NODE SELECTION FOR A FAULT-TOLERANT STREAMING SERVICE ON A PEER-TO-PEER NETWORK

Hyunjoo Kim\textsuperscript{1}, Sooyong Kang\textsuperscript{1} and Heon Y. Yeom\textsuperscript{1}

\textsuperscript{1}School of Computer Science and Engineering
Seoul National University
Seoul, 151-742, Korea
\{hjkim,yeom\}@dcslab.snu.ac.kr

\textsuperscript{1}Department of Computer Science Education
Hanyang University
Seoul, 133-791, Korea
sykang@hanyang.ac.kr

ABSTRACT
Peer-to-Peer (P2P) networks are attracting considerable research interest because of their scalability and high performance relative to cost. One of the important services on a P2P network is the streaming service. However, because each node in the P2P network is autonomous, it is difficult to provide a stable streaming service on the network. Therefore, for a stable streaming service on the P2P network, a fault-tolerant scheme must be provided. In this paper, we propose two new node selection schemes, Playback Node First (PNF) and Playback Node First with Prefetching (PNF-P), that can be used for a service migration-based fault-tolerant streaming service. The proposed schemes exploit the fact that the failure probability of a node currently being served is lower than that of a node not being served. Simulation results show that the proposed schemes outperform traditional node selection schemes.

1. INTRODUCTION
As the demand for streaming service on the Internet is increasing and the quantity of streaming contents is growing rapidly, it has become difficult to provide a stable streaming service using traditional client/server-based architecture. The new network service model, Peer-to-Peer (P2P) service, addresses this problem. The P2P service is a form of reciprocal network service where each node in the P2P network acts as both a client and a server. However, while the P2P service can address the problem of client/server service architecture, i.e., load concentration on the server, it is difficult to provide guaranteed quality of service (QoS) because each node in a P2P network is autonomous. In particular, because the streaming service has real-time characteristics, QoS degradation due to node failures is critical. Therefore, for streaming service on a P2P network, a fault-tolerant scheme that can cope with node/link failures must be developed. Previous works on fault tolerance for streaming service in P2P networks can be classified into: 1) data recovery using redundant data [1], 2) reallocation of the data sending rate from server nodes [2], 3) special encoding for graceful quality degradation [3], and 4) service migration [4].

Before a client node sends a service request to other nodes, it is necessary to select server nodes from many candidate nodes that have the desired object stored. Because the number of nodes participating in a P2P network is large and each node is not only autonomous but all may have different outbound bandwidths, the QoS depends mainly on the node selection scheme. Therefore it is necessary to devise a good node selection scheme for a streaming service on a P2P network.

2. PEER-TO-PEER SERVICE MODEL
In a P2P network each node acts as both a client and a server node. A client node receives data for playback from one or more server nodes. A server node can provide services to multiple client nodes within the predefined outbound bandwidth. Therefore, a node can receive services from multiple server nodes and can provide services to multiple client nodes, simultaneously. Assuming no node/link failures, the service-requesting node should 1) search: find nodes that have the required object, 2) node select: receive available outbound bandwidth information from those nodes and select a suitable number of server nodes that will actually provide the service, and 3) schedule: assign data segments for transmission to the selected nodes. Because the search is beyond the scope of this paper, we assume that the client node already knows all the nodes that have the required object. We also assume that media objects are encoded into constant bit rate data and are sequences of data segments of equal size.

Every node can act as both a client and a server for a P2P streaming service. When a service request arrives from a client node, the server node first advises the client node of the available outbound bandwidth. On receiving the bandwidth information, the client node informs the server node whether it is selected as a server node. If selected, the server node provides the service to the client, and if not, it sends state update information whenever the available outbound bandwidth changes to the client.

After selecting the initial server nodes, the client node requests data transmission and assigns segments to each selected server node, selects a new server node when a fault signal arrives from...
the buffer monitor, and sends a request for data transmission to the selected new server node. Therefore, the server node should have 1) a node selector that chooses server nodes from the many nodes that have the demanded object; 2) a segment handler that assigns data segments to each server node for transmission; 3) a service manager that requests data transmission from the server nodes and sends a service finish request to the node that should cease data transmission because of node/link failure or link state change; and 4) a state manager that receives state update information from nodes that are not currently selected as a server node but can be selected later if a server node fails, and transfers that information to the node selector.

There are two types of fault - failure and state change. Failure means that there can be no more service between server node and client node because of server node shutdown or intermediate link failure. State change means that the actual data transmission rate from the server node to the client node becomes lower than the expected rate because of server overload or network congestion. In this paper, we cope with these in a unified manner.

Before starting playback, the client node prefetches data segments in buffers, with each buffer dedicated to a specific server node. The user can determine the prefetching quantities by configuring the buffering time \( t_p \) in the player. The quantity of data prefetched in a buffer is determined as \( t_p \cdot r_x \), where \( r_x \) is the transmission rate of server \( i \). Therefore, different prefetching quantities for each buffer can be defined to match the transmission rate of each server node. Because at least one segment should be transmitted to each buffer before playback starts, the transmission rate of each server node should be no smaller than \( s_d \cdot t_p \), where \( s_d \) is the segment size. Therefore, the client node has to exclude those nodes that have smaller available outbound bandwidth than \( s_d \) from the possible server nodes.

After prefetching data, during the buffering time, the client node starts playback. The volume of data in a buffer does not change significantly unless a fault occurs. However, if a fault occurs, either no more data arrives at the corresponding buffer (failure) or the rate of arrival decreases (state change) while the consumption rate does not change. Therefore, the amount of data in the buffer decreases as time passes. Therefore, we can determine the occurrence of a fault by monitoring the volume of data in a buffer.

The client node starts to monitor each buffer when the playback begins. Let \( \delta \) be the predicted time length from a fault occurrence to when a newly chosen server node starts service. There should be sufficient data for playback during \( \delta \) in each buffer. Therefore, we can determine the fault detection threshold \( \theta_i \) for buffer \( i \) from the following equation:

\[
\theta_i = \frac{r_x}{s_d} \times \delta
\]

When a fault is detected, the node selector chooses one or more replacement server nodes (R-servers) that will replace the failed node. To prevent the necessity of segment reassignment to all servers, the aggregated service rate of R-servers is the same as that of a failed node. The segments previously assigned to the failed node will be reassigned to the R-servers.

3. SERVER SELECTION SCHEMES

After searching candidate nodes that have the demanded object, the client node should select server nodes that will collectively provide the service. We consider two previous schemes, Larger available outbound Bandwidth First(LBF) and Smaller available outbound Bandwidth First(SBF), and propose new server selection schemes, PNF and PNF-P.

- **Larger available outbound Bandwidth First(LBF):** In the LBF scheme, a node that can provide larger available outbound bandwidth is selected first. Therefore, the number of server nodes can be minimized and we can predict that the probability of a node/link failure will be small. However, the probability that a server node will fail, when using the LBF scheme, is the same as that when the server nodes are randomly selected. This means that the LBF scheme does not decrease the node failure probability. In addition, when a node failure occurs, the overhead to cope with the fault is large because the failed node (which has large available outbound bandwidth) must be replaced by multiple nodes (which may have smaller available outbound bandwidth). Therefore, the recovery time can be large.

- **Smaller available outbound Bandwidth First(SBF):** Using the SBF scheme, a node that can provide smaller available outbound bandwidth is selected first. Therefore, the number of total server nodes is maximized and this also maximizes the probability of a node/link failure. In addition, the SBF scheme cannot decrease the node failure probability itself. Therefore, the overall failure probability of SBF is larger than that of LBF. However, the overhead to recover from a failure is smaller than that of LBF because the outbound bandwidth of the failed node is generally smaller than that of other candidate nodes, which means that only one candidate node is sufficient to replace the failed node.

- **Playback Node First (PNF):** If we know whether a node is currently being served or not, and when a node is currently being served how much time is left before the service will be completed, then we can decrease the failure probability of a server node. Because the probability that a node currently being served will fail in the course of service is relatively small, it will decrease the overall node failure probability when we select the node that is currently being served first. The Playback Node First (PNF) scheme originated from this idea.

The PNF scheme is based on the LBF scheme. However, when there are nodes that are currently being served, they are selected first. Nodes that have recently started playback are more likely to be selected, which means that any node that has the longer remaining playback time is selected first. If there are nodes that have the demanded object and are playing another object or that do not have
Table 1: Simulation parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of nodes</td>
<td>1500</td>
</tr>
<tr>
<td>number of initial servers</td>
<td>20</td>
</tr>
<tr>
<td>outbound bandwidth</td>
<td>30’–100 Kbps</td>
</tr>
<tr>
<td>number of objects</td>
<td>20</td>
</tr>
<tr>
<td>object playback rate</td>
<td>300 Kbps</td>
</tr>
<tr>
<td>segment size</td>
<td>10 Kbits</td>
</tr>
<tr>
<td>buffering time</td>
<td>3600 seconds</td>
</tr>
<tr>
<td>predicted migration time (δ)</td>
<td>2 seconds</td>
</tr>
<tr>
<td>request arrival rate</td>
<td>exponential distr. with mean of 0.012</td>
</tr>
<tr>
<td>network congestion interval</td>
<td>10000 seconds</td>
</tr>
<tr>
<td>network congestion duration</td>
<td>50 seconds</td>
</tr>
<tr>
<td>normal state migration time</td>
<td>exponential distr. with mean of 0.1 sec.</td>
</tr>
<tr>
<td>congestion state migration time</td>
<td>6250’–200000 seconds</td>
</tr>
<tr>
<td>Mean Time Between Failures (node/link)</td>
<td>86400 seconds (24 hours)</td>
</tr>
<tr>
<td>simulation time</td>
<td></td>
</tr>
</tbody>
</table>

4. SIMULATION

In this section, we present and discuss various simulation results. The parameters used for the simulation are shown in Table 1.

The initial servers for a client are those nodes that have the demanded object. We fixed the number of initial server nodes at 20. Therefore, initially, the client should select server nodes from those 20 nodes and the remaining nodes become candidate nodes. The predefined outbound bandwidth to provide services to other nodes varies from 30 Kbps to 100 Kbps with 10 Kbps increments and they are uniformly assigned to all nodes in the P2P network. The predicted migration time is the predicted time from a fault occurrence to when a newly selected server node starts service. The value is used to detect node failure or link state change. The initial buffering time is five seconds, but this does not affect the performance of the system provided that it is larger than the predicted migration time, which was set as two seconds. We assumed that each node requests an average of one streaming service per day. Therefore, the average request arrival rate is \((1/86400) \times 1000 \sim 0.012\). When we considered the link state change, we assumed that each path from a server node to the client node is congested following an exponential distribution with mean interval 10000 seconds and the congested state lasts for 50 seconds. For normal state networks, the actual migration time follows an exponential distribution with a mean of 0.1 seconds, and for a congested network it follows an exponential distribution with a mean of one second. We limited the number of clients that are served concurrently in the P2P network to 50.

Figure 2 shows the simulation results. As the Mean Time Between Failures (MTBF) becomes larger, less failures occur. Figure 2-(a) shows the average number of server nodes for each service. As can be seen from the figure, SBF uses more server nodes than LBF because the SBF selects nodes that have the smallest available bandwidth first, whereas the LBF selects nodes with the largest available bandwidth first. In case of SBF, as the MTBF value becomes smaller, more failures occur, and the average number of server nodes is increased because there can be a server that does not use the whole available bandwidth and has bandwidth left over. For example, consider a failed server with a transmission rate of 70 Kbps and two server nodes, \(N_1\) and \(N_2\), which have data rates of 40 Kbps and 50 Kbps, respectively. Then, \(N_1\) serves with the 40 Kbps data rate and \(N_2\) is sufficient to serve with 30 Kbps data rate. Therefore, as the MTBF value decreases, the number of server nodes increases. However, when the MTBF value is too small, the number of available nodes becomes too small, which means that there are a small number of nodes with small available outbound bandwidth. Therefore, two or more server nodes with small transmission rate can be replaced with one node with large available outbound bandwidth when they fail, which decreases the average number of servers. On the contrary, because the number of nodes with large available bandwidth also decreases as the MTBF value decreases, the average available bandwidth of server nodes selected by LBF decreases, which increases the number of server nodes for a client using LBF. Because the PNF scheme is based on the LBF scheme, the number of server nodes is similar to that of the LBF scheme. However, because the PNF scheme disobeys the LBF rule when there are nodes currently being served, it uses slightly more server nodes than LBF.

Figure 2-(b) shows the average number of service migrations caused by a node failure or network congestion. When a server node fails, the service from the failed node should migrate to another server node selected for replacement. From the figure, the number of migrations increases as the MTBF decreases. Because the PNF scheme guarantees almost no failures during the playback time of the server nodes, the number of migrations is smaller than for the LBF and SBF schemes. For the PNF-P scheme, additional server nodes are used for prefetching. However, because a client can prefetch part of the demanded object in advance and use local...
data instead of receiving data from server nodes when the playback reaches the time when the remaining data are already stored in the local storage, the actual service time decreases. Therefore, the average number of node failures also decreases. In the figure, PNF-P1, PNF-P2, and PNF-P3 use one, two, and three server nodes for prefetching, respectively. As the number of server nodes for prefetching increases, the actual service time decreases and therefore the average number of node failures also decreases.

In Figure 2-(c) and (d) are the number of abnormal service terminations and the number of normally finished services, respectively. When a server node fails and there are insufficient candidate nodes to replace the failed node, the service cannot continue and terminates playback. This is called an abnormal service termination. As more failures occur, the number of candidate nodes decreases and the probability of abnormal service termination increases, because of the lack of suitable candidate nodes. Because the PNF scheme has a smaller number of node failures, as shown in Figure 2-(b), the number of abnormal service terminations is smaller than that for LBF or SBF. Correspondingly, the number of normally finished services is larger than that for LBF or SBF. Because the PNF-P scheme uses more server nodes than the original PNF scheme, the number of abnormal service terminations for PNF-P is slightly larger than that of the original PNF scheme, and, correspondingly, the number of normally finished services for PNF-P is slightly smaller than that of the original PNF scheme.

5. CONCLUSION

To provide a stable streaming service on a Peer-to-Peer network, a fault tolerance scheme must be devised to deal with node failures. One of the traditional fault tolerance schemes is a service migration from the failed node to a new node. When using a service migration-based fault-tolerant scheme, it is important to select relatively stable nodes as server nodes. In this paper, we proposed new node selection schemes for a service migration-based fault-tolerant streaming service on Peer-to-Peer networks. The proposed schemes, Playback Node First (PNF) and Playback Node First with Prefetching (PNF-P), are based on the fact that the failure probability of a node currently being served is lower than that of a node not being served. The proposed schemes decrease the average number of node failures and therefore increase the probability of providing a stable streaming service. From the simulation results, we found that the proposed schemes decrease the number of node failures and therefore provide a more stable streaming service on Peer-to-Peer networks.

6. REFERENCES