5th International Conference on Ambient Systems, Networks and Technologies (ANT-2014)

Loss rate control mechanism for fan-in-burst traffic in Data Center Network

Antriksh Goswami\textsuperscript{a}, K. K. Pattanaik\textsuperscript{a}, Amit Bharadwaj\textsuperscript{a}, Sourabh Bharti\textsuperscript{b}

\textsuperscript{a}Atal Bihari Vajpayee-Indian Institute of Information Technology and Management, Gwalior, India
\textsuperscript{b}Tata Consultancy Services, Noida, India

Abstract

Congestion scenarios in Data Center Network (DCN) arise due to burst traffic and cause packet drop to take place thus reducing the overall throughput. Flow scheduling techniques in DCN do not address well the network congestions. Congestion control techniques use congestion notifications from network core to deal with congestion scenario. Software defined networking techniques use link load information in access switches to react to congestion scenarios. Both the mechanisms operate on post-congestion scenario to deal with sustained burst traffic. In fat tree topology based DCN architectures proactive measures for handling burst traffic at lower layers will be more beneficial. In this paper, we implement traffic shaping mechanism in the edge switch at source that act proactively and prevent the propagation of ill effects due to sustained burst. Further, we evaluate its impact on the overall packet loss and delay. The entire DCN is simulated using Colored Petri Nets (CPN). The packet loss rates observed at the receiver edge switch for various flow patterns reveals cent percent packet transfer which signifies the effectiveness of the proactive congestion control mechanism.

© 2014 Published by Elsevier B.V. Open access under CC BY-NC-ND license.
Selection and Peer-review under responsibility of the Program Chairs.

Keywords: Data center network traffic; Congestion; Loss rate; Delay; Dual-leaky bucket

1. Introduction

In the present time DCNs, fat tree topology is being used globally and the link over-subscription ratio becomes 16:1 as packets move from Top-of-Rack (ToR) switch to Core switches (CS). Due to the link over-subscription packets experience heavy congestion at the higher layers causing increased delay and packet drop. Several static and dynamic flow scheduling techniques aiming at effective link utilization are proposed like Equal Cost Multiple Path (ECMP)[1], Valiant Load Balancing (VLB)[2], Global First fit[1]. Post scheduling scenario can also result in congestion and the above research do not mention about the congestion issues.

\* Corresponding author. Tel.: +91-845-891-9374 ;
\* E-mail address: goswamiantiksh@gmail.com (Antriksh Goswami), kkpatnaik@iiitm.ac.in (K.K.Pattanaik), amit5082930@gmail.com (Amit Bharadwaj), bharti.sourabh90@gmail.com (Sourabh Bharti)
Single congestion control (CC) mechanism and software defined networking are proposed in [3] for DCNs. CC uses congestion notification messages to inform the source in order to limit its rate of packet emission. Software defined networking techniques use link load information in access switches to react to congestion scenarios. Although the focus of these mechanisms is to handle the congestion in the DCN, these are reactive in the sense that they act post congestion scenario. The focus here is on traffic shaping mechanisms that act proactively to burst traffic at the lower layers to prevent the propagation of the ill effects to the higher layers. The traffic shaping mechanism implements a dual leaky bucket which first shapes the arrival rate to ensure no buffer overflow. Second to mitigate the problem of long-waits for small flows it allows flow classification to provide different flow rates to different sized flows. Long-wait is a scenario that arise while a link is occupied by very large flows and small flows wait for the availability of link. Flow classification provides different token rates to flows of different sizes thus reducing long waits for small flows.

The rest of this paper is organized as follows. Section 2 discuss the related work. Proposed mechanism is presented in Section 3. Section 4 discusses the experiment design and simulation. Results and analysis are discussed in Section 5. Finally Section 6 concludes the paper.

2. Related Work

Static flow scheduling techniques [1,2] and the dynamic scheduling techniques (global first fit) used in Hedera [1] mostly focus on flow scheduling with an aim to increase link utilization. Our earlier work DDFS [7] is an enhancement to the above and attempts to load balance and fair link utilization in DCN. However, none of the above research discussed the congestion issue.

Right-sized static buffers in the network makes it responsive to sustained congestion. On the other hand a dynamic shared buffer pool [5] with adaptive thresholding provides burst absorption capabilities to momentary burst traffic. Adding arbitrarily large buffers in the network switches [5] may prove detrimental to latency and responsiveness of applications, as well as significant additional cost. Authors [5] suggest all the links’ bandwidth to be increased to a high capacity related to the flow size of the nodes which is impractical due to various reasons. Application layer flow differentiation [3] sends congestion notification messages back to the source of the congestion to require a lower data sending rate. However, we feel, shaping traffic at the lower layers is a proactive mechanism for congestion control and is essential in the case of link over subscription scenarios like in DCNs. DeTail [14] Used a cross layer approach for dealing with long tail flows. Flash congestion scenario is handled at the link layer whereas network level load balancing done at the network layer. This reduced the likelihood of congestion. D3 (Deadline driven delivery) [12] provides the fraction of bandwidth to the flows on the basis of their deadlines and size. This may result in failure to meet the deadlines for the near deadline flows. Whereas D2TCP (Deadline aware Data Center TCP) [12] resizes its congestion window size according to gamma correction function which allows to meet the deadline. In D2TCP whenever the congestion occurs the near deadline flows backoff partially or not at all. However the far deadline flows backoff aggressively. PDQ (Preemptive distributed quick) flow scheduling [13] uses both Shortest Job First and Earliest Deadline First mechanisms to achieve minimize flow completion time and flow deadline. It works on preemptive philosophy for rescheduling the flows with different deadline requirements. To the best of our knowledge no report is found in the literature regarding traffic shapers to mitigate congestion and packet loss scenarios in DCNs.

3. Proposed Mechanism

Consider a burst traffic scenario when multiple nodes each sending multiple flows simultaneously with their maximum rate are destined for a single node. The burst traffic may be a short burst or a sustained burst. Techniques to deal with short burst in DCN makes the use of adequate provisioning of buffer at ToRs [5]. But such a technique cannot handle sustained burst traffic and creates congestion at different levels in the DCN and particularly its effect is more in terms of packet drops at the destination edge switch link or the bottleneck link. The various congestion control techniques trigger only when congestion like scenario is detected and may sometime also lead to packet drops despite of the mechanism is in place. The simple reason for this is due to the burst traffic offered by the source nodes. Our intuition is if the flow rate can be constrained at the source side as a function of total capacity available at destination edge (i.e., buffer size plus link capacity) then congestion will never take place and so the packet loss. While it is
possible to deal with short bursts by buffer provisioning technique[5], the sustained burst traffic can be dealt through rate control mechanism like leaky bucket[9].

Example packet loss scenario in case of sustained burst traffic:
Let there be \( n \) nodes sending burst traffic for sustained time duration \( T \) (i.e., \( T > 1 \) unit) to a node with buffer capacity \( n \times \text{edge link bandwidth} \).

The receiving node can serve maximum burst traffic equal to the provisioned buffer of size \( n \times \text{edge link bandwidth} \) plus its link capacity for the first time unit. Traffic pertaining to the remaining \( T - 1 \) time units experience packet loss (see Fig. 5). Hence, the buffer provisioning approach alone to deal with sustained burst traffic is not adequate. Thus, if the total burst at source can be shaped to an amount proportional to the number of nodes sharing an outgoing link then the sustained burst will not experience packet loss throughout. In the fat-tree structured DCN with \( k \) source-destination pairs \( n \) nodes sending burst traffic with burstiness \( b_i \) simultaneously to a switch and at rate \( r_i \) proportional to their respective edge links bandwidth over a period \( t \), the gross traffic incident on that switch can be handled successfully when it is less than or equal to its buffer capacity \( B \) and its service rate \( C_i \). Failing to meet the inequality packet loss takes place. Mathematically it can be represented by Eq.(1)

\[
\sum_{i=0}^{n} b_i + r_i \ast t \leq B + C_i \ast t \tag{1}
\]

Whereas the objective function for minimizing the packet loss rate is given by Eq.(2)

\[
O = \text{Minimize} \frac{\sum_{i=1}^{n} C_i \ast t - (C_i \ast t + B)}{\sum_{i=1}^{n} C_i \ast t} \tag{2}
\]

Here \( C_i \) is the long term average rate of sender \( i \).

The proposed mechanism is to limit the traffic arrival rate at the edge switch of sender(s). For a given R-SPEC (Request Specification) in terms of minimal packet loss rate the T-SPEC (Traffic Specification) parameters are the token rate and bucket size. The significance of token rate is to control the average rate of traffic flow while that of the bucket size is the allowable burst. Let \( A(t) \) be the cumulative sum of arrivals in time interval \([0, t] \) defined as

\[
A(t) = \sum_{i=0}^{n} C_i \ast t \tag{3}
\]

Where \( A(t) \) is continuous and \( A(t) = 0 \) for \( t \leq 0 \).

To shape the arrival function \( A(t) \) at the edge switch of the sender(s) we define a function \( \alpha^u \in F \) which applies an upper bound to \( A(t) \). Similarly the lower arrival curve for \( A(t) \) can be zero in the event of node(s) not generating any traffic. Thus the arrival curve shall operate on \( A(t) \) between bounds \( \alpha^u \) and \( \alpha^l \) to give an outgoing flow \( A^*(t) \). In general \( A(t) \geq A^*(t) \).

where

\[
\alpha^u = \sum_{i=0}^{p} b_i + r_i \ast t \tag{4}
\]

Where \( p \) is the number of ports at the edge switch and \( b_i \) is the allowable burst for port \( i \) and \( r_i \) is the long term average rate for port \( i \). The mechanism implements dual leaky bucket and works in two phases. In the first phase it shapes \( A(t) \) and the second phase classifies the flows on the basis of their size. Shapping process constitutes two process as given in Algorithm 1 and Algorithm 2. In Algorithm 1 allowable traffic is constrained by the available token in the bucket. By using Algorithm 2 token is added to the bucket at each time instant till the bucket is full. Similarly, the same mechanism is used for adding tokens to long lived bucket with rate \( rLong \) and to short lived bucket with rate \( rShort \) at each time instant till the bucket is full. Flow classification causes reduction in delay for the short lived flows by providing high token rate on the basis of desired delay specification. Short lived flows are characterized by smaller number of packets and relatively high burst traffic. Whereas long lived flows are characterized by larger number of packets and relatively low burst traffic. Hence, we provisioned high token rate and relatively large bucket size for
short lived flows. On the other hand low token rate and relatively small bucket size for long lived flows. Algorithm 3 classifies the flows and provides different token rate to long lived and short lived flows on the basis of available tokens in their respective buckets.

Algorithm 1: \texttt{ShapeTraffic(destAddr, srcAddr, A(t), availableToken, packetSize, flowSize)} procedure to shape the traffic

Data: \texttt{destAddr, srcAddr, A(t), availableToken, packetSize, flowSize}

Result: Shaped traffic \( A^*(t) \)

\begin{verbatim}
begin
  if \texttt{destAddr.edge = srcAddr.edge} then
    Return flow to dest from edge
  else
    if \texttt{flowSize < availableToken} then
      Then transmit all the flow to the classifier
    else
      Transmit \( \lceil \frac{\texttt{availableToken}}{\texttt{packetSize}} \rceil \) number of packets
end
\end{verbatim}

Algorithm 2: \texttt{AddTokenToBucket(availableToken, rateofToken, bucketSize)}

Data: \texttt{availableToken, rateofToken, bucketSize}

Result: \texttt{tokenBucket}

\begin{verbatim}
begin
  foreach \texttt{timeInstant} do
    if \( \texttt{(availableToken + rateofToken)} > \texttt{bucketSize} \) then
      Add \( \texttt{(bucketSize – availableToken)} \) to \texttt{tokenBucket} and discard
      \( \texttt{(rateofToken – (bucketSize – availableToken))} \)
    else
      Add \texttt{rateofToken} to the \texttt{tokenBucket}
end
\end{verbatim}

4. Experiment Design and Simulation

The proposed mechanism is simulated using CPN. We explain the experiment design and simulation setup in the following subsections. Since DCNs traffic traces are not publicly available due to privacy and security concerns, we model patterns that characterize DCN traffic. Data center traffic flows are characterized in two categories; small or short-lived flows and large or long-lived flows\[1\]. Large flows are very less in number as compared to small flows\[4,10\]. Packet sizes vary depending on application specific flow and it follows discrete random distribution. The sustained burst traffic is modelled as flow arrival at constant rate over an interval. To generate a traffic of 10Mbps at a constant rate, given discrete distribution of packet size over \([64, 1500]\), number of packets as Poisson distributed with rate 1000 packets per unit time; the Poisson distribution of packets would become 100 packets per 0.1 unit time. Thus if packet size is set to more than 1000 bytes then this provides burst traffic of more than 8 Mb with probability of 0.713 which we consider as sustained burst and is shown in the form of cumulative distribution function of inter arrival times in Fig 3.

We simulate the scenario where multiple nodes sending data simultaneously to a single destination node. The offered traffic by the sender nodes is bursty and is modeled as a combination of micro burst and sustained burst. This traffic
Algorithm 3: Classifier(availableTokenLong, availableTokenShort, flowSize, packetSize)

Data: availableTokenLong, availableTokenShort, flowSize, packetSize
Result: Shaped traffic for the upper layer

begin
if (flowSize > 70000Mb) then
  if flowSize < availableTokenLong then
    Then transmit all the flow to the aggregate switch
  else
    Transmit \( \left\lfloor \frac{availableTokenLong}{packetSize} \right\rfloor \) number of packets
  end
else
  if flowSize < availableTokenShort then
    Then transmit all the flow to the aggregate switch
  else
    Transmit \( \left\lfloor \frac{availableTokenShort}{packetSize} \right\rfloor \) number of packets
end

when comes across a bottleneck link associated with the destination node creates congestion leading to packet drops. To deal with this scenario the proposed traffic shaping mechanism is implemented at source edge switch. This enabled the flow control in accordance with the destination nodes’ edge switch capacity. Although this is a static mechanism but ensures the desired R-SPEC for the flow. Our DCN is structured around the fat-tree topology. It comprise of four pods and a pod is further comprise two ToRs and two aggregate switches. The Hierarchical net of our simulation is shown in Fig 1 and the corresponding simulation parameters are in Table 1. The dual leaky bucket mechanism’s subnet is given in Fig 2.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology</td>
<td>Fat-Tree</td>
</tr>
<tr>
<td>RTT(2 time units)</td>
<td>Propagation delay</td>
</tr>
<tr>
<td>( \alpha_u = 5 \text{Mbps} )</td>
<td>Arrival curve parameters</td>
</tr>
<tr>
<td>( \alpha_l = 0 \text{ Mbps} )</td>
<td></td>
</tr>
<tr>
<td>Maximum flow/node = 500</td>
<td>Traffic</td>
</tr>
<tr>
<td>Packet range</td>
<td>Poisson distributed with mean 100 packets per 0.1 unit time</td>
</tr>
<tr>
<td>Node to edge link capacity</td>
<td>10Mb</td>
</tr>
<tr>
<td>T-SPEC Parameters:</td>
<td></td>
</tr>
<tr>
<td>Token rate = 5Mbps</td>
<td>Maximum allowable burst traffic</td>
</tr>
<tr>
<td>Bucket size = 20 Mb</td>
<td></td>
</tr>
<tr>
<td>Edge switch Input Buffer 30 Mb</td>
<td>For Handling micro burst[5]</td>
</tr>
</tbody>
</table>

Table 2. Congestion scenario at E1-1.

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
<th>Route</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mac 5</td>
<td>Mac 1</td>
<td>5-E3-A3-C1-A1-E1-1</td>
</tr>
<tr>
<td>Mac 6</td>
<td>Mac 1</td>
<td>6-E3-A4-C3-A2-E1-1</td>
</tr>
<tr>
<td>Mac 7</td>
<td>Mac 1</td>
<td>7-E4-A3-C2-A1-E1-1</td>
</tr>
<tr>
<td>Mac 8</td>
<td>Mac 1</td>
<td>8-E4-A4-C4-A2-E1-1</td>
</tr>
</tbody>
</table>

5. Results and Analysis

We took a scenario as shown in Table 2 where four nodes (see Fig 1.) sending data to a single node on different routes simultaneously.
For analysing the effect of different queue size over loss rate we generate the constant rate cumulative traffic from all four nodes at the rate of 32 Mb/time unit. For this we fix the size of packets to 1000 bytes and the rate of packet generation to 1000 packets/time unit at each sending node. The traffic are monitored at places in CPN edge switch and nodes. We found that no matter how large queue size is, the packet drop will always be there for sustained burst traffic, however the delay after which the packet drop start will vary according to the flow size as shown in Fig 5. Queue size can’t be taken more than 40 Mb in our case as it crosses the limit of $2C \times RTT$ (double of normal queue size) and if we decrease, the packet drop will start early. Optimum queue size estimation has been an important research. So the other way out is to control traffic at the source in such a way that a congestion scenario at destination edge switch never occurs.
We apply the Dual leaky bucket mechanism on the same traffic scenario with token rate of 2.5 Mb/time unit and bucket size of 10 Mb for each node. Thus the leaky bucket at edge switch will have token rate of 5 Mb/time unit and bucket size of 20 Mb (as there are 2 nodes per edge switch). We first analyze our mechanism with 20 Mb queue size. The aggregate available token at the source side initially is 40 Mb. The traffic generated by the nodes is 32 Mb which utilizes the token and arrived at destination edge switch in first time unit of which 10 Mb is transferred and 20 Mb is stored in queue, remaining 2 Mb data is lost. For the second time unit 10 Mb token are added to the bucket so remaining tokens of 18 Mb is utilized by the sender edge switch and 18 Mb data arrived at the destination edge switch of which 10 Mb is transferred and remaining 8 Mb is lost because queue is already full. After first time unit the incoming traffic will be proportional to the set token rate (10 Mb/time unit in our case) which will ensure flow control limited to 10 Mb/time unit and thus there will be no loss at the bottleneck switch link as shown in Fig 6. With queue size 20 Mb the initial loss is because the 20 Mb queue is not capable to serve the allowable burst of 40 Mb. A number of experiments were carried for different queue sizes and their corresponding loss rate have been recorded. We further examined the effect of different token rates for 10 Mb bottleneck link capacity (see Fig. 7) and found that for different flow classes the token rates can be set to a value depending on the bottleneck edge switch link capacity.

Overall the mechanism applied to data center traffic serves well in terms of proactive on congestion scenarios.

6. Conclusion

The major findings of our work are that by using different queue sizes we can handle the micro burst traffic but it is not efficient for the sustained burst traffic. The traffic shaping mechanism with effective token rate (i.e., first leaky bucket) can mitigate the overall loss due to sustained burst traffic. This proactive mechanism works on the congestion causing sustained burst and does not cause packet loss and overall outperforms other reactive mechanisms. However
it adds delay to the transmission where in we feel this mechanism is acceptable in scenarios where application flows are not critically deadline constrained and heavy tailed. Although delay is another important RSPEC parameter we are working on this to propose a mechanism that can deal with loss rate and delay collectively. Further, the short lived and long lived flows are classified based on their size to have different token rates at the second leaky bucket resulting in low flow completion times for short lived flows.

References