Computer Science and Application 计算机科学与应用, 2021, 11(6), 1689-1697
Published Online June 2021 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2021.116174

Hans汉斯

智能终端海量数据采集与实时分析设计和应用研究

罗健飞[1]，吴　飞[1]，邢亚东[1]，张卫庆[1]，夏代江[1]，吴仲城[2]

[1]安徽中科美络信息技术有限公司智能终端研发中心，安徽　合肥
[2]中国科学院，强磁场科学中心，安徽　合肥
Email: jfluo@mail.ustc.edu.cn

收稿日期：2021年5月14日；录用日期：2021年6月11日；发布日期：2021年6月18日

摘　要

针对各种场景下的海量终端的部署，本文设计了一套海量数据采集与实时分析系统，具体在数据采集模块，通过借助Kafka消息队列，实现数据的高并发接入；在数据分析模块，借助大数据流处理系统Storm，在保证高可靠性的前提下，实现数据的实时处理，并通过相应的优化设计，解决海量终端接入网络时的高并发访问与数据处理需求；通过可视化设计以及实验验证本文方法的有效性，系统具有低延迟，高吞吐，可拓展等特点，能够满足车联网海量数据处理要求，具有很强的实用价值，目前本文提出的方法已经应用在实际场景中，为20多万台北斗定位终端提供服务。

关键词

智能终端，海量数据，实时分析

Design and Application of Massive Data Acquisition and Real-Time Analysis of Intelligent Terminal

Jianfei Luo[1], Fei Wu[1], Yadong Xing[1], Weiqing Zhang[1], Daijiang Xia[1], Zhongcheng Wu[2]

[1]Term R & D Centre, Anhui Zhongkemeiluo Info-Tech Co. Ltd., Hefei Anhui
[2]High Magnetic Field Laboratory, Chinese Academy of Sciences, Hefei Anhui
Email: jfluo@mail.ustc.edu.cn

Received: May 14th, 2021; accepted: Jun. 11th, 2021; published: Jun. 18th, 2021

文章引用: 罗健飞, 吴飞, 邢亚东, 张卫庆, 夏代江, 吴仲城. 智能终端海量数据采集与实时分析设计和应用研究[J]. 计算机科学与应用, 2021, 11(6): 1689-1697. DOI: 10.12677/csa.2021.116174

Abstract

For the deployment of mass terminals in various scenarios, this paper designs a set of mass data acquisition and real-time analysis system. In the data acquisition module, with the help of Kafka message queue, the high concurrent access of data is realized; in the data analysis module, with the help of the big data stream processing system storm, the real-time data processing is realized on the premise of high reliability, and through the corresponding optimization design, the high concurrent access and data processing requirements of massive terminals accessing the network are solved; through the visual design and experimental verification, the effectiveness of this method, the system has the characteristics of low latency, high throughput, scalability, and can meet the requirements of massive data processing in the Internet of vehicles, which has strong practical value. At present, the method proposed in this paper has been applied in the actual scene, providing services for more than 200,000 BD-based positioning terminals.

Keywords

Intelligent Terminal, Massive Data, Real-Time Analysis

1. 引言

随着车联网，以及 5G 网络应用的大规模部署，各种场景下接入云端的终端类型越来越多，例如在典型车联网应用中，涉及到包括车辆位置、状态、速度、加速度、路网、图片、视频等非结构化网络数据，数据量已经达到 TP 甚至 PB 量级，如此海量数据接入服务端时，一方面要求服务端高可用性，另一方面大数据的存储和处理具备实时性[1] [2] [3]，以满足场景应用需求。

现有的系统在处理上述应用中，主要存在：海量终端接入和数据解析的稳定性、数据库无法承载大数据存储以及处理数据时存在吞吐量小，实时性差的问题，本文针对这些问题，从层级精简、功能模块划分等系统架构方面，以及构建高并发数据队列和搭建实时分析框架，和系统性能优化等维度，对现有系统进行升级重构，使得系统满足海量终端接入时的高并发访问与数据处理的实时性要求，并具有高可用、高稳定性和可维护性等优势。

2. 系统设计

本文在传统系统结构上，采用模块化思路，并将模块进行精简，主要分为两大重要模块，一个是数据采集模块包括数据采集接收和报文解析转发，另一个是实时分析模块包括构建基于 Storm 的实时分析框架，实现内容分析和数据存储，应用可视化则主要面向应用场景，通过可视化方式实现数据业务化。图 1 为系统架构模块图，三大模块组成部分的主要功能：

1) 数据采集模块：实现对多源智能终端(支持多协议，如 JT808、私有化协议等)建立网络连接，基于协议中间件完成终端的适配接入，建立终端数据分类，完成数据并发接入。

2) 实时分析模块：对数据进行实时分析计算，借助大数据流处理系统 Storm，在保证高可靠性的前提下，实现数据的实时处理；将实时处理结果存储在分布式文件系统中，实现高速读写。

3) 应用可视：通过数据分析，结合不同应用场景，将结果以可视化的形式展现，典型通过图标等方式动态呈现，通过数据指标辅助用户决策。

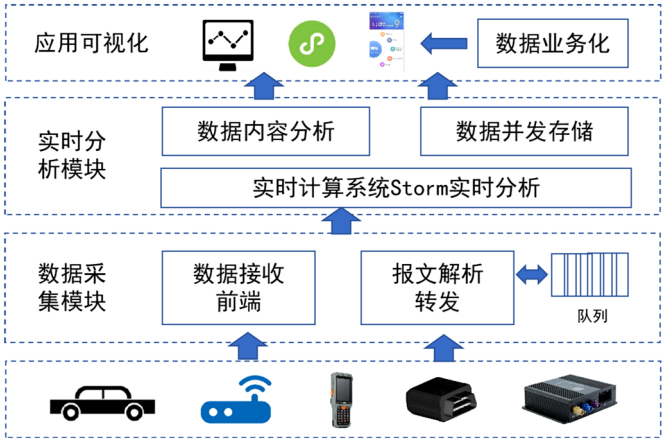

Figure 1. System architecture module diagram
图 1. 系统架构模块图

2.1. 数据采集模块

数据采集模块包括数据接入和数据转发，其中数据接入主要负责接收多源智能终端发送的多协议数据包，一般通过 4G/5G 无线网络实现终端侧和服务端的连接，服务端支持多源协议解析。

面对海量数据接入产生的高并发连接需求，采用 Boost.Asio [4]技术来搭建服务端接口程序，通过集成异步接口函数来实现全异步的事件处理，完成海量数据的可靠接入以及稳定连接。其中，采用 Boost.Asio 来实现前端数据接收服务，是因为在统一的接口层之下，Boost.Asio 提供了大量的类来支持不同的平台(Windows、Unix……)、不同的 IO 类型(同步、异步)及 IO 模型(IOCP、Select、Poll)及网络协议(TCP，UDP，ICMP)，具有很好的跨平台性和支持多种网络连接模式。

海量数据的接入，需要解决大量 TCP 请求所引发的高并发难题，终端数据接收与解析模块之间设计了一个数据缓冲与转发的中转站，通过消息队列实现，同时完成对数据解耦。数据缓冲层的消息队列通过 Kafka 来搭建，实现数据缓存与转发，相对于 RabbitMQ，Kafka 具备根据需求扩展、大吞吐等特点，更加符合海量车联网数据应用场景[5] [6] [7]。数据缓冲层架构如图 2 所示。

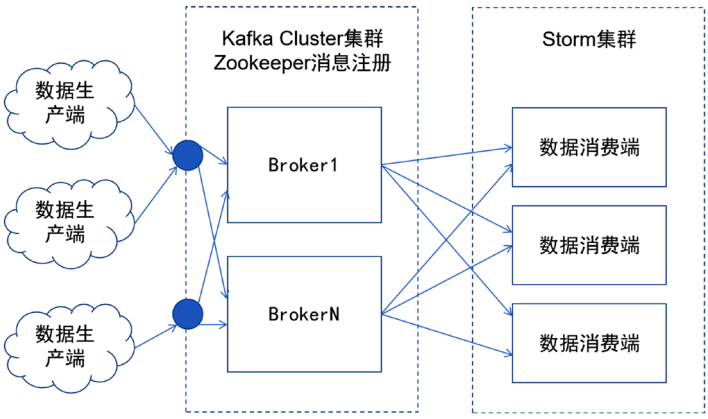

Figure 2. Cache forwarding architecture
图 2. 缓存转发架构图

数据采集模块作为数据生产端，是物联智能终端海量数据接入的第一道关，一方面需要实现高并发下数据接入，另一方面建立数据分类规则，实现对不同类型终端的管理。例如，在实际应用场景下，会存在多种智能终端型号，接入同一个云端服务，不同智能终端所产生的报文信息不同，为了方面管理不同类型的数据，利用 Kafka 的 Topic 分类管理，一类终端数据使用一个 Topic，数据解析模块(Consumer)会根据 Topic 拉取一类终端的报文信息，无需对数据类型进行判断，直接使用相应终端的解析规则对数据进行处理。

2.2. 实时分析模块

数据实时分析模块包括数据分析和数据存储，其中数据实时分析是关键，需要解决数据解析和存储过程中存在的阻塞、延迟以及丢包等问题，以确保数据维度的完整性，为此，本文采用分布式，高容错的实时计算系统 Storm 实时分析[8] [9]，图 3 所示为基于 Storm 的实时分析拓扑图。

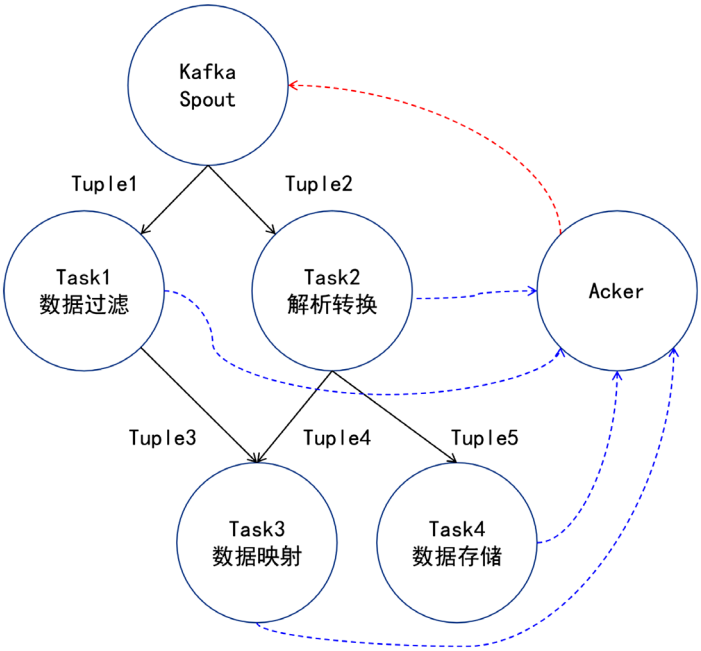

Figure 3. Real time topology analysis based on Storm
图 3. 基于 Storm 的实时分析拓扑

数据存储是实现对实时解析数据的存储服务，特别随着车联网等行业的持续发展，智能终端采集的数据规模爆炸式增长，且终端类型不同，采集的数据格式多样化，原先的存储系统在数据存储与查询管理方面存在一定瓶颈，为此，本文设计的数据存储采用两种方式，基于 HBase 的磁盘存储，实现对离线数据的管理，基于 Redis 的内存存储，实现对数据分析结果的管理，两种方式互补，既保证数据存储安全，也实现数据存储的灵活性。

2.3. 可视化展示

可视化展示主要是实现数据业务化，通过数据分析，结合不同应用场景，将结果以可视化的形式展现。可视化展示主要包括以下几个方面：

1) 终端管理：通过可视化界面，实现对终端的管理，提供故障预警、状态分析以及终端参数配置等管理服务。

2) 业务呈现：对解析后的数据，根据场景需求，完成不同业务功能的开发，实现数据业务化，同时提供业务流程的工具化设计。

3) 信息维护：实现对面向应用层的基础信息设置，例如终端与客户单位的映射维护、组织机构信息的维护以及业务功能配置等服务。

3. 优化设计

考虑到实际应用场景的复杂性，以及系统部署运营所存在的资源有限等情况，本文对上述系统中关键环节进行了一些优化设计。

3.1. 高并发下数据采集优化

随着系统并发访问量的快速增长，服务端资源消耗日益增加，空间上分布于各地的智能终端接入到平台面临着众多问题和挑战，针对上述问题，采用以下优化方案实现高并发的数据采集：

1) 重构服务端接口，利用 Boost 前摄器模式真正使得异步非阻塞通信得以实现并应用，通过异步调用系统内核 I/O 对象，可实现海量终端将所采集的信息进行实时接收与转发，解决了终端集中接入过程中出现的数据丢失、数据重复、异常数据和错误数据等问题。

2) 选择 TCP 协议作为本模块的数据传输协议。TCP 协议的三次握手模式，使得通信双方能够及时确认对方的状态，保证数据的可靠传输与安全到达。

3) 终端将数据发送到数据接收器，对于一些实时性要求较高的任务，终端一般间隔五秒就会发送一次数据。车载终端与服务器之间在发送数据前频繁地创建连接和数据传送结束后频繁地销毁连接会导致大量系统资源的浪费。所以，在车载终端和接收服务器之间利用 TCP 长连接方式进行通信，以降低延迟，提高效率。

前端节点只负责接收数据报文，不进行业务处理。前端节点采用多线程响应机制，能够同时接收来自不同终端的大量报文，利用 Kafka 消息队列完成报文存储。接收前端流程图如图 4 所示。

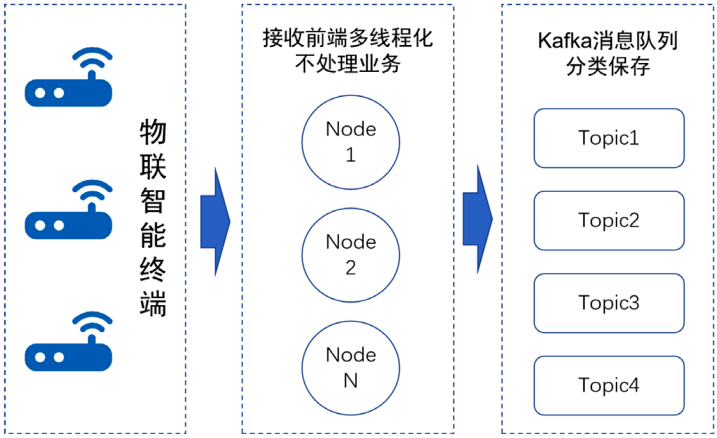

Figure 4. Receiving fore-end optimization flow chart
图 4. 接收前端优化流程图

3.2. 数据解析优化

面对智能终端发送过来的海量数据，数据解析模块既要采用分布式的设计思路，又要引入先进的数据处理技术和科学的优化系统结构，当系统解析任务量剧增时，才能避免系统服务器出现异常，保证数据处理的准确、快速、高效。因此，数据解析模块需要设计地灵活且能够满足不同类型的数据处理要求。

本文基于高并发、低延迟的数据处理要求，使用分布式相关技术，将数据传输、数据解析、缓存等进行分层管理与优化。下面分别从 Kafka 消息队列，Java 线程池，数据缓冲等关键设计要点进行阐述与分析。

1) 消息队列：数据解析模块需要进行数据过滤和报文解析等复杂操作，耗时严重，因此将不需要同步返回解析结果的任务作为消息，并将消息存入 Kafka 消息队列，利用其异步处理的特点，减少了响应时间。因此本文将解析的结果发送到 Kafka 消息中间件，不仅实现了解析模块与下游的存储模块之间的异步通信，也很好地将两者进行有效的解耦。

2) Java 线程池：如果不使用线程池，数据解析模块每次从 Kafka 中拉取数据，就要创建开启一个线程，然后对一条数据进行解析，解析完成之后，再对线程进行销毁。面对海量的车载终端数据，所创建的线程数无法控制，比如短时间内创建上百甚至上千个线程，电脑可能会瞬间崩溃。线程是是系统中相当重要的资源，它的创建会导致系统资源耗费严重，同时会影响系统处理任务的效率和连续性。

因此，本文使用 Java 的 Executor 框架，优势在于把任务的提交和执行进行解耦，只需要定义好任务，然后提交给线程池，而不用关心该任务的处理过程、被哪个线程执行等问题。在线程池中使用 BlockingQueue 阻塞队列，这种方式在面对插入数据队列已满或拉取数据队列已空的情况时，线程会阻塞直到队列非满或非空。BlockingQueue 队列的先进先出既保障了解析数据的有序性和一致性，又能够很好地实现数据解析模块快速高效稳定有序地执行。

3) Memcached 缓存：利用 Memcached 以实现 10 W/秒的数据查询操作，在真正意义上解决高并发的数据访问难题。把解析得到的结果缓冲到内存中，这样既使得数据库读写次数得到降低，更提高了访问效率，利用 Memcached 高性能、高可靠和高效的数据缓冲能力为服务层提供快速简洁的数据查询。

4. 实验分析

实验主要包括两个层面设计，一个是业务可视化设计，二个是系统性能验证。

4.1. 业务可视化实验设计

基于场景设计相应的业务功能，完成数据获取、指标建立到可视化全流程。在 WEB 端展示终端状态，具体包括终端业务操作次数统计，功能点击统计，以及业务与时间关系展示等数据指标呈现。如图 5 所示。

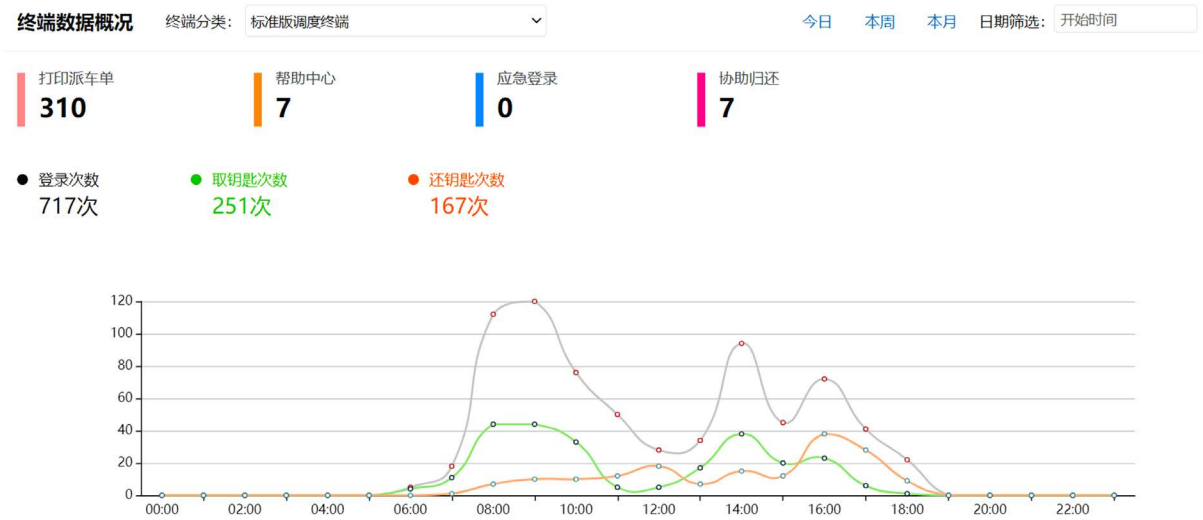

Figure 5. Visualization of function experiment
图 5. 业务可视化展示图

图 5 展示的我们自行开发的一款 AI 调度终端，该终端直接连接云端，分布在全国 21 个省份，主要用来实现对业务流程的自助服务和管理，服务的人群有百万数量，产生的数据包括图片、位置、视频、命令以及第三方平台关联数据等，这些数据=每秒都会发生，特别是 8:00~10:00 am 之间、14:00 pm 等几个时间点，终端的使用达到高峰，我们搭建的 WEB 页面，通过本文搭建的数据采集、存储以及实时处理等系统，会建立指标展示，用来监测每个终端的使用情况以及运营状态，从而方便运维人员，可以高效的管理。

4.2. 系统性能实验验证

性能测试，主要来验证本文提出的采集性能以及实时处理性能两个方面，首先需要搭建一个服务集群，通过部署局域网的 10 台 PC 机，搭建集群进行测试。实验环境配置如表 1 所示：

Table 1. Laboratory environment configuration table
表 1. 实验室环境配置表

序号	项目	版本号	功能说明
1	Storm	V0.10.2	大数据实时框架
2	Centos	V6.7	服务端操作系统
3	JDK	v1.8.0_45-b14	开发包
4	Kafka	v2.11-0.10.0.1	消息队列
5	Zookerper	v3.4.9	服务注册
6	Hbase	v1.0.3	数据存储
7	Redis	v3.2	数据存储
8	Memcached	v1.5.2	数据换成
9	计算机节点配置	内存大小：8 G，CPU 型号：intel Core i5，磁盘 500 G	

4.2.1. 前端采集性能测试

实验利用 Jmeter 进行模拟发送 TCP 消息，对接收程序进行性能测试。服务端机器部署程序，监控前端接收程序的 CPU 和内存信息；针对两个数据采集系统各设置 5 个 TCP 采样器，分别代表 5 个 TCP 客户端；Jmeter 进行模拟发送 TCP 消息，统计 Average (系统的响应时间)和 TPS (Transaction Per Second)。

Table 2. Comparison of experimental results
表 2. 实验结果对比

类型	TCP 采样器	Samples	Average	Error	TPS	KB/Sec
本文系统	1	363,430,205	59	0%	7920	61.9
	2	363,434,739	59	0%	7920	61.9
	3	363,428,438	59	0%	7920	61.9
	4	363,417,282	59	0%	7920	61.9
	5	363,413,564	59	0%	7920	61.9
	合计	1,817,124,227	293	0%	39,602	309.5
传统系统	1	229,662,341	66	1%	5053	53.0
	2	229,649,652	66	1%	5053	53.0
	3	229,645,992	66	1%	5053	53.0
	4	229,646,015	66	1%	5053	53.0
	5	229,652,126	66	1%	5053	53.0
	合计	1,148,256,125	331	1%	25,265	265

在搭建的集群环境下，测试的结果如表 2 所示，表中的指标说明如下：Samples 代表总共接收的报文数，Average 代表系统的平均响应时间(ms)，Error 代表错误率，TPS 代表每秒接收的报文数，KB/Sec 代表带宽吞吐。

实验验证数据表明，传统系统在接收 229,662,341 帧数据时，平台响应时间为 66 ms，发生错误率(也就是丢包或发生数据校验错误)为 1%，TPS 为 5053 以及带宽吞吐为 53；改进后系统，在接收 363,428,438 帧数据时，平台响应时间为 59 ms，TPS 为 7920；从数据来看，改进后的系统，与传统系统相对比，在 Samples、Average、Error、TPS 等数据指标上均有明显的改善和提升，具有在接收数据帧更多的时候，发生的错误率更低，数据接收处理的时效性越高的优点。

4.2.2. 数据解析性能测试

在搭建的集群环境下，我们继续测试数据实时处理的性能，具体通过测试数据解析性能来验证。通过模拟智能终端发送报文信息，模拟多条报文信息发送到解析程序中，测试解析模块的性能，效果如图 6 所示。

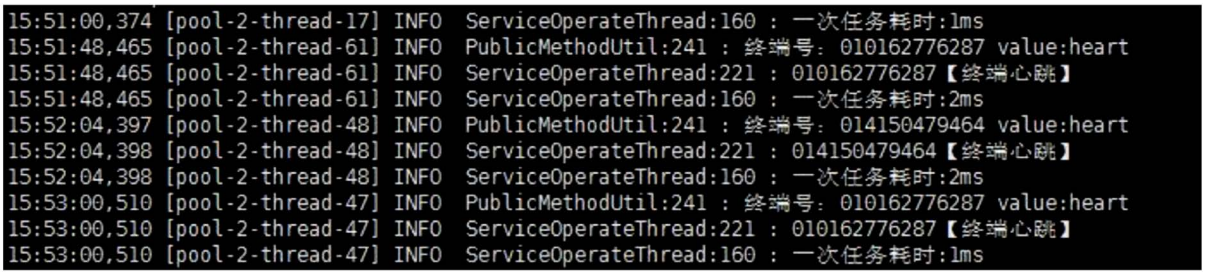

Figure 6. Analysis of time consuming graph
图 6. 解析耗时图

图 6 中，Service Operate Thread 是处理报文的线程类，一个线程解析一条报文，从图中可以看出，Service Operate Thread 一次任务耗时基本上在 1~2 ms 左右，也就是说每秒钟处理 500~1000 条报文帧数据，这个并发数已经完全可以支撑百万级终端数据的接入处理了，可见线程池的应用提高了解析速度，使得数据解析的实时性和效率更加可观。

我们进一步对两种方式下数据解析的时间数据做了对比。如表 3 所示。

Table 3. Comparison of data resolution experimental results
表 3. 数据解析实验结果对比

实验次数	报文帧数据	本文解析时间	传统解析时间
1	36 B	1.2 ms	3.5 ms
2	40 B	1.3 ms	3.8 ms
3	30 B	1.1 ms	3.7 ms
4	52 B	1.4 ms	4.3 ms
5	82 B	1.6 ms	5.1 ms
6	35 B	1.1 ms	3.7 ms

从表 3 可以看出，本文提出的数据解析时间，在处理相同报文帧数据时，在解析效率上，要明显高于传统方法，也就是表明，在相同大量数据接入时，本文在处理时间上，要更加快速，以满足整个应用对实时性的要求。

5. 结论与展望

针对智能终端海量数据采集与实时分析所存在的问题，需要从系统架构，以及性能优化等方面，进行重构，本文设计了一套海量数据采集与实时分析系统，通过借助 Kafka 消息队列，现数据的高并发接入；借助大数据流处理系统 Storm，在保证高可靠性的前提下，实现数据的实时处理，实现了高并发数据的缓冲、转发与处理解析，具有高并发、高稳定性和可维护性；通过测试验证，系统具有低延迟，高吞吐，可拓展等特点，能够满足车联网海量数据处理要求。

下一步将深度围绕资源有限情况，系统性能最大化，以及在大数据分析方面，结合现有技术，如 Hadoop 等[10] [11]，进行更深层数据挖掘，最终为应用服务提供更多有价值的信息。

参考文献

[1] 林平荣, 陈泽荣, 施晓权. 高并发多线程竞争共享资源架构[J]. 计算机工程与设计, 2020, 41(11): 3282-3288.

[2] 袁喆, 文继荣, 魏哲巍, 刘家俊, 姚斌, 郑凯. 大数据实时交互式分析[J]. 软件学报, 2020, 31(1): 162-182.

[3] 戴琳琳. 大数据背景下车联网的应用分析[J]. 电子技术与软件工程, 2020, 187(17): 169-170.

[4] 刘静, 吴仲城, 李芳, 张春风, 陈杰. 基于 Boost.Asio 的智能车载终端数据采集系统[J]. 计算机应用与软件, 2018(2): 248-255.

[5] 狄程, 杨中国, 韩燕波, 等. 面向流数据的实时处理及服务化系统[J]. 重庆大学学报, 2020, 43(7): 75-83.

[6] 王绪亮, 聂铁铮, 唐欣然, 黄菊, 李迪, 闫铭森, 刘畅. 流式数据处理的动态自适应缓存策略研究[J]. 计算机科学, 2020, 47(11): 130-135.

[7] McCreadie, R., Macdonald, C., Ounis, I., *et al*. (2013) Scalable Distributed Event Detection for Twitter. 2013 *IEEE International Conference on Big Data*, Silicon Valley, CA, USA, 6-9 October 2013, 543-549. https://doi.org/10.1109/BigData.2013.6691620

[8] Nair, L.R. and Shetty, S.D. (2018) Applying Spark Based Machine Learning Model on Streaming Big Data for Health Status Prediction. *Computers & Electrical Engineering*, **65**, 393-399. https://doi.org/10.1016/j.compeleceng.2017.03.009

[9] 杨立鹏, 张仰森, 张雯, 等. 基于 Storm 实时流式计算框架的网络日志分析方法[J]. 计算机科学, 2019, 46(9): 183-190.

[10] 董楠楠, 单晓欢, 牟有静. 基于 Hadoop 和 MapReduce 的大数据处理系统设计与实现[J]. 信息通信, 2020(6): 29-31.

[11] 孔德丽, 屈会雪, 卞志勇. 浅析基于 Hadoop 的高校大数据云平台设计[J]. 机械制造与自动化, 2020, 49(1): 101-102. https://doi.org/10.19344/j.cnki.issn1671-5276.2020.01.028