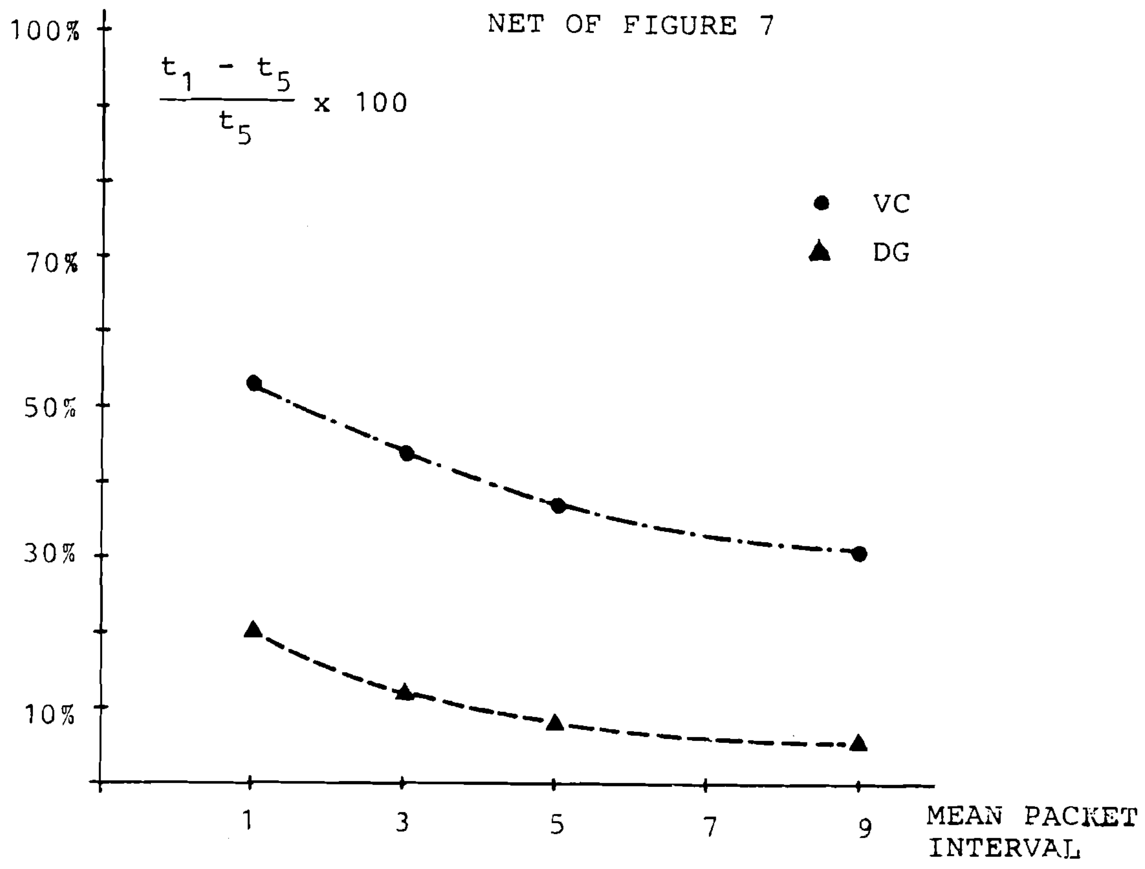


Figure 12. % Difference between first and last packet of a message in VC and DG for different mean intervals between packets of the same message.

t_i = delivery time of the i-th packets in the message

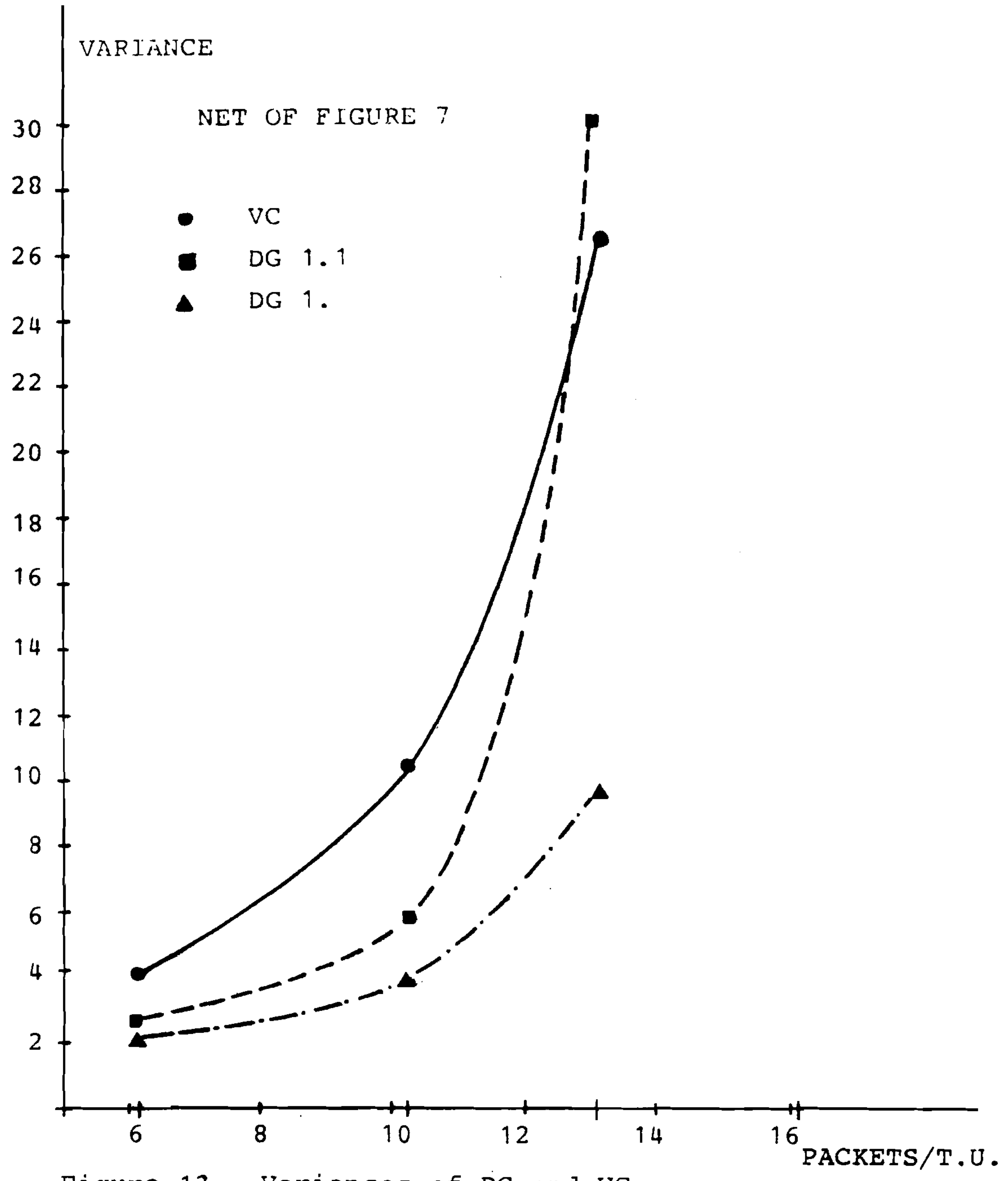


Figure 13. Variances of DG and VC

REFERENCES

[1] CCITT (1977) (The International Telegraph and Telephone Consultative Committee) Provisional Recommendation X.25 - Interface between Data Terminal Equipment (DTE) and Data Circuit Terminating Equipment (DCE) for Terminals Operating in the Packet Mode on Public Data Networks. Pages 8-49 in: Provisional Recommendations X.3, X.25, X.28 and X.29 on Packet-Switched Data Transmission Services. Geneva: CCITT - ISBN 92-61-00591-8.

[2] David, D.W., D.L.A. Barber, W.L. Price, and C.M. Solomonides (1979) Computer Networks and their Protocols. Chichester, England: John Wiley & Sons Ltd.

[3] Sloman, M.S. (1978) X.25 Explained. Computer Communications 1(6):310-326, December.

[4] Butrimenko, A. (1974) Two Criteria of Service and Routing Methods for Packet Switching Networks. Dva Kriteria Kachestva Obslugivanija i Metody Upravlenija Setju Kommutatsee Soobshenij. Pages 142-153 in: Design of Control Systems. Postroenie Upravljanskich Ustrojstv e System. Moscow: "Nauka" (in Russian).