Active Virtual Network Management Prediction

Stephen F. Bush
General Electric Corporate Research and Development
KWC-512, One Research Circle, Niskayuna, NY 12309
bushsf@crd.ge.com (http://www.crd.ge.com/~bushsf)

Abstract

Active Networking provides a framework in which executable code within data packets can execute upon intermediate network nodes. Active Virtual Network Management Prediction (AVNMP) provides a network prediction service that utilizes the capability of Active Networks to easily inject fine-grained models into the communication network to enhance network performance. The models injected into the network allow state to be predicted and propagated throughout an active network enabling the network to operate simultaneously in real time and in the future. State information such as load, security intrusion, mobile location, faults, and other state information found in typical Management Information Bases (MIB) is available for use by the management system both with current values and with values expected to exist in the future. Implementing a load prediction and CPU prediction application has experimentally validated AVNMP. AVNMP implements a distributed, active, and truly proactive network management system. Active Networking enables the implementation of new concepts utilized in AVNMP such as the ability to quickly and easily inject models into a network. In addition, Active Networking enables the ability of messages to refine their prediction as they travel through the network as well as several enhancements to the basic AVNMP algorithm, including migration of AVNMP components and reduction in overhead by means of message fusion.

Introduction: What is AVNMP?

Active Virtual Network Management Prediction (AVNMP) provides a network prediction service designed to facilitate the management of large, complex, active networks [6,11] in a proactive manner. Network management includes a wide variety of responsibilities including configuration, fault, performance, accounting, and security management. A network management system must be able to monitor, control, and report upon the status of all these areas. In addition, the network management system should be more than a tool to generate reports and help fix problems, it should have the capability to anticipate and correct problems before they impact network performance. AVNMP accomplishes prediction and fault anticipation using a novel coupling of concepts from distributed simulation and active networking. A simple example demonstrating AVNMP results on a single node for load prediction is shown in Figure 1. In today's management systems, a Management Information Base (MIB) maintains only current state values. In AVNMP, load is predicted into the future as real-time, called wallclock, advances. Thus anticipated future values are available on the node as well as current values. In Figure 1, the Local Virtual Time (LVT) (future time), runs ahead of Wallclock Time (current time). Predicted load values are refined until Wallclock reaches the LVT of a particular value. This capability, described in detail in this paper, has been enabled by a new proactive network management framework combining three key enabling technologies; namely, distributed simulation, optimistic synchronization, and active networks. The next section discusses a high-level view of the framework and describes its various components.

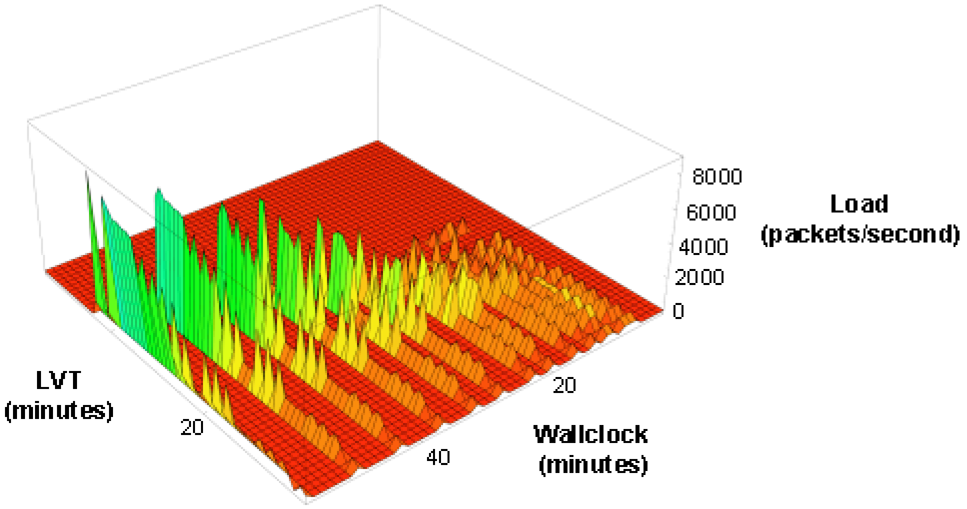

Figure 1: Load Prediction Application Results on a Single Node.

The predictive capability provided by AVNMP facilitates the development of a variety of predictive applications from mobile wireless location management and network security to improved QoS. AVNMP provides an ideal predictive service for mobile systems to predict their location [4]. Because the mobile location can be predicted, hand-off situations are known ahead of time and setup for hand-off can take place prior to the hand-off event resulting in fast hand-off and improved QoS. In the domain of network security, AVNMP can anticipate the progress of an attack along most likely vulnerability paths and incorporate that information into decision-making [13]. An attack can be propagated through the system before it actually occurs in order to determine its impact. With regards to QoS, the load application previously discussed allows resources and routing to be better managed by anticipating traffic in order to optimize load distribution within the network. A few additional selected uses for AVNMP are the ability to choose an optimum management polling interval that minimizes overhead based upon predicted rate of change and fault probability of the monitored data in a managed entity, fault correction before the system is impacted and with time available to perform dynamic optimization of repair parts, service, and solution entities such as software agent or human co-ordination and optimal resource allocation and planning not only for load, but also for CPU utilization that becomes significant in active networks. AVNMP allows “What if…?” scenarios to become an integral part of the network and finally, AVNMP-enhanced components are enabled with the ability to protect themselves by taking proactive, evasive action, such as migrating to safe hardware before anticipated disaster occurs.

A severe limitation of state-of-the-art network management techniques is that they are inherently reactive. They attempt to isolate the problem and determine solutions after the problem has occurred. Proactive management is a necessary ingredient for managing future networks. Part of the proactive capability is provided by analyzing current performance and predicting future performance based on likely future events and the network’s reaction to those events. This can be a highly dynamic, intensely computational operation. This has prevented management software from incorporating prediction capabilities. But distributed simulation techniques take advantage of parallel processing of information. If the management software can be distributed, it is possible to perform computation in parallel and aggregate the results to minimize computation overhead at each of the network nodes. The usefulness of optimistic techniques has been well documented for improving the efficiency of simulations. In optimistic logical process synchronization techniques, also known as Time Warp [7, 8], causality can be relaxed in order to trade model fidelity for speed. If the system that is being simulated can be queried in real time, prediction accuracy can be verified and measures taken to keep the simulation in line with actual performance. AVNMP is implemented in an

Active Network to provide predictive management of an Active Network. AVNMP is designed to utilize the additional processing and flexibility of an Active Network to provide better management of the added complexity in processing and bandwidth in an active network. AVNMP requires extreme network flexibility, primarily in the ability to inject fine-grained component models into the network. A much less flexible version of AVNMP could be implemented in legacy systems by building dedicated network component models directly into legacy network devices such as today's routers. However, these models would be immobile and not easily updated or removed, most likely requiring the network device to be taken down when models are changed or updated. A better mechanism for using AVNMP to manage legacy networks would be to provide an active network overlay capable of monitoring legacy nodes. AVNMP should reside in the Active Network overlay providing a predictive management service for the legacy network. This has the added benefit of transitioning a legacy network to an active network.

AVNMP ARCHITECTURE

This section describes the AVNMP architecture designed to utilize a significant benefit of Active Networking: the ability to utilize fine-grained executable models in the network to enhance communication. These fine-grained models, or Streptichrons, are introduced as active packets expected to exist in the future that carry executable code necessary to represent future behavior. The executable code necessary to represent future behavior is designed in a more compact form than transmitting equivalent, static, non-executable data in a piecemeal fashion. The algorithmic nature of the streptichron allows for high compression. As a simple example, a million digits of π could be transmitted, or more compactly, the description of a circle and the command to divide the circumference by the diameter could be transmitted. Because of the new paradigm and enhanced capabilities of active networks, this work proceeds along the lines that a radical new perspective in understanding and implementing network management architectures should be taken.

TEMPORAL OVERLAY

The approach taken by AVNMP is to inject an optimistic parallel distributed simulation of the network into the active network. This can be viewed as a virtual overlay network running temporally ahead of the actual network**.** As shown in Figure 2, a virtual network, representing the actual network can be viewed as overlaying the actual network. A motivating factor for this approach is apparent when AVNMP is viewed as a model-based predictive control technique where the model resides inside the system to be controlled. The environment is an inherently parallel one; using a technique that takes maximum advantage of parallelism enhances the predictive capability. AVNMP dynamically keeps the predictions within a given tolerance of actual values. Thus the model-based predictive system gains speedup due to parallelism while maintaining prediction accuracy, discussed in greater detail in following sections of this paper.

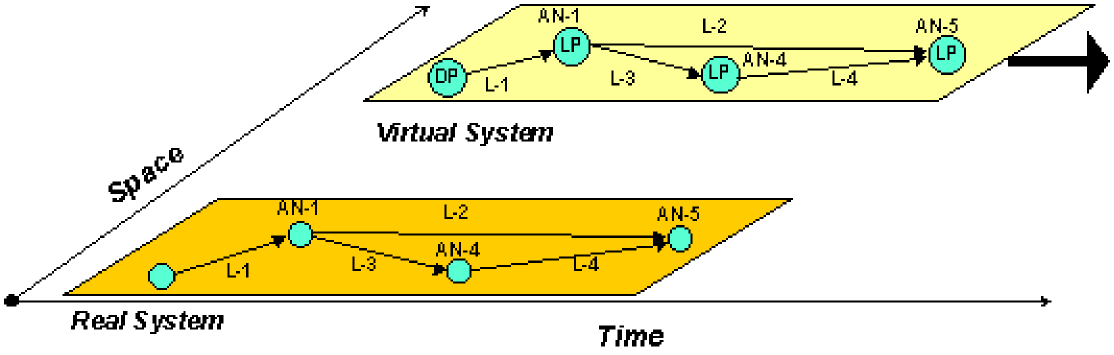

Figure 2: Temporal Overlay.

AVNMP is comprised of Driving Processes, Logical Processes, and Streptichrons. Streptichrons are active messages expected to exist in the future, also referred to as virtual messages. The Logical Processes and Driving Processes execute within an Active Network Execution Environment (EE) on each active network node. The Logical Process manages the execution of the virtual overlay on a single node and is primarily responsible for handling rollback. Rollback can be induced by out-of-order Virtual Message arrivals and by prediction inaccuracy. A tolerance is set on the maximum allowable deviation between the predicted values and the actual values. If this tolerance is exceeded, a rollback to Wallclock time occurs. The Logical Process' notion of the current time increments based upon the value of the virtual messages' Receive Time discussed in detail shortly. A Sliding Lookahead Window is maintained so that a specified distance bounds the Logical Processes' virtual time progression into the future. The Driving Process monitors the input to that portion of the network enhanced by AVNMP and generates the virtual messages that drive the AVNMP Logical Processes forward in time. The driving process monitors the actual application via a general management framework mechanism; such as via the Simple Network Management Protocol [5] implemented within the active network environment. The driving process samples the values to be predicted and generates a prediction. The actual mechanism used for predicting output from any application is application dependent and de-coupled from the system. However, a simple curve-fitting algorithm based upon past history has worked adequately well for prototyping load prediction.

AVNMP ARCHITECTURAL COMPONENTS

In describing the AVNMP architecture, a distinction is made between a Physical Process and a Logical Process. A Physical Process is nothing more than an executable task defined by program code. An example of a Physical Process is the packet forwarding process on a router. AVNMP encapsulates each Physical Process within a Logical Process, labeled LP in Figure 2. A Logical Process consists of a Physical Process, or a model of the Physical Process and additional data structures and instructions that maintain message order and correct operation as the system executes ahead of Wallclock time as illustrated in greater detail in Figure 3. The Logical Process can be designed in a lightweight manner by utilizing the Physical Processes' existing code. The Logical Process can also utilize a model of the Physical Process instead of the actual Physical Process.

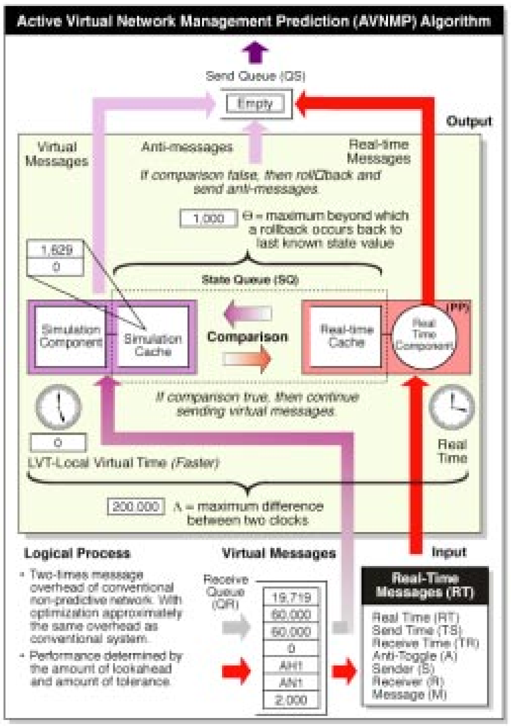
Active Virtual Network Management Prediction (AVNMP) Algorithm
Send Queue (QS)
Empty
Output
Virtual Messages
Anti-messages
Real-time Messages
If comparison false, then roll back and send anti-messages.
1,000
Θ = maximum beyond which a rollback occurs back to last known state value
1,629
0
State Queue (SQ)
Simulation Component
Simulation Cache
Comparison
Real-time Cache
Real Time Component
(PP)
If comparison true, then continue sending virtual messages.
0
LVT-Local Virtual Time (Faster)
Real Time
200,000
Λ = maximum difference between two clocks
Logical Process
• Two-times message overhead of conventional non-predictive network. With optimization approximately the same overhead as conventional system.
• Performance determined by the amount of lookahead and amount of tolerance.
Virtual Messages
Receive Queue (QR)
19,719
60,000
60,000
0
AH1
AN1
2,000
Input
Real-Time Messages (RT)
Real Time (RT)
Send Time (TS)
Receive Time (TR)
Anti-Toggle (A)
Sender (S)
Receiver (R)
Message (M)

Figure 3: AVNMP Logical Process and Message Structure.

The architecture is described in detail by following the flow of messages through the system shown by the arrows entering from the bottom of Figure 3. AVNMP messages contain the Send Time (TS), Receive Time (TR), Anti-toggle (A) and the actual message object itself (M). The Receive Time is the time this message is predicted be valid at the destination Logical Process. The Send Time is the time this message was sent by the originating Logical Process. The "A" field is the anti-toggle field and is used for creating an anti-message to remove the effects of false messages as described later. A message also contains a field for the current Real Time (RT). This is used to differentiate a real message from a virtual message. A message that is generated and time-stamped with the current time is called a real message. Messages that contain future event information and are time-stamped with a time greater than the current wallclock time are called virtual messages. If a message arrives at a Logical Process out-of-order or with invalid information, that is, out-of-tolerance, it is called a false message. A false message causes a Logical Process to rollback. The details of rollback are discussed in a following section of this paper. The Receive Queue, shown in Figure 3, maintains newly arriving messages in order by Receive Time (TR). The Receive Queue is an object residing in an active node's SmallState. SmallState is state left behind by an active packet. The Magician [3] execution environment is used in the implementation described in this work. The Magician execution environment allows any type of information to be stored in SmallState including Java objects; the Receive Queue is a Java object maintaining active virtual message ordering and scheduling. Packets are encapsulated inside Magician SmartPackets following the Active Network Encapsulation Protocol [1] format. Once a virtual message leaves the Receive Queue, the virtual time of the logical process, known as Local Virtual Time, is updated to the value of the Receive Time of that virtual message.

Virtual messages ultimately originate from Driving Processes, shown in the virtual overlay of Figure 2, which predict future events and inject them into the system as virtual messages. All other processes react to virtual messages. Continuing to follow the arrows upward in Figure 3, real messages directly enter the Physical Process and virtual messages enter either the Physical Process, or a model of the Physical Process depending upon how the Logical Process is implemented. The state of the Logical Process is periodically saved in the State Queue (SQ) shown as the Simulation Cache in Figure 3. A particular sample, shown in the Figure 3, contains a value and Local Virtual Time the value existed. These state values are used to restore the Logical Process to a known safe state when false messages are received. State values are continuously compared with actual values from the Physical Process to check for prediction accuracy. If the prediction error exceeds a specified tolerance, shown as Θ in Figure 3, a rollback, which is described in detail in a later section, occurs. An important part of the architecture for network management is the fact that the State Queue within the AVNMP architecture is the network Management Information Base. The AVNMP State Queue values are the Simple Network Management Protocol Management (SNMP) [5] Information Base Object values; but unlike legacy SNMP values, these values are expected to occur in the future.

Continuing along the upward path of the arrows in Figure 3, any virtual messages that are generated as a result of the Physical Process or model computation proceed to the Send Queue (QS). The Send Queue maintains copies of virtual messages to be transmitted in order of their send times. The Send Queue is necessary for the generation of anti-messages during rollback. Anti-Messages annihilate corresponding virtual messages when they meet to correct for previously sent false messages. Rollback is described in detail in the following section. After leaving the Send Queue, virtual messages travel to their destination Logical Process.

Summary

This paper presented the Active Virtual Network Management Prediction (AVNMP) architecture and the mechanism by which that architecture provides a network prediction service that utilizes the capability of Active Networks to easily inject fine-grained models into the communication network to enhance network performance. AVNMP, injected into the network as an active application, was shown to be capable of modeling load and propagating state information in a manner that met the demand for accuracy at a particular active node. Greater demand was met at the cost of AVNMP performance, that is, the ability of AVNMP to predict farther into the future. While this paper has focused on network traffic and load prediction, work is continuing on an AVNMP application to predict CPU utilization for Active Networks in collaboration with National Institute of Standards and Technology [14]. The inherently distributed nature of communication networks and the computational power unleashed by the Active Networking paradigm have been used to mutual benefit in the development of the Active Virtual Network Management Prediction mechanism. Active Networks benefit from AVNMP by continuously receiving information about potential problems before they occur. AVNMP benefits from Active Networks in many ways. The first, and most practical way, was the ease of development and deployment of this novel protocol. This could not have been accomplished so quickly or easily given today's closed, proprietary network device processing. Another benefit is the fact that network packets now have the unprecedented ability to control their own processing. Great advantage was taken of this new capability in AVNMP. Virtual messages, varying widely in content and processing, can adjust their predicted values as they travel through the network. Finally, Active Networks add a level of robustness that cannot be found in today’s networks. This robustness is due to the ability of the AVNMP system components, which are active packets, to easily migrate from one node to another in the event of failure -- or the prediction of failure provided by AVNMP.

References

[1] D. S. Alexander, B. Braden, C. Gunter, A. Jackson, A. Keromytis, G. Minden, and D. Wetherall, editors. *Active Network Encapsulation Protocol (ANEP)*. Active Networks Group, July 1997. Request for Comments: draft.

[2] Y.-C. Ho. Perturbation Analysis: Concepts and Algorithms. In *Proceedings of the 1992 Winter Simulation Conference*, 1992.

[3] A. B. Kulkarni, G. J. Minden, R. Hill, Y. Wijata, S. Sheth, H. Pindi, F. Wahhab, A. Gopinath, and A. Nagarajan. Implementation of a Prototype Active Network. In *OPENARCH '98*, 1998.

[4] E. Lazowaska and Y.-B. Lin. Determining the Global Virtual Time in a Distributed Simulation. Technical Report 90-01-02, University of Washington, 1990.

[5] M. T. Rose. *The Simple Book, An Introduction to the Management of TCP/IP Based Internets*. Prentice Hall, 1991.

[6] D. L. Tennenhouse, J. M. Smith, W. D. Sincoskie, D. J. Wetherall, and G. J. Minden. A survey of active network research. *IEEE Communications Magazine*, 35(1):80--86, Jan. 1997.

[7] P. Tinker and J. Agra. Adaptive Model Prediction Using Time Warp. In *SCS '90*, 1990.

[8] Stephen F. Bush. *Active Virtual Network Management Prediction*. In Parallel and Discrete Event Simulation Conference (PADS) '99, May 1999.

[9] Stephen F. Bush. *Active Jitter Control*. In Intelligence in Services and Networks (ISN) '00, February 2000.

[10] Stephen F. Bush. *Islands of Near-Perfect Prediction*. In *Virtual Worlds Simulation Conference '00 and 2000 Western MultiConference*, January 2000

[11] Bush, Stephen F. and Kulkarni, Amit B. *Active Networks and Active Virtual Network Management Prediction: A Proactive Management Framework*. ISBN 0-306-46560-4. Kluwer Academic / Plenum Publishers. Spring 2001.

[12] A. B. Kulkarni and S. F. Bush. *Active Network Management, Kolmogorov, Complexity, and Streptichrons*. GE-CRD Class I Technical Report 2000CRD107 (www.crd.ge.com/crd_reports/index.htm)

[13] Stephen F. Bush and Bruce Barnett. *A Security Vulnerability Technique and Model*. GE Corporate Research and Development, January 1998. Technical Report 98CRD028.

[14] Stephen F. Bush and Amit Kulkarni (GE), Virinie Galtier and Yannick Carlinet and Kevin L. Mills (NIST), Livio Ricciulli (Metanetworks*). Predicting and Controlling Resource Usage in an Active Network*. DARPA Active Networks PI Meeting, December 6-9, 2000, Atlanta, GA.