International Journal of Engineering & Technology, 7 (2.33) (2018) 600-602

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

An approach for dynamically load balanced flow scheduling(DLBS) in cloud

Mrs. A. Sheela [1] *, Mrs. J. Ranjani [2]

[1] *Asst. Professor, Department of Computer Science and Engineering,Sri Sairam Engineering College*
[2] *Asst. Professor, Department of Computer Applications,Sri Sairam Engineering College*
**Corresponding author E-mail:*

Abstract

The Paper based on load balanced flow scheduling algorithm. This algorithm is used in which huge amount of data send to multiple servers frequently without a few traffic, isolation in open flow network. In existing load balanced algorithm based on huge amount of data send to several server but it suffers from that that algorithm is not support to other open flow networks model and transmission pattern. In this proposed load balanced scheduled algorithm with Round Robin deals with maximizing the network throughput dynamically. The (Dynamically Load Balanced Flow Scheduling) DLBS problem is formulated , considerably a efficient heuristic scheduling algorithms was developed for the two typical Open Flow network model, they have data flow from time slot. The outcome represents load-balanced scheduling algorithms Round Robin and LOBUS with effective improvement in DLBS move toward will carry to the data centers. Plenty of researchers pay large number of attention on software-defined networking.

Keywords: *DLBS; Datacentres; LOBUS*

1. Introduction

Data center networks in clouds are typically built on massive layered switches where a large amount of data needs to be transferred among thousands of servers. To reduce the end-to-end transmission delay and improve the resource utilization ratio, data flows have to be dynamically scheduled in a load-balanced way. However, it is a very desirable but extremely challenging task due to large-scale and dynamical data flows with different demands. The load-balanced scheduling focuses on evenly distributing traffic among all links in a data center network to enable the network to transmit more data flows with lower average end-to-end transmission delay. Traditional hardware-based load balancing techniques can not be widely used due to the high cost and the deficiency in programmable ability. Therefore, more and more researchers pay more attention on software-defined networking (SDN) techniques (e.g., Open Flow) that can improve transmission capacity of data centers

2. Existing system

In the existing system, both DLBS-FPN and DLBS-FTN algorithm were used. This algorithm is used as a load balanced flow algorithm in which large amount of data send to multiple server based on busiest link, link which contain less number of resources. It balancing the load traffic transmission dynamically. During data flow transmission, we monitor the network status to keep load balanced in the whole network. If the current network is under an unbalanced status, i.e., $\delta(t)>\delta*$, we firstly set the busiest link $\ell i,j$, which has the minimal available resource in the ART table, as the scheduled link $\ell s=\ell i,j$. Next, we will select the biggest flow on the scheduled link ℓs as the scheduled flow fs. To transfer fs to an right sub-path, we search for all the replacement sub-paths Pi,j = {p1, p2, . . . , pm}, where any sub-path joins Si with Sj , and then choice the lightest sub-path pk. By the lightest sub-path, we indicate that pk has the minimal path bandwidth with the utilization ratio along with all sub-paths in Pi,j such that $\lambda pk= \min \forall px\in Pi;j\{\lambda px(t)\}$,in this algorithms be able to adapt to dynamical network states and change the traffic supplies through update load imbalance factor $\delta(t)$ and as a result balancing the transmission load slot by slot throughout data transmissions after that this algorithms can worldwide balance transmission traffic in the entire network by way of estimate link, path and network bandwidth utilization ratio.

3. Proposed system

In proposed system, the load balanced flow scheduling algorithm in which huge amount of data send to multiple server based on open flow network, busiest link, link which contain less number of resources. It balancing the load traffic transmission dynamically. During data flow transmission, we monitor the network status to keep load balanced in the whole network. This algorithm does not take more time recursion and support for all the open flow network in data center. The scheduling algorithms concentrate on data flow transmission in data center networks. We do not consider the overhead caused by transferring control signals. We also assume that all the control signals are synchronized and transmitted without perceivable latency. The load distribution is decided according to current traffic states and network conditions during transmissions. It improves execution parallelism and resource utilization without introducing stragglers during Map reduce concept. This algorithm sutiable for all the network and does not take more time to send a data.

4. System diagram

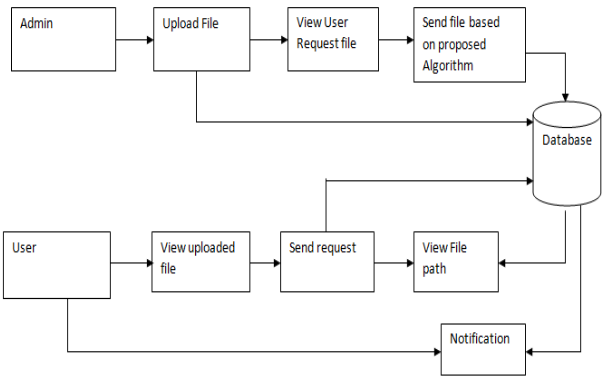

Fig. 1: Block Diagram.

5. Process for implementation

5.1. Sign in module

In this module, user can register in that page and login to next page by using username and password.

5.2. View file module

In this module, user can view the file which is uploaded by admin

5.3. Response file

In this module, if the admin accept your file then you will get a response. If the admin reject your file, then response will not get it.

5.4. File path module

In this module, the user can view the file path of response file.

5.5. Notification

In this module, we can able to download the response file.

5.6. View user

In this module, admin can view the users in which who are all registered.

5.7. Upload module

In this module, admin can upload a file and file type.

5.8. Request file module

In this module, admin can view the request file. There are two options for admin namely accept and reject. If admin accept file, then response file will send. Otherwise it is not possible to send a response file.

6. Load balancing measure metric

The DLBS approach monitors a data center network and detects the unbalancing degree of links all the time. In this Section, we will present how to measure the load imbalance degree and the scheduling trigger threshold. In CDC networks, transmission capabilities and load on links significantly increase from edge layer to aggregation and core layers. As mentioned above, we use link bandwidth utilization ratio $\lambda i,j(t)$ to capture load states of different levels of links $\ell i,j$. Furthermore, we define the network bandwidth utilization ratio.

Network bandwidth utilization ratio $\lambda(t)$ is the average of bandwidth utilization ratio of all the links in the whole network.

$$\lambda (t) = \sum 1 \leq i,j \leq N \ \lambda i,j(t) /N$$

Where N is the number of all the links in a CDC network. $\lambda(t)$ captures the total bandwidth utilization at a time slot t, and can intuitively demonstrates the effectiveness of scheduling algorithms. Now, we define a factor to measure the load imbalance degree in a CDC network.

7. Algorithm

One-Hop DLBS-FPN Input: S2SPT and ART tables, $\delta*$ Output: load-balanced scheduling

1) update $\delta(t)$ using the formula (8) in each time slot t;
2) MAX=-1;
3) Temp= K $\sum k =1 \, t^{fk}(t)$;
4) while ($\delta(t) \geq \delta*$) OR (MAX>Temp) do
5) $\ell s \leftarrow$ the busiest link $\ell i,j$;
6) fs←the biggest flow on ℓs;
7) find out substitute sub-paths Pi,j={p1,p2,...,pm}for fs through S2SPT;
8) select the lightest sub-path pk∈Pi,j in terms of ART;
9) schedule fs to pk;
10) update $\delta(t)$;
11) MAX=Temp;
12) Temp= K $\sum k =1 \, t^{fk}(t)$;
13) end while

In the approach, the remainder bandwidth of a specific link can be obtained from the ART table. To find available substitute paths, a controller just needs to jointly look up the S2SPT and ART tables. Determination of scheduling trigger $\delta*$. In Algorithm 1, $\delta*$ is an important parameter, which can be determined in the following steps. Firstly, we execute the Algorithm 1 under different values of $\delta*$, and figure out corresponding system throughput. Next, we plot the throughput changes with different values of $\delta*$. Finally, according to the relationship between throughput and $\delta*$, we can choose the value of $\delta*$ corresponding to the highest throughput as the threshold. The details can be found in Subsection 5.3. $\delta*$ determines the balance degree of data flows on different links, which mainly depends on the requirement for load balancing and the network topology. If it is set too high, The DLBS-FPN algorithm cannot effectively detects the imbalance state in networks. On the contrary, flows will be migrated too frequently. For a fixed data center network, the $\delta*$ needs to be determined only one time. Frequent rescheduling avoidance. Algorithm 1 works well in most of situations according to our experimental results. In Algorithm 1, however, extremely large flows (elephant flow) may be frequently rescheduled many times, which causes heavy overhead and also leads to a significant increase in the average transmission delay. At the same time, the frequent rescheduling definitely disobeys the original intention for the load balancing. We clearly observed this phenomenon in the experiments. The reason is that once an elephant flow is migrated on a link, this link will easily become over-loaded again. To solve the above problem, we improve the scheduling algorithm. A flow fs is marked with one more flag hfs, which indicates the rescheduled times of fs. hfs is set as zero at the initial scheduling stage. Once a flow is triggered and rescheduled by the scheduling algorithm, hfs will be increased by 1. Based on this mechanism, we improve our scheduling algorithm as Algorithm 2, where fs.hfs refers to the flag hfs of a flow fs.

8. Conclusion

We conclude that algorithms can adapt to dynamical network states and changing traffic requirements through updating load imbalance factor δ (t) and accordingly balancing the transmission load slot by slot during data transmissions. Next, the algorithms can globally balance transmission traffic in the whole network by means of evaluating link, path and network bandwidth. In this approach significantly outperforms the related and representative load-balanced scheduling algorithms Round Robin and LOBUS under multiple transmission patterns, especially in unbalanced data flow distribution. In particular, the higher imbalance degree data flows in data centers exhibit, the more improvement the DLBS approach will bring to the data centers.

9. Future scope

The future scope is going to widen this work to support other Open Flow network models and new transmission patterns.

References

[1] X. Han, J. Li, D. Yang et al., Efficient Skyline Computation on Big Data, IEEE Transactions on Knowledge and Data Engineering, Vol.25, No.11, 2013, pp.2521-2535.
[2] A. G. Erdman, D. F. Keefe, R. Schiestl, Grand Challenge: Applying Regulatory Science and Big Data to Improve Medical Device Innovation, IEEE Transactions on Biomedical Engineering, Vol.60, No.3, 2013, pp.700-706.
[3] K. Greene, TR10: Software-Defined Networking, MIT Technology Review, Retrieved