Volume 3, Issue 4, April 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Load Balancing In Structured Peer To Peer Network Using DSLS and ASH Algorithm

Prakashchandra Suthar
Dept. of Computer
K.J. College of Engg
University of Pune
India.

Vishweshwar Swami
Dept. of Computer
K.J. College of Engg
University of Pune
India.

Rafik Tamboli
Dept. of Computer
K.J. College of Engg
University of Pune
India.

Chaitanya Kshirsagar
Dept. of Computer
K.J. College of Engg
University of Pune
India.

Prof. Sumit Hirve
Dept. of Copmuter
K.J. College of Engg
University of Pune
India.

***Abstract-** A peer-to-peer (abbreviated to P2P) computer network is one in which each computer in the network can act as a client or server for the other computers allowing shared access. For the next generation internet applications P2P are considered to be the most important. For these systems to be effective, load balancing among the peers is critical. Load balancing is essentially the construction of call routing schemes which successfully distribute the changing load over the system and minimize lost calls. In structured P2P systems, data items are spread across distributed computers (nodes), and the location of each item is determined using a distributed hash lookup table (DHT). In this project, we address the problem of load balancing in such P2P systems. We explore the space of designing load balancing algorithms that uses the notion of virtual servers. First, a local search algorithm is used to improve the constructed solution to enhance the search. Secondly, Ant System Heuristic (ASH) which is an instance of the Ant Colony Optimization approach can be reduced to finding good paths through graphs.*

***Keywords** - Ant system heuristics, Distributed hash table, Dual space local search algorithm, Load balancing, Structured peer-peer system.*

I. Introduction

Recently there is dynamic interest for the load balancing problem in P2P in the research community. The load balancing in peer to peer focuses on the notion of Virtual Server (VS) [1] which is mostly based on service for internet applications. A number of design issues, such as how nodes should register with directory nodes, are not adequately addressed. We propose to use the search spaces and develop a corresponding search strategy as an effective approach to tackle the problem.

To assign the load to a particular node, care should be taken that cost incur to assign the load should be minimized. The problem consists of a number of tasks that are to be assigned to various nodes in the network at the least cost. An analogy with the way the dual space local search and ant colonies function has suggested the definition of a new computational paradigm. The rationale for the design is that, even though ASH is an effective randomized restart procedure that can construct a good initial solution in reinforcement style of learning using the pheromone trails, it is not as effective for finding nearby local optima solutions several steps away from the generated solution. DSLS algorithm is the local search component that finds the local optima solution in the neighborhood of a given initial solution.

II. Structured Peer To Peer Network

A peer-to-peer (abbreviated to P2P) computer network is one in which each computer in the network can act as a client or server for the other computers in the network, allowing shared access to files and peripherals without the need for a central server. P2P networks can be set up in the home, a business or over the Internet. Each network type requires all computers in the network to use the same or a compatible program to connect to each other and access other resources found on the other computer.

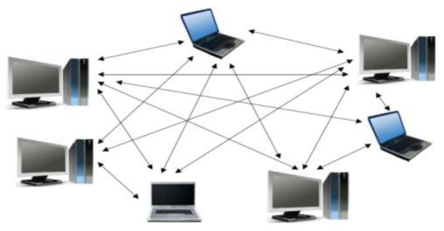

Fig.1. A peer-to-peer system of nodes without central infrastructure.

Peers are equally privileged participants in this application. Each computer in the network is referred to as a node. The owner of each computer on a P2P network would set aside a portion of its resources - such as processing power, disk storage or network bandwidth -to be made directly available to other network participant, without the need for central coordination by servers or stable hosts. With this model, peers are both suppliers and consumers of resources, in contrast to the traditional client server model where only servers supply (send), and clients consume (receive)

III. Load Balancing

Load balancing is a technique to spread work between two or more computers (peers), network links, CPUs, hard drives, or other resources, in order to get optimal resource utilization, maximize throughput, and minimize response time. The load of the peer will be calculated before processing the request. The load of the individual peer will be calculated first. If the individual load is greater than its target load then that peer will be considered as overloaded. Else the peer will be considered as normally loaded. If the peer is normally loaded the load will be handled by the same peer. Else if it is overloaded the number of peers in the region will be counted. If there is only one peer the load will be shared to that peer. If the number of peers is even the peers will be split as pairs of peers and the load is shared to the pair which is having low load. If the number of peers is odd then the peers will be split as pair of peers and there will be one more odd peer. The load will be shared to either peer pair or to the odd peer, which is having less load. The response may be like decompressed file, compressed file, document, zipped folder or unzipped files. The response will be saved in the peer who sends the request. Following Rao et al. [2], let tj denote the target load of physical node j, li denote the current load on VS i, l denote the set of all VSs, and J denote the set of nodes in the system. A P2P system is defined to be balanced if the sum of the load sj a physical node j is smaller than or equal to the target load of the node for every node j ∈ J in the system. That is,

$$Sj = \sum_{i \in I} li\, x_{ij} \leq tj,\ \forall\ j \in J \qquad (1)$$

In (1), the binary variable xij = 1 indicates that VS i is assigned to node j. A physical node j is called heavy if its combined load exceeds its target load, sj > tj; otherwise, the node is called light. When the system is imbalanced, the goal of a load balancing algorithm is to find a way to move VSs from heavy nodes to light nodes in a way that minimizes the total load moved. In such P2P networks, Calls between two points are typically routed through a number of intermediate switching stations, or nodes; in a large network, there are many possible routes for each such call. It is thus possible to relieve actual or potential local congestion by routing calls via parts of the network which have spare capacity. Load balancing is essentially the construction of call –routing schemes which successfully distribute the changing load over the system and minimize lost calls. Of course it is possible to determine the shortest routes from every node to every other node of the network. In this way the average utilization of nodes will be minimized, but this is not necessarily the ideal way to avoid node congestion, as this has to do with how the traffic on the network is distributed.

IV. Distributed Hash Table

A distributed hash table (DHT) is a class of a distributed system that provides a lookup service similar to a hash table; (key, value) pairs are stored in a DHT, and any participating node can efficiently retrieve the value associated with a given key. Responsibility for maintaining the mapping from keys to values is distributed among the nodes, in such a way that a change in the set of participants causes a minimal amount of disruption. This allows a DHT to scale to extremely large numbers of nodes and to handle continual node arrivals, departures, and failures. The physical machines in the network are called nodes and the data items are called objects in the thesis. A typical name for a node in Internet is its IP address. An identifier refers to the unique digital name of a node within an certain integer name space. In P2P system, it can be gotten from hashing the name of a node, such as identifier *P* = *hash* (*IP*). Because we apply distinct integers to mark each node's name here, name and identifier have the same meanings in the thesis if there is
no special claiming. A key is the unique identifier of a data item and can be gained from hashing the name of a
data item *K* = *hash*(*V*).

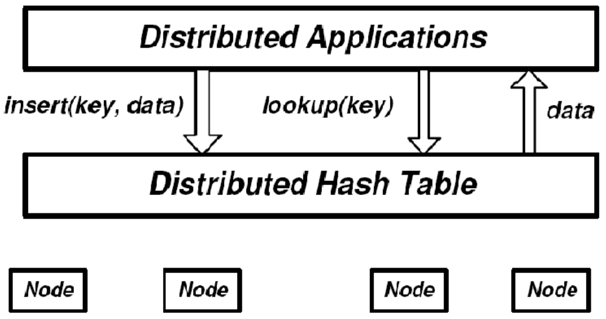

Fig.2. Distributed Hash Table

In DHT-based P2P systems, files are associated to keys(produced by hashing the filename); each node in the system handles a portion of the hash space and is responsible for storing a certain range of keys. After a lookup for a certain key, the system will return the identity (e.g., the IP address) of the node storing the object with that key. The DHT functionality allows nodes to put and get files based on their key, and has been proved to be a useful substrate for large distributed systems and a number of projects are proposing to build Internet-scale facilities layered above DHTs. In DHTs, each node handles a portion of the hash space and is responsible for a certain key range. Routing is location deterministic distributed lookup. The most important enhancements are deterministic locating and load balance.

V. Dual Space Local Search

The overall algorithm we propose is called dual-space local search (DSLS) and is based on the framework by Lourenco and Serra [3].DSLS is an iterative procedure with three main steps performed in each iteration. First, an initial solution is generated using the ant system heuristic (ASH) algorithm. The solution is then improved as much as possible to reach its local minima using the descent local search (DLS) algorithm. The pheromone variables are then updated and the overall procedure is executed again. The rationale for the design is that, even though ASH is an effective randomized restart procedure that can construct a good initial solution in reinforcement style of learning using the pheromone trails, it is not as effective for finding nearby local optima solutions several steps away from the generated solution. A local search algorithm must be used to improve the constructed solution to enhance the search in terms of earlier detection of high-quality solutions. Our DSLS algorithm is the local search component that finds the local optima solution in the neighborhood of a given initial solution. In addition, we distinguish between the search spaces by the purpose they are to achieve and thus can search in significantly smaller search spaces. The DSLS procedure is given as follows:

- Construction.: The algorithm invokes the ASH algorithm to construct an initial solution x for the current iteration.
- Improvement**:** The algorithm then invokes the DSLS procedure to derive a local minimu solution based on initial solution.
- Pheromone trail update**:** The current best solution will be replaced by x'' if it is better that the current best solution. The pheromone trail will be updated to reflect the effect of x''.

DLS is another iterative loop comprising two main phases for searching in the two ejection-chain neighborhoods. First, if the initial solution generated in the first step is not a feasible solution, the algorithm performs a local search procedure in a feasibility- improving ejection-chain neighborhood N(x) to adjust the solution to a feasible one x'. Second, based on the feasible x', the algorithm performs another local search in a cost-reducing ejection-chain neighborhood N'(x) to adjust x' to a lower cost one x''.

VI. Ant System Heuristics

ASH is an instance of the Ant Colony Optimization approach. In computer science and operations research, the ant colony optimization algorithm (ACO) is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs. The first algorithm was aiming to search for an optimal path in a graph, based on the behavior of ants seeking a path between their colony and a source of food[4]. In the natural world, ants (initially) wander randomly, and upon finding food return to their colony while laying down pheromone trails. If other ants find such a path, they are likely not to keep travelling at random, but to instead follow the trail, returning and reinforcing it if they eventually find food. The original idea comes from observing the exploitation of food resources among ants, in which ants' individually limited cognitive abilities have collectively been able to find the shortest path between a food source and the nest.

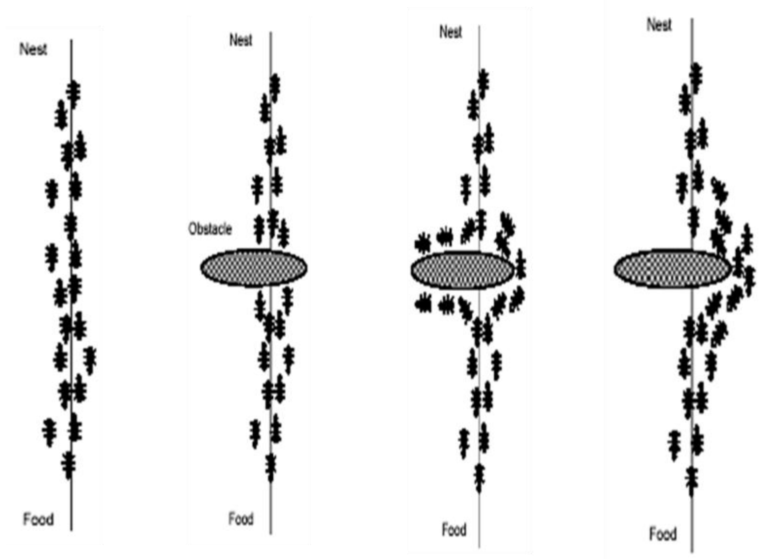

Fig.3. Ant system heuristics

1. The first ant finds the food source (F), via any way then returns to the nest (N), leaving behind a trail pheromone.
2. Ants indiscriminately follow possible ways, but the strengthening of the runway makes I more attractive as the shortest route.
3. Ants take the shortest route, long portions of other ways lose their trail pheromones.

In a series of experiments on a colony of ants with a choice between two unequal length paths leading to a source of food, biologists have observed that ants tended to use the shortest route. A model explaining this behavior is as follows:

1. An ant (called "blitz") runs more or less at random around the colony.
2. If it discovers a food source, it returns more or less directly to the nest, leaving in its path a trail of pheromone.
3. These pheromones are attractive, nearby ants will be inclined to follow, more or less directly, the track.
4. Returning to the colony, these ants will strengthen the route.
5. If there are two routes to reach the same food source then, in a given amount of time, the shorter one will be traveled by more ants than the long route.
6. The short route will be increasingly enhanced, and therefore become more attractive.
7. The long route will eventually disappear because pheromones are volatile.
8. Eventually, all the ants have determined and therefore "chosen" the shortest route.

Thus the basic philosophy of the algorithm involves the movement of a colony of ants through the different states of the problem influenced by two local decision policies, viz., trails and attractiveness. Thereby, each such ant incrementally constructs a solution to the problem. When an ant completes a solution, or during the construction phase, the ant evaluates the solution and modifies the trail value on the components used in its solution. This pheromone information will direct the search of the future ants.

VII. Conclusion And Future Work

For solving the load balance problem in a structured P2P system the first main contribution is an in-depth analysis of the effect of capacity and workload heterogeneity on algorithm performance in both static and dynamic environments and the qualitative relationship between static and dynamic environments. Our system is very adequate if the goals of the project are static in nature or do not change very often. Efforts are been made to have client interaction and include a dynamic nature of goals in the system development phase. We have successfully completed all the desired framework components as set at the beginning of the project. We wish that our effort will provide a backbone structure for the further enhancements of the system. To investigate the following important issues in the future is needed: It is intended to explore other cost reducing neighborhoods to further improve the DSLS algorithm. As the variance of a VS workload has a significant impact on the success ratio performance, the plan is to investigate VS merging and splitting strategies to enhance the performance of the algorithms. Also to perform a more in-depth study of issues in the dynamic scenario in which a node joins and leaves the system.

Acknowledgement

We take this opportunity to express our profound gratitude and deep regards to our guide Prof. Sumit Hirve for his exemplary guidance, monitoring and constant encouragement throughout the course of this thesis. The blessing, help and guidance given by him time to time shall carry us a long way in the journey of life on which we are about to embark. Lastly, we thank almighty, our parents and friends for their constant encouragement without which this assignment would not be possible.

Refernces

[1] C. Chyouhwa, T. Kun-Cheng, “The Server Reassignment Problem for Load Balancing In Structured P2P Systems, “IEEE Trans. Parallel and Distributed Systems, vol. 19, no. 2, Feb. 2008.
[2] Rao, K. Lakshminarayanan, S. Surana, R. Karp, I. Stoica,“Load Balancing in Structured P2P Systems,” Proc. Second Int’l Workshop Peer-to-Peer Systems (IPTPS ’03), Feb.2006.
[3] H.R. Lourenco and D. Serra, “Adaptive Search Heuristics for the Generalized Assignment Problem,” Mathware and Soft Computing, vol. 9, pp. 209-234, 2002.
[4] M. Dorigo and T. Stutzle, Ant Colony Optimization. MIT Press, 2004.