软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2020,31(1):162−182 [doi: 10.13328/j.cnki.jos.005886]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

大数据实时交互式分析*

袁 喆[1], 文继荣[1], 魏哲巍[1], 刘家俊[1], 姚 斌[2], 郑 凯[3]

[1](中国人民大学 信息学院,北京 100872)

[2](上海交通大学 计算机科学与工程系,上海 200240)

[3](电子科技大学 计算机科学与工程学院,四川 成都 610054)

通讯作者: 文继荣, E-mail: jrwen@ruc.edu.cn

摘 要: 实时交互式分析针对多目标和多角度的分析任务,通过多轮次的用户-数据库交互过程,逐步明确分析任务与分析目标,全方位地了解相关领域信息,最终得到科学的、全面的分析结果.相比传统数据库"提交查询-返回结果"的单轮次交互查询方式,实时交互式分析更强调交互的实时性与查询结果的时效性.对实时交互式分析的研究已成为近几年研究的热点.针对当前实时交互式分析面临的若干关键问题,对现有的实时交互式分析研究的理论基础、数据模型与系统构架进行了综述.

关键词: 实时交互式分析;跨模态数据;近似查询处理

中图法分类号: TP311

中文引用格式: 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.软件学报,2020,31(1):162−182. http://www.jos.org.cn/1000-9825/5886.htm

英文引用格式: Yuan Z, Wen JR, Wei ZW, Liu JJ, Yao B, Zheng K. Real-time interactive analysis on big data. Ruan Jian Xue Bao/Journal of Software, 2020,31(1):162−182 (in Chinese). http://www.jos.org.cn/1000-9825/5886.htm

Real-time Interactive Analysis on Big Data

YUAN Zhe[1], WEN Ji-Rong[1], WEI Zhe-Wei[1], LIU Jia-Jun[1], YAO Bin[2], ZHENG Kai[3]

[1](School of Information, Renmin University of China, Beijing 100872, China)

[2](Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai 200240, China)

[3](School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China)

Abstract: Real-time interactive analysis focuses on multi-object and multi-perspective analysis tasks. By employing a multiple user-database interaction process, interactive analysis is able to provide a more comprehensive understanding of the analytic task. Comparing to traditional database where queries are issued and answered in a single interaction, interactive analysis emphasizes on the responses time of the query and timeliness of the results. Real-time interactive analysis has been extensively studied in recently years. In this survey, comprehensive review is provided on the theoretical foundation, data models, and systems of the real-time interactive analysis.

Key words: real-time interactive analysis; cross-modal data; approximate query processing

数据库是大数据时代的中心,是大数据时代能够高速运行的基础保证.传统的数据库与信息检索系统通常

* 基金项目: 国家自然科学基金(61832017, 61972401, 61932001, 61602487, 61922054, 61872235, 61729202, U1636210, 61972069, 61836007, 61532018); 北京高校卓越青年科学家计划(BJJWZYJH01201910002009); 国家重点研发计划(2018YFC1504504, 2016YFB0700502)

Foundation item: National Natural Science Foundation of China (61832017, 61972401, 61932001, 61602487, 61922054, 61872235, 61729202, U1636210, 61972069, 61836007, 61532018); Beijing Outstanding Young Scientist Program (BJJWZYJH01201910002009); National Key Research and Development Program of China (2018YFC1504504, 2016YFB0700502)

收稿时间: 2018-09-14; 修改时间: 2019-06-13; 采用时间: 2019-09-26; jos 在线出版时间: 2019-11-06

CNKI 网络优先出版: 2019-11-06 11:49:20, http://kns.cnki.net/kcms/detail/11.2560.TP.20191106.1148.009.html

采取“提交查询-返回结果”的单次交互的查询方式,由用户提交查询,数据库返回与该查询相关的结果.然而现实世界中,用户对于数据库与信息检索系统的使用往往需经历多次交互.交互式分析针对目标未知或多目标、持续性、多角度的分析任务,通过策略性、多轮次的用户-数据库交互过程,逐步明确分析任务与分析目标,全方位、多角度地了解相关领域信息,最终得到科学的、全面的分析结果.与传统的数据库分析任务不同,交互式分析强调用户与数据库的交互性,而实时交互式分析则进一步强调交互的实时性与结果的时效性,如图 1 所示.(a) 传统的数据库查询.在传统的数据库查询情景下,用户对数据库信息空间的结构和内容完全了解,并且用户在信息空间中的搜索目标是确定的,用户根据搜索目标执行多次查询,最终从信息空间中获取搜索目标,过程如图 1 左图中的结果集 1~3 顺序所示;(b) 实时交互式分析.在实时交互式分析场景下,用户的查询意图是不完全清晰的,故而用户不存在明确的搜索目标,用户根据自身查询意图以及与系统的交互结果确定下一步执行的查询,直至得到满足需求的查询结果,过程如图 1 右图中的结果集 1~9 顺序所示.

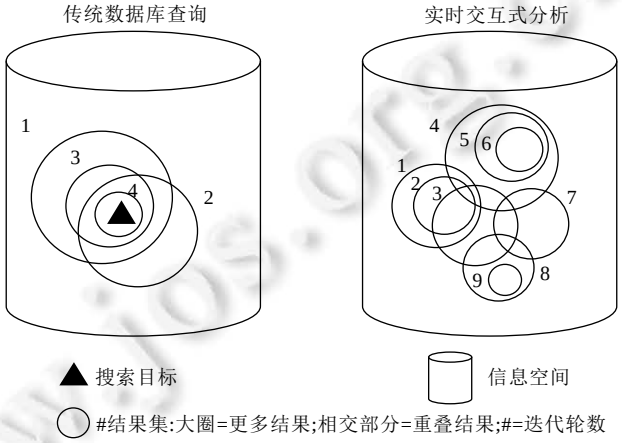

Fig.1 Query with traditional DBMS vs. query with real-time interactive system

图 1 传统数据库查询与实时交互式分析

在很多应用场景中,用户并不完全清楚自己想要的查询结果,因此也无法精确描述查询,往往需要通过与数据库的反复交互来抽象出真实的查询意图,如图 2 所示.

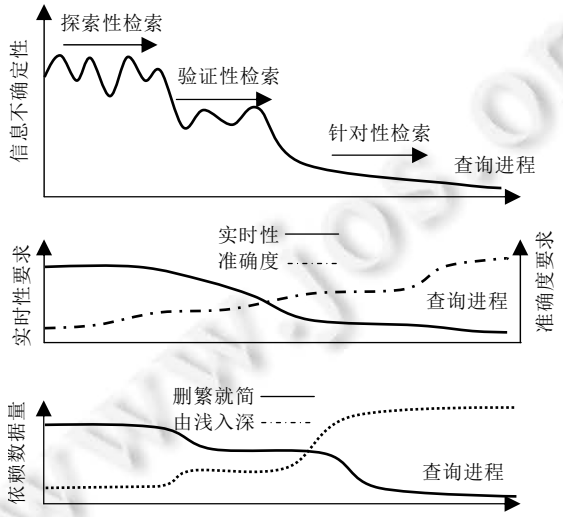

Fig.2 Query situation and analysis of interactive query

图 2 交互式查询情景与分析

数据库领域对实时交互式分析的大量关注,起源于 Google 发布的 Dremel 系统[1],该系统改进了传统的 MapReduce 以批处理为主的数据管理模式,允许用户以交互级响应时间完成 SQL 查询.受到 Dremel 的启发,学界与工业界开始研究在大数据环境下实现实时交互式分析的可能.下为实时交互式分析若干经典应用场景.

(1) 探索式搜索(exploratory search):探索式搜索[2]最早由权威专家 Marchionini 于 2006 年提出,其核心思路是:用户往往对想要搜索的结果要求并不明确,需要通过反复的交互进行探索,其搜索过程是有选择、有策略和反复进行的.例如,天文科学家往往需要在连续的数据流中寻找感兴趣的数据模式,其数据量可达 TB 级别.然而,科学家可能无法精确给出“感兴趣的数据模式”的具体刻画,只有当模式呈现之后才能确定其有效性.因此,这类搜索通常难以形成固定的数据库查询,而必须依赖于利用探索式搜索.在探索式搜索过程中,通常只有通过与数据库的反复交互,才能最终确定用户的查询意图,从而得到满意的查询结果;

(2) 交互式科学假设检验(interactive scientific hypothesis testing):在科学数据库中,科学家通常会建立一个初始假设,并通过对科学数据的反复查询、探索,不断修正、改进假设.通过与数据库的反复交互,科学家最终将获得符合实际数据的假设.如果与数据的交互速度过慢,科学研究的效率就会因此下降.因此在这类应用中,交互的实时性是关键因素;

(3) 敏捷式算法部署(agile algorithm deployment):数据分析师在完成分析算法代码编写工作后,通常需要验证代码是否正确、是否可以有效提升分析结果.在传统数据平台(如 MapReduce)上完成代码部署时,由于数据量巨大,得到最终分析结果通常需要数小时甚至更长时间,不利于对代码的快速修改与更新.在这类应用中,用户希望在修改的代码能实时反应到分析结果中,从而可以快速判断代码的正确性与有效性,并且通过反复交互,最终得到成熟的分析算法代码.

实时交互式分析受到国内外学术界与工业界的广泛关注,对实时交互式分析的研究日益深入.本文将实时交互式系统自底而上分为数据层、系统层、算法层以及分析层,通过对分析层所见主要查询任务的分类介绍,引出为满足相应查询需求而设计并实现的原型及系统;并按照其所基于的计算平台,采用的计算引擎、优化技术、更新策略,以及系统内的核心算法等维度,从实时交互式分析理论基础与系统实现两个方面综述实时交互式分析的研究现状,综述框架及脉络如图 3 所示.

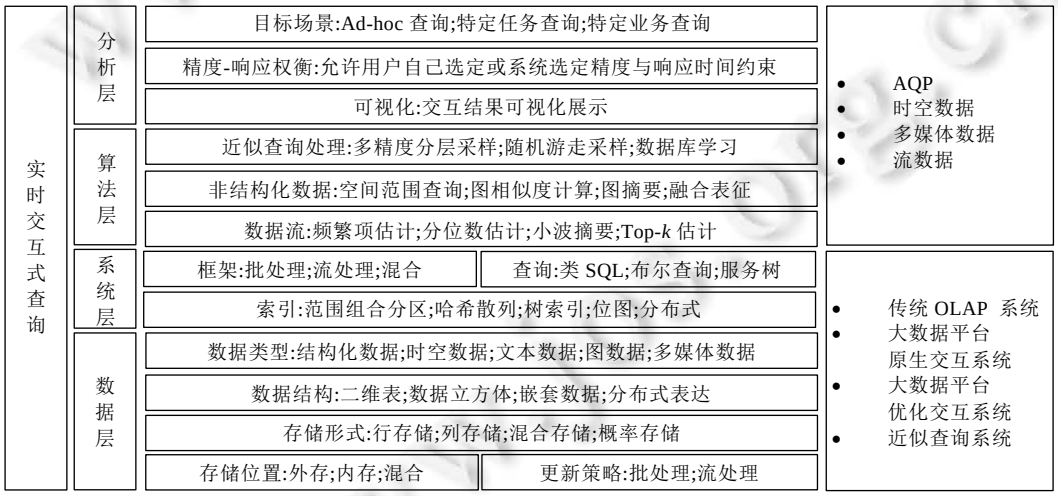

Fig.3 Architecture and core technology of real-time interactive query systemand the survey routine

图 3 实时交互式查询系统架构、核心技术及综述脉络

本文第 1 节对交互式分析的典型场景进行分析,并对所需技术支持作细致分析,引导出当前学术研究与工业应用面临的挑战.第 2 节围绕近似查询算法、图数据与流数据,对支持交互式系统实时性与时效性的核心技

术理论进行介绍,并提出关键研究问题.第 3 节依托于当前支持交互式分析的系统实现方案,沿传统 OLAP 分析,到各类基于大数据分析平台的分析引擎,到基于近似查询算法的引擎,介绍交互式分析的发展历程,分析各形态下的主要技术与性能特点.第 4 节总结国内研究情况并提出关键研究问题及研究方向.

1 实时交互式分析关键问题

相较于传统数据库的查询与分析,实时交互式分析可以让用户更有效地做出决策.然而,对于实时交互式分析的研究需解决以下几个关键问题.

(1) 交互级响应时间(interactive response time).

相比于传统数据库查询同样提供的交互式分析功能,实时交互式分析的核心要求是满足人们“探索式查询”需求的低响应时间.人机交互方向的研究[3,4]显示:为保证用户的活跃度,在交互式查询环境下提供服务的系统,对于包含多种信息需求复杂的查询响应不应超过 10s;对于信息需求较为明确的查询响应时间不应超过 2s;对于以可视化展示结果的查询响应时间不应超过 0.5s.因此,实现交互级响应时间是实时交互式分析系统的先决条件.然而随着数据量的增长,秒级或毫秒级的查询时间对于很多复杂分析查询在理论上是非常困难的,其具体实现也为大数据系统构架提出了非常高的要求.另一方面,在交互式分析过程中,由于交互次数的增加,用户往往能够容忍一定的误差.如果系统能实时给出误差的范围(如置信度与置信区间),用户可以通过多次交互,根据查询结果和误差范围得出分析结论.因此,如何引入可控的误差以获得理想的交互级查询时间,是实时交互式分析面临的首要挑战.

(2) 对跨模态数据(multimodal data)的支持.

实时交互式分析要求对跨模态高维度数据具备处理与分析能力.相对于传统的单模态数据,跨模态数据往往同时包含异构(半结构化和非结构化共存)和异质(不同质量的数据共存)的数据类型,在数据维度、数据分布和数据特征上也变得更加复杂和多样化.这使得传统大数据分析框架下针对单一模态数据的索引结构和查询优化策略无法继续有效地工作.目前,主流的实时交互数据库系统主要针对结构化数据库进行设计与优化,缺乏对于文本数据、时空数据、多媒体数据、图数据等非结构化数据的支持,无法对跨模态数据做出有效分析.

要对跨模态高维度数据具备处理与分析能力,交互式分析系统需要设计良好的组合索引机制.现有的交互式分析系统支持的数据索引类型较为单一,如针对结构化数据的数据立方体、针对空间数据的各类树索引、针对文本数据的倒排表索引以及针对高维连续向量近似查询的各种类局部敏感哈希索引等.在跨模态交互式查询的场景之下,由于不同模态的数据难以使用同一种统一的索引结构,需要在保持同一份数据记录的跨模态表达结构的前提下,使用多种索引来支持在各个模态或维度上进行查询.比如:带有文本的多媒体数据,需要快速融合“文本匹配查询”“视觉语义查询”“视觉近似度查询”等多种查询模型,能在交互的有效时间内从上述 3 个维度去获得查询结果和聚合分析结果,因而对跨模态组合索引的设计提出了极高要求.另一方面,由于跨模态数据具有多种类型的数据表达形式,如何有效地设计分析查询过程中的查询输入和结果更新这一交互过程,也是一个重大挑战.

(3) 实时交互式分析系统(real-time interactive analysis system)的实现.

首先,联机分析处理 OLAP(on-line analytical processing)是传统的数据仓库与商业智能中交互查询的代表,能从数据库中较为灵活与高效地进行复杂、大数据量的统计汇总和聚集,快速得到分析结果.然而,以 OLAP 为主的传统交互查询主要实现方式仍为依赖于 SPJA(select-project-join-aggregation)查询模式,其处理对象也主要为传统关系型数据库中的表数据,无法对跨模态数据进行分析,限制了 OLAP 系统对实时交互式分析的支持.其次,现有的基于分布式大数据平台的交互查询系统,如 Google Dremel、Spark SQL、Presto 等,通过对存储、调度、查询策略等方面进行优化来提供交互级响应时间.其核心优化目标大致可以简化为两类:(1) 高效访问所有查询相关的元数据;(2) 选取所有等价执行策略中效率最高的策略完成查询.然而随着数据量的持续扩张,即使在已知查询所覆盖元数据的前提下,在交互级响应时间内访问所有相关元数据并精确返回查询结果,仍是一个公认的难题.再次,现有的基于近似查询处理 AQP(approximate query processing)的交互式分析系统,通过采用随机

采样、数据摘要等技术,牺牲部分查询精度显著降低查询时间,以适应实时交互式分析.然而,目前的近似查询处理引擎无法为所有 SQL 查询提供预估的误差界.因此,用户必须在查询结束后才能得到查询结果与误差范围,不能完全满足交互式分析的实时性.此外,当前的近似查询处理引擎的插入与删除策略都依赖于批量更新(batch update),不能支持增量更新,难以保证查询结果的实时性.综上所述,实时交互式分析对现有大数据系统的系统资源、系统构架、调度策略等方面都提出了非常大的挑战.

2 实时交互式分析算法理论基础

实时交互式分析是一个大数据领域多个方向融合交叉的课题,目前的实时交互式分析系统也广泛使用了多种优化技术、算法理论与系统构架.为达到交互级响应时间,当期的实时交互式系统通常采用计算升级与数据降级两种思路:计算升级指的是在分布式环境中,通过对 CPU、内存等计算资源的合理调配,实现低响应时间的精确查询;数据降级则指的是利用采样、摘要、略图等近似查询技术将大数据转化为小数据,在满足预定分析结果精度的前提下实现交互级响应时间的查询.第 1 种思路的难点主要在系统实现方面,本文中我们重点总结第 2 种思路的理论基础,包括近似查询处理(approximate query processing,简称 AQP)算法、非结构化数据查询算法与数据流算法.同时,一些新的计算模型和算法概念也被考虑,如 I/O 模型[5]、Cache-oblivious 模型[6]、数据流模型[7,8]、分布式计算模型[9,10]等.

2.1 近似查询处理算法

相较于传统的数据库系统,支持近似查询处理(approximate query processing,简称 AQP)的数据库系统更适用于实时交互式分析,其原因有三.

(1) 传统的数据库系统要求访问所有与查询相关的数据,数据库通过选取不同的查询执行计划(query plan)进行优化.所有传统数据库的优化技术全部基于这一前提.在通常情况下,即使数据库的总数据量很大,和单个查询相关的数据量可能会很小,因此查询效率能够得以保证.然而随着大数据时代的来临,数据量进一步爆炸性扩张,现有硬件受到内存访问速度极限(memory wall)的限制,将会在理论上也不足以提供实时交互式分析所需要的响应时间.而近似查询处理只需访问元数据的一小部分,有可能实现更快的响应时间和更高的空间效率;

(2) 传统的数据库系统无法提前给出查询响应时间,查询结果只有在查询结束后才能展示.近似查询处理可以在更短的响应时间内返回查询结果,基于在线采样的近似查询算法则可以在查询开始时就返回近似查询结果,并且随着查询时间的增长,查询结果逐渐趋于精确;用户可随时终止查询,进行下一轮交互.近似查询处理的这一特性使其更适用于实时交互式分析;

(3) 传统的数据库系统无法复用历史查询,而近似查询算法可以通过历史查询学习数据分布,从而获得对于未来查询的额外信息.

近年来,若干工作提出了基于抽样[11]、摘要[12]等技术的近似查询算法,可在较短时间内提供有置信区间保证、满足一定错误率的近似查询结果.其中,部分工作还进行了系统实现[13−15].通过近似查询算法提供实时的、有保证的近似结果,可满足交互查询对速度和查询质量的要求.按照处理数据的类型分类,近似查询算法主要针对类似 SQL 的查询,如连接、聚集等操作,解决数据量过大导致查询速度较慢的问题,同时考虑 I/O 或分布式模型下算法的优化.对于近似查询处理引擎中查询算子的实现,在算法层面可大致分为离线采样技术(online sampling)和在线采样技术(offline sampling),我们分别对其模式与特点进行综述与对比.

2.1.1 离线采样

离线采样在回答查询之前,就对整个数据库进行预采样并进行存储.数据库通常会存储多个不同精度的采样.给定一个查询,数据库先通过执行计划(query plan)预估该查询所覆盖的数据量,根据这个数据量选取某个精度的预采样,并在预采样上执行精确查询算子.在查询结束后,数据库可以根据 Chernoff 不等式给出近似结果的置信区间和置信度.当前,基于离线采样的近似查询系统的代表为 Agarwal 等人提出的 BlinkDB[16].由于离线采样算法在现有数据库系统中实现时不需要新建索引,编程难度相对较低,因此,离线采样算法是目前支持近似查

询处理引擎的数据库系统中主流的实现方式.然而,离线采样算法存在 3 个明显问题.

(1) 离线采样算法无法给出一个预估的误差界.离线采样算法只有当查询执行完成之后,才能给出其查询结果的误差界.其原因在于:离线采样算法基于预采样执行查询,在查询完成之前无法确定查询结果大小,因此也无法确定误差界.这对于通过离线查询实现交互查询是一个重大缺陷:在查询完成之前,用户无法实时得到查询结果和误差界,也无法提前终止查询进行新一轮交互;

(2) 离线采样算法对于偏态分布数据(S,skewed distributed data)效果不理想.考虑如下 SQL 查询:

Q1: SELECT *B*, SUM(*A*) FROM *T*, WHERE *C*=10, GROUP BY *B*.

预采样的误差界取决于满足条件 *C*=10 的数据个数:如果满足条件的数据个数较多,则预采样可以提供较为精准的查询结果;如果满足条件的数据个数过少,则预采样可能完全无法采集到任何满足 *C*=10 的数据,从而也无法得到合理的估计结果.目前,最新的基于离线采样算法的系统,如 BlinkDB、SnappyData 等,使用分层采样(stratified sampling)方法来规避单一维度上的偏态分布问题.然而,多维度或者联接查询中的偏态分布问题仍未得到解决;

(3) 离线采样算法同样也无法预估查询延迟,因此也不适用于在线交互式分析系统.

2.1.2 在线采样

在线采样算法的核心思路是:通过采样索引(sampling index),在查询处理阶段生成采样,以近似回答查询.在线采样算法将采样本身作为一个逻辑操作符(logical operator),运用于执行计划和物理实现中.给定一个查询,在线采样算法首先通过采样索引找出查询相关的数据范围,之后,通过采样索引逐一产生数据范围内的采样,并利用采样计算近似结果.在线采样算法的优势在于:

(1) 查询在初始阶段样本量较小时即可返回近似结果与置信区间.随着采样数目的增加,查询结果越来越精确,置信区间也逐渐缩小.在这一过程中,用户在得到满意的查询结果后可随时终止查询,并进行下一轮交互;

(2) 在线采样算法可以很好地应对偏态分布数据.在线采样算法所使用的索引技术可以在查询结果覆盖的数据上直接产生采样,因此,其采样效果不受偏态分布影响.以 SQL 查询 Q1 为例,在线采样算法的索引将会在所有满足 *C*=10 条件的数据中产生随机采样,因此,无论满足条件的数据量是否足够,在线采样算法都能获得稳定的近似效果;

(3) 在线采样算法在查询时间足够的前提下,可以收敛到真实查询结果.

在线采样算法的主要缺陷在于必须针对不同的查询设计特定的采样索引.首先,对于某些复杂查询,如何设计高效的采样索引仍是一个待解决的研究问题.例如:对于连接问题(join),近期,Li 等人在 SIGMOD 2016 的最佳文献[13]中提出了 Wander Join 算法.其通过将多表连接(join)查询建模成图,使用随机游走解决在线查询(online aggregation)问题,对多种聚集函数提供无偏且有置信区间保证的估计.算法的重要特性在于不需要事前获取数据的统计知识,可通过进行随机游走选择最优的查询计划.该工作在 PostgreSQL 数据库上进行了整合,称为 XDB,通过在 TPC-H 测试集上的表现证明,其优于已有的基于 Ripple Join[17,18]的 DBO 系统[19,20],可对 GB 量级数据在秒级回答近似查询,且误差不超过 1%的可信度大于 95%.文献[21]将基于抽样的近似查询估计和聚集的预计算两种方法相结合,在查询质量和响应时间上取得更为灵活的折中.然而,Wander Join 只能解决无循环、无星型多表连接采样,对于多表任意连接的采样索引问题仍有待作更进一步的研究.其次,由于在线采样算法需针对不同查询设计不同采样索引,导致这类算法在嵌入已有数据库系统时的实现难度变大.因此,需要设计更多基于在线采样算法的近似查询算子,才能满足实时交互式分析的需求.

最后,文献[22]首次提出了数据库学习(database learning)的概念,通过每次查询,学习到数据分布的知识,提高后续查询结果的质量.从机器学习的角度,即使不同的查询所覆盖的数据子集不同,每一个查询的结果都在一定程度上反映了当前数据库的特征.因此,每一次历史查询都可能涵盖当前查询的部分信息,从而对当前查询有所帮助.然而,传统数据库对于历史查询的复用能力有限,在一定程度上浪费了数据库对于回答历史查询所耗费的 I/O 与计算能力.究其原因,在于查询复用的条件要求较高:给定一个查询,由于查询必须返回精确结果,导致只

有当历史查询所访问的数据为当前查询的子集时,才能被复用.在实际应用中,很难找到这样匹配的历史查询.然而,在面向近似查询处理(AQP)的数据库中,历史查询有可能发挥更大作用.其原因在于:近似查询只要求返回查询的近似结果,而每次历史查询的结果都对于了解数据库内在的数据分布模型有所帮助.数据库学习(database learning)是近年来兴起的一种在近似查询处理数据库中支持历史查询复用的技术,其将数据库中的数据看成由某种未知但固定的统计分布生成.在最理想的情况下,如果能完整地学习到其内在数据模型,就可以在不访问元数据的情况下回答近似查询.在实际中,即使一个不完全准确的数据模型也对近似查询有所帮助,可以将学习到的数据模型配合在线采样算法得到更为精确的查询结果.随着历史查询结果的积累,数据库对于其数据内在的分布建模越来越清晰,回答查询的效率和准确率也会逐渐提升.文献[23]提出了复用近似查询结果的方法,提高了不常见查询的结果精度.更多相关工作可参见综述文献[24,25].

2.2 非结构化数据查询算法

随着互联网对人们生活的渗入以及数据采集能力的提升,非结构化数据越来越多地存在于各类查询任务中.由于非结构化数据的组织结构与语意在特定情境下具有一定的特征,传统的数据库组织方式与优化策略无法针对这些特征进行优化.为了实现在海量数据与非结构化数据的特点下的实时交互式查询,我们需要依托于非结构化数据的具体特点进行查询算法的设计.在本部分内容中,我们对时空数据、图数据以及包括多媒体数据在内的其他数据分别进行综述.

2.2.1 时空数据交互式查询技术

传统的空间数据库和数据分析系统,如 SpatialHadoop、Hadoop GIS 等都基于磁盘存储,其 I/O 代价导致速度较慢.Xie 等人提出了基于 Spark SQL 的内存空间数据分析系统 SIMBA(spatial in-memory big data analytics)[26,27],以保证低延迟和高扩展性,并支持的空间操作有范围查询、*k*NN 查询和基于距离或 *k*NN 的连接.其基本思路包括:通过两阶段的索引策略支持 RDD 上的空间索引,通过 DJSpark 算法进行基于距离的连接.对 *k*NN 连接,实现了基于 Voronoi 图和 z-Value 的连接算法,并提出了基于 R 树的 RKJSpark 算法.在系统优化方面:通过并发执行多个查询提升吞吐量;通过在逻辑和物理优化器上引入对空间数据的支持,进行基于代价的优化(cost-based optimization).对时空数据的近似查询最早由文献[28,29]提及.文献[30,31]研究了空间数据的近似查询算法,其中,范围查询(range query)的算法研究包括文献[32,33].

Christensen 等人首次提出了支持交互时空(spatial-temporal)数据分析的 STORM[34,35],其基于分布式 MongoDB 数据库建立.其通过在线时空数据抽样(online spatial sampling)和聚集支持实时的交互查询分析,且查询质量保证随查询执行数目而提高.为提高查询效率和可扩展性,系统使用了时间-空间索引技术(ST-indexing),提出了基于 R 树的新数据结构 LS-tree 和 RS-tree.具体来讲,LS-tree 通过采用大小符合等比级数的多个 R 树进行分层抽样,RS-tree 基于抽样缓存、拒绝抽样(rejection sampling)和消极搜索策略.LS-tree 和 RS-tree 都可扩展至外存或混合场景.

2.2.2 图数据交互式查询技术

图数据作为一种表达灵活的结构化数据,是数据管理领域所研究的三大经典结构化数据模型(关系、层次、网状)之一.图中一类重要的查询是相似度查询,如节点相似度查询和子图(近似)匹配.其中,节点相似度查询既是基本类型的查询,也在实际应用中使用得最广泛.已知较为著名的节点相似度度量标准有 SimRank[36]、Personalized PageRank(PPR)[37]、Katz[38]、Jaccard 等,在社交网络分析、推荐系统中应用广泛,但部分指标,如 SimRank 和 PPR,其精确计算的复杂度很高,无法扩展到大规模图.然而,通过采样随机路径等方法,可以设计在查询速度和结果精度取得较好平衡的近似算法,如:文献[39–42]等工作设计了基于采样和路径随机游走的方法计算 SimRank 近似查询;文献[43,44]等工作通过结合随机游走、正向搜索和反向搜索的方法计算单源和 Top-*k* PPR 查询,不但对算法的近似效果有理论保证,而且有较低的时延,是目前最新、实际最有效的方法.

图略图(graph sketch)通过线性映射,将 $O(n^2)$的图信息投影到 $O(n \text{ polylog } n)$空间,并以大概率保持图的结构性质.图略图主要解决图不能放进内存以及图规模大到需以分布式或流(stream)方式输入时图的若干查询问题.该模型被称为半流式模型(semi-streaming model)[45].图略图也可扩展至动态图,即允许边的插入和删除以及滑

动窗口模式.已有的图略图工作通过 Spanner[46,47]、Sparsifier[48−50]、抽样和随机游走路径等技术,可回答图是否连通[51−53]、是否为二分图[53]、是否 k-连通、最短路径、(近似)估计最小生成树的权重[51,53]、计算最大匹配[54−56]等.更多查询的内容可见综述文献[57,58].图略图最新的研究方向包括对现有工作的复杂度或近似比进行改进、有向图的略图技术、在随机流上进行设计和分析(stream ordering)、使用更多或更少的内存空间(如 $O(n)$)设计算法等.

2.2.3 其他非结构化数据交互式查询技术

对于多模态的其他非结构数据,交互式查询系统往往使用组合索引的方式来进行实现,如文献[59]中描述的基于文本、视觉、语义等多重特征进行多维度检索及搜索重排序实现的交互式视频查询工具等.由于超高数据维度的挑战,多媒体数据往往无法使用关系型数据库进行完全查询,而通常使用基于哈希的方法来做近似的最近邻查询.在这类方法中,往往利用机器学习的方式,在哈希的过程中保留原始空间的各种距离,如一对一的距离保持、多点之间距离保持、隐含距离保持以及量化.哈希后生成的高维二值特征再在检索系统中使用.另外,近年来,随着跨模态数据不断增多,学界开始研究如何进行多模态特征的统一表达.如文献[60]提出使用对抗学习的方式建立映射函数,将文本及视觉、语义特征融合至统一特征空间,并使得统一空间的融合特征无法被区分为来自哪一个独立特征,以达到有效融合的效果.

2.3 数据流算法

数据流可看成一个由海量数据组成的数据序列,其有 3 个特点.

1) Single Pass:只允许顺序访问一次数据;
2) Small Space:允许存储的空间非常小,通常为对数级别;
3) Small Time:更新(插入、删除)要求的速度快,通常为对数或者常数级别.

数据流算法的理论研究通常基于 3 个常见模型:(1) Cash-register 模型[61−63],该模型中,数据流中的每个元素是一条记录,不允许删除;(2) Turnstile 模型,该模型允许删除元素;(3) Sliding Window 模型[62,64,65],该模型只考虑数据流中最近的元素.在实践方面,目前已经有大规模数据流系统实现,如 S4[66]和 DSTREAM[67].

数据摘要(data summary)是基于数据流模型的数据结构,可利用亚线性空间的数据结构回答近似查询.当前流行的数据摘要有:

1) 随机采样(random sampling).随机采样是近似查询处理引擎中最常见的数据摘要,其原因在于随机采样的表达方式与元数据完全相同,因此任何基于元数据的查询均可在随机采样上完成.此外,随机采样可避免“维度灾难”问题,其近似效果不会随维度提升和下降.当前的近似查询处理引擎往往使用各类随机采样(如有偏采样、权重采样等)以应对复杂的分析查询.然而相较于其他摘要,随机采样存在难以估计连接查询结果、查询优化困难的缺陷;
2) 直方图(histogram)[68]是另一类近似查询算法常用的数据摘要,其将数据分布通过直方的形式进行近似.绝大部分传统数据库系统都支持使用单一维度的直方图来进行执行策略优化,因此直方图也可在不增加系统负担的前提下用于产生近似查询结果.然而,最优直方图(V-optimal histogram)的计算需利用动态规划算法,非最优直方图的近似效果保证仍有待进一步研究;
3) 频繁项摘要(frequent item summary)[69−74],该摘要可近似返回流中出现次数多的元素;
4) FM Sketch[75],该摘要可估计流中不相同元素的数目;
5) AMS Sketch[76,77],该摘要近似返回连接查询的结果集数量;
6) 小波略图(wavelet sketch)[78−80],该略图记录了数据流频率向量小波变化之后最大的 k 个参数,主要用于刻画数据分布;
7) Count-min Sketch[81],该略图可近似查询频繁项与某范围中元素个数;
8) 分位数摘要(quantile summary)[82−84],中位数的扩展可均匀分割数据并返回关于数据累积分布(CDF)的直方图;

摘要技术也常用于一些复杂的分析查询,如范围查询[81]、Top-k 查询[73]、频繁项集估计[85]、l_p 范数估计[86]、

连接结果数目估计[87−89]、频率矩估计[76]等.关于这些查询的更多内容可参见综述文献[90,91].由于数据摘要具备精度高、空间小等特点,也经常被用于传感器网络、网络内数据融合等应用中.

3 实时交互式分析系统

随着人们对交互查询的深入研究,众多基于传统数据库和数据仓库的系统,以及基于 Hadoop/Spark 平台系统均提供了交互查询的功能.同时,部分系统融合了流处理、内存处理等技术,在交互的基础上实现了对数据的连续性操作,并进一步降低了系统的时延.现将目前有代表性的可交互查询系统总结如下.

3.1 基于大数据处理平台原生支持的实时交互式分析系统

得益于 Hadoop(HDFS 与 MapReduce)、Spark、Storm 等分布式大数据处理平台的发展,基于其原生数据分析能力诞生了一系列支持实时交互式分析的系统,现根据其依托的平台,将应对的数据需求情景、发展历程以及主要特点总结如下.

3.1.1 基于 Hadoop 的系统

Apache Hadoop 是一种专用于批处理的处理框架,其处理功能来自 MapReduce 引擎,但由于 MapReduce 过程中每次操作都需要重新写回 HDFS,严重依赖持久存储上的 I/O,因此速度相对较慢.但另一方面,由于磁盘空间通常是服务器上最丰富的资源,这意味着 MapReduce 可以处理非常海量的数据集,也意味着相比其他类似技术,Hadoop 的 MapReduce 通常可以在廉价硬件上运行.同时,因为该技术并不需要将一切都存储在内存中,具备极高的缩放潜力,生产环境中曾经出现过包含数万个节点的应用.

Apache Hive[92]和 Pig[93]是典型的基于 Hadoop 平台的数据仓库系统,该类系统的典型特征是将 SQL/HiveQL 等类型的查询转化为 MapReduce 任务.Hive 原本是为了系统吞吐量而设计的,因此适合于长时间运行的批处理任务.然而,Hive2 引入了 LLAP(live long and process)机制,即预先启动一组进程,并借助内存为中心的架构,可以实现亚秒级的交互查询.和 Apache Hive 类似,Presto[94]是属于 SQL over Hadoop 的分布式 SQL 查询引擎,是为了交互式分析而设计的.尽管 Presto 的速度很快,但它并不保障容错性.相关类型的工作和讨论可见综述文献[95].不同于 MapReduce 将输出进行物化,MapReduce Online[96]允许数据在操作符间管道式(pipelined)传输,缩短了任务执行时间.基于此实现的 HOP(Hadoop online prototype)系统支持在线聚集,即允许用户看到较早的近似结果.类似地,文献[14]通过 Hyracks 实现了基于 MapReduce 的在线聚集,使用贝叶斯框架对结果和置信区间进行估计.通过对 Hadoop 调度程序的改进,其效果优于 HOP.

3.1.2 基于 Storm 的系统

Apache Storm[97]是一个分布式、容错的实时流处理系统,适用于快速响应、中等流量的场景.Storm 令持续不断的流计算变得容易,弥补了 Hadoop 批处理所不能满足的实时要求的缺陷.Storm 中的核心抽象概念是无边界元组(tuples)的序列,称为流.Storm 经常用于在实时分析、在线机器学习、持续计算、分布式远程调用和 ETL 等领域.Storm 的部署管理非常简单,而且在同类的流式计算工具中,Storm 的性能也是非常出众的.由于 Storm 无法确保可以按照特定顺序处理消息,为了实现严格的一次处理,即有状态处理,可以使用一种名为 Trident 的抽象.Trident 会对 Storm 的处理能力产生极大影响,会增加延迟,为处理提供状态,使用微批模式代替逐项处理的纯粹流处理模式.

作为 Storm 的继承者,由 Twitter 在 2016 年开源的 Heron[98]兼容 Storm 的 API,在功能上基本可以互换,且吞吐量更大、对硬件要求更低.Heron 通过把 Storm 的基于线程的计算模型替换为基于进程的模型,克服了 Storm 在性能以及可靠性方面的缺点,同时与 Storm 的数据模型和拓扑 API 完全兼容.Heron 更适合超大规模的机器集群,在稳定性上有更优异的表现;但是在性能上,表现一般甚至稍弱一些;在资源使用上,可以和其他编程框架共享资源,但 Topology 级别会更浪费一些资源.

3.1.3 基于 Spark 的系统

区别于上述基于 Hadoop 和 Storm 发展的系统,Spark 可运行批处理和流处理,运行一个集群即可处理不同类型的任务.尽管这类系统也是使用 MapReduce 的计算模式,但借助 Apache Spark[99]计算引擎的优势(如内存资

源),此类系统在性能上更佳.在内存计算策略和先进的DAG调度等机制的帮助下,Spark可以用更快的速度处理相同的数据集.

典型工作如Shark(Hive on spark)[100],其设计了一个将RDD融合进SQL查询引擎进行深入数据分析的数据仓库系统.其后续工作Spark SQL[101]进一步融合了关系数据处理的优点,如声明式查询和存储优化.对比MapReduce只能处理离线数据,Spark还能支持实时的流计算,Spark Streaming[102]主要用来对数据进行实时的处理.Spark Streaming通过在Spark批处理计算架构下建立的微批处理(mini batch)引擎,将持续不断输入的数据流转换成多个Batch分片,使用一批Spark应用实例进行处理.为流处理系统采用批处理的方法,需要对进入系统的数据进行缓冲.缓冲机制使得该技术可以处理非常大量的传入数据,提高整体吞吐率,但等待缓冲区清空也会导致延迟增高.这意味着Spark Streaming可能不适合处理对延迟有较高要求的工作负载.通过DataFrame API,该系统将声明式和过程式处理结合起来,并实现了一个高可扩展的优化器Catalyst,可提供代码优化、整合不同类型数据以及定义UDF函数等,将Spark速度提高了10倍以上.另一方面,由于内存通常比磁盘空间更贵,因此相比基于磁盘的系统,Spark成本更高.然而处理速度的提升意味着可以更快速地完成任务,在需要按照小时为资源付费的环境中,这一特性通常可以抵消增加的成本.Spark内存计算这一设计的另一个后果是:如果部署在共享的集群中,可能会遇到资源不足的问题.相比Hadoop MapReduce,Spark的资源消耗更大,可能会对需要在同一时间使用集群的其他任务产生影响.

3.2 基于大数据处理平台引擎优化的实时交互式分析系统

由于大数据处理平台在设计之初主要应对相对宽泛的数据应用场景,其中的计算引擎以及其他模块组件在对实时交互分析的支持上并不具有特别的优势.随着商业领域应用场景的聚合和细分,实时交互分析系统愈发成为商业分析系统中不可或缺,甚至是至关重要的一环.故而,一些为具体数据查询情景设计的支持实时交互式分析的计算引擎应运而生.现针对此间典型系统,就其优化思路、性能评价、系统特点以及局限等方面进行总结.

3.2.1 Dremel

大规模交互性数据分析处理在整个行业中的应用越来越广泛,交互型分析对于数据处理的响应时间要求比较高,而原有的BigTable[103]和PNUTS[104]是支持实时查询的典型分布式系统,并没有考虑对于交互式场景的要求,对于大规模交互数据分析处理响应性不够.针对这个问题,Dremel[1]使用了新的SQL执行引擎,实现了交互地查询.

在大规模交互数据分析中,存在一种典型情景如下:待分析的源数据体量非常大,但是最终结果集数据量会很小,以“聚合汇总型(group by)”的查询任务为代表.Dremel针对上述情景,通过:(1) 构建多层次服务树;(2) 嵌套列式存储的方式,巧妙地利用数据结构的构造,在一定程度上规避了传统上多表连接操作,同时,基于对查询执行的优化,利用廉价的大规模分布式磁盘存储与计算,大幅缩小需要访问的数据量与需要执行的操作量.

具体而言,Dremel通过将原生数据组织成嵌套数据的结构,保留了数据的原始结构,使得每个数据之间的关系得以明确,而不需要依赖于表之间的连接显式地申明结构.如何将嵌套数据与能够进行高效压缩与计算的列式存储结合起来,成为Dremel的关键问题,也成为了其创新点.为了把嵌套数据结构转化为列的格式来储存,Dremel作了有趣的转换工作:嵌套结构中的每一个嵌套层级被当作一列来存储,同时,为了在列存储时能够保留结构信息并用作执行查询时复原原始信息,设计了两个重要参数:一个是重复深度(repetition level),另一个是定义深度(definition level).重复深度和定义深度针对重复字段来计算其所在的嵌套层级以及发生重复的字段,以此确定重复的位置.正是通过这两个参数,Dremel巧妙地结合了嵌套数据结构保留原生信息以及列式存储高效的优势.另一方面,Dremel采用的是多层次服务树架构,最上层是Dremel的根服务器,根服务器接收所有的查询请求,读取数据库相关的元数据,并把相关请求下发到下一级查询服务器查询.中间层查询服务器负责根服务器请求的派发和叶子服务器查询结果的处理.叶子服务器与具体存储层直接通信,完成存储系统上相关数据的读取和查询动作.这样一种将复杂查询拆分成为更为基础、简单的小查询的方式,能够很好地应对中间结果远小于分析所依赖原始数据的情况,能够很好地利用大量节点上的磁盘存储空间与计算能力.Dremel查询语句是基

于 SQL 语法的定制语法,能够在上述嵌套数据结构的列存储模式下高效执行.对于 Dremel 查询树结构,会对于根查询服务器收到的查询请求进行层层拆分,最终传递到叶节点的查询服务器,叶节点查询服务器获取数据结果后进行过滤和汇总这样的计算,然后再上传到上级查询服务器,层层汇总结果后,最终返回结果数据.

由于采用了多层次服务树的查询架构,相比于 MapReduce 和其他计算引擎而言,Dremel 在面对特定事务获得效果增益的同时,也使得其应用场景有了更多的局限性:首先,如上所述,具体查询任务产生的中间结果集(及最后结果)要足够地小,使得单个服务器节点的内存能够容纳,只要查询中间的结果有一个无法被装入节点的内存中,则整个查询将失败.对应地,在具体任务之下,要求我们根据具体任务以及系统部署的具体环境,人工设计服务树的层次与各层次上的节点数,这是一个需要经验的工作.相比之下,MapReduce 没有对结果集的大小加以限制,所以 MapReduce 能处理更大规模的数据.但值得注意的是:在实际的使用中,BI 的结果集绝大多数时候要远小于源数据集,所以 Dremel 的这个限制对于实际 BI 查询并不构成重大影响.但对于非聚合运算,如 Sorting、Dremel 将不适合.另一方面,有些聚合运算不能通过服务树进行分拆后合并完成,如 Top-*K* 等.对于这些运算,Dremel 只能进行估算.

3.2.2 PowerDrill

随着 Dremel 的推出,在对于聚合汇总类的任务上,我们有了比 MapReduce 更为高效的交互式分析系统;但是在某些需要“探索式查询”的情景之下,Dremel 秒级的响应时间也不能很好地满足需求.数据探索(data exploration)的场景是指:用户在查询时可能并不明确知道查询目标及意图,在完成一项任务之前需要先向系统多次发出请求,根据得到的结果来修正查询内容并再次向系统发出新的查询,如此反复的过程可能要进行很多次.值得注意的是,这样的查询在某些特定的业务场景下是可以被优化的.

考虑到 Dremel 仍然将待分析的数据存储在节点的外存上,PowerDrill 将着眼点集中在更为具体和高频的使用情景下,通过:(1) 将常用的数据尽量存储在内存上;(2) 将存储在内存上的数据尽量压缩这两个核心思路,为相对稳定的查询事务提供接近实时的查询效果.

具体而言,PowerDrill 采用双层数据字典作为基本的数据结构,其中维护一个全局字典表,存储全局 ID 和与搜索关键字之间的对应关系,同时维护带有块 ID 和全局 ID 的块数据表,在查询时,通过两层数据映射得到数据的真实值.考虑到全局字典在业务中具有:(1) 有序性;(2) 排序后的数据常常有共同的前缀,PowerDrill 通过前缀树的方式构建全局字典,同时每个全局字典块还会维护一个布隆过滤器(Bloom filter)来快速确定某个值是否在字典中,以提升查询效率.另一方面,PowerDrill 团队在通过对其具体工作环境中所遇到的查询事务进行分析后发现:(1) 绝大多数查询是类似和一致的;(2) 系统中存储的数据表单只有一小部分是经常被使用的.在这两个洞察的指导下,PowerDrill 首先对数据进行分块,然后通过一些方法过滤查询中不需要的数据块来减少数据量.PowerDrill 实际采用的是一种组合范围分区方法:由领域专家确定若干个划分的域,然后依次通过设定阈值,利用这几个域对数据进行划分,当每个块的行数达到这个阈值时就停止划分,否则可以进一步划分.为了压缩分块数据在内存中的大小,以能够在内存中装载更多的数据,PowerDrill 采用列存储的方法,并用 LZO 算法的一个变种对数据进行压缩.值得提出的是,结合之前数据使用情景中“二八定律”的洞察,PowerDrill 针对冷热数据采用歧视的压缩策略,即:在内存中保有压缩和未压缩的数据,根据需要对数据进行压缩和解压缩.在冷热数据切换策略中,比较常用的是 LRU 算法.LRU 是 Least Recently Used 的缩写,即最近最少使用页面置换算法.但是 PowerDrill 开发团队认为直接的 LRU 算法效果还不是很理想,为此,他们采用了一种启发式的缓存策略来代替原始的 LRU 算法.

PowerDrill 在 Google 内部的使用过程中得到了较好的验证,其中,(1) 在查询过程中,平均 92.41%的数据被略去,5.02%的数据会直接被缓存命中,一般仅须扫描 2.66%的数据即可得到查询结果;(2) 超过 70%的查询是不需要从磁盘访问任何数据的,这些查询的平均访问延迟大约是 25s,96.5%的查询需要访问的磁盘量不超过 1GB.这说明 PowerDrill 的数据分块策略是比较成功的.

由于 PowerDrill 在数据分块时采用的组合范围分区方式依赖于具体的业务场景以及相应专家知识,其灵活程度以及向其他业务场景平行迁移的能力欠佳.同时,相比于 Dremel 基于外存的查询执行方式,PowerDrill 需要

在内存中维护与业务相关的、需要加载的数据,这使得当支持业务的数据发生变化而需要加载新数据的时候相对不变,使得在其架构之下无法做到增量更新的数据更新策略.

3.2.3 Apache Impala

Impala[105]是 Cloudera 公司受到第 2.3.1 节中提到的 Google 开发的 Dremel 启发而开发的新型查询系统,借鉴了大规模并行处理数据库(MMP)的思路,它提供 SQL 语义,能够查询存储在 Hadoop 的 HDFS 和 HBase 中的 PB 级大数据.Impala 和 Hive 共享元数据和存储数据,使得 Hive 和 SparkSQL 生成的数据可以在 Impala 里刷新后直接查询.Impala 使用 Parquet 实现了 Dremel 中的列存储,未来还将支持 Hive 并添加字典编码、游程编码等功能.Impala 支持 Hadoop 中大多数格式的文件,使用了全新的执行引擎,通过使用 LLVM 来统一编译运行时代码,避免了为支持通用编译而带来的不必要开销,为每个查询产生汇编级的代码,并在本地内存中运行.Impala 采用 C++实现,做了很多有针对性的硬件优化,并使用了支持数据本地化的 I/O 调度机制,尽可能地将数据和计算分配在同一台机器上进行,降低了网络沟通的成本.

Impala 与 Dremel 采用了类似的多层级服务树的查询架构,故而其也将面临支持查询需求相对有限的问题,对系统部署环境的内存有较高要求.另外,Impala 不支持用户定义函数,一定程度上进一步限制了其使用场景.

3.2.4 Druid

Druid[106]是一个为在大数据集之上作实时统计分析而设计的开源数据存储.Druid 集合了一个面向列存储的层,一个分布式、Shared-nothing 的架构和一个高级的索引结构,通过设计倒排索引、巧妙的索引结构布局和高效的压缩、BitMap 以及布尔查询的限定,可以在实时大数据和历史大数据上实现亚秒级的查询,支持实时导入数据,但对 SQL 支持较弱.

具体而言,Druid 在数据导入阶段会进行数据聚合,将相同维度组合的数据进行聚合处理,并采用倒排索引和基于字典编码的列存储方式,将某一个维度的值,按照字典顺序编码成整数,成为支持其查询的基本数据.Druid 索引结构由字典、正排(列存储)以及倒排组成,其中,倒排表采用压缩的位图索引,而位图索引的优势是能支持快速的布尔查询.从操作流程上看,Druid 通过 RealTime 模块采用 LSM-tree 的模型,采用推(push)或者拉(pull)的方式获取流式数据,数据首先添加到内存的增量索引中,内存增量索引采用 SkipListMap.当达到一定阈值后,采用异步线程将内存增量索引转成倒排索引持久化写入到磁盘中,同时生成一个新的内存增量索引接收数据.如此循环往复,当写入磁盘的持久化索引达到设定的分区阈值时,Druid 会将磁盘内的所有持久化的索引转成一个分片,并且推送到存储层中,成为历史数据.值得注意的是:在这些数据成为历史数据被写入存储层之前,对于它们的查询需求由 RealTime 模块来维护.另一方面,Druid 通过 Historical 采用 MMap 的方式加载存储在存储层上的分片数据,并负责来自路由的对这些分片的查询.Historical 模块采用多线程的方式处理高并发,比如一个请求过来涉及多个分片的查询,那么会为每个分片分配一个线程,并发地执行,然后汇总结果返回给路由.注意到:如果索引的总大小小于内存大小,那么 Druid 则变成了内存数据库,瓶颈不在 I/O,也不是在 CPU,索引的压缩以及数据的聚合都很消耗 CPU.

值得注意的是:由于 Historical 处理聚合查询时需要大量的内存存放中间临时结果,故而对系统部署的环境的内存提出了要求.

3.2.5 其 他

此外,交互数据探索与交互查询密切相关[107,108].交互探索的原型系统可见 YmalDB[109]、DICE[110]、Charles[111]、AIDE[112]、SnapToQuery[113]等.Querium[114]、Polaris[115]、VizDeck[116]等工作,它们提出了便于可视化的探索式搜索系统,相关综述参见文献[117].

3.3 基于近似查询处理的交互式分析系统

由于交互精确查询系统面对海量数据存在响应时间慢等问题,近年来,利用近似查询处理(approximate query processing)解决实时交互式分析响应时间问题的工作逐渐成为一个重要的研究趋势.这类系统通过提供近似但有质量保障的查询结果,通过抽样、摘要等技术大幅降低查询延时,提高了系统的扩展性,其系统架构如图 4 所示.

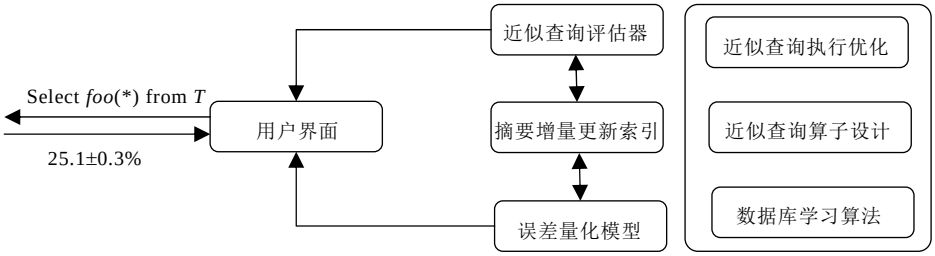

Fig.4 Architecture of AQP-based interactive analysis system

图 4 基于近似查询的交互式分析系统构架

查询算子的执行策略优化(query plan optimization)是数据库领域数十年的重要研究课题.近年来,随着近似查询处理的兴起,对于近似查询算子的优化策略越来越受到关注.传统数据库查询的优化策略受限于其精确性的要求,必须在所有等价的执行策略中选取,因此其重点放在估计查询结果大小、调整算子执行顺序等有限的优化措施上.而近似查询由于在精度上做出放松,允许数据库选取与原查询不完全等价的执行策略,限制条件更为宽泛,而相应最优策略的选取难度也大幅提升.因此,近似查询的优化策略不能套用传统数据库,需要考虑到近似查询的特点.近似查询优化策略研究中的一个重要课题是采样操作与查询算子的可交换性:对于某类查询算子,先在数据上执行采样操作,然后在采样上执行查询,与先执行查询算子然后在查询相关的数据上执行采样,两者近似效果一致.例如:对于键-外键(key-foreign key)双表连接操作,传统的均匀采样是不能与连接操作交换的,其原因在于先采样算法对于偏态分布数据无法给出置信区间和置信度.然而,如果采用异键采样(distinct sampling),该采样方法将对表中每个不同键值均等看待,就去掉了偏态分布的影响,因此异键采样与键-外键连接具有可交换性.设计判断采样操作与查询算子的可交换性的策略,可以为近似查询算子执行优化与传统数据库执行策略优化搭建桥梁.其次,传统数据库查询的执行优化主要考虑其核心资源是可分配的 I/O 代价(即等价于查询时间).在近似查询算子的执行过程中,还应将误差界看成一种可以调配的资源策略优化.因此,传统数据库查询在执行查询代价预估时,一个执行计划对应于一个查询时间;而近似查询的代价预估则应为一条描述执行代价-误差界权衡(query cost-error tradeoff)的曲线.在查询的执行过程中,执行计划对权衡曲线进行修正,并从剩余可行计划中选取允许误差界内代价最低的计划.此外,目前的近似查询处理引擎通常通过置信区间与置信度来衡量查询的误差界.最后,引入近似查询之后,数据库也可以通过历史查询结果来优化当前数据库查询.

AQUA(approximate query answering)[118]系统通过预计算摘要(summary)并存储在 DBMS 中,快速地为聚集查询提供近似结果.STRAT[119]将查询视为两阶段活动的查询模板,并对第 2 阶段进行优化,以支持实时(秒级)的各种类型查询.DBO[19,20]系统实现了 Ripple 连接算法,支持在线聚集(online aggregation)操作.XDB(approXimate DB)[13]基于 PostgreSQL 实现了 Wander Join 算法,支持在线聚集,通过连接图上随机游走的查询优化,其效果优于基于 Ripple 连接的系统.此类系统还包括 SciBORQ[120]等.

BlinkDB[121]是一个基于 Apache Spark 的、用于海量数据上交互 SQL 查询的并行计算引擎,它允许用户通过权衡数据精度来提升查询响应时间.其查询逻辑包括两个核心思想:一是通过假设可预测的查询列集合(predictable query column sets),提出可适应的列存储优化框架来建立和维护一组多维的分层抽样(stratified sample);二是根据查询精度和响应时间要求,通过在多个小规模子抽样上评估并建立错误-延迟画像(error latency profile,简称 ELP),据此选择一个合适大小的抽样进行查询估计.根据上述算法框架,其实现了一个建立在 Hive/Shark 上的基于分层抽样的近似查询处理系统,接受 SQL/HiveQL 式查询语句.系统可在 TB 级数据上以秒级时间返回有置信区间保证的近似查询,其错误率低于 2%~10%.STORM[34]使用在线时空数据抽样对时空数据(spatial-temporal data)支持实时的交互查询分析.Simba[26]是一个基于 Apache Spark 的内存空间数据查询计算引擎,创新性地在 Apache Spark 中引入了索引机制,为范围查询、*k*NN 等空间操作提供可交互的查询接口.

Mozafari 等人首次提出了结合分析处理、事务处理和流数据处理的近似查询处理系统 SnappyData[122].其

混合型查询引擎通过融合 Apache Spark 和 Apache GemFire 建立,并提供了统一的 API,称为 SnappyContext.其采用基于行和列存储的混合存储引擎,并支持“概率存储(probabilistic store)”,如带时间戳的分层抽样、摘要等;通过应用间的状态共享,实现最小序列化;通过低延时的错误发现、应用和数据服务器的分离策略提供高可用性;通过绕过调度器插入细粒度和长时间运行的任务以降低时延或缓存数据,并保证事务一致性.基于数据库学习概念的 Verdict[123]系统通过最大熵估计近似结果,精度较基于抽样的 AQP 系统提高了 20 倍以上.INCAPPROX[124]基于 Spark SQL,使用抽样和 Memorization 技术首次实现了流数据的近似查询和增量计算.文献[125]综述了构建近似查询系统的常用技术,文献[126]总结了目前交互近似查询系统面临的实际挑战和研究方向.

4 总结与展望

实时交互式分析已经引起了国内学术界与工业界的广泛关注,但国内研究机构对于该方向的相关研究尚处于起步阶段.中国人民大学、北京大学、清华大学、上海交通大学、电子科技大学、哈尔滨工业大学、武汉大学、中山大学、西北工业大学、华东师范大学、中国科学院计算技术研究所等在实时交互式分析的相关领域内都有学者从事研究.文献[127,128]分别对现有探索式搜索系统与交互式探索系统作了详细综述,文献[129]对内存集群的交互式分析系统作了总结.总体而言,国内针对实时交互式分析的研究工作刚刚起步,对实时交互式分析的基础理论、计算框架的研究还有待完善,对可用实时交互式分析系统的需求也较为急迫.本文认为:对实时交互式分析的研究面临诸多挑战,主要源于该问题的数据复杂性、计算复杂性和系统复杂性这 3 个关键性科学问题.

4.1 数据复杂性:如何建立支持实时交互式分析的跨模态数据模型?

实时交互式分析要求对跨模态高维度数据具备处理与分析能力.相对于传统的单模态数据,跨模态数据往往同时包含异构(半结构化和非结构化共存)和异质(不同质量的数据共存)的数据类型,在数据维度、数据分布和数据特征上也变得更加复杂和多样化.这使得传统大数据分析框架下针对单一模态数据的索引结构和查询优化策略无法继续有效地工作.目前,主流的实时交互数据库系统主要针对结构化数据库,缺乏对于文本数据、时空数据、多媒体数据、图数据等非结构化数据的支持,无法对跨模态数据做出有效分析.首先,缺乏针对不同的数据模态组合设计有效的特征表达;其次,缺乏跨模态数据的异构表达建立有效的索引组合,缺乏对于非结构化数据的近似查询处理引擎;再次,由于跨模态数据的表征多样性,缺乏针对跨模态高维数据的直观、高效的交互查询方法.因此,对于跨模态数据的恰当表征、高效查询和直观交互是进行实时交互式分析的先决条件.

4.2 计算复杂性:如何在交互级响应时间内支持高计算复杂度的大数据实时分析?

传统数据库要求精确回答查询,数据库需要访问所有与查询相关的数据.实时交互式分析对于系统响应时间有严格要求,而对于查询精度往往可以适当放宽.随着数据量的扩张,精确查询的响应时间最终将无法满足交互级响应时间的要求.近年来,使用支持近似查询的数据库引擎来支持实时交互式分析,是学术界与工业界的共识.然而,目前的近似查询处理引擎与大规模的商业应用之间仍有一定差距,其原因在于,近似查询处理的理论研究仍存在以下缺陷:首先,缺乏支持复杂 SQL 查询在线采样算法,从而无法对现有数据库查询算子提供可控精度保证,不能实现由用户终止查询的实时交互;其次,对于涉及多算子的复杂查询,缺乏针对近似查询算子优化的执行策略,只能套用传统数据库的优化策略,导致优化效率底下,无法针对所有传统数据库查询提供预估精度保证;再次,缺乏针对复杂分析查询的数据流算法,限制了现有近似查询处理引擎的增量更新能力,无法对分析结果提供实时性保证.因此,设计基于在线采样、增量更新的近似查询算子以及优化策略,是实现实时交互近似分析的理论基础.

4.3 系统复杂性:如何设计稳定、高效、易用、可扩展的实时交互式分析系统?

实时交互式分析对当前的大数据处理系统也提出了新的挑战.现有针对实时交互式分析的数据库系统存在以下缺陷.

- 首先,缺乏支持近似查询的高效可扩展的分布式混合索引.高效的分布式近似查询索引是实时交互式分析系统的关键:一方面,需要尽可能地减少数据的网络传输,实现快速检索;另一方面,需要设计支持多种混合数据的索引结构,并支持大数据的增量更新;
- 其次,缺乏支持流-批结合混合应用的高效、稳定的调度方案.大数据平台中同时存在流处理和批处理混合的应用,为了提高应用性能及资源利用率,需要综合考虑应用需求、应用特性,在运行时进行高效的任务调度,确保实时交互式分析任务的快速响应和批处理任务的高吞吐量以及系统的稳定性;
- 再次,缺乏自适应的交互计算模式.交互实时应用在提高响应速度时,要实现查询精度、效率的平衡,从而自适应地满足多种查询分析场景.此外,从应用接口层面支持丰富的查询方法和系统层面的扩展.

因此,最终实现稳定、高效、易用、可扩展的实时交互式分析系统,是该领域发展的核心目标.

References:

[1] Melnik S, Gubarev A, Long JJ, Romer G, Shivakumar S, Tolton M, Vassilakis T. Dremel: Interactive analysis of Web-scale datasets. Communications of the ACM, 2011,54(6):114−123. [doi: 10.1145/1953122.1953148]

[2] Marchionini G. Exploratory search: From finding to understanding. Communications of the ACM, 2006,49(4):41−46. [doi: 10.1145/1121949.1121979]

[3] Miller RB. Response time in man-computer conversational transactions. In: Proc. of the Fall Joint Computer Conf. Part I. New York: ACM Press, 1968. 267−277. [doi: 10.1145/1476589.1476628]

[4] Liu ZC, Heer J. The effects of interactive latency on exploratory visual analysis. IEEE Trans. on Vis Comput Graph, 2014,20(12): 2122−2131.

[5] Aggarwal A, Vitter JS. The input/output complexity of sorting and related problems. Communications of the ACM, 1988,31(9): 1116−1127. [doi: 10.1145/48529.48535]

[6] Frigo M, Leiserson CE, Prokop H, Ramachandran S. Cache-oblivious algorithms. In: Proc. of the 40th Annual Symp. on Foundations of Computer Science. Washington: IEEE Computer Society, 1999. 285−297. [doi: 10.1109/ SFFCS.1999.814600]

[7] Henzinger MR, Raghavan P, Rajagopalan S. Computing on data streams. SRC Technical Note, 1998-011, Boston: American Mathematical Society, 1998. 107−118.

[8] Feigenbaum J, Kannan S, Strauss MJ, Viswanathan M. An approximate L_1-difference algorithm for massive data streams. SIAM Journal on Computing, 2003,32(1):131−151. [doi: 10.1137/S0097539799361701]

[9] Valiant LG. A bridging model for parallel computation. Communications of the ACM, 1990,33(8):103−111. [doi: 10.1145/79173.79181]

[10] Dean J, Ghemawat S. MapReduce: Simplified data processing on large clusters. Communications of the ACM, 2008,51(1):107−113. [doi: 10.1145/1327452.1327492]

[11] Lohr SL. Sampling: Design and Analysis. 2nd ed., San Francisco: CENGAGE Learning, 2010.

[12] Cormode G, Garofalakis M, Haas PJ, Jermaine C. Synopses for massive data: Samples, histograms, wavelets, sketches. Foundations and Trends® in Databases, 2011,4(1-3):1−294. [doi: 10.1561/1900000004]

[13] Li FF, Wu B, Yi K, Zhao ZY. Wander join and XDB: Online aggregation via random walks. SIGMOD Record, 2017,46(1):33−40. [doi: 10.1145/3093754.3093763]

[14] Pansare N, Borkar VR, Jermaine C, Condie T. Online aggregation for large MapReduce jobs. PVLDB, 2011,4(11):1135−1145.

[15] Acharya S, Gibbons PB, Poosala V. Congressional samples for approximate answering of group-by queries. SIGMOD Record, 2000,29(2):487−498. [doi: 10.1145/335191.335450]

[16] Agarwal S, Mozafari B, Panda A, Milner H, Madden S, Stoica I. BlinkDB: Queries with bounded errors and bounded response times on very large data. In: Proc. of the 8th ACM European Conf. on Computer Systems. New York: ACM Press, 2013. 29−42. [doi: 10.1145/2465351.2465355]

[17] Hellerstein JM, Haas PJ, Wang HJ. Online aggregation. SIGMOD Record, 1997,26(2):171−182.

[18] Haas PJ, Hellerstein JM. Ripple joins for online aggregation. SIGMOD Record, 1999,28(2):287−298. [doi: 10.1145/304181.304208]

[19] Dobra A, Jermaine C, Rusu F, Xu F. Turbo-charging estimate convergence in DBO. Proc. of the VLDB Endowment, 2009,2(1): 419−430. [doi: 10.14778/1687627.1687675]

[20] Jermaine C, Arumugam S, Pol A, Dobra A. Scalable approximate query processing with the DBO engine. ACM Trans. on Database Systems, 2008,33(4):23:1−23:54. [doi: 10.1145/1412331.1412335]

[21] Peng JL, Zhang DX, Wang JN, Pei J. AQP++: Connecting approximate query processing with aggregate precomputation for interactive analytics. In: Proc. of the 2018 Int'l Conf. on Management of Data. New York: ACM Press, 2018. 1477−1492. [doi: 10.1145/3183713.3183747]

[22] Park Y, Tajik AS, Cafarella M, Mozafari B. Database learning: Toward a database that becomes smarter every time. In: Proc. of the 2017 ACM Int'l Conf. on Management of Data. New York: ACM Press, 2017. 587−602. [doi: 10.1145/3035918.3064013]

[23] Galakatos A, Crotty A, Zgraggen E, Binnig C, Kraska T. Revisiting reuse for approximate query processing. Proc. of the VLDB Endow., 2017,10(10):1142−1153. [doi: 10.14778/3115404.3115418]

[24] Garofalakis MN, Gibbon PB. Approximate query processing: Taming the TeraBytes. In: Proc. of the 27th Int'l Conf. on Very Large Data Bases. San Francisco: Morgan Kaufmann Publishers Inc., 2001. 725.

[25] Chaudhuri S, Ding B, Kandula S. Approximate query processing: No silver bullet. In: Proc. of the 2017 ACM Int'l Conf. on Management of Data. New York: ACM Press, 2017. 511−519. [doi: 10.1145/3035918.3056097]

[26] Xie D, Li FF, Yao B, Li GF, Zhou L, Guo MY. Simba: Efficient in-memory spatial analytics. In: Proc. of the 2016 Int'l Conf. on Management of Data. New York: ACM Press, 2016. 1071−1085. [doi: 10.1145/2882903.2915237]

[27] Xie D, Li FF, Yao B, Li GF, Chen ZP, Zhou L, Guo MY. Simba: Spatial in-memory big data analysis. In: Proc. of the 24th ACM SIGSPATIAL Int'l Conf. on Advances in Geographic Information Systems. New York: ACM Press, 2016. 86:1−86:4. [doi: 10.1145/2996913.2996935]

[28] Olken F. Random sampling from database [Ph.D. Thesis]. Berkeley: University of California at Berkeley, 1993.

[29] Olken F, Rotem D. Sampling from spatial databases. In: Proc. of the 9th Int'l Conf. on Data Engineering. Washington: IEEE Computer Society, 1993. 199−208.

[30] Azevedo LG, Zimbrão1 G, de Souza JM. Approximate query processing in spatial databases using raster signatures. In: Proc. of the Advances in Geoinformatics: VIII Brazilian Symp. on GeoInformatics (GEOINFO 2006). Berlin, Heidelberg: Springer-Verlag, 2006. 69−86. [doi: 10.1007/978-3-540-73414-7_5]

[31] Belussi A, Catania B, Migliorini S. Approximate queries for spatial data. In: Proc. of the Advanced Query Processing—Volume 1: Issues and Trends. Berlin, Heidelberg: Springer-Verlag, 2013. 83−127. [doi: 10.1007/978-3-642-28323-9_5]

[32] Joshi S, Jermaine C. Materialized sample views for database approximation. In: Proc. of the 22nd Int'l Conf. on Data Engineering. Washington: IEEE Computer Society, 2006. 151−165. [doi: 10.1109/ICDE.2006.90]

[33] Hu XC, Qiao M, Tao YF. Independent range sampling. In: Proc. of the 33rd ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems. New York: ACM Press, 2014. 246−255. [doi: 10.1145/2594538.2594545]

[34] Christensen R, Wang L, Li FF, Yi K, Tang J, Villa N. STORM: Spatio-temporal online reasoning and management of large spatio-temporal data. In: Proc. of the 2015 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2015. 1111−1116. [doi: 10.1145/2723372.2735373]

[35] Wang L, Christensen R, Li FF, Yi K. Spatial online sampling and aggregation. Proc. of the VLDB Endowment, 2015,9(3):84−95. [doi: 10.14778/2850583.2850584]

[36] Jeh G, Widom J. SimRank: A measure of structural-context similarity. In: Proc. of the 8th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. New York: ACM Press, 2002. 538−543. [doi: 10.1145/775047.775126]

[37] Page L, Brin S, Motwani R, Winograd T. The PageRank citation ranking: Bringing order to the Web. Technical Report, Stanford: Stanford InfoLab, 1999. 1−17.

[38] Katz L. A new status index derived from sociometric analysis. Psychometrika, 1953,18(1):39−43. [doi: 10.1007/BF02289026]

[39] Fogaras D, Racz B. Scaling link-based similarity search. In: Proc. of the 14th Int'l Conf. on World Wide Web. New York: ACM Press, 2005. 641−650. [doi: 10.1145/1060745.1060839]

[40] Tian BY, Xiao XK. SLING: A near-optimal index structure for SimRank. In: Proc. of the 2016 Int'l Conf. on Management of Data. New York: ACM Press, 2016. 1859−1874. [doi: 10.1145/2882903.2915243]

[41] Liu Y, Zheng BL, He XD, Wei ZW, Xiao XK, Zheng K, Lu JH. Probesim: Scalable single-source and top-*k* simrank computations on dynamic graphs. Proc. of the VLDB Endowment, 2017,11(1):14−26. [doi: 10.14778/3151113.3151115]

[42] Luo XC, Gao J, Zhou C, Yu X. UniWalk: Unidirectional random walk based scalable simrank computation over large graph. In: Proc. of the 33rd Int'l Conf. on Data Engineering. Washington: IEEE Computer Society, 2017. 325−336. [doi: 10.1109/ICDE.2017.92]

[43] Lofgren P, Banerjee S, Goel A. Personalized PageRank estimation and search: A bidirectional approach. In: Proc. of the 9th ACM Int'l Conf. on Web Search and Data Mining. New York: ACM Press, 2016. 163−172. [doi: 10.1145/2835776.2835823]

[44] Wang SB, Yang RC, Xiao XK, Wei ZW, Yang Y. FORA: Simple and effective approximate single-source personalized PageRank. In: Proc. of the 23rd ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. New York: ACM Press, 2017. 505−514. [doi: 10.1145/3097983.3098072]

[45] Feigenbaum J, Kannan S, McGregor A, Suri S, Zhang J. On graph problems in a semi-streaming model. Theoretical Computer Science, 2005,348(2):207−216. https://doi.org/10.1016/j.tcs.2005.09.013

[46] Baswana S. Streaming algorithm for graph spanners—Single pass and constant processing time per edge. Information Processing Letters, 2008,106(1):110−114. [doi: 10.1016/j.ipl.2007.11.001]

[47] Elkin M. Streaming and fully dynamic centralized algorithms for constructing and maintaining sparse spanners. ACM Trans. on Algorithms, 2011,7(2):20:1−20:17. [doi: 10.1145/1921659.1921666]

[48] Ahn KJ, Guha S. Graph sparsification in the semi-streaming model. In: Proc. of the 36th Int'l Collogquium on Automata, Languages and Programming: Part II. Berlin, Heidelberg: Springer-Verlag, 2009. 328−338. [doi: 10.1007/978-3-642-02930-1_27]

[49] Ahn KJ, Guha S, McGregor A. Graph sketches: Sparsification, spanners, and subgraphs. In: Proc. of the 31st ACM SIGMOD-SIGACT-SIGAI Symp. on Principles of Database Systems. New York: ACM Press, 2012. 5−14. [doi: 10.1145/2213556.2213560]

[50] Goel A, Kapralov M, Post I. Single pass sparsification in the streaming model with edge deletions. arXiv:1203.4900, 2012.

[51] Feigenbaum J, Kannan S, McGregor A, Suri S, Zhang J. On graph problems in a semi-streaming model. In: Proc. of the Automata, Languages and Programming. Berlin, Heidelberg: Springer-Verlag, 2004. 531−543. [doi: 10.1007/978-3-540-27836-8_46]

[52] Ahn KJ, Guha S, McGregor A. Analyzing graph structure via linear measurements. In: Proc. of the 23rd Annual ACM-SIAM Symp. on Discrete Algorithms. Philadelphia: Society for Industrial and Applied Mathematics, 2012. 459−467.

[53] Crouch MS, McGregor A, Stubbs D. Dynamic graphs in the sliding-window model. In: Proc. of the Algorithms—ESA 2013. Berlin, Heidelberg: Springer-Verlag, 2013. 337−348. [doi: 10.1007/978-3-642-40450-4_29]

[54] Kapralov M. Better bounds for matchings in the streaming model. In: Proc. of the 24th Annual ACM-SIAM Symp. on Discrete Algorithms. Philadelphia: Society for Industrial and Applied Mathematics, 2013. 1679−1697.

[55] Ahn KJ, Guha S. Access to data and number of iterations: Dual primal algorithms for maximum matching under resource constraints. In: Proc. of the 27th ACM Symp. on Parallelism in Algorithms and Architectures. New York: ACM Press, 2015. 202−211. [doi: 10.1145/2755573.2755586]

[56] Epstein L, Levin A, Mestre J, Segev D. Improved approximation guarantees for weighted matching in the semi-streaming model. SIAM Journal on Discrete Mathematics, 2011,25(3):1251−1265. [doi: 10.1137/100801901]

[57] McGregor A. Graph stream algorithms: A survey. SIGMOD Record, 2014,43(1):9−20. [doi: 10.1145/2627692.2627694]

[58] Abello J, Finocchi I, Korn J. Graph sketches. In: Proc. of the IEEE Symp. on Information Visualization. Washington: IEEE Computer Society, 2001. 67−70. [doi: 10.1109/INFVIS.2001.963282]

[59] Gao LL, Song JK, Liu XY, Shao JM, Liu JJ, Shao J. Learning in high-dimensional multimedia data: The state of the art. Multimedia Systems, 2017,23(3):303−313. [doi: 10.1007/s00530-015-0494-1]

[60] Wang BK, Yang Y, Xu X, Hanjalic A, Shen HT. Adversarial cross-modal retrieval. In: Proc. of the 25th ACM Int'l Conf. on Multimedia. New York: ACM Press, 2017. 154−162. [doi: 10.1145/3123266.3123326]

[61] Beyer K, Haas PJ, Reinwald B, Sismanis Y, Gemulla R. On synopses for distinct-value estimation under multiset operations. In: Proc. of the 2007 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2007. 199−210. [doi: 10.1145/1247480.1247504]

[62] Datar M, Gionis A, Indyk P, Motwani R. Maintaining stream statistics over sliding windows. SIAM Journal on Computing, 2002, 31(6):1794−1813. [doi: 10.1137/S0097539701398363]

[63] Kane DM, Nelson J, Woodruff DP. An optimal algorithm for the distinct elements problem. In: Proc. of the 29th ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems. New York: ACM Press, 2010. 41−52. [doi: 10.1145/1807085.1807094]

[64] Arasu A, Manku GS. Approximate counts and quantiles over sliding windows. In: Proc. of the 23rd ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems. New York: ACM Press, 2004. 286−296. [doi: 10.1145/1055558.1055598]

[65] Braverman V, Ostrovsky R, Zaniolo C. Optimal sampling from sliding windows. In: Proc. of the 28th ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems. New York: ACM Press, 2009. 147−156. [doi: 10.1145/1559795.1559818]

[66] Neumeyer L, Robbins B, Nair A, Kesari A. S4: Distributed stream computing platform. In: Proc. of the IEEE Int'l Conf. on Data Mining (ICDM). Washington: IEEE Computer Society, 2010. 170−177. [doi: 10.1109/ICDMW.2010.172]

[67] Zaharia M, Das T, Li HY, Hunter T, Shenker S, Stoica I. Discretized streams: Fault-tolerant streaming computation at scale. In: Proc. of the 24th ACM Symp. on Operating Systems Principles. New York: ACM Press, 2013. 423−438. [doi: 10.1145/2517349.2522737]

[68] Poosala V, Haas PJ, Ioannidis YE, Shekita EJ. Improved histograms for selectivity estimation of range predicates. In: Proc. of the '96 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 1996. 294−305. [doi: 10.1145/233269.233342]

[69] Cormode G, Hadjieleftheriou M. Finding frequent items in data streams. Proc. of the VLDB Endowment, 2008,1(2):1530−1541. [doi: 10.14778/1454159.1454225]

[70] Cormode G, Muthukrishnan S. What's hot and what's not: Tracking most frequent items dynamically. In: Proc. of the 22nd ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems. New York: ACM Press, 2003. 296−306. [doi: 10.1145/773153.773182]

[71] Karp RM, Shenker S, Papadimitriou CH. A simple algorithm for finding frequent elements in streams and bags. ACM Trans. on Database Systems, 2003,28(1):51−55. [doi: 10.1145/762471.762473]

[72] Lee LK, Ting HF. A simpler and more efficient deterministic scheme for finding frequent items over sliding windows. In: Proc. of the 25th ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems. New York: ACM Press, 2006. 290−297. [doi: 10.1145/1142351.1142393]

[73] Metwally A, Agrawal D, Abbadi AE. An integrated efficient solution for computing frequent and top-*k* elements in data streams. ACM Trans. on Database Systems, 2006,31(3):1095−1133. [doi: 10.1145/1166074.1166084]

[74] Zhang LF, Guan Y. Frequency estimation over sliding windows. In: Proc. of the 24th Int'l Conf. on Data Engineering. Washington: IEEE Computer Society, 2008. 1385−1387. [doi: 10.1109/ICDE.2008.4497564]

[75] Estan C, Naughton JF. End-biased samples for join cardinality estimation. In: Proc. of the 22nd Int'l Conf. on Data Engineering. Washington: IEEE Computer Society, 2006. 20. [doi: 10.1109/ICDE.2006.61]

[76] Alon N, Matias Y, Szegedy M. The space complexity of approximating the frequency moments. In: Proc. of the 28th Annual ACM Symp. on Theory of Computing. New York: ACM Press, 1996. 20−29. [doi: 10.1145/237814.237823]

[77] Charikar M, Chen K, Farach-Colton M. Finding frequent items in data streams. In: Proc. of the 29th Int'l Colloquium on Automata, Languages and Programming. Berlin, Heidelberg: Springer-Verlag, 2002. 693−703.

[78] Gilbert AC, Kotidis Y, Muthukrishnan S, Strauss M. Surfing wavelets on streams: One-pass summaries for approximate aggregate queries. In: Proc. of the 27th Int'l Conf. on Very Large Data Bases. San Francisco: Morgan Kaufmann Publishers Inc., 2001. 79−88.

[79] Plattner C, Wapf A, Alonso G. Searching in time. In: Proc. of the 2006 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2006. 754−756. [doi: 10.1145/1142473.1142578]

[80] Shaull R, Shrira L, Xu H. Skippy: A new snapshot indexing method for time travel in the storage manager. In: Proc. of the 2008 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2008. 637−648. [doi: 10.1145/1376616.1376681]

[81] Cormode G, Muthukrishnan S. An improved data stream summary: The count-min sketch and its applications. Journal of Algorithms, 2005,55(1):58−75. [doi: 10.1016/j.jalgor.2003.12.001]

[82] Greenwald M, Khanna S. Space-efficient online computation of quantile summaries. In: Proc. of the 2001 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2001. 58−66. [doi: 10.1145/375663.375670]

[83] Guha S, McGregor A. Stream order and order statistics: Quantile estimation in random-order streams. SIAM Journal on Computing, 2009,38(5):2044−2059. [doi: 10.1137/07069328X]

[84] Tao YF, Yi K, Sheng C, Pei J, Li FF. Logging every footstep: Quantile summaries for the entire history. In: Proc. of the 2010 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2010. 639−650. [doi: 10.1145/1807167.1807237]

[85] Yu X, Chong ZH, Lu HJ, Zhou AY. False positive or false negative: Mining frequent itemsets from high speed transactional data streams. In: Proc. of the 30th Int'l Conf. on Very Large Data Bases, Vol.30. San Francisco: VLDB Endowment, 2004. 204−215.

[86] Dobra A, Garofalakis M, Gehrke J, Rastogi R. Processing complex aggregate queries over data streams. In: Proc. of the 2002 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2002. 61−72. [doi: 10.1145/564691.564699]

[87] Alon N, Gibbons PB, Matias Y, Szegedy M. Tracking join and self-join sizes in limited storage. In: Proc. of the 18th ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems. New York: ACM Press, 1999. 10−20. [doi: 10.1145/303976.303978]

[88] Rusu F, Dobra A. Statistical analysis of sketch estimators. In: Proc. of the 2007 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2007. 187−198. [doi: 10.1145/1247480.1247503]

[89] Rusu F, Dobra A. Sketching sampled data streams. In: Proc. of the 25th Int'l Conf. on Data Engineering. Washington: IEEE Computer Society, 2009. 381−392. [doi: 10.1109/ICDE.2009.31]

[90] Babcock B, Babu S, Datar M, Motwani R, Widom J. Models and issues in data stream systems. In: Proc. of the 21st ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems. New York: ACM Press, 2002. 1−16. [doi: 10.1145/543613.543615]

[91] Muthukrishnan S. Data streams: Algorithms and applications. Foundations and Trends® in Theoretical Computer Science, 2005, 1(2):117−236. [doi: 10.1561/0400000002]

[92] Thusoo A, Sarma JS, Jain N, Shao Z, Chakka P, Anthony S, Liu H, Wyckoff P, Murthy R. Hive: A warehousing solution over a map-reduce framework. Proc. of the VLDB Endowment, 2009,2(2):1626−1629. [doi: 10.14778/1687553.1687609]

[93] Olston C, Reed B, Srivastava U, Kumar R, Tomkins A. Pig Latin: A not-so-foreign language for data processing. In: Proc. of the 2008 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2008. 1099−1110. [doi: 10.1145/1376616.1376726]

[94] Sadayuki F. Presto. https://github.com/prestodb/presto

[95] Wang JD, Zhang T, Song JK, Sebe N, Shen HT. A survey on learning to Hash. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2016,40(1):769−790.

[96] Condie T, Conway N, Alvaro P, Hellerstein JM, Elmeleegy K, Sears R. MapReduce online. In: Proc. of the 7th USENIX Conf. on Networked Systems Design and Implementation. Berkeley: USENIX Association, 2010. 21.

[97] Toshniwal A, Taneja S, Shukla A, Ramasamy K, Patel JM, Kulkarni S, Jackson J, Gade K, Fu M, Donham J, Bhagat N, Mittal S, Ryaboy D. Storm@Twitter. In: Proc. of the 2014 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2014. 147−156. [doi: 10.1145/2588555.2595641]

[98] Kulkarni S, Bhagat N, Fu M, Kedigehalli V, Kellogg C, Mittal S, Patel JM, Ramasamy K, Taneja S. Twitter heron: Stream processing at scale. In: Proc. of the 2015 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2015. 239−250. [doi: 10.1145/2723372.2742788]

[99] Zaharia M, Chowdhury M, Franklin MJ, Shenker S, Stoica I. Spark: Cluster computing with working sets. In: Proc. of the 2nd USENIX Conf. on Hot Topics in Cloud Computing. Berkeley: USENIX Association, 2010. 10.

[100] Engle C, Lupher A, Xin R, Zaharia M, Franklin MJ, Shenker S, Stoica I. Shark: Fast data analysis using coarse-grained distributed memory. In: Proc. of the 2012 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2012. 689−692. [doi: 10.1145/2213836.2213934]

[101] Armbrust M, Xin RS, Lian C, Huai Y, Liu D, Bradley JK, Meng X, Kaftan T, Franklin MJ, Ghodsi A, Zaharia M. Spark SQL: Relational data processing in spark. In: Proc. of the 2015 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2015. 1383−1394. [doi: 10.1145/2723372.2742797]

[102] Spark. https://spark.apache.org/streaming/

[103] Chang F, Dean J, Ghemawat S, Hsieh WC, Wallach DA, Burrows M, Chandra T, Fikes A, Gruber RE. Bigtable: A distributed storage system for structured data. ACM Trans. on Computer Systems, 2008,26(2):4:1−4:26. [doi: 10.1145/1365815.1365816]

[104] Cooper BF, Ramakrishnan R, Srivastava U, Silberstein A, Bohannon P, Jacobsen H, Puz N, Weaver D, Yerneni R. PNUTS: Yahoo!'s hosted data serving platform. Proc. of the VLDB Endowment, 2008,1(2):1277−1288. [doi: 10.14778/1454159.1454167]

[105] Kornacker M, Behm A, Bittorf V, Bobrovytsky T, Choi A, Erickson J, Grund M, Hecht D, Jacobs M, Joshi I, Kuff L, Kumar D, Leblang A, Li N, Robinson H, Rorke D, Rus S, Russell J, Tsirogiannis D, Wanderman-milne S, Yoder M. Impala: A modern, open-source SQL engine for Hadoop. In: Proc. of the 2015 Biennial Conf. on Innovative Data Systems Research. 2015.

[106] Yang FJ, Tschetter E, Léauté X, Ray N, Merlino G, Ganguli D. Druid: A real-time analytical data store. In: Proc. of the 2014 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2014. 157−168. [doi: 10.1145/2588555. 2595631]

[107] Idreos S, Papaemmanouil O, Chaudhuri S. Overview of data exploration techniques. In: Proc. of the 2015 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2015. 277−281. [doi: 10.1145/2723372.2731084]

[108] Roy SB, Stefanidis K, Koutrika G, Lakshmanan LV, Riedewald M. Report on the 3rd Int'l workshop on exploratory search in databases and the Web (ExploreDB 2016). SIGMOD Record, 2016,45(3):35−38. [doi: 10.1145/3022860.3022867]

[109] Drosou M, Pitoura E. YmalDB: A result-driven recommendation system for databases. In: Proc. of the 16th Int'l Conf. on Extending Database Technology. New York: ACM Press, 2013. 725−728. [doi: 10.1145/2452376.2452464]

[110] Kamat N, Jayachandran P, Tunga K, Nandi A. Distributed and interactive cube exploration. In: Proc. of the 30th Int'l Conf. on Data Engineering. Washington: IEEE Computer Society, 2014. 472−483. [doi: 10.1109/ICDE.2014.6816674]

[111] Sellam T, Kersten ML. Meet Charles, big data query advisor. In: Proc. of the 2013 Biennial Conf. on Innovative Data Systems Research. 2013.

[112] Dimitriadou K, Papaemmanouil O, Diao YL. Interactive data exploration based on user relevance feedback. In: Proc. of the 30th Int'l Conf. on Data Engineering. Washington: IEEE Computer Society, 2014. 292−295. [doi: 10.1109/ICDEW.2014.6818343]

[113] Jiang LL, Nandi A. SnapToQuery: Providing interactive feedback during exploratory query specification. Proc. of the VLDB Endowment, 2015,8(11):1250−1261. [doi: 10.14778/2809974.2809986]

[114] Golovchinsky G, Diriye A, Dunnigan T. The future is in the past: Designing for exploratory search. In: Proc. of the 4th Information Interaction in Context Symp. New York: ACM Press, 2012. 52−61. [doi: 10.1145/2362724.2362738]

[115] Stolte C, Tang D, Hanrahan P. Polaris: A system for query, analysis, and visualization of multidimensional databases. Communications of the ACM, 2008,51(11):75−84. [doi: 10.1145/1400214.1400234]

[116] Key A, Howe B, Perry D, Aragon C. VizDeck: Self-organizing dashboards for visual analytics. In: Proc. of the 2012 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2012. 681−684. [doi: 10.1145/2213836.2213931]

[117] Schoeffmann K, Ahlström D, Bailer W, Cobârzan C, Hopfgartner F, McGuinness K, Gurrin C, Frisson C, Le D, Del Fabro M, Bai HL, Weiss W. The video browser showdown: A live evaluation of interactive video search tools. Int'l Journal of Multimedia Information Retrieval, 2014,3(2):113−127. [doi: 10.1007/s13735-013-0050-8]

[118] Acharya S, Gibbons PB, Poosala V, Ramaswamy S. The aqua approximate query answering system. In: Proc. of the 1999 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 1999. 574−576. [doi: 10.1145/304182.304581]

[119] Olston C, Bortnikov E, Elmeleegy K, Junqueira F, Reed B. Interactive analysis of Web-scale data. In: Proc. of the 2009 Biennial Conf. on Innovative Data Systems Research. 2009.

[120] Sidirourgos L, Kersten M, Boncz P. SciBORQ: Scientific data management with bounds on runtime and quality. In: Proc. of the 5th Biennial Conf. on Innovative Data Systems Research. 2011. 296−301.

[121] Agarwal S, Iyer AP, Panda A, Madden S, Mozafari B, Stoica I. Blink and it's done: Interactive queries on very large data. Proc. of the VLDB Endowment, 2012,5(12):1902−1905. [doi: 10.14778/2367502.2367533]

[122] Mozafari B, Ramnarayan J, Menon S, Mahajan Y, Chakraborty S, Bhanawat H, Bachhav K. SnappyData: A unified cluster for streaming, transactions and interactice analytics. In: Proc. of the 2017 Biennial Conf. on Innovative Data Systems Research. 2017.

[123] Park Y, Mozafari B, Sorenson J, Wang JH. VerdictDB: Universalizing approximate query processing. In: Proc. of the 2018 Int'l Conf. on Management of Data. New York: ACM Press, 2018. 1461−1476. [doi: 10.1145/3183713.3196905]

[124] Krishnan DR. The marriage of incremental and approximate computing [MS. Thesis]. Dresden: Technical University Dresden, 2016.

[125] Mozafari B, Niu N. A handbook for building an approximate query engine. IEEE Data Engineering Bulletin, 2015,38(3):3–29.

[126] Mozafari B. Approximate query engines: Commercial challenges and research opportunities. In: Proc. of the 2017 ACM Int'l Conf. on Management of Data. New York: ACM Press, 2017. 521–524. [doi: 10.1145/3035918.3056098]

[127] Cheng XQ, Jin XL, Wang YZ, Guo JF, Zhang TY, Li GJ. Survey on big data system and analytic technology. Ruan Jian Xue Bao/Journal of Software, 2014,25(9):1889–1908 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4674.htm [doi: 10.13328/j.cnki.jos.004674]

[128] Wang MX, Li FF, Gu Y, Yu G. Survey on interactive data exploration. Journal of Frontiers of Computer Science & Technology, 2017,11(2):171–184 (in Chinese with English abstract).

[129] Huang L, Sun K, Chen XZ, Zhou MQ. In-memory cluster computing: Interactive data analysis. Journal of East China Normal University (Natural Sciences), 2014,2014(5): 216–227 (in Chinese with English abstract).

附中文参考文献:

[127] 程学旗,靳小龙,王元卓,郭嘉丰,张铁赢,李国杰.大数据系统和分析技术综述.软件学报,2014,25(9):1889–1908. http://www.jos.org.cn/1000-9825/4674.htm [doi: 10.13328/j.cnki.jos.004674]

[128] 王蒙湘,李芳芳,谷峪,于戈.交互式数据探索综述.计算机科学与探索,2017,11(2):171–184.

[129] 黄岚,孙珂,陈晓竹,周敏奇.内存集群计算:交互式数据分析.华东师范大学学报(自然科学版),2014,2014(5):216–227.

袁喆(1994－),男,江西南昌人,学士,CCF 学生会员,主要研究领域为图计算,知识图谱,信息检索.

刘家俊(1984－),男,博士,副教授,主要研究领域为多媒体,计算机视觉和社交媒体中的数据挖掘,数据库,机器学习.

文继荣(1972－),男,博士,教授,博士生导师,CCF 杰出会员,主要研究领域为信息检索,数据挖掘,机器学习.

姚斌(1981－),男,博士,副教授,CCF 专业会员,主要研究领域为数据库管理,大数据分析.

魏哲巍(1986－),男,博士,教授,博士生导师,CCF 专业会员,主要研究领域为图计算,海量数据算法,数据流算法.

郑凯(1983－),男,博士,教授,博士生导师,CCF 专业会员,主要研究领域为数据库.