Semantic Routing for Structured Peer-to-Peer Networks

Luis Enrique Colmenares Guillén[1], Omar Ariosto Niño Prieto[2], Leandro Navarro Moldes[3]

[1] Benemerita Universidad Autonoma de Puebla,
Facultad de Ciencias de la Computación,
BUAP – FCC, Ciudad Universitaria,
Apartado Postal J-32,
Puebla, Pue. México.
lecolme@cs.buap.mx,

[2] Université Claude Bernard Lyon 1
Bâtiment Nautibus
43, Boulevard du 11 novembre 1918
69622 Villeurbanne Cedex France
OMAR.NINO-PRIETO@bvra.etu.univ-lyon1.fr

[3] Universidad Politecnica de Cataluña,
Jordi Girona, 1-3
Barcelona, España
leandro@ac.upc.edu

Abstract. During this work a simulator of a mechanism of searching in collaborative application is made to manage the content through the WWW with structured Peer-to-Peer networks. The principal contribution of this work is one mechanism that joins a metadata in the queries that are sent by the DHT Routing. The Semantic Routing proposed reduces the number of the average hops to reach an object or a document. This routing improves the Bamboo-DHT in the search of documents.

1 Introduction

The distributed computing introduces new challenges because it has new resources that show very different characteristics with heterogeneous architectures connected by distinct networks.

The increasing of Internet resources has been growth up, so the necessity to manage them has been initiated. This management has become an obligation of any distributed system that search the best functionality for future applications.

It is convenient to stand out that the negotiation of the resources consist to facilitate by the optimum way the available resources on Internet, databases, big stored systems or the files systems, among others. It is to be able to predict failures, network errors, traffic and other requirements.

The decentralization is important in applications where having global information is impossible. The decentralization gives to the networks or applications a high scalability and do not allow to have a single point of failure. The peers are autonomous and in some cases they are anonymous. The decentralization generates challenges in the security for different policies of management and dynamism.

The high dynamism of the participants requires new services. These services like the self-organization, replication, searching, among others. Those tasks give performance to the distributed applications in the context of collaboration group.

1.1 Content Management System

There are three important characteristics of the contents that can be used for the proposal of the management system with Peer-to-Peer (P2P) Networks:

1. The growing of the non structured data [1].
2. The management of the content of the network edge is considered an extension of the management of documents. The management of contents must be oriented to the reuse of the content.
3. The collaboration for Internet between the users of universities is frequent and useful in the actuality. The collaborative applications like BSCW [2], MOODLE [3] like others, are client-server in the WWW, the contents is in the server and the participants depend on the robustness of the server for having the active contents.

The idea of using P2P networks is not new in the management of contents. Several proposals exist for the management of the content in the companies [4] and some applications where the content is distributed by the audio, i.e. Gnutella [5].

2 The proposal requirements

The proposal is derived to offer solutions in the scenario that is in these three suggested dimensions, Fig 1:

1. The roles indicate the activity degree that the participants that have and their types of messages specifying protocols based on messages that allows guaranteeing a successful sending and reception of messages.
2. Distributes Hash Table (DHT): they are a weakly-connected_and strongly-connected this last, gives consistency the table routing. Today, the strongly-connected DHT have the lookup with O (log N) to O (log (1)).
3. During the earlier works, the data storage had been oriented from adjust of the system index to the file replication. The cache is emphasized to save temporally, during the restoration mechanisms of the routing tables until the defined roles like in JXTA [10].

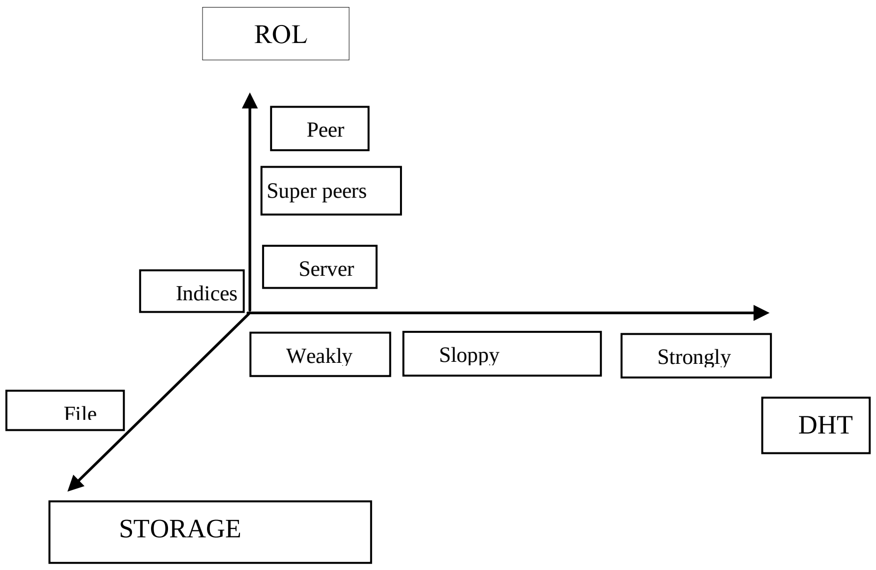

Fig. 1. The application dimensions

3 The Simulator proposal

In this simulator a brief description is made.

1. Classes and scripts with the configuration file. The design and the operation are explained.
2. The different events that the class DHT and semantic has.

3.1 Scripts and Classes with the file of configuration explaining the design and the running

In the programming guide, the design and style of the source code of bamboo [6] can be observed. Two stages or classes were created. The first class gives the number of hops through the DHT routing or the semantic routing. The second class is for locating the puts and generates the gets. The puts will place the objects (documents) through the bamboo nodes and the gets will find the request. Moreover, two scripts that have two classes.

The two Scripts are made in Perl. The first script is the responsible for creating the nodes type bamboo network, moreover of sending the java threads to the nodes of a cluster, balancing on each node in the cluster, the number of bamboo network nodes. For example, if there are 500 Bamboo-DHT nodes and we have ten nodes of a cluster (the nodes have dual microprocessors); we would have 50 bamboo nodes in each node of the cluster having a 25/CPU relation. This script calls the *name.cfg* configuration file and this also calls the DHT class and semantic class.

In the article [9], the chum rate is made with ranks of 8/seconds to 4/minutes. This means that following the experiment, the paper shows that every 8 seconds 8 nodes are disconnected and 8 nodes are connected maintaining the session time average in 0.72 minutes with 500 nodes. The highest rank is about 4 nodes that are disconnected every minute and four nodes that are connected during the same minute maintaining a session time of 86.64 minutes.

Example:

t_{med} = N ln 2/λ where N = number of nodes, with *churn-rate* from 8/seconds to 4/minutes with one t_{med} =1.4 minutes to 3 hours. In the experiment of [9], they have N = 1000 nodes. The estimation ln(2)/ 0.008 = 86.6433976 = 1.44 minutes. In our case N = 500 nodes, then 8/500 =0.016, ln 2/0.016 = 43.3216 = 0.7220 minutes.

In the formula ln(2)/λ . Where, λ= (4/60) / (1000) =0.00006666666. The total is = 10397.2181 = 173.28 minutes almost three hours. With N = 500, (4/60) / 500 = 0.00013333. ln(2)/0.0001333 = 5198.603867 = 86.643397 minutes.

The second script calls the class puts-gets that is located in the class DHT that is of bamboo. The puts follow the law of Zipf and they are made from parallel form in each node. The requests are made by each node that asks for objects with a balanced form. In other words, if there are 1000 requests and if there are 500 nodes, 20 requests are sent per node and each node will follow the Zipf law with random objects that the bamboo network has. Remember than the objects were put before by the puts class.

Using the DHT class calculates the number of hops using the Bamboo-DHT routing. Using the Semantic class, the number of hops is calculated using the semantic routing.

The limits can be changed, for example:

- The number of networks of the contents. It is relation with the distinct gateways of the beginning that will provide configuration file; this will generate different bamboo-network of content. This result that each bamboo network of content can make the next variations
    1. The number of nodes that learn: It is relation with the total number of bamboo nodes, we choose the random number and it will tell us the nodes which will learn, these change can be done with the class DHT or semantic.
    2. The number of objects: they are all the total documents that we can random put in the entire bamboo network; there the class puts/gets is found.
    3. Storage in each node: During the time that the requests are made, there exist a maximum number of objects that can store each node and it's found in the DHT semantic class.
    4. The requests following the Zipf law: The node that makes the request following the Zipf law. Making sure that the popularity of the objects is found in the puts/gets class.

Using the DHT class, calculates the number of hops using the Bamboo-DHT routing. Using the semantic class, the numbers of hops are calculated using the semantic routing.

Example of a configuration files in Bamboo, *name_file. cfg*. Each node of the bamboo network must have a fill of these characteristics

```
<sandstorm>
    <global>
    <initargs>
        node_id ${NodeID}
    </initargs>
    </global>

    <stages>

    <Network>
        class bamboo.lss.Network
        <initargs>
#               udpcc_debug_level 0
                drop_prob 0.1
        </initargs>
    </Network>

    <Rpc>
        class bamboo.lss.Rpc
        <initargs>
        </initargs>
    </Rpc>

    <Router>
        class bamboo.router.Router
        <initargs>
#         debug_level       1
          gateway                 ${GatewayID}
          periodic_ping_period    20
          ls_alarm_period         4
          near_rt_alarm_period    0
          far_rt_alarm_period     10
#           leaf_set_size           8
          digit_values            2
          ignore_proximity      false
            location_cache_size     0
        </initargs>
    </Router>

    <DataManager>
```

```
        class bamboo.dmgr.DataManager
        <initargs>
          debug_level           0
                merkle_tree_expansion 2
               desired_replicas       2
        </initargs>
    </DataManager>

    <StorageManager>
        class bamboo.db.StorageManager
        <initargs>
#         debug_level             0
          homedir       ${CacheDir}
        </initargs>
    </StorageManager>

    <DataManagerTest>
        class bamboo.dmgr.DataManagerTest
        <initargs>
 #               debug_level      0
                to_put           0
                put_size       100
        </initargs>
    </DataManagerTest>

        <Dht>
           class bamboo.dht.Dht
            <initargs>
                #debug_level 1
                storage_manager_stage StorageManager
                min_replica_count     1
            </initargs>
        </Dht>

        <Gateway>
           class bamboo.dht.Gateway
            <initargs>
               # debug_level 1
                port ${GatewayPort}
            </initargs>
        </Gateway>

    <WebInterface>
        class bamboo.www.WebInterface
        <initargs>
                storage_manager_stage StorageManager
        </initargs>
    </WebInterface>
```

```
#Routing dht_bamboo
    <StageSemantic_dht>
        class bamboo.cont_dht.StageSemantic_dht
        <initargs>
          debug_level 1
        </initargs>
    </StageSemantic_dht>

    </stages>
</sandstorm>
```

The example of the *name_file.cfg*, it is a structure in the following way: begins with < sandstorm > and concludes with </sandstorm>. In the global parameters the *node_id* contains the IP of the participant; it in this case, puts the IP of the node of the cluster and the port that distinguish each node. Stages that are bamboo classes are contained. Each stage contains parameters that allow controlling certain characteristics of a physical network. Certain stages were used to allow the creation of a bamboo network like in the reference [7]. The stage *Network* generates parameters of a network and the stage *Rpc* allows the remote procedure call. Router allows using the routes based on the bamboo-DHT considering two groups of neighbors (leaf-Set and table Routing). *DataManager* manages the number of replicas that can store when we use a Put. *StorageManager* allows keeping the data physically in the directory that is indicated. DHT calls to *StorageManager*. Gateway indicates that port used each node for the communication and the step of messages. The stage *WebInterface* allows to make call with the protocol *http* in a navigator to show the bamboo nodes. We conclude with *StageSemantic_dht*, this class allows using our algorithm which adds additional caching to each participant and allows choosing other possible routing.

For example: for a node that initializes the cluster that has IP, 196.0.0.1 the first node, the second 196.0.0.2 and so forth until the node 30 of the cluster, 196.0.0.30. If begins with the port number 3630. The first node 196.0.0.1:3630. The port 3631 reserves for the step of messages (gateway_port of the stage gateway of the file name_file.cfg) and the port 3632 is used to visualize in a graphic window the bamboo nodes. In ten bamboo nodes in a cluster node, would be made increasing of three in three each port and the address IP. Example: 196.0.0.1:3633, 196.0.0.1:3636... 196.0.0.1:3657.

The entire DHT networks have an origin. Bamboo uses the gateway of the stage Router, i.e. all the nodes belong to the bamboo network that is formed with the same gateway. In this case, has the port beginning of the first node 3630. If the bamboo network begins in the nodo1 of the cluster, all the nodes of our application have the following node gateway: 196.0.0.1:3630.

In the stage *StorageManager* to have a directory to locate the components of an object physically: value and content. Moreover, the directory /tmp is used.

This simulator allows that can be used the class java to scenarios with real nodes. It is transparent to the user and can expand the work to nodes PlanetLab [8]. Now, the open-DHT and bamboo are in the nodes PlanetLab.

The simulator works as follows: Once that execute the first script create nodes Bamboo-DHT network. The second script put to the objects in the all bamboo network. The objects are distributed in a uniform way in the bamboo network. The following step is send the class Gets and to do the requests of the objects. For example: suppose that are 100 objects and 50 nodes bamboo, a node requests an object for the first time, the algorithm calculates the number of hops and sends a message to make the copy to the node in its caching and this guarantees that if any node sends a requests the object and if during the journey have the object then has it reduces the number of hops. If makes the request again, now reduce in smaller number of hops because now already has the object.

3.2 The events that to have the class DHT or semantic

The events that use the DHT Routing (DR) or semantic Routing (SR) are:

1. Types of messages.
2. The payload or what is the message.
3. The different logs files. Format of Logs. The logs stay in each node bamboo:
    - Log of requests that indicates the objects that stay in the node.
    - Log of number of hops to reach an object. It registers the object number and the number of hops that it required to arrive to the object.
    - Log of the DHT, is when the bamboo-DHT is used to reach the object.
    - Log of average hops of an object, the maximum of hops of an object, minimum of hops of an object, and number of times of the object.
    - Log that counts number of total objects for each object. In this log, helps identify how many times they are requested each object, remember, that they are objects that are located in the bamboo network.
    - Log that counts the numbers of total hops.

4 Evaluation

4.1 Platform

The platform to measure our architecture this made in Bamboo-DHT [9] and the language scripting is Perl; the code is within a cluster of 16 nodes. The nodes of cluster have dual microprocessors AMD Opteron 64, 1.4 GHz, connected by Ethernet Gigabit, RAM memory of 2 GB a hard disk SCSI of 36 GB and OS fedora Linux Core 6x86-64. The simulations are execution-driven. All the file logs are stored in each participant of the network overlay P2P in run time.

4.2 Evaluation

For the evaluation three modules or classes in Bamboo were added. The first module, is caching, that assigns cache in each participant-DHT and allows the node to choose between two routings. The second module makes simultaneous requests in parallel. This allows making requests of queries from different participant nodes at the same time. The third module is the replication of content. This module will consist basically of the extension of the epidemic algorithm of Bamboo-DHT to update the contents in caching of each node.

4.3 Static Scenes

The evaluations are made both routing SR and DR. In Table 1, shows the parameter that was used for static scenes with Bamboo-DHT routing. The number of tests that were made was approximately 100 by each scene, although from test 45 the graphical one shows a uniform behavior with respect to the growth of the average of the number of hops.

The Table 2 shows the parameter that was used for static scenes with SR. A metadata in query of DHT is used; in addition caching that was made vary in the size of each node or participant. The number of objects is important, if are a small number of requests, 5, 10 or 20 requests. The objects will vary of 100, 250, 500, 750, and 1000 with duration of the object of one week (7*24*3600) or a day (24*3600) guaranteeing that the objects always are in the bamboo network. The communications between nodes of the cluster are stable in approximately 85% of the nodes. This guarantees 85 % of objects. In the number of requests: 10000, 12500, 15000, 20000, observed that the limits of the average are reduced. The size of caching used are: 5, 10, 15, 20. The use of caching represents use of DHT in SR. In scenes 1, 2, 3 of table 2, using small caching, would use of Bamboo-DHT. In experiments 4, 5, 6 used greater caching, would use less the Bamboo-DHT that is reflected in a smaller number of hops.

Table 1. Bamboo-DHT Routing parameters.

Scenes: Bamboo-DHT	No. Nodes (parameter 2)	No. Objects	No. Requests (parameter 1)	Without *caching*, without metadata
1	500	100	10000	-------------
2	500	250	12500	-------------
3	500	500	15000	-------------
4	500	750	15000	-------------
5	500	1000	20000	-------------

Table 2. Semantic Routing parameters.

Scenes: info-semantic (parameter 4)	Nodes (parameter 2)	Objects	Requests (parameter 1)	Size of caching (parameter 3)
1	500	100	10000	5
2	500	250	12500	5
3	500	500	15000	5
4	500	750	15000	10
5	500	1000	20000	15
6	500	1000	20000	20

In Fig. 2, the minimum, 3er quartile and the average of the number of hops by searching each object were found. The graph of Fig. 2 shows the following: The five more popular average objects, the total of objects represent 95% of the total requests. The requests are sent from 500 nodes of a total of 500 available nodes. Each node sends *n* requests of *m* available object. Each experiment of 500 nodes uses 500**n* requests. In *x*-axis, the objects are ordered from the most popular to the least popular, indicated by the named percentage and with their labels from left to right. In the *y*-axis, the number of total hops is shown.

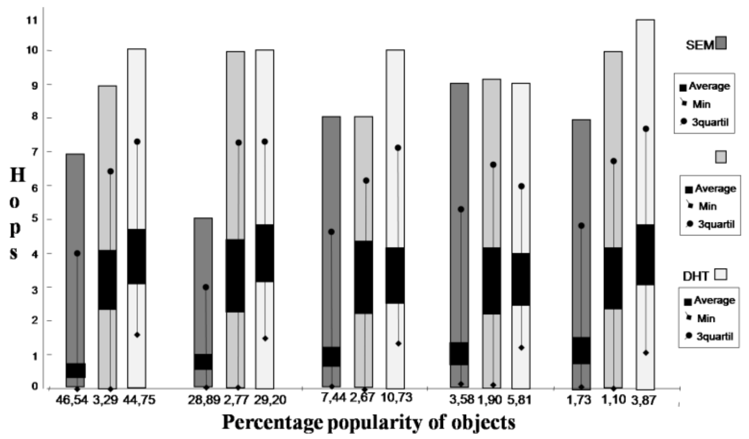

Fig. 2. Comparison of Routings.

In Fig. 2, there are five most popular objects with three bars, respectively. The two first bars of each object represent semantic routing proposed and it is a combination of both routings. The last bar of each object represents Bamboo-DHT routing without alterations. The bars are divided in quartiles; the maximum number of hops for a same object is the high part of the bar. The third quartile represents the superior part of the line that is within the bar, the average or second quartile represents the high part of the black picture and the first quartile is the minimum hop that was obtained by requests of a same object that represents the final part of the line. The average of hops diminishes when the object is more popular. We observed in the graph that improvement in the number of hops by DR and SR are the two initial bars. DR represents the third bar of each popular object; the average is over the two bars.

5 Conclusions and Future Work

The main contribution of this work is to improve the routing of the Bamboo-DHT, with semantic routing that reduces the number of hops average. The culmination of this work will have two contributions, first: 1. a mechanism of content delivery that allows possible routings to reach the content, when adding semantic routing in Bamboo-DHT, obtained a reduction in the number of hops averages to find an object and 2. A search mechanism using semantic information, giving expressivity to DHT, single-key was used and it does not affect the costs of the communication between nodes of DHT.

In future work, the last contribution will end with the class of bamboo that extends the broadcast algorithm, in addition, the evaluation in dynamic scenes, with parameters likes churn-rate and limitation of resources.

References

1. Robert Blumberg, Shaku Atre. The Problem with Unstructured Data. DM Review Magazine, February 2003.
2. BSCW (Basic Support for Cooperative Work) http://bscw.fit.fraunhofer.de/ (2009).
3. *Mo*dular Object-Oriented Dynamic Learning Environment (Moodle) http://moodle.org (2009).
4. “The emergence of distributed content management and Peer-to-Peer Content Networks”, by Gardner Consulting, January 2001.
5. T. Klinberg and R. Manfredi, http://www.gnutella.com/, 2009.
6. http://www.bamboo-dht.org/programmers-guide.html, 2009.
7. http://www.bamboo-dht.org/, 2009.
8. http://www.planet-lab.org/, 2009.
9. S. Rhea, D. Geels, T. Roscoe and J. Kubiatowicz. Handling churn in a DHT. In Proc. of the use USENIX annual technical conference, June 2004.
10. Juxtapose, JXTA. https://jxta.dev.java.net/ 2009.