高性能 LTE 终端通信协议栈缓存机制

于欣峰[1,2] 梁利平[1*]

[1](中国科学院微电子研究所 北京 100029)

[2](中国科学院大学电子与通信工程学院 北京 100029)

摘 要 针对 LTE 终端通信协议栈的处理过程中 Cache 缓存方式效率差,提出一种可以提高处理速度降低延时的数据缓存方式。通过利用便签式存储器(Scratch-Pad Memory,SPM)和直接内存存取方式(Direct Memory Access,DMA)的性能优势,将协议栈处理的过程中需要与处理器频繁交互的数据缓存到 SPM 中,实现在 SPM 中完成数据的包头压缩/解压、加密/解密和重组等协议功能处理,并通过 DMA 方式实现 SPM 与主存之间的数据搬运,减少处理器对大块连续数据的搬运,提高处理效率。该方式避免了 CPU 在通过 Cache 缓存方式进行数据交互容易发生缓存不命中的问题,从而提高了协议栈的数据处理效率。经过理论分析及实验对比,结果表明,在 LTE 协议栈处理过程中,采用 SPM 与 DMA 结合的数据缓存机制,相比 Cache 缓存方式可以使整体性能至少提升 12.65%。

关键词 LTE 协议栈 缓存空间管理 DMA 便签式缓存

中图分类号 TP3 TN92 **文献标志码** A DOI:10.3969/j.issn.1000-386x.2020.03.017

CACHE MECHANISM OF HIGH PERFORMANCE LTE TERMINAL COMMUNICATION PROTOCOL STACK

Yu Xinfeng[1,2] Liang Liping[1*]

[1](*Institute of Microelectronics of The Chinese Academy of Science*, *Beijing 100029*, *China*)

[2](*School of Electrical*, *Electronics and Communication Engineering*, *University of Chinese Academy of Sciences*, *Beijing 100029*, *China*)

Abstract Aiming at the inefficiency of the Cache buffering in the processing of the LTE terminal communication protocol stack, this paper proposes a data buffering method which can improve the processing speed and reduce the delay. By using the performance advantages of SPM and DMA, the data which needed to exchange with the processor frequently in the process of protocol stack processing was cached in SPM. The protocol function processing such as packet head compression/decompression, encryption/ decryption and reorganization were realized in SPM. The data transfer between SPM and main memory was realized by DMA, which could reduce the transfer of large continuous data and improve the processing efficiency. This method avoids the problem that CPU is prone to cache miss in data exchange through cache, and improves the data processing efficiency of protocol stack. After theoretical analysis and experimental comparison, the results show that in the process of LTE protocol stack, the data cache mechanism combined with SPM and DMA can improve the overall performance by at least 12.65% compared with the Cache mode.

Keywords LTE Protocol stack Memory management DMA SPM

收稿日期:2019-04-11。国家自然科学基金项目(61471354)。**于欣峰**,硕士生,主研领域:无线通信,数字集成电路设计。**梁利平**,研究员。

0　引　言

移动互联网和物联网时代的来临,给无线通信应用带来了丰富的业务场景,也对无线通信服务质量提出了更高的要求:超高的数据传输速率、极低的交互时延和良好的移动特性等[1]。更高的性能指标要求意味着协议栈在数据处理能力上面临更大的挑战,协议栈系统需要在极短的时间内对海量的数据进行处理,否则就会影响系统的吞吐量。LTE 协议栈作为数据处理的重要技术模块和无线通信技术演进过程中的重要组成部分,针对如何提高协议栈数据处理速度、提升协议栈上下行吞吐率、降低通信时延方面的技术探究,具有非常的重要实际意义[2]。

无线通信系统中,数据流信息按照 LTE 规定经过三个层次处理:L1 物理层(Physical Layer,PHY)、L2 数据链路层和 L3 应用网络层。其中 L2 层数据链路层又被称为协议栈[3]。协议栈根据不同的业务功能,可以分为控制平面和用户平面;协议栈由三个子层组成,分别为:分组数据汇聚层(Packet Data Convergence Protocol,PDCP)、无线链路层控制层(Radio Link Control,RLC)和媒体接入控制层(Medium Access Control,MAC),整体结构如图 1 所示。

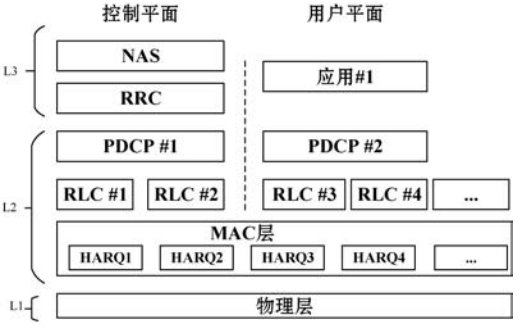

图 1　LTE 协议栈结构示意图

目前针对协议栈的研究,主要基于三层功能的软件实现上,通过软件进行优化;空间管理也是基于主存方式,多采用基于 Cache 的缓存机制,协议栈的数据局部差导致容易产生 Cache Miss[4]。文献[6 - 8]针对数据缓存机制,提出了双缓冲队列,该方式需要采用 2 倍的内存,使得内存占有量大大增加,不利于高速率的嵌入式处理平台。文献[9 - 10]提出改进的硬件机制进行缓存,并提出了基于硬件的快速存储技术,但通用性较差,不能很好地进行复杂协议的开发。

综上所述,目前的 LTE 协议栈实现,主要从软件角度实现算法的优化,从而克服硬件底层瓶颈。本课题基于国家 03 专项(新一代宽带无线移动通信网)面向 LTE-Advanced 的终端软基带技术进行研究,设计高性能的 LTE 无线通信协议栈。采用通用软件协议算法,提出采取“SPM + DMA”结合的数据缓存方案用于 LTE 终端数据处理,提高其性能。

1　原理介绍

1.1　协议栈数据处理流程

各层的协议功能由对应的实体来完成,LTE 终端协议栈在进行上行数据发送过程中,当应用层数据递交给 PDCP 层后,PDCP 层根据无线承载的不同实例化对应的实体进行处理,主要处理过程有:对协议数据单元(Protocol Data Unit,PDU)构建和添加 PDCP 报头、控制信息,进行数据加密和完整性保护并添加包尾,并且将处理好的数据进行缓存,以备在切换场景时对这些数据进行重新处理和发送。

PDCP 层将数据组装成 PDCP PDU 传递给 RLC 层,RLC 层包含三种数据传输模式:透明模式(Transparent Mode,TM)、非确认模式(Unacknowledged Mode,UM)和确认模式(Acknowledged Mode,AM),对应三种协议功能实体,一个无线承载 RLC 例化一个对应实体进行处理。RLC 接收到 PDCP PDU 后,不能立即发送,需要结合 MAC 层的反馈信息,对 PDCP PDU 的格式进行分段和级联,使数据长度能适应 MAC 层指定的大小进行实际传输。重组后数据根据不同的 RLC 实体添加不同的 RLC 报头。另外,在 RLC AM 传输模式中,需要根据接受端给发送端的状态报告和丢包情况,对丢失的 PDU 进行重组和重传,这就要求 RLC 层每次组装好 AM PDU 之后对 PDU 进行备份,并将由状态报告解析而来的重传信息也进行缓存。当下一次发送时刻到来时,根据重传信息,从备份的 AM PDU 中取出相应的部分进行发送。

经过 RLC 层处理后的数据在 MAC 层进行调度处理,一个终端只存在一个 MAC 实体对协议栈上下行所有承载进行统一管理,在功能上,MAC 实体可以分为混合自动重传请求(Hybrid Automatic Repeat reQuest,HARQ)实体、控制器和逻辑信道复用/解复用实体。混合 ARQ 实体执行 HARQ 协议功能,通过不同的重发方案,负责处理传输错误,提供了抵抗传输错误的鲁棒性。复用/解复用实体可以将多个逻辑信道的数据复用到一个传输信道实现逻辑信道和传输信道间的映射。随机接入控制实体负责控制随机接入信道。控制器负责包括调度,上行时间对齐、非连续接收等功能。

终端接收数据时,PHY 层将接收到数据传输给协议栈,经过 MAC 层、RLC 层和 PDCP 层处理,在这个过

程中,处理器与主存交互,读取对应层的实体协议信息和数据包信息,逐层完成数据的解包、去包头包尾、解密、重排序等功能,最后将经过协议栈解析出的专用数据发送给上层。

1.2 缓存机制

目前通用处理器使用片上缓存方式来弥补外存数据存取速率与内核数据速率之间的不匹配问题。片上缓存大多数采用 Sram 结构,主要分为高速缓存(Cache)和便签式存储器(SPM)两种方式[6,11-14]。

Cache 缓存方式是利用连续数据自动高速转存的方式实现数据读写。CPU 计算时,在相邻较短时间内对相邻数据的访问发生概率较大。因此,CPU 在发起一个数据读取时,将数据地址相邻的一段数据同时读取到 Cache 中,这样就可以保证处理器对这一段数据的读取都是通过 Cache 进行,从而减小 CPU 访问外存的巨大时间消耗,提高处理速率。

SPM 是片上具有独立物理地址的存储空间,可由软件进行管理,相比 Cache 缓存,具有结构简单、访存时间短、功耗低、存储密度大等优势。其面积小、功耗低、易于软件编程管理、数据读取周期响应稳定,非常适用于嵌入式终端应用开发。

在处理器对 LTE 协议栈进行数据处理时,处理器需要频繁地读取主存中的各层实体信息和数据信息。使用 Cache 机制进行分级缓存时,数据在不同层间流通处理时,上一时刻从主存中自动搬运缓存到 Cache 上的数据,下一时刻在另一协议层需要读取不同的实体信息,则当前 Cache 中的缓存数据便不再需要,处理器必须重新发起总线操作耗费大量的周期从主存中重新读取数据。尤其当协议栈处理上行数据时,需要不断地通过读取少量数据进行解析,然后根据解析出来的数据特征信息决定接下来需要处理的数据段长度和相应的解析操作,数据局部性差,Cache 缓存机制不能有效发挥,而从主存中读取数据将耗费大量的周期,对处理器性能造成浪费。并且 Cache 对编程者来说是透明的,其硬件结构大小和替换策略决定了其处理过程中不同的数据缓存性能,无法通过软件手段对其进行管理和优化。

2 SPM + DMA 用于协议处理数据缓存

2.1 SPM + DMA 架构

如图 2 所示,通常情况下,处理器可以通过 Cache 方式和 SPM 方式在单周期内访问其内部的数据。但 Cache 机制无法确保处理器需要的数据一定在 Cache 中,一旦处理器需要的数据不在 Cache 中,就会发生 Cache Miss,从而需要 10 ~ 100 个周期从外存访问数据。

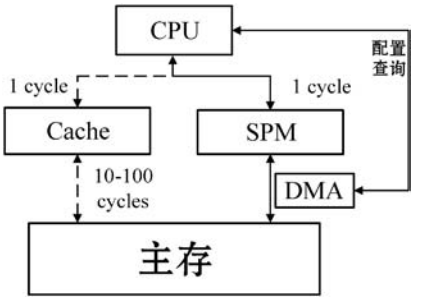

图 2 计算机数据缓存结构示意图

SPM 可以实现“定址”操作,是一个读写数据只需要 1 个周期的“主存”。理想情况下,如果处理器当前时刻需要读写的数据都在 SPM 中,则可以实现存取数据快、无缓存不命中问题的高效数据处理方式。但往往待处理的数据量很大,而片上存储资源十分紧张,无法做到将全部的数据搬运至 SPM 中。

直接存储器存取 DMA[8] 是一种高效的数据搬运机制,CPU 只需要配置目的地址、源地址和所需要搬运的数据长度,就可以自动完成数据块的搬运操作,使 CPU 摆脱数据搬运工作占用的大量的周期,传输速度快,占用较少的 CPU 资源就可以完成大量的数据搬运,有效减轻处理器负担,提升嵌入式系统的性能。

通过使用 DMA 可以实现 SPM 与主存的数据交互。可以结合 SPM 与 DMA 各自的性能优势,配置 DMA 将主存中的待处理数据提前搬运至 SPM 中,待处理完成后将其搬出,并更新下一次待处理数据。便避免 LTE 协议栈处理过程中因为数据局部性特征差 Cache 缓存不命中带来的性能损失,提高处理效率。

2.2 协议栈数据处理结构

协议栈在处理数据时,以数据包为单位,对应用层或物理层传递过来的数据进行处理,主要包括对数据包进行添加报头/解报头、加密/解密、完整性效验、填充包尾/去包尾等操作。

在协议栈处理数据是通过各层的实体来完成的,各层按照不同的服务数据单元(Service Data Unit, SDU),使用相应的实体进行处理。不同的实体代表着不同的协议处理方式,需要缓存实体的数据信息以供处理器在处理数据的过程中调用。

本文 LTE 终端通信协议栈实现基于中国科学院微电子研究所自主研发的 IME-Diamond 多核嵌入式开发平台[15],该平台单核对应的片上 SPM 存储空间为 128 KB,为 4 行 ×8 列结构,一共分为 32 块,如图 3 所示。硬件机制要求不能在同一时刻对同一块进行数据的读和写操作。基于此平台,为了提高处理速度,配合

CPU 处理数据的效率,我们将 SPM 上下两行分为 2 个区域:协议区 A 和数据区 B。协议区负责处理器在处理数据时,提供协议栈各层相应的实体缓存空间等;数据区用于存放各层待处理的数据和控制 SDU\PDU,并为数据在后续各层处理过程中添加报头、报尾、加密、分段等提供相应的缓存空间。通过划分上下空间可以避免 CPU 在读取各层协议缓存数据与 DMA 并行进行数据搬运时发生冲突。

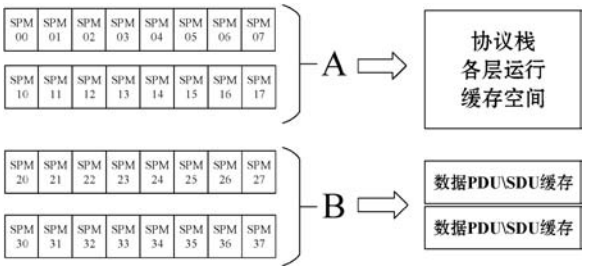

图 3　SPM 缓存空间划分示意图

数据区 B 用于存放 DMA 从主存中搬运的数据缓存空间,并且将数据区按行分为两块,主要目的是为了避免在数据处理过程中,CPU 读取待处理的数据与 DMA 预搬运的数据或 DMA 将上一时刻处理完毕的协议数据搬运至主存时发生块冲突。虽然硬件上如果 CPU 和 DMA 同时对一块 SPM 进行读写时,仲裁器会优先 CPU 的读写操作,但这会让 DMA 等待,浪费时间周期,通过两块空间的“乒乓”操作可以在逻辑上避免 CPU 读写数据块与 DMA 搬运数据块发生冲突,提升效率。

在数据处理时,根据协议要求,CPU 提前通过配置 DMA 的数据源地址(数据包在主存中的地址)、目的地址(在 SPM 中给数据包划分的缓存空间起始地址)和数据包长度,将数据从主存搬运到 SPM 中,以供处理器按照协议要求对数据包进行处理。在数据处理完成后,CPU 再次配置 DMA,从 SPM 中将处理完成的数据搬运回主存中,以备后续应用层使用或通过物理层发送出去。

由于 DMA 搬运要求数据必须是连续的,故需在主存中开辟连续的空间用于存放待处理的数据包,这里通过静态开辟不同大小的空间(64 字节、128 字节、256 字节、2 048 字节)用于存放相应大小的数据,避免在运行过程中内存空间的浪费和碎片化问题。

最终整体缓存方案如图 4 所示,CPU 通过 I-Cache 读取程序指令,通过 D-Cache 和 SPM 中读取数据,D-Cache和 SPM 共同负责对主存数据进行二级缓存。将协议栈处理过程中各层实体缓存空间开辟到 SPM 中,CPU 通过配置 DMA 将数据从主存搬运至 SPM 或从 SPM 搬至主存,实现 CPU 交互数据的缓存空间从 Cache 转换到 SPM 中,在 SPM 中进行高速协议处理的目标。

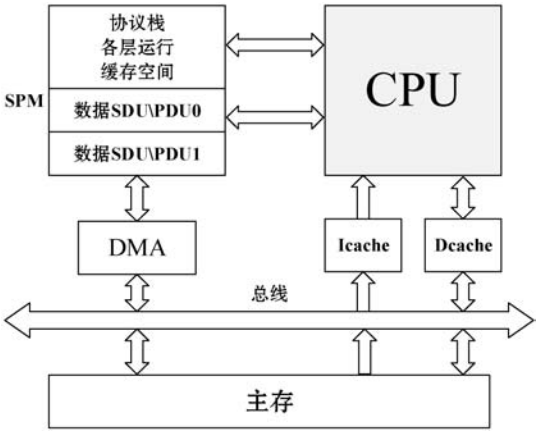

图 4　协议处理过程中数据缓存结构示意图

2.3　协议栈数据运行流程

协议栈运行时,将对应的 PDCP、RLC 和 MAC 实体缓存空间开辟到 SPM 中,如图 5 所示。在终端上行数据进行发送时,首先经过 PDCP 层进行处理,将 PDCP 实体数据缓存到 SPM 对应空间上;然后 CPU 配置 DMA 将数据从主存中搬运至 SPM 数据缓存区,CPU 查询 DMA 任务完成后,在 SPM 中进行添加 PDCP 包头、包头压缩、数据加密、添加包尾等过程;PDCP 数据处理完成后,需要进行备份以场景切换情况下的重传,通过配置 DMA 将数据搬运至主存缓存空间,即相当于完成一次拷贝备份;PDCP 层将 SPM 数据传递给 RLC 层,相应地在 SPM 协议缓存空间中例化相应的 RLC 实体模式(TM 实体、UM 实体或 AM 实体),CPU 读取相应的实体信息,按照协议规定完成数据分段、重组和添加 RLC 报头功能;最后配合 MAC 层的信息,根据不同的信道发送状况,将完成构建的数据通过 DMA 直接发送出去或选择暂存到主存等待合适时机通过物理层发送。

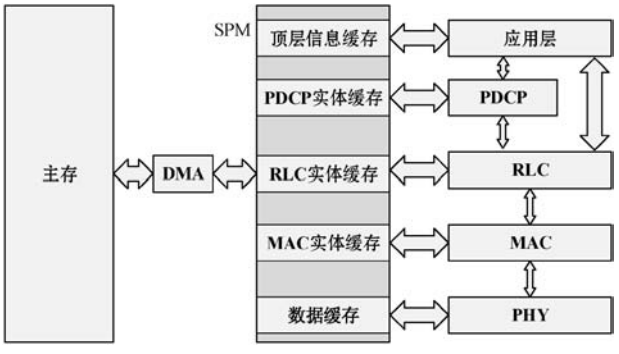

图 5　协议栈数据运行结构示意图

在终端数据下行接收时,PHY 接收的数据包缓存到主存的连续空间,通过 DMA 搬运至 SPM 中的数据缓存中,然后依次经过 MAC\RLC\PDCP 层完成解包,最后通过 DMA 搬运至主存,完成协议栈功能。

3 实验结果与分析

本文将两种缓存方式:传统 Cache 缓存和引入“SPM + DMA”缓存机制所实现的终端协议栈在 IME-Diamond 多核处理器上进行了移植和性能测试,通过模拟搭建基站和终端实现 2 000 帧数据收发。IME-Diamond处理器单核对应片上存储空间 SPM 大小为 128 KB;指令高速缓存 I-Cache 大小为 16 KB;数据高速缓存 D-Cache 大小为 16 KB。采用 SPM + DMA 缓存机制,在 SPM 为协议栈运行静态开辟 45 240 B 空间,开辟了数据缓存空间大小为 64 KB。

在同等吞吐量的情况下,两种缓存方式的运行结果如表 1 所示。

表 1 两种缓存方式运行结果对比

缓存方式	Dcache Miss 数	总指令数	总周期数
Cache	5.032 94e + 07	1.751 33e + 09	5.973 64e + 09
SPM + DMA	3.805 92e + 07	1.748 16e + 09	5.217 59e + 09

使用 Cache 缓存方式总运行周期为 $5.973\ 64 \times 10^9$,采用 SPM + DMA 优化后程序的总运行周期为 $5.217\ 59 \times 10^9$ 个时钟周期,相比之前减少了 12.65%;优化前共发生 $5.032\ 94 \times 10^7$ 次 D-Cache Miss,优化后共发生 $3.805\ 92 \times 10^7$ 次 D-Cache Miss,次数相比优化前减少了 24.38%。

优化前所需要处理器运行频率为:

$$\text{处理器运行频率} = \frac{\text{总周期数}}{\text{运行时间}} = \frac{5.973\ 64 \times 10^9}{20} = 298.68\ \text{MHz}$$

优化后所需要处理器运行频率:

$$\text{处理器运行频率} = \frac{\text{总周期数}}{\text{运行时间}} = \frac{5.217\ 59 \times 10^9}{20} = 260.88\ \text{MHz}$$

处理器的速度提升:

$$\text{性能提升} = \frac{298.68 - 260.929}{298.68} = 12.65\%$$

所以采用 SPM + DMA 缓存机制可以使 LTE 终端协议栈处理速度提升 12.65%。

4 结 语

本文介绍了协议栈的功能与数据处理过程中缓存方式,针对 Cache 缓存的弊端提出了使用“SPM + DMA”结合的高性能数据缓存机制,详细介绍了实现的原理和流程,并在 IME-Diamond 平台进行了性能测试。对比实验结果表明,在同等吞吐量的情况下,SPM + DMA 缓存机制相比 Cache 数据缓存方案在协议栈处理中,可以将运算速度提升 12.65%。

本文在嵌入式平台中,处理器仅仅只运行协议栈一个程序,相对于实际终端设备,处理器需要同时运行系统、切换不同的程序,CPU 处理过程中整体的数据局部性更差,将导致在处理协议栈的时更容易发生 Cache Miss 问题。因此,本文提出的通过 DMA 将协议栈数据搬运至 SPM 中的缓存机制,在实际中应该有更好的性能提升。

参 考 文 献

[1] 曹亘,吕婷,李轶群,等. 3GPP 5G 无线网络架构标准化进展[J]. 移动通信,2018,42(1):7 - 14.

[2] Chen S, Fei Q, Bo H, et al. User-centric ultra-dense networks for 5G: challenges, methodologies, and directions[J]. IEEE Wireless Communications, 2018, 23(2): 78 - 85.

[3] 3GPP TR 36.804. Study on New Radio Access Technology; Radio Interface Protocol Aspects V0.4.0[S]. 2016.

[4] Haw R, Kazmi S M A, Thar K, et al. Cache aware user association for wireless heterogeneous networks[J]. IEEE Access, 2019, 7:3472 - 3485.

[5] Milutinovic S, Mezzetti E, Abella J, et al. Increasing the reliability of software timing analysis for cache-based processors[J]. IEEE Transactions on Computers,2019,68(6):836 - 851.

[6] 张轮凯,宋风龙,王达. 一种针对片上众核结构共享末级缓存的改进的 LFU 替换算法[J]. 计算机应用与软件,2013,30(1):1 - 6,10.

[7] Hui Z, Qiu M, Gai K, et al. Maintainable mobile model using pre-cache technology for high performance android system[C]//IEEE International Conference on Cyber Security & Cloud Computing. 2016.

[8] Kamal R, Arostegui J M M. Design and analysis of DMA based network interface for NoC's router[C]//IEEE International Conference on Recent Trends in Electronics. 2017.

[9] Bernardi P, Cantoro R, Gianotto L, et al. A DMA and CACHE-based stress schema for burn-in of automotive microcontroller[C]//2017 18th IEEE Latin American Test Symposium(LATS). IEEE, 2017.

[10] Rhu M, O'Connor M, Chatterjee N, et al. Compressing DMA Engine: Leveraging Activation Sparsity for Training Deep Neural Networks[C][2018 IEEE International Symposium on High Performance Computer Architecture(HPCA). 2018:78 - 91.

[11] 易俗,殷慧文,张一川,等. 分布式环境下的频繁数据缓存策略[J]. 计算机应用与软件,2017,34(8):12 - 17,86.

[12] 郭俊石,罗轶凤. EarnCache:一种增量式大数据缓存策略[J]. 计算机应用与软件,2017,34(11):44 - 47,102.

(下转第 137 页)

成后的边缘请求成功率约为 52.0%,非常接近理论最优 53.0%。结果证明基于覆盖率需求的内容扩散完成后,针对已经扩散过的热门内容的请求,用户在边缘网络中以约 98.1% 的概率直接在邻域内获取内容服务。用户的访问在边缘网络中得到了高效的服务。同时,约 52% 的内容流量从骨干网络卸载到边缘网络,骨干网络的负载压力也得到了极大的缓解。其中,边缘请求成功率略低于理论最优的数值。原因可能是极少部分局部区域对部分内容的请求频率略高于全局请求频率,导致依据全局请求情况设定的覆盖率需求不能完全满足局部的请求需求。

5 结　语

本文为边缘网络设计了一个去中心化的内容扩散系统。首先定义了一种邻域覆盖率来反映内容副本在邻域的覆盖情况。系统可以根据内容的请求数量设定合适的覆盖率需求,进而模拟谣言传播的方式将内容的副本在网络中进行扩散。实验证明,去中心化的系统可以实现快速的内容扩散。基于设备空闲存储资源的选择策略也均衡了边缘设备的存储负载。基于覆盖率的内容扩散完成后,用户在边缘网络可以拥有接近理论最优的边缘请求成功率。对于已完成扩散的内容的请求,用户有约 98.1% 的概率直接在邻域内获取高效的内容服务。目前的内容扩散系统中还存在管理系统来设定内容的覆盖率需求。在下一步,会进一步利用边缘节点的自治特性,取消管理系统,设计一个完全去中心化的内容扩散系统。边缘节点可以根据获取的局部信息自适应地调整内容的覆盖率需求,每个内容扩散完成后的副本数量也会尽可能正比于内容的流行度。

参 考 文 献

[1] Ma M, Wang Z, Su K, et al. Understanding content placement strategies in smartrouter-based peer video CDN[C]// Proceedings of the 26th International Workshop on Network and Operating Systems Support for Digital Audio and Video, 2016:7.

[2] An D, Lin T. A proactive RAN caching scheme based on content popularity prediction[J]. Journal of Network New Media, 2016, 5(5):1 - 8.

[3] Choi J, Reaz A S, Mukherjee B. A survey of user behavior in VoD service and bandwidth-saving multicast streaming schemes[J]. IEEE Communications Surveys & Tutorials, 2012, 14(1):156 - 169.

[4] Ma G, Wang Z, Chen M, et al. APRank: Joint mobility and preference-based mobile video prefetching[C]//2017 IEEE International Conference on Multimedia and Expo (ICME), 2017:7 - 12.

[5] Li Z, Shen H, Wang H, et al. SocialTube: P2P-assisted video sharing in online social networks[J]. Proceedings—IEEE INFOCOM, 2012, 25(9):2886 - 2890.

[6] Zhang Y X, Gao C L, Guo Y Z, et al. Proactive video push for optimizing bandwidth consumption in hybrid CDN-P2P VoD systems[C]//IEEE INFOCOM 2018—IEEE Conference on Computer Communications, 2018:9.

[7] Gao G, Li R, Wen K, et al. Proactive replication for rare objects in unstructured peer-to-peer networks[J]. Journal of Network and Computer Applications, 2012, 35:85 - 96.

[8] Spaho E, Barolli L, Xhafa F. Data replication strategies in P2P systems: a survey[C]//2014 17th International Conference on Network-Based Information Systems(NBiS). IEEE, 2014:302 - 309.

[9] Trifa Z, Khemakhem M. A novel replication technique to attenuate churn effects[J]. Peer-to-Peer Networking and Applications, 2016, 9(2):344 - 355.

[10] Mundinger J, Weber R. Efficient file dissemination using peerto-peer technology[EB]. Private Communication,2004.

[11] Ku C F, Yang K H, Ho J M. An optimal scheduling for file dissemination under a full binary tree of trust relationship [C]//2012 IEEE Global Communications Conference(GLOBECOM), 2012:751 - 757.

[12] Zheng H, Yu H. Minimizing distribution time for one-to-many file distribution in P2P networks[C]//2015 International Conference on Electrical and Information Technologies for Rail Transportation, 2016:597 - 607.

(上接第 100 页)

[13] 余翔, 王宏刚, 段思睿. LTE-A 系统中基于 QoE 能效的无线资源分配算法[J]. 计算机应用研究, 2019, 36(5):216 - 218,235.

[14] 孟子潇. CoCache:基于文件关联性的协同缓存策略[J]. 计算机应用与软件,2018,35(7):52 - 56.

[15] 王志君,梁利平等. 一种面向多媒体和通信应用的处理器指令集及架构实现[J]. 湖南大学学报(自然科学版), 2014,41(10):108 - 114.

(上接第 108 页)

[16] Wahab O A, Otrok H, Mourad A. VANET QoS-OLSR: QoS-based clustering protocol for Vehicular Ad hoc Networks [J]. Computer Communications, 2013,36(13):1422 - 1435.

[17] Hornig R. An overview of the OMNeT++ simulation environment[C]//International Conference on Simulation Tools & Techniques for Communications. 2008: 11 - 32.