Improving Layered Video Multicast using Active Networks

Paulo A. da S. Gonçalves[1,2], José F. de Rezende[2], Otto C. M. B. Duarte[2] and Guy Pujolle[1]

[1]Network and Performance Group - LIP6
Pierre and Marie Curie University - France

[2]Networking and Automation Group – GTA
COPPE/EE - Federal University of Rio de Janeiro - Brazil

Abstract— *Traditional services for video distribution over the Internet combine layered encoding with multicast transmission by sending different layers on different multicast addresses. This approach trades loss in scalability for its ability to offer many quality levels. Based on the active network paradigm, we propose a new service that achieves a layered video distribution using a single multicast tree. Furthermore, the proposed active service takes into account the available network bandwidth when dynamically adapting to better meet receiver needs. Interactivity among source and receivers is achieved without incurring in feedback-message implosion. This result is a consequence of the fusion mechanism employed. Simulation results show that a high capsule filtering level, around 95%, can be achieved using a transit-stub network topology.*

I. Introduction

The emergence of active and programmable networks [1, 2, 3, 4, 5] has introduced a novel approach to deploy and enhance network services. The programmable nature of an active network allows applications to manipulate network resources (e.g., processing, soft-state and bandwidth) in order to customize network services on an as-needed basis. Active packets or capsules can carry programs to be executed at each node along their path, providing the basis of the network control.

The RLM (Receiver-driven Layered Multicast) [6] protocol pioneered the study of the layered video transmission. Many proposals [7, 8, 9, 10, 11] explore or attempt to improve the properties of RLM, but neither of them exploits the Internet interactivity nor capabilities provided by active networks.

Moreover, these proposals share the following weaknesses. First, each video layer is sent out on a different multicast address. To offer more levels of quality, the source must generate more layers and therefore more multicast trees must be established. Furthermore, increasing the number of multicast trees means to keep more states at routers. As a consequence, a large number of quality levels incur in scaling problems. The second weakness is related to experiments carried out by receivers attempting to improve their perceived video quality. Under these experiments, each receiver subscribes to the multicast group related to the next layer that it wants to receive, and then it measures packet loss. If the loss rate exceeds a threshold, the receiver concludes that there is no available bandwidth and unsubscribes from the group. These experiments may incur in additional network congestion, and as a consequence, receivers will interfere with each other when attempting to improve their own perceived video quality.

In this paper, we propose an active service that overcomes the weaknesses described above and is aware of the increasing necessity for adaptive applications [12, 13] and QoS (Quality of Service) based routing schemes [14, 15] within the network. Based on the active network paradigm, the service takes advantage of customized computations and state within the network to yield enhancements to layered video distribution. The result is an interactive service that can dynamically adapt to individual receiver needs, depending only on the bandwidth availability and the processing capacity of each receiver. In this paper, we also present a performance evaluation of the fusion mechanism used to prevent feedback-capsule implosion.

The remaining of this paper is organized as follows. The proposed active service is described in Section II. Section III presents the implemented protocol. Simulation and analytical results are presented in Section IV. Finally, Section V concludes the paper.

II. The Active Service

The proposed active service is based on a receiver-driven approach in which receivers specify their individual video quality requirements. Moreover, the active service allows receivers to change their individual video quality requirements and connect/disconnect from the service as many times as they want and anytime during a session.

Our service requires a layered video source with QoS dynamic control, i.e. the source generates an on-demand and controlled number of layers. The source learns about the best video transmission rate by periodically receiving feedback capsules from receivers. These capsules carry the video quality requirements of the receivers and perform customized computations within the network to allow for the network bandwidth availability. Thus, the source can generate a number of layers suitable for a range of path characteristics and pro-

This paper was supported by: CAPES, COFECUB, UPMC, UFRJ, CNPq and CNRS.

vide a service that better matches receiver needs.

Video data capsules carry the video information of each layer. The routing of these capsules is aided by soft-state forwarding pointers installed in the active nodes. A video data capsule routes itself along the paths that lead to interested receivers only if the bandwidth availability permits.

Fig. 1 illustrates the active service when three layers are encoded, at 1 Mbps each, by the video source. Link capacities and the video quality desired by each receiver are also presented. All receivers, except the fourth, receive a number of layers suitable to the available network bandwidth. Although the fourth receiver could receive the second enhancement layer, it is only receiving its desired video quality of 2 Mbps, that is, the base layer and the first enhancement layer. Each receiver can attempt to increase its received video quality by using the quality adjustment scale at its own convenience. The upper bound on the scale is related to the processing capacity of the receiver in decoding video. Suppose that the third receiver disconnects from the service. Then the source will only generate two layers to avoid waste of network bandwidth.

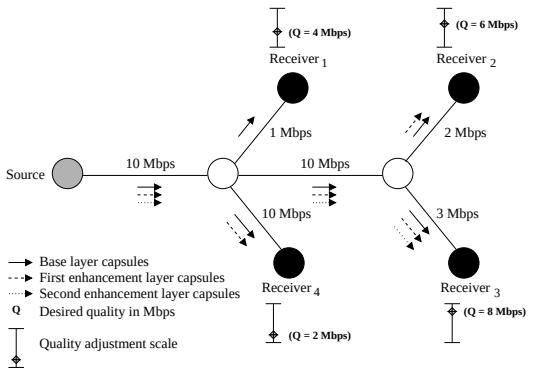


Fig. 1. Example of the layered video distribution.

III. The Active Protocol

The proposed active protocol is composed of five capsule types: subscribe/maintenance, availability, video data, end-of-session, and unsubscribe. Receivers use the unsubscribe capsule to disconnect themselves from the service. The source indicates to the receivers that it has no more data to transmit by sending an end-of-session capsule. The function of the other capsules will be presented throughout this section.

A. Subscribe/Maintenance Capsules

Subscribe/Maintenance capsules employ the basic multicast protocol proposed in the ANTS [16, 17] system to construct and maintain a multicast distribution tree. Thus, subscribe/maintenance capsules are periodically sent by receivers interested in the video reception. As the capsule follows towards the source, it installs or refreshes soft-state forwarding pointers that allow video data capsules to know the paths that lead to receivers. Fig. 2 illustrates this idea. Computations into the network suppress unnecessary capsules to avoid subscribe/maintenance capsule implosion.

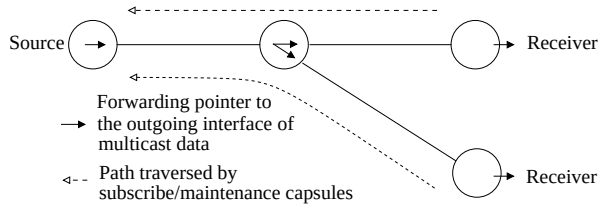
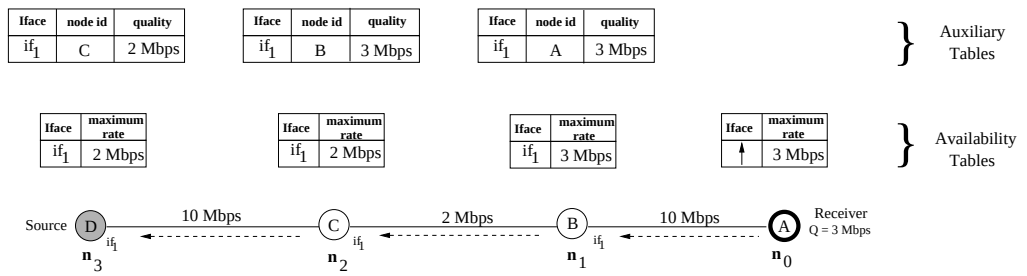


Fig. 2. Multicast tree formation.

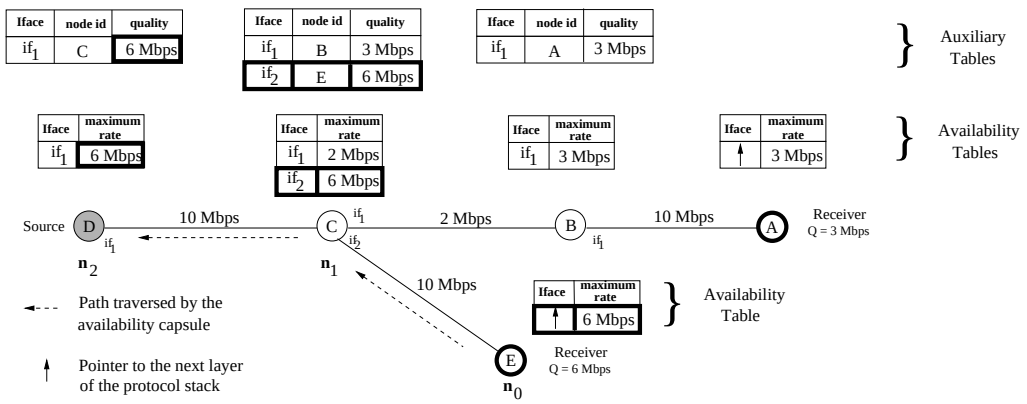
B. Availability Capsules

The availability capsules construct an availability table at each node they pass through to aid the routing of video data capsules. The node's availability table contains a list of the maximum video transmission rate through each outgoing interface belonging to the multicast tree. The availability capsules also construct at each node, except at the receivers, an auxiliary table related to the availability table. From the standpoint of a node, these tables contain both the desired video quality by downstream neighbor nodes belonging to the multicast tree and a reference to the interface that allows to reach each of these neighbors.

Fig. 3(a) shows how these tables are constructed due to the connection of a first receiver (A) to the service. Nodes have been numbered with respect to the standpoint of this receiver. Each node has an identifier indicated inside it. Each link capacity is indicated on the respective link. Immediately after sending the first subscribe/maintenance capsule, the receiver sends an availability capsule that carries its desired video quality Q. This information is carried in a capsule header field (TX) and it is expressed in terms of bandwidth. The receiver's availability table only contains an entry that informs the parameter Q. Upon reaching n_1, the capsule constructs an auxiliary table. This table contains an entry that is a compound of the identifier of the previous node visited by the capsule, the TX value carried by the capsule and the interface that allows to reach the previous node. Next, the capsule verifies the available bandwidth for the video on the link to the previous node. The capsule assigns in the node's availability table the minimum value between the available bandwidth for the video and the indicated quality in the auxiliary table. The capsule updates its TX field with this minimum value and follows to the next node. The



(a) Connection of the first receiver.



(b) Connection of a second receiver.

Fig. 3. Construction and update of auxiliary and availability tables.

same algorithm is performed when the capsule passes through n_2 and n_3. Upon receiving the capsule, the source knows exactly how many layers to generate in order to better serve the receiver.

Fig. 3(b) illustrates how a second receiver (E) connects to the service. Node numbering is related to the standpoint of this receiver. After constructing the availability table in E, the availability capsule sent by the receiver reaches n_1 carrying in the TX field the video quality desired by E. A new entry into the auxiliary table that contains the tuple $\{if_2, E, 6\ Mbps\}$ is added. That is, the quality of 6 Mbps desired by the receiver E, which is reachable through interface 2. The new value of the maximum video transmission rate on this interface is computed as the minimum value between the available bandwidth for video on the link to receiver E and the quality in the auxiliary table related to interface 2. The field TX is updated with this computed value and the capsule follows to n_2. At this node, the entry in the auxiliary table is updated with the TX value. Next, it is computed the minimum value between the available bandwidth for the video and the value in the auxiliary table field labeled *quality*. The entry in the availability table as well as the TX field are updated with this information, and next, the capsule follows to the next layer of the protocol stack. Upon receiving the capsule, the video source can adapt to better serve the new receiver.

From the standpoint of the node B, its auxiliary table contains the desired quality by the receiver A. From the standpoint of the node C, its auxiliary table contains the quality desired by the nodes B and E. Thus, more precisely, the auxiliary table contains the desired quality by each downstream neighbor node that belongs to the multicast tree.

More generally, the algorithm performed to add or update an entry in the auxiliary table follows (1).

$$\Gamma^{(i)}_{n,n-1} = TX_{n-1}, \quad \text{for } n \geq 1 \tag{1}$$

where i is the incoming interface of the capsule, $\Gamma^{(i)}_{n,n-1}$ represents, from the standpoint of the node n, the quality desired by the node $n-1$, which is reachable through interface i. The value carried in the TX field of a capsule upon leaving node $n-1$ is represented by TX_{n-1}.

The algorithm performed to add or update an entry in the availability table follows (2).

$$tx_n^{(i)} = \begin{cases} \min\{\max(\Lambda_n^{1,(i)}, \ldots, \Lambda_n^{\sigma^{(i)},(i)}), b_n^{(i)}\}, \\ \qquad \text{if } n \geq 1; \text{ for } \sigma^{(i)} \geq 1 \\ Q, \qquad \text{if } n = 0 \end{cases} \quad (2)$$

where i is the incoming interface of the capsule, $b_n^{(i)}$ represents the available bandwidth for video on the link that is reachable through interface i of the node n. If $n \geq 1$, $tx_n^{(i)}$ represents the maximum video transmission rate on the link reached through interface i. Otherwise, it represents the quality Q desired by the receiver. The number of auxiliary table entries related to interface i is represented by $\sigma^{(i)}$. Values from $\Lambda_n^{1,(i)}$ to $\Lambda_n^{\sigma^{(i)},(i)}$ represent entries related to interface i in the auxiliary table field labeled *quality* in the node n.

Equation (3) gives the value of the field TX in the capsule that will follow towards the source.

$$TX_n = \begin{cases} \max(\Omega_n^1, \ldots, \Omega_n^\theta), & \text{if } n \geq 1; \text{ for } \theta \geq 1 \\ Q, & \text{if } n = 0 \end{cases} \quad (3)$$

where Q is the video quality, in terms of bandwidth, desired by the receiver. TX_n represents the TX field value in the availability capsule header upon the capsule leaving node n. The number of entries in the availability table is represented by θ. Values from Ω_n^1 to Ω_n^θ represent entries in the field labeled *maximum rate* in the availability table of the node n.

B.1 Feedback and Fusion Mechanisms

A feedback mechanism is used in order to keep the video source updated about the better video transmission rate. Thus, all receivers must periodically send availability capsules towards the source. Note that interactivity among source and receivers is achieved since the source can adapt according to the information carried by the capsules. This information represents, directly or indirectly, the desired video quality by each receiver or the video quality supported by the network. Quality adjustments carried out by receivers are automatically placed into the new availability capsules that follow towards the source.

The drawback of periodically sending availability capsules towards the source is twofold. First, updates in the availability and auxiliary tables will be uncoordinated, and second, the source will be prone to feedback-capsule implosion. We use a fusion mechanism to solve these problems. The fusion of availability capsules occurs according to the following conditions: (i) at least one availability capsule related to each entry in the auxiliary table has arrived; (ii) the fusion timer has expired.

Suppose that a capsule related to each entry in the auxiliary table arrives at a node before the expiration of the fusion timer. In this case, on the last expected capsule arrival, entries in the auxiliary table related to each cached capsule are updated according to (1). Next, base on (2), the availability table is completely updated and only one availability capsule follows to the next node, carrying in TX the result given by (3). To update an availability table entry, at least one capsule from the interface related to this entry must be in the node's cache. Suppose the expiration of the fusion timer before the arrival of all expected availability capsules, then it is updated for each cached capsule its respective entry in the auxiliary table following (1). Upon finishing these updates, based on (2), the entries in the availability table are updated. Next, only one capsule follows to the next node carrying in TX the result of (3). If more than one capsule related to the same auxiliary table entry arrives before the expiration of the fusion timer or the arrival of all expected availability capsules, then the previous cached capsule is dropped. Thus, only the most updated information will be used during the fusion.

C. Video Data Capsules

Video data capsules route themselves along the paths that lead to interested receivers only if the bandwidth availability permits. These capsules use the information in their TX field and the node's availability table to make the routing decisions. The source assigns the field TX of each video data capsule with the cumulative rate for the layer that the capsule belongs to, that is, the field TX is assigned according to the following rule:

$$TX = \sum_{i=1}^{c} TC_i \text{ for } 1 \leq c \leq N \quad (4)$$

where c is the layer that the capsule belongs to, TC_i denotes the rate of the i^{th} layer and N is the maximum number of layer that the source can generate.

Upon arriving at a node, the video data capsule checks the soft-state for its multicast group. If there is no soft-state, then the capsule drops itself, otherwise, it verifies in the availability table if the maximum rate on each interface is greater than or equal to the value placed in its TX field. If true, the capsule follows through the interface. If more than one interface respect this condition, the video data capsule must reproduce or deploy copies of itself on these interfaces. If there are no interfaces that respect this

condition, then the capsule drops itself. Note that the mechanism for constructing and maintaining the multicast tree combined with the filtering performed by video data capsules themselves has the positive effect of eliminating the need for establishing multiple trees when multicasting layered video. There is only a single multicast tree that is traversed by base layer capsules. Capsules of each enhancement layer can traverse all or some branches of this tree. Furthermore, as a consequence of the mechanism for constructing and updating the availability tables combined with the filtering performed by the video data capsules themselves, receivers get a number of layers according to the characteristic of their individual path from the source and their desired video quality Q.

IV. Results

In this section, we evaluate the performance of the fusion mechanism. We first derive an analytical result for the maximum filtering level achieved by the fusion mechanism when the network topology is represented by a tree. The filtering level is defined as the percentage of the total number of availability capsules suppressed into the network. Subsequently, we evaluate the performance of the fusion mechanism via simulation using a network topology model that correlates well with the domain structure found in the Internet.

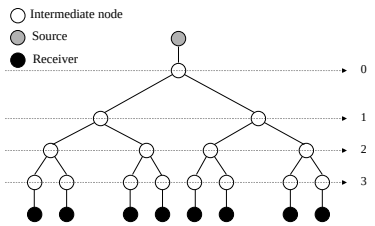


Fig. 4. Example of a complete m-ary tree topology with height $h = 3$ and $m = 2$.

The maximum filtering level for topologies such as the one represented in Fig. 4 can be easily derived by considering both the network has identical links and the receivers are synchronized, that is, the connection-time of them as well as the sending time of availability capsules are the same. Thus, fusions will be synchronized at all levels of the tree. For such topologies, the maximum filtering level is given by the following equation:

$$F_{max} = \left(1 - \frac{1}{m^h}\right) 100\% \quad \text{for } m \geq 1 \text{ and } h \geq 0 \qquad (5)$$

Note that the maximum filtering level for a complete 3-ary tree topology with $h = 3$ is 96.3%.

The proposed active service was implemented in the ns-2 simulator [18, 19]. A modified version with the basic support to active networks was used. New agents, new capsule types, the fusion mechanism, and mechanisms for monitoring and estimating the available bandwidth on links were implemented [20].

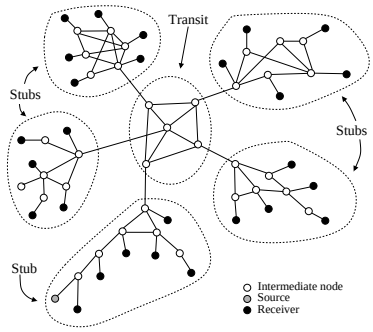


Fig. 5. Network topology used in simulations.

The transit-stub topology represented in Fig. 5 was used to evaluate the performance of the fusion mechanism. Link capacities are 10 Mbps. Each link delay was randomly chosen uniformly on the interval [1, 10] ms. The timeout interval of the fusion timer was set to 100 ms. The connection-time of each receiver was randomly chosen uniformly on the interval [0, 10] s. Each simulation was run for 100 seconds. All simulation results have a confidence interval of 90%.

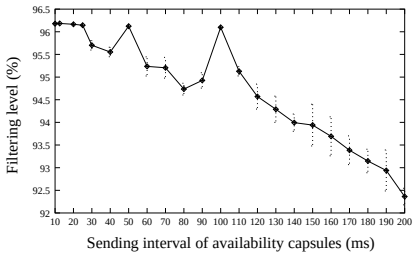


Fig. 6. Availability capsule filtering level.

Fig. 6 shows how the filtering level of availability capsules varies with the sending interval of these capsules. This curve shows an availability capsule filtering level from 92% to 96.18%. The maximum filtering levels are observed when the sending interval of the availability capsules is less than 25 ms, or equal to or half of the timeout interval of the fusion timer.

Fig. 7 shows how the fusions occur into the network as the sending interval of availability capsules is varied. The result shows that when the sending interval is less than 25 ms, or equal to or half of the timeout interval of the fusion timer, the majority of the fusions occurs due to the arrival of the expected capsule related to each auxiliary table entry. It means that table updates are much more coordinated and the information received by the source better represents the bandwidth availability

of the network rather than the available bandwidth in paths from some receivers to source.

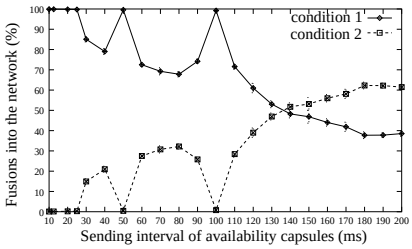


Fig. 7. Behavior of the fusions into the network.

As a final result, Fig. 8 shows the behavior of the average interval arrivals of availability capsules at the source according to the availability capsule sending interval. This result shows that when the sending interval is less than 25 ms, or equal to or half of the timeout interval of the fusion timer, the average interval arrivals of availability capsules is almost the same of the sending interval of the availability capsules specified at receivers.

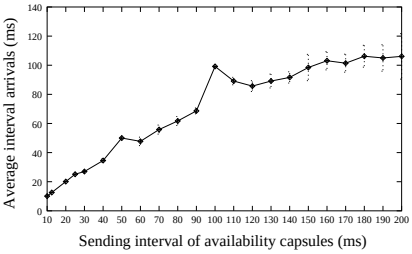


Fig. 8. Average interval arrivals of availability capsules.

V. Conclusion

We have presented an active service for multicasting layered video. Simple programming capabilities and knowledge that is available only within the network were exploited to yield a more interactive and scalable service. Our approach has the positive effect of achieving a layered video distribution using a single multicast tree. This effect is a consequence of the mechanism for constructing and maintaining the multicast tree combined with the filtering performed by video data capsules themselves. Furthermore, in contrast with the service provided by the RLM protocol and other proposals, receivers of the service act independently without interfering with each other when attempting to improve their perceived video quality. The performance of the fusion mechanism showed that a high capsule filtering level, around 95%, can be achieved using a transit-stub network topology model. Thus, this result proves that interactivity among source and receivers is achieved without degrading the scalability of the service.

References

[1] J. Smith *et al.*, "Activating networks: A progress report," *IEEE Computer*, vol. 32, pp. 32–41, Apr. 1999.

[2] K. L. Calvert, S. Bhattacharjee, and J. Sterbenz, "Directions in active networks," *IEEE Communications Magazine*, vol. 36, pp. 72–78, Oct. 1998.

[3] J. Biswas *et al.*, "The IEEE P1520 standards initiative for programmable network interfaces," *IEEE Communications Magazine*, vol. 36, pp. 64–70, Oct. 1998.

[4] D. L. Tennenhouse *et al.*, "A survey of active network research," *IEEE Communications Magazine*, pp. 80–86, Jan. 1997.

[5] D. L. Tennenhouse and D. J. Wetherall, "Towards an active network architecture," *Computer Communication Review*, vol. 26, Apr. 1996.

[6] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven layered multicast," in *SIGCOMM'96*, (Stanford, CA), pp. 117–130, Aug. 1996.

[7] R. Gopalakrishman *et al.*, "A simple loss differentiation approach to layered multicast," in *IEEE INFOCOM'2000*, (Tel Aviv, Israel), Mar. 2000.

[8] A. Legout and E. W. Biersack, "PLM: Fast convergence for cumulative layered multicast transmission schemes," in *ACM SIGMETRICS 2000*, (Santa Clara, CA), June 2000.

[9] L. Vicisano, L. Rizzo, and J. Crowcroft, "TCP-like congestion control for layered multicast data transfer," in *IEEE INFOCOM'98*, (San Francisco, CA), Apr. 1998.

[10] X. Li, S. Paul, and M. H. Ammar, "Layered video multicast with retransmission (LVMR): Evaluation of hierarchical rate control," in *IEEE INFOCOM'98*, (San Francisco, CA), pp. 1062–1072, Apr. 1998.

[11] L. Wu, R. Sharma, and B. Smith, "ThinStreams: An architecture for multicasting layered video," in *NOSSDAV'97*, (St. Louis, MO), pp. 173–182, May 1997.

[12] M. D. de Amorim, O. C. M. B. Duarte, and G. Pujolle, "Multi-criteria arguments for improving the fairness of layered multicast applications," in *Proceedings of Broadband Communications, High Performance Networking, and Performance of Communication Networks - Networking'2000*, no. 1815 in Lecture Notes in Computer Science, pp. 1–10, May 2000.

[13] C. Diot, C. Huitema, and T. Turletti, "Multimedia applications should be adaptive," in *HPCS'95*, Aug. 1995.

[14] G. Apostolopoulos *et al.*, "Quality of service based routing," in *SIGCOMM'98*, pp. 17–28, Sept. 1998.

[15] L. H. M. K. Costa, S. Fdida, and O. C. M. B. Duarte, "Distance-vector qos-based routing with three metrics," in *Proceedings of Broadband Communications, High Performance Networking, and Performance of Communication Networks - Networking'2000*, no. 1815 in Lecture Notes in Computer Science, pp. 847–858, May 2000.

[16] D. Wetherall, J. Guttag, and D. Tennenhouse, "ANTS: Network services without the red tape," *IEEE Computer*, vol. 32, pp. 42–48, Apr. 1999.

[17] D. J. Wetherall, J. V. Guttag, and D. L. Tennenhouse, "ANTS: A toolkit for building and dynamically deploying network protocols," in *IEEE OPENARCH'98*, (San Francisco, CA), Apr. 1998.

[18] K. Fall and K. Varadhan, "NS notes and documentation," tech. rep., The VINT Project, July 1999.

[19] S. Bajaj *et al.*, "Improving simulation for network research," tech. rep., University of Southern California, Department of Computer Science, Mar. 1999.

[20] P. A. da S. Gonçalves, J. F. de Rezende, and O. C. M. B. Duarte, "An active service for multicast video distribution," *Journal of the Brazilian Computer Society*, vol. 7, pp. 43–51, July 2000.