A NEW MULTI-DIMENSIONAL HYPERBOLIC STRUCTURE FOR CLOUD SERVICE INDEXING

Telesphore Tiendrebeogo[1] and Oumarou Sié[2]

[1]Department of Mathematics and Computer Science, Polytechnic University of Bobo-Dioulasso, Bobo-Dioulasso, Burkina Faso
[2]Department of Computer Science, University of Ouagadougou, Ouagadougou, Burkina Faso

ABSTRACT

Nowadays, the structure of cloud uses the data processing of simple key value, which cannot adapted in search of closeness effectively because of the lack of structures of effective indexes, and with the increase of dimension, the structures of indexes similar to the tree existing could lead to the problem " of the curse of dimension". In this paper, we define a new cloud computing architecture in such global index, storage nodes is based on a Distributed Hash Table (DHT). In order to reduce the cost of calculation, store different services on the hyperbolic tree by using virtual coordinates thus that greedy routing based on hyperbolic space properties in order to improve query storage and retrieving performance effectively. Next, we perform and evaluate our cloud computing indexing structure based on a hyperbolic tree using virtual coordinates taken in the hyperbolic plane. We show through our experimental results that we compare with others clouds systems to show our solution ensures consistence and scalability for Cloud platform.

KEYWORDS

*Hyperbolic Geometry, Hyperbolic Tree, Data Indexing, Virus, Cloud Computing, Dist*ributed Hash Table.

1. INTRODUCTION

The recent appearance of cloud computing resolutely changed everybody is the perception of architectures of infrastructure, the software delivery and models of development. Thus, the Cloud computing is an emergent business solution. He can send the requirements of every software service to distribute the storage space and any sorts of the service on the association of resource. The user does not have to buy any material or software and flexible to update the quantity of resource according to their own real demand of the supplier. The system of Cloud produced the business opportunity and the future trend in the software industry. According to the studies of Merill Lynch [18], before 2011, the profit of the market of Cloud computing should be estimated to reach $160 billion, including $95 billion in the business and $65 billion in the on-line advertising. Because of the commercial potential of the platform of Cloud, more IT companies increase their investments in the Cloud develop. The infrastructures of Cloud of existing include the Blue Cloud of IBM [20], Google Map Reduce [10], Amazon's Elastic Computing Cloud (EC2) [6], etc. There is a number of characteristics associated with the summary of Cloud computing as follow: the variety of resources, central Internet, virtualization, the adaptability, the automatic adaptation, the optimization of resource, maintains SLAS (the Service-Level-Agreements) and the infrastructure SLAs [5]. The Cloud computing corresponds to a virtual IT resource with the possibility of maintaining and of managing remotely. From a structural point of view, the resource can be:

- IaaS (Infrastructure as a Service);
- PaaS (Platform as a Service);
- SaaS (Software as a Service).

Thus, it can understand for example, them for many structures of large-scale server cluster, including servers of calculation, servers of storage, the resources of bandwidth [26]. The platform of Cloud allows to manage both large number of computing resources and to store a large number of data. The allocation of resource is made in the platform of Cloud as a user considers that it uses an infinite resource. In this paper, we make the following contributions:

- We introduce a new structure of cloud computing using an hyperbolic tree in which each node is associated to a resource server and takes of virtual coordinates into hyperbolic space of the model of the Poincaré disk.
- We show how cloud infrastructures can communicate by greedy routing algorithm using [23].
- We show that this structure provides scalability and consistence in database services like data storage and retrieving.
- We perform some simulations and we show that our solution is comparable and sometimes better than others structures based on existing index structures, such as R-tree [15], k-d tree [14] and RT-CAN [30] which can support multi-attribute/multi-dimensional indexing or spatial indexing.

The remainder of this paper is organized as follows. Section II gives an overview onto the related works. Section III presents some properties of the hyperbolic plan as well as the model of the record of Poincaré. Section IV shows the process of addressing and indexing in our system of cloud. Section V defines the binding algorithm of our cloud system. Section IV presents a synthesis of the experimental evaluation which we have to realize to know the performances of our system and we conclude in Section VII.

2. RELATED WORK

The structure of indexation in the environment of cloud can be divided into one dimensional index and the multidimensional index. For single-dimensional data, we can build one dimensional index, as an index of B-tree and an hashed index. We propose evolutionary distributed B-tree to support the index of data of large scale data index in a cluster [2]. In this system, compute nodes can be divided into server nodes and client nodes. B-tree is distributed through servers, the client lazily replicates all the nodes of internal B-tree and the servers maintain synchronically a table of version of B-tree for the validation. The uses of scalable distributed B-tree is associated to distributed transactions to make changes to B-tree nodes, and B-tree nodes can be migrated on-line between servers for the load balancing. The main advantages of this index are the massive adaptability, the low cost, the fault tolerance and the possibility of management. But its weakness is that client's nodes lazily have to reproduce all the corresponding internal nodes, which incur the high maintenance cost. We propose a general frame of indexation for the system of cloud computing in [34] and there are three layers in this context. In the average layer, we supply computing resources to the users. To accelerate the queries processing, in every data stored is associated a local index built on every node. Afterward, a world index is establishes by choosing and by publishing a part of the local indexes in the overlay network. In the lower layer, processing nodes are organized in a structured overlay for routing and facilitating nodes joining and leaving. And in the top layer, it supplies an interface of access of data to user's applications based on the global index. The users can choose methods of access of different data as different questions. Another structure of hierarchical index is presented in [33] to process one dimensional data in the cloud. First of all, it builds an index of B-tree for the local data stored in every node.

Then, all the nodes are organized in a structured network BATON. Furthermore, he proposes an adaptive algorithm to choose the node of B^+-tree to publish according to models of question. This structure of index can support the range question and improve the efficiency of question. Both plans of index in [33, 34] use two levels of index and build the local index for the data stored in slave's nodes and global index in server nodes. During the request, they locate the local index through the global index and look for then in data in slave's nodes. Although these can reduce the time of question, they cannot support the multidimensional question effectively. Contrary to the structure of tree index, the bitmap index is slightly allocated by the growth of data, thus, we propose a mechanism of indexation with bitmap in [7]. Besides, there are the diverse types of systems of distributed storage for the management of the big quantities of data as the System of files [29] (GFS) Googles, which serves the applications of Google with an important volume of data. Big Table [8] establishes a type of system of distributed storage to manage the data structured by very large scales.

Also, there are some open source versions of Big Table and GFS for HDFS [27], the Hyper Table and HBASE, who indeed established a good platform adapted for the research and development. Yahoo supplied PNUTS [9], one accommodated, the system of database in the center controlled, parallel and distributed for the applications of Yahoo. These systems, data of break and constitute some fragments, spread then randomly the latter in groups to improve the parallelism of access of data. Some core servers working as routers are responsible for the orientation of questions in the nodes which contain results of question. In [28], we proposed the insertion roughly, the effective insertion of data in these systems. Contrary to this work, we propose an evolutionary mechanism using the model of the Poin caré disk which supplies the distributed storage of data and the algorithms of covering based on it in the hyperbolic space. Our plan of indexation is designed to route in a greedy way a big quantity of queries among a big group of nodes of storage by the hyperbolic coordinates using. The dynamo of the Amazon [12] is a store of easily available key-value based on the geographical reproduction and he can assure the final coherence. Every cluster in Dynamo uses a structure of ring to organize nodes, which supply the coherent hashing to retrieve data items. The coherent hashing proposed by the previous works is designed to support key-based data retrieval, but is not adapted to support the range and the multidimensional questions.

There are some solutions which support the processing of question on multidimensional data, as CAN (Content Addressable Network) [24]. It allows building a storage system of database by dividing rectangular zones. The Advertising distributed systems of storage as the Amazon S3 (the System of Simple Storage) and CloudDB of Microsoft tends to have small published details of implementation. Some other systems as Ceph [31], Sinfonia [3], etc., are designed to store objects and they aim at supplying the high performance in the object based retrievals instead of the set based database retrieval. Another solution as MapReduce [11] suggests dealing with large sets of data spread among clusters. MapReduce assigns mappers and reducers handle tasks, where mappers produces the intermediary succeeds parallel and reducers pull the intermediary result from mappers to make accumulations. The recent work, as [36] and [1], tries to integrate MapReduce into systems of database. These works constitute a parallel frame to deal with large sets of data. Our work has to build the second index of level overview, which can be also used on systems as GFS and MapReduce, among cluster nodes.

Besides, many algorithms of structured DHT (MSPastry [19], the Tapestry [35], Kademlia [16], CAN and Chord [4]) who supports multidimensional questions of range can be used to implement some services of cloud as Chord the Frame of Management of Session based for the Software as a service of Cloud (CSMC) [22], MingCloud based on the algorithm Kademlia [32], the Search on the Cloud is built on Pastry [25] and to improve fault tolerance, scalability and coherence.

Furthermore, Xiangyu proposed an Effective Multidimensional Index with the Cube of Knot for the system of Cloud computing [37] Zhang and others. Also the index of RT-CAN in their management system of data of Cloud in 2010 [30] was built by Jinbao Wang and others. These two plans are respectively based on k-d tree and R-tree.

Our work follows the framework proposed in [34]. However, to support multi-dimensional data, new routing algorithms and storage and retrieve query processing algorithms are proposed. Besides, our structure of indexation reduces the number of hops realized inside the system to index a service which is there or to facilitate the deployment of back-end database applications.

3. POINCARÉ MODELS OF HYPERBOLIC GEOMETRY

The first model of the hyperbolic plane that we will consider is due to the French mathematician Henri Poincaré. This model is called, Poincaré Disk Model. In this latter, the hyperbolic plane is represented by the open unit disk of radius 1 centered at the origin (coordinates associated with the origin are (0; 0)). In this specific model:

- Points within this open unit disk is associated to the points;
- Arcs of circles intersecting the disk and meeting its boundaries at right angles correspond to the lines.

In this model, we address the nodes by using complex coordinates. Thus, fact to be able to tile the hyperbolic space into polygons of any size that we called p-gon constitutes an important property. Every transformation in hyperbolic mosaic is in compliance with the notation {p, q} where every polygon has p sides and q vertex. There is a transformation in hyperbolic mosaic {p, q} for which every couple {p, q} obeyed in: $(p-2) \times (q-2) > 4$. In a tiling, p corresponds to the number of sides of the primal polygons (the black edges and blue vertices in Figure 1) and q is the number of sides of the polygons of the dual (the red triangles in figure 1). Our purpose is to partition the plane and address each node uniquely. We set p to infinity, thus transforming the primal into a regular tree of degree q. The dual is then tessellated with an infinite number of q-gons. This particular tiling splits of the hyperbolic plan in different spaces allows to build a tree which we use to address in a unique way the nodes of our system. Figure 1 illustrates an example with q = 3 of such a hyperbolic tree.

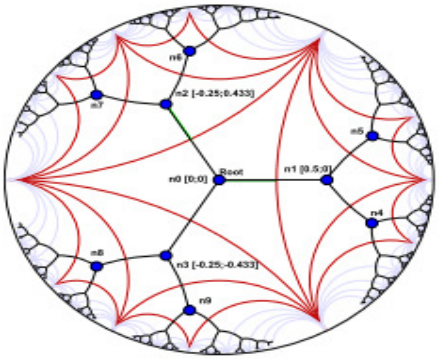

Figure 1. 3-regular tree in the hyperbolic plane.

In the model of Poincaré disc, the distances between two any points z and w are given by curves minimizing the distance between the latter. This curve is called geodesics of the hyperbolic plan. The determination of the length of geodesic between two points z and w allows to give the

hyperbolic distance d_H, we use the metrics of Poincaré which is invariable isometric given by the formula:

$$d_H(z, w) = \arg\cosh\left(1 + 2\frac{|z-w|^2}{\left(1-|z|^2\right)\left(1-|w|^2\right)}\right) \quad (1)$$

Greedy routing algorithm uses this formula to determine an optimal path in the routing process (see next section).

4. INDEXING PROCESS IN THE HYPERBOLIC PLANE

We now explain in this section how we create the hyperbolic addressing tree for database servers joins and how queries can be routed in our cloud structure. We propose here a dynamic and scalable hyperbolic greedy routing algorithm [21] that permits to index resources on our cloud. The first step in the creation of our system based on hyperbolic-tree of servers nodes is to start the first database server and to choose the degree of the addressing tree.

We notice that the hyperbolic coordinates (that is, a complex number) of a server node of the addressing tree are used as the address of the services server corresponding in our system of cloud. A server node of the tree can give the addresses corresponding to its children in the hyperbolic-tree. The degree of the tree determines the number of addresses which every services server will be capable of giving to new nodes servers for their connections. The degree of the hyperbolic-tree is defined at the beginning for all the lifetime of the cloud. The process of construction of the cloud is incremental, with every new services server joining one or several services servers already existing in the system. Over time, the servers shall leave the system until there is not a server anymore. This established fact will entail the death of the system. Thus, for every service which must be stored in the system, a Service IDentifier (SID) is associated with it and map then into pair key-value. The key will allow determining in which one of the servers of services will be stored (as in Section 5). The key will allow determining in which data servers the object will be stored (like in the section 5). Besides, when a service is deleted, the system must be able to update this operation in all the system by making follow the request in the direction of the table of storage. This method is scalable because unlike [13], we do not have to make a two-pass algorithm over the whole spatial system to find its highest degree. Also in our system, a server can connect to any other server at any time in order to obtain an address.

The first step is thus, to define the degree of the tree because it allows the construction of the dual, namely the regular q-gon. We fix the root of the tree at the origin of the primal and we begin the tilling at the origin of the disk depending of the q. The creation of sub-spaces separated pass by subdivision of the space. In this context, we are certain that half-spaces are tangent; where from the primal is an infinite q-regular tree. Our representation uses a theoretical infinite q-regular tree to build the greedy embedding of our hyperbolic tree. In this context, the degree of the regular tree corresponds to the number of sides of polygon used to build the dual (see Figure 1). In other words, a space is assigned for q services servers. Each repeats the same calculation principle in its own half of the space, and then the space is again assigned in q - 1 child from a level to other one of the tree. The distribution of the addresses of every node child is made in a half space which is associated with it. The algorithm 1 presents the process of servers' addresses computation as well as addresses of these various q children. In fact, the root of the hyperbolic tree possesses by default the coordinates (0; 0) which are associated with the address of the first services server of our system.

Algorithm 1 Calculating the coordinates of a database server's children.

1: procedure CALCCHILDRENCOORDS(server, q)
2: step ← argcosh(1/sin(π/q))
3: angle ← 2π/q
4: childCoords ← server.Coords
5: for i ← 1, q do
6: ChildCoords.rotationLeft(angle)
7: ChildCoords.translation(step)
8: ChildCoords.rotationRight(π)
9: if ChildCoords = server.P arentCoords then
10: STORECHILDCOORDS(ChildCoords)
11: end if
12: end for
13: end procedure

This distributed algorithm ensures that the database servers are contained in distinct spaces and have unique coordinates. Our algorithmic approach is adapted to the distributed and asynchronous computing in each of its steps. Our algorithm allows the allocation of address as hyperbolic coordinates in every element of the dynamic topology of our system. In our context, we consider, sub-tree by sub-tree. Therefore, the knowledge of all the nodes of the system is not necessary and every new server can obtain an address by asking a server of the system to be its parent. If the asked server has already given all its addresses, the new server must ask an address to another existing database server. When the server in the we ask an address has no it, because it already given every its addresses, the request is forwarded to another existing server of services in the system. When to a new server, system attributes an hyperbolic address, this last computes the addresses (i.e. hyperbolic coordinates) of its future children. At the same time as the construction of our system, we thus attend the construction of our structure of hyperbolic tree of addressing in an incremental way.

When a new server of service wants to connect to the servers) of services already inside the system and obtained an address of one of these servers of services, it can begin to send requests to store or lookup services in the system. The process of indexation is made on every server of services through the path (by leaving from the sender) by using the greedy routing algorithm 2 based on the hyperbolic distances between the servers. When a request is received by a server of resources, the latter calculates the distance of each of its neighbours in the destination and making follow the request step by step until the destination (we present destination server of resources computing in Section 5). If no neighbour is closer than the server itself, the query has reached a local minimum and is dropped.

Algorithm 2 Indexing process on our cloud system.

1: function GETNEXTHOP(servserver, query) return servserver
2: w = query.destinationServerCoords
3: m = server.Coords
4: $d_{min} = \arg\cosh\left(1 + 2\frac{|m-w|^2}{\left(1-|m|^2\right)\left(1-|w|^2\right)}\right)$
5: p min = server
6: for all neighbour ∈ server.Neighbours do
7: n = neighbour.Coords

8: $d = \arg\cosh\left(1 + 2\frac{|n-w|^2}{\left(1-|n|^2\right)\left(1-|w|^2\right)}\right)$

9: if $d < d_{min}$ then

10: $d_{min} = d$

11: p_{min} = neighbour

12: end if

13: end for

14: return p_{min}

15: end function

In a real network environment, link and server failures are expected to happen often. If the addressing hyperbolic tree is broken by the failure of a server of resources or a link, we reassign new addresses beyond the broken down server (some servers should reconnect at first to other servers to restore the connectivity). But, we are not treat this case in this paper.

5. NAMING AND BINDING PROCESS IN CLOUD SYSTEM

In this section we explain how our cloud system stores and retrieves the resources by using these latter names that is map to its address (virtual coordinates where it's possible to find resources servers). Our solution uses a structured Distributed Hash Table (DHT) system that with the virtual addressing mechanism of resource servers and the greedy routing algorithms presented in Section 4. At start-up, each new resource server uses a name that identifies the service (Application, Platform, and Infrastructure) that is shared in the system. This name will be kept by the resource server containing the service during all the lifetimes of the cloud. When the new resource server obtains an address, it stores the names of these services on different others resources servers. This storage uses the structured DHT of our cloud to store a fragment of key obtain by hashing of service name (explain in the follow). If the same sub-key is already stored in the cloud, an error message is sent back to the resource server containing concerned service in order to change this service name. Thus the DHT structure used in our cloud itself ensures that services names are unique.

A *(name, address)* pair, with the name mapping as a key is called a binding. Figure 2 shows how and where a given binder is stored in our cloud. A binder is any resource server that stores these pairs. The depth of a services server in the tree of hyperbolic addressing can be considered as the number of parental servers through whom we pass from root to server itself (including the root). In the creation of our cloud system, a maximal depth of the potential binder is chosen. This depth permits to define the maximum number of resources that can connect and share different services. Also the depth d is chosen such d minimize the inequality 2:

$$p \times \left(\frac{1-(p-1)^d}{2-p} \right) + 1 \geq N \quad (2)$$

With p the degree of our hyperbolic tree. Thus, this value is defined as the depth required from the hyperbolic tree.

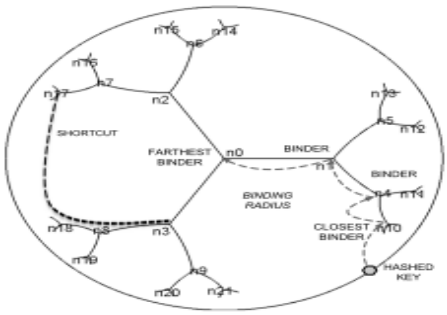

Figure 2: Hyperbolic cloud system.

To join the cloud, a new server of resource connects to other servers of resources and obtains an address of one of these servers of resources. Thus, each service name of the resource server is mapped into a key by hashing its name with the SHA-1 algorithm (SHA-1 gives 160-bit key). Next, the new resource server divides the 160-bit key into 5 equally sized 32-bit sub-keys (for redundancy storage). By linear transformation, the resource server selects the first sub-key and converts it to an angle. The angle is given by:

$$\alpha = 2\pi \times \frac{32 - bits \quad sub - key}{0 \times FFFFFFFF} \quad (3)$$

The resource server then computes a v representing a virtual point on the unit circle by using this angle:

$$v(x, y) \ \ with \ \begin{cases} x = \cos(\alpha) \\ y = \sin(\alpha) \end{cases} \quad (4)$$

Afterward, the server of resource computes the binder coordinates closest to the virtual point determined on the unit circle of Poincaré by using among others the depth of the tree. In Figure 2, we choose the depth value to three in the aim to do not overload Figure and make it incomprehensible. It's important to note that this closest service (name of binder) may not really exist. If no resource server currently owns this address. This latter then sends a store query (containing service name and address) to this closest resource server. This query is routed inside the cloud by using the indexing process describes in Section 4. If the request fails because the computed binder does not exist or because of the failures of node/link, it is redirected in the binder following the closest which is the father of the calculated binder. This process continues until the query reaches to the root resource server having the address (0; 0) (which is the farthest binder) or the number of resources servers is equal to (radial strategy):

$$S \leq \left\lfloor \frac{1}{2} \times \frac{\log(N)}{\log(q)} \right\rfloor \quad (5)$$

With N equal to number of distributed services servers, q to degree of hyperbolic-tree.

To reduce the impact of the dynamic (the leave or the join of the resource server also the adding or the deleting of the service) of the cloud, uses a redundancy mechanism that consist to increase the number of copies. The number of copies stored of a pair along the binding radius can be an arbitrary value (chosen in the creation of cloud) set at the system creation. Similarly the division

of the key in 5 sub-keys is arbitrary and could be increased or reduced depending on the redundancy (circular strategy) needed. To conclude, we can define two redundancy mechanisms (represented by Figure 3 and Figure 4) for storing copies of a given binding:

1. radial strategy of replication;
2. circular strategy of replication.

These mechanisms enable our cloud system to cope with a non-uniform growth of the system and they ensure that every will be stored by a replicated way in the aim to increase luck of success during the indexation of service.

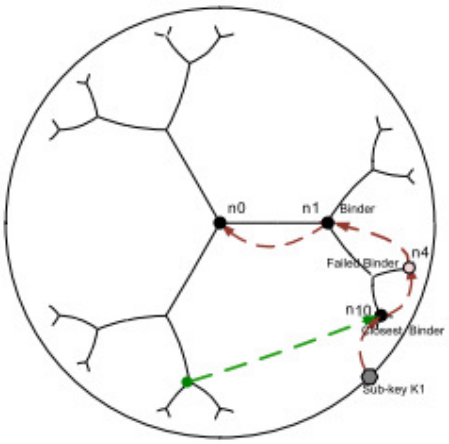

Figure 3: Radial replication in the cloud.

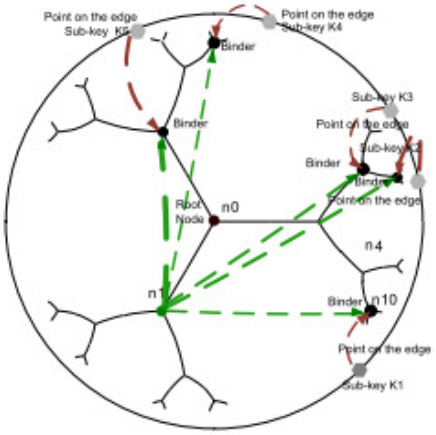

Figure 4: Circular replication in the cloud.

We present here a solution with the property of consistent hashing: if one resource server (or a service of the resource server) fails, only its keys are lost, but the other binders are not impacted and the whole cloud system remains coherent. As in many existing systems, the pairs will be stored through a hybrid soft mechanism and hard state mechanism. The creator of every service has to store periodically the pair (service_name, address) in the system to maintain its existence and show that the service is available. If not, it will have to be to delete by its binder. A message of deletion can be sent by the creator of every service to proceed to its removal of the storage table and consequently to an update of the existing services in the system.

6. SIMULATIONS

In the context of our experiments, we present the results of the simulations that we made it seen to show the practicability of our model as well as the scalability of our addressing system of services servers. Our cloud system is considered as dynamic (the phenomenon of churn is applied). We use the *Peersim* [17] simulator for cloud computing system simulation and it allows to service name following the uniform distribution. The study involved the following parameters of the DHT used in our cloud. These parameters are valid for all the DHTs that we compare:

- number of resources servers connected and used to store different kind of services is equal to 10000 in the starting up;
- we consider that our cloud system is dynamic and the rate of churn varying from 10% to 60%;
- we consider a simulation performed during 2 hours;

- the leaving and joining of the resources servers follows an exponential distribution as well as the inserting and the deleting of the services of the clouds associated;
- we suppose that the system receives 6 millions of queries (for access to different services) following an exponential distribution with a median equal to 10 minutes;
- we fix the maximum number of services by server to 2000.

6.1. Load Balancing Characteristics in Our System

Figure 5 shows an experimental distribution of points corresponding to the scatter plot of the distribution of resources servers in our cloud. Thus, we can mark that hyperbolic-tree of our cloud system is balanced. Indeed, we can noticed by part and others around the unit circle which we have resources servers. This has an almost uniform distribution around the root, what implies that our system builds a well-balanced tree what will more easily allow to reach a load balancing of storage.

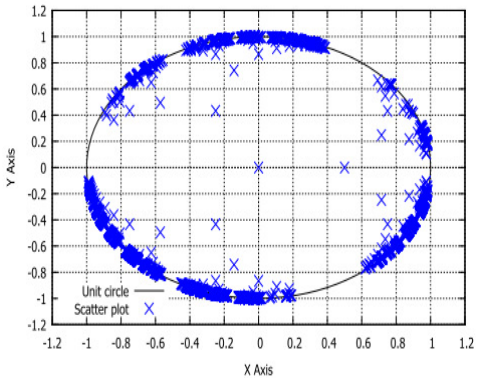

Figure 5: Scatter plot corresponding to the distributed

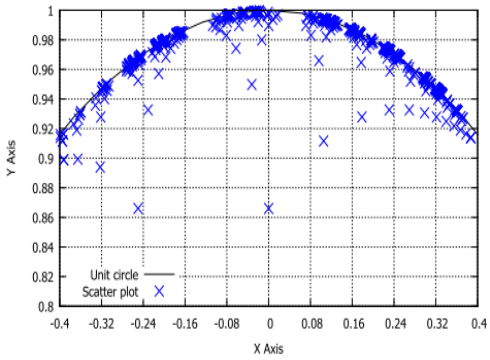

Figure 6: Distribution of database servers in the neighbourhood

6.2. Performances Analysis

Figure 7 shows that in a context where churn phenomenon is equal to 10%, we can notice that all the success rates are between 83% and 88% when the churn rate becomes 60%, according to the number of replication, the success rate is between 18% and 67%. This result indicates that the replication strategy permits to reduce impact of the churn phenomenon on our cloud performance.

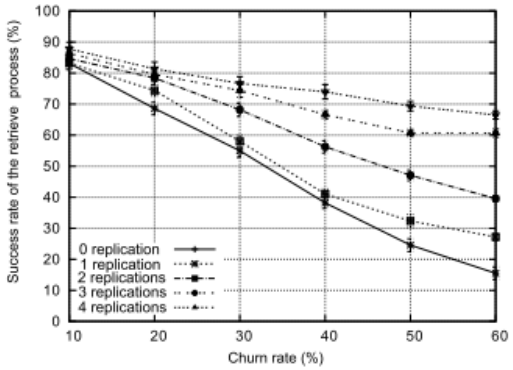

Figure 7: Evaluation of resilient strategy against the churn phenomenon.

Figure 8 indicates that when the churn rate is between 10% and 60%, the average number of hops to send a query from the source (resource server where the user is connected) to target (resource server that contains the cloud service searched) is approximately the same. Thus, through this result, we show that our system behaves well with compared with the others such as Chord, MSPastry and Kademlia. So the cloud that we propose that is based on this DHT algorithm have a good performance.

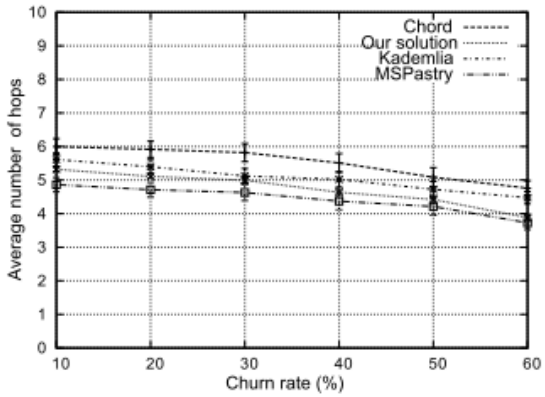

Figure 8: Comparison of the success rate depending to DHTs algorithms.

7. CONCLUSION

In this paper, we have presented the Cloud platform, providing scalable services is an essential requirement. There were few research reports proposed for multi-dimensional indexing schemes for a Cloud platform to manage the huge and variety services to address the complex queries efficiently. We have shown in this study that our solution using an hyperbolic-tree with virtual coordinates is consistent, scalable et hard because it support enough well churn phenomenon. Furthermore, our system gives good results than cloud system based on others DHTs algorithm. In the perspectives of our works, we shall analyze the incidence of the churn phenomenon issue and thus to develop a new multi-dimensional index structure that ensure a best result in a dynamic context in the aim to supply a variety of cloud services.

REFERENCES

[1] A. Abouzeid, K. Bajda-Pawlikowski, D. Abadi, A. Silberschatz, and A. Rasin. Hadoopdb: An architectural hybrid of mapreduce and dbms technologies for analytical workloads. Proc. VLDB Endow., 2(1):922–933, Aug. 2009.

[2] M. K. Aguilera, W. Golab, and M. A. Shah. A practical scalable distributed b-tree. Proc. VLDB Endow., 1(1):598–609, Aug. 2008.

[3] M. K. Aguilera, A. Merchant, M. Shah, A. Veitch, and C. Karamanolis. Sinfonia: A new paradigm for building scalable distributed systems. SIGOPS Oper. Syst. Rev., 41(6):159–174, Oct. 2007.

[4] N. Antonopoulos, J. Salter, and R. M. A. Peel. A multi-ring method for efficient multi-dimensional data lookup in p2p networks. 2005.

[5] H. B. and P. Y. X. An efficient two-level bitmap index for cloud data management. 2011.

[6] M. Balduzzi, J. Zaddach, D. Balzarotti, E. Kirda, and S. Loureiro. A security analysis ofamazon's elastic compute cloud service. In Proceedings of the 27th Annual ACM Symposium on Applied Computing, SAC '12, pages 1427–1434, New York, NY, USA, 2012. ACM.

[7] G. C., L. J., Z. Q., C. H., and G. Z. The characteristics of cloud computing. pages 275–279, 2008.

[8] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. Bigtable: A distributed storage system for structured data. In Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation Volume 7, OSDI '06, pages 15–15, Berkeley, CA, USA, 2006. USENIX Association.

[9] B. F. Cooper, R. Ramakrishnan, U. Srivastava, A. Silberstein, P. Bohannon, H.-A. Jacobsen, N. Puz, D. Weaver, and R. Yerneni. Pnuts: Yahoo!'s hosted data serving platform. Proc. VLDB Endow., 1(2):1277–1288, Aug. 2008.

[10] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. In Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6, OSDI'04, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association.

[11] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. Commun. ACM, 51(1):107–113, Jan. 2008.

[12] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: Amazon's highly available key-value store. SIGOPS Oper. Syst. Rev., 41(6):205–220, Oct. 2007.

[13] V. Gaede and O. G¨unther. Multidimensional access methods. ACM Comput. Surv., 30(2):170–231, June 1998.

[14] H. Garcia-Molina, J. D. Ullman, and J. Widom. Database System Implementation. Prentice-Hall, 2000.

[15] A. Guttman. R-trees: A dynamic index structure for spatial searching. SIGMOD Rec., 14(2):47–57, June 1984.

[16] X.-s. HOU, Y.-d. CAO, and Y. ZHANG. P2P multi-dimensional range query system based on kademlia. 2008.

[17] I. Kazmi and S. F. Y. Bukhari. Peersim: An efficient & scalable testbed for heterogeneous cluster-based p2p network protocols. pages 420–425, Washington, DC, USA, 2011. IEEE Computer Society.

[18] L. Merrill. The cloud wars: $100+ billion at stake. 2008.

[19] C. Miguel, C. Manuel, and R. Antony. Performance and dependability of structured peer-to-peer overlays. 2004.

[20] M. Naghshineh, R. Ratnaparkhi, D. Dillenberger, J. R. Doran, C. Dorai, L. Anderson, G. Pacifici, J. L. Snowdon, A. Azagury, M. VanderWiele, and Y. Wolfsthal. Ibm research division cloud computing initiative. IBM J. Res. Dev., 53(4):499–508, July 2009.

[21] F. Papadopoulos, D. Krioukov, M. Bogu˜n´a, and A. Vahdat. Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces. In Proceedings of the 29th Conference on Information Communications, INFOCOM'10, pages 2973–2981, Piscataway, NJ, USA, 2010. IEEE Press.

[22] Z. Pervez, A. M. Khattak, S. Lee, and Y.-K. Lee. Csmc: Chord based session management framework for software as a service cloud. In Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication, ICUIMC '11, pages 30:1–30:8, New York, NY, USA, 2011. ACM.

[23] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In Proceedings of the 9th Annual International Conference on Mobile Computing and Networking, MobiCom '03, pages 96–108, New York, NY, USA, 2003. ACM.

[24] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. SIGCOMM Comput. Commun. Rev., 31(4):161–172, Aug. 2001.

[25] R. Savage, D. T. Nava, N. E. Ch´avez, and N. S. Savage. Search on the cloud file system, 2011.

[26] Z. Shufen, Z. Shuai, C. X., and W. S. Analysis and research of cloud computing system instance. pages 88–92, 2010.

[27] K. V. Shvachko. HDFS scalability: The limits to growth. j-LOGIN, 35(2):??–??, apr 2010.

[28] A. Silberstein, B. F. Cooper, U. Srivastava, E. Vee, R. Yerneni, and R. Ramakrishnan. Efficient bulk insertion into a distributed ordered table. In Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, SIGMOD '08, pages 765–778, New York, NY, USA, 2008. ACM.

[29] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner. A break in the clouds: Towards a cloud definition. SIGCOMM Comput. Commun. Rev., 39(1):50–55, Dec. 2008.

[30] J. Wang, S. Wu, H. Gao, J. Li, and B. C. Ooi. Indexing multi-dimensional data in a cloud system. In Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data, SIGMOD '10, pages 591–602, New York, NY, USA, 2010. ACM.

[31] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. E. Long, and C. Maltzahn. Ceph: A scalable, high-performance distributed file system. In Proceedings of the 7th Symposium on Operating Systems Design and Implementation, OSDI '06, pages 307–320, Berkeley, CA, USA, 2006. USENIX Association.

[32] J.-y. Wu, J.-l. Zhang, T. Wang, and Q.-l. Shen. Study on redundant strategies in peer to peer cloud storage systems. 2011.
[33] S. Wu, D. Jiang, B. C. Ooi, and K.-L. Wu. Efficient b-tree based indexing for cloud data processing. Proc. VLDB Endow., 3(1-2):1207–1218, Sept. 2010.
[34] S. Wu and K. lung Wu. An indexing framework for efficient retrieval on the cloud. IEEE Data Engineering Bulletin, page 2009.
[35] Z. Y., H. Ling, S. Jeremy, C. R. Sean, D. J. Anthony, and D. K. John. Tapestry: A resilient global-scale overlay for service deployment. 2004.
[36] H.-c. Yang, A. Dasdan, R.-L. Hsiao, and D. S. Parker. Map-reduce-merge: Simplified relational data processing on large clusters. In Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, SIGMOD'07, pages 1029–1040, New York, NY, USA, 2007. ACM.
[37] X. Zhang, J. Ai, Z. Wang, J. Lu, and X. Meng. An efficient multi-dimensional index for cloud data management. In Proceedings of the First International Workshop on Cloud Data Management, CloudDB '09, pages 17–24, New York, NY, USA, 2009. ACM

AUTHORS

Telesphore Tiendrebeogo PhD degree in Computer Networks from the University of Bordeaux, France, in June 2013. His research was in the peer-to-peer Networks. He obtained Master Degree in Multimedia and Internet Information System from the Jules Verne University of Picardie, France, in 2008 and an other Master in Real Time Information System, in Polytechnic University of Bobo-Dioulasso, in 2007. He is now Assistant Professor in Computer Science Department, Polytechnic University of Bobo-Dioulasso, Burkina Faso and member of the laboratory of analysis in mathematics and computer science of the University of Ouagadougou. His research interests include the peer-to-peer networks , Distributed Hash Table (DHT), and database system management and his indexing.

Oumarou Sié PhD at the University of Rennes I France. He is Leader of the Research Team LTIC Department of Computer Science, Professor at the Department of Science and Technical of University of Ouagadougou . Advisor at the Regulatory Authority for Electronic Communications of Burkina-Faso.