ELAB：基于端系统的新型拥塞感知负载均衡机制

陈果，张潍丰
（湖南大学信息科学与工程学院，湖南 长沙 410082）

摘 要：良好的负载均衡机制是有效利用数据中心网络带宽的必备条件。现有的等价多路径路由（ECMP）的负载均衡方法由于负载均衡粒度过粗，且不具备对路径拥塞状态感知的能力，因此负载均衡效率较低。为解决此问题，近年来出现了一系列细粒度且具备拥塞感知能力的负载均衡研究工作。然而，这些研究或者需要修改交换机硬件以实时搜集网络各部位的拥塞情况，难以部署；或者虽不需要修改交换机硬件，仅需对端系统的软件进行修改，但由于缺乏准确的网络拥塞信息而导致负载混合效果不佳。针对该问题，提出了一种实现于端系统上的软件解决方案 ELAB，该方案不需要对网络中的硬件进行修改，就可以达到良好的负载均衡效果。ELAB 创造性地采用了基于可用带宽的方式进行流量负载均衡，相比于现有的基于端系统的方法，ELAB 性能提升达 20%。

关键词：数据中心网络；负载均衡；基于边缘；拥塞感知
中图分类号：TP393.1
文献标识码：A
doi: 10.11959/j.issn.1000−436x.2019054

ELAB: end-host-based congestion aware load balancing for data center network

CHEN Guo, ZHANG Weifeng
College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China

Abstract: A good load balance mechanism is the key to effectively use the network of the data center network. In current production data center, ECMP is the de facto load balancing scheme. However, it has two drawbacks. 1) the load balance unit is too coarse-grained, 2) it's not congestion aware. To solve these problems, several fine-grained and congestion-aware load balancing works have emerged in recent years. These works either need to modify the switch hardware to collect congestion in various parts of the network in real time, and it is difficult to deploy; or only need to modify the end system, but the inaccurate sense of congestion cannot achieve a good load balancing effect. A novel edge-based load balancing scheme ELAB was proposed, which addresses above existing problems and improves the network performance up to 20%.

Key words: data center network, load balancing, edge-based, congestion aware

1 引言

数据中心承载着人们日常生活中所用到的各种各样的互联网服务，已成为当前云计算时代最重要的基础设施之一。为满足日益增长的性能需求，数据中心网络（DCN, data center network）作为连接数据中心内所有计算和存储资源的“交通枢纽”，需要提供越来越高的网络带宽以支持数据中心内部的大量通信需求[1]。因此，现有数据中心网络多采用基于 Clos 结构的密集网络拓扑架构[2]，通过大量的并行路径，为数据中心提供极高的内部网络带宽。

为了有效地利用整个网络的带宽，目前，在实际环境中多采用等价多路径路由（ECMP, equal cost multi-path）[3]负载均衡机制将网络流量分发到 DCN 内部的多条并行路径上进行传输。在 ECMP 机制

收稿日期：2018−06−25；修回日期：2018−08−24

中，网络中的每个交换机对途经数据分组头部的五元组（源 IP、目的 IP、源端口、目的端口、协议号）通过散列函数计算散列值，随后根据散列值为数据分组选择传输路径，将不同的数据流散列至不同并行路径上。然而，ECMP 方法主要存在以下 2 个问题：1) 基于流（五元组）进行负载均衡的粒度过粗，常常由于散列冲突导致网络负载不均衡；2) 缺少对路径拥塞状态的感知，因此无法在网络出现非对称的情况下根据各路径的状况进行流量分发。针对 ECMP 的这 2 个问题，近年来，出现了一系列工作研究如何为数据中心网络提供更细粒度且具备拥塞感知能力的负载均衡机制[4-8]。

将负载均衡的单元切分为比数据流更细的粒度较为容易，有基于小流（flowlet）[9]、流信元（flowcell）[10]、数据分组[11]等多种方式，但提供高效的动态拥塞感知能力却一直是研究难点。依据负载均衡机制实现拥塞感知的位置，可将现有研究方案分为 2 类：基于网络的拥塞感知[4-6]和基于边缘的拥塞感知[7-8]。基于网络的拥塞感知以 CONGA（congestion aware balancing）[6]为代表。此类方案依赖于在网络中交换机各端口上进行实时的流量监控，判断路径的拥塞程度。由于是对网络内部进行检测，基于网络的拥塞感知可以动态地选择拥塞程度较轻的路径。然而，需要对现有交换机硬件做较大改动，因此难以在实际 DCN 环境中进行部署。基于边缘的拥塞感知以 Clove[7]和 Hermes[8]为代表。这类方案实现在端系统上只需修改端系统软件代码即可完成，易于部署。然而，由于缺乏对网络路径实时状况的准确监测信息，因此只能通过往返时延（RTT, round-trip time）或者显示拥塞控制通知（ECN, explicit congestion notification）等信号来猜测最优路径，进行启发式选路。这些信号往往只能从侧面反映路径的状态，而不能准确地表明路径的拥塞程度，尤其是在信号反馈时延较长时对路径拥塞程度的判断更加不准确。因此，这类方案在复杂的真实网络状况下往往性能表现不佳。

由上述分析可知，目前缺乏一种易于部署的具备良好动态拥塞感知能力的负载均衡机制。基于此，本文提出了一种基于边缘的利用可用带宽进行拥塞感知的负载均衡机制（ELAB，edge-based load balancing according to path available bandwidth）。ELAB 结合了已有基于网络和基于边缘 2 类方案的优点，较好地解决了目前负载均衡拥塞感知的难点。ELAB 实现在端系统上，不需要对现有网络硬件做任何修改。ELAB 接收端通过检测数据分组到达速率，实时计算各条路径的传输速度，从而辅助发送端判断出各条路径的可用带宽。基于此，ELAB 可以在不修改现有网络硬件的情况下，在端系统上实现效果良好的拥塞感知负载均衡。

本文在网络仿真平台 NS2 上实现了 ELAB，并将其与现有基于边缘的拥塞感知方法 Clove 和 Hermes 进行对比。实验结果表明，在非对称网络情况下 ELAB 的吞吐率较 Clove 和 Hermes 提高了 10%~20%。

2　相关工作

数据中心网络的负载均衡是近年来相关领域的研究重点，出现了一系列的相关工作成果。依据是否具备全局的动态拥塞感知能力，负载均衡的方法可总结为 2 类：静态负载均衡与动态拥塞感知的负载均衡。

静态负载均衡具有代表性的方法研究主要有 Presto[10]、DRB[11]和 Packet Spray[12]。这类工作主要通过减小负载均衡调度的粒度（如以分组为单位）来实现负载均衡最优的目标，流量均衡过程中不对网络状态进行感知。在对称的网络结构中，静态负载均衡方法能有效地提升负载均衡的效果，然而因为缺乏动态感知拥塞的能力，在网络出现非对称的情况下（如链路故障）表现不佳。

动态拥塞感知的负载均衡动态地监测网络的拥塞状态，从而做出相应的选路决策。该类型工作又可进一步细分为以下 2 类。

1) 集中式拥塞感知的负载均衡，其典型代表有 Hedera[13]、MicroTE[14]和 Mahout[15]。这类工作通过集中式控制的方式，监控网络中各部位的负载信息以及数据流情况，随后将数据流重新路由到负载低的路径上。得益于全局的集中式控制，这类工作可以很好地均衡调度数据中心网络中的长流。但其主要的缺陷在于集中式的监控和调度时延太长。具体来说，集中式方法从感知到网络路径状态的变化，到做出集中式调度，往往需要经历数十秒甚至更长的时延[13]，难以适应突发短流的负载均衡需求（微秒级）。而本文提出的 ELAB 机制能在数十微秒内感知路径拥塞并做出相应的负载均衡调整。

2) 分布式拥塞感知的负载均衡，除前文提到的 CONGA[6]、Clove[7]、Hermes[8]外，具有代表性的工

作还有 Flowtune[16]、DRILL[5]等。分布式的方式大大降低了这类工作获取网络拥塞情况的时延，使其可以更好地应对网络拥塞变化。但正如前文所述，这类工作或者需要修改交换机硬件以实时搜集网络各部位的拥塞情况，难以部署，如 CONGA；或者虽仅需对端系统的软件进行修改，但无法达到良好的负载均衡效果，如 Clove 和 Hermes。通过本文实验可见，现有最新的基于端系统的负载均衡机制的性能皆远低于理论最优性能。而本文提出的 ELAB 机制在现有方法的基础上显著地提升了性能，将网络平均流完成时间优化了 10%~20%。

3 研究思路

本节首先简单介绍现有基于边缘的负载均衡机制如何进行拥塞感知，通过一个简单的例子论述现有机制选路性能不佳的原因；然后，分析基于可用带宽分配的负载均衡机制能很好地解决该问题，从而引出 ELAB 的设计出发点；最后，讨论如何在端系统中探测可用带宽，介绍 ELAB 的设计思路。

3.1 现有基于边缘的负载均衡机制

现有基于边缘的具备动态拥塞感知能力的负载均衡机制主要有 2 个代表性的方法：Clove 和 Hermes。与基于网络的方法不同，基于边缘的方法无法准确、实时地获取网络中各路径的拥塞程度，因此多采用 RTT 或 ECN 等信号来推测路径的拥塞程度。

Clove 为每条并行路径维护一个权重作为选路的指标，初始阶段各路径权重相同，当某条路径收到 ECN 标记时则对其权重进行一定量的衰减（文献[7]中衰减为原权重的$\frac{2}{3}$），降低该路径的发送速率。Hermes 除 ECN 外，还检测每个路径的 RTT，根据每条路径的 ECN 和 RTT 的值是否处于一定的范围内来判断每个路径的好坏，从而选择好路避开坏路。

下面通过一个简单的例子，说明现有利用 RTT 或 ECN 信号的方法无法做出最佳的选路决策。考虑如图 1 所示情形（以下简称简单情形），网络中存在 2 条并行路径，其中路径 2 的带宽为 6 Gbit/s，路径 1 因为发生故障带宽降为 3 Gbit/s。存在一个 TCP 流经过这 2 条并行路径发送数据。为了考察网络负载均衡的性能，假设该流入口和出口带宽无限大，从而排除输入/输出端口的影响，使带宽瓶颈出现在网络中。在仿真环境中，流量产生端和接收端到叶子交换机 1/叶子交换机 2 的带宽设置为 10 Gbit/s，该网络中每一跳的时延根据实际数据中心环境设置为 20 μs，端到端时延为 160 μs，叶子交换机至骨干交换机的链路出口队列长度设置为 1.8 MB（3 倍带宽时延积），ECN 阈值设置为 600 KB（1 倍带宽时延积）。图中 spine 表示骨干交换机，leaf 表示叶子交换机。

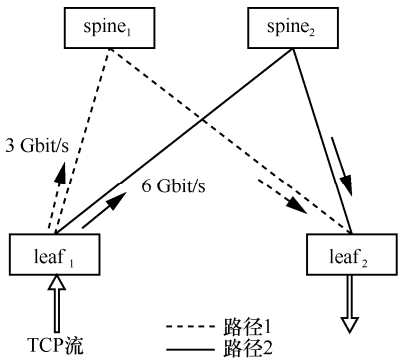


图 1 简单情形下的网络负载均衡

对图 1 所示简单情形网络采用不同的负载均衡方案进行仿真，不同方案下网络带宽利用率如图 2 所示。

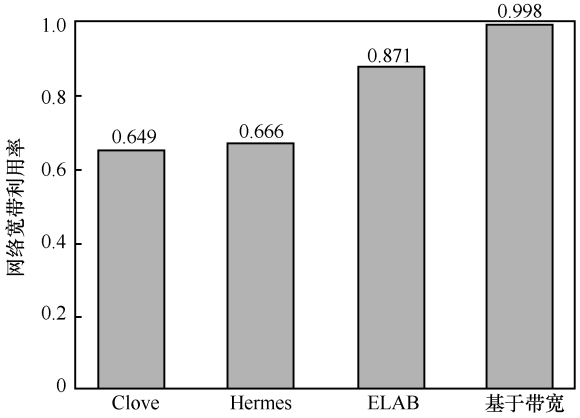


图 2 简单情形下各负载均衡机制的网络带宽利用率对比

理论上最优的负载均衡情况下，该 TCP 流应将网络中所有并行路径完全利用，利用率达到 100%。然而，Clove 和 Hermes 这 2 种方法都远没有达到最优的网络利用率，具体原因如下。

1) 在 Clove 中，初始阶段 2 条路径权重相同，流量按照 1:1 的比例均分到 2 条路径上。当 TCP 流速度增至 6 Gbit/s，即 2 条路径的速度皆为 3 Gbit/s 时，路径 1 上将收到 ECN。此时路径 1 的权重衰减为初始值的$\frac{2}{3}$，则后续流量将按照 2:3（即$\frac{2}{3}$:1）

进行分配。此时 TCP 流不会立即降速（Clove 仅在所有路径皆收到 ECN 时将 ECN 信号返回给上层 TCP），2 条路径的流量速度分别调整至 2.4 Gbit/s 和 3.6 Gbit/s，且后续保持 2:3 的比例分配。当 TCP 流吞吐量增至 7.5 Gbit/s 时，2 条路径的速度分别为 3 Gbit/s 和 4.5 Gbit/s，此时路径 1 再次收到 ECN，权重衰减为初始值的$\frac{4}{9}$，2 条路径的流速按照 4:9（即$\frac{4}{9}$:1）进行重新分配，此时 2 条路径的流速分别为 2.3 Gbit/s 和 5.2 Gbit/s。随后，当 TCP 流吞吐量增至 8.6 Gbit/s，2 条路径的速度分别为 2.6 Gbit/s 和 6 Gbit/s 时，路径 2 也收到 ECN，TCP 开始降速。此时路径 2 的权重衰减为初始值的$\frac{2}{3}$，则后续流量将按照 2:3（即$\frac{4}{9}:\frac{2}{3}$）进行分配（与路径 1 权重降低后情况相同）。如果把这个过程延长，那么 2 条路径的流量比例将会一直在 2:3 和 4:9 之间振荡，而始终无法以最优的比例 1:2 同时将 2 条并行路径完全利用。图 3 展示了 Clove 对 2 条路径分配的流量速度比例（路径 1 流速:路径 2 流速）。每次振荡时都会收到 ECN，TCP 流随之降速。因此，Clove 中 TCP 流无法同时完全利用 2 条并行路径，达到最优的 9 Gbit/s 吞吐量。

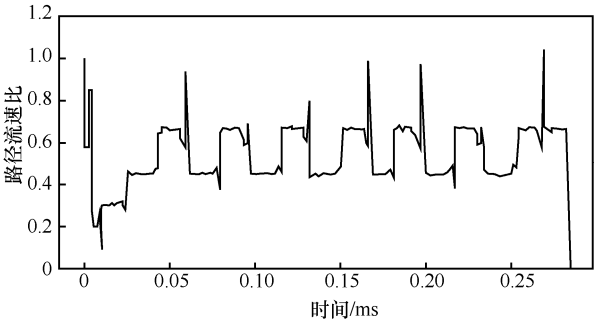


图 3　简单情形下 Clove 中 2 条路径的流速比
（路径 1 流速:路径 2 流速）

2) 在 Hermes 中，初始阶段 2 条路径的 RTT 都很小（无排队时延），也没有 ECN，因此流量以 1:1 分到 2 条路径上。当 TCP 流速度增至 6 Gbit/s，即 2 条路径速度皆为 3 Gbit/s 时，路径 1 的 RTT 开始上升并收到 ECN，被判断为坏路。此时 TCP 流收到 ECN 降速并重新进入线性增速阶段。Hermes 会将后续的流量全部分发至路径 2。直至 TCP 流吞吐量达到 6 Gbit/s 时，路径 2 也会出现 ECN 且 RTT 上升，被判断为坏路，此时 TCP 流收到 ECN 再次降速。流量在 2 条路径之间振荡，而无法保持以最优的比例 1:2 同时将 2 条并行路径完全利用。图 4 展示了 Hermes 对 2 条路径分配的流量速度比例（路径 1 流速:路径 2 流速）。如图 4 所示，虽然 Hermes 分配速度比例的振荡中心在 0.5 附近，但流量一直在 2 条路径之间大幅切换。这会导致某条路径被用满时 TCP 流收到 ECN 降速，而此时另一条路径负载还比较轻。因此，Hermes 也无法同时完全利用 2 条并行路径，达到最优的 9 Gbit/s 吞吐量。

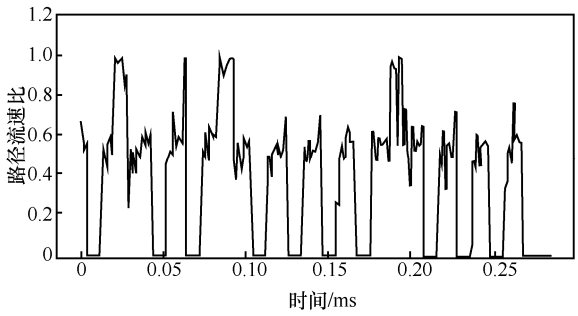


图 4　简单情形下 Hermes 中 2 条路径流速比
（路径 1 流速:路径 2 流速）

3.2　基于可用带宽的负载均衡

由上述实验分析可知，现有基于边缘的方法无法做出最优负载均衡决策，其原因在于，仅通过 RTT 或 ECN 信号无法准确地反映一条路径的拥塞程度。具体来说，一条路径只有被完全利用出现拥塞时才有可能出现 ECN 或 RTT 上升，此时再根据相关信号来选择其他路径为时已晚，TCP 已经出现降速，因此无法达到最优吞吐量。此外，即使 Clove 刻意隐藏 ECN 信号，虽可避免在探测到某一条路径拥塞时立即出现 TCP 降速，但仍无法知道其他路径当前的拥塞程度，因此同样不能以最优的比例进行流量分配，后续依然无法使所有路径同时被完全利用。

若想使网络中数据流的吞吐量最大，最优的负载均衡决策需要让网络中所有并行链路利用率相等（即同时完全利用）。要实现这一目标，可以通过基于各条路径当前的可用带宽比例进行流量分配。

同样地，考虑图 1 所示的简单情形。如果严格按照实时可用带宽的比例进行流量分配，由图 2 可知，采用基于带宽的方案 TCP 流的吞吐量可以达到最优的 9 Gbit/s，达到 99.8%的网络利用率。

因此，基于以上观察，本文所提出的方案 ELAB 采用基于可用带宽分配流量的负载均衡原则。

3.3 在端系统中探测可用带宽

现有的商用网络设备不支持直接在网络中监测各路径的可用带宽。若想设计一种易于部署、不需要对现有网络硬件进行修改的负载均衡机制，需要在端系统上对网络路径的可用带宽进行探测，这也是 ELAB 的设计难点所在。

经分析，可以利用接收端的配合来探测路径可用带宽。具体来说，在接收端可以实时计算每条路径上收到数据的传输速度（此处假设接收端知道每个数据分组的发送路径）。假设已知第 i 条路径的流量速度为 R_i，若已知其物理带宽为 B_i，则该路径的可用带宽为

$$A_i = B_i - R_i \tag{1}$$

值得注意的是，式(1)在实际应用中容易出现不准确的情况，其原因如下：1) 即使预先知道网络中链路的物理带宽 B_i，现实中链路也可能由于网络故障而发生带宽骤降；2) 网络中不同节点对之间可能同时存在多个并发数据流，数据流仅知道其自身在某条路上的流速 R_i，却无法实时获取其他流在各个路径上的流量速度，因此无法获知路径上准确的可用带宽 A_i。为解决以上问题，可以利用现有 ECN 信号来逐步探测准确的带宽。具体来说，对于一个数据流，一旦某条路径上收到 ECN，则将该路径的物理带宽 B_i 更新为当前该路径上的流量速度。若网络中相同路径上只有一条数据流，则该流量速度为准确的物理带宽（同样能很好地探测到网络故障降速后的带宽）；若网络中有其他数据流共享该路径，则 B_i 会更新为各数据流对于该链路的共享比例（例如 2 个数据流共享一条 40 GB 链路，若两者流量速度相等，则 2 个数据流的 B_i 皆更新为 20 GB），因此 $B_i - R_i$ 同样能准确反映各数据流在该路径共享比例下的可用带宽。

4 ELAB 详细设计

本节将介绍 ELAB 机制的详细设计，描述 ELAB 如何在边缘上利用可用带宽进行拥塞感知的负载均衡。

4.1 设计综述

ELAB 算法实现在端系统上，不需要对现有网络硬件进行修改，其结构如图 5 所示。ELAB 作为中间层，实现在端系统协议栈中的网络层和传输层之间，通过隧道封装/解封装的方式在端系统上实现负载均衡选路。ELAB 对上层传输层完全保持透明，不需要对现有传输层协议做任何修改。

在发送端，ELAB 将传输层发来的数据分组进行隧道封装，加入一个传输层隧道分组头（L4 tunnel header）用于选路。因为现有数据中心网络硬件皆采用 ECMP 的选路方式（即对数据分组头的五元组进行散列选路），所以在不改变已有网络硬件的前提下，可以通过改变隧道头中五元组的方式来在端系统上进行主动选路。本文中每个不同的隧道五元组在被称为一条虚拟路径（VP, virtual path）。如前文所述，ELAB 为每一对通信节点之间维护一个 VP 状态表，不断监测各条 VP 上的可用带宽，从而进行拥塞感知的动态选路。

在接收端，ELAB 将接收到的数据分组去除隧道头，将数据分组还原后交给上层协议处理。同时，它也为每一对通信节点之间维护一个 VP 状态反馈表。该表的功能是记录每个虚拟路径上最近一段时间内接收到的数据分组数量，并在适当时将这个信

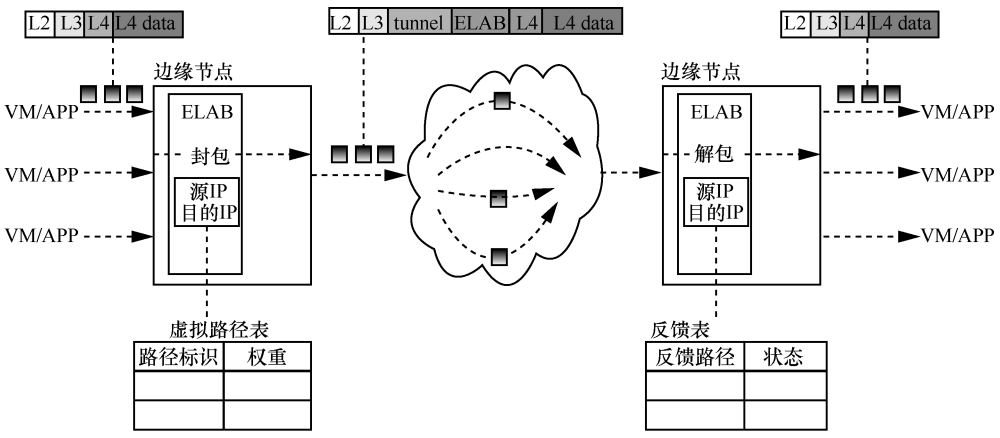


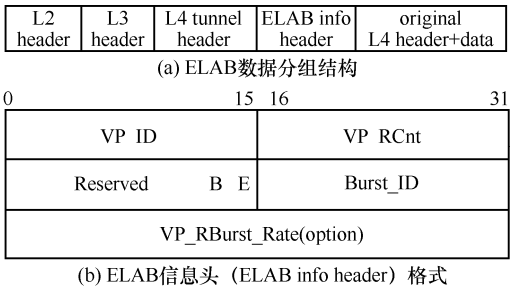
图 5 ELAB 总体结构

息送至发送端以计算路径拥塞状态。

本文在没有特殊说明的情况下，所有的讨论均为一对通信节点之间。通信节点对可以通过源/目的 IP 地址来进行标识。所有的 ELAB 节点间通信为双向通信，即每一个节点既是 ELAB 发送端，也是 ELAB 接收端。

4.2　ELAB 数据分组头部格式

ELAB 数据分组结构如图 6(a)所示。ELAB 在原来数据分组的网络层头部和传输层头部之间加入了一个传输层隧道头（L4 tunnel header）。此外，ELAB 在发送端和接收端之间还需传输一些必要的信息以计算路径的拥塞状态，因此，ELAB 在传输层隧道分组头之后插入了一个自定义的 ELAB 信息头（ELAB info header），具体分组格式如图 6(b)所示。



(a) ELAB数据分组结构

(b) ELAB信息头（ELAB info header）格式

图 6　ELAB 数据分组结构及 ELAB 信息头格式

传输层隧道头在保持现有网络 ECMP 路由不变的情况下，在端系统上选择不同的网络传输路径。现有数据中心网络通常采用的传输层隧道格式有 VXLAN[17]和 SSTP[18]等。本文为描述简单，ELAB 采用了分组头长度最小的 UDP 隧道头。具体地，在原有数据分组的三层分组头和四层分组头之间插入一个新 UDP 头部，其内容如下：1) 用一个特殊的 UDP 目的端口号来标识该流量是 ELAB 产生的流量；2) 将 UDP 头中不同的源端口号作为不同 VP 的标识。因此，ELAB 通过对不同数据分组设置不同的 UDP 隧道头中的源端口号，即可实现主动选路。

ELAB 信息头用于在 ELAB 发送端和接收端之间传输计算拥塞状态所需的必要信息。此处先仅对各字段的含义做简要介绍，后文将详细介绍在发送端和接收端如何设置这些字段以及如何利用这些字段计算各路径的拥塞状态。ELAB 信息头具体包括如下字段。

1) VP_ID（16 位）和 VP_RCnt（16 位）。用于接收端反馈某条虚拟路径在最近一段时间内接收到的数据分组数量。VP_ID 表示当前所反馈的 VP 的标识。VP_RCnt 为 VP 状态反馈表中该 VP 接收的数据分组数目。

2) Reserved（14 位）。保留字段，暂未使用。

3) B（1 位）和 E（1 位）。这 2 个标志位皆由 ELAB 接收端反馈时设置。E 位表示最近在该虚拟路径上是否接收到被 ECN 标记的数据分组。B 位表示该数据分组中是否携带有路径探测速率的反馈信息。

4) Burst_ID（16 位）。由发送端设置。若该数据分组为路径速率探测分组，则发送端将该字段设置为该探测分组的探测序号。

5) VP_RBurst_Rate（32 位）。可选字段，由接收端反馈时设置。当 B 位为 1 时，该字段表示该路径的探测速率；B 位为 0 时，无该字段。

4.3　虚拟路径空间的选取

本节简单介绍 ELAB 对虚拟路径空间的选取。前文提到利用传输层隧道头中的源端口号作为虚拟路径的标识。然而，不同的虚拟路径可能被底层网络的 ECMP 路由散列到同一条物理路径之上。如果盲目地选择一段源端口号范围作为传输中使用的虚拟路径空间，则既有可能无法覆盖网络中所有的并行路径，也有可能导致大部分虚拟路径对应的是少数相同的物理路径，这 2 种情况都会严重地影响选路效率。针对此问题，ELAB 采用了经典 Paris traceroute 机制[19]，在初始化阶段对虚拟路径进行探测，从而选出对应不同物理路径的虚拟路径。

在新通信节点对之间产生流量时，ELAB 会对该通信节点对之间的虚拟路径空间进行初始化。初始化时，随机产生源端口号，向目的地址发送一组带有不同源端口号的 traceroute。借助这些 traceroute 的反馈信息，即可知道不同源端口号经过的具体路径。为了完善虚拟路径表，需要随机产生多组源端口号不同的探测分组，尽可能地将网络中所有的并行路径探测出来。当反馈回的物理路径信息与已完成探测的虚拟路径不一致时，即被认为是新的并行物理路径，加入虚拟路径空间中。

值得注意的是，以上虚拟路径空间的初始化探测在新通信对之间第一次出现流量时进行，后续过程中仅需以极低的频率周期性地探测即可（为应对网络拓扑变化），引入的网络开销很低。

4.4 基于可用带宽的选路

本节将具体介绍 ELAB 如何根据可用带宽动态地选路。

4.4.1 选路函数

在 ELAB 发送端，为每一条虚拟路径维护上限带宽、实际发送速度以及可用带宽几个状态，分别如下。

1) B_i：第 i 条 VP 的上限带宽。

2) R_i：第 i 条 VP 的实际发送速度。

3) A_i：第 i 条 VP 的可用带宽，$A_i = B_i - R_i$。

ELAB 按照可用带宽比例进行流量分配。假设共有 n 条 VP，则第 i 条 VP 上的流量比例 w_i 为

$$w_i = \frac{A_i}{\sum_{j=i}^{n} A_i} \tag{2}$$

具体做法为在每次有数据分组从发送端发出时，ELAB 都会以上述比例轮询地从各条 VP 中选择一条，将数据分组从该 VP 发送出去。

4.4.2 路径发送速度 R_i 的探测

ELAB 发送端依赖于接收端的配合来实时探测每条路径的实时发送速度。ELAB 接收端为每一对通信节点之间维护一个 VP 状态反馈表。该表的功能是记录每个虚拟路径上最近一段时间内接收到的数据分组的数量。具体来说，ELAB 接收端每收到一个数据分组时，会将其对应的 VP 接收数量加 1；而每当接收端有数据分组发回发送端时，ELAB 会以轮询的方式依次将 VP 状态反馈表中各条 VP 收到的数据分组数量反馈给发送端，同时将该轮反馈过的 VP 接收到的数据分组数量清零。

发送端利用上述接收端反馈的路径接收分组数量来计算一条 VP 的实时发送速度。发送端对第 i 条 VP 的实际发送速度 R_i 的计算式为

$$R_i = (1-\alpha)R_i^* + \alpha\frac{\mathrm{RCnt}_i}{T_i} \tag{3}$$

其中，R_i^* 是上一时刻第 i 条 VP 的实际发送速度；RCnt_i 表示从上一时刻到现在接收端总共反馈的第 i 条 VP 接收的数据分组数量；T_i 是上一时刻到现在经过的时间长度；$\frac{\mathrm{RCnt}_i}{T_i}$ 则表示最近一段时间内该 VP 的实际发送速度；α 是一个加权平均参数，表示最近一段时间内的发送速度在计算整体发送速度时所占的比重，$\alpha \in [0,1]$。

4.4.3 路径上限带宽 B_i 的探测

如 2.3 节所述，网络可能出现故障或者有其他流竞争，所以需要动态地对 VP 的上限带宽进行探测。发送端对于第 i 条 VP 的上限带宽 B_i 的更新有如下 3 种方式。

1) 初始化。B_i 在初始阶段被初始化为链路的物理带宽，利用该值初始化能够减少选路稳定的时间。链路的物理带宽可由网络管理员手动输入。

2) ECN 重置。当一条个路径收到 ECN 标记，则说明该路径当前处于满负载的状态。基于这一点，可以根据当前该路径上的实际发送速率来估计其上限带宽。具体地，当收到接收端发回的数据分组中 E 字段被置位时，则将该路径的上限带宽 B_i 更新为 R_i。

3) 超时探测。收到 ECN 时所探测的上限带宽可能在一段时间之后失效（可能链路物理带宽发生了变化，或者竞争数据流出现/消失）。因此，有必要在收到 ECN 标记的一段时间后重新对该路径的可用带宽进行探测。具体来说，在某条 VP 的上限带宽被 ECN 重置后，发送端会等待一段时间 T_{explore}。若 T_{explore} 内该路径没有再次收到 ECN，则会对该 VP 带宽进行超时探测。发送端会把接下来发出去的 C_{burst}（探测数量）个数据分组作为一组探测分组。这组探测分组不采用可用带宽比例选路，而是均发送到超时探测的路径上。发送端会为每一轮超时探测记录一个唯一的探测轮号，写入这组数据分组的 Burst_ID 字段中。接收端根据该标识记录这组数据分组到达的数量及各分组到达的间隔时间，从而计算出到达速率。随后将该速率写入反馈分组包中的 VP_RBurst_Rate 字段中发回。发送端收到反馈后将 B_i 更新为 VP_RBurst_Rate。

5 实验评价

5.1 实验环境介绍

在 NS2[20]网络仿真平台上实现了 ELAB 机制。为了对比性能，在 NS2 中还实现了目前已有的 2 个基于边缘的动态拥塞感知算法—Clove 和 Hermes。因为 CONGA 等方法需要修改网络硬件，无法满足本文的需求，所以故不做对比。

如无特殊说明，各方法的参数配置如下。ELAB 中 T_{explore} 设置为 200 个 RTT（考虑 ECN 触发指示当前网络状态，在频繁探测和保证数据实时性中均衡，设置 100~500 个 RTT 区间），C_{burst} 设置为 10 个数据分组（为了提高准确度，单次探测的数据量

至少为半个带宽时延积），α 设置为 0.3（使新值较快对计算值造成影响）。Clove 中权重降低比例设置为 0.33。Hermes 中 T_{ecn} 设置为 40%，$T_{\text{RTT\_low}}$ 设置为 20 μs+baseRTT，$T_{\text{RTT\_high}}$ 设置为 2baseRTT。Clove 和 Hermes 参数设置均采用文献[7-8]中的设置值/推荐值。

5.2　专门实验深入分析 ELAB

5.2.1　简单非对称场景

第 2 节中对 ELAB 在图 1 中所示的场景进行评价，从图 2 所示的仿真结果可以看到，ELAB 的网络带宽利用率达到了 87%。

图 7 展示了 ELAB 对 2 条路径路所分配的流量速度比例（路径 1 流速：路径 2 流速）。可以看出，可见 ELAB 的算法能很好地模拟出基于可用带宽的分配。

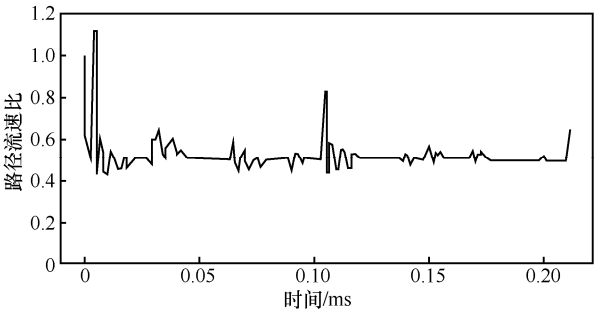


图 7　简单情形下 ELAB 中 2 条路径路所分配的流速比（路径 1 流速:路径 2 流速）

5.2.2　非对称网络下与其他流竞争的场景

除上述简单非对称场景外，本文还进一步设计了非对称网络下与其他流竞争的场景（以下简称竞争场景），来评估 ELAB 的性能。实验网络如图 8 所示。

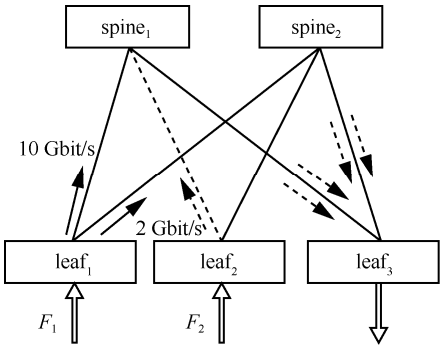


图 8　非对称网络下与其他流竞争的场景

交换机之间的链路正常带宽为 10 Gbit/s。$\text{spine}_1 \leftrightarrow \text{leaf}_2$ 因为故障，降速为 2 Gbit/s。交换机的缓冲队列长度设置为 0.6 MB，ECN 阈值设置为缓存队列长度的 30%。网络中每一跳的时延设置为 20 μs。在竞争场景中，引入 2 个大小皆为 300 MB 的 TCP 流——F_1 和 F_2，分别从 leaf_1 和 leaf_2 进入，去往同一个目的 leaf_3。

图 9 展示了竞争场景下各方法的平均流完成时间（FCT, flow completion time）对比。图中各方法的流完成时间均对基线方法 ECMP 进行了归一化。可以看到，在此场景下，ELAB 的性能依然最佳，相比于 ECMP、Clove 和 Hermes 分别将流完成时间缩短了 31%、24%、3%。

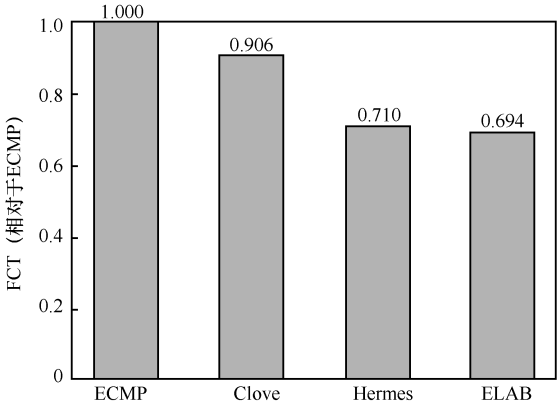


图 9　竞争场景中各方法的平均流完成时间对比

图 10 展现了竞争场景下各方法中 2 条流 F_1 和 F_2 各自的分路比例随时间的变化。其中比例计算方式皆为经 spine_1 发送的流量:经 spine_2 发送的流量。从图 10 可知，ELAB 中 F_2 主要将流量分配至经 spine_2 的好路径，而避开了经 spine_1 的故障路径；同时 F_1 也相应地将主要流量分配至经 spine_1 的路径，从而避免了与 F_2 在 spine_2 上竞争。因此，ELAB 能在此情景下取得较好的平均流完成时间。相反，不论是 Clove 还是 Hermes，从图 10 都能看出对于 2 条路径的分流比例并不合理。其中 F_2 还会频繁地将流量发送至故障路径 spine_1 上，因此流完成时间不佳。

5.3　大规模仿真实验

本文构造了一个大规模 leaf-spine 拓扑网络。如图 11 所示，在该拓扑结构中有 4 个核心层交换机节点和 4 个接入层交换机节点。每个核心层交换机与所有的接入层交换机相连，组成了一个全链接的网络。每个接入层交换机与 16 台底部主机相连，整个网络共有 64 台主机。交换机之间的链路带宽为 40 Gbit/s，交换机与主机之间的链路带宽为 10 Gbit/s。交换机的缓冲队列长度设置为 3 MB，并且 ECN 阈值设置

为缓存队列长度的 30%。网络中每一跳的时延设置为 20μs。本文将利用这种拓扑结构分别评估各方法在对称网络和非对称网络（网络中某些链路出现故障）下的负载均衡性能。

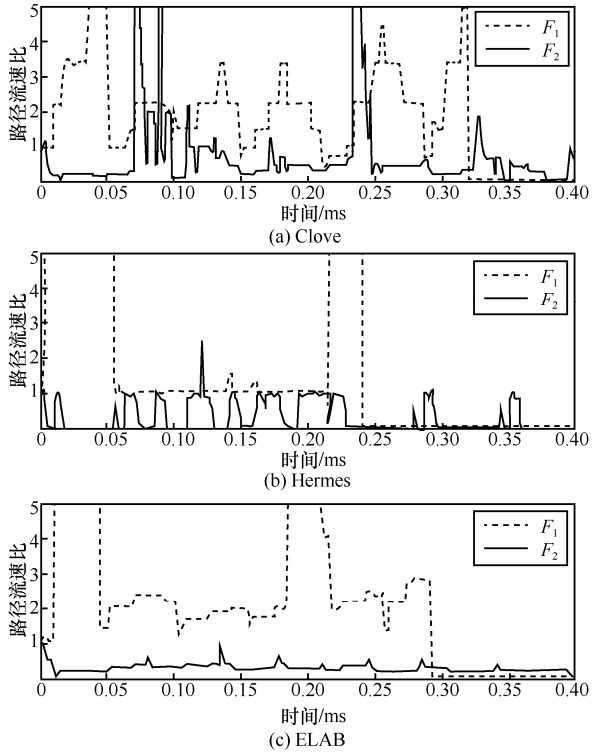


图 10　竞争场景中各方法的路径流速比随时间的变化

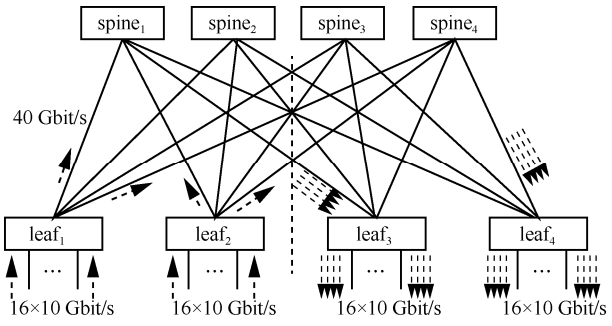


图 11　大规模网络仿真

根据真实数据中心的网页搜索（Web-search）[8]流量记录，在该网络拓扑中生成流量。由于本文研究的是负载平衡对网络性能的影响，因此以图 11 左半部分网络的 32 台主机为每个流量的源主机，在图 11 右半部分网络的不同的位置随机选择每个流量的目的主机，从而使所有流量都经过网络。通过调整节点生成流量的间隔长度来达到不同的网络负载强度。遵循先前工作的惯例，本文使用 FCT 作为主要的性能衡量指标。

5.3.1　对称网络

图 12 展示了大规模网络情形下各机制的平均流完成时间（对 ECMP 进行归一化）。从图 12 可知，无论在轻度负载还是重度负载情况下，ELAB、Clove 和 Hermes 表现都类似，稍微优于 ECMP。ELAB 整体较 Clove、Hermes 有 1%~3%的优势。

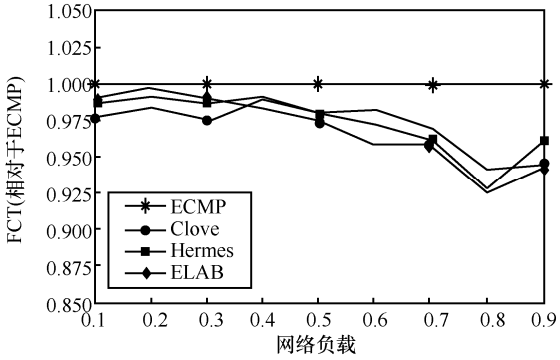


图 12　对称网络中，不同负载下各机制的平均流完成时间（大规模仿真）

5.3.2　非对称网络

将图 11 中 $leaf_2 \leftrightarrow spine_1$ 和 $leaf_2 \leftrightarrow spine_2$ 这 2 条链路的带宽设置为 10 Gbit/s，来模拟网络故障引起链路降速的非对称情形。

图 13 展示了该情形下各机制的平均流完成时间（对 ECMP 进行归一化）。从图 13 可知，在网络负载较轻时，ELAB、Clove 和 Hermes 都表现类似，优于 ECMP。当负载高于 0.5 时，ELAB 开始呈现出明显的优势，平均流完成时间较 Clove 和 Hermes 分别降低了 7%~22%和 17%~26%。

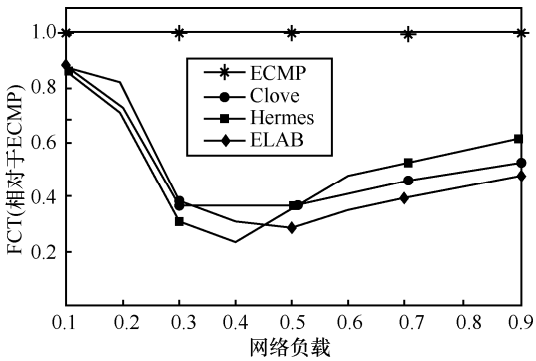


图 13　非对称网络中，不同负载下各机制的平均流完成时间（大规模仿真）

6　结束语

人们对网络带宽需求的日益增高，也对数据中心网路的负载均衡机制提出了更高的要求。当前生

产环境中采用的 ECMP 负载均衡机制选路粒度粗，缺乏动态拥塞感知能力，性能不佳。现有其他细粒度且具备动态拥塞感知能力的方案，或需要对网络硬件进行修改，难以部署；或仅需修改端系统软件，但拥塞感知不准，性能不佳。本文提出的 ELAB 方案是一种基于边缘的利用可用带宽进行拥塞感知的负载均衡机制，既不需要对网络硬件进行改动，又能准确感知拥塞实现良好的负载均衡效果。实验结果表明，在非对称网络情形下，ELAB 能将现有方法的吞吐量提升 10%以上。ELAB 利用可用带宽进行流量分配的思路，解决了现有方法的难点，为数据中心网络的负载均衡机制提供了一种新的思路。

参考文献：

[1] BENSON T, AKELLA A, MALTZ D A. Network traffic characteristics of data centers in the wild[C]//The 10th ACM SIGCOMM Conference on Internet Measurement. 2010: 267-280.

[2] SINGH A, ONG J, AGARWAL A. Jupiter rising: a decade of clos topologies and centralized control in google's datacenter network[J]. ACM SIGCOMM Computer Communication Review, 2015, 45(4): 183-197.

[3] HOPPS C E. Analysis of an equal-cost multi-path algorithm[R]. Request for Comments 2992, 2000.

[4] KATTA N, HIRA M, KIM C. Hula: scalable load balancing using programmable data planes[C]//The Symposium on SDN Research. 2016: 10.

[5] GHORBANI S, GODFREY B, GANJALI Y. Micro load balancing in data centers with DRILL[C]//The 14th ACM Workshop on Hot Topics in Networks. ACM, 2015: 17.

[6] ALIZADEH M, EDSALL T, DHARMAPURIKAR S. CONGA: distributed congestion-aware load balancing for datacenters[J]. ACM SIGCOMM Computer Communication Review, 2014, 44(4): 503-514.

[7] KATTA N, HIRA M, GHAG A. CLOVE: how I learned to stop worrying about the core and love the edge[C]//The 15th ACM Workshop on Hot Topics in Networks. 2016: 155-161.

[8] ZHANG H, ZHANG J, BAI W. Resilient datacenter load balancing in the wild[C]//The Conference of the ACM Special Interest Group on Data Communication. 2017: 253-266.

[9] VANINI E, PAN R, ALIZADEH M. Let it flow: resilient asymmetric load balancing with flowlet switching[C]//USENIX Symposium on Networked Systems Design and Implementation. 2017: 407-420.

[10] HE K, ROZNER E, AGARWAL K, et al. Presto: edge-based load balancing for fast datacenter networks[J]. ACM SIGCOMM Computer Communication Review, 2015, 45(4): 465-478.

[11] CAO J, XIA R, YANG P. Per-packet load-balanced, low-latency routing for clos-based data center networks[C]//The 9th ACM Conference on Emerging Networking Experiments and Technologies. 2013: 49-60.

[12] DITTMANN G, HERKERSDORF A. Network processor load balancing for high-speed links[C]//The 2002 International Symposium on Performance Evaluation of Computer and Telecommunication Systems. 2002, 735.

[13] AL-FARES M, RADHAKRISHNAN S, RAGHAVAN B, et al. Hedera: Dynamic flow scheduling for data center networks[J]. USENIX Symposium on Networked Systems Design and Implementation, 2010, 10: 19.

[14] BENSON T, ANAND A, AKELLA A. MicroTE: fine grained traffic engineering for data centers[C]//The 17th Conference on emerging Networking Experiments and Technologies. 2011: 8.

[15] CURTIS A R, KIM W, YALAGANDULA P. Mahout: low-overhead datacenter traffic management using end-host-based elephant detection[C]//INFOCOM. 2011: 1629-1637.

[16] PERRY J, BALAKRISHNAN H, SHAH D. Flowtune: flowlet control for datacenter networks[C]//USENIX Symposium on Networked Systems Design and Implementation. 2017: 421-435.

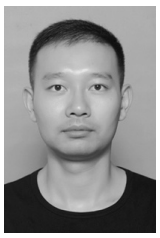
[17] MAHALINGAM M, DUTT D, DUDA K. Virtual extensible local area network (VXLAN): a framework for overlaying virtualized layer 2 networks over layer 3 networks[R]. Request for Comments, 7348,

[18] KIM Y J, KOLESNIKOV V, KIM H. SSTP: a scalable and secure transport protocol for smart grid data collection[C]//2011 IEEE International Conference on Smart Grid Communications (SmartGridComm). 2011: 161-166.

[19] AUGUSTIN B, CUVELLIER X, ORGOGOZO B, et al. Avoiding traceroute anomalies with Paris traceroute[C]//The 6th ACM SIGCOMM conference on Internet measurement. 2006: 153-158.

[20] ISSARIYAKUL T, HOSSAIN E. Introduction to network simulator NS2[M]. Springer Science & Business Media, 2011.

[作者简介]



陈果（1989– ），男，湖南长沙人，博士，湖南大学副教授，主要研究方向为计算机网络系统、数据中心网络。

张潍丰（1997– ），男，广东新会人，主要研究方向为计算机网络系统、计算机系统结构。