# The Scalable Networking Scheme for High-Speed Networks

Chi-Hsiang Yeh
Dept. of Electrical and Computer Engineering
Queen's University
Kingston, Ontario, K7L 3N6, Canada
chi-hsiang.yeh@ece.queensu.ca

Emmanouel A. Varvarigos
Dept. of Electrical and Computer Engineering
University of California
Santa Barbara, CA 93106-9560, USA
manos@ece.ucsb.edu

*Abstract*— **We propose the scalable networking scheme for high-speed networks where quality of service (QoS) is important. The main objective of the scheme is to provide QoS guarantees and to achieve scalability to very large traffic volume and link bandwidth. Other important goals include extensibility, small to moderate buffer requirements, high throughput, small latency, and loss-free communication. The proposed scheme can service a wide variety of traffic classes and is applicable to various switching and multiplexing techniques.**

*Keywords*— **scalability, quality of service (QoS), signaling, resource reservation, routing, flow control, communication protocols.**

## I. Introduction

Many communication protocols have been proposed in the literature or implemented in practice, some of which deal with flow control [3], [7], [8], [13] and some others deal with connection establishment [4], [6], [10], [12]. We have also proposed the conflict-sense routing (CSR) protocol [14], the ready-to-go virtual circuit (RGVC) protocol [15], [16] and the efficient reservation virtual circuit (ERVC) protocol [16], [17] to be used in the 100-Gbps Thunder and Lightening network testbed, and the virtual circuit deflection (VCD) protocol [19] to be used in the MOST Tera Switch under development at UCSB.

With the advance of networking technologies, the bandwidth of communication networks is increasing rapidly and terabit networks are expected to be available in the near future. However, the exponential growth of Internet traffic and the requirements of emerging networking applications, such as mobile communications and quality of service (QoS) for multimedia traffic, are imposing great challenges on the future networking environment. In particular, scalability is a critical issue to next-generation Internet, where very high-speed links are used, an extremely large and rapidly growing number of sessions are serviced, and QoS guarantees are mandatory.

Previous solutions for guaranteeing QoS may not be scalable to future networks that carry very large traffic volumes. A reason is that these protocols require intermediate network nodes to maintain QoS information for each of the QoS-guaranteed sessions that they service. In the future Internet core, the number of sessions serviced by a node is expected to be extremely large so that a QoS-guaranteed communication protocol that requires every node to maintain per-flow state will be infeasible. Also, the resource reservation protocol (RSVP) [5] requires refreshing messages sent frequently along the established connections, which may lead to scalability problems due to large processing overhead.

Research partially supported by the DARPA MOST project under Contract DABT63-93-C-0039

Another problem with the future networking environment is that in order to guarantee loss-free communication, transmission at full link speed, and small latency, previous protocols require each link to have a buffer space at least equal to the product of link bandwidth and the round-trip link propagation delay. Although this is currently feasible, the buffer requirements will be prohibitively high in future networks with very high-speed links. On the other hand, the rate-based flow-control mechanism can guarantee QoS and loss-free communication using a small buffer space. Protocols based on such a flow-control mechanism, however, require a round-trip delay for connection establishment before data packets can be transmitted, which is nonnegligible compared to typical session holding times in high-speed networks, leading to lower link utilization and thus lower achievable throughput. Also, the increase in latency may not be acceptable to latency-sensitive sessions.

In this paper, we propose the *scalable networking scheme (SNS)* to tackle the above two challenging issues in high-speed networks with very large traffic volume and QoS-guarantee requirements. We propose the *scalable resource reservation subscheme* and the *scalable routing subscheme* for QoS guarantees without maintaining per-flow state. We also propose the *scalable flow-control subscheme* for traffic engineering and loss-free communication in very high-speed networks with small to moderate buffer requirements. We also present the aggressive transmission technique where data packets can be transmitted before a connection is fully established, considerably reducing the latency, and timed/activated resource reservation and allocation/locking, where resources are reserved and allocated/locked for a session only during the period when resources are actually used by that session. The scalable resource reservation subscheme is the first that combines both timed reservation and aggressive transmission, achieving small latency and high throughput at the same time. Also, activated resource allocation/locking relieves SNS-based network of the need for accurately synchronized clocks, and when combined with the immediate reservation technique, enables SNS-based networks to allocate and/or lock resources efficiently without relying on the availability of any clock signals. Moreover, the proposed conservative transmission technique guarantees QoS/class of service (CoS) and loss-free communication at the expense of a round-trip delay for connection establishment; the aggressive transmission technique combined with the credit-first flow-control mechanism guarantees loss-free communication with a small offset delay; and the proposed aggressive reservation technique guarantees QoS/CoS and loss-free communication with a small or even zero offset delay, by allowing additional resources to be reserved in advance (e.g., for virtual paths (see Subsection III-B)).

The remainder of the paper is organized as follows. In

Section II, we present the scalable networking scheme. In Section III, we present the scalable resource reservation scheme and several reservation mechanisms. In Section IV, we present the scalable routing scheme. In Section V, we present the scalable flow-control scheme and a novel flow control mechanism. In Section VI, we conclude the paper.

## II. The Scalable Networking Scheme (SNS)

The objective of this work is to develop a networking scheme for next-generation networks with the following capabilities and properties:

- *QoS/CoS guarantees* and effective *traffic engineering.*
- *scalability* to very large traffic volume and link bandwidth,
- *extensibility* so that new mechanisms and advanced features can be easily added to the protocols, if desired, and simpler nodes with basic features and complex nodes with advanced features can coexist in the same network.
- high throughput, low latency, small call-setup rejection rate, and effective tradeoffs between them for connection-oriented sessions.
- small to moderate buffer requirements.

Other important goals include (1) servicing a wide range of traffic classes efficiently, (2) guaranteeing loss-free communication, if required, or providing packet-loss probability no more than the ones requested by sessions, (3) applicability to various implementation technologies and underlying switching and multiplexing techniques, (4) effective tradeoffs between the efficiency and complexity of the resultant protocols, as well as the buffer requirements (and thus effective tradeoffs between the hardware cost and network performance), (5) efficient internetworking with QoS/CoS guarantees between different SNS-based networks as well as efficient internetworking between SNS-based protocols and other popular protocols/technologies, and (6) the capability to incorporate other techniques to satisfy other important requirements, such as security and reliability.

To achieve these goals, the scalable networking scheme is divided into several subschemes dealing with (I) signaling and resource reservation (based on the scalable resource reservation scheme), (II) routing and forwarding (based on the scalable routing scheme), (III) flow control and traffic engineering (based on the scalable flow-control scheme), (IV) application to various switching and multiplexing techniques, (V) internetworking between different protocols and interfacing with different layers, (VI) reliability and failure recovery and reoptimization, and (VII) other important issues such as security. These subschemes are developed separately, and one or several mechanisms can be taken from each of the subschemes and combined to handle a particular session. The mechanisms and features are selected according to the properties and requirements of that session, network traffic, and the actual implementation of the network. Such flexibility enables a single protocol to service certain functions of various traffic classes efficiently, rather than requiring different protocols for different traffic classes; it also enables the resultant protocols to be adaptive to traffic conditions and implementation technologies.

QoS/CoS and loss-free communication are guaranteed through the reservation and allocation/locking of resources (Part I) and an appropriate flow control mechanism (Part III); low latency and high throughput are achieved through fast establishment of connection (Part I) and efficient reservation of resources (Part I), in addition to appropriate routing algorithms (Part II) and traffic engineering (Part III); scalability is achieved by reducing buffer requirements and providing QoS/CoS guarantees without per-flow state at every intermediate node, where the former is made possible by the scalable flow-control scheme (Part III) and the latter by appropriate reservation mechanisms (Part I), in addition to scalable routing and forwarding mechanisms (Part II) and prioritizing data packets when desired. Extensibility is achieved by appropriate design of resource reservation and allocation/locking techniques and routing and flow-control mechanisms [20]. In what follows, we briefly present the functions of and the interaction between these subschemes, and elaborate on some of them in the next few sections.

- **I. Signaling and Resource Reservation:**

SNS provides service to both connection-oriented and connectionless traffic. For a connection-oriented session, we first establish a connection between the source and destination using the scalable resource reservation scheme, and then transmit data packets along the reserved route using the resources reserved for that connection. A setup packet is first sent to reserve the required capacity, followed by data packets after an offset time $\delta_S$. The setup packet is routed in the network using the scalable routing scheme (Part II), while the data packets are handled at the source, destination, and intermediate nodes using the scalable flow-control scheme (Part III) to prevent them from dropping.

- **II. Routing and Forwarding:**

For connectionless sessions, the data packets are routed using the scalable routing scheme. For connection-oriented sessions, the setup packets are routed using the scalable routing scheme, and then the data packets follow using the path(s) established by the setup packet(s). We have derived several session/tunnel routing algorithms for the scheme, which will be reported in the near future.

- **III. Flow Control and Traffic Engineering:**

In order to guarantee a minimum transmission bandwidth, loss-free communication or small dropping rate, and balanced and effective utilization of aggregate network resources, the scalable flow-control scheme is employed to utilize the resources reserved by the scalable resource reservation scheme so as to achieve high throughput and small latency and to guarantee the quality of service (QoS).

- **IV. Switching and Multiplexing:**

The resource reservation mechanisms, routing algorithms, and flow control mechanisms will be combined with various underlying multiplexing techniques and switching techniques, such as virtual circuit, circuit switching, virtual cut-through, wormhole routing, as well as other new switching techniques. Details will be reported in the near future.

- **V. Internetworking and Interfacing:**

In order to adapt to different physical media (such as dense wavelength division multiplexing (DWDM) or wireless networks) and different switching techniques, SNS may lead to different protocols; different requirements and constraints in local area networks and wide area networks may also lead to different protocols/networks. Networks implementing these SNS-based protocols as well as existing technologies, such as Ethernet, asynchronous transfer mode (ATM) [1], [11], and multiprotocol label switching (MPLS) [9], may need to be connected together. Therefore, internetworking is an important issue for SNS in that it determines whether QoS guarantees and other goals such as high throughput, low latency, loss-free communication, reliability, and security can be met across different networks. Efficient interfacing with higher and lower layers is also important.

- **VI. Reliability and Failure Recovery:**

When a network link or node fails, it is important for the network to recover from the failure in a short time (possibly by redundant/precalculated alternative routes), to maintain reasonable QoS for QoS-guaranteed sessions, to optimize new routing tables, and to balance the load by reengineering the traffic. Incorporating reliability and reoptimization techniques into the networking scheme, especially for QoS-guaranteed sessions, is crucial to its applicability.

- **VII. Others:**

Other components may be added to the basic scheme to satisfy emerging application requirements, to enhance performance, and/or to reduce implementation cost as new networking and VLSI technologies mature and application environments change. For example, security for both connection-oriented sessions and datagrams is of particular interest.

## III. Signaling and Resource Reservation

In this section, we present the scalable resource reservation scheme that achieves high throughput and small latency in high-speed networks. We also present several resource reservation mechanisms for scalable resource reservation without maintaining per-flow state at every intermediate node.

### A. The Scalable Resource Reservation Scheme (SRRS)

The scalable resource reservation scheme consists of seven phases. These phases are "pipelined" in that a latter phase can start at a node before a former phase is completed in the network.

- **Phase 1 (Setup Phase):**

In order to establish a connection (i.e., a route with reserved capacity) for the transmission of data packets, the source node (for source-initiated reservation) sends a setup packet, which carries with it part or all of the following information:
*(i) basic information:* (a) source and destination addresses, (b) traffic class, and (c) loose/tight routing path (if source routing is used),
*(ii) parameters:* (a) transmission bandwidth, (b) optimization criteria (e.g., throughput, latency, and/or reliability), (c) sensitivity to delay, (d) time-to-live (TTL), and (e) other QoS/CoS parameters,
*(iii) SNS information:* (a) the starting time $t_S$ for transmission, (b) the amount of data to be transmitted or the duration $d_i$ of the session,
*(iv) SNS options:* (a) choice of dropping, buffering, or deflecting data packets when connection is not available (see Phase 4), (b) the routing and forwarding mechanism to be used, (c) the flow control mechanism to be used, and/or
*(v) other information:* any information that may be useful, such as the way resource reservation and locking are initiated and terminated (e.g., timed reservation, activated locking).
When an intermediate node $i$ receives the setup packet, it finds an appropriate outgoing link (possibly by looking at its routing table based on the information and parameters carried by the setup packet). Any unicast or multicast routing algorithm (see Section IV) can be used to route the setup packet.
Node $i$ then computes the reservation time $t_i$, after which data packets are expected to arrive, and the reservation duration $d_i$, during which all data packets are supposed to finish transmission at that node. If there are several priority levels for packets, setup and other control packets should be assigned higher priority for transmission.
For receiver-initiated reservations, the destination, instead of the source, is the one that sends the setup packet to establish the connection in the reverse direction of the path followed by the setup packet. In the remaining of the paper, we omit receiver-initiated reservations for simplicity.

- **Phase 2 (Reservation Phase):** Each node $i$ on the established part of a connection reserves the capacity required by the connection at reservation time $t_i$ (according to its local clock or the clock signal it receives).

The reservation time $t_i$ may range from the time node $i$ receives the setup packet (that is, the capacity is reserved right after the outgoing link is selected) to a time long enough for the entire connection to be established and for the first data packet to arrive at the node (which will happen at least a round-trip delay after the setup packet arrives at node $i$ if the source node waits for an acknowledgement to start transmission). Note that capacity at downstream nodes should be reserved with time lags approximately equal to the propagation delay on the link connecting two neighboring nodes plus the transmission delay, since approximately that much time is required for the first data packet to arrive from an upstream node to a downstream neighbor. In other words, $t_j - t_i$ for neighboring nodes $i, j$ is approximately equal to the delay of link $(i, j)$ and once $t_S$ is specified, the reservation times $t_i$ for all downstream nodes $i$ do not need to be carried in the setup packet.
Note that if the required capacity is reserved right after a setup packet is processed at an intermediate node, the reservation technique is called *immediate reservation* and the network node does not need to have a clock at all; otherwise, we call the reservation technique *timed reservation* and the network node need to have a local clock or to receive a clock signal from the network. Note, however, that an SNS connection can contain network nodes without a clock using immediate reservation (usually for nodes that are not performance bottlenecks) and nodes with clocks using timed reservation (usually for nodes where resources are scarce relative to the application requirements). Note also that we only need to synchronize clocks between neighboring nodes (to a certain accuracy) in order to reserve the capacity (and to allocate/lock the reserved capacity accurately (see Phase 3)), since each node on the connection can compute and accumulate the time that has elapsed since the setup packet was generated in order to determine the remaining time to turn on the connection. A global clock or synchronized distributed clocks (within small error bounds) may help but are not necessary.

- **Phase 3 (Allocation/Locking Phase):** Each node $i$ on the established part of a connection allocates/locks the reserved capacity for the connection

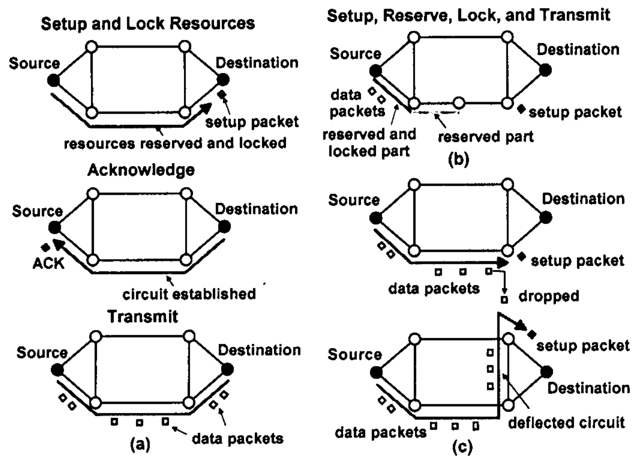

Fig. 1. (a) Setup, acknowledgement, and transmission phases are not pipelined in ordinary reservation schemes. (b) Pipelining setup, reservation, locking, and transmission phases in the scalable resource reservation scheme (SRRS). (c) Possible actions (e.g., dropping and deflection) taken by SRRS when data packets catch up with the setup packet. Note that buffering or other actions are also possible when appropriate flow-control mechanisms (see Section V) are used.

*(i)* upon the reception of an allocation/locking control packet (i.e., a special case of *activated allocation/locking*),
*(ii)* upon the reception of the first data packet (i.e., another special case of activated allocation/locking),
*(iii)* at time $b_i$, $b_i \geq t_i$ (i.e., *timed allocation/locking*), or
*(iv)* whichever happens first among a subset of the preceding events.

The option taken may depend on the type of the connection or the default of the node. Note that different nodes on a connection can initiate locking based on different options. Note also that the reserved interval may be made somewhat larger than the locked interval since the former does not waste as many network resources, and a longer reserved interval can prevent data packets from dropping due to inaccurate clocks.

Note that the capacity reserved by a connection may still be used by other traffic, unless it is "locked" for that connection. In other words, the purpose of reservations is bookkeeping, while locking explicitly forbids other sessions from using the locked capacity. Instead of locking the reserved capacity, we can "allocate" the capacity to that connection so that the "owner session" has the highest priority to use the capacity, while other connections or fill-in traffic may use the allocated capacity when no packets from the owner session are available.

This phase can be combined with Phase 2 as in most previous reservation-based protocols, so that reservation and allocation/locking are performed at the same time. It may also be completely eliminated if allocation or locking of resources is not required.

• **Phase 4 (Transmission Phase):** The source node $S$ starts transmission when it receives an acknowledgement for the establishment of a connection or at transmission time $t_S$ (of its local clock or according to the clock signal it receives), which is computed when the setup packet is sent out (see item (iii) of Phase 1) and may change due to a postponement or rejection control packet from an intermediate node, depending on the option specified for the connection.

Note that $t_S$ may range from the time the source node $S$ sends the setup packet to a time long enough for the entire connection to be established and acknowledged. The transmission technique where a source node is allowed to start transmitting data packets before it receives an acknowledgement for the establishment of the connection (see Fig. 1b) will be referred to as *aggressive transmission.* The transmission technique where a source node is not allowed to transmit any data packet until it receives an acknowledgement from the destination and until its local time is at least $t_S$ will be referred to as *conservative transmission.*

If the data packets catch up with the setup packet or arrive at a node before the reserved capacity is available (e.g., due to inaccuracy/error of clocks), one of several possible actions is performed according to the option specified in the setup packet or the default of the network node:

– **Option 4.1 (Dropping packets):** The simplest action is to drop the data packets.

Note that there are several ways to guarantee that data packets are never dropped due to overflow, including using conservative transmission or an appropriate flow-control mechanism (see Option 4.2 and Subsection V-A).

– **Option 4.2 (Buffering packets):** Higher throughput and smaller latency may be achieved by storing the data packets in buffers until the connection to the next node is available, especially when the packets arrive only slightly before the outgoing link is expected to become available.

To implement this option, network nodes need sufficient buffer space and an effective flow control mechanism such as credit-first flow control (see Section V).

– **Option 4.3 (Deflecting packets):** Deflecting the setup and data packets to an available outgoing link reduces the buffer requirements and may also reduce latency.

When no appropriate outgoing link is available, the data packets are dropped, buffered, redirected, or transmitted over several split subconnections.

– **Option 4.4 (Other alternatives):**

More options, such as bifurcated routing, can be added later as new networking and VLSI technologies mature and applications with different requirements emerge.

• **Phase 5 (Renegotiation/Rerouting Phase):** If a session's transmission rate or duration changes, or if the session has to be rerouted to accommodate other connections or to increase its transmission bandwidth, the source node or an intermediate node sends a control packet to adjust the resources and/or the path occupied by the connection. The Renegotiation/Rerouting Phase can be repeated several times during the holding time of a session, if so desired.

• **Phase 6 (Release Phase):**

Each node $i$ on the connection release the allocated/locked resources, upon the reception of a *release control packet*, upon the reception of the last data packet, at (local) time $b_i + d_i$, until node $i$ times out after the connection stays idle for a long time, or whichever happens first among a subset of these events. The option taken may again depend on the type of the connection or the default of the node. Note that any option for the initiation of allocation/locking can be paired with any option for the termination of allocation/locking.

- **Phase 7 (Teardown Phase):** Each node $i$ on the connection tears down the connection (by deleting associated information, such as virtual channel numbers, and terminating the reservation interval) when it receives a *teardown control packet* or the *last* data packet, or at (local) time $t_i + d_i + \triangle_i$, depending on the option of the connection or the default of the node, where $\triangle_i$ is chosen so that $t_i + \triangle_i \geq b_i$. The teardown control packet may be sent by the destination in the reverse direction on the connection path or by the source after it receives an acknowledgement from the destination for the last data packet. Note that Phases 6 and 7 may be merged into a single phase.

Other phases, actions, and options such as those dealing with bursty traffic may be added to the preceding basic scheme to enhance the performance as new networking and VLSI technologies mature and application requirements change. Furthermore, even though SRRS is reservation-based and connection-oriented, it can coexist with other routing mechanisms such as those used for connectionless traffic to maximize the network throughput.

### *B. Resource Reservation Mechanisms*

In SRRS, we allow several different mechanisms for the reservation of network resources.

The first mechanism is to reserve resources for each of the virtual circuits that require QoS guarantees and record the QoS requirements of that virtual circuit on all the intermediate nodes along the virtual circuit. Each network node maintains a table with the incoming and outgoing virtual channel numbers and the QoS requirements for each of the virtual circuits that pass through it. A data packet of such a session only needs to carry with it a virtual channel number and does not have to carry any QoS parameters. This resource reservation mechanism is similar to that of ATM networks [1], [11] and those of ordinary virtual circuit protocols that have appeared in the literature [15], [17], and can work with the first routing mechanism of Section IV.

The second mechanism is to reserve the required aggregate resources at a network node for all the connections that require CoS/QoS guarantees that pass through it. The intermediate nodes do not need to maintain the information concerning incoming and outgoing virtual channel numbers or CoS/QoS requirements for each of the connections. Instead, a data packet of such a session carries with it the CoS/QoS parameters. This reservation mechanism has to be combined with a routing mechanism that always routes the data packets of the same session along the same path, unless some intermediate node(s)/link(s) fail (e.g., the second or third routing mechanism of Section IV). It is not necessary for these sessions to send refreshing messages along the connections, although they may send them infrequently if desired.

The third mechanism is to reserve the required aggregate resources at a network node for all the connections sharing a virtual path. The network node does not maintain specific information (e.g., for QoS or routing) for every connection. Usually, we require data packets of such connections to carry with them their CoS parameters. However, if a virtual path has default CoS/QoS requirements/guarantees, then the data packets of that virtual path do not need to carry CoS/QoS parameters unless they have different CoS/QoS requirements. This mechanism can work with the fourth and fifth routing mechanisms presented in Section IV. This mechanism allow resources (e.g., 5% or 10% additional resources other than those allocated or locked) to be reserved in advance so that when packets of new sessions arrive, they can be transmitted with QoS/CoS guarantees right away without waiting for a round-trip delay for connection establishment. The setup packets of these new sessions can then be used to further reserve additional resources. Note that these additional reserved resources can be used by any other sessions or fill-in traffic before they are allocated/locked for certain sessions. We refer to such reservation technique as *aggressive reservation.*

We can use the first mechanism for sessions with smaller data packets, since such sessions would require larger overhead for carrying CoS parameters if the second or third mechanism were used. When traffic is heavy and the table entries for virtual circuits are about to be exhausted, we simply use the second and/or third mechanism to make reservation for more sessions. Other reservation mechanisms with respective advantages can also be used in SRRS if desired. Since the second and third mechanisms do not require a network node to maintain information for every connection that passes through it, SNS-based networks using a mix of these mechanisms are scalable to very high traffic volumes without requiring prohibitively large tables at network nodes.

The mechanisms for resource allocation and locking are similar to the preceding resource reservation mechanisms. When a resource allocation/locking mechanism similar to the second or third resource reservation mechanism is used, we assign the highest priority to the sessions that are allocated the resources. Note that if a session uses the first (second or third) mechanism for resource reservation, it should use a resource allocation/locking mechanism similar to the first (second or third, resp.) mechanism for resource allocation/locking. The details for resource allocation/locking mechanisms are omitted in this paper.

## IV. The Scalable Routing Scheme (SRS)

In this section we present the scalable routing scheme for scalable routing without maintaining per-flow routing information at every network node.

SRS supports both connectionless and connection-oriented sessions. For connectionless traffic, any unicast or multicast routing algorithm [1], [11], [18] can be used to route a datagram; for a connection-oriented session, any unicast or multicast routing algorithm can be used to route the setup packet, and the data packets follow the same path traversed by the setup packet. Different algorithms are allowed for routing different setup packets and datagrams. Note that in an SNS-based network, it is also possible that all sessions are connection-oriented.

In what follows, we present several mechanisms for routing and forwarding the data packets of a connection-oriented session. The first mechanism is the same as the one used in ordinary virtual circuit protocols, where setup packets are forwarded to appropriate outgoing links according to their destination addresses or the routes specified by the source nodes to establish the virtual circuits. Data packets are then forwarded based on their virtual channel identifiers (VCI), by looking up the switching table at each network node.

The second mechanism is similar to that used in datagram networks, where a network node select the outgoing link for a data packet according to its destination address (possibly based on a routing table). Therefore, each data packet carries with it a destination address rather than a VCI or a virtual path identifier (VPI). But we require that in the routing tables, the entries used to forward data packets of a session have to be "fixed" during the holding time of that session. For example, we can use shortest paths (in terms of the minimum number of hops) as the criteria to route packets, which do not change with time. Note that we do not change these entries even if some nodes fail or new nodes are added, unless they are on the virtual circuit. We can also use the measured/estimated traffic conditions to calculate these entries, but we do not change nor delete them until all the sessions that use them are torn down (e.g., by using a counter and timeout to determine it). An intermediate network node knows which entries to use to route a data packet simply by looking at the time stamp the data packet carries with it, which records the time the first packet of that session was sent. Therefore, routing using this mechanism can still be adaptive to traffic conditions to a certain degree, since we can add new entries when traffic conditions change and use these new entries for newer sessions.

The third mechanism is similar to source routing, where a network node has to forward a data packet to an appropriate outgoing link according to the next-hop address or port number the packet carries with it. Since we only need a small number of bits to specify a port number, the overhead for this method is not too large when the packet size is not small and the number of intermediate nodes is not large. When node addresses are used, the number of bits, and thus the overhead, can be reduced by using loose source routing. The latter results in a datagram routing mechanism rather than a virtual circuit routing mechanism but it may improve QoS of ordinary datagram routing mechanisms since the data packets are now forced to traverse certain nodes where resources may have been reserved.

The fourth mechanism is to use a connection concatenating one or several virtual paths and/or virtual subcircuits to route the packets of the same session. At the end of a virtual path along a connection, one of the following three submechanisms is used to change the VPI/VCI of a data packet and to forward the packet. The first submechanism (corresponding to the first routing mechanism) looks up into a switching table to find an appropriate VCI and VPI according to the current VCI. The second submechanism (corresponding to the second routing mechanism) forwards the data packet to an appropriate virtual path, virtual subcircuit, or outgoing link to another node according to its destination address (possibly based on a routing table). The VCI and VPI of the packet are updated accordingly. Note that, similar to the second routing mechanism, we require the associated routing table entries remain "fixed" during the lifetimes of all the sessions using those entries. The third submechanisms (corresponding to the third routing mechanism) requires each data packet to carry with it the VPI for the next virtual path to be traversed, and the data packet is forwarded to that virtual path accordingly. Note that these three submechanisms can be employed in the same connection if desired. Other submechanisms may also be added to this routing mechanism at a later time. Moreover, virtual paths can be classified into several hierarchical levels if desired.

The fifth mechanism is to use a connection consisting of segments of virtual paths (to be referred to as virtual subpaths) to route the setup and data packets, rather than bridging several complete virtual paths. Each data packet carries with it the VPIs for all the virtual paths it traverses and the address(es) of the node(s) where the connection swaps from one virtual path to another before it ends. Note that for this routing mechanism to work, at least some network nodes have to be able to determine whether a data packet needs to be swapped to a different virtual path. Any of the three submechanisms for the fourth routing mechanism can be used for connecting two virtual subpaths. The total number of virtual paths required is only $\Theta(\sqrt{N})$ if we use $\Theta(\sqrt{N})$ "roughly" horizontal virtual paths and $\Theta(\sqrt{N})$ "roughly" vertical virtual paths, and allow data packets to swap virtual paths once or several times at their intersections, where $N$ is the number of nodes in the network. As a comparison, previous methods using a complete mesh of VPIs to route packets require $\Theta(N^2)$ virtual paths, which is not scalable to networks with an extremely large number of nodes. Note also that we can use one or several virtual subcircuits to bridge several virtual subpaths to establish a connection.

The second mechanism can be used to route sessions that transmit a relatively small amount of data so that routing them across a temporarily congested area does not worsen the traffic conditions considerably. Also, other mechanisms and/or datagram routing algorithms that are more adaptive to traffic conditions can be used for other sessions so that congestion can be avoided by using a mix of these mechanisms and algorithms. The third mechanism can be used for sessions whose data packets that are relatively large so that the overhead is smaller. These two mechanisms do not require intermediate nodes to maintain specific routing information for the sessions they service. The fourth and fifth mechanisms with the second or third submechanism only require routing information for virtual paths, whose number is considerably smaller than the number of sessions. A session using the fourth or fifth mechanism with the first submechanism requires routing information at intermediate nodes where it is swapped from a virtual path/subpath to another, which is considerably fewer than the total number of nodes along the connection when a proper set of virtual paths are available. If all or a sufficiently large part of nodes can serve as such intermediate nodes for swapping virtual (sub)paths, the required switching table sizes are considerably smaller than those in a network using virtual circuits alone for most of its connections. Also, the required processing power using the fourth or fifth mechanism for forwarding data packets using the second submechanism is considerably smaller than that using the second routing mechanism. Therefore, by using some or all of these four mechanisms for a sufficient fraction of traffic, SNS-based networks can be made scalable to very large traffic volumes without requiring prohibitively large routing tables. Other mechanisms that have respective advantages may be added to SRS at a later time. Note that some or all of the above five routing mechanisms can be employed in the same connection without difficulties.

To see the advantages of the proposed routing scheme, consider an SRS-based network where 40% of the traffic is connection-oriented and 60% of the traffic is connectionless. Among the connection-oriented sessions, 20% of them use the first mechanism and 80% of them use the last four mechanisms. Then the required size for routing tables in the SRS-based network is approximately 8% that in a network using virtual circuits without virtual paths for most connections. Moreover, we can further reduce the required table size by forcing more sessions to use the last four routing mechanisms or datagram routing algorithms (the latter may be combined with prioritization similar to the one used in differentiated service [2], or reservation with refreshing messages as in RSVP to improve QoS/CoS). From this example, we can see that SRS enables SNS-based networks to be scalable to very large traffic volume.

## V. Flow Control and Traffic Engineering

The aggressive transmission technique of our scalable resource reservation scheme allows data packets to be sent before the connection is fully established, imposing additional challenges on the required flow-control mechanisms. In this section, we first introduce a novel flow-control mechanism and then present the scalable flow-control scheme for scalable flow control without requiring extremely large buffer space for loss-free communication in high-speed networks.

### *A. Credit-First Flow Control*

Rate-based flow control can guarantee QoS, while credit-based flow control can achieve higher throughput, especially for bursty traffic. In this subsection, we introduce a hybrid flow control mechanism that can be combined with the scalable resource reservation scheme to considerably reduce latency and increase throughput of rate-based flow control, while achieving QoS guarantees and loss-free communication at the same time. Moreover, the buffer space required by the hybrid mechanism is considerably smaller than that required by networks using credit-based flow control alone. These desirable properties are attained by taking advantage of reserved connections in reservation-based schemes.

In the *credit-first flow control* mechanism, we use credit-based flow control before the end-to-end connection is fully established, and use rate-based flow control after the connection is established and acknowledged. Upon reception of a setup packet, the destination sends an acknowledgement along the reverse direction of the established connection. When an intermediate node receives the acknowledgement, it starts forwarding data packets belonging to that connection using rate-based flow control, without further wasting its credits. In this way, the relative transmission time $\delta_S$ in the scalable resource reservation scheme can be made very small or even zero without the risk of packets being dropped, reducing the latency considerably. Since data packets can still be transmitted and are guaranteed to arrive at the destination even if the connection is not established as soon as expected or at the rate originally requested, the link bandwidth is not wasted for retransmissions and does not stay idle unnecessarily so that the maximum achievable throughput is increased. Moreover, once a connection is established and the acknowledgement is received by the source, the session can transmit at the requested rate, guaranteeing the QoS. Since credits are required only during the connection establishment phase, the total credit and thus the buffer space required to sustain high throughput is much smaller than in networks using ordinary credit-based flow control.

Note that before a connection is completely established, the average transmission rate should not exceed the reserved bandwidth even if additional credits are available, so that data packets do not accumulate at intermediate nodes in most cases. If the connection is established extraordinarily slowly so that some data packets have accumulated at some intermediate nodes, the source can temporarily reduce the transmission rate or even cease transmission for an appropriate interval following a timeout, or upon reception of a control packet from a congested intermediate node or the late acknowledgement from the destination. A source node that ceases transmission may send a control packet to release the reserved/locked capacity during the corresponding interval so that it can be utilized by other sessions.

### *B. The Scalable Flow-Control Scheme (SFS)*

Reservation-based protocols usually reserve certain bandwidth for a connection and use rate-based flow control for all connections. In our scalable flow-control scheme, more than one flow control mechanisms can be used in a network. Using a combination of flow control mechanisms can often improve throughput and/or reduce buffer requirements, achieving scalability in high-speed networks.

The scalable flow-control scheme uses rate-based flow control as its basic feature for some connections, and credit-first flow control or other flow control mechanisms for other connections. Note that these additional flow control mechanisms are *global features* for the scalable flow-control scheme, where a global feature is an advanced feature that has to be implemented at all network nodes for it to work efficiently and/or correctly. When loss-free communication is not required by a session (e.g., for some unspecified-bit-rate (UBR) traffic), we can also transmit data packets of that session without reservation or credit, and drop them when the buffer overflows. Alternatively, we can transmit data packets without reservation or credit before a connection is fully established, but use rate-based flow control afterwards. We refer to such a flow-control mechanism as *dropping-first flow control* (see Option 4.1 of SRRS). Note that we can assign priority classes to data packets and drop low priority packets first when a node is running out of buffer space. When data packets are dropped, a control packet may be sent back to the source to request retransmissions (e.g., starting with the first packet or for the dropped packets only); alternatively, the source node may retransmit all the packets not acknowledged by the destination node, initiated either by the network-layer protocol or a transport-layer protocol.

In what follows, we compare conventional networks and SFS-based networks to illustrate why the latter are scalable to very high link speeds. In the first example network, a conventional flow-control mechanism such as credit-based flow control [7] is used. In order to guarantee loss-free communication and transmission at full link speed, the required buffer space for a link is lower bounded by the product $P$ of the link bandwidth and the round-trip link propagation delay. Since the link propagation delay is lower bounded by the physical link length divided by the speed of light, it cannot be reduced; the link bandwidth, however, is increasing rapidly, leading to extremely large buffer requirements in high-speed networks. Other previously proposed flow-control mechanisms such as the one used in the RGVC protocol [15] also require buffer space at least equal to $P$. Therefore, these flow control mechanisms and protocols are not scalable to very high link speeds due to the prohibitively large buffer space required.

In the second example network the scalable flow-control scheme is used. We use credit-first flow control for *Type-I-A sessions* that require very small latency and *strict loss-free communication* (i.e., no packets are allowed to be dropped due to buffer overflow even before the connection is fully established). For *Type-I-B sessions* that require strict loss-free communication but are not latency-sensitive (i.e., a round-trip delay for connection-establishment is allowed before data packets are transmitted), we use conservative transmission and rate-based flow control so that the buffer space required at intermediate nodes is very small. For *Type-I-C sessions* that require small latency and *loose loss-free communication* (i.e., data packets are allowed to be dropped before the connection is fully established, but are not allowed to be dropped due to buffer overflow after the connection is established), we use dropping-first flow control so that only very small buffer space is required for these sessions. In addition to the above connection-oriented sessions, we may have connectionless traffic. For *Type-II-A datagrams* that require loss-free communication, we use credit-based flow control; for *Type-II-B datagrams* that require dropping rate smaller than a certain requested value, we assign an appropriate priority class to them, taking into account the traffic conditions, without making reservation for buffers and link capacity or consuming any credits; for *Type-II-C datagrams* that can be dropped, we simply transmit them with best efforts, without using credits or reserved buffers.

Consider a communication network where 40% of the traffic is connection-oriented and 60% of the traffic is connectionless. Also, the ratio of Types-I-A, I-B, and I-C traffic is 1:2:4 and 25% of Type-I-A data packets are sent before a connection is established, which require credits. Then the buffer requirement for all traffic type except for Type-II-A datagrams is approximately $0.014P$ for a link with delay-bandwidth product $P$ in the second example network based on SFS (i.e., 1.4% that of a link of the same speed in the first example network using credit-based flow control or RGVC). Suppose available technologies can implement $0.05P_i$ buffers for each link $i$ with delay-bandwidth product $P_i$ at an affordable price. We can then allow up to 6% of the datagrams to use credit-based flow control for loss-free communication. Note that in this example, up to about 43% of the traffic is guaranteed to be loss-free (assuming that 25% of Type-I-B data packets are sent before a connection is established, which may be dropped due to buffer overflow), and up to about 35% of the traffic is transmitted with QoS/CoS guarantees. These percentages appear to be sufficient for the requirements of future networking applications, and they can be further increased by using conservative transmission and dropping-first flow control for more sessions. Moreover, Type-II-B datagrams can usually achieve satisfactory QoS and low dropping rate as long as Types-I, II-A, and II-B traffic does not exceed a certain threshold (e.g., 80% of the maximum achievable throughput), since most packets that are dropped or delayed for a long time are Type-II-C datagrams when a sufficient fraction of traffic belongs to Type II-C. If the link bandwidth is very large so that $P$ is very large and we can only implement $0.02P$ buffers for a link, we simply reduce the percentage of datagrams that use credit-based flow control, and set higher price for using credit-first flow control so that some sessions switch to dropping-first flow control or conservative transmission using rate-based flow control, further reducing the buffer requirements. From this example, we can see that SFS enables SNS-based networks to be scalable to very high link speeds, without the need for extremely expensive or even infeasible buffer space.

## VI. Conclusions

In this paper, we have proposed the scalable networking scheme for high-speed networks with QoS guarantees. The important features of SNS include scalability, QoS guarantees, and extensibility. The proposed scheme also guarantees loss-free communication when desired, achieves high throughput and small latency, and requires small to moderate buffer space. By appropriately selecting the SNS parameters and options, we can obtain effective tradeoffs between latency, throughput, and call-setup rejection rate so that the proposed scheme is adaptable to traffic conditions and application requirements.

## References

[1] Bertsekas, D.P. and R. Gallager, *Data Networks*, Prentice Hall, Englewood Cliffs, N.J., 1992.

[2] Blake, S., D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services," RFC 2475, Dec. 1998.

[3] Bonomi, F. and K.W. Fendick, "The rate-based flow control framework for the available bit rate ATM service," *IEEE Network*, Vol. 9, no. 2, Mar.-Apr. 1995, pp. 25-39.

[4] Boyer, P.E. and D.P. Tranchier, "A reservation principle with applications to the ATM traffic control," *Computer Networks and ISDN Systems*, Vol. 24, no. 4, May 1992, pp. 321-334.

[5] Braden, R., Ed., L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP) – Version 1 functional specification," RFC 2205, Sep. 1997

[6] Cidon, I., Gopal, I. S., and Segall A., "Connection establishment in high-speed networks," *IEEE/ACM Trans. on Networking*, Vol. 1, no. 4, Aug. 1993, pp. 469-481.

[7] Kung, H.T. and K. Chang, "Receiver-oriented adaptive buffer allocation in credit-based flow control for ATM networks," *Proc. INFOCOM'95*, 1995, pp. 239-252.

[8] Ohsaki, H., M. Murata, H. Suzuki, C. Ikeda, H. Myahara, "Rate-based congestion control for ATM networks," *Computer Communication Review*, Vol. 25, no. 2, Apr. 1995, pp. 60-72.

[9] Rosen, E.C., A. Viswanathan, R. Callon "Multiprotocol label switching architecture," Internet Draft draft-ietf-mpls-arch-06.txt, Aug. 1999.

[10] Suzuki, H. and F.A. Tobagi, "Fast bandwidth reservation scheme with multi-link and multi-path routing in ATM networks," *Proc. INFOCOM'92*, May 1992, pp. 2233-2240.

[11] Tanenbaum, A.S., *Computer Networks, 3rd Edition*, Prentice Hall, N.J., 1996.

[12] Tranchier, D.P., P.E. Boyer, Y.M. Rouaud, and J.Y. Mazeas, "Fast bandwidth allocation in ATM networks," *Proc. Int'l Switching Symp.*, 1992, pp. 7-11.

[13] Tzeng H.-Y. and K.-Y. Siu, "Comparison of performance among existing rate control schemes," *ATM Forum Contribution*, 94-1078, Nov. 1994.

[14] Varvarigos, E.A. and D.P. Bertsekas, "A conflict sense routing protocol and its performance for hypercubes," *IEEE Trans. Computers*, Vol. 45, no. 6, Jun. 1996, pp. 693-703.

[15] Varvarigos, E.A., and V. Sharma, "The ready-to-go virtual circuit protocol: a loss-free protocol for multigigabit networks," *IEEE/ACM Trans. Networking*, Vol. 5, no. 5, Oct. 1997, pp. 705-718.

[16] Varvarigos, E.A., "Control protocols for multigigabit-per-second networks," *IEICE Trans. Communications*, Vol. E-81B, no. 2, Feb. 1998, pp. 440-448.

[17] Varvarigos, E.A. and V. Sharma, "An efficient reservation connection control protocol for Gigabit networks," *Computer Networks and ISDN Systems*, Vol. 30, no. 12, Jul. 1998, pp. 1135-1156.

[18] Varvarigos, E.A. and C.-H. Yeh, "Network routing algorithms," *Encyclopedia of Electrical and Electronics Engineering*, John Wiley & Sons, 1999.

[19] Varvarigos, E. A. and J.P. Lang, "A virtual circuit deflection protocol," *IEEE/ACM Trans. Networking*, Vol. 7, no. 3, Jun. 1999, pp. 335-349.

[20] Yeh, C.-H., E.A. Varvarigos, B. Parhami, and V. Sharma, "Scalable communication protocols for high-speed networks," *Proc. Int'l Conf. Parallel and Distributed Computing and Systems*, Vol. 1, 1999, pp. 417-422.