基于 DAG 的 Hive 数据溯源方法

杜 娟[1]，苏秋月[2]

（1.61646 部队，北京 100192；2.四川大学，四川 成都 610065）

摘 要： 数据溯源能够快速定位数据的来源及其产生过程，对审计、数据生命周期管理都有很大帮助，是数据安全防护中的重要一环。针对传统数据溯源机制难以应用于 Hive 数据仓库中大规模、复杂的数据处理的问题，提出了一种基于 DAG 的数据溯源方法，实现了对 Hive 中数据的溯源。最后通过实验对该方法的有效性以及性能进行了测试与验证。

关键词： 数据溯源；大数据；有向无环图

中图分类号： TP311.13　　**文献标识码：** A　　**DOI：** 10.19358/j.issn.2096-5133.2020.11.005

引用格式： 杜娟，苏秋月. 基于 DAG 的 Hive 数据溯源方法[J].信息技术与网络安全，2020，39(11)：31-37.

Hive data provenance method based on DAG

Du Juan[1], Su Qiuyue[2]

(1.Unit 61646 of PLA, Beijing 100192, China; 2.Sichuan University, Chengdu 610065, China)

Abstract: Data provenance can quickly locate the source of data and its production process, which is of great help to audit and data life cycle management. It is an important part of data security protection. Aiming at the problem that traditional data provenance mechanisms are difficult to apply to large-scale and complex data processing in Hive data warehouses, this paper proposes a data traceability method based on Directed Acyclic Graph (DAG). It can implement the traceability of Hive data. Finally, the effectiveness and performance of this method are tested and verified by experiments.

Key words: data provenance; big data; directed acyclic graph

0 引言

Hive 是基于 Hadoop 的开源数据仓库工具，它提供了丰富的 SQL 查询方式来分析存储在 Hadoop 分布式文件系统中的数据：可以将结构化的数据文件映射为一张数据库表，并提供完整的 SQL 查询功能；可以将 SQL 语句转换为 MapReduce 任务运行，通过自己的 SQL 查询分析需要的内容。这套 SQL 简称 Hive SQL，使不熟悉 MapReduce 的用户可以很方便地利用 SQL 语言查询、汇总和分析数据[1]。由于 Hive 在数据存储和分析上的灵活性，众多企业用它存储重要数据。这些敏感的商业数据被大量企业内部人员访问和操作，一旦发生人为误操作或违规操作，很容易导致数据的泄露。现有大数据平台上的数据安全防护方案缺乏对敏感数据灵活的访问控制，难以对数据的生命周期及用户操作行为进行精确的追踪溯源，无法提供对大数据合规审计管理的支撑。因此，如何提供有效的安全防护机制来保障 Hive 中数据的安全，是目前研究的重点。

数据溯源也称为数据血缘、数据谱系等，数据溯源技术根据追踪路径重现数据的历史、状态和演变过程，实现数据历史档案的追溯[2]。通过数据溯源能追踪到异常发生的原因，还能帮助人们确定数据仓库中各项数据的来源。国内外学者在数据溯源技术上进行了深入研究。在数据溯源模型方面，汪洪昕[3]提出了数据染色体溯源模型，更加完善地揭示数据传播过程中的变化及数据的关系，并在 Hadoop 平台中得以实现。郝鹏飞[4]通过对大数据模型分析平台工作流特征分析，讨论了基于 Oozie 模型工作流的数据溯源问题。

目前针对数据库的数据溯源追踪主要有两种方法：(1)基于标注的方法[5]，此类方法虽然实施起来比较简单，但需要额外的存储空间且随着处理的

数据量增加其执行效率会降低，难以直接应用于维护着海量数据的 Hive 数据仓库；(2) 基于逆置函数的方法[6]，此类方法需要的存储空间较小，但不是所有的数据处理都可以逆置，且其溯源追踪的性能完全取决于逆置机制。对于 Hive 数据仓库中复杂的数据处理，要构造一个良好的逆置机制难度较大。Hive 数据的溯源重点在于数据沿袭问题，而给定数据的数据沿袭问题可以概括为建立数据的血缘关系，得到其产生过程以及源数据。

对于数据仓库中数据溯源问题，柯洁[7]等人基于 W3C 的 PROV 模型对 ETL 过程的数据溯源进行了深入分析，并提出了相应的数据溯源算法。文献[8-9]讨论了数据仓库中的数据谱系跟踪问题，提供了谱系跟踪算法以及溯源过程中属性映射和转换起源集的求解方法。但这些研究均针对传统数据仓库中的数据溯源，难以应用于大数据环境下 Hive 的数据溯源。针对大数据环境，文献[10]提出了一种基于层的数据溯源架构，其中包括大数据来源的捕获及可视化，并且在溯源数据中引入了一种访问控制机制。文献[11-13]总结了数据库中的数据溯源技术，分析了在 Hadoop 环境下数据溯源面临的研究挑战，并从数据溯源模型、溯源数据存储、溯源查询语言等方面梳理了现有解决方案。Apache Atlas 是 Hadoop 社区为解决 Hadoop 生态系统的元数据治理问题而产生的开源项目，它为 Hadoop 集群提供了包括数据分类、集中策略引擎、数据溯源、安全和生命周期管理在内的元数据治理核心能力[14]，因此可以将 Apache Atlas 引入到 Hive 数据溯源中。

针对传统数据溯源机制难以满足 Hive 中大规模、复杂的数据处理问题，本文提出了基于有向无环图(Directed Acyclic Graph，DAG)的数据溯源方法。通过对 Apache Atlas 进行扩展，在 Hive 中实现了该数据溯源方法，并通过实验证明该方法可为 Hive 提供准确、高效的数据溯源机制，也为数据安全审计提供了有力支撑。

1 基于 DAG 的数据溯源方法

针对 Hive 中数据处理的特点，本文提出了基于 DAG 的数据溯源方法。该方法包含了两部分内容：数据血缘图的定义和基于数据血缘图的溯源追踪算法，下面针对这两部分进行具体的阐述。

1.1 数据血缘图定义

Hive 数据仓库中数据的变化主要源于 HQL 任务，用户可以通过 Hive 访问接口提交 HQL 语句，对 Hive 中的数据执行操作。下面对 HQL 语句中所涉及的相关实体及关系进行定义。

定义 1 数据项(Dataset)，表示数据处理过程中所涉及的数据。例如 Hive 中的数据库、表、列，以及 HDFS 中的文件。

定义 2 操作(Process)，表示数据处理过程中用户执行的具体操作，例如 CRETAE_AS_SELECT、INSERT、SELECT、DTOP，以及 EXPORT、IMPORT 等基本 HQL 操作。

定义 3 关系(Relationship)，表示实体之间的关联关系，通过关系建立起了数据实体与操作实体之间的联系。

定义 4 数据血缘图 $G=(V,E,R,A)$ 为一个有向无环图，其中：

$V \subseteq \mathrm{Du}\times\mathrm{OP}\times\mathrm{Dr}$

$E \subseteq (\mathrm{Di}\times\mathrm{OP}\times R)\cup(\mathrm{OP}\times\mathrm{Do}\times R)$

$R=\{\mathrm{usedBy},\mathrm{generated}\}$

$A=$

$\{\mathrm{guit},\mathrm{name},\mathrm{typeName},\mathrm{createTime},\mathrm{createBy},\mathrm{version}\}$

其中 Du 表示使用数据顶点的集合，OP 表示操作顶点的集合，Dr 表示结果数据顶点的集合，Di 和 Do 分别表示第 i 和第 o 项数据。E 表示边的集合，R 是指边类型的集合，包含了 usedBy 和 generated 两种类型。其中 usedBy 表示数据操作所使用的数据，建立了使用数据与操作之间的关系。generated 表示一个结果数据产生的过程，建立了操作与结果数据之间的关系。A 表示血缘图中顶点和边包含的属性，具体定义如表 1 所示。其中 guid 表示实体或关系的唯一标识；name 表示顶点(或边)所表示的实体(或关系)的名称；typeName 表示顶点或边的类型，这里除了边具有两种类型外，顶点的类型也包括 Dataset 和 Process 两种类型。

在一条 HQL 任务中，会存在嵌套子查询或多表

表 1 属性定义

名称	类别	描述
guid	String	唯一标识
name	String	实体名称
typeName	String	类型名称
createTime	Date	创建时间
createBy	String	创建者
version	String	版本号

关联查询等子操作。所以一条 HQL 语句可能会产生多个使用数据的顶点，但只产生一个结果数据顶点和一个操作顶点。例如对于某 HQL 语句“CREATE TABLE table_c AS SELECT … FROM table_a a JOIN table_b b ON…”，这里表 table_a 和 table_b 均为使用数据，表 table_c 为产生的结果数据，CREATE_AS_SELECT 为数据的操作。通过对该数据处理过程进行建模，产生了由四个实体顶点和三条边关系的血缘图，如图 1 所示。

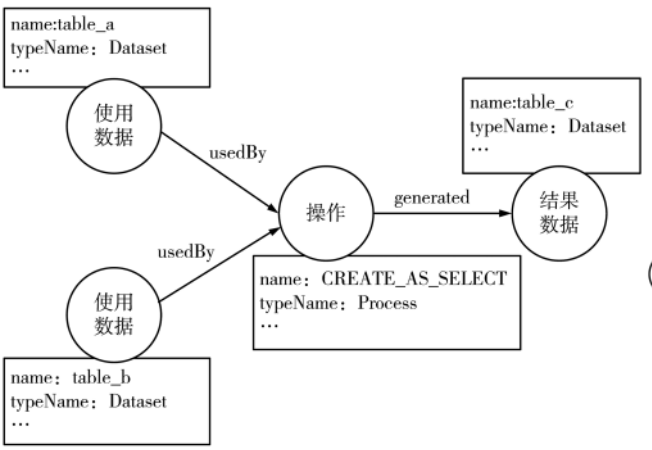

图 1 数据血缘图

1.2 基于 DAG 的溯源追踪算法

上节利用 DAG 构建了 Hive 中的数据的血缘关系，在此基础上，将给定数据的溯源追踪问题转变为了图的连通性问题，利用图连通算法找到数据顶点能连通的所有顶点，从而得到数据的血缘关系图，实现对数据的溯源追踪。因此，本文提出基于 DAG 的溯源追踪算法。引入深度优先搜索(Depth First Searching，DFS)算法[15]的思想，根据给定数据顶点进行深度优先查询，首先得到与该顶点相连的所有边，然后根据邻接边得到相邻顶点，递归执行，最终得到由给定数据顶点能连通的所有顶点构成的数据血缘图。基于 DAG 的溯源追踪算法流程如图 2 所示，具体步骤如下：

(1)初始化用来描述数据血缘关系的图 G；

(2)将表示给定数据的顶点 vertex 添加至图 G，并标记该顶点已被访问；

(3)获取所有与该顶点 vertex 相连的边，若不存在与之相连的边，则跳至步骤(5)，否则进行下一步骤；

(4)根据该边相连的两个顶点 vertex 和 vertex′，

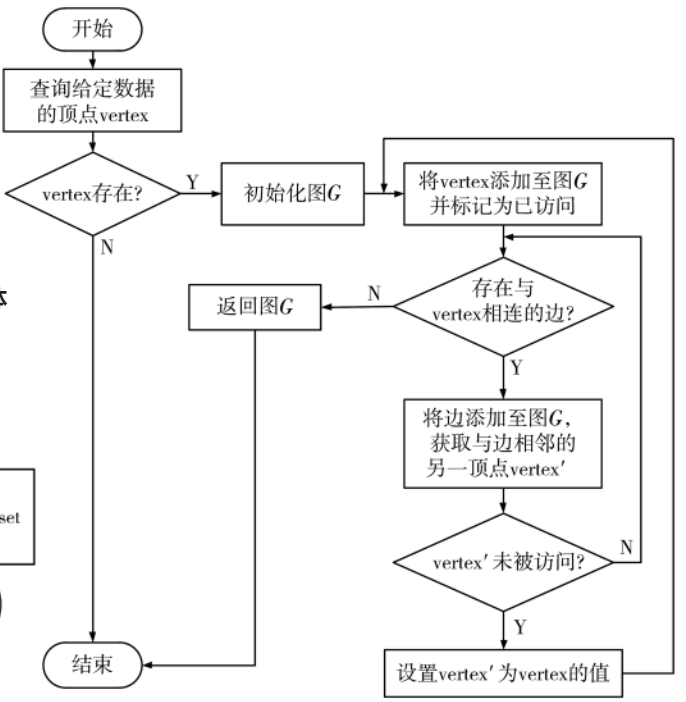

图 2 基于 DAG 的溯源追踪算法流程

如果 vertex′未被访问，则设置该顶点为 vertex 的值，然后跳转至步骤(2)，否则循环该步骤遍历其他相连的边；

(5)到此对给定数据的溯源已经完成，返回数据的血缘图 G。

1.3 基于 Hive 的实现框架

1.3.1 整体框架

在 Apache Atlas 的基础上进行了扩展，实现了基于 DAG 的数据溯源方法，整体框架如图 3 所示。该框架主要包括溯源收集、溯源信息建模和溯源追踪三个模块。该框架的简要工作流程如下：在用户执行 HQL 操作之后，内嵌于 Hive 中的溯源收集模块收集数据操作过程中的溯源信息，并发送给溯源信息建模模块；溯源信息建模模块在收到溯源信息后，对溯源信息进行建模，将收集的溯源信息转换成数据血缘图，并持久化存储在图数据库中；当对给定数据进行溯源追踪时，溯源追踪模块利用基于 DAG 的溯源追踪算法对给定数据进行追踪溯源，最终返回数据的血缘关系图。

实现框架采用的图数据库为 JanusGraph[16]，并利用 AtlasGraphManagement 接口实现对图数据的操作。由于溯源追踪模块主要通过基于 DAG 的溯源追踪算法实现给定数据的溯源追踪，后面就不再介

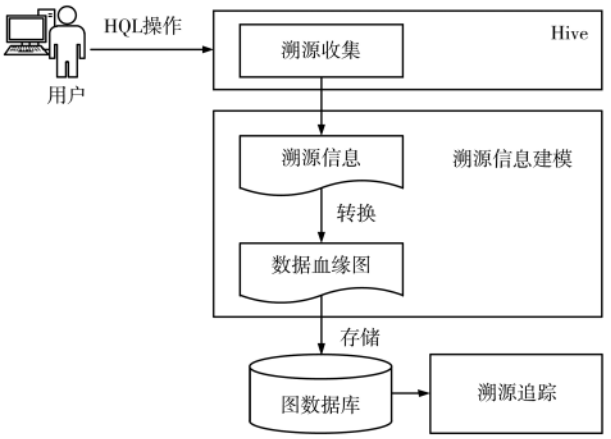

图 3 基于 Hive 的实现框架

绍该模块的实现。

1.3.2 溯源收集

溯源收集是 Hive 数据溯源实现的基础，本文主要关注溯源收集的方式和内容两方面。为了在保证数据准确性的前提下不对 Hive 本身带来过高的负载，本文采用在 Hive 中实现 Hook 钩子函数的形式，在 HQL 执行完成后，实时地收集数据的变化过程。这里基于 Apache Atlas 实现对 Hive 溯源信息的收集，具体是在 Hive 配置文件中添加如图 4 所示信息。

```
<property>
    <name>hive.exec.post.hooks</name>
    <value>org.apache.atlas.hive.hook.HiveHook</value>
</property>
```

图 4 Hive 的 Hook 配置

在确定收集的方式和时机之后，需要考虑应该收集数据处理过程中的哪些内容，这些内容必须与溯源相关。在对 Hive 进行数据溯源时，不能完全使用 Atlas 收集的元数据来构建数据的溯源关系。因此，本文参考了 W7 模型中以数据为中心的思想[17]，追溯在何时、何地以及何种原因，由何人通过何种方式对数据进行了何种操作，重新定义了收集的溯源信息，如表 2 所示。这些内容可以为后续利用数据血缘图模型建立数据的血缘关系提供重要的基础。

由表 2 可以看出，定义的 Hive 溯源信息表示一次 HQL 任务中所涉及的相关信息，主要分为三类：数据相关信息、操作相关信息和上下文信息。

1.3.3 溯源信息建模

利用数据血缘图模型对收集的溯源信息进行建模，首先提取出溯源信息中的数据和操作等基本对象，根据对象类型创建对应的实体；然后根据实体创建图顶点，以及根据实体属性中包含的与相关实体的关系创建边，并保存于图数据库中。每种类型的实体均由唯一标识和一组基本属性集合组成。唯一标识是在创建实体或关系时产生的，而属性集合来自于溯源收集的内容。数据实体的属性包含了溯源信息中数据相关的信息，操作实体的属性包含了操作和上下文相关的信息。

表 2 溯源信息

名称	类别	描述
dataType	String	数据类型
dataName	String	数据名称
owner	String	数据所有者
comment	String	注释
userName	String	操作者
operationType	String	操作类型
queryText	String	HQL 语句
evtTime	Data	操作时间
clientIP	String	IP 地址
inputs	Array	输入数据集合
outputs	Array	输出数据集合

对一条溯源信息建模，以数据和操作为顶点，它们之间的关系作为有向边，以此形成数据血缘图的实现流程如图 5 所示，具体的步骤如下：

(1)提取溯源信息中的基本对象，包括使用数据、操作以及结果数据。

(2)为使用数据和结果数据分别创建 Dataset 类型的实体，为其分配唯一标识 guid 以及设置基本属性。根据实际情况，这里可能会产生多个输入 Dataset 类型的实体。

(3)为溯源信息中的操作对象创建 Process 类型的实体，并根据操作以及上下文信息设置基本属性。其中在 inputs 集合和 outputs 集合中分别添加使用数据实体和结果数据实体。

(4)根据 Process 实体属性中的 inputs 集合和 outputs 集合包含的数据实体，创建 Relationship 依赖关系，分别为 usedBy 类型和 generated 类型。

(5)根据实体创建顶点，并设置基本属性。

(6)根据关系创建边，并设置边的两个邻接点及基本属性。

(7)将顶点和边添加至图中，并以邻接表的方式持久化存储在图数据库中。

通过上述流程，将 Hive 中数据处理过程收集的

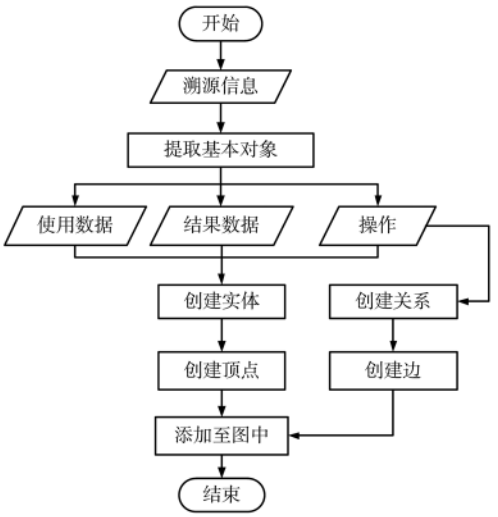

图 5 溯源信息建模过程

溯源信息构建成了具有关联关系的数据血缘图，并且利用图数据库将血缘关系进行了持久化的存储，为后续基于 DAG 的溯源追踪算法对给定数据进行溯源，还原出数据的血缘关系图提供了可能。

2 实验与分析

2.1 实验环境

通过构建 Hive 环境进行实验，在其中一个从节点上部署 Kafka，创建了一个副本因子为 1、分区数为 1 的主题，用于存储 Hive 中的溯源信息。在另外一个从节点上部署了 JanusGraph 图数据库，其中使用 Hbase 作为图数据的存储后端，Solr 作为图数据的搜索引擎。另外由于本文在 Apache Atlas 的基础上进行扩展，实现了基于 DAG 的数据溯源方法，因此在主节点上部署了 Atlas。详细信息如表 3 所示。

表 3 节点相关信息

名称	类别	描述
主节点	Hbase	1.2.6.1
	Solr	7.2.0
	Atlas	1.0.0
从节点 1	Kafka	2.10-2.1.0
从节点 2	Zoopeeper	3.4.5

2.2 有效性测试

为了验证 Hive 中数据处理过程中的数据是否被正确的溯源，本节对 Hive 中的数据执行了部分常见的 HQL 操作作为测试用例，如表 4 所示。从表 4 可以看出，Hive 中数据处理过程中的数据被正确溯源。

表 4 数据操作测试用例

编号	HQL 语句	描述
1	IMPORT TABLE empinfo FROM '/home/empinfo.csv';	测试是否对 IMPORT 操作过程构建了数据血缘
2	INSERT INTO TABLE empinfo SELECT d.name name, e.name emp_name, u.salary salary FROM deportment d LEFT JOIN employee e on d.name=e.depart;	测试是否对 INSERT 操作过程构建了数据血缘
3	CREATE TABEL avgsalary AS SELECT name,avg(salary) avgsalary FROM empinfo GROUP BY name;	测试是否对 CREATE 操作过程构建了数据血缘
4	EXPORT TABLE avgsalary TO '/home/data/avgsalary';	测试是否对 EXPORT 操作过程构建了数据血缘
5	SELECT * FROM avgsalary ORDER BY avgsalary DESC;	测试是否对 SELECT 操作过程构建了数据血缘

利用 Apache Atlas 的可视化管理页面查询 Hive 中数据的血缘信息，这里查询了表 avgsalary 的数据溯源关系，如图 6 所示。

由图 6 可以看出，表 avgsalary 的血缘关系分为两种类型：一种关系是与表 avgsalary 中数据的来源有关；另一种关系是与表 avgsalary 中数据的去处有关。首先查找表 avgsalary 的来源，以该表为起点往前追溯，可知它是由用户对表 empinfo 执行了 CREATE AS SELECT 操作得到的。而表 empinfo 数据的产生总共有三部分的来源，第一部分是 HDFS 中的 empinfo.csv 文件；另外两个部分是来自 Hive 中原有的表 department 和表 employee。到此，发现没有可以往前追溯的节点了，然后返回到表 avgsalary，往后查找其产生了哪些数据。发现该表的数据有两个去处，一是经过 EXPORT 操作产生了 HDFS 文件‘avgsalary.csv’；二是经过 SELECT 操作产生了临时文件。

2.3 性能测试

为了测试本文所提出的数据溯源方法对性能的影响，本节从溯源信息采集对 HQL 执行的额外时间开销以及溯源追踪效率两方面进行了测试。

(1)溯源信息采集时间开销实验

本文对 Hive 进行了扩展，在 Driver 驱动中增加了溯源收集模块，因此本实验旨在对扩展前后的 Hive 进行对比测试，检验溯源收集过程对整体性能的影响。本实验执行相同的 HQL 任务处理不同规

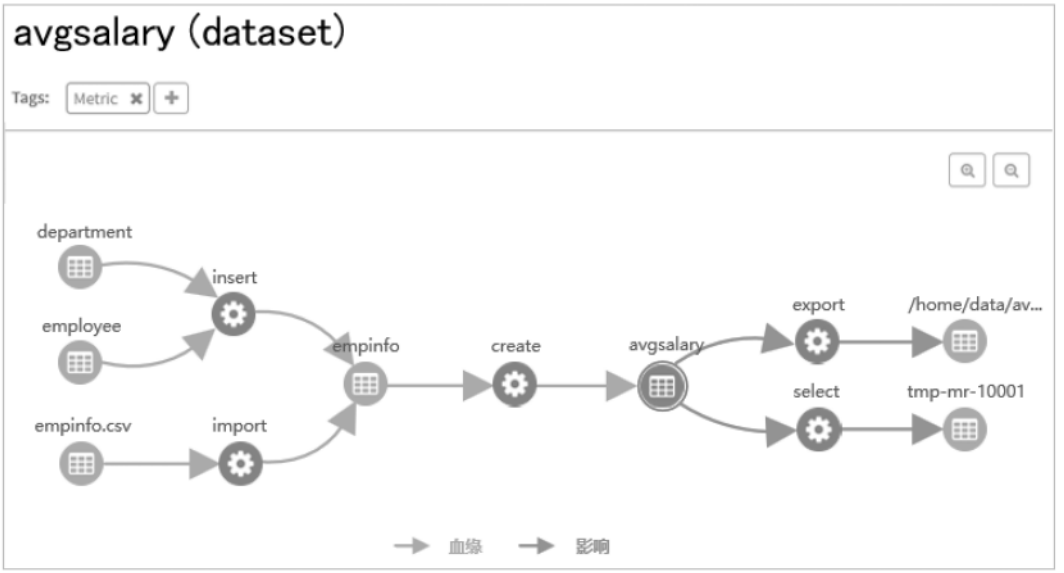

图 6 表 avgsalary 的数据溯源图

模的数据，测试对不同输入数据规模下，加入溯源信息采集模块前后 HQL 执行时间的对比，以及在 HQL 过程中溯源采集的时间开销。该测试实验设置为 8 组，其中数据量从 64 MB 到 512 MB，分别测试其执行时间，每次的执行时间取同一实验执行 3 次的平均值。实验结果如图 7 所示。

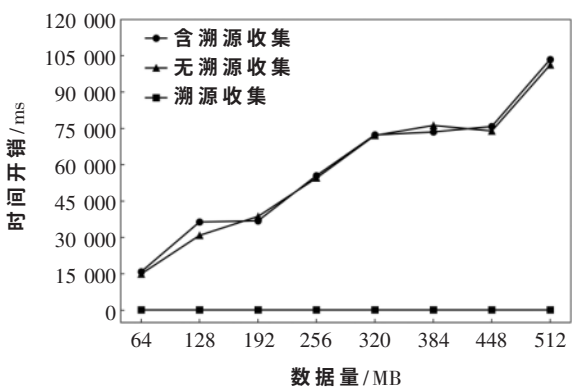

图 7 溯源收集对 HQL 执行时间性能影响

由图 7 可以看出，加入溯源收集模块与无溯源收集模块的 HQL 执行时间均随着输入数据量的增加而呈线性增长，但有溯源收集模块与无溯源收集模块的时间开销差距较小。另外，在整个执行过程中，溯源收集的时间开销趋近于零。由此可知，溯源采集对 HQL 的额外时间开销可以忽略不记。

(2)溯源追踪效率实验

为了实现根据给定数据项的溯源追踪，本文提出了基于 DAG 的溯源追踪方法，利用图论查询算法完成对数据的来源以及产生过程的追踪溯源。在不同输入数据规模的情况下，对溯源追踪的时间开销进行了测试，每次的执行时间取同一实验执行 3 次的平均值。实验结果如图 8 所示。可以看出，随着数据规模的增加，数据溯源追踪的时间基本上保持在 300 ms 左右，且波动范围很小。由此可知，本文提出的溯源追踪方法整体性能较好。

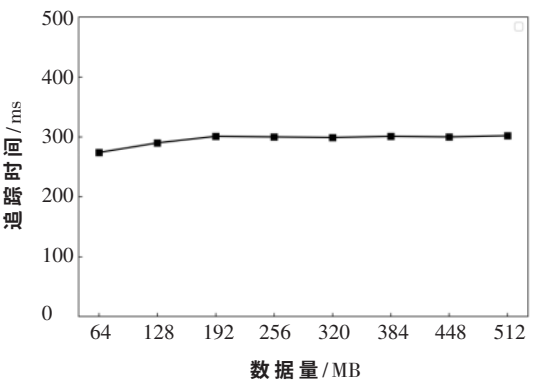

图 8 不同输入数据量下溯源追踪时间开销

3 结论

本文对 Hive 数据仓库的数据溯源进行了研究，首先介绍了数据溯源的相关理论与机制，并针对传统数据溯源难以应用于 Hive 中大规模、复杂的数据处理的问题，提出了基于 DAG 的数据溯源方法；然后对该方法中数据血缘图的构建以及基于 DAG 的溯源追踪算法进行了阐述；接着基于 Apache Atlas 实现了该方法对 Hive 中数据溯源及可视化的展示；

最后通过实验验证了本文所提的数据溯源方法的有效性，并从溯源收集和溯源追踪两方面的性能测试，验证了该方法的整体性能较好。实验表明，基于 DAG 的数据溯源方法不仅实现了对 Hive 中数据准确、高效的溯源，也为数据操作审计提供了有力支撑。

参考文献

[1] 万川梅，谢正兰.HADOOP 应用开发实战祥解(修订版)[M].北京：中国铁道出版社，2014.

[2] 明华，张勇，符小辉.数据溯源技术综述[J].小型微型计算机系统，2012，33(9)：1917－1923.

[3] 汪洪昕.Hadoop 环境下的数据溯源方法的应用研究[D].南京：南京邮电大学，2016.

[4] 郝鹏飞.大数据模型分析平台下的数据溯源关键技术研究[D].成都：电子科技大学，2017.

[5] ZHAO Y，WILDE M，FOSTER I.Applying the virtual data provenance model[C].International Provenance and Annotation Workshop，Springer，Berlin，Heidelberg，2006：148－161.

[6] DEZANI－CIANCAGLINI M，HORNE R，SASSONE V.Tracing where and who provenance in linked data：a calculus[J].Theoretical Computer Science，2012，464(464)：113－129.

[7] 柯洁，董红斌，梁意文.基于 PROV 的 ETL 起源信息统一表达机制[J].四川大学学报(工程科学版)，2015，47(5)：123－129.

[8] TOMINGAS K，JÄRV P，TAMMET T.Computing data lineage and business semantics for data warehouse[C].International Joint Conference on Knowledge Discovery，Knowledge Engineering，and Knowledge Management.Springer，Cham，2016：101－124.

[9] TOMINGAS K，JÄRV P，TAMMET T.Discovering data lineage from data warehouse procedures[C].The 8th International Conference on Knowledge Discovery and Information Retrieval，2016：101－110.

[10] AGRAWAL R，IMRAN A，SEAY C，et al.A layer based architecture for provenance in big data[C].2014 IEEE International Conference on Big Data(Big Data).IEEE，2014：1－7.

[11] LOHIA P，GUPTA H，BRAHMA S，et al.Efficiently processing workflow provenance queries on SPARK[J].arXiv preprint arXiv：1808.08424，2018.

[12] CUZZOCREA A.Provenance research issues and challenges in the big data era[C].2015 IEEE 39th Annual Computer Software and Applications Conference.IEEE，2015，3：684－686.

[13] GUPTA H，MEHTA S，HANS S，et al.Provenance in context of Hadoop as a Service(HaaS)－state of the art and research directions[C].IEEE International Conference on Cloud Computing，2017.

[14] Data governance and metadata framework for Hadoop [EB/OL].(2017－06－15)[2019－03－12].https：//cwiki.apache.org/confluence/display/ATLAS.

[15] 陶华，杨震，张民，等.基于深度优先搜索算法的电力系统生成树的实现方法[J].电网技术，2010，34(2)：120－124.

[16] JanusGraph distributed graph database[EB/OL].(2017－01－20)[2019－03－12].https：//janusgraph.org/.

[17] RAM S，LIU J.A new perspective on semantics of data provenance[C].Proceedings of the First International Conference on Semantic Web in Provenance Management，2009：35－40.

(收稿日期：2020－09－01)

作者简介：

杜娟(1985－)，通信作者，女，博士研究生，助理研究员，主要研究方向：态势感知。E－mail：28927066@qq.com。