基于广域无损网络的QoS参数自适应优化技术应用研究

Application Research of QoS Parameter Automatic Optimization Technology Based on Wide Area Lossless Network

杨锋/Yang Feng[1]，尹远阳/Yin Yuanyang[2]，
王江龙/Wang Jianglong[1]，杨顺兴/Yang Shunxing[2]
（1. 中国电信集团有限公司，中国 北京 100035；
2. 中国电信股份有限公司研究院，中国 广州 510630）
(1. China Telecom Group Corporation, BeiJing 100035, China;
2. Research Institute of China Telecom Co. ,Ltd. , Guangzhou 510630, China)

DOI：10.12142/ZTETJ.202605011
网络出版地址：https://link.cnki.net/urlid/34.1228.TN.20241012.1529.002
网络出版日期：2024-10-12
收稿日期：2024-09-10

摘要：随着生成式人工智能（AIGC）大模型计算应发展，当前广域网基于DiffServ模型的主动队列管理（AQM）技术，难以满足业务在高吞吐、低时延以及低抖动下的不同流量、多样化的业务场景需求。研究了广域网络流量的场景分类技术，并结合AQM技术，提出了一种广域服务质量（QoS）参数自适应优化方法，实现跨数据中心的无损转发，优化吞吐量和时延性能。

关键词：拥塞控制；主动队列管理；服务质量；概要统计

Abstract: With the development of generative artificial intelligence (AIGC) large model computing, the active queue management (AQM) technology based on the DiffServ model in current wide area networks can hardly meet the requirements of services for different traffic and diverse business scenarios under high throughput, low latency, and low jitter. The traffic scenario classification technology for wide area networks is studied, and combined with AQM technology, an adaptive optimization method for wide area quality of service (QoS) parameters is proposed to achieve lossless forwarding across data centers and optimize throughput and latency performance.

Keywords: congestion control; active queue management; quality of service; summary statistics

引用格式：杨锋, 尹远阳, 王江龙, 等. 基于广域无损网络的QoS参数自适应优化技术应用研究 [J]. 中兴通讯技术, 2026, 32(5): 82-90. DOI: 10.12142/ZTETJ.202605011

Citation: Yang F, Yin Y Y, Wang J L, et al. Application research of QoS parameter automatic optimization technology based on wide area lossless network [J]. ZTE technology journal, 2026, 32(5): 82-90. DOI: 10.12142/ZTETJ.202605011

随着网络条件和应用需求的不断演进，针对服务质量（QoS）的相关研究如雨后春笋般持续涌现，40年来仍然经久不衰，每年都有大量研究成果发表[1]。而随着云计算和数据中心的崛起，数据中心网络（DCN）需要为海量数据的快速存储和高效处理提供强有力的保障。为了应对数据中心网络巨大的流量需求、日益扩大的网络规模以及时延敏感业务与高吞吐业务的综合承载需求，业界推出了各种流量控制方案[2]。这些无损技术提高了数据中心的数据吞吐量，降低了网络时延。

近年来，随着分布式训练和高通量传输需求的不断增长[3]，如何提升广域网络性能以满足新的业务需求，已成为业界研究的热点。其中，广域无损技术成为一个重要的技术方向。广域无损技术是数据中心无损网络在广域网的延伸，主要目标是减少丢包，满足高吞吐业务的转发需求，优化吞吐与时延性能。然而，直接将数据中心的无损技术应用于广域网，将会带来性能问题。

数据中心无损网络技术依赖于动态基于优先级的流量控制（PFC）、数据中心量化拥塞通知（DCQCN）以及动态显式拥塞通知（ECN）等技术来实现无损流量转发。然而，在广域网络中，由于链路长度通常达到数十千米，导致时延较大，这限制了PFC和ECN等流控技术的有效性，使其在广域网环境下难以充分发挥作用。其中，PFC的操作对象是端口优先级队列，粒度较粗，因此在广域网中部署PFC比在数据中心更容易引发PFC风暴，进而导致PFC不公平现象，损

害无辜的数据流。对于DCQCN而言，由于广域网中的中继节点数量远超数据中心，非数据中心流量的竞争加剧了触发ECN的概率，使得遵循DCQCN算法的发送端不得不保持较低的发送速率，从而无法有效提升数据中心流量的吞吐性能。

1 无损新技术研究介绍

当前广域网主要采用基于区分服务（DiffServ）模型的主动队列管理（AQM）技术[1]。在AQM技术基础上，业界提出了一系列AQM改进算法和新技术，主要策略是控制拥塞、联动端侧降速、减少数据传输过程中的丢包。

广域无损的目标并非绝对零丢包，而是尽量减少丢包，其技术方向包括端侧优化和网络侧优化：1）端侧传输层协议优化，通过改进传输控制协议（TCP），以提供更高的性能和可靠性。但是，无论TCP如何在算法上进行优化，其拥塞控制算法的核心都是尽量占用带宽，如瓶颈带宽与往返传播时间（BBR）协议不断探寻最小往返时延（RTT），实现最大传输速率，最终还是会抢占链路带宽，导致拥塞丢包，并不能真正实现广域无损的目标。2）网络侧AQM技术，如随机早期检测（RED）[2]、加权随机早期检测（WRED）[3]以及尾部丢弃（TD）等，在广域路由器上主动控制丢包策略，其本质是在拥塞产生之前提前进行检测识别，通过早期丢包触发端侧降速，避免拥塞恶化[4–7]。

在网络侧AQM技术中，RED算法得到了广泛部署。如图1所示，RED算法模型通过设定队列长度的最小门限和最大门限来控制丢包行为。每当分组数据包到达时，算法会测量当前队列的平均深度，并将其与预设的高、低门限进行比较，以决定是否需要丢弃数据包。若平均队列深度小于低门限，则将新到达的分组数据包放入队列进行排队；若超过高门限，则丢弃；若在低门限和高门限之间，则按照某一丢弃概率丢弃。这种丢包机制使得网络侧拥塞能够被端侧感知，进而通过TCP协议的速率调整机制实现拥塞避免。

尽管RED算法得到了广泛应用，但其实际效果并不尽如人意，主要原因在于RED算法模型涉及参数较多，例如低门限、高门限等，这些静态配置的参数难以适应多变的流量场景。此外，它们与端侧的TCP等传输协议之间的协同作用不足，导致难以有效提升广域网传输质量。

基于RED技术，自适应随机早期检测[8]、流随机早期检测（FRED）[9]以及控制延时（CoDel）[10]等新技术应运而生。这些技术都对RED做了不同的算法改进，但也只在特定业务流量模型下表现较好，在广域路由器上并没有得到广泛应用。

自适应调整门限参数（ARED）方法虽然解决了RED的参数敏感性问题，但也带来了新的参数设置问题。α、β设置过大或过小会导致maxp多次震荡后再达到期望值，不利于网络性能稳定。FRED相对于RED在流的公平性上做出了重大改进，但该算法需要维护每个流的信息，计算量较大，增加了路由器的负载，很难实际部署。CoDel算法的优化重

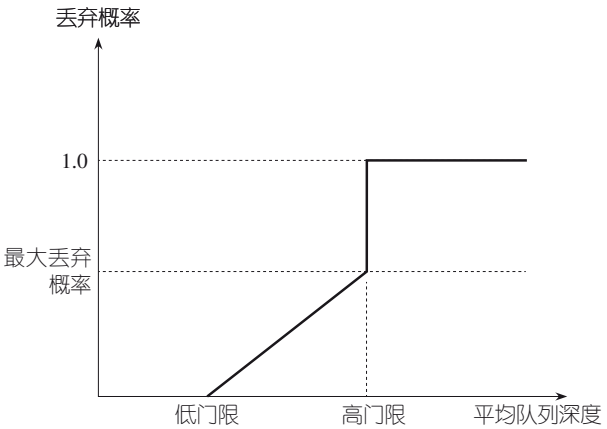


图1　随机早期检测算法模型

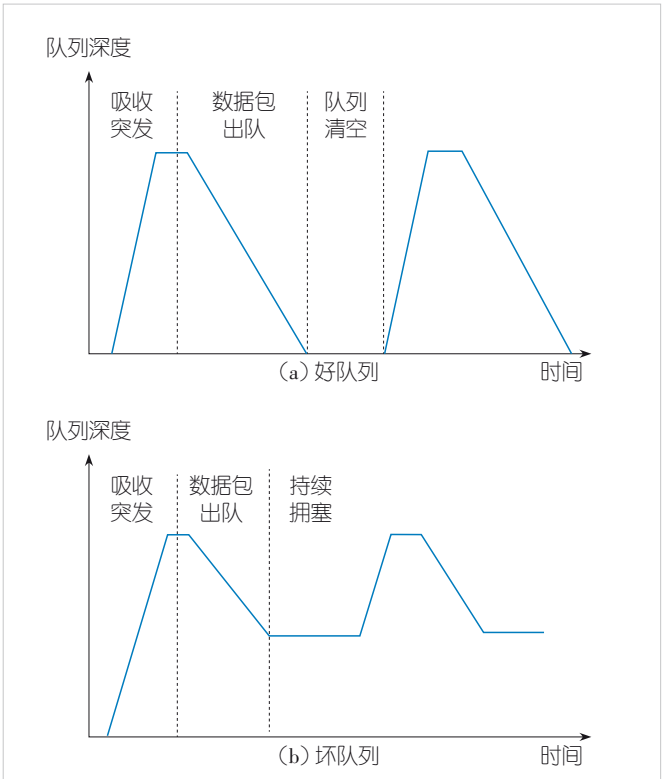


图2　CoDel算法的好坏队列运行曲线

点是降低时延，研究者为此提出了好队列与坏队列的概念。如图2所示，CoDel算法将频繁清空、数据包停留时间接近于零的队列视为运行良好的队列；相反，拥塞队列会延迟每个数据包，并且最小停留时间永远不会接近于零。CoDel通过监控目标报文的停留时间来决定是否丢弃报文。算法使用5 ms作为目标停留时间，如果超过该停留时间，就会丢弃报文以控制队列深度。但实际上，目标时间数值的选择需要大量实验和模拟，很难适应复杂场景，尤其是在大量突发场景下，由于丢包严重，会导致吞吐损失较大。

广域网和数据中心网络的差异主要体现在时延增大，这导致反馈时间较长；同时，网络流量复杂、瓶颈链路多变，传统的数据中心拥塞算法以及缓存不能很好地适应广域网场景[11-12]。业界还提出了多种改进方案，如源端流控（SFC）[13-14]、选择性优先级流控（SPFC）[15]、健壮的拥塞控制（RoCC）[16]等。但这些方案依赖于流量特征，或者仅针对长稳流进行优化。精细刻画流量特征是众多方案的基础，但相关文献都没有就此展开研究。综合以上分析可知，当前各种AQM技术和改进方案在应对广域无损问题上都有不足，无法自适应不同流量场景，只能在特定场景下取得较好的效果[17-18]。

2 广域网QoS参数自适应优化技术

本文提出的QoS参数自动优化技术主要由流量建模与QoS参数自动优化两个核心功能组成，其中流量建模是QoS参数自动优化的前置条件。

流量建模通过对QoS队列流量长短期波动的检测与学习，建立流量基线，实现如下统计功能：

1）大小流分布：实现队列大小流分布与占比统计。

2）微突发的检测与预测：输出队列微突发的最大规模及持续时间。

3）时延检测与预测：检测并学习队列时延变化，包括最差时延与平均时延。

QoS自适应优化架构如图3所示，包括流量建模和QoS参数化自动优化，流量模型作为输入参数用于自动优化QoS参数。通过启动QoS参数自适应优化过程，系统在多个学习周期内收集队列的实时统计信息。基于这些信息，系统动态调整QoS组合参数，如RED算法的参数，包括缓存的低门限、高门限和丢弃概率等，以提升吞吐量并降低时延。执行QoS参数自动优化动作后，结果会反馈至流量建模，以便进行下一轮优化调优[14]。经过多轮调优后，如果队列吞吐和时延性能达到最优值，则终止优化。

2.1 大小流分布算法

在广域网上，大流占总流的数量比例较小，大部分都是小流。大流的特点是持续时间较长，占用带宽较多，对吞吐敏感；而小流持续时间短，实时性要求高，对时延敏感。现有广域网AQM技术并没有针对队列中的大小流进行优化处理，静态配置的QoS参数无法同时满足大流高吞吐和小流低时延的需求。

Sketch是一种概要数据结构，基于Hash数据结构实现，是记录活跃流统计信息的有效方法，适用于流量测量、流量分类、攻击检测等多种应用领域。Sketch存在多种结构，包括基础的Count-min Sketch[19]以及各种变体[20-21]。本文大小流检测方案对Count-min Sketch结构进行改进，只需进行多次Hash表的counter计数统计与数值比较，就可以完成大小流的检测。

大小流分布算法是一种基于Sketch概要统计与基数估计相结合的大小流检测算法，能够在占用更少芯片计算与存储资源的情况下实现有效的大小流检测。大小流统计Sketch架构如图4所示，共有N行哈希桶。当报文到达后，并行计算N个哈希值，每个哈希值对应的counter计数增加；计算N个

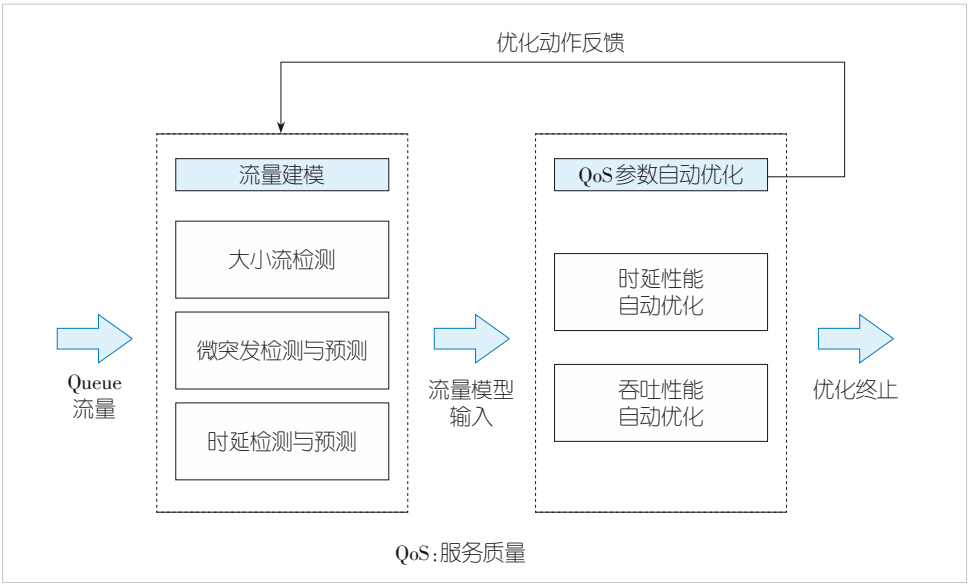


图3 QoS自适应优化流程架构

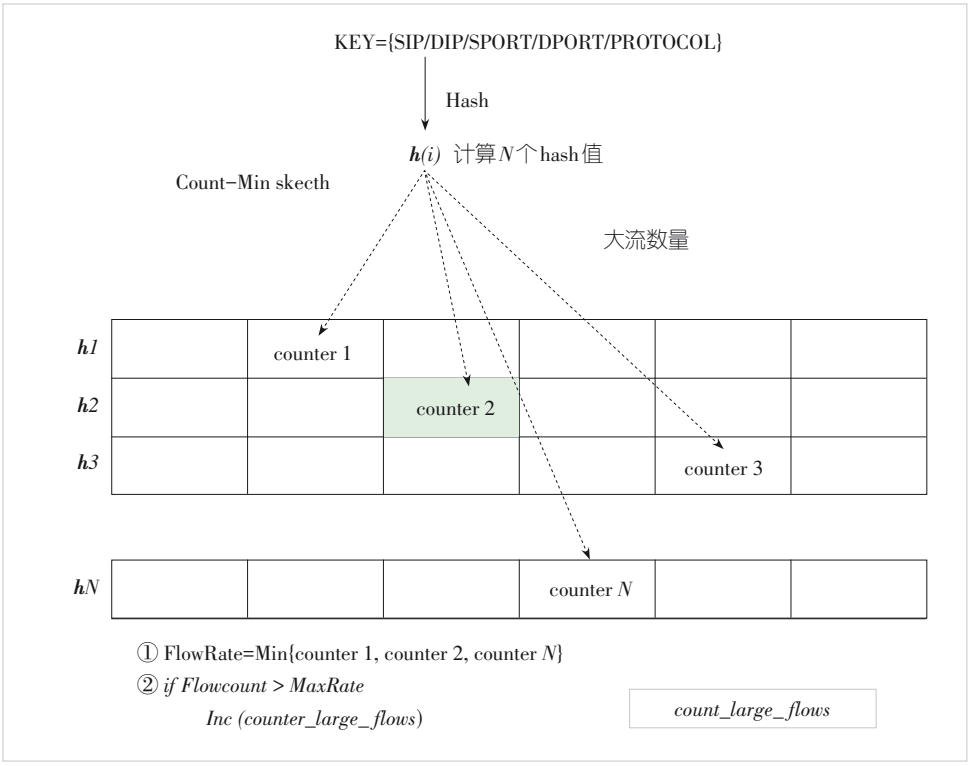


图4 Sketch大小流统计架构

counter中的最小值，如果最小值超出大流的速率阈值MaxRate，则确定为大流，同时在转发面增加大流统计数count_large_flows。

为了使counter计数能够反映一定周期内的报文速率，实现对大流的稳定测量，需要由软件定时读清大流的counter基数。队列大小流的统计决定了队列QoS参数自适应优化的目标：如果大流占比较高，优化的主要目标就是提升吞吐性能；如果小流占比较高，优化的主要目标则是降低时延。

2.2 微突发与时延测量算法

广域网络存在大量微突发。在较大的观测时间窗口上（如1 s以上），网络整体平均流量相对平稳，基本观测不到微突发；但在较小的时间粒度上，瞬时速率远超平均速率。1 ms级的微突发最大可达100 ms级的十几倍以上，10 ms级微突发大致是100 ms级微突发的几倍，500 ms的流量则接近1 s的平均流量。

微突发产生的主要原因与TCP等传输协议有关。TCP的间断式连续发包会造成短时突发，即周期性的突发波动流量。此外，在广域组网上，并发多打一以及高速带宽向低速带宽发送也是微突发的常见原因。在缓存buffer不足的情况下，微突发可能导致大量数据丢包。这种丢包现象会触发所有端侧传输协议同步减小发送窗口，进而导致链路利用率持续降低，最终造成总体吞吐量降低。

为了应对微突发问题，常见的方法是增大队列缓存大小，因为更大的缓存能够吸收更多突发流量。但芯片的缓存资源有限，不能无限制地扩大缓存容量。此外，较大缓存不利于小流的低时延转发。因此，仅通过调整缓存大小很难同时满足大流高吞吐和小流低时延的性能要求。

QoS参数自适应优化技术的微突发检测方案在转发面嵌入微秒级定时机制，周期性地统计队列缓存与队列实际调度能力，输出队列微突发最大规模以及最大时延。转发面每100 μs进行一次队列突发报文字节数统计，最大缓存100 ms的历史数据；同时每100 ms进行一次突发缓存数据读取，汇总后计算出100 ms期间的最大积压buffer。结合检测到的队列实际调度能力，计算出最大微突发buffer以及最大时延，具体算法方案如图5所示。

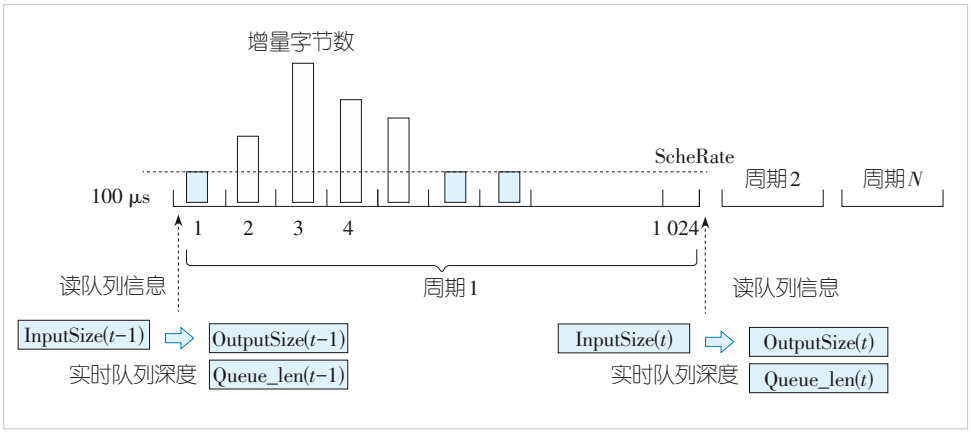


图5 队列微突发检测模型方案

ScheRate为预测的队列实际调度速率，该速率采用两次周期读取出队列字节数的差值除以周期进行计算，公式如公式（1）：

$$\mathrm{ScheRate}(t) = (\mathrm{OutputSize}(t) - \mathrm{OutputSize}(t-1))/\mathrm{Period} \quad (1),$$

其中，OutputSize(t)是t时刻出队列的字节数，Period是周期时长。

算法模型采用循环遍历计算每个时间片的累积最大buffer，计算公式为：

$$\mathrm{CumBuffer}(t) = \mathrm{CumBuffer}(t-1) + \mathrm{BurstSize}(t) - \mathrm{ScheSize}(t) \quad (2),$$

其中，CumBuffer为当前时间片t的最大累积buffer；BurstSzie为当前时间片的突发统计字节数；ScheSize为当前时间片的可调度字节数，由ScheRate计算得出。

当完成所有时间片的CumBuffer计算后，就可以得到最大累积缓存MaxCumBuffer，最差时延MaxDelay，计算公式为：

$$\mathrm{MaxDelay} = \mathrm{MaxCumBuffer}/\mathrm{ScheRate} \quad (3)。$$

2.3 QoS参数自动寻优算法

本文提出服务质量（QoS）参数自动调整技术，并通过参数寻优算法实现，算法模型主要采用启发式算法。结合流量场景，如微突发情况和大小流分类情况，确定参数优化的方向，主要包括如下3种优化方向：

1）丢包性能优化：主要针对微突发丢包场景。如果当前队列配置的是尾部丢弃（TD）模型，则只进行buffer缓存大小优化，综合多个检测周期的最大微突发大小自动修改队列缓存大小，实现微突发情况下零丢包，避免对队列造成吞吐损失。

2）吞吐性能优化：对于大流占比高的场景，在当前吞吐未达到最优门限前启动吞吐优化，采用启发式算法多轮迭代探寻最优QoS参数，包括缓存高门限、低门限以及丢弃概率等。本文主要基于低门限进行动作优化。经过实验对比，低门限相对高门限以及丢弃概率效果最优。

3）时延性能优化：对于小流占比高的场景，启动时延性能优化。对于大流，通常会在完成吞吐性能优化之后也进行时延优化，即在优先保证吞吐性能的前提下进一步降低时延。

对应的算法整体流程如图6所示，包括输入、动作、输出等多个环节。

（1）输入：包括多周期最大时延、最大缓存积压、大小流占比。如果大流占比较高，例如超过30%（实际广域网在不同场景中大小流占比也各不相同），则可认为持续的buffer积压由大流引起。因此，调低MinThresh低门限，通过早丢包控制大流的发送速率，可以减少积压buffer，降低时延。另外，如果小流占比相对较高，则无论是否降低门限都存在一定风险，会有较大概率丢弃小包；丢包则会造成小流停止发包，导致时延恶化。在这种情况下，可以将大小流分流，并降低大流的QoS调度优先级，优先保证小流的时延性能。

（2）动作：主要基于WRED低门限MinThresh进行动态调整，设置调整步长step = 5%。其中，MinBuffer为队列允许的最小buffer，用于避免大包在不稳定情况下丢包。MinThresh计算公式为：

$$\mathrm{MinThresh}(t) = \mathrm{MAX}\{\mathrm{MinBuffer}, \mathrm{MinThresh}(t-1) \times (1 - step)\} \quad (4)。$$

（3）反馈：下一个周期的最大时延、最大缓存积压buffer以及大小流占比。完成一次动作后，等待下一周期的模型输入；当输入满足终止条件，即达到需要优化的时延目标时，则停止策略动作。

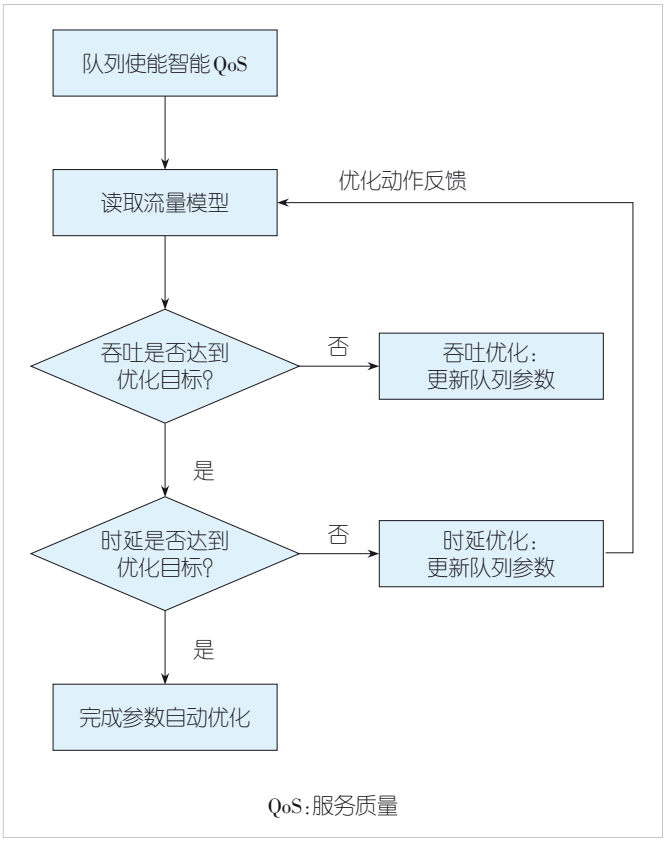


图6 QoS参数自动寻优算法

3 测试与验证

3.1 大小流检测算法测试

基于运营骨干网流量数据，我们选择一个时间片段内的流量数据进行大小流检测算法验证，如表1所示。Range表示该时间片内的报文个数，entries表示流的数目。总体上，小流数量占比较高，大流数量占比较少，但大流占用带宽较大。

我们采用如下混淆矩阵作为数据指标，对算法进行有效性验证：

1）准确率：（TP + TN）/（TP + FP + FN + TN）

2）精确率：TP /（TP + FP）

3）召回率：TP/（TP + FN）

4）F1分数：2 ×（Precision × Recall）/（Precision + Recall）

表1 大小流流量模型参数表

特征	数值
流总带宽	2 Mbit/s
流总条数	363 512
大流数量	5 658
流大小分布	0～9：335 027条 10～99：25 988条 100～999：2 397条 1 000～4 999：99条 5 000～9 999：1条 10 000+：0条

其中，假阳性（FP）表示模型将一个实际为负类别的样本错误地分类为正类别，即模型假阳性错误。假阴性（FN）表示模型将一个实际为正类别的样本错误地分类为负类别，即模型假阴性错误。真阳性（TP）表示模型将一个实际为正类别的样本正确地分类为正类别。真阴性（TN）表示模型将一个实际为负类别的样本正确地分类为负类别。

在测试验证中，我们重点关注大流检测精确率。我们对Sketch哈希桶数以及哈希表大小选择不同的参数组合进行验证，分别得到不同哈希表数精确度指标数据和不同哈希桶数精确度指标数据。

如表2所示，本文中我们固定哈希桶数，设置不同的哈希表大小进行对比验证。可以看出，哈希表规模越大，大流识别的精确度指标越高。表3中，我们固定哈希表大小，设置不同的哈希桶数进行对比验证。可以看出，哈希桶数越大，大流识别的精确率指标越高。但相比之下，哈希表大小对大流识别精确度指标的影响更大。路由器产品芯片转发面资源有限，因此需要结合芯片资源情况，选择最优的Sketch哈希表大小与哈希桶数参数组合。

3.2 队列参数自动寻优算法测试

我们对微突发以及队列参数自动寻优算法进行组网测试验证，测试组网如图7所示。测试仪支持4—7层协议，通过10GE链路连接2台路由器。测试仪PORT-1作为应用服务的Server端，开启超文本传输协议（HTTP）、文件传输协议（FTP）、视频（VIDEO）混合流；PORT-2作为Client端，不

表2 固定哈希桶对应不同哈希表数精确度指标

哈希表大小	哈希桶数	TP	FN	TN	FP	召回率/%	精确率	准确率	F1
1 024	3	100	0	5 691	357 721	100	0.002 795	0.015 931	0.000 559
2 048	3	100	0	349 169	14 243	100	0.006 972	0.960 818	0.013 848
4 096	3	100	0	363 232	180	100	0.357 143	0.999 505	0.526 316
8 192	3	100	0	363 383	29	100	0.775 194	0.999 92	0.873 362
16 384	3	100	0	363 397	15	100	0.869 565	0.999 959	0.930 223
32 768	3	100	0	363 397	15	100	0.869 565	0.999 959	0.930 233

FN：假阴性　TN：真阴性　TP：真阳性

表3 固定哈希表对应不同哈希桶数精确度指标

哈希表大小	哈希桶数	TP	FN	TN	FP	召回率/%	精确率	准确率	F1
4 096	1	100	0	340 001	23 411	100	0.004 253	0.935 598	0.008 647
4 096	2	100	0	361 792	1 620	100	0.058 140	0.995 543	0.109 89
4 096	3	100	0	363 232	180	100	0.357 143	0.999 505	0.526 316
4 096	4	100	0	363 322	90	100	0.526 316	0.999 752	0.689 655
4 096	5	100	0	363 339	73	100	0.578 035	0.999 799	0.732 601

FN：假阴性　TN：真阴性　TP：真阳性

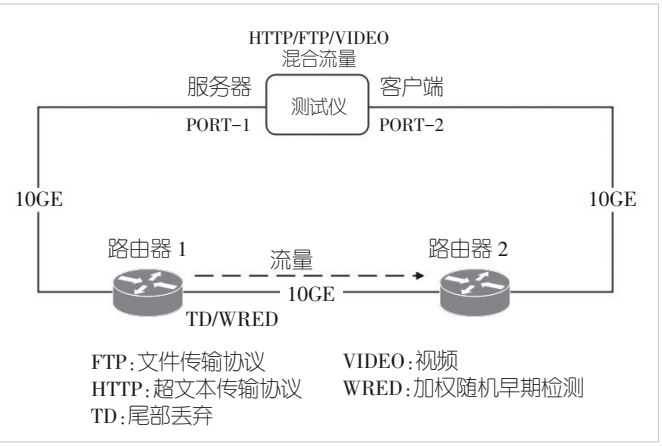


图7　面向微突发场景的队列参数自动寻优算法组网测试拓扑

同会话通过发送字节数区分大小流。在路由器Router-1到Router-2的出向链路上配置限速，构造拥塞场景，验证TD队列丢包优化以及RED队列吞吐与时延优化。

TD队列缓存优化算法的测试对比如表4所示。队列初始缓存大小设置为128 kB，出口限速为1 Gbit/s。在所有应用服务开启并达到流量相对稳定后，未优化前统计的平均队列深度约为120 kB，小于最大缓存128 kB，但微突发大小已经超过最大缓存128 kB，此时检测到大量微突发丢包。

启动微突发丢包优化后，流量建模预测的微突发最大值约为240 kB。经过3次迭代，算法最终收敛至250 kB的最优缓存配置，对微突发最大缓存的预测误差小于5%。此外，当缓存得到优化后，可以观测到出向流量吞吐同时得到提升。这是由于减少丢包避免了TCP波动带来的流量损失。

RED队列未启动QoS参数自动寻优算法时，系统初始设置低门限为64 kB，高门限为1 MB。应用服务全部开启且流量相对稳定后，队列统计数据如表5所示。队列持续丢包，

表4　队列缓存优化对比数据

	缓存大小/kB	队列丢包数	队列平均深度/kB	出口平均速率/（Mbit•s^{-1}）
未优化前	128	787k	120	821
第1次迭代	240	758	143	943
第2次迭代	245	41	152	951
第3次迭代	250	0	151	963

表5　RED队列未优化数据

	入队列速率/（Mbit•s^{-1}）	出队列速率/（Mbit•s^{-1}）	队列深度/kB	丢包数/k
最大值	1 211	1 000	165	2 862
均值	886	761	83	787

吞吐始终在70%～80%之间波动，无法达到1 GE的最大链路带宽。

启动QoS参数自动寻优算法后，系统通过多次迭代逐渐提升低门限，优化吞吐与时延性能。图8和图9分别展示吞吐与时延随着低门限的变化曲线。从图8中可以得出，随着低门限增加，微突发导致的丢包减少，吞吐性能上升；当低门限接近最大缓存时，吞吐达到最大。从图9中可以得出，随着低门限增加，时延性能下降；当达到最大吞吐后，低门限回调到512 kB时，可以在90%的吞吐情况下取得相对最优的时延，约为600 μs。

图10展示动态寻优过程中各队列参数随时间的变化情况，横坐标表示时间，数据采样间隔为10 ms；纵坐标自上而下依次为入队列速率in_rate、出队列速率out_rate（单位为Gbit/s），入队与出队字节数差值ΔS、队列深度Queue Depth、丢包字节数Drop Bytes（单位为MB），以及队列低门

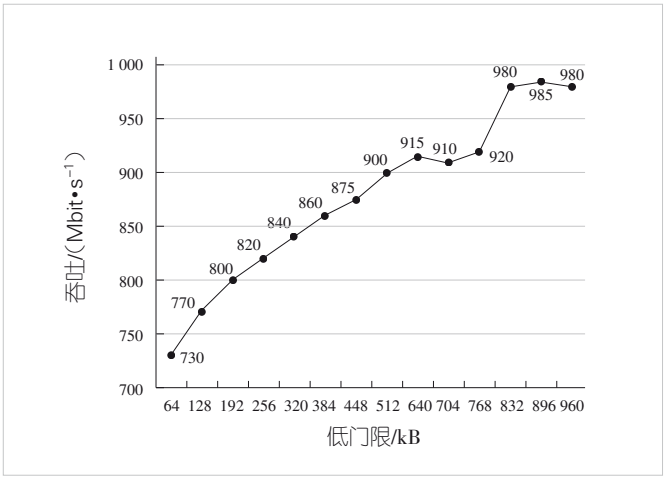


图8　低门限动态优化过程吞吐变化曲线

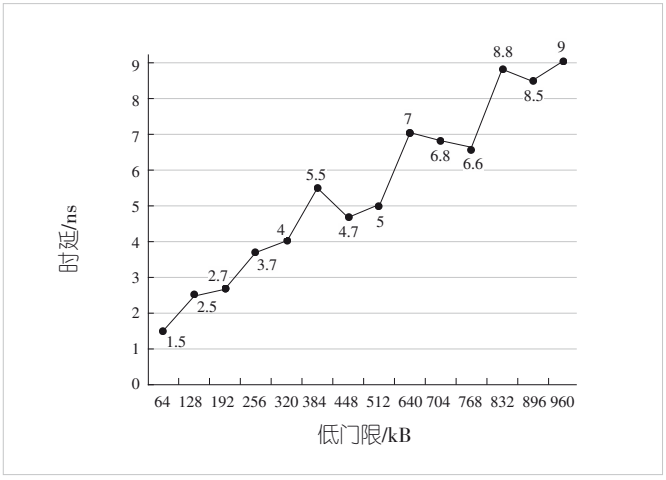


图9　低门限动态优化过程时延变化曲线

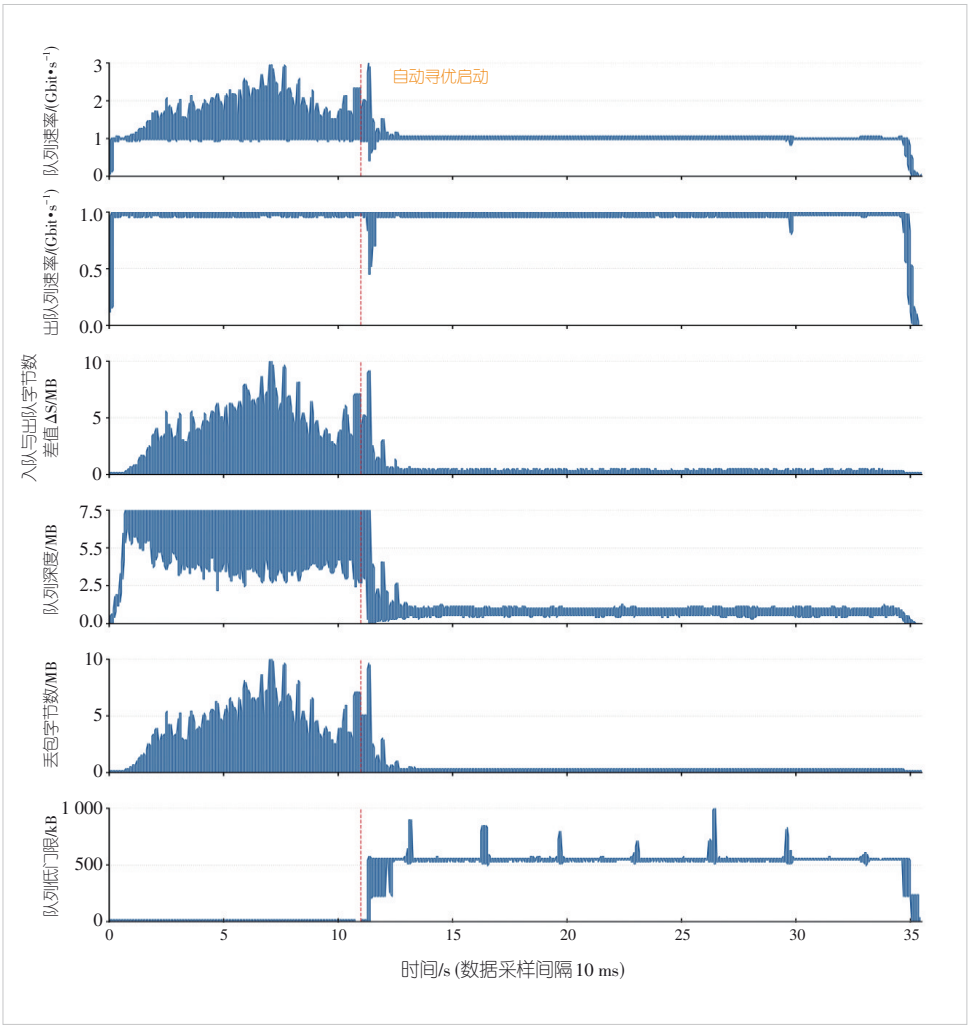


图10　动态寻优过程中各队列参数随时间的变化

限 Min Threshold（单位为kB）。可以看到，RED参数自动优化在12 s启动，并在1～2 s左右完成优化。TCP端侧显著降速，入向速率与出向速率达到相对平衡，队列丢包保持在较低水平，吞吐则保持在最高水平，队列深度可以保持在较低水平，时延性能同时得到优化。

4 结束语

本文提出了一种针对广域网的QoS参数自适应优化技术，并在路由器产品中进行了测试验证。该技术通过流量建模实现精确的场景分类，能够准确检测出流量微突发的规模、最大时延与平均时延，并通过Sketch概要统计结构有效实现大小流检测。相对于数学公式化的参数拟合和静态参数配置，本文采用的启发式算法能够适应不同流量场景，并可在多次迭代后收敛到最优的QoS参数。针对TD队列中的微突发丢包问题，自适应优化通过设置零丢包的最小缓存，有效减少了缓存资源占用并避免了突发丢包。对于RED队列，自适应优化QoS参数能够在提升广域网流量吞吐性能的同时，降低平均队列深度，进而优化时延性能。

参考文献

[1] Jain R. The catch-up game quest for the impact [EB/OL]. (2017-08-22) [2026-08-02]. https://conferences. sigcomm. org/sigcomm/2017/files/program/keynote-1.pdf
[2] 杜鑫乐, 徐恪, 李彤. 数据中心网络的流量控制：研究现状与趋势 [EB/OL]. [2026-08-02]. https://lib. cqvip. com/Qikan/Article/Detail? id=7105063887
[3] King D, Chown T, Rapier C, et al. Current state of the art for high performance wide area networks [EB/OL]. (2025-03-01) [2026-08-04]. https://www. ietf. org/archive/id/draft-kcrh-state-of-art-hp-wan-00.txt
[4] FLOYD S. A review of FRED: flow random early detection [J]. ACM SIGCOMM Computer Communication Review, 2000, 30 (4): 62-73
[5] RFC 2309 Recommendations on queue management and congestion avoidance in the internet [S]
[6] Wurtzler M. Analysis and simulation of weighted random early detection (WRED) queues [EB/OL]. [2026-08-05]. https://www.ittc.ku.edu/research/thesis/documents/mark_wurtzler_thesis.pdf
[7] Hassan S, Rufai A. Modified dropping-random early detection (MD-RED): a modified algorithm for controlling network congestion [J]. International journal of information technology, 2023, 15(3): 1499-1508. DOI: 10.1007/s41870-023-01201-1
[8] 任金霞, 蒋梦倩, 温春晖. 基于S型函数动态自适应改进的RED算法 [J]. 高技术通讯(中文), 2019, 29(5): 449-454. DOI: 10.3772/j.issn.1002-0470.2019.05.005
[9] Patel S. Sensitivity analysis of queue-based AQM over network parameters [J]. IET networks, 2019, 8(4): 280-288. DOI: 10.1049/iet-net.2018.5160
[10] Patel S. Nonlinear performance evaluation model for throughput of AQM scheme using full factorial design approach [J]. International journal of communication systems, 2020, 33(8): e4357. DOI: 10.1002/dac.4357
[11] Floyd S, Gummadi R, Shenker, S. Adaptive RED: an algorithm for increasing the robustness of RED´s active queue management [EB/OL]. (2001-08-01)[2026-08-03]. https://www.icir.org/floyd/papers/adaptiveRed.pdf
[12] Yan F Y , Ma J, Hil G D, et al. Pantheon: the training ground for Internet congestion-control research [EB/OL]. [2026-08-03].

https://pantheon. stanford. edu/static/pantheon/documents/pantheon-paper.pdf
[13] Le Y, Lee J, Blendin J, et al. SFC: Near-source congestion signaling and flow control [PP/OL]. arXiv[2026-08-02]. https://arxiv.org/pdf/2305.00538
[14] Huang H L, Du X L, Li T, et al. Re-architecting buffer management in lossless Ethernet [J]. ACM transactions on networking, 2024, 32(6): 4749-4764. DOI: 10.1109/tnet.2024.3430989
[15] Taheri P, Menikkumbura D, Vanini E, et al. RoCC: robust congestion control for RDMA [C]//Proceedings of the 16th International Conference on Emerging Networking EXperiments and Technologies. ACM, 2020: 17-30. DOI: 10.1145/3386367.3431316
[16] Nichols K, JacobsonV. Controlling queue delay [J]. ACM Queue, 2012, 10(5): 68-81.DOI: 10.1145/2209249.2209264
[17] Cormode G, Muthukrishnan S. An improved data stream summary: the count-min sketch and its applications [J]. Journal of algorithms, 2005, 55(1): 58-75. DOI: 10.1016/j.jalgor.2003.12.001
[18] Estan C, Varghese G. New directions in traffic measurement and accounting: focusing on the elephants, ignoring the mice [J]. ACM transactions on computer systems, 2003, 21(3): 270-313. DOI: 10.1145/859716.859719
[19] Tang L, Huang Q, Lee P P C. MV-sketch: a fast and compact invertible sketch for heavy flow detection in network data streams [C]//Proceedings of IEEE INFOCOM 2019 - IEEE Conference on Computer Communications. IEEE, 2019: 2026-2034. DOI: 10.1109/infocom.2019.8737499
[20] Yang T, Zhang H W, Li J Y, et al. HeavyKeeper: an accurate algorithm for finding top- $k $ elephant flows [J]. ACM transactions on networking, 2019, 27(5): 1845-1858. DOI: 10.1109/tnet.2019.2933868
[21] Yang T, Jiang J, Liu P, et al. Elastic sketch: adaptive and fast network-wide measurements [C]//Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication. ACM, 2018: 561-575. DOI: 10.1145/3230543.3230544

作 者 简 介



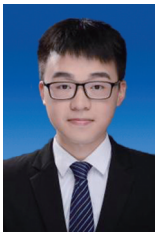
杨锋，中国电信集团有限公司高级工程师；主要从事IP网路技术、新型城域网建设规划与技术应用推进等工作。



尹远阳，中国电信股份有限公司研究院高级工程师；主要从事IP网络技术研究、新型城域网技术研究与评测、4G/5G 承载网新技术研究与评测等工作。



王江龙，中国电信集团有限公司高级工程师；主要从事IP网路技术、新型城域网建设规划与技术应用推进等工作。



杨顺兴，中国电信股份有限公司研究院工程师；主要从事IP网络技术研究与评测、IP骨干网、数据中心网络新技术评测等工作。