Improving Performance Among Client-Server For Continuous Distributed Interactive Applications

Shahenaz  
Dept. of Computer Science & Engg.  
UCEK, JNTUK, Kakinada  
Andhra Pradesh, India  
Email: shahenaz.lbrces@gmail.com

Syed Shabana  
Dept. of Computer Science & Engg.  
UCEK, JNTUK, Kakinada  
Andhra Pradesh, India  
Email: shabana.612@gmail.com

ABSTRACT:
In today’s situation consumers mainly use the network as an interactive medium for multimedia entertainment and communication purpose. In the entertainment arena, new applications involve multiple users participating in a single interactive session, for example, online gaming. In distributed server architecture, the interactivity performance depends on not only client-to-server network latencies but also interserver network latencies. All of these factors are directly affected by how the clients are assigned to the servers. We propose a Latency Equalization (LEQ) service, which equalizes the perceived latency for all clients participating in an interactive network application.

We achieve equalized-latency paths by using a few routers in the network as hubs, and interactive application packets from different clients are redirected through these hubs to the servers. We formulate the hub placement problem, and provide the greedy algorithm and the optimal algorithm to solve this problem. Through extensive simulations, we show that our LEQ hub routing architecture significantly reduces delay difference in large network topologies. The LEQ architecture is incrementally deployable in today’s networks, requiring only a few nodes to be modified to perform the LEQ service.

KEYWORDS: Distributed Interactive Application, Client Assignment, Interactivity, Consistency, Equalized-Latency Path.

I. INTRODUCTION:  
Distributed interactive applications (DIAs), such as multiplayer online games and distributed interactive simulations, allow participants at different locations to interact with one another through networks. Thus, the interactivity of DIAs is important for participants to have enjoyable interaction experiences. Normally, interactivity is characterized by the duration from the time when a participant issues an operation to the time when the effect of the operation is presented to the same participant or other participants. We refer to this duration as the interaction time between participants. Network latency is known as a major barrier to provide good interactivity in DIAs. It cannot be eliminated from the interactions among participants and has a lower theoretical limit imposed by the speed of light. Here we focus on reducing network latency for improving interactivity in DIAs.

We propose a general network-based LEQ service for interactive applications by providing equalized-latency paths between the clients and servers in each application. A direct solution would be to buffer packets on routers. However, this requires large router buffers and complex router scheduling mechanisms to decide how long to buffer each packet, and will entail significant changes to today’s routers. Equalized-latency paths are achieved by setting up a few ‘hubs’ in the network, and packets from different clients are redirected through these hubs to the servers. Since the redirection is implemented through IP encapsulation, our hub routing mechanism can be deployed by leveraging today’s routing architecture. We show that our mechanism is easier to manage.

We formulate the hub placement problem, which decides where to place the hubs and the assignment of hubs to different clients to minimize delay difference. Therefore we propose a greedy algorithm and an optimal algorithm which works well in achieving equalized-latency paths. Through extensive simulation studies, we show that our LEQ architecture significantly reduces delay difference in both intra-domain and inter-domain settings.

II. MOTIVATION FOR NETWORK-BASED LATENCY EQUALIZATION SUPPORT:
To achieve equalized delay for interactive applications, previous approaches are implemented either at the client or server side without any network support. Client-side latency compensation techniques are based on hardware and software enhancements to speed up the processing of event updates and application rendering. These techniques cannot compensate for network-based delay differences among a group of clients. Buffering event update packets at the client side is
Due to some problems of client-side solutions, several delay compensation schemes are implemented at the server side. However, while introducing CPU and memory overhead on the server, they still do not completely meet the requirements of fairness and interactivity. Based on the limitations of the end-system based techniques, we conclude that it is difficult for end-hosts and servers to compensate for delay differences without network support. There is a pressing need to provide a network support for latency equalization as a general service to improve user experience for all interactive network applications. With network support for LEQ, the network delay measurement can be offloaded from the application server and performed more accurately.

Network-based solutions can achieve LEQ and also react faster to network congestion or failure. By providing LEQ service, ISP can gain larger revenue through significantly improving the user experience of interactive applications. Network support for LEQ is complementary to server-side delay compensation techniques. Since network-based LEQ service can reduce both delay and delay difference among participants of the interactive applications, the application servers can better fine-tune their performance.

II. LITERATURE SURVEY:
THE AUTHOR, Farzad Safaei (ET .AL), AIM IN [1], this article assess network and server communications needs to hold up real-time flows linked with networked activity applications. These comprise the state information flow to inform the status of the near environment and immersive communication flows such as voice, video, gesture, and haptics communication. The article shows that extent these applications to large physical spreads of member requires distribution of calculation to get together the latency constraints of the applications. The article gives detailed results on distributed server architectures for two of these real-time flows, state information and immersive voice communication. It also makes out a generic set of requirements for the causal network and server infrastructure to hold up these applications and suggest a new design, called switched overlay networks, for this purpose.

THE AUTHOR, Luis Diego Briceño (ET .AL) AIM IN [2], the surroundings careful in this research is a huge multiplayer online gaming (MMOG) environment. Each user controls an avatar (an image that stands for and is influenced by a user) in a practical world and interrelates with other users. A significant characteristic of MMOG is upholding a fair environment among users (i.e., not give an unjusbenefit to users with faster connections or additional powerful computers). The knowledge (either positive or negative) the user has with the MMOG environment is dependent relative on how speedily the game world takes action to the user’s actions. This study focuses on range the system based on demand, while keep an environment that guarantees fairness. Believe a situation where there is a main server (MS) that controls the state of the virtual world. If the act falls lower than adequate standards, the MS can off-load adding up to secondary servers (SSs). An SS is a user’s computer that is transformed into a server.

IV. PROBLEM DEFINITION:
The client assignment problem is assigning the clients to servers effectively to maximize the interactivity and reducing the latency of continuous DAs. It is formulated as the client assignment that minimizes the maximum length of interaction paths between all client pairs. Normally interactivity is considered by the duration from the time when a participant issues an operation to the time when the effect of the operation is presented to the same participant or other participants. We refer to this duration as the interaction time between participants. Network latency is known as a chief barrier to provide good interactivity in DIAs. It cannot be eradicated from the interactions surrounded by contributor and has a minor theoretical limit imposed by the speed of light. Our main focus is on performance improvement which can be improved by having equalized latency paths.

V. PROPOSED APPROACH:
Our proposed network-based LEQ service provides equalized-latency paths between the clients and servers by redirecting interactive application traffic from different clients along paths that minimize their delay difference. We achieve equalized-latency paths by using a few routers in the network as hubs, and interactive application packets from different clients are redirected through these hubs to the servers. Two algorithms namely greedy algorithm and the optimal algorithm are proposed. These algorithms results minimal latency time delays for all the clients.

The edge router first identifies interactive application packets by their port number and server IP addresses and then redirects the packets via one of its assigned hubs. Each edge router monitors the end-to-end delay to the server edge routers through each of its assigned hubs. When congestion is detected along a path, the client edge router can redirect packets to another assigned hub to get around the point of congestion. Hubs are just
conventional routers that are either in the core network or at the edge. Packet indirection through hubs is implemented through IP encapsulation without any changes to today’s routing architecture. Since we consider all edge routers as potential client edge routers, the selected hub nodes can be shared among different interactive applications.

VI. SYSTEM ARCHITECTURE:
LEQ routing architecture

VII. PROPOSED METHODOLOGY:
SERVER:
This serves as the centralized server to decide the hub selection and assignment. We choose an offline hub selection algorithm. Hubs are selected with the goal of minimizing the delay difference across all client edge routers.

HUB:
Some routers in the network are used as hubs and packets are redirected only through these hubs. Hubs are just conventional routers that are either in the core network or at the edge. Packet indirection through hubs is implemented through IP encapsulation without any changes to today’s routing architecture. Since we consider all edge routers as potential client edge routers, the selected hub nodes can be shared among different interactive applications. Using a small number of hubs in the network to redirect application packets, we equalize the delays for interactive applications.

EDGE ROUTER:
The edge router first identifies interactive application packets by their port number and server IP addresses and then redirects the packets via one of its assigned hubs. Each edge router monitors the end-to-end delay to the server edge routers through each of its assigned hubs. When congestion is detected along a path, the client edge router can redirect packets to another assigned hub to get around the point of congestion.

CLIENT:
Client interactivity is improved by using greedy and optimal algorithm. Service is provided from server through routers and also through hubs based on user request. We achieve equalized-latency paths by using a few routers in the network as hubs, and interactive application packets from different clients are redirected through these hubs to the servers.

GREEDY ALGORITHM:
1. The Router which is nearer to Client is treated as Edge-Router and we continue traversing.
2. As the maximum delay is known we neglect the paths where the D-max time delay is encountered while reaching the server.
3. The rest of the paths which reaches the server are minimal latency time delay paths and termed as Heuristic Hubs.
4. Thus the time delay is minimized, if there are more than one minimal latency paths we choose the path which has least time-delay.
5. This procedure is repeated along all the clients in the multiparty networks so as to minimize or equalize the time-delay.

Algorithm 1 Basic hub placement for min b
Step 1. Sort all the delays from client edge router $c_i$ to server $s_i$ through hub $h_j$ in increasing order, denote this array $A$
Step 2. For each $A[i]$, binary search to find the min delay difference
for each delay $A[i]$
$left = 0$, $right = D_{max} - A[i]$
while $left$ not equal $right$
$\delta = (left + right) / 2$
$L_k = greedycover(A[i], \delta, m, G, \{d(u, v)\})$
if $(|L_k| > M) left = \delta$ else $right = \delta$
Step 3. pick $L_1$ with smallest $(\delta, A[i])$ in terms of lexicographical order.

The pseudo-code of basic hub placement algorithm

OPTIMAL ALGORITHM:
1. Suppose that if the network is larger and the determination of D-max is complex we cannot use Greedy algorithm thus we need to implement another technique “Optimal Algorithm”.
2. This algorithm chooses the router as Edge-router which is nearer to the client, if there is more than one nearer router the router with least time delay is treated as edge-router
3. The traversing starts from this edge router as there are no interactive packets involved the timing delays are complex to calculate thus the traversing continuing by choosing all the paths to reach the server, and the path with least
time-delay is treated as minimal time-delay path
4. Thus the condition for this algorithm is \( m = M \)
   where \( m \) denotes number of hubs for an edge router and \( M \) denotes total number of hubs in
   the network.
5. Let \( B \) be a set of all candidate hub records where each record \( b \in B \) has three fields: hub ID \( h_id \), minimal delay \( min_d \) and maximum delay \( max_d \).
6. Denote the records sorted in increasing order of \( min_d \) and \( max_d \) by \( B_1 \) and \( B_2 \) respectively (Step 1).
7. For each candidate hub record \( b_1 \), the algorithm computes a candidate solution \( L_1 \) by
   adding in \( M \) hubs with smallest \( b_2 \), \( max_d \) whose \( b_2 \), \( min_d \) > \( b_1 \), \( min_d \) (Step 2).
8. The algorithm then picks the \( L \) with minimal \( \delta_t \) (Step 3).

\[ \text{Algorithm 2 Optimal algorithm for } m = M \]

Step 1. Let \( B_1, B_2 \) be the candidate hub records sorted by their \( min_d \) and \( max_d \) in increasing order.
Step 2. For each \( b_2 \in B_2 \)
\[ L_t = \{ b_1, h_id \} \]
\[ \delta_t = D_{max} \]
   for each \( b_2 \in B_2 \)
   \[ L_t = L_1 \cup \{ b_2, h_id \} \]
   if \( (b_2 \), \( min_d \) > \( b_1 \), \( min_d \) and \( |L| < M \) )
\[ \delta_1 = b_2 \), \( max_d \) - \( b_1 \), \( min_d \)
Step 3. Pick \( L_1 \) such that \( \delta_1 \) is the smallest.

The pseudo-code of algorithm for \( m = M \) (Optimal Algorithm).

VIII. RESULTS:

It demonstrates the sharing of the numbers of assignment modifications carry out in the simulation. We count up each effort made to alter the assigned server of a client concerned in a longest interaction path as one assignment modification, even if it is determined not to modify the assigned server of the client after reckoning. The algorithm executes less than 50 assignment modifications in 94 percent of the simulation runs. In fact, the normal number of assignment modifications performed is less than 18. Under K-centre and K-median placements, the number of assignment adjustments are about 55 and 43 on average. These results entail that only a small proportion of clients require adapting their servers in the implementation of Distributed-Modify Assignment.

IX. CONCLUSION:
The edge router first identifies interactive application packets by their port number and server IP addresses and then redirects the packets via one of its assigned hubs. Each edge router monitors the end-to-end delay to the server edge routers through each of its assigned hubs. When congestion is detected along a path, the client edge router can redirect packets to another assigned hub to get around the point of congestion. Hubs are just conventional routers that are either in the core network or at the edge. Packet indirection through hubs is implemented through IP encapsulation without any changes to today’s routing architecture. Since we consider all edge routers as potential client edge routers, the selected hub nodes can be shared among different interactive applications.

X. FUTURE WORK:
The basic hub placement problem can be extended to account for access delays. Low-overhead mechanism for fine-grain latency and loss measurement can also be incorporated.

XI. REFERENCES:


**Ms Shahenaz** is a student of University College of Engineering, JNTUK, Kakinada. Presently she is pursuing her M.Tech [Information Technology] from this college and she received her B.Tech from Lakireddy Bali Reddy College of Engineering, affiliated to JNTUK, Kakinada in year 2013.

**Ms Syed Shabana** is a student of University College of Engineering, JNTUK, Kakinada. Presently she is pursuing her M.Tech [Information Technology] from this college and she received her B.Tech from Lakireddy Bali Reddy College of Engineering, affiliated to JNTUK, Kakinada in year 2012.