软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2020,31(7):2221−2244 [doi: 10.13328/j.cnki.jos.006050]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

数据中心网络负载均衡问题研究*

沈耿彪[1], 李 清[2,3], 江 勇[1,3], 汪 漪[2,3], 徐明伟[4]

[1](清华大学 深圳国际研究生院,广东 深圳 518055)

[2](南方科技大学 未来网络研究院,广东 深圳 518055)

[3](鹏城实验室 网络通信研究中心,广东 深圳 518055)

[4](清华大学 计算机科学与技术系,北京 100084)

通讯作者: 李清, E-mail: liq8@sustech.edu.cn

摘 要: 数据中心网络是现代网络和云计算的重要基础设施,实现数据中心网络负载均衡是保证网络吞吐并提高服务体验的关键环节.首先分析了数据中心网络与传统互联网之间的区别,总结其特点及特殊性在负载均衡方案设计方面的优势.然后从数据中心的复杂性和多样性角度分析其负载均衡方案设计所面临的挑战.将现有数据中心网络负载均衡方案根据不同的实现层次从网络层、传输层、应用层和综合方案 4 个角度进行分析,对比各个方案的优缺点,并从控制结构、负载均衡粒度、拥塞感知机制、负载均衡策略、可扩展性和部署难度几个方面进行综合评价.最后对现有数据中心网络负载均衡方案进行总结,并指出未来可能的研究方向.

关键词: 数据中心网络;负载均衡;云计算;流量调度;传输管理

中图法分类号: TP303

中文引用格式: 沈耿彪,李清,江勇,汪漪,徐明伟.数据中心网络负载均衡问题研究.软件学报,2020,31(7):2221−2244. http://www.jos.org.cn/1000-9825/6050.htm

英文引用格式: Shen GB, Li Q, Jiang Y, Wang Y, Xu MW. Research on load balancing in data center networks. Ruan Jian Xue Bao/Journal of Software, 2020,31(7):2221−2244 (in Chinese). http://www.jos.org.cn/1000-9825/6050.htm

Research on Load Balancing in Data Center Networks

SHEN Geng-Biao[1], LI Qing[2,3], JIANG Yong[1,3], WANG Yi[2,3], XU Ming-Wei[4]

[1](Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China)

[2](Insitute of Future Network, Southern University of Science and Technology, Shenzhen 518055, China)

[3](PCL Research Center of Networks and Communications, Peng Cheng Laboratory, Shenzhen 518055, China)

[4](Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

Abstract: Data center networks are the important infrastructure of the modern Internet and cloud computing. It is critical to achieve load balancing in data center networks for guaranteeing high throughput and improving service experience. The difference between the data center network and traditional Internet is firstly analyzedand the features of data center networks and the facilitation for designing load balancing schemesare concluded. Then, the challenges of designing load balancing schemes are analyzedin data center networks from the perspective of complexity and diversity. Existing load balancing schemes in data center networksare classified into four types, i.e., the schemes based on the network layer, the transport layer, the application layer, and the synthetic designs, according to different modification types. The advantages and disadvantages of these schemes are detailedand they are evaluatedfrom the point of the control

* 基金项目: 国家自然科学基金(61872420); 广东省重点领域研发计划(2018B010113001); 鹏城实验室大湾区未来网络试验与应用环境(LZC0019); 深圳市软件定义网络重点实验室(ZDSYS20140509172959989)

Foundation item: National Natural Science Foundation of China (61872420); Guangdong Province Key Area R&D Program (2018B010113001); PCL Future Greater-bay Area Network Facilities for Large-scale Experiments and Applications (LZC0019); Shenzhen Key Laboratory of Software Defined Networking (ZDSYS20140509172959989)

收稿时间: 2018-09-28; 修改时间: 2019-05-08; 采用时间: 2020-02-17; jos 在线出版时间: 2020-04-21

structure, the granularity of load balancing, the congestion sensing mechanism, the strategy of load balancing, scalability and the difficulty of deployment. Finally, all load balancing schemes are summarizedand some future feasible directionsare presented.

Key words: data center network; load balancing; cloud computing; flow scheduling; traffic management

随着现代网络的发展和云计算的兴起,数据中心承载了当前互联网中的大部分数据流量和应用功能.用户对网络资源的需求不断增大,导致了数据中心的规模越来越大.作为数据中心的基础设施,数据中心网络(data center network,简称 DCN)由于其拓扑结构和传输能力的限制,使其在当前背景下实现用户所需的性能受到了前所未有的挑战.目前,国内外互联网巨头,包括 Amazon、Google、Facebook、Microsoft、Baidu、Alibaba 等,针对自己不同的性能需求,实现了其内部数据中心的数据存储、网络搜索以及大范围的网络计算等功能,为自己的用户提供了满足需求的保障.传统网络发展的半个世纪以来,涌现了许多提升网络传输性能并提高用户体验的机制,包括各种 TCP 版本和 QoS 机制等.然而,数据中心的特殊性使得这些机制并不能直接应用或效果很差.因此,为了使数据中心提供更好的服务,关键在于针对数据中心网络设计专门的负载均衡策略.

数据中心承载了不同用户的不同需求,对于不同用户的不同应用流量,理想情况下是为这些流量提供差异化服务.数据中心中的流量从其大小来看,可分为大流和小流,大小流的区分可根据流的传输数据量或流的持续时间来进行区别;从应用层面来看,可将流分为吞吐量敏感的流和时延敏感的流,需为吞吐量敏感的流提供足够的带宽保障,为时延敏感的流提供满足时延要求的调度;从流的时域特性来看,可将流分为突发流和稳定流,稳定流可利用流量工程方法进行长时负载均衡,而突发流的不稳定性会影响网络的整体性能.数据中心处理的流量种类繁多,从总体来看,数据中心最理想的状态是提供高吞吐量低时延的网络服务.对数据中心网络中的流量进行管理能够提高网络链路的整体利用率,降低网络的拥塞情况,减少传输过程中的重传,因此设计合理、高效的数据中心网络负载均衡方案十分关键.数据中心与传统互联网有较大不同,具有以下几个特点.

(1) 结构化的拓扑.为提高数据中心服务能力,并实现更好的网络可扩展性,数据中心网络采用结构化的、规整的拓扑来连接数据中心中的服务器.结构化的拓扑可以提供有保障的传输能力,并降低了网络的扩展难度.典型的结构化的数据中心网络拓扑包括 Clos[1]、VL2[2]和 Jellyfish[3]等,这些结构化的拓扑以图理论为基础,提供了可能的最大传输能力.在数据中心网络的负载均衡方案设计过程中,可利用网络结构化的特性,设计简单、有效的流量调度策略,以实现高效的数据传输.

(2) 易获取的流量需求.数据中心由服务供应商进行维护,为自己或用户提供服务能力,以满足不同应用的需求.因此,数据中心容易获得进入及离开其网络的流量统计信息,以及不同应用类型在其网络内部传输的流量.对于大部分应用来说,流量需求是可获知的,因此可利用已获得的网络流量信息,感知网络的传输情况,从而执行相应的负载均衡机制,实现最优的流量部署.

(3) 统一的管理机制.数据中心一般具有统一规格的网络设备以及一致的管理策略.运营商为网络设备设置计划的配置,部署相应的管理策略,并统一进行维护和升级.因此,在数据中心中设计专用的负载均衡方案,相较于互联网来说能够更容易地部署,管理难度和开销也更小.

数据中心的特殊性使得为其设计专有的负载均衡方案成为当前数据中心网络的研究热点.现有的数据中心网络负载均衡方案充分利用了数据中心的特点,在互联网负载均衡机制的基础上进行方案改进或重新设计,以解决数据中心网络对于高带宽低时延的根本需求.然而,随着数据中心规模的不断扩大,以及数据中心承载的功能的不断增加,原来满足需求的数据中心网络负载均衡方案难以实现新的目标,因此,在新的应用场景下设计合适的负载均衡方案亟待研究.已有的负载均衡机制从 4 个层面进行考虑,即网络层、传输层、应用层和综合方案.网络层负载均衡方案中,利用数据中心中的网络设备对网络状态和流量信息进行收集,实现流量的合理调度;传输层方案则从主机端对数据中心网络状态进行感知,调整发送端的发送速率或动作使得数据中心网络不出现或少出现拥塞情况,以此实现网络的负载均衡.应用层主要针对应用相关的服务器之间的负载均衡,通过不同的流映射机制平衡服务器的负载,从而保证数据中心的服务能力.综合方案则是通过跨层设计来提升数据中心网络负载均衡效果,保证传输性能.这些类型的负载均衡方案从不同层次来实现数据中心网络的高吞吐量低时延需求.本文对这些方案进行了总结,并对比了方案之间的联系、差异以及优缺点等.已有的数据中心网络负

载均衡方案并不能适用于所有应用场景,在不同的网络环境中,设计怎样的负载均衡策略需要进一步的探讨.

本文对数据中心网络的负载均衡方案进行了深入分析.首先,列举了数据中心网络负载均衡方案设计过程中存在的挑战,包括网络动态性、拥塞感知难度、包乱序、设备故障以及方案可部署性等几个方面的问题.然后,将目前已有的数据中心网络负载均衡方案分为网络层、传输层、应用层和综合方案 4 类,并对各层实现方案根据负载均衡粒度或不同负载均衡目标进行细分,对各种类型的负载均衡机制进行详细的分析,对比不同负载均衡机制的优缺点.之后,在这些分析的基础上,将各方案从不同优化角度、有效性、可扩展性和部署难度等方面进行评估,总结各个机制的性能差异.最后,总结已有的数据中心网络负载均衡机制,并提出了设计数据中心网络负载均衡机制的新思路,为负载均衡机制提供了可行的研究方向.

本文第 1 节对数据中心网络中设计负载均衡方案存在的挑战进行阐述.第 2 节对已有数据中心网络负载均衡方案进行分类,对各个方案进行总结并比较各个方案的优缺点.第 3 节对已有数据中心网络负载均衡方案进行对比和评估.第 4 节对本文进行总结,并提出未来设计数据中心网络负载均衡方案可能的研究方向.

1 数据中心网络负载均衡方案存在的挑战

数据中心网络作为连接数据中心服务器的基础结构,其传输性能的好坏直接决定了数据中心的服务能力.由于目前数据中心的复杂性和多样性,使得设计满足要求的数据中心网络负载均衡方案具有一定的挑战.

(1) 流量动态性.数据中心网络的流量是动态变化的.数据中心中少部分的大流产生了网络中的大部分流量,大部分的流为小流,持续时间为几百毫秒;同时,大部分的流量在 rack 内部传输,一个 rack 内每秒的动态流可能达到 10 000 条[4,5].大量的小流造成网络状态变化剧烈,使得流量调度呈现滞后性.数据中心中流量的传输方式并不是以固定模式进行发送,由于传输协议的控制,流量传输呈现明显的开关特性,违背了大部分负载均衡方案对流量发送稳定性的假设[6].同时,在某些数据中心中,流量传输的开关特性却并不明显[7],使得数据中心网络负载均衡设计问题变得更加复杂.

(2) 拥塞感知难度.负载均衡的目的是实现网络资源的充分利用,从而增大网络的传输能力.因此,负载均衡实现过程中首先要针对网络状态进行感知,对目前网络中各链路或各路径的拥塞情况进行度量,从而为数据包决定合适的传输端口.由于数据中心网络流量特有的高动态性,使得对网络拥塞的感知具有时滞性,即当前感知到的拥塞信息为过时的状态,在集中式的负载均衡方案中,这种影响尤为明显.因此,在负载均衡方案中,感知网络拥塞情况的准确性和及时性直接决定了负载均衡的性能.

(3) 包乱序.大多数网络负载均衡方案基于流粒度进行调度,基于流粒度的方案难以接近网络流模型的最优性能.数据中心中由于拓扑存在等价多路径,因此性能较好的数据中心网络负载均衡方案一般基于比流更小的粒度进行调度,在多个路径上进行数据传输.由于传输层协议的确认机制,这类将流进行分割后传输的方案会造成数据包的乱序问题,即接收端接收到的数据包顺序错乱.传输层将乱序包的确认视为网络丢包并进行重传,严重降低了数据的传输效率,增大了流的完成时间,同时也降低了网络吞吐量.因此,如何在提高网络吞吐量的同时减少包乱序,或设计相应机制降低包乱序的影响,是在设计数据中心网络负载均衡方案中必须考虑的问题.

(4) 设备故障.数据中心中部署着大量网络设备,随着网络规模的不断扩大,设备出现故障的情况变得频繁.网络设备故障对数据中心数据传输有很大影响,尤其是转发设备的故障会导致连接的中断,降低了数据传输的吞吐量,增大了网络时延.当网络设备发生故障后,负载均衡方案需及时对故障情况进行响应,对网络中被影响的流量进行重新调度,以实现网络的最优传输.因此,在设计数据中心网络负载均衡方案时,需考虑对网络故障情况的响应,避免由于网络故障导致网络特性变化而影响负载均衡方案的性能.

(5) 部署难度.数据中心虽然由运营商进行统一管理,但实际维护过程中,数据中心网络的升级并不十分频繁,而且在数据中心规模庞大的情况下,难以进行统一的更新.同时,数据中心网络设备替换周期较长,一些新的技术无法在旧的网络设备中实现.因此,在设计数据中心网络负载均衡方案过程中,需考虑方案的可部署性,避免对网络进行较大规模的改动,同时还需要考虑方案在实际部署过程中的成本开销等.

数据中心的特性,使得数据中心网络负载均衡方案的设计具有较大的挑战性,需要综合考虑网络的结构特

点和流量规律,并要兼顾实际部署过程中方案的可实现性,因此需要进一步的深入研究.

2 数据中心网络负载均衡方案

现有的数据中心网络负载均衡方案种类繁多,国内外学者从不同角度和不同层面设计了相应策略[8].负载均衡方案的设计主要从5个方面进行考虑.(1) 方案控制结构,可分为集中式控制和分布式控制;(2) 负载均衡粒度的选择,基于更小的粒度进行调度能够提高网络的利用率并实现更好的负载均衡效果;(3) 网络拥塞信息感知策略,可分为全局拥塞信息感知和本地拥塞信息感知;(4) 负载均衡策略,可分为主动负载均衡策略和被动负载均衡策略;(5) 负载均衡方案的部署难度,尽可能地减少对数据中心网络设备的修改以达到或接近所需要的性能需求.从数据中心网络负载均衡方案实现层面的角度来分,可将其分为网络层方案、传输层方案、应用层方案和综合方案4类.根据之前的分析,数据中心网络负载均衡方案分类角度如图1所示.之后,首先介绍了最常用的数据中心网络负载均衡基础方案,然后根据不同修改层次对各类方案进行详细的阐述和归纳总结.

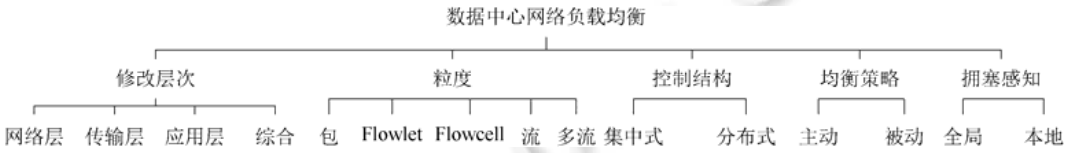

Fig.1 The classification of load balancing schemes in data center networks

图1 数据中心网络负载均衡方案分类角度

2.1 基础方案

数据中心网络由于规则的拓扑,端到端之间存在多条等价路径,因此通常采用ECMP[9]和WCMP[10]两种机制实现网络的负载均衡.这两种方案在交换机本地根据数据包头的哈希结果进行路径选择,实现简单且高效,目前已广泛在数据中心网络交换机中部署.

(1) ECMP

数据中心网络存在多条等价转发路径,使得负载均衡机制能够将流量均匀分配到各个路径上,实现网络传输能力的最大化.ECMP是数据中心网络中最常用的负载均衡方案.其在交换机本地实现哈希机制,交换机对到达数据包的包头域,比如五元组(包括了源IP地址、目的IP地址、源端口号、目的端口号以及协议号),进行哈希,然后根据哈希结果选择相应的转发端口进行数据转发.ECMP本质上是基于流的转发策略,其对相同流的五元组有相同的哈希结果,保证了流的转发一致性.同时,ECMP是一种主动的负载均衡机制,其主动地进行流哈希,不感知网络拥塞情况,对哈希结果的好坏不进行评价.ECMP在数据中心中的流量都是小流的情况下,性能很好且可接近最优的网络传输性能.然而,ECMP在采用哈希机制的过程中,会出现哈希冲突的情况.因此当网络中存在大流时,ECMP会出现大流冲突的现象,即多条大流哈希到同一条链路或路径,导致网络拥塞的情况发生,从而降低网络的吞吐量.虽然ECMP具有明显的缺点,但其简单且可部署性好,使其在数据中心网络中被广泛采用.

(2) WCMP

WCMP是针对ECMP存在的不平衡问题而提出的一种加权哈希负载均衡方案.ECMP机制设计过程中假设网络拓扑是对称的,网络中的流量是平衡的.然而实际情况中,这些假设大都无法满足.数据中心中由于网络设备的异构性或设备故障的发生,会导致网络产生不对称的情况;同时,数据中心中由于不同功能的服务器在机架上的部署,使得网络中的流量不是均匀分布的.在这些情况中采用ECMP进行负载均衡会造成网络传输的不平衡,降低网络的性能.WCMP对ECMP机制进行改进,利用权重对流哈希产生的交换机端口进行权重选择.网络中利用中央控制器进行网络状态感知,当发现网络出现不平衡性时,中央控制器计算出最优的权重分配并发送给相应交换机,交换机本地修改哈希权重执行数据包转发.WCMP可以通过修改权重的方式来应对各种网络不平衡的情况.然而,由于其采用集中式的网络状态感知方式,对不对称情况的感知有一定的周期,因此在动态性较大的数据中心中难以迅速响应网络的变化.但总体上来说,其能在一定程度上满足网络负载均衡的需求.

2.2 网络层方案

基于网络层的数据中心网络负载均衡方案是最直接、有效的解决方案,其通过修改数据包或流的调度机制直接影响网络中链路或路径的利用率.从负载均衡粒度角度来分,粒度从小到大可将网络层的负载均衡方案分为包粒度、flowlet 粒度、flowcell 粒度和流粒度等,其分类结果如图 2 所示.

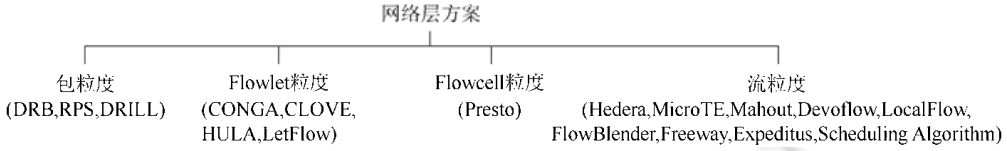

Fig.2 The classification results of the schemes base on the network layer

图 2 网络层方案分类结果

2.2.1 包粒度

基于包进行的负载均衡机制理论上是最优的策略.包作为网络传输的基本单元能够最大限度地平衡各链路的负载.然而,由于包乱序问题的存在,基于包粒度的负载均衡机制仍有许多问题需要解决.目前,典型的网络层基于包粒度的数据中心网络负载均衡方案包括 DRB[11]、RPS[12]和 DRILL[13,14]等.

(1) DRB

DRB(位逆转反弹)利用数据中心中 Clos 拓扑的结构特点实现基于包粒度的网内负载均衡.终端选择每个包转发路径时,选择 Clos 拓扑的顶层交换机作为反弹交换机,以确定上行转发路径和下行转发路径.因此,DRB 通过基于位逆转的顶层交换的选择策略,实现了包发送过程中链路的交错选择,从而实现了各链路的负载均衡,减少了网络中的排队长度以及避免了包乱序现象的出现.部署过程中,DRB 采用 IP-in-IP 包封装技术实现与顶层反弹交换机的数据交互,利用静态路由实现封装报文转发,并设计专门的拓扑更新协议实现终端对网络状态的感知.与传统的 RB(随机反弹)、RRB(轮询反弹)和 ECMP 相比,DRB 能够实现更大的吞吐量、更短的排队长度、更低的排队时延以及更小的 RTT 时间.与 MPTCP 相比,DRB 可在保证较高网络利用率的同时实现更低的平均时延.与 DeTail 相比,DRB 在吞吐量和时延方面的性能都更好.DRB 是一种主动负载均衡技术,通过网络结构特性平衡流量,然而,其无法对网络的拥塞情况进行感知,需要结合 TCP 实现终端的发送速率调节来处理网络拥塞情况.同时,在网络非对称情况下,DRB 的负载均衡性能取决于其对网络变化的感知速率.

(2) RPS

利用数据中心网络对称拓扑的特点,以及 TCP 的重复 ACK 和 DSACK 机制对少量乱序包的容忍能力,RPS 将流的每个包随机地通过所有等价路径进行发送.相同的源和目的地路径上的链路可组成一个等价类,等价类中的链路组成的路径具有相似的队列状态,因此能够保证近乎相等的时延.在网络对称的情况下,RPS 在不同的过订阅率和不同的流量模式场景中,相比于 ECMP 和 MPTCP 都具有更大的吞吐量和更小的流完成时间.此时,网络中各路径的排队长度差别很小,而且 RPS 造成的重复 ACK 相较于 ECMP 也要少.然而实际数据中心网络中经常存在非对称的情况,此时,RPS 利用的对称拓扑特性被破坏,造成乱序包大量增加,从而大大降低了网络吞吐量.RPS 对被非对称影响的所有流的包进行标记,设计 SRED 机制在交换机中对标记的包执行 RED 策略,实现主动的队列管理,从而保证路径时延的近似.通过非对称场景下的实验验证了 SRED 的性能,相比于 Droptail 和 RED 机制能够实现更高的吞吐量和更小的流完成时间,相比于 MPTCP 也具有明显的优势.然而,由于非对称场景出现情况复杂,RPS 的队列管理机制可能造成网络利用率下降,而且 SRED 的丢包会进一步降低网络吞吐量.

(3) DRILL

DRILL 是一种分布式的、随机的网内本地负载均衡机制,在交换机本地存储对称路径集合.当一条流的包到达时,DRILL 利用 power-of-two-choices[15]机制,首先利用包的五元组哈希并根据权重选择一个对称路径集合,然后随机选择 d 个集合中的路径出口,并结合 m 个上一时刻记录的最不拥塞出口,在 $d+m$ 个出口中选择队列最小的出口进行包的转发.DRILL 在交换机本地监测并采集出口队列信息,通过选择本地监测的对称路径集合的

最优端口实现负载均衡,保证非常少量的乱序包.在非对称场景中,对称路径集合能够保证包传输的路径具有近乎相等的时延.为了完全消除乱序的影响,还可在终端部署已有的处理乱序的网络栈来提高乱序恢复能力.在数据中心网络对称场景中,DRILL 相比于 ECMP、CONGA 和 Presto 能实现更小的流完成时间,在不同网络负载情况下其网络的平均排队时间也最小.在网络扩展和不同网络拓扑场景下,DRILL 仍然具有最优的性能.在非对称网络中,只有非常少量的流会触发 TCP 重传,因此 DRILL 具有非常有效的负载均衡能力.DRILL 基于包进行负载均衡相较于其他方案更接近最优性能,然而网络状态的感知速度和对称路径的计算是 DRILL 性能保证的关键.同时,由于网络存在动态性,故障恢复后对称路径的更新机制需要进一步考虑.

2.2.2 Flowlet 粒度

数据中心网络流量具有一定的特性,通过分析 TCP 传输层协议的流量发送特点,发现 TCP 的突发流量会造成 flowlet 的出现.Flowlet 的定义是一条流突发的包集合[16],当相邻包的发送间隔超过一定阈值时表示新 flowlet 的出现.Flowlet 是 TCP 拥塞控制造成的现象,当网络发生拥塞时,发送端会减小发送窗口,导致数据包发送的间隔增大,从而出现 flowlet[17].Flowlet 是网络出现拥塞后所造成的现象,因此可根据 flowlet 的出现情况对网络拥塞情况进行判断,从而调整转发策略实现网络负载均衡.由于不同 flowlet 之间的间隔大部分情况下是基于数据中心网络的传输时延而确定的,因此,基于 flowlet 的调度能够有效地避免包乱序问题的出现.目前典型的网络层基于 flowlet 粒度的数据中心网络负载均衡方案包括 CONGA[18]、CLOVE[19]、HULA[20]和 LetFlow[21]等.

(1) CONGA

CONGA 是一种拥塞感知的分布式负载均衡方案,其大部分功能在数据中心网络边缘交换机实现.在典型的叶脊拓扑中,CONGA 在叶子交换机中维护 Congestion-To-Leaf 和 Congestion-From-Leaf 两张表来实现路径的选择和网络状态的存储,同时在本地维护 flowlet 表并根据网络情况设定的阈值实现 flowlet 的检测.当流的包到达时,叶子交换机查找本地 flowlet 表并根据表项中的出端口进行转发;否则,交换机在所有可行的出端口和 Congestion-To-Leaf 表中选择最不拥塞的出端口进行转发.CONGA 利用不同源和目的地对之间传输的包实现路径拥塞情况的检测,利用基于 VXLAN[22]的 overlay 技术实现拥塞信息的传递.封装的信息包括 LBTag(4 bits)、CE(3 bits)、FB_LBTag(4 bits)和 FB_Metric(3 bits).当包在源交换机发出过程中时,会将源交换机的出端口号记录在 LBTag 中,并在 CE 中记录该端口对应的拥塞程度;在网内转发过程中,转发交换机在 CE 中记录较大的拥塞程度.当包到达目的交换机后,目的叶子交换机根据接收到的路径拥塞信息更新 Congestion-From-Leaf,然后将拥塞信息写入 FB_LBTag 和 FB_Metric 域实现反馈.相比于 ECMP 和 MPTCP 方案,CONGA 在网络负载不同的情况下都能保证最小的流完成时间,性能提升对大流尤为明显,对小流可实现与 ECMP 相似的效果.在网络故障情况下,CONGA 能够很好地处理网络故障后的负载均衡决策,实现最优的性能.虽然 CONGA 具有非常突出的性能,但其在叶子交换机存储网络路径拥塞信息带来的开销较大,因此 CONGA 只能应用在 2 层网络,使其扩展性受到一定限制.而且,CONGA 的全局拥塞感知会造成拥塞感知不及时,从而影响其负载均衡效果.

(2) CLOVE

现有的数据中心网络负载均衡机制需要更新网内交换机或是改变虚拟机网络栈,会带来很大的成本开销和更新周期.CLOVE 从网络边缘角度出发,在虚拟机控制程序上实现位于虚拟边界的拥塞感知的负载均衡机制.在 CLOVE 中,数据中心网络仍然采用现有的 ECMP 转发技术,负载均衡中的路径发现和选择以及拥塞感知都在虚拟机监控程序上实现.虚拟机监控程序发送具有不同的源端口号和生存时间的探针以发现源和目的地之间的路径.利用 ECMP 五元组哈希的不同结果,将源端口号和不同路径进行映射.在虚拟机监控程序中实现基于 flowlet 的调度,每个 flowlet 的转发路径根据拥塞情况进行权重路由,拥塞感知可利用 ECN 或 INT[23]技术实现.在叶脊网络拓扑中将 CLOVE 与 ECMP 和 CONGA 进行对比,发现 CLOVE 相比于 ECMP 能够实现更短的流完成时间,并且能够达到接近 CONGA 的性能,在高负载情况下,CLOVE 能够达到约 80%的 CONGA 的性能增益.CLOVE 不需要昂贵的硬件替换就能实现非常好的性能,成本较低.但是路径探针需要在 overlay 网络中实现,基于反馈感知拥塞无法及时响应,而且利用虚拟机监控程序维护路径状态也是不可忽略的开销.

(3) HULA

由于 CONGA 在边缘交换机保存大量拥塞状态,从而限制了其可扩展性,并且 CONGA 在硬件中的实现导致其存在不小的部署开销和难度,因此提出了 HULA,一种分布式的、单跳的负载感知的架构以实现数据中心网络负载的均衡.HULA 中,每个交换机维护路径利用率表,表中存有 Dst_ip、Best hop 和 Path util,分别表示对应的 ToR 的 IP 地址、发往该 ToR 的最优下一跳出口以及该出口到 ToR 路径的最大链路利用率.类似传统距离向量协议,路径拥塞状态在 ToR 之间进行逐跳传播,交换机只知道到达目的地的最优路径下一跳出口.路径状态感知过程中,ToR 交换机周期性地发送设计的探针,其中包含 torID(24 bits,表示发出探针的 tor)和 minUtil(8 bits,表示最优路径的最小利用率),当探针到达交换机后,会将其对应入口的出口利用率和原 minUtil 中的较大值保存在 minUtil 中,更新后交换机复制该探针并向其邻居发送.所有交换机执行相同策略,从而实现拥塞信息的传播.HULA 以 flowlet 粒度进行负载均衡,交换机本地维护 flowlet 表,新的 flowlet 到达后根据路径利用率表选择最优下一跳进行该 flowlet 包的发送.为了降低部署难度,HULA 采用 P4[24]来实现其包头解析、状态维护和探针处理等功能.在 3 层 Fat-Tree 拓扑的实验中发现,在对称情况下,HULA 在不同数据集中相比于 ECMP 和扩展的 CONGA 都具有更小的流完成时间.在非对称情况下,HULA 仍然具有最小的排队长度以及最短的流完成时间.虽然 HULA 具有很好的可扩展性,但其逐跳传播的机制限制了其拥塞感知的及时性,同时,在网络拓扑很大时探针的广播开销不可忽略,降低了网络的有效传输能力.

(4) LetFlow

LetFlow 充分发掘 flowlet 对拥塞自适应的特性,提出一种非常简单的交换机本地决策的 flowlet 负载均衡方案.通过对数据中心网络流量的观察,发现 flowlet 的大小能够直接反映网络的拥塞情况,拥塞严重的路径其 flowlet 更小,不拥塞的路径其 flowlet 更大.因此利用 flowlet 自身特性能够实现自适应的决策.在交换机本地维护 flowlet 表并进行 flowlet 检测,然后为每个新的 flowlet 在所有可行出口中随机选择一个出口进行转发.通过这种简单的机制,能够保证拥塞路径上的 flowlet 数据逐渐转移到更不拥塞的路径上,从而实现负载均衡.在不同的数据集中进行比较,LetFlow 相比于 ECMP 和 CONGA,能够实现更小的流完成时间.在网络非对称情况下,LetFlow 也能达到最优的性能.同时,LetFlow 对不同的传输层协议具有很好的鲁棒性.LetFlow 机制非常简单,其在交换机中造成的硬件开销相比于已有机制小很多,同时能够实现非常好的性能.但是 flowlet 的拥塞后感知的缺点使其无法对网络的突发拥塞进行感知,需要等到网络拥塞后造成新的 flowlet 出现时才能进行处理,因此对于需要处理突发流的情况难以满足性能要求.

2.2.3 Flowcell 粒度

已有的 flowlet 机制能够很好地感知网络的拥塞情况,但存在几个缺点.一方面,flowlet 大小不固定,不拥塞的路径上一个 flowlet 的数据量很大,拥塞路径上一个 flowlet 的数据量很小,网络中存在不同大小的 flowlet,使得对 flowlet 的调度存在难度.另一方面,flowlet 调度是网络出现拥塞后进行的调度机制,当交换机感知到 flowlet 后网络已经产生了拥塞或丢包,这种被动机制一定程度上会降低网络的传输性能.为了避免 flowlet 的缺点,He 提出基于大小相等的 flowcell 进行负载均衡的方案[25].Flowcell 粒度介于包和流之间,避免了基于流调度粒度过大的缺点,同时,一定程度上缓解了基于包粒度调度的乱序问题.目前,网络层基于 flowcell 粒度的数据中心网络负载均衡方案是 Presto[25].

(1) Presto

Presto 是一种在边界虚拟机监控程序执行基于 flowcell 粒度的主动负载均衡方案.该方案源于 ECMP 对小流可实现接近最优性能的思考,在端系统将流量分成大小一致的 flowcell 进行多路传输.在边界虚拟机监控程序中,将虚拟机发送的流量分割成与网络场景相关的、固定大小的 flowcell,考虑到现有虚拟机的 TCP 协议中 TSO(TCP segment offload)大小为 64KB,因此选择 flowcell 的大小为 64KB 来减少小流被分割的可能性.在实现多路传输的过程中,将网络中各路径进行标签化,并在虚拟机监控程序采用 shadow MAC[26]技术将包的目的 MAC 地址修改为转发路径的标签,以实现相应的路径发送.将包的源 MAC 地址修改为 flowcell 的 ID 来为终端提供数据还原的标识.在每个 flowcell 路径选择过程中,采用轮询或是权重调度方式实现不同路径上 flowcell 的

负载均衡.并设计一个中央控制器实现网络故障情况的感知,修改端系统的路径选择策略,直接删除故障路径.为了缓解包乱序问题造成的性能下降,Presto 还在 TCP 的 GRO(generic receive offload)中修改缓存机制以处理乱序包,有效地对少量的乱序包进行处理,减少了乱序包的重传.通过对 Presto 机制的测试,发现 Presto 基本上能够消除包乱序带来的影响,同时只会引起约 6%的 CPU 开销.在不同数据集和网络场景中,与 ECMP 和 MPTCP 相比,Presto 更接近最优性能.当网络出现故障导致不对称情况时,Presto 仍可以实现较好的性能.Presto 是一种主动负载均衡的机制,通过小粒度的固定大小的 flowcell 可以实现非常好的效果.但是由于 Presto 对网络拥塞不感知,因此其无法及时地对网络情况进行响应,在网络突发流严重的场景中,网络拥塞会大大降低其传输效果.

2.2.4 流粒度

基于流粒度进行调度是大多数负载均衡方案的策略.这类方案主动或被动地对网络每条流执行路径选择.通过集中式或分布式机制感知网络当前负载情况,对网络中传输的流进行路径选择.当网络状态发生变化后,通过预定策略选择部分流进行整体迁移,从而实现网络的负载均衡.目前,典型的网络层基于流粒度的数据中心网络负载均衡方案除了 ECMP 和 WCMP 之外,还包括:Hedera[27]、MicroTE[28]、Mahout[29]、Devoflow[30]、LocalFlow[31]、FlowBender[32]、Freeway[33]和 Expeditus[34]等.

(1) Hedera

Hedera 是一种集中式的、可扩展的动态流调度系统,其利用一个中央控制器从边界交换机中收集网络的流量信息,并分析网络中存在的大流以实现大流的调度,从而实现网络的负载均衡.Hedera 利用估计算法实现网络大流的检测,根据从边界交换机获取的流量信息迭代地计算各边界交换机对间所有流的稳态带宽需求,并将超过链路带宽一定阈值的流视为大流,在调度过程中为计算出的大流重新选择路径.Hedera 设计了两种大流路径选择算法:Global First Fit 和模拟退火算法.Global First Fit 算法中,调度器维护链路的剩余带宽,并为每个大流选择第 1 个可行的路径进行迁移.模拟退火算法则对所有可行路径空间进行搜索,尽量选择最优的路径实现调度.Hedera 提出的集中式控制机制对大流的负载均衡非常有效,在不同的流量模式场景下,Hedera 相比于 ECMP,具有更大的平均对半带宽,可接近最优性能.由于 Hedera 是一种集中式的控制机制,因此其需要收集的网络状态很多,在现有信息获取机制下,会造成非常大的传输开销.Hedera 中采用 OpenFlow 协议进行数据收集和路径安装,受限于现有设备的性能,调度器只能实现 5s 周期的调度,使其无法处理流量变化迅速的网络场景.

(2) MicroTE

由于数据中心网络流量具有短期的部分可预测性,因此可利用该特性优化流量的部署.MicroTE 是一种细粒度的流量管理机制,其通过分析 ToR 交换机对之间流量的可预测性,对可预测的流量进行细粒度调度.在服务器实现本地流量信息采集,并每隔一定时间(如:0.1s)将本地流量发送到指定的服务器进行流量信息的聚合以及流量可预测性分析,然后指定服务器再周期地向中央控制器通告聚合后的 ToR 对之间的流量信息.中央控制器生成全局流量信息,并利用装箱启发算法对可预测的流量进行最优路径选择.对不可预测流量则根据可预测流量路径选择后的剩余网络带宽能力,执行加权的 ECMP 调度策略.在不同的网络拓扑中,MicroTE 相比于 ECMP 具有更高的带宽利用率,且十分接近最优的带宽利用.在不同的流量可预测程度下,MicroTE 都能接近最优性能,且能尽量避免 ECMP 造成的丢包情况的发生.同时,在网络可预测性不高的情况下,MicroTE 与 ECMP 能够实现无缝衔接,因此其具有很好的可扩展性.然而,网络状态采集仍然是控制器决策的瓶颈,秒级粒度的调度无法处理突发流的情况,而且利用服务器进行信息采集不可避免地会带来巨大的开销.

(3) Mahout

为实现网内不同流量类型的差异化调度,Mahout 对网络中的大流和小流采取不同的调度策略来实现数据中心网络的负载均衡.Mahout 在终端检测 socket 的缓存大小,将缓存超过一定阈值的流视为大流,周期性地(如:1s)将大流的包进行标记,通过这种带内信号的方式实现大流通告.当边界交换机接收到包时,若没有标记,则该包对应的流为小流,交换机对小流执行 ECMP 策略进行转发;若该包属于大流,则交换机会将该包复制并发送给中央控制器,由中央控制器进行大流管理.控制器利用 Increasing First Fit[35]算法进行大流路径选择,并周期性地从网络中拉取链路利用率和大流统计信息,实现大流路径的优化.在不同的大流阈值场景下,Mahout 的大流

带内通告机制相比于端口采样和状态拉取方案具有更小的控制信息开销,在 OpenFlow 场景中只消耗少量的流表资源.与 Hedera 相比,Mahout 对不同大小的流都能实现更快速的大流检测过程.然而,Mahout 对大流的调度仍是基于周期性的网络状态拉取,因此调度结果存在滞后性.同时,其在终端进行大流检测虽然极大地提高了大流检测的速率,但数据中心中虚拟化技术的广泛应用,使得各虚拟机 socket 的访问成为部署挑战.

(4) Devoflow

Devoflow 是一种集中式和分布式相结合的负载均衡方案,其将大部分流的控制转移到交换机中以减小控制器的开销以及网络的响应时间.Devoflow 尽量在交换机本地实现网络中小流的路径选择,利用 OpenFlow 协议的通配符特性,设计本地路由克隆机制.当新流到达后,数据包会匹配通配符直到该流表匹配超过一定阈值后,交换机会本地克隆该流表来为命中的流生成新流表,并根据概率分布为新流表在多个可行端口中选择转发出口.同时,交换机本地对流表阈值进行检测,当其匹配命中数据大于一定阈值后,将该流视为大流并通报给控制器,控制器为该流选择最不拥塞的路径进行重路由.在不同拓扑和数据集中,Devoflow 相较于 ECMP 能够实现更大的吞吐量,提升的效果与不同的数据集类型相关.相较于基于拉状态的机制,Devoflow 与控制器之间的报文交换数据量很小,同时其在交换机中需要的流表数量也很少,因此 Devoflow 的可扩展性非常好.Devoflow 尽量将流的处理在网络数据平面中完成,因此能够更快地对网络进行本地响应.但是由于集中式机制固有的缺点,使其对网络变化不敏感,交换机不具有全局状态,使得提升出口选择效果成为难点.

(5) LocalFlow

LocalFlow 是一种交换机本地的负载均衡方案,其利用装箱算法将流分配到各个端口实现传输.网络中每个交换机都周期性地(如:10ms)执行调度过程,测量本地流量的速率,根据速率和出端口数量执行装箱算法,从而实现流的放置.算法将聚合流量均匀地分配在各出端口,并对无法放置的流量进行分流,当剩余能力填满后将剩下的流量继续分配,同时利用松弛条件减少分割流的数量,尽量保证小流不被分割.当网络发生故障造成不对称时,交换机通过底层协议进行感知,并在调度过程中简单地修改箱子数量大小来实现流量重平衡.实际部署中,LocalFlow 设计多精度的分割,采用扩展的匹配头或 OpenFlow 中的流表计数器实现精确的流分割,并与端系统 TCP 协议相配合,降低包乱序所带来的影响.在不同的拓扑中,LocalFlow 可以实现近优的吞吐量,相比于 ECMP、MPTCP 和 CONGA,具有明显的吞吐量提升.相比于简单的包散射机制,LocalFlow 具有更少的流分割,从而实现更小的流完成时间.适当增加 TCP 的重复 ACK 阈值能够有效地避免受到包乱序的影响.同时,低的流分割的精度会降低 LocalFlow 的性能.LocalFlow 是一种主动的拥塞避免机制,利用灵活的负载感知的流分割来实现网络负载均衡.但是由于其无法感知网络的拥塞信息,因此对拥塞不敏感.同时,其通过底层协议进行故障检测,使不同交换机对不对称情况感知存在时延差异,可能造成短时间的流量黑洞,从而导致性能下降.

(6) FlowBender

Flowlender 是一种在端系统本地实现的负载均衡机制.现有数据中心采用 ECMP 实现流的哈希转发,在不改变交换机硬件的条件下,FlowBender 通过在交换机执行扩展的哈希来实现流的路径选择.将原有的哈希函数从五元组扩展为五元组加上 TTL(time-to-live)值,并在端系统利用已有的 ECN 标记和 TCP 超时时间来检测每条流的拥塞情况,根据策略修改拥塞流的 TTL 值以实现流的换路.为了减少路径切换和包乱序情况的出现,FlowBender 在每个 RTT 时间都对接收到的拥塞信息进行评价,并设计拥塞感知阈值以提高拥塞感知的准确性.在 3 层 Fat-Tree 拓扑的仿真中,FlowBlender 相比于 ECMP 能够实现更小的流完成时间.在不同负载情况下,FlowBender 相比于 DeTail 具有更小的流延迟,接近 RPS 的性能.在测试床实验中,FlowBender 在不同的网络负载情况下,相比 ECMP 能够有效地减小流的时延,随着负载的增加,优化效果更明显.FlowBender 是一种简单、有效的负载均衡机制,部署成本低,效果好.但其扩展的哈希不能保证改变 TTL 值一定就能改变转发路径,因而会延长拥塞响应的周期,造成拥塞处理不及时.而且,基于 ECN 和 TCP 超时来感知网内拥塞无法处理流量动态性强的场景.

(7) Freeway

数据中心网络存在不同的流量需求,吞吐敏感和时延敏感的流量对于网络路径的时延要求不同.Freeway

是一种将大流和小流隔离后进行传输的方案.其利用数据中心网络拓扑的等价路径特性,将网络中的 ToR 对之间的路径根据路径的利用率和排队情况分成低时延路径和高吞吐路径.利用 SDN 网络中的控制器收集网络中各链路的利用率信息,将网络中的 ToR 路径根据核心交换机的位置分为固定数量的低时延路径和高吞吐路径,剩余路径作为动态路径并根据网络流量情况在两类路径中进行切换.交换机接收路径类型信息并为小流作本地决策,通过最不拥塞路径进行转发.端系统在发送大流前和控制器交互大流信息,控制器根据优化算法为大流选择合适的路径并加以部署,然后通知端系统发送相应数据.当网络中路径类型发生变化时,执行对应的操作实现调度.在 Fat-Tree 拓扑中进行仿真,实验中低时延路径和高吞吐路径的数量变化反映出 Freeway 对网络状态感知准确.在真实数据集场景下,Freeway 相比于 ECMP、Hedera 和包散射方案能够实现更大的大流吞吐量和更小的小流时延,同时能够保证更少的 TCP 超时事件.Freeway 实现了大流和小流路径的隔离,但如何在网络流量变化剧烈的情况下实现快速的路径分类需要进一步加以考虑,将不同类型流量分开传输还可能导致出现链路利用率不高的问题.

(8) Expeditus

Expeditus 充分利用 Fat-Tree 拓扑中聚合层交换机之间存在成对连接的特性,提出了一种二阶段的分布式拥塞感知的负载均衡机制.数据中心网络中交换机对其上行链路的入口和出口拥塞情况进行本地测量.第 1 阶段中,当一个流从源 ToR 交换机发送到目的 ToR 交换机时,该流的第 1 个包到达源 ToR 后,会将本地所有上行链路的出口拥塞信息携带在以太网包头中,通过 ECMP 机制将包发送到目的 ToR 交换机.目的 ToR 交换机提取包头中的拥塞信息并与本地所有上行链路入口拥塞信息进行比较,选择最不拥塞的目的聚合层交换机进行数据返回.第 2 阶段中,当被选择的目的聚合层交换机接收到返回的包后,其将本地所有上行链路入口拥塞信息携带在返回包中继续通过 ECMP 进行发送.源聚合层交换机接收到返回包后,将包中拥塞信息与本地所有上行链路出口拥塞信息进行综合,选择最不拥塞的核心层交换机安装转发规则.当第 1 个返回包返回源 ToR 交换机后,该流的路径建立成功,之后的包根据网内交换机安装好的转发规则进行传输.在不同的数据集实验中,Expeditus 相比于 ECMP 具有更短的流完成时间.在不同的流量模式场景下,Expeditus 相比于 ECMP 能够实现更高的吞吐量,性能接近最优的调度机制.当网络存在不对称情况时,Expeditus 仍可以实现网络的负载均衡,性能好于 ECMP.在 2 层叶脊网络中,Expeditus 相比于 ECMP 和 CONGA 能够实现更短的流完成时间.Expeditus 利用二阶段的 ToR 和聚合层交换机的选择来接近最优的路径,在一个 RTT 时间内实现最不拥塞的路径部署.然而,当网络动态性高且路径拥塞情况不稳定时,对各路径的拥塞情况感知不及时会降低路径选择的效果.

(9) Scheduling Algorithm

由于现有数据中心网络中经常出现非对称情况,传统 ECMP 机制无法根据非对称状态调整流量分配策略,因此,基于网络状态的调度算法是解决这类问题的一种方法.Mehrnoosh 等人提出一种在线不分流的低复杂度且具有拥塞感知的调度算法[36].该算法计算不同路径的开销并将流分配到最小边际开销的路径上,实现最小化的平均网络开销,同时证明了算法的渐进性能.实验结果表明,该算法在各种流量环境和数据中心网络结构场景中都具有更好的性能,而且在对称情况中,算法的性能也非常好.考虑到当前数据中心网络中的多流传输现状,Mehrnoosh 等人还提出一种性能更好的、整体完成时间更短的多流调度算法[37].利用基于排序变量的线性规划模型和简单的序列调度策略来设计启发式调度算法,在具有多流开始时间和没有开始时间的情况下,证明了该算法能够显著提升最优估计率.根据实际的仿真结果来看,提出的多流调度模型能够获得更短的平均多流完成时间,实现多流传输之间的负载均衡.由于现有的调度算法通常需要建立全局最优模型,因此网络状态采集会产生较大的开销,同时,集中式的全局管理也限制了这类调度算法对网络状态响应的及时性.

2.2.5 网络层方案对比

网络层方案设计针对性的网络流量调度机制或交换机转发策略,通过不同粒度的传输管理来实现数据中心网络负载均衡目标.通过分析现有网络层方案,从方案的关键设计和方案性能两个角度对各方案进行对比,比较结果见表 1.

方案关键设计包括方案特点、是否对包进行标记、是否区分网络大小流进行差异化处理、如何获取网络

拥塞信息、是否考虑网络非对称情况、是否设计机制克服突发流情况以及是否会造成传输失序等方面.从比较结果来看,网内负载均衡方案为了获取网络状态通常会采用包标记来携带相应信息;不进行包标记的方案通常为集中式状态采集方案或是主动负载均衡方案,这类方案对网络状态感知较慢或不感知.集中式调度方案通常会显式地进行大流调度,而基于 flowlet 调度的方案利用 flowlet 的特性隐式地实现了大小流的区分.对于分布式方案来说,大多采用本地信息采集或网络反馈来获取网络状态;而集中式的方案通常会对全网状态进行收集,一定程度上会增加状态采集难度.现有网络层负载均衡方案大多通过全局网络信息考虑了网络非对称情况,但大部分方案没有考虑突发流的处理,只有依靠交换机本地信息或网络拥塞反馈的方案能够在一定程度上响应突发流情况.对于最关键的失序问题,基于流粒度的确定的不分流的调度不会造成传输失序,但对于动态调度的方案则有可能导致失序的出现,尤其是包粒度的方案出现失序的可能性非常大.虽然基于 flowlet 的方案能够很大程度地避免失序情况的发生,但大多数方案仍然需要修改传输层以降低失序带来的影响.

方案性能方面,从方案的最优性、复杂度和方案执行速度这 3 个角度进行比较.集中式的方案通常具有网络的全局信息,因此能够实现全局最优的调度策略;分布式的方案一般都会根据本地采集状态进行决策,但部分分布式方案会通过网络状态反馈等机制获取全局信息,也能实现全局最优的决策.对于固定策略的负载均衡方案,通常复杂度较低,需要对网络多个部分进行修改的方案具有较高的复杂度.一般具有较低复杂度的方案能够实现更快的方案执行速度,复杂度高的方案通常执行速度较慢.但一些集中式方案通常会部署相匹配的分布式策略,通过综合的管理方式能够弥补集中式方案的低速缺点.

Table 1 The comparison of load balancing schemes based on the network layer in data center networks

表 1 数据中心网络负载均衡网络层方案对比

方案	关键设计							方案性能		
	特点	包标记	大小流	拥塞信息	非对称	突发流	失序	最优性	复杂度	速度
DRB	位逆转选择路径	无	无	无	不考虑	不考虑	无	无	低	快
RPS	包散射	无	无	无	不考虑	不考虑	严重	无	低	快
DRILL	交换机选择最不拥塞出口发送包	无	无	交换机队列	考虑	考虑	可能	本地	一般	快
CONGA	边界交换机维护 flowlet 并选路	有	隐含	包携带	考虑	不考虑	可能	全局	高	一般
HULA	交换机维护最优路径下一跳	有	隐含	探测包	考虑	不考虑	可能	全局	高	一般
CLOVE	端系统为 flowlet 选路	无	隐含	ECN 标记	考虑	考虑	无	全局	一般	一般
LetFlow	交换机本地为 flowlet 选路	无	隐含	Flowlet 间隔	考虑	不考虑	可能	本地	一般	一般
Presto	端系统为 flowcell 选路	有	隐含	无	考虑	不考虑	可能	无	一般	快
ECMP	交换机本地哈希	无	无	无	不考虑	不考虑	无	无	低	快
WCMP	交换机本地执行权重哈希	无	无	无	考虑	不考虑	无	无	低	快
Hedera	中央控制器周期性调度大流	无	显示	控制器收集	考虑	不考虑	可能	全局	一般	慢
MicroTE	收集流信息并进行集中调度	无	显示	控制器收集	考虑	不考虑	无	全局	一般	一般
Mahout	端侧检测 socket 并集中调度	无	显示	控制器收集	考虑	不考虑	无	全局	一般	一般
Devoflow	交换机克隆规则,控制器综合调度	无	显示	控制器收集	考虑	不考虑	无	全局	高	一般
LocalFlow	交换机周期性利用装箱算法分配流量	无	无	交换机本地	不考虑	不考虑	可能	本地	低	慢
FlowBlender	五元组和 TTL 值进行哈希选路	无	无	ECN 和包超时	不考虑	考虑	无	无	一般	快
Freeway	路径分类执行全局和分布式选路	无	显示	链路利用率和排队情况	考虑	不考虑	无	全局	高	慢
Expeditus	二阶段选路策略	有	无	包携带	考虑	不考虑	无	全局	高	一般
Scheduling Algorithm	构建全局最优调度模型进行调度	无	无	控制器收集	考虑	不考虑	无	全局	高	慢

2.3 传输层方案

数据中心网络由于其拓扑特性存在多条等价路径,因此将流量分发到多条路径是实现负载均衡目标的一种方法.传输层方案将网络中的流分成多条子流,并通过子流的综合管理来调整各流的发送策略,从而提高网络的吞吐量并实现较小流的完成时间.目前,典型的基于传输层的数据中心网络负载均衡方案包括 MPTCP[38]、XMP[39]、RackCC[40]、MMPTCP[41]和 VMS[42]等,其分类结果如图 3 所示.

(1) MPTCP

为了利用数据中心网络的多路特性提高网络传输能力,如何将流分配到多个路径传输并加以管理是关键.MPTCP 是一种将流分成多个子流,并利用多个 TCP 连接进行子流传输的数据中心流量传输方案.多个 TCP 连接综合进行流量管理,通过修改端口或 IP 地址并结合数据中心网络常用的 ECMP 路由方式,实现子流的多路传输,并将流量向更不拥塞的等价路径进行转移,从而实现数据中心网络的负载均衡.各个子流 TCP 的拥塞窗口变化遵循如下规则:当子流 r 接收到一个 ACK 时,该子流的窗口 w_r 增加 $\min(a/w_{\text{total}},1/w_r)$;当子流 r 出现一个丢包时,该子流的窗口减少 $w_r/2$,其中,w_{total} 为所有子流的窗口之和,a 决定了子流窗口的增加速度.将子流窗口与整个流的传输带宽相结合,可以保证流传输的公平性.同时,通过 a 的调节,可以提高 MPTCP 的鲁棒性.在不同的拓扑和流量模式的仿真中,MPTCP 在给定足够的子流数量的情况下,能够实现最大的网络吞吐量.对于常见的数据中心拓扑,8 条子流就能实现很好的吞吐量和公平性.在不同的拥塞控制机制中,MPTCP 相比于独立 TCP 流和相等权重 TCP 流,具有更低的丢包率,并能实现更小的小流完成时间.MPTCP 能够充分利用数据中心网络的多路特性实现负载均衡,但是由于需要将流分成多个子流进行传输,因此需要传输层协议进行相应的修改,而且,当网络出现多个瓶颈链路时,MPTCP 吞吐量会被限制,从而降低了传输能力.

Fig.3 The classification results of the schemes base on the transport layer

图 3 传输层方案分类结果

(2) XMP

传统 MPTCP 方案利用多路特性实现高网络利用率,而实际数据中心网络中存在不同的流量需求.理想情况下,大流采用 MPTCP 而小流采用 TCP 进行传输.XMP 结合网内的链路缓存占用率,提出了一种平衡吞吐量和带宽的拥塞控制机制.XMP 提出 BOS 算法,结合 ECN 机制和网络的带宽时延积以修改子流的窗口,通过调整子流发送速率从而控制链路的缓存占用率.同时,XMP 还提出 TraSh 算法,根据效用函数和平衡情况,调整各子流窗口以实现流量从拥塞路径转移到不拥塞的路径.将 BOS 和 TraSh 算法相结合从而实现大流的高吞吐和小流的低时延.在测试床实验中,XMP 对网络动态性具有快速的响应能力,能够将流迅速地根据网络拥塞情况进行迁移.当拥塞控制参数设置合适时,XMP 能够实现很好的公平性.在不同流量模式的仿真实验中,XMP 相比于 DCTCP 能够实现更大的吞吐量,且能够有效地处理多对一传输时吞吐崩溃的问题.同时,在不同的 RTT 分布中,XMP 也能实现最小的传输时间.XMP 提出一种结合网内状态的多路传输的拥塞控制机制,但其利用 ECN 进行拥塞感知的方式使其对突发流量不敏感,而且拥塞控制的关键参数选择与网络流量分布情况和网络结构相关,对不同场景需通过大量实验才能获得最优参数.

(3) RackCC

在数据中心网络中,每条路径上都传输许多流,因此,相同路径被多条流探测过传输状态,这种探测是在 rack 间的层次而不仅仅是端系统间的层次.为了消除重复探测造成的性能损失,RackCC 实现了一种 rack 层次的拥塞控制机制,将 rack 之间传输的流聚合成巨大流以执行 rack 间的传输层控制,实现更好的公平性和传输性能.RackCC 实现了快速的流启动、相同 rack 对之间流的公平分享以及多流协作的拥塞检测.其在 ToR 交换机上维护不同路径的 ECN 到达速率、子流数量以及吞吐量信息,并周期性地通告连接到该 ToR 的端系统.RackCC 的拥塞控制主要分为巨大流的拥塞控制和子流的拥塞控制,根据接收端和发送端的约束,子流发送速率变化为 $s_{target}=\min(f_{target},D_{target})$,其中,$f_{target}$ 表示子流窗口根据 ToR 交换机收集的各路径信息所进行的速率调节,D_{target} 表示目的接收端根据其接入链路能力所返回的可行速率.当新流到达时,巨大流会根据新流到达的数量直接给新

流分配公平的发送速率以实现新流的快速启动;当流结束后,将剩余的资源平均分配给发送中的流.与传统 TCP 进行对比,RackCC 可在更少的 RTT 时间内获得公平分享速率.在与 DCTCP 的比较过程中,RackCC 在不同的子流数量和流大小情况下可缩短尾部流完成的时间.RackCC 通过 rack 间巨大流的拥塞控制,实现了不同路径的负载均衡,然而,这种方案需要 ToR 交换机维护状态,并与端系统进行交互,其部署难度较大.

(4) MMPTCP

传统 MPTCP 方案对数据中心网络中所有流进行多路传输,然而,对于时延敏感的小流来说,多路传输会增大其控制复杂性和拥塞的概率,从而延长了小流的流完成时间.MMPTCP 提出一种二阶段的传输协议.在第 1 阶段,MMPTCP 通过一个拥塞窗口将流的包随机地发送到网络中,利用网络中的多路特性提高传输效率.当发送的包超过一定阈值时,MMPTCP 进入第 2 阶段,将继续发送的包通过正常的 MPTCP 进行传输.二阶段的方式可以在保证小流的完成时间的同时保证大流的吞吐量.为了处理包乱序问题,MMPTCP 利用网络拓扑特性,在端系统推测流量传输的路径长度,从而自适应地调整该流对应的重复确认阈值,降低包乱序造成的影响.在子流数相同的情况下,MMPTCP 相比于 MPTCP,能够实现更小的短流完成时间,同时能够保证大流的吞吐量不变,还能维持较高的网络利用率和较低的丢包率.相比于 TCP 和包散射机制,MMPTCP 也具有更好的传输性能.MMPTCP 将包散射和 MPTCP 加以结合,实现了差异化的传输和负载均衡效果,但包散射造成的包乱序需要通过修改 TCP 重传机制来处理,关键阈值的设计与网络状态相关联,在动态性强的数据中心网络中是一个难以解决的挑战.

(5) VMS

现有的数据中心网络负载均衡机制需要修改网络设备或虚拟机协议栈,VMS 则通过修改端系统部署的虚拟交换机,在端系统传输过程中修改传输路径,实现新的流量负载均衡方案.VMS 在发送端部署 VMS 发送模块,通过修改包的 5 元组中的源端口号为每个流的每个包选择不同的转发路径进行包散射传输,并在接收端部署 VMS 接收模块对每个流的包进行还原并发送给相应虚拟机.在每个包的路径选择过程中,VMS 采用虚拟窗口机制,根据网内不同路径 ECN 的反馈情况选择具有足够传输能力的路径进行传输.VMS 在流传输过程中维护流的状态,并在连接断开时移除相应状态信息.在实验中对 VMS 的路径数进行评估,表明 8 条路径即可实现可接受的性能.同时,VMS 可实现非常接近 MPTCP 的性能,而且设计的方案使其不易受网络拓扑非对称的影响.通过原型实验,发现 VMS 与传统虚拟交换机方案的 CPU 开销相差不大.VMS 提出在端系统虚拟交换机实现负载均衡减小了原有机制的成本开销,降低了负载均衡方案部署的难度.然而,在虚拟交换机进行负载均衡时需要对相关的虚拟机流量综合处理,对于不同需求的流量无法获得差异化传输,同时,在网络高负载情况下,发送端和接收端模块的协议栈优化需要进一步加以考虑.

(6) 传输层方案对比

传输层方案利用多路传输优势,修改端侧协议栈以提高传输效率并实现数据中心网络负载均衡目标.通过分析现有传输层方案,从方案的关键设计和方案性能两个角度对各方案进行了对比,比较结果见表 2.

Table 2 The comparison of load balancing schemes based on the transport layer in data center networks

表 2 数据中心网络负载均衡传输层方案对比

方案	关键设计					方案性能		
	特点	控制类型	控制因素	大小流	多阶段	状态数	差异化	复杂度
MPTCP	多子流 TCP 传输	窗口	丢包	不考虑	无	少	不考虑	低
XMP	大流采用 MPTCP 传输,小流采用 TCP 传输	窗口	ECN、带宽时延积	考虑	无	一般	考虑	一般
RackCC	ToR 之间巨大流控制	速率	路径探测信息	不考虑	无	多	考虑	高
MMPTCP	二阶段 MPTCP 传输	窗口	发送量、拓扑特性	考虑	有	一般	考虑	一般
VMS	端侧包散射	窗口	ECN	不考虑	无	一般	不考虑	一般

方案关键设计从方案特点、拥塞控制类型、拥塞控制影响因素、大小流差异化处理和是否多阶段控制等几个方面进行比较.传输层方案都是通过多路管理来平衡网络中的数据传输性能,多路之间拥塞控制通常采用

传统 TCP 中的拥塞窗口进行控制;随着新型协议栈的出现,也使得基于发送速率的控制被应用到传输层方案中,相比于基于窗口的方案能够更快地达到均衡目标.大部分传输层方案采用常用的拥塞控制影响因素,包括丢包和 ECN 等进行发送状态调整;部分方案还考虑了网络结构特性或进行网络状态探测,这类方案对网络状态感知相对准确.早期传输层方案通常不考虑大小流对传输的影响,由于小流在多路传输情况下受影响较大,因此后续方案通常实现了大小流的单独处理,部分方案还设计了多阶段的处理方式,进一步提升了传输性能.

在方案性能方面,对传输层方案需要维护的状态数、是否实现了差异化管理以及方案复杂度等几个方面进行对比.大部分传输层方案维护的状态数不多,但部分方案的拥塞管理与网内状态紧密相关,因此会产生较多的状态维护需求.大部分方案考虑了网内的差异化管理,显式或隐式地对大流和小流进行差异化处理,实现大小流之间的公平.考虑不同传输层方案的管理开销和对网络的修改程度,大部分方案不具有很高的复杂度,但是对于网内多部分配合的方案,其高复杂度会降低方案的可部署性.

2.4 应用层方案

数据中心中部署大量服务器来执行不同任务,因此不同任务的流需要调度以实现相应的需求.从应用角度执行负载均衡决策是最能满足应用需求的方式.数据中心网络中服务器之间的负载均衡是最经典的负载均衡问题.数据中心中部署大量不同应用的服务器来提供不同类型的服务.当某个应用的流到达数据中心后,数据中心需要将该流路由到某个服务器进行处理,由于服务器负载情况的不同以及网络状态的变化,将流导入不同的服务器会造成不同的开销和性能,尤其是服务器的利用率会导致不同的流处理完成的时间,从而直接影响服务的完成质量.因此,实现服务器之间的负载均衡能够提高数据中心的服务能力,并保证数据中心的健壮性.目前,典型的数据中心服务器负载均衡方案包括 Ananta[43]、Duet[44]、Maglev[45]、SilkRoad[46]和 Beamer[47]等,其分类结果如图 4 所示.

Fig.4 The classification results of the schemes base on the application layer

图 4 应用层方案分类结果

(1) Ananta

Ananta 实现了多租户云平台中可扩展的软件负载均衡器和 NAT,其将网络分成 3 层:顶层是路由器,在网络层基于 ECMP 实现负载分发;第 2 层是 Mux,即一系列可扩展的专用服务器来实现负载均衡,其在内存中维护网络连接的状态并实现应用服务器的 4 层负载均衡;第 3 层是虚拟交换机,通过在每个服务器中部署虚拟交换机来提供有状态的 NAT 功能.Ananta 设计了可扩展的网内处理机制,使网内多个网络元素能够同时处理相同 VIP 的包,从而不需要同步地维护每个流的状态;并下放部分功能到端系统,让端系统只维护本地虚拟机的状态,从而提高 Ananta 在数据中心的可扩展性.当一条流到达云数据中心后,路由器会将流路由到 Mux,然后 Mux 为该流选择并记录 DIP 后利用 IP-in-IP 的封装机制将包发送到端系统,由端系统的 HA(host agent)进行解析.当流从数据中心内部向源返回数据时,端系统会与 AM(ananta manager)交互来选择路由以实现绕过 Mux 的快速路径转发,从而降低 Mux 的负载并实现更快速的数据传输.通过性能测试,发现 Ananta 的快速路径返回机制能够有效地减少 Mux 的 CPU 开销,同时更高的 Mux 的 CPU 利用率会延长其检测 SYN 洪泛攻击的时间.在实际公有云的部署中,绝大部分 SNAT 端口分配可以得到及时的响应,并且 Ananta 在长时间内可维持较高的可用性,从不同 Mux 的带宽和 CPU 利用率来看,ECMP 可以实现 Mux 之间均匀的负载均衡.Ananta 通过软件负载均衡机制实现了很好的扩展性,然而,软件负载均衡器受限于其处理能力,其性能是否匹配得上流量的增长,还需要进一步加以验证.

(2) Duet

为了克服软件负载均衡器固有的有限能力和高处理时延的缺点,Duet 提出一种将硬件负载均衡器

(Hardware Mux,HMux)和软件负载均衡器(Software Mux,SMux)相结合的可扩展的负载均衡策略.利用现有商业交换机中已实现的 ECMP 和隧道机制,在剩余的表项中维护一定数量的 VIP-DIP 映射,在每个 HMux 中维护部分 VIP 集合以及对应的所有 DIIP 表项,并在交换机之间利用 BGP 通告 VIPs 来保证其他交换机能够根据 VIP 路由到该交换机.同时,Duet 在网络中部署少量 SMuxes,每个 SMux 中维护所有 VIPs 信息以实现灵活性并保证高可用性.Duet 设计贪婪算法实现在不同存储能力和带宽约束情况下 VIPs 在交换机之间的分配,并利用 SMuxes 的部署来提高网络的故障容忍能力.测试结果表明,Duet 能够实现低时延并保证高可用性,同时能够有效地处理故障情况,在 VIP 迁移过程中可以快速地对网络情况进行响应.在不同的流量场景中,Duet 相较于 Ananta 只需要更少的 SMuxes 就能达到需要的性能,并能实现更低的处理时延.在故障情况下,Duet 能够实现平滑的过渡,从而保证最大链路利用率的稳定性.Duet 充分利用了现有交换机的剩余能力,因此部署成本低,具有较高的容量和较低的处理时延,可实现高可用性.由于利用分布式方式实现 VIPs 在 HMuxes 中的存储,因此不同场景中需要设计合适的分配算法以实现期望的负载均衡效果.

(3) Maglev

Maglev 是一个快速、可靠的分布式软件负载均衡系统,目前已部署在 Google 数据中心之中.Maglev 将所有入站的包通过 ECMP 机制转发到设计的 Maglev 机器上执行软件负载均衡决策,根据决策结果将包通过通用路由封装机制(generic routing encapsulation,简称 GRE)发送到目的服务器以进行服务,服务器返回的数据包绕过软件负载均衡器直接返回给源端.Maglev 的关键部分在于软件负载均衡器的优化,其绕过 Linux 内核,并设计 forwarder 模块,直接在网卡和 forwarder 之间交互数据和维护连接跟踪表,同时设计了新的一致性哈希算法实现后端资源的公平分享.从 Maglev 在 Google 数据中心部署的效果来看,Maglev 设计的 forwarder 相较于内核处理方式具有明显的处理优势.对于不同的包类型,Maglev 可以实现近乎相等的吞吐量,同时在一定带宽情况下设计的 forwarder 不会成为瓶颈.在与不同的哈希算法比较的过程中,Maglev 的哈希在不同哈希表大小的情况下均能实现近乎完美的负载均衡效果.Maglev 通过设计和优化软件负载均衡器实现了可扩展的、灵活的负载均衡系统,在实际运行中效果显著.然而,在测试过程中,在更高速的网络环境下(40Gbps),forwarder 中的 steering 模块会成为性能瓶颈,导致网卡与 steering 性能不匹配,因此,如何让 Maglev 适应更高速的网络场景需要进一步设计.

(4) SilkRoad

SilkRoad 是一种在交换芯片上实现的快速、有状态的 4 层负载均衡机制,其利用单个交换芯片代替软件负载均衡器,直接在交换机本地进行负载均衡.SilkRoad 在 P4 交换机中实现,通过存储流五元组的哈希值来减小匹配域大小的存储空间,并设计 DIP 池的版本号来减少匹配后的动作数据的存储开销.为了保证每个连接的一致性,在 DIP 池更新过程中,SilkRoad 利用布隆过滤器存储更新过程影响的流,并设计转移表来保证这些流处理的正确性.在 P4 交换机有限的存储空间中,SilkRoad 实现了片上的负载均衡.在流级别的仿真环境中,SilkRoad 在不同数据中心网络位置上能够实现相应的负载均衡效果,在相同的需求情况下相比于软件负载均衡器只需要一半甚至更少的设备.与 Duet 相比,SilkRoad 在不同的更新情况和新连接到达时能够实现更好的连接一致性效果,同时转移表能实现明显的性能提升.SilkRoad 通过设计合适的存储和匹配机制,在 P4 交换机的芯片中实现了有状态的负载均衡,然而,P4 交换机无法实现比较复杂的功能,且其性能与硬件负载均衡器存在一定差距,因此,SilkRoad 的性能需要进一步加以验证.

(5) Beamer

在服务器负载均衡过程中,维护每条流的状态会给负载均衡器带来巨大的开销以及其他问题,包括标准的扩展事件造成的连接断开、SYN 洪泛攻击以及在软件负载均衡器中维护状态会极大地降低吞吐量.Beamer 是一种无状态的可扩展的数据中心负载均衡器,充分利用后端服务器本地维护的连接状态信息来保证连接可靠性,能够支持 TCP 和 MPTCP 的状态维护.Beamer 为了实现连续的哈希空间,将服务器分成多个箱子,利用稳定哈希算法将每个连接分配到每个箱子中的服务器进行处理.在网络状态迁移和更新过程中,为了保证连接一致性,Beamer 利用 daisy chaining 技术将到达新服务器的旧连接发回给旧服务器.与维护状态的负载均衡机制相比,Beamer 对不同大小的包都具有更快速的处理能力,在网络故障的情况下能够实现更平滑的吞吐量过渡,并能均

匀地进行流量分配.在对 TCP 和 MPTCP 的流传输过程中,能够保证短的流完成时间.在服务器数量较多的情况下,Beamer 能够在短时间内进行新配置的部署.Beamer 设计无状态的负载均衡方案,减轻了负载均衡器的存储和处理开销,但当网络状态变化影响流的数量较多时,其 daisy chaining 机制会导致大量流量通过服务器进行转发,使得服务器开销增大,成为网络传输性能瓶颈.

(6) 应用层方案对比

应用层方案考虑数据中心网络中服务器之间的性能差异,保证服务能够均匀地分配到相应的服务器,同时维护连接的一致性,实现数据中心网络的负载均衡.通过分析现有应用层方案,从方案的关键设计和方案性能两个角度对各方案进行对比,比较结果见表 3.

Table 3 The comparison of load balancing schemes based on the application layer in data center networks

表 3 数据中心网络负载均衡应用层方案对比

方案	关键设计				方案性能			
	特点	架构	状态维护	硬件需求	瓶颈	速度	适应性	复杂度
Ananta	3 层负载均衡结构	软件	服务器	交换机+服务器	软件性能	慢	高	低
Duet	现有商业交换机和 软件交换机共同维护	软件+硬件	交换机	交换机+服务器	分布式存储	一般	高	一般
Maglev	绕过操作系统的软件负载均衡器	软件	服务器	服务器	运行模块	一般	一般	一般
SilkRoad	利用交换芯片代替 软件负载均衡器	硬件	交换机	P4 交换机	特殊芯片性能	快	低	高
Beamer	利用连续空间哈希和 daisy chaining 技术	软件	服务器	服务器	Daisy chaining 技术	一般	高	一般

方案关键设计包括方案特点、设计架构、状态维护方式和方案硬件需求几个方面.从比较结果来看,现有传输层负载均衡方案主要通过软件和硬件负载均衡器的部署实现负载均衡功能,为了提高负载均衡速度通常会采用相结合的部署方式,部分方案还对软件负载均衡器进行优化.硬件负载均衡器的优点在于具有更快的处理速度,软件负载均衡器的优点在于能够维护更多状态并执行更复杂的负载均衡策略.因此,结合软/硬件是现有应用层方案的趋势.大多数负载均衡状态都在软件负载均衡服务器进行维护,但部分利用新型交换芯片的方案会在交换机维护部分状态,这类方案会造成更多的硬件需求.

在方案性能方面,对应用层方案的性能瓶颈、方案速度、适应性和复杂度几个方面进行比较.应用层方案的性能主要受限于负载均衡器的性能,但是对于利用特殊存储结构或状态维护算法的方案,其设计的关键结构和算法也是负载均衡性能的瓶颈.硬件负载均衡方案通常相比于软件负载均衡方案具有更快的速度,但软件负载均衡方案由于其部署难度较低,因此具有更高的适应性.对于能够利用现有网络设备的方案其复杂度相对较低,但对于修改现有网络设备或需要专用设备的方案则具有更高的复杂度,其部署难度也更大.

2.5 综合方案

数据中心网络流量具有高动态性,其与传统网络特点不同,因此优化传统的负载均衡机制只能在一定程度上提升负载均衡效果.影响数据中心网络负载的因素很多,包括拓扑结构、流量特性和调度机制等,针对数据中心网络设计合适的负载均衡方案需要综合考虑这些因素,通过对数据中心网络进行多层设计来达到需要的性能.目前典型的、综合的数据中心网络负载均衡机制包括 DeTail[48]、Fastpass[49]、Hermes[50]、NDP[51]和 AuTO[52]等,其分类结果如图 5 所示.

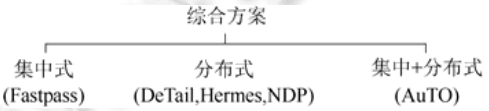

Fig.5 The classification results of the schemes base on the synthetic design

图 5 综合方案分类结果

(1) DeTail

DeTail 是一种跨层网络栈设计,其在底层网络快速检测拥塞,驱动上层网络执行路由决策以选择更不拥塞的路径,从而实现网内的负载均衡并保证流的时间约束.DeTail 在链路层采用 PFC 机制[53],利用 Pause 和 Unpause 信息控制交换机上下游的包发送来实现无损传输.DeTail 在交换机本地执行每个包的负载均衡,根据端口队列占用反映的拥塞信息动态地选择包的下一跳.由于无损结构网络不会因为拥塞造成包丢失,因此传输层采用 ECN 机制实现拥塞感知.同时根据不同流的时延敏感程度指定流优先级,网络传输过程中保证高优先级的包先处理.DeTail 能够完全避免拥塞相关的丢包,相比于流哈希机制能够极大地缩短流完成时间.同时,其能有效地让包绕过拥塞热点,相比于无损的包随机发送也能实现更小的流完成时间,对不同的流模式,包括序列流和分裂聚合流,都能提升其传输性能.DeTail 通过设计跨层结构实现每个包的负载均衡,然而,由于其利用 PFC 实现无损的 2 层传输,对数据中心网络设备有一定需求,而且在网络拓扑较大的情况下,PFC 的消息通告方式会造成不同位置流的相互影响,从而降低 2 层数据的传输效果.

(2) Fastpass

理想的数据中心网络传输性能包括低时延、高利用率和拥塞避免等,为了实现网络资源的最优分配,Fastpass 提出利用一个中央控制器(arbiter)控制每个发送端发送状态的方案.当每个包需要进行传输时,需要和 arbiter 进行交互来获得包的发送时间和路径.Fastpass 设计 FCP(fastpass control protocol)来实现低带宽开销和低时延的端系统和 arbiter 之间的消息传递协议,还设计多个 arbiter 之间的复制机制以实现故障容忍的恢复策略.Fastpass 设计最大最小公平分配的时隙分配算法来实现包发送时间分配,并利用快速边涂色算法进行包路径选择,保证 arbiter 为每个包进行时隙和路径分配的效率.将 Fastpass 方案与传统基于 TCP 传输的数据中心网络方案进行对比,Fastpass 能够实现近乎零队列的高吞吐量,显著地缩短了交换机队列的平均大小以及时延,同时能够保证公平的、快速的吞吐量分配.Arbiter 中算法的设计有效地提高了其可扩展性,并能实现细粒度的时隙控制.Fastpass 利用集中式控制的优点,实现了全局感知的包调度策略,提高了网络传输性能,并消除了端系统拥塞控制的需求.然而,由于大部分调度工作在集中式控制器完成,因此,在网络负载较大的情况下,集中式控制器性能容易成为瓶颈,限制了其应用场景.

(3) Hermes

Hermes 是一种基于端系统的不需要修改交换机的负载均衡方案,其在端系统设计感知模块进行不同路径的拥塞感知,并设计重路由模块来确定是否进行重路由.Hermes 根据 RTT 和 ECN 共同感知路径拥塞状态,将路径分为 3 种类型,并利用包的重传和超时事件来推测交换机故障情况.为了提高端系统对网络的可见性,Hermes 采用 power-of-two-choices[15]机制探测部分路径时延,从而有效地辅助负载均衡策略.当包对应的流出现超时情况,或包的路径发生拥塞时,触发重路由过程,从最好的路径开始选择,并考虑重路由为传输带来的收益,从而进行重路由决策.在对称拓扑的实验中,Hermes 相比于 ECMP 和 CLOVE-ECN 机制能够实现更小的流完成时间,性能接近 Presto 和 CONGA.在非对称拓扑实验中,Hermes 由于其及时的重路由特性,相比于 CONGA,能够实现更小的流完成时间,利用其主动探测的特性,相比于 CLOVE-ECN 和 LetFlow,也能实现更小的流完成时间.Hermes 相比于其他方案能够及时、有效地检测故障情况.Hermes 在端系统进行拥塞感知和负载均衡减少了部署成本,并实现了相当的性能.然而,端系统感知拥塞有一定延迟,因此无法及时处理突流情况,而且路径感知和重路由算法中参数较多,如何选择合适的参数需要结合网络情况进行选择.

(4) NDP

NDP 提出一种新的网络传输架构,可为短流实现接近最优的完成时间,并在多种场景下提供大吞吐量.NDP 对交换机和端系统协议栈都进行修改.在交换机本地采用浅缓存策略(8 个包)并执行 CP 机制[54],维护两个优先级队列,低优先级队列存储包数据,高优先级队列存储修剪的包头、ACKs 和 NACKs 包.当一个包到达后,如果低优先级队列溢出,则以 50%的概率选择修剪新到达的包或低优先级队列的尾包.在调度过程中,两个优先级队列采用权重调度方式来提高传输能力.NDP 中发送端进行每个包的路径选择,发送端获取到达目的地的所有路径链表并随机组合,发送包时按照组合的链表顺序发送,当所有路径都发送过一个包后重新组合路径链表继续发

送过程.NDP 采用接收端驱动的传输层协议,在第 1 个 RTT 时间内发送一定窗口的包,然后根据确认的 ACKs 和 NACKs 信息,接收端发送 PULL 包实现未收到数据的重新拉取.当交换机高优先级队列溢出时,直接交换新生成的修剪包头的源地址和目的地址,将包返回给发送端.在端系统,NDP 不需要执行拥塞控制策略.NDP 相比于 DCTCP、MPTCP 和 DCQCN[55]能够实现更小的短流完成时间.在网络高负载情况下,NDP 在交换机队列长度为 8 个包的情况下能够实现近乎最优的最大网络能力,在多对一传输场景中能够实现近优的时延和公平性.NDP 通过重新设计网络协议栈,利用接收端驱动机制实现网络传输的负载均衡,保证小流的低时延和大流的高吞吐量.但 NDP 在高负载网络中,没有拥塞控制机制会造成过多的重传开销,在非对称网络中,NDP 无法处理长度不同的路径分配,而且 NDP 对网内和端系统都进行修改,部署难度较大.

(5) AuTO

AuTO 通过模仿生物神经系统结构设计了一种二级的深度强化学习(deep reinforcement learning,简称 DRL)系统[56]以实现数据中心网络中的流量传输优化.AuTO 基于 PIAS[57]的架构,在主机和交换机本地实现多级反馈队列,根据不同优先级队列对应的流大小阈值,将不同流分配到不同优先级中执行严格的优先级调度.AuTO 在主机实现 PS(peripheral system)系统,收集本地流量信息,并根据控制器发送的不同优先级流大小阈值进行本地流量决策;设计一个控制器 CS(central system),根据网络状态信息通过 sRLA(short flow DRL agent)计算最优的多级反馈队列流阈值,并根据主机上报的大流信息,通过 lRLA(long flow DRL agent)实现大流的路径选择、速率控制和优先级设置.在不同负载情况的实验中,AuTO 相比于最短任务优先和最少服务优先机制能够实现更小的流完成时间.在时间和空间同构或异构的环境中,AuTO 能够明显地提升网络传输能力,实现稳定的性能提升.而且,AuTO 对端系统造成的开销小,能够实现 10ms 内的状态更新.AuTO 将人工智能(artificial intellengence,简称 AI)应用于数据中心网络中的流量传输优化,利用 AI 的自学习自优化特性实现阈值自适应以保证传输效果.但是 AI 方法的可行性和适用性仍是一个问题,当网络场景与训练场景不一致时,基于 AI 的方法是否能够适应不同场景的变化仍需要通过进一步的实验进行验证.

(6) 综合方案对比

综合方案对网络中多个层面进行修改,设计多种机制克服单一修改某个层可能产生的缺陷,实现性能更好的负载均衡效果.通过分析现有综合的方案,从方案的关键设计和方案性能两个角度对各方案进行对比,比较结果见表 4.

方案的关键设计包括方案特点、方案实现关键需求、拥塞信息获取、方案对大小流的考虑情况以及是否考虑不同优先级传输等方面.大部分综合方案对网络具有较大的修改,包括特殊的设备需求、特殊的传输策略以及高性能的处理需求等.通常,综合方案会采集网络拥塞信息作为决策参考,但对于全局最优的方案来说,拥塞信息可以不考虑.综合方案根据设计不同对大小流会进行不同的处理,对于有明显大小流需求的场景,会直接考虑大小的流差异化管理.大部分综合方案都会考虑传输数据的优先级,优先处理控制信息从而进行及时响应.

方案性能主要考虑综合方案的速度、适应性和复杂度等几个方面.对网内设备修改较多的方案通常执行本地决策,因此具有更快的处理速度,而集中式管理的方案会由于管理周期影响其速度.对于设计过程中综合考虑网络特性的方案来说,其相对于考虑特殊情况的方案来说具有更好的适用性,不会由于网络情况变化造成性能的明显下降.大部分综合方案都需要修改网络中的多个部分,因此大多数方案复杂度较高,除了少数只修改端侧的方案外,综合方案的部署难度都会相对较大.

Table 4 The comparison of load balancing schemes based on the synthetic designs in data center networks

表 4 数据中心网络负载均衡综合方案对比

方案	关键设计					方案性能		
	特点	需求	拥塞信息	大小流	优先级	速度	适应性	复杂度
DeTail	采用 PFC 技术实现无损传输	PFC 实现无损二层	交换机队列	不考虑	有	快	高	高
Fastpass	为每个包分配路径和时间	高性能中央控制器	无	不考虑	无	慢	一般	高
Hermes	端侧推测感知拥塞并重路由	端侧修改	RTT 和 ECN	不考虑	无	一般	高	低
NDP	接收端驱动协议	交换机和端侧修改	控制包	考虑	有	一般	一般	高
AuTO	利用二级 DRL 实现传输管理	控制器运行 DRL 模块并决策	流状态	考虑	有	慢	一般	高

3 方案对比和讨论

数据中心网络负载均衡方案通过不同角度的设计保证了低时延和高吞吐的网络性能.表 5 从各个不同方面整体分析并评估了各个方案的特点.根据负载均衡机制之间的差异,从方案类型、负载均衡粒度、方案控制结构、负载均衡类型、拥塞感知机制、网络修改位置、方案扩展性和部署难度几个方面进行了对比.

从不同方案对比中可以发现,大部分数据中心网络负载均衡方案通过网络层传输调度直接进行优化,综合方案普遍性能更好,但部署难度较大.从不同负载均衡粒度来看,粒度较小的方案实现的效果较好.在不同控制结构中,集中式方案会造成巨大开销,从而限制其扩展性.不同的负载均衡类型对网络拥塞的控制效果也不同,被动负载均衡在拥塞后进行处理降低了拥塞响应速度,主动方式可以避免拥塞的发生.对于不同的感知策略,全局拥塞信息感知对负载均衡具有明显的促进作用,可以实现全局最优的调度.现有方案中,越复杂且对网络的修改越多的方案其性能相对来说要更好.然而,实际数据中心网络部署中,可部署性是非常重要的问题,无法简单升级或迭代更新的方案在实际数据中心中难以应用.因此,数据中心网络负载均衡方案的设计需要从性能、可扩展性和实用性等角度综合考虑,在满足传输性能的同时还要尽可能地保证低成本和开销.

Table 5 The evaluation of load balancing schemes in data center networks

表 5 数据中心网络负载均衡方案评估

方案	方案类型	粒度	控制结构	均衡类型	拥塞感知	修改位置	效果	扩展性	部署难度
DRB	网络层	包	分布式	主动	无	端系统+交换机	中	中	容易
RPS	网络层	包	分布式	主动	无	交换机	中	好	容易
DRILL	网络层	包	分布式	主动	本地	交换机	好	好	中等
CONGA	网络层	Flowlet	分布式	被动	全局	ToR	好	中	困难
HULA	网络层	Flowlet	分布式	被动	全局	交换机	好	好	中等
CLOVE	网络层	Flowlet	分布式	被动	全局	端系统	中	好	容易
LetFlow	网络层	Flowlet	分布式	被动	无	交换机	中	好	容易
Presto	网络层	Flowcell	分布式	主动	无	端系统	中	好	容易
ECMP	网络层	流	分布式	主动	无	无	中	好	容易
WCMP	网络层	流	分布式	主动	无	交换机	中	好	容易
Hedera	网络层	流	集中式	主动	全局	控制器	中	差	容易
MicroTE	网络层	流	集中式	主动	全局	控制器+端系统	中	好	容易
Mahout	网络层	流	集中式	主动	全局	控制器+端系统	中	好	容易
Devoflow	网络层	流	集中+分布	主动	全局	控制器+交换机	好	好	中等
LocalFlow	网络层	流	分布式	主动	本地	交换机	中	中	困难
FlowBlender	网络层	流	分布式	主动	全局	端系统	中	好	中等
Freeway	网络层	流	集中式	主动	全局	端系统+交换机	中	中	中等
Expeditus	网络层	流	分布式	主动	本地	交换机	好	好	中等
Scheduling Algorithm	网络层	流	集中式	主动	全局	控制器	好	差	困难
MPTCP	传输层	多流	分布式	被动	全局	端系统	中	好	容易
XMP	传输层	多流	分布式	被动	全局	端系统	中	好	中等
RackCC	传输层	多流	分布式	被动	全局	端系统+ToR	好	中	困难
MMPTCP	传输层	多流	分布式	被动	全局	端系统	好	中	中等
VMS	传输层	包	分布式	被动	全局	端系统	中	中	中等
Ananta	应用层	流	分布式	主动	本地	软件 Mux	中	中	容易
Duet	应用层	流	分布式	主动	本地	软件 Mux+交换机	好	好	容易
Maglev	应用层	流	分布式	主动	本地	软件 Mux	中	中	容易
SilkRoad	应用层	流	分布式	主动	本地	P4 交换机	中	中	困难
Beamer	应用层	流	分布式	主动	本地	软件 Mux	好	中	容易
DeTail	综合	包	分布式	被动	本地	交换机	好	中	中等
Fastpass	综合	包	集中式	主动	全局	控制器+端系统	好	差	困难
Hermes	综合	流	分布式	被动	全局	端系统	好	中	容易
NDP	综合	流	分布式	主动	无	端系统+交换机	好	中	困难
AuTO	综合	流	集中+分布	主动	全局	控制器+端系统	好	中	中等

4 总结与展望

数据中心网络是现代网络和云计算的重要基础架构,数据中心网络的传输性能直接影响网络的传输能力

和服务体验.因此,设计高效的数据中心网络负载均衡方案以满足低时延高吞吐的数据中心网络需求十分关键.现有的数据中心网络负载均衡方案可按照不同解决层面分为:网络层、传输层、应用层和综合方案四大类,其他角度则可从负载均衡粒度、方案控制结构、负载均衡类型和拥塞感知策略等方面对负载均衡方案进行归纳.虽然现有方案设计了多种机制以提高网络传输能力,但仍存在一些弊端:(1) 方案基于传统 TCP 协议栈,性能受限于TCP基于窗口的传输策略;(2) 负载均衡方案性能依赖于现有数据中心典型的网络拓扑;(3) 现有负载均衡方案针对某个场景优化,适用性不强;(4) 现有方案未充分利用数据平面控制能力;(5) 没有充分开发新型传输设备的优势.鉴于现有方案的分析,总结了数据中心网络负载均衡未来可能的研究方向.

(1) 设计新型协议栈

由于传统 TCP 传输协议栈不适用于数据中心的高速场景,在负载均衡过程中无法快速达到最优的网络性能,因此重新设计协议栈是解决传统TCP问题的关键.先前设计的新型协议栈NDP是重新设计网络协议栈的探索,近期提出的 Homa[58]也是通过设计新型协议栈来提高网络传输性能.但是,这些新型协议栈的主要目的是通过进行网络的拥塞控制来提高传输能力,对网络负载均衡的效果只能起到部分提升的作用.因此设计兼顾负载均衡目的的新型协议栈是未来重要的研究方向.

(2) 优化数据中心拓扑

现有数据中心网络负载均衡方案主要基于典型的数据中心网络拓扑(Clos 拓扑)进行设计,负载均衡性能受限于拓扑约束,因此优化数据中心网络拓扑结构是提高网络最大传输能力的重要途径.近期提出的 Flat-tree[59]即是利用可变换的交换结构实现不同场景中的拓扑改变来提升传输性能.负载均衡方案设计时可以结合拓扑优化机制实现较高的网络利用率,并设计最优的引流策略以保证负载均衡的效果.

(3) 结合机器学习和人工智能技术

由于很多数据中心网络负载均衡机制与网络应用场景和流量特性相关,而且设计的机制性能依赖于相关参数的选取,因此可利用机器学习和人工智能技术,通过分析历史数据实现自适应的参数调整.先前的 AuTO 方案即是根据网络中流的状态自适应地给定相关阈值,并进行自学习过程.近期提出的 DRL-TE[60]方案则是利用 DRL 实现流的最优分流传输.目前,此类方案在特定应用场景中效果显著,其适用性仍有待进一步验证.因此结合机器学习和人工智能技术的数据中心网络负载均衡方案是未来研究的一个热点.

(4) 设计新型数据中心网络数据平面

随着网络传输需求的不断增大和各种传输模式的涌现,传统数据中心网络数据平面已经成为提升传输性能的瓶颈,因此出现了许多新型数据平面技术,包括智能网卡的应用[61]、新型网络数据面架构[62]以及网内缓存技术[63]等.这些新型数据平面为网络带来更多的管理维度,结合新维度实现数据中心网络负载均衡是进一步提高数据中心服务能力的关键.利用智能网卡提供的管理能力,可以直接提升端系统均衡吐流能力,同时结合新型数据面架构进一步提高网络负载管理性能.尤其是最近提出的网内缓存技术,可以利用网内缓存对网络负载的吸收特点,设计更深层次的差异化管理,从而保证数据中心网络的均衡负载.因此,结合数据平面新技术是设计数据中心网络负载均衡方案的重要发展方向.

(5) 结合可编程网络技术

受限于传统网络硬件更新周期长和定制能力差等缺点,可编程网络提供了一种高效易管理的网络抽象.典型的可编程网络技术包括 P4[24]和各种可编程交换机技术[64,65],为网络管理者提供转发平面控制接口,从而提供更灵活且高性能的调度能力.因此结合可编程网络技术,可以直接在转发平面定制负载均衡功能,直接感知数据平面状态并做出决策,同时还能根据网络和流量状态变化修改转发平面配置,实现动态负载均衡策略.随着可编程网络技术的不断发展,网络中的可编程设备能够维护和存储的信息量不断增大,使得复杂的负载均衡方案能够在数据中心网络中部署,增大了方案的可扩展性和适用性.随着可编程网络逐渐成为未来研究热点,结合可编程网络技术来设计数据中心网络负载均衡方案也成为了未来的研究热点.

References:

[1] Al-Fares M, Loukissas A, Vahdat A. A scalable, commodity data center network architecture. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2008. 63–74. [doi: 10.1145/1402958.1402967]

[2] Greenberg A, Hamilton JR, Jain N, Kandula S, Kim C, Lahiri P, Maltz DA, Patel P, Sengupta S. VL2: A scalable and flexible data center network. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2009. 51–62. [doi: 10.1145/1592568.1592576]

[3] Singla A, Hong CY, Popa L, Godfrey PB. Jellyfish: Networking data centers randomly. In: Proc. of the Conf. on Networked Systems Design and Implementation. Berkeley: USENIX, 2012. 225–238.

[4] Kandula S, Sengupta S, Greenberg A, Patel P, Chaiken R. The nature of data center traffic: Measurements & analysis. In: Proc. of the ACM SIGCOMM Conf. on Internet Measurement. New York: ACM, 2009. 202–208. [doi: 10.1145/1644893.1644918]

[5] Benson T, Akella A, Maltz D A. Network traffic characteristics of data centers in the wild. In: Proc. of the ACM SIGCOMM Conf. on Internet Measurement. New York: ACM, 2010. 267–280. [doi: 10.1145/1879141.1879175]

[6] Benson T, Anand A, Akella A, Zhang M. Understanding data center traffic characteristics. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2010. 92–99. [doi: 10.1145/1672308.1672325]

[7] Roy A, Zeng H, Bagga J, Porter G, Snoeren A C. Inside the social network's (datacenter) network. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2015. 123–137. [doi: 10.1145/2829988.2787472]

[8] Noormohammadpour M, Raghavendra CS. Datacenter traffic control: Understanding techniques and tradeoffs. IEEE Communications Surveys & Tutorials, 2018,20(2):1492–1525. [doi: 10.1109/COMST.2017.2782753]

[9] Hopps C. Analysis of an equal-cost multi-path algorithm. RFC 2992, 2000. [doi: 10.17487/RFC2992]

[10] Zhou J, Tewari M, Zhu M, Kabbani A, Poutievski L, Singh A, Vahdat A. WCMP: Weighted cost multipathing for improved fairness in data centers. In: Proc. of the European Conf. on Computer Systems. New York: ACM, 2014. 1–14. [doi: 10.1145/2592798.2592803]

[11] Cao J, Xia R, Yang P, Guo C, Lu G, Yuan L, Zheng Y, Wu H, Xiong Y, Maltz D. Per-packet load-balanced, low-latency routing for clos-based data center networks. In: Proc. of the ACM Conf. on Emerging Networking EXperiments and Technologies. New York: ACM, 2013. 49–60. [doi: 10.1145/2535372.2535375]

[12] Dixit A, Prakash P, Hu YC, Kompella RR. On the impact of packet spraying in data center networks. In: Proc. of the Int'l Conf. on Computer Communications. Piscataway: IEEE, 2013. 2130–2138. [doi: 10.1109/INFCOM.2013.6567015]

[13] Ghorbani S, Godfrey B, Ganjali Y, Firoozshahian A. Micro load balancing in data centers with DRILL. In: Proc. of the ACM Workshop on Hot Topics in Networks. New York: ACM, 2015. 17–23. [doi: 10.1145/2834050.2834107]

[14] Ghorbani S, Yang Z, Godfrey P, Ganjali Y, Firoozshahian A. DRILL: Micro load balancing for low-latency data center networks. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2017. 225–238. [doi: 10.1145/3098822.3098839]

[15] Mitzenmacher M. The power of two choices in randomized load balancing. IEEE Trans. on Parallel Distribution System, 2001, 12(10):1094–1104. [doi: 10.1109/71.963420]

[16] Sinha S, Kandula S, Katabi D. Harnessing TCP's burstiness with flowlet switching. In: Proc. of the ACM Workshop on Hot Topics in Networks. New York: ACM, 2004.

[17] Kandula S, Katabi D, Sinha S, Berger A. Dynamic load balancing without packet reordering. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2007. 51–62. [doi: 10.1145/1232919.1232925]

[18] Alizadeh M, Edsall T, Dharmapurikar S, Vaidyanathan R, Chu K, Fingerhut A, Lam VT, Matus F, Pan R, Yadav N, Varghese G. CONGA: Distributed congestion-aware load balancing for datacenters. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2014. 503–514. [doi: 10.1145/2619239.2626316]

[19] Katta N, Hira M, Ghag A, Kim C, Keslassy I, Rexford J. CLOVE: How I learned to stop worrying about the core and love the edge. In: Proc. of the ACM Workshop on Hot Topics in Networks. New York: ACM, 2016. 155–161. [doi: 10.1145/3005745.3005751]

[20] Katta N, Hira M, Kim C, Sivaraman A, Rexford J. Hula: Scalable load balancing using programmable data planes. In: Proc. of the Symp. on SDN Research. ACM, 2016. 10–23. [doi: 10.1145/2890955.2890968]

[21] Vanini E, Pan R, Alizadeh M, Taheri P, Edsall T. Let it flow: Resilient asymmetric load balancing with flowlet switching. In: Proc. of the Conf. on Networked Systems Design and Implementation. Berkeley: USENIX, 2017. 407–420.

[22] Mahalingam M, Dutt D, Duda K, Agarwal P, Kreeger L, Sridhar T, Bursell M, Wright C. Virtual extensible local area network (VXLAN): A framework for overlaying virtualized layer 2 networks over layer 3 networks. RFC 7348, 2014. [doi: 10.17487/RFC7348]

[23] Pan T, Song E, Bian Z, Lin X, Peng X, Zhang J, Huang T, Liu B, Liu Y. INT-path: Towards optimal path planning for in-band network-wide telemetry. In: Proc. of the Int'l Conf. on Computer Communications. Piscataway: IEEE, 2019. 487–495. [doi: 10.1109/INFOCOM.2019.8737529]

[24] Bosshart P, Daly D, Gibb G, Izzard M, McKeown N, Rexford J, Schlesinger C, Talayco D, Vahdat A, Varghese G, Walker D. P4: Programming protocol-independent packet processors. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2014. 87–95. [doi: 10.1145/2656877.2656890]

[25] He K, Rozner E, Agarwal K, Felter W, Carter J, Akella A. Presto: Edge-based load balancing for fast datacenter networks. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2015. 465–478. [doi: 10.1145/2829988.2787507]

[26] Agarwal K, Dixon C, Rozner E, Carter J. Shadow Macs: Scalable label-switching for commodity ethernet. In: Proc. of the ACM Workshop on Hot Topics in Software Defined Networking. New York: ACM, 2014. 157–162. [doi: 10.1145/2620728.2620758]

[27] Al-Fares M, Radhakrishnan S, Raghavan B, Huang N, Vahdat A. Hedera: Dynamic flow scheduling for data center networks. In: Proc. of the Conf. on Networked Systems Design and Implementation. Berkeley: USENIX, 2010. :89–92.

[28] Benson T, Anand A, Akella A, Zhang M. MicroTE: Fine grained traffic engineering for data centers. In: Proc. of the ACM COnf. on Emerging Networking EXperiments and Technologies. New York: ACM, 2011. 8–20. [doi: 10.1145/2079296.2079304]

[29] Curtis AR, Kim W, Yalagandula P. Mahout: Low-overhead datacenter traffic management using end-host-based elephant detection. In: Proc. of the Int'l Conf. on Computer Communications. Piscataway: IEEE, 2011. 1629–1637. [doi: 10.1109/INFCOM.2011.5934956]

[30] Curtis AR, Mogul JC, Tourrilhes J, Yalagandula P, Sharma P, Banerjee S. DevoFlow: Scaling flow management for high-performance networks. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2011. 254–265. [doi: 10.1145/2018436.2018466]

[31] Sen S, Shue D, Ihm S, Freedman MJ. Scalable, optimal flow routing in datacenters via local link balancing. In: Proc. of the ACM Conf. on Emerging Networking Experiments and Technologies. New York: ACM, 2013. 151–162. [doi: 10.1145/2535372.2535397]

[32] Kabbani A, Vamanan B, Hasan J, Duchene F. Flowbender: Flow-level adaptive routing for improved latency and throughput in datacenter networks. In: Proc. of the ACM Conf. on emerging Networking EXperiments and Technologies. New York: ACM, 2014. 149–160. [doi: 10.1145/2674005.2674985]

[33] Wang W, Sun Y, Zheng K, Kaafar MA, Li D, Li Z. Freeway: Adaptively isolating the elephant and mice flows on different transmission paths. In: Proc. of the IEEE Int'l Conf. on Network Protocols. Piscataway: IEEE, 2014. 362–367. [doi: 10.1109/ICNP.2014.59]

[34] Wang P, Xu H, Niu Z, Han D, Xiong Y. Expeditus: Congestion-aware load balancing in clos data center networks. In: Proc. of the ACM Symp. on Cloud Computing. New York: ACM, 2016. 442–455. [doi: 10.1109/TNET.2017.2731986]

[35] Correa JR, Goemans MX. Improved bounds on nonblocking 3-stage clos networks. SIAM Journal on Computing, 2007,37(3): 870–894. [doi: 10.1137/060656413]

[36] Shafiee M, Ghaderi J. A simple congestion-aware algorithm for load balancing in datacenter networks. IEEE/ACM Trans. on Networking (TON), 2017,25(6):3670–3682. [doi: 10.1109/TNET.2017.2751251]

[37] Shafiee M, Ghaderi J. An improved bound for minimizing the total weighted completion time of coflows in datacenters. IEEE/ACM Trans. on Networking (TON), 2018,26(4):1674–1687. [doi: 10.1109/TNET.2018.2845852]

[38] Raiciu C, Barre S, Pluntke C, Greenhalgh A, Wischik D, Handley M. Improving datacenter performance and robustness with multipath TCP. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2011. 266–277. [doi: 10.1145/2018436.2018467]

[39] Cao Y, Xu M, Fu X, Dong E. Explicit multipath congestion control for data center networks. In: Proc. of the ACM Conf. on Emerging Networking Experiments and Technologies. New York: ACM, 2013. 73–84. [doi: 10.1145/2535372.2535384]

[40] Zhuo D, Zhang Q, Liu V, Krishnamurthy A, Anderson T. Rack-level congestion control. In: Proc. of the ACM Workshop on Hot Topics in Networks. New York: ACM, 2016. 148–154. [doi: 10.1145/3005745.3005772]

[41] Kheirkhah M, Wakeman I, Parisis G. MMPTCP: A multipath transport protocol for data centers. In: Proc. of the Int'l Conf. on Computer Communications. Piscataway: IEEE, 2016. 1–9. [doi: 10.1109/INFOCOM.2016.7524530]

[42] Li Z, Bi J, Zhang Y, Dogar AB, Qin C. VMS: Traffic balancing based on virtual switches in datacenter networks. In: Proc. of the IEEE Int'l Conf. on Network Protocols. Piscataway: IEEE, 2017. 1–10. [doi: 10.1109/ICNP.2017.8117566]

[43] Patel P, Bansal D, Yuan L, Murthy A, Greenberg A, Maltz DA, Kern R, Kumar H, Zikos M, Wu H, Kim C, Karri N. Ananta: Cloud scale load balancing. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2013. 207–218. [doi: 10.1145/2486001.2486026]

[44] Gandhi R, Liu HH, Hu YC, Lu G, Padhye J, Yuan L, Zhang M. Duet: Cloud scale load balancing with hardware and software. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2014. 27–38. [doi: 10.1145/2619239.2626317]

[45] Eisenbud DE, Yi C, Contavalli C, Smith C, Kononov R, Hielscher EM, Cilingiroglu A, Cheyney B, Shang W, Hosein JD. Maglev: A fast and reliable software network load balancer. In: Proc. of the Conf. on Networked Systems Design and Implementation. Berkeley: USENIX, 2016. 523–535.

[46] Miao R, Zeng H, Kim C, Lee J, Yu M. SilkRoad: Making stateful layer-4 load balancing fast and cheap using switching ASICs. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2017. 15–28. [doi: 10.1145/3098822. 3098824]

[47] Olteanu V, Agache A, Voinescu A, Raiciu C. Stateless datacenter load-balancing with beamer. In: Proc. of the Conf. on Networked Systems Design and Implementation. Berkeley: USENIX, 2018. 125–139.

[48] Zats D, Das T, Mohan P, Borthakur D, Katz R. DeTail: Reducing the flow completion time tail in datacenter networks. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2012. 139–150. [doi: 10.1145/2342356.2342390]

[49] Perry J, Ousterhout A, Balakrishnan H, Shah D, Fugal H. Fastpass: A centralized zero-queue datacenter network. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2015. 307–318. [doi: 10.1145/2619239.2626309]

[50] Zhang H, Zhang J, Bai W, Chen K, Chowdhury M. Resilient datacenter load balancing in the wild. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2017. 253–266. [doi: 10.1145/3098822.3098841]

[51] Handley M, Raiciu C, Agache A, Voinescu A, Moore AW, Antichi G, Wojcik M. Re-architecting datacenter networks and stacks for low latency and high performance. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2017. 29–42. [doi: 10.1145/3098822. 3098825]

[52] Chen L, Lingys J, Chen K, Liu F. AuTO: Scaling deep reinforcement learning for datacenter-scale automatic traffic optimization. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2018. 191–205. [doi: 10.1145/3230543. 3230551]

[53] Mittal R, Shpiner A, Panda A, Zahavi E, Krishnamurthy A, Ratnasamy S, Shenker S. Revisiting network support for RDMA. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2018. 313–326. [doi: 10.1145/3230543.3230557]

[54] Cheng P, Ren F, Shu R, Lin C. Catch the whole lot in an action: Rapid precise packet loss notification in data center. In: Proc. of the Conf. on Networked Systems Design and Implementation. Berkeley: USENIX, 2014. 17–28.

[55] Zhu Y, Eran H, Firestone D, Guo C, Lipshteyn M, Liron Y, Padhye J, Raindel S, Yahia MH, Zhang M. Congestion control for large-scale RDMA deployments. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2015. 523–536. [doi: 10.1145/2829988.2787484]

[56] Silver D, Huang A, Maddison C, Guez A, Sifre L, Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lilicrap T, Leach M, Kavukcuoglu K, Graepel T, Hassabis D. Mastering the game of Go with deep neural networks and tree search. Nature, 2016,529(7587):484–489. [doi: 10.1038/nature16961]

[57] Bai W, Chen L, Chen K, Han D, Tian C, Wang H. Information-agnostic flow scheduling for commodity data centers. In: Proc. of the Conf. on Networked Systems Design and Implementation. Berkeley: USENIX, 2015. 455–468.

[58] Montazeri B, Li Y, Alizadeh M, Ousterhout J. Homa: A receiver-driven low-latency transport protocol using network priorities. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2018. 221–235. [doi: 10.1145/3230543. 3230564]

[59] Xia Y, Sun XS, Dzinamarira S, Wu D, Huang XS, Ng TSE. A tale of two topologies: Exploring convertible data center network architectures with flat-tree. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2017. 295–308. [doi: 10.1145/ 3098822.3098837]

[60] Xu Z, Tang J, Meng J, Zhang W, Wang Y, Liu CH, Yang D. Experience-driven networking: A deep reinforcement learning based approach. In: Proc. of the Int'l Conf. on Computer Communications. Piscataway: IEEE, 2018. [doi: 10.1109/INFOCOM.2018.8485853]

[61] Le Y, Chang H, Mukherjee S, Wang L, Akella A, Swift MM, Lakshman TV. UNO: Uniflying host and smart NIC offload for flexible packet processing. In: Proc. of the Symp. on Cloud Computing. New York: ACM, 2017. 506–519. [doi: 10.1145/3127479. 3132252]

[62] Li Y, Wei D, Chen X, Song Z, Wu R, Li Y, Jin X, Xu W. DumbNet: A smart data center network fabric with dumb switches. In: Proc. of the EuroSys Conf. New York: ACM, 2018. 9:1–9:13. [doi: 10.1145/3190508.3190531]

[63] Jin X, Li X, Zhang H, Soule R, Lee J, Foster N, Kim C, Stoica I. NetCache: Balancing key-value stores with fast in-network caching. In: Proc. of the Symp. on Operating Systems Principles. New York: ACM, 2017. 121–136. [doi: 10.1145/3132747. 3132764]

[64] Chole S, Fingerhut A, Ma S, Sivaraman A, Vargaftik S, Berger A, Mendelson G, Alizadeh M, Chuang ST, Keslassy I, Orda A, Edsal T. DRMT: Disaggregated programmable switching. In: Proc. of the Conf. on Special Interest Group on Data Communication. New York: ACM, 2017. 1–14. [doi: 10.1145/3098822.3098823]

[65] Sharma NK, Liu M, Atreya K, Krishnamurthy A. Approximating fair queueing on reconfigurable switches. In: Proc. of the Conf. on Networked Systems Design and Implementation. Berkeley: USENIX, 2018. 1–16.

沈耿彪(1991－),男,硕士生,主要研究领域为数据中心,负载均衡,流量调度,协议栈,软件定义网络,内容分发网络.

汪漪(1983－),男,博士,CCF 专业会员,主要研究领域为未来网络体系架构,信息中心网络,软件定义网络,高性能网络器件设计与实现,智能化网络.

李清(1985－),男,博士,CCF 专业会员,主要研究领域为网络体系架构,路由协议与算法,网络传输调度,边缘计算.

徐明伟(1971－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为计算机网络.

江勇(1975－),男,博士,教授,博士生导师,CCF 专业会员,主要研究领域为计算机网络体系结构,下一代互联网,人工智能网络.