·实用技术·

基于 Spark Streaming 的实时数据采集分析系统设计*

党寿江[1,2] 刘 学[1] 王星凯[1,2] 刘春梅[1]

([1]中国科学院声学研究所 国家网络新媒体工程技术研究中心 北京 100190
[2]中国科学院大学 北京 100049)

摘要:大数据量的实时数据分析系统,需要快速的处理和响应。为了实现实时数据分析,本文设计了基于 Spark Streaming 的实时数据采集分析系统,并对有状态计算操作的基数计算的精确计算方法和估算方法进行了比较分析。实验表明,基于 HyperLogLog++ 的基数估算方法在处理时间和存储占用空间上有明显优势,而计算偏差基本可以忽略不计,更适于大数据的基数估算。

关键词:基数计算,实时数据分析,Spark 流式处理,不重复计数,HyperLogLog++

Design of Real – time Data Collection and Analysis System Based on Spark Streaming

DANG Shoujiang[1,2], LIU Xue[1], WANG Xingkai[1,2], LIU Chunmei[1]

([1]National Network New Media Engineering Research Center, Institute of Acoustics, Chinese Academy of Scicence, Beijing, 100190, China,
[2]University of Chinese Academy of Sciences, Beijing, 100049, China)

Abstract: Real – time data analysis systems with large amounts of data require fast processing and response. In order to realize real – time data analysis, this article designs a real – time data collection and analysis system based on Spark Streaming and analyzes the performance between exactly and approximately counting in cardinality counting for the stateful operation. Experiments show that the cardinality estimation method based on HyperLogLog++ has obvious advantages in processing time and storage space, and the impact caused by relative deviation is negligible, which is more suitable for the cardinality counting in real – time large data analysis.

Keywords: Cardinality Counting, Real – time Data Analysis, Spark Streaming, Distinct Count, HyperLogLog++

1 引言

互联网时代对数据分析的实效性和分析粒度提出了更高的要求。目前,社交应用、电子商务和新媒体应用,都有海量的数据产生。为了实时的挖掘和分析数据流中的信息价值,需要设计面向大数据分析的实时数据处理系统。

面向大数据处理的框架和技术,从 Hadoop 平台的并行分布式处理框架 Map Reduce 开始,到 Storm 流式处理、Spark 生态系统,一直在不断的发展。这些框架和技术提供了面向分布式的大数据处理方法和编程接口,极大的方便了海量数据分析程序的开发。

本文于 2017 – 08 – 07 收到。
* 中国科学院战略性先导科技专项:新型传播技术研究与系统研制(XDA06040602)。

数据分析，经常需要基数计算，即计算数据中不同元素的个数[1]。例如：统计一个网站 5 分钟内的独立 IP 地址数。针对此类问题，可以通过精确的基数计算算法进行计算，但需要将 5 分钟内的所有的 IP 地址都要缓存下来，需要消耗过多的资源。而基数估算算法是基于概率统计理论估算给定数据集中不同元素基数的算法，通过牺牲一定的数据精度，来提升计算效率[2]。

本文内容以如下组织：第 2 节设计了基于 Spark Streaming 的实时数据采集分析系统；第 3 节针对有状态计算操作的基数计算的精确计算方法和估算方法进行了比较分析，验证了基于 HyperLogLog++ 的基数估算方法更适于大数据的基数估计的实时统计；第 4 节对本文进行总结和展望。

2　基于 Spark Streaming 的实时数据采集分析系统

实时数据采集分析系统，既需要满足并发性要求，同时还需要数据处理的实时性和一定的数据容灾性保障。本文基于开源系统，设计了包括数据采集服务、数据队列服务、以及数据分析服务在内的实时数据采集分析系统[3,4]，总体架构如图 1 所示 ，其包括了数据采集客户端、数据采集服务器 OpenResty、Kafka（分布式发布/阅读消息系统）集群和基于 Spark Streaming 的数据分析计算程序。

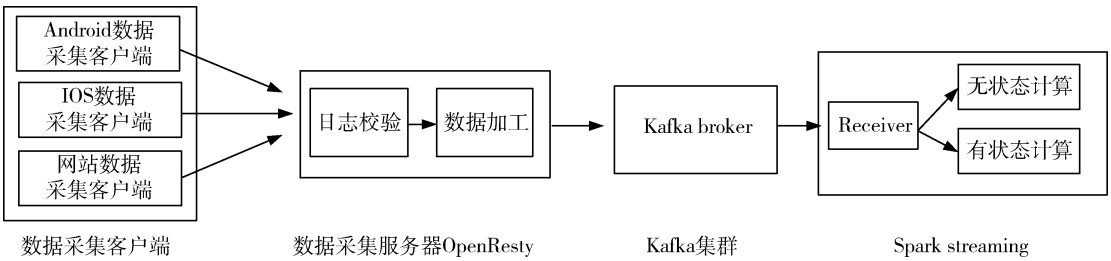

图 1　基于 Spark Streaming 的实时数据采集分析系统架构

2.1　数据采集客户端

为了方便第三方应用进行数据集成和数据统一处理，我们设计和开发了数据采集客户端（Open Resty），并在采集客户端中对采集数据的格式进行了统一定义，按照事件发生的主体、事件类别、事件涉及的属性、事件发生的时间、事件发生的位置以及事件的结果来定义日志信息，主要包括几个域：应用标识域、设备信息域、用户标识域、动作事件域、动作对象域、动作时间域、动作地理域和动作结果域，如表 1 所示。

表 1　日志记录信息域定义

日志信息域	描述
应用标识域	定义了应用的标识、版本及类型等
设备信息域	定义了设备的操作系统、分辨率、设备 model、分辨率及 useragent 等信息
用户标识域	定义了设备标识和用户标识
动作事件域	定义了常见的浏览事件、异常事件、播放相关事件、搜索、评论、分享等事件，同时支持自定义事件
动作对象域	定义了对应事件域所涉及的业务属性，比如播放的节目标识、url 地址、流地址、播放时间点等
动作时间域	定义了事件的开始和结束时间
动作地理域	定义了事件发生的经纬度、网络类型、ip 地址等
动作结果域	定义了事件发生的结果

2.2　基于 OpenResty 的实时数据采集服务

Nginx 是一个高性能的 HTTP 服务器，相较于 Apache、Lighthttpd 等具有占用内存少、稳定性高等优势。OpenResty 是一个基于 Nginx 服务器与 Lua 解释器的高性能 Web 平台，充分利用 Nginx 的非阻塞 I/O 模型，使用 Lua 脚本语言调用 Nginx 支持的各种 C 及 Lua 模块，不仅仅能对 HTTP 客户端连接请求，而且对远程后端如 Mysql、Redis 及 Kafka 等都能进行一致的高性能响应。

基于 OpenResty 的实时数据采集服务，通过 Lua 模块基于定义的日志记录格式，实现对上报日志的格式

校验,避免非正常日志进入和污染系统,同时针对日志中的 UserAgent 信息和 IP 地址信息进行补充和加工,对处理后的日志通过 Kafka Lua 生产者客户端,向 Kafka 的不同分区进行发布日志数据,完成数据的采集。

2.3 基于 Kafka 的实时数据缓存队列

Kafka 是一个易于扩展的基于主题发布/订阅的消息队列系统。Kafka 主要包括生产者 Producer、代理 Broker 和消费者 Consumer 组成[5]。Producer 负责将消息收集并发送给 Broker,Broker 接收 Producer 的消息,并将消息进行持久化,Consumer 是消息的使用者,从 Broker 获取消息。Kafka 作为数据汇聚的缓冲,是整个数据架构的核心,可以为多个消费者提供数据,使实时数据服务于多个场景和服务。

Kafka 集群对实时数据流进行解耦,实现了数据的生产者和消费者的异步处理。数据采集服务器 OpenResty 作为实时数据的生产者 Producer,基于事件类型,向 Kafka 集群的主题进行发布。Spark Streaming 作为消费者 Consumer 从 Kafka 集群中读取数据,形成了实时数据处理管道。

2.4 基于 Spark Streaming 的实时数据分析

Spark Streaming 是 Spark 大数据分析系统中用于实时计算的一个框架。Spark Streaming 将实时数据流以时间片为单位进行拆分形成离散数据流(DStream, Discretized Stream)。离散数据流 (DStream) 作为 Spark Streaming 中的一个基本抽象,其内部维护了一组离散的以时间轴为键的弹性分布式数据集(RDD, Resilient Distributed Datasets)序列[6],这些 RDD 序列分别代表着不同时间段内的数据集,而对于 DStream 的各种操作最终都会映射到内部的 RDD 上,从而实现了和 Spark 的无缝衔接。

基于 Spark Streaming 的数据分析,通常包括两类计算:无状态类计算和有状态类计算,例如:计算网站每 5 分钟的页面浏览量 PV(Page View)数量和独立 IP 数量,如果 Spark Streaming 的批次处理间隔为 5 分钟,那么每天的 PV 数量,可以直接对每批次的 PV 数量进行相加并计算出每天的 PV 数量。而独立 IP 数量,则需要把 5 分钟内所有的 IP 数量都进行状态保存,才可以进行统一计算。针对有状态的计算则需要更多的资源才能完成,而且针对需要每天、每周等进行多个粒度的统计,更是需要很长时间才能响应。因而对数据计算进行分类,并对不同的分类采用不同的计算方法才能更有效和实时的进行分析。

3 针对有状态操作的基数计算的精确计算方法和估算方法比较分析

基数计算方法是决定一个数据流中不同元素个数的方法。基数计算有两类方法:精确计算算法和近似估算算法。精确计算算法通常能够被线性空间复杂度 $O(N)$ 算法轻易计算[1],但是往往需要大量的内存,而且响应时间较长,面对海量数据时往往力不从心。因而针对海量数据的使用较少资源的基数估算方法应运而生。常见的基数估计算法有 Linear Counting、LogLog Counting、HyperLogLog Counting[2] 以及 Adaptive Counting 等[1]。HyperLogLog++[7]是在 HyperLogLog Counting 算法基础上提出的一个算法,具有较低的误差。本文主要基于 HyperLogLog++ 算法进行每天独立 IP 数量的基数估计。

3.1 基数精确计算方法

基于 Spark Streaming 的精确基数计算方法,在每一批次处理时,依据历史状态算出每一批次中每天第一次出现的 IP 数量,然后把每一批次的 IP 数量相加,即为每天的独立 IP 数量,具体流程如图 2 所示,伪代码如下:

```
JavaDStream < PcIpByDay > pcIpByDay = pcLogs
. mapPartitionsToPair ( new PcIpByDayStatistics1 ( ) )//映射数据为( dag,ip > ,1) 的键值对
. updateStateByKey ( new PcIpByDayMergeFuntion ( ) )//基于历史状态判断对应用 Key 是否计算过.
. mapPartitionsToPair ( new PcIpByDayStatistics2 ( ) )//映射数据为( < dag > ,1/0) 的键值对.
. updateStateByKey ( new PcIpByDayADDFuntion ( ) )//基于历史值累加当前增量值
. mapPartitions ( new PcIpByDayFromPairFuntion1 ( ) ) ;
pcIpByDay. foreachRDD ( new SavePcIpByDay1 ( ) ) ;
```

3.2 基于 HyperLogLog++ 的基数估算方法

基于 Spark Streaming 的基数估算方法,针对每天保存一个 HyperLogLog++ 对象到历史状态中,并将每一

批次的 IP 添加到对应的 HyperLogLog++ 对象中。计算时，调用 HyperLogLog++ 对象的基数方法即得到每天的独立 IP 数量，具体流程如图 3 所示，伪代码如下：

```
JavaDStream < PcIpByDay > pcIpByDay = pcLogs
. mapPartitionsToPair( new PcIpByDayStatisticsHll( ) )//映射数据为(day,ip)的键值对.
. updateStateByKey ( new PcIpByDayHllUpdateFuntion ( ) )//基于历史更新 HyperLogLog++ 对象
. mapPartitions( new PcIpByDayFromHll( ) );
pcIpByDay. foreachRDD( new SavePcIpByDay1( ) );
```

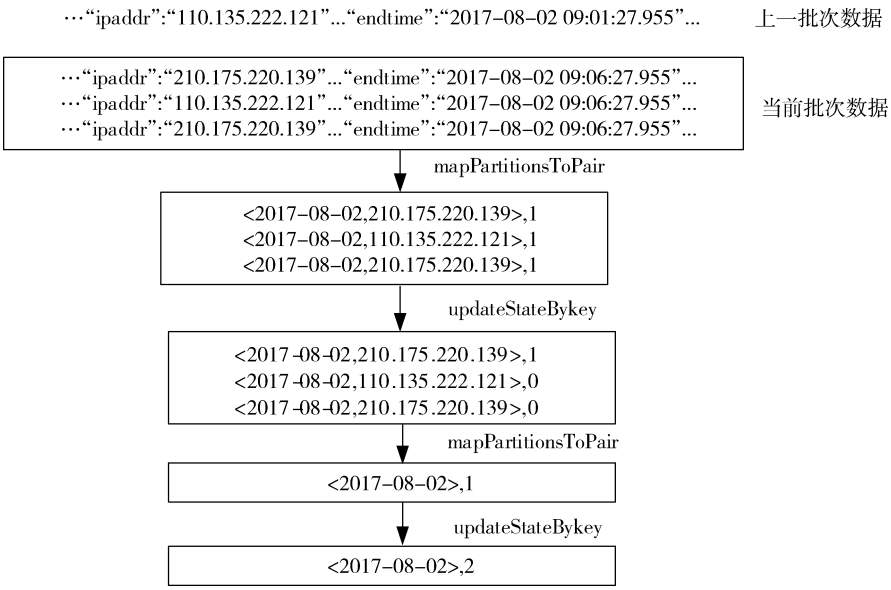

图 2　基于 Spark Streaming 统计每天独立 IP 数量的精确计算方法流程

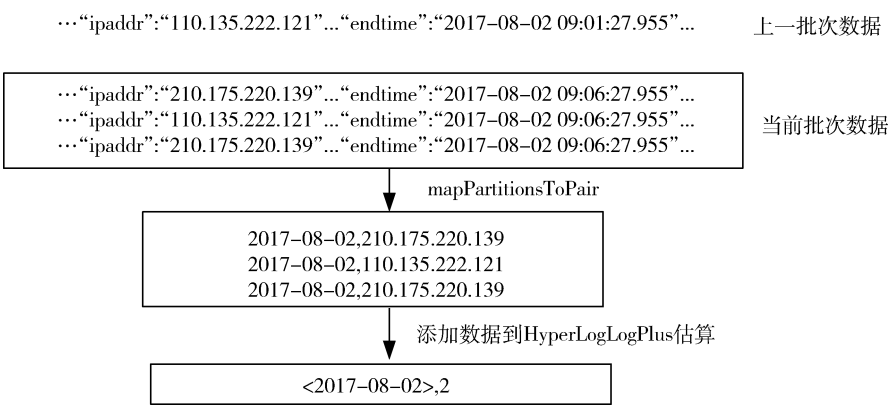

图 3　统计每天独立 IP 数量的 HyperLogLog++ 估算方法

3.3　实验比较分析

按照每批次产生 3000 条、5000 条、10000 条日志记录分三次进行实验，每次实验运行 12 个批次，共计 36 个批次，其中每批次中有 1/2 的 IP 数量一样。实验中，Spark Streaming 的批处理间隔为 5 分钟，即每 5 分钟产生 3000 条、5000 条或 10000 条日志记录，每条日志记录大小为 1296byte，其中日志中的 IP 地址字段为 15 个字节(XXX. XXX. XXX. XXX)。

对日志分别运行基数精确计算方法和基于 HyperLogLog++ 的基数估算方法，在运行中，程序以 local 模式进行部署提交，统计运行过程中计算出的独立 IP 数量、每批次的处理时间及 checkpoint 路径下文件的大小。

估算计数方法中各批次统计的独立 IP 数量的误差百分比计算结果如表 2 所示。从表 2 可知估算计数

方法和精确计数方法相比,误差率在 1.5% 以下,针对统计每天的独立 IP 数量而言,误差基本可以忽略不计。而且从批次处理的日志记录数量来看,日志数量越多,误差相对越小。

表 2 计数误差比较

批次	估算计数	精确计数	误差百分比	批次	估算计数	精确计数	误差百分比	批次	估算计数	精确计数	误差百分比
1	1500	1500	0	13	2500	2500	0	25	5000	5000	0
2	3000	3000	0	14	5000	5000	0	26	10001	10000	0.01
3	4500	4500	0	15	7501	7500	0.01	27	15019	15000	0.13
4	6001	6000	0.02	16	9999	10000	0.01	28	20158	20000	0.79
5	7501	7500	0.01	17	12437	12500	0.5	29	25054	25000	0.22
6	9001	9000	0.01	18	14987	15000	0.09	30	30232	30000	0.77
7	10502	10500	0.02	19	17536	17500	0.21	31	35171	35000	0.49
8	12002	12000	0.02	20	19973	20000	0.14	32	40197	40000	0.49
9	13652	13500	1.11	21	22416	22500	0.37	33	45222	45000	0.49
10	15142	15000	0.94	22	24862	25000	0.55	34	50427	50000	0.85
11	16651	16500	0.91	23	27556	27500	0.2	35	55419	55000	0.76
12	18224	18000	1.23	24	30082	30000	0.27	36	60537	60000	0.9

每一批次处理时间的统计结果如图 4 所示,从图 4 可以看出相同数量的日志数量,精确计数方法每批次的平均处理时间比基于 HyperLogLog++ 的基数估算方法要多出 1 倍。而且,随着日志记录数的增加,精确计数方法每批次的平均处理时间比基于 HyperLogLog++ 的基数估算方法增加的更多。由此可以看出,在第 3 次实验的批次中,精确计数方法的批处理时间已经大于 Spark Streaming 的批处理间隔,产生了较大的调度延迟。

图 5 显示了 checkpoint 占用空间随批次的变化情况,从图 5 中可以看出,基于 HyperLogLog++ 的基数估算方法的存储空间使用量基本上趋于稳定,波动较小,而且占用空间的绝对量远远低于精确计数方法。而精确计算方法,由于要保存所有记录的状态,其 checkpoint 占用空间随着日志记录数量的变大而单调直线上升。

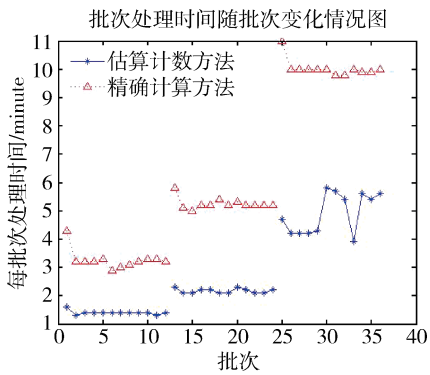

图 4 批次处理时间统计图

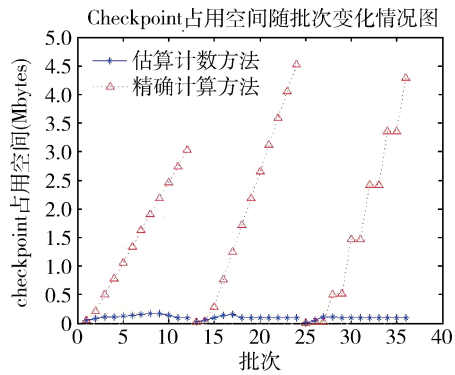

图 5 checkpoint 占用空间统计图

由上述分析可以看出,基于 HyperLogLog++ 的基数估算方法,相对于精确计算方法,在处理时间和 checkpoint 占用空间上有明显优势,而误差率基本在 1.5% 以下,误差的影响基本可以忽略不计。因而,基于 HyperLogLog++ 的基数估算方法更适于大数据的基数估计的实时统计。

4 结束语

本文设计了基于 Kafka、Spark Streaming 的实时数据采集分析系统,并针对有状态计算操作的基数计算的精确计算方法和估算方法进行了实验分析。实验结果表明,基于 HyperLogLog++ 的基数估算方法,相对于精确计算方法,在处理时间和 checkpoint 占用空间上有明显优势,而误差率基本在 1.5% 以下,误差的影响基

本可以忽略不计。因而，基于 HyperLogLog++ 的基数估算方法更适于大数据的基数估计的实时统计。后续将继续挖掘采集数据的业务特征，研究基于业务特征的基数估算和相关统计量的近似估算算法。

参 考 文 献

[1] 魏剑龙. 面向海量数据的分布式 OLAP 引擎的研究与实现[D]. 东北大学，2015.

[2] 刘绍记，曹阳，崔梦天. 基数估计算法参数的分析与优化[J]. 计算机科学，2017，44(02)：279 - 282 + 301

[3] 韩德志，陈旭光，雷雨馨，戴永涛，张肖. 基于 Spark Streaming 的实时数据分析系统及其应用[J]. 计算机应用，2017，37(05)：1263 - 1269

[4] 郑文俊，彭明喜. 大并发、高吞吐量实时数据平台的研究[J]. 电信快报，2016，(10)：28 - 34

[5] 孙大为，张广艳，郑纬民. 大数据流式计算：关键技术及系统实例[J]. 软件学报，2014，25(4)：839 - 862

[6] 薛瑞，朱晓民. 基于 Spark Streaming 的实时日志处理平台设计与实现[J]. 电信工程技术与标准化，2015，28(9)：55 - 58

[7] Stefan Heule, Marc Nunkesser and Alexander Hall. HyperLogLog in practice: algorithmic engineering of a state of the art cardinality estimation algorithm[C]//Proceedings of the 16th International Conference on Extending Database Technology (EDBT'13). New York, NY, USA: ACM, 2013.

作者简介

党寿江，男，(1982.10 -)，博士研究生，主要研究方向：新媒体技术。

刘学，男，(1979.1 -)，副研究员，主要研究方向：云计算、多媒体通信、大数据。

王星凯，男，(1992.8 -)，博士研究生，主要研究方向：新媒体技术。

刘春梅，女，(1985.1 -)，硕士，助理研究员，主要研究方向：智能终端。

(上接第 28 页)

[8] A. K. Nayak, A. Reimers, N. Feamster, and R. Clark. Resonance: dynamic access control for enterprise networks[M]. in Proceedings of the 1st ACM workshop on Research on enterprise networking. ACM, 2009. 11 - 18

[9] 李凤华，王巍，马建峰，梁晓艳. 基于行为的访问控制模型及其行为管理[J]. 电子学报，2008，(10)：1881 - 1890

[10] 林莉，怀进鹏，李先贤. 基于属性的访问控制策略合成代数[J]. 软件学报，2009，(2)：403 - 414

[11] 董理君，余胜生，杜敏，周敬利. 一种基于环境安全的角色访问控制模型研究[J]. 计算机科学，2009，(1)：51 - 54 + 59

[12] H. Zhang, Y. He, and Z. Shi. A formal model for access control with supporting spatial context[J]. Science in China Series F: Information Sciences, 2007, 50(3): 419 - 439

[13] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. Openflow: enabling innovation in campus networks[J]. ACM SIGCOMM Computer Communication Review, 2008, 38(2): 69 - 74

[14] T. L. Saaty and L. G. Vargas. Models, methods, concepts & applications of the analytic hierarchy process[J]. Springer US, 2012, 7(2): 159 - 172

[15] V. Jain, P. Congdon, J. R. Vollbrecht, A. Smith, R. Yavatkar, B. Rosen, and J. Roese. Port - based network access control[M]. IEEE Draft, 1999.

[16] MIT Lincoln Laboratory. Lincoln laboratory scenario (ddos) 1.0. [EB/OL]. (2000) Available: http://www.ll.mit.edu/ideval/data/2000/LLS DDOS 1.0.html.

[17] MIT Lincoln Laboratory. 2000 ddos 2.0.2 intrusion detection dataset hosts. [EB/OL]. (2000) Available: http://www.ll.mit.edu/ideval/docs/2000 LLS DDOS 2.0.2 hosts.html.

作者简介

谢德俊，男，(1991 -)，硕士研究生，主要研究方向：网络与系统安全。

王利明*，男，(1978 -)，副研究员，主要研究方向：网络安全、软件定义网络、云计算和数据安全。

宋晨，女，(1984 -)，助理研究员，主要研究方向：网络安全。

杨倩，女，(1989 -)，研究实习员，主要研究方向为：网络安全。

* 为通信作者。