REAL-TIME PDU FILTERING IN THE GATEWAYS OF WIDE AREA SIMULATION NETWORKS

M. A. Bassiouni and M. Chiu
Computer Science Department
University of Central Florida
Orlando, FL 32816

M. P. Garnsey
STRICOM
12350 Research Parkway
Orlando, FL 32826

ABSTRACT

Achieving the real-time linkage among multiple, geographically-distant, local area networks that support distributed interactive simulation (DIS) is one of the major technical challenges facing the implementation of future large-scale training systems. Data filtering is a technique that can help achieve this real-time linkage. In this paper, we present the results of a detailed study to design and evaluate the performance of data filtering for DIS systems. An approach suitable for the implementation of data filtering in the gateways of DIS networks is given and detailed performance results are presented. The paper is concluded by discussing methods to solve the problem of inaccurate state information at high filtering rates.

ABOUT THE AUTHORS

M. A. Bassiouni received his Ph.D. degree in Computer Science from the Pennsylvania State University in 1982. He is currently an Associate Professor of Computer Science at the University of Central Florida, Orlando. He has been actively involved in research on computer networks, distributed systems, concurrency control, and relational databases. Dr. Bassiouni is a member of IEEE and the IEEE Computer and Communications Societies, the Association for Computing Machinery, and the American Society for information Science.

M. Chiu received his masters degree in Computer Science from the University of Central Florida in 1991. He is currently a Ph.D. student and Research Assistant in the Department of Computer Science, UCF.

Mike Garnsey is a System Engineer with the U.S. Army Simulation Training and Instrumentation Command (STRICOM). He is currently working in the Combined Arms Technology Division on Distributed Interactive Simulation efforts such as the Army's Battlefield Distributed Simulation program. He holds a BS in Computer Engineering from the University of South Florida and a MS in Simulation Systems from the University of Central Florida.

REAL-TIME PDU FILTERING IN THE GATEWAYS OF WIDE AREA SIMULATION NETWORKS

M. A. Bassiouni and M. Chiu
Computer Science Department
University of Central Florida
Orlando, FL 32816

M. P. Garnsey
STRICOM
12350 Research Parkway
Orlando, FL 32826

INTRODUCTION

Today, there is a strong emphasis being placed on the development of efficient Distributed Interactive Simulation (DIS) systems[1]. DIS systems are composed of live, constructive, and virtual simulations interacting in a common synthetic or "virtual" battlefield from multiple geographically distributed locations via communications networks. Real-time data filtering and data compression are two complimentary techniques that can help improve the networking efficiency of DIS systems. The design of real-time compression for DIS protocol data units (PDUs) has been treated in a previous publication[2]. In this paper, we concentrate on the problem of real-time PDU filtering and present the results of a detailed performance evaluation study which we have recently conducted to assess the benefits of PDU filtering at the gateway level of wide area DIS networks.

PDU filtering refers to the process of analyzing the contents of communication messages in order to select only the data that is required to be received or transmitted by simulation entities. For example, if two simulated vehicles, say V_1 and V_2, are separated by a large distance in the simulated environment, then a state update message from vehicle V_1 would be irrelevant to (and would not therefore have to be delivered to) vehicle V_2. This example shows the most obvious method of filtering, namely, filtering based on distances in the simulated environment. Other factors (e.g., type of vehicles) can also affect the filtering process. For example, state update messages from a vehicle submersed in water could normally be ignored by vehicles on the ground. Filtering is used when the total traffic is large enough to overwhelm the small bandwidth of a local site or when the slow nodes in this site cannot handle the fast rate of message arrival. For example, if a high-speed FDDI backbone[3-5] is used to interconnect several 10 Mbits per second Ethernet simulation networks[6-7], filtering could then be used to reduce the size of the traffic flowing from the FDDI backbone to each individual Ethernet LAN. In large scale simulator-based training exercises, a simulated vehicle would normally need to receive information from only a small subset of the total simulated vehicles at any given time; state update messages from the rest of the vehicles would not be important and can be discarded.

PDU filtering at the WAN gateway level represents a level of abstraction of the Simulation Networking (SIMNET) program key design principle that all simulation entities be cognizant of each other. Basically, WAN gateway level PDU

filtering enables an efficient DIS system implementation principle that the cognizance of all simulation entities is not the responsibility of the underlying simulation entities of a LAN, but rather the responsibility of the WAN gateways. The WAN gateway must monitor and process all PDUs transmitted by other WAN gateways, and then relay only those PDUs of external entities that are within the "sphere of influence" of its underlying LAN entities to those entities.

AN APPROACH FOR DATA FILTERING

In this section, we shall give the high level details of an approach that can be used for implementing on-the-fly (i.e., real-time) filtering of state update messages. For the purpose of illustrating the basic ideas of the filtering scheme, we shall discuss methods relevant to simulators of ground vehicles and we shall use the distance separating these vehicles as the main criterion for filtering.

The filtering scheme uses a one-dimensional vector of distances for each simulated vehicle. The vector can be stored in the gateway of the LAN (cluster) where the simulator resides. Assuming that vehicles in the simulated environment are numbered 1 through n, the vector for the i^{th} simulator will be stored in the form

$D_i = (d_{i1} , d_{i2} , \ldots , d_{ii} = 0 , \ldots , d_{in})$

where d_{ij} is the distance (in the simulated environment) between vehicle V_i and vehicle V_j. For each vehicle, say vehicle V_i, we define a *"reachability region"* which specifies a neighborhood region such that the vehicles located within that region are tactically important to vehicle V_i (e.g., they are electronically visible to vehicle V_i). State update messages from vehicles outside this reachability region need not be delivered to vehicle V_i. For purposes of illustration, we shall assume that the simulated region is a flat terrain free from obstacles. In this case, the reachability region can be simply represented by a reachability radius R_i that gives the maximum distance from vehicle V_i at which another vehicle is reachable (visible). In addition to the distances vector D_i, a bit vector B_i is maintained for vehicle V_i and is defined by

$B_i = (b_{i1} , b_{i2} , \ldots , b_{ii} = 1 , \ldots , b_{in})$

where $b_{ij} = 1$ if $d_{ij} \leq s R_i$

$= 0$ otherwise

and s is a safety scale factor that suppresses the filtering of messages from vehicles that are outside the reachability region but which are close enough to its border. As shown in Figure 1, a safety ring of depth $(s-1)R_i$ is created to guard against any delay by the filtering mechanism in resuming the delivery of messages sent by a fast vehicle that suddenly entered the reachability region. Thus for example, if s is equal to 1.2, then vehicle V_i will start receiving messages from another vehicle even though that vehicle is at a distance 20% larger than the actual reachability radius.

B_i is a binary vector and is therefore more suitable than D_i for real-time filtering decisions. Upon receiving a state update message, say M_j, sent by vehicle V_j, the gateway will perform the following algorithm to update the vector B_i.

```
Update position of Vj based on Mj
for i = 1 to n and i ≠ j do
  if bij= 0 and dij ≤ sRi
   then bij = 1
   else if bij = 1 and dij > sRi
        then bij = 0; endif;
endfor
```

Because of the safety region, the above procedure does not represent a time critical computation; it can in fact be performed as a background job. Furthermore, even if the scale factor is set to 1 (i.e., no safety region), the above procedure can be easily implemented in real-

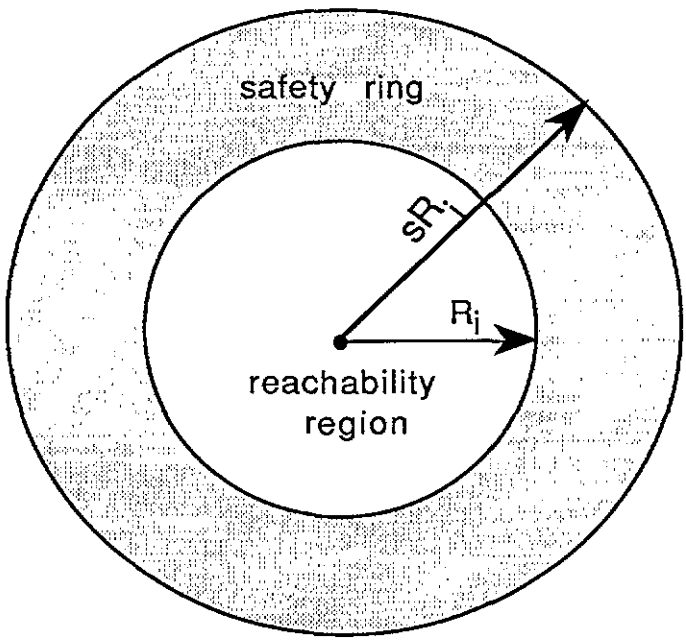

Figure 1. The reachability region

time using clever data structures or alternatively using parallel processing technology.

Using the above scheme, the filtering decision becomes an easy task. For example, to determine whether vehicle V_i needs to receive a message M_j sent by vehicle V_j, the following code is executed

if $b_{ij} = 1$ then send M_j to V_i
else discard M_j

IMPLEMENTATION OF DATA FILTERING

Filtering should be performed by network gateways at the transmission and reception of a message as well as during its routing in intermediate gateways. Filtering at transmission is the main process that could eliminate the majority of the unneeded messages. Filtering at reception performs a final check and could eliminate the unneeded messages that have not been detected during the transmission and routing phases. Notice that the gateway handles simulator messages in two different ways: 1) the gateway receives messages from nonlocal simulators (called external senders) and distributes them to the simulators on its local site, and 2) the gateway receives messages sent by the local simulators (called local senders) and transmits them over long-haul links to the simulators in other sites. The first case requires *filtering at reception* (i.e., filtering after receiving a message via long-haul links) and the second case requires *filtering at transmission* (i.e., filtering before transmitting a message onto long-haul links). We

shall start by discussing filtering at reception then proceed to examine filtering at transmission.

The receiving gateway would need to keep accurate information about the positions of the vehicles simulated by the local nodes connected to it. This can be done without much difficulty since the gateway receives every state update message transmitted by any node in its local site. Without loss of generality, let us assume that the total number of nodes (simulators) in all sites is n, and that the local site under our consideration contains the first m nodes, i.e., its nodes are numbered 1 through m. Using our filtering scheme, the gateway in this site maintains a collection of binary vectors equivalent to a binary matrix, called the filtering matrix B. In the case of filtering at reception, this matrix is defined as

$B = [b_{ij}] \quad 1 \le i \le m, \; m+1 \le j \le n$

where b_{ij} is a filtering flag that is set to 0 if messages from the external simulator j are not relevant (i.e., need not be delivered) to the local simulator i. As before, the safety scale factor is denoted by s and the reachability region of vehicle V_i is represented by a circle of radius R_i. Filter-at-transmission uses a logic similar to that used in the case of reception. The main idea can be briefly described as follows. If a local simulator sends a message, the gateway will perform filtering to transmit the message to only those external simulators that can be affected by it (or discard the message if it is not important to any external simulator).

PERFORMANCE RESULTS

A detailed simulation program written in Concurrent C was developed and used to evaluate the different design alternatives of filtering at transmission and filtering at reception. Several performance tests were conducted to assess the benefits of data filtering and compute the expected filtering rate in DIS networks. Fig. 2 gives the value of the overall filtering rate for various values of the reachability range and number of clusters (LANs). All the points shown in Fig. 2 use a total of 400 vehicles with a maximum vehicle's speed of 45 miles/hour. Fig. 3 shows the range of the filtering rate (low and high values) when 400 vehicles are distributed over different number of clusters (ranging from 4 to 25 clusters). An example plot showing the relationship between the reachability range and the overall filtering rate is shown in Fig. 4. The plot is for 80 vehicles distributed evenly among four clusters. Three cases of initial placement are tested in this experiment.

* Case A: vehicles in each cluster are initially grouped together and separated (in the simulated world) from other clusters.

* Case B: same as Case A, but one vehicle in each cluster is initially detached from its group and placed with some other cluster.

* Case C: All clusters initially overlap using random placement.

A similar plot for the relationship between the overall filtering rate and the number of clusters is given in Fig. 5. A reachability range of 50 miles is used in all the cases of this figure. In general, the filtering rate is more influenced by changes in the reachability range than by the number of clusters (using the same total number of vehicles).

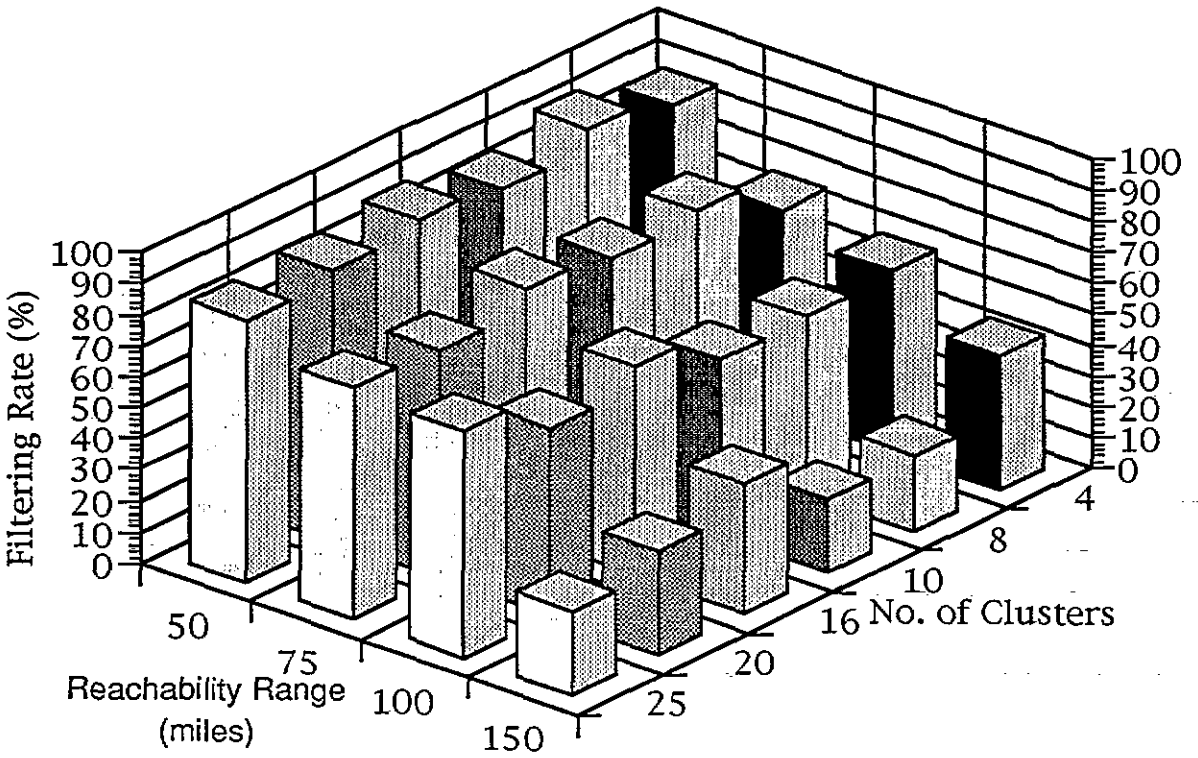

Fig. 2. Overall Filtering Rate For Various Values of Reachability Range and Number of Clusters

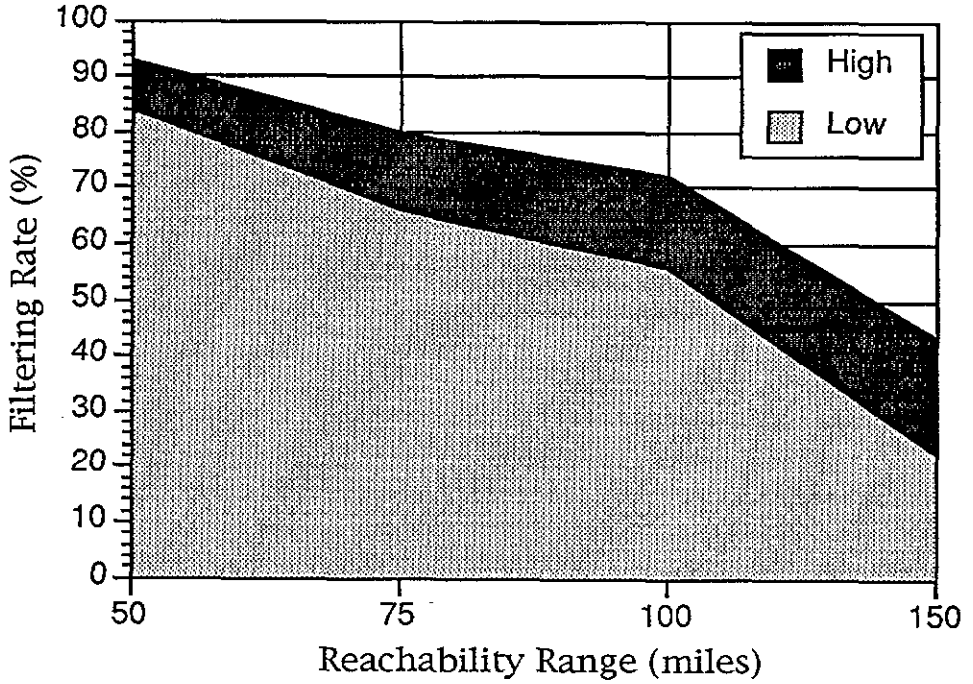

Fig. 3. Low and high values of filtering rate

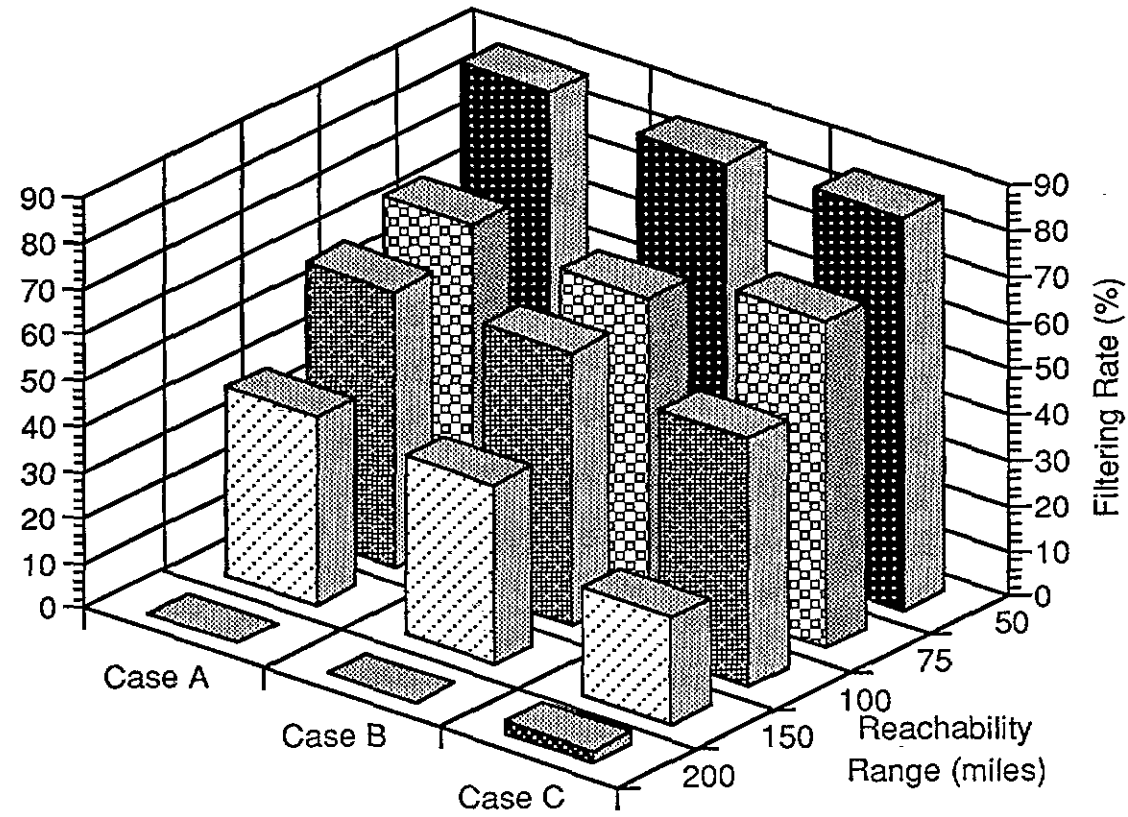

Fig. 4. Filtering Rate vs. Reachability Range

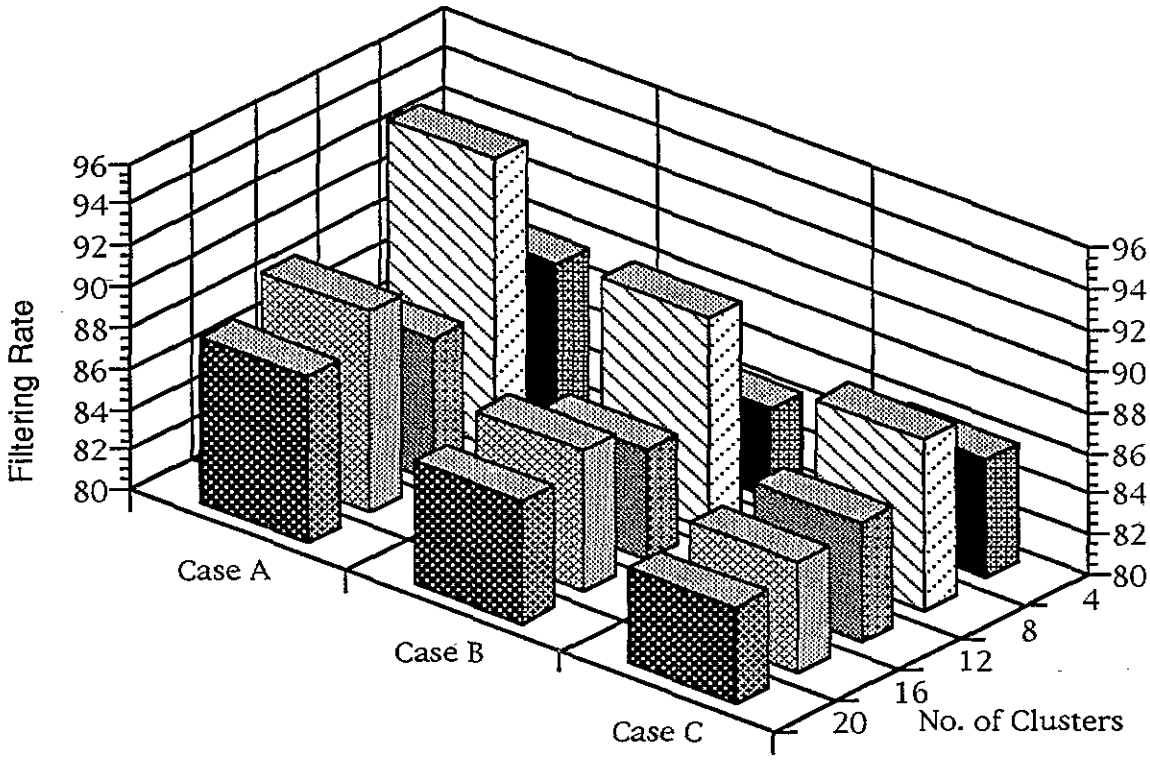

Fig. 5. Filtering rate vs. number of clusters

Fig. 6 shows the average, maximum, and minimum values of the overall filtering rate for the three cases discussed before. The total number of vehicles used in Fig. 6 is 80, the reachability range is 50 miles, and the number of clusters varies from 4 to 20.

DEAD-RECKONING AT THE GATEWAY LEVEL

If the filtering mechanism becomes very successful, the gateways will be deprived of receiving messages from some external simulators. This in turn will make the information (on external vehicles) maintained by each gateway less accurate and can render the filtering decisions incorrect.

A simple example will be used to illustrate this problem. Consider two vehicle simulators V_1 and V_2 located in two different DIS sites (LANs). The two sites communicate over long-haul links using the services of two gateways G1 and G2. Initially, the two vehicles are quite far from each other, i.e., each vehicle is outside the reachability region of the other vehicle.

Now assume that vehicle V_1 started moving towards vehicle V_2. Gateway G1 will execute the Filtering at Transmission strategy and will find that the state update messages emitted by V_1 need not be delivered to V_2. Gateway G1 will therefore refrain from sending these messages to G2. Thus this latter gateway continues to have the initial position of vehicle V_1. Now if vehicle V_2 moves towards V_1, gateway G2 may determine that the state update messages emitted by V_2 need not be delivered to V_1. G2 will therefore refrain from sending these messages to G1. The result is that G1 will have inaccurate information about the position of V_2. A situation can subsequently arise where the two vehicles V_1 and V_2 are near each other but each one of them is deprived of receiving the state update messages of the other.

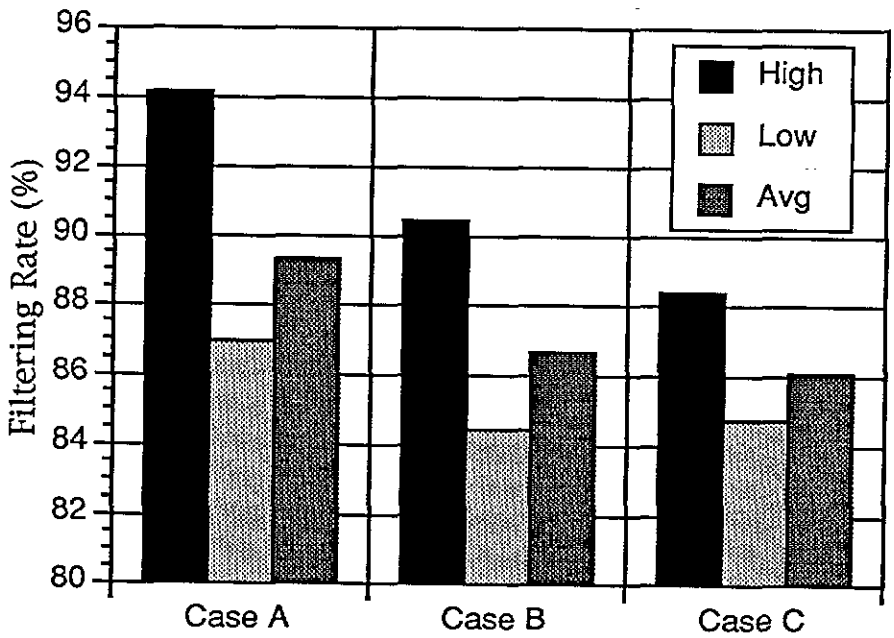

Fig. 6. Maximum, minimum, and average filtering rate

There are two ways to solve this problem. In the first method, each gateway would periodically suppress filtering for a short duration to ensure that the (simulated) positions of its local vehicles are recorded correctly in the other gateways. The second method is to provide the gateway with a **dead-reckoning algorithm** similar to that used by the vehicle simulators themselves. This approach is described next.

The concept of dead reckoning is used to reduce the number of state update messages that need to be transmitted by each simulator for the purpose of maintaining accurate state representation. Simply, each simulator has a high fidelity model which maintains accurate information (position, speed, velocity, etc.) about its own state. Each simulator also maintains a less accurate model, called the dead reckoning model, for each simulator (including itself) participating in the exercise. The dead reckoning model of a vehicle is periodically updated by extrapolating the information reported in the last state update message of that vehicle. Using first-order extrapolation, the anticipated position of a simulator is obtained by extrapolating its last reported position based on its last reported velocity as follows:

$$X(t + \tau) = X(t) + V_X(t)\,\tau$$
$$Y(t + \tau) = Y(t) + V_Y(t)\,\tau$$
$$Z(t + \tau) = Z(t) + V_Z(t)\,\tau$$

where $X(t)$, $Y(t)$, $Z(t)$ are the World Coordinates of the simulated vehicle at time t as reported in the last state update message, $V_X(t)$, $V_Y(t)$, $V_Z(t)$ are the x, y, z components of the velocity vector of the vehicle at time t, and $X(t + \tau)$, $Y(t + \tau)$, $Z(t + \tau)$ are the new coordinates predicted at τ units of time after the last state update message.

The prediction of the dead reckoning algorithm can be generally improved by resorting to higher order extrapolation equations. For example, the dead reckoning equations using second-order extrapolation are as follows

$$X(t + \tau) = X(t) + V_X(t)\,\tau + 0.5\,A_X(t)\,\tau^2$$
$$Y(t + \tau) = Y(t) + V_Y(t)\,\tau + 0.5\,A_Y(t)\,\tau^2$$
$$Z(t + \tau) = Z(t) + V_Z(t)\,\tau + 0.5\,A_Z(t)\,\tau^2$$

where $A_X(t)$, $A_Y(t)$, $A_Z(t)$ are the x, y, z components of the acceleration vector at time t. Whenever a state update message is received from a simulator, the information of that message is used to correct the extrapolated information of the dead reckoning model. Finally, when the state of a simulator actually changes, the simulator updates its own high fidelity model and compares it with the extrapolated information of its own dead reckoning model. If there is a large enough discrepancy between the two models, the simulator transmits a new state update message to all other simulators. The corresponding dead-reckoning approach in network gateways can now be described as follows:

1) Each gateway will maintain accurate information (position, speed, velocity, etc.) about each of the local simulators in its own site. This information (called the high fidelity model) should be reasonably accurate since the gateway receives every message transmitted by a local node.

2) Each gateway also maintains a less accurate model (called the dead reckoning model) for external simulators. The dead reckoning model is obtained by extrapolating the last reported location of each external vehicle based on its last reported velocity. Whenever a message is actually received from an external simulator, the information of that message is used to correct the extrapolated

information of the dead reckoning model.

3) Finally each gateway also keeps a dead reckoning model for its local simulators. When the gateway receives a message from a local simulator, it updates its high fidelity model and compares it with the extrapolated information of the dead reckoning model. If there is a large enough discrepancy between the positions of the local vehicle in the two models, the gateway transmits the message over the long-haul links.

Our tests so far have indicated that both methods discussed above are effective and give comparable results.

CONCLUSIONS

In this paper, we presented high level details of data filtering designs for DIS wide area networks. Our performance evaluation tests have shown that data filtering can provide a significant saving in the amount of traffic transmitted over the network. The paper presented various results showing the relationship between the overall filtering rate, number of clusters, and the reachability range. Gateway dead-reckoning methods to solve the problem of inaccurate state information at high filtering rates are presented.

ACKNOWLEDGMENT

This work has been supported by the Army's Simulation, Training, and Instrumentation Command (STRICOM) under contract N61339-92-C-0016.

REFERENCES

[1] LORAL System Company. Strawman DIS Architecture Document. Vol. I: Summary Description. ADST Program Office, March 1992.

[2] Bassiouni, M. and Mukherjee, A. "Data compression in real-time distributed systems" Proc. of IEEE Global Telecommunications Conference, 1990, pp. 967-971.

[3] ANSI Standard X3.148 "FDDI token-ring physical layer protocol (PHY)" 1988.

[4] ANSI Standard X3.139 "FDDI token-ring media access control (MAC)" American National Standard, 1987.

[5] Bassiouni, M. and Thompson, J. "Application of FDDI/XTP network protocols to distributed simulation" Proc. of the 12th I/ITS Conference, 1990, pp. 226-233.

[6] IEEE/ANSI Standard 8802/3 "Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specification" IEEE Computer Society Press, 1985.

[7] Bassiouni, M.; Georgiopoulos, M. and Thompson, J. "Real time simulation networking: network modeling and protocol alternatives" Proceedings of the 11th (I/ITSC) Conference, 1989, pp. 52-61.

[8] STRICOM *"Ditsributed Interactive Simulation", Standards Developement Draft 2.0*, September 1992.