ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Specialized algorithm for Client Server Communication for cost effectiveness and Load Balancing over Distributed Systems.

Annavarapu Mahendra Babu
M.Tech, Student,
Computer Science Engineering Department,
Rao & Naidu Engineering College, Ongole.

N. Narasa Reddy
M.Tech, Asst Professor,
Computer Science Engineering Department,
Rao & Naidu Engineering College, Ongole.

Abstract:

An Internet distributed system is a software system in which components located on networked computers communicate and coordinate their actions by passing messages. The components interact with each other in order to achieve a common goal. Three significant characteristics of distributed systems are: concurrency of components, lack of a global clock, and independent failure of components. For an optimized performance for such a method requires appropriate client-server assignment. We can accomplish such a distributed system by primarily taking into account the factors like full communication load and load balancing of the servers.

Optimizing the overall performance of such a system then can be formulated as a client-server assignment problem whose aim is to assign the clients to the servers in such a way to satisfy some prespecified requirements on the communication cost and load balancing.

We show that 1) the total communication load and load balancing are two opposing metrics, and consequently, their tradeoff is inherent in this class of distributed systems; 2) in general, finding the optimal client-server assignment for some prespecified requirements on the total load and load balancing is NP-hard, and therefore; 3) we propose a heuristic via relaxed convex optimization for finding the approximate solution. Our simulation results indicate that the proposed algorithm produces superior performance than other heuristics, including the popular Normalized Cuts algorithm. sIn this paper we examined a method depending on an algorithm which obtains an approximately optimal solution for the client-server assignment problem.

Keywords:

Client-Server, Communications, Distributed Systems, Metrics, Messaging systems and Protocols

Introduction:

The word distributed in terms such as “distributed system”, “distributed programming”, and “distributed algorithm” originally referred to computer networks where individual computers were physically distributed within some geographical area.

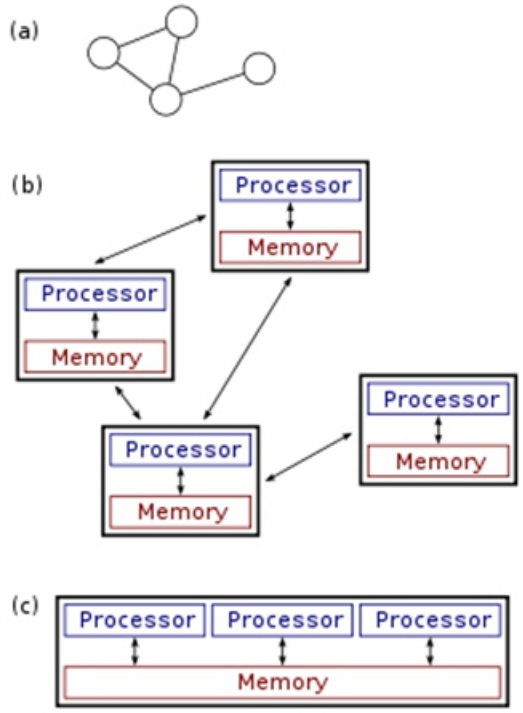
Architecture:

Client/Server System : The Client-server architecture is a way to provide a service from a central source. There is a single server that provides a service, and many clients that communicate with the server to consume its products. In this architecture, clients and servers have different jobs. The server’s job is to respond to service requests from clients, while a client’s job is to use the data provided in response in order to perform some tasks.

Peer-to-Peer System :

The term peer-to-peer is used to describe distributed systems in which labour is divided among all the components of the system. All the computers send and receive data, and they all contribute some processing power and memory. As a distributed system increases in size, its capacity of computational resources increases. In a peer-to-peer system, all components of the system contribute some processing power and memory to a distributed computation.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal



(a)–(b) A distributed system.
(c) A parallel system.

The figure above illustrates the difference between distributed and parallel systems. Figure (a) is a schematic view of a typical distributed system; as usual, the system is represented as a network topology in which each node is a computer and each line connecting the nodes is a communication link.

Figure (b) shows the same distributed system in more detail: each computer has its own local memory, and information can be exchanged only by passing messages from one node to another by using the available communication links. Figure (c) shows a parallel system in which each processor has a direct access to a shared memory.

Client Server Model:

The client–server model of computing is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. Often clients and servers communicate over a computer network on separate hardware, but both client and server may reside in the same system. A server host runs one or more server programs which share their resources with clients.

A client does not share any of its resources, but requests a server's content or service function. Clients therefore initiate communication sessions with servers which await incoming requests. Examples of computer applications that use the client–server model are Email, network printing, and the World Wide Web.

Fig: A computer network diagram of clients communicating with a server via the Internet.

The client–server characteristic describes the relationship of cooperating programs in an application. The server component provides a function or service to one or many clients, which initiate requests for such services. Servers are classified by the services they provide. For instance, a web server serves web pages and a file server serves computer files.

A shared resource may be any of the server computer's software and electronic components, from programs and data to processors and storage devices. The sharing of resources of a server constitute a service.

Whether a computer is a client, a server, or both, is determined by the nature of the application that requires the service functions. For example, a single computer can run web server and file server software at the same time to serve different data to clients making different kinds of requests.

Client software can also communicate with server software within the same computer. Communication between servers, such as to synchronize data, is sometimes called inter-server or server-to-server communication.

Client and server communication:

In general, a service is an abstraction of computer resources and a client does not have to be concerned with how the server performs while fulfilling the request and delivering the response.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

The client only has to understand the response based on the well-known application protocol, i.e. the content and the formatting of the data for the requested service.

Clients and servers exchange messages in a request-response messaging pattern: The client sends a request, and the server returns a response. This exchange of messages is an example of inter-process communication. To communicate, the computers must have a common language, and they must follow rules so that both the client and the server know what to expect. The language and rules of communication are defined in a communications protocol.

All client-server protocols operate in the application layer. The application-layer protocol defines the basic patterns of the dialogue. To formalize the data exchange even further, the server may implement an API (such as a web service). The API is an abstraction layer for such resources as databases and custom software. By restricting communication to a specific content format, it facilitates parsing. By abstracting access, it facilitates cross-platform data exchange.

A server may receive requests from many different clients in a very short period of time. Because the computer can perform a limited number of tasks at any moment, it relies on a scheduling system to prioritize incoming requests from clients in order to accommodate them all in turn.

To prevent abuse and maximize uptime, the server's software limits how a client can use the server's resources. Even so, a server is not immune from abuse. A denial of service attack exploits a server's obligation to process requests by bombarding it with requests incessantly. This inhibits the server's ability to responding to legitimate requests.

An Internet distributed system consists of a number of nodes (e.g., computers) that are linked together in ways that allow them to share resources and computation. An ideal distributed system is completely decentralized, and that every node is given equal responsibility and no node is more computational or resource powerful than any other. However, for many real-world applications, such a system often has a low performance due to a significant cost of coordinating the nodes in a completely distributed manner.

In practice, a typical distributed system consists of a mix of servers and clients. The servers are more computational and resource powerful than the clients. A classical example of such systems is e-mail. When a client A sends an e-mail to another client B, A does not send the e-mail directly to B. Instead, A sends its message to its e-mail server which has been previously assigned to handle all the e-mails to and from A. This server relays A's e-mail to another server which has been previously assigned to handle e-mails for B.

B then reads A's e-mail by downloading the e-mail from its server. Importantly, the e-mail servers communicate with each other on behalf of their clients. The main advantage of this architecture is specialization, in the sense that the powerful dedicated e-mail servers release their clients from the responsibility associated with many tasks including processing and storing e-mails, and thus making e-mail applications more scalable. E-mail systems assign clients based primarily on the organizations that the clients belong to.

Two employees working for the same company are likely to have their e-mail accounts assigned to the same e-mail server. Thus, the clientserver assignment is trivial. A more interesting scenario is the Instant Messaging System (IMS). An IMS allows realtime text-based communication between two or more participants over the Internet.

Each IMS client is associated with an IMS server which handles all the instant messages for its clients. Similar to e-mail servers, IMS servers relay instant messages to each other on behalf on their clients. In an IMS that uses the XMPP (Jabber) protocol such as Google Talk, clients can be assigned to servers independent of their organizations. Furthermore, the client-server assignment can be made dynamic as deemed suitable, and thus making this problem much more interesting.

Existing System:

Any general distributed system contains a combination of servers and clients. The servers are equipped with additional computational capabilities and enhanced hardware configuration when compared to clients. A typical instance of such systems is electronic mail. When a client A transmits an email to another client B, it is not sent directly.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Instead, A transmits its message to its e-mail server which has been previously assigned to handle all the e-mails to and from A. This server transmits Client-A's e-mail to another server which has been formerly allocated to handle emails for Client-B. B then reads A's e-mail by downloading the e-mail from its server. Prominently, the e-mail servers communicate with each other on behalf of their clients.

The major benefit of this design is specialization, which is high configured e-mail servers relieve their clients from the task related with many tasks including processing and storing messages, and consequently making e-mail applications more scalable.

Instant messaging is a set of communication technologies used for text-based communication between two or more participants over the Internet or other types of networks. IM–chat happens in real-time. Of importance is that online chat and instant messaging differ from other technologies such as email due to the perceived quasi-synchrony of the communications by the users.

Some systems permit messages to be sent to users not then 'logged on' (offline messages), thus removing some differences between IM and email (often done by sending the message to the associated email account).

IM allows effective and efficient communication, allowing immediate receipt of acknowledgment or reply. However IM is basically not necessarily supported by transaction control. In many cases, instant messaging includes added features which can make it even more popular.

For example, users may see each other via webcams, or talk directly for free over the Internet using a microphone and headphones or loudspeakers. Many applications allow file transfers, although they are usually limited in the permissible file-size.

Every IMS client is connected with an IMS server which handles all the instant messages for its clients. Similar to e-mail servers, IMS servers relay instant messages to each other on behalf on their clients. In an IMS that uses the XMPP (Jabber) protocol such as Google Talk, clients can be assigned to servers independent of their organizations.

Drawbacks of Existing System:

We use multiple servers; we also need to balance the communication load among the servers for the following reasons:

* If one server is overloaded, we are require to add another server to share out the load, which is cost-effectively inefficient and typically increases the overall communication load.

* As a heavily loaded server classically exhibits a low performance, we would like to avoid the situation.

* To minimize the amount of total communication load, assigning all clients to one server is optimal. Nevertheless, it is not possible due to overloading and completely loses the load balance. Simple load balancing does not usually take account of reducing the overall communication load.

Proposed System:

In the proposed system the primary contribution is a heuristic algorithm via relaxed convex optimization that takes a given communication pattern between the clients as an input, and produces an approximately optimal client-server assignment for a pre-specified tradeoff between load balance and communication cost.

We must strike a balance between reducing the overall communication load and increasing the load fairness among the servers, i.e., the load balance.

Merits of Proposed System:

The advantages of the proposed system have follows:

» The two groups have different number of servers, a server within a group with fewer servers will likely to have a higher load than a server in the group with more servers. This reduces the load balance.

» We describe a number of emerging applications that have the potential to benefit from the client-server assignment problem.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

» It optimizes the performance of a class of distributed systems over the Internet.

» Reduction in the overall communication load

» Increases the load fairness among the servers, i.e., the load balance.

Applications:

» 1. Facebook: It allows circles of friends to exchange messages assigning them to the same server which will reduce the inter-server communication and will result in reducing the overall communication load.

» 2. Twitter: Like Facebook, same operation is performed in twitter for optimizing the overall performance.

» 3. eBay: In this case, it lets a user log in to the server that is likely to have contents of interest to that user which will raise the efficiency of the system.

The client-server assignment problem is also relevant to a host of emerging applications ranging from social network applications such as Facebook and Twitter to online distributed auction systems such as eBay.

Facebook is a system that allows circles of friends to exchange messages and pictures among themselves.

Since friends are likely to communicate with each other than non-friends, assigning friends to the same server will reduce the inter-server communication and will result in reducing the overall communication load.

At the same time, it is preferable to balance the communication load.

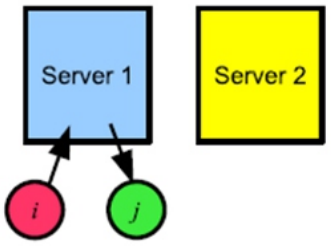
Application of this work is as well appropriate to a lot of up-and-coming applications ranging from social network applications such as Facebook and Twitter to online distributed auction systems such as eBay.

This is exactly the client-server assignment problem encountered in the IMS. Online Distributed auction systems is another application.

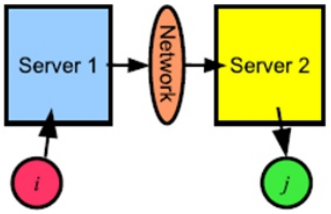
If a user logged in a server which has contents that are mostly not of interest to user, then on average, every search by a user will generate a larger communication overhead, as the search must be done across multiple servers.

Therefore letting a user log in the server that is likely to have contents of interest to a user will increase the efficiency of the system.

System Architecture:



(a) Client i and j are assigned to server 1. The messages between them are passed only via server 1.



(b) Client i is assigned to server 1 and client j is assigned to server 2. The messages are passed through server 1 and 2, which doubles the overall communication load.

Fig: Example of client assignment to servers

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Algorithms Used:

Algorithm 1 (Two-server Algorithm).

1. We start with picking up one arbitrary x_i and set it 0.
2. Afterward, we solve (27) or (28) by convex optimization.
3. However, in most cases, other elements of x will still remain nonbinary. Therefore, we choose x_c whose value is closest to 0 or 1, then set it 0 or 1 whichever x_c is closer to.
4. Repeat 2 and 3 until no more nonbinary element exists in x.

Algorithm 2 (General Algorithm).

1. Split the number of servers into two groups with $m = \lceil \frac{M}{2} \rceil$ and $M - m = \lfloor \frac{M}{2} \rfloor$ servers in each group.
2. Run Algorithm 1 with modified load balance metric (29) or (30).
3. Repeat 1 and 2 for each of the two groups, until the number of servers in each group equal to 1.

Source: Hiroshi Nishida and Thinh Nguyen, Optimal Client-Server Assignment for Internet Distributed Systems, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 24, NO. 3, MARCH 2013.

Conclusion:

In this work paper we discussed and examined an approximation algorithm for solving the client server assignment problem in internet distributed systems.

The most important aspects that are investigated in this paper for optimizing client-server assignment are optimizing the total communication cost and load balance.

We also observed that assigning the search keywords which are frequently queried together to the same servers will decrease the inter-server communication.

Application of this work is as well appropriate to a lot of up-and-coming applications ranging from social network applications such as Facebook and Twitter to online distributed auction systems such as eBay.

References:

[1] Hiroshi Nishida and Thinh Nguyen, Optimal Client-Server Assignment for Internet Distributed Systems, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 24, NO. 3, MARCH 2013.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

[2] "Finding good nearly balanced cuts in power law graphs," Yahoo Research Labs, Tech. Rep., 2004.

[3] Nishida, H.; Thinh Nguyen; , "Optimal Client-Server Assignment for Internet Distributed Systems," Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on , vol., no., pp.1-6, July 31 2011-Aug. 4 2011 .

[4] Lu Zhang; Xueyan Tang; , "Client assignment for improving interactivity in distributed interactive applications," INFOCOM, 2011 Proceedings IEEE , vol., no., pp.3227-3235, 10-15 April 2011 7 doi: 10.1109/INFCOM.2011.5935173 .

[5] Zhang, L.; Tang, X.; , "Optimizing Client Assignment for Enhancing Interactivity in Distributed Interactive Applications," Networking, IEEE/ACM Transactions on , vol.PP, no.99, pp.1, 0 doi: 10.1109/TNET.2012.2187674.

[6] Bortnikov, E., Khuller, S., Li, J., Mansour, Y. and Naor, J. S. (2012), The load-distance balancing problem. Networks, 59: 22–29. doi: 10.1002/net.20477 .

[7] Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. Linear Matrix Inequalities in System and Control Theory. SIAM, 1994.

[8] U. Feige and M. Langberg. The rpr2 rounding technique for Semidefinite programs. J. Algorithms, 60(1):1–23, 2006.

[9] T. Duong-Ba, T. Nguyen, "Distributed client-server assignment," Sch. of EECS, Oregon State Univ., Corvallis, OR, USA.

[10] S. Balakrishna and Dr. Ling Ding, "An efficient client-server assignment for Internet distributed systems," Dept. Comp. Sc. & Sys., Univ. of Washington, Tacom.

[11] S. C. Han, "Distributing link stress for Internet TV systems," Dept. Comput. Eng., Myongji Univ., Yongin, South Korea.

[12] Rajesh D. Bharati, Ishan Naidu, Anurag Kiran, Kalpesh Khune, Chirag Vyas, An Enhanced Client-Server Assignment for Internet Distributed Systems, International Journal of Engineering Trends and Technology (IJETT) – Volume 10 Number 4 - Apr 2014

[13] P. Morillo, J. M. Orduna, M. Fernandez, and J. Duato, "Improving the performance of distributed virtual environment systems," IEEE Transactions on Parallel and Distributed Systems, vol. 16, pp. 637–649, 2005.

[14] I. Dhillon, Y. Guan, and B. Kulis, "Weighted Graph Cuts Without Eigenvectors a Multilevel Approach," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 29, no. 11, pp. 1944-1957, Nov. 2007.