计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2020.20190776
57(7): 1347-1368, 2020

一种面向主干网的器件级动态功率感知节能机制

张金宏[1] 王兴伟[1] 易 波[1] 黄 敏[2]
[1](东北大学计算机科学与工程学院 沈阳 110169)
[2](东北大学信息科学与工程学院 沈阳 110819)
(neuzjh@aliyun.com)

A Component-Level Dynamic Power-Aware Energy-Saving Mechanism for Backbone Networks

Zhang Jinhong[1], Wang Xingwei[1], Yi Bo[1], and Huang Min[2]
[1](*School of Computer Science and Engineering, Northeastern University, Shenyang* 110169)
[2](*School of Information Science and Engineering, Northeastern University, Shenyang* 110819)

Abstract With a progressive increase of Internet traffic year by year, power consumption in the Internet is rising at an alarming rate, and the consequent environmental problems, e.g. the greenhouse effect caused by the surging carbon footprint and so on, have also aroused continuous concerns on a global scale, which are more serious especially in the backbone network where the aggregated traffic is transmitted. The oversupply principle for traditional Internet resources further aggravates these severe situations. With regard to this situation, a component-level dynamic power-aware energy-saving mechanism is devised over the backbone network in this paper. In the proposed mechanism, firstly, the incoming traffic size of nodes is dynamically predicted for a short term; then the fine-grained port number conversion algorithm is adopted to determine the number of ports to be regulated; then the corresponding ports convert their power states according to the sleeping and awakening rules; finally a novel hierarchical scheduling algorithm is devised to schedule the packets. In the simulation, based on the real traffic distribution traces over three typical backbone networks, we determine prediction parameters, test the proportionality of tracing load by power efficiency, explore the impacts of adopting different prediction time slot series and the different number of traffic load counters on the accuracy of load prediction, analyze the impacts of overestimation error and underestimation error of traffic load prediction that might appear on power consumption and discuss the tradeoff between power efficiency and actual performance in different application scenarios. Results demonstrate that the component-level power control mechanism proposed in the paper can control the power consumption of each network component dynamically and proportionally with a fine granularity and has a significantly energy-saving benefit.

Key words component-level energy saving; dynamic power awareness; fine-grained control; load prediction; hierarchical scheduling

收稿日期:2019-11-05;**修回日期**:2020-04-13

基金项目:国家重点研发计划项目(2017YFB0801701);国家自然科学基金项目(61872073)
This work was supported by the National Key Research and Development Program of China (2017YFB0801701) and the National Natural Science Foundation of China (61872073).

通信作者:王兴伟(wangxw@mail.neu.edu.cn)

摘 要 随着互联网流量逐年递增,网络功耗正以惊人的速度攀升,由此激增的碳足迹导致的温室效应等环境问题也引起了全球范围内的持续关注,尤其是在流量汇聚之后的主干网,这些问题更为突出.传统互联网资源的过供给原则进一步加剧了这种严峻的状况.鉴于此,面向主干网提出了一种器件级动态功率感知节能机制.该机制首先对节点入流量大小进行动态短期预测,进而采用细粒度的端口数转换算法确定需要调整的端口数目,之后依据休眠唤醒规则和速率调节规则控制相应的端口进行功率状态的转换,最后采用层次调度算法进行分组的调度.在实现方面,基于3个典型主干网中的真实流量分布轨迹,确定了预测参数,测试了功效随负载变化的比例性,探索了采用不同的预测时隙序列以及不同的流量负载计数器数目对负载预测准确度的影响,分析了可能出现的流量负载预测过估计误差和低估计误差对功耗和性能产生的影响,讨论了在不同应用场景下功效与实际性能之间的权衡.结果表明:提出的器件级功控机制能够动态、细粒度和比例性地控制各网元功耗,具有显著的节能收益.

关键词 器件级节能;动态功率感知;细粒度控制;负载预测;层次调度

中图法分类号 TP393

近些年,随着互联网用户数不断激增,互联网规模持续壮大.思科在其年度互联网报告白皮书中指出:预计全球互联网用户数和设备数将分别从2018年的39亿人和184亿台激增到2023年的53亿人和293亿台[1],由此引发互联网能耗急剧攀升.预计到2030年信息与通信技术(information and communication technologies, ICT)行业耗电量将高达82 650亿kW·h,其中,互联网耗电量高达66 900亿kW·h,约占80.94%,主干网高达26 410亿kW·h,约占31.95%[2].全球电子可持续发展倡议组织(Global e-Sustainability Initiative, GeSI)在SMART 2020,SMARTer2020,SMARTer2030一系列报告中指出,ICT行业的二氧化碳排放当量将以每年6%的速度递增,2020年将达到12.7×10^8t,约占全球总排放量的2.3%[3-5],预计到2030年此比重将增至23%[6].在波士顿咨询公司(Boston Consulting Group, BCG)2017年11月发行的关于互联网对气候变化影响的报告中指出互联网每年释放大约10×10^8t温室气体(其中主干网约占1/3),约占全球二氧化碳排放量的2%[7].面对如此严峻的状况,针对互联网尤其是主干网的节能变得刻不容缓.

尽管在终端用户设备和"最后一公里"相关技术与产品中已经采用了一些高能效的方法,但是主干网仍然处于高耗能低能效的状态[8].这主要源于目前主干网的设计遵循"过供给原则",即无论当前网络中流量大小均提供恒定的冗余网络资源以增加网络可靠性和容纳网络峰值流量需求[9-10].然而,实际上网络中的流量大小是动态变化的,并且在峰值和非峰值情况下流量差距很大[11].主干网的最大平均链路利用率低于30%,在非峰值流量期间甚至低于5%[12].在非峰值期间,由于现有网络设备不具备功率感知能力,其功耗无法随着资源利用率的变化而进行相应的调节,因此网络资源未得到充分利用,其峰值功耗造成了巨大的能量浪费,导致了严重的低功效[13-15].面对此种情况,需要设计一种功率感知机制以实现网元的功耗随其流量负载变化而自适应调节,即网元功耗能够紧随入流量大小而变化,网络只维持能够为所有入流量提供足够处理性能的必需数目的网元部件而休眠其余部件,这样能够消除不必要的功耗.该机制的实现需要基于对网元部件功耗状态的控制,但是如果我们仅仅对其进行粗粒度的功耗控制,则无法实现较为满意的功率感知效果.因此,我们提出了一种基于网元细粒度控制的功率感知路由机制,将网元的构成部件做进一步的功能拆分,提取出最小的可控器件单元—线卡端口,同粗粒度的路由器底架级控制和线卡级控制相比,基于线卡端口级的控制使得网元功耗能够更加精细地随着网络中流量负载的变化而进行相应的状态调整,进而实现网络资源更加充分的利用.此外,我们综合考虑了2种网元功耗调整策略,即基于动态功率管理(dynamic power management, DPM)技术的低功率闲置(low power idle, LPI)策略和基于动态电压频率调节(dynamic voltage and frequency scaling, DVFS)技术的自适应链路速率(adaptive link rate, ALR)策略,当网元满足一定的条件和阈值时可以采用这2种策略进行功耗状态转换,从而获得最大的节能收益.

作为功率感知的基础,流量负载情况的获取至关重要.由于主干网在一些时间段具有接近峰值和变化迅速的流量负载,因此我们需要动态获知主干

网中流量的变化情况并基于此实现网络中各个网元对其相应部件功耗的动态控制.一般说来,基于不同的动态流量获取方法所做出的管理控制决策可以分为反应式决策和前瞻式决策[16-17].反应式决策需要监测单元实时测量网络上的流量信息,并将监测结果发送给决策单元供其对相应部件进行管理控制,然而此过程所花费的时间对性能影响很大,因此通常对该策略的处理反应时间有较为苛刻的要求,这样才能降低因流量信息实时获取处理而导致的决策滞后性对网络管理控制产生的负面影响.反应式决策的优势在于对当前即时流量进行决策,因此决策准确度一般有保障.前瞻式决策依据预测流量进行决策,因此能够避免反应式决策固有的决策滞后问题,从而能够更加及时地对网络进行管理控制,但其准确性严重依赖于流量负载预测的准确性.鉴于主干网基础设施和流量自身的特点以及本文功率感知控制的场景需求,本文采用前瞻式决策对主干网中的流量信息进行及时预测.

迄今为止,对于网络流量需求的预测方法有许多研究,其中一些预测方法,如基于神经网络或者小波技术等的重量级预测方法,有着较高的计算复杂度和时间开销,通常具有秒级甚至更长的预测时间[18-19].但是,由于对主干网流量突变期间部件级的功率状态动态控制需要更加迅速的流量预测方法来满足器件级的功率状态动态控制,因此本文的预测方法使用滑动平均和滑动标准差模型[20],这样可以实现对主干网中流量波动情况的快速预测,而且从本文开展的实验中可以看到,通过进一步调整预测参数能够提高这种方法的预测准确度,从而提高了决策单元对网元器件的控制精度.此外,本文还使用并行化的预测计数器并扩展入流量的计数窗口,这样可以增加流量预测中使用的样本数目,从而使预测模块对流量负载预测的准确性和稳定性均得到改善.

一般说来,无论我们使用什么样的方法进行流量预测都不可能完全消除所有的预测误差,因为实际流量大小可能超过或低于预测流量,而这将引发入流量速率和器件处理速率之间的差异,进而可能会导致数据分组丢失.为了尽可能避免数据分组丢失,我们可以使用缓冲区.尽管缓冲区的使用能够在一定程度上减少数据分组丢失,但是缓冲区过大将导致处理延迟显著增长,使路由器的性能严重下降.鉴于此,本文工作在控制分组丢失的同时,尽可能减少数据分组的处理延迟,在决策模块中考虑缓冲区的使用状况以减少缓冲区中因数据分组累积而引发的处理延迟.

此外,基于 DiffServ 模型,本文在考虑各节点节能收益的同时,还考虑其对不同应用的服务质量(quality of service, QoS)支持,尽可能在获得最大节能收益的同时,提供必要的 QoS 支持.由于目前的典型主干网核心路由器间采用捆绑链路进行互连[21],因此本文中节点间的互连链路均为捆绑链路.

综上所述,本文的主要贡献包括 6 个方面:

1) 面向主干网,提出了基于捆绑链路的动态功率感知路由器模型,且在最大化各节点节能的同时兼顾应用 QoS 需求,在节点调度引擎中提出了层次调度算法,使得不同类型分组的 QoS 在节点处得以保证.

2) 与大多数研究工作中假设静态流量需求的做法不同,本文提出的动态功率感知节能机制能够使得网元功耗自适应于动态流量负载.

3) 不同于基于粗粒度的路由器底架级动态功率控制和线卡级动态功率控制,本文提出了基于线卡端口级的更加细粒度的动态功率控制方案,对不同的流量变化情况给出了相应的定量求解方法,使得网元的功耗能够更加精细地随着网络中流量负载的变化而进行相应的状态调整.

4) 综合考虑基于 DPM 技术的 LPI 策略和基于 DVFS 技术的 ALR 策略,使得这 2 种功耗调整策略互为补充.

5) 针对主干网基础设施和流量自身变化的特点,为了避免信息获取的滞后性以实现及时的控制管理决策,本文采用滑动平均和滑动标准差模型对流量进行快速准确的预测,并且在预测时隙序列的选取上考虑了同期预测和环期预测,并比较了基于两者的流量负载预测准确性差异,进而针对流量预测过估计误差和流量预测低估计误差对节点功耗和节点性能的影响进行了全面的评价.

6) 在不同应用场景下,探索本文机制在功效与最差延迟之间的权衡以及在功效与缓冲区占用率之间的权衡.

1 相关工作

目前,针对主干网器件级节能的研究工作,按控制策略可以分为 ALR,LPI 和混合策略等;按控制粒度可以分为粗粒度和细粒度;按实现目标可以分为在性能可接受的前提下仅以最大化节能为目标的

约束节能和在节能与性能之间寻求最佳平衡点的权衡节能;按演进范畴可以分为完全打破且不依赖于原有的网络架构和网络协议等而进行全新设计的革新式以及在原有的网络架构和网络协议等的基础上进行扩充和改进的增补式.尽管革新式通常可以获取更为显著的节能效果,但其实现成本巨大;而增补式通常实现的节能效果较前者有限,但代价也较前者小很多.

ALR[22]根据链路/节点端口的负载情况,自适应动态调节链路/节点端口中数据传输/处理速率,使链路/节点端口能够在低负载时减小功耗,实现节能.它使不同的链路/节点端口工作在不同的服务速率和相应的功耗等级上,在链路/节点端口处于低利用率时降低链路/节点端口的传输/处理速率,能够在保证有限的性能影响下有效地降低功耗.网络中的任一链路/节点端口可以根据链路传输负载情况/节点端口处理负载情况,使用基于阈值的方法选择处于某一工作状态的传输/处理速率机制,当链路/端口利用率超过或低于某一阈值时,相应地调整传输/处理速率.

LPI[23-25]通过在低链路利用率期间关闭链路任一端相连的网元或网元部件实现网络设备功率的节省,在需要正常传输数据时恢复对已关闭部分的正常供电,以此节约网络运营成本和实现网络通信的高效节能.

文献[18,26-30]的研究工作均着眼于器件级节能.文献[18]提出了一种时钟频率调节路由器架构,允许其内部模块依据流量负载采用不同的时钟频率运行.它给出了在不同网络环境下的4种频率切换策略:休眠唤醒切换策略、上边界切换策略、双边界切换策略和组合切换策略,以减少路由器功耗从而实现网络节能,但是通常这样频繁的调节会导致功率控制复杂化,而且产生的控制延迟不可忽略.文献[26]针对路由器线卡,提出了一种时钟频率自适应调节策略以最小化主干网能耗,该策略依据路由器线卡实际承载的流量需求,自适应调节路由器线卡时钟频率,这样,路由器线卡不需要总保持在全速运行状态,从而实现节能.在假设已知不同时隙的不同节点对之间的预测流量需求矩阵的前提下,它给出了一个混合整数线性规划模型,在不同时隙为每个路由器线卡选择最优时钟频率,从而使所有路由器线卡的总能耗最小.文献[27]提出了一种广义的ALR策略.它将网元的休眠视为服务速率为0的特殊情形.基于真实的分组轨迹,它分析了使用该策略的路由器对其邻居路由器产生的影响.结果表明:当一个路由器采用该策略时,其下游路由器可获得高达30%的节能.文献[28]提出了一种被称为"GreenRouter"的新型路由器架构,将一个线卡分成2个部分:网络接口卡和分组处理卡,且在这2部分之间通过一个2级交换结构相连.在该架构中,从所有网络接口卡进入的流量能够共享所有分组处理卡,而且这些流量可以按需聚集到一部分分组处理卡上,这样其余的分组处理卡可以被关闭以实现节能.文献[29]提出了一种将能效以太网(energy efficient Ethernet, EEE)协议和eBond(energy-aware bonding)协议相结合的混合协议"eeeBond",在每个路由器的网络接口内部执行EEE协议来休眠与唤醒网络接口,在不同网络接口间执行eBond协议进行网络接口切换,从而使路由器能耗自适应动态带宽需求.它给出了一个统一的协议性能分析模型,推导出了用于设置协议最优参数的闭型表达式.结果表明,通过使用eeeBond协议和设置最优参数,网络可以实现最大节能.文献[30]基于NetFPGA平台,研究网络设备功率的刻画方法及其调节机制.它给出了一个量化的NetFPGA交换机/路由器部件能耗的测量框架,提出了一个功率调节算法,根据实际流量负载调节FPGA(field programmable gate array)内核和以太网接口的运行时钟频率,但是它仅仅是基于硬件的节能.文献[26-27]是基于路由器线卡级的粗粒度控制,而文献[18,28-30]是基于路由器线卡端口级的细粒度控制,它们均未考虑QoS支持,大多采用单一节能策略,其中文献[28-29]基于LPI策略,文献[18,26,30]基于ALR策略,少数(如文献[27])考虑了LPI和ALR混合策略.

相比以上研究工作,本文提出的机制是面向主干网的增补式细粒度短期动态功率感知的约束节能,它在实现各节点功耗最小化的同时,兼顾对不同应用的QoS支持和性能保证,同时综合运用多种节能策略.

2 问题描述

2.1 网络模型

本文将主干网建模为连通图$G(V,E)$,其中连通图顶点集合$V=\{v_1,v_2,\cdots,v_n\}$表示所有网络节点的集合,连通图边集合$E=\{e_1,e_2,\cdots,e_m\}$表示所有网络链路的集合.

2.2 节点模型

本文提出的节点结构如图 1 所示,包括主控引擎、背板、底架、交换结构、缓冲区、调度引擎、线卡、转发引擎、复制引擎和端口等构件以及负载预测、缓冲区观测、决策、休眠唤醒控制和速率调节等功能模块.

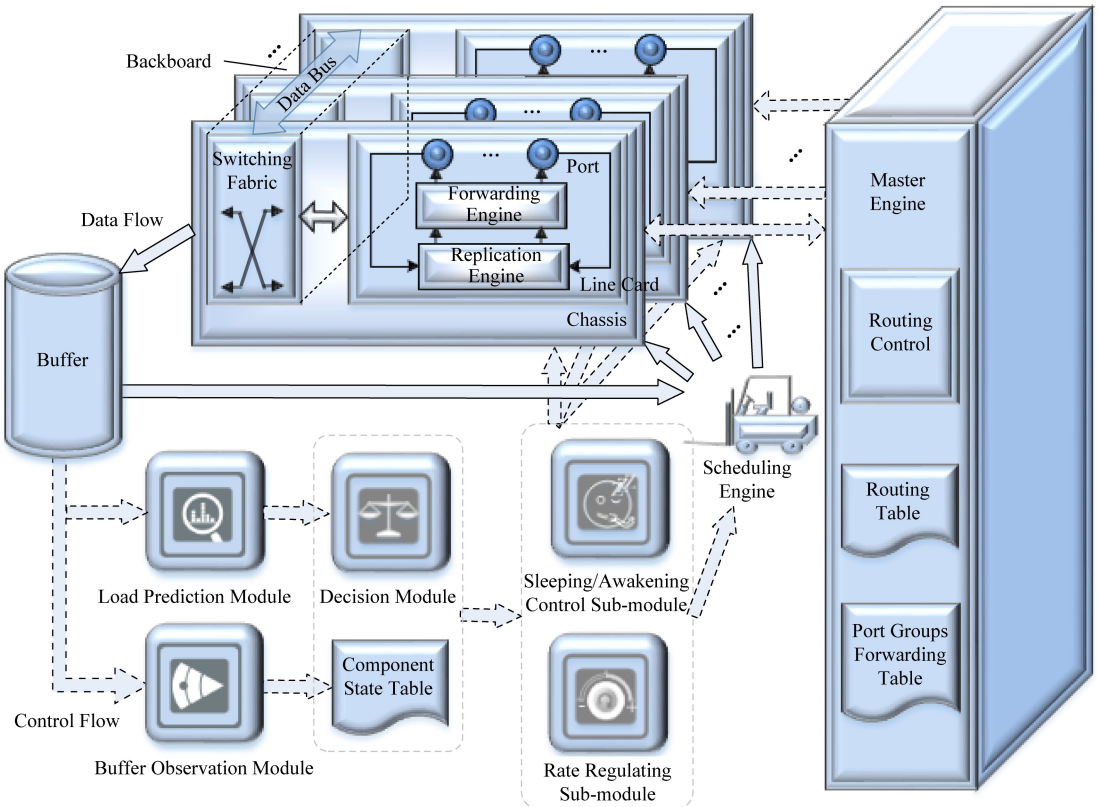


Fig. 1 Node structure

图 1 节点结构

主控引擎是路由器的控制中心,用于运行路由协议、实现配置管理和路由表查找等功能.背板由数据总线和交换结构组成,是路由器内部数据交换通道.底架用于承载线卡和交换结构,为线卡提供连接槽位.交换结构用于在路由器内部连接线卡的输入端口和输出端口.线卡用于实现分组处理、队列调度和流量管理等功能.转发引擎用于完成分组输入、存储与转发等功能.复制引擎用于组播所需的分组复制.端口用于连接路由器和外部线路,并在两者之间进行数据传输.缓冲区调节数据入口速率和数据出口速率之间的差异,部署缓冲区能够应对流量预测错误的发生.使用较大的缓冲区能够容忍更大的预测错误从而避免包丢失,但是包转发时延将变大.因此,缓冲区尺寸应该根据延迟容忍来决定.缓冲区观测模块对缓冲区的当前使用情况进行周期性观测并将此观测结果发送给决策模块.负载预测模块使用负载的历史统计信息和当前的负载情况来估计未来负载.在负载预测模块中,负载计数器统计入口负载的字节数,预测计算器根据负载计数器统计出的字节数计算预测值.决策模块根据由负载预测模块计算出的预测结果和由缓冲区观测模块获取到的缓冲区使用量综合决策,确定需要调节速率的端口数和/或需要唤醒/休眠的端口数(负载需求增加时,优先使用速率调节策略,如果将所有活动端口的速率都调至最大后仍无法满足负载需求时,则使用休眠控制策略唤醒必要数量的端口;否则,反之.)之后,决策模块将端口状态转换结果,即需要调整速率的端口数和需要休眠唤醒的端口数,发送到状态控制模块,即速率调节子模块和休眠唤醒子模块.

本文把当前节点中所有连接相同下一跳节点的不同端口分到 1 组,这样形成的 1 组端口称之为端口组.不同的下一跳节点对应不同的端口组,这样形成的转发表称之为端口组转发表,如表 1 所示:

Table 1 Forwarding Table Based on Port Groups

表 1 端口组转发表

Component	Next Hop Node
1#chassis 1#line card 1#port	v_j
1#chassis 1#line card 2#port	v_j
1#chassis 1#line card 3#port	v_k
1#chassis 1#line card 4#port	v_k
⋮	⋮
2#chassis 1#line card 1#port	v_l
2#chassis 1#line card 2#port	v_m
2#chassis 1#line card 3#port	v_k
2#chassis 1#line card 4#port	v_m
⋮	⋮
n#chassis m#line card p#port	v_k
⋮	⋮

休眠唤醒控制模块基于决策模块的端口数量调整信令和器件状态表中记录的器件已休眠/已唤醒时间,依据已唤醒时间越长休眠优先级越高的休眠原则和已休眠时间越长唤醒优先级越高的唤醒原则,具体得出休眠/唤醒端口组中的哪些端口,并向调度引擎和相应的端口下发功率状态转换信令.速率调节模块基于决策模块的端口数量调整信令和器件状态表中记录的器件所处不同速率等级的持续时间,依据同一速率等级持续时间越长调节优先级越高的调节原则,具体得出调节端口组中的哪些端口的转发速率,并向调度引擎和相应的端口下发功率状态转换信令.调度引擎在得到来自休眠唤醒控制模块和速率调节模块的相关信令后,向相应端口并行转发数据分组.相应的端口根据接收到的来自休眠唤醒控制模块和速率调节模块的相关信令进行相应的功率状态转换操作.节点内部各构件和功能模块间的逻辑关系如图 2 所示:

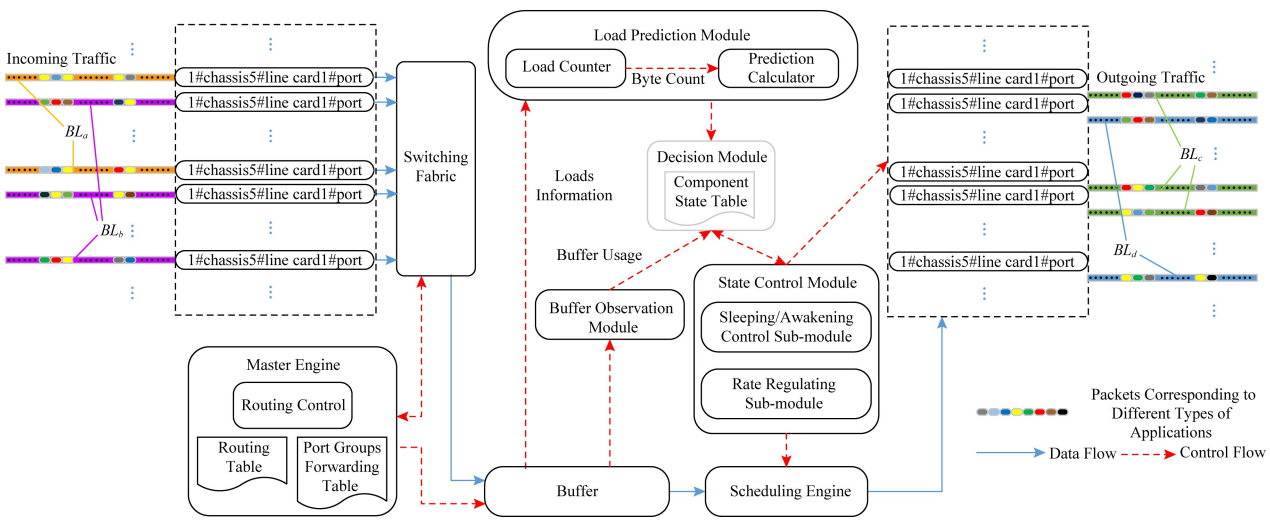


Fig. 2 Logical relationship among components and function modules inside the node

图 2 节点内部各构件和功能模块间逻辑关系

2.3 链路模型

典型情况下,主干网每对节点间由多条物理链路互连,这些链路形成一条逻辑捆绑链路[21].本文采用文献[31]的做法假设节点 v_i 和节点 v_j 之间的捆绑链路 BL_{ij} 由 n_{ij} 条容量相同的物理链路组成,表示为:$BL_{ij}=\{l_{i_1j_1},l_{i_2j_2},\cdots,l_{i_{n_j}j_{n_j}}\}$.本文抽象每条物理链路的结构如图 3 所示,包括功率放大器、在线放大器、光再生器和前置放大器等中间设备.其中,功率放大器用来提高信号发送功率,在线放大器用来延长信号传输距离,光再生器用来对信号进行整形,前置放大器用来改善接收端灵敏度.

定义 1. 物理链路速率集.假设每条物理链路都

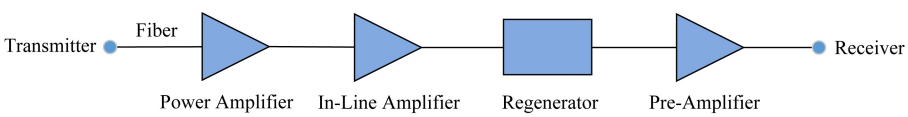


Fig. 3 Link structure

图 3 链路结构

遵循 ALR 策略,则对于不同的链路利用率 LU,其自适应链路速率 R 可以确定:

$$R=\begin{cases}R_1, & LU\leqslant\theta_1;\\ R_2, & \theta_1<LU\leqslant\theta_2;\\ \vdots & \\ R_i, & \theta_{i-1}<LU\leqslant\theta_i;\\ \vdots & \\ R_{k-1}, & \theta_{k-2}<LU\leqslant\theta_{k-1};\\ R_k, & LU>\theta_{k-1}.\end{cases}\tag{1}$$

其中,$\theta_i(i=1,2,\cdots,k-1)$表示划分阈值.我们将这些不同速率组成的集合称为物理链路速率集,记为 $R_{\text{link}}=\{R_1,R_2,\cdots,R_k\}$.

2.4 功耗模型

定义 2. 端口休眠唤醒状态集与合并状态集.当端口处于活动状态 S_a 时,端口正常运行和处理分组;当端口处于浅层休眠状态 S_s 时,端口不处理分组,但是使用一小部分功耗以维持端口的初始化状态,这样可以实现其从该状态返回到活动状态的快速苏醒;当端口处于深层休眠状态 S_d 时,端口被完全关闭,其功耗为 0.因此,端口的休眠唤醒状态集可以表示为 $S=\{S_a,S_s,S_d\}$.此外,再考虑到与上述物理链路速率集相对应的端口速率集 $R=R_{\text{link}}=\{R_1,R_2,\cdots,R_k\}$,我们得到端口的合并状态集 $S=\{S_{a_1},S_{a_2},\cdots,S_{a_k},S_s,S_d\}$.

在休眠唤醒 3 个状态中只有端口处于活动状态才能够处理数据分组,如果处于活动状态的端口不足以满足流量需求,则将引起包丢失,因此处于休眠状态的端口必须在可接受的时间内按需激活;相反,如果因流量需求减少而不再需要过多处于活动状态的端口,则需使多余的端口进入休眠状态以实现节能.图 4 展示了端口状态之间的转换.

Fig. 4 Transforming among port states

图 4 端口状态之间的转换

从深层休眠状态到活动状态或浅层休眠状态,需要包括更新内存和存储初始信息等操作.在浅层休眠状态,功率被持续供给以保持内存芯片上所有的数据,只是没有提供时钟信号,因此唤醒器件从浅层休眠状态到活动状态只需要重新提供时钟信号.同深层休眠状态的唤醒时间相比,从浅层休眠状态到活动状态的唤醒时间很短.表 2 比较了 3 种状态的属性细节,本文设从活动状态到浅层/深层休眠状态和从浅层休眠状态到深层休眠状态的转换时间为 0,因为这些转换仅是简单地停止供电.

Table 2 Property Comparison Among Different Port States

表 2 不同端口状态间的属性比较

Power State	Running Memory	Keeping Information	Power Consumption	Awakening Time
Active State	Yes	Yes	Much	No
Shallow Sleeping State	No	Yes	A Little	Negligible
Deep Sleeping State	No	No	No	Non-Negligible

定义 3. 端口之外的器件(如线卡和底架)状态.开启状态,器件正常运行和处理分组;关闭状态,完全关闭器件,其功耗为 0.

当且仅当一个线卡上的所有端口都处于深度休眠状态,这个线卡才能关闭;同理,当且仅当一个底架上的所有线卡都关闭了,这个底架才能关闭.

表 3 是器件状态表,包括所有器件当前的状态以及该状态所持续的时间.

根据节点内部各构件的工作原理、彼此间的交互方式、DPM 技术、LPI 策略以及功率状态划分定义,抽象出节点功耗模型:

$$P_{v_i}=(P_i^{\text{ctrl}}+P_i^{\text{sch}}+\sum_{k=1}^{N_i^{\text{chass}}}(P_{i,k}^{\text{chass}}\times ChaSt_i^k+\sum_{lc=1}^{N_{i,k}^{\text{card}}}((P_{i,k,q}^{\text{ford}}+P_{i,k,q}^{\text{repl}})\times LcdSt_{i,k}^q)))\times NodeSt_i,\tag{2}$$

其中,P_{v_i} 表示节点 v_i 的功耗,P_i^{ctrl} 表示节点 v_i 主控引擎的功耗,P_i^{sch} 表示节点 v_i 调度引擎的功耗,$P_{i,k}^{\text{chass}}$ 表示节点 v_i 中第 k 个底架(背板交换结构的功耗计入到对应的底架中,数据总线功耗忽略不计)的基准功耗(不插任何线卡时底架的功耗),$P_{i,k,q}^{\text{ford}}$ 表示

节点 v_i 中第 k 个底架上第 q 个线卡的转发引擎的功耗，$P_{i,k,q}^{\text{repl}}$ 表示节点 v_i 中第 k 个底架上第 q 个线卡的复制引擎的功耗；N_i^{chass} 表示节点 v_i 的底架总数，$N_{i,k}^{\text{card}}$ 表示节点 v_i 中第 k 个底架上的线卡总数；$NodeSt_i$，$ChaSt_i^k$，$LcdSt_{i,k}^q$ 分别表示节点 v_i 的开关标识符、节点 v_i 中第 k 个底架的开关标识符和节点 v_i 中第 k 个底架上第 q 个线卡的开关标识符，这 3 个标识符都是二进制变量，其值为“1”表示对应的节点或器件处于正常工作的活动“开”状态，其值为“0”表示对应的节点或器件处于休眠“关”状态.

Table 3 Component State Table

表 3 器件状态表

Component	Current State	Rate Level	State Duration
1# chassis	Open		t_1
1# chassis 1# line card	Open		$t_{1,1}$
1# chassis 1# line card 1# port	Shallow Sleeping		$t_{1,1,1}$
1# chassis 1# line card 2# port	Deep Sleeping		$t_{1,1,2}$
1# chassis 1# line card 3# port	Active	1	$t_{1,1,3}$
1# chassis 1# line card 4# port	Active	2	$t_{1,1,4}$
1# chassis 2# line card	Close		$t_{1,2}$
⋮	⋮	⋮	⋮
2# chassis	Close		t_2
2# chassis 1# line card	Close		$t_{2,1}$
2# chassis 1# line card 1# port	Deep Sleeping		$t_{2,1,1}$
2# chassis 1# line card 2# port	Deep Sleeping		$t_{2,1,2}$
2# chassis 1# line card 3# port	Deep Sleeping		$t_{2,1,3}$
2# chassis 1# line card 4# port	Deep Sleeping		$t_{2,1,4}$
⋮	⋮	⋮	⋮
n# chassis	Open		t_n
n# chassis 1# line card	Open		$t_{n,1}$
⋮	⋮	⋮	⋮
n# chassis m# line card p# port	Active	3	$t_{n,m,p}$
⋮	⋮	⋮	⋮

基于 DVFS 技术、上述的 ALR 策略和先前给出的链路模型，抽象捆绑链路功耗模型：

$$P_{l_{ij}}=\sum_{k=1}^{n_j}P_{i_kj_k}^{\text{link}}=\sum_{k=1}^{n_j}((P_{i_k}+P_{j_k}+(P_{\text{PA}}+P_{\text{ILA}}\times N_{i_kj_k}^{\text{ILA}}+P_{\text{REG}}\times N_{\text{REG}\,i_kj_k}+P_{\text{PREA}}))\times LinkSt_{i_kj_k}) \tag{3}$$

其中，$P_{l_{ij}}$ 表示节点 v_i 和节点 v_j 之间的捆绑链路 BL_{ij} 的功耗；$P_{i_kj_k}^{\text{link}}$ 表示组成捆绑链路 BL_{ij} 的物理链路 $l_{i_kj_k}$ 的功耗；P_{PA} 表示功率放大器的功耗；P_{ILA} 表示在线放大器的功耗；P_{REG} 表示光再生器的功耗；P_{PREA} 表示前置放大器的功耗；$N_{i_kj_k}^{\text{ILA}}$ 和 $N_{i_kj_k}^{\text{REG}}$ 分别表示物理链路 $l_{i_kj_k}$ 中在线放大器和光再生器的数目；$LinkSt_{i_kj_k}$ 是物理链路 $l_{i_kj_k}$ 的开关标识符，这个标识符是二进制变量，其值为“1”表示该物理链路及其两端端口处于活动状态或浅层休眠状态，其值为“0”表示该物理链路及其两端端口处于深层休眠状态；P_{i_k} 和 P_{j_k} 分别表示物理链路 $l_{i_kj_k}$ 的入端口 $port_{i_k}^{ij}$ 和出端口 $port_{j_k}^{ij}$ 的功率：

$$P_{i_k}=\begin{cases}P_{i_k}^{\text{base}}+(P_{i_k}^{\max}-P_{i_k}^{\text{base}})\times(R_{i_kj_k}^{\text{link}}/\psi_{ij}^{\text{link}})^h, & S_{i_k}^{ij}\in S_{\text{a}},\\ \varepsilon\times P_{i_k}^{\text{base}}, & S_{i_k}^{ij}=S_{\text{s}},\\ 0, & S_{i_k}^{ij}=S_{\text{d}},\end{cases} \tag{4}$$

$$P_{j_k}=\begin{cases}P_{j_k}^{\text{base}}+(P_{j_k}^{\max}-P_{j_k}^{\text{base}})\times(R_{i_kj_k}^{\text{link}}/\psi_{ij}^{\text{link}})^h, & S_{j_k}^{ij}\in S_{\text{a}},\\ \varepsilon\times P_{j_k}^{\text{base}}, & S_{j_k}^{ij}=S_{\text{s}},\\ 0, & S_{j_k}^{ij}=S_{\text{d}},\end{cases} \tag{5}$$

其中，$S_{i_k}^{ij}$ 和 $S_{j_k}^{ij}$ 分别表示物理链路 $l_{i_kj_k}$ 的入端口 $port_{i_k}^{ij}$ 和出端口 $port_{j_k}^{ij}$ 的功率状态；$P_{i_k}^{\text{base}}$ 和 $P_{j_k}^{\text{base}}$ 分别

表示物理链路 $l_{i_kj_k}$ 的入端口 $port_{i_k}^{ij}$ 和出端口 $port_{j_k}^{ij}$ 的基准功率;$P_{i_k}^{\max}$ 和 $P_{j_k}^{\max}$ 分别表示物理链路 $l_{i_kj_k}$ 的入端口 $port_{i_k}^{ij}$ 和出端口 $port_{j_k}^{ij}$ 的最大功率;ψ_{ij}^{link} 和 $\lambda_{ij}^{\text{link}}$ 分别表示组成捆绑链路 BL_{ij} 的单个物理链路的单侧端口容量和缓冲区大小;ε 是功率比重调节系数,用于调节浅层休眠端口功耗在端口基准功率中所占的比重;h 是功耗和链路速率之间的相关系数,用于描述链路速率与功率之间的对应关系;$R_{i_kj_k}^{\text{link}}$ 表示物理链路 $l_{i_kj_k}$ 的链路速率.由于链路速率单阈值切换策略较为粗糙会导致节能效果不理想,而链路速率多阈值切换策略的速率频繁切换和速率滞后调整会导致速率抖动问题[32],因此本文采用链接速率双阈值切换策略,其表达式:

$$R_{i_kj_k}^{\text{link}}=\begin{cases}R_1^{ij}, & LU_{i_kj_k}^{\text{link}}\leqslant\eta_1^{i_kj_k}.\\ R_2^{ij}, & \eta_1^{i_kj_k}<LU_{i_kj_k}^{\text{link}}\leqslant\eta_2^{i_kj_k},k=1,2,\cdots,n_j.\\ R_3^{ij}, & LU_{i_kj_k}^{\text{link}}>\eta_2^{i_kj_k}.\end{cases}\tag{6}$$

其中,$LU_{i_kj_k}^{\text{link}}$ 表示链路利用率,$\eta_1^{i_kj_k}$ 和 $\eta_2^{i_kj_k}$ 分别为 $LU_{i_kj_k}^{\text{link}}$ 的上阈值和下阈值,$R_{i_kj_k}^{\text{link}}$ 随着 $LU_{i_kj_k}^{\text{link}}$ 所处的不同变化范围而进行相应等级的切换,R_1^{ij},R_2^{ij},R_3^{ij} 分别为 $R_{i_kj_k}^{\text{link}}$ 由低到高划分的 3 个等级.

基于上述的节点功耗模型和链路功耗模型,全网的功耗模型为

$$P_{\text{net}}=\sum_{v_i\in V}P_{v_i}+\sum_{l_{ij}\in E}P_{l_{ij}}.\tag{7}$$

3 器件级功控机制

器件级功控机制既要考虑流量预测又要考虑缓冲区的占用情况.图 5 展示了对从当前节点 v_i 流向下一跳节点 v_j 的数据分组,器件级功控机制(component-level power controlling mechanism,CPCM)是如何进行工作的,其中,I_{pre} 为预测时隙,I_{obs} 为缓冲区观测时隙,$I_{\text{pre}}=I_{\text{obs}}$,$I_{\text{swi}}$ 为端口进行功率状态转换的时隙,D_{cal} 为因预测计算而产生的时延,D_{dec} 为因确定需要进行状态转换的端口数目而产生的时延,D_{ide} 为因落实需要进行功率转换的具体端口而产生的时延,D_{tra} 为因端口进行功率状态转换而产生的时延.

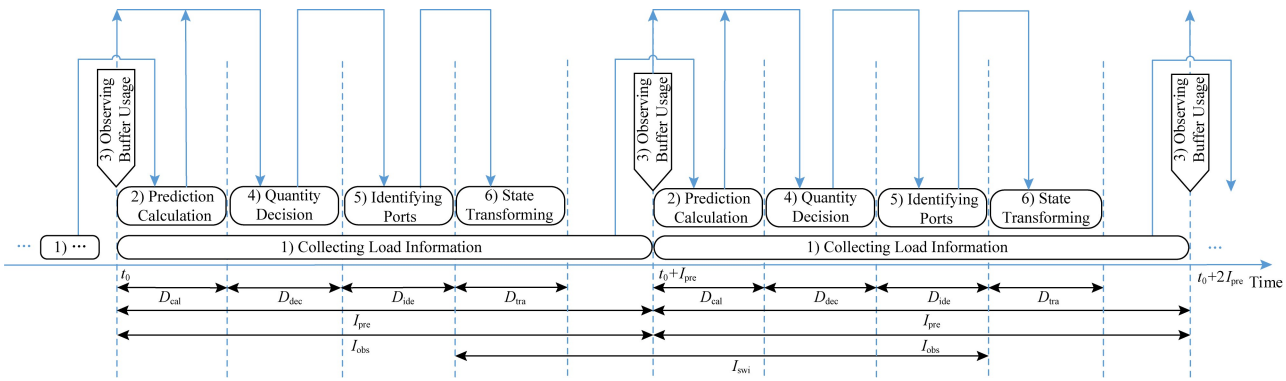


Fig. 5 Schematic diagram for component-level power controlling mechanism

图 5 器件级功控机制示意图

从图 5 可以看出,器件级功控机制主要包括 6 个阶段:

1) 负载预测模块收集流量负载信息(从 $t=t_0$ 到 $t=t_0+I_{\text{pre}}$);

2) 根据第 1 阶段收集的信息计算出预测的负载大小(从 $t=t_0+I_{\text{pre}}$ 到 $t=t_0+I_{\text{pre}}+D_{\text{cal}}$);

3) 通过缓冲区观测模块获取缓冲区使用量(在 $t=t_0+I_{\text{pre}}$);

4) 根据从第 2,3 阶段获取的结果,决策模块决定当前节点未来需要运行所处的功耗状态等级(需要多少器件运行并运行在怎样的速率级别上),并将决策结果发送给休眠唤醒控制模块和速率调节模块(从 $t=t_0+I_{\text{pre}}+D_{\text{cal}}$ 到 $t=t_0+I_{\text{pre}}+D_{\text{cal}}+D_{\text{dec}}$);

5) 休眠唤醒控制模块和速率调节模块根据休眠/唤醒/调节原则确定需要进行状态转换的端口,并发送信令给调度引擎和相应的端口(从 $t=t_0+I_{\text{pre}}+D_{\text{cal}}+D_{\text{dec}}$ 到 $t=t_0+I_{\text{pre}}+D_{\text{cal}}+D_{\text{dec}}+D_{\text{ide}}$);

6) 所有收到信令的端口开始进行相应的状态转换,调度引擎根据信令将数据分组发送给相应端口(从 $t=t_0+I_{\text{pre}}+D_{\text{cal}}+D_{\text{dec}}+D_{\text{ide}}$ 到 $t=t_0+I_{\text{pre}}+D_{\text{cal}}+D_{\text{dec}}+D_{\text{ide}}+D_{\text{tra}}$).

3.1 预测模块

在器件级功控机制中,为了减少因负载预测而产生的时延和避免因为缓冲区溢出而导致的包丢失,轻量级的准确而快速的短期负载预测对于路由

器细粒度功率控制是必需的.为了满足这样的预测需求,本文使用自回归滑动平均模型的滑动平均和滑动标准差进行高精度轻量级的负载预测.用 A_t^{ij} 表示在预测时隙 t 当前节点 v_i 中流向下一跳节点 v_j 的负载的平均大小,其可以通过在第 1 阶段(即在时隙 $t-1$)测量窗口中观测到的分组字节计数的滑动平均得到;用 E_t^{ij} 表示在预测时隙 t 当前节点 v_i 中流向下一跳节点 v_j 的负载大小的标准差,其可以通过在第 1 阶段(即在时隙 $t-1$)测量窗口中观测到的分组字节计数的滑动标准差得到.

3.1.1 预测时隙序列的选取

选取不同的时隙序列$(1,2,\cdots,t-1)$进行预测直接影响到预测准确性,本文尝试采用 3 种方法确定预测所需的时隙序列$(1,2,\cdots,t-1)$以对时隙 t 内的负载大小进行较为全面地预测.

定义 4. 广义同期预测.将基于先前每天中与当前时隙相同时隙负载大小对当前时隙负载大小进行的预测称为广义同期预测(general same-period prediction, GSP).

定义 5. 狭义同期预测.将基于先前每周对应工作日中与当前时隙相同时隙负载大小对当前时隙负载大小进行的预测称为狭义同期预测(special same-period prediction, SSP).

定义 6. 环期预测.将基于先前连续时隙负载大小对当前时隙负载大小进行的预测称为环期预测(continuous-period prediction, CP).

3.1.2 计数器数目的选取

在预测模块中,C_{t-1}^{ij} 的获取将显著影响负载预测的准确度.

1) 单一计数器

只使用一个负载计数器获取 C_{t-1}^{ij},并设置与预测时隙相同的值作为负载计数器窗口的宽度,如图 6 中的单个计数器窗口所示.此时,A_t^{ij} 和 E_t^{ij} 可计算得到:

$$A_t^{ij}=\begin{cases}(1-2^{-\alpha})\times A_{t-1}^{ij}+2^{-\alpha}\times C_{t-1}^{ij}, \\ \quad t\geqslant 3 \text{ 且 } t\in\mathbb{N}, \\ C_1^{ij},\ t=2,\end{cases} \tag{8}$$

$$E_t^{ij}=\begin{cases}(1-2^{-\beta})\times E_{t-1}^{ij}+2^{-\beta}\times|A_t^{ij}-C_{t-1}^{ij}|, \\ \quad t\geqslant 3 \text{ 且 } t\in\mathbb{N}, \\ 0,\ t=2,\end{cases} \tag{9}$$

其中,α 和 β 分别表示滑动平均和滑动标准差的平滑参数.从式(8)和式(9)可以看出,由于预测计算能够通过移位寄存器来实现高速运算,因此在负载预测模块中的计算时延可以忽略,器件级功控机制的总时延可以表示为 $D=D_{\text{cal}}+D_{\text{dec}}+D_{\text{ide}}+D_{\text{tra}}\approx D_{\text{dec}}+D_{\text{ide}}+D_{\text{tra}}$.由于此预测方法计算量较小,因此适合负载高的核心路由器.

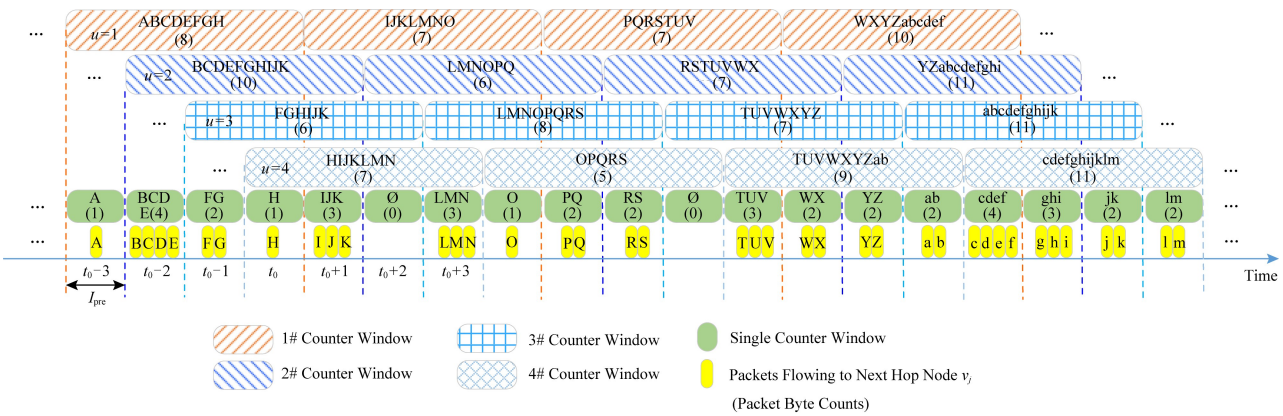


Fig. 6 Comparison among single counter and multiple counters

图 6 单个计数器和多个计数器的比较

2) 多计数器

使用单个计数器窗口可能会因被测量的 C_{t-1}^{ij} 缺乏测量样本而引起大的波动,导致在统计上是不可靠的,这将恶化预测准确度.出于这个原因,本文采用扩展计数窗口(使用比预测时隙更大的值作为计数窗口的宽度)并使用多个负载计数器的方法获取 C_{t-1}^{ij},可以显著改善负载预测准确度.

以图 6 为例,如果单一计数器,则在一个计数窗口中的分组数在$[0,4]$之间变化,而如果 4 个计数器,则在一个计数窗口中的分组数在$[5/4,11/4]$之间缓慢变化.可见,多计数器窗口的使用增强了 C_{t-1}^{ij} 的统计稳定性,可以显著改善流量负载预测准确度以及进一步避免频繁的预测决策带来的网元器件功率状态切换震荡.

据此,A_t^{ij} 和 E_t^{ij} 的计算可修改为

$$A_t^{ij}=\begin{cases}(1-2^{-\alpha})\times A_{t-1}^{ij}+2^{-\alpha}\times C_{t-1}^{ij}(u)/K,\\ \quad u=1+t \bmod K, t\geqslant 3 \text{ 且 } t\in\mathbb{N},\\ C_1^{ij}(u), u=1+t \bmod K, t=2,\end{cases} \tag{10}$$

$$E_t^{ij}=\begin{cases}(1-2^{-\beta})\times E_{t-1}^{ij}+2^{-\beta}\times|A_t^{ij}-C_{t-1}^{ij}(u)/K|,\\ \quad u=1+t \bmod K, t\geqslant 3 \text{ 且 } t\in\mathbb{N},\\ 0, t=2,\end{cases} \tag{11}$$

A_t^{ij} 和 E_t^{ij} 均依次在不同的计数窗口周期性地进行取值计算.其中,K 为计数器的总数,$C_{t-1}^{ij}(u)$ 为从时隙 $t-K$ 到时隙 $t-1$ 期间在第 $u\in\{1,2,\cdots,K\}$ 个计数窗口中包含的分组字节计数.以图 6 为例,我们不妨假设时隙 $t_0 \bmod 4=0$,则 $(t_0+1) \bmod 4=1$,$(t_0+2) \bmod 4=2$,$(t_0+3) \bmod 4=3$,$C_{t_0}^{ij}(u=1)$ 为从时隙 t_0-3 到时隙 t_0 期间第 $u=1$ 个计数窗口中的分组字节计数,$C_{t_0+1}^{ij}(u=2)$ 为从时隙 t_0-2 到时隙 t_0+1 期间第 $u=2$ 个计数窗口中的分组字节计数,$C_{t_0+2}^{ij}(u=3)$ 为从时隙 t_0-1 到时隙 t_0+2 期间第 $u=3$ 个计数窗口中的分组字节计数,$C_{t_0+3}^{ij}(u=4)$ 为从时隙 t_0 到时隙 t_0+3 期间第 $u=4$ 个计数窗口中的分组字节计数.

3.2 缓冲区

无论采用何种预测方法都会有误差存在,因此本文在节点内部引入缓冲区来容忍这些误差.然而,这些误差可能会逐渐累积在缓冲区中,从而增加缓冲区溢出风险,因此需要定期观测缓冲区的使用量.

为了避免缓冲区溢出导致数据分组丢失,本文设置缓冲区警戒阈值.当缓冲区使用量超过该阈值时,缓冲区观测模块启用观测值加倍机制,即将双倍的当前缓冲区实际使用量作为观测值向决策模块进行反馈,以便预留足够的端口容量.

假设在时刻 t 缓冲区观测模块向决策模块反馈的当前节点 v_i 的缓冲区中等待转发到下一跳节点 v_j 的数据分组大小的观测值为 B_t^{ij},则其计算为

$$B_t^{ij}=\begin{cases}b_t^{ij}\times(I_{\text{pre}}/I_{\text{obs}}), t=n\times(I_{\text{obs}}/I_{\text{pre}}),\\ \quad \sum\limits_j b_t^{ij}/B<\zeta,\\ 2\times b_t^{ij}\times(I_{\text{pre}}/I_{\text{obs}}), t=n\times(I_{\text{obs}}/I_{\text{pre}}),\\ \quad \sum\limits_j b_t^{ij}/B\geqslant\zeta, n\in\mathbb{N}\\ B_{t-1}^{ij}, t\neq n\times(I_{\text{obs}}/I_{\text{pre}}),\end{cases} \tag{12}$$

其中,b_t^{ij} 表示在时刻 t 当前节点 v_i 缓冲区中等待转发到下一跳节点 v_j 的数据分组的实际缓冲区使用量,ζ 为警戒阈值.

由于使用缓冲区会增加数据分组的等待时延,因此为了既实现节省网络功耗又避免节点处理性能下降,需要在功效和缓冲区占用率之间进行必要的均衡(详见 4.3.2 节).

3.3 决策模块

决策模块根据负载预测模块的预测结果和由缓冲区观测模块获取到的缓冲区使用量,综合确定如何调整端口状态和数目.

以当前节点 v_i 中对应下一跳节点为 v_j 的端口组 $G^{ij}=\{Port_{i1}^{ij}, Port_{i2}^{ij},\cdots, Port_{in_j}^{ij}\}$ 为例,阐述端口数量的确定.综合端口组速率等级集 $R^{ij}=\{R_1^{ij}, R_2^{ij}, R_3^{ij}\}$ 和端口组休眠唤醒状态集 $S^{ij}=\{S_{\text{a}}^{ij}, S_{\text{s}}, S_{\text{d}}\}$,得到端口组的合并状态集 $S^{ij}=\{S_{\text{a}_1}^{ij}, S_{\text{a}_2}^{ij}, S_{\text{a}3}^{ij}, S_{\text{s}}, S_{\text{d}}\}$.

在任一时隙 t,总端口数 N_t^{ij} 满足 $N_t^{ij}=N_{t,\text{a}}^{ij}+N_{t,\text{s}}^{ij}+N_{t,\text{d}}^{ij}$,其中,$N_{t,\text{a}}^{ij}$ 表示处于活动状态的端口数,$N_{t,\text{s}}^{ij}$ 表示处于浅层休眠状态的端口数:

$$N_{t,\text{s}}^{ij}=\lceil E_t^{ij}/R_3^{ij}\rceil, \tag{13}$$

其中,$N_{t,\text{d}}^{ij}$ 表示处于深层休眠状态的端口数,进一步地,$N_{t,\text{a}}^{ij}=N_{t,1}^{ij}+N_{t,2}^{ij}+N_{t,3}^{ij}$,其中 $N_{t,1}^{ij}, N_{t,2}^{ij}, N_{t,3}^{ij}$ 分别表示速率等级处于第 1,2,3 级的端口数.流量负载变化 Δ_t^{ij} 的计算:

$$\Delta_t^{ij}=(A_t^{ij}+B_t^{ij})-(A_{t-1}^{ij}+B_{t-1}^{ij}). \tag{14}$$

定义 7. 各状态端口间的数量转换矩阵 $\boldsymbol{N}_{p\times t}^{ij}$:

$$\boldsymbol{N}_{p\times t}^{ij}=\begin{array}{c|ccccc}& 3 & 2 & 1 & \text{s} & \text{d}\\ \hline 3 & — & N_{23}^{ij} & N_{13}^{ij} & N_{\text{s}3}^{ij} & N_{\text{d}3}^{ij}\\ 2 & N_{32}^{ij} & — & N_{12}^{ij} & N_{\text{s}2}^{ij} & N_{\text{d}2}^{ij}\\ 1 & N_{31}^{ij} & N_{21}^{ij} & — & N_{\text{s}1}^{ij} & N_{\text{d}1}^{ij}\\ \text{s} & N_{3\text{s}}^{ij} & N_{2\text{s}}^{ij} & N_{1\text{s}}^{ij} & — & N_{\text{ds}}^{ij}\\ \text{d} & N_{3\text{d}}^{ij} & N_{2\text{d}}^{ij} & N_{1\text{d}}^{ij} & N_{\text{sd}}^{ij} & —\end{array}. \tag{15}$$

矩阵元素 N_{xy}^{ij} 表示端口组 G^{ij} 中功率状态从列状态 S_x^{ij} 转换到行状态 S_y^{ij} 的端口数,$S_x^{ij}, S_y^{ij}\in\{S_{\text{a}_1}^{ij}, S_{\text{a}_2}^{ij}, S_{\text{a}3}^{ij}, S_{\text{s}}, S_{\text{d}}\}$.

在决策模块中嵌入应对不同流量负载变化趋势的端口数转换算法(详见算法 1 和算法 2),用于得出在任一预测时隙的数量转换矩阵,从而明确各状态端口间的数量变化情况,决策模块据此将端口数量调整信令分别发送给休眠唤醒控制模块和速率调节模块.

为了确保节能收益最大化,同时考虑到唤醒端口需要付出一定的转换代价(使端口复苏至正常工作状态需要一定的时延,并且该时延不可忽略[12])

而休眠端口是瞬间完成的(只需对端口停止供电即可),算法 1 和算法 2 分别基于不同的流量变化趋势权衡休眠/唤醒和速率调整所需付出的代价以及所能获得的节能收益.且在算法 1 和算法 2 中,为了书写简洁,我们将端口组 G^{ij} 中在时隙 $t-1$ 处于状态 S_x^{ij} 的全部端口切换到状态 S_y^{ij}($S_y^{ij}\neq S_x^{ij}$)而产生的容量变化量$|N_{t-1,x}^{ij}\times(R_y^{ij}-R_x^{ij})|$简记为 μ_{xy}^{ij}.

对于流量负载增加的情形,首先进行提升速率操作,即反复将当前处于最低速率等级的端口的速率等级进行逐级提升,直至能够满足流量增长需求为止;若所有开启端口的速率等级升至最高后仍不能满足流量增长需求,则进行唤醒端口操作,先唤醒浅层休眠端口,后唤醒深层休眠端口,并且为了尽可能减少唤醒端口数量,将休眠端口直接唤醒到速率等级 3(对于流量负载增加剩余量不足速率等级 3 的部分,按流量负载实际剩余量将休眠端口唤醒至速率等级 1 或速率等级 2).算法 1 的行①初始化各状态端口间的数量转换矩阵为零矩阵;行②③是对流量负载增幅较小(将端口速率升至 2 级即可满足)时的调整;行④~㉘是对流量负载增幅较大(所有开启端口速率提升至最高级仍无法满足)时的调整;行㉙~㉝是对流量负载增幅居中(通过提升速率操作可以满足)时的调整;行㊱返回更新后的各状态端口间的数量转换矩阵.

算法 1. 流量负载增加时各状态端口间的数量转换算法.

输入:A_t^{ij}, B_t^{ij}, E_t^{ij}, A_{t-1}^{ij}, B_{t-1}^{ij}, R_1^{ij}, R_2^{ij}, R_3^{ij}, $N_{t-1,1}^{ij}$, $N_{t-1,2}^{ij}$, $N_{t-1,3}^{ij}$, $N_{t-1,s}^{ij}$, $N_{t,s}^{ij}$;

输出:$\boldsymbol{N}_{p\times t}^{ij}$.

① 初始化 $\boldsymbol{N}_{p\times t}^{ij}\leftarrow\boldsymbol{0}$;

② if $\mu_{12}^{ij}\geqslant\Delta_t^{ij}\leftarrow(A_t^{ij}+B_t^{ij})-(A_{t-1}^{ij}+B_{t-1}^{ij})$ then

③ $N_{12}^{ij}\leftarrow\left\lceil\dfrac{\Delta_t^{ij}}{R_2^{ij}-R_1^{ij}}\right\rceil$;

④ else if $\Delta_t^{ij}>\mu_{23}^{ij}+\mu_{13}^{ij}$ then

⑤ $N_{13}^{ij}\leftarrow N_{t-1,1}^{ij}$, $N_{23}^{ij}\leftarrow N_{t-1,2}^{ij}$;

⑥ if $\mu_{s3}^{ij}\geqslant\Delta_t^{ij}-\mu_{23}^{ij}-\mu_{13}^{ij}$ then

⑦ $N_{s3}^{ij}\leftarrow\left\lfloor\dfrac{\Delta_t^{ij}-\mu_{23}^{ij}-\mu_{13}^{ij}}{R_3^{ij}}\right\rfloor$;

⑧ if $R_1^{ij}\geqslant\Delta_t^{ij}-\mu_{23}^{ij}-\mu_{13}^{ij}-R_3^{ij}\times N_{s3}^{ij}$ then

⑨ $N_{s1}^{ij}\leftarrow1$;

⑩ else if $R_2^{ij}<\Delta_t^{ij}-\mu_{23}^{ij}-\mu_{13}^{ij}-R_3^{ij}\times N_{s3}^{ij}$ then

⑪ $N_{s3}^{ij}\leftarrow N_{s3}^{ij}+1$;

⑫ else

⑬ $N_{s2}^{ij}\leftarrow1$;

⑭ end if

⑮ end if

⑯ $N_{ds}^{ij}\leftarrow N_{t,s}^{ij}-N_{t-1,s}^{ij}+N_{s1}^{ij}+N_{s2}^{ij}+N_{s3}^{ij}$;

⑰ else

⑱ $N_{s3}^{ij}\leftarrow N_{t-1,s}^{ij}$, $N_{d3}^{ij}\leftarrow\left\lfloor\dfrac{\Delta_t^{ij}-\mu_{23}^{ij}-\mu_{13}^{ij}-\mu_{s3}^{ij}}{R_3^{ij}}\right\rfloor$;

⑲ if $R_1^{ij}\geqslant\Delta_t^{ij}-\mu_{23}^{ij}-\mu_{13}^{ij}-\mu_{s3}^{ij}-R_3^{ij}\times N_{d3}^{ij}$ then

⑳ $N_{d1}^{ij}\leftarrow1$;

㉑ else if $R_2^{ij}<\Delta_t^{ij}-\mu_{23}^{ij}-\mu_{13}^{ij}-\mu_{s3}^{ij}-R_3^{ij}\times N_{d3}^{ij}$ then

㉒ $N_{d3}^{ij}\leftarrow N_{d3}^{ij}+1$;

㉓ else

㉔ $N_{d2}^{ij}\leftarrow1$;

㉕ end if

㉖ end if

㉗ $N_{ds}^{ij}\leftarrow N_{t,s}^{ij}$;

㉘ end if

㉙ else if $\Delta_t^{ij}>\mu_{23}^{ij}+\mu_{12}^{ij}$ then

㉚ $N_{23}^{ij}\leftarrow N_{t-1,2}^{ij}$, $N_{13}^{ij}\leftarrow\left\lceil\dfrac{\Delta_t^{ij}-\mu_{12}^{ij}}{R_3^{ij}-R_2^{ij}}\right\rceil-N_{23}^{ij}$, $N_{12}^{ij}\leftarrow N_{t-1,1}^{ij}-N_{13}^{ij}$;

㉛ else

㉜ $N_{12}^{ij}\leftarrow N_{t-1,1}^{ij}$, $N_{23}^{ij}\leftarrow\left\lceil\dfrac{\Delta_t^{ij}-\mu_{12}^{ij}}{R_3^{ij}-R_2^{ij}}\right\rceil$;

㉝ end if

㉞ end if

㉟ end if

㊱ return $\boldsymbol{N}_{p\times t}^{ij}$.

对于流量负载减少的情形,为了尽可能增加休眠端口数量,首先尽可能多地休眠当前处于低速率等级的端口,之后进一步通过降低速率,尽可能多地减少当前处于高速率等级的端口数量.算法 2 的行①初始化各状态端口间的数量转换矩阵为零矩阵;行②~⑮是对流量负载降幅较小(将处于速率等级 1 的端口休眠即可满足)时的调整;行⑯~㉙是对流

量负载降幅居中(将处于速率等级 1 和 2 的端口休眠即可满足)时的调整;行㉛～㊵是对流量负载降幅较大(将处于速率等级 1,2,3 的端口休眠才可满足)时的调整;行㊸返回更新后的各状态端口间的数量转换矩阵.

算法 2. 流量负载减小时各状态端口间的数量转换算法.

输入:A_t^{ij}, B_t^{ij}, E_t^{ij}, A_{t-1}^{ij}, B_{t-1}^{ij}, R_1^{ij}, R_2^{ij}, R_3^{ij}, $N_{t-1,1}^{ij}$, $N_{t-1,2}^{ij}$, $N_{t-1,3}^{ij}$, $N_{t,s}^{ij}$;

输出:$\boldsymbol{N}_{p\times t}^{ij}$.

① 初始化 $\boldsymbol{N}_{p\times t}^{ij}\leftarrow\boldsymbol{0}$;

② if $\mu_{1s}^{ij}\geqslant|\Delta_t^{ij}\leftarrow(A_t^{ij}+B_t^{ij})-(A_{t-1}^{ij}+B_{t-1}^{ij})|$ then

③ $N_{1s}^{ij}\leftarrow\left\lfloor\dfrac{|\Delta_t^{ij}|}{R_1^{ij}}\right\rfloor$;

④ if $\mu_{32}^{ij}<|\Delta_t^{ij}|-R_1^{ij}\times N_{1s}^{ij}\leqslant\mu_{21}^{ij}+\mu_{32}^{ij}$ then

⑤ $N_{32}^{ij}\leftarrow N_{t-1,3}^{ij}$,

$N_{21}^{ij}\leftarrow\left\lfloor\dfrac{|\Delta_t^{ij}|-R_1^{ij}\times N_{1s}^{ij}-\mu_{32}^{ij}}{R_2^{ij}-R_1^{ij}}\right\rfloor$;

⑥ else if $|\Delta_t^{ij}|-R_1^{ij}\times N_{1s}^{ij}>\mu_{31}^{ij}+\mu_{21}^{ij}$ then

⑦ $N_{31}^{ij}\leftarrow N_{t-1,3}^{ij}$, $N_{21}^{ij}\leftarrow N_{t-1,2}^{ij}$;

⑧ else if $\mu_{32}^{ij}\geqslant|\Delta_t^{ij}|-R_1^{ij}\times N_{1s}^{ij}$ then

⑨ $N_{32}^{ij}\leftarrow\left\lfloor\dfrac{|\Delta_t^{ij}|-R_1^{ij}\times N_{1s}^{ij}}{R_3^{ij}-R_2^{ij}}\right\rfloor$;

⑩ else

⑪ $N_{21}^{ij}\leftarrow N_{t-1,2}^{ij}$,

$N_{31}^{ij}\leftarrow\left\lfloor\dfrac{|\Delta_t^{ij}|-R_1^{ij}\times N_{1s}^{ij}-\mu_{32}^{ij}}{R_2^{ij}-R_1^{ij}}\right\rfloor-$

$N_{21}^{ij}, N_{32}^{ij}\leftarrow N_{t-1,3}^{ij}-N_{31}^{ij}$;

⑫ end if

⑬ end if

⑭ end if

⑮ $N_{sd}^{ij}\leftarrow N_{t-1,s}^{ij}+N_{1s}^{ij}-N_{t,s}^{ij}$;

⑯ else if $\mu_{1s}^{ij}+\mu_{2s}^{ij}\geqslant|\Delta_t^{ij}|$ then

⑰ $N_{1s}^{ij}\leftarrow N_{t-1,1}^{ij}$, $N_{2s}^{ij}\leftarrow\left\lfloor\dfrac{|\Delta_t^{ij}|-\mu_{1s}^{ij}}{R_2^{ij}}\right\rfloor$;

⑱ if $\mu_{32}^{ij}\leqslant|\Delta_t^{ij}|-\mu_{1s}^{ij}-R_2^{ij}\times N_{2s}^{ij}<\mu_{32}^{ij}+(N_{t-1,2}^{ij}-N_{2s}^{ij})\times(R_2^{ij}-R_1^{ij})$ then

⑲ $N_{32}^{ij}\leftarrow N_{t-1,3}^{ij}$,

$N_{21}^{ij}\leftarrow\left\lfloor\dfrac{|\Delta_t^{ij}|-\mu_{1s}^{ij}-R_2^{ij}\times N_{2s}^{ij}-\mu_{32}^{ij}}{R_2^{ij}-R_1^{ij}}\right\rfloor$;

⑳ else if $|\Delta_t^{ij}|-\mu_{1s}^{ij}-R_2^{ij}\times N_{2s}^{ij}\geqslant\mu_{31}^{ij}+(N_{t-1,2}^{ij}-N_{2s}^{ij})\times(R_2^{ij}-R_1^{ij})$ then

㉑ $N_{31}^{ij}\leftarrow N_{t-1,3}^{ij}$, $N_{21}^{ij}\leftarrow N_{t-1,2}^{ij}-N_{2s}^{ij}$;

㉒ else if $\mu_{32}^{ij}>|\Delta_t^{ij}|-\mu_{1s}^{ij}-R_2^{ij}\times N_{2s}^{ij}$ then

㉓ $N_{32}^{ij}\leftarrow\left\lfloor\dfrac{|\Delta_t^{ij}|-\mu_{1s}^{ij}-R_2^{ij}\times N_{2s}^{ij}}{R_3^{ij}-R_2^{ij}}\right\rfloor$;

㉔ else

㉕ $N_{21}^{ij}\leftarrow N_{t-1,2}^{ij}-N_{2s}^{ij}$, $N_{31}^{ij}\leftarrow\left\lfloor\dfrac{|\Delta_t^{ij}|-\mu_{1s}^{ij}-R_2^{ij}\times N_{2s}^{ij}-\mu_{32}^{ij}}{R_2^{ij}-R_1^{ij}}\right\rfloor-$

$N_{21}^{ij}, N_{32}^{ij}\leftarrow N_{t-1,3}^{ij}-N_{31}^{ij}$;

㉖ end if

㉗ end if

㉘ end if

㉙ $N_{sd}^{ij}\leftarrow N_{t-1,s}^{ij}+N_{1s}^{ij}+N_{2s}^{ij}-N_{t,s}^{ij}$;

㉚ else

㉛ $N_{1s}^{ij}\leftarrow N_{t-1,1}^{ij}$, $N_{2s}^{ij}\leftarrow N_{t-1,2}^{ij}$, $N_{3s}^{ij}\leftarrow\left\lfloor\dfrac{|\Delta_t^{ij}|-\mu_{1s}^{ij}-\mu_{2s}^{ij}}{R_3^{ij}}\right\rfloor$;

㉜ if $(R_3^{ij}-R_2^{ij})\times(N_{t-1,3}^{ij}-N_{3s}^{ij})\geqslant|\Delta_t^{ij}|-\mu_{1s}^{ij}-\mu_{2s}^{ij}-R_3^{ij}\times N_{3s}^{ij}$ then

㉝ $N_{32}^{ij}\leftarrow\left\lfloor\dfrac{|\Delta_t^{ij}|-\mu_{1s}^{ij}-\mu_{2s}^{ij}-R_3^{ij}\times N_{3s}^{ij}}{R_3^{ij}-R_2^{ij}}\right\rfloor$;

㉞ else if $(R_3^{ij}-R_1^{ij})\times(N_{t-1,3}^{ij}-N_{3s}^{ij})<|\Delta_t^{ij}|-\mu_{1s}^{ij}-\mu_{2s}^{ij}-R_3^{ij}\times N_{3s}^{ij}$ then

㉟ $N_{31}^{ij}\leftarrow N_{t-1,3}^{ij}-N_{3s}^{ij}$;

㊱ else

㊲ $N_{31}^{ij}\leftarrow\left\lfloor\dfrac{|\Delta_t^{ij}|-\mu_{1s}^{ij}-\mu_{2s}^{ij}-R_3^{ij}\times N_{3s}^{ij}}{R_3^{ij}-R_1^{ij}}\right\rfloor$,

$N_{32}^{ij}\leftarrow N_{t-1,3}^{ij}-N_{31}^{ij}$;

㊳ end if

㊴ end if

㊵ $N_{sd}^{ij}\leftarrow N_{t-1,s}^{ij}+N_{1s}^{ij}+N_{2s}^{ij}+N_{3s}^{ij}-N_{t,s}^{ij}$;

㊶ end if

㊷ end if

㊸ return $\boldsymbol{N}_{p\times t}^{ij}$.

3.4 状态控制模块

当需要唤醒/休眠某一类状态端口,或者需要调节某一类状态端口的速率时,为了获取最大的节能收益、减少不必要的 QoS 退化以及减少器件故障率、延长器件使用寿命,应该尽量避免出现端口短时间内在唤醒和休眠之间频繁切换以及在不同速率之

间频繁切换的情况.为此,本文分别在休眠唤醒子模块和速率调节子模块中定义相应的规则.

3.4.1 休眠唤醒子模块

定义 8. 休眠规则.假设 G^{ij} 在时隙 $t-1$ 开启的端口为 $Port^{ij}_{t-1,i_{a_1}}$, $Port^{ij}_{t-1,i_{a_2}}$, …, $Port^{ij}_{t-1,i_{a_m}}$, 对应器件状态表中记录的活动状态持续时间为 t_{a_1}, t_{a_2}, …, t_{a_m}, 则优先选取满足 $\{Port^{ij}_{t-1,i_{a_x}} \mid t_{a_x}=\max(t_{a_1}, t_{a_2}, \cdots, t_{a_m})\}$ 的端口进行休眠.

定义 9. 唤醒规则.假设 G^{ij} 在时隙 $t-1$ 浅层休眠的端口为 $Port^{ij}_{t-1,i_{s_1}}$, $Port^{ij}_{t-1,i_{s_2}}$, …, $Port^{ij}_{t-1,i_{s_n}}$, 深层休眠的端口为 $Port^{ij}_{t-1,i_{d_1}}$, $Port^{ij}_{t-1,i_{d_2}}$, …, $Port^{ij}_{t-1,i_{d_o}}$, 对应器件状态表中记录的浅层休眠持续时间为 t_{s_1}, t_{s_2}, …, t_{s_n}, 深层休眠持续时间为 t_{d_1}, t_{d_2}, …, t_{d_o}, 则对于浅层休眠优先选取满足 $\{Port^{ij}_{t-1,i_{s_x}} \mid t_{s_x}=\max(t_{s_1}, t_{s_2}, \cdots, t_{s_n})\}$ 的端口进行唤醒,对于深层休眠优先选取满足 $\{Port^{ij}_{t-1,i_{d_x}} \mid t_{d_x}=\max(t_{d_1}, t_{d_2}, \cdots, t_{d_o})\}$ 的端口进行唤醒.

休眠唤醒控制模块的工作分为 2 个阶段:

1) 接收来自决策模块的端口数量调整信令,确定哪几种状态端口需要进行休眠唤醒操作以及每种状态端口需要调整多少个.

2) 对于同种状态端口,依据休眠/唤醒规则,具体得出休眠/唤醒端口组中的哪些端口,并向调度引擎和相应的端口下发状态转换信令.

3.4.2 速率调节子模块

定义 10. 速率调节规则.假设 G^{ij} 在时隙 $t-1$ 处于速率等级 m 的端口为 $Port^{ij}_{t-1,i_{m_1}}$, $Port^{ij}_{t-1,i_{m_2}}$, …, $Port^{ij}_{t-1,i_{m_n}}$, 对应器件状态表中记录的持续时间分别为 t_{m_1}, t_{m_2}, …, t_{m_n}, 则在当前处于速率等级 m 的端口中优先选取满足 $\{Port^{ij}_{t-1,i_{m_x}} \mid t_{m_x}=\max(t_{m_1}, t_{m_2}, \cdots, t_{m_n})\}$ 的端口进行速率转换.

速率调节模块的工作分为 2 个阶段:

1) 接收来自决策模块的端口数量调整信令,确定哪几种速率端口需要进行速率调节操作以及每种速率端口需要调整多少个.

2) 对于同种速率端口,依据速率调节规则,具体得出调节端口组中的哪些端口的转发速率,并向调度引擎和相应的端口下发速率转换信令.

3.5 调度引擎

为了避免单一采用优先级排队调度算法(priority queuing, PQ)可能出现的低优先级队列"饿死"现象以及单一采用轮询类算法(如轮询调度(round robin, RR)、加权轮询调度(weighted round robin, WRR)、赤字加权轮询调度(deficit weighted round robin, DWRR))可能出现的无法对延迟、抖动和丢包率等敏感的关键分组提供优先级保证的问题,并考虑面向主干网节点和高速链路的应用场景(面向低速链路的公平排队(fair queuing, FQ)类调度算法不适用该情形),本文在调度引擎设计中提出了一种层次调度算法 PDL(PQ-DDWRR-LRMDTF).它的队列由 2 大类组成:1 个 PQ 队列和 m 个动态赤字加权轮询调度(dynamic deficit weighted round robin, DDWRR)队列,且 PQ 队列优先级高于所有 DDWRR 队列.只有当 PQ 队列中的所有分组都被调度完成,才会对 DDWRR 队列中的分组进行调度,并且两者之间是抢占式的,以此确保后面随时到来的紧急分组可以被高优先级调度.

定义 11. RMDT(remaining maximum delay time).对于流入节点的某个数据分组 $packet_k$,假设其对应的应用类型为 $type_k$,这种应用类型所允许的最大延迟为 $delay_{\max k}$,该数据分组转发至当前节点已经花费的延迟时间为 $delay_k$,则该数据分组所允许的剩余最大延迟时间 RMDT 为 $delay_{\max k}-delay_k$.

PQ 队列和每个 DDWRR 队列均采用最小 RMDT 优先(least RMDT first)对分组进行排序,RMDT 最小的分组排在所在队列的最前面,调度引擎在每个队列内部总是从队首分组开始执行调度,并且在 2 种队列的内部均采用非抢占式调度,即正在处理的分组即使没有新进来分组的 RMDT 小,也不必让位于新来的分组,而是直到当前分组处理完成后才处理新来的分组.这样设计的原因是为了避免同种队列内部的抢占式调度造成的频繁切换而形成的颠簸现象.在某一时刻,如果所有 DDWRR 队列中存在小于 PQ 队列队尾 RMDT 的分组,则按该分组 RMDT 大小将其移至 PQ 队列相应位置,以防止被"饿死".

依据国际电信联盟标准 ITU-T Y.1541[33] 中关于 QoS 类别的划分和定义以及 ITU-T Y.1221[34] 中关于业务合约(traffic contracts)的划分和定义,将分组分为 4 大类:会话类(如 VoIP(voice over Internet protocol), VTC(video teleconferencing), IPTV(Internet protocol television))、流类(如流媒体、VOD(video on demand))、交互类(如 Web 访问、数据库检索)和背景类(如 FTP(file transfer protocol)和 Email),不同类型分组需满足的 QoS 参数指标如表 4 所示:

Table 4 QoS Constraints Corresponding to Different Types of Packets

表 4 不同类型分组对应的 QoS 约束

QoS Parameter	Conversational Class	Streaming Class	Interactive Class	Background Class
Guaranteed Bandwidth	Yes	Yes	No	No
Delay	Rigorously Limited	Limited	Loose	Unlimited
Delay Jitter	Rigorously Limited	Limited	Loose	Unlimited
Low Error Rate	No	No	Yes	Yes
DSCP Value (PHB)	High (EF, AF4, AF3)	Relatively High (AF2)	Relatively Low (AF1)	Low (BE)

Note: PHB(per-hop-behavior), EF(expedited forwarding), AF(assured forwarding), BE(best effort).

当出现差分服务代码点(differentiated services code point, DSCP)的每跳行为 PHB 标识码为加急转发 EF、确保转发 AF4, AF3 或 AF2 的分组时,则由分类器将其放入 PQ 队列,优先处理;当新到分组的 PHB 为 AF1 或 BE 时,则由分类器依据其所属的端口组,将其放入对应的 DDWRR 队列.DDWRR 队列之间采用动态赤字加权轮询方式进行调度,每个队列维护一个赤字计数器(deficit counter, DC),DC 值表示每次允许调度该队列的字节总数.每次轮询调度时,首先初始化每个非空队列的 DC 值为本队列上次剩余的 DC 值和按当前各自权重计算所得的带宽之和,调度引擎依次访问所有非空队列,如果当前队列队首分组大小不大于 DC 值,则 DC 值减去此分组的大小,并由调度引擎发送该分组到输出端口,如此不断更新 DC 值,发送分组到输出端口,直到队首分组大小大于 DC 值为止,将此时剩余的 DC 值累加到当前队列下次轮询时使用,如果队列已空,则设置 DC 值为零,调度引擎移向下一非空队列进行轮询调度.

具体地,分配给第 j 个 DDWRR 队列的权重可以计算得出:

$$w_j=\frac{(\sum_{k=1}^{n_j}R_{i_kj_k}^{\text{link}}(t))-C_{PQ_j}(t)}{bw_i^{\text{total}}(t)-C_{\text{PQ}}(t)}\times 100\%,\quad(16)$$

其中,$C_{\text{PQ}}(t)$表示时刻 t 在 PQ 队列中所有分组的总大小,$C_{PQ_j}(t)$表示时刻 t 在 PQ 队列中隶属于端口组 G^{ij} 的分组总大小,$bw_i^{\text{total}}(t)$表示在时刻 t 节点 v_i 所有开启出链路的总带宽.$bw_i^{\text{total}}(t)$可计算得到:

$$bw_i^{\text{total}}(t)=\sum_j\sum_{k=1}^{n_j}R_{i_kj_k}^{\text{link}}(t),\quad(17)$$

其中,物理链路 $l_{i_kj_k}$ 的传输速率 $R_{i_kj_k}^{\text{link}}(t)$依据休眠唤醒控制模块和速率调节模块发送来的控制信令进行相应的动态调整.

综上所述,PDL 层次调度算法整体工作流程如图 7 所示,调度引擎通过采用 PDL 层次调度算法对进入当前节点的所有数据分组进行不同优先级的调度转发,可以确保最紧急的数据分组最先被转发,而非紧急的数据分组也不会被"饿死",这样使得不同应用类型的分组所需的 QoS 在节点处得以保证.

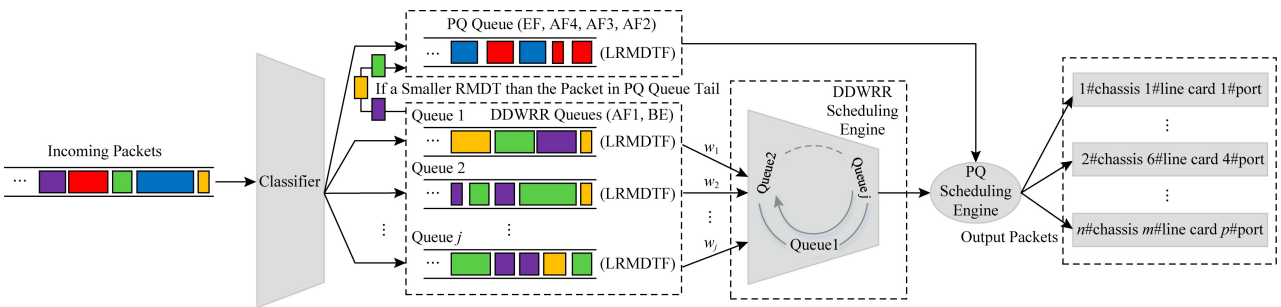


Fig. 7 Schematic diagram for PDL scheduling algorithm

图 7 PDL 调度算法示意图

4 仿真实现与综合测评

本文以功耗作为网络图权值在节点间采用 SPT(shortest path tree)算法路由每对流量需求,同时对所有网元都采用器件级功控机制.

在机制测评中,对于探索节能与性能之间的权衡以及探索过估计误差和低估计误差、计数窗口数目、环期预测和同期预测方式对预测准确度、节能和性能的影响,由于这些涉及的均是本机制的特有属

性,我们与未采用该机制时进行对比;而对于反映机制功效的比例性,我们选取文献[32]中提出的节能机制 BDTP(buffer dual-threshold policy)作为对比机制,该机制预设端口缓冲区占用率双阈值,对不同的流量负载动态调整链路传输速率实现节能.此外,为了对比公平起见,我们将 LPI 策略引入对比机制 BDTP,假设此时物理链路同样具有浅层休眠和深层休眠状态,当无分组传输时其可以立即进入浅层休眠状态,超过预设时间转为深层休眠状态.

4.1 仿真环境

所有方案的仿真环境为:

硬件配置 CPU 为 Intel Quad-Core i5-4590@3.30 GHz,RAM 为 4 GB(DDR3,1 600 MHz);操作系统为 Windows 7 professional 64 bits;开发平台为 Microsoft Visual Studio 2010;开发语言为 C++.

4.2 仿真数据集

仿真用例采用 3 个典型的主干网 CERNET2(20 个节点和 22 条链路)、GéANT (41 个节点和 65 条链路)和 INTERNET2(64 个节点和 78 条链路),拓扑结构如图 8 所示,特征属性如表 5 所示.

Table 5 Topological Structure Properties

表 5 拓扑属性

Property	CERNET2	GéANT	INTERNET2
# nodes	20	41	64
# links<10 Gbps	18	8	0
10 Gbps≤# links<100 Gbps	4	30	0
# links≥100 Gbps	0	27	78

流量数据集.对于 CERNET2 拓扑,通过教育网 Aladdin 网管中心信息平台① 提取在观测期间网络中所有节点间的进出流量监测数据,得到节点间流量分布及交互情况.对于 GéANT 拓扑和 INTERNET2 拓扑,采用文献[35]中提供的流量数据.

环期预测数据集.由于每周对应工作日之间的流量特征具有相似性,本文选取从 2016-04-10—2016-04-16 期间每天 24 h 流量轨迹.

同期预测数据集.选取以上 3 个拓扑从 2016-02-28—2016-09-24 每天 4:00—5:00,14:00—15:00,21:00—22:00 这 3 个时间段的流量轨迹.

需要指出的是,除了在 4.4.5 节与环期预测进行比较时我们使用同期预测数据集之外,其余各处仿真均使用环期预测数据集.而且除了 4.4.4 节之外,其余各处仿真均使用 8 个计数器.

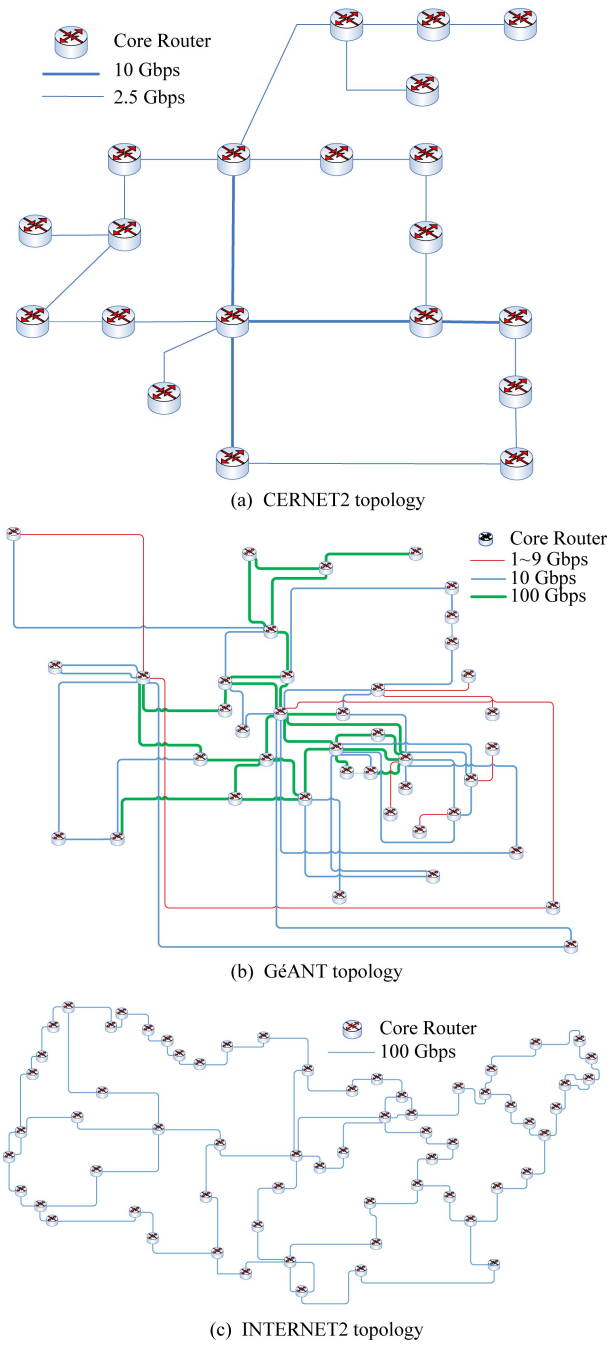


Fig. 8 Topology use-case diagram

图 8 拓扑用例

4.3 参数的设置和确定

参考思科 12000 系列路由器② 设置仿真中使用的功耗和器件配置等参数,如表 6 所示:

① 阿拉丁网络信息管理系统,http://219.243.208.6/snmp/index.php

② Cisco XR 12000 Series and Cisco 12000 Series Routers. http://www.cisco.com/c/en/us/products/routers/12000-series-routers/datasheet-listing.html

Table 6 Simulation Parameters Setting

表 6 仿真参数设置

Parameters	Values
Base Power Consumption of a Chassis/W	100
Power Consumption of the Master Engine/W	356
Power Consumption of the Scheduling Engine/W	420
Power Consumption of a Replication Engine/W	100
Power Consumption of a Forwarding Engine/W	446
Power Consumption of the Power Amplifier/W	4.8
Power Consumption of an In-Line Amplifier/W	10
Power Consumption of a Regenerator/W	26
Power Consumption of the Pre-Amplifier/W	4.8
Maximum Power Consumption of a Port/W	100
Base Power Consumption of a Port/W	40
The Number of Chassis in a Node	4
The Number of Line Cards in a Chassis	4
The Number of Ports in a Line Card	4
Bundled Size n_{ij}	5
The Lower Threshold of LU in CPCM	$0.3\times\psi_{ij}^{\text{link}}$
The Upper Threshold of LU in CPCM	$0.7\times\psi_{ij}^{\text{link}}$
Link Rate at Level 1 in CPCM	$0.3\times\psi_{ij}^{\text{link}}$
Link Rate at Level 2 in CPCM	$0.7\times\psi_{ij}^{\text{link}}$
Link Rate at Level 3 in CPCM	$1\times\psi_{ij}^{\text{link}}$
The Upper Threshold of Port Buffer in BDTP	$0.3\times\lambda_{ij}^{\text{link}}$
The Lower Threshold of Port Buffer in BDTP	$0.7\times\lambda_{ij}^{\text{link}}$
Low Link Rate in BDTP	$0.3\times\psi_{ij}^{\text{link}}$
High Link Rate in BDTP	$1\times\psi_{ij}^{\text{link}}$
Holding Time from S_s to S_d in BDTP/min	5
Correlation Coefficient between Power and Rate	3
Power Proportion Accommodation Coefficient	0.01

对于预测参数 α,β,I_{pre},I_{obs}的取值需要根据不同的流量负载状况进行调整,本文在 3 个典型的代表低负载(4:00—5:00)、中负载(14:00—15:00)和高负载(21:00—22:00)特征的轨迹下对这些参数值进行调整.在这些轨迹下,分别通过式(8)或式(10)来获取参数 α 的值,之后分别通过式(9)或式(11)来获取参数 β 的值.流量增加时,对这 2 个参数值向下取整,反之向上取整.根据不同主干网中各节点的历史流量日志和器件部署情况设置 I_{pre} 值,而 I_{obs} 取值的标准是在保证没有分组丢失的条件下选取使当前节点功耗最小化时对应的 I_{pre} 值的整数倍.

按上述方法,CERNET2 拓扑中的沈阳节点、GéANT 拓扑中的丹麦节点以及 INTERNET2 拓扑中的匹兹堡节点在不同流量等级(traffic level, TL)的负载轨迹下的预测参数设置如表 7 所示:

Table 7 Prediction Parameters Setting

表 7 预测参数设置

Node(Topology)	TL	α	β	I_{pre}/ms	I_{obs}/ms
Shenyang (CERNET2)	Low Load	5	5	3.4	125.8
	Mid Load	3	2	1.3	14.3
	High Load	2	2	0.4	5.6
Denmark (GéANT)	Low Load	4	4	2.4	43.2
	Mid Load	2	3	0.8	6.4
	High Load	1	2	0.3	1.8
Pittsburgh (INTERNET2)	Low Load	3	4	1.3	10.4
	Mid Load	2	2	0.3	0.9
	High Load	1	2	0.1	0.1

4.4 机制测评

采用定义 12~14 中的功效、缓冲区占用率和最差延迟作为参数度量指标评价本文提出的器件级功控机制 CPCM.

定义 12. 功效(power efficiency, PE).将当前所有休眠器件以及低速率器件共同节省的总功耗与所有器件都活动且运行在最大速率时的总功耗的比值称为功效.功效越高越节能.

当预测开启的处于各种速率的器件不能满足实际到来的流量时,即实际入口负载到达速率大于节点处理输出速率,则需要使用缓冲区来容纳未被及时处理的分组,从而增加缓冲区的使用,因此除了需要对功效进行度量之外,还需要对缓冲区的使用情况进行度量.

定义 13. 缓冲区占用率.将当前已经使用的缓冲区大小占缓冲区总容量的百分比称为缓冲区占用率.缓冲区占用率越小,分组的等待延迟越小.

定义 14. 最差延迟.将数据分组进出缓冲区的最大时间间隔称为最差延迟,用来度量负载延迟.

此外,本文还讨论采用不同数目的计数窗口以及环期预测和同期预测对预测准确性的影响,以及过估计误差和低估计误差对器件级功控机制的影响.

4.4.1 比例性

比例性可以反映功效随负载变化的紧密程度,以此评价设计机制的节能潜力.最理想的情况是两者完全成正比例线性变化,此时节能收益最大化.

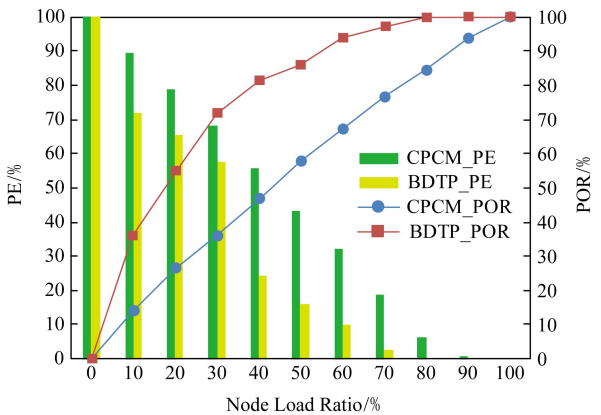
定义 15. 节点负载率.将节点流量负载与节点容量的百分比称为节点负载率.

定义 16. 端口开启率(port opening ratio, POR).

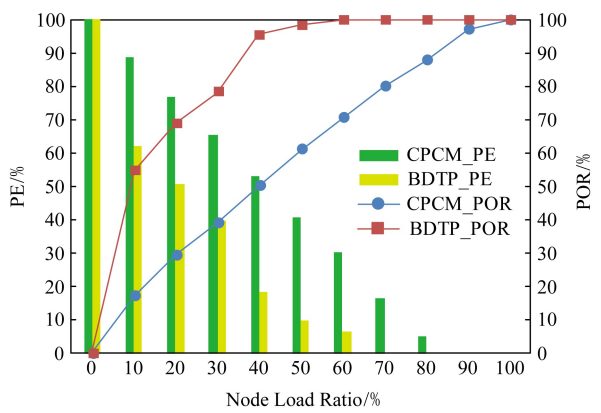
将节点中开启端口数与总端口数的百分比称为端口开启率.

定义 17. 比例性.将端口开启率随节点负载率动态变化的线性程度称为比例性.

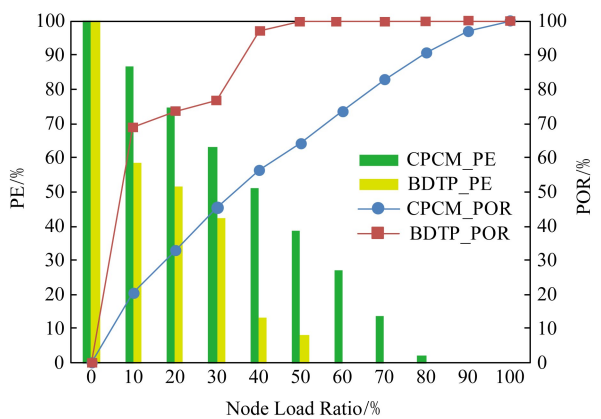
从图 9 可以看出,在 CPCM 机制下,节点功效和端口开启率都紧随节点负载率的变化而趋近线性比例变化,尤其对于规模较小的拓扑比例性更好.而且,通过观察功效变化情况可以发现,CPCM 机制



(a) Shenyang (CERNET2)



(b) Denmark (GéANT)



(c) Pittsburgh (INTERNET2)

Fig. 9 Proportionality test

图 9 比例性测试

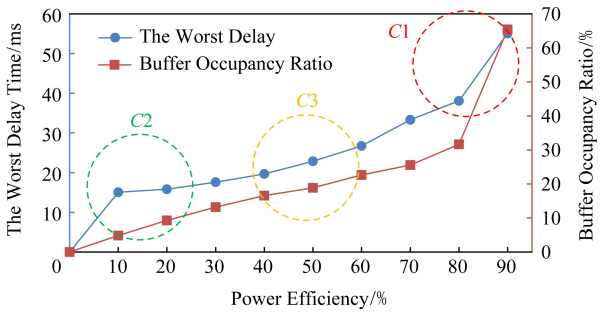
的引入能够节省大量的功耗,甚至当节点负载率高达 70%时,仍能维持 14%～20%的功效.由此可见,CPCM 机制能够依据流量负载变化非常有效地控制功效和端口状态.而在 BDTP 机制下,节点功效在低负载和高负载时差异很大,近似阶跃式变化;端口开启率始终较高,对节点负载率的变化不敏感,并且拓扑结构越复杂的节点越明显.2 种机制的显著差异主要来源于前者对捆绑链路中全部物理链路规划端口的开关和速率的调整,较大限度地保证了所开即所需,而后者捆绑链路各端口之间是相互独立的,没有聚合流量,于是开启了多余的端口或者使用了较大而不必要的高速率,导致功效显著降低.

4.4.2 节能与性能之间的权衡

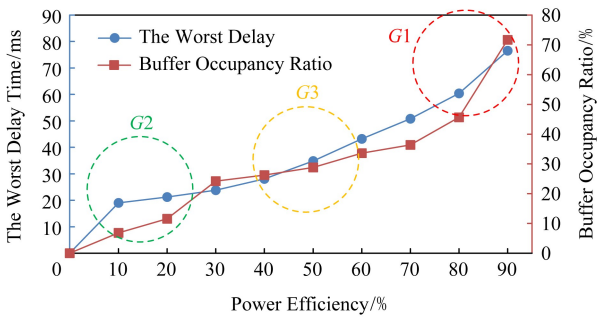
权衡是既要保证节能效果又要保障一定的运行性能而不得不在两者之间做出的折中考量.本文关注功效与最差延迟之间的权衡以及功效与缓冲区占用率之间的权衡.

图 10 显示了在不同的缓冲区使用量警戒阈值 ζ 下的功效与最差延迟权衡分布以及功效与缓冲区占用率权衡分布.可以看出,当设置最差延迟为某个确定值时,能够获得在此限制下的最大功效;反之,当需要实现特定级别的功效时,能够获得此时的最坏延迟.例如,观察区域 $C1$,$G1$,$I1$ 中 $\zeta \geqslant 0.8$ 的点,功效达到 80%以上,但最差延迟至少 30 ms,使用这样的 ζ 值能够以增大最差延迟为代价换取功效最大化.这种情况适用于在注重节能的环境下进行部署,仅需满足最低的延迟需求即可.又如,在区域 $C2$,$G2$,$I2$ 中的 $\zeta \leqslant 0.2$ 的点,此时缓冲区频繁刷新,与区域 $C1$,$G1$,$I1$ 相比,功效降低超过 50%,但最差延迟减小可达 2/3,即最差延迟被最大程度地减小,但功效被最小化.这种情况适用于对负载延迟有苛刻约束而对节能不敏感的环境.最后,在区域 $C3$,$G3$,$I3$ 中的 $\zeta \in [0.6, 0.7]$ 的点,这样的点是权衡后的“甜蜜点”,既没有太糟糕的最差延迟(比第 1 种情况少 50%),也能获得一定的节能效果(比第 1 种情况少 30%).因此,本文通过选择适当的控制参数 ζ 以满足在某一环境下的部署需求.

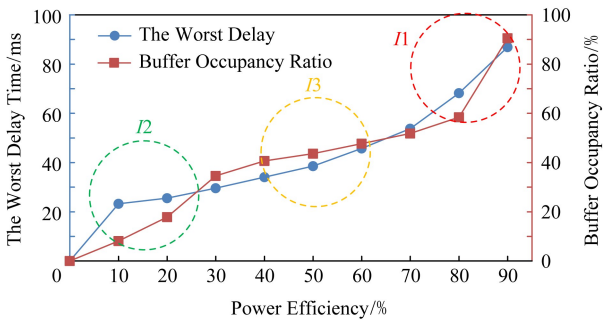
从图 10 还可以看出,如果使用增加功效的参数,则缓冲区占用率将增长,反之功效将减小.这是因为缓冲区中数据分组的累积会导致最差延迟的增长.当选择减少最差延迟的参数,缓冲区占用率也将减少,因此可以通过恰当地使用缓冲区减小最差延迟.



(a) Shenyang (CERNET2)



(b) Denmark (GéANT)



(c) Pittsburgh (INTERNET2)

Fig. 10 Tradeoff between energy saving and performance

图 10 节能与性能之间的权衡

4.4.3 过估计误差和低估计误差的影响

当负载预测模块得到的预测性能偏离实际需要的性能时会产生过估计或低估计.

如果是过估计,则意味着投入了比实际需要更多的器件用于处理入口负载,降低了功效,造成能源的浪费,但对节点的处理性能没有影响.

如果是低估计,则意味着投入了比实际需要更少的器件用于处理入口负载.这会导致数据分组在缓冲区中的不断累积从而导致负载最差延迟的增长,甚至可能导致数据分组累积过多而使缓冲区溢出从而导致分组丢失,严重影响节点的处理性能.

图 10 中的区域 $C1$,$G1$,$I1$ 中的点可以看作是发生低估计时的状况,尽管获得了功效收益,但是最差延迟的增长恶化了节点的性能.区域 $C2$,$G2$,$I2$ 中的点可以看作是发生过估计的状况,牺牲了功效换取了负载最差延迟的减小.对于运行在主干网中的核心节点来说,与过估计相比,低估计的发生更需要被避免.

4.4.4 计数窗口数目对预测的影响

定义 18. 端口数平均误差.将当前开启端口数 N_{active} 和实际所需开启端口数 N_{actual} 间的平均误差称为端口数平均误差,记为 χ,计算为

$$\chi=\frac{1}{n_{\text{pre}}}\sum(N_{\text{active}}-N_{\text{actual}}), \tag{18}$$

χ 能够反映过估计/低估计的趋势.

定义 19. 端口数均方根误差.将当前开启端口数 N_{active} 和实际所需开启端口数 N_{actual} 之间的均方根误差称为端口数均方根误差,记为 δ,计算为

$$\delta=\sqrt{\frac{1}{n_{\text{pre}}}\sum(N_{\text{active}}-N_{\text{actual}})^2}, \tag{19}$$

δ 能够反映预测值追踪实际负载的准确程度.其中,n_{pre} 是预测的总次数.

记处于过估计状态的端口数均方根误差为 δ_{over},计算为

$$\delta_{\text{over}}=\sqrt{\frac{1}{n_{\text{over}}}\sum_{N_{\text{active}}>N_{\text{actual}}}(N_{\text{active}}-N_{\text{actual}})^2}\times\frac{n_{\text{over}}}{n_{\text{pre}}}, \tag{20}$$

δ_{over} 值越小,表示过估计的影响越小,可以实现更高的功效控制.

记处于低估计状态的端口数均方根误差为 δ_{under},计算为

$$\delta_{\text{under}}=\sqrt{\frac{1}{n_{\text{under}}}\sum_{N_{\text{active}}<N_{\text{actual}}}(N_{\text{active}}-N_{\text{actual}})^2}\times\frac{n_{\text{under}}}{n_{\text{pre}}}, \tag{21}$$

δ_{under} 值越小,意味着低估计的影响越小,可以实现更小的负载延迟.其中,n_{over} 和 n_{under} 分别是过估计和低估计发生次数.

图 11 展示了节点预测模块采用不同数目的计数器时的预测准确度比较,在 3 个拓扑中,在高负载情形下 δ_{under} 均大于 δ_{over} 而占据主导地位,在低负载情形下 δ_{over} 都大于 δ_{under} 而占据主导地位,在所有情形下平均误差 χ 都趋于 0.上述说明,计数器数目不影响预测趋势.与单个计数器相比,当使用 8 个计数器时,所有的 δ_{over} 和 δ_{under} 均大幅降低,这是由于增加计数器数目可以减少流量负载频繁波动对预测准确度的影响.

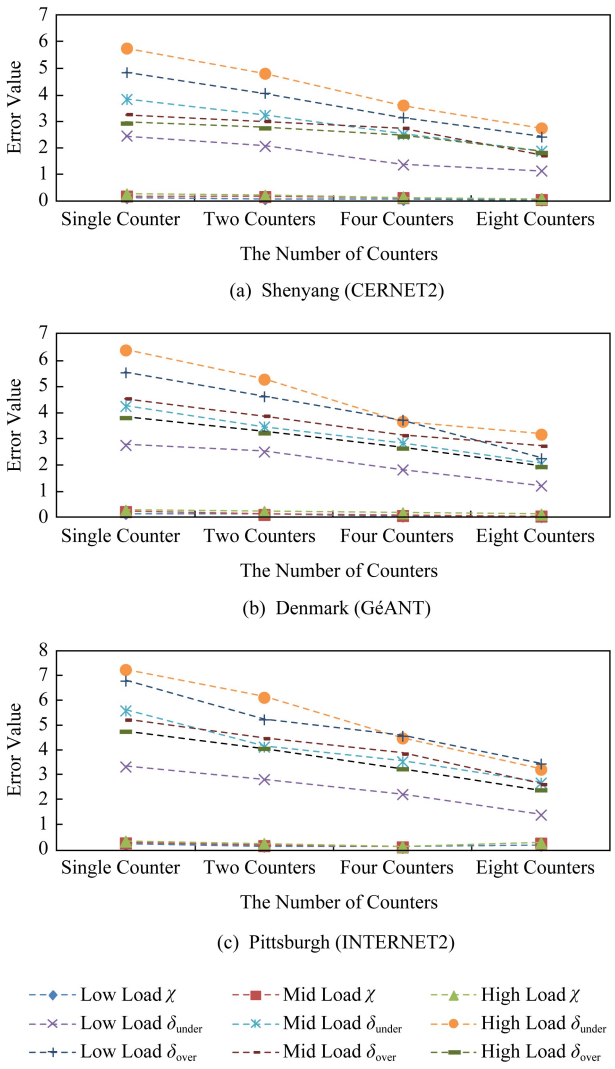


Fig. 11 Comparison on prediction accuracy among different number of counters

图 11 不同数目计数器间的预测准确度比较

4.4.5 环期预测和同期预测

记环期预测、广义同期预测和狭义同期预测下的端口数均方根误差分别为 δ_{CP}, δ_{GSP}, δ_{SSP}, 均通过式(19)计算得到.

图 12 展示了当选取不同的预测时隙序列时对预测准确度的影响.可以看出,低负载时,采用环期预测的均方根误差明显小于广义同期预测和狭义同期预测;高负载时,大多数情况下环期预测的准确度与广义同期预测较为接近.这是因为环期预测对网络流量低平稳高抖动的日夜规律依赖性更大,呈现低负载预测准确度高而高负载预测准确率低的特征.在所有负载情形下,对于时间跨度敏感的流量负载,如图 12(b)所示,狭义同期预测准确度明显下降;而对于时间跨度不敏感的流量负载,如图 12(a)所示,狭义同期预测的准确度超过了广义同期预测.这主要是由于在前者情形下,与广义同期预测相比,狭义同期预测的流量负载时间序列抖动偏差变大,导致其预测误差增大;而在后者情形下,狭义同期预测的由不同周相同工作日相同时隙组成的流量时间序列比广义同期预测的由不同工作日相同时隙组成的流量时间序列更好地保留了流量相似性,因而具有更小的抖动,提高了预测准确度.

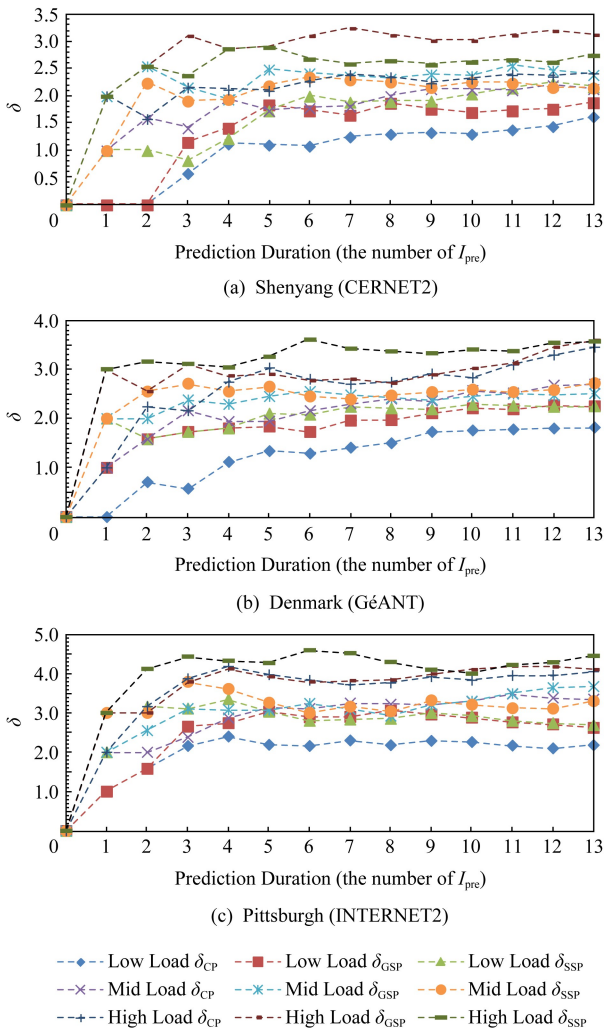


Fig. 12 Accuracy comparison among CP, GSP and SSP

图 12 环期预测和同期预测的准确度比较

5 总 结

本文面向主干网提出了一种细粒度器件级功控机制,该机制依据流量负载大小动态调整网元器件功率状态实现了节能,且在节能的同时兼顾用户 QoS 需求的满足,在提供 QoS 保证的前提下尽可能最大化节能收益.此外,本文还考虑了在功效和负载

最差延迟之间以及在功效和缓冲区占用率之间的权衡问题,同时揭示了预测过估计和预测低估计、环期预测和同期预测以及扩展入口负载计数器的计数窗口对器件级功控机制的影响.仿真结果表明:本文提出的器件级功控机制能够细粒度、动态和比例性地控制各网元功耗.

由于本文提出的机制是增补式的,并不需要"大刀阔斧"式地更换目前现有的基础设施、网络架构和网络协议等而进行各个层面的全新设计,在实际部署中仅需要适当扩充和增添少许网络功能模块以及在现有网络协议基础上作相应修改即可,且机制依据的 DPM 和 DVFS 等功率调节技术以及 ALR 和 LPI 等节能策略也较为成熟,因此本文提出的机制具有实际可行性和可能性.在实际部署本文提出机制时,网络运营商需要根据不同应用需求场景通过参考我们的实验结果选取适当的缓冲区使用量警戒阈值使得节能与性能之间达到权衡或者侧重于二者之一,同时根据不同的拓扑结构和历史流量分布情况选择合适的预测时隙获取方式和计数器数目,严格避免低估计的发生,也尽量避免过估计的发生.

我们未来的工作将着重于探索端口速率调节方法对基于链路状态的开放最短路径优先路由算法等网络级路由机制的影响以及两者之间的协同交互,通过综合运用器件级和网络级节能机制,使网络获取更大的节能收益.

参 考 文 献

[1] Cisco. Cisco annual Internet report (2018—2023) white paper, document ID: 1581105709073148 [R]. San Jose, CA: Cisco Systems, Inc, 2020

[2] Lorincz J, Capone A, Wu Jinsong. Greener, energy-efficient and sustainable networks: State-of-the-art and new trends [J]. Sensors, 2019, 19(22): No.4864

[3] The Climate Group. SMART2020: Enabling the low carbon economy in the information age [R/OL]. Brussels, Belgium: Global eSustainability Initiative (GeSI), 2008 [2019-11-01]. https://www.theclimategroup.org/sites/default/files/archive/files/Smart2020Report.pdf

[4] The Climate Group. SMARTer2020: The role of ICT in driving a sustainable future [R/OL]. Brussels, Belgium: Global eSustainability Initiative (GeSI), 2012 [2019-11-01]. http://www.truevaluemetrics.org/DBpdfs/ClimateChange/Mr-Neves-Presentation.pdf

[5] The Climate Group. SMARTer2030: Report ICT solutions for 21st century challenges [R/OL]. Brussels, Belgium: Global eSustainability Initiative (GeSI), 2015 [2019-11-01]. http://environmentportal.in/files/file/Full_report.pdf

[6] Addis B, Capone A, Carello G, et al. Energy management in communication networks: A journey through modeling and optimization glasses [J]. Computer Communication, 2016, 91/92: 76-94

[7] Ryan K. Is the Internet's carbon footprint becoming a dangerous concern? [R/OL]. Lincoln, United Kingdom: Blue & Green Tomorrow, 2018 [2019-11-01]. http://blueandgreentomorrow.com/environment/internet-carbon-footprint-becoming-dangerous-concern/

[8] Kastell K A. Challenges and incentives for the convergence of backbone and last mile of mobile and fixed line communication networks [C] //Proc of the 6th IEEE Int Conf on Management of Innovation & Technology (ICMIT). Piscataway, NJ: IEEE, 2012: 663-667

[9] Coudert D, Kodjo A, Phan T K. Robust energy-aware routing with redundancy elimination [J]. Computers & Operations Research, 2015, 64: 71-85

[10] Chen Ruobin, Wang Xingwei, Ma Lianbo, et al. An energy-efficient routing algorithm in green networks [J]. Chinese Journal of Computers, 2018, 41(11): 2612-2623 (in Chinese)
(陈若宾,王兴伟,马连博,等.绿色主干网络中一种高效的节能路由算法[J].计算机学报,2018,41(11):2612-2623)

[11] Zhang Jinhong, Wang Xingwei, Huang Min, et al. A distributed topology management scheme for energy saving in green Internet [J]. Chinese Journal of Computers, 2017, 40(7): 1517-1529 (in Chinese)
(张金宏,王兴伟,黄敏,等.绿色互联网中面向节能的分布式拓扑管理机制[J].计算机学报,2017,40(7):1517-1529)

[12] Lin Chuang, Tian Yuan, Yao Min. Green network and green evaluation: Mechanism, modeling and evaluation [J]. Chinese Journal of Computers, 2011, 34(4): 593-612 (in Chinese)
(林闯,田源,姚敏.绿色网络和绿色评价:节能机制、模型和评价[J].计算机学报,2011,34(4):593-612)

[13] Rihab M, Lamia C, Bernard C. Energy saving in carrier-grade networks: A survey [J]. Computer Standards & Interfaces, 2018, 55: 8-26

[14] Lin Gongqi, Soh S, Chin K W. Energy-aware traffic engineering with reliability constraint [J]. Computer Communications, 2015, 57: 115-128

[15] Wang Xingwei, Zhang Jinhong, Huang Min, et al. A green intelligent routing algorithm supporting flexible QoS for many-to-many multicast [J]. Computer Networks, 2017, 126: 229-245

[16] Caârpa R, Assunção M D, Glück O, et al. Responsive algorithms for handling load surges and switching links on in green networks [C] //Proc of IEEE Int Conf on Communications (ICC 2016). Piscataway, NJ: IEEE, 2016: 5369-5375

[17] Yousef A H, Fahmy A F, Mohamed H K. On the use of predictive analytics techniques for network elements failure prediction in telecom operators [C] //Proc of the 13th Int Computer Engineering Conf (ICENCO). Piscataway, NJ: IEEE, 2017: 250-255

[18] Fu Wenliang, Song Tian. A frequency adjustment architecture for energy efficient router [C] //Proc of ACM SIGCOMM 2012. New York: ACM, 2012: 107-108

[19] Takeshita H, Ishii D, Yamanaka N. High-energy efficient layer-3 network architecture based on solitary universal cloud router and optical aggregation network [C] //Proc of the 9th Int Conf on Optical Internet (COIN 2010). Piscataway, NJ: IEEE, 2010: 138-140

[20] Basu S, Mukherjee A, Klivansky S. Time series models for internet traffic [C] //Proc of the 15th Annual Joint Conf of the IEEE Computer Societies: Networking the Next Generation (INFOCOM'6), vol 2. Los Alamitos, CA: IEEE Computer Society, 1996: 611-620

[21] Doverspike R D, Ramakrishnan K K, Chase C. Structural overview of ISP networks [M] //Guide to Reliable Internet Services and Applications. Berlin: Springer, 2010: 19-93

[22] Nordman B, Christensen K. IEEE 802.3 Tutorial Reducing the Energy Consumption of Network Devices [S]. Piscataway, NJ: IEEE, 2005

[23] Patel-Predd P. Update: Energy-efficient Ethernet [J]. IEEE Spectrum, 2008, 45(5): 13

[24] Christensen K, Reviriego P, Nordman B, et al. IEEE 802.3az: The road to energy efficient Ethernet [J]. IEEE Communications Magazine, 2010, 48(11): 50-56

[25] IEEE P802.3az Energy Efficient Ethernet Task Force. IEEE Standard 802.3az—2010 Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications [S]. Piscataway, NJ: IEEE, 2010

[26] Zhao Xuejiao, Shen Gangxiang, Shao Weidong, et al. Energy-minimized design and operation of IP over WDM networks with traffic-aware adaptive router card clock frequency [J]. IEEE Journal on Selected Areas in Communications, 2015, 33(12): 2847-2862

[27] James C, Carlsson N. Green domino incentives: Impact of energy-aware adaptive link rate policies in routers [C] //Proc of the 6th ACM/SPEC Int Conf on Performance Engineering. New York: ACM, 2015: 211-221

[28] Kai Yi, Wang Yi, Liu Bin. GreenRouter: Reducing power by innovating router's architecture [J]. IEEE Computer Architecture Letters, 2013, 12(2): 51-54

[29] Carlsson N. Optimized eeeBond: Energy efficiency with non-proportional router network interfaces [C] //Proc of the 7th ACM/SPEC on Int Conf on Performance Engineering. New York: ACM, 2016: 215-223

[30] Nam P N, Thanh N H, Trong V Q, et al. A new power profiling method and power scaling mechanism for energy-aware NetFPGA gigabit router [J]. Computer Networks, 2015, 78: 4-25

[31] Lin Gongqi, Soh S, Chin K W, et al. Efficient heuristics for energy-aware routing in networks with bundled links [J]. Computer Networks, 2013, 57(8): 1774-1788

[32] Gunaratne C, Christensen K, Nordman B, et al. Reducing the energy consumption of Ethernet with adaptive link rate (ALR)[J]. IEEE Transactions on Computers, 2008, 57(4): 448-461

[33] ITU-T. ITU-T Y.1541 Network Performance Objectives for IP-based Services [S]. Geneva: ITU, 2011

[34] ITU-T. ITU-T Y. 1221 Traffic Control and Congestion Control in IP Based Networks [S]. Geneva: ITU, 2010

[35] Orlowski S, Wessäly R, Pióro M, et al. SNDlib 1.0-survivable network design library [J]. Networks, 2010, 55(3): 276-286

Zhang Jinhong, born in 1982. PhD candidate. His main research interests include network energy-saving mechanism, green routing, network topology management, etc.

Wang Xingwei, born in 1968. PhD, professor, PhD supervisor. Senior member of CCF. His main research interests include future Internet, cloud computing and cyberspace security, etc.

Yi Bo, born in 1988. PhD, lecturer. His main research interests include network function virtualization and service function chain, etc.

Huang Min, born in 1968. PhD, professor, PhD supervisor. Her main research interests include modeling and optimization for logistics and supply chain management, etc.