面向带宽碎片最小化和QoS保障的数据中心网络流量调度算法

唐 宏 王欣欣* 刘亦星

(重庆邮电大学移动通信技术重庆市重点实验室 重庆 400065)

摘 要：随着数据中心网络流量的迅速增长，如何提高数据中心网络性能和服务质量成为了研究热点。然而现有的流量调度算法在网络负载加大时，一方面会导致网络带宽碎片化从而使得网络吞吐量降低，另一方面忽视了流量应用需求导致网络服务质量较差。为此，该文提出一种面向带宽碎片最小化和QoS保障的动态流量调度算法，算法综合考虑了带宽敏感的大流、时延与丢包敏感的小流的不同需求，首先根据待调度流的源地址和目的地址建立最短路径集，其次从中筛选出满足待调度流的带宽需求的所有路径，然后根据路径剩余带宽信息和小流应用需求情况为每条路径建立权重函数，最后根据权重函数值利用轮盘赌算法选择转发路径。实验仿真结果显示，与其它算法相比，所提算法降低了小流的丢包率和时延，同时在网络负载较大时提升了网络吞吐量。

关键词：数据中心网络；流量调度；带宽碎片；服务质量

中图分类号：TP393 **文献标识码**：A **文章编号**：1009-5896(2019)04-0987-08

DOI: 10.11999/JEIT180466

A Traffic Scheduling Algorithm for Bandwidth Fragmentation Minimization and QoS Guarantee in Data Center Network

TANG Hong WANG Xinxin LIU Yixing

(*Chongqing Key Laboratory of Mobile Communications Technology, Chongqing University of Post and Communications, Chongqing* 400065, *China*)

Abstract: With the rapid growth of Data Center Network (DCN) traffic, how to improve the performance and service quality of data center network become a research hotspot. However, when the network load increases, the existing traffic scheduling algorithm on the one hand may cause bandwidth fragmentation results in the network throughput decrease, on the other hand, it neglects the traffic application requirements to lead to poor QoS. Therefore, a dynamic traffic scheduling algorithm for bandwidth fragmentation minimization and QoS guarantee is proposed. The algorithm takes into account the different requirements of the bandwidth-sensitive large flows, and delay sensitive and packet-loss sensitive small flows. Firstly, the shortest path set is established according to the source address and destination address of the to-be-scheduled flow. Secondly, all the paths that satisfy the bandwidth requirement of the to-be-scheduled flow are selected. Then, the weight function is established for each path according to the free bandwidth of the path and the application requirements of the small flow. Finally, the forwarding path is selected based on the weight function value by roulette algorithm. The network simulation results show that when the network load increases, the proposed algorithm reduces the packet loss rate and delay of small flows, and improves the network throughput compared with other algorithms.

Key words: Data Center Network(DCN); Traffic scheduling; Bandwidth fragmentation; QoS

1 引言

近年来，随着虚拟化的广泛应用，云计算和物联网的不断发展，相关业务如并行计算、容灾备份等，需要大量服务器组成集群系统协同完成工作，这使得数据中心内部通信流量呈现指数级增长的趋势[1]。此外，随着互联网在线业务(如网页搜索、在线游戏、社交网络和视频流)的日益流行，用户对业务可靠性、时延保障方面也提出了更苛刻的诉求[2]。通常，数据中心网络中的流量主要有两种类型：(1)与用户任务(例如网页流量或搜索查询)相关的流

收稿日期：2018-05-16；改回日期：2018-11-16；网络出版：2018-12-04
*通信作者： 王欣欣 17782358734@163.com
基金项目：长江学者和创新团队发展计划(IRT_16R72)
Foundation Item: The Changjiang Scholars and Innovative Research Team Program in University (IRT_16R72)

量，该类型流量的特点是持续时间较短，对服务质量要求较高，例如有指定的流量完成时间(FCT)等；(2)通过虚拟机迁移，数据备份或分布式计算(MapReduce)等产生的流量，该类型流量的特点是传输字节数较大、持续时间较长，一般要求足够的吞吐量，但没有严格的时间限制[3]。这两种类型的流量混合在数据中心网络中，如何解决提高网络吞吐量和保证服务质量之间的矛盾成为一项挑战。因此，根据数据中心网络的流量特征设计流量调度算法，在提升用户服务质量的同时保证网络吞吐量，对于提升网络整体性能至关重要。

目前的数据中心网络流量调度算法主要分为两类：静态流量调度算法和动态流量调度算法。传统的静态流量调度算法如等价多路径路由(Equal-Cost Mutipath Routing, ECMP)算法[4]，通过对数据包头部散列值进行哈希运算，从而得到相应路径的端口地址进行转发。该策略虽然效率较高，但其将可用路径视为等成本路径的思想，没有考虑链路状态和流量特征，容易产生大流碰撞造成网络拥塞，从而导致网络吞吐量降低，并增大了小流的时延和丢包率。

目前的动态流量调度算法，通常是基于软件定义网络(Software Defined Networking, SDN)实现对网络的细粒度流量调度。SDN的控制层面与转发层面解耦，控制层面可以获取整个网络的负载信息，从而可以根据流量特征和需求进行更加灵活的管理和控制[5]。鉴于持续时间长、字节数多的大流是造成数据中心网络拥塞的主要原因，因此目前基于SDN的数据中心动态流量调度算法主要是针对大流的研究[6−11]，而持续时间短、字节数少的小流采取静态流量调度算法ECMP进行调度。如Hedera[7], Mahout[8], Nimble[9]等，都属于区分大小流的调度方案。大流的检测在边缘交换机(Hedera, Nimble)或者终端主机(Mahout)进行，然后根据流传输字节数是否超过阈值作为判别大小流的依据，最后为大流计算一条轻负载路径，如Hedera根据全网视图通过模拟退火(simulated annealing)算法计算大流转发路径。这几种基于SDN的动态流量调度算法，由于大流转发路径的剩余可用带宽都远大于该流本身所需带宽，在网络负载较轻时可以实现网络的负载均衡，提高网络链路利用率，但是当网络负载较重时容易造成带宽碎片化，导致多条路径剩余带宽之和大于当前流带宽需求，却无任何单条路径能够满足流带宽分配需求，从而造成网络拥塞和吞吐量降低[10]，而且忽略了小流的服务质量要求，导致用户访问数据中心的体验感急剧下降。

针对上述分析，本文首先提出最先满足带宽算法(First Meet the Bandwidth Algorithm, FMBA)，算法根据全网资源视图建立最短路径集，从中选择第1条满足大流带宽需求，同时保证小流服务质量的路径作为转发路径。该算法有效地降低了小流的时延和丢包，但是在网络负载加重时容易产生带宽碎片，降低网络吞吐量。因此，本文在FMBA的基础上进行优化，提出了一种面向带宽碎片最小化和QoS保障的调度算法(Bandwidth Fragmentation minimization and QoS guarantee, BFrag)，算法根据全网资源视图建立最短路径集，并从中筛选出满足大流带宽需求的所有路径，然后结合路径剩余带宽信息和小流应用需求情况建立权重函数，根据权重函数值利用轮盘赌算法选择转发路径。仿真验证该算法有效地降低了小流的丢包率和时延，同时在网络负载较大时保证了网络吞吐量。

本文第2节提出面向带宽碎片最小化和QoS保障的流量调度算法，首先介绍了该算法的系统架构，然后介绍了算法的具体设计；第3节进行仿真数据分析，验证算法合理性；第4节进行结论分析。

2 面向带宽碎片最小化和QoS保障的流量调度算法

2.1 算法的系统架构

所提算法的系统架构如图1所示，主要分为以下4个模块。

(1) 拓扑信息检测模块：该模块主要目的是获取网络拓扑，继而得到流的源交换机和目的交换机之间的K条最短路径。控制器通过链路层发现协议(Link Layer Discovery Protocol, LLDP)获取网络拓扑[12]。具体方法如下：

第 1 步　控制器在OpenFlow通道建立后，通过握手信息获取交换机的物理端口信息；

第 2 步　控制器向交换机周期性下发LLDP数据包，收到该数据包的交换机通过端口发送给邻接交换机。邻接交换机收到该LLDP数据包后，通过

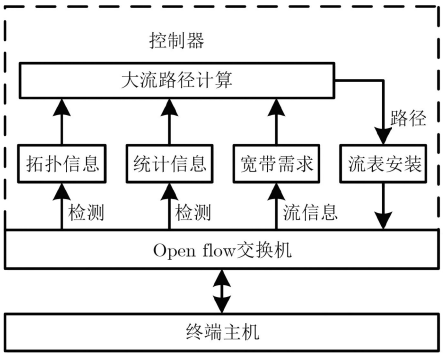

图 1 算法系统架构图

Packet-In消息发送给控制器；

第 3 步　控制器根据Packet-In消息以及保存的物理端口信息，获得交换机的连通信息，从而得到整个网络拓扑信息[11]；

得到网络拓扑信息后，控制器根据流的源交换机地址和目的交换机地址，基于跳数计算最优K条路径，组成K条最短路径集。

(2) 统计信息检测模块：该模块主要目的是获取大流信息和链路剩余带宽信息。主要方法是：控制器通过OFPT_STATS_REQUEST消息，固定时间间隔轮询每个OpenFlow交换机，查询并存储所有来自交换机的流信息和端口统计信息。

根据流信息记录并保存传输字节数变化量超过阈值的大流。传输字节数变化量计算方式为

$$\psi = \frac{b_{t_2} - b_{t_1}}{(t_2 - t_1)B} \tag{1}$$

其中，b_{t_2}, b_{t_1}分别表示交换机在t_2, $t_1(t_2 > t_1)$时刻接收到的字节数，B表示链路最大可用带宽，ψ表示流的速率占链路带宽的比例，通过对ψ进行界定即可区分大小流。本文设定，当ψ超过10%时，就认定该流为需要进行动态调度的大流[7]。

统计信息检测模块实时监测网络中各端口和链路的流量信息，进而计算出每条链路的可用带宽B_m为

$$B_m(t) = B - \mathrm{ld}_m(t) \tag{2}$$

其中，$\mathrm{ld}_m(t)$表示链路m上传输数据流已占用带宽值，B表示链路最大可用带宽。假设路径P包含m条链路，则该路径的可用带宽为

$$B_{\min} = \mathrm{Min}(B_1, B_2, \cdots, B_m) \tag{3}$$

该路径的链路最大剩余带宽为

$$B_{\max} = \mathrm{Max}(B_1, B_2, \cdots, B_m) \tag{4}$$

(3) 带宽需求检测模块：该模块的主要目的是获取大流的真实带宽需求。因为流的实际传输速率通常会受到网络传输的限制和主机的限制，因此并不能反映其真实带宽需求。网络传输的限制包括传输介质的最大传输速率、网络设备速率等，而主机的限制主要是发送端和接收端主机网卡的带宽限制[13]。带宽需求估算的目的就是计算得到网络流在网络传输无限制的理想情况下的自然带宽需求。在这种情况下，所有流的传输速率只会受到发送方网卡的发送速率限制和接收方网卡的接受速率限制，而不会受到中间网络链路的传输速率的限制[14]。需求估算函数的主要步骤如下：

第 1 步　根据流的源地址和目的地址集合组成一个$N \times N$的矩阵M。矩阵中第i行、第j列的元素由N_{ij}和B_{ij}组成，N_{ij}表示主机i发往主机j的大流数目，B_{ij}表示每条大流的带宽需求，这里假设发送主机将所有可用带宽资源均等分配给所有的大流；

第 2 步　S_i表示主机i的发送带宽限值，初始化为1。则第i行每条流发送带宽最大为$B_{ij}^1 = S_i \Big/ \sum_{j=1}^{N} N_{ij}$；

第 3 步　R_j表示主机j的接收带宽限值，初始化为1。则第j列每条流接受带宽最大为$B_{ij}^2 = R_j \Big/ \sum_{i=1}^{N} N_{ij}$；

第 4 步　若存在任意i, j使得$B_{ij}^1 \neq B_{ij}^2$，则$B_{ij} = \mathrm{Min}\left(B_{ij}^1, B_{ij}^2\right)$，更新矩阵$M$和剩余$S_i$和$R_j$，返回第2步。否则，进入第5步；

第 5 步　若存在任意i, j都有$B_{ij}^1 = B_{ij}^2$，输出矩阵M。

(4) 流表安装模块：该模块主要根据以上3个模块得到的K条最短路径信息、链路剩余带宽信息和大流带宽需求信息，根据流量调度算法计算得到大流转发路径，然后通过OFPT_FLOW_MOD消息向OpenFlow交换机下发流表，交换机根据流表中的路径信息完成转发。

2.2 流量调度算法设计

基于QoS保障的考虑，本文首先提出了最先带宽满足算法FMBA，基本思路是在路径集中随机选择第1条满足大流带宽需求，同时保证小流服务质量的路径作为转发路径。FMBA算法在数据中心网络流量负载较轻时，能够有效地降低小流的时延和丢包，而且大大减少了路径查找时间，降低了算法复杂度。但是通常情况下，该算法的大流转发路径的剩余带宽远大于流所需带宽，因此在网络负载较重时容易产生带宽碎片，降低网络吞吐量。

因此，在此基础上进行改进提出了基于带宽碎片最小化和QoS保障的流量调度算法BFrag。BFrag算法不仅考虑到小流的服务质量问题，还考虑到带宽碎片问题。算法的主要思路是，结合路径剩余带宽信息和小流应用需求情况为每条备选路径建立权重函数，根据权重函数值利用轮盘赌算法选择转发路径。

2.2.1 FMBA算法设计

假设从带宽需求估算模块得到待调度大流的带宽需求为B_n。由于流经过的链路数目越多，相对占用的资源越多，因此算法首先根据大流的源交换机地址和目的交换机地址，由信息统计模块计算出K条最短路径组成最短路径集P=$(P_1, P_2, \cdots, P_i, \cdots, P_k)$。然后为第$i$条路径建立函数$f_i$如式(5)

$$f_i = \mu B_{\min} - B_n, \mu \in (0.5, 1.0) \tag{5}$$

式中，B_n为大流的带宽需求，$B_{\min}$为第i条路径的可用带宽，μ为小流权重因子，取值范围为$(0.5, 1.0)$。该算法在最短路径集中随机找到第1条满足$f_i > 0$的路径作为转发路径。算法的伪代码见表1

表 1 FMBA算法伪代码

```
(1) for each path ∈ K_src→dst do
(2)   if B_min · μ > B_n
(3)   selectedPath ← path .
(4) return selectedPath
```

2.2.2 BFrag算法设计

已知K条最短路径集$P = (P_1, P_2, \cdots, P_j, \cdots, P_k)$，得到最短路径集中每条路径的剩余可用带宽$B_{\min}$后，筛选出满足大流带宽需求$B_n$的路径组成$m$条候选路径集$C = (C_1, C_2, \cdots, C_j, \cdots, C_m)$。

首先，为了解决带宽碎片问题，本文借鉴最佳适应(best fit)算法，优先选择与带宽需求最接近的路径，从而充分利用剩余带宽资源。在此基础上，结合FMBA算法中保证小流服务质量的方法，为第j条候选路径提出第1条选择函数${f_j}^1$

$$f_j^1 = \frac{\mu B_{\min} - B_n}{B}, \mu \in (0.5, 1.0) \tag{6}$$

其中，B_n为大流的带宽需求，$B_{\min}$为第j条路径的剩余可用带宽，B为链路最大可用带宽，μ为小流的权重因子。

其次，由于一条路径的剩余带宽仅取决于最小链路剩余带宽，而如果一条路径上的链路剩余带宽波动较大，不仅会导致剩余带宽大的链路产生一定程度上的资源浪费，而且对小流的时延抖动也会产生不利的影响。因此为第j条候选路径提出第2条选择函数${f_j}^2$

$$f_j^2 = \frac{B_{\max} - B_{\min}}{B} \tag{7}$$

其中，$B_{\max}$为路径上最大链路剩余带宽，$B_{\min}$为路径上最小链路剩余带宽，B为链路最大可用带宽。

综合以上两点，为第j条候选路径建立如式(8)的权重函数

$$\varphi_j = \alpha \cdot f_j^1 + (1 - \alpha) \cdot f_j^2, \alpha \in (0, 1) \tag{8}$$

其中，${f_j}^1$和${f_j}^2$分别是第j条路径的两条选择函数，α为选择函数的权重因子，φ_j为第j条候选路径的权重函数值。

最后，为了防止剩余带宽较大的路径由于等不到带宽需求相符合的大流而长期处于空闲状态，从而导致网络负载不均衡的情况出现。算法采用轮盘赌选择，又称比例选择算子，基本思想是路径被选择的概率与该路径的权重函数值φ_j成反比。具体步骤如下：

第 1 步 通过计算m条备选路径的权重函数值φ，得到第j条路径被选择的概率P_j为

$$P_j = \frac{1/\varphi_j}{1/\varphi_1 + 1/\varphi_2 + \cdots + 1/\varphi_j + \cdots + 1/\varphi_m} \tag{9}$$

第 2 步 计算第j条路径的累积被选择概率CP_j为

$$\mathrm{CP}_j = \sum_{n=1}^{j} P_n \tag{10}$$

第 3 步 随机产生一个0到1之间的随机数$\mathrm{r}(0, 1)$，如果第j条路径的累积被选择概率CP_j满足式(11)

$$\mathrm{CP}_{j-1} < \mathrm{r}(0, 1) \le \mathrm{CP}_j \tag{11}$$

则第j条路径即为大流转发路径。算法伪代码见表2。

表 2 BFrag算法伪代码

```
(1)  for each path ∈ K_src→dst do
(2)    if B_min > B_n
(3)      candidatePath.add(path);
(4)  if candidatePath is empty then
(5)    k+ = 1
(6)  for each p ∈ candidatePath do
(7)    weight1 ← (B_min · μ − B_n) /B
(8)    weight2 ← (B_max − B_min)/B
(9)    weight ← weight1 · α + weight2 · (1−α)
(10)   totalProbability+ = 1/weight
(11)  for each p ∈ candidatePath do
(12)    probabilty ← totalProbability/weight
(13)    cumulativeProbabilty+ = probabilty
(14)    while cumulativeProbabilty ≥ fslice do
(15)      selectedPath ← p
(16)  return selectedPath
```

3 仿真结果

为了验证算法的效果，本文采用轻量级仿真实验平台Mininet来搭建Fat-Tree网络，在Ryu控制器上进行实现。Fat-Tree拓扑结构如图2所示，它由于结构简单、易于部署和扩展，在数据中心网络中认可度较高。此外，本实验采用文献[15]提供的数据中心网络各种通信模式的测试套件作为本实验的流量产生模式。

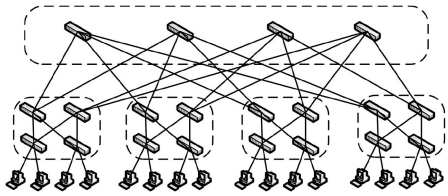

图 2 Fat-Tree拓扑结构图

3.1 仿真参数设置

本节将所提算法BFrag和MFBA，与Hedera和ECMP进行实验对比，主要从网络吞吐量，小流往返时延和丢包等5个方面进行测试和验证。利用Mininet产生4个传输区(point of delivery, pod)，4个核心交换机的Fat-Tree结构，交换机之间链路带宽设置为1 Gbps。每个主机按照流量模型概率向其他主机发送数据流，使用Iperf发送数据模拟大流(默认发送数目为1)，每条大流持续时间为60 s，占用带宽在100 Mbps-1 Gbps之间；使用Ping发送数据包模拟小流(默认发送数目为600)，发送间隔为0.1 s，每个数据包流持续时间在0.5 s以内，占用带宽在50 Mbps以下。

通过增加每个主机发送大流的数目来增大网络负载，使用网络监控软件bwm-ng进行实时网络负载监控，网络性能测试时常60 s。算法参数μ和α的设置将通过仿真实验进行分析确定。首先固定参数$\alpha=0.5$，参数μ的取值对算法影响的仿真实验结果如表3所示。当参数$\mu=0.7$时，算法整体效果较好。

同理，固定参数$\mu=0.7$，参数α的取值对算法影响的仿真实验结果如表4所示，从表中可以看出，当$\alpha=0.6$时，算法整体效果较好。因此在接下来的对比试验中，我们将设置参数值$\mu=0.7, \alpha=0.6$。

3.2 算法效果检验1：网络传输性能

网络的平均吞吐量指的是网络系统在当前流量模型下获得的单位时间吞吐量的平均值，网络的标准化吞吐量指的是网络系统在当前流量模型下获得的总吞吐量与该系统在最理想的流量模型下获得的最大吞吐量的比值，两者皆是衡量网络传输性能的重要指标。在图3中，分别给出所提算法BFrag与FMBA, Hedera和ECMP的平均吞吐量与标准化吞吐量随网络负载变化的情况，由图3(a)和图3(b)可以看出，由于BFrag算法在网络负载较轻时会造成一定程度的负载不均衡，因此平均吞吐量与标准化吞吐量均低于Hedera和FMBA算法。但是当网络负载持续增加(主机发送大流数目达到4以上时)，BFrag和ECMP算法逐渐呈现出更好的效果。而MFBA算法由于仅致力于保证小流的服务质量而忽略了带宽碎片，导致网络负载增加时吞吐量明显降低。

表 3　μ取不同值时算法结果对比

μ取值	0.6	0.7	0.8	0.9
平均吞吐量(Gbps)	7.821	7.756	6.601	6.416
标准化吞吐量	0.488	0.484	0.413	0.401
小流时延(ms)	14.43	12.69	17.50	10.18
小流时延偏差(ms)	19.27	14.82	10.64	12.82
小流丢包率(‰)	0.127	0.094	0.104	0.115

表 4　α取不同值时算法结果对比

α取值	平均吞吐量(Gbps)	标准化吞吐量	小流时延(ms)	小流时延偏差(ms)	小流丢包率(‰)
0.1	6.765	0.423	17.93	10.09	0.142
0.2	6.778	0.424	26.88	10.81	0.280
0.3	6.527	0.408	14.73	10.13	0.135
0.4	6.672	0.417	28.39	10.83	0.176
0.5	6.945	0.434	80.55	14.79	0.091
0.6	7.019	0.438	20.13	15.49	0.129
0.7	7.100	0.443	20.07	16.33	0.200
0.8	7.095	0.443	21.24	14.81	0.164
0.9	7.228	0.452	25.20	16.45	0.221

3.3 算法效果检验2：小流服务质量

平均往返时延指的是数据包从客户端发出到收到服务器端回复的数据包的平均时延，是检测流服

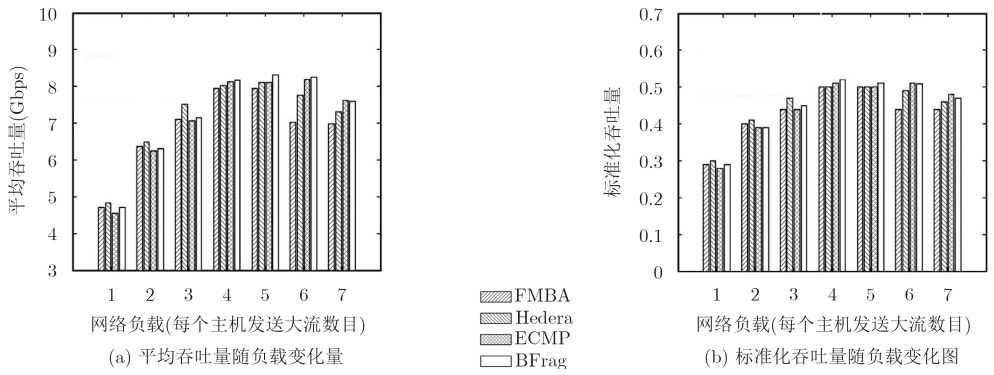

图 3 不同调度算法的吞吐量随负载变化图

务质量的重要依据。在图4中，分别给出所提算法BFrag与FMBA, Hedera和ECMP的小流平均往返时延随网络负载变化的情况。如图4所示，由于ECMP是将可用路径视为等成本路径随机转发大小流，容易出现将多条大流分配到同一条路径上的情况，导致小流的平均往返时延最高。Hedera算法采用静态调度小流，动态调度大流的方法，在一定程度上降低了小流的平均时延，但是大流调度策略只考虑到路径的剩余带宽是否满足大流带宽需求，忽视了网络中数目占比更大的小流对带宽的需求。而MFBA和BFrag算法，在同样采取静态调度小流，动态调度大流的基础上，在大流调度策略中提出了小流权重因子，链路剩余带宽不仅要满足大流带宽需求，还要为小流保留一定的剩余带宽，从而保证了小流的正常转发。因此MFBA和BFrag在平均往返时延方面相比Hedera有所改进，相比ECMP算法有明显的优势。

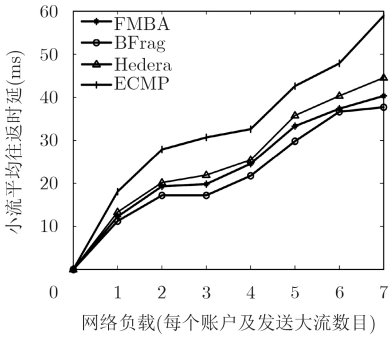

图 4 不同调度算法的小流平均往返时延对比

平均往返时延偏差指的是数据流的每一个数据包的往返时延与平均往返时延差的平均值，值越小说明数据包的往返时延分布得越集中，与时延抖动有一定的相似之处，音视频等数据的传输对此有较高的要求。在图5中，分别给出所提算法BFrag与FMBA, Hedera和ECMP的小流平均往返时延偏差随网络负载变化的情况。如图所示，FMBA和Hedera都是动态调度大流，因此时延偏差无明显区别。ECMP是大小流等概率随机发送所以剩余带宽分布较平均，时延偏差低于FMBA和Hedera。BFrag算法在为大流计算转发路径时，提出函数f_j^2优先选择链路剩余带宽波动较小的路径，使得网络链路上的剩余带宽分布较为平均，因此与其他3种算法相比明显降低了小流的往返时延偏差。

丢包率指的是数据包丢失发生的概率。在图6中，分别给出所提算法BFrag与FMBA, Hedera和ECMP的小流丢包率随网络负载变化的情况。发生网络拥塞是导致小流丢包的主要原因之一，由图6

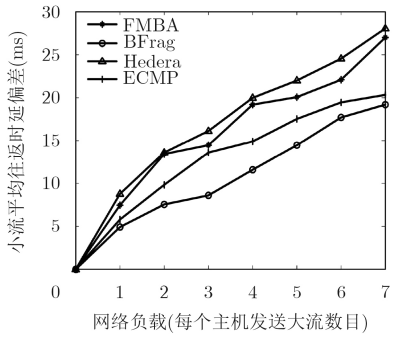

图 5 不同调度算法的小流平均往返时延偏差对比

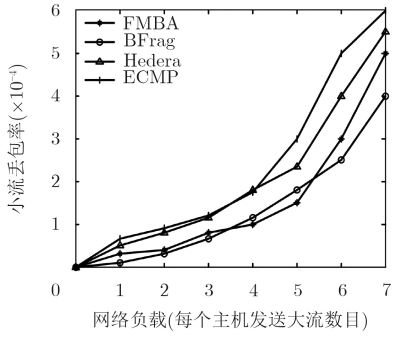

图 6 不同调度算法的小流丢包率对比

可以看出，在网络负载较轻时4种算法的丢包率普遍较低，而随着负载加重(主机发送大流数目达到5以上时)，网络发生拥塞概率增加，小流的丢包率也随之迅速增加。其中容易发生大流碰撞的ECMP丢包率最高。而BFrag和FMBA算法在调度策略中增加了小流权重因子，因此相比Hedera算法丢包率略低，但是FMBA是以牺牲吞吐量为代价，相比而言BFrag的综合性能更优。

3.4 算法复杂度分析

在Fat-Tree架构中，假设有k个pod主机组，则一共有$\left(k^3/4\right)$个主机，2台主机之间共有$(k/2)^2$条可选的最短路径。假设小流的总数为N，大流的总数为M，$(k/2)^2$条可选最短路径中剩余带宽大于大流带宽需求的路径有L_1条，剩余带宽与小流权重因子的乘积大于大流带宽需求的路径有L_2条。

ECMP将可用路径视为等成本路径，随机分配大小流，因此算法复杂度为$O(M+N)$。

Hdera的全局首次适应算法，从最短路径集中随机找到一条满足大流带宽需求的路径动态转发大流，用ECMP的方法静态转发小流，时间复杂度为$O\left(M\left((k/2)^2-L_1\right)\right)+O(N)$。

MFBA从最短路径集中随机找到一条剩余带宽与小流权重因子的乘积大于大流带宽需求的路径，则总时间复杂度为$O\left(M\left((k/2)^2-L_2\right)\right)+O(N)$。

BFrag首先从最短路径集中筛选出剩余带宽与

小流权重因子的乘积大于大流带宽需求的路径，然后为每条路径建立权重函数，用轮盘赌算法得到最终转发路径，因此计算大流转发路径的时间复杂度为 $O\left(M\left((k/2)^2+2L_2\right)\right)$，总的时间复杂度为 $O\left(M\left((k/2)^2+2L_2\right)\right)+O(N)$。

为使时间复杂度对比结果更加清晰，取流量产生模式中的stag_0.1_0.2为例产生流量(仿真主机向同一接入层交换机的其他主机以0.1的概率发送数据，向同一pod的主机以0.2概率发送数据，向其他剩余主机以0.7概率发送数据)，发送大流数目从1～5递增。4种算法的运行时间如表5所示，由于ECMP算法是静态流量调度算法，不需要控制器为大流计算路径，因此运行时间最低，而其他3种算法运行时间无明显差距。

表 5　算法运行时间结果对比(s)

发送大流数目	1	2	3	4	5
FMBA运行时间	198.34	258.81	300.12	349.97	403.88
BFrag运行时间	200.56	246.20	295.54	345.68	391.74
Hedera运行时间	201.39	255.50	298.81	343.10	392.15
ECMP运行时间	141.58	190.96	239.35	286.76	335.00

4　结束语

本文对现有的数据中心网络动态流量调度算法进行了分析和研究，首先提出了最先满足带宽算法—FMBA，该算法虽然有效提高了小流服务质量，但是在网络负载较大时容易产生带宽碎片，降低网络性能。因此本文对FMBA进行优化，提出了一种面向带宽碎片最小化和QoS保障的流量调度算法—BFrag，算法在最佳适应算法(best fit)基础上进行改进以减少带宽碎片，并结合链路剩余带宽波动情况建立权重函数，根据权重函数值利用轮盘赌算法得到最终路径。仿真实验表明，本文所提算法BFrag能够有效地保证小流的服务质量，并在网络负载较重时呈现出更高的网络吞吐量。然而，本文所提算法仅针对Fat-Tree型数据中心网络架构设计，并进行了实验分析，而在其他拓扑结构(如BCube, DCell)下的应用情况还需要进一步研究分析。

参 考 文 献

[1] GAO Yongqiang and WU Yonghao. Profit-aware workload management for geo-distributed data centers[C]. International Conference on Parallel and Distributed Computing, Applications and Technologies, Taipei, China, 2017: 60–66. doi: 10.1109/PDCAT.2017.00019.

[2] MRUDUAL S and SWAPNASUDHA K. A dynamic and energy efficient greedy scheduling algorithm for cloud Data Centers[C]. IEEE International Conference on Cloud Computiong in Emerging Markets, Bangalore, India, 2017: 1–3. doi: 10.1109/CCEM.2017.9.

[3] SONG Ziyan and ZHANG Ting. START: Sensible traffic scheduling in dynamic data center networks[C]. IEEE International Performance Computing and Communications Conference, San Diego, USA, 2017: 1–8. doi: 10.1109/PCCC.2017.8280435.

[4] ZHANG Hailong and GUO Xiao. SDN-based ECMP algorithm for data center networks[C] Computing, Communications and IT Applications Conference, Beingjing, China, 2015: 13–18. doi: 10.1109/ComComAp.2014.7017162.

[5] LI Cong and WU Yonghao. Strategy of data manage center network traffic scheduling based on SDN[C]. International Conference on Intelligent Transportation, Big Data & Smart City, Changsha, China, 2016: 29–34. doi: 10.1109/ICITBS.2016.61.

[6] LIU Jing and LI Jie. SDN based load balancing mechanism for elephant flow in data center networks[C]. International Symposium on Wireless Personal Multimedia Communications, Sydney, Australia, 2015: 486–490. doi: 10.1109/WPMC.2014.7014867.

[7] ALFARES M, RADHAKRISHNAN S, RAGHAVAN B, *et al*. Hedera: Dynamic flow scheduling for data center networks[C]. NSDI'10 Proceedings of the 7th Usenix Symposium on Networked Systems Design and Implementation, San Jose, USA, 2010: 19.

[8] CURTIS A, KIM W, and YALAGANDULA P. Mahout: Low-overhead datacenter traffic management using end-host-based elephant detection[C] IEEE INFOCOM, Shanghai, China, 2011: 1629–1637.

[9] 李龙, 付斌章, 陈明宇. Nimble: 一种适用于OpenFlow网络的快速流调度策略[J]. 计算机学报, 2015, 38(5): 1056–1068. doi: 10.3724/SP.J.1016.2015.01056.
LIN Long, FU Zhangshou, and CHEN Mingyu. Nimble: A fast flow scheduling strategy for OpenFlow networks[J]. *Journal of Computer*, 2015, 38(5): 1056–1068. doi: 10.3724/SP.J.1016.2015.01056.

[10] 陈琳, 张富强. 面向SDN数据中心网络最大概率路径流量调度算法[J]. 软件学报, 2016, 27(2): 254–260.
CHEN Lin and ZHANG Fuqiang. Maximum probability path scheduling algorithm for elephant flow in data center networks based on SDN[J]. *Journal of Software*, 2016, 27(2): 254–260.

[11] 段洁, 高江明, 程克非, 等. 基于流类型的SDN数据平面故障恢复算法[J]. 重庆邮电大学学报(自然科学版), 2018, 30(1): 134–140. doi: 10.3979/j.issn.1673-825X.2018.01.017.
DUAN Jie, GAO Jiangming, CHENG Kefei, *et al*. Failure

recovery algorithm based on flow type in SDN data plane[J]. *Journal of Chongqing University of Posts and Telecommunications*(*Natural Science Edition*), 2018, 30(1): 134–140. doi: 10.3979/j.issn.1673-825X.2018.01.017.

[12] 左青云, 陈鸣, 赵广松, 等. 基于OpenFlow的SDN技术研究[J]. 软件学报, 2013, 24(5): 1078–1097. doi: 10.3724/SP.J.1001.2013.04390.
ZUO Qingyun, CHEN Ming, ZHAO Guangsong, *et al*. Research on OpenFlow-based SDN technologies[J]. *Journal of Software*, 2013, 24(5): 1078–1097. doi: 10.3724/SP.J.1001.2013.04390.

[13] SONG Tao, LIU Yuchen, WANG Yiding, *et al*. Ashman: A bandwidth fragmentation-based dynamic flow scheduling for data center networks[J]. *Computer Journal*, 2017, 60(10): 1498–1509. doi: 10.1093/comjnl/bxx042.

[14] 林智华, 高文, 吴春明, 等. 基于离散粒子群算法的数据中心网络流量调度研究[J]. 电子学报, 2016, 44(9): 2197–2202. doi: 10.3969/j.issn.0372-2112.2016.09.026.
LIN Zhihua, GAO Wen, WU Chunming, *et al*. Data center network flow scheduling based on DPSO algorithm[J]. *Acta Electronica Sinica*, 2016, 44(9): 2197–2202. doi: 10.3969/j.issn.0372-2112.2016.09.026.

[15] VAHADAT A, AlFARES M, and LOUKISSAS A. Scalable commodity data center network architecture[P]. USA Patent, US8483096, 2013.

唐 宏：男，1967年生，教授，研究方向为计算机网络、移动通信.

王欣欣：女，1994年生，硕士生，研究方向为数据中心网络、软件定义网络.

刘亦星：男，1992年生，硕士生，研究方向为数据中心网络、软件定义网络.