Lakehouse Architecture and Query Optimization for Time-Series Data in Vehicle-Road-Cloud Collaboration

Xiang Zhang[1] Dong Han[2] Yuchen Zhang[2*]

1. Beijing Municipal Engineering Design and Research Institute Co., Ltd. - National Engineering Laboratory for Big Data Collaborative Security Technology, Beijing, 100082, China

2. Beijing Municipal Engineering Design and Research Institute Co., Ltd. - Road Traffic Institute, Beijing, 100082, China

Abstract

The vehicle road cloud collaborative data includes vehicle status, roadside perception, and traffic events, which need to support real-time writing and retain historical records required for trajectory tracing. This article organizes this type of time-series data on Apache Iceberg: partitioning by time and device type, clustering within partitions by (vehicle_i, event_ts), and then using Bloom filters, Min Max statistics, small file merging, and dynamic cropping to limit the scanning range. In the four types of query experiments, the latency of scheme D decreased by more than 75% compared to scheme A, and the number of files and I/O were also significantly reduced.

Keywords

vehicle-road-cloud collaboration; time-series data; Lakehouse; query optimization; Apache Iceberg

面向车路云协同的时序数据湖仓一体架构与查询优化

张翔[1] 韩冬[2] 张雨晨[2*]

1.北京市市政工程设计研究总院有限公司 大数据协同安全技术国家工程实验室，中国•北京 100082

2.北京市市政工程设计研究总院有限公司 道路交通院，中国•北京 100082

摘要

车路云协同数据同时包含车辆状态、路侧感知和交通事件，既要支持实时写入，也要保留轨迹回溯所需的历史记录。本文在Apache Iceberg上组织这类时序数据：按时间和设备类型分区，在分区内按(vehicle_id,event_ts)聚类，再用布隆过滤器、Min−Max统计、小文件合并和动态裁剪限制扫描范围。四类查询实验中，方案D的延迟较方案A下降75%以上，文件数和I/O也明显减少。

关键词

车路云协同；时序数据；湖仓一体；查询优化；Apache Iceberg

1 引言

车路协同依托车载单元、路侧设施与云平台实现多主体信息交互，持续产生车辆轨迹、传感器读数、交通事件和V2X消息。其高吞吐、强时空关联与长期积累特征，为交通态势分析、风险预警和算法训练提供数据基础。

其管理难点包括：数据源与格式异构，协议升级引起频繁Schema演化；实时预警和历史分析并存，对低延迟流处理与海量批处理均有要求；不同模态、不同阶段的数据分散存放，增加关联分析和长期回溯成本。

数据仓库SQL分析能力强，但Schema-on-Write和扩展成本不利于接纳快速变化的原始数据；数据湖存储灵活、成本低，却缺少完备的事务、质量和元数据管理，易形成小文件和“数据沼泽”。湖仓一体以开放表格式在对象存储上提供ACID事务、Schema演化和统一元数据，融合两类能力。

现有湖仓研究多面向通用分析，对车路协同数据的时间、车辆标识和空间关联考虑不足。单一时间分

【基金项目】面向现代城市交通及城市治理的全生命周期一体化智慧场景技术研发及示范应用（项目编号：北京市2025年科技创新专项资金支持项目BG2025−0109−SZZY−FYH）。

【作者简介】张翔（1973−），男，中国安徽巢湖人，硕士，大数据高级工程师，从事大数据、市政基础设施研究；韩冬（1983−），男，北京市人，硕士研究生，从事交通规划、智慧交通研究。

【通讯作者】张雨晨（1994−），男，中国北京人，硕士，从事智慧交通、信号控制研究。

区难以兼顾轨迹回溯、区域统计与事件关联，多维数据组织与查询下推仍有优化空间[1]。

针对上述问题，本文以Apache Iceberg为表格式，构建批流一体的车路云时序数据湖仓；提出时间—设备类型分区、Z-Order聚类、布隆过滤器、小文件合并及动态分区裁剪等协同优化方法；通过对比实验从写入、存储、查询和资源消耗四方面验证有效性。

2 湖仓一体架构设计

车路协同数据具有高吞吐、强时序、多源异构和模式演化特征，涵盖结构化状态、半结构化日志及图像视频。实时预警关注秒级数据，仿真与模型训练依赖长期历史数据，平台需兼顾扩展写入、批流处理、统一治理和跨模态关联，避免形成数据孤岛。

数据仓库采用Schema-on-Write，适合稳定数据上的复杂SQL，但对模式变化及非结构化数据适应性较弱。数据湖采用Schema-on-Read，可低成本保存原始数据，却常缺乏事务、一致性约束和精细元数据。车路协同既需要开放存储，也需要可靠治理和查询性能，单一方案难以覆盖全流程。

开放表格式研究涵盖Apache Iceberg[2]、Delta Lake[3]和通用湖仓架构[4]。本文据此设计接入、计算、表格式管理、统一存储与数据服务五层架构，在共享数据副本上完成批流处理。

对象存储承担容量，Iceberg保障事务、演化与追溯，Flink、Spark和Trino分别支持流式接入、离线治理与交互分析，各层通过标准接口解耦并独立扩容。

如图1所示，Kafka实时流与历史文件经Flink、Spark处理后写入同一Iceberg表，Parquet文件存于对象存储或HDFS，上层查询共享数据副本，避免跨系统复制。

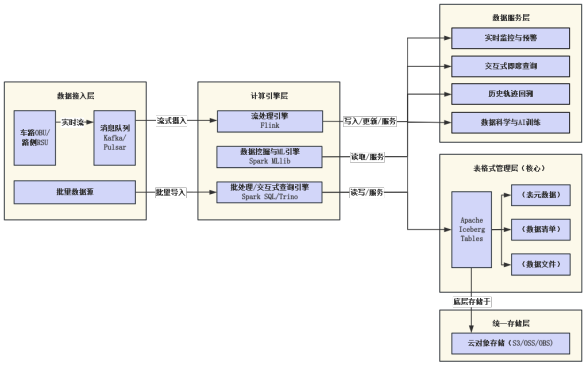


图1 车路云时序数据湖仓一体架构

统一存储层采用S3、OSS、OBS或HDFS保存Parquet、ORC及图像视频，版本、分区和文件关系由Iceberg维护。结构化时序表优先使用Parquet，通过列式压缩、投影和谓词下推减少扫描。

Iceberg以快照和清单描述文件，支持原子提交、并发读写、回滚和时间旅行。基础Schema包含vehicle_id、event_ts、device_type、位置及速度等字段，以properties MAP承载可变属性，字段变更通过Schema演化完成。

数据按event_ts的天粒度隐藏分区，必要时细化到小时，再按device_type隔离设备；分区内依据(vehicle_id,event_ts)计算Z值并排序，使同车相近时段数据落入相邻文件。查询先裁剪分区，再以文件统计缩小候选范围。

实时数据经Kafka[5]进入Flink完成解析、去重、校验及微批写入，历史文件由Spark执行ETL和重排后写入同一Iceberg表；Trino/Presto提供交互式SQL，Spark承担离线分析和机器学习。

3 查询优化方法

针对轨迹点查、时空聚合、事件关联和窗口计算四类负载，本文从数据组织、文件索引和执行计划三个层次逐级缩小候选范围，避免单项优化被后续扫描开销抵消。

物理布局先以时间和设备类型划分稳定分区，再用Z-Order改善分区内多维局部性并周期性重写。Z-Order将归一化维度的二进制位交错为一维Z值，本文选择vehicle_id哈希值与event_ts，兼顾同车聚合和时间连续。

以单车轨迹查询为例，系统先按日期剪枝，再根据文件级vehicle_id和event_ts统计过滤，最后读取必要行组。Z-Order使目标时段记录集中于少量相邻文件，从而减少随机I/O。

流式微批产生的小文件会造成清单膨胀和随机I/O。后台按分区及Z-Order范围收集小文件，数据达到阈值后由Spark重排为接近128MB的新文件，并以Iceberg事务原子替换；合并频率依据文件增长和查询

负载调整。

对高基数vehicle_id建立布隆过滤器，点查时在读取Parquet前跳过确定不含目标车辆的文件；对event_ts、speed等数值字段使用文件和行组级Min-Max统计，若查询区间与[min,max]不相交则直接剪枝。

事件关联查询先获取小表traffic_events的时间范围和区域，再生成vehicle_telemetry的分区及文件过滤条件并下推，仅对候选文件执行空间判断与连接，减少大表读取和网络交换。

Parquet按列读取，Trino批量向量处理。查询仅投影所需列，连续内存访问提高CPU缓存命中率并便于SIMD并行；批大小按列选择率配置，以平衡并行度和内存。

4 实验与结果分析

实验在6节点集群上进行，每节点配置24核CPU和128GB内存，使用Ceph对象存储与万兆网络；软件包括Hadoop 3.3.4、Spark 3.3.2、Iceberg 1.0.0、Trino 405和Flink 1.16.2。

实验生成30天模拟数据，包含10万辆车、86.4亿条记录，原始容量约2.0TB，经Parquet和Snappy压缩后约0.5TB。车辆位置按速度、方向和时间更新，并叠加高斯噪声模拟定位误差。

对比方案A为传统Parquet数据湖，方案B增加Iceberg并按时间分区，方案C加入设备类型分区和Z-Order，方案D进一步加入布隆过滤器、小文件合并及执行优化。指标包括写入吞吐、文件数量、查询响应时间及CPU、网络和磁盘I/O。

实验结果显示，方案A吞吐量最高，为45.2万条/秒，但延迟波动和文件数最大；方案D吞吐量为36.2万条/秒，下降约20%，延迟标准差由1250.3ms降至98.3ms，文件数由28452降至1089，存储容量基本不变。

表1 典型查询响应时间（s）

查询类型	方案A	方案B	方案C	方案D	性能提升(DvsA)
Q1: 单车轨迹查询	12.5	8.3	3.2	1.8	85.6%
Q2: 区域车辆统计	25.8	18.5	9.7	6.3	75.6%
Q3: 事件关联分析	186.4	124.7	67.3	42.5	77.2%
Q4: 时序聚合分析	45.2	32.1	15.8	10.4	77.0%

1.Q1响应时间由12.5s降至1.8s，Q2由25.8s降至6.3s，Q3由186.4s降至42.5s，Q4由45.2s降至10.4s，降幅分别为85.6%、75.6%、77.2%和77.0%。结果表明，Z-Order、索引、动态裁剪和向量化对不同负载形成互补。

资源监测与响应时间变化是一致的：方案D将CPU利用率从85.2%降至52.1%，网络I/O从420.5MB/s降至123.6MB/s，磁盘I/O从1250.8MB/s降至285.7MB/s。这里的下降并非减少了查询任务，而是因为分区、文件统计和列投影先排除了大量无关数据，剩余数据再由向量化执行处理。

把方案A、B、C、D按步骤对照，可以看出性能改善具有叠加关系。B先提供Iceberg快照和时间分区，C再加入设备类型与Z-Order，D补充过滤器、合并和执行阶段优化。四个方案的差异说明，单靠表格式不能解决车辆点查，单靠排序也不能处理小文件，必须把布局、索引和执行计划放在同一条链路上。

文件数从28452降到1089，是本实验中最直观的维护收益。小文件减少后，查询不必反复解析大量清单，调度任务和文件句柄也随之减少；合并过程通过Iceberg事务替换旧文件，在线读者仍按原快照读取。128MB只是本实验的目标值，生产环境应结合写入速率和对象存储吞吐重新标定。

四类查询的变化并不相同。Q1从12.5s降到1.8s，说明车辆过滤和Z-Order对点查最有效；Q2、Q4分别降到6.3s和10.4s，主要受益于时间分区、列投影和向量化；Q3由186.4s降到42.5s，降幅较大但绝对耗时仍最高，表明事件范围传递和大表连接仍是后续优化重点。

加速并非没有代价。方案D吞吐量为36.2万条/秒，比方案A低约20%，代价来自排序、索引构建和文件重写；相应地，延迟标准差从1250.3ms降至98.3ms，文件数也大幅下降。对于需要频繁写入而很少查询的短期任务，可以降低重排频率；对于长期回溯型任务，牺牲部分写入吞吐是可接受的。

如果将该方案部署为线上服务，监控项应直接对应本文的优化链：记录候选文件数、实际扫描字节、过滤器命中率、合并队列长度和查询分位延迟。只有当这些指标持续偏离基线时才触发重写，避免把维护

任务本身变成新的I/O高峰。

本实验使用模拟数据，尚未覆盖迟到、乱序、重复及设备时钟漂移。真实接入时应以事件时间设置水位线，并用vehicle_id、event_ts和消息序列号构造幂等键；字段完整性、坐标范围和速度突变等规则需要单独统计，异常记录不能直接混入训练样本。

云边协同可以沿用本文的分层思路：边缘侧完成协议解析、短时缓存和初步过滤，中心湖仓负责统一目录、长期保存和跨区域分析。近期轨迹采用较细分区，冷数据降低维护频率；权限控制应区分车辆、道路和事件字段，避免把原始轨迹无差别开放给所有分析任务。

分区粒度、Z-Order维度和合并周期也不宜永久固定。可用近期谓词分布、扫描放大比和文件增长速率生成候选布局，在离线样本上比较查询节省与重写成本后再发布；当重写成本更高时保留原布局。这样的反馈机制比定期盲目重排更适合负载变化明显的车路云系统。

跨主体共享还需要审计边界。车辆标识可采用分级脱敏或可撤销映射，敏感位置字段按角色授权；对象存储启用传输和静态加密，Iceberg快照、提交记录与查询日志形成追溯链。本文只验证查询性能，安全策略仍需在真实权限模型和合规要求下单独测试。

5 结论

本文构建面向车路云协同的时序数据湖仓一体架构，并形成从分区、Z-Order聚类、文件索引和小文件合并到执行下推的完整优化链。完整方案虽使写入吞吐降低约20%，但四类查询响应时间均降低75%以上，资源开销同步下降。

研究仍受模拟数据和经验参数限制，后续需在真实道路、设备故障及数据倾斜条件下验证，并结合代价模型或在线学习自适应调整分区、聚类和合并策略，进一步扩展云—边—端协同与数据库智能优化。

参考文献

[1] 王小丽, 陈华辉. ZFT索引: 基于分段线性回归的学习型多维索引[J]. 计算机应用与软件, 2024, 41(10): 24-31.

[2] MISHRA S. Comparing Apache Iceberg and Databricks in building data lakes and mesh architectures[J]. International Journal of AI, BigData, Computational and Management Studies, 2022, 3(4): 37-48.

[3] ARMBRUST M, DAS T, SUN L, et al. Delta Lake: high--performance ACID table storage over cloud object stores[-J]. Proceedings of the VLDB Endowment, 2020, 13(12): 3411-3424.

[4] JHA V Y. Data lakehouse architectures: bridging structure d and unstructured data[J]. International Journal of Advanced Research in Computer Science & Technology, 2024, 7(6): 11218-11221.

[5] 谢星星, 张翠平, 田小燕, 等. 消息队列技术综述[J]. 计算机科学与应用, 2023, 13(12): 2180-2190.