文章编号:1674-2869(2019)02-0179-05

空气质量监测大数据区间的统计问题

刘黎志,何经纬
智能机器人湖北省重点实验室(武汉工程大学),湖北 武汉 430205

摘 要:为降低客户端和服务端之间的远程过程调用(RPC)通讯,提高对存储空气质量监测数据的HBase表的区间统计效率,提出了一种基于协处理器的大数据区间统计方法。使用终端协处理器可以将区间统计过程放在服务端运行,通过特定的协议将区间统计所需的参数从客户端传递到服务端,协处理器调用结束后,将结果返回到客户端,客户端对返回的消息进行处理汇总,最终得到区间统计结果。实验证明,使用终端协处理器进行空气质量监测数据区间统计较使用客户端扫描方式至少快一个数量级,极大地提高了统计效率。

关键词:协处理器;大数据;区间统计;HBase
中图分类号:TP311 **文献标识码**:A **doi**:10.3969/j.issn.1674-2869.2019.02.015

Big Data Interval Statistics for Air Quality Monitoring

LIU Lizhi, *HE Jingwei*
Hubei Key Laboratory of Intelligent Robot (Wuhan Institute of Technology), Wuhan 430205, China

Abstract: To reduce the remote procedure call communications between the client and the server, and improve the interval statistical efficiency of the HBase table that stores air quality monitoring data, we proposed a big data interval statistics method based on co-processor. Interval statistics process was put into an endpoint co-processor running on server side, then the required parameters for interval statistics were transmitted from the client to the server through a specific protocol. The response message which was processed and summarized was returned to client when calling co-processor was finished, thus the interval statistics result was finally obtained. The experiments prove that using endpoint co-processor to do interval statistics for air quality monitoring data is at least one order of magnitude faster than using client scan method, so the statistics efficiency is promoted dramatically.

Keywords: co-processor; big data; interval statistics; HBas

城市空气质量监测站的监测过程需要记录大量实时数据,以及根据实时数据计算出的小时均值数据、日均值数据和评价数据[1-3]。湖北省环境中心站所管辖的102个自动化站每天产生的海量数据,如果使用关系型数据库存储,数据检索的实时性和效率将无法保证。基于Hadoop的大数据解决方案的研究为有效存储和快速检索空气质量监测数据提供了新途径,其中HBase是建立在Hadoop之上,具有高可靠性、高性能、列存储、可伸缩、实时读写等特点的数据库系统,HBase通过指定行键(row key)的范围来查询数据,为海量的数据提供高效率的数据维护及检索功能[4-11]。

在对空气质量监测数据进行查询时,通常需要对某个监测值或评价值进行区间统计,如统计宜昌市全年NO_2的实时浓度值在0.00~41.00 μg/m^3,43.05~82.00 μg/m^3,84.05~123.00 μg/m^3,125.05~164.00 μg/m^3,166.05~205.00 μg/m^{3}区间的分布情况;统计宜昌市的伍家岗站2016年6月轻度污染

收稿日期:2018-11-29
作者简介:刘黎志,硕士,副教授。E-mail:llz73@163.com
基金项目:武汉工程大学第十届研究生教育创新基金
引文格式:刘黎志,何经纬.空气质量监测大数据区间的统计问题[J].武汉工程大学学报,2019,41(2):179-183.

以上的天数，即AQI指数分别在101~150，151~200，201~ 300，>300的天数。HBase提供的Scan方法，每执行一次next操作，只会从服务端读取一行数据，因此扫描多个Region会在客户端和服务端之间形成大量的远程过程调用（remote procedure call，RPC）通讯，从而影响查询效率。HBase0.92版本中提出的终端（Endpoint）协处理器可以在服务端完成计数、求和、求最大值等统计工作，并将结果返回到客户端，减少了客户端到服务端的RPC调用，从而极大地提高了统计查询的效率[12-14]。本文将对如何使用终端（Endpoint）协处理器对空气质量监测大数据进行区间统计进行讨论。

1 空气质量监测大数据存储模式设计

基于HBase的空气质量监测大数据的存储模式设计如图1所示。

空气质量存储模式的具体描述见文献[15]，实际的应用证明该模式可以有效地对空气质量监测数据进行存储及满足地区、站点之间的数据同比、数据环比、趋势分析等查询所需的要求。

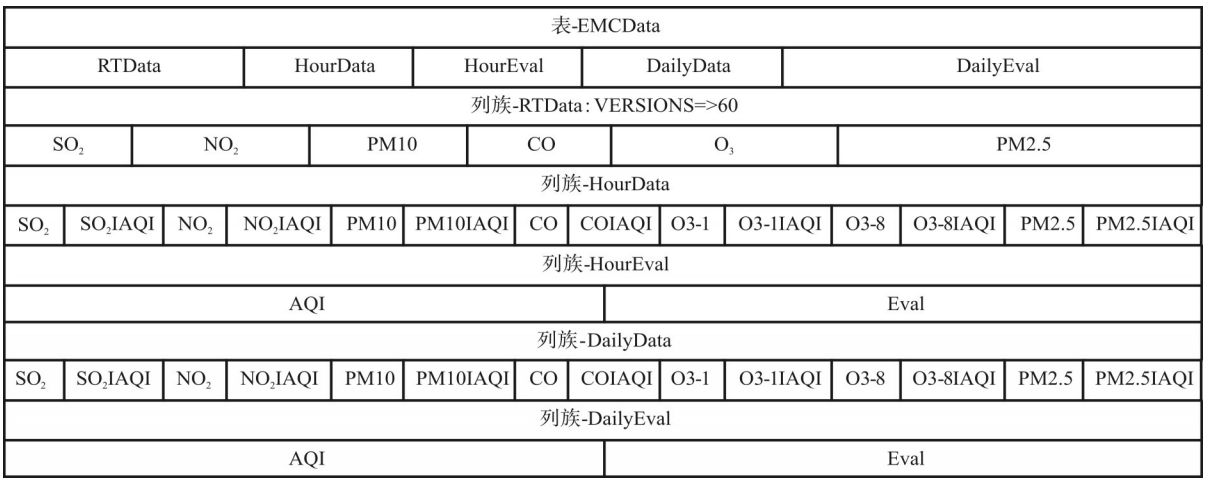

图1 空气质量监测数据存储模式

Fig. 1 Store schema of air quality monitoring data

2 区间统计协处理器

协处理器分为观察者（Observer）模式及终端（Endpoint）模式两种。终端协处理器可以将数据检索统计过程放在服务端完成，减少客户端到服务端的远程过程调用产生的通讯开销，从而提高统计效率，使用终端协处理器对数据进行区间统计的过程如图2所示。

数据区间统计的步骤为：1）定义EMCStat.proto文件，按照protobuf协议定义区间统计协处理器

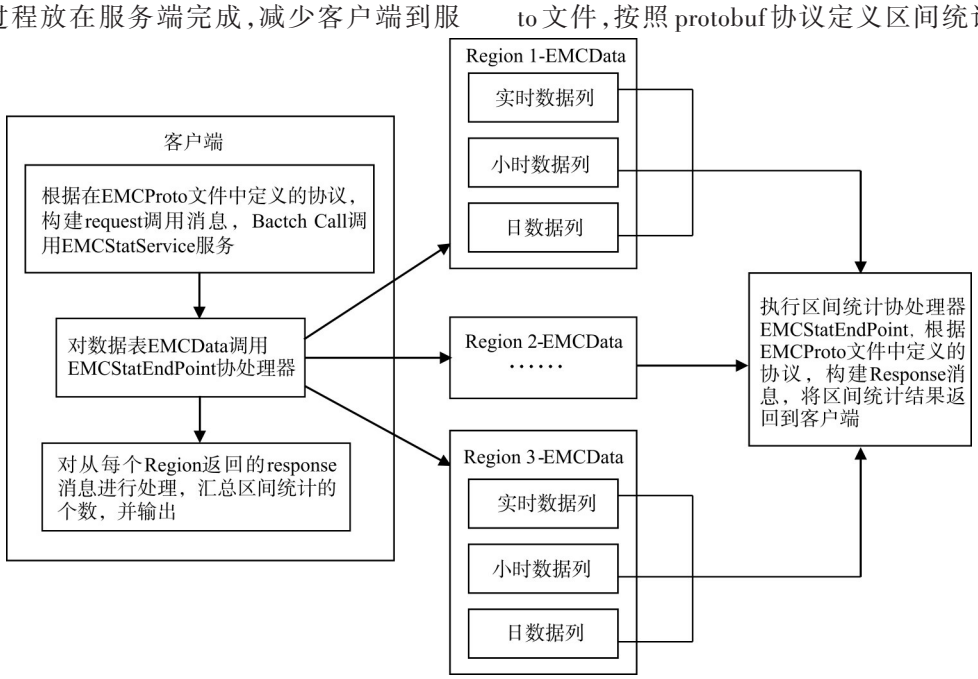

图2 协处理器调用过程

Fig. 2 Process procedure of co-processor

的request，response消息格式及RPC服务；2）定义协处理器类EMCStatEndPoint，实现EMCStat.proto文件中定义的RPC服务EMCStatService，服务中的getEMCStat方法实现区间统计的业务逻辑；3）为EMCData表加载EMCStatEndpoint协处理器；4）客户端调用EMCStatEndpoint协处理器，对分布在不同Region上的数据进行区间统计，并输出结果。

2.1　Protobuf协议

终端协处理器使用protobuf协议来定义客户端与服务端进行通信的消息格式，实现空气质量区间统计终端协处理器的protobuf协议的定义为：

```
message EMCStatRequest
{  //定义客户端请求协议格式
required string areacode =1;//地区码
   optional string ssid=2 ;//站点编码
   required string stattime =3;//统计开始时间
   required string endtime =4;//统计结束时间
   required string cf =5;//列簇名
   required string qual=6;//列限定符
   message LHLimit//区间嵌套消息
   {
required float  ll=1;//区间下限
     required float  hl=2 ; //区间上限
   }
repeated LHLimit lh = 7;
//区间消息可重复，表示可以定义多个区间
 }
message EMCStatResponse
{  //定义服务端返回协议格式
   required string areacode =1;//地区码
   optional string ssid=2 ;//站点编码
   required string cf =3;//列簇名
   required string qual=4 ;//列限定符
   message LHCount/区间统计结果嵌套消息
   {
    required float  ll=1 ;//区间下限
    required float  hl=2 ;  //区间上限
    required int64  count =3 ;//区间计数
}
repeated LHCount lhc = 5;
 }//可以输出多个区间统计结果
service EMCStatService
{  //协议服务名
  //rpc调用方法名
  rpc getEMCStat(EMCStatRequest)
  returns (EMCStatResponse);
}
```

客户端在调用服务端的终端协处理器时，会根据EMCStatRequest协议的格式，向协处理器传递参数，包括：区间统计的地区码、站点编码、统计时间段，需要统计的列簇名及列限定符名，统计区间集合列表。协议服务EMCStatService表示其RPC方法getEMCStat以EMCStatRequest消息为输入，在获取其定义的所需参数后，执行区间统计程序，服务端协处理器按照EMCStatResponse协议格式将区间统计的结果返回给客户端。所有协议被定义在EMCStat.proto文件中，使用protoc工具，执行protoc --java_out=./src EMCStat.proto命令，可以在项目中生成EMCStatProtos.java文件，该Java文件是区间统计协处理器数据交换协议的代码实现，文件中定义了EMCStatService抽象类以及抽象方法getEMCStat。

2.2　区间统计协处理器实现

定义EMCStatEndPoint类为区间统计协处理器的实现逻辑类，该类继承于EMCStatService抽象类，并实现了Coprocessor和CoprocessorService接口，EMCStatEndPoint类中的getEMCStat方法用于实现区间统计过程，主要过程如下：

算法 getEMCStat（RpcController rpcCt，EMCStatRequest emcsRequest，RpcCallback <EMCStatResponse> done）

{

输入：emcsRequest；

输出：done；

1：Scan sc = new Scan()；sc.setMaxVersions()；

2：读取emcsRequest消息的各个字段，包括地区码、站点编码、统计时间段、列簇名、列限定符赋值到对应的变量；

3：根据emcsRequest消息提供的统计区间对区间类集合列表进行初始化，将每个区间的计数设置为0；

4：if（站点编码为空）{将地区下的所有站点编码添加到lstSSIDS集合，表示统计所有站点}；

5：else{将站点编码添加到lstSSIDS集合}

6：EMCStatResponse response = null；InternalScanner itScanner = null；

7：for（String assid：lstSSIDS）{

8：sc.setStartRow（startKey）；//区间统计startKey为地区码_站点编码_统计开始时间

9：sc.setStopRow（endKey）；//区间统计endKey为地区码_站点编码_统计结束时间

//判断是否需要对该region进行统计

10：if（startKey > env.getRegion(). getEndKey() || endKey < env.getRegion(). getStartKey()）{break；}

11：sc.addColumn（Bytes.toBytes（列簇名），Bytes.toBytes（列限定符））；

12：itScanner = env.getRegion(). getScanner（sc）；

13：List<Cell> cellResults = new ArrayList<Cell>()；boolean isHasMore = false；

14：do {

15： isHasMore = itScanner.next(cellResults);

16： for (Cell cell : cellResults) {根据cell的值,确定其所在的区间后,将其集合列表中对应的记数加1; }

17： cellResults.clear();}}

区间统计协处理器在对每个Region进行统计时,可以根据Region的StartKey和EndKey来判断该Region是否参与统计,当进行区间统计的StartKey大于Region的EndKey或区间统计的EndKey小于Region的StartKey时,可直接跳过该Region。在如图3所示的5个区间统计中,Region-A参与区间统计2、3、4,不参与区间统计1、5。在区间统计过程中跳过不需要进行统计的Region,可以加快扫描速度,提高统计效率。

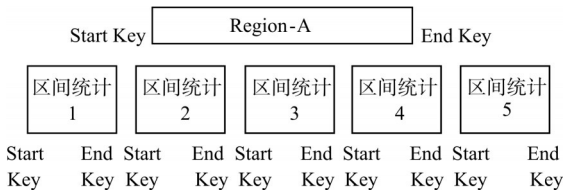

图3 Region统计逻辑

Fig. 3 Statistic logic of region

算法中的区间类的定义和EMCStatResponse中的消息LHCount的格式一致。EMCStatEndPoint类编译成功后,将其所在的jar包导出,并上传到Hadoop集群的HDFS分布式文件系统中,使用alter 'EMCData','coprocessor'=>'hdfs:///jar包的路径/jar包|EMCStatEndPoint协处理器的完整类名|表示优先级的整数值|参数'命令将协处理器加载到EMCData表。

2.3 客户端调用

客户端区间统计业务逻辑按EMCStatRequest的消息格式定义协处理器统计过程所需要的参数后,以Batch Call方式调用EMCData表的区间统计协处理器,由于Batch Call只负责对每个Region进行区间统计,所以还需要对每个Region的区间统计结果进行汇总后 输出,过程如下:

算法: main(String[] args){

输入: args[0]:地区码,args[1] ssid :站点编码,若为"-",表示查询所有子站点;args[2]startTime-endTime 以-分隔;args[3]列簇名;限定符;args[4]lh-hh:lh-hh表示统计区间

输出: 每个region的统计结果;

1: 根据args数组,将用户输入的参数地区码、站点编码、统计时间段、列簇名、列限定符赋值读取到对应的变量;

2: 将统计区间读取到LHLimit类型的集合列表中; 构造EMCStatRequest消息;

3: long beginTime=System.currentTimeMillis(); Configuration config=HBaseConfiguration.create();

4: HTable htb=new HTable(config,"EMCData");

5: Map<byte [] , String> resultMaps = htb.coprocessorService(EMCStatService.class, null, null,

6: new Batch.Call<EMCStatService,String>() {//调用协处理器

7: public String call(EMCStatService emcStat){

8: ServerRpcController srController = new ServerRpcController();

9: BlockingRpcCallback<EMCStatResponse> bRpcCb =

10: new BlockingRpcCallback<EMCStatResponse>();

11: emcStat.getEMCStat(controller, request,bRpcCb);

12: EMCStatProtos.EMCStatResponse emcsResponse = bRpcCb . get();//得到Response返回消息

13: if(emcsResponse ! = null) {List<LHCount> lstlhCount = emcsResponse . getLhcList();

14: for(LHCount lhc:lstlhCount){输出每个region的统计结果; 将每个region的区间统计结果进行累加;}}

15: return "";}});

通过记录区间统计的开始和结束时间,得到协处理器区间统计所需的时间,可以快速地与直接使用Scan操作进行区间统计所需的时间进行比较。

3 实验部分

实验环境的安装和配置和文献[16]中描述的一致。模拟的数据写入程序按每个监测项目,每小时40~60个实时值写入EMCData表,实时数据写入完成后,自动计算并写入小时均值及评价,全天的小时均值计算完成后,自动计算并写入全天的日均值及评价。

在数据的写入过程中,当Region的数量分别为3、5、7、9、11时,对存储NO_2实时浓度数据的列RTData按0.00~41.00 μg/m³, 43.05~82.00 μg/m³, 84.05~123.00 μg/m³, 125.05~164.00 μg/m³, 166.05~205.00 μg/m³进行区间统计,参数为:地区码4201,站点编码为空,表示统计该地区下的所有站点(9个子站+城区),统计时间段覆盖所有Region。为减少客户端Scan统计过程RPC调用,可以为Scan操作设置一个扫描缓存值,表示一次RPC调用可以从服务端读取的行数,从而减少客户端RPC请求次数,但扫描缓存值不能设置太高,否则会过多消耗客户端内存,严重时还会导致内存溢出,且延长next操作的时间,反而降低了查询效率。扫描缓存值的设置需要在减少RPC请求及客户端内存消耗之间取得平衡,实验中将扫描缓存值设置为256。客户端Scan的统计过程的具体实现算法类似于区间统计协处理器,这里不再具体描述。各

区间值统计的结果，使用协处理器进行区间统计及客户端Scan进行统计所需的时间如表1所示，时间对比如图4所示。

表1　实验结果

Tab. 1　Experimental results

Region数量/个	数据总量/个	协处理器统计时间/s	Scan统计时间/s
3	1 301 299	4.6	48.2
5	2 607 730	4.8	97.2
7	3 572 336	6.2	140.7
9	4 856 571	20.3	233.4
11	5 980 808	28.3	290.8

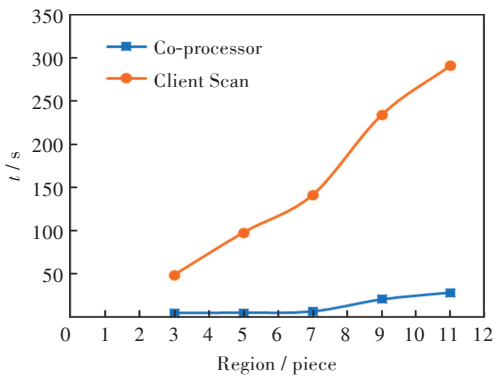

图4　协处理器和客户端Scan区间统计时间对比

Fig. 4　Time comparison of interval statistics by co-processor and client Scan

从实验结果分析，随着区间统计需要扫描Region数量的增长，客户端Scan统计所需的时间呈直线增长，而使用协处理器所需的时间则增长平缓，且当Region数量较少时，时间几乎没有增长。使用协处理器进行区间统计较使用客户端Scan至少快一个数量级（10倍）。

4　结　语

在服务器端使用Endpoint协处理器对城市空气质量监测数据进行常规的统计工作，会显著的减少统计所需的时间。理论上，若HBase的数据表在Hadoop集群的每个数据节点上的Region数量相同，且每个Region的大小相同，由于可以进行并行计算，此时Endpoint协处理器的工作效率达到最佳，这也是实验中，当Region的数量较少时，区间统计的时间几乎没有增长的原因。但随着数据的增长，Region的不断分裂导致其数量的增加，Region在每个数据节点上的数量不再相同，数据在各个Region上的分布也不再均衡，实验中在进行区间统计时，客户端和ZooKeeper服务进行RPC通讯时会出现延迟阻塞的现象，从而导致Region数量从7增加到9时，区间统计所需时间发生突变（增加近3倍）。如何有效解决这一问题，将是今后的研究方向。

参考文献

[1] 中华人民共和国环境保护部. 环境空气质量标准：GB 3095—2012[S]. 北京：中国环境科学出版社，2012.

[2] 中华人民共和国环境保护部. 环境空气质量指数AQI技术规定：HJ 633—2012[S]. 北京：中国环境科学出版社，2012.

[3] 中华人民共和国环境保护部. 环境空气质量评价技术规范（试行）：HJ 663—2013[S]. 北京：中国环境科学出版社，2013.

[4] 王珊，王会举，覃雄派，等. 架构大数据：挑战、现状与展望[J]. 计算机学报，2011，34(10)：1741-1751.

[5] 孟小峰，慈祥. 大数据管理：概念、技术与挑战[J]. 计算机研究与发展，2013，50(1)：146-169.

[6] 陈吉荣，乐嘉锦. 基于Hadoop生态系统的大数据解决方案综述[J]. 计算机工程与科学，2013，35(10)：25-35.

[7] MBARKA S，FOUTSE K，SOFIENE T. Task scheduling in big data platforms：a systematic literature review[J]. The Journal of Systems & Software，2017，134：170-189.

[8] YI Z，SHUBBHI T，GAUTAM D，et al. Towards thermal-aware Hadoop clusters[J]. Future Generation Computer Systems，2018，88：40-54.

[9] WU W T，LIN W W，HSU C H，et al. Energy-efficient hadoop for big data analytics and computing：A systematic review and research insights[J]. Future Generation Computer Systems，2018，86：1351-1367.

[10] Apache Software Foundation. Apache Hadoop[EB/OL]. (2014-06-30)[2018-09-06]. http://hadoop.apache.org.

[11] Apache Software Foundation. Apache HBase[EB/OL]. (2014-10-16)[2018-10-18]. http://hbase.apache.org.

[12] MINGJIE L，EUGENE K，ANDREW P. Coprocessor introduction[EB/OL]. (2012-02-01)[2018-10-26]. https://blogs.apache.org/hbase/entry/coprocessor_introduction.

[13] VASHISHTHA H，STROULIA E. Enhancing query support in HBase via an extended coprocessors framework[C]//4th European Conference on Towards a Service-Based Internet. Berlin：Springer-Verlag，2011：75-87.

[14] HAN D，STROULIA E. A three-dimensional data model in HBase for large time-series dataset analysis[C]//IEEE 6th International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA). Piscataway：IEEE，2012：47-56.

[15] 刘黎志，吴云韬. 环境空气质量监测大数据存储模式研究[J]. 环境科学与技术，2016，39(5)：123-128.

[16] 刘黎志，邓介一，吴云韬. 基于HBase的多分类逻辑回归算法研究[J]. 计算机应用研究，2018，35(10)：3007-3010.

本文编辑：陈小平