MULTICAST

INTRODUCTION

One of the ways communication can be characterized is by the *number* of parties involved. Traditional communication modes have been *unicast*, i.e., one-to-one, and *broadcast*, i.e., one-to-all. Between these two extremes we find multicast, the transmission of a single message or data stream to a set of receivers. Thus, multicast is a generalization of both unicast and broadcast and a unifying communication mode. For this reason it is receiving increasing attention in modern networking architectures.

The above definition of multicast is still traditional in the sense that it is sender-centric and unidirectional. An even more general term would be *multipoint communication*. Note also that multicast is considered mostly in the context of digital, and particularly, packet-switching networks.

Multicast is examined on its own because the specification of receivers through a set introduces features and complications that are not present in traditional communication modes. We can distinguish two cases of increasing complexity with respect to the set of receivers: (1) it is fixed and known (e.g., to the sender), and (2) it is unknown and/or dynamic.

The multicast model of communication supports applications where data and control are partitioned over multiple actors, such as updating replicated databases, contacting any one of a group of distributed servers of which the composition is unknown (more appropriately termed *anycast*), and interprocess communication among cooperating processes, e.g., distributing intermediate computational results from one processor to others in parallel computers.

A demanding application of multicast is Distributed Interactive Simulation (*DIS*). Targeted news and information distribution in near real-time has potential global impact and could normally be less demanding, even though specific applications, such as stock-quote information distribution, might have very stringent requirements (e.g., needing *atomic multicast*, the semantics of which imply that the message should be received by all receivers in the group or none at all). However, the prototypical multipoint communication application is probably real-time interactive multimedia tele-conferencing, possibly including shared workspaces, and being under the umbrella of Computer Supported Collaborative Work (*CSCW*).

Efficient multicast is a fundamental issue for the success of group applications. Here the selective multicast service takes the place of the indiscriminate broadcasting so as to reduce the waste of resources caused by transmitting all the information or channels to all receivers. The basic means of conserving resources via multicast is *sharing*: instead of transmitting information from a sender to each receiver separately, we can arrange for routes that share links to carry the information only once over the shared links. We can picture a multicast route as a *tree* rooted at the sender with a receiver at each leaf and possibly some receivers on internal nodes. The tree can be designed so as to maximize shared links and thus minimize resource consumption.

While multicast is increasingly recognized as a valuable service for packet networks, many architectures still do not support it directly. Many shared-medium networks such as Ethernet support options for broadcast and multicast packets and the corresponding addressing mechanisms. However, processors are often required to perform extra processing when receiving a multicast packet. When switches are used in point-to-point networks, we would like the hardware to automatically recognize multicast addresses as such and transmit multicast packets through multiple links copying packets on the fly, as required. ATM switch designs increasingly support parallel transmission of multicast cells over multiple links in hardware, increasing peak switching speeds.

Another set of issues is concerned with extending the feedback mechanisms employed by unicast-oriented protocols to deal with flow, congestion and error control. For example, transport-layer protocols such as TCP adapt their behavior according to the prevailing network conditions at any given point in time by measuring loss rates as experienced by receivers. When extending these protocols for multicast, there is the possibility of *feedback implosion* when many receivers send such reports towards the sender, thus swamping the network and the source with control information. Apart from the obvious scalability problems of such schemes, there is also the issue of how to adapt the sender's behavior when conflicting reports arrive from the various receivers.

Multicast Groups and their Dynamics

The difference between multicasting and separately unicasting data to several destinations is best captured by the Internet host group model: a host group is a set of network entities sharing a common identifying multicast address, all receiving (traditionally, via best-effort service) any data packets addressed to this multicast address by senders that may (closed group) or may not be members of the group (open group) and have no knowledge of the group's membership. This definition implies that the behavior of the group over time is unrestricted in multiple dimensions; it may have local (*LAN*) or global (*WAN*) membership, be transient or persistent in time, and have constant or varying membership. From the sender's point of view, this model reduces the multicast service interface to a unicast one. This implies that the network software is accordingly burdened with the task of managing the multicasts in a manner transparent to the users. From the network designer's point of view, this extra work is expected to result in a more efficient usage of resources. This is the primary motive for network providers to support multicast in the first place.

These goals for multicast service impose specific requirements for the network implementation. First, there must be a means for routing packets from a sender to all group members whenever the destination address of a packet is a multicast one, which implies that the network must locate all members of the relevant group and make routing arrangements. Second, since group member-

J. Webster (ed.), *Wiley Encyclopedia of Electrical and Electronics Engineering*.

ship is dynamic, the network must also continuously track current membership during a session's lifetime, which can range from a short to a very long period of time. Tracking is required both to start forwarding data to new group members and for stopping the wasteful transmission of packets to destinations that have left the group. Both tasks must be carried out without assistance from the sending entity as defined by the host-group model. The dynamic nature of multicast groups has important implications for multicast routing.

A very different model is adopted by the Asynchronous Transfer Mode (*ATM*) technology. The first and currently the only supported model is that of a point-to-multipoint Virtual Channel (*VC*), where the source signaling agent knows exactly the addresses of all destinations, which are included in the VC, typically by being added one-by-one, during the initial connection set-up time. Proposals for receiver-initiated dynamic modifications to VCs are being investigated.

MULTIPOINT ROUTING AND MULTICAST TREE ALGORITHMS

Unicast routing intends to minimize transmission cost or delay, depending on the metric used for the optimization. These two apparently different goals are equivalent from an algorithmic point of view, both leading to the use of shortest-path algorithms, with Dijkstra's and the Bellman-Ford algorithm the two common cases. These algorithms find optimal routes between one node (the sender) and all other nodes in the network (including all receivers) in the form of shortest path trees. Thus, a straightforward (but not optimal) solution to the multicast routing problem can be based on the shortest path trees produced by these algorithms by pruning off any branches that do not lead to any receivers in the group.

Although details vary according to the base algorithm, there are some observations that generally apply. On the up side, these algorithms are easy to implement, as direct extensions of existing ones, and thus fast to deploy. Additionally, each path is optimal by definition, regardless of changes in group membership, and this optimality comes essentially for free since shortest paths need to be computed for unicast routing as well. On the down side, these algorithms optimize the wrong metric. Also, for large internetworks with widely dispersed groups, either the scale of the network or continuous network changes restrict their use to subnetworks that already employ their unicast counterparts. Similar problems (e.g. processing complexity for Dijkstra and instability for Bellman-Ford) have also forced unicast routing algorithms to rely on hierarchical routing techniques for large networks.

Cost optimization in multicast can be viewed from another angle: overall cost optimization for the distribution tree. The shortest path algorithms concentrate on pairwise optimizations between the source and each destination and only conserve resources as a side effect, when paths overlap. We can instead try to build a tree that exploits link sharing as much as possible, and by duplicating packets only when paths diverge, minimize total distribution cost, even at the expense of serving some receivers over longer paths. What we need is a tree that reaches all receivers and may use any additional network nodes on the way. This is equivalent to the Steiner tree problem, where a cost-labeled graph and a set of nodes, the Steiner points, are given and we want a minimal-cost tree connecting all Steiner points, consisting of the sender and the receivers. Note that if all the nodes in the graph were Steiner points, the problem coincides with finding a spanning tree for the graph, for which efficient algorithms are known. Instead the Steiner tree problem is intractable.

However, even though Garey *et al.* (10) have shown that this problem is NP-complete, approximation algorithms exist with proven constant worst-case bounds. Implementations of such algorithms have been shown to produce low-cost multicast trees with very good average behavior. As an example, trees built with the heuristic by Kou *et al.* (15) have at most twice the cost of Steiner trees, while simulations of realistic network topologies have shown their cost to be within 5% of the optimum. The advantage of this approach is its overall optimality with respect to a single cost metric, such as transmission cost. However, the disadvantages are also important: the algorithm needs to run in addition to the unicast algorithms, and it will itself have scaling problems for large networks. Furthermore, optimality is generally lost after group membership changes and network reconfigurations if the tree is not recomputed from scratch. Thus, Steiner tree algorithms are best suited to static or slowly changing environments since changes lead to expensive recalculations to regain optimality.

Both approaches discussed above suffer from an inability to maintain their measure of optimality in large and dynamic networks. Approaches for extending these algorithms to deal with changes in group membership without complete tree reconfigurations include extending an existing tree in the cheapest way possible to support a new group member and pruning the redundant branches of the tree when a group member departs. The quality of the trees after several local modifications of this sort will deteriorate over time, eventually leading to a need for global tree reconfiguration.

A different approach to the routing problem opts for a solution in realistic settings by adopting the practical goal of finding good rather than optimal trees that can also be easily maintained. The departure point for this approach is the center-based tree, which is an optimal-cost tree that, instead of being rooted at the sender, is rooted at the topological center of the receivers. Even though such a tree may not be optimal for any one sender, it can be proven to be an adequate approximation for all of them together. The implication is that one basic tree can serve as a common infrastructure for all senders. Thus, maintenance of the tree is greatly simplified and nodes on the tree need only maintain state for one shared tree rather than many source-rooted trees. Since this method has been developed for broadcasting rather than multicasting, the theoretical investigation does not hold when we prune the broadcast trees to get multicast ones. In addition, the topological center of the tree, apart from being hard to find (the problem being NP-complete), will not even be of use in a dynamic multicast environment.

Practical proposals for multicast routing abandon the concrete optimality claims discussed above, but keep the basic idea of having a single shared multicast tree for all senders to a group. This is a departure from approaches that build one tree for each sender. Routing is then performed by defining one or more core or rendez-vous points to serve as the basis for tree construction and adding branches by separately routing packets optimally (in the unicast sense) from the senders to these fixed points and then from there to the receivers. Again, merging of paths is exploited whenever possible, but it is not an explicit goal of the routing calculations. Instead, because of the concentration of paths around the fixed points, common paths are expected to arise.

A single shared multicast tree is not optimal in any strict sense since no attempt is made to find the topological center of the tree, both due to its computational cost and the limited lifetime of any topological center for a dynamic environment. But, the advantages of shared multicast trees are numerous. First, a shared tree for the whole group means that this approach scales well in terms of maintenance costs as the number of senders increases. Actually, there is still a tree emanating from each sender, but all these trees merge near the fixed points and the distribution mesh is common from there on to the receivers. Second, the trees can be made quite efficient by clever choice of the fixed points. Third, routing is performed independently for each sender and receiver, with entering and departing receivers influencing only their own path to the fixed points of the single shared tree, employing any underlying mechanism available for unicast routing. This last property means that network and group membership dynamics can be dealt with without global recalculations and by using available mechanisms.

In practice, these multicast algorithms are expected to use the underlying unicast algorithms, but are independent of them. Interoperability with different unicast schemes, coupled with the scalability of the shared trees, make these algorithms ideal for use on very large-scale heterogeneous networks. The fixed points can also be selected so as to facilitate hierarchical routing for very large internetworks, further enhancing scalability properties.

Group dynamics are an obstacle in maintaining optimality, whatever the method of constructing the initial trees. Since repeating all routing computations whenever members join or leave the group may be prohibitively expensive, an alternative is to prune extraneous links when a member leaves the group, and add the most economical extension path towards a new member, either from a fixed or from the optimal location in the existing group. Rather than making modifications blindly, the most advanced algorithms store some of the state accumulated during tree construction and make only local calculations that still satisfy the requirements of the application.

However, simulations have shown that even simple multicast routing using the shortest path tree is not significantly worse in terms of total tree cost from the optimal solutions or the near-optimal heuristics. For realistic network topologies Doar and Leslie (9) have found that the cost of a shortest path tree is less than 50% larger than that of a near-optimal heuristic tree, while path delays for heuristic trees are 30% to 70% larger than shortest path delays. Since shortest path trees are easily built and modified using the underlying unicast routing and they never deteriorate in terms of delay, but simply vary in their inefficiency in terms of total cost, if an application is prepared to accept the overhead, it can avoid special multicast-tree construction and maintenance methods by employing the shortest delay paths.

A similar cost versus simplicity trade-off is involved when using shared trees for all senders to a group. For shared trees, optimality is hard to achieve and even harder to maintain, as discussed earlier, but a simple approach is to choose the center among group members in a way so that only as many trees as group members will have to be considered. For these trees, when path delay is optimized, simulations show that delays are close to 20% larger than the shortest paths, and tree cost is about 10% lower than that of shortest path trees. Furthermore, a single tree constructed using the underlying unicast routing mechanisms minimizes state and maintenance overhead. Unfortunately, apart from their moderate sub-optimality, shared trees also suffer from traffic concentration, since they route data from all senders through the same links. Simulations show that delay-optimal member-centered shared trees can cause maximum link loads to be up to 30% larger than in a shortest path tree.

For these reasons, recent proposals try to combine shared trees and shortest paths by starting each group connection in the shared tree mode and then changing individual paths to shortest delay ones upon receiver requests. This approach can also support traditional source-rooted trees. Additional complications arise when links and paths are asymmetrical. Most of the algorithms and approaches discussed above need modifications and in some cases they do not apply at all.

An interesting problem arising from resource sharing in multicast is how to split the total distribution costs among the receivers, or how to allocate the savings compared to using separate unicasts. This issue is orthogonal to the problem of what the total costs are, a question also arising in the unicast case. Whether these costs are used for pricing or for informational purposes, they are a primary incentive to use multicast.

Multicast Routing with Quality-of-Service Constraints

The motivation for routing multicast traffic along trees rather than along arbitrary paths is to minimize transmission cost through link sharing. For continuous media, the volume of data transferred makes this goal even more important. However, for real-time multimedia applications we must take into account two additional factors: delay constraints, particularly for interactive applications, and media heterogeneity. Separate handling of media streams is useful in order to use the most effective coding techniques for each stream. The question arises then whether we should use the same or separate distribution trees for each stream. Considering the load that continuous media puts on network links and the interaction among admission control and routing, it seems better to use separate trees. Thus, each media stream could ask for the appro-

priate Quality-of-Service (*QoS*) parameters and get routed accordingly, with receivers choosing to connect to any subset of the trees. On the other hand, the management overhead of multiple trees per source may be prohibitive. In addition, routing each media stream separately exacerbates the inter-media synchronization problem.

Turning to delay requirements, if we use delay as the link metric during routing, we can easily see that the shortest delay tree, made up from the shortest paths from sender to each receiver, is not the same as the tree of total minimal cost that maximizes link sharing at the expense of individual path delays. We have then a global tree metric (tree cost) and many individual receiver-oriented metrics (path delays) that are potentially in conflict. Since we cannot hope to optimize on all fronts, we can try to optimize cost subject to the constraint that delay is tolerable. Interactive applications can be characterized by upper bounds on end-to-end delay and/or limits on jitter. In this sense, it is reasonable to design the tree so as to optimize total cost while keeping individual paths within their respective bounds. Normally, all receivers would be satisfied by the same limits, as these are determined by human perception properties.

This new problem is essentially a version of the Steiner tree problem with additional constraints on the paths. Even though it is NP-complete, fast heuristic algorithms that are nearly optimal have been developed. Almost identical formulations could be obtained when the constraints are delay jitter or a probabilistic reliability constraint. For example the latter could be modeled in the case of independent link losses by a loss probability assigned to each link. Then, using logarithms the reliability metric could be expressed in linear form between a source and each destination by adding the logarithms along the path. This maps the problem to the previous one since the goal is tree cost minimization with a constraint on additive path-based metric. Finally, a similar formulation can be used when the constraint is link capacities which must not be exceeded, instead of a delay bound. Again, heuristics exist to solve this variant of the problem.

FEEDBACK CONTROL AND RELIABLE MULTICAST

Whether a network provides a simple connectionless service or a complicated connection-oriented service for unicast, generalizing it for multicast is not trivial. Flow, congestion, and error control depend on feedback to the sender, according to network and receiver-triggered events. For simple network services, no such information is provided by the network itself, but instead end-to-end reports must be exchanged.

Error control ensures that packets transmitted by the sender are received correctly. Packets may be received corrupted (detected by error-detection codes) or they may be lost (detected by missing sequence numbers). Flow control assures that the sender does not swamp the receiver with data that cannot be consumed in time. Congestion control deals again with the problem of insufficient resources, but this time at network elements between sender and receiver. Although packets may be dropped at intermediate nodes, in many networks this loss can be detected only by the receiver, resulting in confusion between errors and congestion.

In the unicast case, lost or corrupted packets are retransmitted based on feedback received from the network or the receiver. When packets are multicast, simple feedback schemes face the feedback-implosion problem: all receivers respond with status information, swamping the sender with possibly conflicting reports. Ideally, senders would like to deal with the multicast group as a whole and not on an individual receiver basis, following the host-group model. However, the sender cannot simply treat all receivers identically, because this would lead to either ignoring the retransmission requests of some receivers, or to wasting resources by retransmitting to all of them.

Since there is no evident solution that satisfies all requirements, several approaches exist emphasizing different goals. The simplest approach of all is to ignore the problem at the network layer and provide a best-effort connectionless service. Delegating the resolution of transmission problems to the higher layers may be an adequate solution in many cases, since they may have additional information about the application requirements and thus can implement more appropriate mechanisms than what is possible at this layer.

A second solution sacrifices the host-group model's simplicity by keeping per-receiver state during multicasts. After transmitting a multicast packet, the sender waits until a stable state is reached before sending the next one. For flow control, this slows down the sender enough so as not to swamp the slowest receiver. For error control, retransmissions are made until all receivers receive the data. This may not be possible even after multiple retransmissions, so the sender may have to take special action, e.g., removing some receivers from the group. Retransmissions may be multicast when many receivers lose a packet, or unicast when few do. Since feedback implosion is always a possibility, all such schemes should use negative rather than positive acknowledgments, i.e., send responses when problems occur rather than confirming that packets are received correctly and in time. In a negative acknowledgment scheme, some responsibilities are moved to the receivers, complicating their operation. However, additional opportunities arise, such as multicasting the negative acknowledgments to all receivers after random periods of time to minimize the number of negative acknowledgments returned to the sender. Assigning such responsibilities to receivers can lead to higher throughput.

However, the scalability of such schemes is doubtful, even for very reliable links and rare congestion or overflow problems. The problem is that the sender is still the control center, and as the number of group members grows, receivers and network paths become more heterogeneous. With these essentially symmetric schemes, the service provided to a group member is the lowest common denominator, which may be the slowest or most overloaded receiver, or the slowest or most congested network link. Sophisticated approaches exist that follow these general directions, but their complexity and inefficiency makes them appropriate only for applications that require very high reliability and uniform member treatment. Note that such reliable

solutions can be implemented as transport services over a simple connectionless network service.

A third solution is to distribute the feedback control mechanism over the entire multicast tree, and follow a hierarchical scheme. A receiver's feedback need not propagate all the way to the sender. Instead, intermediate nodes may either respond directly or merge the feedback from many downstream receivers to a summary message and then recursively propagate it upwards. In this case, feedback implosion is avoided in terms of messages, but the problem of dealing with possibly conflicting requests remains. If the added complexity of making local decisions on each network node (not only group members) is acceptable, we can narrow down the impact of problems to specific parts of the tree, relieving the sender from dealing with individual receivers.

A non-hierarchical method for distributed feedback control targeted to recovery of lost messages is to let all receivers and senders cooperate in handling losses, thus extending the sender-oriented model. When receivers discover a loss, they multicast a retransmission request, and anyone that has that message can multicast it again. To avoid feedback implosion, these requests and replies are sent after a fixed delay based on the distance from the source of the message or the source of the request respectively, plus a (bounded) randomized delay. The result is that most duplicate requests and replies are suppressed by the reception of the first multicasts. By varying the random-delay intervals, the desired balance among recovery delay and duplicates can be achieved. In contrast to hierarchical schemes and because location-independent multicasts are used, only group members participate but recovery cannot be localized without additional mechanisms.

A scalable feedback mechanism that can be used to estimate network conditions without creating implosion problems has been proposed by Bolot *et al.* (4): it first estimates the number of receivers in a group and then what the average quality of reception is (the averaging depends on the application), using probabilistic techniques. This method has been used in applications for senders to detect congestion problems and adapt their output rates (to relieve congestion) and error redundancy factors (to increase the chances of error recovery). Cheung and Ammar (5), proposed a further enhancement to scalable feedback control, by splitting the receivers in groups according to their reception status and capabilities and only sending them the data that each group can handle. This avoids problems created by very slow or very fast machines dragging the whole group towards one extreme.

Finally, another approach (mostly orthogonal to the above), tries to minimize the need for feedback by taking preventive rather than corrective action. For error control, this is achieved by using Forward Error Correction (*FEC*) rather than simple error detection codes. For flow and congestion control, this is achieved by reserving resources so that both receivers and intermediate network nodes are able to support the sender's data rate. The cost of these techniques are increased overhead and network complexity. FEC imposes processing and transmission overhead, but requires no additional mechanisms in the network. Resource reservation on the other hand needs additional control mechanisms to set up and maintain the resources for a session.

Message Ordering and Atomic Multicast

Some applications require delivery of messages in order. In some cases this requirement is expressed across sources, leading to a synchronization problem. The required ordering of messages could be *causal* or *total*. Causal ordering is based on the "happens before" relationship and might not be total (i.e., messages could be concurrent). A multicast protocol that ensures reliability and total ordering is called *atomic*. Such protocols might be necessary for secure distributed computing in the presence of failures and malicious agents, while causal ordering is sufficient to ensure consistency in updates to replicated databases.

MULTICASTING REAL-TIME CONTINUOUS MEDIA

Host and Network Heterogeneity

Several representational formats for various media types coexist. This is a problem with traditional data communications, but it is more of an issue with images, audio and video. Such issues are typically addressed at the presentation layer in the OSI model. Translation between formats can be provided at three points: at the transmitter, at the receiver, or inside the network. In the latter case, format converters are required to be deployed in the network. This may be appropriate for converting protocols or text encodings between autonomous systems with different standards (placing the converters in the gateways), but it is not effective when the terminals themselves can use different encodings within the same area. Thus, it is more realistic to move the translation services to the hosts.

With unicast, translation can be effectively done at either the sender or at the receiver. However, heterogeneity problems are aggravated with multicast. For example, translation at the sender requires the stream to be duplicated and translated for each different type of receiver, precluding link sharing over common paths. This approach also does not scale for large heterogeneous groups since the sender's resources are limited. Finally, it requires the sender to be aware of the receiver's capabilities, which is incompatible with the host-group model. The sender may use different multicast groups for each encoding to avoid this, but the other problems remain. Translation at the receiver is the most economical and scalable approach in this case since it fully exploits sharing and moves responsibilities away from the sender.

Since continuous media impose heavy demands on both networks and hosts, it is likely that not all receivers will be able to receive all of a sender's traffic. This argues in favor of prioritization of the traffic generated through hierarchical coding. Hierarchical or layered coding techniques decompose a signal into independent or hierarchically dependent components that can be used in specific subsets to provide partial reconstruction of the signal. In this case receivers can choose to get only those parts of the media that they can use or are most important to them. Thus, appropriate hierarchical coding can be easily combined with

and facilitates translation and reconstruction of the signal at the receivers, according to their needs and abilities. For example, a high resolution component of a video could be dropped from a congested subnetwork, allowing low resolution components to be received and displayed in that subnetwork, but without impacting other subnetworks that are not congested.

Resource Reservations

Resource reservations at the network switches are needed if any service guarantees are to be provided. The exact nature of these reservations differ according to the required service guarantees and the approach taken towards satisfying them, so resource reservation along transmission paths could be viewed as a subset of the general switch-state establishment mechanisms. An alternative to reserving resources for an indefinite period of time during connection establishment is to make advance reservations for a future connection with a given lifetime. This allows more sessions to be admitted (due to their deterministic timing) and also permits negative responses for reservation requests to be dealt with more gracefully.

The first component of resource reservation schemes is a specification model for describing flow characteristics, that depends heavily on the model of service guarantees supported by the network. Then, an appropriate protocol is required to communicate these specifications to the receivers and reserve resources on the transmission path so that the service parameters requested can be supported. Simple unicast approaches to resource reservations are generally source-based. A set-up message containing the flow specification is sent to the destination with the intermediate nodes committing adequate resources for the connection, if available. Resources are normally over-allocated early on in the path, so that even if switches encountered further along the path are short of resources, the connection can still be set up. After the set-up message reaches its destination, assuming the connection can be admitted along the path, a response message is returned on the reverse path, allowing the intermediate switches to relax commitments in some cases.

Similarly, for multicast, there must be a way for senders to notify receivers of their properties, so that appropriate reservations can be made. In a perfectly homogeneous environment, the reservations will be made once on each outgoing link of a switch for all downstream receivers, so that resource usage can be minimized. Reserved resources can also be shared among data transmitted from multiple senders to the same group (e.g., in applications such as conferencing where the number of simultaneous senders is much smaller than the total). However, receiver and network heterogeneity often prohibits use of this simplistic scheme. One approach is to allocate resources as before during the first message's trip and then have all receivers send back their relaxation (or rejection) messages. Each switch that acts as a junction will only propagate towards the source the most restrictive relaxation among all those received. However, since paths from such junctions towards receivers may have committed more resources than are now needed, additional passes will be required for convergence, or resources will be wasted. To handle dynamic groups without constant source intervention, this model can be augmented with receiver-initiated reservations that propagate towards an already established distribution tree.

An alternative approach is to abandon reservations during the sender's multicast set-up message and instead reserve resources based on the modified specifications with which the receivers respond to the initial message. Again, resource reservations will be merged on junction points, but since the (now upstream) requests are expected to be heterogeneous, each junction will reserve adequate resources for the most demanding receivers and reuse them to support the less demanding ones. Even though it is still unclear how aggregation of reservations should be performed, this approach has the potential to support both heterogeneous requests and resource conservation, possibly without over-committing resources, thus maximizing the possibility for a new session to be admitted. Since this mechanism converges in one rather than in multiple passes, the reservation state in the switches can be periodically refreshed, turning the fixed hard state of a static connection into adaptive soft state suitable for a dynamic environment. In this way this mechanism can accommodate both group membership changes and routing modifications without involving the sender.

The interaction of routing and resource reservations further complicates matters. Even in the simple case of static routing, success in building a multicast tree depends on the adequacy of resources on each switch. We would like to construct the tree using the switches that pass the admissibility tests, thus favoring the sender-initiated reservation approach. On the other hand, we do not want the construction to fail due to over-allocation, so receiver-initiated reservations are preferable because they may avoid over-committing resources and converge in one pass. Now however, the tree constructed by the routing algorithm may be inadequate to support the reservations, again rejecting a session that could in principle be set up.

MULTICASTING ON THE INTERNET

The Internet has been extensively used as a testbed for algorithms and protocols supporting multicast. The extensions of the IP model to support multicast are the provision of special (class D) multicast addresses and IGMP (the Internet Group Management Protocol), which supports the host-group model. Multicast-aware routers periodically multicast, on a well-known address, membership queries on their LANs and gather replies from interested hosts in order to discover which groups have members present in their area.

To achieve multicasting in a wide area network, we need a mechanism to keep track of the dynamic membership of each group and another mechanism to route the multicast datagrams from a sender to these group members without unnecessary duplication of traffic. IP multicasting implements these mechanisms in two parts: local mechanisms track group membership and deliver multicasts to the correct hosts within a local network, and global mechanisms

route datagrams between local networks. Distinguishing local from global mechanisms is appropriate for IP since it is an internetworking protocol: each local network can use mechanisms appropriate to its technology, while cooperation among networks is achieved by hiding local differences behind a common interface.

In each local network, at least one router acts as a multicast router. A multicast router keeps track of local group membership and is responsible for forwarding multicasts originating from its network towards other networks, and for delivering multicasts originating elsewhere to the local network. Multicast delivery of either externally or locally originated datagrams to local receivers, as well as reception of local multicasts by the router for subsequent propagation to other networks, depend on the underlying network technology. Accordingly, the information needed within the local network regarding group membership in order to achieve local multicast delivery may vary. In contrast, cooperation among multicast routers with the purpose of delivering multicast datagrams between networks is based on a network independent interface between each local network and the outside world. The information needed in order to decide if multicasts should be delivered to target networks is whether at least one group member for a destination group is present there. A multicast router uses the information for each of its attached local networks along with information exchanged with its neighboring routers to support wide area multicasting. Irrespective of the group membership information tracked by a multicast router for local purposes, the interface between local information and global routing is a list of groups present at each attached network. Based on this interface, alternative algorithms can be used for routing among networks, without affecting local mechanisms. Conversely, as long as this interface is provided by the local mechanisms, they can be modified without affecting routing.

A variety of global, wide area multicast routing, mechanisms exist, with the earliest and most widespread being the Distance Vector Multicast Routing Protocol (*DVMRP*). DVMRP v.1 is a variant of *Truncated Reverse Path Broadcasting*. Routers construct distribution trees for each source sending to a group, so that datagrams from the source (root) are duplicated only when tree branches diverge towards destination networks (leaves). Each router identifies the first link on the shortest path from itself to the source, i.e., on the shortest reverse path, using a distance vector algorithm. Datagrams arriving from this link are forwarded towards downstream multicast routers, i.e. those routers that depend on the present one for multicasts from that source. A broadcast distribution tree is thus formed with datagrams reaching all routers. Since each router knows which groups are present in its local networks, redundant datagrams are not forwarded by truncating the tree. DVMRP v.3 implements the improved Reverse Path Multicasting mechanism, which prunes tree branches leading to networks that have no members, and grafts them back when members appear, thus turning the group distribution tree to a real multicasting one.

Another protocol discussed by Moy (12), the Multicast Open Shortest Path First (*MOSPF*), uses a link state algorithm: routers flood their membership lists among them, so that each one has complete topological information concerning group membership. Shortest path multicast distribution trees from a source to all destinations are computed on demand as datagrams arrive. These trees are real multicast ones (i.e., not broadcast), but the flooding algorithm introduces considerable overhead. A radically different proposal for multicast routing is the Core Based Trees (*CBT*) protocol, which employs a single tree for each group, shared among all sources. The tree is rooted on at least one arbitrarily chosen router, called the core, and extends towards all networks containing group members. It is constructed starting from leaf network routers towards the core as group members appear, thus it is a multicast tree composed of shortest reverse paths. Sending to the group is accomplished by sending towards the core; when the datagram reaches any router on the tree, it is relayed towards tree leaves. Routing is thus a two stage process which can be sub-optimal. The first stage may propagate datagrams away from their destinations until the tree is reached, thus increasing delay, and in addition, traffic tends to concentrate on the single tree rather than being spread throughout the network. Finally, the Protocol Independent Multicast (*PIM*) protocol by Deering *et al.* (7), employs either shared or per source trees, depending on application requirements. There are two main modes of operation of the PIM protocol depending on the distribution of the multicast group members throughout the network. In the *Sparse Mode* (*PIM-SM*), receivers are assumed to be sparsely distributed throughout the network and therefore any router with downstream group members must explicitly inform its upstream multicast routers of its interest in joining a multicast group. The resulting shared tree is rooted at a group-specific *Randez-vous Point* (*RP*). Routers can later join a source-specific, shortest path, distribution tree and prune themselves of the shared tree. In the *Dense Mode* (*PIM-DM*), the opposite assumption is made, i.e., it is assumed that most of the routers are interested in receiving multicast traffic. Therefore, each router forwards multicast data to all of its neighboring routers. Routers not interested in joining the multicast group, explicitly prune themselves off the constructed source-rooted multicast tree.

Networks supporting IP multicasting may be separated by multicast unaware routers. To connect such networks, tunnels are used: tunnels are virtual links between two endpoints, that are composed of a, possibly varying, sequence of physical links. Multicasts are relayed between routers by encapsulating multicast datagrams within unicast datagrams at the sending end of the tunnel and decapsulating them at the other end. The *MBone* is a virtual network composed of multicast aware networks bridged by such tunnels. Multicast routers may choose to forward through the tunnels only datagrams that have Time-to-Live (*TTL*) values above a threshold, to limit multicast propagation.

In contrast to global mechanisms, only a single set of local mechanisms exists. These local multicasting and group management mechanisms were based on shared medium broadcast networks such as Ethernet, and this is evident on some of the design decisions made. Delivery is straightforward on these LANs, as all hosts can listen to all datagrams and select the correct ones. If a LAN supports multicasting

as a native service, class D IP addresses may be mapped to LAN multicast addresses to filter datagrams in hardware rather than in software. Multicasts with local scope do not require any intervention by the multicast router, while externally originated multicasts are delivered to the LAN by the router. The router also monitors all multicasts so that it can forward to the outside world those for which receivers exist elsewhere. Both unicasts and multicasts are physically broadcast on these LANs, so the only issue for the router when delivering externally originated multicasts is whether at least one member for the destination group exists in the network. The router only has to keep internally a local group membership list, which coincides exactly with the information on which global multicast routing is based.

Both versions of the Internet Group Management Protocol (*IGMP*) provide a mechanism for group management well suited to broadcast LANs, since only group presence or absence is tracked for each group. In IGMP v.1 the multicast router periodically sends a query message to a multicast address to which all local receivers listen to. Each host, on reception of the query, schedules a reply, to be sent after a random delay, for each group in which it participates. Replies are sent to the address for the group being reported, so that the first reply will be heard by all group members and suppress their own transmissions. The router monitors all multicast addresses, so that it can update its membership list after receiving each reply. If no reply is received for a previously present group for a number of queries, the group is assumed absent. In steady state, in each query interval the router sends one query and receives one reply for each present group. When a host joins a group it sends a number of unsolicited reports to reduce join latency for the case where it is the first local member of the group. No explicit action is required when a host leaves a group, as group presence times out when appropriate.

In IGMP v.2 a host must send a leave message when abandoning a group, but only if it was the last host to send a report for that group. However, since this last report may have suppressed other reports, the router must explicitly probe for group members by sending a group specific query to trigger membership reports for the group in question. It can only assume the group absent if no reports arrive after a number of queries. All IGMP v.2 queries include a time interval within which replies must be sent: general queries may use a long interval to avoid concentrating reports for all groups, while group specific queries may use a short interval to speed up group status detection. The time between the last host leaving a group and the router stopping multicast delivery for that group is called the leave latency.

Other Internet Protocols and Services

The Resource ReSerVation Protocol (*RSVP*), designed by Zhang *et al.* (21), acts as an overlay on routing protocols, supporting receiver-initiated resource reservations over any available multicast routing scheme. In addition, RSVP supports dynamic reservation modifications and network reconfigurations.

A transport protocol supporting continuous media has been developed by Schulzrinne *et al.* (18): RTP (Real Time Protocol). It provides support for timing information, packet sequence-numbers and option specification, without imposing any additional error control or sequencing mechanisms. An application can use this basic framework adapted to its requirements to add whatever mechanisms seem appropriate, such as error control based on loss detection using sequence numbers, or intra-media and inter-media synchronization based on timing information. A companion control protocol, RTCP (Real Time Control Protocol), can be used for gathering feedback from the receivers, again according to the application's needs. For example, an application can use RTP for transport and RTCP adapted for scalable feedback control, along with appropriate FEC and adaptation mechanisms.

Another relevant protocol is SDP (Session Description Protocol), which provides a mechanism for applications to learn what streams are carried in the network, describing them in adequate detail so that anyone interested can launch the appropriate receiver applications.

MULTICAST IN ATM AND OTHER TECHNOLOGIES

ATM technology supports point-to-multipoint VCs, with the source responsible for setting up the VC. Two basic models have been proposed to support multicast in ATM. The first is based on a mesh of point-to-multipoint VCs from each source to the destinations. The second resembles the center based trees and uses a multicast server (*MCS*) with point-to-multipoint VCs to all destinations; sources then establish and use point-to-point VCs to forward their data to the MCS, which then forwards them to the destinations. Both models have advantages and disadvantages. Group management and dynamic join-leave is probably more complex and slow with the mesh, but throughput and delay should be better. Also the MCS is a single point of failure and concentrates traffic, not only to the particular server, but also to the subnetwork surrounding it. LAN emulation service on top of ATM offers a very similar solution with that of the MCS for multicast and broadcast.

A third, multipoint-to-multipoint solution has also been suggested based on the shared tree approach and an access control protocol that allows sources to alternate in using the common infrastructure. However, this last proposal is less compatible with the various ATM protocols and techniques currently adopted. Finally, an important problem for ATM is the mapping of high-level multicast addresses (or group names) to specific destination end-points and point-to-to-multipoint VCs. A key solution is based on a Multicast Address Resolution Server (*MARS*) based on the notion of an ATM ARP server. This service is obviously necessary for full implementation of IP over ATM.

Similarly, when multiple hosts and applications are communicating, as in a multi-party conference, there is usually a need to mediate transmission and reception of data among participants at the application layer. As the specific needs of each application and conference setting may vary, one way to support multiple control policies is to use either a specialized server or a logical conference control channel as a shared mechanism through which control messages are exchanged. Floor control and session management applications can then employ this mechanism for

their needs.

Many other applications and networking technologies have also had to confront the multicast problem. For example multi-hop light-wave technologies using multiple wavelengths assigned to source-destination pairs in the case of unicast, have to modify their architectures for multicast. For Mobile IP and IP Multicast, even though straightforward solutions do exist, they suffer from various problems that are still being investigated. Finally, in some cases multicast is proposed as a solution to other problems. For example, in order to minimize delay and loss during hand-offs in mobile packet communications, it has been proposed to multicast the packets to all base stations near the mobile so that the information will be immediately available in case of hand-off.

Multicast Security Issues

The traditional security issues for communications, which are typically thought of in the context of unicast (i.e., point-to-point communications), also exist for multicast. They typically relate to data confidentiality and integrity and service availability. However, multicast amplifies the existing problems and poses new ones. In addition, straightforward extensions of unicast either do not apply or are uninteresting. For example, using $O(n^2)$ independent end-to-end unicast secure channels between all pairs of participants can provide secure group communication. However, no benefits from multicasting can be drawn in this case. In particular, no efficient transport, nor flexible membership.

With the original Internet protocols session membership is typically not known (except perhaps in an indirect way at the application level) and cannot be controlled. This makes the problems of eavesdropping, unauthorized injection of messages, and even service denial to authorized members, even more important. However, research into these issues is just beginning and experience with real systems is almost non-existent. Some obvious but central requirements for approaches to secure multicast, in particular in the context of global networks such as the Internet and ATM, are: (1) compatibility with existing network protocols, (2) scalability, and (3) transparency to higher level services and applications.

APPLICATION-LEVEL MULTICAST

The scalability problems faced in the deployment of IP multicast in wide area networks have led to an interest in application-level multicast over peer-to-peer overlay networks. As suggested by Ratnasamy *et al* (26), the main target is to eliminate the need for a multicast routing algorithm to construct distribution trees. Towards this direction, peer-to-peer overlay networks provide a scalable, fault-tolerant, self-organizing routing substrate. Application-level multicast aims at leveraging these underlying overlay routing properties.

Peer-to-peer overlay routing is performed over an abstract namespace. A randomly chosen portion of the namespace is assigned to each participating node. Each node of the overlay network holds routing information only for a small subset of nodes whose namespace portions are neighboring its own in the global namespace. In this way, routing information is distributed among all nodes, yielding a scalable routing infrastructure. For the logical namespace proximity to reflect the actual networking proximity (in terms, for example, of round trip time), among all neighboring nodes in the logical namespace, a node holds routing information only for those closest to it in the actual networking topology. Rowstron *et al.* (23) and Zhao *et al.* (24) show that overlay routes are approximately 30% longer than the routes followed in the case of direct IP routing based on complete routing tables. This is considered as an acceptable cost in view of the fact that each node holds routing information only for a small subset of the overall topology. Messages are destined to points in the namespace (otherwise termed as *keys*). The overlay network routes a message to the node that has been assigned the portion of the namespace containing the destination point, i.e. the owner of the *key*. This is accomplished by each node forwarding the message to the node with the closest namespace to the key. The average number of overlay hops required for a message to reach its destination is a logarithmic function of the number of nodes constituting the overlay network.

Two main approaches have been followed towards application-level multicast. The first aims at building the multicast distribution tree on top of the overlay network. Zhuang *et al.* (25) propose the creation of source-specific trees on top of a Tapestry (24) overlay network. The construction of the multicast tree follows the hierarchical character of the underlying rooting mechanism, i.e. closely neighboring nodes in the namespace belong to the same tree level. However, each *join* message reaches the source node, so that the construction of the multicast tree is coordinated there, weakening the scalability of the proposed scheme. Castro *et al.* (27) overcome this limitation by handling group joins locally. They propose the creation of source-rooted trees or trees rooted at a randomly chosen *randez-vous* node, on top of a Pastry (23) overlay network. In this case, a *join* message is not propagated towards the root of the tree, but it is suppressed by an intermediate node that has already joined the group. Both approaches result in the creation of well-balanced source-specific trees due to the randomization of the overlay addresses. However, these trees may contain nodes not belonging to that multicast group.

The second approach, followed by Ratnasamy *et al.* (26), aims at the creation of an overlay network for each multicast group, avoiding the construction of a multicast tree. Multicast data is then broadcasted in the overlay network. Contrary to the first approach, there is no restriction on the number of sources, i.e. multiple nodes may broadcast in the overlay network resulting in a multipoint-to-multipoint communication model. Moreover, the dissemination of data is performed only by nodes actually belonging to the multicast group. Nevertheless, Castro *et al.* (28) show that, the tree-building approach achieves lower delay and signaling overhead than broadcasting over per group overlays due to the significant delay overhead incurred by the routing state establishment during the construction of the overlay network in the latter case.

Streaming Applications

Streaming applications, such as live audio and video delivery, pose additional requirements for the efficient point-to-multipoint data distribution. The bandwidth requirements of these applications are significant, imposing the need for load balancing measures during multicast tree construction. The end-to-end delay from the source to the receivers may be high if the content traverses long paths of nodes until it reaches the leaf nodes of the multicast tree. Hence, the need for a small tree height is apparent. Furthermore, traffic bottlenecks may appear in the tree topology if non-leaf nodes are required to forward the multicast content to a large number of descendant nodes. In effect, the fan-out degree of each node in the tree must be bounded. Overall, bandwidth availability is an essential criterion for the construction of efficient multicast trees and the adequate placement of each node in the hierarchical topology.

A significant approach addressing these issues, followed by Castro *et al.* (30) and Padmanabhan *et al.* (31), is based on the creation of multiple multicast trees per group. The multicast content is encoded in several separate, decodable streams (*stripes)* of lower quality and each stream is transmitted over a separate tree. All trees share the same root (source) and leaf nodes, but consist of disjoint sets of intermediate-level nodes. The main goal is to distribute the forwarding load among the participating nodes and this is achieved by each interior node bearing the burden of forwarding a single, lighter stream. Each node participates in all trees, either as an interior or a leaf node, in order to receive all *stripes* and reconstruct the original content. It is noted that the reception of all stripes is necessary only for the reconstruction of the content in its initial quality. The reception of fewer stripes typically still results in the reconstruction of the content, but at a lower quality. In addition to the balancing of the forwarding load, this approach may also achieve robustness to node failures. A node failure will only result in the loss of a single stripe by the leaf nodes served by the failing node. In effect the leaf nodes will only experience the degradation of the quality of the received content rather than the interruption of the streaming service.

In addition to the aforementioned approach, Duc *et al.* (29) propose the construction of the multicast tree based on a multilayer hierarchy of bounded-size clusters of nodes. In each cluster, a *head* node is responsible for monitoring membership in the cluster and an *associate head* node is responsible for transmitting the content to cluster members. Thus, membership management is distributed, relieving the source node from the burden of overall tree management. The height of the resulting tree is at most logarithmic to the size of the node population and the fan-out degree of each node is bounded by a constant.

BIBLIOGRAPHY

1. M. Ahamad, *Multicast Communication in Distributed Systems*. IEEE Computer Society Press, Los Alamitos, CA, 1990.

2. M. H. Ammar, G. C. Polyzos, and S. Tripathi, (eds.), Special Issue on " Network Support for Multipoint Communication," *IEEE Journal on Selected Areas in Communications*, **15**: 273–588, 1997.

3. K. P. Birman, A. Schiper, and P. Stephenson, *Lightweight Causal and Atomic Group Multicast*, National Aeronautics and Space Administration, Washington, D.C., 1991.

4. J. C. Bolot, T. Turletti, and I. Wakeman, Scalable feedback control for multicast video distribution in the Internet. *Computer Communications Review*, **24**(4): 58–67, 1994.

5. S. Y. Cheung and M. H. Ammar, Using destination set grouping to improve the performance of window-controlled multipoint connections. *Computer Communications*, **19**: 723–736, 1996.

6. S. E. Deering and D. R. Cheriton, Multicast routing in datagram internetworks and extended LANs. *ACM Transactions on Computer Systems*, **8**(2): 85–110, 1990.

7. S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. Liu, and L. Wei, The PIM architecture for wide-area multicast routing. *IEEE/ACM Transactions on Networking*, **4**: 153–162, 1996.

8. C. Diot, W. Dabbous, and J. Crowcroft, Multipoint communication: a survey of protocols, functions, and mechanisms. *IEEE Journal on Selected Areas in Communications*, **15**: 277–290, 1997.

9. M. Doar and I. Leslie, How bad is naive multicast routing? Proc. IEEE INFOCOM'93: 82–89, 1993.

10. M. R. Garey, R. L. Graham, and D. S. Johnson, The complexity of computing Steiner minimal trees. *SIAM Journal on Applied Mathematics*, **34**: 477–95, 1978.

11. S. L. Hakimi, Steiner's problem in graphs and its implications. *Networks*, **1**: 113–133, 1971.

12. J. Moy, Multicast routing extensions for OSPF. *Communications of the ACM*, **37**(8): 61–66, 1994.

13. B. K. Kabada and J. M. Jaffe, Routing to multiple destinations in computer networks. *IEEE transactions on communications*, **31**: 343–351, 1983.

14. V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, Multicast routing for multimedia communication. *IEEE/ACM Transactions on Networking*, **1** (3): 286–292, 1993.

15. L. Kou, G. Markowsky, and L. Berman, A fast algorithm for Steiner trees. *Acta Informatica*, **15**: 141–145, 1981.

16. J. C. Pasquale, G. C. Polyzos, E. W. Anderson, and V. P. Kompella, The multimedia multicast channel. *Internetworking: Research and Experience*, **5**(4): 151–162, 1994.

17. J. C. Pasquale, G. C. Polyzos, and G. Xylomenos, The multimedia multicast problem. *Multimedia Systems*, **6**(1): 43–59, 1998.

18. H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, *RTP: A Transport Protocol for Real-Time Applications*. Internet Request For Comments, RFC 1889, 1996.

19. D. Towsley, J. Kurose, and S. Pingali, A comparison of sender-initiated and receiver-initiated reliable multicast protocols. *IEEE Journal on Selected Areas in Communications*, **15**: 398–406, 1997.

20. G. Xylomenos and G. C. Polyzos, IP Multicast for Mobile Hosts, *IEEE Communications Magazine*, **35**(1): 54–58, 1997.

21. L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, RSVP: a new resource ReSerVation Protocol. *IEEE Network*, **7**(5): 8–18, 1993.

22. W. D. Zong, Y. Onozato, and J. Kaniyil, A copy network with shared buffers for large-scale multicast ATM switching. *IEEE/ACM Transactions on Networking*, **1**(2): 157–165, 1993.

23. A. Rowstron and P. Druschel, Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems.

IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany, pp. 329–350, November 2001.

24. B. Y. Zhao, Ling Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz. Tapestry: A resilient global-scale overlay for service deployment. *IEEE Journal on Selected Areas in Communications*, **22**(1), 2004.
25. S. Zhuang, B. Zhao, A. Joseph, R. Katz, and J. Kubiatowicz, Bayeux: An Architecture for Scalable and Fault-tolerant Wide-Area Data Dissemination, Proc. NOSSDAV, pp. 11–20, 2001.
26. S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, Application-level multicast using content-addressable networks, Proc. International Workshop on Networked Group Communication (NGC), November 2001.
27. M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron, Scribe: A large-scale and decentralized application-level multicast infrastructure, *IEEE JSAC*, **20**(8), 2002.
28. M. Castro, M. B. Jones, A.-M Kermarrec, A. Rowstron, M. Theimer, H. Wang, and A. Wolman, An evaluation of scalable application-level multicast built using peer-to-peer overlays, *Proc. IEEE INFOCOM*, **2**: 1510–1520, 2003.
29. D. A. Tran, K. Hua, T. Do, A peer-to-peer architecture for media streaming, *IEEE Journal on Selected Areas in Communications*, **22**(1): 121–133, 2003.
30. M. Castro, P. Drushel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, SplitStream: High-bandwidth Multicast in Cooperative Environment, *ACM Symposium on Operating Systems Principles*, 2003.
31. V. N. Padmanabhan, J. J. Wang, P. A. Chou, and K. Sripanidkulchai, Distributing streaming media content using cooperative networking, Proc. NOSSDAV, pp. 177–186, 2002

GEORGE C. POLYZOS
K. KATSAROS
Athens University of Economics and Business, Athens, Greece