支持差异化可协商的数据通信机制

寇文龙[1,2]，李凤华[1,2,3]，董秀则[4]，曹晓刚[2,3]，耿魁[2]，李青[5]

（1. 西安电子科技大学网络与信息安全学院，陕西 西安 710071；2. 中国科学院信息工程研究所，北京 100093；
3. 中国科学院大学网络空间安全学院，北京 100049；4. 北京电子科技学院电子通信工程系，北京 100070；
5. 中国电信股份有限公司研究院，北京 100033）

摘 要：针对云环境和物联网中存在海量、差异化的计算单元和终端设备，硬件资源和运算能力的差异对高可靠高性能的数据通信提出了新的挑战这一问题，提出了支持差异化可协商的数据通信机制。提出参数协商方法，发送端根据接收端能力不同进行参数协商，实现差异化、可协商的数据通信；设计重传反馈机制，发送端通过接收端反馈的数据接收情况动态调整发送速率和重传数据，提高通信效率和可靠性。实验结果表明，所提机制能够根据接收端能力差异，进行动态自适应、高效、并行通信。

关键词：自适应；差异化；可协商；滑动窗口；通信机制

中图分类号：TN92

文献标识码：A

DOI: 10.11959/j.issn.1000−436x.2021183

Differentiated and negotiable mechanism for data communication

KOU Wenlong[1,2], LI Fenghua[1,2,3], DONG Xiuze[4], CAO Xiaogang[2,3], GENG Kui[2], LI Qing[5]

1. School of Cyber Engineering, Xidian University, Xi'an 710071, China
2. Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
3. School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China
4. Department of Electronic and Information Engineering, Beijing Electronics Science and Technology Institute, Beijing 100070, China
5. China Telecom Research Institute, Beijing 100033, China

Abstract: In view of the massive and differentiated computing units and terminal devices in the cloud environment and the Internet of things, the differences in hardware resources and computing capabilities posed new challenges to high reliability and high performance data communication, a data communication mechanism supporting differentiation and negotiation was proposed. A parameters negotiation method was proposed, and parameters were negotiated by the sender according to the difference of the receiver capability to achieve differentiated and negotiable data communication. A retransmission feedback mechanism was designed, and the transmission rate was dynamically adjusted and the data was retransmitted based on the data reception feedback of the receiver by the sender to improve communication efficiency and reliability. The experimental results show that the proposed mechanism can perform dynamic adaptive, efficient and parallel communication according to the difference of the receiver capability.

Keywords: adaptive, differentiated, negotiable, sliding window, communication mechanism

1 引言

随着云计算和物联网等新型网络的大规模部署和应用，5G 移动通信的全球化商用[1]，服务器与终端设备、服务器与硬件计算资源、不同终端设备之间、终端设备与硬件计算资源之间的通信越来越频繁[2]，设计高效、可靠的通信方法是数据通信中亟须解决的问题。

收稿日期：2021−03−16；**修回日期**：2021−06−10

通信作者：耿魁，gengkui@iie.ac.cn

基金项目：国家重点研发计划基金资助项目（No.2017YFB0801802）

Foundation Item: The National Key Research and Development Program of China (No.2017YFB0801802)

云环境和物联网中存在海量、差异化的硬件计算资源和终端设备[3]。数据通信时面临以下 2 个挑战：如何根据计算资源和终端设备的资源情况和接收能力进行协商，实现差异化、可协商的通信；如何根据不同硬件计算资源和设备的状态，动态调整通信速率，进行数据发送和重传，降低丢包率，提高通信的稳定性、可靠性和效率[4]。此外，对 B5G 和 6G 移动通信的研究已经引起了学术界和工业界的广泛关注，通信与计算融合是未来数据通信的一大趋势[5]，实现差异化可协商的数据通信对通信与计算融合的数据通信至关重要。

光纤等高速网络通信设备的快速发展为云计算和物联网带来了更大的带宽支撑，但是数据通信性能却没有随着通信带宽的提升而得到显著提高。其原因是目前云环境和物联网中的大部分设备是基于 TCP/IP 协议族进行数据通信的[6]，而 TCP/IP 协议族在传统的低带宽低时延网络中运行稳定且性能良好，但不适用于高带宽低时延的网络环境。TCP 流量控制的一种方法是快速重传，基于重复确认（ACK, acknowledge）来间接判断重传，其每次只能重传一个数据段，导致通信速度远低于理论的传输带宽；另一种方法是选择重传，一个 ACK 中可包含多个待重传数据段，但是需要设置相当容量的缓存。TCP 的拥塞避免算法会产生一些不必要或者错误的拥塞判断，进而由于指数级的拥塞避让策略导致通信性能大大降低[7]。针对上述问题，研究人员提出了多种优化方案来提高 TCP 在高速通信环境[8]中的性能，但是收效甚微。此外，大部分改进协议在设计时没有考虑通信双方硬件资源和处理能力的差异，导致某些改进协议只适用于高性能的设备[9]，不能根据设备的不同实现差异化的数据通信。并且由于处理能力的差异，通信双方在进行数据通信的同时还需要兼顾数据处理等其他工作，现有的通信协议不能根据通信双方处理能力的差异进行动态调整[10]，降低了通信的效率。

针对上述问题，本文提出了支持差异化、可协商的数据通信机制。本文主要贡献如下。

1) 提出了参数协商方法，根据接收端计算能力和接收能力的不同，发送端与接收端协商合适的滑动窗口大小和同步包序号，实现差异化、可协商的数据通信。

2) 设计重传反馈机制，提出 3 种 ACK，分别是正常接收 ACK、重传 ACK 和丢弃 ACK。其中，正常接收 ACK 表示数据包被接收端正确接收，重传 ACK 表示数据包需要发送端重新发送，丢弃 ACK 表示数据包被接收端丢弃。正常接收 ACK 采用累计确认机制，重传 ACK 包含一个或多个需要重传的包序号，提高了数据重传的效率，同时 ACK 准确反馈接收端的数据接收状态，发送端根据接收端数据接收情况动态调整发送速率，进行数据发送和重传，有效缓解通信过程中的链路拥塞情况，降低通信的丢包率，提高通信效率和可靠性。

3) 发送端和接收端采用不同滑动窗口大小以不同通信速率进行多对多、并行异步的数据通信。通过服务器和计算单元架构模式进行实验仿真，实验结果证明了该机制能够根据接收端能力差异进行动态自适应的数据通信，提高了通信的可靠性和效率。

2 相关工作

由于云计算和物联网中各种应用对高性能数据通信的迫切需求以及传统数据通信协议本身在云环境和物联网中的不足，使研究适用于新环境下的高速数据通信方法成为热点，本文主要从数据通信方法本身入手，对相关研究进行论述。

He 等[11]提出可靠增强 UDP（RBUDP, reliable blast user datagram protocol），采用 UDP+TCP 的方式实现可靠传输，其中，UDP 用来传输数据，TCP 用来传输控制信息。其主要思路是发送端利用 UDP 发送所有数据，发送完成后利用 TCP 发送一个传输完成信号，接收端接收数据并记录，待接收到传输完成信号后利用 TCP 反馈数据接收情况，发送端根据反馈信息重新发送丢失的数据，重复上述过程直到所有数据被成功接收。Kachan 等[12]在 10 Gbit/s 链路上对 RBUDP 的性能进行了验证，其性能、资源消耗等均表现较好。但是 RBUDP 主要针对批量数据传输，不适用于流式应用；需要事先获取链路速度来决定发送端数据包的发送速度；对传输的数据量有限制，因为接收端需要缓存所有的数据，数据量过大可能导致接收端异常。Meiss[13]针对 RBUDP 的不足提出了 Tsunami 协议，其改进之处有以下两点：定时反馈数据接收情况，根据反馈信息动态调整发送时延或者速度以实现拥塞控制。Tsunami 协议解决了 RBUDP 对传输内容大小的限制，但其拥塞控制机制过于简单，在长距离网络传输中不能准确反映传输链路的拥堵状况。

Syzov 等[14]在高速数据通信环境下提出多线程

收发和自适应资源分配的通信方案，以解决单通路 UDP 数据收发时带宽利用率不足的问题。该方案在高速大数据量传输时性能表现很好，但是线程管理和冗余资源回收机制的不足导致该方案不能根据传输速度的变化动态调整系统资源的消耗。

基于 RUDP，Luo 等[15]针对航空自组织网络[16]中可靠高效数据传输问题提出了带喷泉码的可靠 UDP（FRUDP, fountain code and reliable UDP），Nasser 等[17]提出了一种紧凑型可靠 UDP（compact-RUDP, compact reliable UDP），陈波等[18]提出了一种新型可靠数据传输协议 ARUDP（argument reliable UDP），上述 3 种协议都是在可靠 UDP 基础上添加控制机制来降低通信过程中的丢包率，以达到提高通信效率的目的。Christensen 等[19]提出在现场可编程门阵列（FPGA, field programmable gate array）和数字信号处理器（DSP, digital signal processor）等硬件设备上实现可靠 UDP 通信，通过修改网络最大包长度来提高传输性能。

文献[20-21]针对高速长距离网络中大量数据传输的性能需求，提出了一种高速批量数据传输协议 DCUDP（double cubic UDP），通过三次方速率控制提高协议在高带宽产品网络上的可伸缩性，并采用随机算法来减轻连续丢失和丢失同步的影响。该协议在带宽利用率、CPU 资源占用率和协议公平性等方面均表现良好，但是在进入稳定状态之前会出现拥塞窗口抖动的现象，为数据传输带来了不稳定因素，并且对丢包的原因分析不透彻，同样会导致传输效率降低。

Gu 等[22]提出基于 UDP 的数据传输（UDT, UDP-based data transfer），其通过接收端周期性发送反馈信息，丢包时立即发送否定式确认信息来通知发送端丢包情况，采用随着减少增加的加性增乘性降（DAIMD, additive increase and multiplicative decrease with decreasing increase）算法来调整发送端的数据发送速率，同时 UDT 采用类似于 TCP 的滑动窗口机制来控制发送端，减少发送无意义数据包的数量，达到拥塞控制的效果。UDT 协议本身复杂度较高，这在很大程度上限制了协议本身的传输效率。

Eckart 等[23]提出了性能自适应的 UDP（PA-UDP, performance adaptive UDP），该协议主要针对文件传输而设计，通过简单比较剩余文件大小和接收端缓冲区可用空间来实现。PA-UDP 还通过数学模型来调节发送速率，避免接收端缓冲区溢出。但 PA-UDP 的数学模型出发点并不准确，即磁盘读写速率与数据传输速率的快慢情况不能完全确定，因此可能存在传输效率不高的问题。

现有的数据通信方法大致从两方面来设计拥塞避免算法，分别是丢包率和接收端缓冲区利用率，即根据丢包率或者接收端缓冲区利用率来调整发送端的数据发送速率，进而提高通信效率。但这些协议实现的复杂度较高，需要考虑的因素较多，对通信的性能影响较大，并且大多数改进协议是针对特定应用场景设计的，通用性不强。

除了对通信协议本身的研究之外，对 TCP/IP 栈在不同平台上实现的研究也较多。Sidler 等[24]在 Xilinx VC709 FPGA 开发板上设计并实现了协议栈，最多支持 10 000 个连接，万兆环境下通信速度达到 8.5 Gbit/s，但是资源消耗较大，尤其是块随机存取内存（BRAM, block random access memory）占用达到 21%，对在 FPGA 上实现的其他应用影响较大。在此基础上，Sidler 等[25]提出使用双倍速率同步动态随机存取内存（DDR SDRAM， double data rate synchronous dynamic random access memory）进行数据存取，内存消耗降低了 50%，虽然数据通信的时延降低了，但 FPGA 逻辑资源占用却有增无减，主要原因是需要额外消耗逻辑资源来存储控制 DDR SDRAM 存取数据的逻辑。因此，数据通信机制需要根据不同设备硬件资源的差异来动态调整用于数据通信的资源消耗。

本文所提数据通信机制从以下三方面进行研究：设计参数协商机制，根据设备处理能力的差异动态调整通信的资源消耗和通信速度；设计简洁高效的滑动窗口机制，保证通信的高效性；设计自定义协议，实现反馈确认和选择重传来保证通信的可靠性，根据接收端当前处理能力动态调整发送端的发送速度，降低通信链路的拥塞风险。

3　差异化、可协商的数据通信机制模型

本节以服务器与计算单元/终端设备应用场景为例，对服务器与计算单元之间通信的各类需求进行分析，提出了差异化、可协商的数据通信机制的模型。

3.1　应用场景分析

云计算环境下，服务器与计算单元或者服务器与终端设备通信的应用场景如图 1 所示。一个服务

器需要与多个计算单元或者终端设备进行并行、高效、可靠通信。而各计算单元或者终端设备的计算资源、带宽等均不相同，导致不同计算单元或者终端设备的数据处理能力具有很大的差异性。

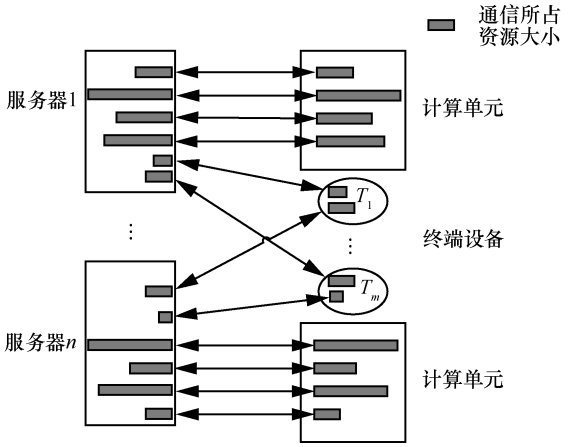


图 1 应用场景

假设计算单元为 FPGA 计算单元，每个 FPGA 计算单元功能不同，带宽不同，数据处理能力不同，每个 FPGA 计算单元可以承受的最大通信速率以及需要的通信速率也不相同。假设 FPGA 计算单元为密码计算单元，不同密码算法所需的计算资源不同，数据处理能力不同、带宽也不相同。服务器与 FPGA 计算单元进行通信时，需要根据每个 FPGA 计算单元的资源情况，采用合适的通信速率进行差异化数据通信。此外，实际通信过程中，服务器需要根据每个 FPGA 计算单元实际的数据接收情况和数据处理情况动态调整通信速率，及时准确判断数据丢包情况，进行数据发送和重传，降低丢包率，提高通信的效率和可靠性。

下面以密码服务为例阐述本文所提数据通信机制在密码服务系统中的应用。密码服务系统模型由 5 个模块组成，如图 2 所示。其中，属于串联关系的模块有数据接收、密码服务调度、密码服务汇聚和数据发送模块，负责整个密码作业数据流的处理工作；属于并联关系的模块是密码算法运算模块，密码算法运算模块由不同种类、不同数量的密码算法知识产权（IP, intellectual property）核组成，各个密码算法 IP 核之间是并联关系，即不同类型的密码算法处于并行工作状态，在密码算法运算过程中互不影响。

数据接收模块采用滑动窗口和选择重传机制保证密码服务数据包的正确性和完整性，提高数据传输的性能；密码服务调度模块将正确接收的密码服务数据包根据密码算法类型的不同分发到密码算法运算模块中不同的密码算法的 IP 核进行处理；密码服务汇聚模块对运算完成后的密码服务数据包进行聚合；数据发送模块同样采用滑动窗口和选择重传机制来保证数据传输的正确性和性能。

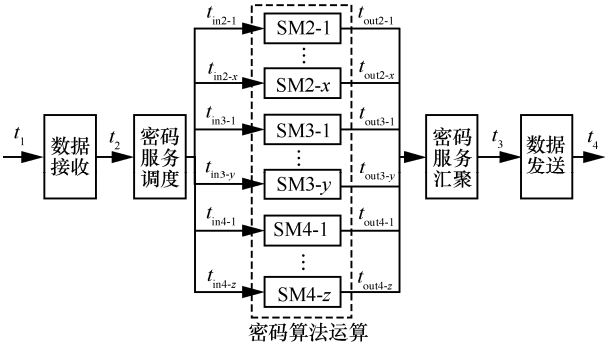


图 2 密码服务系统模型

令数据接收模块从接收数据包完成到开始接收下一个数据包的时间间隔为 t_1；密码服务调度模块从数据接收模块读取一个数据包完成到开始读取下一个数据包的时间间隔为 t_2；密码算法运算模块中的密码算法 IP 核从密码服务调度模块排队读取数据包的时间间隔为 t_{in}，根据算法 IP 核的不同分别表示为 t_{in2}、t_{in3}、t_{in4}；密码服务汇聚模块从密码算法 IP 核中取数据包的时间间隔为 t_{out}，根据算法 IP 核的不同分别表示为 t_{out2}、t_{out3}、t_{out4}；数据发送模块从密码服务汇聚模块读取数据包的时间间隔为 t_3；数据发送模块从发送数据包完成到开始发送下一个数据包的时间间隔为 t_4；则密码服务系统处理一个密码服务数据包的额外时间开销 Δt 为

$$\Delta t = t_1 + t_2 + \max(t_{\text{in2-}i}, t_{\text{in3-}j}, t_{\text{in4-}k}) + \max(t_{\text{out2-}i}, t_{\text{out3-}j}, t_{\text{out4-}k}) + t_3 + t_4 \tag{1}$$

其中，i=1,⋯,x，j=1,⋯,y，k=1,⋯,z。

为降低密码服务处理的额外时间开销，可充分利用 FPGA 并行工作的特性，算法 IP 核采用全流水的设计方式，处理模块之间使用先进先出（FIFO, first input first output）存储器进行解耦，实现数据包的不间断处理，使密码服务调度、密码算法运算和密码服务汇聚等模块间的时间间隔降为 0，即

$$t_2 = 0 \tag{2}$$

$$\max(t_{\text{in2-}i}, t_{\text{in3-}j}, t_{\text{in4-}k}) = 0 \tag{3}$$

$$\max(t_{\text{out2-}i}, t_{\text{out3-}j}, t_{\text{out4-}k}) = 0 \tag{4}$$

$$t_3 = 0 \tag{5}$$

则密码服务数据包的额外时间开销仅与数据发送和接收的时间相关，即

$$\Delta t = t_1 + t_4 \tag{6}$$

3.2　通信机制模型

针对上述应用场景，本文提出了支持差异化、可协商的数据通信机制。通信机制模型如图 3 所示。通信机制包含三部分，分别为参数协商、数据发送和数据接收。

参数协商是指通信开始之前，通信双方根据自身硬件资源大小和数据处理能力协商出合适的滑动窗口大小、同步包序号等关键参数。参数协商分为参数协商请求和参数协商反馈。参数协商请求由发送端发给接收端，内含发送端期望的接收端滑动窗口大小和同步包序号。参数协商反馈由接收端发给发送端，内含接收端滑动窗口大小和同步包序号。参数协商充分考虑不同数据通信终端硬件资源和数据处理能力的差异，通过协商避免因数据通信的原因影响设备其他功能的正常运转，同时避免因通信两端数据处理能力差异导致的数据包丢失，保证数据传输的高效性。

数据发送包括发送数据和处理 ACK。发送端根据自身当前滑动窗口状态和接收端当前数据处理情况综合判断是否满足数据发送条件，如果满足数据发送条件则发送数据，否则接收并处理接收端反馈的 ACK 信息。ACK 信息分为 3 种，分别是正常接收 ACK、重传 ACK 和丢弃 ACK。其中，正常接收 ACK 表示该包序号对应的数据包被接收端正确接收，正常接收 ACK 采用累计确认机制，即发送端收到某一正常接收 ACK，则表示该包序号之前的数据包均被接收端接收到。重传 ACK 表示该包序号对应的数据包需要发送端重新发送，重传 ACK 中可包含一个或多个待重传包序号，以提高数据重传效率。丢弃 ACK 表示该包序号对应的数据包不在接收端滑动窗口范围内，被接收端丢弃。数据发送根据反馈的 ACK 信息及时地对数据进行确认和重传，保证数据传输的稳定性和性能。

数据接收包括接收数据和反馈 ACK。接收端对接收的数据包进行解析，首先根据数据包序号和接收端滑动窗口状态确定数据包的接收状态，然后根据数据包的接收状态生成正常接收 ACK、重传 ACK 和丢弃 ACK。除了 ACK 信息，接收端还将数据处理情况反馈给发送端，使发送端据此动态调整数据发送速率，避免通信链路的拥塞，提高数据传输的高效性。

本文采用时间自动机对所提数据通信机制进行数学建模，定义 3 个模块，分别是参数协商、数据发送和数据接收模块。

3.2.1　参数协商模块

参数协商模块的时间自动机定义为 $A=<L_a, L_{a0}, \sum_a, X_a, I_a, E_a>$。

- $L_a = \{\text{Init,Request,Done}\}$
- $L_{a0} = \{\text{Init}\}$
- $\sum_a = \{\text{send\_request,recv\_reply}\}$
- $X_a = \{x\}$
- $I_a = I_a(x)$
- $E_a = L_a \sum_a \Phi(X_a) 2^{X_a} L_a$

其中，L_a 表示参数协商模块的位置集合，包含 3 个位置，Init 表示初始位置，Request 表示发起参数协商请求，Done 表示参数协商完成；L_{a0} 表示参数协商模块的起始位置，即 Init；$\sum_a$ 表示参数协商模块的有限字符集合，send_request 表示发送参数协商请求包，recv_reply 表示接收参数协商反馈包；X_a 表示参数协商模块的有限时钟集合；I_a 表示参数协商的映射，为 L_a 中的每一个位置指定 $\Phi(X_a)$ 的一个时钟约束；E_a 表示参数协商模块位置迁移关

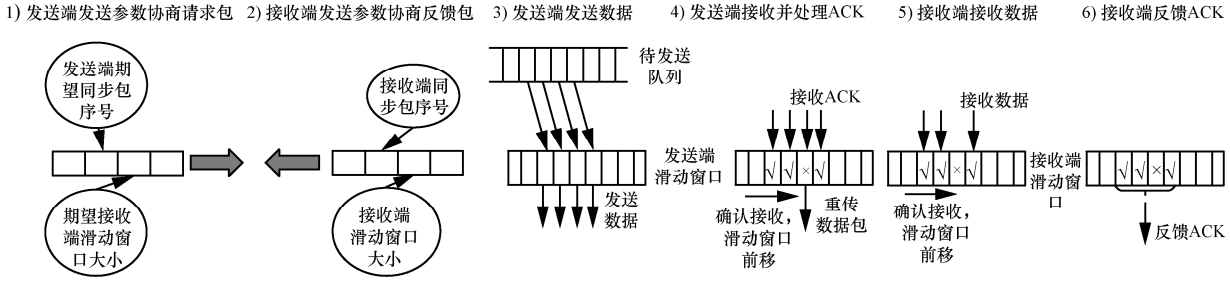


图 3　通信机制模型

系集合。

参数协商模块时间自动机如图 4 所示。通信双方开始参数协商时，发送 send_request 消息，消息中包含发送端期望同步包序号以及期望接收端滑动窗口大小，接收端收到 send_request 消息后根据自身接收能力和资源使用情况，确定接收端滑动窗口大小和同步包序号，并通过 recv_reply 消息返回给发送端，如果发送端在等待 MAX_DELAY 时间后仍未收到 recv_reply 消息，则返回初始状态。

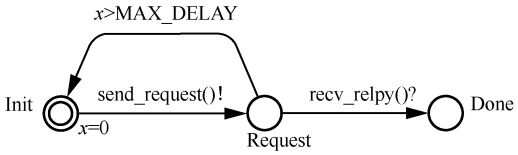


图 4 参数协商模块时间自动机

以本文实验所用的 FPGA 计算单元为例，设备当前接收能力由设备可用作滑动窗口的逻辑资源量决定，而滑动窗口在 FPGA 计算单元内部的实现主要由三部分组成，分别是块存储器 BRAM、查找表（LUT, look up table）和触发器（FF, flip flop）。这 3 种逻辑资源的剩余量决定了设备的当前接收能力。

令 FPGA 计算单元 BRAM 剩余量为 R，LUT 剩余量为 L，FF 剩余量为 F。设单位滑动窗口 BRAM 占用量为 α，LUT 占用量为 β，FF 占用量为 γ，滑动窗口大小为 w。则 w 需满足以下条件

$$\begin{cases} w\alpha < R \\ w\beta < L \\ w\gamma < F \end{cases}$$

由此可以推出 $w = \min(\frac{R}{\alpha}, \frac{L}{\beta}, \frac{F}{\gamma})$。

3.2.2 数据发送模块

数据发送模块的时间自动机定义为 $S =< L_s, L_{s0}, \Sigma_s, X_s, I_s, E_s >$。

- L_s = {Idle,SendData,RetransData, RecvAck,SendSuccess,UpdateStatus}
- L_{s0} = {Idle}
- Σ_s = {msg[id],ack[id]}
- $X_s = \{x\}$
- $E_s = L_s \Sigma_s \Phi(X_s) 2^{X_s} L_s$
- $I_s = I_s(x)$

其中，L_s 表示数据发送模块的位置集合，包含 6 个位置，Idle 表示空闲，SendData 表示发送数据，RetransData 表示重传数据，RecvAck 表示接收 ACK，SendSucess 表示数据发送成功，UpdateStatus 表示更新滑动窗口状态；L_{s0} 表示数据发送模块的起始位置，即 Idle；Σ_s 表示数据发送模块的有限字符集合，msg[id] 表示发送的数据内容，ack[id] 表示接收的 ACK；X_s 表示数据发送模块的有限时钟集合；I_s 表示数据发送的映射，为 L_s 中的每一个位置指定 $\Phi(X_s)$ 的一个时钟约束；E_s 表示数据发送模块位置迁移关系集合。

数据发送模块时间自动机如图 5 所示。数据发送时首先判断是否满足数据发送条件，即发送端滑动窗口未满并且接收端滑动窗口空闲节点数量大于正在发送的数据包数量。其中，发送端滑动窗口未满是指发送端滑动窗口中有空闲节点来发送新的数据包，如式(7)所示；接收端滑动窗口空闲节点数量是指接收端滑动窗口中可用来接收新数据包的空闲节点，如式(8)所示；正在发送的数据包数量是指发送端已经发出但尚未收到确认的数据包数量，如式(9)所示。

$$w + 1 \neq r \tag{7}$$

$$\text{num}_{\text{sending}} = \text{sseq} - \text{cseq} \tag{8}$$

$$\text{num}_{\text{free}} = \text{WIN} - \text{rseq} + \text{pseq} \tag{9}$$

然后根据接收端反馈的 ACK 包来解析正常接收数据包、重传数据包和丢弃数据包，进行数据包的确认和重传，并获取接收端的接收包序号和处理包序号，以此判断接收端当前的数据接收能力，进而动态调整数据发送速率，降低通信中的拥塞情况和丢包率，提高通信效率。

3.2.3 数据接收模块

数据接收模块的时间自动机定义为 $R =< L_r, L_{r0}, \Sigma_r, X_r, I_r, E_r >$

- L_r = {Idle,RecvData,SendAck}
- L_{r0} = {Idle}
- Σ_r = {msg[id],ack[id]}
- $X_r = \{x\}$
- $E_r = L_r \Sigma_r \Phi(X_r) 2^{X_r} L_r$
- $I_r = I_r(x)$

其中，L_r 表示数据接收模块的位置集合，包含 3 个位置，Idle 表示空闲，RecvData 表示接收数据，

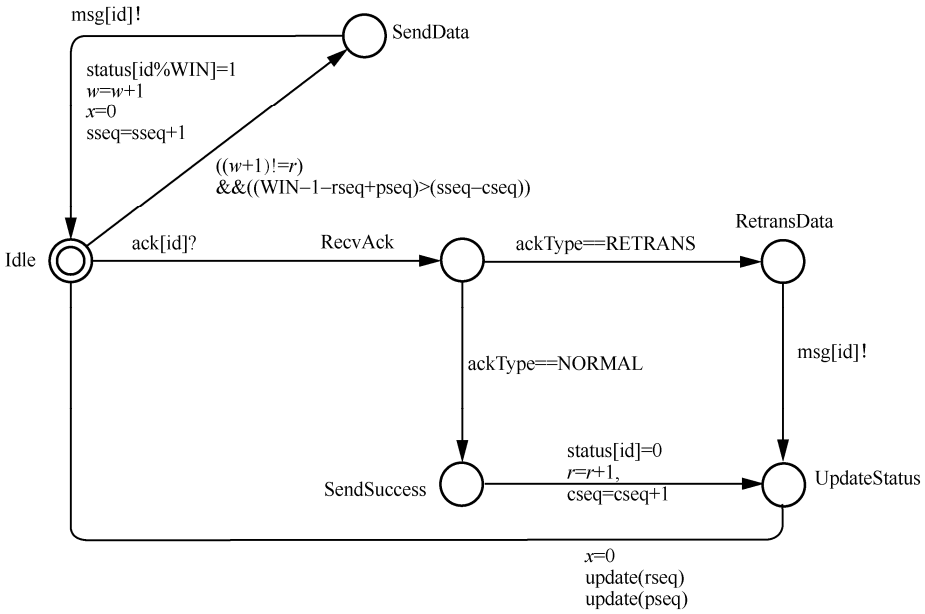


图 5　数据发送模块时间自动机

SendAck 表示发送 ACK；L_{r0} 表示数据接收模块的起始位置，即 Idle；$\sum_r$ 表示数据接收模块的有限字符集合，msg[id] 表示接收的数据内容，ack[id] 表示发送的 ACK；X_r 表示数据接收模块的有限时钟集合；I_r 表示数据接收的映射，为 L_r 中的每一个位置指定 $\Phi(X_r)$ 的一个时钟约束；E_r 表示数据接收模块位置迁移关系集合。

数据接收模块时间自动机如图 6 所示。数据接收模块主要作用是接收数据，并根据数据接收情况返回 ACK。接收端根据当前滑动窗口状态对接收的数据包进行判断，将数据包标记为正常接收数据包、重传数据包和丢弃数据包，并结合当前接收端处理能力的变化生成 ACK 返回给发送端。通过反馈接收端的数据处理能力，通信双方能够自动适应通信速度的变化。

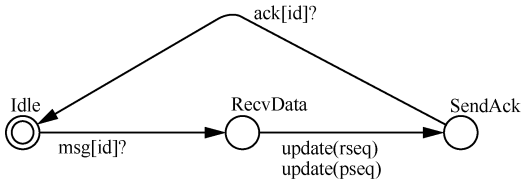


图 6　数据接收模块时间自动机

4　差异化、可协商的数据通信流程

发送端和接收端在完成参数协商之后进行数据发送和接收。本文设计了 ACK 包，在实际通信时，接收端根据数据接收情况生成 ACK 包。发送端根据 ACK 包及时了解接收端数据的接收和处理情况，动态调整数据发送速率；解析需要重传的数据，对需要重传的数据及时进行重传。此外，本文设计了发送端滑动窗口状态和接收端滑动窗口状态，分别表示发送端和接收端数据发送和接收情况。

4.1　参数协商

发送端和接收端在数据通信前，通过参数协商得到双方合适的滑动窗口大小和同步包序号。

4.1.1　参数协商中的包结构

参数协商主要根据参数协商包和参数协商反馈包进行协商。

参数协商包结构可表示为

AgreePkt={ExpectAgreeSeq,ExpectWindowSize}

其中，ExpectAgreeSeq 表示期望同步包序号，ExpectWindowSize 表示期望接收端滑动窗口大小。

参数协商反馈包结构可表示为

AgreeRespPkt={RecvAgreeSeq,RecvWindowSize}

其中，RecvAgreeSeq 表示接收端同步包序号，RecvWindowSize 表示接收端滑动窗口大小。

4.1.2　参数协商流程

发送端设置期望同步包序号、期望接收端滑动窗口大小，构造参数协商包发送给接收端。

接收端收到参数协商包后，解析出参数协商包

中的期望同步包序号和发送端期望滑动窗口大小。接收端根据参数协商包中的期望同步包序号确定同步包序号，根据自身接收能力和资源使用情况、发送端期望滑动窗口大小等确定接收端滑动窗口大小；然后根据同步包序号和接收端滑动窗口大小构造参数协商包反馈包返回给发送端。

发送端接收到参数协商反馈包后，解析并记录同步包序号和接收端滑动窗口大小，然后确定双方共同的同步包序号和滑动窗口大小。

发送端和接收端采用参数协商步骤协商出通信双方合适的滑动窗口大小、同步包序号等关键参数，使通信双方以一个合适的速度开始数据通信，避免了通信刚开始时速率过快或者过慢导致的通信效率低下。此外，参数协商机制充分考虑通信双方资源使用情况和处理能力的差异，针对不同的设备采用不同的速率进行通信，实现差异化、可协商的数据通信。

4.2 数据通信

发送端和接收端完成参数协商之后即可进行正常的数据通信。

4.2.1 数据通信中的包结构

数据通信主要包括数据包和 ACK 包。数据包表示待传输的有效数据，ACK 包表示接收端根据数据接收和处理情况生成的反馈包。

数据包结构可表示为

DataPkt={Seq,Cmd,Length,Data}

其中，Seq 表示数据包的包序号，Cmd 表示数据包的命令字段，Length 表示数据包中 Data 的长度，Data 表示待传输的数据。

ACK 包结构可表示为

$$\begin{aligned}&\text{AckPkt}=\{\text{RecvSeq},\text{ProcessSeq},\\&\{\text{NormalAck}_i,\text{RetransAck}_j,\text{AbortAck}_k\}^*\mid\\&i\geqslant 0,j\geqslant 0,k\geqslant 0,\text{且}i+j+k\geqslant 1\}\end{aligned}$$

其中，RecvSeq 表示一段时间内接收端滑动窗口内已确认接收的连续数据包的最后包序号；ProcessSeq 表示一段时间内接收端滑动窗口内已处理的连续数据包的最后包序号；NormalAck 表示正常接收数据包集合，即数据包按序正确到达接收端；RetransAck 表示重传数据包集合，即在传输过程中被丢弃或者出现错误的、需要发送端重新发送的数据包；AbortAck 表示丢弃数据包集合，即不在接收端接收范围内，被接收端丢弃的数据包。

4.2.2 数据通信流程

数据发送时，发送端设计了待发送队列、待确认队列、ACK 队列，分别用来保存待发送的数据包、待确认的数据包以及接收端反馈的 ACK 包。

当满足数据包发送条件时，发送端从待发送队列中取出一个或多个待发送的数据包按照 4.2.1 节数据包结构来构造数据包发送给接收端，并保存到待确认队列中。

当不满足数据包发送条件时，发送端从 ACK 包队列中取出一个 ACK 包，解析接收端接收包序号、接收端处理包序号。读取 ACK 包中正常接收数据包集合，将正常数据包序号对应的数据包从发送端待确认队列中删除；读取 ACK 包中的重传数据包集合，将重传数据包序号对应的数据包从发送端待确认队列中重新发送给接收端；读取 ACK 包中的丢弃数据包集合，如果丢弃数据包序号对应的数据包在发送端待确认队列中，继续判断发送端滑动窗口状态是否正常，如果发送端滑动窗口状态不正常，调整发送端滑动窗口状态后，再将发送端待确认队列中对应的数据包发送给接收端，如果丢弃数据包序号对应的数据包不在发送端待确认队列中，不作任何处理。

数据发送完成或者 ACK 包处理完成后，更新发送端滑动窗口状态。

发送端滑动窗口状态可表示为

SendWindow=
{SendRptr,SendWptr,SendSeq,SendStatus,RecvSeq,
ProcessSeq,SendConfirmSeq,SendWindowSize}

其中，发送端读指针 SendRptr 表示发送端滑动窗口的下界；发送端写指针 SendWptr 表示发送端滑动窗口的上界；发送端发送包序号 SendSeq 表示发送端一段时间内已经发送的数据包的最后包序号；数据包发送状态 SendStatus 表示在滑动窗口内数据包的发送状态，数据包发送状态包括已发送、已确认接收；接收端接收包序号 RecvSeq 表示一段时间内接收端滑动窗口内已确认接收的连续数据包的最后包序号；接收端处理包序号 ProcessSeq 表示一段时间内接收端滑动窗口内已处理的连续数据包的最后包序号；发送端确认包序号 SendConfirmSeq 表示一段时间内发送端已经确认发送成功的连续数据包的最后包序号；发送端滑动窗口大小 SendWindowSize 表示发送端滑动窗口所占资源空间的大小。

数据接收时，接收端根据数据接收和处理情况生成 ACK 包反馈给发送端。

首先，接收端解析接收到的数据包，获取数据包序号，并判断包序号是否在接收端滑动窗口范围内，如果不在范围内，则标记该包序号为丢弃数据包序号。如果在接收端滑动窗口范围内，则搜索接收端滑动窗口处理包序号至当前数据包序号之间是否有数据包的状态为未收到，如果接收端滑动窗口处理包序号至当前数据包序号之间的数据包都收到，则将接收端滑动窗口处理包序号至当前数据包序号之间的数据包序号标记为正常接收数据包序号；如果接收端滑动窗口处理包序号至当前数据包序号之间有数据包的状态为未收到，则标记未收到的数据包序号为重传数据包序号，其余为正常数据包序号。然后，根据数据包的接收状态生成正常接收数据包集合、重传数据包集合和丢弃数据包集合，将接收端接收包序号、接收端处理包序号、正常接收数据包集合、重传数据包集合和丢弃数据包集合按照 4.2.1 节 ACK 包结构来构造 ACK 包发送给发送端。最后，更新接收端滑动窗口状态。

接收端滑动窗口状态可表示为

RecvWindow={RecvRptr,RecvWptr,
RecvSeq,ProcessSeq,RecvStatus, RecvWindowSize}

其中，接收端读指针 RecvRptr 表示接收端滑动窗口的下界；接收端写指针 RecvWptr 表示接收端滑动窗口的上界；接收端接收包序号 RecvSeq 表示一段时间内接收端滑动窗口内已确认接收的连续数据包的最后包序号；接收端处理包序号 ProcessSeq 表示一段时间内接收端滑动窗口内已处理的连续数据包的最后包序号；数据包接收状态 RecvStatus 表示在接收端滑动窗口内的数据包的接收状态，数据包接收状态包括收到、未收到；接收端滑动窗口大小 RecvWindowSize 表示接收端滑动窗口所占资源空间的大小。

数据通信时，发送端根据数据包发送条件，充分考虑接收端接收速率的变化，动态调整发送端的数据包发送速率，能够有效降低数据传输的丢包率；同时，充分利用了接收端反馈的 ACK 包，对发送端待确认队列中的数据包进行确认和重传，发送端根据接收端不同接收能力和状态动态自适应地发送和重传，提高了通信效率和可靠性。

数据通信时，发送端可以给不同接收端发送数据，接收端可以同时作为发送端，原来的发送端也可以同时作为接收端，进行双向通信，即一台设备或一个系统或一个组件或一个线程或一个进程中可以同时部署发送端和接收端，以实现发送端和接收端的多对多、并行、全双工、双向通信。

5　实验与结果分析

5.1　实验环境

为验证本文所提通信机制的性能，本文在服务器−计算单元的架构中实现了该通信机制，包括一台 X86 平台的服务器和一套高级通信计算架构（ATCA, advanced telecommunications computing architecture）平台系统。其中，服务器使用 Intel 公司的 82599ES 10 Gbit/s 网卡或者 Mellanox 公司的 100 Gbit/s 网卡进行数据通信；ATCA 平台系统由交换板和业务板组成，其架构如图 7 所示，交换板和业务板之间、业务板和子卡之间均通过背板数据总线进行数据交换，子卡上每个 FPGA 计算单元的带宽为 10 Gbit/s。

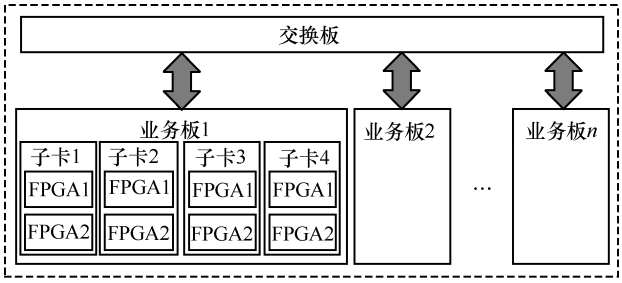


图 7　ATCA 平台系统架构

服务器和 ATCA 平台系统通过 10 Gbit/s 多模光纤或者 100 Gbit/s 电缆进行数据通信。实验所用的软硬件环境如表 1 所示。本文实验是在服务器和 FPGA 计算单元上分别使用 C 语言和 Verilog 语言实现的，服务器端使用数据平面开发套件（DPDK, data plane development kit）接收和发送网络数据包。

5.2　通信机制容错效果分析

容错实验的基本原理是服务器端向 FPGA 计算单元发送文件，FPGA 计算单元收到文件后将文件返回给服务器端，服务器端在发送文件内容的过程中随机丢弃若干个数据包，最后查看服务器端接收到的文件和发送的文件内容是否一致，以此来检验通信协议的重传机制是否有效，同时检验通信协议在链路拥塞情况比较严重的情况下的通信效率。实

验共设置 4 组丢包测试，丢包率依次增加，实验结果如表 2 所示。

表 1 实验环境

设备名称	规格说明
服务器	OS：CentOS7.7 DPDK：18.11.2-stable CPU：Intel Xeon Gold 6246 3.3 GHz 内存：DDR3 256 GB
100 Gbit/s 网卡	Mellanox ConnectX-5
10 Gbit/s 网卡	Intel 82599ES
100 Gbit/s 电缆线	华为 QSFP28+高速电缆
6UATCA 平台系统	恒光 6U6SAC，一个交换槽位，5 个业务槽位
交换板	时代通信 AS100
业务板	ATCA1400，共 4 块，每块可插 4 块 AMC 子卡
AMC 子卡	共 32 块，每块含 2 个 FPGA 计算单元，型号均为 Xilinx XC7K325T

表 2 通信协议容错实验结果

丢包率	传输耗时/s	传输速度/(Gbit·s^{-1})
1%	77.60	7.62
5%	86.09	5.70
10%	115.17	5.29
20%	124.12	3.97

从表 2 可以看出，随着丢包率的增加，文件传输所需时间逐渐增加，传输速度随之降低。但即使丢包率达到 20%，即模拟通信系统网络严重拥塞的情况下，本文所提通信协议依然能够正确完成数据传输任务，并且传输速度仍然维持在较高的水平，实验证明本文所提通信协议具有较强的容错能力，并且通信效率高。

5.3 通信机制性能分析

5.3.1 转发性能分析

转发实验的基本原理是在测试服务器和 FPGA 计算单元分别设置不同大小的滑动窗口，测试服务器分别使用不同数据包大小发送大量数据给 FPGA 计算单元，FPGA 计算单元收到数据之后再转回测试服务器，数据传输完成后根据传输数据量和所用时间计算通信速率。

转发实验共配置 2 种实验环境，一种是测试服务器通过 10 Gbit/s 网卡与 FPGA 计算单元进行通信，另一种是测试服务器通过 100 Gbit/s 网卡与 FPGA 计算单元进行通信。每种环境在实验时共设置 8 组滑动窗口大小，5 种数据包大小。

实验结果分别如图 8 和图 9 所示。从实验结果可以看出，随着通信两端滑动窗口大小和数据包数据长度的增加，通信速率也随之增长。当滑动窗口大小较小时，通信速率较低，旨在模拟运算性能较差，或者通信能力较弱的设备，如物联网通信设备；在滑动窗口大小为 14 和 16 时，通信速率达到最大值，接近 10 Gbit/s 网口的线速。使用 100 Gbit/s 网卡通信时的最高速度与使用 10 Gbit/s 网卡通信时的速度基本一致，是因为 FPGA 计算单元的物理网口的规格是 10 Gbit/s，达到了端口性能的上限。

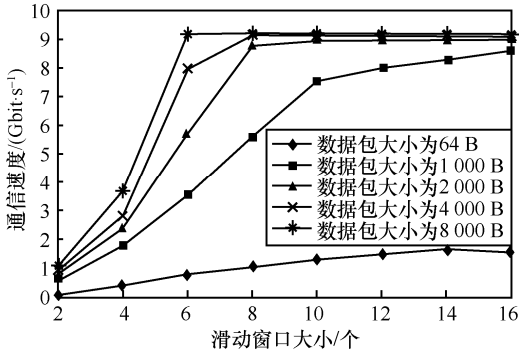


图 8 通信协议速率 10 Gbit/s 网卡实验结果

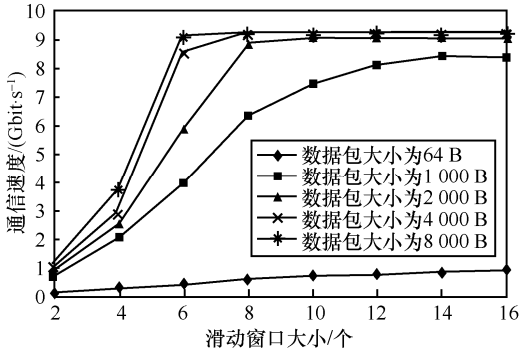


图 9 通信协议速率 100 Gbit/s 网卡实验结果

搭建实验环境，将 2 个测试服务器通过 10 Gbit/s 光纤连接，并在 2 个服务器上对 UDTv4 协议、RBUDP 和本文所提通信协议进行对比实验，通过设置丢包率来模拟网络环境的变化，以测试在不同网络环境下 3 种协议的性能，实验结果如图 10 所示。

UDTv4 协议在无丢包的情况下通信速率仅有 4 Gbit/s，在 0.1%丢包率的情况下通信速率降低到不足 200 Mbit/s，性能较差。RBUDP 的性能与本文所提机制较为接近，但随着丢包率的增大，RBUDP 与本文所提通信机制的性能差异逐渐增大。

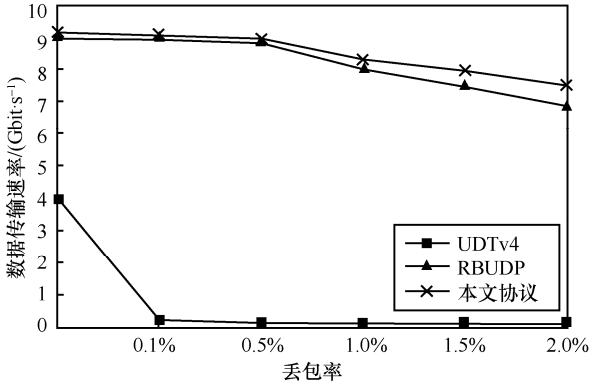


图 10　10 Gbit/s 网络环境下 3 种通信机制的性能对比

在上述实验基础上，本文搭建测试环境，测试服务器通过 100 Gbit/s 网卡与 8 个 FPGA 计算单元同时进行通信，每个 FPGA 计算单元均设置滑动窗口大小为 16，数据包大小为 1 000 B，测试结果如图 11 所示。

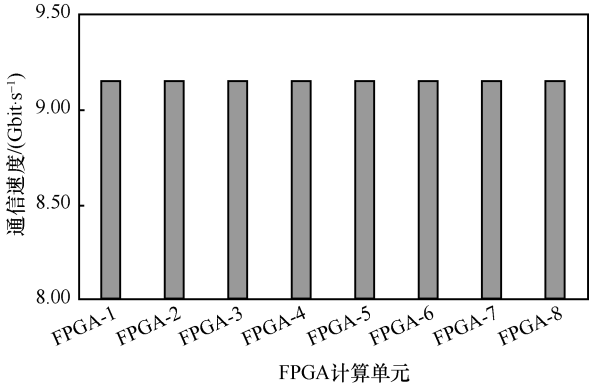


图 11　通信协议速率 100 Gbit/s 网卡多 FPGA 计算单元实验结果

由于每个 FPGA 型号都相同，计算资源和处理能力一致，因此每个 FPGA 计算单元的通信速率均约为 9.15 Gbit/s，接近 10 Gbit/s 网口的线速，从实验结果可以看出，本文所提通信机制在多数据流同时通信时具有很好的公平性。

此外，按照 3.1 节所述密码服务系统模型搭建测试环境，验证本文所提数据通信机制的稳定性和性能，并在所提数据通信机制的基础上验证整机对外提供密码服务的能力。单个 FPGA 计算单元的商用密码 4 算法电码本（ECB, electronic codebook）模式加解密速度为 7.34 Gbit/s。由于密码算法运算的计算开销会导致密码运算的性能比转发的性能低，随着 FPGA 计算单元数量的增加密码服务系统整体的商用密码 4 算法 ECB 模式加解密性能基本达到了线性增长，整机性能达到 234.88 Gbit/s。

5.3.2　资源占用情况分析

表 3 列出了文献[24-25]方案和本文方案在 XC7K325T 芯片上的资源占用情况。表 3 数据显示，本文方案所占硬件资源总体较少，既节省了硬件资源，又能保证数据通信的可靠性、稳定性和高性能。

表 3　　XC7K325T 资源占用情况

方案	BRAM/个	LUT/个	FF/个
文献[24]方案	279	19 026	20 611
文献[25]方案	353	22 468	24 744
本文方案	133	14 971	22 493

6　结束语

本文针对云计算、物联网中，服务器与海量差异化的计算单元和终端设备之间高速率、高可靠性、低时延、自适应的数据通信需求，设计了支持差异化、可协商的数据通信机制；通过参数协商根据接收端能力的不同，协商出合适的滑动窗口大小和同步包序号，采用不同通信速率，实现差异化可协商的数据通信；设计重传反馈机制，根据 ACK 准确反馈接收端数据接收状态，发送端动态自适应地进行数据的发送和重传，有效缓解通信过程中的拥塞情况，降低丢包率，提高通信可靠性和效率。本文在服务器-FPGA 计算单元架构上进行实验，对本文设计的数据通信机制的有效性和性能进行了测试，结果表明所提机制能够根据计算单元能力的不同，进行差异化、并行、高效、可靠的数据通信。本文所提通信机制已应用在某密码设备中，验证了通信机制的高效性。

参考文献：

[1] ZHOU Y Q, LIU L, WANG L, et al. Service-aware 6G: an intelligent and open network based on the convergence of communication, computing and caching[J]. Digital Communications and Networks, 2020, 6(3): 253-260.

[2] AL-SARAWI S, ANBAR M, ALIEYAN K, et al. Internet of things (IoT) communication protocols[C]//Proceedings of 2017 8th International Conference on Information Technology (ICIT). Piscataway: IEEE Press, 2017: 685-690.

[3] CHAI L, REINE R. Performance of UDP-lite for IoT network[J]. IOP Conference Series: Materials Science and Engineering, 2019, 495: 012038.

[4] DIZDAREVIC J, CARPIO F, JUKAN A, et al. A survey of communication protocols for Internet of things and related challenges of fog and cloud computing integration[J]. ACM Computing Surveys, 2019, 51(6): 1-29.

[5] ZHOU Y Q, TIAN L, LIU L, et al. Fog computing enabled future mobile communication networks: a convergence of communication and computing[J]. IEEE Communications Magazine, 2019, 57(5): 20-27.

[6] SASAKI Y, YOKOTANI T. Performance evaluation of MQTT as a

communication protocol for IoT and prototyping[J]. Advances in Technology Innovation, 2019, 4(1): 21-29.

[7] 王伟杭, 任勇毛, 唐明洁, 等. 终端性能自适应传输协议[J]. 软件学报, 2010, 21(7): 1635-1645.
WANG W H, REN Y M, TANG M J, et al. End-system performance aware transport protocols[J]. Journal of Software, 2010, 21(7): 1635-1645.

[8] FAIRHURST G, TRAMMELL B, KUEHLEWIND M. Services provided by IETF transport protocols and congestion control mechanisms[R]. RFC Editor, 2017.

[9] AL-DHIEF F T, SABRI N, LATIFF N M A, et al. Performance comparison between TCP and UDP protocols in different simulation scenarios[J]. International Journal of Engineering & Technology, 2018, 7(4): 172-176.

[10] WHEEB A H. Performance evaluation of UDP, DCCP, SCTP and TFRC for different traffic flow in wired networks[J]. International Journal of Electrical and Computer Engineering, 2017, 7(6): 3552.

[11] HE E, LEIGH J, YU O, et al. Reliable blast UDP: predictable high performance bulk data transfer[C]//Proceedings of IEEE International Conference on Cluster Computing. Piscataway: IEEE Press, 2002: 317-324.

[12] KACHAN D, SIEMENS E. Comparison of contemporary protocols for high-speed data transport via 10 Gbps WAN connections[C]//Proceedings of the 2nd International Conference on Applied Innovations in IT. Berlin: Springer, 2014: 21-27.

[13] MEISS M R. Tsunami: a high-speed rate-controlled protocol for file transfer[R]. Indiana University, 2004.

[14] SYZOV D, KACHAN D, SIEMENS E. High-speed UDP data transmission with multithreading and automatic resource allocation[C]//Proceedings of the 4th International Conference on Applied Innovations in IT. Berlin: Springer, 2016: 51-55.

[15] LUO Q, WANG J F. FRUDP: a reliable data transport protocol for aeronautical ad hoc networks[J]. IEEE Journal on Selected Areas in Communications, 2018, 36(2): 257-267.

[16] ZHENG Z G, SANGAIAH A K, WANG T. Adaptive communication protocols in flying ad hoc network[J]. IEEE Communications Magazine, 2018, 56(1): 136-142.

[17] NASSER B, RABANI A, FREILING D, et al. An adaptive telerobotics control for advanced manufacturing[C]//Proceedings of 2018 NASA/ESA Conference on Adaptive Hardware and Systems. Piscataway: IEEE Press, 2018: 82-89.

[18] 陈波, 陶威, 王运明. 基于ARUDP的指挥控制网络数据传输协议[J]. 火力与指挥控制, 2016, 41(4): 157-160.
CHEN B, TAO W, WANG Y M. Command and control network data transmission protocol based on ARUDP[J]. Fire Control & Command Control, 2016, 41(4): 157-160.

[19] CHRISTENSEN M J, RICHTER T. Achieving reliable UDP transmission at 10 GB/s using BSD socket for data acquisition systems[J]. Journal of Instrumentation, 2020, 15(9): T09005.

[20] ZHANG X, GU N J, SU J J. DCUDP: scalable data transfer for high-speed long-distance networks[J]. Concurrency and Computation: Practice and Experience, 2017, 29(4): e3846.

[21] 张旭. 面向网络服务的传输协议设计与任务调度优化[D]. 合肥: 中国科学技术大学, 2017.
ZHANG X. Transfer protocol design and task scheduling optimization on network service system[D]. Hefei: University of Science and Technology of China, 2017.

[22] GU Y H, GROSSMAN R L. UDT: UDP-based data transfer for high-speed wide area networks[J]. Computer Networks, 2007, 51(7): 1777-1799.

[23] ECKART B, HE X B, WU Q S. Performance adaptive UDP for high-speed bulk data transfer over dedicated links[C]//Proceedings of 2008 IEEE International Symposium on Parallel and Distributed Processing. Piscataway: IEEE Press, 2008: 1-10.

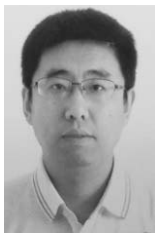
[24] SIDLER D, ALONSO G, BLOTT M, et al. Scalable 10Gbps TCP/IP stack architecture for reconfigurable hardware[C]//Proceedings of 2015 IEEE 23rd Annual International Symposium on Field-Programmable Custom Computing Machines. Piscataway: IEEE Press, 2015: 36-43.

[25] SIDLER D, ISTVÁN Z, ALONSO G. Low-latency TCP/IP stack for data center applications[C]//Proceedings of 2016 26th International Conference on Field Programmable Logic and Applications. Piscataway: IEEE Press, 2016: 1-4.

[作者简介]

寇文龙（1990− ），男，河南许昌人，西安电子科技大学博士生，主要研究方向为信息安全。

李凤华（1966− ），男，湖北浠水人，博士，中国科学院信息工程研究所研究员、博士生导师，主要研究方向为网络与系统安全、信息保护、隐私计算。



董秀则（1976− ），男，山东莒县人，北京电子科技学院副教授，主要研究方向为信息安全、密码工程。

曹晓刚（1996− ），男，河北邢台人，中国科学院信息工程研究所博士生，主要研究方向为信息安全。

耿魁（1989− ），男，湖北红安人，博士，中国科学院信息工程研究所高级工程师、硕士生导师，主要研究方向为网络安全、信息保护。

李青（1973− ），男，陕西宝鸡人，中国电信股份有限公司研究院高级工程师，主要研究方向为电信增值业务平台及产品开发、网络演进、云计算/大数据等运营技术。