THE IMPLEMENTATION AND PERFORMANCE EVALUATION OF A DYNAMIC ADAPTIVE MULTICAST WWW SYSTEM

Jyh-Jean Wu, Chuan-Chi Hu
Department of Electronics Engineering
Chung Cheng Institute of Technology
Email:{g891302, g901313}@ccit.edu.tw

Chen-Kow Yuan, Ching-Chuan Chiang

Department of Computer Science
Chung Cheng Institute of Technology
Email:{yuan, ccchiang}@ccit.edu.tw

ABSTRACT

Internet provides users with ubiquitous information accesses and World Wide Webs (WWW) are the most popular applications in the internet. Enormous web accesses lead to a steep increase in the amount of traffic on the internet and the load on the server, thus increasing the transmission delay and decreasing the quality of service. In this paper we explore an adaptive web service scheme which uses multicast to reduce network bandwidth and server overhead, and thus improves the performance of web accesses. The adaptive web server is able to switch responses to clients' requests from TCP unicast to multicast and vise versa by monitoring the clients' requests. When the requests meet a threshold, the server switches to multicast the responses instead of multiple unicasts. It is very useful for hot web pages and audio/video applications.

Keyword

Multicast delivery, dynamic adaptive, WWW, FEC

1. INTRODUCTION

Internet provides users with ubiquitous information accesses and World Wide Webs (WWW) are the most popular applications in the internet. Enormous web accesses lead to a steep increase in the amount of traffic on the internet and the load on the server, thus increasing the transmission delay and decreasing the quality of service. The increased traffic which can be solved by adding the network bandwidth, and the increased load can be shared and balanced by distributing the requests to duplicate servers (server cluster). However, those solutions are uneconomical and cannot scale with the user population. In this paper we explore an adaptive multicast web scheme which uses multicasting for delivering hot web pages instead of multiple unicast transmissions of the same page to lots of clients. The adaptive method is capable of dynamic switching between unicast TCP transmission and multicast UDP delivery by monitoring the requests of the web pages. The benefits of multicast are network bandwidth reduction, sever load release, and well scalability [3]. The scalability issue is very important for the hot pages since it is independent with the population of clients. There are additional overheads for the multicast transmissions (such as join delay, switching latency, etc.) and not all web pages are suitable for multicasting. In this paper we use a dynamic adaptive scheme to switch to multicasting. By monitoring the requests during a time period, the system will adopt multicast when the number of requests reaches the threshold, otherwise using the traditional unicast. In addition to the requests, the switching metrics are also based on the properties of web pages. Audio/video and short lifetime data such as stocks, electrical news, etc. are more suitable for multicasting [4, 5]. Traditional web accesses are unicast TCP connections and the multicast transmissions are using UDP. To recover the data loss and enhance the reliability, we adopt FEC(Forward Error Correction Code) scheme and TCP retransmissions. In this paper we implement the adaptive multicast web system on Windows 98/NT systems and analyze various performance issues via an experimental network infrastructure.

Section 2 compares the unicast and multicast web transmissions. Section 3 explains our adaptive multicast web system. Section 4 describes the development environments. Section 5 discusses the performance evaluation. Section 6 concludes the paper.

2. UNICAST WWW AND MULTICAST WWW

2.1 HTTP Protocol

The Hypertext Transfer Protocol (HTTP) [6, 7] is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, object-oriented protocol which can be used for many tasks, such as name servers and distributed object management systems, through extension of its request methods. A feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred. HTTP has been in use by the World-Wide Web global information initiative since 1990. HTTP builds on the discipline of reference provided by the Uniform resource Identifier (URI) , as a location (URL), for indicating the resource to which a method is to be applied. Messages are passed in a format similar to that used by Internet mail as defined by the Multipurpose Internet Mail Extensions (MIME).

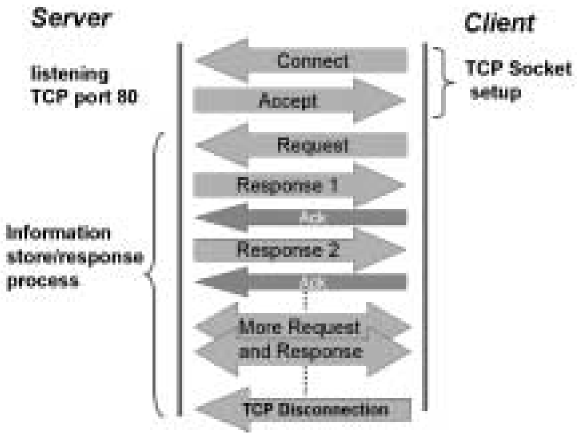
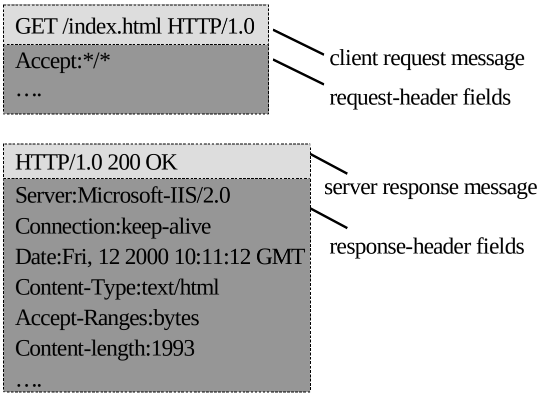


Figure 1 : unicast HTTP server/client connection

2.2 Unicast HTTP server and client

HTTP is similar to other TCP/IP applications such as SMTP and FTP. The client sends requests to the server and the server replies responses to the client. The HTTP client is called the web browser and the HTTP server is called the web server. Usually the web server uses TCP port 80 for connection.

Traditional web connection between a browser and the web server is as follows: (1) the browser initiates the TCP connection and sends the request to the web server, (2) the web server replies the requested objects, and (3) the browser terminates the TCP connection. Figure 1 shows the process of traditional HTTP server/client connection and Figure 2 shows the contents of the GET command.



```
GET /index.html HTTP/1.0
Accept:*/*
….
```

client request message

request-header fields

```
HTTP/1.0 200 OK
Server:Microsoft-IIS/2.0
Connection:keep-alive
Date:Fri, 12 2000 10:11:12 GMT
Content-Type:text/html
Accept-Ranges:bytes
Content-length:1993
….
```

server response message

response-header fields

Figure 2 : request/response of GET command

2.3 Multicast HTTP server and client

The multicast HTTP server/client scheme proposed in this paper uses a 2-phase TCP/UDP connection. In the first phase the TCP connection is used between the server and the client. If the client is capable of multicast functions, it sends the request with the keyword “MULTICAST” in the header. The server replies the multicast information (such as multicast address, TTL, port, etc.) in the response’s header if it also has multicast capability. After receiving the server’s response, the client terminates the TCP connection and sends the IGMP message to join the multicast group which was informed in the header of previous response and waits for the objects which are delivered by multicasting. The detail process will be explained in the next section. Figure 3 shows the connection steps of multicast HTTP server/client and figure 4 shows the headers of request/response messages.

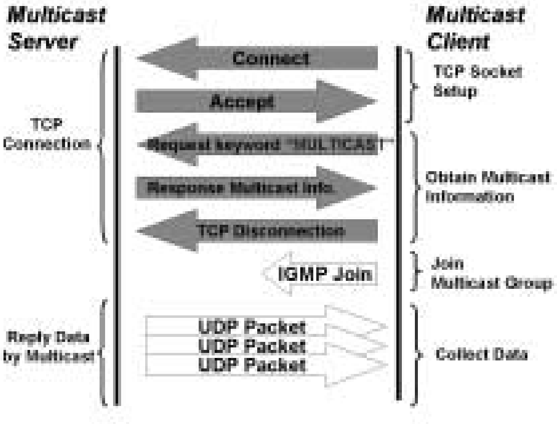
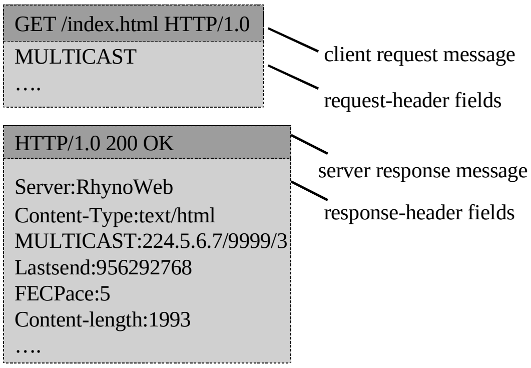


Figure 3 : multicast HTTP server/client connection



```
GET /index.html HTTP/1.0
MULTICAST
….
```

client request message

request-header fields

```
HTTP/1.0 200 OK
Server:RhynoWeb
Content-Type:text/html
MULTICAST:224.5.6.7/9999/3
Lastsend:956292768
FECPace:5
Content-length:1993
….
```

server response message

response-header fields

Figure 4 : request/response of GET command

3. ADAPTIVE MULTICAST WWW

3.1 Mapping of URL to Multicast Address

When the web server receives the multicast requests and decides to switch to transmit the web page by multicasting, it must tell the clients the multicast group to join. For IP multicast [1, 2, 11, 12, 13, 15], the multicast group is defined by the multicast address (class D IP) and the UPD port. How to efficiently select the multicast group to avoid using the same multicast group as other web servers is a challenging problem. If there are more than one server using the same multicast group, the clients may receive the wrong objects. Since the range of multicast addresses is smaller than unicast addresses, the mapping is many to one. The method in [3] used DNS services to query the multicast servers. However, the assignment and management of multicast addresses via DNS is not practical. The scheme in this paper uses a simple hierarchical mapping. As in figure 5, the multicast address format is partitioned into three parts. The first part is the WWW address allocation which represents the different locations such as countries. The second part is the server address which stands for the multicast servers. The third part is the page address which means different hop web pages. For an URL http://host/path/mediafile.xxx, "host" is mapped to the server address part and "host/path" is mapped to the page address part, and "host/path/mediafile.xxx" is mapped to the port. The mapping is to sum the ASCII value of each character and module 256 or 65536 to get the corresponding addresses. To further control the multicast traffic and avoid the conflict, TTL is used to limit the scope and reduce the conflict probability. An ID number is also specified in the response's header to allow the clients to distinguish multicast data from different servers.

0	1	2	3	4~15	16~23	24~31
1	1	1	0	WWW Address Allocation	Server Address	Page Address

Figure 5 : Multicast address allocation format

3.2 Client request processing

The procedure of the client connecting to the multicast web server is as following steps:

(1) If the client is capable of multicast functions, it sends the request to the web server with "MULTICAST" keyword in the header and waits for the response from the server. If the response indicates that the server is going to use multicast to transmit the objects, the client terminates the current TCP connection and goes to step (2). If either the client or the server cannot support multicast, the HTTP connection uses traditional unicast transmissions.

(2) The client joins the multicast group by sending an IGMP join message to the router. The multicast group information is retrieved from the header of server's response. According to the multicast protocol in the router, the multicast tree is constructed and multicast data for this group will be forwarded. The client is listening to the multicast address and port and waiting for data

(3) If the client does not receive any reply after the timeout, it resends the request but without including the "MULTICAST" keyword in the header.

(4) Since the multicast data is transmitted via UDP protocol, the client must examine the integrity and correction of the received data. The detail of data error detection and correction will be described in section 3.4. Figure 6 shows the diagram of multicast client processing.

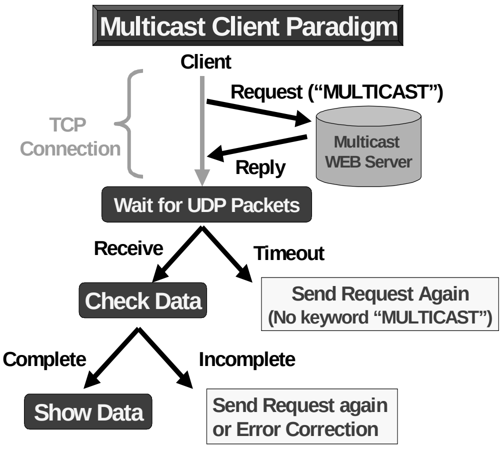


Figure 6 : Multicast client processing

3.3 Multicast server request processing

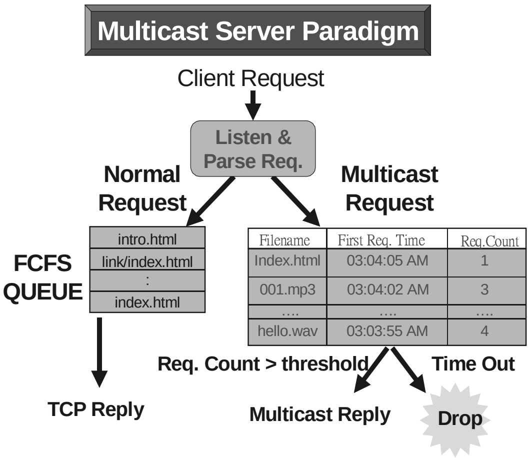
(1) When the server receives a request, it goes to step (2) if the request contains "MULTICAST". Otherwise it goes to step (3).

(2) Multicast requests for the hop page will be recorded. If the number of requests is greater than the threshold within a specified time window, the server replies the responses, which include the multicast address and port (using the mapping rule described in section 3.1), to all clients which are held during the time window. Then, the requested object is sent to the multicast group. If the number of requests is not reaching the threshold after the timeout of the time window, all requests are dropped. The clients will resend the requests without "MULTICAST" in the header if they do not receive the responses after timeout.

(3) The server replies the requested object using traditional TCP connection. Figure 7 shows the diagram of multicast server processing.

3.4 Error Detection in Client

Multicast transmission is using UDP protocol which is not reliable. In order to detect the integrity, the packet header includes IDnumber, SequenceNumber, and TotalNumber fields, which are used to detect the packet loss. The IDnumber is referring to the multicast group. As explained in section 3.1, the multicast group may be not unique. When a client receives a packet, it will check the IDnumber to verify the membership. If the received IDnumber is different from the one in the server's response, the packet is coming from other server which uses the same multicast address and port.



Filename	First Req. Time	Req.Count
Index.html	03:04:05 AM	1
001.mp3	03:04:02 AM	3
....	....	....
hello.wav	03:03:55 AM	4

Figure 7 : Multicast server processing

3.5 Request Scheduling at the Server

In the multicast server, there are two queues for the coming requests. One is for the unicast requests and the other is for the multicast requests. Queue for the unicast requests is served in the FCFS order. When the server receives a multicast request, the request is marked with current timer and inserted into the multicast queue. If the number of requests in the multicast queue is greater than the threshold value, the server replies the responses to all the requests to notify the multicast group information and the queue is empty. A queued request is removed if its timer is expired. The interval of time window greatly affects the delay of transmissions and the load of the server, thus becoming the performance tradeoff.

3.6 Reliable Multicast using FEC

One of the major problems of multicasting is "reliability". Traditional HTTP uses TCP connections while multicast adopts UDP connections. It is a challenging problem to reliably deliver data to multiple receivers. There are lots of researches focusing on reliable multicast [14,17]. In this paper we refer to [6, 8, 9] and use FEC and TCP retransmissions to recover the packet loss of UDP transmissions. FEC (Forward Error Correction Code) appends the code into the original data to enable receivers to detect or correct the errors. This scheme avoids the retransmission overhead of packet loss but increases the computation load of servers (senders) and clients (receivers). Two major coding schemes of FEC are Tornado codes and Reed Solomon Erasure (RSE). In our implementation we adopt RSE which transmits n packets (k data packets and n-k coded packets) and receivers are able to recover the loss if any k packets are received correctly. To simply our first version, we encode the coded packet by using XOR operations among k data packets and then transmit the k+1 packets to clients. The client is able to recover one packet loss as long as it receives any k packets. Figure 8 shows the diagram of FEC scheme.

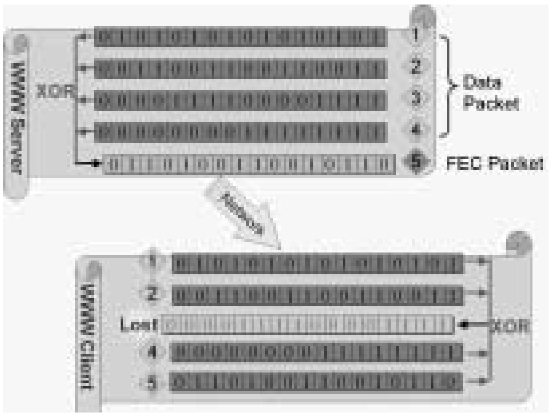
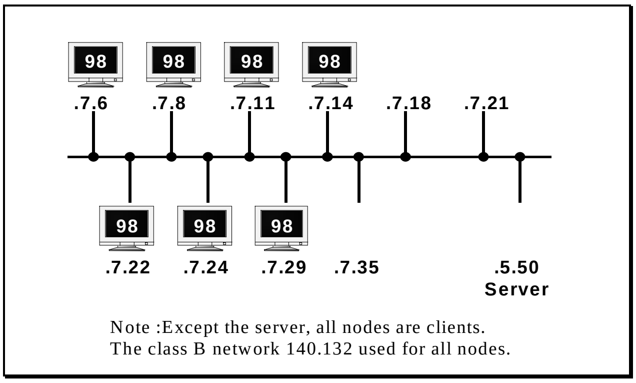


Figure 8 : FEC scheme

4. SYSTEM DEVELOPMENT ENVIRONMENT

The implementation of adaptive multicast web system is developed on Microsoft Windows 98/NT systems. There are two major programs: server and client. The server program provides web multicasting and the client provides the web browsing. All programs are coded with Borland C++ Builder 4.0 Professional version. The socket functions are provided by Microsoft Winsock 1.1/2.0[10]. To improve the performance and enhance the stability, advanced techniques, such as threads and asynchronous I/O, are used.



Note :Except the server, all nodes are clients.
The class B network 140.132 used for all nodes.

Figure 9: Network Infrastructure

5. PERFORMANCE EVALUATION

5.1 Network Infrastructure

The network infrastructure of our experiment is shown in figure 9. There are one server on the subnet 140.132.5.0 and 10 clients on the subnet 140.132.7.0. The bandwidth of each NIC (network interface card) is 10 Mbps. Each client randomly sends the requests to the server in the interval of 1 to 10 seconds. If the client does not received the response from the server after the max. waiting time (20 sec), it resends the request without “MULTICAST” keyword to use unicast connection. When the server receives the requests, it put the requests into the multicast queue. The queue is checked every 3 seconds to see if the number of queued requests is greater than the threshold value. The queued requests are dropped after timeout (15 sec). Table 1 lists the system parameters. Three performance metrics (table 2) are evaluated : (1) the average delay from the request to the reply for the client, (2) the probability of the client receiving multicast replies (that is how many requests are replied with multicast responses), and (3) the ratio of total data transmission using multicast to total data transmission using unicast (the ratio is 1 if no multicast delivery is experienced). The experimental results for various clients and multicast threshold are shown in figure 10, 11, 12.

Table 1 : System Parameters

System Parameters		
Server	Max. Holding Time	15 seconds
	Request Checking Interval	3 seconds
Client	Random Request Interval	10 seconds
	Max. Waiting Time	20 seconds
	Total Requests	~200
Object files	5 Small files (Avg. size = 10 KB)	
	5 large files (Avg. size = 60 KB)	

Table 2: performance Evaluation Metrics

Performance Evaluations	
PA	Average delay from the request to the reply for the client. (seconds)
PB	Probability of multicast reply
PC	Data Transmission with Multicast / Data Transmission without multicast

5.2 Discussion

To understand the performance gains of switching from unicast to multicast, we measure the metrics (PA, PB, PC) for various number of clients (2, 4, 6, 8, 10). Three switching threshold values are analyzed, namely 2 requests, 3 requests, and 4 requests. Figure 10 shows the results of PA : the average delay of retrieving objects computed from the client’s request to the reception of the objects. Two types of files are measured: large files and small files. Both cases show the average delay is decreased when the number of requests are increased. That means that multicasting reduces the overhead of duplicate transmissions and thus improves the efficiency of object retrieving. Figure 11 shows the probability of receiving “MULTICAST” responses when the clients issue the requests. The results show that the more clients, the higher the probability is. For hot web pages, the large population of clients greatly overload the server’s load which can be seemed as the threshold value, thus increasing the probability of getting multicast responses and decreasing the average delay. Figure 12 shows the ratio of MD/UD. MD represents the total data traffic when using adaptive web multicasting, and UD is the total data traffic when using traditional web server. The ratio indicates the bandwidth saving when using multicast. Without multicast, the ratio is defined as 1 (no bandwidth saving). With multicast, the bandwidth saving is increased with the number of clients. For example, when the number of clients is 10 in the small files case, multicasting traffic is reduced to 60% of unicasting traffic.

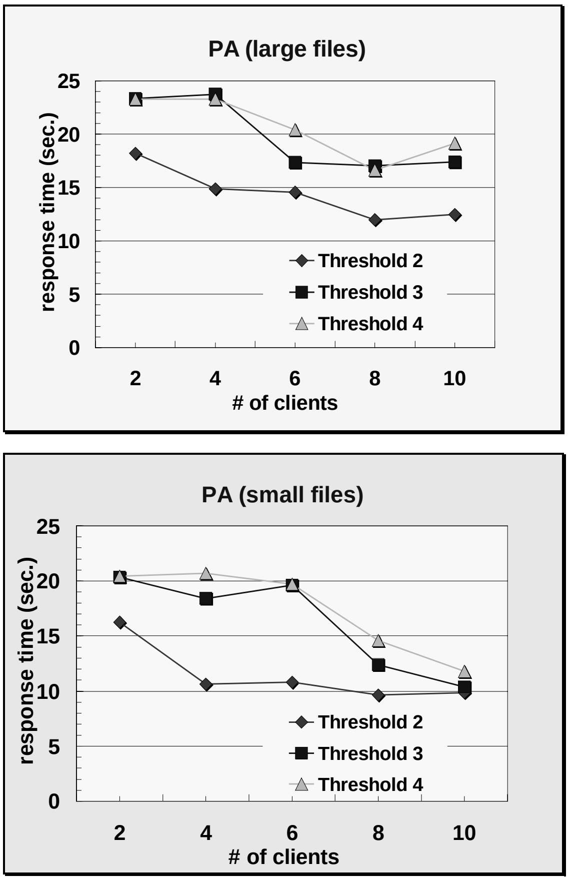


Figure 10 : Evaluation of metric PA of large files (up) and small files (down)

The experimental results show that the adaptive web multicast reduces the bandwidth requirements and thus releases the load of server when the requests are considerable. The response delay latency is reduced when the client populations are increased. Objects with small files are more suitable for multicast than large file applications. The threshold for switching from unicast to multicast services is a performance tradeoff and should be related with the server's connection capability. More elaborate experiments will be studied in the future, including various system parameters, client populations covering campus area, TANET/I2, and Internet.

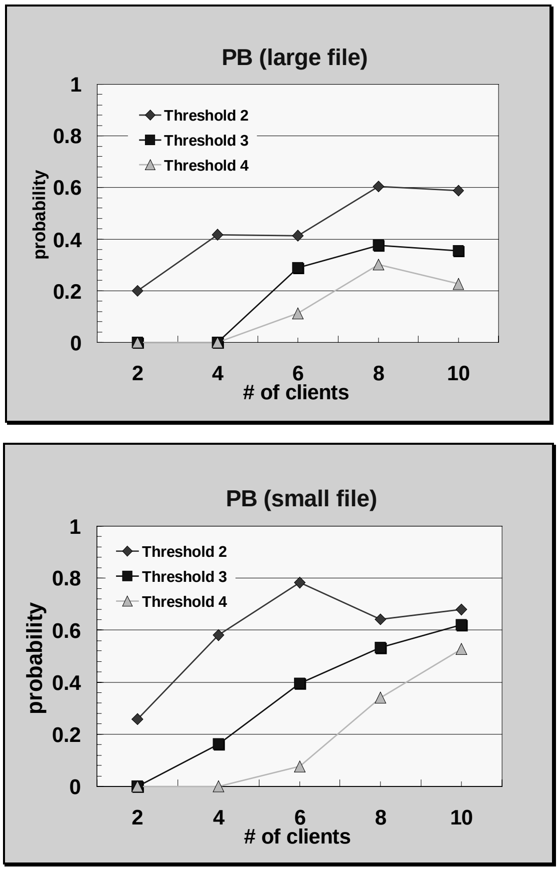


Figure 11 : Evaluation of metric PB of large files (up) and small files (down)

6.CONCLUSIONS AND FUTURE WORK

In this paper we develop and implement a adaptive multicast web server/browser system in Windows 98/NT. The performance results collected from the experimental environments show that multicast provides an efficient delivery for the hop web pages by reducing the network bandwidth requirement and the web server's load. Future work will extend to the multicast proxy servers and the multicast web caching. Another issue to be explored is the mapping of URLs to multicast addresses. Multicast services are more and more important in the future. All backbone network devices (routers, switches, etc.) will be capable of multicast and new IP protocol (Ipv6) will provide the default multicast protocol and adequate IP addresses, thus making multicast services feasible.

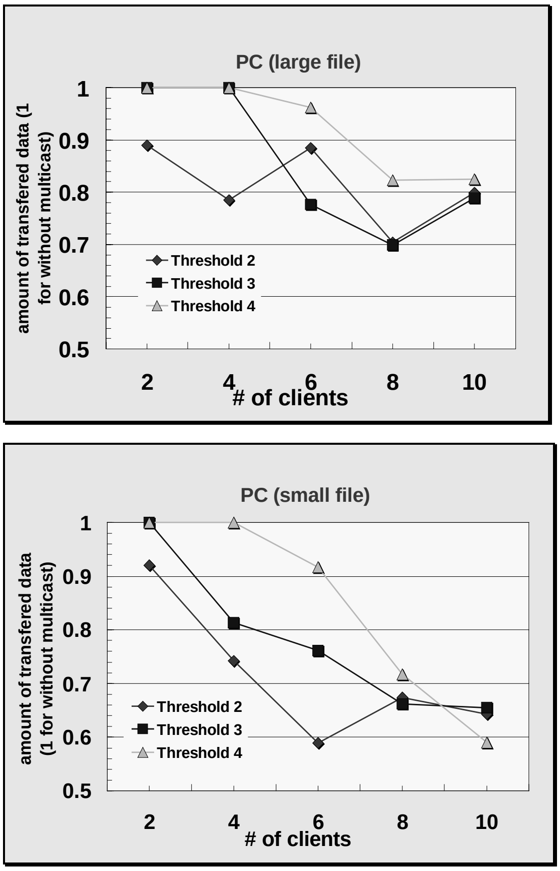


Figure 12 : Evaluation of metric PC of large files (up) and small files (down)

7.REFERENCES

[1] H. Eriksson, "MBONE: The Multicast Backbone", Communication of the ACM, vol. 37, no. 8, Aug. 1994, pp. 54-60.

[2] Frequently Asked Questions (FAQ) on the Multicast Backbone (MBONE), http://www.cs.columbia.edu/~hgs/internet/mbone-faq.html

[3] Russell J. Clark and Mostafa H. Ammar. "Providing Scalable Web Services Using Multicast", Proceedings of 2nd International Workshop on Services in Distributed and Networked Environments (SDNE 95), Jun. 1995.

[4] Kevin C. Almeroth, Mostafa H. Ammar, Zongming Fei, "Scalable Delivery of Web Pages Using Cyclic Best-Effort Multicast", Proceedings of IEEE INFOCOM '98, the Conference on Computer Communications.

[5] Pablo Rodriguez, Ernst W. Biersack, "Continuous multicast push of Web documents over the Internet", IEEE Network, vol.12, no.2, March/April 1998

[6] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1" , RFC 2068, January 1997.

[7] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee "Hypertext Transfer Protocol -- HTTP/1.1." , http://src.doc.ic.ac.uk/computing/internet/rfc/rfc2616.txt

[8] C. Perkins, O. Hodson, “Options for Repair of Streaming Media”(IETF RFC2354), http://src.doc.ic.ac.uk/computing/internet/rfc/rfc2354.txt

[9] M. Luby, L. Vicisano, L. Rizzo, J. Gemmell, J. Crowcroft, B. Lueckenhoff, “Reliable Multicast Transport Building Block: Forward Error Correction Codes”, http://search.ietf.org/internet-drafts/draft-ietf-rmt-bb-fec-00.txt

[10] Stardust.com, http://www.winsock.com/

[11] Steven Edward Deering, "Host Extensions for IP Multicasting", RFC 1112, August 1989.

[12] Steven Deering and Deborah Estrin and Dino Farinacci and Van Jacobson and Ahmed Helmy and Liming Wei, “Protocol Independent Multicast Version 2, Dense Mode Specification", Internet Draft, March 1997.

[13] Steven Edward Deering, “"Multicast Routing in a Datagram Internetwork", PhD Dissertation, Stanford University, December 1991.

[14] S. Floyd and V. Jacobson and S. McCanne and Ching-Gung Liu and Lixia Zhang, "A reliable multicast framework for light-weight sessions and application level framing", ACM SIGCOMM, pages 342—356, 1995.

[15] L. Wei and D. Estrin, "Multicast routing in dense and sparse modes: simulation study of tradeoffs and dynamics", Proceedings of the 1995 International Conference on Computer Communications and Networks (ICCCN'95), pages 20—23, 1995.

[16] Tony Ballardie and Paul Francis and Jon Crowcroft, "Core Based Trees (CBT) An Architecture for Scalable Inter-Domain Multicast Routing", ACM SIGCOMM, pages 85—95, 1993.

[17] Elena Pagani and Gian Paolo Rossi, “Reliable Broadcast in Mobile Multihop Packets Networks.” In ACM MOBICOM, pp. 34-42, 1997.