INTERNET NOTEBOOK ~~3~~

POSTEL

IEN # 1

INDRA Note 637
PSPWN 76
29 July 1977

INDRA
Working
Paper

Section 2.3.3.1

ISSUES IN THE INTERCONNECTION OF DATAGRAM NETWORKS

C. J. Bennett
S. W. Edge
A. J. Hinchley

ABSTRACT

This working paper attempts to analyse the main problems of interconnection of datagram networks using a suitable end-to-end protocol such as TCP. The paper expands on many of the comments and discussion which have taken place over the last two years.

Dept of Statistics & Computer Science
University College, London

1 INTRODUCTION.

Although considerable thought has been given to the problems of network connection over the last couple of years little progress has been made to a satisfactory set of solutions. One of the main reasons for this has been the impossibly wide set of bounds for such solutions. Are we attempting to provide a framework for all types of traffic, for all types of users, on all types of network that the technology can provide? If we are providing for a subset of that broad requirement which subset is it to be?

The first delineation which seems use is the three groupings:

interconnection of private nets
interconnection of PTT (X25) nets
interconnection of private nets through PTT nets.

These three groups obviously have problems in common but the constraints are so different that if solutions are to be obtained they should be discussed separately. The first group is the subject of this paper, and has been the main preoccupation of the ARPANET internetworking community. The second problem is being specifically addressed by CCITT and will produce a specification (X7X) by early 1978, which will address the connection of nets providing an X25 interface with limited transit net capability. The third problem has not yet been formally addressed in any detail but will be a preoccupation at least in the European situation.

Having isolated one group, we can then continue to narrow further by postulating datagram networks, as this network type is clearly the one favoured within the ARPANET community for its versatility in routing possibilities. A further constraint is the selection of a suitable end-to-end protocol to be used across the internet path, such as TCP. Although the protocol itself does not need to be tied into the internet solution, the reverse is not true. For example, gateway fragmentation requires specific operations relating to the end-to-end packet formats provided. Finally we come to the network medium. Here, we must decide whether to attempt to provide a framework which can utilise broadcast technologies to the full, or merely one in which broadcast networks operate satisfactorily. This working paper does not deal with broadcast networks particularly. However, it is assumed that internetworking solutions within the current framework will need to be fairly receptive to the needs and benefits of broadcast operation.

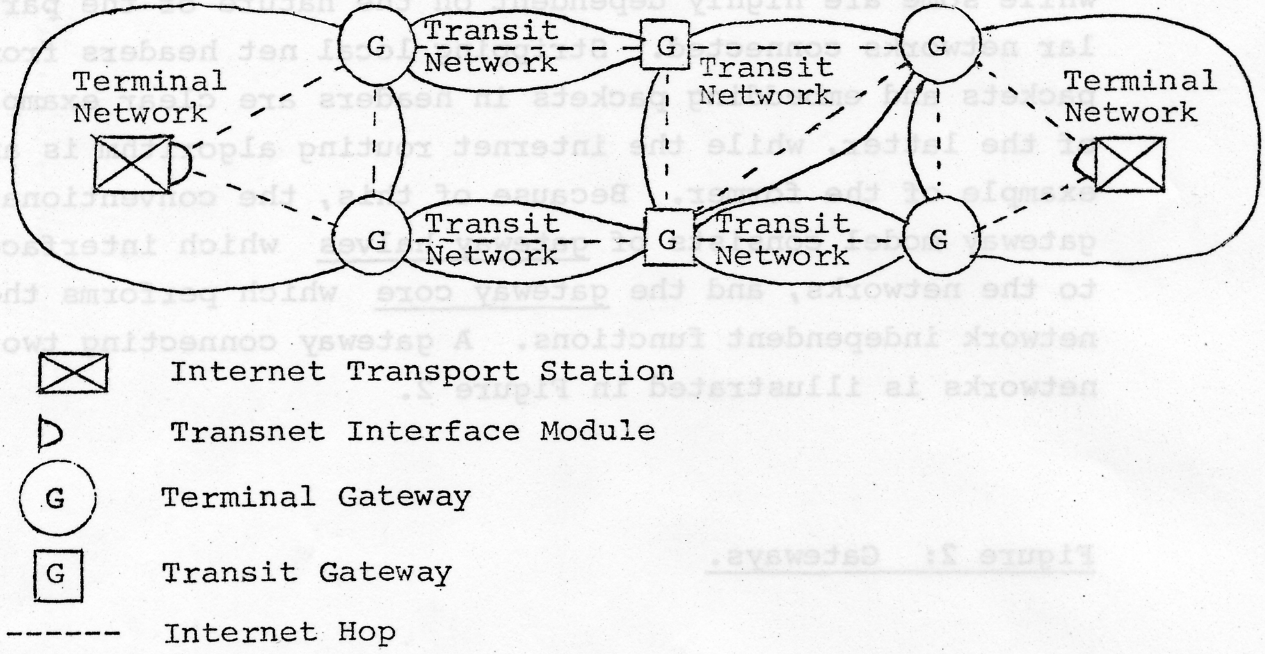
2 THE COMPONENTS OF TRANSNET COMMUNICATION.

There has been considerable debate on the appropriate internet model to be used in discussing transnet communication systems. Partly in order to clarify our own position and partly to help the debate move forward, we briefly present here our version of the transnet environment. This is essentially a supernet model (Cerf 77a) and is illustrated in Figure 1.

The physical basis for the model is a set of networks which are connected via gateway machines, the structure of which we shall examine later. The topology of the supernet is flexible in that there are no inherent restrictions on the way networks may be interconnected. This topology is unlikely to be constrained in the sense that the topology for a local net can be optimally designed. Network interconnections are more likely to continue on the existing pattern - they will be made at the points which seem most convenient to the networks being connected and the overall topology may well continue to be as unbalanced as it is at present. The distributed nature of the topology may be reflected in distributed internet management structures though in some cases more centralised schemes may be preferred.

Communication between two processes in different nets is managed by an internet transport station which is a communication process obeying some end-to-end protocol, of which the TCP is one example. We point out here that the transport station is essentially a software concept, in that the term does not necessarily carry any hardware implications. We shall return to this point in Section 3.3. The purpose of the transport station is to render the structure of the supernet and of the component networks invisible to the user. For any particular transnet conversation we will denote the networks in which the source and destination transport stations reside as terminal networks; intermediate networks which the internet traffic crosses are transit networks. It should be noted that these differences are conversation-dependent: one man's terminal network is another man's transit network.

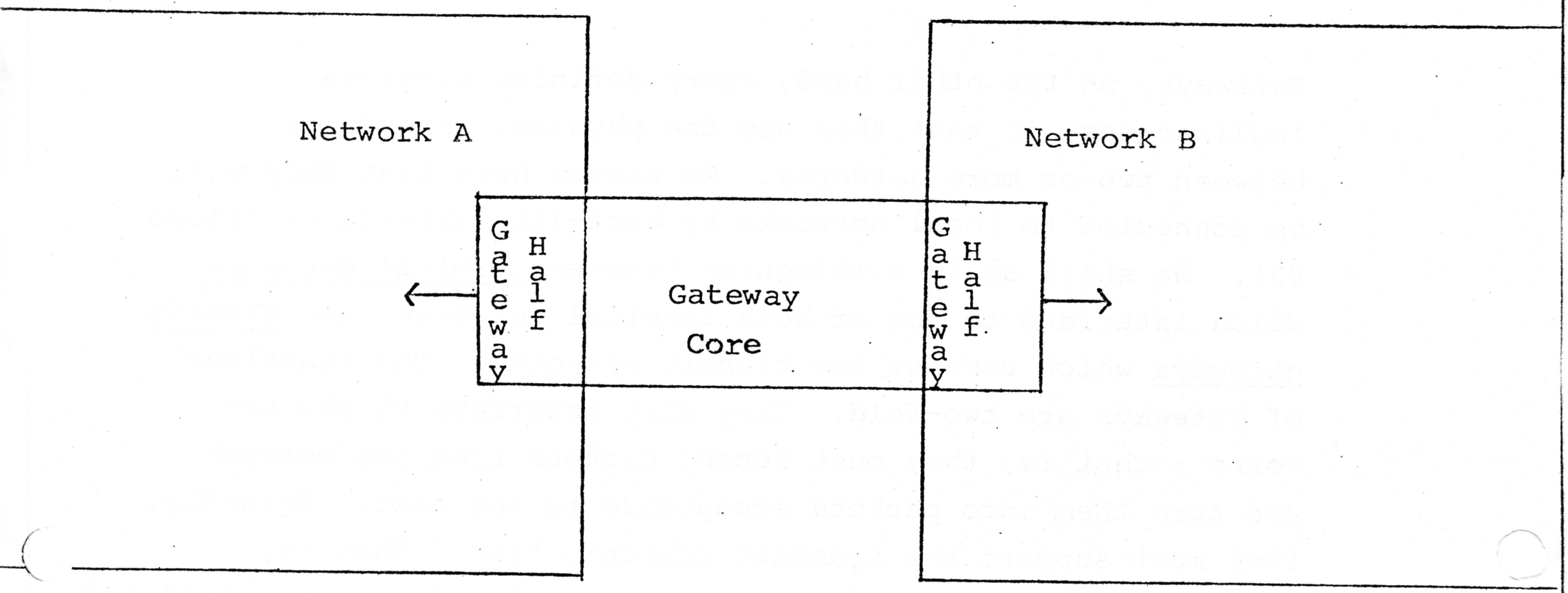
Figure 1: A Supernet Topology.



Gateways, on the other hand, carry definite hardware implications, in that they are the physical interfaces between two or more networks. We assume here that they will be connected to local networks by host-like interfaces (Lloyd 75). We shall again distinguish between terminal gateways which interface to one or both terminal networks, and transit gateways which connect two transit networks. The functions of gateways are two-fold. They must interface to the networks - that is, they must accept packets from one network and turn them into packets acceptable to the next. Secondly, they must support the transnet communication. That is, they must provide internet routing, and if they are required to fragment packets they must (in TCP at least) make the fragments compatible with the end-to-end protocol. In addition, in order to provide gateway stability and to support various end-to-end facilities we may have to build in other functions, such as flow control and error control between gateways, and interpretation of priority information and status, access and accounting information. Much of this was listed in more detail in (Beeler 75)

Some of these functions are clearly network independent, while some are highly dependent on the nature of the particular networks connected. Stripping local net headers from packets and embedding packets in headers are clear examples of the latter, while the internet routing algorithm is an example of the former. Because of this, the conventional gateway model consists of gateway halves which interface to the networks, and the gateway core which performs the network independent functions. A gateway connecting two networks is illustrated in Figure 2.

Figure 2: Gateways.



The fundamental reason why we have preferred the supernet model to the transit gateway model of (Cerf 77a) arises from considering the relationship between the transport stations and the terminal networks. A transport station is a network independent concept. However, in order for it to function, it must physically reside in some machine on a terminal network. The packets produced by it must be formatted to be compatible with this terminal network, and they must be routed to and from the terminal gateways. These are precisely the functions we listed above as distinguishing a gateway, and if we consider the transport station as a gateway core process the analogy becomes even stronger. Thus, conceptually at least, a transport station is co-resident with a gateway half, with possibly some gateway core functions as well. In our view, therefore, it is an absolute requirement that a host providing internet transport facilities must also provide some basic gateway facilities. However, the 'gateway' component of this entity is clearly not an ordinary gateway as it does not interconnect two networks. We propose to call it a transnet interface module (TIM). We stress that although a TIM is not a true gateway it is in many ways functionally equivalent to one, and for many purposes it will be implicitly included in the following discussion as a 'gateway'. In the remaining sections of this paper we will examine the major areas of internet design in the light of the concepts presented by this model.

3 ADDRESSING.

3.1 Introduction.

When one conducts a computer dialogue over several networks, the problem of assigning a name to the destination process assumes critical importance. It has been recognised (Sunshine 75) that there is a need for universal naming space, which will provide unique names over all networks, but this has to be provided in the face of autonomous networks each with their own naming schemes based on highly diverse network architecture. The TCP as it currently stands (Cerf 74, Cerf 77) satisfies these criteria by providing a hierarchical scheme with three levels as follows:

<internet ID>::=<net ID> <TCP ID> <port ID>

This scheme seems generally adequate, and we will accept it as a suitable basis for internet addressing in the following discussion. The key to uniqueness is provided by <net ID>, which is a universally agreed code for distinguishing individual networks. The <TCP ID> satisfies the requirements of network autonomy, as it is decided purely by the local network without reference to others, the only restriction being that it can be fitted into the ample space of 24 bits. The <port ID> identifies a port on the TCP by which the destination process is addressed.

Although the exact interpretation of the <net ID> subfield is clear, less attention has been given to the <TCP ID> and <port ID> subfields, especially the former. In practice, the transport station has usually been identified with the host in which it is physically resident, and so the <TCP ID> chosen is simply the local host ID. The <TCP ID> has been chosen on this basis in both the TCP measurement experiments (Kirstein 77), and the gateway experiment currently being undertaken in the SATNET project. Also, the <port ID> interpretation has assumed that the destination process is co-resident with the TCP. For this reason we shall denote this scheme as the 'physical scheme'.

3.2 Special Topologies.

In this section of this note we shall examine a number of local network configurations to see how visible these will be to the user attempting to access a particular transport station.

3.2.1 Changes in Topology.

If a local net addressing scheme is physical, as in ARPANET, then a change in the local net topology can easily result in a change in the local address of a host. If this address is used directly to determine the ⟨TCP ID⟩, as in the physical scheme, then the ⟨TCP ID⟩ must also change, which means updating the tables of every process, transport station and gateway which is aware of the transport station. In the SATNET project, the BBN gateway has undergone several changes of identity for precisely this reason, although none of these should have had relevance at an internet level.

3.2.2 Co-resident Transport Stations.

The physical scheme breaks down when faced with the possibility of several transport stations co-resident in the same host, as it cannot distinguish between them. This situation has again been encountered in the SATNET project, where it was envisaged that the gateway could support 'raw internet' processes, TCP and secure TCP. In order to make the distinction, a revision was proposed (Burchfiel 76) which effectively introduced an extra level of addressing, the ⟨format field⟩. Even this approach will fail if the transport processes being supported are identical, i.e., we could distinguish between TCP and some other end-to-end protocol but not between two TCPs.

3.2.3 Multiply-Connected Hosts.

There are two possible situations in which multiply-connected hosts can arise. The first is hosts multiply-connected within a network (e.g. by multiple lines, or in a broadcast network). The second is the case in which a host is connected to several networks. A gateway machine running a TCP is an obvious example of this. Both these situations can lead to a host which has several identities. In a physical scheme, the source transport station would have to be aware that the various identies referred to the same TCP. However, in a properly constructed network with multiply-connected hosts, it should be possible to specify the host by a single logical address, and so the physical scheme may not encounter the difficulty.

On the other hand, a host connected to multiple networks is bound to have multiple addresses, as the <net ID> field must be differently specified for each network to which the host is attached. However, the <TCP ID> may also be necessarily different for each network in the physical scheme, and in fact this has happened on the gateway machines in the SATNET experiment. The UCL gateway, for instance, is known as host 74 on network 4 (SATNET) and host 352 on network 12 (ARPANET). The host numbers are different and hence the <TCP ID>s are different.

3.3 Logical Addressing.

We have seen that identifying local hosts with the transport stations which reside in them leads to a number of situations in which the structure of a local net is forced on an internet user. The way to avoid this, thereby keeping the local networks transparent to the internet user is to interpret the <TCP ID> strictly as a logical identifier for a transport station which is unrelated in any way to its environment. In this 'logical' scheme, we would retain the three TCP levels, but propose different interpretations of the <net ID> and <TCP ID> fields in which the burden of determining the physical address of a transport station is placed on the terminal gateways. The <net ID> , as before, is a universally agreed network number. It is also the key to the physical address: when the packet arrives at a terminal gateway, that gateway will look up its internal indicate that the TCP is to be found on a host connected to that network; thus a⟨net ID⟩ of 33 would indicate that the host is connected to both 5 and net Ø. However, this particular scheme severely restricts the number of networks that could interconnect; in the current TCP there would be at most 8. Other more complex schemes could be devised, but all of these would require a radical revision of the TCP header to accommodate a reasonable number of networks.

The obvious cost of logical addressing is the need to maintain tables which map the logical address to a physical one. This working paper is proposing that a set of tables must be created for the association between TCPs and hosts. These tables are limited to the TCPs within a particular network, and any particular table will only be known to the terminal gateways for that network. Updating the tables is consequently also managed solely by the local network, and will normally be carried out when the network would have to acknowledge the change in any case. Thus the costs of maintaining these tables are limited by the extent of the committment the local net has made to internetworking, and is contained within that network.

A further cost comes in considering the allocation of logical addresses. An initial choice could well be to use the local host ID, unless there are co-resident transport stations. In a net where the local addressing is logical, such as packet radio net (PRN), this choice may prove permanently satisfactory. In a network where the local host-naming scheme is physical, such as ARPANET, this can lead to a divergence of local net and internet structures. If at a later time the local host address is changed, this is not reflected in the logical internet address. In particular, if an old physical address is reused within the net for a new host which has internet capability a different transport station address would have to be allocated. Essentially, this reflects the fact that indicate that the TCP is to be found on a host connected to that network; thus a ⟨net ID⟩ of 33 would indicate that the host is connected to both 5 and net Ø. However, this particular scheme severely restricts the number of networks that could interconnect; in the current TCP there would be at most 8. Other more complex schemes could be devised, but all of these would require a radical revision of the TCP header to accommodate a reasonable number of networks.

The obvious cost of logical addressing is the need to maintain tables which map the logical address to a physical one. This working paper is proposing that a set of tables must be created for the association between TCPs and hosts. These tables are limited to the TCPs within a particular network, and any particular table will only be known to the terminal gateways for that network. Updating the tables is consequently also managed solely by the local network, and will normally be carried out when the network would have to acknowledge the change in any case. Thus the costs of maintaining these tables are limited by the extent of the committment the local net has made to internetworking, and is contained within that network.

A further cost comes in considering the allocation of logical addresses. An initial choice could well be to use the local host ID, unless there are co-resident transport stations. In a net where the local addressing is logical, such as packet radio net (PRN), this choice may prove permanently satisfactory. In a network where the local host-naming scheme is physical, such as ARPANET, this can lead to a divergence of local net and internet structures. If at a later time the local host address is changed, this is not reflected in the logical internet address. In particular, if an old physical address is reused within the net for a new host which has internet capability a different transport station address would have to be allocated. Essentially, this reflects the fact that internet transport stations are different entities from local net hosts.

3.4 Process to Port Association.

The foregoing discussion has concentrated mainly on the question of establishing an end-to-end connection between transport stations. However, we are ultimately interested in establishing an end-to-end connection between two 'user' processes sitting above the transport station, and in our discussion of addressing we should also look at the meaning of the ⟨port ID⟩ subfield. The allocation of ⟨port ID⟩ s, and the determination of a remote ⟨port ID⟩ are essentially implementation issues which we do not propose to discuss here; the question at issue is the interpretation.

The ⟨port ID⟩ in the existing TCP specification is essentially a logical address. However, since the TCP is an end-to-end protocol, there has been an implicit assumption in the TCP literature and discussions that a process is resident in the same host as the TCP, and that the ⟨port ID⟩ thus represents a direct association between the TCP and the user process. This model is reasonable on a restricted view of a user process as a creator and consumer of receive and transmit buffers. However, such a process need not necessarily be the ultimate destination process for the internetwork dialogue, which is a more natural view of a 'user'process.

The de-coupling of a user process and a transport station could occur in several ways, the most obvious example is the use of front-end machines to manage network communication. A user process which is physically de-coupled from a transport station must first create a server process on that host (using the local network protocol), which will manage the internetwork connection for it, but may well merely act as the implementor for commands decided by the 'true' user process. To the process on the foreign socket, this situation is not apparent as it is disguised by the logical nature of the ⟨port ID⟩ field (just as the relationship between the transport station and the host would be disguised in a logical ⟨TCP ID⟩ scheme). Indeed, the foreign socket may well contain a precisely equivalent server process, with the true foreign user process resident elsewhere in the foreign net.

4 ROUTING.

4.1 Fundamental Approaches.

Previous discussion of internet routing have tended to favour one of two alternatives. Discussion within the PTT community has favoured fixed routes nominated by terminal gateways (Guy 75) which it appears will by adopted in CC1TT recommendation X7X. The ARPANET based community, on the other hand, has tended to favour distributed adaptive routing techniques (Sunshine 75) of the kind discussed in (McQuillan 74) to optimise net usage. Although a detailed discussion of how adaptive techniques would be applied in an internet environment has not been made, it is clear that the gateways will play the fundamental role. Gateways will maintain routing tables, either network-based (Sunshine 75) or on some other basis, and will exchange internet routing messages between gateways at regular intervals. In order to function properly, TIMs would have to be included in this process, as they would make the initial routing decision for packets being transmitted; just as terminal gateways will make the final routing decision for traffic being received.

Since the adaptive procedure is the most familiar alternative in the current context we will emphasise this scheme in the following discussion. This concentrates on the problems raised by adaptive routing in an internet context, and examines possible modifications which may help to remedy these problems.

4.2 Measuring and Updating Routing Tables.

In order to operate successfully, an adaptive routing algorithm needs to be able to measure some quantity which is contained in the entries of its tables. The algorithm seeks to minimise (or maximise) this object function on an end-to-end basis given its values on a node-to-node basis.

Thus, in order to run an internet adaptive routing algorithm an internet object function must be measured over internet hops. There are a number of functions which could be chosen; we will consider three, namely path length, delay and bandwidth.

Given the great variability in network sizes and structures we have already noted, it is not likely that 'path length' is a reasonable choice for an internet object function. Probably the most reasonable interpretation would be to regard 'path length' as the number of hops from one gateway to the next. But to acquire this information requires the active co-operation of the local net, and if the net does not already use this measure it requires providing a routing scheme over and above that already used in the net, which is a substantial modification to the net's internal architecture. This choice could therefore seriously compromise the principle of network independence.

Measuring delay is probably the simplest object function to adopt, as it makes no demands of the network. It does make demands of the gateway however. Essentially, delay would have to be measured on a round trip basis. We therefore require that gateways be able to recognise and acknowledge at least a special delay test-message. We also require that all gateways be able to convert delay measurements to a common clock unit.

Measuring bandwidth potentially presents the greatest problem of all. The bandwidth between two nodes is a function of the minimum capacity cut set between those nodes. For local net measures this cut set is simply the line joining two nodes and so the throughput is determined by measuring how long it takes to send a message off down the line. For the internet hop, however, the minimum cut set could be anywhere in the network being crossed and determining accurately what that bandwidth is could require doing a large data transfer, thus reducing the bandwidth to nil for a period! An adequate estimate may be obtainable by clocking packets across the gateway interface, or by determining the network's zero-load minimum capacity in special cases. However, the first choice is not accurate, and the second quantity is fixed. Its general use destroys the point of having an adaptive algorithm.

Since transnet traffic is likely to have a longer packet lifetime than local net traffic, it may well be felt that it should be given priority over local net traffic. The implications of this will be discussed in more detail in section 7. We note here that internet routing packets must also therefore be given priority over local net packets, and that the internet object functions used must also reflect this priority, where possible.

An allied question is how often routing updates should be made for adaptive routing. Clearly, there is no point in making these too often, as it takes a certain time for changes within a net to become known throughout the net. For example, routing updates in ARPANET are made roughly twice a second. These are based on information sent from IMP to IMP which propagates one hop per routing cycle; thus if a significant change occured at some IMP it will take several seconds for the whole subnet to register the change. Gateways at opposite ends of ARPANET should therefore only exchange internet routing messages every 3 or 4 seconds - in any shorter interval no significant change will be reflected.

Finally, it is worth considering the effectiveness of such an algorithm once implemented. In any network, traffic between two hosts will tend to be routed along certain preferred paths. These paths will be stable unless the object function between two adjacent nodes becomes unacceptably high, and we can define a tolerance limit for a given hop based on end-to-end considerations. Clearly, we can apply this idea to a transnet communication as well and we can say that on an internet preferred path, a given internet hop is stable provided the internet object function does not become too large. Now, the variation in the object function is normally due to variations in the overall traffic load applied to the hop. In the case of the 'internet hop', however, traffic is injected and removed at several points between the gateways, and so what is observed is the variation in the mean traffic load - the contribution of traffic injected or removed at any single point (including the gateways) is thus reduced. Therefore, an internet object function shows relatively less variation than a local net one does, and so on a preferred path, the object function is relatively more likely to stay within acceptable bounds. In other words, a preferred transnet path is more likely to be stable than a preferred local net path.

Implementing an adaptive routing algorithm thus represents considerable problems if one is attempting to optimise use of the network. Measuring an object function can be difficult to do at all and very difficult to do accurately. A preferred internet path will however, be stable, both in elapsed time (as routing updates are performed less often) and relative to the lifetime of a packet. In view of these considerations, it is worth reconsidering whether an optimising algoritm is worth the effort.

An alternative aim which is compatible with the points raised in this section would be the use of the adaptive algorithm to determine internet reachability. The object function here is simple - one counts the number of internet hops needed to reach a destination. This would only need to be updated in the event of network or gateway failure being detected, and accords well with the use of area routing. An indication of optimality could be given by assigning each network a fixed 'diameter', this would give an order of magnitude indication of the distance to a network. Essentially, we could answer the objections raised in this section by converting to a fixed routing scheme which performs adaptive failure recovery. This algorit'hm will be discussed in more detail in section 4.5.

4.3 Routing by Area.

Sunshine (Sunshine 75) suggested that one of the major advantages of a hierarchical addressing scheme was that it could be exploited for use in an adaptive area-routing technique, where the 'areas' are naturally the constituent networks of the supernet. The routing tables of the gateway have a simple structure in this scheme - a local net portion, for use by the gateway in a terminal role which contains local net routing information for the transport stations in nets it is connected to; and a remote net portion which is used by the gateway in a transit role. In this portion, the transport station is ignored, and the only information kept refers to the shortest route to a terminal network. This scheme accords well both with PTT attitudes and the logical addressing scheme discussed in section 3, in that a gateway is only require to know about transport stations within its own net.

In his discussion of area-routing McQuillan points out that it is a technique to be used in very large networks where the size of large routing messages will become prohibitive. In certain circumstances it is possible for the algorithm to declare falsely that a destination is inaccessible, and except for certain favourable configurations it will provide suboptimal routing. This sub-optimality can be minimised, however, if one chooses areas which are small enough and are suitably configured.

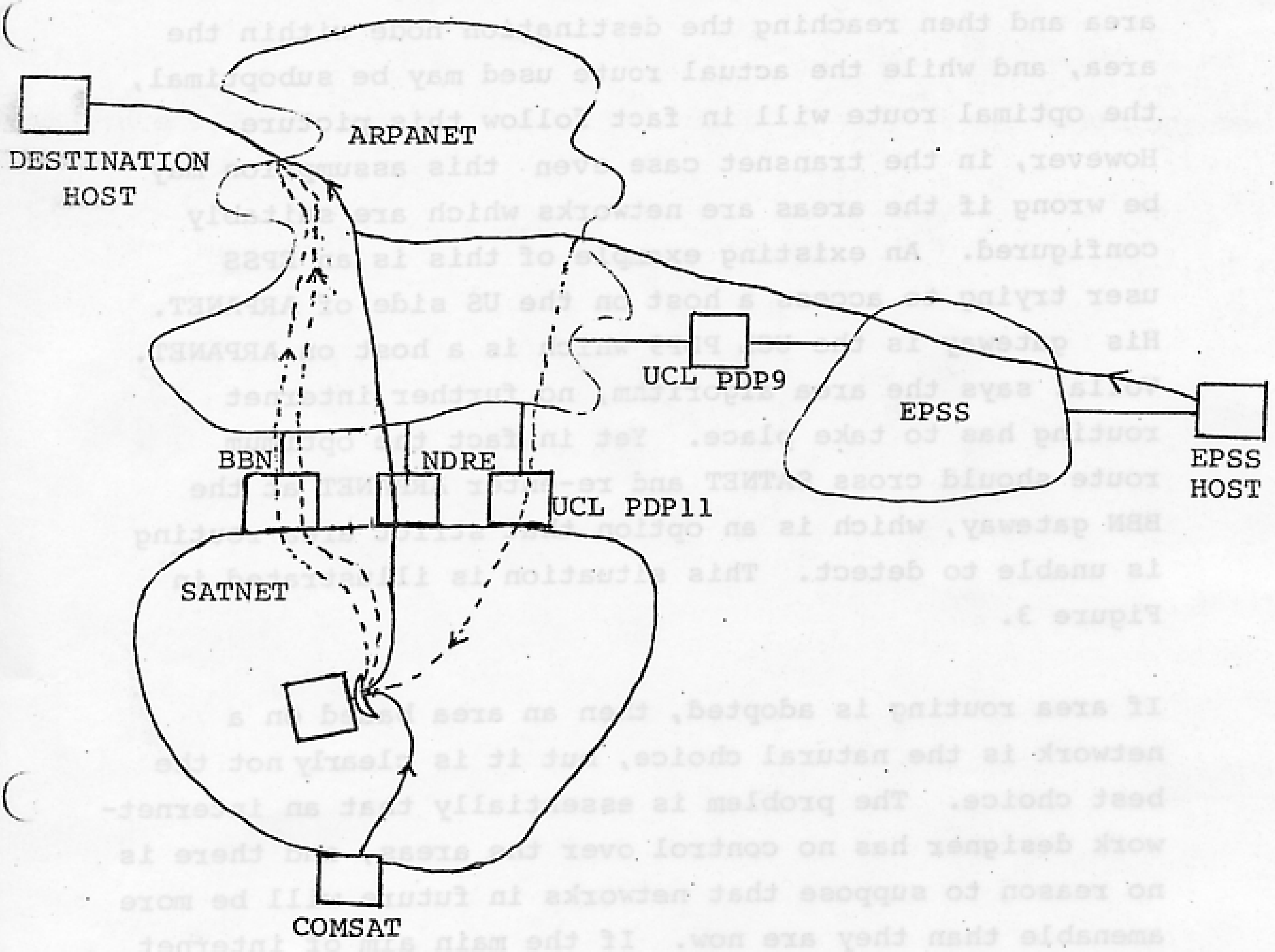
These comments, of course apply to the use of adaptive area-routing techniques in transnet communication. It is therefore worthwhile discussing their impact here. Initially one can expect the supernet to be small, but the possible upper limit of 2^{32} transport stations distributed over 2^8 nets places the potential routing tables light years into the prohibitive category. In practice, however, is it likely that the size of routing tables will become too large without area techniques? It would seem so. Packet radio networks (PRNs - see Frank 75) in particular have been designed with the aim of giving a packet terminal transnet capability. Although our discussion in section 3.4 shows that it is quite possible to treat these terminals as processes on a larger transport station, which would then act as a kind of internet concentrator, PRNs are not being designed along these lines. Other networks may also require that transnet capability should appear to be identical to local net capability, and so the supernet design must assume that these networks at least will contain a large number of internet transport stations.

Secondly, how bad are the drawbacks mentioned above?. The reliability weakness can probably be overcome by modifying the algorithm to seek for terminal gateways rather than terminal networks (though in a well-connected supernet this is likely to increase the size of routing tables again). The objection of suboptimality is not so easily overcome, however. Principally this is because one has no control over the designation of "areas" - these are assumed to be networks which come in all shapes and sizes. Networks already in existence vary in size from 2 or 3 nodes (e.g. EPSS) to nearly 70 (ARPANET). Future networks will have thousands of nodes, and a packet-based replacement for a telephone network could be handling literally millions of nodes. The bigger a network is, the more seriously suboptimal area routing can become for transnet routing. Thus, for example, if traffic were to be sent from the COMSAT small station to a host in ARPANET via SATNET, ARPANET could be entered at BBN, NDRE or UCL with equal probability. Yet the last two choices would cause an unnecessary half-second delay.

The basic assumption behind area routing is that finding the shortest route requires first reaching the terminal area and then reaching the destination node within the area, and while the actual route used may be suboptimal, the optimal route will in fact follow this picture. However, in the transnet case even this assumption may be wrong if the areas are networks which are suitably configured. An existing example of this is an EPSS user trying to access a host on the US side of ARPANET. His gateway is the UCL PDP9 which is a host on ARPANET. Voila! says the area algorithm, no further internet routing has to take place. Yet in fact the optimum route should cross SATNET and re-enter ARPANET at the BBN gateway, which is an option that strict area-routing is unable to detect. This situation is illustrated in Figure 3.

If area routing is adopted, then an area based on a network is the natural choice, but it is clearly not the best choice. The problem is essentially that an internetwork designer has no control over the areas, and there is no reason to suppose that networks in future will be more amenable than they are now. If the main aim of internet routing is to optimise use of the supernet, then consideration should be given to other choices of area, although this may have a drastic effect on internet addressing. It may well prove desirable to break large networks into notional areas, purely for internetwork purposes. These areas could be based on the terminal gateways to a network. These may, however, be lopsidely distributed, in which case this will not be a satifactory choice. The three gateways between ARPANET and SATNET, for instance, are all tied to the Atlantic spur of ARPANET, for good experimental reasons. The 'notional area' of the BBN gateway would probably have to be the whole US side of ARPANET, whereas the NDRE and UCL gateways, would only have a few hosts in their 'areas'. Other alternatives should be investigated.

Figure 3: Suboptimal Internet Routing.



Suboptimal routes selected shown as continuous lines.
Optimal routes shown as dashed lines.
COMSAT to destination: suboptimal due to internet hop measure.
EPSS to destination: suboptimal due to internet topology.

4.4 Access Constraints.

Another problem, which is in part peculiar to transnet communication is how to handle access constraints. These include internet accounting, which we do not propose to discuss here, although it is a subject of great interest to PTTs. We are mainly concerned with handling security restrictions on types of traffic. These restrictions can be end-to-end or they can be imposed by a local net. (Beeler 75) cites as an (unlikely) example of the former the desire of the US Government to prevent its transnet traffic to Europe going via Havana (for that matter, the Cuban Government would like to prevent its transnet traffic to Russia going via the Pentagon!). Local net constraints might include a desire to forbid transit traffic from using that net, or to forbid traffic from or to a certain network or (more perversely) to forbid traffic which has come via a certain transit net.

Restrictions of this sort require a fairly detailed knowledge of a packet's history, or its future. The easiest way to do this is to use source routing, where the networks involved can inspect the entire route at connection set up. Other methods do not lend themselves readily to this. With fixed routing and TCP one could perhaps send a 'route collector' as part of the SYN packet for a connection, which picked up the identities of the transit nets as it went along, and the complete list is returned along with the SYN/ACK packet. However, if someone objected to the route thus described the gateways would have to provide a mechanism for selectively rerouting data for that connection only. That is, with fixed routing algorithm, gateways may be required to understand details of the end-to-end protocol, and to suspend the multiplexing of internet traffic.

The more adaptive the routing scheme becomes, the more formidable are the problems of handling these constraints.

With full adaptivity the proposal discussed above would be inadequate for handling end-to-end constraints as the routing history would have to be attached to every packet. By the time a data packet had reached the destination, it would be too late to object to a regulatory breach. This problem will also arise with the semi-fixed routing advocated in section 4.5, although not as viciously; possibly a 'routing history' packet could be generated for each connection after a routing update. However, this is still a costly approach. In general, it would seem that if it becomes important to handle access constraints, the only adequate techniques will involve some form of source routing.

4.5 Alternatives.

Internet routing would appear to be far more difficult than is currently believed. Area-based adaptive routing, which appears to be the method most in favour in the literature is difficult to implement, suboptimal in many cases, and may be unnecessarily intelligent due to the increased relative stability of preferred transnet paths. Moreover, there are some internet requirements such as the access constraints discussed in section 4.4, which are totally incompatible with adaptive routing.

The problems arise because of the nature of the communications medium. For routing within a single net, we have control over the network topology, we can inspect the traffic at each packet switch, and we can make reasonable assumptions about the homogeneity of the net. For transnet routing, precisely the reverse is the case. Each internet hop has its own characteristics, and these will often depend on traffic within the net, which is invisible to the gateways.

In our view the most reasonable course is to low the demands we make on routing. Many of the problems discussed in sections 4.2 and 4.3 arise because we are trying to chose the optimal route at any given point in time. It seems reasonable to suggest that we should learn to live with suboptimality, provided we can still vary the route in 'catastrophic situations. Similarly, it seems reasonable to question how important it is to be able to impose access restrictions on the supernet.

If we do this, then the internet routing problem becomes one of finding a path which is 'good enough'. One approach would be to use fixed routing with some alternate paths built in in case of failure. However, one is loath to discard adaptivity altogether as it has many advantages - in particular, it is easy to incorporate new internet links, and the algorithm can automatically handle failures in the supernet.

There are two modifications which could be made to the adaptive algorithm to suit our needs. The first is to abandon measurement of the internet object function. This would be replaced by a table of 'reasonable' pre-assigned values. The only other value possible is 'infinity', which is inserted when network failure or overload is detected. The second change that follows from this is that routing updates should be event-triggered. There are very few events that need cause a routing update. These are: network (or gateway) failure, network (or gateway) recovery, special command from a gateway control centre (e.g. to assign a new value to an object function - see section 6), and the arrival of an internet routing packet.

These two modifications create a semi-fixed routing scheme, as updates will now only occur when there are significant changes in the supernet. We have lost the ability to optimise the internet path, which it is doubtful we ever had, but we have retained the flexibility that adaptive routing gives us when we need it.

5 FLOW CONTROL.

5.1 Aims of Transnet Flow Control.

In general, flow control in networks should achieve two objectives - optimising some user service requirement while achieving a balance between efficient of the host and the network resources, and the need to protect these resources from saturation (Pouzin 76). Flow control in its optimising role is usually on a global or an end-to-end basis. In the internet context, efficient use of the end-to-end transport station resources involves the kind of buffering strategies discussed in detail by Edge (e.g. Edge 77), whereas efficient use of the supernet is to a very large extent a routing function. Many of the problems with adaptive routing raised in section 4, are therefore also relevant to a discussion of efficient use of the supernet.

Congestion control is essentially an interface problem: hosts must be protected against flooding from the network and vice versa; nodes in the subnet must be protected in a similar fashion. In the supernet case, we must protect transport stations and the supernet from each other, and within the supernet, we must protect local nets against gateways and TIMs and vice versa. Some of these topics have been broached elsewhere - for instance it is clear (Cerf 77) that the TCP window protects the transport station against flooding from the supernet. Others have been neglected in internetwork discussion.

5.2 Methods of Flow Control.

Optimising transnet flow amounts essentially to removing all intermediate obstacles on a particular connection. On an end-to-end basis, in a protocol such as TCP, this means granting sufficient credit (window space in TCP) to a sender to permit unhindered transmission, and the receiver should be prepared to buffer a sizeable amount of out of order or undeliverable data (Edge 77). On a hop-by-hop basis, intergateway signalling (discussed below) and local subnet congestion controls should also not inhibit data flow, though of course some form of flow constriction will be present as various levels of protocol try to protect resources. Optimising flow may then use priority transmission at each gateway, which will be useful for improving flow at the expense of other connections. Adaptive routing techniques could also be used to improve flow, subject to the problems discussed in section 4.

As mentioned in section 5.1, resource protection is an interface problem, and most of the approaches we shall discuss in this section will be based on the needs of an individual resource. There is one relevant technique which takes a global approach. Isarithmic control (Davies 72) seeks to limit the total number of packets flowing in a subnet by use of permits which can be exchanged for packets at an interface. In the internet context, this would imply placing limits on the total amount of permitted internet traffic. In order to distribute the traffic load away from congested nodes it would probably be desirable to have high inter-gateway connectivity with permits being distributed in a randomised fashion. This amounts to a disguised form of random routing, introducing a conflict with our other aim of optimising supernet usage. If isarithmic methods are used with the kind of semi-fixed routing advocated in section 4.5, it degenerates into a crude form of end-to-end control.

Other protection mechanisms can operate on a local basis. The simplest of these is blocking (Cerf 77a) imposed by heavily congested gateways or TIMs. Incoming packets are discarded, on the assumptions that they will be retransmitted eventually and that by that time the congestion will have cleared up. This could also be done on this basis would seem to be a suitable mechanism for transport interfaces protecting a slow terminal network. In general, however, blocking is a very crude method, to be used as a last resort and if it is to be used effectively, it must be associated with some fast feedback mechanism.

The need for such feedback raises the possibility of intergateway signalling. This could be done either directly or indirectly, either in some standard fashion or by exploiting the facilities provided by the local net. Direct signalling from gateway to gateway could well use a standard method of flow control such as the TCP window, or a cruder credit return system. Selective control of individual streams or groups of streams (e.g. all streams with the same priority) could be achieved by demultiplexing incoming traffic in our semi-fixed routing scheme, although this implies that gateways are aware of many aspects of TCP. The alternative is to exchange signals between the gateway and subnet requiring action from either party. Signals sent by a gateway would often cause the net to send signals to another gateway. This indirect signalling would imply a subnet interface which again could either be standard (e.g. X-25) or local net dependent. While this may well require that local nets provide a special gateway interface, the scheme does have a number of advantages not available with direct signalling. It could be used to give gateways advance knowledge of congestion in the local net, and it provides a method of protecting the local net against flooding from the gateway. Especially if it were combined with rapid direct signalling this could well be the most effective method of congestion control.

The signals used have to indicate two states in order to achieve congestion control. There must be signals for a blocking condition, and there must be signals to indicate way these are implemented is open to discussion - as we have indicated, anything may be possible, from a blocked/unblocked signal to a full windowing scheme.

5.3 Gateway Storage Requirements.

In normal operation, small queues (of the order of a few packets) might arise at gateways due to variations in traffic arrival, gateway processing delays, and packet acceptance by attached subnets. Catastrophic conditions occur when output to attached subnet or adjacent gateway is blocked or constricted and an output queue builds up. The nature of the blocking signal depends on the form of gateway signalling chosen (see section 5.3).

With short term blocking conditions - of the order of a local (not-satellite) net round-trip delay - gateway storage should be sufficient to contain arriving traffic until senders can be warned, and while the blocking condition persists. The required storage may be estimated using the pipe capacity of each adjacent gateway-gateway pair connection. This is defined as the maximum data a amount which can travel, on average, in the subnet "pipe" between connected gateways and hosts, and is given by the product of the mean round trip delay and the network bandwidth (where the network bandwidth depends on the minimum cut set, as in section 4.2).

After signalling a blocking condition to all neighbours, a gateway can receive from each neighbour data equal to a gateway-neighbour pipe capacity. Thus gateway storage data of, at most, the sum of these pipe capacities will be sufficient to contain arriving data, while the blocking condition lasts. In general, this policy may also cause packet pile up in preceding gateways, as constriction propagates backwards. After removal of the blocking condition, traffic would be released in turn as restart signals propagate backwards.

When blocking conditions are long term (exceeding an internet round trip delay or satellite delay), dropping packets at a blocked gateway may not matter, since blocking delays may now exceed retransmission delays. However, senders should still be warned to prevent needless retransmission while the blocking condition persists. When an alternate route exists, bypassing the blockage, then dropping packets will be preferable, since preceding gateways can re-route retransmissions and subsequent traffic around the blockage, as discussed in section 4.5. High inter-gateway connectivity would make this policy especially attractive. With this policy of signalling blockages and re-routing retransmissions, gateway storage need not exceed a few packets, except in very long-delay or high bandwidth nets.

6 MONITORING AND CONTROL OF GATEWAYS.

As with other aspects of the design and operation of gateways, the use of the 'supernet node' analogy with the nodes of a single net is only partly suitable as a basis for establishing gateway requirements. Gateways between heterogeneous networks are by definition unique functional entities between those two networks. The example of ARPANET/SATNET is unusual in that three gateways exist between those two networks, and therefore those gateways happen to be identical. A more normal occurrence would be that the gateways not only operate different software functions due to connecting dissimilar networks   , but may also operate on different hardware. Certain gateway functions will be common to all gateways - the exact proportion of these common functions being one of the subjects under discussion. Although simple monitoring functions are easy to specify, total control of the gateway, extending to real control over the supernet must assume a very substantial common proportion of common gateway functions, a common way of exercising control over these functions and a common way of exercising some control over each subnet through gateways.

6.1 Monitoring Functions.

Gateway monitoring covers those tasks needed to be completed to ensure that the gateway continues to function as a software system. The method of achieving monitoring and control is discussed in a later section. Software systems can be designed to be substantially self-sufficient by virtue of multiple processors and specific software checking tasks. To achieve recovery under all circumstances as well may not be straightforward, and the overall solution may not be cheap. For network nodes, it is simpler therefore to rely on the connectivity of a communications network to place the intelligence elsewhere for fault detection and recovery. This approach is helped by the fact that the amount of information held in a switching node is comparatively small, and in the event of a restart it can be built up again fairly quickly. Whether the same approach can be applied to gateways depends on whether our approach to gateway design is sufficiently close totthat for switching node, together with considerations on how self-sufficient we would really like the gateway to be.

Gateway State can be summarised by trouble reports indicating excessive queues, excessive delays in servicing queues, or corruption of critical values. Some 'null' report is presumably also needed from time-to-time to indicate satisfactory functioning. Gateway Traffic Flow can be fairly simply reported within the previously indicated monitoring framework. The state of lines and networks attached to a particular gateway can be monitored by exception reports with presumably some filtering on very frequent changes of state. Network changes may be expressed in the appropriate protocol for that network, and outages reported back in an intelligent fashion. Line outages will presumably also be seen by the network control of the network in which the line occurs, and mutual arrangements for reporting outages will be required where appropriate.

6.2.1 Gateway Control.

Here,we refer to the control of individual gateways in order to modify the behaviour of a particular gateway, rather than a consorted 'supernet' approach which is discussed in the next section. Gateway control will consist of several functions.

System reload/restart is a minimum control function triggered by a sufficient set of trouble reports of the the type described in the previous section. For diagnostic reasons, some data on the machine state prior to restart may be required to be forwarded to a suitable database. Reload can take place from a local backing store given a suitably robust medium, or can be from a designated mainframe. In a centralised control situation, reload may be from the central control machine, as for ARPANET nodes. Administratively, of course, a large number of different binary images will exist for all gateways.

Routing updates will be needed at intervals to take account of changes in network connection topology, or policy on transit network usage. Although elsewhere we suggest that terminal gateways should specifically deal with routing to hosts within that network, the main internet routing tables are presumed to be held in a similar structure in each gateway in the common 'gateway core'. In a fixed/alternate scheme, a particular topology change will require a check on which gateways are affected by the change, and specific table updates will need to be individually arranged for those gateways. In a routing scheme of a more dynamic nature, apart from perhaps universally extending the gateway table, there is no need to make manual table changes, as the effect of the new additon will be propagated through all gateways.

Access control changes go hand-in-hand with routing changes. The degree of sophistication of network access is a design consideration. Control can of course be provided by source, destination, user group, or individual user identification as discussed in section 4.4. Control can be exercised at all gateways, or at the traffic source. Considerable additional table space will be required with associated updating problems for an extensive control scheme. For a pure datagram gateway, checks will need to be carried out for every single message passing through the gateway. In a fixed routing scheme, some connection information can be retained in the gateway, although of course this brings its own set of problems. Carrying access information increases the reload problem. As a specific example observed in the UK, the British network EPSS carries terminal descriptions, identifiers and passwords for all character terminals dialling the exchange. This information is not always reloaded on a restart, or may get corrupted without the main switching software being aware of the problem. Thus service is apparently provided as far as the switch is concerned, but in fact the user is unable to get his terminal recognised by the exchange.

Gateway throughput regulation may require changing from time to time to govern the volume of messages passed through specific transit networks.

6.2 Supernet Control.

A supernet view of interconnected networks implies that control should be exercised over all gateways in a common fashion in the same way as nodes of a network are controlled. Advantages of such a control scheme are obvious in that the entire global flow of data across net boundaries can be modified in reaction to particular events such as the loss of particular transit networks, in a way which is optimal for the supernet. Particular abnormal operational quirks-lockups, routing failures leading to network overloadings etc., can be more easily observed from the supernet point of view. Disadvantages are that such a hierarchical structure may not be administratively agreed, and if organised in a centralised fashion may require too great a flow of information to a single point in the supernet. Furthermore, such a centralisation of operations implies network component reliability of an exceptional nature if a large set of interconnected nets are to be controlled.

Some compromise is obviously required which is satisfactory from a technical point of view and also satisfies administrative criteria. We believe that some supervisory control will be needed over sets of gateways, but that gateways are likely to achieve some degree of independence by implementation of suitable checking operations, and possibly by use of adjacent gateways to participate in monitoring exercises on each other, at least to the degree of listening for regular 'gateway up' messages. This degree of independence can be helped by perhaps limiting the degree of adaptive routing allowed by gateways, to the point where flexibility is retained but uncontrollable states are unlikely to develop. Access control information could be used in such a way that if checks are impossible at one gateway, overall checking still continues on the overall traffic throughput.

We assume, therefore, that one or more gateway control centres will exist each controlling a group of gateways, which in the general case will be running dissimilar software systems in dissimilar hardware, but with a common gateway core set of functions as outlined earlier in this working paper. The precise number and organisation of gateway control centres is an administrative problem, based on the separation between networks, and the total number of gateways involved.

To implement the gateway control centres in a reasonably implementation independent fashion, just as a gateway-gateway protocol may be required to implement flow control between gateways, we need a gateway-gateway control centre protocol (which could use a common base) in order to specify a gateway command interface, and a monitoring interface back to the control centre. The command interface must be able to achieve in an independent fashion the functions indicated in section 6.1.

In addition, rudimentary debugging functions should also be specifiable at this interface so that dumping of important tables and critical values can be done without resort to system-dependent tools such as the XNET used to control existing PDP11 gateways. Such low-level tools should not be part of the control environment, but rather engineering tools to be used when the particular system is deemed to be out-of-service. Similarly patching of code modules (such as carried out by the ARPANET NCC) can be reduced if system changes likely to be encountered such as looping of lines, temporary routing loops, and buffering allocations can be encoded in an independent fashion. Undoubtedly, the addition of such an interface will add to the complexity of the gateway. However, we suggest that this a reasonable tradeoff towards the simplification of gateway functions. We note that the design for the gateway/ SATNET control centre does provide a common interface to the human operator (Estep 77) for some of these functions, but does not attempt to take the interface any deeper.

In essence we have added a new interface to the gateway core - its interface to the gateway control centre. Some attention should be paid to the addressing of the gateway from other gateways, and from the gateway control centre. Should the TCP be used as the main method of communication? If not, should an appropriate simple protocol be adopted, and should this have the ability to address several different processes in the gateway? Should the gateway processes be addressable in different ways for nets connected to the gateway, or should a separate supernet designation be made to cater for gateway addressing? Certainly without it the connectivity of the gateway cannot be taken advantage of by the gateway control centre.

7 PRIORITY AND DELAY CLASSES OF INTERNET TRAFFIC.

To date, little use has been made in packet-switched networks of extensive delay/priority classifications. The store-and-forward nature of the medium is different from classical message switch systems, which expect to queue items for some time before onward delivery. In a packet-switch system queues are kept short by passing items on at the earliest opportunity. Given high-throughput lines with minimal node storage, different classes of item would move through the network at very similar rates. We must assume therefore that the main reason for introducing delay/priority classifications is to handle very heavily loaded conditions, or cases where due to a slow rate of takeup by a host, or limited line bandwidth, only a limited flow can be accepted. In these conditions it is quite possible, by suitable buffer reservation strategies to severely limit traffic flow of lower priority classes. In a virtual call net, where connections are always identified to the subnet, it is possible to refuse the call altogether or to allocate that call negligible buffer space at switching nodes. In a commercial network, tariffing can be used to extract a higher revenue from the higher priority traffic class (as proposed in T ranspac in France.)

SATNET represents an unusual case of priority/delay in that traffic waiting for transmission at all ground stations can be ordered in a CPODA system into a transmission queue summed for all waiting traffic. Here, priority/delay can obviously make a more significant difference.

The first question for internet priority/delay classification is whether a uniform definition can be established which is at all meaningful. It is necessary to state whether priority only applies within a single delay class, or whether priorty means an overall traffic ordering, but giving respect to delay classes when necessary. Delay class implies an absolute time value. To achieve an accurate figure for this require sufficient gateway synchronisation to accumulate delay figures on a per packet basis. If no timing is to be carried out, then only a rough estimate of total delay likely for a particular class can be made. Priority is even more difficult to tie down given that its effect depends totally on the sum of traffic moving through a network at a particular time. To give an example, if all Transpac users opt for the priority service, Transpac will be unable to provide the guaranteed service level to <u>any</u> customer given that traffic will then be undifferentiated.

In the internet case, it is also necessary to ask the question whether internet traffic takes priority over local traffic.

It can be concluded therefore that even given a common acceptance of priority/delay definition, some control over the internet traffic is required in order to make delay and priority classes very meaningful. We can assume that is virtually impossible for every net to have a similar definition of behaviour for a particular traffic class. What can be achieved is an internet limited set of classes encoded in the internet header, and translated by each gateway into the appropriate local definition for that particular class. That translation may range from non-existent for a net offering no distinct classifications, to a translation (in the case of SATNET) into an ordering which will always be taken account of in a CPODA-type system.

A slightly different approach is being taken in PTT nets, where classification and flow control are united in the concept of a throughput class. At the time a call is set up, its throughput class is defined, and sufficient space is reserved to meet that throughput specification. Presumably after a certain point, only calls of a more limited throughput class will be accepted, in anticipation of overloading. The method is not completely foolproof of course because some calls will not deliver the class they have requested, so it is still necessary to overallocate to ensure efficient use of resources (for example, airline reservations). We are therefore back much closer to our traditional queuing picture of random arrivals than at first would have seemed apparent.

8 CONCLUSIONS.

This paper has explored a number of areas in network interconnection. There are three fundamental points which have emerged from this discussion. The first concerns the use of the supernet model. This model implies a fairly uniform structure to the supernet,which would be closely analagous to a homogenous distributed subnet. Yet at several points we have had to draw attention to the diversity of the networks which make up the supernet. Very rigorous and careful design is needed to make sure this diversity does not intrude on the basic supernet functions. This is the fundamental reason why we need to define a TIM, and why we need to make sure our addressing scheme refers only to internet objects. The intrusion of local networks also lies behind the objections we have raised to full adaptive internet routing, and to our arguments for heavy gateway involvement in flow control.

The second point which emerges is that there is a strong case for a gateway to gateway protocol. This would handle several functions - notably exchange of routing packets; access control; congestion and flow control; and management of classes of priority traffic. Some of these depend on the type of gateway involved (e.g. routing); some can be conducted without considering the end-to-end protocols and others are deeply dependent on it - especially management of priority traffic. When this protocol is designed in detail careful attention should be given to this last division. It may be held desirable to support several end-to-end protocols, including TCP. The decisions made on this division will also affect the ease with which future versions of TCP can be implemented.

Finally, in order to achieve gateway control we need a new, specialised, gateway control interface. This must be designed in a common fashion for all gateways, and must incorporate what amounts to a special gateway control language.

These conclusions apply to the problem of connecting datagram nets to other datagram nets. Other problems, such as the interconnection of virtual circuit nets, or the connection of virtual circuit to datagram nets will require different solutions. This choice is the main limitation we have placed on the networks considered, and one way of interpreting our conclusions would be to argue that the problems we have raised show that we must re-examine the constraints we place on the structure of the networks we can interconnect.

REFERENCES.

Beeler 75: M. Beeler, J. Burchfiel, R. Rettburg, V. Strazisar, D. Walden - 'Gateway Design for Computer Network Interconnection'. October 1975.

Burchfiel 76: J. Burchfiel, W. Plummer, R. Tomlinson - 'Proposed Revisions to the TCP'. INWG Protocol Note 44, September 1976.

Cerf 74: V. Cerf, Y. Dalal, C. Sunshine - 'Specification of Internetwork Transmission Control Program'. INWG Note 72, December 1974.

Cerf 77: V. Cerf - 'Specification of Internetwork Transmission Control Program (Version 2)'. March 1977.

Cerf 77a: V. Cerf - 'Gateways and Network Interfaces'. PSPWN 65, April 1977.

Davies 72: D. Davies - 'The Control of Congestion in Packet Switched Networks'. IEEE Trans. on Comm., June 1972.

Edge 77: S. Edge - 'Evaluation Through Simulation of Possible Retransmission Acknowledgement and Flow Control Schemes for TCP'. INDRA Note in course of preparation.

Estep 77: J. Estep - 'Overview of SATNET Monitoring Centre Design'. PSPWN 61, March 1977.

Frank 75: H. Frank, I Gitman, R. van Slyke - 'Packet Radio System: Network Considerations'.

Guy 75: M. Guy - 'Bridging Protocol Specification'. EPSS Study Group 3 BIG CP(75)7, June 1975.

Kirstein 77: P. Kirstein - 'ARPANET Project: Annual Report (1 January 1976 - 31 December 1976)'. UCL TR34, February 1977.

Lloyd 75: D. Lloyd, M. Galland, P. Kirstein - 'Aims and Objectives of Internet Experiments'. INWG Experiment Note 3, February 1975.

McQuillan 74: J. McQuillan - 'Adaptive Routing Algorithms for Distributed Computer Networks'. BBN Report 2831, May 1974.

Pouzin 76: L. Pouzin - 'Flow Control in Data Networks: Methods and Tools'. INWG Note 118, April 1976.

Sunshine 75: C. Sunshine - 'Interprocess Communication Protocols for Computer Networks'. SU-DSL TR 105, December 1975.