To allow better network utilization, Systems Network Architecture (SNA), the IBM data communications architecture, includes flow control procedures to guard against data overrun to devices and to prevent network congestion. The measurement of "congestion" used by SNA to regulate traffic flow is performed by various SNA products. This paper describes the flow control protocols in SNA and the implementation of these protocols in the Network Control Program (ACF/NCP/VS Release 3).

SNA flow control: Architecture and implementation

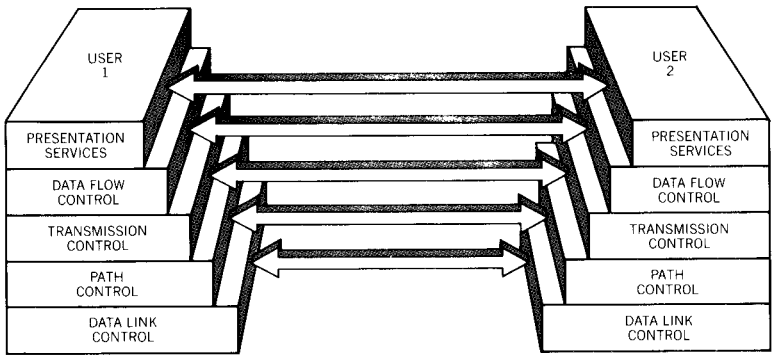
by F. D. George and G. E. Young

The cost of building and maintaining any computer communication network motivates the network owner to obtain the maximum throughput from the network. Paradoxically, if each network user transmits as many messages as the network can accept, the optimal service level will not be realized. Indeed, the network may deadlock and cease all data transmissions.[1,2]

The cost for a network designed to accept the maximum anticipated traffic from any source is prohibitive. The imposition of static limits to the amount of incoming data from each individual source might reduce the network cost, but the result would still be an ineffective network utilization. (For example, first-shift limits are not appropriate for off-shift traffic.) A better approach is to have the network dynamically regulate the load it may accept from an individual source, based on the over-all network utilization. A lightly loaded network can accept traffic more freely than a heavily loaded network in which scarce resources must be allocated fairly among all network users.

Systems Network Architecture (SNA) defines protocols for communication between various network components.[3–11] Through the use of these protocols, a compatible line of products has been developed by IBM to support distributed data processing. SNA is intended to maintain efficient network utilization without overrunning the receiving capacity of a device and without overcommitting a particular communication link or depleting network buffer capacity.

Figure 1 Layers of SNA



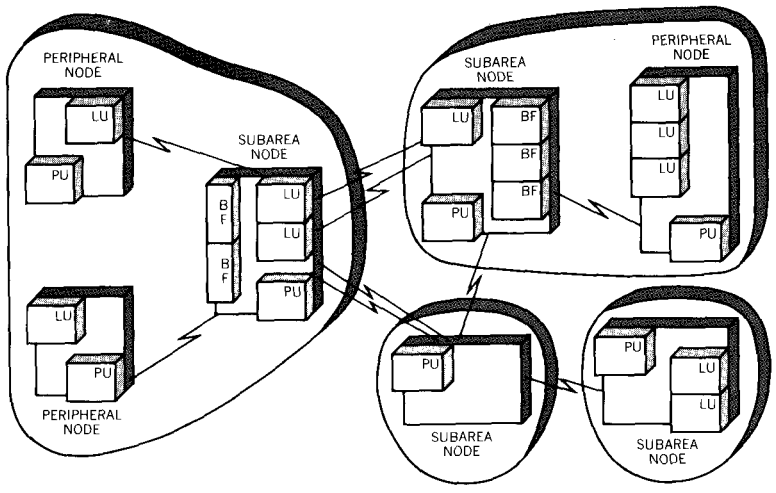
SNA defines flow control protocols to prevent the occurrence of the above situations while allowing as much data as possible to flow in the network. If one of these situations threatens, SNA selectively restricts traffic from contributors to the situation, rather than penalizing all network users. For this approach to be effective, however, each product that implements SNA must integrate the SNA flow control protocols within its own internal scheduling and resource management algorithms. This paper describes these SNA flow control protocols and their implementation in one IBM program product, the Network Control Program (ACF/NCP/VS Release 3).

SNA architectural overview

As a layered architecture, SNA defines the communication between peer-level components, as is illustrated in Figure 1.[10] In SNA, a network user (application program or terminal operator) accesses a network through a port by using a set of these components. The port is called a *Logical Unit* (LU). An LU resides in a network node, which itself is controlled by a *Physical Unit* (PU). A PU is responsible for managing a node's LUs and other resources such as communication links. (A glossary of abbreviations and terms used here is given in the Appendix at the end of this paper.)

Network nodes (and the corresponding types of PUs) defined by SNA can be classified into the following two broad categories: *subarea nodes* (which are PU types 4 and 5) and *peripheral nodes* (which are PU types 1 and 2). Subarea nodes participate in network routing to all other subarea nodes. Peripheral nodes attach to a subarea node to use its routing capability to the entire network. SNA allows multiple active links to connect each pair of subarea nodes simultaneously. These links are grouped into one or more *Transmission Groups* (TGs), each of which acts as a single logical connection between adjacent subarea nodes. A peripheral node attaches to a subarea node through a single link.

Figure 2 Sample network of subarea nodes and peripheral nodes



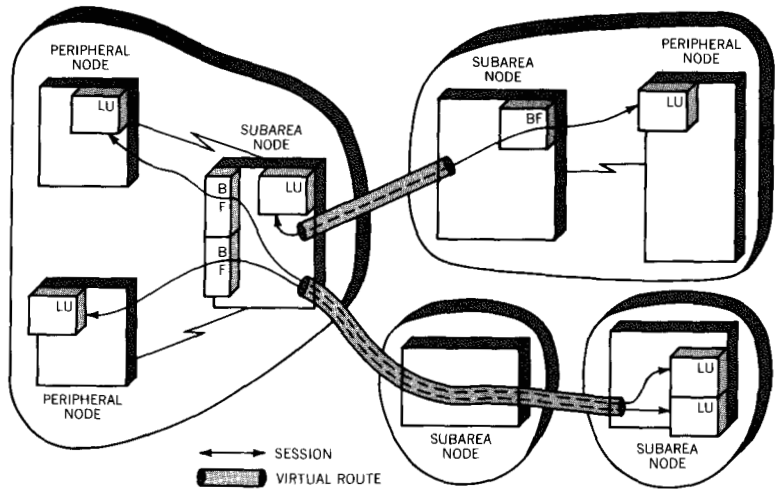
BF: BOUNDARY FUNCTION COMPONENT
LU: LOGICAL UNIT
PU: PHYSICAL UNIT

For each LU in a peripheral node attached to a subarea node, the subarea node allocates a *boundary function* component to support the attached LU. Collectively, a subarea node, the peripheral nodes attached to it, and all the resources controlled by PUs in these nodes comprise a *subarea*. Subareas, PUs, LUs, and boundary functions are the key components for flow control in SNA. Figure 2 depicts a sample network of these components.

One LU communicates with another LU through a *session*. SNA defines protocols for the activation, deactivation, and control of these LU-LU sessions. The LU that creates a session is called the *primary LU* of the session, and it always resides in a host subarea node. Hence, a boundary function is never required for a primary LU. The other LU (which cooperates in the session creation) is called the *secondary LU* of the session. Note that the primary and secondary roles are for a particular session; an LU could be primary on some sessions, secondary on others.

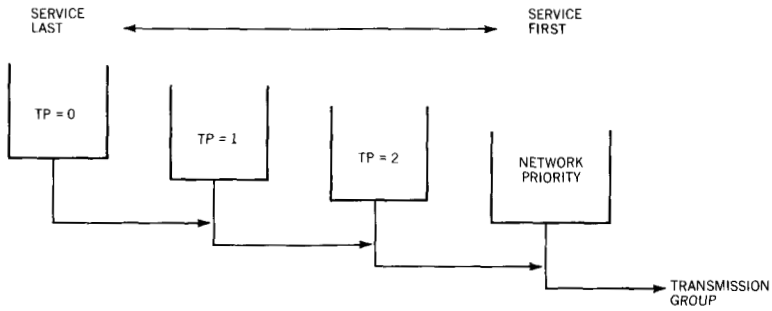
Subarea nodes in an SNA network with their interconnecting transmission groups form either a hierarchical or a meshed topology. A logical path defined through this topology is called a *Virtual Route* (VR). A VR begins, traverses, and ends only in subarea nodes; the connection between a subarea node and a peripheral node is not part of a VR. Each session is assigned a VR for its duration. A VR is activated when a session needs a path, and it is deactivated when the last session on the VR terminates. Figure 3 shows the relationship of sessions and VRs.

Figure 3 Sessions assigned to virtual routes



BF: BOUNDARY FUNCTION UNIT
LU: LOGICAL UNIT

Figure 4 Transmission priority determines network service



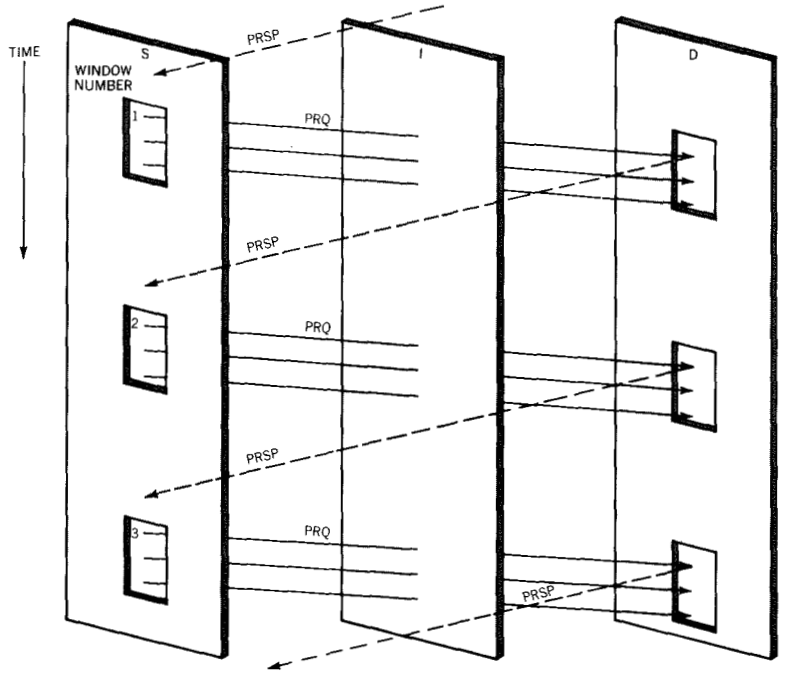
One of the characteristics of a virtual route is its *Transmission Priority* (TP). Transmission priority influences the service level of the messages of a virtual route relative to other messages in the network, i.e., the order in which messages are transmitted over a transmission group between two subarea nodes. In network congestion situations, SNA implementations tend to restrict lower transmission priority traffic first, which is consistent with the meaning of transmission priority. The use of transmission priorities in the flow control algorithms of NCP is discussed later in this paper. As shown in Figure 4, SNA supports three transmission priorities for user traffic, and an additional, higher network priority for certain supervisory traffic.

Pacing

pacing windows

SNA uses *pacing window* techniques for network flow control between corresponding layers of the architecture.[10] The pacing window form

Figure 5 Flow control with pacing windows

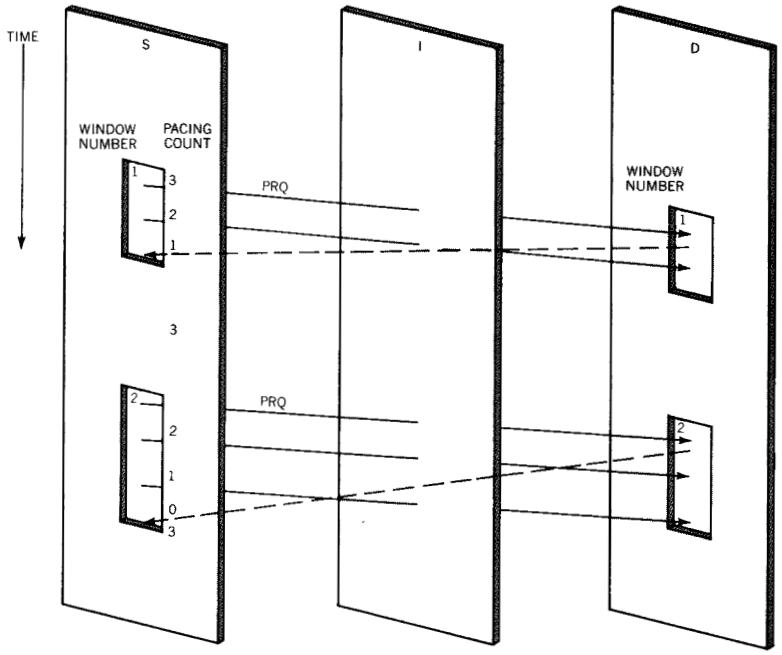


PRQ: PACING REQUEST
PRSP: PACING RESPONSE

of flow control permits only a certain number of sequential messages to be outstanding in the network between traffic end points before an end-to-end acknowledgment is received.[1] In SNA, the first message in a pacing window is called a *pacing request,* and it is explicitly tagged as such. After receiving a pacing request, the destination may send back a *pacing response,* which is an acknowledgment of its willingness to receive another pacing window. Figure 5 is a time diagram depicting the pacing windows from a source S to destination D, with a single intermediate point I. The pacing window size is three messages per window. Hence, S sends three messages through I to D, and then awaits the pacing response from D. D sends the pacing response upon receiving the pacing request (first message) from S. The pacing response is permission for S to send another pacing window of messages.

The destination may choose to withhold the pacing response under certain conditions. For example, if the receiver does not have the buffers necessary to receive another pacing window, the receiver temporarily withholds the pacing response. The traffic source cannot commence the transmission of a second window until the pacing response from the first window has been received, although completion of the transmission of the first window is allowed.

Figure 6 Reset pacing protocol



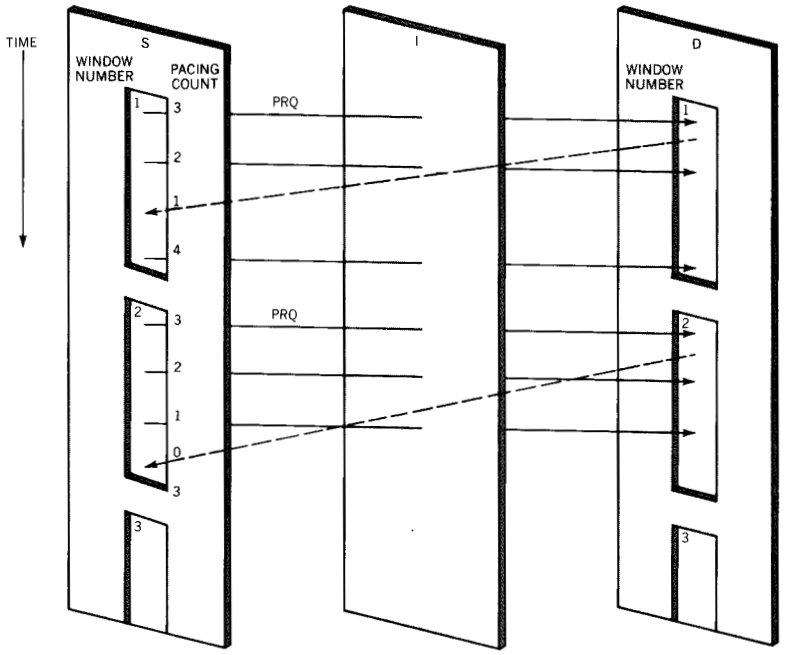
PRQ: PACING REQUEST

If a pacing response arrives at the traffic source before all the messages of the current window have been sent, two options are available to the traffic source. It may cease transmission in the current window, in which case it marks the subsequent message as a pacing request (the first message in a new window), and assumes permission to send one complete window. This is called *reset pacing.* Alternately, the traffic source may interpret the pacing response as permission to send an additional window after the current window is complete. This is called *cumulative pacing.*

The tracking of the pacing window in either type of pacing is the responsiblity of the traffic source. The destination needs only to remember whether a pacing response is owed. Hence, the *pacing count,* i.e., the number of messages that may be sent into the network, is maintained only by the traffic source. Figure 6 illustrates reset pacing, and Figure 7 illustrates cumulative pacing for a fixed pacing window size of three messages. The pacing count maintained by source S is shown for each technique.

Pacing techniques can be further classified according to the method by which pacing responses are transmitted. If a pacing response is sent as a separate message it is said to be isolated. Since, however, a

Figure 7 Cumulative pacing protocol

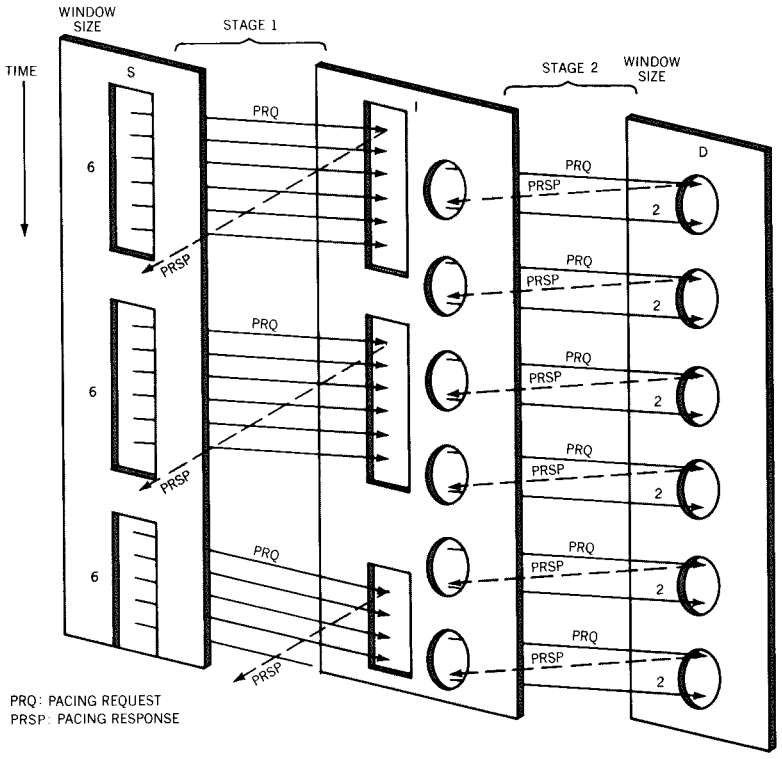


pacing response is trivial information to communicate, it may be combined with traffic flowing in the proper direction, in which case the pacing response is said to be piggy-backed.

Multiple instances of the basic pacing protocol described here are combined in SNA. Since traffic in only one direction is paced with the basic protocol, bidirectional transmissions (such as in virtual routes) have two instances of the pacing protocol operating, one for traffic in each direction. More elaborate combinations within each direction are also possible. Improved operation may result if segments of a path with different capacities and capabilities independently exercise instances of the basic pacing protocol with different window sizes. Each such segment with a pacing protocol is termed a *stage*.

Figure 8 depicts a two-stage pacing protocol in which one basic pacing protocol is executed from source S to intermediate point I, and another from I to destination D. Assume that the bandwidth is the same between the pairs of points, but that the delay from S to I is several times that from I to D. Also assume that D has limited buffer space. The pacing window size for the first stage is set to six to maintain the proper delivery rate through the segment with longer delay, and the pacing window size for the second stage is set to two to

Figure 8 Operation of two pacing stages



prevent overrun in small-buffered D. As depicted, I takes advantage of the larger pacing count from S so that data are always ready for D, even though D itself cannot handle a surge of six messages.

SNA paces both sessions and virtual routes, which differ in pacing window size, in the types of traffic paced, and in the number of pacing stages allowed in each direction. Table 1 lists the features of each type of pacing.

session pacing

Session pacing in SNA is an attempt to match the rate at which a network user can receive data with the rate at which another network user is transmitting.[1] If the transmitter is too fast, the transmission rate should be reduced rather than allowing the network to accumulate data, or worse, allowing the receiver to discard it for lack of buffers.

Session pacing uses fixed window sizes established at session setup. Session pacing may have up to two stages in each direction, each of which can be set appropriately for the particular network situation. The stages of session pacing are illustrated in Figure 9. For outbound pacing, the stages are from the primary Logical Unit (LU) to the

Figure 9 Stages for session pacing

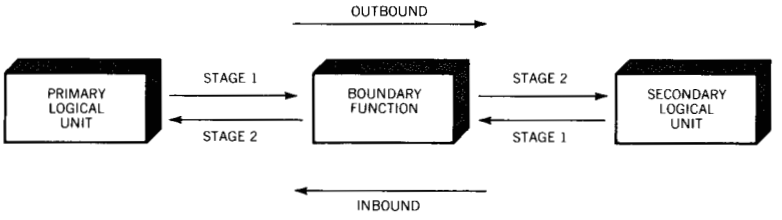


Table 1 Pacing protocols in SNA

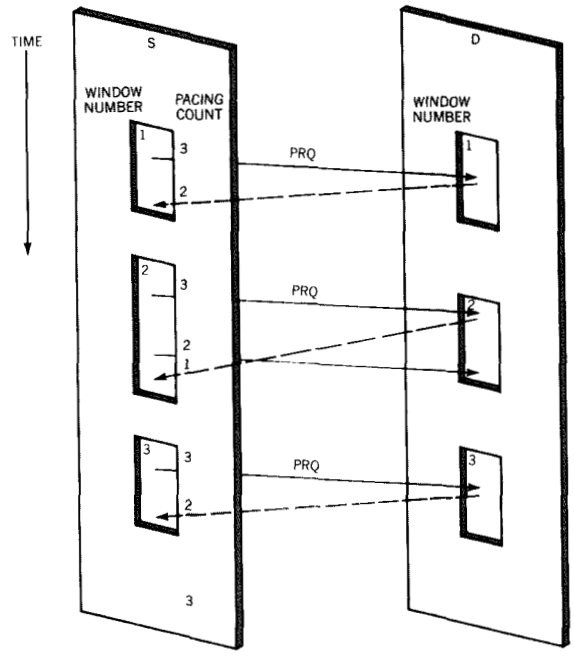
Characteristic	*Session*	*Virtual Route*
Window sizes	Fixed	Dynamic
Stages per direction	0, 1, or 2	1
Type	Cumulative	Cumulative
Pacing response	Isolated or piggy-back	Isolated
Paced traffic	Information messages between LUs	All except pacing responses
Size of units	Variable	Variable

boundary function, and from the boundary function to the secondary LU. For inbound pacing, the stages are from the secondary LU to the boundary function, and from the boundary function to the primary LU.

The first two stages in Figure 9 are referred to as "outbound pacing" because the primary LU resides in a host, and the traffic is being paced outbound from the host. Similarly, the last two stages are referred to as "inbound pacing" stages for traffic coming into the host. If a boundary function is not involved, only one pacing stage is used for each direction between the LUs.

The first outbound pacing stage is set to deliver data to the destination subarea at a rate that can keep the receiver occupied. The longer the network delay, the larger the pacing value required to maintain the proper data delivery rate. The second outbound pacing stage is set to prevent overrunning the local buffers of the secondary LU and is generally set to a lower value. If one considers these factors when setting the pacing values, the secondary LU always has adequate data without being overrun. The inbound pacing stages serve similar purposes, particularly if the secondary LU is a distributed data

Figure 10 Excessive pacing responses on reset pacing



application that is sending large quantities of data to a host application. The pacing stages protect the boundary function and the primary LU from overrun in this case. The boundary function, however, is not required to participate in the pacing. In this case, the pacing window sizes for the two stages are equal, and pacing responses are simply passed through the boundary function.

A session may run unpaced in either direction. The pacing protocol is negated by declaring a pacing count of zero. Unpaced sessions do not involve the overhead of pacing responses, yet these sessions need not threaten to overrun a device if each session is naturally paced. An example of such a session is an interactive query application with each request requiring a reply before the next request is made. The nature of such a session obviates a pacing protocol.

Session pacing is cumulative. The source of traffic for each stage maintains a pacing count of the messages allowed to be transmitted. Upon receiving a pacing response, the pacing count is incremented by an amount equal to the pacing window size of the particular pacing stage. Cumulative pacing avoids excessive supervisory traffic (isolated pacing responses) during relatively slack periods of terminal activity. If reset pacing is used, then—in low-session-traffic situations—almost every request is marked a pacing request and requires a pacing response. The excessive pacing responses on such a session

may contribute to an overall Virtual Route (VR) or network congestion situation, though this particular session is only lightly loaded. This undesirable reset pacing situation is depicted in Figure 10, where the window size of three is never exhausted. Hence, reset pacing is not used for sessions.

Session pacing responses can be isolated, or be combined (i.e., piggy-backed) with the header of response data flowing back to the source. Although it is advantageous to piggy-back, it is a rare luxury in practice. One reason is that paced sessions often involve large messages that do not request responses (acknowledgments) unless errors occur. Also, the node resources (such as buffer space) are often managed by a common node service that executes independently of the LU from which a response flows. Some implementations of SNA have decoupled the pacing from response generation and never use piggy-backing.

Session pacing regulates *data requests* only (i.e., informational message packets between the LUs). Control traffic (such as contention for session control) and all *responses* (i.e., acknowledgments to data or control requests) are not subject to regulation. Since the concern justifying pacing deals with data overruns at the primary or secondary LU, only data requests require explicit pacing control. The volume of all traffic on the session is naturally regulated by simply controlling the normal data traffic.

In early hierarchical SNA networks, transport network congestion (i.e., congestion in the subarea nodes of the network) could be controlled (1) through judicious settings of the pacing stages between the boundary function and the primary LU for all sessions, and (2) through restricting the number of sessions. As the networks grew in size and complexity, however, the delicate balancing of necessary data delivery to the boundary function and the avoidance of network congestion became precarious. This situation was further aggravated by the need to coordinate and control more and more sessions. With the advent of recent additional architecture, it has become possible to dedicate session pacing to "keeping the printer busy" by introducing another pacing protocol—*virtual route pacing*—for transport network congestion control.

virtual route pacing

Virtual route pacing in SNA prevents the overuse of the key network resources, i.e., buffers in the subarea nodes and capacities of the transmission groups.[12,13] Virtual route pacing has entry-to-exit controls exercised by the virtual route end points, with the subarea-to-subarea connection level control mechanisms exercised by nodes along the path of the virtual route, yielding a unique flow control protocol.[1]

Virtual route pacing uses dynamically varying window sizes with the size range established at virtual route activation.[10] (Recall that

virtual routes are activated as needed for sessions.) Although all implementations allow for user exits to modify the range of window sizes, the default range is h to $3h$ where h is the number of subarea-to-subarea connections between the end points of the virtual route. The default selection of h and $3h$ offers a significant reduction in response time compared to other obvious ranges (e.g., fixed windows of $3h$, dynamic windows of 1 and $3h$, and dynamic windows of $h-1$ and $3h$).[9]

Dynamic windowing allows the flow on the virtual route to be sensitive to changes in network load. In lightly loaded situations, large throughput is available for each individual VR. In periods of significant network resource competition, the network reduces traffic contributions from individual VRs to acceptable overall network traffic levels. The network attempts to regulate only the VRs contributing to a specific congestion situation, and to equitably restrict each such VR.

Virtual routes are always paced. Both directions of the VR are paced independently and may have radically different pacing window sizes at any given moment, although the range of pacing window sizes is the same for both directions. Such an imbalance occurs if there is contention for network resources in one direction (e.g., when a full-screen application writes to many terminals), but relatively little competing traffic in the opposite direction (e.g., only short queries from a few terminals).

Virtual route pacing is cumulative, as is session pacing. The rationale for reducing the frequency of pacing responses in low traffic situations applies equally to virtual route pacing. A pacing count is maintained to track the number of messages that can be sent at any given time. The pacing count of a newly activated VR is set to zero. A pacing response is required from the receiving subarea node before any data are allowed to flow. This precaution allows a subarea node to activate (or allow activation of) a VR without concern for immediate resource commitments associated with the VR.

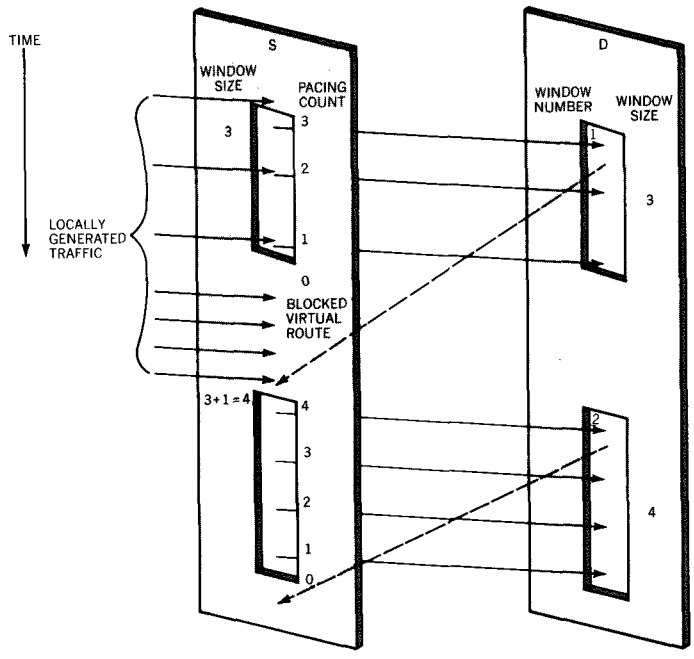
Virtual route pacing responses are transmitted as isolated pacing responses. The network priority is used for these isolated pacing responses to minimize their transmission time. Network priority ensures that the pacing responses pass any other data in the network that are contending for the same links. As explained later in this paper, it is beneficial to the window adjustment algorithm that pacing responses take minimal time.

Congestion

network congestion determination

The window size at VR activation is the minimum window size, and it is successively increased as required until excessive network buffering

Figure 11 Blocked virtual route



begins. This excessive buffering is called network *congestion.* After this point, the window size is continuously adjusted while the VR is active. Decreases in the window size occur only when there is congestion; increases occur only when there is both a need and no congestion. Need is determined as follows.

If the pacing count for a virtual route is exhausted, that VR is considered to be *blocked.* This condition occurs if a VR accumulates more traffic from sessions than it can send within its current pacing window and has not received the VR pacing response allowing another window before the pacing count is exhausted. The VR is obliged to queue all pending messages until the VR is unblocked by the arrival of a pacing response. Implementations of this mechanism take steps to restrict traffic sources that are feeding a blocked VR so as to minimize the VR pacing queue depth. Figure 11 depicts a blocked situation when the pacing window size is three. Since more traffic is arriving at the VR than can be transmitted under the current window size, and since there is idle transmission time, the VR can support a larger window. If congestion is not detected and if the window size is not at its maximum value, the window size is increased by one for the next window.

Having data queued for transmission by a VR (i.e., even more messages than the window size) is not equivalent to being blocked. In

Figure 12 Transmission-limited virtual route

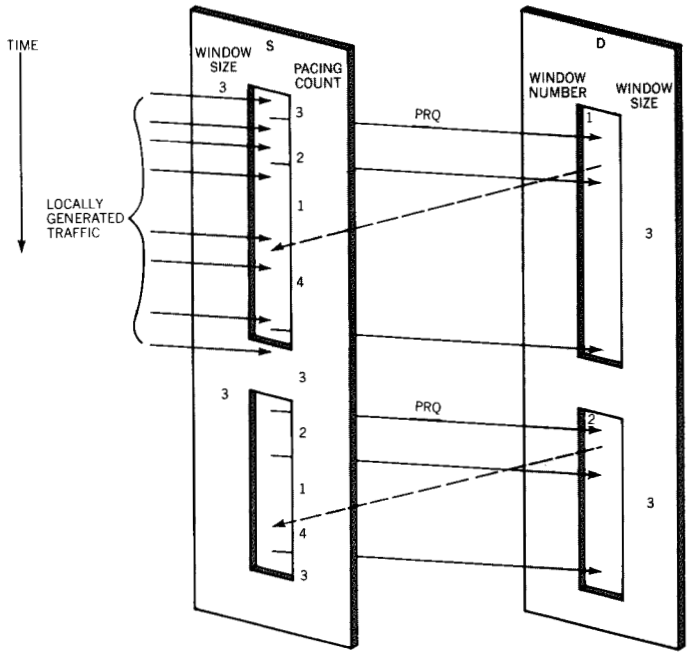


Figure 12, the rate of arrival of messages for the VR has exceeded the transmission rate from the node. In this case, however, there is no need for an increase in the window size since the network is transmitting as fast as possible. A window size increase does not affect the transmission rate.

The need for a "fast" pacing response can be observed from Figure 11. A slowly propagating pacing response may create an artificial idle period for transmission, thereby encouraging a larger window size. A faster pacing response maintains constant data transmission with smaller window sizes, and does not introduce the potential for unnecessary queuing into the network.

reactions to congestion

The reaction to a congestion situation varies with the severity of the congestion. The determination of the severity of the congestion is left to the various implementations of SNA. The congestion criteria for the Network Control Program (NCP) are given later in this paper. Some subarea congestion situations can be relieved through manipulation of a single VR or a couple of contributing VRs. Other congestion situations require actions affecting all VRs.

In mild congestion cases, a subarea node can reduce the window size of a VR by the quantity one. This is the appropriate action for a VR

Table 2 Summary of congestion situations and actions

Situation	*Technique*	*Results*
Moderate congestion	CWI, CWRI in transmission header	Reduce next window by one
Severe congestion	RWI in transmission header	Reduce next window to minimum window size
Early buffer depletion	Withhold pacing response	Stop transmission on VR after current window
Severe buffer depletion	Cease receiving data	Stop all windows at current point

CWI: Change Window Indicator
CWRI: Change Window Reply Indicator
RWI: Reset Window Indicator
VR: Virtual Route

with a pacing window that has been gradually increasing from the initial minimum value to the point at which it becomes too large, resulting in congestion. This is also the appropriate action for a VR with a stable pacing window size, when that VR begins to experience resource competition from newly activated virtual routes.

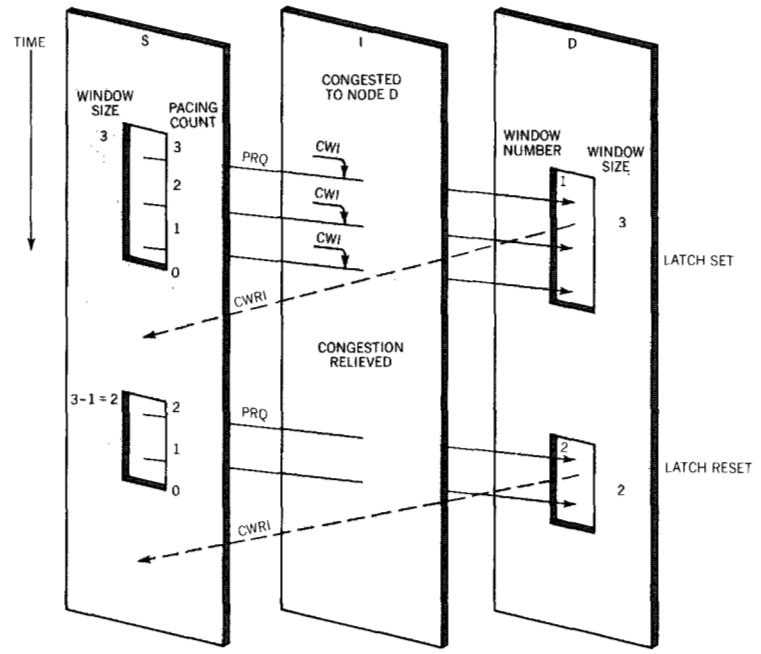
A slightly stronger congestion control involves a subarea node causing the immediate reduction of the pacing window size of a VR to its minimum value. If many VRs are active, with occasional light traffic surges, pacing window values may have been increased toward maximum levels. If sudden, steady traffic occurs on many VRs at once, network congestion can occur. A congested subarea node can immediately reduce the VR window sizes to minimum values (rather than by one message per window for several windows), thereby providing immediate relief at the point of congestion. After this reduction, windows can gradually increase in size toward a new point of network traffic balance.

If an entire subarea node is in danger of buffer depletion, it can withhold the pacing response entirely for all VRs for which it is an end point, effectively freezing each VR transmitter at the end of the current window. When congestion is sufficiently relieved, the pacing responses can be forwarded to the VR end points.

The strongest action taken for overall buffer depletion in a subarea node is the cessation of receiving any traffic by the node. This causes subsequent messages, even in current VR windows, to be queued by nodes adjacent to the congested node until the situation can be relieved, usually by transmitting data that the congested node itself has queued.

Table 2 summarizes the various congestion situations in order of increasing severity. For each one, the technique involved and the

Figure 13 Invocation of CWI/CWRI protocol



PRQ: PACING REQUEST
CWI: CHANGE WINDOW INDICATOR
CWRI: CHANGE WINDOW REPLY INDICATOR

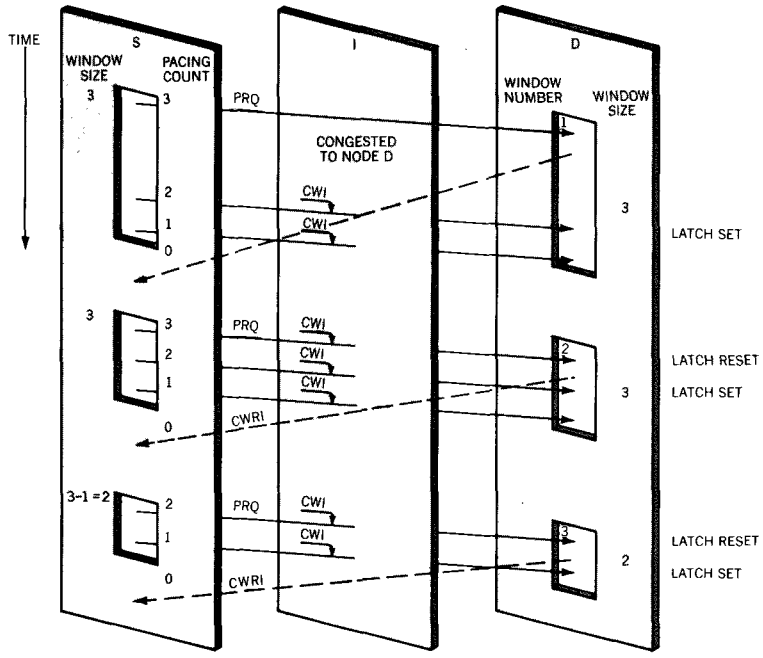
effect on network traffic are shown. The first two techniques involve the use of special indicators carried in the transmission headers of the data flowing in the network—Change Window Indicator (CWI), Change Window Reply Indicator (CWRI), and Reset Window Indicator (RWI). The protocols involving their use are now described in detail.

decrement window protocol

If mild congestion is detected on a transmission group by a subarea node in the network, the traffic passing from that subarea node on that transmission group is tagged with the Change Window Indicator (CWI). The receiving subarea node maintains a latch for each VR to track the appearance of a CWI. The latch is always reset after a pacing response is sent to the transmitting subarea node. The latch is set if a message is received tagged with a CWI, or if the receiving subarea node itself is mildly congested. If the latch is set when a pacing response is to be sent, the subarea node tags the VR pacing response with the Change Window Reply Indicator (CWRI). The transmitting subarea node decrements the window size when it receives a pacing response tagged with a CWRI (but it never decrements the window size below the minimum value).

As long as the congestion persists, traffic is continuously tagged with CWIs. The CWRI sent in reply to a CWI decrements each successive

Figure 14 Delay in CWRI-induced window reduction



PRQ: PACING REQUEST
CWI: CHANGE WINDOW INDICATOR
CWRI: CHANGE WINDOW REPLY INDICATOR

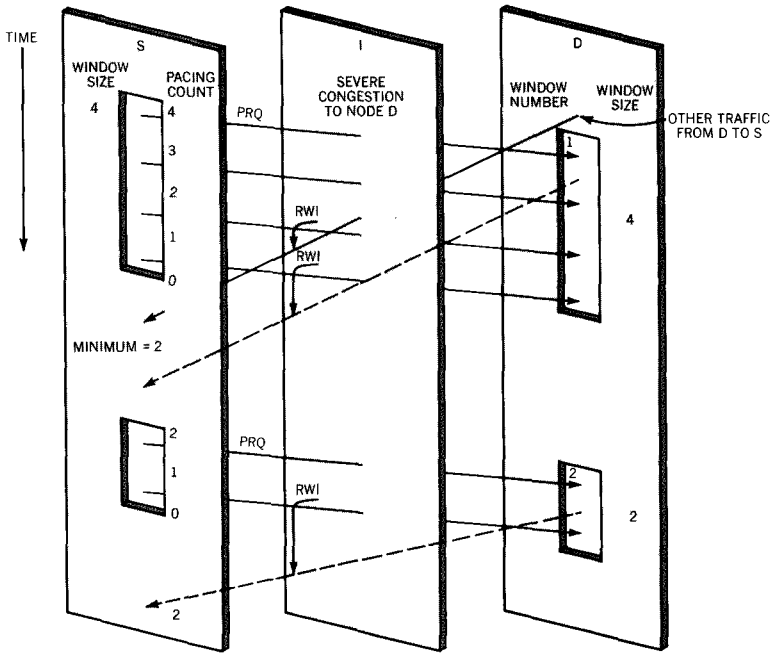
window by the quantity one. Figure 13 gives an example of a node (node I) that is suffering congestion. Node I sets CWIs on messages flowing from node S to node D, and eventually causes the window size for the VR from node S to be decremented from the value of three to two.

A delay of one window can occur before a CWRI-induced reduction is effected. In Figure 14, the pacing response associated with the current window has already been sent by node D before a message tagged with a CWI is received. The next pacing response (in which a CWRI may be set) is not sent by node D until the first message in the next window is received. Hence, a CWRI set in this later pacing response affects the size of the window after the next window.

reset window protocol

If more severe congestion is detected on a transmission group by a subarea node than might be dissipated by the CWI/CWRI protocol, the subarea node begins to tag messages with the Reset Window Indicator (RWI). As the name indicates, this action causes an immediate reduction in the window size to the originally established minimum value. The RWI protocol differs from the CWI/CWRI protocol in that the congested subarea node tags traffic with the RWI that it receives over the congested transmission group. Traffic so tagged is flowing in the direction opposite that of the traffic causing the congestion. The

Figure 15 Invocation of RWI protocol



PRQ: PACING REQUEST
RWI: RESET WINDOW INDICATOR

received traffic serves as the most expedient vehicle for notifying the sources that are contributing to the congestion situation. Since virtual routes are always paced, at least virtual route pacing responses are being received over the congested transmission group for tagging with an RWI. The reduction in the pacing window size, however, usually takes place without waiting for the VR pacing response.

When a subarea node receives a message on a VR with the RWI set, that node sets its own pacing window size for the VR (as a transmitting subarea node) to the minimum value, even though it is the receiving subarea node for the traffic tagged with the RWI. Figure 15 illustrates the reaction to an RWI situation in subarea node I. Here, the traffic from node S is throttled from 4 to a minimum window size of 2 by node I tagging traffic flowing to node S with the RWI.

One additional difference between the RWI protocol and the CWI/CWRI protocol is that the RWI protocol causes a reduction of the current pacing count (in addition to the reduction of window size) to the minimum window size if the current value of the pacing count is greater than the minimum window size value. This occurs if the pacing window size has increased toward the maximum value and light traffic demands have caused very little use of the current pacing count. If the pacing count has already dropped below the minimum pacing window size, the count is not modified when an RWI is detected.

Network Control Program implementation

The Network Control Program[14] (ACF/NCP/VS Release 3, which we refer to as NCP) performs as a subarea node in an SNA transport network. NCP, which provides an attach point for terminals and clusters in the network (peripheral nodes), is an IBM program product that executes in the IBM 3705 Communications Controller. NCP supports meshed connectivity of its 3705 controller with other 3705 controllers through multilink transmission groups, and may be channel attached to multiple host nodes.

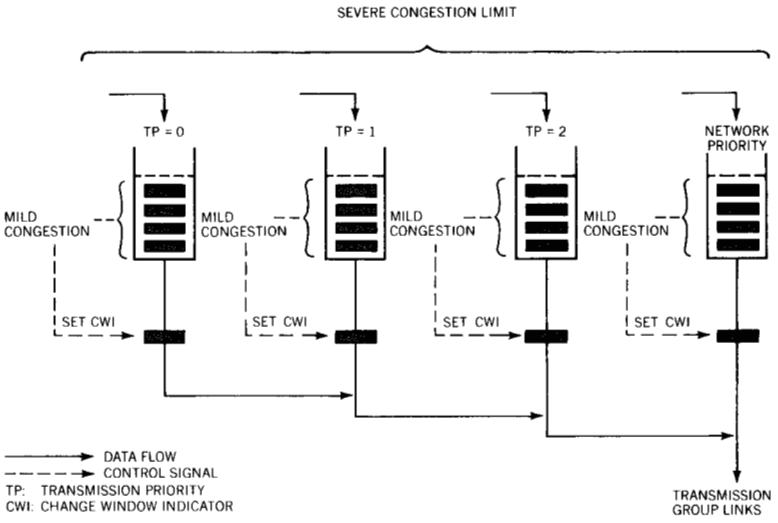
Other subarea node implementations include Virtual Telecommunications Access Method (ACF/VTAM Release 3) and Telecommunications Access Method (ACF/TCAM Version 2 Release 3), both of which execute on System/370 architecture hosts. NCP has been selected to illustrate the SNA flow control implementations, since it serves as both an end point and an intermediate node for virtual routes, whereas VTAM and TCAM serve only as VR end points. Also, NCP is more buffer constrained and is therefore more subject to congestion.

NCP session pacing

As a subarea node, NCP has implemented SNA flow control procedures. With peripheral nodes attached, NCP participates in session pacing through its boundary functions (supporting the LUs of the peripheral nodes). A boundary function in NCP serves as a pacing-stage end point for outbound pacing to an LU in an attached peripheral node. If one outbound stage is paced, both outbound stages must be paced. NCP allows the specification of one parameter beyond those that are required by SNA: the designation of which message in a window is marked as the pacing request. (SNA specifies that the first message in a window be tagged as the pacing request.) This NCP parameter is applicable only to the second outbound pacing stage (boundary function to secondary LU) and defaults to tagging the first message as the pacing request. This extension was originally part of SNA, and NCP has retained the function from its implementation of that original architecture.

A boundary function in NCP also serves as a pacing stage end point for inbound pacing from an LU in an attached peripheral node, but in a more restricted fashion than outbound pacing. NCP requires that the values for the inbound pacing stages be equal, as would be the case for single-stage inbound pacing. Yet the NCP boundary function may intercept and withhold the session pacing response flowing from the primary LU to the attached secondary LU. This withholding may be necessary to prevent congestion in the NCP by not soliciting further data from attached secondary LUs. NCP takes this action on each session associated with a blocked virtual route. When the virtual route is no longer blocked, NCP forwards the session pacing response to the attached secondary LU.

Figure 16 Queues associated with a transmission group



NCP virtual route pacing

NCP has implemented virtual route pacing. When examining this support, it is helpful to classify the role of NCP into three categories:

- Intermediate node along the path of a VR.
- Origin node of a paced VR flow.
- Destination node of a paced VR flow.

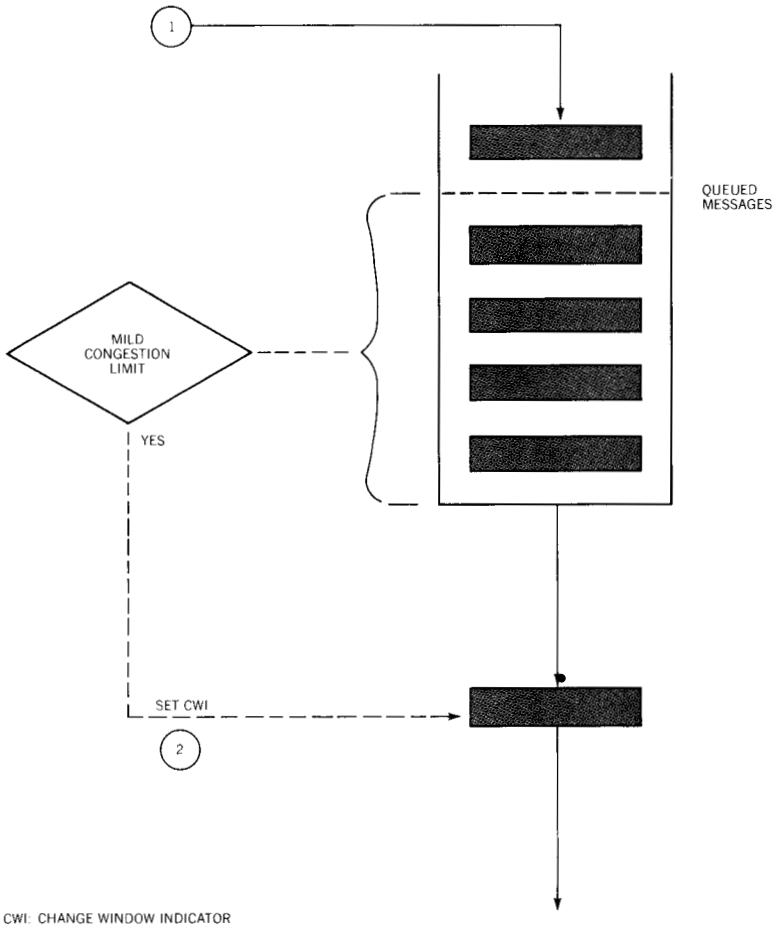
The actions in each of these environments are somewhat independent, and acting as the destination node of a paced VR flow involves complex management of all demands for NCP buffers.

intermediate node support

If an NCP lying on the path of a virtual route is not one of the end points of the virtual route, that NCP serves as an *intermediate node* for that virtual route. When NCP is acting as an intermediate node, only the flow control procedures CWI/CWRI and RWI are exercised, as indicated in Table 2.

As an intermediate node, NCP should buffer data only to the extent needed by the link protocols. Excess data that might otherwise be queued by NCP are prevented from entering the network through pacing window size adjustments. Toward this end, NCP uses buffer availability levels (in bytes) as the criterion to invoke SNA flow control protocols. NCP maintains four logical transmission queues for each transmission group, one for each of the three possible transmission priorities and the fourth for network priority, as shown in Figure 16. (Actually, NCP has a single transmission group queue, and inserts a message by scanning the single queue for the appropriate point of insertion of the new message. Higher-priority messages are enqueued

Figure 17 CWI protocol invoked for a queue



CWI: CHANGE WINDOW INDICATOR

before lower-priority messages.) The buffering limits are tracked separately to an adjacent subarea node for each transmission group.

mild congestion

NCP defines mild congestion for a transmission priority queue as the situation in which the sum of the message sizes on the given transmission priority queue exceeds an established mild-congestion limit, as indicated in Figure 16. Once the queue size exceeds that limit, each message removed from the queue is tagged with a CWI. This tagging continues until the queue size drops below the limit.

The received message that causes the limit to be exceeded is not necessarily flagged with a CWI; rather, the next message dequeued for the congested transmission priority is flagged with a CWI, as shown in Figure 17. At step 1, a newly arrived message is enqueued with other messages of the same transmission priority. The message

causes the mild-congestion limit to be exceeded. Step 2 shows the oldest message being dequeued and tagged with a CWI, since the CWI threshold has been exceeded.

severe congestion

Severe congestion for a transmission group occurs when the sum of the message sizes on the queues for all transmission priorities associated with a transmission group (including network priority) exceeds an established severe-congestion limit, as illustrated at the top of Figure 16. Once this limit is exceeded, every inbound message from the congested transmission group is tagged with RWI. This procedure continues until the aggregate queued traffic for the transmission group is less than the severe-congestion limit.

Using these two flow control protocols in conjunction with the handling of transmission priority, observations on the dynamics of flow control in NCP can be made. Assume the network has reached a stable point, with pacing windows appropriately adjusted. If high-priority traffic is suddenly increased (to accommodate a new VR, for example), lower-priority traffic is serviced less by the transmission group. Yet the lower-priority VRs continue to contribute at their previous rates. As queues develop in NCP for the lower-priority traffic, the CWI/CWRI protocol begins to throttle this traffic by reducing pacing window sizes, thus moving toward a new network traffic mixture.

If the gradual throttling of lower-priority traffic is not sufficiently quick, the total buffer space of NCP is threatened. This situation, triggered by the severe-congestion threshold, causes all traffic contributing to the threat to be throttled by RWI tagging. Upon relief of the threat to NCP, the pacing windows may increase to a stable level. Hence, if gradual change under the CWI/CWRI protocol is not sufficient for handling a radical traffic shift, the more radical RWI protocol is invoked, which will reduce pacing window sizes to the minimum for all priorities within the transmission group.

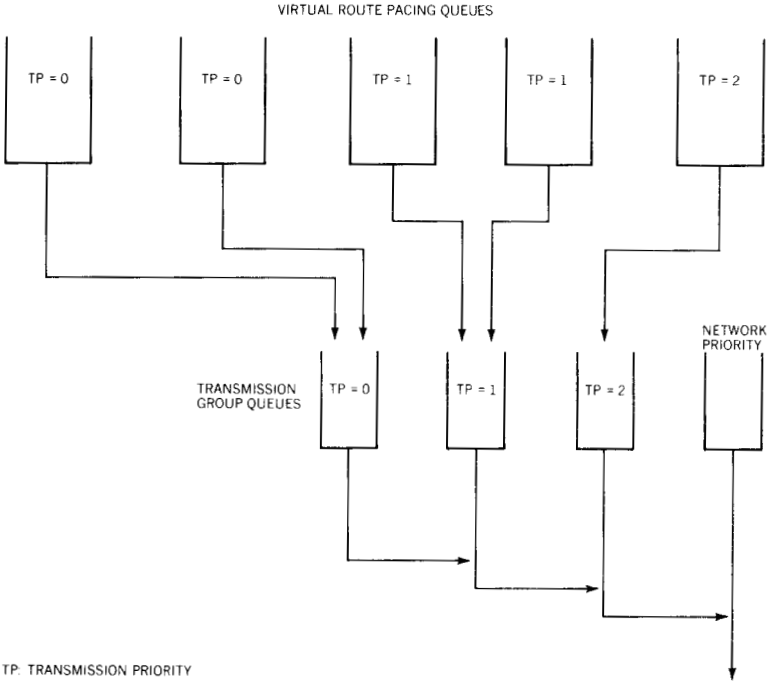
data origin support

As an originator of virtual route data, NCP attempts to send as much data into the network as possible to provide the best throughput for its attached devices. The data flow from NCP is constrained only by the receptiveness of destinations as expressed through flow control protocols.

If the destination node or an intermediate node does not return a VR pacing response, NCP ceases transmissions on that VR at the completion of the current window. Data received after that point are queued in the VR pacing queues. NCP then begins to restrict its own internal traffic sources, as described later in this paper.

If the node adjacent to NCP refuses to receive the data, NCP accumulates the data in its transmission group queues (mixed with intermediate node traffic) until the VR pacing counts are exhausted.

Figure 18 NCP queuing structure



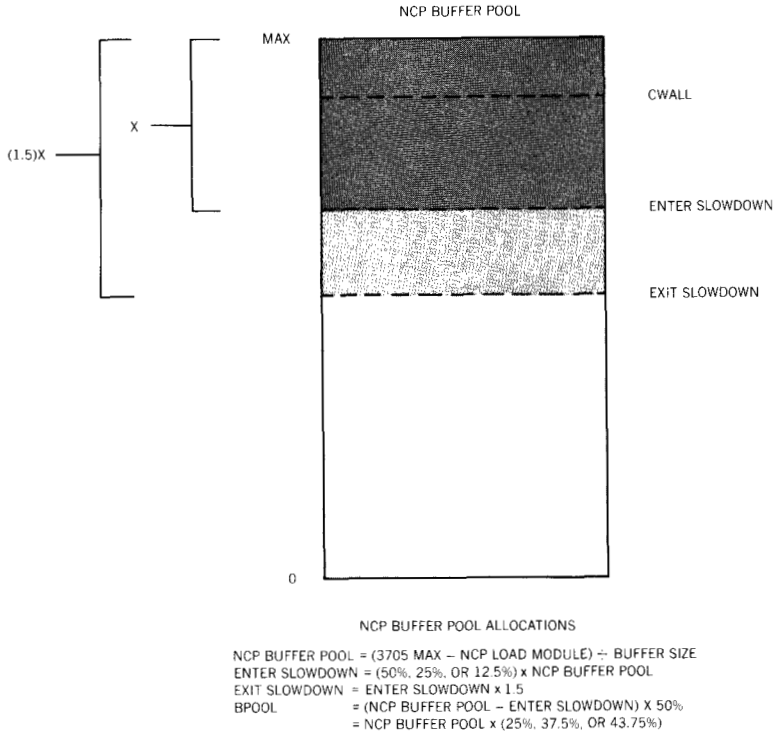
After that point, data are accumulated in VR pacing queues, since VR pacing responses cannot arrive. Figure 18 depicts both the transmission group queues and the VR pacing queues in NCP.

If NCP receives a message on a virtual route that is tagged with an RWI, NCP immediately sets the VR pacing window size to the minimum value established at VR activation. (The default is *h*, i.e., the number of subarea-to-subarea connections.) No manipulation of the current pacing count is made, however.

Upon the arrival of a VR pacing response, pacing window size manipulation is also performed by NCP. If the pacing response is tagged with a CWRI, the pacing window size is decremented by one (but not lower than the minimum pacing window size). If the pacing response is not tagged with either a CWRI or an RWI, and if the VR is blocked (pacing count currently exhausted), the pacing window size is increased by one (but not higher than the maximum pacing window size). After appropriate pacing window size manipulation (increase, decrease, or no change), the current pacing count is incremented by the pacing window size.

If the network restricts the data flow from NCP, messages accumulate in NCP itself. This accumulation affects the overall buffer availability of NCP, and throttling mechanisms for restricting device data input may be invoked. These mechanisms are described in the following section.

Figure 19 NCP buffer pool with key thesholds



data destination support

As the receiver of data from both the subarea nodes (via VRs) and the peripheral nodes directly attached, NCP constantly tracks the state of its buffer pools. Before discussing when NCP uses the SNA flow control protocols, NCP buffer management terminology and the critical congestion thresholds are defined.

After NCP is loaded into a 3705 communications controller, all remaining storage of the 3705 is allocated for buffering data. Thus the amount of buffer space varies from one node to the next, and is directly influenced by the size needed for the NCP (unnecessary functions are not loaded) and by the size of the 3705 in which the NCP resides.

Figure 19 represents a memory map of the total buffer pool of NCP, with several thresholds marked. Assuming that buffers are allocated from the bottom up, the following thresholds are established at NCP generation:

- *CWALL* or communications wall is specified as a number of buffers available, with a default of 8 buffers.
- *Enter slowdown* threshold is set at a percentage of the total buffers at initialization of the NCP. Only three values are valid—50, 25, and 12.5 percent of the buffers.

Table 3 BPOOL region congestion reactions

Region	*Maximum Percent of BPOOL Used*	*Set RWI for*	*Withhold VR Pacing Response*
1	67.5	no TP	no TP
2	75.0	TP = 0	no TP
3	87.5	TP = 1	TP = 0
4	100.0	TP = 2	TP = 0,1
5	>100.0	no TP	TP = 0,1,2

RWI: Reset Window Indicator
TP: Transmission Priority

- *Exit slowdown* threshold is calculated from the enter slowdown threshold, and it is also a percentage of the total buffers. The enter slowdown threshold is simply multiplied by 1.5 to arrive at the percentage for the exit slowdown threshold.

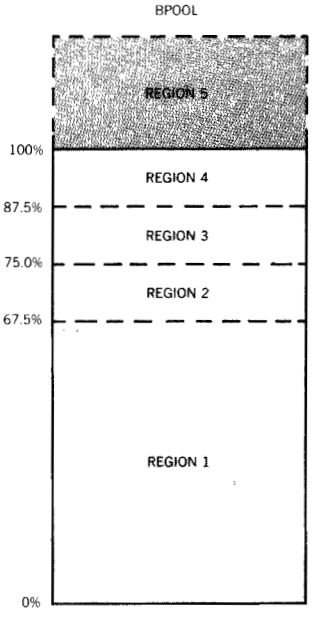
CWALL is the minimum number of buffers needed by NCP to communicate with an attached device. If fewer than the CWALL-specified buffers are available, the CWALL threshold is considered to be exceeded. When more than the CWALL-specified buffers are available, the CWALL condition no longer exists.

If one less than the enter-slowdown-specified buffers are available, NCP is in a *slowdown state.* The slowdown state is left when more buffers are available than the exit slowdown threshold requires. Having separate thresholds for entering and exiting the slowdown state ensures that the conditions that induced slowdown are truly past before the slowdown-induced congestion procedures are terminated.

For virtual route pacing, a logical subpool of the NCP buffer pool is established. The size of this subpool, called the BPOOL, is established at one-half the buffers of NCP before its entering-slowdown threshold is reached. Thus the BPOOL size is 25, 37.5, or 43.75 percent of the total NCP buffer pool. (This corresponds to enter-slowdown thresholds of 50, 25, and 12.5 percent.) The BPOOL is subdivided into regions, with regional demarcations as indicated in Figure 20. Note that region 5 can extend beyond 100 percent, since the BPOOL is a subpool of the total NCP buffer pool.

If a message is received by NCP from a transmission group and if that message is for that NCP or an attached peripheral node, NCP logically allocates BPOOL buffers to contain the message. By totaling all such BPOOL buffers, the region of the BPOOL in which NCP is operating is determined. Based on the region, NCP invokes the RWI protocol or withholds pacing responses. As summarized in Table 3, the lower

Figure 20 BPOOL used for VR pacing with regions shown



Transmission Priorities (TPs) are penalized first in congestion situations. This is consistent with the treatment of transmission priorities in the network to provide preferential service to higher transmission priorities.

The NCP buffer algorithms and thresholds having been described, the implementation of the SNA flow control procedures can be succinctly stated. If NCP receives a message on a virtual route that is tagged with a CWI, NCP sets the CWRI in the next VR pacing response on that VR, as prescribed by SNA. The next pacing response is not tagged with a CWRI unless another message has been received with CWI since the last pacing response was sent.

NCP sets an RWI in a message if the BPOOL rules in Table 3 indicate that that action should be taken (transmission priority of message matches the appropriate BPOOL region), or if the severe-congestion limit described previously is exceeded for the message's transmission group.

NCP withholds a VR pacing response if the BPOOL rules indicate that action or if NCP is in slowdown. Once both of these conditions have passed, the withheld VR pacing response is transmitted.

related NCP flow control procedures

NCP can support many active links to both subarea nodes and peripheral nodes. To avoid overcommitting available buffer resources, NCP may temporarily refuse to receive data from attached peripheral nodes or adjacent subarea nodes. For peripheral nodes, the refusal to receive data is made by not polling the node. For adjacent subarea nodes, the refusal is made by rejecting any data transmitted by the subarea node. The refusal to receive data does not alter the ability of NCP to send data to those same nodes to relieve its own congestion.

There is one receiving-restriction procedure—called "commit trigger to slowdown"—that is applied to all peripheral nodes and all link-attached subarea nodes (but not to channel-attached subarea nodes). Under this procedure, an estimate is made of the number of buffers required to service each adjacent node. This number is the greater of either the number of buffers needed for the node as estimated by the user or the number of buffers required to service the node on the last successful receiving operation. Once a receiving operation is allowed, the estimated count of buffers needed for servicing the node is accumulated for threshold value comparisons.

Using the estimate of the results of a successful receiving operation, NCP does not accept data from a particular node if by receiving data an NCP receiving threshold would be passed by more than one buffer. The receiving threshold for refusing data varies with the type of connection. Peripheral nodes are not polled while NCP is in the

slowdown state, and data from subarea nodes are rejected if the CWALL threshold is passed. Actually, the comparisons are not made directly against the threshold; rather, the threshold is lowered by the number of buffers committed to receiving, and then the modified threshold is compared with the number of buffers in use before making a commitment to receive more data. With this procedure, NCP attempts to avoid congestion situations altogether.

There are two exceptions to the decision to refuse data. First, if a message has been divided into multiple logical parts (called *segments*) for transmission and if the last segment of a message from a peripheral node has not yet been received, NCP continues to receive from that node until the last segment of the message is detected. Second, if a particular transaction is still in progress as indicated by the *bracket* state of a session, receiving continues for that node until the transaction is complete. With both of these exceptions, the completion of the activity stands to relieve congestion by freeing resources already committed to the activity.

If the commit-trigger-to-slowdown mechanism prevents data from being received from a node, that node is queued for later service until the appropriate threshold situation is relieved. There are two separate service queues, one for subarea nodes and another for peripheral nodes, with preference given to the service queue for subarea nodes. Data from a subarea node are solicited if the subarea node service queue is empty, whereas data from a peripheral node are solicited only if both service queues are empty.

When a resource-committed receiving operation completes, appropriate adjustments are made to the available-buffer count and to the committed-buffer count to reflect the actual data received versus the amount anticipated.

NCP does not estimate the data that might be received across a channel from an adjacent subarea node. Rather, whenever the buffers in use exceed the CWALL threshold, NCP simply ceases receiving data across any channel. Receiving resumes when the CWALL condition is relieved.

Another receiving restriction procedure is invoked by NCP when a virtual route becomes blocked. In this situation, NCP restricts local traffic sources that are contributing to the pacing queue of the blocked VR. Two mechanisms are used, sometimes in combination. In one case, if a paced session is feeding the blocked VR, the NCP intercepts and withholds any session-level pacing responses that are flowing to the attached LU on that session. This effectively restricts a source of traffic to the blocked VR at the end of the pacing window for that session. By the other mechanism, NCP can refuse to poll a node whose LUs have sessions that are using the blocked VR. This

completely eliminates a traffic source, but it may also restrict traffic from an LU that does not use the blocked VR.

Concluding remarks

SNA provides flow control protocols for both sessions and virtual routes, based on pacing window techniques. Although the same mechanism is used in both protocols, the purpose of each pacing protocol is different. To match the rate at which an LU can receive data from another LU, bidirectional pacing of the session between LUs is defined in SNA. The two pacing stages in each direction allow traffic-rate matching for transmission both through the backbone network (of subarea nodes) and to the attached peripheral node.

For protecting the transport network from congestion, SNA defines a pacing protocol on a transport network construct, the virtual route. By manipulating its virtual routes, a subarea node exercises any one of four separate mechanisms to reduce congestion, depending on the severity of the situation. Although basically an end-to-end protocol, virtual route pacing is also influenced by intermediate subarea nodes along the path of the virtual route, a unique SNA combination.[1]

An implementation of an SNA subarea node incorporates the SNA flow control protocols into its own resource management algorithms. As an intermediate node, NCP defines buffer limits for each transmission group as its criterion for congestion. As an end point for virtual routes, NCP uses elaborate buffer monitoring techniques for equitably restricting traffic sources when congestion occurs. The traffic restriction is accomplished through combinations of virtual route congestion mechanisms, session pacing interruptions, and selective cessation of polling activities.

The interactions among these mechanisms further indicate the capabilities of SNA flow control protocols when they are applied in concert with the resource scheduling algorithms of a particular implementation. If the network is not accepting traffic at the rate required by NCP, virtual routes become blocked. NCP then begins to withhold session pacing responses destined for LUs in attached peripheral nodes. In cases where sessions with LUs in attached peripheral nodes are not paced, cessation of polling is invoked. By combining the capabilities of these flow control protocols, NCP is able selectively to restrict traffic sources contributing to congestion with a granularity not available with a single level of flow control.

As SNA users' networks continue to grow in size and complexity, enhancements to the SNA flow control procedures will evolve. Static calculations of flow control parameters will become less reliable and will be replaced with distributed, reactive flow control algorithms that are more sensitive to the current status of the network. Although

these distributed algorithms will probably be intricate, they relieve user specifications of complex flow-control parameters.

ACKNOWLEDGMENTS

Many persons have contributed to the development of the flow control protocols in SNA. It has been developed jointly by designers from Communication Systems Architecture, Communications Programming, and the SNA implementations including NCP, TCAM, and VTAM. The authors would also like to acknowledge the critical reviews of the paper by Paul E. Green, Jacob Hagouel, Berton D. Moldow, the referees, and the editors.

Appendix: Glossary of abbreviations and terms

boundary function	A boundary function is a component in a subarea node that supports an LU in an attached peripheral node.
BPOOL	This is the subpool of buffers in NCP that is used to store incoming data destined for that NCP. Its size is one-half the number of NCP buffers allocated before the enter-slowdown threshold is reached.
CWALL	The CWALL is the threshold in NCP at which communication with attached devices is no longer possible because of buffer depletion.
CWI	Change Window Indicator is a tag in a message set by a subarea node on a virtual route to indicate congestion in that subarea node.
CWRI	Change Window Reply Indicator is a tag in a VR pacing response set by a subarea node at one end of a virtual route to indicate to the subarea node at the other end that congestion has occurred. Pacing window size reductions are made if possible.
enter-slowdown	This is an NCP buffer threshold that invokes certain congestion protocols (slowdown) in NCP.
exit-slowdown	This is an NCP buffer threshold which halts the congestion protocols invoked when the enter-slowdown threshold was crossed.
LU	Logical Unit is the data communication port in SNA for a user or application.
NCP	Network Control Program is an IBM program product that resides in the 3705 Communications Controller, acts as a subarea node in SNA, and provides an attachment point for peripheral nodes.

NCP buffer pool — This is a buffer space allocated by NCP from all the available storage in the 3705 after NCP is loaded. The buffers are used to contain the messages being transmitted to, from, and through NCP.

pacing window — This is a set of messages grouped for flow control algorithms. The first message in a pacing window is tagged as such, and the pacing window must be acknowledged before another pacing window can be sent. SNA uses pacing windows for flow control for sessions and virtual routes.

peripheral node — This is SNA implementation that attaches to a subarea node to gain connectivity to the total network. A peripheral node supports one or more LUs.

PRQ — Pacing Request is the first message of a pacing window. Pacing requests are explicitly tagged as such.

PRSP — Pacing Response is the acknowledgment of the receipt of a pacing window. Pacing responses may be combined with other traffic (piggy-backed) or sent as independent messages (isolated).

PU — Physical Unit is the entity in SNA that manages the resources of a node, including LUs, links, and buffers.

RWI — Reset Window Indicator is a tag in a message set by a subarea node along a virtual route to indicate severe congestion in that subarea node. RWI causes an immediate reduction in a pacing window size to an established minimum value.

session — The communication protocol between two LUs is termed a session. Sessions may be paced, and may have as many as two stages in each direction.

stage — A stage is a complete pacing protocol that may be used in concert with other pacing protocols to regulate traffic in a single direction. Stages allow more complex pacing protocols to be built that utilize intermediate buffering along the path on which data are flowing.

subarea node — This is an SNA implementation which interacts with other subarea nodes to form a network with full awareness, routing, and congestion control among the subarea nodes. A subarea node may support one or more LUs and boundary functions.

subarea A subarea is the region of a network consisting of a single subarea node and its attached peripheral nodes, plus all the resources these nodes manage.

TG Transmission Group is a logical link between a pair of adjacent subarea nodes. A transmission group has one or more physical links. There may be more than one transmission group between a pair of adjacent subarea nodes, with each transmission group having one or more physical links.

TP Transmission Priority is a relative priority assigned to a message and used to schedule messages over a transmission group. There are three levels of transmission priority, 0 (lowest), 1, and 2 (highest). A transmission priority is permanently assigned to each virtual route.

VR Virtual Route is a logical path defined between a pair of subarea nodes through the network of subarea nodes and transmission groups. Traffic on virtual routes is always paced.

CITED REFERENCES

1. M. Gerla and L. Kleinrock, "Flow control: A comparative survey," *IEEE Transactions on Communications* **COM-28,** No. 4, 553–574 (April 1980).
2. M. C. Pennotti and M. Schwartz, "Congestion control in store and forward tandem links," *IEEE Transactions on Communications* **COM-23,** No. 12, 1434–1443 (December 1975).
3. J. P. Gray and C. R. Blair, "IBM's Systems Network Architecture," *Datamation* **21,** No. 4, 51–56 (April 1975).
4. J. H. McFadyen, "Systems Network Architecture: An overview," *IBM Systems Journal* **15,** No. 1, 4–23 (1976).
5. T. F. Piatkowski, D. C. Hull, and R. J. Sundstrom, "Inside IBM's Systems Network Architecture," Special Report, *Data Communications,* 34–48 (February 1977).
6. J. P. Gray, "Network services in systems network architecture," *IEEE Transactions on Communications* **COM-25,** No. 1, 104–116 (January 1977).
7. R. J. Cypser, *Communications Architecture for Distributed Systems,* Addison-Wesley Publishing Co., Reading, MA (1978).
8. J. P. Gray and T. B. McNeill, "SNA multiple-system networking," *IBM Systems Journal* **18,** No. 2, 263–297 (1979).
9. V. Ahuja, "Routing and flow control in Systems Network Architecture," *IBM Systems Journal* **18,** No. 2, 298–314 (1979).
10. J. D. Atkins, "Path control: The transport network of SNA," *IEEE Transactions on Communications* **COM-28,** No. 4, 527–538 (April 1980).
11. *Systems Network Architecture, Format and Protocol Reference Manual: Architecture Logic,* SC30-3112; available through IBM branch offices.
12. G. A. Deaton, "Flow Control in Packet-Switched Networks with Explicit Routing," *Proceedings of the Flow Control in Computer Networks Conference,* Paris, France, February 12–14, 1979.

13. M. Reiser, "A queueing network analysis of computer communication networks with window flow control," *IEEE Transactions on Communications* **COM-27,** No. 8, 1199–1209 (August 1979).
14. W. S. Hobgood, "The role of the Network Control Program in Systems Network Architecture," *IBM Systems Journal* **15,** No. 1, 39–52 (1976).

F. D. George is located at the IBM Thomas J. Watson Research Center, P.O. Box 218, Route 134, Yorktown Heights, NY 10598, and G. E. Young is located at the IBM Communication Products Division, P.O. Box 12195, Research Triangle Park, Raleigh, NC 27709.

Reprint Order No. G321-5166.