An Integration of Overlay and Physical Network Architecture Resilient to any Network Disruptions

Naziya Husna
M. Tech 4th sem Dept. of CSE
T. John Institute Technology
Bangalore, India

Dr. T. R Mahesh
Professor & head, Dept. of CSE
T. John Institute of Technology
Bangalore, India

***Abstract*— In an overlay-based parallel networking architecture, data is distributed among the servers and the data is processed by servers employing the overlay network which can achieve high scalability. However, it is difficult to provide services in overlay-based parallel architecture when there is a physical network disruption occur that is caused by routers. To overcome this issue our proposed architecture is designed based on the integration of overlay and physical networks and overlay network topology for maximizing the connectivity against server breakdowns. Overlay network construction and task allocation schemes are proposed for maximizing the service availability against physical network disruption. In the super node(SN) we will add one cache memory which stores the query details and the node that contains the required resource so that the search time can be reduced or sometimes not needed. We have used multiple source node concept where one node will be provided the resource by multiple nodes at a time. Each node will provide some part of the resource so that the requested node will get the node in less time. Further we are enhancing to improve the service by selecting the servers in each group based on the types of services like multimedia, sports, documents etc. Once any resource is updated in one server then, which are the other servers are providing the similar services in other group, will be updated.**

Keywords*— *Overlay Network, neighbour selection, physical network disruption, service availability, task allocation.

I. INTRODUCTION

The existing architecture provides parallel data mining architectures such as MapReduce [3] and Hadoop [12] to fulfill these requirements. In those architectures, the data processing is executed by distinct nodes (called processing nodes) but system management task is served by a central node. While such a management scheme simplifies the design and implementation, this scheme lacks scalability because the central management overseen by a central node may decrease the system performance when the number of nodes increases [7]. Additionally, since the central node(called as master node) is a single point of failure, the service availability can dramatically decrease when the central node ceases to function. From these reasons, scalability and service availability are critical issues for parallel overlay network architecture.

As a remedy for improving scalability, an overlay-based parallel overlay network architecture has been proposed. Since all the nodes execute both management and processing functions by using overlay network, this architecture can balance the management load. Additionally, this architecture achieves higher service availability against the breakdown of central node because it keeps providing the overlay network until overlay network is disrupted.

However, this architecture cannot ensure the service availability against physical network disruption (e.g., router breakdown due to hardware trouble or DDoS attacks) [1]. The

physical network disruption does not only lead to the cease of function of the damaged router but also disrupts the communications of the servers, which are connected with the damaged router. In other words, numerous nodes are removed from the overlay network by the physical network disruption.

To deal with the above-mentioned problem, we propose an overlay-based parallel overlay network architecture that is tolerant to physical network disruption. Our proposed architecture is designed based on the integration of overlay and physical networks for maximizing the connectivity against server breakdowns.

Furthermore, we are enhancing to improve the service by selecting the servers in each group based on the types of services like multimedia, sports, documents, text files etc. Once any resource is updated in one server then, which are the other servers are providing the similar services in other group, will be updated and then will add one cache memory which stores the query details and the node that contains the required resource so that the search time can be reduced or sometimes not needed.

II. RELATED WORK

The distributed networks such as Peer-to-Peer (P2P) networks and grid networks have attracted much attention due to their scalability[1]. While the distributed networks have the advantage of allowing the node(s) to join or leave the network easily, the issue of lack of resiliency to both attacks and faults still remains. Here Distributed Mechanism take place where a method that construct a network following bimodal degree distribution[1], which is robust to

deal with both attacks and faults and can achieve higher resilience compared with other existing networking approaches.

Here they study the energy efficiency problem for such MapReduce clusters in private cloud environments that are characterized by repeated, batch execution of jobs. Here we are discussing about Energy efficiency problem, the energy efficiency problem for such MapReduce[2] clusters that are characterized by repeated, batch execution of jobs. Simultaneously improving job performance. Only one process taken at a time.

MapReduce is a programming model and an associated implementation for processing and generating large datasets[3]. Users specify the computation in terms of a map and a reduce function, and the under- lying runtime system automatically parallelizes the computation across large-scale clusters of machines. Functional model with user-specified mapreduce[3] operations allows us to parallelize large computations easily. It hides the details of parallelization, fault tolerance, locality optimization, and load balancing .

Designed and implemented the Google File System, a scalable distributed file system for large distributed data-intensive applications. It provides fault tolerance while running on inexpensive commodity hardware, and it delivers high aggregate performance to a large number of clients[4]. While sharing many of the same goals as previous distributed file systems, This has led us to re-examine traditional choices and explore radically different design points[4]. Here Checking Mechanism of chunk files take place and it Provides fault tolerance by constant monitoring. File system interface extensions designed to support distributed application treating component failures as the normal rather than the exception.

To get better routing efficiency, a new routing algorithm named Double-Layer Ring Structured Topology(DLRT) was presented here[5]. In DLRT, nodes were dynamically divided into a number of clusters which used sub-super nodes to manage, and a short routing table was defined at constant level. The route table maintaining algorithm in DLRT for nodes entering and exiting was presented, and the clusters building method and distributed election algorithm for sub-super nodes was presented too. Improves the performance in regard to routing table maintaining, routing hops, and network delay[5].The DLRT has great high fault tolerance the disadvantage is that the systems require high bandwidth, scalability, and a long start time and the network requires a very long time and large bandwidth.

Load imbalance in a distributed file system, that is, the file chunks are not distributed as uniformly as possible among the nodes[6]. Emerging distributed file systems in production systems strongly depend on a central node for chunk reallocation. This dependence is clearly inadequate in a large-scale, failure-prone environment because the central load balancer is put under considerable workload that is linearly scaled with the system size, and may thus become the performance bottleneck and the single point of failure. Here, a fully distributed load rebalancing algorithm is presented to cope with the load imbalance problem[6]. We present a load rebalancing algorithm for distributing file chunks as uniformly as possible and minimizing the movement cost as much as possible. Taking advantage of physical network locality and node heterogeneity. Allocate their resources on-demand without sophisticated deployment and management of resources.

In Monitoring File System master node can complete only a few thousand operations per second. MapReduce may have a thousand tasks wanting to open a number of files[7]. Initial conception of Google did not include new file system no other choice, so GFS born – Monitoring, error detection, fault tolerance, auto recovery all part of file system. Anticipated throughput requirements necessitated changing traditional assumption – I/O operations and block sizes – Scalability.

One way of achieving this goal is to optimize the execution of Mapreduce jobs on the cluster[8]. For a set of production jobs that are executed periodically on new data, they can perform an off-line analysis for evaluating performance benefits of different optimization techniques. In this work, they consider a subset of production workloads that consists of MapReduce jobs with no dependencies. They observe that the order in which these jobs are executed can have a significant impact on their overall completion time and the cluster resource utilization. the goal is to automate the design of a job schedule that minimizes the completion time (make span) of such a set of MapReduce jobs[8].

Building on this new wave of mobile devices are personal computing activities such as micro blogging [9], social networking, and photo sharing, which are intrinsically mobile phenomena that occur while on the go. Mobility is now propagating to more professional activities such as data analytics, which need no longer be restricted to the workplace. In fact, the rise of big data increasingly demands that they are able to access data resources anytime and anywhere, whether to support decisions and activities for travel, telecommuting, or distributed team work[9].

Many systems take the form of networks—sets of vertices joined together by edges—including social networks, computer networks, and biological networks. A variety of models of networks have been proposed and studied in the physics literature, many of which have been successful at reproducing features of networks in the real world. However, there is an important element missing from these models: many networks show ‘‘assortative mixing’’[10] on their degrees, i.e., a preference for high- degree vertices to attach to other high-degree vertices, while others show disassortative mixing—high-degree vertices attach to low-degree ones.

III. EXISTING PARALLE DATA MINING ARCHITECTURE

Here, we introduce the parallel data mining architecture based on the centralized management mechanism. Then we describe the existing works that aim to improve the service availability, followed by the shortcomings of these existing schemes. Moreover, we describe an overlay based parallel overlay network architecture that can overcome the weakness of the conventional architecture.

A. Conventional Parallel Data Mining Architecture

MapReduce is the most popular architecture for parallel data mining[12], [4]. In MapReduce, servers are classified into two types of nodes, i.e., a single central node and multiple processing nodes. While the central node schedules mapping and reduction processes and manages file name space operations (i.e., open, close, and rename), the processing nodes store data and execute mapping and reduction processes.

When a data processing request is injected, the central node partitions the task into some data blocks, which are distributed to distinct processing nodes. Then, each processing node (called mapper) performs the mapping process, which classifies a large amount of information and picks out the information required for the next process. After the mapping process, the central node selects a reducer, which performs the

reduction process, from mappers. The reducer integrates the information extracted in the mapping process and outputs the analyzed results.

Since mapping and reduction processes are executed in distributed manner, MapRecue can execute the data mining at the speed proportional to the number of servers. Additionally, valuable existing works conducted in [8]–[15] developed high performance parallel data mining architectures in terms of processing speed, network resource efficiency, computational resource efficiency, and energy efficiency. Despite the significant advantages, those architectures still suffer from server breakdowns because the success probability of overlay network decreases when the servers fail due to hardware troubles or software bugs [16], [17].

To cope with this issue, the common MapReduce architecture (e.g., current Hadoop [18]) replicates each data block and distributes the replicated ones to distinct nodes, which increases the service availability against server breakdowns.

Additionally, current Hadoop utilizes multiple central nodes mechanism to increase service availability against the breakdown of central node. However, it is difficult to ensure the service availability under real environment since the optimal numbers of replications and central nodes depend on the probability and scale of breakdowns.

The works [19], [20] proposed processing scheduling technique that can shorten execution time of the data mining under failure-prone environment. However, because these works assume the scale of server breakdowns is small, the capability of overlay network is dramatically decreased when a larger scale of breakdown, such as physical network disruption, occurs. Therefore, a parallel overlay network architecture that is tolerant to physical network disruption is absolutely imperative to provide future “ubiquitous big overlay network service”.

B. Overlay-based Parallel Data Mining Architecture

Overlay-based parallel data mining is one of architectures that needs to improve the service availability against server breakdowns [6]–[22]. In this architecture, all the servers execute both management and processing functions. The overlay network is constructed by all servers and utilized to find processing nodes, similar to the central nodes in the conventional architecture. This architecture can keep providing the service even if some nodes are removed from the overlay network.

When a data processing request is injected, a node that received the requests executes a reception function by using the overlay network. In other words, the node finds mappers by using flooding message, where mappers are randomly selected. Then, a mapper that initially finished the mapping process becomes a reducer, and it requests to other mappers to transmit the processed data to itself, where the request message can be forwarded by using flooding scheme.

After receiving the processed data from mappers, the reducer executes the reduction process and outputs the analyzed result.

In this architecture, since the connectivity of overlay network dramatically affects the service availability of overlay network, there are numerous works, which tackled the connectivity issue from the various viewpoints, i.e., context-aware, graph theory based, and complex network theory based overlay network construction schemes [23]–[5]. These works make overlay networks that are tolerant to small-scale server breakdowns but do not consider the large-scale server breakdowns, i.e., physical network disruption. Therefore, this paper develops an overlay-based parallel overlay network architecture that is tolerant to physical network disruption so that overlay network is available at anytime, anywhere.

IV. OVERLAY BASED PARALLEL NETWORKING ARCHITECTURE

Here, we proposes a overlay-based parallel network architecture to improve the service availability against server breakdowns and physical network disruption by utilizing physical network information. First, we introduce an overlay network topology following a bimodal degree distribution for maximizing connectivity against server breakdowns. Then, we propose a neighbor selection scheme in order to improve the number of available nodes after physical network disruption occurs. Furthermore, a task allocation scheme, which succeeds in overlay network under physical network disruption is proposed. In the task allocation process we are intern allocating service by making two servers in each group and allocate service to each group (services like; multimedia, documentation, text files, sports etc).

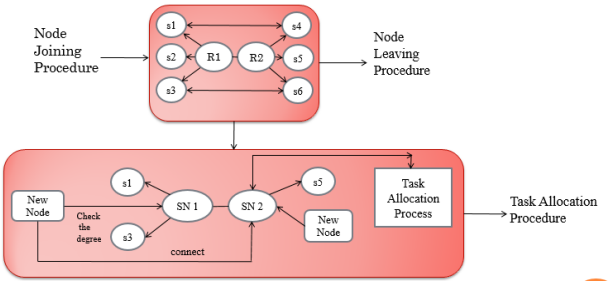

Fig 1. Integration of Overlay and Physical network

A. *Overlay network topology based on bimodal degree distribution*

We introduce an optimal overlay network topology that is tolerant to server breakdowns caused by hardware troubles and DDoS attacks. To achieve high connectivity against both hardware troubles and DDoS attacks, this paper focuses on overlay Based on the optimal bimodal degree distribution, an optimal network topology for maximizing connectivity against server breakdowns has been developed [5]. Fig. 2 shows an optimal network topology where The LNs are classified into Normal Leaf Nodes (NLNs), which connect with an SN, and Extra Leaf Nodes (ELNs), which do not connect with an SN but connect with other ELNs. This topology is divided into multiple smaller groups. Each group network following a bimodal degree distribution.

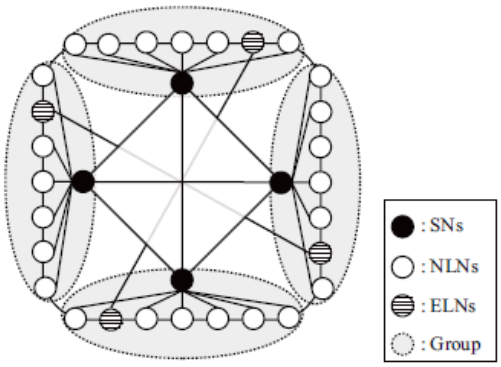

Fig. 2. An optimal network topology which is tolerant to server breakdowns.

B. *Physical network aware neighbor selection*

Since the neighbor selection scheme affects the connectivity of overlay network, we propose a neighbor selection scheme to construct an overlay network that achieves higher connectivity against physical network disruption. The physical network disruption has a specific characteristic such as "locality", i.e., servers connecting with the malfunctioning router are removed from the overlay network. Therefore, it is desirable that the servers that are connecting with the same router (or located in the same area) in the physical network become neighboring nodes (or belong to same group) in the overlay network, as shown in Fig. 3. In this neighbor selection principle, most of the links of the removed nodes are also the links to other removed nodes. In other words, this scheme can achieve higher connectivity against the physical network disruption because there remain a lot of links between the surviving nodes.

In order to construct an overlay network based on our neighbor selection principle and optimal network topology, we

propose two procedures: (i) node joining procedure and (ii) Service maintenance procedure. The node joining procedure is autonomously executed by a newly joined node (NJN) to distribute the management load to all nodes. In the node joining procedure, an NJN selects the servers that are located in the same area in the physical network as neighboring nodes in the overlay network. On the other hand, the network maintenance procedure, which reconstructs network to keep the optimal network topology.

Group construction process – The objective of this process is to construct new groups with the increase in the number of nodes. First, the NJN selects a node from the biggest group as a new SN. The ideal degree of the newly created SN is calculated according to (1). Then, the ESN constructs a new group by dividing the AG of the newly created SN into two smaller groups evenly.

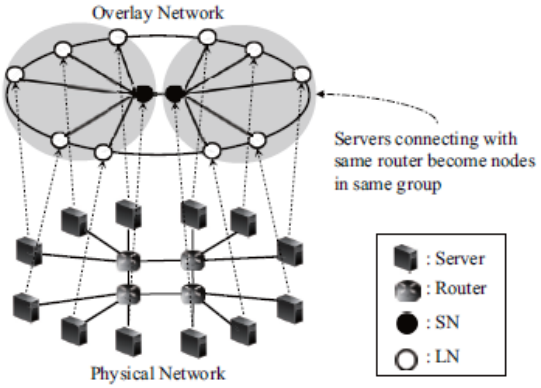

Fig. 3. An overlay network constructed based on the proposed neighbor selection principle.

Algorithm 1:Node Joining Algorithm

1. Select group g as Affiliation Group(AG)
2. Check the degree of SN in group g
3. If (degree<other SNs degree)
4. then join the node to that group
5. get intra-group list L^{g}_{intra} which is shared in group g
6. establish connection with nodes of group g
7. get intra-group list $L^{g'}_{intra}$ which is shared in group g'
8. establish connection with LNs of group g'
9. update L^{g}_{intra} and $L^{g'}_{intra}$
10. else
11. join to the other group g
12. repeat
13. end if

Regrouping process – In this process, the ESN restructures the groups so that the size of each group is the same as others.

C. Physical network aware task allocation

While the proposed neighbor selection scheme achieves higher connectivity of the overlay network and increases the available number of nodes against physical network disruption, the overlay-based overlay network architecture fails to output processing result when all mappers that have same data block are removed.

Algorithm 2: Service Allocation/Distribution Algorithm

1. Select a group g€G
2. Select a server s from group g
3. Allocate a Service to s
4. if (d>0)
5. then select a server s' belonging to group g'
6. Allocate a service to s'
7. d←d-1 , G←G-{g'}
8. if d>0
9. Repeat the same
10. end if
11. end if

V. SERVICE AVAILABILITY ANALYSIS

In this section, we mathematically analyze the service availability of overlay-based parallel overlay network architecture after physical network disruption.

A. *Node removal probability*

The physical network disruption causes numerous nodes to be removed from the overlay network and the probability that a node is removed differs depending on neighbor selection schemes. Therefore, we model the node removal probability in overlay networks that is constructed based on the existing and proposed neighbor selection schemes, respectively. Here, we define the node removal probability, fk, which denotes the probability that a node with degree k will be removed from the overlay network.

$$f_k^{\text{rand}} = \frac{N^R}{N}.$$

B. *Giant cluster ratio*

Since the node removal affects the degree distribution of overlay network, we derive the degree distribution after physical network disruption, p k, by using the node removal probabilities, which are modeled in previous subsection.

A cluster that has maximum number of nodes after physical network disruption is referred to as the "giant cluster". The giant cluster ratio, Gc, is defined with the number of nodes in a giant cluster, Ngc, and the total number of nodes in the original overlay network, N, as follows.

$$G_c = \frac{N_{\text{gc}}}{N}.$$

C. *Success probability of overlay network*

In the parallel processing architecture, each task is partitioned into some data blocks, which are replicated and distributed to distinct mappers, and a reducer successfully executes reduction process if it receives at least one replication of each data block from the mappers. Therefore, the probability that a overlay network task is successfully processed, P success, is decreased by the node removal due to physical network disruption. P success is expressed with the number of partitioned data blocks, B, the number of replicas, D, and the probability that there exists a node that has replication i of partitioned data block j in giant cluster, ai, j , as follows.

$$P_{\text{success}} = \prod_i^B \left\{ 1 - \prod_j^D (1 - a_{i,j}) \right\}.$$

TABLE I
A List of Notations used in Task Allocation Procedure.

G	Group List
{g,g',…}	Number of groups of group G
d	Number of replicas of each server
{s, s',..}	Number of Servers

VI. ASSESSMENT OF PHYSICAL NETWORK DISRUPTION EFFECT ON SERVICE AVAILABILITY

In this section, we aim to investigate the effect of physical network disruption on the service availability of overlay network. Additionally, we confirm the effectiveness of our proposed architecture in comparison with existing architecture that are designed without considering physical network, i.e., neighboring nodes are randomly selected and data blocks are distributed in a random manner. In this evaluation, we show the number of available nodes and number of tasks that are successfully processed in order to verify the effectiveness of the proposed neighbor selection and task allocation schemes, respectively. Mathematical expressions in previous section are used for our performance evaluation.

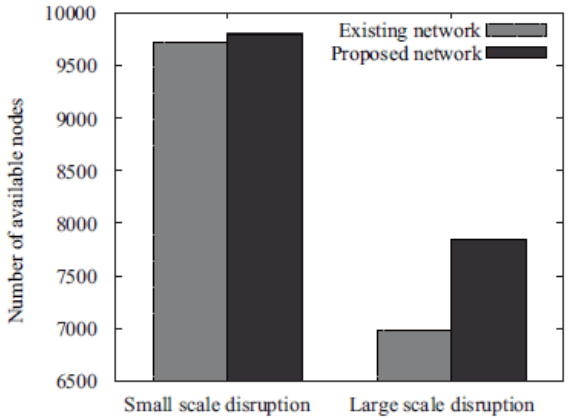

Fig. 4. The number of available nodes in different physical network disruption scenarios.

We suppose that the physical network follows power-law degree distribution, which is a well-known fact, and its topology is a tree structure, where the number of nodes including servers and routers is set to 104. The overlay

network is constructed by all nodes and follows the bimodal degree distribution, where the degree of leaf nodes is set to 3. We suppose that a processing task is partitioned into 5 data blocks and the total number of processing tasks is 103. We evaluate the performance of data processing after different types of physical network disruptions occur.

A. *Performance comparison of neighbor selection schemes*
In order to verify the effectiveness of the proposed neighbor selection scheme, we evaluate the number of available nodes after a physical network disruption occurs in two overlay networks, i.e., (i) overlay network that is constructed based on the proposed neighbor selection scheme (shortly referred to as the "proposed network"), and (ii) overlay network that is constructed based on the random neighbor selection scheme (shortly referred to as the "existing network"). We suppose two kinds of physical network disruption, i.e., small scale disruption (where approximately 2% of nodes are removed from overlay network) and large-scale disruption (where approximately 20% of nodes are removed from overlay network). Fig. 4 depicts the number of available nodes in different physical network disruption scenarios. While the number of available nodes in the existing network represents the lower value, the proposed network achieves maximum number of available nodes regardless of the physical network disruption scenarios. This is because the proposed overlay network is not disrupted since the removed nodes are located in the same area. Moreover, the proposed network attains much better performance when the number of removed nodes increases. Therefore, we can confirm the effectiveness of the proposed neighbor selection scheme.

B. Performance comparison of task allocation schemes
In the remainder of this section, we verify the effectiveness of the proposed task allocation scheme by comparison of existing scheme in terms of the number of successful task after physical network disruption. While the proposed task allocation replicates 2 times, the number of replicas is set to either 2, 3, or 4 in existing task allocation. In the both cases, the overlay network is constructed based on the proposed neighbor selection scheme.

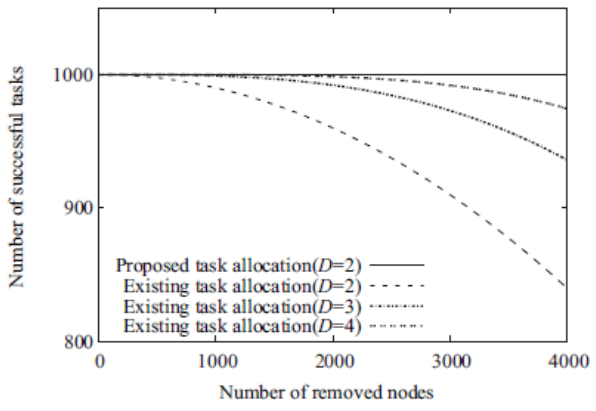

Fig. 5. Impact of the number of removed nodes on the number of successful tasks.

Fig. 5 demonstrates the number of successful tasks when the number of removed nodes by physical network disruption is varied from 0 to 4000. The existing task allocation scheme falls to an extremely low availability with a progressive increase of number of removed nodes even if the number of replicas increases. On the other hand, the proposed task allocation scheme achieves 100% success probability of overlay network with minimum replications regardless of the number of removed nodes because it ensures existence of the nodes that have a data block for processing in giant cluster. It can be concluded that the overlay-based overlay network architecture with the proposed neighbor selection and task allocation schemes can execute big overlay network with higher success rate and lower processing cost.

VII. CONCLUSION & FUTURE ENHANCEMENT

An overlay-based overlay network architecture, which fully distributes management and processing functions by using overlay network technologies, can potentially provide scalable
overlay network in large-scale network. However, due to physical network disruption, this architecture dramatically decreases service availability of overlay network. To solve this problem, we proposed neighbor selection and task allocation schemes based on integration of the overlay and physical networks. In order to improve the success probability of overlay network against physical network disruption, our neighbor selection scheme constructs overlay network based on node location in physical network and our task allocation scheme selects nodes from different diagonally-cornered groups in the overlay network as servers

In the super node(SN) we will add one cache memory which stores the query details and the node that contains the required resource so that the search time can be reduced or sometimes not needed. We have used multiple source node concept where one node will be provided the resource by multiple nodes at a time. Each node will provide some part of the resource so that the requested node will get the requested node in less time. Further we are enhancing to improve the service by selecting the servers in each group based on the types of services like multimedia, sports, documents etc. Once any resource is updated in one server then, which are the other servers are providing the similar services in other group, will be updated.

REFERENCES

[1] K. Suto, H. Nishiyama, X. S. Shen, and N. Kato, "Designing P2P networks tolerant to attacks and faults based on bimodal degree distribution," *Journal of Communications*, vol. 7, no. 8, pp. 587–595, Aug. 2012.
[2] M. Cardosa, A. Singh, H. Pucha, and A. Chandra, "Exploiting spatiotemporal tradeoffs for energy-aware MapReduce in the cloud," *IEEE Transactions on Computers*, vol. 61, no. 12, pp. 1737–1751, Dec. 2012.
[3] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," in *Proc. of 6th Symposium on Operating Systems Design and Implementation*, San Francisco, USA, Dec. 2004, pp. 137–150.

[4] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," in *Proc. of 19th Symposium on Operating Systems Principles*, New York, USA, Oct. 2003, pp. 29–43.
[5] K. Suto, H. Nishiyama, N. Kato, T. Nakachi, T. Fujii, and A. Takahara, "THUP: A P2P network robust to churn and dos attack based on bimodal degree distribution," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 247–256, Sept. 2013.
[6] H.-C. Hsiao, H.-Y. Chung, H. Shen, and Y.-C. Chao, "Load rebalancing for distributed file systems in clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 5, pp. 951–962, May 2013.
[7] K. McKusick and S. Quinlan, "GFS: Evolution on fast-forward," *ACM Queue Magazine*, vol. 7, no. 7, pp. 1–11, Aug. 2009.
[8] A. Verma, L. Cherkasova, and R. H. Campbell, "Orchestrating an ensemble of MapReduce jobs for minimizing their makespan," *IEEE Transactions on Dependable and Secure Computing*, vol. 10, no. 5, pp. 314 327, Sept.-Oct. 2013.
[9] N. Elmqvist and P. Irani, "Ubiquitous analytics: Interacting with big data anywhere, anytime," *IEEE Computer Magazine*, vol. 46, no. 4, pp. 86–89, Apr. 2013.
[10] M. E. J. Newman, "Assortative mixing in networks," *Physical Review Letters*, vol. 89, no. 208701, May 2002.
[11] F. Gebara, H. Hofstee, J. Hayes, and A. Hylick, "Big data text oriented benchmark creation for hadoop," *IBM Journal of Research and Development*, vol. 57, no. 3/4, pp. 10:1–10:6, May-Jul. 2013.
[12] "Hadoop [Online]." Available: http://hadoop.apache.org/.
[13] Y. Zhang, Q. Gao, L. Gao, and C. Wang, "Priter: A distributed framework for prioritizing iterative computations," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 9, pp. 1884–1893, Sept. 2013.
[14] D. Warneke and O. Kao, "Exploiting dynamic resource allocation for efficient parallel data processing in the cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 6, pp. 985–997, Jun. 2011.
[15] M. Asahara, S. Nakadai, and T. Araki, "Load Atomizer: A locality and I/O load aware task scheduler for MapReduce," in *Proc. of IEEE 4th International Conference on Cloud Computing Technology and Science*, Taipei, Dec. 2012, pp. 317–324.
[16] A. Rabkin and R. H. Katz, "How hadoop clusters break," *IEEE Software Magazine*, vol. 30, no. 4, pp. 88–94, Jul.-Aug. 2013.
[17] H. Jin, X. Yang, X.-H. Sun, and I. Raicu, "Large-scale distributed systems at google: Current systems and future directions," in *Keynote Speech at the 3rd ACM SIGOPS International Workshop Large Scale Distributed Systems and Middleware*, Montana, USA, OCT. 2009.
[18] "HDFS Fereration [Online]." Available: http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/Federation.html.
[19] Q. Zheng, "Improving MapReduce fault tolerance in the cloud," in *Proc. of IEEE International Symposium on Parallel & Distributed Processing, Workshop and Phd Forum*, Atlanta, USA, Apr. 2010, pp. 1–6.
[20] J. Dean, "Adapt: Availability-aware MapReduce data placement for nondedicated distributed computing," in *Proc. of 32nd IEEE International Conference on Distributed Computing Systems*, Macau, Jun. 2012, pp. 516–525.
[21] F. Azzedin, "Towards a scalable HDFS architecture," in *Proc. of International Conference on Collaboration Technologies and Systems*, San Diego, USA, May 2013, pp. 155–161.
[22] J. Zhang, G. Wu, X. Hu, and X. Wu, "A distributed cache for hadoop distributed file system in real-time cloud services," in *Proc. of ACM/IEEE 13th International Conference on Grid Computing*, Beijing, China, Sept. 2012, pp. 12–21.
[23] Z. Yao, X. Wang, D. Leonard, and D. Loguinov, "Node isolation model and age-based neighbor selection in unstructured P2P networks," *IEEE/ACM Transactions on Networking*, vol. 17, no. 1, pp. 144–157, Feb. 2009.
[24] W. Xiao, M. He, and H. Liang, "Cayleyccc: A robust P2P overlay network with simple routing and small-world," *Academy Publisher Journal of Networks*, vol. 6, no. 9, pp. 1247–1253, Sept. 2011.
[25] P. Flocchini, A. Nayak, S. Member, and M. Xie, "Enhancing peer-topeer systems through redundancy," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 1, pp. 15–24, Jan. 2007.
[26] T. Tanizawa, G. Paul, R. Cohen, S. Havlin, and H. E. Stanley, "Optimization of network robustness to waves of targeted and random attacks," *Physical Review E*, vol. 71, no. 4, Apr. 2005.