USE OF ACTIVE NETWORK TECHNOLOGIES FOR DISTRIBUTED SIMULATION

Dr. Stephen Zabele
Litton-TASC, Inc.
Reading, MA

Thomas Stanzione
Litton-TASC, Inc.
Reading, MA

ABSTRACT

While distributed simulation infrastructures have evolved dramatically over the past several years to provide ever increasing levels of flexibility, abstraction, and interoperability, the scalability and performance of the simulation infrastructure continues to be a critical limiting factor. In particular, it is now becoming apparent that the limitations of the supporting networking technologies are a significant impediment to achieving needed levels of scalability and performance. Advancing the state-of-the-art for large-scale distributed simulations therefore requires significant advances both in the underlying network technologies and in the ability of simulations to exploit these new capabilities.

Under the Specialized Active Networking technologies for Distributed Simulation (SANDS) project sponsored by the Information Technology Office (ITO) of the Defense Advanced Research Projects Agency (DARPA), TASC and the University of Massachusetts, Amherst (UMass) are developing Active Networks-based capabilities to improve significantly the performance of network-based distributed simulations[1]. Our primary objective is reducing the substantial amounts of irrelevant network traffic delivered to simulation hosts in order to both improve bandwidth efficiency and to reduce the considerable overhead associated with reading and discarding unneeded data. Our approach involves installing dynamic packet filters *within the network* that act on behalf of each host to eliminate unneeded packets as early as possible. Our goal is a seamless integration with the High Level Architecture (HLA) Declaration Management (DM) and Data Distribution Management (DDM) services.

Use of Active Networks to provide interest management services offers several important benefits to large scale simulations: (i) Because each entity can install its own filters, information filtering is accomplished in a "receiver-driven" manner, allowing each entity to customize its filters according to its own need. This decentralized approach allows active filtering to scale well as the number of entities grows large. (ii) Because active filtering is performed at a routing point, filtering can also be dependent on the state (e.g., congestion-level) at that router. In particular, this allows both entities and network routers to determine which data should be shed in times of congestion overload, and provides an effective means for mediating among the conflicting demands of different entities.

ABOUT THE AUTHORS

DR. STEPHEN ZABELE is Manager of Advanced Networking and Information Systems at Litton-TASC, Inc., in Reading, MA, where he is Principal Investigator on a number of DARPA projects in the advanced networking area. His current research activities include the design and analysis of scalable reliable and semi-reliable multicast transport protocols, the design and analysis of protocols and services for specialized applications using Active Networking technologies, and the design and analysis of techniques for secure group communications in wireless networking environments

THOMAS STANZIONE is the manager of the Simulation Technology Section at Litton TASC, Reading, MA. He manages a staff of 12 engineers investigating several simulation technology areas including computer generated forces, synthetic environments, software engineering, simulation infrastructure development, and simulation networking. Mr. Stanzione has a Master of Science degree in Photographic Science from the Rochester Institute of Technology.

[1] This work is supported by the Defense Advanced Research Projects Agency (DARPA) under contract N66001-9117-411V.

USE OF ACTIVE NETWORK TECHNOLOGIES FOR DISTRIBUTED SIMULATION

Dr. Stephen Zabele
Litton-TASC, Inc.
Reading, MA

Thomas Stanzione
Litton-TASC, Inc.
Reading, MA

ACTIVE NETWORKS

Active Networks (AN) [1] are being touted by many in the industry as the next major advance in computer networking technologies. The core idea behind Active Networks is that placing computational resources directly within the network specifically to support end user needs will not only provide higher performing traditional service offerings, but will allow faster deployment of new service offerings as well as enable a whole new class of service capabilities.

Advantages of Active Networks

From a performance perspective, the design of networking protocols and services is fundamentally one of balancing scarce resources (e.g., buffer space, computational resources and bandwidth) in order to optimize performance (e.g., throughput, latency, robustness, and resource sharing). Traditionally, network design efforts have quite reasonably assumed severe limitations on network buffer and computational resources and, as such, have pushed the burden to the end hosts whenever possible. However, since the cost of processing and memory continues to drop at a far faster rate than the cost of link capacity, alternate design points are becoming not only feasible but also increasingly attractive. For example, our work in the reliable multicast area [2] has shown that the placement of small amounts of session-specific computation at key routing nodes can significantly increase throughput, decrease latency, and reduce congestion to make more bandwidth available for other sessions.

From a deployment perspective, developing and fielding new network protocols and services has also been a significant undertaking. The current painstaking Internet Engineering Task Force (IETF) standardization process is both slow and cumbersome, and increasingly lags technology development cycles where a "net year" can be measured in terms of a few months. Moreover, when a "best guess" design proves erroneous, correcting and updating the implementation presents a major problem. The Herculean effort by DARPA to get Jacobson's enhancements into TCP before widespread release of the protocol is the rare exception: the IETF adoption process is the rule.

The development and deployment problem is compounded by the unfortunate reality that newer, more complex protocols and services necessarily require testing on larger scale networks with large numbers of users to truly prove a design. Network simulations, while invaluable, are inherently limited because of the all-too-necessary simplifications that must be made in modeling network behavior. Furthermore, laboratory experiments with limited network topologies are generally too small to really understand the full impact and dynamical range inherent in a given approach. Active Networks promises to provide a highly flexible mechanism for deploying and testing new concepts and ideas by allowing their direct insertion into the internetwork fabric itself.

Active Networks for Distributed Simulation

Active networks can be used to increase the scalability and performance of distributed simulations in numerous ways. Opportunities to leverage active networks include:

- *Enhanced Network Transport* – Our work on the use of active networks for reliable and semi-reliable multicast transport protocols has demonstrated that AN-based implementations can provide substantially reduced latency and significantly improved throughput and scalability over traditional end-host approaches. Since multicast protocols include unicast protocols as a proper subset (i.e., a unicast session is equivalent to a multicast session with only two participants), this work also shows that the performance of traditional protocols such as TCP can be significantly enhanced using active networks.
- *Interest Filtering* - An analysis of packet traces from a Synthetic Theater of War (STOW) scenario emulation [4] revealed that overall network traffic could easily be cut by over 60% using otherwise simple partitioning methods to reduce the amount of irrelevant data transmitted and received. However,

achieving this straightforward partitioning requires a network infrastructure that can support large numbers of interest groups. Unfortunately, this number is substantially larger than the number of groups that can be supported efficiently by the current suite of Run Time Infrastructure (RTI) implementations -- a shortcoming that is first and foremost a limitation imposed by current generation network infrastructures. As such, an active networks-based approach that allows the network to support large numbers of interest groups offers tremendous value.

Other areas of opportunity, which are outside the scope of this paper, include:

- *Priority-based congestion management,*
- *Data caching,*
- *Computational caching,*
- *Consistency maintenance,*
- *Clock synchronization, and*
- *Sequence management.*

The area of Interest Filtering is of particular interest due to the substantial, quantifiable performance gains that can potentially be achieved. As such, the primary focus of our work (and the focus of this paper) is on developing techniques that will allow network infrastructures to efficiently support large numbers of interest groups using active network technologies.

Advantages of Active Interest Filtering

To illustrate the potential power of using active networks to provide an *active interest filtering* capability, consider a simulation scenario of a fast flying aircraft at low altitudes with limited sensor range. With a large number of ground vehicles sparsely distributed over a correspondingly larger geographical region, the aircraft simulation is interested in relatively small numbers of ground vehicles at any given time, but relatively large numbers of ground vehicles over time. If communication grouping is based on a spatial partition of the battle space (as in [4]), the region specification process must weigh the disadvantages of a few large groupings (which will likely include too many ground vehicles) against a larger number of smaller groupings (requiring more frequent maintenance of group membership).

In contrast, consider a customized dynamic interest filter in the form of agents injected into selected network routers on behalf of the aircraft simulation. From their locations within the network, these agents can easily examine position information in state update messages from ground vehicles and forward only those within a given distance of the aircraft's current position.

Active interest filters can easily be configured to examine more than simple geographic proximity in making forwarding decisions. For example, consider a JSTARS simulation, which has a large field of view of the battlefield, but is only sensing moving vehicles. The active interest filters can filter static vehicle data before it reaches the JSTARS simulation, so that the simulation need not spend time weeding out non-moving vehicle data.

An active interest filtering approach is conceptually flexible enough to implement several forms of multi-resolution processing as well. For example, filters could be used to forward update messages for entities further away on a less frequent basis. Similarly, it could also be used to implement a priority-based congestion management mechanism. For example, state data from Surface to Air Missiles and enemy aircraft would have higher priority over data from lesser ground-based threats due to their substantially higher lethality potential. When faced with a choice of which packets to drop during heavy network congestion, the active interest filter could selectively forward the higher priority packets.

Finally, from the perspective of current generation RTIs which rely on IP/Multicast as a convenient and network-efficient means for delivering event data to interest groups, the only manner in which an entity can control the data it receives is through the joining and leaving of multicast groups. Technically, this is accomplished through the Internet Group Management Protocol (IGMP) and network multicast routing protocols (e.g., Distance Vector Multicast Routing Protocol (DVMRP), Protocol Independent Multicast (PIM)) which provide only coarse control over data delivery at relatively long time scales (e.g., 100s of milliseconds to join a group, and tens of seconds to leave a group). Here, active interest filtering offers a means to introduce fine-time-scale, flexible (programmable) per-entity information filters into the routers' forwarding function that can react to changes in receiver interests nearly instantaneously.

Interest Filtering Using Active Networks

In a previous paper [3], the authors describe the use of multicast addressing for HLA data distribution management (DDM) services [5], providing example approaches for mapping routing spaces to multicast addresses. These examples share the common problem of potentially requiring a tremendously greater number of multicast groups than can realistically be supported by the network infrastructure. One solution to this problem is to merge the groups into some smaller number of aggregate groups that can be supported, with the goal of minimizing the amount of irrelevant data delivered to the end hosts (and possibly within the network)[2].

An initial analysis of an aggregation approach shows that the aggregation operation itself is potentially quite expensive computationally. For example, the number of ways that *M* groups can be aggregated to form *K* groups (*K*<*M*) where each of the *M* groups is assigned to one and only one of the *K* aggregate groups is given by the Stirling number of the second kind, given by:

$$S_M^{(K)} = \frac{1}{K!}\sum_{j=0}^{K}(-1)^{K-j}\left(\frac{K}{j}\right)j^M$$

The last term in the expansion of the summation is given by:

$$\frac{1}{K!}K^M \;>\; K^{M-K}$$

such that the number of such partitions is approximately:

$$O(K^M)$$

An obvious alternative to aggregation for resolving the mismatch between the number of groups needed and the number of groups that can be supported is simply to increase the number of groups the network can support. A straightforward mechanism for achieving this is simply to "extend the usable address space" by adding sufficient resources at the hosts and within the routers to support the needed number of multicast groups. The mapping problem is then trivial: each grid cell or region is assigned a unique multicast address.

Some practical considerations make this otherwise straightforward mechanism unworkable. For example, consider the partitioning of a 700km x 500km x 20km play box (typical of some recent STOW exercises) into 500m x 500m x 500m grid cells. This partitioning yields 56 million grid cells in total, and requires 24 bits of addressing capability. If the gridding also provides separation by other attributes (e.g., vehicle speed, EM emitter frequency, etc.) the addressing requirements easily exceed the 28-bit multicast address limit of IPv4. Moreover, if a subscriber wants information from all EM emitters over the entire battlefield, this might require joining (at least!) 56 million groups, with serious implications on the number of data structures and associated memory requirements that must be maintained by the host and network routers.

Virtually Extended Address Space

The alternative, active-networks based approach we are developing can be thought of a providing a *virtual* extension to the usable address space. Our approach combines native network support for multicast (to provide efficient delivery to multiple subscribers) with active network support for installing and modifying *customizable interest filters* within the network routers (to selectively prune irrelevant data from the multicast streams). The core approach embodies the following key aspects:

- *Coarse-grain Multicast Grouping* – The routing space is first coarsely divided into a number of groups that can be reasonably supported by the network infrastructure. This is entirely consistent with interest group management approaches historically used within the HLA/RTI.
- *Embedded Interest Headers* – Methods for encoding and organizing information relevant for interest management (e.g., latitude, longitude, altitude, etc) in a simulation-wide common data construct are defined and agreed to by all federates. Then, for each state update sent by a publisher, the data construct is filled out according to its associated interest information and subsequently embedded as header information within the corresponding state update message. *Note that a perfectly acceptable mechanism here is to define the interest header as being comprised of state*

[2] Controlled coarsening of the quantization grid to reduce the number of grid cells, and therefore the number of groups required, can be cast as a form of aggregation.

update information already embedded within the messages. In this situation, the header is really a virtual entity that imposes no additional processing or message overhead.

- *Multicast Group Subscription* – During the course of the simulation exercise, subscribers and publishers join and/or leave the coarse-grain multicast groups as necessary to assure delivery of needed information. Again, this is consistent with historical interest group management approaches used within the HLA/RTI.
- *Interest Filter Specification* – During the course of the simulation exercise, subscribers submit, modify, and/or retract fine-grained interest filters to active nodes within the network. Interest filters are specified as a set of operations performed on interest headers to determine which update messages on a specific multicast group should be delivered to the subscriber.
- *Interest Filter Distribution* – Active routers propagate interest filter specifications from subscribers (receivers) to publishers (senders) in the reverse direction along the multicast path from sender to receiver. In general, the active routers merge interest filters from downstream entities (subscribers or other active routers) before sending filter specifications upstream both to reduce the overall number of filters needed as well as to reduce the complexity of the filtering expressions.
- *Multicast Routing* – Routers that are not also active routers provide standard support for setting up and maintaining multicast routes in response to join and leave requests from subscribers and publishers. These routers also provide the standard best-effort services for routing multicast packets between interfaces according to its current multicast routing tables. This is also consistent with historical interest group management approaches used within the HLA/RTI.
- *Active Filtering* – In addition to standard multicast routing decisions, routers that are also active routers will perform an additional step of running any relevant interest filters to determine whether a packet should be replicated onto a given outbound interface. Hence message replication onto an outbound interface requires both an admissible multicast route through that interface for the multicast group and that the message contain interest information that is admissible by the associated interest filter for that interface.

Note that the collection of bits constituting the interest header in update messages can well be interpreted as address information since the bits are used to decide which of a given set of end hosts should be sent the message. Hence, the approach can be viewed as providing a virtual extension to the usable address space.

A Simple Interest Filtering Example

The notional operation of a simple interest filter specification and distribution capability is illustrated in Figure 6. Here, the routing space is represented as larger squares, and needed data regions are represented as colored cells within the squares. Interest filters from each subscriber (receiver) are forwarded upstream towards the publisher (sender). Active routers along the path intercept interest filters from both subscribers and downstream active routers and merge the filters into a single, more general interest filter that is subsequently forwarded upstream.

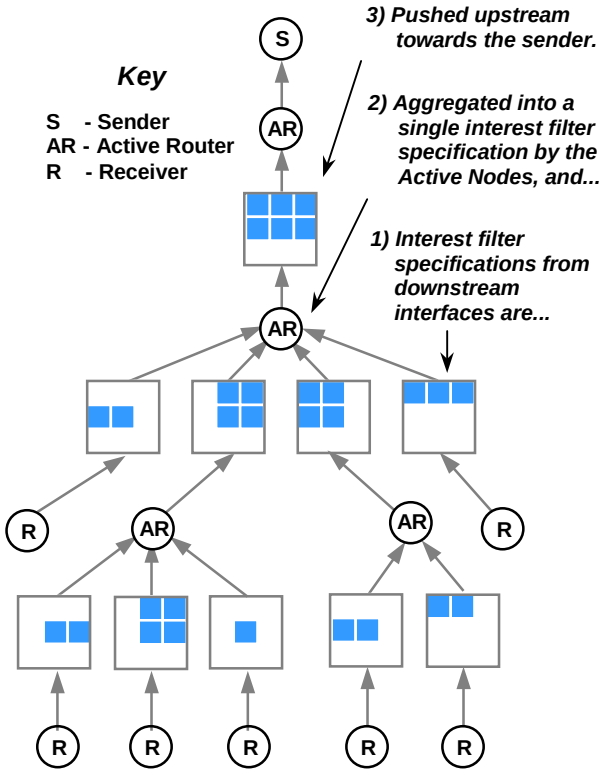

Figure 6: Simple Interest Filter Specification and Distribution Example

The corresponding operation of the active interest filtering function is illustrated in Figure 7. Here, the sender transmits a message with an embedded interest header to the multicast group. (Note: it is assumed that all subscribers shown are members of the group). Upon receipt of the message, each active router compares the information in the interest header with the interest filter associated with each outbound interface. The message is replicated only onto those interfaces with interest filters that will admit the message.

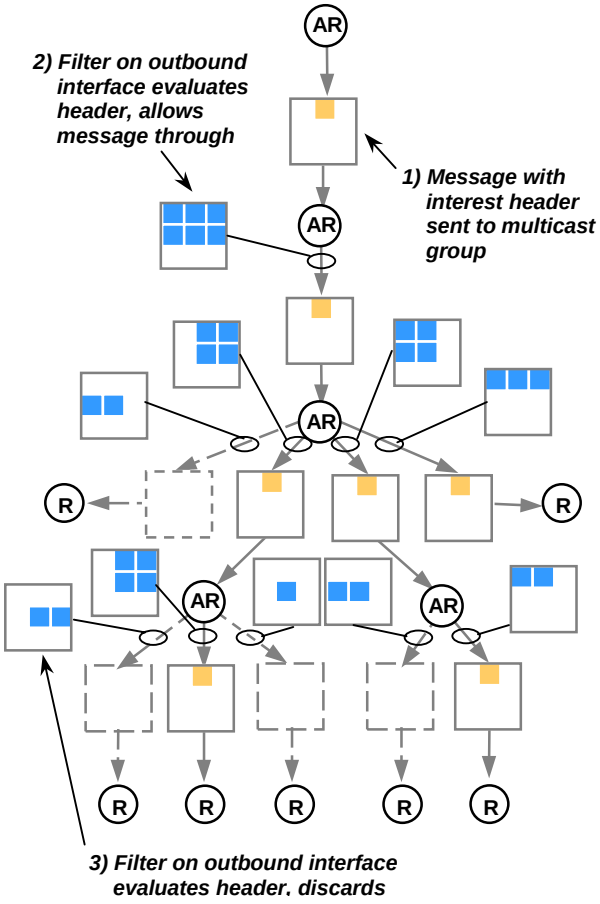

Figure 7: Simple Active Filtering Example

Note that in this example, each of the active routers has forwarded an aggregate filter specification that is *exact* in the sense that no simplifications were made by any of the active routers (e.g., as needed to meet performance goals.) Hence, even though all subscribers are in fact members of the single multicast group, only those subscribers needing the data actually receive the message. As such, no irrelevant data is received, and the system behaves as if a larger number of more precisely specified groups are in use.

Distributed Filtering: Further Improvements

This simple example, while straightforward architecturally, hints at potential scalability and performance limitations. In particular, for distributed simulation exercises with large numbers of simulations, it is easy to imagine that aggregate interest filters could quickly become arbitrarily complex. Once filters become sufficiently complex, routers will no longer be able to complete filtering operations at line speeds and overall throughput suffers.

An interesting approach for improving overall filter performance arises from the observation that a significant amount of the interest filtering performed at each node in the previous example is in fact highly redundant and therefore unnecessary. As a thought experiment, suppose that all receivers in the simple example have identical interest filters. Then, each router node downstream from the first (uppermost) node is bound to repeat verbatim each of the filtering operations performed by the first node – when in fact *no* additional filtering need be performed after the first node. The key insight here is that if active nodes can be provided with accurate and timely knowledge of the filters being executed upstream, then the overall filtering load can be reduced.

A reasonable approach for distributing the filter load involves using a two-pass filter specification technique. In this approach, the first (upstream) pass is identical to that of the original example. As before, receivers express interests as individual interest filters that are forwarded upstream towards the sender. Similarly, active routers along the path intercept interest filters from both receivers and downstream active routers, and merge the filters into a single, more general interest filter that is then forwarded upstream. For clarity, we here refer to filters propagating upstream as *request* interest filters.

The second (downstream) pass begins once a request interest filter reaches the sender. The sender takes the request interest filter and performs an intersection with its update region (i.e., the region describing what information the sender can actually provide). The resultant, possibly reduced interest filter is then sent downstream towards the receivers as a *response* interest filter. Each active routers along the forwarding path intercepts the response interest filter and uses it to calculate a supplementary filter for each downstream interface. A supplementary defines what additional filtering is to be done to best support the aggregate needs of downstream

constituents for a given interface. The aggregation of the intercepted response interest filter with the supplementary filter for a given interface becomes the response interest filter for that interface, which is subsequently forwarded downstream.

The potential advantage of a distributed filtering approach over the simple filtering example is shown in Figure 8. Here, the filtering decisions made at each stage are (for the most part) single decision planes, with a total of 11 decision planes needed to implement the entire tree. This number contrasts sharply with the simple example, which would require 23 decision planes to implement the same filtering structure.

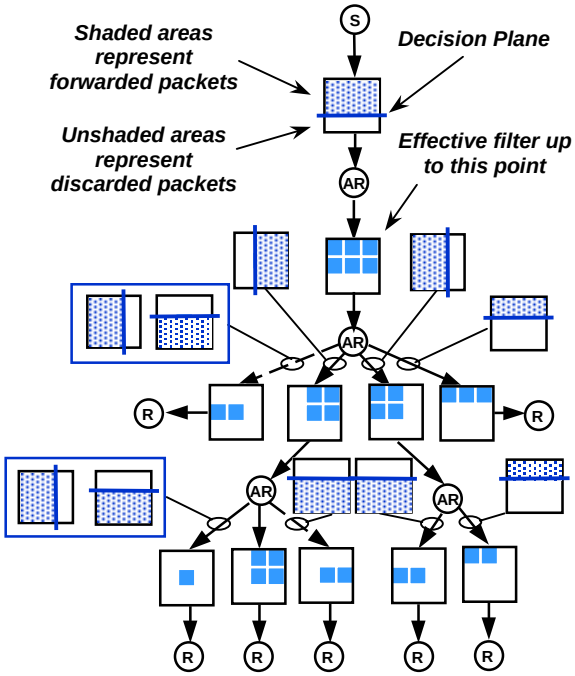

Figure 8: Cooperative Distributed Interest Filtering Example

Filter Simplification to Improve Throughput

As a further mitigation strategy for maintaining or improving throughput, the aggregation process may also elect to reduce the filter complexity with a simpler set of approximate filters. For example, a single larger interest region may "correctly" be used in lieu of a large number of small interest regions as long as the larger region contains all of the smaller regions. Although this potentially allows additional, unwanted network traffic through, it is still perfectly acceptable under the correctness criteria since no needed data is discarded.

The results of "optimally" merging a set of 8 interest filters into a smaller target number of filters is shown in Figure 9. Here, optimal means that the added volume (i.e., the volume within the simplified filter that was not in one of the original filters) is minimized. Assuming a uniform traffic density model, minimizing the volume is equivalent to minimizing the additional, unwanted traffic, and hence is reasonable.

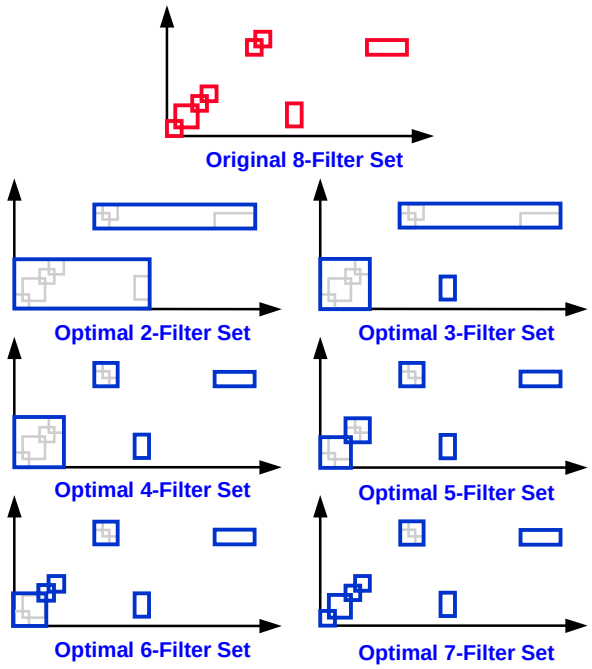

Figure 9: Optimal Minimum Volume Filter Simplification

Although we have implemented an optimal N-dimensional filter simplification capability as part of our experimental work under SANDS, optimal filter simplification is for the most part impractical as the problem has NP-Hard computational complexity [6]. To gain insight into the origins of this complexity, consider the number of ways that a set of *M* filters may be partitioned into *K* unique subsets. This number is in fact given by the Stirling number of the second kind described in a preceding section, and which was shown to be

$$O(K^M)$$

For added perspective, measurements of the execution time required to reduce some number of original filters (from 7 to 14) to a fixed target number of filters (4) is presented in Figure.10.

Given the nearly linear relationship between the number of original filters and the logarithm of the run time indicated in this figure, the exponential relationship between the execution time and the number of input filters is revealed.

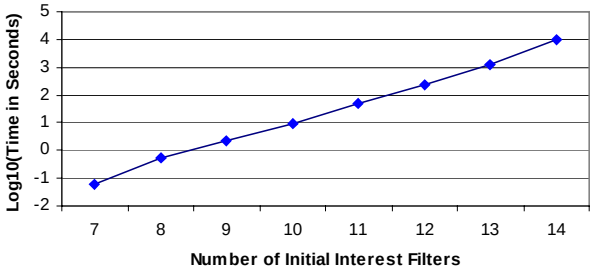

Figure 10: Execution Time for Optimal Filter Simplification on a 700 MHz Pentium III

To address the computational complexity problem inherent with optimal filter simplification, we have developed a very promising albeit suboptimal approach based on a technique used in designing Vector Quantizer (VQ) codebooks for data compression applications. In general, the VQ codebook design problem is one of constructing a low cardinality set (the set of VQ code vectors) that is used to represent a much larger cardinality set (the space of all possible input vectors) so that the distortion inherent in the representation is in some sense minimized. Clearly the notion of representing a large number of filters by a smaller number of filters is analogous with the VQ cardinality reduction objective, and the notion of doing so in a way that minimizes excess traffic or excess volume is aligned with the VQ optimization goals.

Drawing from this analogy, we have adapted the Fast Pairwise Nearest Neighbor (PNN) algorithm [7] to perform filter simplification. The Fast PNN algorithm basically works as follows: the current set of elements (initially, the set of original filters for our purposes) is recursively split into smaller and smaller groups (based on proximity) until the number of elements in each group reaches some objective value (e.g., 8 elements per group). Within each group, the pair of elements that if merged would produce the smallest amount of distortion (e.g., excess volume) is identified as the candidate pair for the group. Here, merging two elements translates to approximating the pair by a single element containing the pair. The candidate pairs, one from each of the groups, are then ordered according to the amount of distortion, and some fraction (e.g., 30%) of the pairs are replaced by corresponding single elements that result from the merge process. All elements are then lumped back into a single group, and the process repeats until the target number of elements is reached.

The Fast PNN algorithm is attractive in that not only have its results proven quite good operationally for VQ applications, but it has an *extremely* attractive computational complexity - particularly in comparison with the optimal approach. Specifically, the computational complexity for reducing a number of elements M to some target number K using the Fast PNN algorithm is:

$$O(M \log(M))$$

which is essentially independent of K.

Initial experimental results using this approach have been quite promising. For example, testing using the Fast PNN algorithm against the previous example problem of reducing eight filters to a target number between two and seven filters yielded reduced filters that were identical to those produced by the optimal algorithm. Moreover, from a processing speed perspective, the optimal implementation took nearly three hours to reduce a set of 14 filters to a set of four filters. By comparison, the Fast PNN was able to reduce a set of 24 filters to a set of four filters in *under one one-hundredth of a second*. Using the optimal algorithm to solve the 24 filter problem would take an estimated 354 years.

The near linear scaling properties for the Fast PNN-based filter simplification approach are shown in Figure 11 for a range of 10 to 10,000 input filters and a reduction target of four filters.

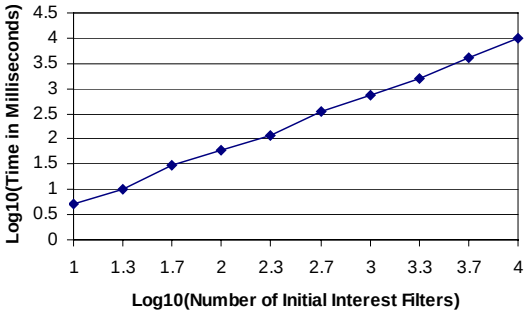

Figure 11: Execution Time for Fast PNN-Filter Simplification on a 400 MHz Pentium III

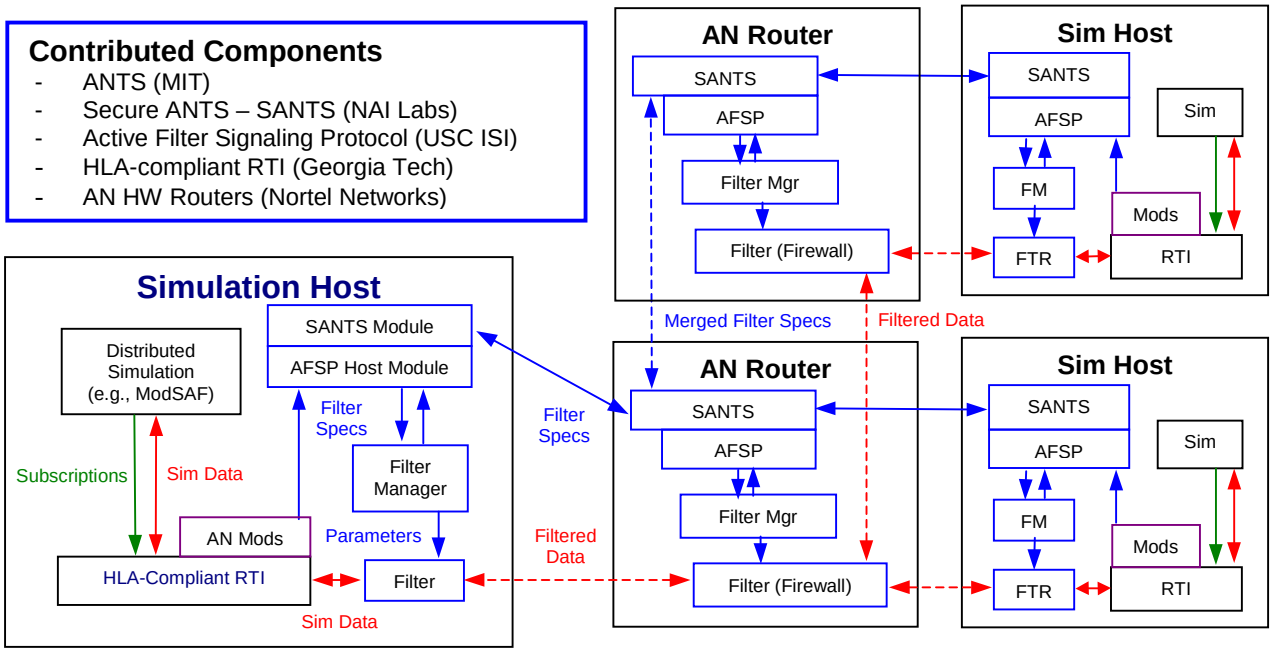

Figure 12: SANDS Active Filtering Architecture for Distributed Simulation

SANDS Active Filtering Architecture

The active filtering architecture we are developing under the SANDS project leverages numerous components being developed by other researchers as shown in Figure 12. For example, the Active Filter Signaling Protocol (AFSP) shown in Figure 12 is derived from a Java implementation of the ReSerVation Protocol (RSVP) [8] designed and implemented by the Information Sciences Institute at the University of Southern California (ISI/USC). AFSP provides the core signaling mechanism used to communicate filter information between hosts and active routers, and is the result of a collaborative design effort between us and ISI. AFSP is currently being implemented by ISI as part of our Distributed Simulation Team demonstration for DARPA's Active and High Confidence Networking (AHCN) Program.

As the Filter Manager and Filtering mechanisms we are developing are fundamentally dynamic, programmable firewalls, such capabilities could cause substantial denial of service outages if used maliciously or even incorrectly. Consequently, provisioning active network security mechanisms to authenticate and authorize filter installation requests is crucial. Towards this end, all communications between AFSP components is handled by Secure ANTS (SANTS), designed and implemented by Network Associates, Inc. Laboratories (NAI Labs). SANTS is basically a secure version of the Active Network Transport System (ANTS) [9] developed by the Massachusetts Institute of Technology (MIT).

Although one of our main objectives is developing a capability that requires *no* changes to HLA-compliant simulations, our approach *does* require modest modifications to the supporting RTI. Basically, the modifications are used to provide the necessary information (i.e., the interest region definitions, the location of interest region variables within the Protocol Data Units (PDUs), and multicast session information) to the active network via the AFSP host module. At the time we began the SANDS effort, source code for HLA-compliant RTIs was singly unavailable. Fortunately, researchers at the Georgia Institute of Technology agreed to extend their RTI [10] to provide the suite of HLA interfaces needed by simulations such as the pervasive Modular Semi-Automated Forces (ModSAF) simulation and its variants, as well as the DM and DDM components needed for our work. It is to their RTI that we are adding the active filtering enhancements. For this work we are in their debt.

As our goal is *improving* the performance of distributed simulations over what is currently achievable over existing networks using commercial routers, it is essential for our demonstrations that we use network equipment of comparable performance. As described earlier, one of our key design considerations was

developing an approach that can leverage high-speed programmable hardware such as the Nortel Accelar series. Towards this end, Nortel is also working with us as part of our demonstration team for DARPA's AHCN Program.

Preliminary Active Filtering Performance

Although our goal is to leverage programmable hardware such as the Nortel routers, at this point such equipment is not yet capable of supporting the types of user-defined filtering needed with our approach (although the needed capabilities are expected to be in place by the Winter of 2000.) In order to evaluate the potential performance impacts of user filtering (as well as support interim demonstrations) we have therefore developed and integrated the needed capabilities as extensions to the kernel packet firewall code (*fw*) within Linux-based software routers.

The results from initial experiments, shown in Figure 13, are extremely encouraging. The experimental setup consisted of a multicast sender and a multicast receiver connected to separate ports (network interfaces) on a Linux-based software router. The software router consisted of a 300MHz Pentium II with a pair of 100 Mbps ethernet cards running the modified Linux kernel described above. The experiment consisted of the send host multicasting packets with interest information embedded within the packet payload as fast as possible, and the receive host simply counting the number of packets received over the duration of the transmission. The software router between the sender and receiver was configured with N-1 "miss" filters (i.e., filters that did not match the information within the packet) and a final "hit" filter (i.e., a filter that did match the information within the packet) with the rule for the "hit" filter being to forward the packet. Hence the router was forced to evaluate N filters before forwarding the packet, with values of N ranging from 10 to 2000 in this experiment.

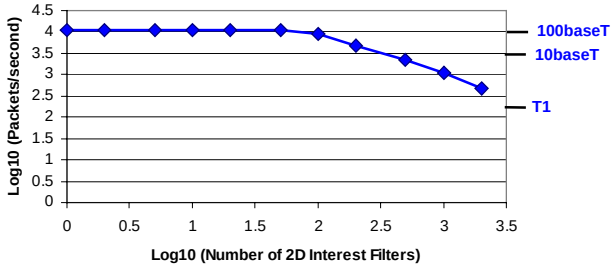

Figure 13: 2D Interest Filtering Performance on 300 MHz Pentium II Running Linux

As shown in Figure 13, even this relatively "slow" router (by current standards at least) was able to sustain full 100 Mbps line rate throughput for values of *N* well beyond 50, and appears capable of sustaining T1-speed or better throughput (1.5 Mbps, typical of WAN links) out to around 5000 filters. This *linear* filtering test (i.e. where all filters are applied in a single linear sequence) provides significant evidence that the prescribed active interest filtering approach will well support the needs of large simulations.

This possibility is considerably strengthened by the notion that a set of filters can potentially be organized and processed as a binary tree rather than as a linear sequence. A binary tree-based filtering approach potentially enables a *substantial* reduction in the number of filters that must be applied to any given packet. If realizable, this number would be proportional to the logarithm of the total number of interest filters expressed by the distributed simulation as a whole.

The truly exciting implication here is that if filter sets can in fact be effectively organized as binary trees, then it is entirely possible that *no* simplification of filter sets will be needed or required. If no filter simplification is performed, all filtering is therefore exact. What remains, then, is the enticing possibility that this approach can achieve the ultimate interest filtering objective, where all needed data and *no* unneeded data are delivered to the simulation applications. To date, we have made excellent progress towards developing techniques for organizing interest filters as binary trees, and the development of these techniques has now become one of the key thrusts of the TASC-UMass team effort.

SUMMARY

We are leveraging the capabilities of Active Networks to provide enhanced HLA interest management services. For distributed simulation applications, an Active Networks approach for interest management offers several important benefits:

(i) Because data forwarding is decoupled from multicast route setup, an entity can rapidly and dynamically select which data a router will forward. This selection can be done at a sub-millisecond timescale, as compared to the multiple-second timescale imposed by the existing multicast group join/leave mechanisms.

(ii) Because each entity can install its own filters, information filtering is accomplished in a "receiver-driven" manner, allowing each entity to customize its filters according to its own need. This decentralized approach allows active filtering to scale well as the number of entities grows large.

(iii) Because active filtering is performed at a routing point, filtering can also be dependent on the state (e.g., congestion-level) at that router. In particular, this allows both entities and network routers to determine which data should be shed in times of congestion.

Active interest filtering also enables an effective means for routers to mediate among conflicting demands.

Although we are currently still in the development stages, our end goal is to demonstrate that increased distributed simulation scalability and performance can be achieved with these techniques using an approach that is fully HLA-compliant. Towards this end, we are integrating the developed capabilities with an HLA-compliant RTI and intend to demonstrate its improved performance in the context of a meaningful, realistic set of simulation scenarios. Initial performance results for the two performance-critical components (i.e., the Filter Manager and the Filter) have been extremely encouraging. Based on these results we expect that an integrated system will provide a strong and convincing demonstration of the viability of this approach as well as the potential power of active networking technologies.

REFERENCES

[1] D. Tennenhouse, J. Smith, W. Sincoskie, D. Wetherall, and G. Minden, "A Survey of Active Network Research", IEEE Communications Magazine, Vol. 35, No. 1, pp80-86. January 1997.

[2] S. Kasera, S. Bhattacharyya, M Keaton, D. Kiwior, S. Zabele, J. Kurose, and D. Towsley, "Scalable Fair Reliable Multicast Using Active Services", IEEE Network Magazine, Vol. 14, No. 1., January-February 2000.

[3] S. Zabele and T. Stanzione, "Interest Management Using an Active Networks Approach", 00S-SIW-030, Spring 2000 Simulation Interoperability Workshop, March 26-31, 2000.

[4] D. Van Hook, S. Rak, and J. Calvin, "Approaches to Relevance Filtering", 94-11-144, Eleventh Workshop on Standards for the Interoperability of Distributed Simulations, September 26-30, 1994.

[5] Department of Defense High Level Architecture Interface Specification, Version 1.3, DMSO, April 1998

[6] R. E. Burkard and M. M. Miatselski, "Volume Maximization and Orthoconvex Approximation of Orthogons", Computing, 63, 1999

[7] W. H. Equitz, "A new Vector Quantization Clustering Algorithm", IEEE Trans. ASSP, Vol. 37, No. 10, October 1989

[8] R. Braden, Ed, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification" IETF RFC 2205, September 1997

[9] D. Wetherall, J Guttag, and D. Tennenhouse, "ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols" IEEE OPENARCH'98, San Francisco, CA, April 1998

[10] R. M. Fujimoto and P. Hoare, "HLA RTI Performance in High Speed LAN Environments", Proceedings of the Fall Simulation Interoperability Workshop, September 1998