Reservation-Based Session Routing for Broadband Communication Networks with Strict QoS Requirements

Chi-Hsiang Yeh, Emmanouel A. Varvarigos, Dimitri P. Bertsekas, and Hussein T. Mouftah

Abstract

In this paper, we propose novel features and algorithms for resource reservation and session/burst routing. In particular, we present the scalable resource reservation scheme (SRRS) for reservation in broadband networks and the constraint-based deflection routing with backtracking and bifurcation (CD/BB) for applications with strict quality of service (QoS) requirements. CD/BB routing consists of a collection of routing algorithms that are particularly suitable for networks that use reservations on a session-by-session or burst-by-burst basis. Important features of CD/BB routing include probe deflection routing with backtracking, bifurcated session/burst routing, probe flooding, and redirected routing. When combined with SRRS, they lead to several important advantages, such as guaranteed QoS, small buffer requirements, high throughput, and small latency under light load.

1. Introduction

Internet traffic has been growing exponentially for the last few years due to the rapid increase in user population and the proliferation of new applications. In the near future, the traffic volume is expected to become unprecedentedly high, fueled by the extraordinarily large number of future mobile Internet handsets, the bandwidth requirements of emerging multimedia applications, and the availability of high-speed home access networks. Among various advances in networking technologies, wavelength division multiplexing (WDM) has the capability to provide very high-bandwidth transmissions and is thus expected to dominate the future Internet core.

In addition to providing high bandwidth to network applications, another important emerging requirement for the future Internet is the provision of guaranteed quality of service (QoS) for real-time multimedia applications. The resource reservation protocol (RSVP) [4] is a signaling protocol that reserves the required resources for time-critical applications so that they can be served with a guaranteed minimum bandwidth and bounded delay and jitter, in order to meet the required QoS. Various extensions to RSVP are currently being standardized, while many other reservation protocols have also been proposed [3, 5, 10, 12]. In most of the previous reservation protocols, a connection has to be fully established before the transmission of any data packet can start. A problem with such protocols in future broadband networks is that the round-trip or one-way propagation delay required for establishing a connection is nonnegligible compared to the time required to transmit the data, which leads to low link utilization since the required capacity is reserved long before it is actually used, considerably affecting the maximum achievable throughput (see Fig. 1a). Another problem with such protocols is that the time required to set up the connection considerably increases the latency, which is undesirable and may be unacceptable to some applications, especially those involving latency-sensitive bursty traffic.

In this paper we present the *scalable resource reservation scheme (SRRS)* and the *constraint-based deflection routing with backtracking and bifurcation (CD/BB)* for QoS management over the next-generation Internet. SRRS and CD/BB routing can be viewed as an evolution of our previous protocols, including the conflict-sense routing (CSR) protocol [14] for multiprocessors, the ready-to-go virtual circuit (RGVC) protocol [15] and the efficient reservation virtual circuit (ERVC) protocol [16] for the Thunder and Lightening ATM network testbed, the virtual circuit deflection (VCD) protocol [17] to be used in the MOST all-optical Tera Switch under development at UCSB. The objective of this work is to develop scalable, flexible, and aggressive reservation mechanisms and routing algorithms that will enable the next-generation Internet (with multiprotocol label switching (MPLS) [9]) to efficiently support existing and emerging multimedia and data applications with strict QoS requirements. SRRS has three categories of features: (1) the *aggressive/optimistic/conservative transmission disciplines*, (2) *timed reservation* and *activated locking*, and (3) *adaptable reservation*. The timed reservation feature and the aggressive transmission discipline of SRRS can solve the aforementioned problems and guarantee very small call-blocking rate and packet-loss ratio without relying on buffering. Constraint-based routing and the resource reservation mechanisms also facilitate traffic engineering in the Internet. The features of SRRS and CD/BB routing are tightly integrated together to efficiently support each other in order to achieve the objective.

Another noticeable trend for the future Internet is the de-

Chi-Hsiang Yeh and Hussein T. Mouftah are with the Dept. of Electrical and Computer Engineering, Queen's University, Kingston, Ontario, K7L 3N6, Canada.

Emmanouel A. Varvarigos is with the Dept. of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106-9560, USA. Work performed while at TU Delft, Netherlands.

Dimitri P. Bertsekas is with the Laboratory for Information and Decision Systems, Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.

0-7695-0951-7/01 $10.00 © 2001 IEEE

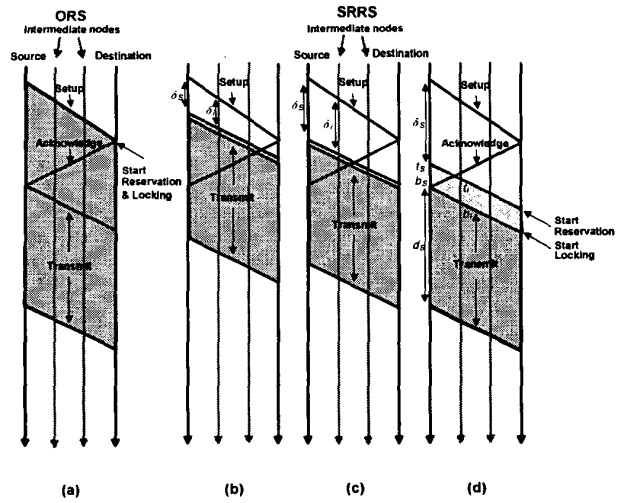

Figure 1. Comparison of SRRS and ordinary reservation schemes (ORS). The light shaded parts represent time segments during which capacity is reserved but not locked, while the dark shaded parts represent time segments during which capacity is both reserved and locked. (a) Ordinary reservation schemes. (b) SRRS with small δ_S to reduce latency. (c) SRRS with a medium value for δ_S to provide a tradeoff between latency and the probability that a session is not successfully established in time (in which case data packets may have to be dropped and the throughput may decrease). (d) SRRS with larger δ_S to increase throughput for latency-insensitive sessions or latency-sensitive messages with aggressive reservation. When conservative transmission is used, the acknowledgement has to be received at the source node before the transmission of any data packet can begin so that no packets are dropped due to buffer overflow. Note that in cases (b) and (c) the latency is smaller than in (d), but packets may be dropped unless an appropriate flow control mechanism, such as credit-first flow control, is used.

velopment and growth of wireless/mobile networks. The number of mobile handsets is expected to exceed that of wireline telephones in several years. The number of mobile wireless Internet devices including handsets, personal digital assistants (PDA), and laptop computers with wireless modems is expected to exceed that of computers with wireline connections; intelligent appliances (IA) and home/local-area networks (LAN) with wireless connections also have great potential. The adaptable reservation feature of SRRS makes it particularly suitable for signaling voice/multimedia applications in mobile wireless networks, where the bandwidth available to a mobile terminal changes due to handoffs, interference, and moving speeds. Moreover, adaptable reservation enables effective protection over QoS connections without wasting any resources unnecessarily. The aggressive transmission discipline also facilitates fast restoration after a fault occurs. These important applications constitute another drive for deploying an adaptable and aggressive reservation protocol like SRRS.

2. The scalable resource reservation scheme

In this section we present the scalable resource reservation scheme (SRRS) for efficient signaling and resource reservation. In the next section, we will then propose routing algorithms that are particularly suitable to be used in combination with SRRS.

2.1. The basic structure of SRRS

SRRS consists of seven phases. These phases are "pipelined" in the sense that a latter phase can start at a node before a former phase is completed in the network.

- **Phase 1 (Signaling Phase):** In order to establish a connection (i.e., a route with reserved capacity) for the transmission of data packets, the source node S (for source-initiated reservation) sends a setup packet for signaling.

- **Phase 2 (Reservation Phase):** Node i on the established part of a connection reserves the capacity required by the connection at reservation starting time t_i (according to its local clock).

- **Phase 3 (Locking Phase):** Node i on the established part of a connection locks the reserved capacity for tr he connection.

- **Phase 4 (Transmission Phase):** The source node S starts transmission at transmission starting time t_S (of its local clock) or when it receives an acknowledgement for binding (e.g., in MPLS networks [9]) or the establishment of the connection. Node i forwards the data packets after they are received.

  If the data packets catch up with the setup packet or arrive at node i before the reserved capacity is available (e.g., due to inaccuracy/error of clocks), node i performs one of the following actions: (1) dropping, (2) buffering, or (3) deflection.

- **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.

- **Phase 6 (Release Phase):** Node i on the connection release the reserved, allocated, and/or locked resources. Go to Phase 2 if there are reservation subintervals remaining for the session.

- **Phase 7 (Tear-down Phase):** Node i on the connection tears down the connection.

More details concerning these phases and several scalable reservation mechanisms can be found in [18]. In the following subsections, we present several advanced features that are particularly suitable for inclusion in SRRS. Most of them are *extensible features*, which are advanced features that do not have to be implemented at all network nodes in order to work correctly and efficiently.

2.2. Timed reservation and activated locking

We first investigate several issues concerning resource reservation and locking in Phases 1, 2, 3, 6, and 7 of SRRS.

2.2.1 Timed reservation

In SRRS, the reservation, allocation, and locking of network resources may be separated. More precisely, a fraction of the capacity of a link and other resources, such as buffer space, are reserved for a certain interval by each connection using the link, but the capacity reserved by a connection may still be used by other traffic, unless it is "locked" for that connection. The purpose of reservations is bookkeeping, while locking explicitly forbids other sessions from using the locked capacity.

When *timed reservation* is employed, the capacity is not reserved right after an intermediate node i receives a setup packet, but is reserved at time t_i, which is δ_i time units after the setup packet is received. δ_i is referred to as the *reservation lag time*; similarly, δ_S (associated with the transmission starting time t_S) is referred to as the *transmission lag time*. Note also that when the timed reservation concept is used, a connection can be established and scheduled for transmission at a later time when the required links are not available, rather than being rejected as in ordinary reservation-based protocols.

Depending on the underlying multiplexing and switching techniques and the actual implementation, we may or may not want to lock the reserved capacity. If locking is implemented, the reserved capacity or a fraction of the reserved capacity may be locked at time t_i (i.e., the starting time for the reserved interval) or at a later time depending on the option of the connection or the default of the node.

In order to reserve and utilize the capacity efficiently, each node maintains a *reservation profile*, which records the intervals during which capacity is reserved for each session. The reservation profile contains the curve for the reserved capacity with respect to time, which may be a straight line, a step function, or a more complicated function that changes values whenever applicable.

Several primitive forms of timed reservation have already been used in CSR [14] and ERVC [16], where the capacity is reserved for a time interval during which it is actually used for transmitting data. In CSR the reserved time interval is a time slot for a packet while in ERVC the reserved time interval is the duration of a session. A difference between SRRS and ERVC is that ERVC is designed for constant-bit-rate (CBR) traffic and the time interval reserved by ERVC is locked for the session at the same time. Another difference is that SRRS allows more complicated piecewise step functions to specify the reservation.

When the number of sessions is large, it may be prohibitively expensive to maintain a reservation profile that records the starting and/or ending times for all the sessions using timed reservation. To achieve better scalability, we may use *slotted reservation*, where the reserved intervals/subintervals are rounded to contain the minimum numbers of slots that cover the entire intervals/subintervals. Each slot only needs one or several counters to record the aggregate bandwidth reserved during that slot. The number of slots maintained is determined by the processing and storage capacity of a node as well as the intended overhead for refreshments. Note that the durations of slots are not equal to the transmission time of a packet and the former are usually considerably larger. The last slot maintained does not have an ending time (i.e., it extends to infinity), and has a dynamic starting time that is slid to a later time whenever the (original) first slot is passed and a new slot is added before the last slot. There are two types of reservations: *soft reservations* that have to be refreshed and will be canceled if not refreshed in time, and *hard reservations* that do not need to be refreshed. The bandwidth reserved in the last slot consists of *soft reservations* only, while other slots may contain either hard reservations, soft reservations, or both in general.

A new request for reservation or the refreshment for a soft reservation may contain a subinterval for hard reservation and another subinterval for soft reservation. The subinterval for hard reservation should not extend into the last slot maintained at a node; otherwise, special handling may be required for that request/refreshment. When a new request arrives, the node adds the additional *hard reserved bandwidth* to each of the slots in the subinterval for hard reservation and adds the additional *soft reserved bandwidth* to each of the slots in the subinterval for soft reservation; if the requested interval extends into the last slot, the node adds the additional soft reserved bandwidth to the last slot. When a refreshment arrives, the node adds the additional hard reserved bandwidth to the refreshed slots and subtracts the corresponding soft reserved bandwidth from those slots. When the (original) first slot is passed, a new slot is added before the (original) last slot, and the starting time of the last slot is slid to a later time, the new second last slot keeps the soft reserved bandwidth originally reserved in the last slot, but it can now accept hard reservations. Soft or hard reserved bandwidth may be released when a node receives a release/tear-down control packet indicating the reservation intervals/subintervals and the reserved bandwidth to be canceled.

If a session has made hard reservation for a certain bandwidth, it marks its packets (that are transmitted within the reserved bandwidth) using several bits to indicate that these packets should not be dropped or delayed. Since the required bandwidths for QoS sessions have been reserved and refreshed before they are used, QoS can be guaranteed for these sessions. It can also be seen that network nodes do not need to maintain per-flow states for reservations when slotted reservation is used so that the processing and storage requirements are flexible and independent of the number of sessions. Also, the total duration of the maintained slots except for the last one (i.e., the number of slots (except for the last one) times the duration of a slot) can be made reasonably large, so that the required number of refreshments per session and thus the processing overhead can be made small. Thus, slotted reservation can be scalable and provide QoS guarantees at the same time. When the number of slots maintained is not small, the required processing capability

may become high but still feasible. Note that the numbers of slots at different network nodes may be different, but it is preferred that the total durations of the maintained slots except for the last one are similar at all network nodes that use slotted reservation. Note also that the path of a connection may contain network nodes that use *unslotted timed reservation* and nodes that use slotted reservation as long as the setup packet and refreshments of the connection contains the information required for both types of reservations. More details concerning slotted reservation, its implementations, and the operation for a connection with both slotted and unslotted reservations will be reported in the near future.

2.2.2 Activated locking

The reserved capacity at node i or part of the capacity can be locked when an activation control packet or the first data packet is received (this case will be referred to as *activated locking*), and can be terminated when a release control packet, a tear-down control packet, or the last data packet is received. A characteristic of activated locking/allocation is that multiple activation control packets and release control packets may be sent during the lifetime of a session. An average transmission rate may be reserved, while a higher bandwidth may be activated when required. When bursts from many sessions are transmitted along the same link, statistical multiplexing can effectively share the bandwidth among these sessions. Note that the reserved interval may be made somewhat longer than the locked interval since the former does not waste as many network resources, and the reserved capacity can prevent (at least part of the) data packets from dropping due to overflow.

A comparison of the bandwidth reserved and locked by SRRS and ordinary reservation schemes is shown in Fig. 1. The light shaded areas of Figs. 1bcd represent the durations when capacity is reserved but not locked, while the dark shaded areas of Fig. 1 represent the durations when the capacity is both reserved and locked. It can be seen that when the round-trip propagation delay is not negligible, the capacity is more efficiently utilized in SRRS than in ordinary reservation schemes, since the resources of the former are reserved and locked only when necessary.

In order to achieve better scalability with low price and avoid the requirement for clock synchronization, a node can start reserving capacity since it receives a setup packet (instead of using timed/slotted reservation) but lock (or release) the reserved resources upon the reception a corresponding control packet (i.e., activated locking or releasing is used). In this way resources are locked only when necessary, leading to higher utilization, while no synchronized clock is needed and dropping of data packets due to inaccurate clocks can be avoided. Activated locking can also improve the efficiency of slotted reservation since the reserved intervals/subintervals are rounded to the boundary of slots so that the reserved bandwidth at the beginning and/or end of the slots may not be used.

2.3. Aggressive, optimistic, and conservative transmission disciplines

In some networking environments, a network node may expect a certain amount of additional capacity that will be requested by new connections in the next few time units. In some applications, a source node may know in advance that it will send a message some time later. Taking advantage of the timed-reservation feature of SRRS, an intermediate network node or a source node may reserve resources in advance for its coming requests or future messages so that when the requests for connections arrive or when the message is ready, data packets can be transmitted using the reserved capacity right away without waiting for a round-trip delay or any offset time as in previous reservation protocols. Moreover, for such a message, we can initiate reservation earlier and make δ_S and δ_i's larger without increasing the latency experienced by the message, so that the probability that data packets catch up with their setup packet is kept low, increasing the network throughput. We refer to this strategy as *aggressive reservation*. Permanent virtual circuits/paths that reserve fixed capacity form a special case of aggressive reservation. Another example is to reserve a certain amount of capacity for coming high priority sessions/bursts or handoffs in mobile networks.

If we insist that a source node does not transmit any data packets until it receives an acknowledgement from the destination and until its local time is at least t_S (the transmission starting time, which is used for computing reservation starting times t_i at intermediate nodes i), data packets are never dropped due to buffer overflow. This strategy may lead to higher throughput and is called *conservative transmission* (see Fig. 1d for an example). Note that resource reservation and locking are separated in SRRS, in contrast to being combined as in previous protocols such as UNITE, ERVC, and RGVC, so that resources do not stay idle unnecessarily between t_i and the time data packets/bursts actually arrive, which may be large when the traffic is heavy and conservative transmission is used. Combining aggressive reservation and conservative transmission leads to small latency and high throughput when applicable.

If aggressive reservation is not feasible (e.g., in an interactive application), we can still use *aggressive transmission* to reduce latency, where a source node can start transmitting data packets before it receives an acknowledgement for the establishment of the connection (see Fig. 1bc). If the data packets/bursts arrive before the reserved capacity is available, a certain action, such as deflection, buffering, or dropping, is taken to deal with these packets/bursts. Note that a typical and appropriately-operated SRRS network should have some sessions using conservative transmission if they are not sensitive to the latency for connection establishment, while some other sessions or bursts using aggressive transmission if they are latency sensitive and/or have higher priority. The conservative transmission discipline has the advantage that it does not waste any network resources through deflections, packet loss, or excessive buffer consumption unnecessarily. In Fig. 1 we compare typical situations of SRRS and ordinary reservation protocols. It can be seen that SRRS with aggressive transmission can considerably reduce the latency. Moreover, if timed reservation is not available and locking of capacity is required, aggressive transmission can also considerably increase link utilization and network throughput in broadband networks by reducing the duration links remain idle during call setup.

An important characteristic of aggressive transmission is its adjustable transmission lag time δ_S. By using a larger δ_S, deflection with backtracking can be used to find an ap-

propriate route with high probability (see Section 3.1). If δ_S is small as in RGVC [15], just enough time (JET) protocol [8], or other tell-and-go (TAG) typ of protocols [6, 13], all the data packets/bursts of a session will be sent back and forth when backtracking is used, wasting network resources. Also, if δ_S is sufficiently large (e.g., larger than the round-trip propagation delay of the upstream link), the setup packet does not need to reserve any bandwidth to the upstream node so that the success rate for connection establishment is considerably increased.

The *optimistic transmission discipline* is a strategy between the aggressive transmission discipline and the conservative transmission discipline. Under this discipline, a source node S transmits after it receives an acknowledgement (and the associated binding information) from the next node in the path and after its local clock has reached its transmission starting time t_S; similarly, an intermediate node i forward the received packets after it receives an acknowledge (and associated binding information) from the next node in the path and after its local clock has reached its reservation starting time t_i. This transmission discipline is particularly useful for reducing the numbers of table entries at intermediate nodes in MPLS networks [9], achieving better scalability. As a result, when the table space is limited, a session should use the optimistic transmission discipline when the increased latency (relative to that of the aggressive transmission discipline) is acceptable, while only higher-priority sessions should be permitted to use the aggressive transmission discipline.

2.4. Adaptable reservation

The reservation made by an SRRS connection may contain more than one set of QoS requirements, one for the normal mode and others for degraded modes. A node may then adapt to the traffic conditions and provide acceptable service quality to all the connections maximizing user satisfaction and minimizing aggregate penalty. This feature will be referred to as *adaptable reservation*.

When adaptable reservation is used for a connection, its setup packet is required to carry with it the resource requirements under the degraded modes, and the session has to be able to cope with such a degraded bandwidth enforced by the network when there are no sufficient network resources. Network nodes (including base stations) with such an advanced feature have to record the requirements of each connection or forwarding equivalence class (FEC), including the maximum tolerable delay and delay variation, the peak, average, and/or minimum bandwidths without degradation, and the bandwidth requirements under degradation level 1, level 2, and so on, along with the priority of the session/FEC and the penalty for each degradation level.

When there is sufficient bandwidth available, a network node allocates the required bandwidths without degradation to all the connections as ordinary communication protocols; when there is not sufficient bandwidth left, a network node first holds a certain amount of traffic that is time-noncritical. If this is still insufficient to accommodate the traffic for new connections, handoffs, and/or reactivating/renegotiating connections, the network node reallocates the required bandwidth eunder degradation to some connections with lower priority.

The network node should try to provide the required bandwidths without degradation to high-priority connections if possible, while reducing the bandwidths assigned to some (or even all) of the connections when necessary. A goal of the bandwidth reallocation algorithm is to minimize the aggregate penalty. Note that the possibility of such a degradation is within the agreement made between the network node and the sessions when they established the connections or when they renegotiated for new bandwidth. Degrading some services while still at a tolerable level can achieve higher satisfaction to all users as a whole, as compared to degrading, for example, real-time multimedia sessions without delay guarantees as in current best-effort networks. Also, it helps prevent resource wasting when the traffic is heavy and the resources are most needed.

In some networks it may be advantageous for an application to specify the circumstances for it to use a reduced bandwidth even when there is still bandwidth available in a wireless cell or at a network node. For example, a future mobile telephone company may offer lower price per Kbps in lightly-loaded cells, while asking for higher price in heavily-loaded cells. A user may then specify the ideal and/or degraded bandwidths for each of the price ranges. Other criteria such as power consumption may also be taken into account for determining the bandwidth to be used. We refer to this type of techniques as *economic reservation*.

We have incorporated the adaptable reservation feature in a medium access control (MAC) protocol proposed in [19]. The resultant MAC protocol can provision an acceptable QoS even when there are many handoffs and/or new mobile stations arrive at a cell and suddenly increase the aggregate bandwidth requirement considerably. We have also proposed an architecture for implementing the adaptation feature at the application layer for network/distributed applications [7].

2.5. The admit-and-forward discipline

When the processing time and queueing delay for the setup packet and/or other control packets is large, we may prefer to use the *admit-and-forward discipline* to speed-up the connection setup. Under this discipline, an intermediate node uses the minimum possible time to determine whether it will admit the request, and then forwards the setup packet to the outgoing link (with or without modifications). The setup packet can continue to be processed at the node for the reservation of the bandwidth, buffers etc. The processing for admittance and forwarding should be given higher priority so that several setup packets may be partially processed and forwarded first, postponing more time-consuming processing for a later time. This way the reservation starting time δ_S can be decreased (since it is lower bounded by the sum of the processing times for the setup packet of a connection before it is forwarded) and the latency is thus reduced.

This feature is particularly useful for a network implementing adaptable reservation, in that an optimization algorithm may have to be run to determine the reserved bandwidth and to reallocate the bandwidths for other sessions.

3. Probe-based session routing algorithms

It is straightforward to apply existing unicast and multicast algorithms to route sessions/bursts, and the basic features for routing in virtual circuit networks can be found in a variety of papers and books [1, 2, 11]. However, packet routing algorithms, which select routes on a packet-by-packet basis and have been intensively studied in the literature and widely implemented in practice, may be quite different from the best algorithms for performing session routing, which routes data packets on a session-by-session basis, especially when QoS requirements are taken into account. Also, several new features proposed for SRRS, such as timed reservation, enable the use of more powerful routing algorithms (e.g., probe flooding) with negligible increase in the traffic load.

In this section, we develop new session/burst routing algorithms that are particularly suitable to be used in combination with SRRS. These routing algorithms are extensible features in that they do not have to be implemented at every network node to work correctly and efficiently. For example, when probe flooding (see Subsection 3.3) is used in a network, a setup packet may spawn other setup packet(s) at network nodes that implement the feature to enhance performance, while not at network nodes that do not have this capability, and the routing function remains correct.

3.1. Probe deflection routing with backtracking

If a call is rejected whenever a link on the shortest path does not have sufficient bandwidth during the requested time interval, the call-setup rejection rate and the average latency will be large when the traffic is heavy; if, with a non-negligible probability, data packets (transmitted under the aggressive transmission discipline) catch up with their setup packet (which may be delayed for processing or waiting for an outgoing link on the shortest path) and are dropped, network resources are wasted due to retransmissions and the network throughput decreases. A method to mitigate the latter problem is to implement large buffers, at least at nodes that are performance bottlenecks. This, however, may be too expensive, especially for all-optical networks. Alternatively, a cost-effective method to solve both problems is to use deflection (hot-potato) routing for the setup packet, with data packets following the path on which the setup packet is deflected [17].

One way to implement probe deflection routing with SRRS is to send a deflection-enabling control packet from the source node along the established part of the connection some time before data packets are transmitted. If the connection is completely set up before the deflection-enabling control packet reaches the destination, the connection is established along the shortest path; otherwise, after the control packet catches up with the setup packet, the intermediate node i tries to find the most appropriate outgoing link among the outgoing links that have sufficient bandwidth BW available during the reservation interval(s), instead of the link on the shortest path. If no outgoing links with bandwidth BW are available, backtracking is used and the setup packet is sent back to upstream nodes to find an alternative route. In this way we try to reserve shortest paths when possible and avoid the dropping of data packets, both of which may increase throughput and reduce latency. Note that only the setup packet is sent back and forth in backtracking and the data packets are not. In summary, deflection in our probe deflection routing can be triggered

(i) by the first data packet,

(ii) by a locking control packet,

(iii) by a deflection-enabling control packet (we refer to items (i)-(iii) as *activated deflection*),

(iv) after a predetermined time (referred to as *timed deflection*),

(v) if the session has to wait longer than a threshold (by looking at the reservation profile at that node) for the desired outgoing link to become available (referred to as *conditional deflection*), or

(vi) whichever happens first among a subset of them,

depending on the type of the connection, the implementation of the network, and the decision made by the intermediate network node. Note that we can assign the predetermined time for a connection as $-\infty$ so that deflection is used from the very beginning.

Deflections in probe deflection routing happen on a session-by-session basis (referred to as session deflection) as is also done in the VCD protocol [17] or on a burst-by-burst basis (referred to as burst deflection), rather than on a packet-by-packet basis (referred to as datagram deflection). As argued in [17], session deflection can considerably reduce the reassembly cost compared to datagram deflection. A difference between probe deflection routing and VCD is that in probe deflection routing we send a setup packet (i.e., a probe) before data packets so that the probe may backtrack, if so desired, and find a better route without sending data packets back and forth and waste resources. This is particularly important for bursty traffic where a large bandwidth is requested, which may not be available at some intermediate nodes even when deflection routing is used, leading to dropping of data packets or preemptions of existing sessions [17] unless probe deflection routing is used. Therefore, the success rate for call setup can be considerably increased by appropriately selecting δ_S since the probability that none of the upstream nodes (when backtracking) have outgoing links with sufficient bandwidth is considerably lower when δ_S is not small. Also, the network throughput can be increased since resources are not wasted due to dropping or forwarding data packets back and forth or along a longer path. Another difference is that the probe in probe deflection routing may wait for a desired link that is currently unavailable so that the probability that shortest paths are used is higher, while the data packets in VCD without probes cannot wait at intermediate nodes unless a very large buffer space is available. A third difference is that an SRRS session does not always require a unique label or virtual channel identifier (VCI) and a dedicated table entry at every intermediate node in order achieve better scalability; instead, it may share a label with other sessions. This, however, requires additional mechanisms and control information when the routing path is deflected at least once. The scalability issues and several proposed mechanisms to solve the problems associated with

deflection and backtracking will be reported in the near future.

The deflection feature is particularly useful for latency-sensitive sessions. For latency-insensitive sessions, the source may specify in the setup packet an option other than deflection, such as dropping or buffering. An intermediate node may still override such an option and deflect the setup packet if the utilization of the required outgoing link is too high or when the locking control packet or the first data packet arrives, if so desired. Livelocks can be avoided by assigning an appropriate time-to-live (TTL) field in the setup packet so that the deflected packets are dropped after TTL.

3.2. Bifurcated session/burst routing

Even with deflection routing, an intermediate node may still have no links with sufficient bandwidth *BW* to accommodate a new session, or it may be preferable to send data packets along links other than the ones that have sufficient bandwidth *BW* (e.g., to reduce latency and/or increase throughput). In addition to backtracking in probe deflection routing, another possible strategy is to split a connection into several subconnections with an aggregate bandwidth equal to *BW*, and demultiplex the data packets to these subconnections. If the total outgoing bandwidth of a node is not smaller than the total incoming bandwidth, we can always find such a set of subconnections, assuming that redirected routing (Subsection 3.4) is implemented at nodes attached with hosts or the transmissions originating at the intermediate node where the connection is split can be throttled when necessary. Even if some connections cannot be throttled due to QoS constraints (when a host is attached to the intermediate node) and redirected routing (see Subsection 3.4) is not available, this advanced feature can still reduce the occurrence of packet dropping significantly. Note, however, that the intermediate node must then have the capability to spawn multiple setup packets and other control packets to continue setting up and controlling these subconnections, and to demultiplex the data packets over them; also, the destination host must have the capability to buffer and reassemble these split subsessions/subbursts. In networks where the implementation of such an advanced feature is too expensive, we may choose not to implement the function at some nodes and destinations.

An important application of bifurcated session/burst routing is to establish multiple paths for bursty traffic. We may split the bursty data appropriately at the source node so that the reassembly cost is reduced compared to the case where a session/burst is split and demultiplexed at an intermediate node. Since each path requires a considerably smaller bandwidth, the probability that some or all of them are established successfully is considerably higher than that of a single high-bandwidth connection. When only part of the requested paths are established successfully, we can still demultiplex the data to those established subconnections, which considerably improves the time required for transmission compared to a single low-bandwidth connection. Note that to use this strategy, the requested duration should be made longer, and/or the aggregate bandwidth requested by these paths should exceed the required bandwidth by a certain amount. This, however, will not waste network resources when timed reservation is used and the reserved interval and/or the reserved bandwidth are adjusted before/when the data packets arrive. We can implement bifurcated session/burst routing at source nodes only, or at both source nodes and intermediate nodes. In the latter case, we can also request a single high-bandwidth connection at the beginning, and split the connection only when necessary.

3.3. Probe flooding for small latency and reliability

In addition to establishing multiple connections for bifurcated session/burst routing, multiple setup packets can be utilized to improve performance in several other ways. In particular, when only a fraction of the requested bandwidth is allocated under a degraded mode (when adaptable reservation is used), or when a setup packet is waiting at an intermediate node for sufficient capacity of the link on the shortest path to become available, the node may spawn another or several setup packets to find one or several alternative routes that have sufficient bandwidth for the same session. If the link on the shortest path becomes available before data packets arrive, they are routed over that link and the intermediate node sends a control packet to terminate the alternative route(s); otherwise, they are routed along one or several alternative routes. In the latter case the original setup packet can either be dropped or be used to establish a better route, which may be used for rerouting in Phase 5 of SRRS if the reserved interval is long, the QoS requirements can still be met, the network traffic is heavy, and the resources consumed by the alternative route are significantly larger than the minimum required.

The above feature is particularly useful for sessions with large holding times. For sessions that are not latency-sensitive, conservative transmission can be used so that only the setup packets, rather the data packets, are sent along alternative routes. For a very high-priority and latency-sensitive session, aggressive transmission can be used so that the intermediate node may copy the first few data packets to more than one routes, provided that the destination node has the capability to drop duplicate data. After a connection to the destination node is established, the destination node may send control packets to terminate all but one connection (e.g., the first one that reaches the destination and satisfies the QoS requirements). When timed reservation is implemented, the former strategy barely wastes any resources since the reservations are canceled before they start. In this way, advantages similar to those of (selective) flooding can be attained, while considerably smaller network bandwidth is consumed. We refer to the approach as *probe flooding*. Probe flooding may also be useful in networks whose topology changes frequently with time, such as wireless/mobile networks. Multiple setup packets can also be used for multicast.

3.4. Redirected routing

When no security problems can arise (e.g., data are encrypted) and some additional delay can be tolerated, an egress/edge router or a host attached to or near an intermediate node may intercept the data packets by pretending that it is the destination node and initializing a new connection to forward the data packets to the final destination when the required links become available. The probe is used

to find such a router/host that has the capability to receive and store large messages/bursts on-the-fly. The intermediate node may then send a control packet to the source node, informing it to stop or slow down transmissions in order to "reconnect" the two connections. Such a routing option does not waste network resources when the outgoing and incoming links connected to the router/host are underutilized, while implementing it may considerably increase the network throughput; it may also be useful in wireless/mobile networks, when arriving packets find that the destination node has moved to a new location/cell or is temporarily unavailable.

3.5. Constraint-based deflection routing with backtracking and bifurcation

In this subsection we present *constraint-based deflection with backtracking and bifurcation (CD/BB)* routing, which combines constraint-based routing and the features in the preceding subsections.

In CD/BB routing, a session that has QoS constraints uses a constraint-based routing algorithm to determine the explicit route(s) for the session. The information concerning the loose or tight explicit path is carried by its setup packet and the setup packet is forwarded along the specified path by intermediate nodes. If a session does not have special QoS requirements or if its requirements can be satisfied by an existing forwarding equivalence class (FEC) or virtual path, its setup packet only carries an appropriate label or virtual path identifier (VPI)/VCI and is forwarded according to the binding information at the intermediate nodes.

If an intermediate node finds that the outgoing link specified in the explicit path or routing/switching table does not have sufficient bandwidth, it uses probe deflection with backtracking to find an alternative path. The deflected route may be found using a constraint-based routing algorithm again if time permits, or using a simpler and faster algorithm. If none of the outgoing links have sufficient bandwidth, the intermediate node may use bifurcated routing to split the session if this is allowed for the destination of the session, which may be indicated in the setup packet or recorded in the local information base. If the session has strict QoS requirement, the intermediate node may use probe flooding (i.e., another form of bifurcation) to increase the probability of finding an appropriate route. Redirected routing may also be employed if it is appropriate and an egress/edge router or host is available. Note that an intermediate node does not have to support all these features, especially when it is not expected to form a performance bottleneck. Also, a setup packet may carry instructions indicating the preferred way for the session to be handled.

4. Conclusions

In this paper, we presented SRRS for signaling and reservation in broadband communication networks and CD/BB routing for in reservation-based networks with strict QoS requirements. SRRS and CD/BB routing combine important advantages of several previous reservation and routing protocols such as guaranteed QoS, loss-free communication, high throughput, and small latency. Most features of SRRS and CD/BB routing are extensible, facilitating the manufacture of inexpensive routers/switches and network interface cards for network nodes and hosts that do not have to implement the more advanced features.

References

[1] Ali, I.A., H.T. Mouftah, and A.H. El-Sawi, "A dynamic routing protocol for broadband networks," *IEEE Symp. Computers and Communications*, Jul. 1997, pp. 495-500.

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

[3] 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.

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

[5] 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.

[6] Hjalmtysson, G. and K.K. Ramakrishnan, "UNITE–an architecture for lightweight signaling in ATM networks," *Proc. IEEE INFOCOM'98*, 1998, pp. 832-840.

[7] Malamos, A., T. Varvarigou, E. Malamas, and C.-H. Yeh, "MEQA^3 – A multi-end QoS application adaptation architecture," *Int'l J. for Computer Research*, to appear.

[8] Qiao, C. and M. Yoo, "Choices, features and issues in optical burst switching," *Optical Networks*, to appear.

[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] Turner, J.S. "WDM Burst Switching for Petabit Networks," *Proc. of OFC*, 2000, to appear.

[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. 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.

[17] 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.

[18] Yeh, C.-H. and E.A. Varvarigos, "The scalable networking scheme for high-speed networks," *IEEE Int'l Conf. Communications (IEEE ICC'2000)*, 2000, pp. 1335-1342.

[19] Yeh, C.-H. and O.Y. Chan, "A QoS-guaranteed MAC protocol for wireless/mobile communications," *Proc. Int'l Conf. Information Technology and Communication at the Dawn of the New Millennium*, 2000, pp. 411-420.