BIG DATA NETWORKING: REQUIREMENTS, ARCHITECTURE AND ISSUES

Mohammed S. Al-kahtani

Dept. of Computer Engg., Prince Sattam bin Abdulaziz University, Saudi Arabia

Abstract

A flexible, efficient and secure networking architecture is required in order to process big data. However, existing network architectures are mostly unable to handle big data. As big data pushes network resources to the limits it results in network congestion, poor performance, and detrimental user experiences. This paper presents the current state-of-the-art research challenges and possible solutions on big data networking theory. More specifically, we present the state of networking issues of big data related to capacity, management and data processing. We also present the architectures of MapReduce and Hadoop paradigm with research challenges, fabric networks and software defined networks (SDN) that are used to handle today's idly growing digital world and compare and contrast them to identify relevant problems and solutions.

Keywords

Big Data; MapReduce; Hadoop; SDN; Fabric Networks

1. INTRODUCTION

In today's times, applications are constantly increasing the rate that data is growing in the world. Studies have shown that by 2020 the world will have increased 50 times the amount of data it had in 2011, which was currently 1.8 zettabytes or 1.8 trillion gigabytes of data [18]. The basic reason for the sharp increase in data being stored over the years simply comes down to cost of storage.

The IT industry has made the cost of storage so cheap that applications are capable of saving data at exponential rates. This brings the challenge of having existing network infrastructure learn how to manage and process this big data so that it can be utilized into useful information [1, 5 - 6, 16].

Many big data applications work in real-time. Hence, these applications need to create, store and process large amount of information which produces a great deal of volume and demand on the network. When looking at data from a networking perspective, many different areas are needed to be explored. These include network topology optimization, parallel structures and big data processing algorithms, data retrieval, security, and privacy issues [13]. The topic of big data is still a new exciting area of research among the IT community and will be requiring much attention for the years to come.

According to network theory, big data can be described as any aggregate or data-in-motion and many of its applications have real-time needs to be effective especially in an education/research environment [8]. In the industry, common applications use for big data include analyzing the data

DOI: 10.5121/ijwmn.2016.8604

to come up with efficient conclusions in the application domains such as astrophysics, particle physics (e.g., CERN research), biological science (e.g., healthcare), geological science, social networking (e.g., Facebook), trend prediction (e.g., google flu), and optimizing route delivery (e.g., UPS package delivery, fuel tracking) [13].

A typical organization has a limited network infrastructure and resources capable of handling these volumes of traffic flows which cause regular services (e.g., Email, Web browsing, video streaming) to become strained. This can reduce network performance affecting bandwidth and exposing hardware limitations of devices such as firewall processing being overwhelmed [13].

This paper presents a comprehensive survey on the network theory of big data. This work starts by introducing the concept of big data and networking theory by giving some background information on the state of networking issues related to capacity, management and data processing in Section II. In Section III briefly presents the architecture of Big Data which includes the MapReduce paradigm and Hadoop distributed architecture, fabric network infrastructure and software defined networks (SDN). We present architectures, designs and techniques using theorized solutions to handle today's idly growing digital world and compare and contrast them to identify relevant problems and solutions. Finally, we discuss, evaluate, and state conclusions on the analysis of the defined Big Data networking concepts in Section IV.

2. BIG DATA REQUIREMENTS

This section briefly presents the network, data types, and data flow requirements of Big Data.

2.1. Big Data Network Requirements

Big data requires big performance demands on a networks infrastructure which means the network needs to be resilient, consistent, and have some form of application awareness. Ideal big data network architecture must be designed with a distributed architecture in mind in order to deal with the availability of distributed resources all while simultaneously working in parallel on a single task [7]. Big data applications begin by processing large volumes of information which becomes larger as the data replicates across the network. Table 1 presents a comparison of the traditional and big data types [11].

Table 1: Traditional vs. Big data types

Feature	Traditional Data	Big Data
Data type	Structured	Partial structure or unstructured
Data Volume	Giga/Terabytes	Peta/Exabytes
Data Relationship	Simple and Known	Complex and Unknown
Data Model	Used fixed schema	No schema
Network Model	Centralized/Distributed	Only Distributed

The most demanding part of this process is not just the size of information, but the ability for the network to break larger jobs into smaller ones in order to work with the data effectively [12]. The

six main set of network metric requirements used to evaluate whether a network can appropriately handle big data are:

Network resiliency: the availability of a network is critical to communication of distributed resources and nothing can happen without it. Building a network that is perfectly available is impossible, but we can try to create one that resilient to failures. The core idea behind this strategy is implementing path diversity and failover [7]. Path diversity refers to being able to travel a different route when trying to get between resources. Failover is having the ability to recognize issues quickly then switching over to an alternate path.

Network congestion mitigation: many big data applications obviously need to process large amount of data in real-time which requires their data flow methods to work in a type of "burst mode", where a small set of a large data amount may be sent across the network for immediate analysis [8]. This can cause congestion problems which in turn cause queuing delays, dropped packet issues, and retransmission triggers [8]. Network architecture must take into account mitigating congestion through variant path diversity so the network traffic is able to fan out across many different paths in order to reach the resources.

Network performance consistency: typically, big data networks are not affected by network latency that is measured in seconds or thousands of nanoseconds because it does not really affect the big data application processing time [7]. The more important issue to big data applications is maintaining high synchronicity. The different jobs of these applications need to be executed in parallel in order to assist in accurate analysis, so any large decreases in network performance may trigger failures in the outcome.

Network future scalability: As big data becomes more popular it is important to design network infrastructure with the ability to add future nodes to the bid data cluster environment. According to a study done by the Hadoop wizard in 2013, the average number of nodes is approximately 100, but yahoo runs 42,000 nodes, and these node environments increase every year [7].

Network partitioning: Implementing a parallel infrastructure approach to big data networks is essential in creating an environment that can handle both big and regular traffic efficiently. The common "burst" data flow detrimental effects found in big data applications can be avoided by separating the networks so they reduce the impact of every-day network services. If this architecture is implemented, the network will have to monitor multiple tenants and jobs in order to maintain performance, compliance regulation, and audit/logging methods. Network partitioning may take the form of a logical separation (e.g., Virtualization) or physical separation depending on the requirements of the organization [7].

Network application awareness: Big data network architectures are typically based using clustered environments so the nodes can build large data sets. Different applications require different requirements, for example, some may need high bandwidth while others are latency sensitive [7]. If a network is to support different big data applications requirements and multiple tenants, then it needs to have the ability to differentiate, separate, and process the various workloads independently. This coexistence of data flows, processes, and application resource factors all must be considered when creating a big data network that must function in a common environment.

2.2. Data Type Requirements

The characteristics of big data types are very different from that of traditional data types. Big data applications that require real-time performance requires the large data sets be broken down into small incremental sizes for processing and analyzing. This system does not perform well in typical online transaction processing (OTP) or when using popular SQL analysis methods [11]. Instead, big data needs a more flexible flat horizontally scaled database environment that is capable of handling various unstructured data types with complex and unknown data relationships amongst each other in distributed network architecture. Some older databases models have attempted to be modified to create NewSQL, but most have abandoned SQL for new models such as NoSQL [8]. These new databases make it possible to utilize big data's analytical tools and capabilities. Figure 1 demonstrates the throughput comparison of RDMS and No SQL database over the volume of data.

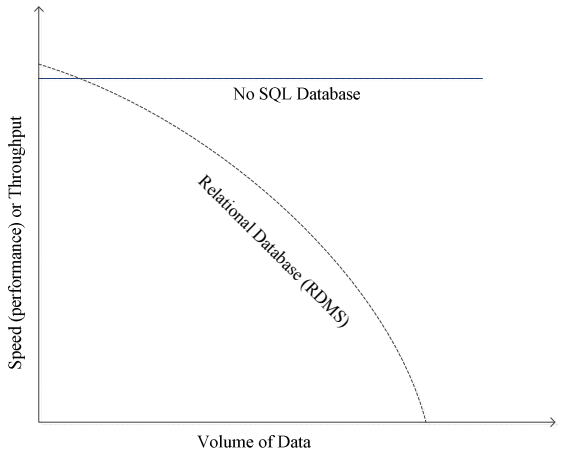

Figure 1: Comparison of scalability of RDMS and NoSQL

2.3. Network Flow Requirements

Typical network infrastructure over the years has been created in a three-tier architecture consisting of computing, storage, and networking. Basically the direction of network traffic is referred to as "north-south" which means it usually begins at the end user then goes down the integrated stack to the server than to the database. With the implementation of big data in networks, the traffic pattern needs to change. Big data architecture is based on a horizontal scale and uses a distributed approach among the nodes, so the traffic demands between the server and storage nodes are much higher than the typical data flow between servers and end users. Data is no longer just being created by external sources, but are generated from a number of devices and applications. This type of traffic flow is referred to as machine-to-machine/virtual machine network traffic or "east-west" and it needs to be supported when creating a high-performance scalable big data network.

3. BIG DATA ARCHITECTURE

In this section, we present architecture and research challenges of MapReduce and Hadoop [2,3,4] that are used for big data processing. Then, we present the fabric network topology [19] used in big

data infrastructure along with its advantages and limitations. Finally, we briefly present software defined networks (SDN), which is very popular for big data processing.

3.1. MapReduce and Hadoop

MapReduce is the core component of the Hadoop Apache software framework and is a type of programming model that can be implemented in variety of languages (e.g., Java, C++) that is used for processing big data [17]. This type of software tool can divide applications into smaller fragments or blocks which are then sent out to nodes in a cluster or map. It uses a map function that will able to filter, sort, and distribute jobs to various nodes and also uses a reduce function to collect the results from those jobs so they can resolved into a single value to be used for efficient analysis (Figure 2).

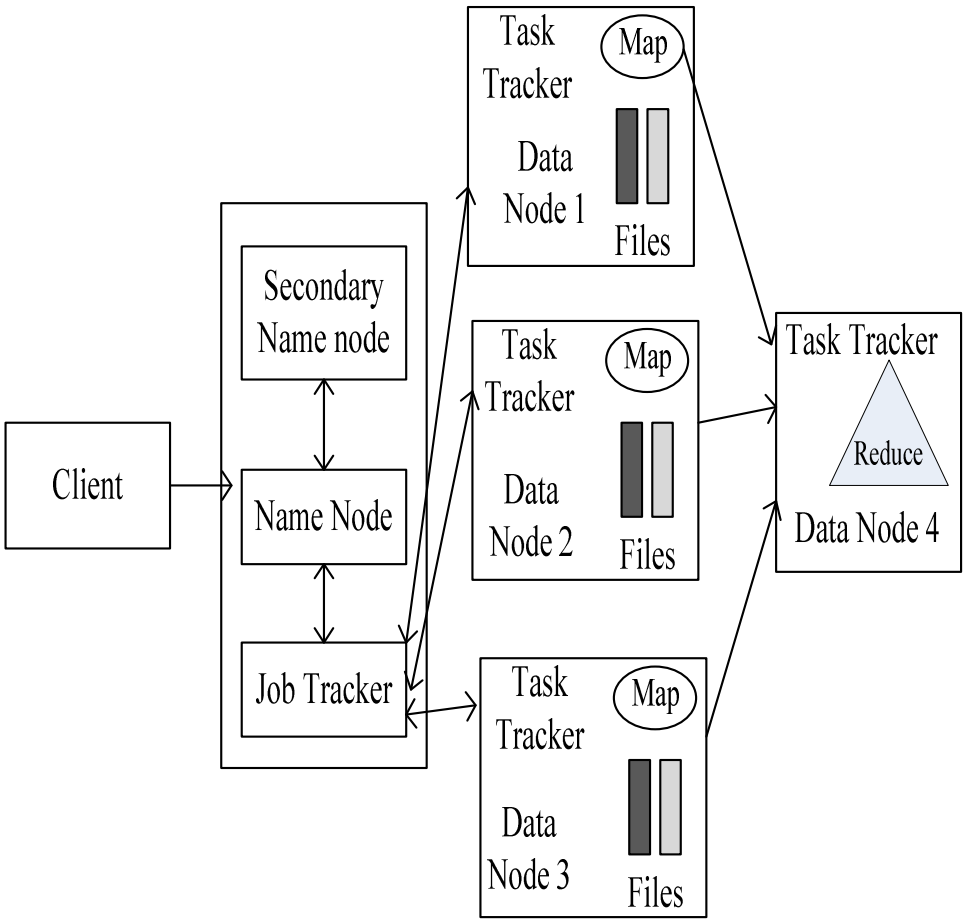

Figure 2: Hadoop based on MapReduce paradigm

The MapReduce consists of a job tracker, task trackers, and sometimes a job history server [23]. The job tracker is used as the master node that is in charge of managing resources and jobs. The task tracker is used to be deployed to each node in order to run the map and help with some of the cluster task load. The job history server is used to track finished jobs and can be deployed as an

independent function. MapReduce operates in parallel across vast cluster sizes, while jobs can be divided across many different servers [17]. MapReduce has fault-tolerance where each node sends status updates to the master node, who can re-assign jobs to functioning nodes in cases of node failure.

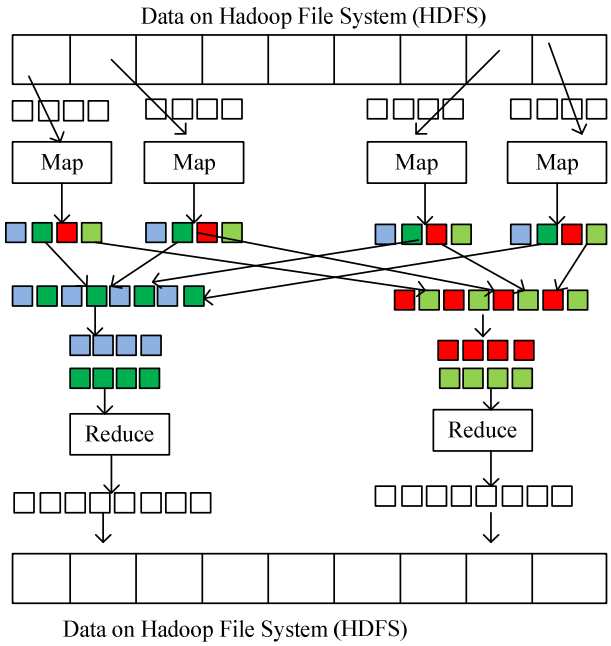

Figure 3: Hadoop that reduces data and processes in parallel

On the other hand, Hadoop is a management software framework that plays an important role in big data analytics. Hadoop is capable of cataloging, managing, distributing, and querying unstructured large data sets rapidly across many nodes within a distributed network environment. It uses a Hadoop distributed file storage system (HDFS) for storage that divides data into blocks which is distributed and stored on multiple nodes. In order to process the data, Hadoop uses MapReduce to break down data for the nodes to process and sort in parallel (Figure 3). The map procedure is responsible for filtering and sorting and the reduce procedure focuses on summary tasks. It supports high speed transfer rates and is capable of resilient uninterrupted operation in situations when there is node failure [20]. The infrastructure divides up the nodes into groups or racks. Hadoop is an excellent framework for applications using large search engines such as Google and Yahoo.

Together Hadoop and MapReduce provide the current popular choice for implementation of big data infrastructure [10]. A typical Hadoop network structure consists of a slave (data node, task tracker), master (name node, job tracker) and a client [9]. The client is basically the user interface or query engine. The data nodes are used as storage for the data that contain smaller databases systems and are horizontally distributed across the network. The task tracker is used to process the broken down fragments of the task that has been distributed to a node. The name node maintains a location index of all the other nodes found in the network, so it knows where the specific data is located in which data node. The job tracker is the software job tracking

mechanism that is used to transfer and aggregate request search queries (tasks) through to the task tracker nodes so the end user can perform information analysis on the result.

Research Challenges in Hadoop-MapReduce Paradigm

One of the main issues with this infrastructure is that when data is broken into smaller elements, so the nodes can store that data and communicate the data to the overall Hadoop cluster through designated rack switches, this process will obviously increase latency throughout the network [14]. Many big data networks begin to suffer from performance bottleneck issues because of the additional hop counts that increase between the client, job tracker, and data nodes when the data must travel through one rack switch to another rack switch as illustrated in Figure 4

Big data and many cloud architectures [24, 25] demand much more performance from connecting nodes as opposed to the connecting clients. When dealing with big data, a single client interaction can produce thousands of node to node interactions in order to get the intended result needed. "The traditional client/server architecture that has been implemented in the last 20 years cannot handle the burden of these new big data functioning requirements because it assumed the client supplied much of the computational overhead, rather than the backend infrastructure" [11].

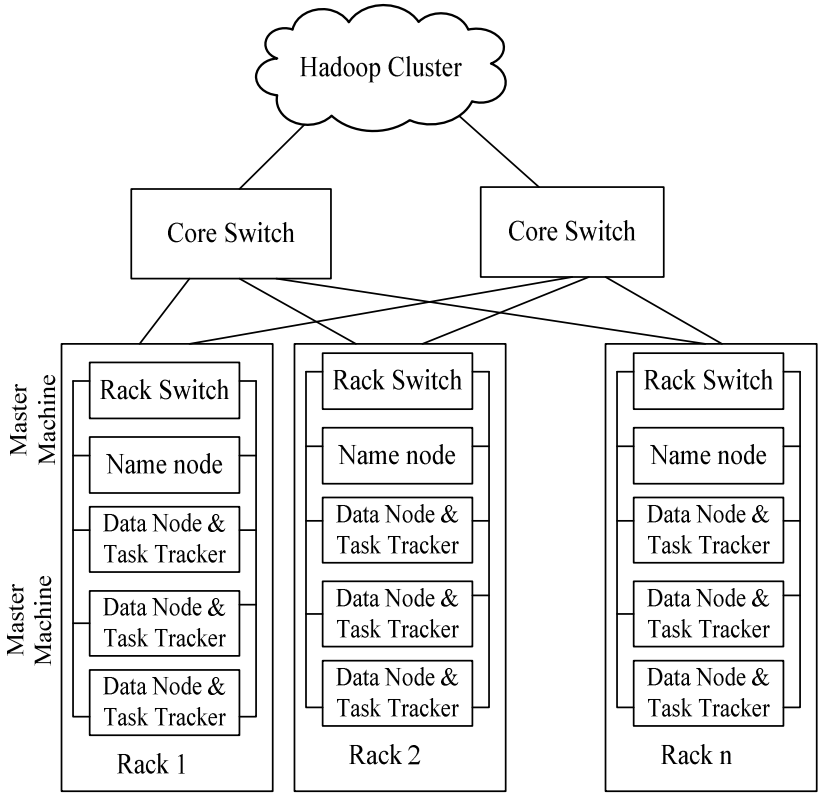

Figure 4: Hadoop cluster using rack switches

The Hadoop node architecture runs tasks in parallel operating in various clusters over many nodes. This results in burdens on a network that can cause problems. The first is that requires to process data in a locality manner which basically means to have the ability in processing the

information where you actually store it (locally). Moreover, the "shuffle and sort" process which occurs between the nodes that are processing parallel jobs produces east-west traffic. This results in poor network connectivity. The map phase process of Hadoop consists of the job tracker trying to use data locality when scheduling map tasks, and may cause problems because of the dependency on the data nodes which is the locations where the data is locally stored. The problem here is what happens when a task is scheduled on a node that does not contain the locally available data?

In order to guarantee prime Hadoop performance, the network must deliver high bandwidth, low latency, and reliable any-to-any node connectivity. Second, any Hadoop deployment that is running on underlying network architecture needs to plan for seamless scaling of cluster growth in order to provide predictable performance. Third, the increased "east-west" node to node traffic flows requires a combination of unicast and multicast communications between nodes, which also demands extremely high bandwidth and low latency results.

Hadoop critically requires that the network have these issues resolved as well as low latency node hops in order to function effectively, which leaves room for improvement in the future big data endeavors.

3.2. Fabric Network Infrastructure for Big Data

Point-to-point switching fabric architectures are emerging in the area of big data and even cloud implementations as a solution to big data network issues that arise in the traditional infrastructure. Much of the current network environments are based on a hierarchical structure were the bottom of the tree is where the hosts connect to network using the access layer and in big data networks they will be divided using rack switches [22]. The middle layer is the distribution (aggregation) layer, which goes up to the top core layer (Figure 5 – 6). The top core layer is responsible for routing and providing services to outside the network and remote locations. Common problems are bottlenecks when upper layers are oversubscribed [23].

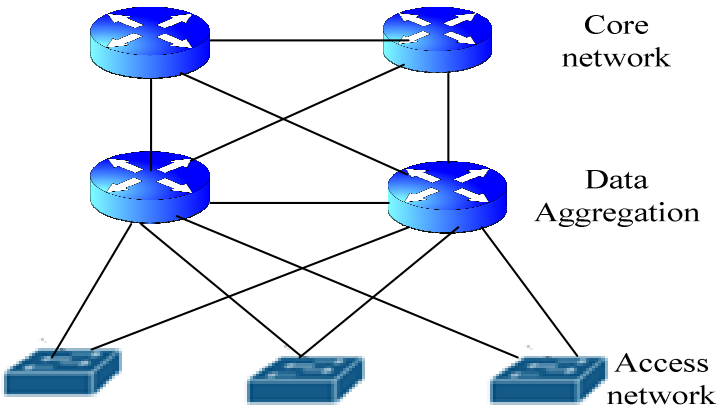

Figure 5: Basic Hierarchical network architecture

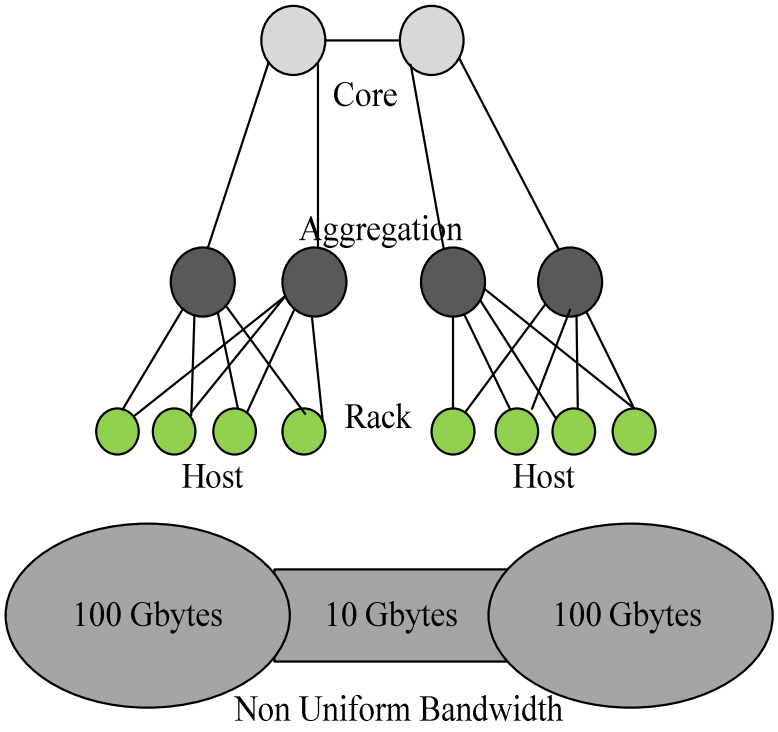

Figure 6: Big Data Hierarchical topology not capable of uniformed bandwidth

The fabric topology is based on a series of switches called leafs that form the access layer and these switches are further meshed to spine switches basically creating a "fabric" network (Figures 7 - 8). Every access-layer switch is only one hop away from each other, immediately reducing the probability of bottlenecks and minimizing latency [22].

The leaf-spine topology can function in layer 2 and 3 which allows switching and routing between the leaf and spine layer. Fabric is said to increase connectivity and flexibility in networks that implement dynamic virtual environments and are prepared for a converged scalability of devices. The fabric network topology is defined as "when network nodes are connected to each other using one or many network switches" [19].

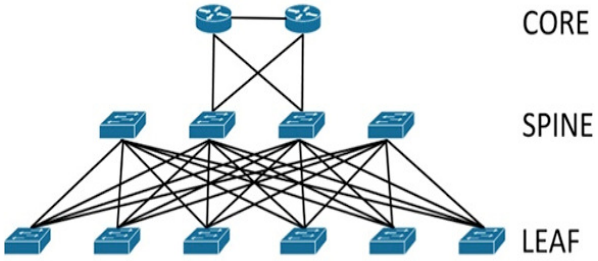

Figure 7: Basic Fabric network architecture

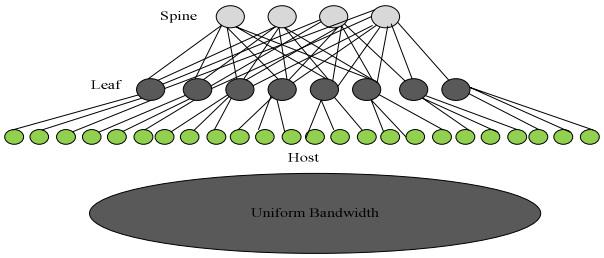

Figure 8: Big Data Fabric topology capable of uniformed bandwidth

Fabric networks are based on spreading the flow of network traffic across many physical links, which outputs an increase in higher throughput, bandwidth and performance over typical broadcast networks [15]. A flatter converged network is simpler and cheaper to install, while increasing traffic efficiency and reducing congestion. Fabric networks have the benefit over tree topology because they use layer 2 network paths which help with load balancing and reducing redundancy [21]. A fabrics convergence feature allows multiprotocol communication and transparency extending to all locations in the network under a single environment. This allows the implementation of consistent services and a greater network intelligence management policy that can provide efficient and secure communication [21].

There are various advantages to fabric over the hierarchical tree based topology. The first advantage in fabric topology is based on establishing a point-to-point connection between nodes using a single hop distance in relation to the switching process, which greatly decreases latency in the network. Second, it is possible to implement a switching fabric using virtualization which allows different networking components to act as one and help improve switching management. It is possible to create a pool of virtual switches that can reduce manual work for the administrator. Third, by having a flat architecture of a single extended environment, it is easier to provide scalability in many areas of the network, especially when creating a data center fabric topology [15]. Expansion and enhancement can be done easily by creating virtual domains or by creating point-to-point connections.

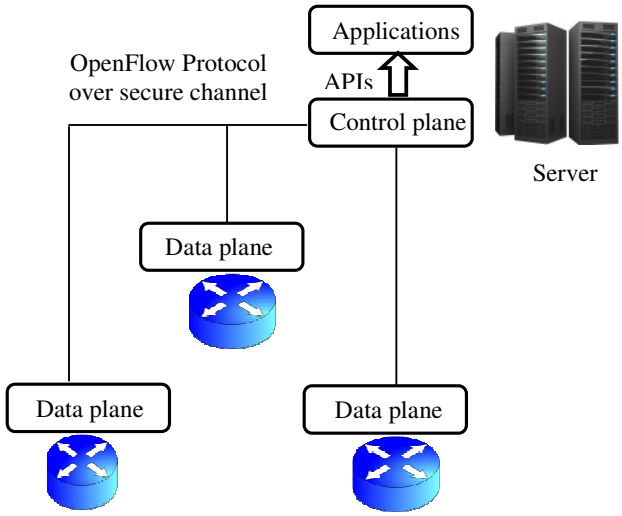

Figure 9: SDN Architecture layer breakdown

3.3. Software Defined Networking Infrastructure

Software-Defined Networking (SDN) [22, 26] was developed for network configuration in order to help network administrators manage processing planes. Big Data communications utilizes to fill the one to many communications from client to databases to provide services and run high-end applications in order to carry huge amounts of dynamic unstructured data that traditional networks are unable to handle. Through the data-planes devices connect and interact in a standard fashion. Separating the control plane and data plane (packet forwarding) changes what once was a closed system to an open one, making it possible to configure network equipment processing. The network separates into three-layers: application, control and infrastructure/data (Figure 9).

In order to effectively use SDN architecture in providing programmable network connectivity one is able manage the operation of the network via software programs. Directly managing the planes gives an innovated way to effectively adapt the control plane to provide a flexible and efficient topology that is able to handle various networked applications and services. The Open Networking Foundation provides the most well received definition of SDN:

"Software-Defined Networking is an emerging network architecture where network control is decoupled from forwarding and is directly programmable" [22]. Figure 10 demonstrates the architecture of big data that comprises operations monitoring, big data controlling, reporting, scheduling and triggering.

3.3.1. Operation of Software Defined Network

A purpose of SDN is to use software that has been defined to run a network [27], with the layer of abstraction the aim is to make a flexible network management system able to process all Big Data requests effectively and efficiently. Switches mainly are forwarding devices controlling routing decisions that gets processed by the central controller.

The idea of programmable networks and decoupled data and control plan reveals the base at which the SDN controller effectively operates. The control plane is a communications channel used for exchanging signals (messages) among logical network devices to do forwarding and management. Most of the available products for SDN set it up with a North Bound plane where users can use programmability within an application programing interface (API) to retrieve real-time statics and service usage. The data plane aka forward plane where all the information goes to be switched in the network where the traffic is accounted and measured for service use. Using this network is the application layer from which custom-made applications are used.

Virtualized networks [27] are used to bring flexibility. As a result, clients start a pay as you use service that handle virtual network infrastructure using Virtual Private Server (VMs) that create customized topologies facilitating dynamic network configuration using centralized control plane implemented in software.

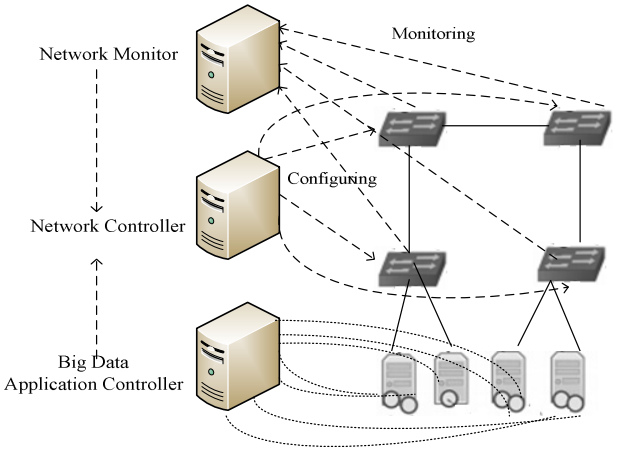

Figure 10: Hybrid awareness operations monitoring, big data controlling, reporting, scheduling, and triggering

3.3.2. Traffic-Aware Architecture

The Northbound API interfacing the SDN controller can use extra functionality for security or routing. Open Flow has worked very well in adapting the standardization of network device programming and configuration between controllers and switches. In operation Open Flow contains two rules that match fields, filtering packet headers and instructions to take with matches. During transit the controller sends the switch a rule and route of the packet. The activity of the controller provides pre-defined match rules that can be managed through static or dynamic SDN control software. It's up to the manager to define the flow of traffic from usage patterns, applications and cloud resources. Though Open Flow is highly regarded technology being used to processes SDN control plane data it has its flaws when considering real-world hardware limitations. As it is the de facto to become more main stream a modification proposed by Curtis [28] the authors of Networking For Big Data break down Curtis look at designing Devo Flow to "devolve control by cloning rules whenever flow is created using wildcards." The work done in [13] reduces overhead by Open Flow in order to have more efficient flow management for Big Data analytic applications networking uses local actions so switches can have local routing decisions without having to contact a controller. Figure 11 shows SDN controller and flow table communications with Open Flow.

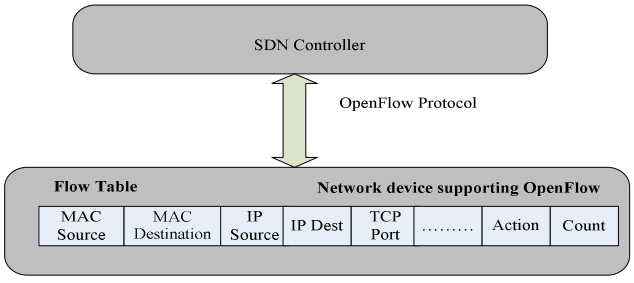

Figure 11: SDN Controller and Flow Table communications with Open Flow.

3.3.3. Application-Aware Architecture

The goal of the big data networks is to be constantly improving to handle the performance of its networks through network configuration and programmability. A part of doing this is the necessity for the underlying network to be aware of application requirements to be able to service its needs. One of the approaches to service application needs is to use switches pre-rendered flow information in the control plane in order to find and configure scheduled flows. To further the flows application-level inputs and requirements are included at the transport layer from the Big Data applications. Applications would be part of several data transfers known as shuffles from mappers and flow tables correlating to specific sorting of flows. In order to optimize flows both hardware and applications will need to orchestrate a correlation for seamless reconfiguration of resources and upper-layer requirements. Due to poor performance by Hadoop in heterogeneous clusters optimizers have been proposed. OFScheduler executes MapReduce jobs, asses network traffic, and load-balances traffic on the links decreasing the time it takes jobs to finish based on the demand of MapReduce jobs to improve performance [29]. Heavy loaded links are offloaded by preference of load balancing flows especially duplicated tasks, larger flows can affect global cost so offloading occurs for big flows to maximize performance and avoid rescheduling [13].

4. DISCUSSION AND CONCLUSION

This paper presents a comprehensive survey on the network theory of big data: capacity, management and data processing requirements of big data, architecture of big data that include MapReduce paradigm and Hadoop distributed architecture, fabric network infrastructure and software defined networks (SDN).

The need to handle big data effectively is quickly growing every year and more research will be needed to come up with solutions. Big data has specific network requirements that must be considered when trying to implement this technology, such as network resiliency, congestion, performance consistency, scalability, and partitioning. The capabilities of current network infrastructures are definitely being challenged by big data's demanding characteristics. It is important to implement various tools, methods and techniques into networks in order to help support big data processing such as MapReduce and Hadoop, which helps manage and process the data more efficiently by breaking up the work to distribute simultaneously across the network. Research in the area of big data has proven that the ideal choice for implementing big data in a network is using a fabric-based topology. It comes with many advantages over a hierarchical architecture such as improved throughput, performance, load balancing, scalability and a reduction in bottlenecks and latency. When looking at theories for Big Data networking many sources referenced the eventual integration with Software Defined Networks which led us to describe the ways SDN can be used to elevate Big Data problems that can be addressed through programmability with the right kind of topology, scheduling, and optimization techniques. In conclusion, this paper presents an overview of how big data functions, compares network topologies, implementation of controlling network traffic flow, discusses methods used to process, store and manage big data, and explores research challenges with proposed solutions.

REFERENCES

[1] Han, Jing, et al. "Survey on NoSQL database". *Pervasive computing and applications (ICPCA), 2011 6th international conference on*. IEEE, 2011.

[2] Lam, Chuck. *Hadoop in action*. Manning Publications Co., 2010.

[3] M. R. Jam, L. M. Khanli, M. S. Javan and M. K. Akbari, "A survey on security of Hadoop", *2014 4th International Conference on Computer and Knowledge Engineering (ICCKE)*, Mashhad, 2014, pp. 716-721.

[4 Borthakur, Dhruba. "The hadoop distributed file system: Architecture and design", *Hadoop Project Website* 11, No. 21, 2007.

[5] X. W. Chen and X. Lin, "Big Data Deep Learning: Challenges and Perspectives," in *IEEE Access*, vol. 2, pp. 514-525, 2014.

[6] Sagiroglu, Seref, and Duygu Sinanc. "Big data: A review." *Collaboration Technologies and Systems (CTS), 2013 International Conference on*. IEEE, 2013.

[7] Bushong, Michael. "Six considerations for big data networks", *searchsdn.techtarget.com*, Accessed Web on March 2016

[8] Kvernvik Tor and Matti Mona. "Applying big-data technologies to network architecture", *Ericsson Review*, 2012

[9] Guohui Wang, T.S. Eugene Ng, and Anees Shaikh, "Programming your network at run-time for big data applications", In *Proceedings of the first workshop on Hot topics in software defined networks* (HotSDN '12). ACM, New York, USA, pp. 103-108, 2012

[10] Idiro Analytics, "Hadoop", http://idiro.com/about-idiro/our-technology/ Accessed Web in May 2016

[11] Introduction to big data: infrastructure and networking considerations, Juniper Networks White Paper 2012.

[12] Jain, Raj. "Networking issues for big data", *Class Lecture,* Washington University in St. Louis 2013

[13] Lin Xiaodong, Misic Jelena, Shen Xuemin, and Yu Shui, "Networking for big data", *Boca Raton: CRC Press*, 2015 Print.

[14] Bertolucci, Jeff. "Big data development challenges: talent, cost, time". *Informationweek.com.* Web. Aug 18, 2012.

[15] Borovick, Lucinda and Villars and L. Richard, "The critical role of the network in big data applications". *Cisco White Paper*, April, 2012.

[16] Chen, Min, Shiwen Mao, and Yunhao Liu. "Big data: a survey." *Mobile Networks and Applications* 19.2 (2014): 171-209.

[17] Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: simplified data processing on large clusters." *Communications of the ACM* 51.1 (2008): 107-113.

[18] Mearin, Lucas. "World's data will grow by 50x in next decade, IDS study predicts". *Computerworld.com*, Web Accessed on May 2016.

[19] Eneboe, Michael K., and Andrew D. Hospodor. "Isochronous switched fabric network", *U.S. Patent* No. 7,002,926. 21 Feb. 2006.

[20] Jeffrey Dean and Sanjay Ghemawat, "MapReduce: simplified data processing on large clusters", *The ACM Communication Magazine*, Vol. 51, Issue. 1, pp. 107-113, January 2008

[21] Borovick Lucinda and Villars L. Richard, "The critical role of the network in big data applications". *Cisco White Paper*, April 2012.

[22] Banks, Ethan. "Data center network design moves from tree to leaf". *Searchdatacenter.techtarget.com*, Packet Pushers Interactive, Accessed on June 2016

[23] Jeffrey Dean and Sanjay Ghemawat, "MapReduce: simplified data processing on large clusters". *The ACM Communication Magazine,* Volume 51, Issue 1, pp. 107-113, January 2008.

[24] IBM, "Bringing big data to the enterprise", https://www01.ibm.com/software/in/data/bigdata/ Accessed Web on June 2016.

[25] McDougall, Richard. "Is your cloud ready for big data?" *Strata Conference – Co-presented by O'Really Cloudera*, New York, Oct 28- 30, 2013

[26] Open Network Foundation, "Software-defined networking: The new norm for networks," *White Paper*, April 2012

[27] Open Networking Foundation, "SDN Architecture Overview", *White Paper V1.0*, December 2013.

[28] Curtis, A. R., Mogul, J. C., Tourrilhes, J., Yalagandula, P., Sharma, P., and Banerjee, S., "Devoflow: Scaling flow management for high-performance networks", *ACM SIGCOMM Computer Communication Review*, 41(4): 254-265, July 2011

[29] Zhao Li, Yao Shen, Bin Yao, Minyi Guo, "OFScheduler: A Dynamic network Optimizer for MapReduce Heterogenous Cluster", *International Journal of Parallel Programming*, Volume 43, Issue 3, pp. 472-488, June 2015.