

基于帕累托的 LTE 小基站 PHY-MACAPI 时间窗选择方法^①

武 一^{②*} 于振浩^{*} 萧 放^{*} 孙 刚^{**}

(^{*} 河北工业大学电子信息工程学院 天津 300401)

(^{**} 中国科学院计算技术研究所 北京 100190)

摘 要 针对协议栈子系统 Linux 的实时性引起协议栈任务运行时间具有抖动性而导致采用选择固定大小时间窗的传统方法降低协议栈处理效率的问题,在中科院计算所 LTE 协议栈的基础上,按照 Femto Forum 的应用程序接口 (FAPI) 提出了一种基于帕累托分布构建模型的方法,从而达到提高下行链路协议栈处理效率的目的。针对不同架构、主频等参数的 LTE 小基站的设计,根据该方法能够高效地选择适合自身的应用程序接口 (API) 时间窗。实际测试结果表明,与大多数厂家采用的固定时间窗的方法相比较,这种方法能够明显提升下行链路协议栈处理效率。本研究能够为动态时间窗的实现提供参考。

关键词 PHY-MAC 交互, 时间窗, 构建模型, 帕累托分布, 效率优化

0 引 言

长期演进 (long term evolution, LTE) 小基站物理层 (physical layer, PHY) 与媒体接入控制层 (media access control, MAC) 之间交互的应用程序接口 (application programming interface, API) 是小基站重要的组成部分,时间窗是 API 的一个重要参数^[1]。协议栈处理效率表示在时间窗内,协议栈单位时间处理传输流量的多少。传统选择时间窗的方法是对小基站的协议栈软件性能、协议栈处理器计算能力、物理层算法性能以及物理层处理器计算能力等各个模块分别评估。根据评估结果,考虑最坏的情况,对 API 的时间窗进行选择^[2-4]。这种分模块评估的方法强化了模块间的独立性,弱化了模块间的相互作用和联系。协议栈软件运行在操作系统上,其任务运行时间在一定范围内具有抖动性^[5-8];由于处理器计算体系结构和操作系统实时性能的不同,造成了任务运行时间具有不同的抖动特性。传统方法没

有充分考虑这种抖动性对时间窗选择带来的影响。

大多数设备厂商 (如英特尔、飞思卡尔和爱立信) 为了兼顾大多数厂家以及保证 MAC 层下发的消息落入时间窗内,选择固定的、“宽裕”时间窗作为 PHY-MACAPI 的时间窗。但是针对不同架构、参数的小基站设计来说,这种传统方法忽略了时间窗选择对协议栈处理效率的影响。如果时间窗选择偏大,则造成处理器计算资源浪费,协议栈处理效率降低。

本文分析协议栈子系统 Linux 软实时性对协议栈软件任务运行时间的影响。然后设计 PHY-MAC 交互实验,统计 MAC 层响应 PHY 消息所用时间,提取时间分布特征。进而在得出分布特征符合帕累托分布的基础上,分析帕累托分布对时间窗选择的影响,提出了构建协议栈处理效率模型的方法。可以根据该模型快速准确地选择合适的时间窗,达到提高下行协议栈处理效率的目的。最后通过实际测试验证说明了该方法的合理性和高效性。

① 国家科技重大专项基金 (2015ZX03001026-002) 资助项目。
② 女,1964 年生,博士,教授;研究方向:移动通信网络,物联网等;联系人,E-mail: 937144106@qq.com
(收稿日期:2018-01-22)

1 PHY-MAC API 简析

LTE 小基站由物理层子系统、协议栈子系统和射频子系统组成。物理层子系统,主要完成 PHY 所有收发功能,以及空口同步等功能。PHY 运行在无操作系统的数字信号处理器(digital signal processor,DSP)上。协议栈子系统主要完成协议栈软件对数据的处理。协议栈软件包括 MAC、无线链路控制(radio link control,RLC)、分组数据汇聚协议(packet data convergence protocol,PDPC)和无线资源控制(radio resource control,RRC)4 个子层。协议栈软件运行在 Linux 操作系统上。通过 API 传输消息完成 PHY-MAC 之间的数据交互。

在 LTE 传输时间间隔(transmission time interval,TTI)内,PHY-MAC API 需要将两者交互消息数据完全处理^[3]。在每个 TTI 起始处,PHY 通过 TTI 中断向 MAC 层上报消息。MAC 层响应 TTI 中断并做出相应的消息处理,然后下发中断通知 PHY,如图 1 所示。根据 Femto Forum 的应用程序接口(femto forum application programming interface,FAPI)时序,PHY 只接收并处理在时间窗内收到的 MAC 层下发的消息。对于在时间窗外收到 MAC 层消息的情况,称作超时^[1]。

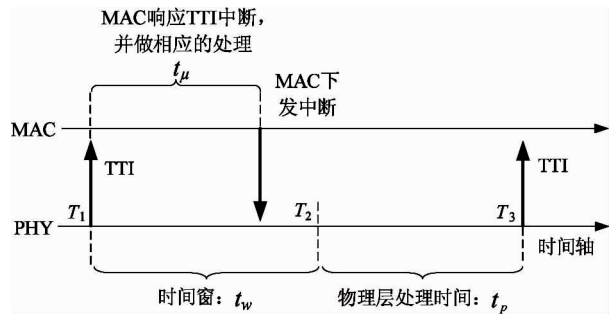
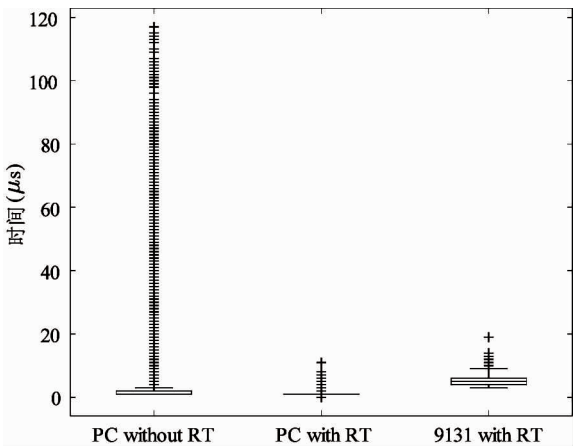


图 1 PHY-MAC API 时序以及时间窗示意图

2 协议栈子系统 Linux 实时性分析

由于 PHY 是在无操作系统的 DSP 上实现,运行平稳,下面只分析协议栈子系统 Linux 的实时性对时间窗选择的影响。协议栈子系统采用 Linux 操作系统,下面对 Linux 的实时性进行分析。cyclicttest

是开源的 Linux 内核实时性能测试工具,本文使用 cyclicttest 对装有 Linux 操作系统的桌面 PC 与飞思卡尔的 BSC9131 片上基站分别进行测试并采样,样本容量均为 10^6 。使用箱线图表示样本数据,如图 2 所示。结果表明:(1)PC without RT 与 PC with RT 对比:Linux 实时性优化后,实时性有了改善,但其仍然是一个软实时系统。图 2 中 99% 的情况分布在 $10\mu s$ 以内,但是仍有少量的异常数值抖动较大。这些异常值对时间窗的选择有很大影响。(2)PC with RT 与 9131 with RT 对比:在相同的内核版本与相同的补丁版本情况下,不同参数的设备表现的处理能力和实时性是不同的。这些样本数据的特征是对处理器计算能力和操作系统实时性的综合反映。因此,针对不同架构、不同参数的小基站设计来说,协议栈子系统的处理能力不同,需要的 PHY-MAC 交互时间窗大小也不同。



PC without RT:PC 未打实时性补丁;PC with RT:PC 打实时性补丁;9131 with RT:9131 打实时性补丁

图 2 实时性测试数据箱线图

Linux 是软实时操作系统。软实时系统即从统计的角度来说,一个任务能够在确定的时间内执行完成,但是仍会存在超出确定时间的情况^[9,10]。多次执行同一个任务,其每次执行时需要的任务调度时间、任务切换时间与中断响应时间不固定,在一定范围内浮动变化,具有抖动性^[11,12]。

PHY-MAC 进行数据交互时涉及到 Linux 的任务调度、任务切换、中断响应等过程。因此,每次交互的时间参数 t_μ 具有不确定性,从而影响时间窗的选择。如果时间窗 t_w 太小,容易造成 MAC 层下发

的中断落入时间窗之外,以致超时;如果时间窗太大,造成 t_p 减少,吞吐量下降,协议栈处理效率降低^[13]。

3 MAC 响应 PHY 消息的时间分布特征

MAC 层响应来自 PHY 的消息中断,并根据消息内容做出相应的处理,如图 1 中 t_μ 所示。下面通过设计实验测试 t_μ ,并采集样本数据。然后对样本数据进行分析,提取时间 t_μ 分布特征。根据分布特征,使用帕累托分布对样本数据的概率密度进行拟合,并进行合理性论证。

3.1 设计 PHY-MAC 交互实验

该实验的目的是测试 t_μ ,并采集样本数据。设计的实验的线程模型如图 3 所示。由 Iperf_Client 产生下行数据,经过线程调用 UDP_Recv→PDCP_Tx→RLC_Tx→DL_Mem_Pool→MAC_Tx 实现下行链路协议栈数据发送。API 触发 MAC_Tx,将上行数据经过线程调用 UL_Mem_Pool→RLC_Rx→PDCP_Rx→UDP_Send 传送给 Iperf_Server,实现上行链路协议栈数据接收。采用回环测试的方法,实现数据的下行发送和上行接收。

为了缩短实验周期,实验中没有引入真实的物理层,使用高精度 Timer 定时器模拟 1msTTI 中断。当 1ms TTI 中断到来时,MAC 层响应 TTI 中断,并去

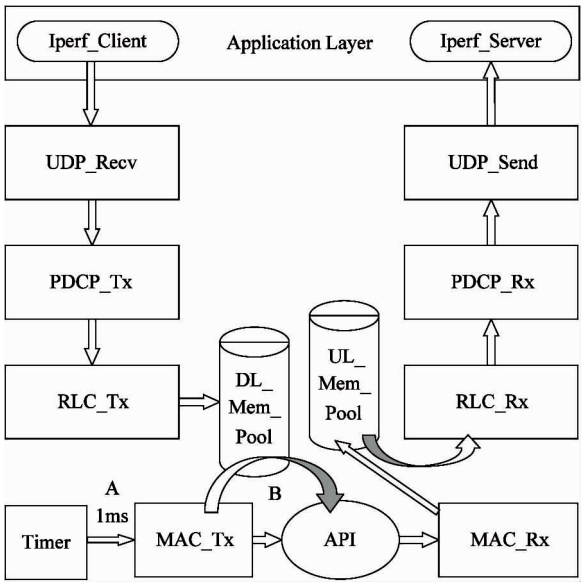


图 3 实验线程模型与观测点

下行内存池 (DL_Mem_Pool) 中做相应的消息处理,处理完成后调用 API。

本文主要是对下行协议栈处理效率优化,因此在观测点 A 与 B 处使用代码插桩的方法,测量下行链路情况下的参数 t_μ 。同时采集数据,样本容量为 10^6 。该实验是在表 1 中 3 种不同的实验环境参数中分别进行,这 3 种实验环境的处理器计算能力和 Linux 操作系统的实时性不同。通过多组实验,测量多组样本数据,目的是得到更加真实、准确的实验结果。

表 1 实验环境参数

参数类型	环境 a	环境 b	环境 c
厂商	飞思卡尔	苏州国芯	英特尔
平台	BSC9131	CCFC9000PA	X86
处理器内核	PowerPC e500	PowerPC 476FP	Intel i5-7400
CPU 核数	单核	2 核	4 核
主频	1GHz	800MHz	2GHz
操作系统	Linux 2.6.33.9-rt31	Linux 3.4.106-rt143	Linux 3.11.6

3.2 样本数据分析

样本数据是对协议栈处理器计算能力、Linux 系统实时性和协议栈软件的一个综合反映。通过对 3 种实验环境下的样本数据进行分析,提取出样本数

据的数字特征,如表 2 所示。表 2 中众数与平均数相差较大,并且众数靠近最小值的位置,说明数据分布不均匀,大多数为小数值。

表 2 样本数据的特征			
参数类型	环境 a	环境 b	环境 c
最小值(μs)	49	42	38
最大值(μs)	250	307	248
平均值(μs)	132	163	137
众数(μs)	72	83	56

将样本数据处理,得出 t_{μ} 的概率密度曲线,如图 4 所示。这 3 种实验环境下的概率密度曲线图具有共同特征:绝大多数的数据集中分布在左边数值较小的区间,只有少量的数据分布在右边数值较大

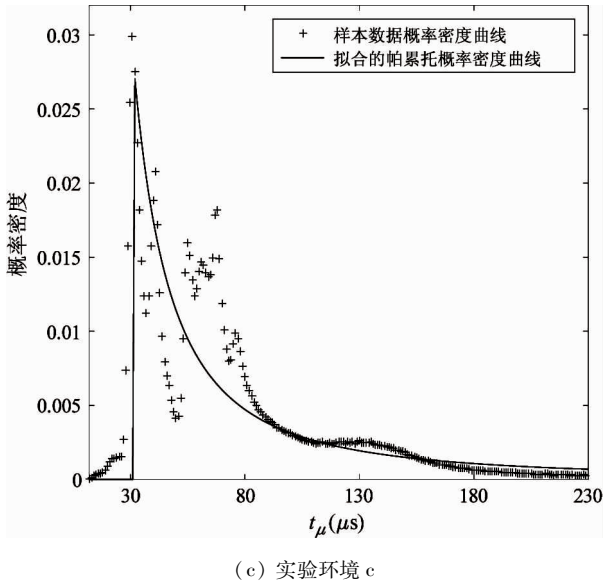
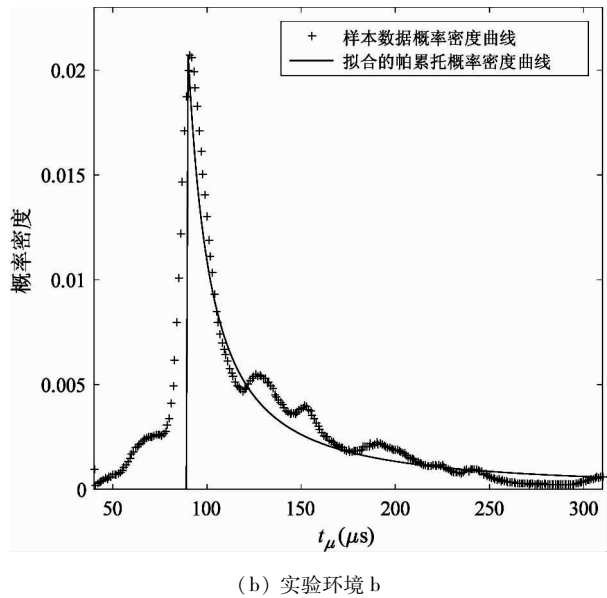
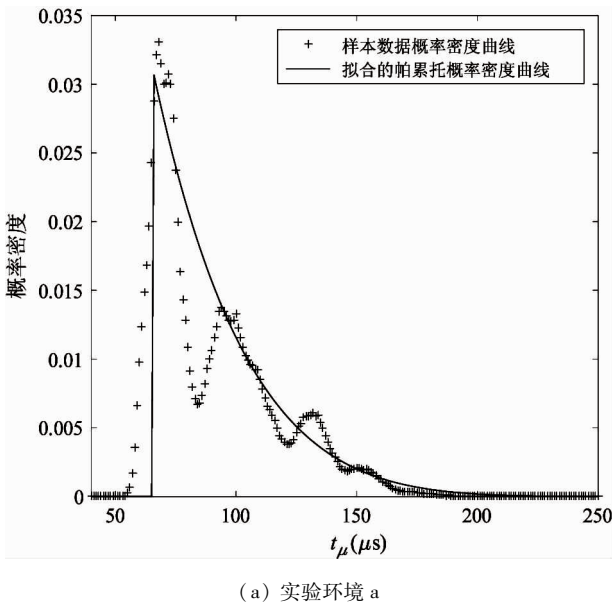


图 4 样本数据概率密度

的区间;曲线右侧存在一个“长尾”,而且“长尾”部分发生的概率很低。该“长尾”分布特征表明,MAC 层在绝大多数情况下可以在相对较短的时间内完成对 PHY 消息的响应,但仍然存在极少数情况需要较长时间。需要较长时间的“长尾”部分发生的概率很低。这些数据特征符合帕累托分布。本文选择帕累托分布对样本数据的概率分布进行拟合,下文对此进行论证。

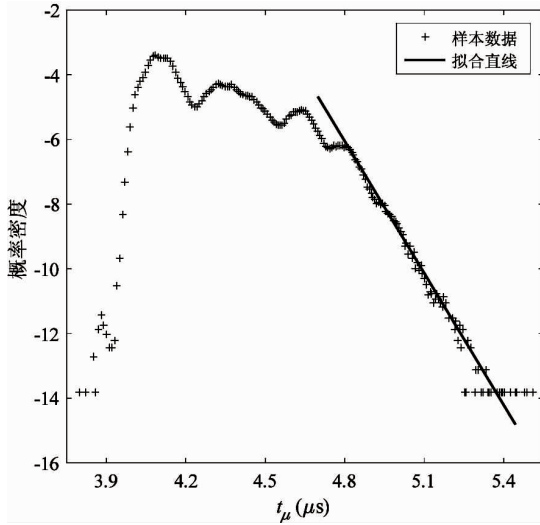
3.3 帕累托分布论证

通过对样本数据分析,选择帕累托分布对样本数据的概率密度进行拟合。帕累托分布是幂律分布的一种,其有两个重要的特征:(1) 长尾:帕累托分布包含许多小数值和少量大数值。(2) 无标度:短尾分布集中在一定范围内,通常称为“标度”^[14-16]。帕累托分布的概率密度函数式为:

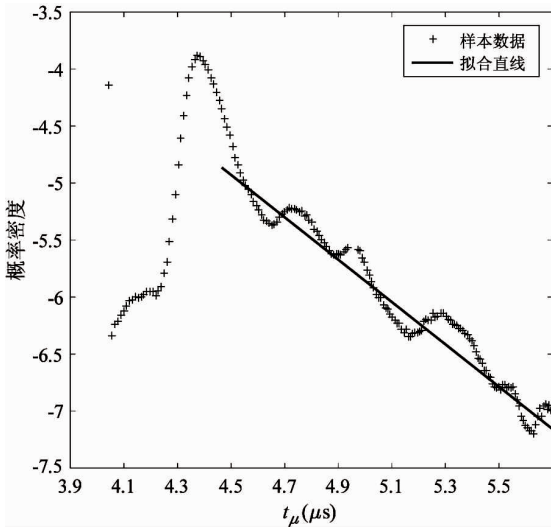
$$f(x \mid k, \sigma, \theta) = \left(\frac{1}{\sigma}\right) \left(1 + k \frac{(x - \theta)}{\sigma}\right)^{-1 - \frac{1}{k}} \quad (1)$$

其中, k 为形状参数, σ 为尺度大小参数, θ 为阈值, x 大于阈值 θ 。

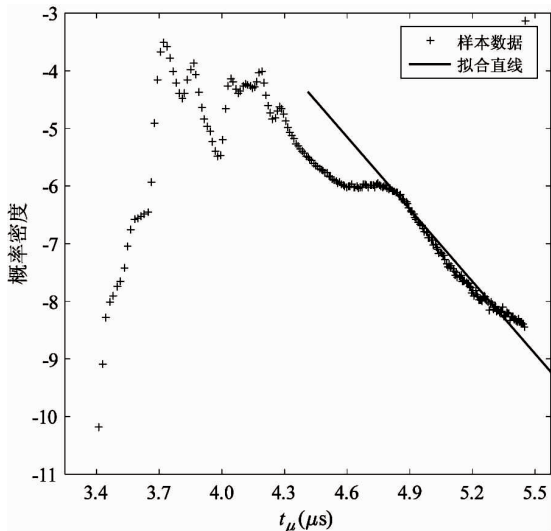
在双对数坐标中,幂律分布表现为一条斜率为幂指数的负数的直线,这一线性关系是判断给定的实例中随机变量是否满足幂律的依据^[8,14]。采用一元线性回归和最小二乘法对样本数据处理,可得样本数据在双对数坐标中的回归直线方程,如图 5 所示。由于操作系统内部调度因素的影响,图5前半



(a) 实验环境 a



(b) 实验环境 b



(c) 实验环境 c

图5 在双对数坐标下“尾部”的线性拟合

部分呈现为直线的程度较低,而后半部分(对应图4的尾部)则近乎为一条直线,该直线的斜率的负数为幂指数。因此,使用帕累托分布拟合是合理的,尤其是长尾部分,符合程度非常高。

帕累托分布“长尾”部分的大数值发生的概率很低,是小概率事件,但是在使用传统方法选择时间窗时,却影响固定的、“宽裕”时间窗的大小。因此,与大多数帕累托分布的应用意义相同,在选择时间窗时,会更加关注帕累托的“尾部”。

4 基于帕累托的协议栈处理效率模型构建与验证

协议栈效率与时间窗有关,下面基于帕累托分布构建协议栈处理效率模型,通过模型寻找合适的时间窗以实现下行协议栈处理效率达到最优。

4.1 模型构建

从宏观统计的角度考虑,在同样的应用场景下,传输数据中的长包、中包与短包的比例是固定的,每个包的平均处理时间的期望值为常数。本文使用线性关系处理下行链路传输流量与协议栈处理时间的关系:

$$\eta_0 = \frac{L_0}{T_0} \quad (2)$$

其中,用 η_0 表示下行链路协议栈处理效率。下行协议栈处理效率表示在时间窗内,协议栈单位时间内处理的下行传输流量。 L_0 为下行链路传输流量。在不考虑协议栈子系统为非实时的情况下, T_0 表示 MAC 层响应 PHY 消息并做相应处理所需时间的理想值。 η_0 和 T_0 由处理器的硬件结构、操作系统和协议栈软件决定。

然而在实际情况中,协议栈子系统 Linux 是软实时系统,其任务运行时间具有抖动性。MAC 层响应 PHY 消息并做相应处理的时间 t_μ 随协议栈子系统抖动而动态变化。因此,在附加了时间窗的情况下,带来下行协议栈相对处理的效率降低。相较于理想的实时情况,在非实时系统中,相同的传输流量并不能在 T_0 时刻被协议栈处理完成。因此,需要在式(2)的基础上再考虑:传输流量被完全处理时的概率问题:

$$\eta_1 = \eta_0 \times H \tag{3}$$

$$H = \frac{T_0 \times P(T < T_2)}{T_2} \tag{4}$$

其中, η_1 表示实际情况下的下行协议栈处理效率。 T_0 取值为帕累托分布概率密度函数式(1)中的阈值 θ 。 $P(T < T_2)$ 表示在时间窗内协议栈将传输流量处理完成的概率,通过式(1)求出。 H 表示由协议栈软件的抖动性和时间窗带来的相对处理效率。 η_0 表示下行协议栈绝对处理效率,由式(2)可知,其值是由处理器的硬件结构、操作系统和协议栈软件决定的。

当 η_1 达到最大时,求解出对应的时间窗 T_2 。换句话说,此时求出的时间窗为实际下行链路协议栈处理效率达到最大值时的时间窗。当求 η_1 的达到最大值时的 T_2 值,即求下行协议栈相对处理效率 H 达到最大值时的 T_2 值。

在实验环境(a)的数据基础上,得出式(1)的参数值: $k = -0.12939, \sigma = 32.6269, \theta = 65$ 。将这些参数值和式(1)代入式(4)中,解出下行协议栈相对处理效率达到最大化时的时间窗值, T_2 为 $145\mu s$ 。求解过程曲线如图 6 所示。

4.2 模型验证

为了验证该方法的合理性,通过设置小基站系统 API 的实际时间窗,步进为 $10\mu s$,依次增大,测量统计相应的超时率并计算下行协议栈相对处理效率,实际测试设置的时间窗如表 3 所示。

表 3 实际时间窗设置

时间窗 (μs)	50	60	70	...	300	310	320
--------------------	----	----	----	-----	-----	-----	-----

在实验环境(a)中进行实际测试,将实际测试的数据与通过模型得出的数据对比,两者的求解曲线基本吻合,如图 6 所示。随着时间窗的增大,PHY 收到 MAC 层消息的概率增加,协议栈处理的下行传输流量增加,协议栈处理效率增大;当到达最大值点时,协议栈子系统的计算资源利用率达到最大,下行协议栈处理效率达到最大;当超过最大值点后,由于时间窗较大,超时的概率几乎为 0,随着时间窗继续增大,协议栈处理效率开始下降。将设置实际时间

窗并测试数据与使用该模型方法的求解过程对比发现,该方法思路正确、合理。实际测试曲线图中,最大值点 T_2 为 $160\mu s$ 。该模型方法和实际测试的结果值相比较,两者误差约为 9.38%,误差较小。因此,根据此模型可以比较准确地得出下行协议栈处理效率最大时的时间窗。

此外,在传统的选择时间窗方法中,时间窗是一个固定的、“宽裕”的经验值。大多数厂家采用传统的时间窗值为 $250\mu s$ 。如图 6 所示,如果采用传统方法的时间窗,下行协议栈处理效率并不是最优。与之相比,采用该模型构建的方法可以使得下行链路协议栈处理效率 η 提高大约 16.77%。

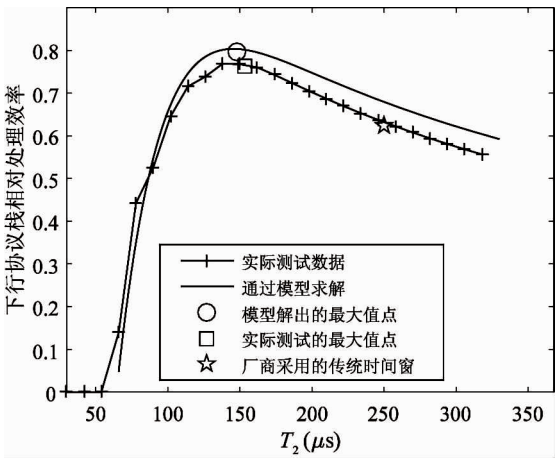


图 6 模型求解最大值点与实际测试求解最大值点对比

5 结论

本文对协议栈子系统 Linux 实时性以及协议栈软件任务运行时间的浮动变化特征进行研究。对 MAC 层响应物理层消息并做相应处理所消耗的时间进行测试、采集数据和数据分析,发现其符合帕累托分布的规律。结合协议栈下行处理效率与时间窗的关系,构建协议栈处理效率的数学模型。利用数学统计的思想解决工程问题,通过模型导出协议栈下行处理效率最大时的时间窗大小。最后通过实际测试数据验证说明构建模型法的合理性与高效性。相比传统方法,通过该方法选择合适的时间窗,使下行协议栈处理效率得到明显提高。该模型利用帕累托分布“以 20% 的投入得到 80% 的产出”的思想,

主要考虑大多数小数值的情况,舍弃少数大数值对时间窗的影响。帕累托分布参数的本质是对协议栈处理器的计算能力、操作系统的实时性能和协议栈软件的综合性反映。

下一步的研究将会进一步考虑 MAC 下发的消息中断落入时间窗之外产生的影响,综合更多的因素以得到一个更加接近实际情况的结果。

参考文献

- [1] Small Cell Forum. NFAP1 and FAPI specification release 7.0[EB/OL]. <https://www.smallcellforum.org/>; Small Cell Forum, 2016
- [2] Mroueh L, Vivier E. Energy efficient relaying PHY-MAC strategy for LTE-advanced networks[C]. In: Proceedings of the International Symposium on Wireless Communication Systems, Paris, France, 2012. 361-365
- [3] Soekartono A, Chen Y, Darwazeh I. PHY-MAC cross layer based modelling for LTE QoS optimization[C]. In: Proceedings of the 10th International Symposium on Communication Systems, Networks and Digital Signal Processing, Prague, Czech Republic, 2016. 1-6
- [4] Zogovic N, Dimic G, Bajic D. PHY-MAC cross-layer approach to energy-efficiency and packet-loss trade-off in low-power, low-rate wireless communications[J]. *IEEE Communications Letters*, 2013, 17(4):661-664
- [5] Wang H H, Duan C B, GuoS Y. Design of TD-LTE PHY and MAC layer in the interactive system primitives[J]. *Applied Mechanics & Materials*, 2014, 513-517:901-904
- [6] Kroshin F S, Jivetiev A V, Drozdova V G. Simulation of probability-time characteristics of wireless MAC protocols WiMAX-like with fixed super frame formats[C]. In: Proceedings of the 14th International Conference of Young Specialists on Micro/nanotechnologies and Electron Devices, Novosibirsk, Russia, 2013. 68-70
- [7] Drozdova V, Akhpashev R. Analysis of probability-time characteristics of the LTE dynamic MAC-protocol[C]. In: Proceedings of the International Multi-Conference on Engineering, Computer and Information Sciences, Novosibirsk, Russia, 2017. 232-235
- [8] Kuo C S, Huang C Y. A study of applying the bounded Generalized Pareto distribution to the analysis of software fault distribution[C]. In: Proceedings of the 2010 IEEE International Conference on Industrial Engineering and Engineering Management, Macao, China, 2010. 611-615
- [9] Cheng K, Bai Y, Wang R, et al. Optimizing soft real-time scheduling performance for virtual machines with SRT-Xen[C]. In: Proceedings of the 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Shenzhen, China, 2015. 169-178
- [10] Bovet D, Cesati M A 著,陈莉君,张琼声,张宏伟译. 深入理解 Linux 内核[M]. 第三版. 北京:中国电力出版社,2008. 258-266
- [11] 张晓龙,郭锐锋,陶耀东,等. Linux 实时抢占补丁研究及实时性能测试[J]. 计算机工程, 2014, 40(10): 304-307
- [12] Sheahan R, Lipsky L, Fiorini P. The effect of different failure recovery procedures on the distribution of task completion times[C]. In: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium, Denver, USA, 2005. 282-283
- [13] 文志成. LTE 多用户下行吞吐量性能建模分析[J]. 电信科学, 2016, 32(10): 150-156
- [14] Zhang T, Gu G F, Xu H C, et al. Power-law tails in the distribution of order imbalance[J]. *Physica A Statistical Mechanics & Its Applications*, 2017, 483:201-208
- [15] 胡海波,王林. 幂律分布研究简史[J]. 物理, 2005, 34(12):889-896
- [16] Huang C Y, Kuo C S, Luan S P. Evaluation and application of bounded generalized Pareto analysis to fault distributions in open source software[J]. *IEEE Transactions on Reliability*, 2014, 63(1):309-319

A timing window selection method for PHY-MACAPI of LTE small cell based Pareto

Wu Yi^{*}, Yu Zhenhao^{*}, Xiao Fang^{**}, Sun Gang^{**}

(^{*} School of Electronic and Information Engineering, Hebei University of Technology, Tianjin 300401)

(^{**} Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)

Abstract

Considering that using the traditional method of selecting a fixed-size timing window reduces the protocol-processed efficiency due to the jitter of the task runtime of protocol stack caused by the Linux real-time performance of protocol-stack subsystem, a method for constructing model based on Pareto distribution is presented on the basis of LTE(long term evolution) protocol stack given by the Institute of Computing Technology of Chinese Academy of Sciences according to the FAPI (femto forum application programming interface) to achieve the improvement of downlink protocol-processed efficiency. For the design of small cell of different architectures, frequency and other parameters, the appropriate timing window could be quickly and accurately selected according to the method. The test results illustrate that the proposed method works well, bringing an effective improvement of downlink protocol-processed efficiency, compared with the method of selecting a fixed-size timing window used by most manufacturers. This study is of referential significance to realization of dynamic timing window.

Key words: interaction between PHY and MAC, timing window, modeling, Pareto distribution, efficiency optimization