Available online at www.sciencedirect.com
ScienceDirect
Procedia Computer Science 50 (2015) 235 – 243
Procedia Computer Science

2nd International Symposium on Big Data and Cloud Computing (ISBCC'15)

Switch Bandwidth Congestion Prediction in Cloud Environment

Sivakumar Kuppusamy[a], Vivekanandan Kaniappan[b], Devi Thirupathi[c] and Ramasubramanian.T[d]

[abc] *Department of Computer Applications, Bharathiar Univeristy, Coimbatore, India.*
[d] *EMC^2 Karnataka, India*

Abstract

Cloud environment is mainly used to store and analyze large volumes of data .The major benefits of hosting the applications using cloud environment are storing large volume of data and performing live data analytics. Cloud systems have switches, which serves as an interface connecting and sharing the data with each other. Network bandwidth congestion is caused when large volumes of data are moved to node or cluster in the cloud environment which in- turn creates Service Level Agreement (SLA) violation for the cloud application service. The result of SLA violation includes impact on working of the entire cloud service .The proposed Network Bandwidth Congestion Prediction framework (NBCPF) predicts the bandwidth congestion using weighted random early detection, to avoids the SLA violation for applications services. NBCPF arrests the bandwidth congestion in cloud environment by prediction of switches in cloud environment resulting in accurately identifying the exact task involved in creation of jobs or services.

Keywords: SLA violation; switch bandwidth congestion; Cluster bandwidth congestion; WRED; NBCPF.

1. Introduction

The very important take away of using cloud environment apart from application hosting and storing big data are backing up large volumes of data, historic analysis and analyze shared resources for large volume of data execution using disk, memory, processor and network. The disk, memory and processor supports expansion network remains unexpandable when it is subjected to transferring large volume of data in a shared environment. The data flow rates can be controllable in applications, but in distributed application, the flow rates can't be controlled. Example - Hadoop based data processing. In Hadoop system, the data flow rates are un-controllable, when job gets invoked, it takes all distributed data block from across network and start process. In a shared environment like cloud, this type of execution will create network congestion for moving large volume of data and it affects the other application process and creates the SLA violations for the application response [1]. A switch in cloud environment routes data from one cluster to another. When bandwidth congestion occurs the cloud environment takes a big hit with its performance and data availability causing data loss and brakes the network flow. Each cluster consists of one more host is connected with another cluster using switches. Example: Below Figure.1 illustrates Working Data Center

doi:10.1016/j.procs.2015.04.054

Switch Architecture mainly consisting of various types of clusters like Hadoop Cluster, Application X Cluster, Application Y Cluster, Data Processing Cluster and Analytic Cluster. Clusters share common switch for transmitting and receiving high volumes of data from data processing cluster to Hadoop cluster. Due to the large amount of data involved, sharing switch might get congested resulting in data loss from the application cluster and SLA violation with respect to client response.

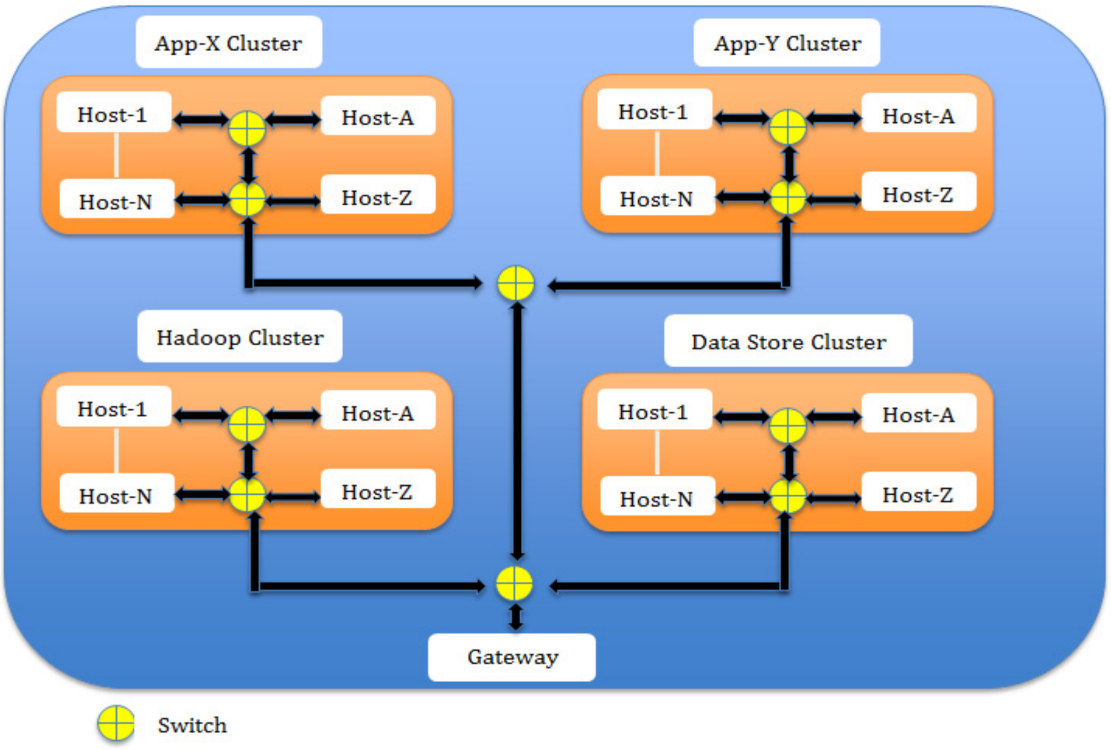


Figure.1 Data Centre Switch Architecture

2. Network congestion-A background study

Network bandwidth congestion prediction tools predicts the bandwidth congestion based on data flow rate, packet loss and SNMP technique. The monitoring tools has option to setup alert for some threshold value, once the bandwidth congestion reached the threshold value, it will send alert to configured mail[2][3]. Network bandwidth prediction tools have no options to control the jobs across the cluster. The weighted random early detection (WRED) technique is added inside switch to control the congestion and alert the node to control the flow rate, but this technique is not used for the hadoop based large data processing and multiple application hosting cluster to control the bandwidth congestion. The prior art system is not having early detection approach in complex condition across different applications and cluster level. The proposed system will collect the bandwidth usage from all nodes and switches in the datacenter and helps to create the mapping for switches and nodes, this mapping helps to easily predict the data flow, bandwidth congestions of switches and clusters, it controls the jobs across different clusters based on bandwidth congestion.

.

3. Insights into architecture of cloud data center

In a cloud environment, applications are hosted in different data center for backup, failover and handling more application request grouped by each region. Below Figure.2 is the Data Center Architecture for handling different application request.

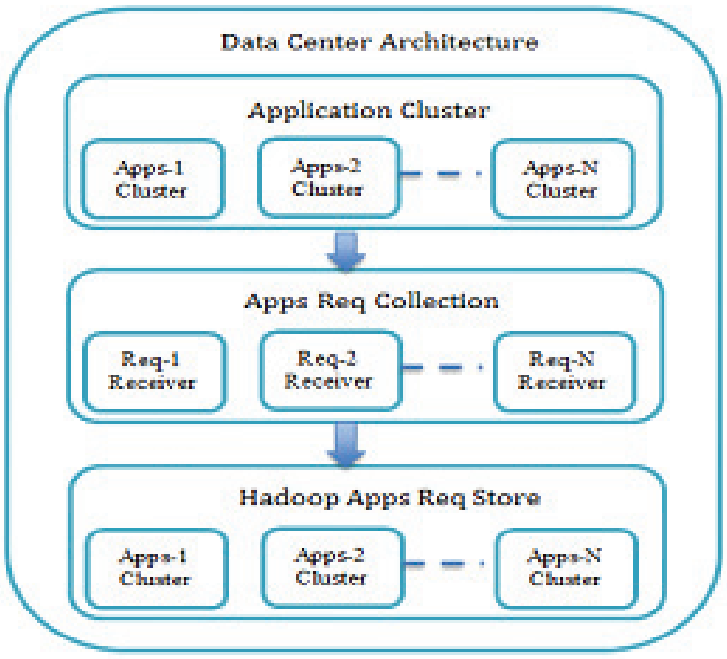


Figure.2 Data Center Architecture

Each application cluster has specific agent for publishing request details to request receiver for widget analytical processing. Request receivers accumulate all application data and store it in the Hadoop cluster of same data center. The applications Hadoop data is moved to another common data processing cluster for analytics and this analytics results is published to business users for application specific use cases.

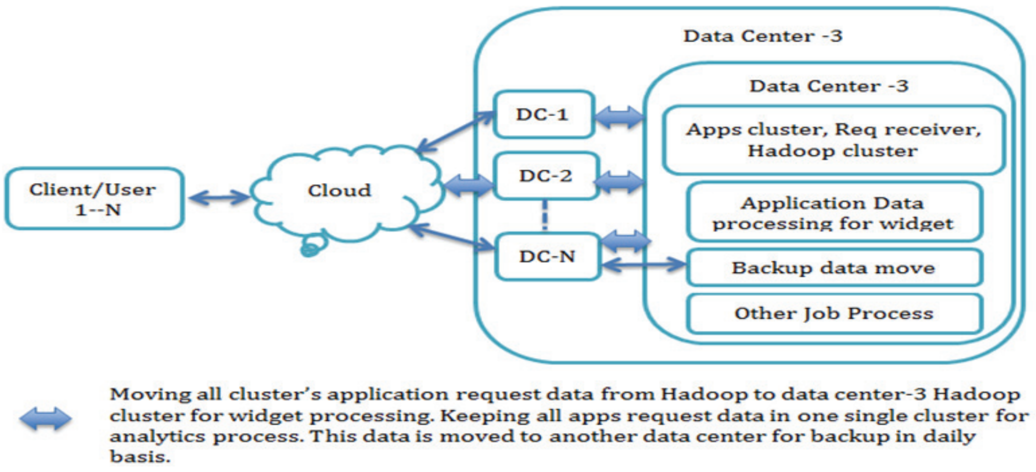
4. Working methodology of cloud data processing environment

Cloud applications are accessed by cloud user by sending the application request to the cloud application, which is further processed and stored, in the cloud. When storing the data in cloud, the request details are collected for analytic process, to know the user info, location, type of application access etc,

The following tasks are performed by each module in the cloud data processing environment

- Analytic information is extracted from each request and stored in application-hosted machine and sent to request receiver by an agent in the application. Agent in the application is responsible for sending the request data periodically to request receiver based on message count threshold and time limit.
- Request receivers retrieves all application data periodically and store it in Hadoop cluster in request received data center.
- The request data of all applications are then moved to job processing data center for analytical process, which implies medium volume size data movement from one data center to another data center.
- The job processing data center schedules jobs of all application data process, analytical for widget data process and backup data move for all applications data to backup Hadoop cluster in another data center[6][7].
- Backup data movement usually involves large volume of data move, which in-turn it triggers, the switch bandwidth congestion while moving the data.
- The job processing data center has job-tracking agent, which is used to track the all scheduled job and data size of job, priority of job and dependency. This tracking mechanism provides ability to track data periodically and updates the proposed system with information, which is used in congestion prediction algorithm for avoiding the congestion in switch by controlling the job execution.

The following Figure. 3 illustrates the working methodology of data processing environment



⇔ Moving all cluster's application request data from Hadoop to data center-3 Hadoop cluster for widget processing. Keeping all apps request data in one single cluster for analytics process. This data is moved to another data center for backup in daily basis.

Figure.3 Working methodology of ata processing environment

5. Proposed System

The proposed solution will predict bandwidth congestion from switches and detect the issue creating jobs from clusters and applications. As per design it will automatically controls the factor creating jobs based on priority and it kill the jobs based on congestion creation ratio in switch thereby averting SLA violation for application access. The following attributes influence detection of congestion and provides pre- determined data to avoid SLA violation in cloud environment,

- Job to expose the source data file path and schedule time
- Priority of job
- Job dependency

The job data collector agent is setup in job processing system in each cluster. This agent is used for collect the running and scheduled job's data size, job completion percentage, priority of job and dependency jobs for running and scheduled job.

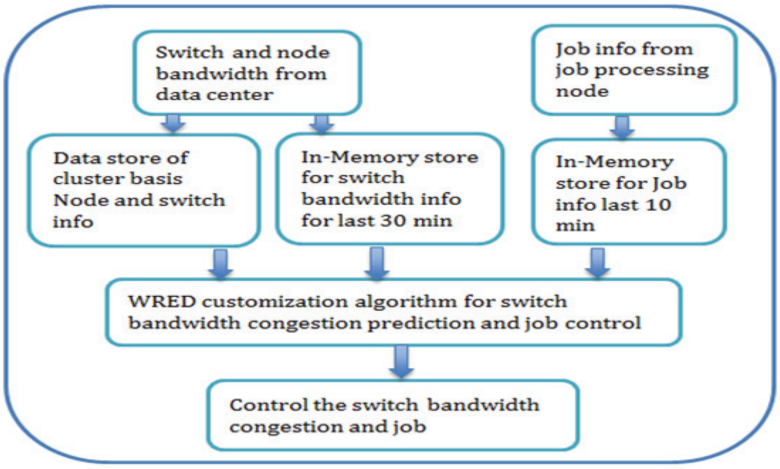

Switch and node bandwidth from data center
Job info from job processing node
Data store of cluster basis Node and switch info
In-Memory store for switch bandwidth info for last 30 min
In-Memory store for Job info last 10 min
WRED customization algorithm for switch bandwidth congestion prediction and job control
Control the switch bandwidth congestion and job

Figure.4 The Proposed Solution Architecture

Cluster is a collection of one or more systems integrated together. Generally, cloud environment has more number of clusters where in each cluster will perform their specific tasks Ex. Hadoop Cluster, Applications Cluster, Data Backup Cluster etc. In cloud environment, some clusters say Hadoop Cluster where data flow and storage rates are very high, Hadoop Cluster will store and process large amount of data, so the data movement across the systems is heavy and it will affect the bandwidth usage of other applications and creates the bandwidth congestion [8]. The other use case is moving large volume of data to store in backup cluster, The resultant effect of it will create the bandwidth congestion and it affects the SLA of other applications usages. Switches, which are the main interface for system integration, are finally affected. The proposed solution will predict cluster bandwidth congestion prediction. It will allow and properly aid multi-tenant application hosting platform to perfectly manage the defined SLA.

Algorithm for WRED Customization

```
List<Switch>switchLst=getSwitchDetailsForCluster(clusterName);
List<Switch>mainSwtLst=getMainSwitchList(switchLst);
List<Switch>parentSwtLst=getParentSwitchLst(mainSwtLst);
for(inti=0;i<parentSwtLst.size();i++) {
      wredImpl(parentSwtLst.get(i),clusterName);
}

public void wredImpl(Switchswobj,StringclusterName) {

Portdetail[] pdetail= swobj.getPortDetail();
doubleportBandwidth[] = newdouble[pdetail.length];
double sum=0;
for(inti=0;i<pdetail.length;i++) {
   if(pdetail[i].isActive()) {
        if(pdetail[i].getPortType().equals("node")) {
                portBandwidth[i]=pdetail[i].getBandwidth();

        }  elseif(pdetail[i].getPortType().equals("switch")) {
                portBandwidth[i]=pdetail[i].getBandwidth();
        }
        sumSWBW=sumSWBW+portBandwidth[i];
   }
 }
doubleswitchCapacity = swobj.getCapacity();
double p65 = switchCapacity*(65/100);
doubleswitch_stddev=getStddevofSwitch_30min();
intestmin=getCriticalEstimation(switch_capacity,switch_stddev,p65)

While(true) {
  if(estmin<5) {
          List<Job>jobsList=getJobsDetails(clusterName);
        //Latest Running Job : Low Priority jobs
        Job latestJob= getlatestRunningJob(jobsList,lowPriority);
                If(latestJob.getRunMin()<15){
                   latestJob.killJob();
                   List<Job>dependencyLst=latestJob.getDependencyJob();
                   latestJob.killJob(dependencyLst);
                }
        //Latest Running Job : High Priority jobs
        latestJob= getlatestRunningJob(jobsList,highPriority);
        If(latestJob.getRunMin()<5){
           latestJob.killJob();
           List<Job>dependencyLst=latestJob.getDependencyJob();
           latestJob.killJob(dependencyLst);
        }
        //Max Datasize Job termination
        Job maxDatasizeJob = getRunningMaxDatasizeJob(jobsList, lowPriority);
         maxDatasizeJob.killJob();
         maxDatasizeJob = getRunningMaxDatasizeJob(jobsList, highPriority);
         if(maxDatasizeJob.getRunMin< 10) {
             maxDatasizeJob.killJob();
         }

         Job lngRunJob= getLongRunningJob(jobsList);
         Double pCompl = lngRunJob.getPercentageCompletion();
          If(lngRunJob.getRunMin()>75 &&pCompl<95) {
          //Long running jobs
             lngRunJob.killJob();
          } else if((lngRunJob.getRunMin()>60 &&lngRunJob.getRunMin()<75) && pCompl<80)) {
               lngRunJob.killJob();
             }
          switch_stddev=getStddevofSwitch_30min();
          estmin=getCriticalEstimation(switch_capacity,switch_stddev,p65);
      } else {
             break;
      } // end of estmin check()
    } // end of while()
  }   // end of wredImpl()
```

Algorithm for WRED Customization -Switch Bandwidth Congestion Prediction

- To collect each minute bandwidth data from each hosts and switches
- Group the hosts and switches by cluster.
- To create mapping for hosts to switches and switch to sub-switches.
- To collect the job information running and scheduled jobs, job data size jobs priority and jobs dependencies from each cluster.
- Calculate the critical estimation value for main switches in each cluster
- To control the congestion for switch when estimation value is low, and then control the job execution in the cluster based on the job information running and scheduled jobs, jobs data size, jobs priority and jobs dependencies.
- This algorithm is implemented as parallel way of congestion prediction.

Critical Estimation Computation for Switch

The critical estimation value is used for predicting the congestion in the switch using customized weighted RED. This technique is not in the existing weighted RED**.** Following is the steps for calculating the critical estimation value.

- Using SNMP to read the bandwidth from each port in the switch in the mode of Ethernet-in and Ethernet-out [9][10].
- To collect each minute switch port bandwidth and store in the in-memory database, to store last 4 hours data.
- Gather last 30 minute data for critical estimation calculation.
- Using RED, to calculate the mean and standard deviation for bandwidth of switch
- Defines percentile value for 65% from switch capacity (P65).
- Based on the switch capacity, standard deviation of switch bandwidth and percentile P65 is used to calculate the critical estimation value for the switch bandwidth to reach the critical bandwidth usage stage.
- Migrate the historical data (4 hours old) to no-sql database for analytical purpose.

Mathematical approach for critical estimation calculation for bandwidth usage

Let S_{PORT} is no of switch port in the switch

$$S_{BW} = \text{SUM}(\text{PORT\_BANDWIDTH}_{1\rightarrow N}) \quad (1)$$
$$S_{MEAN} = S_{BW}/S_{PORT} \quad (2)$$
$$S_{STD} = \text{SQRT}(1/ S_{PORT} * \text{SUM}(\text{PORT\_BANDWIDTH}_{1\rightarrow N} - S_{MEAN})^{\wedge 2}) \quad (3)$$
$$C_{EST\_MIN} = (P_{65} - S_{BW})/S_{STD} \quad (4)$$

S_{BW} : Switch Bandwidth
S_{MEAN} : Switch Bandwidth Mean
S_{STD} : Switch Bandwidth Standard deviation
C_{EST_MIN} : Switch Critical Bandwidth Estimation

6. Performance and Evaluation

We have done a case study between existing frameworks available and based our solution to actually bridge the gap in our implementation based on our findings. Weighted random early deduction technique is used to avoid the congestion, but it would not control the job execution in cluster in a cloud environment. The effect of this will enable switches to get more bandwidth and incurs packet loss thereby propelling the congestion in the switch. Our proposed solution will not only manage the switch bandwidth but also controls the job execution in cluster to predict the congestion early and avoid the congestion in switch. Following specifications describes the job execution status in the Hadoop Cluster. This cluster will process high volume of application's request process for widget publish and moving the collected applications data to backup cluster, this backup data movement creates the bandwidth congestion to applications data processing and creates SLA violations to applications clusters.

Job Name	Job Detail
Application's widget update for all cloud applications	**Job Description**: This job will publish the widget detail of all cloud application. The widget contains the unique user information for application access for every hour. **Period of execution**: Every Hour **Job Details**: No of Application : 63 One hour data size of 63 App: 410GB No of Request : 2.2 billion Type of job : Analytical process No of node : 210 No of switches: 5 (48 port)
Application Request log data backup into other cluster	**Job Description**: This job will collect all the application's request log data into another backup cluster. **Period of execution**: Daily **Job Details**: No of application : 63 Daily data size : 4.7 TB No of request: 59 billion Type of job : Move to backup cluster No of node : 530 No of switches: 12 (48 port)

Before solution implementation

The switch has WRED technique to control the data flow rate based on the bandwidth congestion, but it would not applicable for large cluster for bandwidth congestion and it would not support to control the job execution across the clusters. Following table has congestions details of most data flow main switches. The congestion control in the main switch not supporting to control the bandwidth congestions and it creates the SLA violation to application cluster for application request processing.

S.No	Switch Name	Switch Capacity	Switch Bandwidth	SLA Violation	Data Loss	status
1	St-main-p001	100Gbits	64.5 Gbits	Yes	Yes	Congestion
2	St-main-p045	100Gbits	63 Gbits	Yes	Yes	Congestion

NBCPF Solution implementation

The customized WRED algorithm has features of job control in across the clusters and it predict the congestion in the switch and it controls the job execution in the clusters as low priority and long running jobs in the cluster and it helps to control the SLA violation in the application clusters. Following table has proposed solution results and it controls the congestion.

S.No	Switch Name	Switch Capacity	Switch Bandwidth	P_{65}	Before		After	
					Bandwidth STD for 30min	Estimation min. for critical condition	After job Control STD	Estimation min. for critical condition
1	St-main-p001	100 Gbits	60.5 Gbits	65 Gbits	850.57 Mbits	3	432 Mbits	10
2	St-main-p045	100 Gbits	61Gbits	65 Gbits	727.00 Mbits	4	347 Mbits	11

STD : Bandwidth Standard Deviation
P_{65} : Percentile Critical Bandwidth limit

The WRED customization algorithm predicts the switch bandwidth congestion. The critical estimation minute less than five minute, the algorithm controls the job execution and avoids the bandwidth congestion in the switch and avoids the SLA violation.

7. Conclusion

By taking into consideration of various design and model for Network prediction in the market we have designed this solution for cloud system to accurately predict the bandwidth congestion .Our proposed Network Bandwidth Congestion Prediction Framework (NBCPF) uses weighted random early prediction technique to avoid SLA violations for application services. NBCPF extracts the information on switches as well as identifies and fixes the exact cause causing the creation of jobs or services which is responsible for bandwidth congestion. Using NBCPF in cloud environment, one can easily leverage upon the technique to manage any number of clusters with zero concerns about its performance and availability thereby enhancing the overall productivity of cloud based systems.

References

[1] Ruey-Maw Chen, Yueh-Min Huang, "Competitive neural network to solve scheduling problems", 09252312/01/$ - see front matter _ 2001 Elsevier Science B.V.
[2] Tran Xuan Truong, Le Hung Lan, "Congestion Control In TCP/IP Differentiated Services Network Using Neural Network", 9781424499533/11, 2011.
[3] Harvey B. Newman, Iosif C. Legrand, A Self-Organizing Neural Network for Job Scheduling in Distributed Systems, Kent State University, Kent, OH, USA, Doctoral Thesis, 2003.
[4] Martin H. Luerssen, Character Recognition with Neural Networks, School of Informatics & Engineering Flinders University of South Australia GPO Box 2100 Adelaide 5001 Australia.

[5] Heikki N. Koivo, NEURAL NETWORKS: Basics using MATLAB Neural Network Toolbox, February 1, 2008.
[6] V. Venkatesa Kumar and K. Dinesh, Job Scheduling Using Fuzzy Neural Network Algorithm in Cloud Environment, Bonfring International Journal of Man Machine Interface, Vol. 2, No. 1, March 2012.
[7] Martin Hagan, Neural Network Design ECEN 4120 (Call # 45307) / ECEN 5120 (Call # 45308), July 2007.
[8] KlimisSymeonidis, Hand Gesture Recognition Using Neural Networks, School of Electronic and Electrical Engineering On August 23, 2000.
[9] Qinghui Lai1, Haiye Yu1, Haitao Chen, Yong Sun, Prediction of Total Power of Agricultural Machinery Using Artifical Neural Networks, IEEE International Conference on Computing, Control and Industrial Engineering, 2010.
[10] DibyajyotiGuha, S.S. Pathak, Load Balancing using Past Information of Queue, Kent State University, IEEE 2010.