doi:10.3772/j.issn.1002-0470.2016.06.003

面向海量 NetFlow 数据的存储和查询处理方法研究①

陈重韬②*,**,*** 王伟平*** 孟 丹*** 崔 甲**** 胡 斌*,**

(*中国科学院计算技术研究所计算机应用研究中心 北京 100190)

(**中国科学院大学 北京 100049)

(***中国科学院信息工程研究所 北京 100093)

(****中国信息安全测评中心 北京 100085)

摘 要 针对全国骨干网高速海量 NetFlow 数据到来速度快、数据量大以及对所存数据进行频繁多维查询操作的特点,提出了一种多维属性聚簇存储(MACS)模型。该模型根据实际应用环境中查询的特点对数据进行空间分片,以并行加流水的方式对数据进行存储。此外,为 NetFlow 提出了一种超多面体的查询模式。真实环境实验结果表明,运用 MACS 模型实现的系统单点数据实时存储速度达到 270 万条/s,远远快于其他的数据分析系统,并且多维属性查询的速度优于 Hive 和 Impala。

关键词 NetFlow,多维属性聚簇存储(MACS)模型,实时数据存储,超多面体

0 引 言

伴随着信息技术和网络的快速发展,网络安全形势日益严峻[1,2]。NetFlow[3]作为收集和监控网络流数据的一种网络协议,被广泛用于网络流量统计、拒绝服务监控、入侵检测等方面[4-7],具有很高的应用价值和实际意义。目前,国内外在 NetFlow 数据的存储和分析方面的工作主要面向小规模网络,当面对全国范围的网络时,现有的解决方案均存在问题,因此需要对海量 NetFlow 数据的存储管理框架进行重新设计。

本文面向全国范围广域网骨干路由器发回的海量 NetFlow 数据,主要面临以下挑战:(1)高速写入。一般区域数据到来速度可达到 130 万条/s,对数据的实时存储提出了很高的要求;(2)数据近实时可查。由于应用场景要求,数据由存储到可查近似实时,不能有过高的访问延迟;(3)数据海量。现有数据针对全国各主要省市骨干路由器,每省每天存储数据压缩后接近 1TB,全国数据存储总量可达到 PB 级。伴随着网络扩容,以及各地方路由器的不断接入,势必会造成数据量成倍甚至数十倍的增长;(4)高频多维检索。NetFlow 数据会被频繁的检索访问,其中含有大量同时对流记录内不同属性的筛选类查询操作,因而需要对数据进行有针对性的组织,保证数据检索能力。

针对以上挑战,结合相关应用场景特征,本文提出了一种针对海量 NetFlow 数据的多维属性聚簇存储(multidimensional attributes clustering storage model,MACS)模型,简称 MACS 模型。该模型根据现有负载特征选择聚簇属性,通过对数据进行空间切分,有效过滤检索数据,在存储过程中采用并行加流水的方式,保证数据的快速实时存储,并在查询过程中,提出以超多面体的方式描述查询条件,进一步贴合场景要求。实验结果表明,运用 MACS 模型实现的系统单点实时存储速度达到 270 万条/s,并能提供高效的多维数据检索性能。

① 国家科技支撑计划(2012BAH46B03),国家自然科学基金(61402473),核高基(2013ZX01039-002-001-001)和中国科学院先导专项(XDA06030200)资助项目。

② 男,1986 年生,博士生;研究方向:海量数据的存储与处理;联系人,E-mail:chenzhongtao@ncic.ac.cn

(收稿日期:2016-03-29)

1 相关工作

传统的数据管理技术以关系数据库为代表,得益于成熟的索引及查询机制,其在数据查询方面的优势非常明显。但由于关系数据库在事务处理上具有强一致性的要求,导致其牺牲了数据加载的性能,并且系统结构难以扩展。同时,数据库的存储结构面向更新而设计,导致其数据的读写效率较低,而 NetFlow 数据采用实时传输并以追加的方式进行存储,数据存储之后不再进行修改,不需要上述严格的要求。

SILk[8] 作为面向 NetFlow 数据的专有系统,是由 CERT NetSA 开发并应用于大型网络的安全分析工具。相比于关系数据库,SILk 的存储格式简练,数据实时存储性能优异,能够满足较大流量的加载性能要求。但其只能采用全扫描的方式对数据进行处理,这对于需要进行多维度属性查询的应用来说,将造成大量的冗余数据扫描,严重影响查询性能。

以 Google 为代表的互联网公司针对各自应用中的海量数据特点,提出了一系列的 NoSQL 分布式数据管理技术,成为目前海量数据存储与管理的主流技术。针对海量数据的存储问题,Google 提出了分布式文件系统 GFS[9];面向高并发、低延迟的在线访问类应用,Google 提出了分布式多维表 BigTable[10],提供了基于键值的快速检索能力。Hadoop[11] 作为 Google Mapreduce[12] 和 GFS 的开源实现,与上层的 Hive[13] 以及 Hbase[14] 一起被广泛地应用在 Yahoo、Facebook、百度等大型互联网公司,取得很好的应用效果。但是,这些 NoSQL 技术都较为通用,应用到管理 NetFlow 数据上,在数据存储效率、加载性能以及查询性能方面还存在不足。

Hbase 作为 BigTable 的一个开源实现,以其良好的扩展性和快速查询能力,被广泛应用于结构化数据的存储。但在面对海量 NetFlow 数据时,首先,由于 Hbase 采用列存储,原始数据载入到 Hbase 时会有很多关于列和列簇的信息加入,从而造成严重的数据膨胀,直接导致存储开销成倍增长;其次,Hbase 的数据实时加载性能远低于海量 NetFlow 数据的到来速度;最后,现有的 Hbase 只针对包含主键列的查询拥有良好的筛选能力,对于其他属性列的查询只能通过全扫描的方式来实现,虽然针对多维属性查询问题,可以通过在 Hbase 之上建立二级索引的方式来解决,但是这种方式势必会进一步影响已经很有限的加载性能,并且二级索引所占用的存储空间也不容忽视。因此,Hbase 的解决方式并不适用于海量 NetFlow 数据。

Hive 作为当前海量数据仓库通用的解决方案,在面对海量 NetFlow 数据时,其数据加载以及查询的性能均不能满足要求。首先 Hive 作为离线分析系统而设计,因而并不能提供高效的数据实时载入性能;其次其查询的实质是强行全扫描数据,并不能对数据进行有效的筛选。与 Hive 对应的 Impala,受 Dremel[15] 的启发由 Cloudera 开发,采用类似并行数据库的大规模并行处理引擎(MPP),有效地提高了查询的性能,但底层仍然是以 HDFS 或 Hbase 作为存储介质,直接导致其实时存储性能不能满足要求,而且与 Hive 相似,Impala 的数据扫描方式也是全扫描,因而多维数据筛选类查询的性能并不十分出色。

综上所述,传统的解决方案都基于较小的数据集,当数据规模达到 TB 及以上级别时,均存在问题,而目前新型的通用海量数据解决方案又不能满足 NetFlow 数据对于实时存储以及多维度属性检索性能的基本要求,因而需要研究面向海量 NetFlow 数据存储与管理的新型体系结构。

2 MACS 模型

2.1 模型定义

给定一个具有 n 维属性的关系 $R(A_1, A_2, \cdots, A_n) \in D_1(D_2(\cdots(D_n$,其中 D_i 为属性 A_i 的定义域,关系 R 被视为 n 维数据空间 $S = D_1 \times D_2 \times \cdots \times D_n$ 的一个子集。多维聚簇存储模型首先从 n 维属性中选定一个或多个属性,针对每一维选定的属性进行划分,最终将空间 S 划分为若干子空间,原则是使 R 在每个子空间中具有近似相等的元组数。关系 R 即被划分成若干近似相等的子空间。依据上述描述,MACS 模型即选择数据记录的某个或某几个属性对整体数据进行空间切分,将海量数据划分到不

同的逻辑区域内存储。

下面以一个例子来说明多维属性聚簇模型在数据的存储和查询过程中所起到的作用。如图 1 所示在 3 个维度上做空间切分。对于值域范围内任意一条网络流记录所对应的 3 维属性的值,都可以在此空间内找到唯一的一块子空间与之对应。在数据存储过程中,随着流数据的不断到来,依据模型中对于各个属性的划分规则,将逐渐填充相应的空间。而根据已有规则查找数据的过程与存储过程相似,依据查询条件得到一个或多个子空间作为候选集,然后对候选集内的数据进行筛选查找,最终得到查询结果。例如查找的一条记录 Rec 的属性满足:$time_1 < Rec.time < time_2 \cap dip_0 < Rec.dip < dip_1 \cap sip_0 < Rec.sip < sip_1$,则在图示中深色的数据逻辑分片中进行查找。

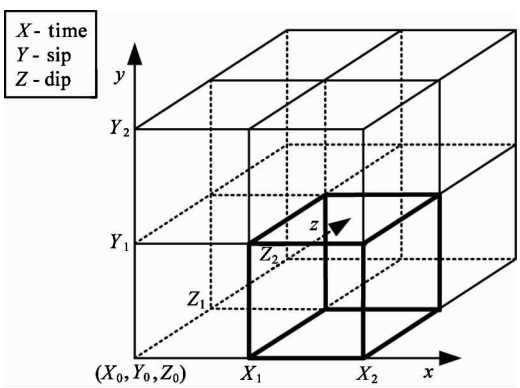

图 1 三维空间聚簇存储模型逻辑示意图

2.2 模型建立

聚簇属性选择的好坏将直接影响到查询效率,本文通过构建属性频度关联矩阵并统计属性整体出现频率的方式来选取聚簇属性。作为准备,需要统计查询负载信息,主要包括应用环境中常用的查询及其频度、查询中涉及的筛选类限制属性,算法中涉及到的符号描述如表 1 所示。

表 1 聚簇属性选择算法输入符号描述

符号	定义
R	包含 n 维属性的关系,$R = \{R_1, R_2, \cdots, R_n\}$
Q	在关系 R 上运行的查询的集合,$Q = \{Q_1, Q_2, \cdots, Q_m\}$
f_i	查询 Q_i 的查询频度

定义 1 关系 R 的属性频度关联矩阵的行表示作用在该关系上的不同查询,列表示关系中的所有属性,矩阵中若查询 Q_i 中涉及 R_j 属性,则该位置取值为 f_i,反之,取值为 0。

聚簇属性选取的第一步将根据负载统计信息依据定义 1 中描述构建属性频度关联矩阵,如表 2 所示,Q_1 包含属性 A_1 和 A_3,则矩阵中相应的值为 Q_1 的出现频度 f_1,以此类推;第二步基于该矩阵统计每一维属性出现的频度,A_1 的属性出现频度即为相应矩阵值求和;最后根据设置的阈值得到聚簇属性集合。关于阈值的设置,会根据应用场景中数据量的大小进行动态的变化。

表 2 属性频度关联矩阵示例

	A_1	A_2	A_3	A_4
Q_1	f_1	0	f_1	0
Q_2	f_2	f_2	f_2	0
Q_3	f_3	0	0	0

3 存储模式

针对 NetFlow 数据到来速度快的特点,并充分利用数据服务器的计算资源,数据服务器采用并行加流水的方法来保证数据的实时存储性能。数据存储引擎采取多线程压缩,最终多线程写盘,并且压缩与写盘过程分开的策略,其中压缩采用 zlib 算法进行。数据存储引擎在接收到数据后,将数据按照不同的聚簇属性区间进行划分,并添加到内存相应的一级缓冲区中,当缓存的数据量达到设定的阈值时,相应的数据存储线程会将缓存的数据进行压缩,并将压缩块写入到指定的二级缓冲区中,最终当二级缓冲区达到阈值之后,以追加的方式写到相应的数据文件中。图 2 给出了多维数据聚簇存储执行框架。

采用第二级缓冲区的目的是尽量保证数据的顺序写入。磁盘顺序读写的性能远远高于随机读写的性能,因而在数据存储的过程中需尽量保证数据的顺序写入。由于本文对 MACS 模型的实现需要存储大量的文件,在数据写盘过程中会同时打开多个文件进行后追加写,使得数据随机写入的概率增加。

虽然现有的文件系统本身有预分配策略,来尽量保证数据的顺序写入,但是该策略只在单线程写操作的情况下效果明显,对于多线程同时写多个文件,尤其是每次写数据量都很小的情况下并不能保证数据的连续性,从而严重影响数据的读写性能。因此我们采用增加写操作数据量的方式来尽量保证数据顺序写入磁盘,同时也会根据磁盘挂载的情况调整写队列的数量。

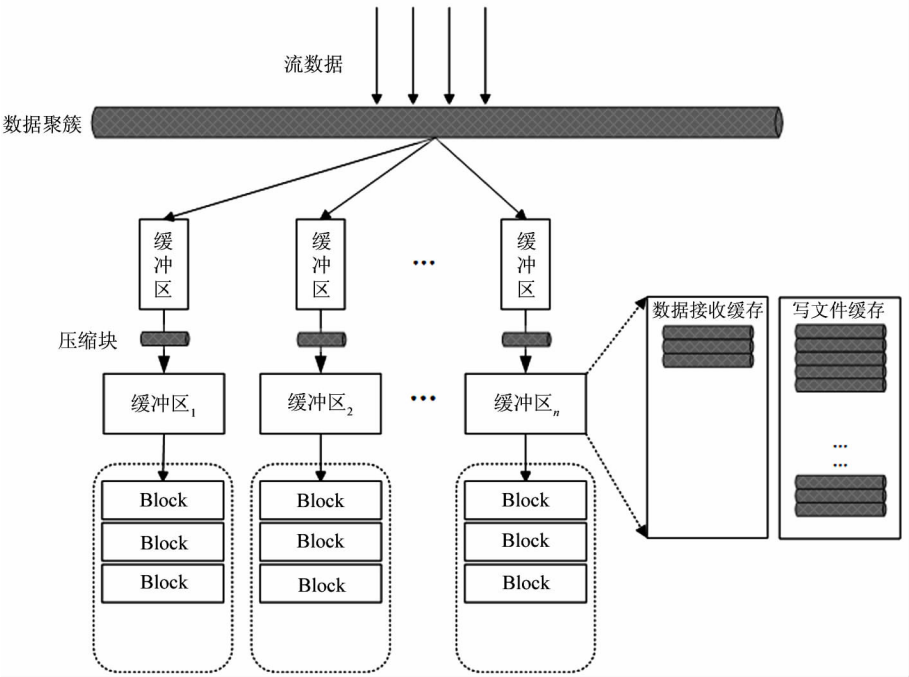

图 2　多维数据聚簇存储执行框架

为进一步提高数据的过滤能力,本研究对文件内部数据采用一种新的方式进行组织,如图 3所示。文件由数据头和数据块两部分组成,数据头用来记录数据块的基本信息,数据块内部则是一定时间范围内的流记录的集合。数据头内部 TimeRange 代表所对应的数据块的时间跨度,offset 记录该数据块在文件中的位置信息,File Head Offset 记录数据头在文件中的位置。每个文件都包含若干个数据块,在

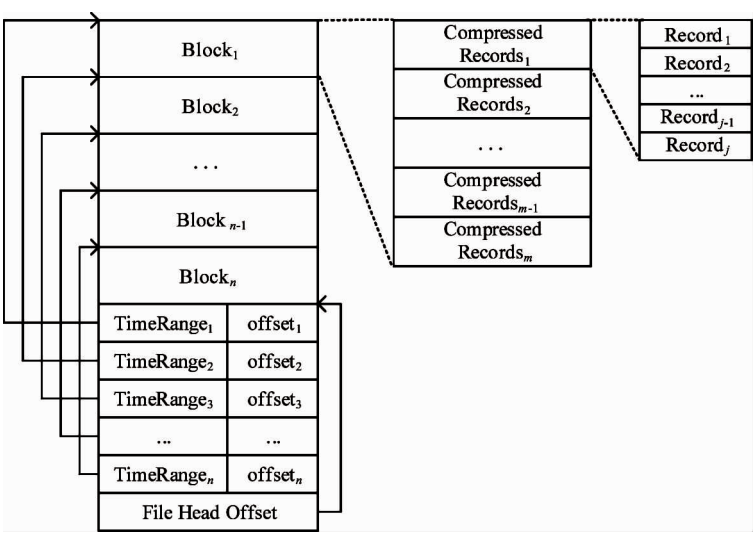

图 3　文件内部结果示意图

写入过程中每当缓存达到阈值时,便将处理后的数据块以后追加的形式写入到相应的数据文件中。文件头则一直保存在内存中,每当有新的数据块完成写入时,便在文件头中添加相应信息。直到文件关闭,数据头才会以后追加的方式写入到文件中,并同时更新 File Head Offset 信息。

在进行查询的过程中,会先读取文件头信息,查找符合查询条件的 TimeRange,并依据对应的位置信息读取相应的数据块。通过对文件进行如上组织,可以进一步减少查询中读取的数据量。

4　查询模式

本文所面向的应用场景为一个跨区域环境,查询系统采用两级分布式查询模型,如图 4 所示。

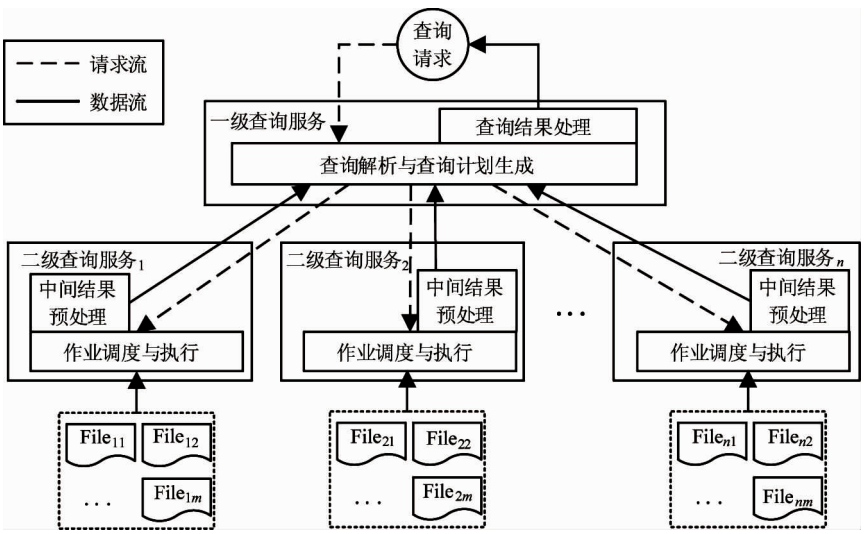

图 4　两级查询模型示意图

一级查询服务根据查询请求生成查询计划,通过查找元数据信息,直接将查询下发给相应的二级查询服务,最终由二级查询服务进行数据的扫描和处理。二级查询服务会根据查询的类型采用不同的结果回传方式,若为普通筛选类查询则进行实时分段回传,若为聚集排序等复杂查询,则先对中间结果进行预处理,之后返回给一级查询服务。一级查询服务也会针对不同的查询类型进行结果处理,包括分组、排序等,最终将查询结果返回。

在二级服务查询的过程中,依据场景特点,定义了超多面体模型,以超多面体的并、交、差来描述查询条件,以快速定位数据文件。

假设查询涉及关系的维度为 N, 那么这 N 维属性便构成了一个 N 维空间。任意一条值域空间范围内的流记录 Rec: $(r_1, r_2, \cdots, r_n)$, 都可以看做此 N 维空间中的一个点。同理,查询条件都可转化为此 N 维空间中的一个或多个超多面体,而查找满足查询条件的流记录,就等价于:(1)找到查询条件所对应的超多面体集合;(2)找到该集合中所包含的点。图 5 给出了一个查询条件转化示例。

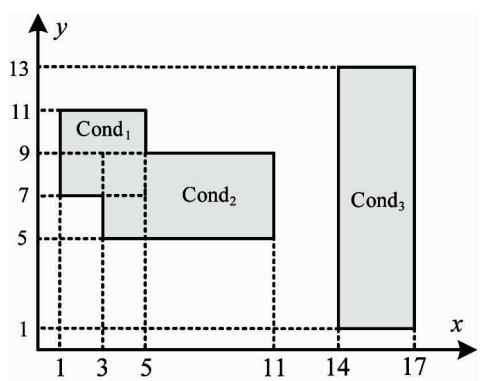

图 5　超多面体空间查询示例

图 5 中所示查询条件转化为 3 个超多面体的并:$\mathrm{Cond}_1 \cup \mathrm{Cond}_2 \cup \mathrm{Cond}_3$,其中:

$Cond_1$ 等价于:$1 \leqslant x \leqslant 5 \cap 7 \leqslant y \leqslant 11$

$Cond_2$ 等价于:$3 \leqslant x \leqslant 11 \cap 5 \leqslant y \leqslant 9$

$Cond_3$ 等价于:$14 \leqslant x \leqslant 17 \cap 1 \leqslant y \leqslant 13$

用超多面体来表达的查询条件具有析取范式的形式:$Cond_1 \cup Cond_2 \cup \cdots \cup Cond_m$,其中每个超多面体 $Cond_i$ 都是一系列属性条件的合取式:$A_1 \cap A_2 \cap \cdots \cap A_n$,这里每个属性条件 A_j 决定了 $Cond_i$ 的一条边,即流记录 Rec 若包含于 $Cond_i$,那么它在任意第 j 个查询属性上必须满足的条件。由于任何逻辑表达式都存在对应的析取范式,因此对于任意的查询条件,也总能找到与之等价的一组超多面体。用超多面体的方式描述查询条件,不仅直观,而且合乎 NetFlow 数据查询条件的常见方式。

5 实验结果与分析

以 MACS 模型为基础,采用上文提到的存储和查询模式,本研究实现了在线系统 SIEVE,根据场景要求,SIEVE 为跨区域的系统,已于 2012 年上线运行,本节所涉及的测试均采用 SIEVE 系统来进行。测试平台由 9 台服务器组成,服务器硬件为每个节点 4 个频率为 2.2GHz 八核 CPU,32GB 内存和 1TB 的存储容量。节点运行的操作系统为 Red Hat 6.2,内核版本号 2.6.32,节点间通过千兆网互连。其中,1 台节点作为 SIEVE 的查询控制节点,同时作为 Hadoop 集群的 master 节点,其余 8 台节点作为数据节点。实验中运行的 Java 环境为 1.7.0,使用的 Hadoop 版本为 2.6.0,Hive 版本为 1.1.0,Impala 版本为 2.3.0,Hbase 版本为 0.96.12,存储在 HDFS 上的文件副本数为 3,Hive 以及 Impala 所涉及数据均采用 Parquet 的文件格式进行存储。SIEVE 系统采用 C++实现,测试采用的数据均来自于实际线上环境。SIEVE 默认时间粒度设置 1h,选取源 IP、目的 IP 作为聚簇存储属性,而在每个维度上分为 4 个分片。

5.1 数据压缩比

本节测试数据的存储效率,方法为统计使用多维属性聚簇模型存储的数据所占空间大小以及数据中包含的 NetFlow 数据条数,即得到压缩后每条记录所占空间大小。经测试,压缩后单条数据大小约为 17 个字节,而原始数据单条数据大小为 48 个字节,压缩比约为 3:1,压缩效果明显。有如此明显的压缩效率主要原因是数据经过 MACS 模型组织后,数据块内均为相近的数据,并且最终采用二进制方式进行存储。

5.2 实时存储速率

对比 SIEVE、Hive 以及 Hbase 3 种数据存储方式的数据加载性能,其中 Hive 采用批量加载的方式,而 SIEVE、Hbase 采用在线实时加载的方式,比较三者的加载完成时间。图 6 给出了数据加载平均时间的对比。

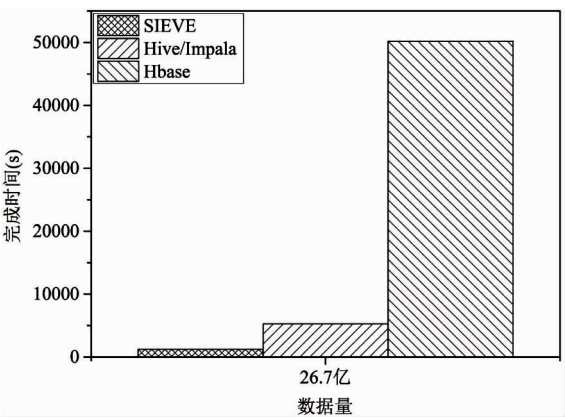

图 6 数据实时存储速度

图 6 中本文存储约 26.7 亿条数据,来测试数据存储速率,其中纵坐标为完成数据加载所需时间,由于测试环境为千兆以太网,因而 SIEVE 采用一台数据服务器进行接收。本研究可知,SIEVE 的数据实时存储速度最快,单点接收速度大概为 270 万条/s,基本接近了千兆以太网卡的流速上限,远高于 Hive 和 Hbase。

5.3 查询性能

对比 SIEVE、Hive 以及 Impala 的查询性能。实验中包括应用环境中最常出现的点查询以及聚集排序类查询,点查询的查询语句形式为"select * from table where sip = *** and dip = *** ",而聚集排序类查询的查询语句形式为"select sip,dip,dport,sum(bytes) as sum_bytes from table where sip = *** and dip = *** group by sip,dip,dport order by sum_bytes desc"。图 7 给出了查询执行的平均时间对

比。

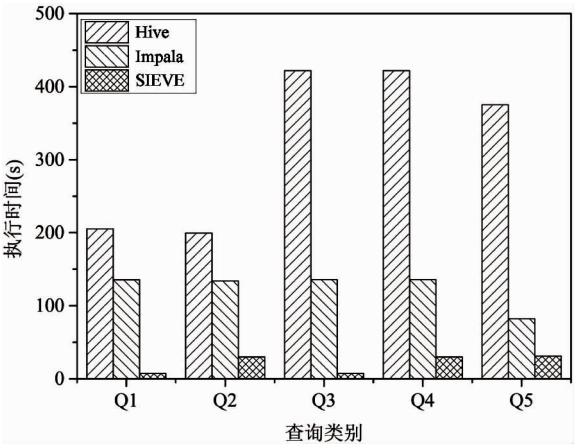

图 7 查询性能对比测试

图 7 中本文对约 26.7 亿条数据进行查询,测试查询完成时间,横坐标为查询类型,其中 Q_1、Q_2 为点查询,Q_3、Q_4、Q_5 为聚集排序类查询,Q_1、Q_3 查询条件中包括对源 IP 、目的 IP 的筛选限制,Q_2、Q_4、Q_5 查询条件仅包括对目的 IP 的筛选限制,Q_4、Q_5 区别在于最终结果列的数量,Q_5 仅对两维数据进行统计并最终排序。本研究可知,SIEVE 具有最好的查询性能。对于 Hive 和 Impala,虽然 Parquet 存储格式中有关于 block 内数据列最大最小值的统计信息,但是由于 NetFlow 单条记录小,导致每个数据块中所含的数据条数多,而 IP 分布本身具有分散的特性,这就使得统计信息的作用有限,最终仍需要扫描绝大部分的数据,因而执行时间均较长,尤其是 Hive 在执行聚集排序类操作时需要启动两个 MapReduce 作业来实现,导致耗费时间最长。Q_5 在执行时间上比同样进行目的 IP 筛选条件的 Q_4 更快是因为 Parquet 存储格式采用的是纯列存,因而查询结果中涉及的数据列越少,查询性能便越高。

图 8 中本文对不同选择率的查询性能进行比较,其中横坐标为点查询的查询结果条数,同时也为聚集排序类查询的符合条件的原始数据条数,查询条件中包括对源 IP 、目的 IP 的筛选限制。本研究可知,SIEVE 在不同的数据选择率下具有稳定且优越的查询性能。而 Hive 和 Impala 在不同选择率下查询也没有太大的波动,进一步说明 Parquet 存储格式的统计信息优化在本场景中作用有限。

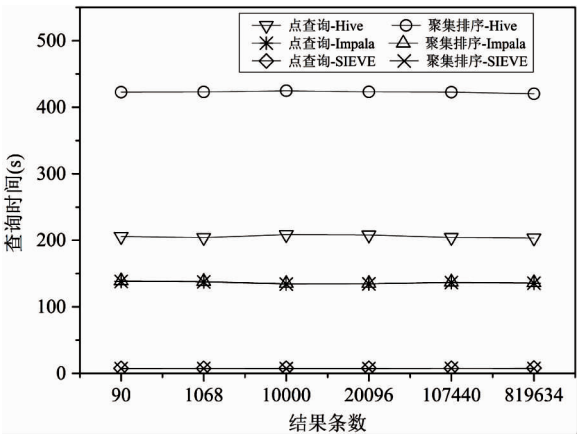

图 8 查询性能对比测试

5.4 系统扩展性

本节固定对约 26.7 亿条数据进行查询,通过增加机器的数量,查看查询时间变化趋势,如图 9 所示,由 5.3 节可知点查询与聚集排序查询性能表现一致,因而本节仅选取点查询进行测试。本研究可知,通过增加节点的数量,系统的查询性能近似线性提高。在线环境已运行超过 100 台节点,运行状况良好。

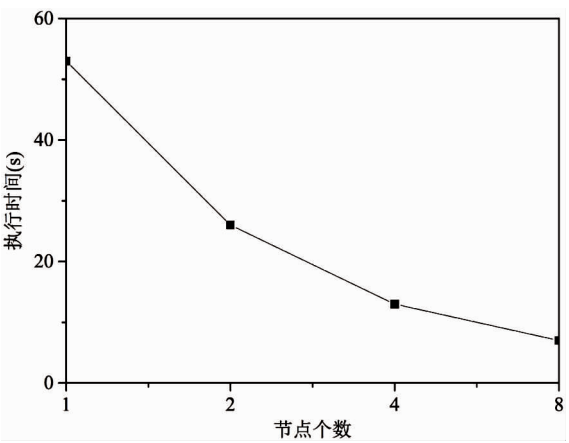

图 9 系统扩展性测试

6 结 论

面对快速到来的海量 NetFlow 流数据给存储和查询分析带来的挑战,本文提出了多维属性聚簇存储(MACS)模型。它通过分析现有负载查询特点,合理选择一个或多个属性作为聚簇属性,划分数据空间,从而减少数据处理过程中待处理的数据量,通过并行加流水的方式,充分利用各节点的计算和存

储资源,保证每一维数据的实时快速存储,并在查询中定义了超多面体的查询描述方式,使之更贴近应用场景,更加直观和高效地表示对于 NetFLow 数据的查询操作。实验结果表明,MACS 模型实现的系统单点实时存储性能达到 270 万条/s,并且多维数据查询能力也优于 Hive 和 Impala,能够提供稳定持续的查询服务。未来我们将研究新的文件组织形式来作为 MACS 模型的实现,进一步保证数据在磁盘上的顺序读写,保证读写性能,此外还将研究 MACS 模型在 Hive、Impala 等查询分析系统中的应用。

参考文献

[1] Cncert/Cc. 2013 年我国互联网网络安全态势综述. http://www.cert.org.cn/:国家互联网应急中心,2014

[2] Cncert/Cc. 2013 年中国互联网网络安全报告. 北京:人民邮电出版社, 2014

[3] Li B, Springer J, Bebis G, et al. A survey of network flow applications. *Journal of Network and Computer Applications*, 2013, 36(36):567-581

[4] Wang Z, Wang X. NetFlow based intrusion detection system. In: Proceedings of the 2008 International Conference on MultiMedia and Information Technology, Three Gorges, China, 2008. 825-828

[5] 吴斌, 丘劲松, 金连甫. 基于 NetFlow 的流量计费系统的设计与实现. 计算机工程, 2004, 30(7):189-191

[6] 陈宁, 徐同阁. NetFlow 流量采集与存储技术的研究实现. 计算机应用研究, 2008, 25(2):559-561

[7] Liu B, Lin C, Ruan D H, et al. Netfiow based flow analysis and monitor. In:Proceedings of the 2006 IEEE International Conference on Communication Technology, Vienna, Austria, 2006. 1-4

[8] Gates C, Collins M, Duggan M, et al. More netflow tools for performance and security. In: Proceedings of USENIX Conference on System Administration, Atlanta, USA, 2004. 121-132

[9] Ghemawat S, Gobioff H, Leung S. File and storage systems: the google file system. *Acm Symposium on Operating Systems Principles Bolton Landing*, 2003, 37:29-43

[10] Chang F, Dean J, Ghemawat S, et al. Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems*, 2008, 26(2): 205-218

[11] Bialecki A, Cafarella M. Hadoop: A Framework for Running Applications on Large Clusters Built of Commodity Hardware, http://lucene.apache.org/hadoop: Apache, 2008

[12] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters. *Communications of The ACM*, 2008, 51(1): 107-113

[13] Thusoo A, Sarma J S, Jain N, et al. Hive: a warehousing solution over a map-reduce framework. *Proceedings of the Vldb Endowment*, 2009, 2(2):1626-1629

[14] George L. HBase: The Definitive Guide. Sebastopol, USA: O'Reilly Media, 2011

[15] Melnik S, Gubarev A, Long J J, et al. Dremel: Interactive analysis of web-scale datasets. *Communications of the Acm*, 2011, 3(12):114-123

Research on storage and query processing for massive NetFlow data

Chen Zhongtao[*,**,***], Wang Weiping[***], Meng Dan[***], Cui Jia[****], Hu Bin[*,**]

([*] Center of Computer Application Research, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)

([**] University of Chinese Academy of Sciences, Beijing 100049)

([***] Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093)

([****] China Information Technology Security Evaluation Center, Beijing 100085)

Abstract

Considering that China backbone network's NetFlow data has the features of high arrival rate, large amount and need of frequent multidimensional query operation, the study proposed a multidimensional attributes clustering storage (MACS) model. According to the properties of real applicable queries, the proposed MACS model conducts space partition on NetFlow data, and stores the data in the way of parallel pipelining. Moreover, a hyper-polyhedron query mode for NetFlow data was presented. The experiments performed in real application environments show that the real time data storing rate of a single system realized with the model can achieve the storing rate up to 2.7 million records per second, which is more faster than all the other systems. Especially, the speed of the proposed multidimensional query is faster than Hive and Impala.

Key words: NetFlow, multidimensional attributes clustering storage (MACS) model, real time data storage, super polyhedron