Research article Available online www.ijsrr.org ISSN: 2279–0543

International Journal of Scientific Research and Reviews

Dyanamic Load Balancing Strategies For A Distributed Computer System – A Survey

A. Poornima[1] and D. Maheswari[2*]

[1]Rathinavel Subramaniam College of Arts and Science, Sulur, Coimbatore.
[2*] Head & Research Co-Ordinator, School of Computer Studies, Rathinavel Subramaniam College of Arts and Science, Sulur, Coimbatore.

ABSTRACT

A distributed system consists of independent workstations connected usually by a local area network. The IT infrastructure is playing an increasingly important role in the success of a business. Market share, customer satisfaction and company image are all intertwined with the consistent availability of a company's web site. Network servers are now frequently used to host ERP, e-commerce and a myriad of other applications. The foundation of these sites, the e-business infrastructure is expected to provide high performance, high availability, secure and scalable solutions to support all applications at all times. However, the availability of these applications is often threatened by network overloads as well as server and application failures. Resource utilization is often out of balance, resulting in the low-performance resources being overloaded with requests while the high-performance resources remain idle. Server load balancing is a widely adopted solution to performance and availability problems.

KEYWORDS: Resource Utilization, Load Balancing, Distributed system.

***Corresponding author:**

D. Maheswari[2*]

Associate Professor,

Head & Research Co-Ordinator,

School of Computer Studies,

Rathinavel Subramaniam

College of Arts and Science, Sulur, Coimbatore.

1. INTRODUCTION

Due to the growing popularity of the Internet, data centers and network servers are anticipated to be the bottleneck in hosting network-based services, even though the network bandwidth continues to increase faster than the server capacity. It has been observed that network servers contribute to approximately 40 percent of the overall delay, and this delay is likely to grow with the increasing use of dynamic Web contents. For Web-based applications, a poor response time has significant financial implications. For example, E-Biz reported about $1.9 billion loss in revenue in 1998 due to the long response time resulting from the Secure Sockets Layer which is commonly used for secure communication between clients and Web servers. Even though SSL is the de facto standard for transport layer security, its high overhead and poor scalability are two major problems in designing secure large-scale network servers. Deployment of Secure Socket Layer can decrease the server's capacity up to two orders of magnitude.

Application servers provide dynamic contents and the contents require secure mechanisms for protection. Generating dynamic content takes about 100 to 1,000 times longer than simply reading static content. Moreover, since static content is seldom updated, it can be easily cached. Several efficient caching algorithms have been proposed to reduce latency and increase throughput of front-end Web services. While dynamic content is generated during the execution of a program, caching dynamic content is not an efficient option like caching static content. Recently, a multitude of network services have been designed and evaluated using cluster platforms. Specifically, the design of distributed Web servers has been a major research thrust to improve the throughput and response time. The IT infrastructure is playing an increasingly important role in the success of a business. Market share, customer satisfaction and company image are all intertwined with the consistent availability of a company's Web site. Network servers are now frequently used to host ERP, e-commerce and a myriad of other applications. The foundation of these sites, the e-business infrastructure is expected to provide high performance, high availability, secure and scalable solutions to support all applications at all times. However, the availability of these applications is often threatened by network overloads as well as server and application failures. Resource utilization is often out of balance, resulting in the low-performance resources being overloaded with requests while the high-performance resources remain idle. Server load balancing is a widely adopted solution to improve performance and availability problems. Server load balancing is the process of distributing service requests across a group of servers.

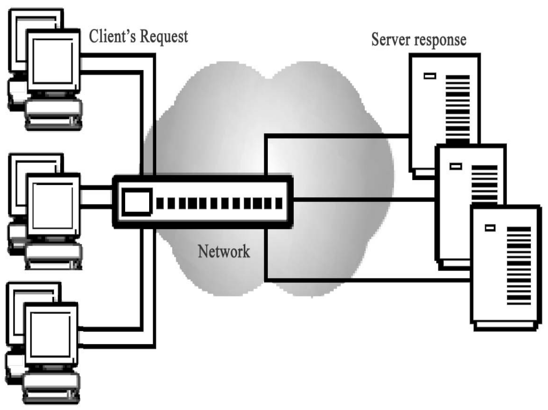

Figure-1: Client – Server interaction

Figure-1 represents how clients interact with the server. End-user requests are sent to a load-balancing device that determines which server is most capable of processing the request. Then it forwards the request to that server. Server load balancing can also distribute workloads to firewalls and redirect requests to proxy servers and caching servers.

Many content-intensive applications have scaled beyond the point where a single server can provide adequate processing power. Both enterprises and service providers need the flexibility to deploy additional servers quickly and transparently to end-users. Server load balancing makes multiple servers appear as a single server a single virtual service by transparently distributing user requests among the servers. The highest performance is achieved when the processing power of servers is used intelligently. Advanced server load-balancing products can direct end-user service requests to the servers that are least busy and therefore capable of providing the fastest response times. Necessarily, the load-balancing device should be capable of handling the aggregate traffic of multiple servers. If a server load-balancing device becomes a bottleneck it is no longer a solution, it is just an additional problem. The other benefit of server load balancing is its ability to improve application availability. If an application or server fails, load balancing can automatically redistribute end-user service requests to other servers within a server farm or to servers in another location. Server load balancing also prevents planned outages for software or hardware maintenance from disrupting service to end-users. Distributed server load-balancing products can also provide disaster recovery services by redirecting service requests to a backup location when a catastrophic failure disables the primary site.

Some applications require persistent sessions between clients and servers. Persistence is the ability to ensure that a user's session with a server will continue to be connected to that particular server. The reasons to preserve a specific session to a particular server can vary from optimizing the cache performance of the server to ensure a session that is not broken. A broken session can result in a shopping cart losing its contents on an e-commerce site. Persistence based on IP destination address enables service providers and web content providers to optimize repetitive Web hits to specific content. Persistence based on source IP address ensures that a client remains connected to a specific server for the duration of a state full transaction. Simple persistence based on source IP address works for nearly all Internet applications, except for those clients that might be located behind web proxy farms. It is possible in this scenario for a user's source IP address to change during a single session. This can be overcome using persistence based on the source IP address with a mask. As a result, any sessions from a given set of Web proxies will be aggregated to single server.

Some applications may have special needs, such as URL/cookie persistence or SSL session ID persistence for more secure transactions. The most common motivation for considering URL/cookie persistence is to improve database cache hits on servers and preserve session integrity for clients situated behind proxy server farms. This persistence requires all client sessions to terminate on the server load-balancing device and then be reconnected from the server load-balancing device to all servers. The performance impact of this approach is significant. The negative aggregate performance on the server load balancing device may be greater than any "cache hit" benefits to the server. Session integrity is often more easily handled by utilizing the previously discussed IP source address persistence with an appropriate mask. In this way, each group of proxy server farms maintains session integrity with individual servers.

There are various Internet applications such as financial transactions, database access are always running on the Internet Server. Those applications must be able to run on multiple servers to accept an ever increasing number of users and networks need the ability to scale the performance to handle large volumes of client requests without creating unwanted delays. So, load balancing concept must be implemented for better performance as well as to increase the ability to handle more number of users. For these reasons, clustering is of wide interest to the enterprise. Clustering enables a group of independent servers to be managed as a single system for higher availability, easier manageability, and greater scalability.

2. DISTRIBUTED SYSTEM

A number of computers like minicomputer, workstations, PCs which handle all the process are distributed physically and connected through a communications network. It is network structure in which the network resources, such as switching equipment and processors, are distributed throughout the geographical area being served. Arrangement of networked computers in which several processors (the CPUs) are located on scattered machines, but are capable of working both independently and jointly as required. The computer programming and the data to be worked on are spread out over more than one computer, usually over a network. For efficient network management, the multi-level domain approach has been used. Management function (Configuration, Fault tolerance, Performance management) is distributed in each management domain. Various hardware and software architectures are used for distributed computing. At a lower level, it is necessary to interconnect multiple CPUs with some sort of network, regardless of whether that network is printed onto a circuit board or made up of loosely-coupled devices and cables. At a higher level, it is necessary to interconnect processes running on those CPUs with some sort of communication system.

The distributed networking architecture has multiple sub server systems to share a particular resource efficiently. Every PC server (Server1, Server 2 …) has more number of sub systems and every main server has more number of sub-servers connected to the central server. Distributed networks have the centralized server which may have more number of sub-servers. Each system has number of sub-system and local data. The central data will be maintained in the central server. There are few advantages of using the distributed systems are as follows:

- **Localized mobility:** Latency is lowered, while the probability for a controlled, smooth handover is increased due to an end-to-end delay reduction.
- Elimination of a single point of failure.
- **Better backhauling capacity utilization:** Backhauling costs for download diversity traffic, peer-to-peer traffic, and volume of IP traffic with local PSTN connectivity are reduced to nearly zero.

The Distributed Network Protocol (DNP) defines communication procedures for components of process automation systems. Distributed Network protocol performs multiplexing, data fragmentation, error checking, link control, prioritization, and addressing services for user data, operating at the Data Link Layer of the OSI Reference Model. When discussing about distributed network and network sharing, Web clusters are used to represent the communication between the servers and the sub-servers based on the requests.

3. LOAD BALANCING

A cluster is a set of computers that work together to provide a service to the clients. The use of a cluster enhances both the availability and scalability of the service. Network Load Balancing provides a software solution for clustering multiple computers running networked client/server applications. Complex systems make increasing demands on Web servers. Multiple objects can interfere, high volumes can overwhelm systems. The fixes need to be identified early in the system, and clients have scalability concerns, and must warrantee some level of scalability with industry accepted metrics. The basic performance challenges both the browser and the server sides of the equation and advises on an overall approach for identifying and attacking performance bottlenecks. Identifying load of the servers is more a complicated process. The identification of load refers to the practice of modeling the expected usage of a software program by simulating multiple users accessing the program concurrently. As such load identification which is most relevant for multi-user systems; often one built using a client/server model, such as Web servers. However, other types of software systems can also be used to test the load.

Load Balancing can be done by implementing several techniques. The steps in each technique involve the network construction with multiple sub servers. Allocating sub-servers is the major part of efficient load distribution. As the researcher mentioned earlier, some load balancing techniques failed to request efficient navigation, although the server load is distributed, the efficient redirection has failed to provide proper response.

4. RELATED WORKS

In a single Web server environment, the cost of the SSL layer was studied by Apostolopoulos et al.[1] using the Netscape Enterprise Server and Apache Web server, and it was shown that the session reuse is critical for improving the performance of Web servers. This study was extended to a cluster system that was composed of three Web server nodes[1]. The paper has described the architecture of the L_5 system and has presented two application experiments: Routing HTTP session based on Uniform Resource Locators (URLs) and Session-aware dispatching of SSL connections. The SSL-session reuse scheme is also investigated[2], which presented a session-based adaptive overload control mechanism based on SSL connections differentiation and admission control.

Guitart et al.[3] proposed a possible extension of the Java Secure Socket Extension (JSSE) API to allow the differentiation of resumed SSL connections from new SSL connections. Recent studies on data centers have focused on cluster based Web servers[4,5,6] and the following works are related to the researcher research.

Aron et al.[4] has proposed the backend request forwarding scheme in cluster-based Web servers for supporting HTTP1.1 persistent connections. The client requests are directed by a content-blind Web switch to a Web server in the cluster by a simple distribution scheme such as the RR Domain Name System (DNS). The first node that receives the request is called the initial node. The initial node parses the request and determines whether to service it locally or forward it to another node based on the cache and load balance information. The forwarded request is sent back to the initial node for responding to the client. However, this study does not consider the impact of user-level communication and SSL-enabled application servers. The first effort that has analyzed the impact of user-level communication on distributed Web servers is the PRESS model[5]. The clients in the PRESS model communicate with the cluster using TCP over a Fast Ethernet, whereas the intra-cluster communication uses VIA over connectionless local area network (cLAN)[5] . It is shown that the server throughput can improve up to 30 percent by deploying VIA.

J.H. Kim et al.[6] shows that, in addition to taking advantage of a user-level communication scheme, co-scheduling of the communicating processes reduces the average response time by an additional 25 percent. Due to the low cost of the intra-cluster communication, reading a file from a remote cache turns out to be faster than reading the file from the local disk. Implementation on an 8-node cluster shows that PRESS can improve the server throughput by about 29 percent compared to the TCP/IP model.

Zhou et al.[7] have deployed VIA between a database server and the storage subsystem. They implemented the interface, called Direct Storage Access (DSA), to support the Microsoft SQL Server to use VIA. However, none of these studies has investigated the application server performance with SSL offering. Amza et al.[8] have explored the characteristics in several Web sites, including auction, online bookstore, and bulletin board sites, using synthetic benchmarks. In their study, the online bookstore benchmark reveals that the CPU in the database server is the bottleneck, whereas the auction and bulletin board sites show that the CPU in the Web server is the bottleneck. Cecchet et al.[9] examines the performance and scalability issues in Enterprise Java Beans (EJB) applications. They have modeled an online auction site like eBay and have experimented on it by several EJB implementations. Their test shows that the CPU on the EJB application server is the performance obstacle. In addition, the network is also saturated at some services.

The design of In fini B and data centers is studied[10]. It compares the performance between Socket Direct Protocols (SDP) and native sockets implementation over InfiniBand (IPoIB). This paper only uses the user-level communication in a data center without any intelligent distribution algorithm or architectural support for secure transactions. Yingvu zhu et al.[11] has proposed the structured Peer-to-peer systems implemented by several techniques such as hash functions. The paper efficient, proximity- aware load balancing for structured P2P systems has proposed an algorithm for the efficient distributed systems. This has been developed to guide the load balancing system efficiently by instructing them in a right way. This has been done in all load balancing systems, those system stores all proxy details in a tree structure so that load could be shared in the structured peers. This is effectively working in the structured systems but fails to work on unstructured systems. The idea behind this paper presents the guidance for the load balancing system may help to simplify the load distribution. In particular, the main contributions of this system are:

(1) A self-organized, fully distributed K-nary tree structure is constructed on top of a Distributed Hash Table (DHT) for load balancing information collection/dissemination and load reassignment.

(2) Load balancing is achieved by aligning those two skews in both load distribution and node capacity in here in P2P systems - that is, to have higher capacity nodes carry more loads.

(3) Proximity information is utilized to guide load balancing such that virtual servers are assigned and transferred between physically close heavy nodes and light nodes, thereby minimizing the load transferring overhead and making load balancing fast and efficient.

The idea behind this paper by using a hash table and guiding through the tree based tables may help to increase the load balancing performance.

Quang Hieu Vu et al.[12] have the graphical representations of the peers handled in the histogram load balancing concept. For effective resource sharing the server maintains all details about the peers globally. It has two key components: (1) A histogram manager maintains a histogram that reflects a global view of the distribution of the load in the system. (2) A load-balancing manager that redistributes the load whenever the node becomes over or under loaded. It exploit the routing metadata to partition the P2P network into non-overlapping regions corresponding to the histogram buckets. It proposes mechanism to keep the cost of constructing and maintaining the histograms low. It shows that the scheme can control and bound the amount of load imbalance across the system. Finally, it demonstrates the effectiveness of the proposed system by instantiating it over three existing structured P2P system.

Yuh-Ming Chin et al.[13] has proposed load balancing system which is distributing the resource effectively across the global networks. When the load distribution has implemented, the feasibility analysis must be considered. The server and the proxy servers must act quickly according to the client's requests. The bandwidth and the response time decide the efficiency of the load balancing system. The paper 'Minimizing File Download Time in stochastic peer-to peer networks' has involved in decrementing the download time. The peer-to-peer (P2P) file-sharing applications are becoming increasingly popular and account for more than 70% of the Internet's bandwidth usage. Measurement studies show that a typical download of a file can take from minutes up to several hours depending on the level of network congestion or the service capacity fluctuation. The author considers two major factors that have significant impact on average download time, namely, the spatial heterogeneity of service capacities in different source peers and the temporal fluctuation in service capacity of a single source peer. It points out that the common approach of analyzing the average download time based on average service capacity is fundamentally flawed. It rigorously proves that both spatial heterogeneity and temporal correlations in service capacity increase the average download time in P2P networks and then analyzes a simple, distributed algorithm to effectively remove these negative factors, thus minimizing the average download time. It shows through analysis and simulations that it outperforms most of other algorithms currently used in practice under various network configurations. The adoption of SSL (Secure Sockets Layer) to secure data across the Internet and World Wide Web is growing at a dramatic rate. However, SSL connection setup and processing can severely impair standard servers. There are several technologies available today to help offload servers from these computationally intensive tasks.

5. SERVER LOAD BALANCING TECHNIQUES

5.1 Round Robin

Round Robin load balancing allows the client to distribute their requests across multiple servers. Load balancers improve server fault tolerance and end-user response time. Load balancer distributes client requests across multiple servers to optimize resource utilization. In a scenario with a limited number of servers providing service to a large number of clients, a server can become overloaded and degrade server performance. Load balancing is used to prevent bottlenecks by forwarding the client requests to the servers best suited to handle them.

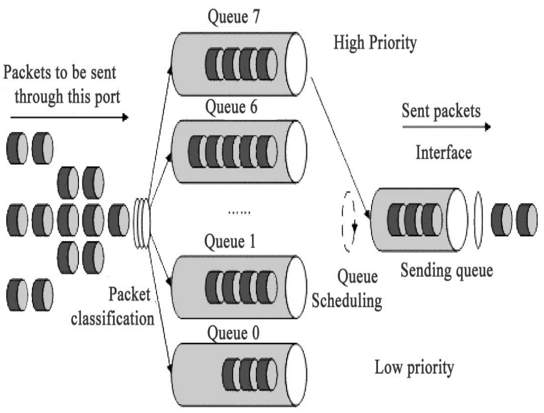

Figure-2: Data Processing using RR method

Figure-3 shows the RR scheduling method. In a Round-Robin algorithm, the IP sprayer assigns the requests to a list of the servers on a rotating basis. The first request is allocated to a server picked randomly from the group, so that if more than one IP sprayer is involved, not all the first requests go to the same server. For the subsequent requests, the IP sprayer follows the circular order to redirect the request. Once a server is assigned a request, the server is moved to the end of the list. This keeps the servers equally assigned.

5.2 Weighted Round-Robin Allocation

Weighted Round-Robin is an advanced version of the Round-Robin that eliminates the deficiencies of the plain Round Robin algorithm. In case of a Weighted Round-Robin, one can assign a weight to each server in the group so that if one server is capable of handling twice as much load as the other, the powerful server gets a weight of 2. In such cases, the IP sprayer will assign two requests to the powerful server for each request assigned to the weaker one. It takes care of the capacity of the servers in the group. It does not consider the advanced load balancing requirements such as processing times for each individual request. The configuration of a load balancing software or hardware should be decided on the particular requirement. For example, if the Website contains static HTML pages or light database driven dynamic Web pages, Round Robin will be sufficient. However, if some of the requests take longer than the others to process, then advanced load balancing algorithms are used. The load balancer should be able to provide intelligent monitoring to distribute the load, directing them to the servers that are capable of handling them better than the others in the cluster of server.

Table-1: Sample WRR Weights

Sub-Group	Server	Weight/Connections per cycle
1.	S_1	8
2.	S_2	8
3.	S_3	2
4.	S_4	2
5.	S_5	4
6.	S_6	3
TOTAL		**27**

Table-1 illustrates a server farm consisting of four groups of six real servers in total that are assigned various weights. The total number of connections per cycle in this example is 27.

5.3 Least Connections

With the least-connections algorithm, as the name suggests, the content switch forwards new requests to real servers with the fewest connections. The content switch maintains the concurrent number of existing connections to each real server. When a real server receives a new connection, the content switch increments the count. When clients or servers tear down connections, the content switches will automatically decrements the amount. The benefit of the least-connections load distribution mechanism is that it creates an even distribution of connections across the real server's. Real server weighting is also available for the least-connections predictor algorithm those real's with higher relative weights receive a larger proportion of the available connections. The difference with least-connection weighting and the weighting mechanism in WRR is the way in which the content switch uses the weight to determine the distribution of connections. For example, say that user gives the same weights to sub-groups of real server's within the server farm as given previously in Table-2`.

Table -2: Sample Weighted Least-Connections Proportion Calculations

Server	Weight	Percentage of connections
S1	8	8/27 = 29%
S2	8	8/27 = 29%
S3	2	2/27 = 7%
S4	2	2/27 = 7%
S5	4	4/27 = 14%
S6	3	3/27 = 11%
TOTAL	27	27/27 =100%

Consider a server farm consisting of N subgroups of real server's, with N different weights 1, 2 ... N. During one cycle, the real subgroup with weight 1 would receive 1 / (1 + 2 + ... + N) connections, the real server with weight 2 would receive 2 / (1 + 2 + ... + N) connections, and so forth. Table-2 illustrates how the least-connections algorithm distributes the load with the same weights as given previously with Weighted Round Robin in Table-2.

The weighted least connections algorithm specifies that the next real server chosen from a server farm for a new connection to the virtual server is the server with the fewest active connections. Each real server is assigned a weight for this algorithm, also. When weights are assigned, the server with the fewest connections is based on the number of active connections on each server, and on the relative capacity of each server. The capacity of given real server is calculated as the assigned weight of that server divided by the sum of the assigned weights of all of the real servers associated with that virtual server, or n1 / (n_1+n_2+n_3...).

For example, assume a server farm comprised of real server A with n = 3, Server B with n = 1, and Server C with n = 2. Server A would have a calculated capacity of 3/(3+1+2), or half of all active connections on the virtual server, Server B one-sixth of all active connections, and Server C one-third of all active connections. At any point in time, the next connection to the virtual server would be assigned to the real server whose number of active connections is farthest below its calculated capacity.

6. CONCLUSION

Various algorithms have been proposed in the literature, and each of them varies based on some specific application domain. Some load balancing strategies work well for applications with large parallel jobs, while others work well for short, quick jobs. Some strategies are focused towards handling data-heavy tasks, while others are more suited to parallel tasks that are computation heavy. Some load balancing techniques failed to efficient request navigation, although the server load is distributed, the efficient redirection has failed to provide proper response.

REFERENCES

1. Apostolopoulos G., V. Peris and D. Saha. Transport Layer Security: How Much Does It Really Cost?. Proc. INFOCOM, 1999; 2 : 717-725.
2. Apostolopoulos G., D. Aubespin, V. Peris, P. Pradhan, and D. Saha. Design, Implementation and Performance of a Content-Based Switch. In Proceedings of the 2000 IEEE Computer and Communications Societies Conference on Computer Communications (INFOCOM-00), Los Alamitos, 2000; 3 : 1117–1126.
3. Guitart J., D. Carrera, V. Beltran, J. Torres, and E. Ayuade. Session-Based Adaptive Overload Control for Secure Dynamic Web Applications. Proc. International Conference Parallel Processing (ICPP '05), 2005; 341 – 349.
4. Aron M., P. Druschel, and W. Zwaenepoel. Efficient Support for P-HTTP in Cluster-Based Web Servers. Proc. Usenix Ann. Technical Conf., 1999; 185-198.
5. Carrera E.V., S. Rao, L. Iftode, and R. Bianchini. User-Level Communication in Cluster-Based Servers. Proc. Eighth International Symp. High-Performance Computer Architecture (HPCA '02), 2002; 275-286.
6. Kim J.H., G.S. Choi, D. Ersoz, and C.R. Das. Improving Response Time in Cluster-Based Web Servers through Co-scheduling. Proc. 18th International Parallel and Distributed Processing Symposium, 2004; 88-97.
7. Zhou Y., A. Bilas, S. Jagannathan, C. Dubnicki, J.F. Philbin, and K.Li. Experiences with VIA Communication for Database Storage. Proc. 29th Ann. International Symp. Computer Architecture, 2002; 257-268.
8. Amza C., A.Chanda, A.L.Cox, S. Elnikety, R. Gil, E. Cecchet, J. Marguerite, K.Rajamani, and W. Zwaenepoel. Specification and Implementation of Dynamic Web Site Benchmarks. Proc. IEEE Fifth Annual Workshop Workload Characterization (WWC-5), 2002; 3-13.
9. Cecchet E., J. Marguerite, and W. Zwaenepoel. Performance and Scalability of EJB Applications. Proc. 17th ACM SIGPLAN Conf. Object-Oriented Programming, Systems, Languages, and Applications, 2002; 246-261.

10. Balaji P., S. Narravula, K. Vaidyanathan, S. Krishnamoorthy and D.K. Panda. Sockets Direct Procotol over InfiniBand in Clusters: Is It Beneficial?. Proc. IEEE International Symp. Performance Analysis of Systems and Software (ISPASS '04), 2004; 28-35.
11. Yingvu zhu, Yiming Hu. Efficient, Proximity-Aware Load Balancing for DHT-Based P2P Systems. Peer-to-Peer Computing, Proceedings, Third International conference, 2003; 16(4) :349-361.
12. Quang Hieu Vu, Beng Chin Ooi, Rinard.M, Kian-Lee Ton. Histogram based Global Load Balancing in structured Peer-to-Peer Systems. IEEE Transaction on Knowledge and Data Engineering, 2009; 21(4) : 595-608.
13. Yuh-Ming Chin, Do Young Eun. Minimizing File Download Time in Stochastic Peer-to-Peer Networks. IEEE / ACM, 2006; 16(2): 253-266.