CHAPTER 1

INTRODUCTION

1.1 Motivation

As the Internet continues to grow in size and popularity, web traffic and network bottlenecks are major issues in the network world. The continued increase in demand for objects on the Internet causes severe overloading in many sites and network links. Factors affecting the performance of web are heterogeneous network connectivity, real-world distances and congestion due to unexpected demand. Many users have no patience in waiting more than few seconds for downloading a web page [1]. Web traffic reduction techniques are necessary for accessing the web sites efficiently with the facility of existing network. Although the Internet communication capacity increases 60% per year, the demand for bandwidth is likely to be more than the supply [2].

A significant amount of research effort has been elaborated upon investigating solutions to improve the response time. Researchers want to improve the web performance by increasing bandwidth using better communication medium or utilizing existing infrastructure more efficiently using software technologies. Since it is expensive to increase network infrastructure and bandwidth capacity, most users prefer to use some software technologies for improving the performance of the web. Some kind of solution is to be undertaken for the problems caused by rapid growth of the web, otherwise Internet would become too congested and its entire appeal would eventually be lost. Frequently accessed documents can be stored at client side for reducing the delay in accessing the
documents and the amount of information to be transmitted through the Internet. The present work broadly aims to improve web performance by reducing web traffic/latency. Simplistic view of the web, existing techniques available for web traffic/latency reduction, general architecture of web caching and pre-fetching techniques are discussed in the subsequent section.

1.2 Simplistic View of the Web

World Wide Web (WWW) is an architectural framework for accessing linked documents spread out all over the Internet. A web site is a collection of files that are managed by a web server. A simple architecture of the web is shown in Figure 1.1. Web consists of a set of servers, clients and the infrastructures that connects them. The user runs a web client known as browser and the user requests are given through the browser to the web servers. Web servers satisfy user requests by providing web pages to the clients. Hypertext Transfer Protocol (HTTP) specifies the interaction between web clients, servers and intermediaries. In order to retrieve a particular web page, the client gives the corresponding Uniform Resource Locator (URL) request through the browser and attempts to communicate over the Internet to the origin web server.

The client establishes a connection with the server by making a request. In order to connect to the server, the client needs its Internet Protocol (IP) address. It queries the Domain Name Servers (DNS) to translate the domain name to its IP address. After getting IP address, the client establishes a connection to the server. The web servers accept the URL requests from client and transmit the response to it. The time to retrieve a resource when a new connection is required can be
approximated by two round-trip times (RTT) plus the time to transmit the response [1] as shown in Figure 1.2.

**Figure 1.1 Simple web Architecture**

**Figure 1.2 HTTP Transfer Timing Cost for a Connection**

(Redrawn from [1])
1.3 Techniques for Improving Web Performance

As web traffic increases, improving the performance of web has become a challenging issue. Content simplification, compression, web server pushing, load balancing, bandwidth management, intelligent routers, object packaging, traffic dispersion, optimization of Hypertext Markup Language (HTML) codes, web caching and pre-fetching are some of the techniques available for improving the performance of the web by reducing web traffic/latency [3]. Latency is the elapsed time between sending the request of an object and receiving the response of that object.

In content simplification technique, web designers use common sense to reduce page complexity or special tools to optimize image coding [4]. But some data such as medical images, broadcast quality videos and executable software cannot be simplified without loss of meaning. Redundant bits within a single transfer are reduced using compression techniques. In web server pushing technique, web server pushes documents to some place near the client site. When the client accesses a document, it will go to a site that is nearer to the client [5]. This approach reduces the latency in accessing the documents. In this technique, it is difficult for a web content provider to know the proper place to push documents. In browser initiated server pushing (BIP), upon receiving a HTTP request, the server actively pushes embedded contents if the permission is given by the client [5].

In load balancing approach, the jobs of overloaded servers are shared by under loaded servers. Load balancing systems monitor the health of the available
servers and make decisions to route the traffic for optimizing the performance and availability. This ensures that the users will be connected to the most available server, providing excellent and predictable quality of service to the end user [6, 7]. Network bottleneck such as congestion can be avoided by managing the bandwidth properly [8, 9]. If infinite bandwidth is available, there is no need for bandwidth management. It is obviously very unrealistic to implement infinite bandwidth in any organization. Intelligent routers can route the traffic efficiently by reducing the network bottlenecks such as congestion. Intelligent routers can dynamically route the web traffic and provide reliable service even if some routes are failed.

The server side overhead can be reduced if number of requested files is reduced. In object packaging, multiple files such as HTML text and image files will be packed in an object package file for efficient transmission. At the receiving side, a web browser should unpack the files [10]. In traffic dispersion technique, burst nature of the web traffic is converted into sub bursts for reducing congestion [11]. The consequences of bloated HTML codes unnecessarily increase storage costs, transmission costs, download times and browser rendering times [12]. Optimization of HTML generated by WYSIWYG programs can be done for reducing the size of the web page to be transmitted.

1.4 Web Caching and Pre-fetching

Cache stores cacheable responses for handling subsequent requests. In web caching, copies of remote data are kept locally to reduce latency of repeatedly accessed data. If a new request that is same as a stored request arrives, then cache
sends the stored response rather than accessing the resource from the web server [13, 14]. Web caching is one of the most effective approaches to reduce bandwidth consumption, latency reduction and alleviating network bottleneck such as congestion [15, 16]. Pre-fetching is an active caching scheme which reduces latency by finding the anticipated sites and fetching it before actually required [17]. But it induces a little burden on the proxy server and the network. It may be difficult to predict accesses very far into the future, since client requests often depend on recent history [18].

The general architecture for web caching and pre-fetching [19] is shown in Figure 1.3. In this architecture, clients are connected to the web servers through proxy servers or cache servers. Proxy servers are used as intermediary for forwarding the user requests to the origin server and sending the response back to the client. Caching, pre-fetching, filtering and firewall techniques can be included in proxy servers. Since cache is included in proxy servers, it is also used as cache servers. The clients are connected to the web servers via two-tier or three tier architectures. In the two-tier architecture, the proxy server is not present and the client is connected to the server via a fast connection. In homogeneous three-tier architecture, proxy server is connected via a fast connection to both the server and the client. The middle tier is the proxy server, which might be used to provide security using a firewall, reduce latency by proxy caching and filter contents. In the heterogeneous three-tier architecture the proxy server is connected to the client via a slow connection. In these architectures, clients have browser cache and in three tier architecture, proxy servers have local cache.
Liu et al. shown that a single stand alone proxy cache does not always reduce response time [20]. Collaboration of proxy caches or cache servers is needed for improving response time. Web caching can occur at many locations including the...
client, server or proxy server. Caches of proxy servers throughout the world exhibit different access patterns [21]. The request stream exhibits temporal, spatial and geographical localities [22, 23, 24]. Repeated access to the same object within short time periods are called temporal locality. Access to an object causes provision to access some other objects are called spatial locality. Geographical locality implies that objects accessed by a client are likely to be accessed again by nearby clients. Web caching techniques exploits temporal or geographical locality and web pre-fetching technique utilizes the spatial locality of web objects. The principle of sharing suggests that if one client accesses a web document, other clients are likely to access that document. The principle of association states that access to a web document increases the probability of access to the documents linked from that document [25]. While web performance can be improved using caching, a key problem with caching is consistency for making the cached data up to date as origin server [26]. HTTP 1.1 defines headers for maintaining cache consistency [27]. Many browsers ignore the document expiry and serve the stale documents as current [28]. Due to the heterogeneity of the user requests and the limitation of maximum size of the cache, efficient cache replacement policies are needed for making space for placing a new object in cache. There are many cache replacement policies available in web caching and many factors are affecting their performance [29, 30].

Pre-fetching technique increase web traffic and consume more bandwidth for fetching the anticipated sites. The main cost factor of any pre-fetching system is the amount of fetched data which are never used [31]. In order to reduce the latency, pre-fetching can be used in a controlled manner without increase of much network overload. Both caching and pre-fetching techniques have not
attained the satisfactory results due to dynamic documents, CPU overhead, administrative overhead and memory spending for caching/pre-fetching. Caching the data increases storage costs, but it reduces transmission costs, delay and error experienced by the average user [32].

Even if several methods are available for web traffic/latency reduction, no single scheme is adequate enough for accessing the web sites instantly without delay. Many studies show that the web caching has the maximum limit of cache hit ratio of 50% [33, 34]. Shi et al. says that pre-fetching can improve the hit ratio to 60% or even more than 80% [35]. Experimental results show that pre-fetching increase cache hit ratio to 30-75% [36]. Caching reduces latency up to 28% [37]. Combined caching and pre-fetching proxy could reduce latency as much as 60% [38]. If pre-fetching decision is left to the user, latency is reduced up to 81% in a homogeneous environment and 63% in a heterogeneous environment [19].

1.5 Objectives

The top level aim of thesis was to evolve methods to better the information access over computer networks, in the context of caching and pre-fetching. The specific objectives were as follows.

1. To evolve better methods to reduce the amount of information to be transmitted through Internet.
2. To evolve better methods for efficient utilization of the existing bandwidth of current network.
3. To evolve better methods to avoid network bottleneck such as congestion.
4. To evolve better methods to increase cache hit ratio
5. To evolve better methods to reduce latency

6. To evolve better methods to permit accessing of desired cacheable web pages even if the connection to the web server is lost.

1.6 Dissertation Outline

This thesis is organized in seven chapters as follows.

Chapter 1: Introduction

This chapter gives the background information about the web traffic/latency reduction techniques. The objectives of the present work and organization of the thesis are discussed in this chapter.

Chapter 2: Web Caching: Trends and Techniques

Chapter 2 presents basic concepts of web caching and techniques related to web caching. It also focuses on the web cache communication, hypertext transfer protocol support for web caching, caching protocols, existing caching system architectures, replacement strategies, consistency mechanisms, design issues of web caches and limitations of web caching. This chapter is included because the foundation of the present work is based on web caching.

Chapter 3: Web Pre-fetching Techniques

Chapter 3 contains an overview of web pre-fetching techniques. The different types of pre-fetching techniques, existing pre-fetching techniques and the recent surveys conducted in pre-fetching are covered in this chapter. This chapter sets the stage for the original contribution of this dissertation by identifying the integration of web caching and pre-fetching.
Chapter 4: Browser Aware Clustered Cache Sharing Architecture

Chapter 4 contains an overview of clustered caching. A new browser aware clustered cache sharing architecture and its benefits are also reported in this chapter. In this architecture, proxy servers are grouped into clusters and a coordinator is elected in each cluster using election algorithm in order to survive from failures.

Chapter 5: Dynamic Web Pre-fetching Technique

Chapter 5 reports a new pre-fetching technique by integrating caching and pre-fetching. Dynamic pre-fetching technique is developed for maximum utilization of the existing capacity of the network bandwidth. This technique takes advantage of the idle time of the network to pre-fetch the anticipated sites. The benefits and working of this dynamic pre-fetching model are demonstrated in this chapter in a meaningful and applicable way.

Chapter 6: Adaptive Web Traffic/Latency Reduction

Chapter 6 embodies the main original contribution of the present research in web traffic/latency reduction. Adaptive traffic/latency reduction technique combines the existing traffic/latency reduction techniques such as web caching, pre-fetching, compression, optimization of HTML codes, object packaging, traffic dispersion. Adaptive traffic/latency reduction algorithms for client side and web server side are reported in this chapter.

Chapter 7: Conclusion

Chapter 7 summarizes the findings of the research and discusses the scope for future work.