Performance Enhancement of Scheduling Algorithms in Web Server Clusters using Improved Dynamic Load Balancing Policies

Hemant Kumar Mehta
mehtahk@yahoo.com

Priyesh Kanungo
priyeshkanungo@hotmail.com

Manohar Chandwani
chandwanim1@rediffmail.com

School of Computer Science, Devi Ahilya University, Indore, India.

ABSTRACT
Processing requirements of internet servers are increasing exponentially. This problem is aggravated with the growing complexity of web based applications and services. It is becoming difficult to manage the website traffic using a single powerful server. To improve the scalability, it is necessary to use web server cluster. However, in a server cluster, processing workload on different servers may vary drastically. Therefore, it is necessary to use dynamic load balancing (DLB) for evenly distributing this bursty workload. With the higher rate of increase in network bandwidth, DLB has now become more relevant. Although, DLB provides a cost effective solution for improving the performance of scheduling algorithms in a server cluster, the existing DLB techniques are not capable of meeting the challenges presented by the rapidly increasing processing requirements.

In this paper, we suggest some improvements in present DLB techniques to meet the extra ordinary situations faced by present day web applications. These suggestions are: improving the load balancing strategies to incorporate various essential features, load balancing strategies in grid computing, load balancing strategies for middleware and backend systems, improved dynamic content handling techniques. All the suggested modifications are currently under development.

KEYWORDS
Distributed Systems, Server Cluster, Dynamic Load Balancing, Grid Computing, Dynamic Content Handling

INTRODUCTION
Web servers are the means of interoperating between different applications running on variety of platforms, operating systems and programming languages. Most of the commercial application servers support web services. Web services are presenting enormous opportunities as well as a number of challenges by fundamentally changing the way of doing business and recasting the vendor customer relationship. More and more businesses are deploying network solutions being used by increasing number of people. Although in recent years, both communication speed and server capacity have improved, web applications are no more used for simple communication and browsing for getting static information. The World Wide Web has become complex medium of conducting personal and commercial transaction that require dynamic computation and secure communication with multiple servers through middleware and application software. With the phenomenal growth of IP traffic due to market expansion, server sites are overwhelmed with processing load. Even smaller companies are establishing internet and intranet presence to survive. In some special mega events like Olympics, the web site hit rate is over 1 million per minute. This is also true for the rapidly growing e-commerce applications such as banking, online shopping and B2C websites like eBay. Due to diverse load levels and sudden increase in the number user requests, servers become overloaded. In order to provide better response to the users in this bursty load situation, it is necessary to process different users’ requests at multiple locations in a transparent manner.

There are two ways for server sites to manage increased traffic, either deploy a more powerful server or add additional servers to a cluster of replicated servers without disrupting service. A group of servers, called a server cluster or a server farm, is used for servicing users’ requests as it is not possible to process the requests using a single server. In a distributed computing system (DCS), server farms are generally used for cluster computing or high traffic website hosting. They provide a cost effective solution that can be used in place of mainframe computers or scale-up servers. Server farm enables transparent growth as physical servers can be added without externally visible network changes.

With the increase in heterogeneous client devices and network bandwidth, the use of techniques for improving performance of web server system has become necessary. For the proper distribution of user requests, a distributed system requires middleware software between the client and the server. As shown in figure 1, this middleware receives user requests and
sends them to a specific server in the cluster according to some distribution logic and is called load balancer. Generally, load balancing enables effective allotment of a bottleneck resource to improve the overall performance of the system. Load balancing (LB) can be mainly classified as static, dynamic and adaptive. In case of static load balancing, the decision regarding the allotment of a request to a resource is known well in advance. In case of dynamic load balancing (DLB), the request allotment decisions are taken at run time based on the current state of the system. In adaptive load balancing (ALB), the parameters and policies can change depending on the run time information about the previous decisions, their results and the current state of the system. The overall performance of the DCS system depends on how the load managed at bottleneck device from the web servers to middleware, middleware to databases, database to operating systems and operating system to router.

Various types of distributed systems have evolved over a period of time. Some of them are general purpose and the others support only specific applications. The server farming, clustering and grid computing are examples of general purpose distributed systems while peer-to-peer computing supports only file sharing between various heterogeneous systems over the large networks or internet. Load balancing strategies must be evaluated for the various aspects of DCS. Our objective is to suggest enhancements in existing load balancing services and techniques being used for network and web servers.

LITERATURE REVIEW

DLB in network and web server is an important research issue owing to development of rapidly growing distributed application. Research community has given a considerable attention towards developing efficient technique for server load balancing. DLB in networks and web servers is a challenging research interest and provides an opportunity to the researchers to come with newer ideas. The area has attracted various researchers from variety of perspectives and application during past ten years as revealed from the work that has been reported in the literature. The goal of this literature survey is to provide an overview and future direction of research in the area of DLB in networks and web servers. There is a large body of literature in server load balancing. We, however, limit our coverage to the following topics:

(i) Improving the load balancing strategies
(ii) Load balancing strategies in grid computing
(iii) Load balancing strategies for middleware and backend systems
(iv) Improved dynamic content handling techniques

In this section, we will present the summary of research efforts made primarily in the past decade in the stated areas. The related state of the art is also presented.

(i) Improving the load balancing strategies: As the demands of resource intensive distributed applications have grown, the need for improved overall throughput and scalability has also increased. A cost-effective way to address requirements of these applications is to use dynamic or adaptive load balancing techniques. These load balancing techniques distribute client workload equitably among various servers in order obtain the best possible response time. In the following paragraphs, we will describe several advanced load balancing features that address the inability of present load balancing services to satisfy the demanding optimization and quality of service (QoS) requirements raised by complex distributed systems. Othman et al. have suggested a number of essential features that may be supported by DLB and ALB techniques for middleware. These features includes server-side transparency, stateful replica, different load monitoring granularity levels, federated load balancing architecture, fault tolerant load balancing, extensible load balancing strategy support and dynamic replica activation. In their subsequent work, the features are implemented in a middleware developed for the CORBA [8, 9]. These features may also be used for load balancing strategies for web servers and distributed databases.

(ii) Load balancing strategies for grid computing: Grids are attractive platforms for deploying parallel and distributed applications at large scale and/or for high performance. Application scheduling is a key issue for achieving these goals that is the decision process by which application components are allocated to available resources for optimizing some metric of performance. Scheduling applications onto grid platforms poses new challenges. Distinguishing feature of grid platforms is the large number of individual resources, with current systems contain thousands of individual devices and future systems may incorporate more than tens of thousands of devices. Such a large numbers of resources raise some scalability issues like resource discovery, resource monitoring etc. The solution for these problems is discussed in the following paragraphs.

Several works have been reported in literature recently for applying DLB in grid computing [12, 13, 14, 15]. The load balancing algorithms were developed with the assumption that the homogeneous sites are linked with similar networks. But in grids use of these algorithms is not practically possible due to heterogeneity, scalability and adaptability of resources in grids. A layered algorithm is proposed by Yaguobi and Slimani to achieve dynamic load balancing in a grid. This algorithm is independent of the physical architecture of grid that supports heterogeneity and scalability. A generic load balancing model has been suggested, which is based on incremental tree. The load balancing strategy is also hierarchical and has three levels, Intra-site load balancing, Intra-cluster load balancing and Intra-grid load balancing. Authors have implemented the algorithm and a simulator is developed to test its performance. The results show that the proposed model can lead to a better load balancing between computing elements of a grid without high overhead. A significant improvement in mean response time was observed and the communication cost between clusters is also reduced. The model was tested on newly developed simulator [12]. In order to validate practicality of the model, it
can be experimented on the real grid environments using some grid application.

Zhang et al. have compared the performance of several algorithms that represents two major approaches to scheduling in three different grid environments [13]. These two approaches are the list based and the level-based. Results show how the list-based algorithms can be applied successfully to the grid environment and how different factors in a grid computing environment affect the performance of the scheduling algorithms. The work is recently extended by further. It shows that if we use effective aggregate computing power (EACP) in the selection phase of a two-level algorithm, scheduling of the resulting virtual grid with a standard algorithm can produce improved results as compared to the standard version of the algorithm. In subsequent work, Zhang et al. have presented a decoupled mechanism that leverages the concept of a virtual grid to schedule workflow applications onto large-scale grid environments. This approach improves scalability when compared to traditional scheduling approaches as schedules can be computed dramatically faster [14, 15].

(iii) Load balancing strategies for middleware and backend systems: The distributed database comes into existence allowing users to retrieve information from different locations. However, there exist the cases where some database servers are heavily loaded while others may be lightly loaded or even no load which affects overall performance of the database system. Therefore, it is necessary to balance the workload among database servers to provide users with reasonable response time and throughput rate.

Middleware is increasingly used for applications with quality of service (QoS) requirements and scalability. Several efforts have been described in the literature for improving the DLB and ALB strategies.

Othman et al. have made several contributions towards adaptive middleware load balancing techniques. Authors have highlighted deficiencies in presently used load balancing techniques such unnecessary overhead or unable to adapt dynamically changing load conditions. They have also proposed Cygnus which is an adaptive load balancing service. Their findings show that adaptive middleware load balancing is a feasible solution for improving the scalability of distributed applications. They emphasize on developing load balancing strategies for CORBA middleware [9]. We feel that a generalized strategy should be designed that can be implemented on any middleware.

(iv) Improved dynamic content handling techniques: Performance and management of web servers has been an active area of research in recent years. Various proposed approaches to improve the performance of web servers are effective for web sites that have predominantly static web contents. However, all of these techniques are limited in terms of handling dynamic requests. The current generation web service used in the e-commerce environment suffers from several serious problems. The high proportion of dynamic content results in the poor response time for accessing these sites. A number of approaches have been proposed earlier for improving performance of web servers using dynamic content handling and caching. These approaches are being described in following paragraphs.

Candan et al. presents an architectural framework CachePortal system for enabling dynamic content caching for database-driven e-commerce sites. More specifically, they develop techniques for intelligently invalidating dynamically generated webpages in the caches thus enabling caching of web pages generated based on database contents. This is achieved by using a bi-layered architecture, the proposed architecture consists of two independent components, a sniffer, which collects information about user requests and an invalidator, which removes cached pages that are affected by updates to the underlying data [2, 3].

Examination of several dynamic pages on the web revealed that in most of the cases, only a part or a few portion of the pages are dynamic in nature. Other portions of these pages constitute static images or text. However, for every access of the dynamic page, the entire page gets constructed and delivered to the clients. Thus the static characteristics of these pages are unexploited. Mohapatra et al. have developed a framework, called WebGraph, with a motivation to exploit this nature of the dynamic pages. WebGraph uses a graphical representation of web pages to serve dynamic pages very efficiently. In addition, the framework is also enriched such that it can be used for other important attributes such as overload control, QoS assurances, caching efficiency, personalized services and load balancing in web servers. In addition to the performance benefits viz. overload control and QoS support, WebGraph also facilitates personalized and value-added services like easy management and publishing of web contents. The results justify these claims [7].

The above mentioned dynamic content caching schemes should be exhaustively analyzed to find their limits. Instead of caching the complete responses, it is easy to envision scenarios where caching database query results, specific fragments of responses, or compiled script-caching would be valuable option.

A detailed implementation study on behavior of web servers serving the static requests under the transient overload has been suggested by Biersack et al. and Schroeder and Balter. These papers suggest kernel level modifications in the web servers to change the policy from standard FAIR (processor sharing) policy to shortest remaining processing time (SRPT). Results show that significant performance improvement can be achieved by serving just the small size request first, without much affecting the performance of larger jobs. The work is further investigated and it has been concluded that the policy reduces the mean response time. A similar policy, called least-attend-service (LAS) that can be used if job size is not known in advance or it is difficult to estimate the job size [1]. Similar policies should be explored for dynamic content websites. Also, the effect of some parameter like dynamic content caching and content awareness is not covered in the study.
SCOPe OF SERVER LOAD BALANCING

DCS have a number of features like resource sharing, improved response time, performance, scalability, heterogeneity etc. In our research work we are going to address important issues like resource utilization and improvement in response time of processes in a distributed computing environment and emphasizing the need of DLB to solve the problem in a cost effective manner. Our investigations may help in solving the problem of server load balancing by modifying the existing server load balancing policies. Modifications in DLB schemes proposed in this paper will be useful in effective utilization of resources and improving response time in following areas in a DCS:

(i) Load-balancing clusters: This system consists of a load balancing server that manages the load among the collection of the replicated servers forming the load-balancing (LB) cluster. The primary focus of the load-balancing clusters is to improve performance of the system.

(ii) High-performance computing clusters: High-performance computing (HPC) clusters also increase the performance of the system but they split a computational task among different nodes in a cluster. Sometimes, communication among the nodes in the cluster may be required to synchronize computations among these nodes.

(iii) Grid computing: A computational grid (or simply Grid) consists of set of resources distributed over wide area networks (WAN) to support large and complex applications. Grid provides easy and pervasive access to computing utilities and data in the manner similar to electrical power Grid.

(iv) High-availability clusters (fail-over clusters): Distributed system is supposed to be more reliable owing to the multiplicity of the resources. In order to achieve reliability, distributed system must have some mechanism for fault handling. The tradeoff between HA and LB clusters can be created to improve the performance while maintaining the reliability.

ENHANCEMENTS IN DLB

Load balancing is the focus of research in distributed computing from late 80’s to till date. In the literature, we can find several works in this direction of improving the load balancing services, developing new strategies for load balancing, applying the load balancing techniques in cluster computing, grid computing, dynamic content handling, caching techniques load balancing for various software for example middleware like CORBA, various databases etc. In order to efficiently process a request it must be allocated to appropriate server. Despite of all the efforts, there remains a number of unresolved issues. In this paper, we have identified issues in and proposing solutions in load balancing services and their essential features, application of load balancing strategies in grid computing and at software level applying load balancing in various middleware, backend database systems etc. Websites serving the dynamic contents generates the entire page for each request. Improvements in dynamic content handling and caching techniques will results in better performance. We are also proposing possible strategies to solve these problems.

(i) Improving the load balancing strategies: The existing load balancing strategies must be modified to incorporate various features like server-side transparency, stateful replicas, different load monitoring granularity levels, federated load balancing architectures, fault tolerance, extensible load balancing strategy support, run-time control of replica life times. Server side transparency refers to replication transparency in handling client request. Stateful replicas maintain the state of client between subsequent requests. Load monitoring granularity refers to the interval for collecting the load balancing information and performing load balancing. In the federated load balancing architecture, distributed load balancers are used to provide more reliability and scalability. The distributed applications must be fault tolerant. Further study on improvements in adaptive and dynamic load balancing strategies is required. If the load on the system changes significantly, new replicas may have to be added to the cluster [8]. We have planned to implement these features for load balancing strategies used in web servers and distributed databases.

(ii) Enhancement of existing load balancing strategies for grid computing: The emergence of new applications demands more computational power. A number of business constraints on the organizations have led to the development and enhancement to a altogether different collaborative computing environment called grid computing. Grid computing is a recent approach to perform parallel and distributed computing. Grid combines a pool of servers, storage systems and networks into a large system. The users and applications using the grid have a illusion that this environment is a virtual uniprocessor system. Grid connects these systems that may be in the single room or distributed over a LAN, MAN or WAN with multiple hardware platforms and different operating systems. The systems may be under different organizations where every node has its own resource manager that behaves like an independent entity. Certain resource management and load balancing polices need to provide resource management in a secured manner to avoid the bottlenecks [13, 14, 15, 5, 12]. The existing models are tested on newly developed simulators, we have planned to tested on some standard grid schedulers. In order to validate practicality of the models, we have also planned, to experiment the models on the real grid environments using some realistic grid application.

(iii) Improvements in the performance of load balancing strategies for middleware and database systems: Most of the research on Internet servers concentrates on web servers but with explosion in the traffic on websites, the attention needs to be shifted to the entire system including middleware and the databases. Scaling issues for these systems are hard to deal compared to the web-servers because the business logic often involves the entire data repository, while web-servers only deal with a few files at a time. All the performance related issues of web servers apply to middleware and backend systems. Effective load balancing strategies can be explored on heavily
loaded middleware and backend systems [9, 4]. Rather than developing solutions for specific middleware, generalized middleware load balancing solutions is planned to be developed in this research.

(iv) Suggesting improved dynamic content handling techniques: The contents of most of the web service providers are dynamic that are either created on the fly while serving the users’ requests or their content changes very frequently. The changes may be due to modifications in the actual data in the database such as changes in the inventory data of an e-commerce website, the variation in the values such as change in the prize of shares etc. Investigations can be made on improving the dynamic content handling protocol and dynamic content delivery architectures. Only few parts of the dynamic pages actually change while rest of the portion is static image or text. But presently most of the services are providing such pages by recreating the whole page for each request. This property of dynamic web pages can be further explored to properly handle the load on dynamic websites. An effective caching scheme can also be developed for handling of dynamic content. The research should concentrate on “what, when and where” to cache the data while keeping the consistent data in the cache [3, 6, 7, 2, 10]. We have planned a detailed analysis of the above mentioned dynamic content caching schemes to find their limits. Instead of caching the complete responses, we are also planning to implement strategies involving caching database query results, specific fragments of responses, or compiled script-caching.

Recent trends in the scheduling are based on job sizes. In past few years, research has been primarily focused on the workload and job size distribution in variety of contexts including web file size, FTP file transfer etc. The change in the sequence of the execution of job can reduce the response time drastically. In [11], a detailed implementation study on behavior of web servers serving the static requests under the transient overload was suggested. These papers suggest kernel level modifications in the web servers to change the policy from standard FAIR (processor sharing) to shortest-remaining-processing time (SRPT). This would improve the performance drastically. Results show that significant performance improvement can be achieved just by serving just the small size request first without much affecting the performance of larger jobs. Similar policy should be explored for dynamic content websites. Also, the effect of some parameters like dynamic content caching and content awareness is not covered in the above research. The work is further investigated and it has been concluded that the policy reduces the mean response time [1]. A similar policy, called least-attend-service (LAS) can be used if job size is not known in advance or it is difficult to estimate the job size. We are going to further explore the LAS and SRPT policies for the dynamic content scheduling.

EXPERIMENTAL SETUP
The strategies discussed above may also be tested on a real environment. We may also use the server cluster and campus-wide network available in Information Technology Centre (ITC) at Devi Ahilya University (DAU), Indore. The campus-wide network is established to facilitate students and research community. The university is having two lease lines having 2 Mbps and 512 Kbps bandwidth. The 2 Mbps connection is used to provide internet access to students, staff members and teachers of DAU. University is also hosting its own mail server, web portal and other applications on the same connection. University network is divided into three regions using a CISCO ASA, a gateway firewall. These regions are external zone (EZ), internal zone (IZ) and de-militarized zone (DMZ). External zone is the outside network. IZ is the LAN of the university, where the users of university access the Internet and other facilities using a proxy server. Proxy server is so designed that only authorized user can access the internet. EZ consists of three distant campuses administration, academic and engineering college of DAU. Central administration of the network is IT Centre (ITC), a department situated at academic campus. These campuses are connected through fiber optic cable with ITC. Individual campus network is designed using mix communication media (Copper, Fiber optic and radio waves). The next region of the network is DMZ, which includes externally accessible servers, viz. mail server, web portal, DNS and other applications. The DMZ is created for providing secure access to the servers. This region is accessible from both IZ and EZ. The services are available on 24X7 basis. All the applications and services are running on servers with high processing power (dual and quad processor) and storage capacity.

![Network architecture of campus-wide network of IT Centre, DAU, Indore](Image)

The 512 Kbps leased line used to facilitate research and other academic activities. This connection is used to provide Internet access at ITC lab, which consists of around 60 PCs and thin
clients controlled by firewall, proxy server and terminal server. A software based firewall is implemented. Proxy server is used to provide internet access to authentic users. To manage thin clients the terminal server is running NFS service. Network of the university using a LDAP based directory server that maintains the users’ database. This server is used to authenticate user accessing proxy and mail services. The architectural of the DAU campus wide network is shown in the fig. 2.

CONCLUSION
Scheduling of jobs in web cluster of servers is a major research activity in distributed systems. One of the critical scheduling problems in distributed computing environment is constant pressure of increased network traffic and diverse load levels in a server cluster. Load balancing is an important performance booster for a server system. Managing the bottleneck resources effectively has great impact in improving the performance of the system. This issue has drawn the attention of researchers since the emergence of the distributed computing and seemed to be saturated. However, the recent requirement of the servers for handling bursty traffic and few successful steps in this direction has resulted in resurgence of the field.

FUTURE SCOPE
This paper presents some important enhancements in the load balancing algorithms for server load balancing. Possible methodology to incorporate the issues in current LB algorithm has been briefly proposed. Modifications in DLB methodology suggested above for improving the performance of scheduling algorithms in web servers are currently under development.

REFERENCES