An Appropriate scheduling using Dynamic load balancing approach for bigdata in cloud

S. Abinaya [1], R. Kirthika devi [2], M. Manjudevi [3] B. Muthulakshimi [4]

[1,2,3]Students Member, Department of Computer Science and Engineering
[4]Assistant Professor, A.V.C. College of Engineering

Abstract—*Load-balanced flow scheduling is technique, in which we transfer the large amount of data frequently among thousands of interconnected servers, is a key and challenging issue. Here we consider the open flow is the solution to balance data flows in a data center network through its programmatic traffic controller. Existing Open Flow based scheduling schemes, however, statically assaign the path routes for data transmission only at the initilisation stage which suffers from dynamical flow distribution and changing network states in data centers and often results in poor system performance. In this paper, we propose a dynamical load-balanced scheduling (DLBS) approach for dynamically balancing the work load and also increase the maximizing through put. We firstly formulate the DLBS problem, and then for two typical openflow modules here we use the heuristic algorithm which balance data flows time slot by time slot. Experimental results demonstrate that our DLBS approach significantly outperforms other representative load- balanced scheduling algorithms Round Robin and LOBUS.*

Keywords— Flow scheduling, load balancing, data center, Cloud Computing, OpenFlow

I. 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 Analyze the load rebalancing problem in distributed file systems specialized for large-scale, dynamic and data-intensive clouds. 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 differentdemands. 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 through programmable load balanced flow control Many schemes have been proposed for load-balanced flow scheduling in Open Flow based networks. They focus on the initial route selection only before the flow transmission.

1. We identify a new flow scheduling problem in big data centers in clouds, i.e., dynamical load-balanced scheduling (DLB and formulate the DLBS problem. The objective is to optimize network throughput on condition that load balancing is guaranteed on all links during every time slot.

2. We propose a trigger mechanism for dynamical data flow scheduling. We firstly propose a factor δ(t) to capture the load imbalance degree of data center networks, and then define the link scheduling trigger threshold $\delta * . \delta(t)$ is calculated slot by
slot, and the OpenFlow controller initiates our DLBS scheduling algorithms once $\delta(t) > \delta *$.

3. We propose a set of heuristic scheduling algorithms to address the DLBS problem. They are implemented in two representative OpenFlow architectures: FPN and FTN. These algorithms dynamically migrate the flows which occupy the largest amount of bandwidth on the most congested link to the lightest links.

4. We implement a system to simulate a cloud data center, and evaluate our DLBS approach through comparing the DLBS with other classical methods. The experimental results demonstrate that our algorithms significantly outperform the representative Round Robin and LOBUS, especially in unbalanced .

5. To Analyze the load rebalancing problem in distributed file systems specialized for large-scale, dynamic and data- intensive clouds.

6. The Secure Socket Layer_with_BF model is very helpful for load balancing of the server. This will reduce the load of the server while the server is being busy. These are the advantages of our proposed system.

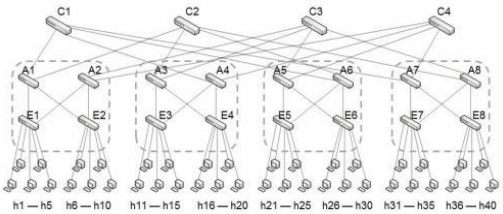
7. The ssl with bf scheme can minimize the average latency by about 40 percent and improve throughput across a variety of workloads.

II. OPEN FLOW FRAMEWORK

In traditional networks (e.g., TCP/IP-based Internet), routers are responsible for both discovering routes and forwarding packets according to their routing tables. In large- scale data centers, such the routing scheme results in two disadvantages. Firstly, routers have to be designed more and more powerful and expensive due to large amount of data transmission tasks. Moreover, it is difficult to balance data flows dynamically. To provide the programmable packet processing ability to meet these converging needs [32], researchers in Stanford, Berkeley and MIT proposed an open network switch

III. MOTIVATION SCENARIO

(Fig.No.2)



IV. 4. DLBS Problem

The objective of our DLBS is to maximize system throughput through dynamically balancing data flows in cloud data centers.

Let there be K concurrent data flows fk (k=1, 2, . . . , K) in T time slots in a CDC network. We use t fk (t) to denote the traffic transmitted from a source sk to a destination dk for a flow fk=(sk, dk) during a slot t. The total transmitted traffic in the whole network during T slots can be summed up as ∑ T t=1 ∑ K k=1 t fk (t).

In CDC networks, links at different levels have different capabilities and oversubscription factors. We use link bandwidth utilization ratio to capture load states on different levels of links, which is defined as follows.

V. 5. Notations and Description

i, j switchs i and j, respectively

f K the number of concurrent data flows in T slots framework OpenFlow in 2007. In the OpenFlow framework,

k actual data forwarding components and route setup

fk the kth data flow fk=(sk, dk) (1≤k≤K). components are separately deployed on switches and controllers [23] [24]. The controllers determine data forwarding rules. the openflow protocols ensure communications between switches and controllers [33]. In $_t$ (t)

Þi , j n Capacity of the link ℓi,j between i and j the set of neighboring switches of $I_{i,\ j}$ (t) bandwidth utilization ratio of ℓi,j during a slot t

VI. DYNAMIC AND LOAD-BALANCED SCHEDULING (DLBS)

we will present our DLBS approach, which mainly consists of two stages: (1) initial flow scheduling when a new flow arrives at the network and (2) dynamical flow scheduling during the flow transmission..

6.1 Initial Flow Scheduling :

Initial flow scheduling. Data transmissions in big data centers exhibits the following “3H” characteristics: high transmission frequency, huge transmission volume and hard transmission deadline. Once a switch receives multiple transmission requests from hosts, it has to schedule these concurrent transmission requests in a specified order.

$$p(fk) = a/\, d(fk) - b \times s(fk)$$

6.2 Dynamical Load-Balanced Scheduling Algorithms for FPN Networks:

During data flow transmission, we monitor the network status to keep load balanced in the whole network.

Output: load-balanced scheduling
1: update δ(t) using the formula (8) in each time slot t; 2: MAX=-1;
3: Temp= ∑ K k=1 t fk (t);
4: while (δ(t) ≥ δ ∗) OR (MAX>Temp) do
5: ℓs← the busiest link ℓi,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 δ(t);
11: MAX=Temp;
12: Temp= ∑ K k=1 t fk (t); 13: end while

Output: load-balanced scheduling
1: update δ(t) using the formula (8) in each time slot t; 2: MAX=-1;
3: Temp= ∑ K k=1 t fk (t);
4: while (δ(t) ≥ δ ∗) OR (MAX>Temp) do
5: ℓs← the busiest link ℓi,j ;
6: fs← the biggest flow on ℓs;
7: fs.hfs ++; //increase rescheduled times hfs of fs. 8: while (fs.hfs>ξ) do
9: drop fs;
10: fs←the next biggest flow on ℓs; 11: end while
12: find out substitute sub-paths Pi,j={p1, p2, . . . , pm} for fs through S2SPT;
13: select the lightest sub-path pk∈Pi,j in terms of ART;
14: schedule fs to pk;
15: update δ(t);
16: MAX=Temp;
17: Temp= ∑ K k=1 t fk (t); 18: end while

VII. Dynamical Load-Balanced Scheduling Algorithm for FTN Networks:

Algorithm 3: Multi-Hop DLBS-FTN Input: S2SPT and ART tables, δ ∗
Output: load-balanced scheduling
1: update δ(t) using the formula (6) in each time slot ; 2: MAX=-1.
3: Temp= ∑ K k=1 t fk (t);
4: while (δ(t) ≥ δ ∗ and Σ(t)≠ Φ) OR (MAX>Temp) do
5: fs← the flow that covers the largest subset of Σ(t);
6: Ps←{p1, p2, . . . , pm};
7: ps← pbest ∈Ps;
8: schedule fs to ps;
9: update δ(t);
10: MAX=Temp;
11: Temp= ∑ K k=1 t fk (t); 12: end while

VIII. Performance Evaluation

We implemented a simulation system to evaluate our dynamical load-balanced scheduling algorithms Improved OneHop DLBS-FPN (Algorithm 2) and Multi-Hop DLBS- FTN (Algorithm 3) for FPN and FTN data center networks, respectively.

IX. Evaluation Schemes

A. Uniform pattern: Flows are initiated and distributed symmetrically among all hosts. In each unit of time, each host transmits a packet with equal probability; and packets are destined to other hosts with equal probability.

B. Semi-uniform pattern: For a specified source host hi , one half of flows generated by the hi are distributed in the intra-pod that connects with the hi directly while other flows are evenly distributed among all the inter-pods.

C. Center-based pattern: It is the most unbalanced traffic pattern, where more than 80% of data flows are generated by a single hot host.

We compared our DLBS approach with the

following representative algorithms

LOBUS: It is a simple load-balanced scheduling scheme through the greedy selection [8]. The basic idea in LOBUS is even transmission delay of all the links as much as possible. More specifical, it greedily picks out the pair of host and path that yields the lowest total response delay for each request. So, flows on the link with the longest transmission delay are migrated to the link with the shortest delay.

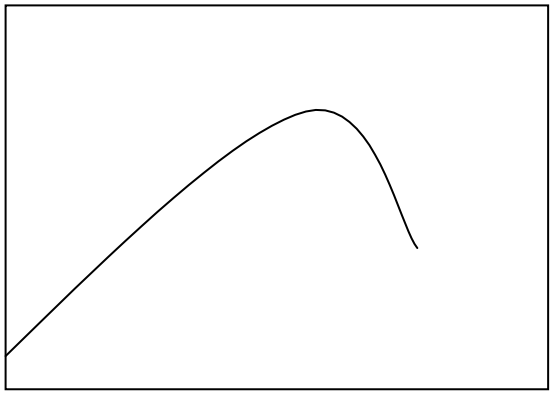
Round Robin (RR): This algorithm is one of the most classical static load balancing algorithms. In the Round Robin [23], flows are assigned evenly among all switches. Each new flow is assigned to available switch in the round robin order. Furthermore, the flow allocation order is locally maintained on each switch's flow table.

X. System Setting

A. FPN model (fully populated): In the FPN network, there are two hundreds of core switches, four hundreds of aggregation switches and four hundreds of ToR switches. Each ToR switch directly connects with four hosts.

B. FTN model (fat-tree): This network consists of two hundreds of core switches and two hundreds of pods. Each pod contains two aggregation switches and two end switches.
Each end switch
directly connects with
five hosts.

Evaluating the performance



XI. RELATED WORK

A. Flow Scheduling:

Existing researches on flow scheduling can be classified as static and dynamical schemes. Static load balancing schemes distribute the traffic mainly based on a fixed set of rules according to characteristics of the input traffic, which can not feedback real-time information of traffic and network states on links.

B. OpenFlow-Based and Data Center Oriented Scheduling:

OpenFlow is a leading software-defined networking architecture, which allows for quick experimenting and optimizing switching/routing policies. Handigol et al. [8proposed the OpenFlow-based LOBUS algorithm to balance the load, which applies greedy selection strategy to pick the (server, path) pair that yields the least total response time at every request. Anitha et al. proposed an idea similar to flow tables in OpenFlow switches, which applies a load table on the Dispatcher node to record changing states and then applies corresponding transmission policy

XII. CONCLUSIONS AND FUTURE WORK

In this work, we address the load-balanced scheduling problem through balancing transmission traffic dynamically and globally in cloud data centers. Aiming at two typical OpenFlow architectures: FPN and FTN, we proposed and implemented a set of efficient scheduling algorithms DLBSFPN and DLBS-FTN respectively. Compared with existing scheduling schemes for load balancing and route selection, our algorithms have two main advantages. Firstly, our algorithms can adapt to dynamical network states and changing traffic requirements through updating load imbalance factor $\delta(t)$ and accordingly balancing the transmission load slot by slot during data transmissions. Next, our algorithms can globally balance transmission traffic in the whole network by means of evaluating link, path and network bandwidth utilization ratio proposed in this paper.

REFERENCES

[1]. Z.Z.Cao, M.Kodialam and T.V.Lakshman. Joint Static and Dynamic Traffic Scheduling in Data Center Networks. in Proceedings of IEEE INFOCOM 2014, pp.2445-2553.

[2]. J.Lu, D.Li, Bias Correction in Small Sample from Big Data, IEEE Transactions on Knowledge and Data Engineering, Vol.25, No.11, 2013, pp.2658-2663.

[3]. 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, 60(3) (2013) 700-706.

[4]. 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.

[5]. 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.

[6]. K.Greene, TR10: Software-Defined Networking, MIT Technology Review, Retrieved Oct. 7, 2011.