第 43 卷 第 9 期 计 算 机 工 程 2017 年 9 月
Vol. 43 No. 9 Computer Engineering September 2017

·先进计算与数据处理· 文章编号:1000-3428(2017)09-0034-05 文献标志码:A 中图分类号:TP391

分布式信息网数据库管理系统的动态数据划分研究

马 杨,刘梦赤

(武汉大学 计算机学院,武汉 430072)

摘 要:在分布式信息网数据库管理系统中,数据是否被合理划分会影响系统的负载均衡以及节点之间的通信开销。为此,提出一种基于查询的动态数据划分算法。根据历史查询信息挖掘数据之间潜在的关联性,将关联性较大的数据动态调整到同一个处理节点上,使查询在较少的节点上处理完成,减少不必要的通信开销。实验结果表明,在保证系统负载均衡的情况下,该算法可减小通信开销,加快查询速度,优化分布式环境的整体性能。

关键词:动态数据划分;通信开销;关联性;负载均衡;分布式系统

中文引用格式:马 杨,刘梦赤. 分布式信息网数据库管理系统的动态数据划分研究[J]. 计算机工程,2017,43(9):34-38.

英文引用格式:MA Yang, LIU Mengchi. Research on Dynamic Data Partition of Database Management System in Distributed Information Network[J]. Computer Engineering,2017,43(9):34-38.

Research on Dynamic Data Partition of Database Management System in Distributed Information Network

MA Yang, LIU Mengchi

(School of Computer, Wuhan University, Wuhan 430072, China)

【Abstract】 For database management system in distributed information network, whether data is partitioned reasonably affects not only load balancing of the system but also the communication overhead between nodes. Aiming at this problem, this paper proposes a query-based dynamic data partition algorithm. According to the historical query information, it mines the potential relevance between data and dynamically adjusts the data with larger relevance to one processing node, so as to make the query processing completed in fewer nodes and reduce the unnecessary communication overhead. Experimental results show that, in the case of system load balancing, this algorithm can reduce the communication overhead, speed up the query and optimize the overall performance of the distributed environment.

【Key words】 dynamic data partition; communication overhead; relevance; load balancing; distributed system

DOI:10.3969/j.issn.1000-3428.2017.09.007

0 概述

数据划分是分布式系统实现中一个十分重要的问题,数据是否被合理地划分会直接影响分布式系统的性能。衡量分布式系统的性能有 2 个指标:负载均衡和通信开销。传统的数据划分算法主要从负载均衡的角度出发[1],研究数据的划分策略,其中具有代表性的多级划分算法[2-4],通过细化、划分、粗化 3 个阶段取得了较好的数据划分效果。这种划分策略对于彼此之间相互独立的数据具有较好的分布式性能,然而对于相互关联较大的数据,则会产生较大的通信开销。

信息网模型[5-7]是一种新型的面向对象的语义关联数据模型。在信息网模型中,现实世界中的每个实体对应于数据库中的一个对象,实体之间的关联表示成对象之间的关联。实体的所有信息保存在一个对象中,关联以源对象出发指向目标对象。因此,当从一个数据对象出发想要查询另一个数据对象的内容时,只需要从源对象出发沿着关联路径“跳入”目标对象中即可,不需要繁琐的 join 操作。然而在分布式环境下,数据被划分到各个处理节点上,如果相互之间关联的数据没有被划分到同一个处理节点上,那么频繁的跳对象操作会产生很大的通信开销。因此,如何进行数据划分使负载均衡的同时最小化通信开销成为一个研究难点。

针对数据划分中的通信开销问题,研究者提出

基金项目:国家自然科学基金(61202100);软件工程国家重点实验室开放基金(SKLSE2012-09-20)。

作者简介:马 杨(1993—),女,硕士研究生,主研方向为数据库技术、大数据;刘梦赤,教授。

收稿日期:2016-07-12 **修回日期**:2016-09-06 **E-mail**:mayang@whu.edu

大量基于K-balanced的图分割算法[8]。K-balanced图分割算法旨在均衡数据划分的同时,最大化减少节点之间的通信开销。文献[9]证明了 K-balanced 图分割算法是一个 NP 难问题,此后很多近似算法以及多层次启发式算法相继被提出[10]。但随着图规模不断增大,这些数据划分算法需要较大的时间开销,在实际开发中使用的可能性并不大。文献[11]采用一种细粒度的数据划分策略,将相关的元组分配在一个划分单元中,进而使得相关查询在一个节点上就可以完成。最小切边算法[12-15]则从最小化切边数出发减少通信开销。这些方法虽然在理论上可以取得较好的减少通信开销的效果,但是随着数据规模不断增大,算法本身实现起来过于复杂,使得数据划分本身的开销基本上抵消了减少的通信开销,因此,在实际的应用开发中也不具备可行性。

为解决并行系统中任务本身差异带来的负载不均衡问题,文献[16]提出一种基于日志的动态数据调整算法。该算法通过历史统计信息预估下一次迭代过程中的查询速度,并根据捕捉到的负载差异进行数据划分的动态调整,进而解决并行系统中因任务差异导致的负载不均衡问题。

结合文献[16]算法的设计思想,本文提出一种基于查询的动态数据划分算法。与文献[16]算法不同,该算法利用历史查询信息挖掘数据对象之间的关联性,将相互间关联度较高的数据动态调整到同一个处理节点上,进而减少因数据划分不合理导致的大量通信开销。同时还引入对象相关性的概念,用于表示 2 个对象之间的关联度,从侧面反映由一个对象出发跳至另一个对象中的可能性。当查询过程中产生跳对象操作,则动态修正 2 个对象之间的相关系数,进一步减少通信开销。

1　信息网模型

信息网模型[5-7]是一种新型的面向对象的语义数据模型。在信息网模型中,实体的所有信息都保存在一个对象中,实体之间的关联映射成关联对存放在源对象和目标对象中。该模型通过引入各种不同的关联类型以及关联层次,实现对于复杂语义的支持。

在信息网模型中,对象中包含对应实体的属性信息以及与其他对象之间的关联信息。当从一个源对象出发想要查询目标对象的内容时,只需要从源对象出发沿着指定的路径跳入目标对象进而获取到最终的查询结果。例如查询“武汉大学教授的论文发表情况”,此时便可以从“武汉大学”这个实体对象出发,沿着“教学部门”//“教授”这个关联路径跳入目标对象中,查询其“论文发表”对应的内容,并最终返回查询结果。

完成一个查询进行的跳对象次数除了与跳对象层数有关,同时也与每层跳对象中涉及的目标对象的个数密切关联。例如上述查询中,跳对象的次数即等于从实体对象“武汉大学”出发,关联“教学部门”与关联“教授”分别对应的目标对象的个数之和。由此可知,如果该查询涉及到的数据不在同一个处理节点上,处理节点之间频繁的跳对象操作将会导致大量的通信开销。

在分布式环境中,减少不同处理节点之间的通信开销的一个常见设计思路是使一个查询尽可能在一个处理节点上处理完成。本文即从该思路出发,结合历史查询信息,研究数据划分策略,使相互间关联度较高的数据尽可能被划分到同一个处理节点上,进而实现一个查询在一个处理节点上执行,减少因数据划分不合理导致的不必要的通信开销。

2　系统架构

本部分主要介绍分布式信息网数据库管理系统的整体架构,并结合此架构说明动态数据划分的实现机制。

如图 1 所示,集群中的节点分为 2 类:主节点和处理节点。其中主节点主要进行任务分发与数据划分,处理节点则是负责任务的具体执行和数据的动态调整。

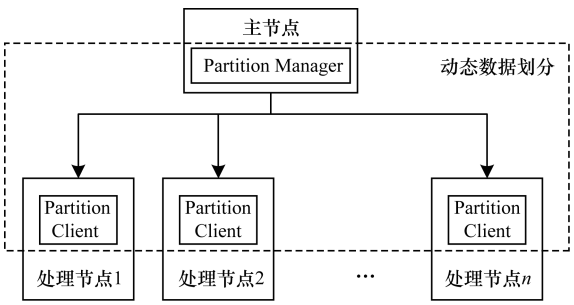

图 1　分布式信息网数据库管理系统架构

数据的动态划分包含 2 个阶段:数据的初始划分和数据的动态调整。考虑到实现的复杂程度以及数据划分算法本身的开销,本文采用一致性哈希作为数据的初始划分策略。为快速有效地定位请求数据所在的处理节点,系统动态地维护了一张对象 id-处理节点 id 表,用于存储任意一个已经存在的数据对象所在的处理节点。当用户发出一个请求时,主节点通过该表获取请求对象所在的处理节点并将任务下发,处理节点接收到任务进行处理的同时,记录处理过程中产生的通信开销,动态更新数据对象之间的相关指数。如果通信开销超过预设的阈值,则会反馈给主节点。主节点根据数据对象之间的相关指数,调用动态调整策略对处理节点的数据进行动态调整。

在数据的动态调整过程中,只需要根据查询信息动态维护对象之间的相关指数,并且在通信开销超过预设阈值之后才进行数据的调整。由于数据动态调整过程只依赖于历史查询信息而相对独立于数

据的初始规模,因此相对于以往的数据划分算法,动态数据划分算法在大规模数据的处理中具有更好的数据划分性能。不仅如此,由于动态调整过程不依赖于具体的数据划分算法和分布式环境的具体架构,因此该算法具有较好的通用性和可扩展性。

3 动态数据调整算法

3.1 对象相关指数的动态维护

本文引入对象相关指数 R_{ij}($R_{ij}\geqslant 1$)来表示对象 i 和对象 j 的关联程度。对象相关指数越高,表示查询过程中从一个对象出发跳入另一个对象的可能性越高。数据的动态调整分为 2 个阶段:对象之间相关指数的维护和数据的动态调整。在第一阶段中,R_{ij} 的初始值均为 1,并根据查询进行动态调整。一旦查询的过程中涉及到跳对象操作,则对应的 R_{ij} 进行加 1 操作。进入第二阶段之后,数据的动态调整算法根据第一阶段得到的 R_{ij} 进行数据的动态调整。

在数据操作的过程中,会统计上一次动态调整之后到目前为止数据操作所带来的通信开销 C_{total}。通信开销 C_{total} 包括节点内通信开销 C_{in} 以及节点之间的通信开销 C_{be},由于 C_{in} 相比 C_{be} 可以忽略不计,因此此处只统计跨节点的数据操作带来的通信开销 C_{be}。当 C_{total} 超过预设的阈值时,则对数据进行动态调整。为保证统计数据的有效性,同时防止数据划分调整过于频繁,根据统计数据指定最小调整周期 $T_{\min}$,即在最小调整周期之内,即使 C_{total} 已经超过了预设阈值,也不会马上进行调整,而是延迟一段时间进行。

为量化表示,此处用跳对象次数来表示通信开销,并引入一个字段 *flag* 来标识是节点内跳对象还是节点之间跳对象。在查询过程中,如果需要从一个对象出发跳入另一个对象中,则对应的 R_{ij} 加一。虽然笔者认为 C_{in} 相对于 C_{be} 可以忽略不计,但为评估数据动态调整的代价,对节点内跳对象也进行同样的统计。

3.2 数据的动态调整

当满足上述需要进行数据动态调整的条件时,即某段时间内 C_{total} 超过了预设阈值 $T_{\max}$,此时主节点根据数据的动态调整策略进行数据调整。在数据的动态调整过程中,一般方法是优先处理 C_{total} 较大的处理节点,因此,首先将各个处理节点按照 C_{total} 降序排列,然后按照顺序依次处理。

为说明方便,此处引入一个新的概念:通信开销贡献值。前文介绍了对象相关指数 R_{ij},用来度量从一个对象出发跳入另一个对象中的可能性。数据的动态调整是指将数据对象 O_{ij} 从原来所在的处理节点 $node_i$ 移动到另一个处理节点 $node_j$ 上。数据对象 O_{ij} 的通信开销贡献值则用于衡量由于 O_{ij} 移动到处理节点 j 上使得处理节点 j 减少的通信开销量,记为 T_{dec},其计算公式如下:

$$T_{\text{dec}}=\sum_{t\in node_j} RtO_{ij} \tag{1}$$

其中,RtO_{ij} 表示对象 t 与对象 O_{ij} 的相关指数,t 是 $node_j$ 上的任意数据对象。T_{dec} 即是指 $node_j$ 上所有对象与数据对象 O_{ij} 的相关指数之和。

针对每个待处理的节点,筛选出产生 C_{be} 的请求对象,并按照对应的 T_{dec} 降序排列,按照 T_{dec} 从高到低依次处理。针对每个请求对象,如果满足如式(2)所示的条件,则将请求的对象移动至该节点。

$$T_{\text{dec}}-T_{\text{inc}}>0 \tag{2}$$

其中,T_{dec} 表示该请求数据对该处理节点造成的通信开销贡献值;T_{inc} 表示请求对象原所在的节点由于请求对象被移动可能增加的通信开销。因此,两者的差表示数据移动带来的整体减少的通信开销。约束条件式(2)使得任意一次动态数据调整过程都会减少整体的通信开销。

对所有需要处理的节点重复上述过程,直至所有的节点都处理结束,至此一次数据的动态调整过程结束。分布式信息网数据库管理系统重新开始执行用户请求,对象之间的 R_{ij} 回归到初始值,重新按照查询进行动态维护,循环上述过程。

考虑到负载均衡问题,各个处理节点上的数据条目应满足以下条件:

$$n_i\leqslant\lambda(N/p) \tag{3}$$

其中,n_i 表示处理节点 i 上数据条目的个数;N 表示总的数据条目;p 表示集群中的节点总数;λ 为调整因子,可以侧面反映对负载不均衡的容忍程度,其值越大,表示容忍程度越高。λ 值可以根据统计信息获取。在数据的动态调整过程中,为保证负载均衡,须对式(3)进行验证。

在数据的动态调整过程中,会先根据不同对象产生的通信开销结合条件式(2)制定数据的调整计划,计划制定完成之后再进行条件式(3)的验证,在满足的情况下计划执行,否则修正调整计划。数据的动态调整策略可在减少通信开销的基础上,保证划分结果具有较好的负载均衡性。动态调整算法的具体描述如下:

算法 动态调整算法
输入 各个处理节点上的查询统计信息
输出 数据动态调整计划

```
1. 根据通信开销对各个处理节点进行降序排列;
2. For 排序后的每个处理节点
3.     根据查询的统计信息,筛选出产生跨节点通信开销的请求对象,并按通信开销贡献值进行降序排列(优先处理通信开销贡献值较大的数据对象);
4.     For 排序后的每个请求对象
5.         If 满足数据转移条件,即条件式(2)
6.             将该数据加入转移列表中;
7.         End If
8.         If 破坏负载均衡条件,即条件式(3)
```

```
9.            跳出循环;
10.       End If
11.    End For
12. End For
13. 返回数据调整计划;
```

4　实验与结果分析

本文利用实验对比分析一致性哈希与动态数据划分 2 种数据划分算法下,测试用例的通信开销与查询时间。分布式集群包括 1 个主节点和 6 个处理节点。实验在 Intel Xeon CPU E5-2630 +2.60 GHz,内存 100 GB,系统 Linux6.4 环境下运行。实验基于一个 8 GB 的数据集(包含 250 万个数据对象)进行,其中数据集中的数据是由 TPC-H 平台生成,并结合信息网模型中的数据格式转换得到。表 1 给出了实验中要使用的测试用例。Q1 ~ Q4 语句涉及到的数据量逐渐增加,其中,Q1 比较特殊,用于查询一个具体对象的内容,不涉及节点之间的通信。Q1 的设计主要用于测试不同时刻网络环境差异等客观因素对查询速度等指标的影响,对其他查询起到一个参考作用。Q2 ~ Q4 中涉及的节点间通信次数逐渐增加。Q2 ~ Q4 的设计旨在测试动态调整算法对于不同数据请求的优化能力。

表 1　测试用例

Q	查询语句	含义
Q1	query $x = nation0/supplier: $y construct $y;	查询 nation0 所有供应商
Q2	query $x = nation0/supplier: $y/s_phone: $z construct $y/$z;	查询 nation0 所有供应商的电话
Q3	query $x = region0/nation: $y/supplier: $z/s_phone: $k construct $y/$z/$k;	查询区域 region0 所有供应商的电话
Q4	query $x = nation0/supplier: $y/part: $z/p_type: $k construct $y/$z/$k;	查询 nation0 所有供应商提供的所有零件类型

4.1　算法性能比较

Q1 ~ Q4 在一致性哈希算法与动态调整算法下的查询时间与通信开销比较如表 2 所示。对数据进行分析可知,随着跳对象数据越多,产生的通信开销越多,同时查询时间也越长。当通信开销太大时,因通信造成的时间开销就成为了影响查询速度的首要因素,因此,通信开销的减少也会间接加快查询速度。分析实验数据可知,动态调整算法可以使得通信开销减小 40% ~60% ,同时查询时间减小 40% 左右。实验结果表明,该优化策略可以有效地减少因数据划分不合理导致的通信开销,同时较大程度提高查询性能。

表 2　2 种划分算法下的通信开销与查询时间对比

Q	一致性哈希算法		动态调整算法	
	通信次数	查询时间/ms	通信次数	查询时间/ms
Q1	0	8.747	0	9.113
Q2	35	39.274	15	17.466
Q3	412	174.043	223	89.577
Q4	6 043	1 347.000	3 611	911.522

4.2　负载均衡测试

动态调整前后各个处理节点上数据对象的个数比较如图 2 所示。可以看出,动态调整之后虽然各个处理节点上数据量都发生了改变,但是整体来说仍属于一种相对均衡的状态。实验结果表明,动态数据划分算法在根据对象之间关联性动态调整数据划分的同时,可使各个处理节点上的数据仍处于相对均衡的状态,确保了整体的负载均衡。

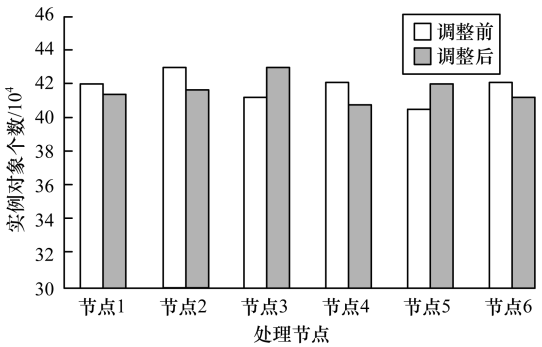

图 2　动态调整前后各处理节点上的数据量对比

5　结束语

本文针对信息网模型分布式实现中通信开销较大的问题,提出一种基于查询的动态数据划分算法。该算法利用历史查询信息挖掘数据对象之间的关联程度,进而将关联程度较高的数据动态调整到一个处理节点上,减少因数据跳对象操作而导致的节点间通信开销。实验结果表明,与一致性哈希算法相比,该算法可减少 40% ~60% 的通信开销,加快查询速度,优化分布式环境的整体性能。

考虑到实现的复杂程度以及数据划分本身的开销,本文中最初的数据划分基于一致性哈希算法实现。一致性哈希算法使数据均匀随机地被划分到各个处理节点上,导致相互关联的数据极大可能不在同一个处理节点上,因此,动态调整算法将会伴随大量的数据交换。一致性哈希算法虽然实现比较简单,但却很大程度地限制了动态调整算法的优化能力。因此,下一步将研究如何结合信息网模型的特点,设计初始数据划分算法,使相互关联的数据在最

初阶段就被划分到同一个处理节点上,结合本文的动态调整算法,使分布式信息网数据库管理系统达到更好的处理性能。

参考文献

[1] 杨小虎,王新宇,毛 明.基于数据划分的分布式模型及其负载均衡算法[J].浙江大学学报(工学版),2008,42(4):602-607.

[2] HENDRICKSON B, LELAND R. A Multi-level Algorithm for Partitioning Graphs[C]//Proceedings of 1995 ACM/IEEE Conference on Supercomputing. Washington D. C., USA:IEEE Press,1995:28.

[3] KARYPIS G, KUMAR V. Parallel Multilevel Graph Parti-tioning Schemes [D]. Minneapolis, USA: University of Minneapolis,1995.

[4] ARORA A, KAUR K. Enhanced Multilevel Hybrid Algorithm for Graph Partitioning [J]. International Journal of Computer Applications,2015,120(6):16-19.

[5] LIU Mengchi,HU Jie. Information Networking Model[M]// LAENDER A H F, CASTANO S, DAYAL U, et al. Conceptual Modeling-ER 2009. Berlin, Germany: Springer, 2009:131-144.

[6] 胡 捷,刘梦赤.信息网模型 INM 研究[M].武汉:科学出版社,2011.

[7] 徐 倩,胡 婕,刘梦赤.复杂语义关系的描述与操作[J].计算机科学与探索,2014,8(12):1432-1441.

[8] GAREY M R, JOHNSON D S, STOCKMEYER L. Some Simplified NP-Complete Problems[C]//Proceedings of ACM Symposium on Theory of Computing. New York, USA: ACM Press,1974:47-63.

[9] ANDREEV K, RACKE H. Balanced Graph Partitioning [J]. Theory of Computing Systems, 2006, 39(6): 929-939.

[10] GUY E, JOSEPH N, SATISH R, et al. Fast Approximate Graph Partitioning Algorithms[C]//Proceedings of the 8th ACM-SIAM Symposium on Discrete Algorithms. [S. l.]: Society for Industrial and Applied Mathematics, 1997: 2187-2214.

[11] TATAROWICZ A L, CURINO C, JONES E P C, et al. Lookup Tables: Fine-grained Partitioning for Distributed Data-bases[C]// Proceedings of the 28th IEEE International Conference on Data Engineering. Washington D. C., USA:IEEE Press,2012:102-113.

[12] GUTTMANN-BECK N, HASSIN R. Approximation Algorithms for Minimum K-cut[J]. Algorithmica,2000, 27(2):198-207.

[13] GHAFFARI M, KUHN F. Distributed Minimum Cut Approxima-tion [M]//Afek Y. Distributed Computing. Berlin, Germany: Springer, 2013:1-15.

[14] NAGAMOCHI H, IBARAKI T. A Fast Algorithm for Computing Minimum 3-way and 4-way Cuts [J]. Mathematical Programming,2000,88(3):507-520.

[15] DAN G, ÉVA T. A Faster Parametric Minimum-cut Algorithm[J]. Algorithmica,1994,11(3):278-290.

[16] XU Ning, CHEN Lei, CUI Bin. LogGP: A Log-based Dynamic Graph Partitioning Method[J]. Proceedings of the VLDB Endowment,2014,7(14):1917-1928.

编辑 金胡考

(上接第 33 页)

[5] 陈 佐,谢 赤,陈 晖.基于小波聚类方法的股票收益序列时间模式与挖掘[J].系统工程,2005,23(11):102-107.

[6] MAHESHWARY P, SRIVASTAVA N. Wave Cluster for Remote Sensing Image Retrieval[J]. International Journal on Computer Science and Engineering, 2011, 3(2):976-979.

[7] MICHAEL B. Mammographic Segmentation Using Wave Cluster[J]. Algorithms,2012,5(3):318-329.

[8] LI Xingyi, LU Junyun, SHI Huaji, et al. An Approach for Treatment of the Incomplete Data Based on WaveCluter and Weighted 1-nearest Neighbor [C]// Proceedings of 2009 International Association of Computer Science and Information Technology Conference. Berlin, Germany: Springer, 2009:3-8.

[9] ANGGRAINI E L, SUCIATI N, SUADI W. Parallel Computing of Wave Cluster Algorithm for Face Recognition Application[C]//Proceedings of International Conference on QIR. Washington D. C., IEEE Press, 2013: 1195-1203.

[10] YILDIRIM A A, ÖZDOğAN C. Parallel Wavelet-based Clustering Algorithm on GPUs Using CUDA [J]. Procedia Computer Science,2011,3(1):396-400.

[11] 王治和,杨 晏.基于双层网格和密度的数据流聚类算法[J].计算机工程,2014,40(4):146-150.

[12] 林秀丹,毛国君.基于密度网格的分布式数据流聚类算法[J].计算机工程,2012,38(16):70-73.

[13] 张书春,孙秀英.基于网格结构的 CLARANS 改进算法[J].计算机工程,2012,38(6):56-59.

[14] 单世民,张 宁,江 贺,等.基于网格和密度的簇边缘精度增强聚类算法[J].计算机工程与应用,2008,44(8):143-146.

[15] 余灿玲,王丽珍,张元武.基于网格密度方向的聚类簇边缘精度加强算法[J].计算机研究与发展,2010,15(5):816-822.

编辑 索书志