Apache Flink

Flink在快手的应用与实践

董亭亭 · 快手 / 高级研发工程师

Apache Flink Meetup 北京 - 2019年06月29日

Apache Flink

CONTENT

目录 >>

Apache Flink

01

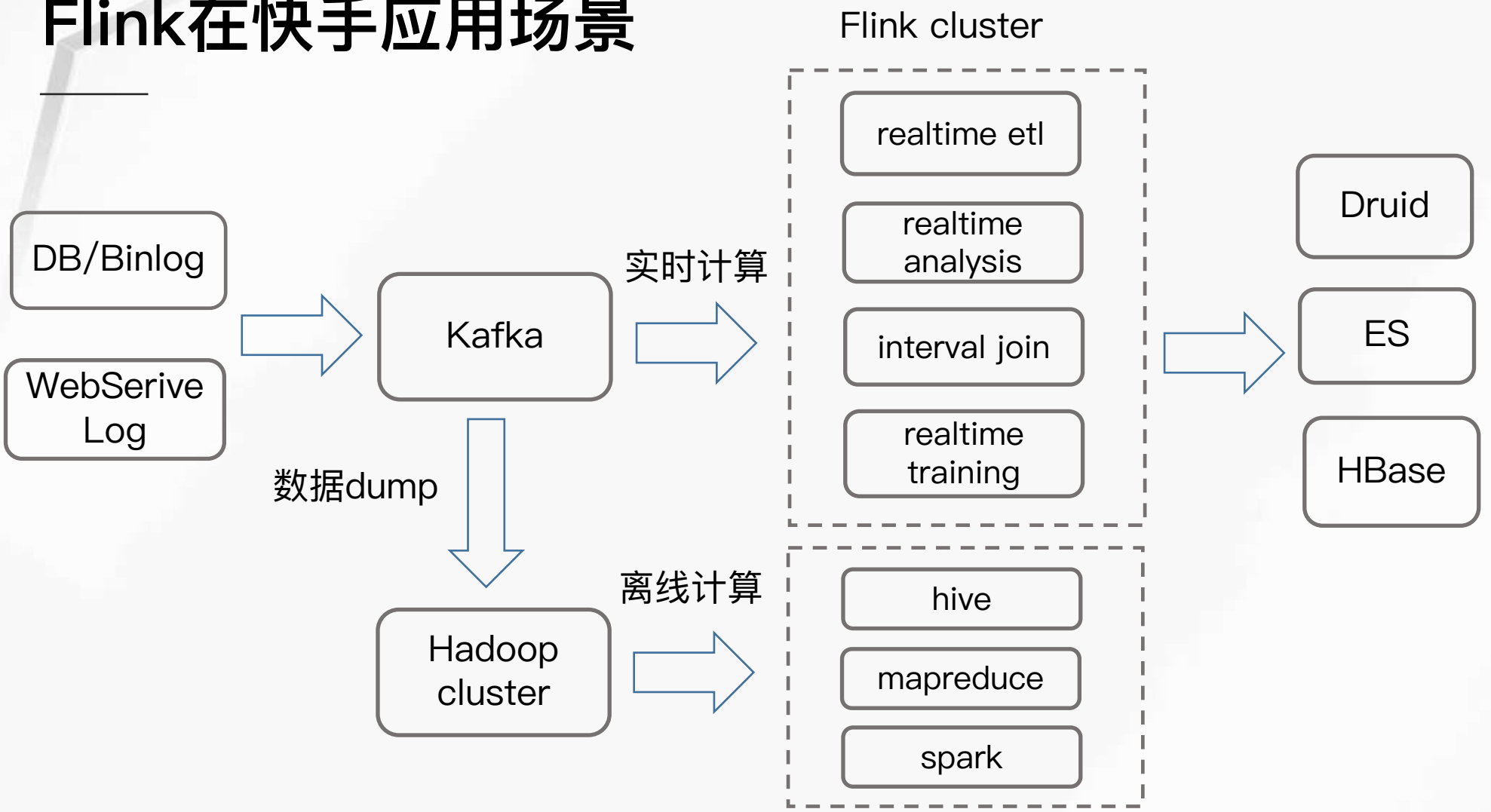
Flink在快手应用场景与规模

Apache Flink

Flink在快手应用

- Flink在快手应用场景
- Flink集群规模

Flink在快手应用场景



数据接入 | 数据计算 | 数据应用

Apache Flink

Flink在快手应用场景

实时统计各类指标、监控项

辅助业务实时分析、监控

对数据清洗、拆分、join等处理

大topic的数据拆分、清洗

实时业务处理

特定业务逻辑处理：实时调度

Flink在快手应用场景

短视频、直播质量监控

实时统计直播观众端、主播端播放量、卡顿率、开播失败率等多种监控数据。

用户增长分析

实时统计各投放渠道拉新情况，根据效果实时调整各渠道投入

实时数据处理

广告展现流、点击流实时join

客户端日志的拆分

直播cdn调度

实时监控各cdn厂商质量，调整各cdn厂商流量配比

Flink集群规模

1500
集群规模

1.7万亿
日处理条目数

3.7千万
峰值处理

- **集群部署：On Yarn**
- **离线集群：标签物理隔离**
- **实时集群：业务隔离，稳定性要求极高**

02

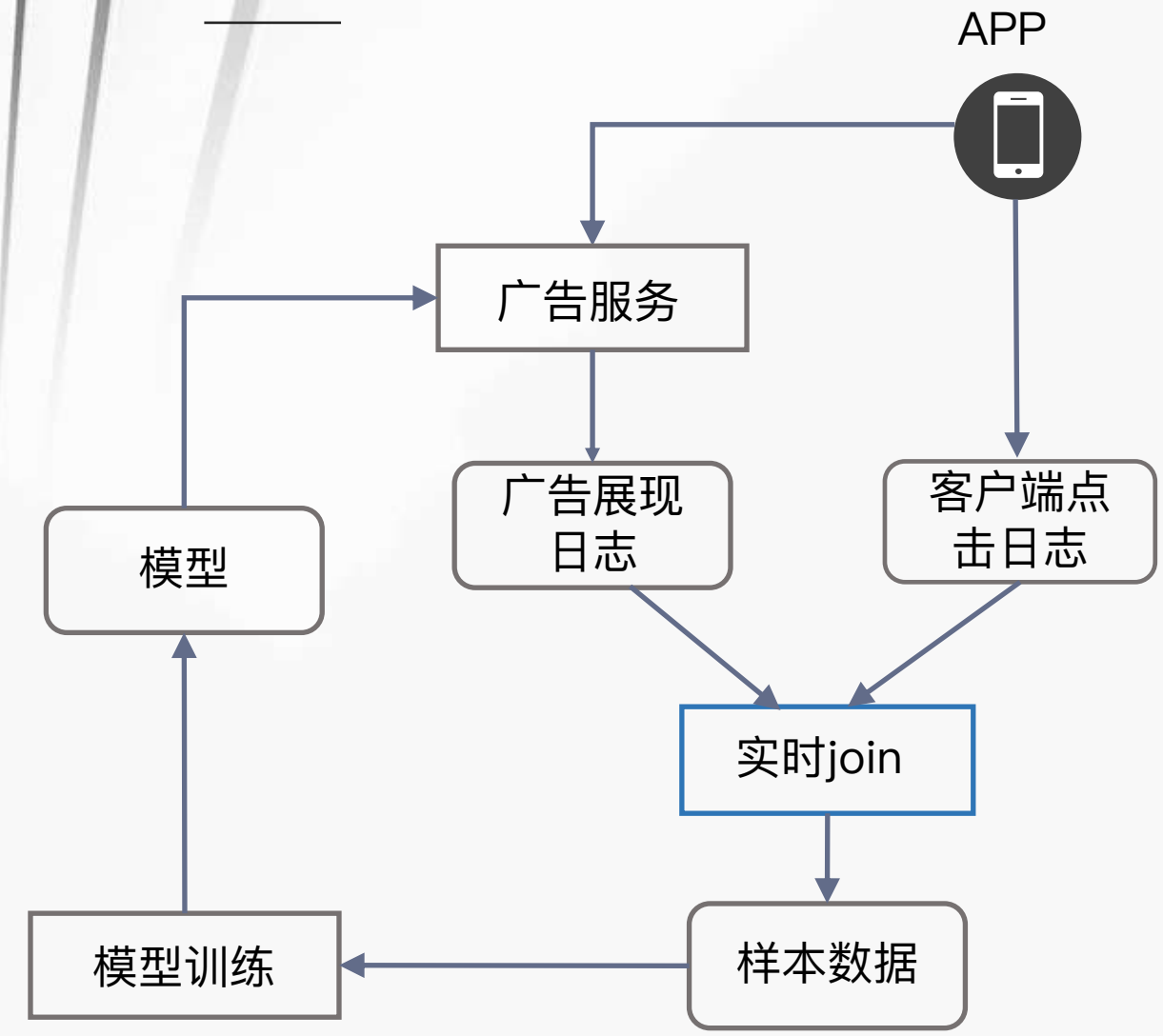
快手Flink技术演进

Apache Flink

快手Flink技术演进

- 场景优化
  - Interval join场景优化
- 稳定性改进
  - 数据源控速
  - JobManager稳定性
  - 作业频繁失败
- 平台建设

Interval Join应用场景



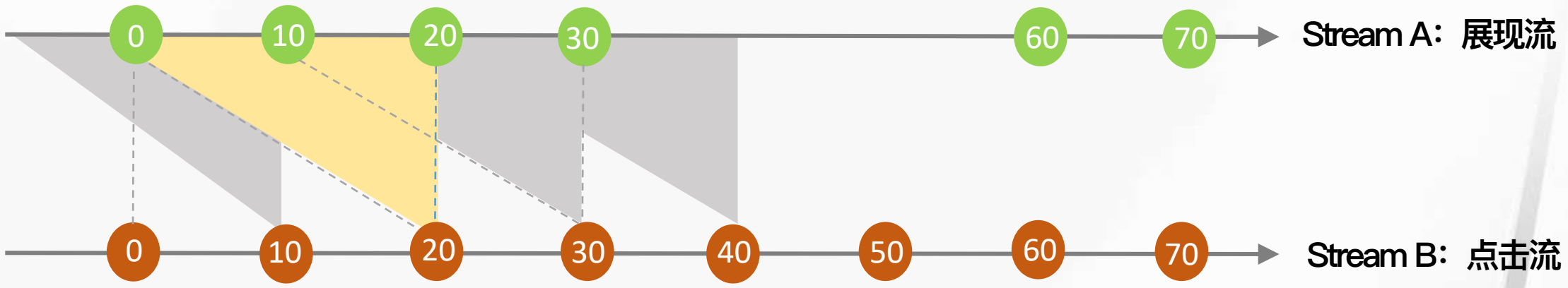
- 广告展现点击流实时join场景：
  - 有效点击：展现流以后20min的点击
  - Join逻辑：点击数据join过去20min展现
  - 数据量大：展现流20min数据1TB+
- 最初模式
  - 业务自己实现：kafka+redis
  - 实时性不高
  - 堆积机器、运维成本高
- Flink实现
  - Interval join完美切合
  - 实时性高
  - 维护成本低

Interval Join场景优化

Interval Join原理：

- 两条流数据缓存在state中
- 任一数据到达，获取对面流相应时间范围数据
- join到数据后应用joinFunction

```
KeydStreamA.
intervalJoin(KeyedStreamB)
.between(Time.minutes(0), Time.minutes(20))
.process (joinFunction)
```



b.Timestamp ∈ [a.timestamp, a.tmestamp+20]

Interval Join场景优化

状态存储策略选择

Backend方式	特点
FsStateBackend	State存储在内存 checkpoint时持久化到hdfs
RocksDBStateBackend ✓	State存储rocksdb 可增量checkpoint 适合超大state

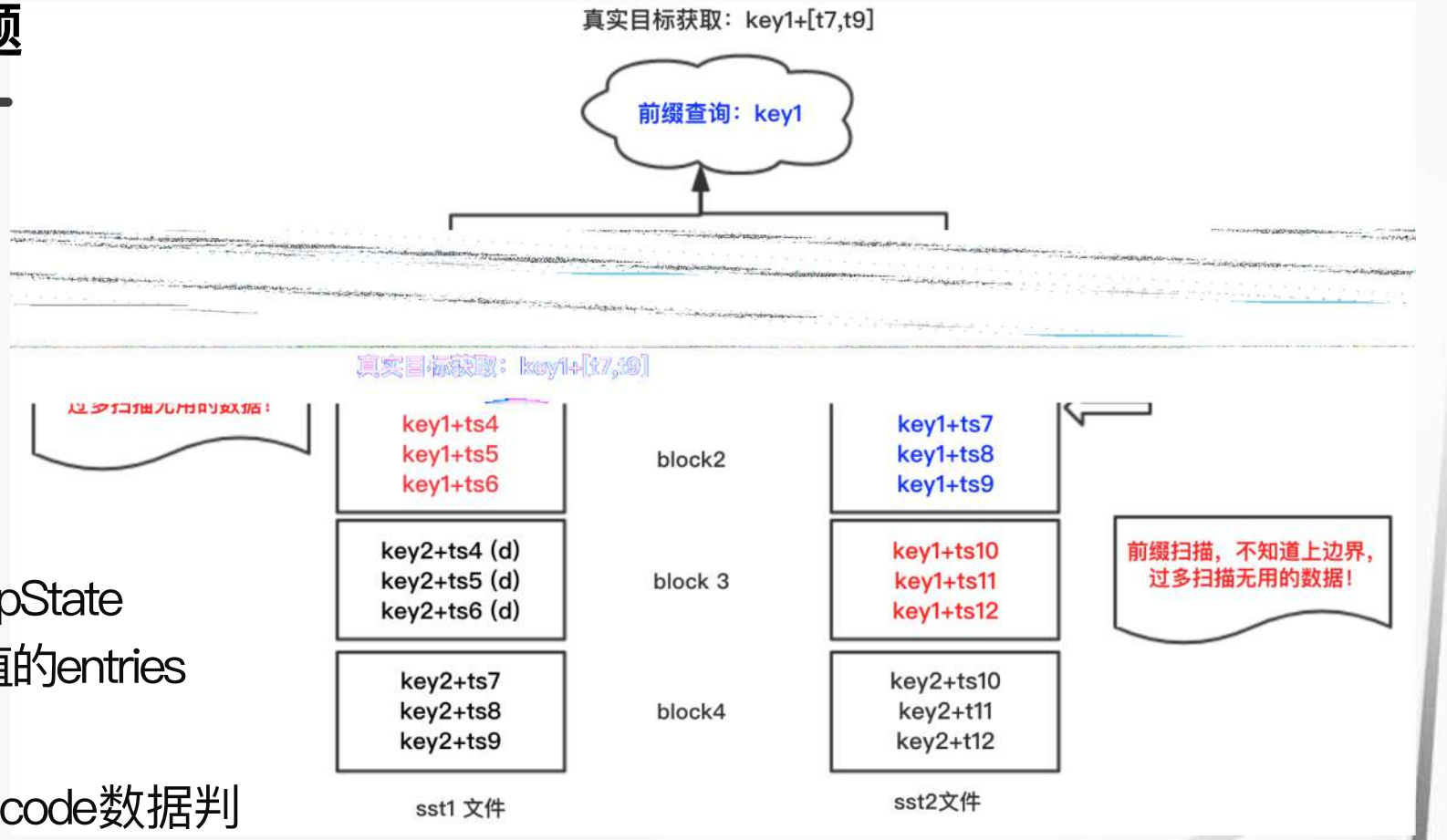
Rocksdb状态存储方式：

rowKey (keyGroupId+joinKey+ts)	cf1 (StreamA)	cf2 (StreamB)
1,key1,ts1	record1	record1’
2,key2,ts2	record2	record2’

Interval Join场景优化

Rocksdb访问性能问题

- 作业出现反压，吞吐下降
- 频繁rocksdb get
- 单线程CPU被打满



问题分析

- Flink内部实现RocksdbMapState
- 前缀扫描：获取某个key值的entries
- 扫描大量无效数据
- 数据缓存pagecache，decode数据判断是否删除消费cpu

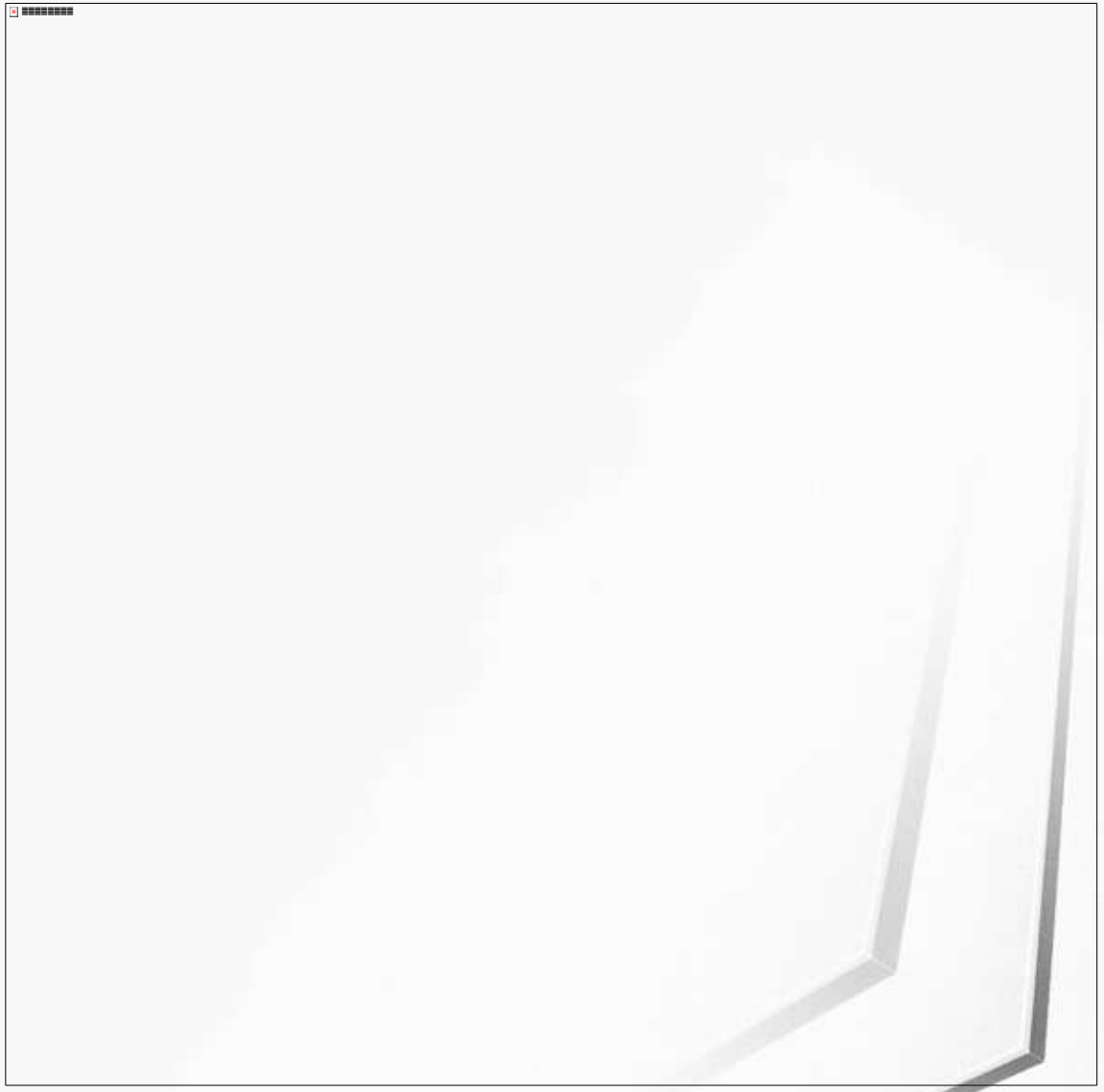
Interval Join场景优化

针对rockdb访问性能优化：

- 全key扫描： 拼出查询上下边界full key：keygroupId+join key+ts[lower, upper]
- 范围查询rocksdb：更加精确seek到上下边界，避免无效数据扫描和校验

◘ 效果：

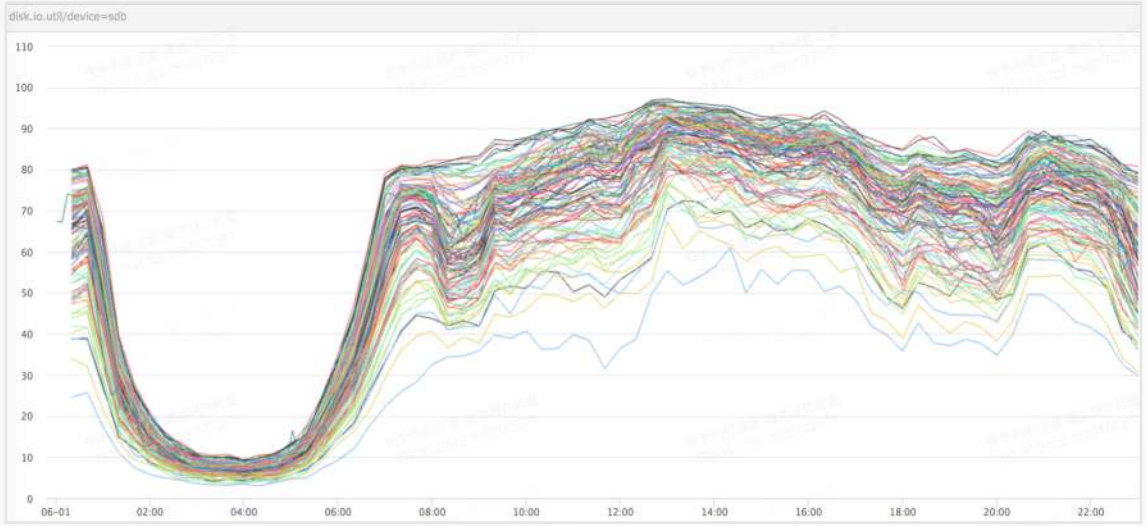
- 性能提升： p99查询性能提升10倍:nextkey由1000毫秒 ---> 100毫秒以内
- 作业吞吐反压问题解决



Interval Join场景优化

Rocksdb磁盘压力问题

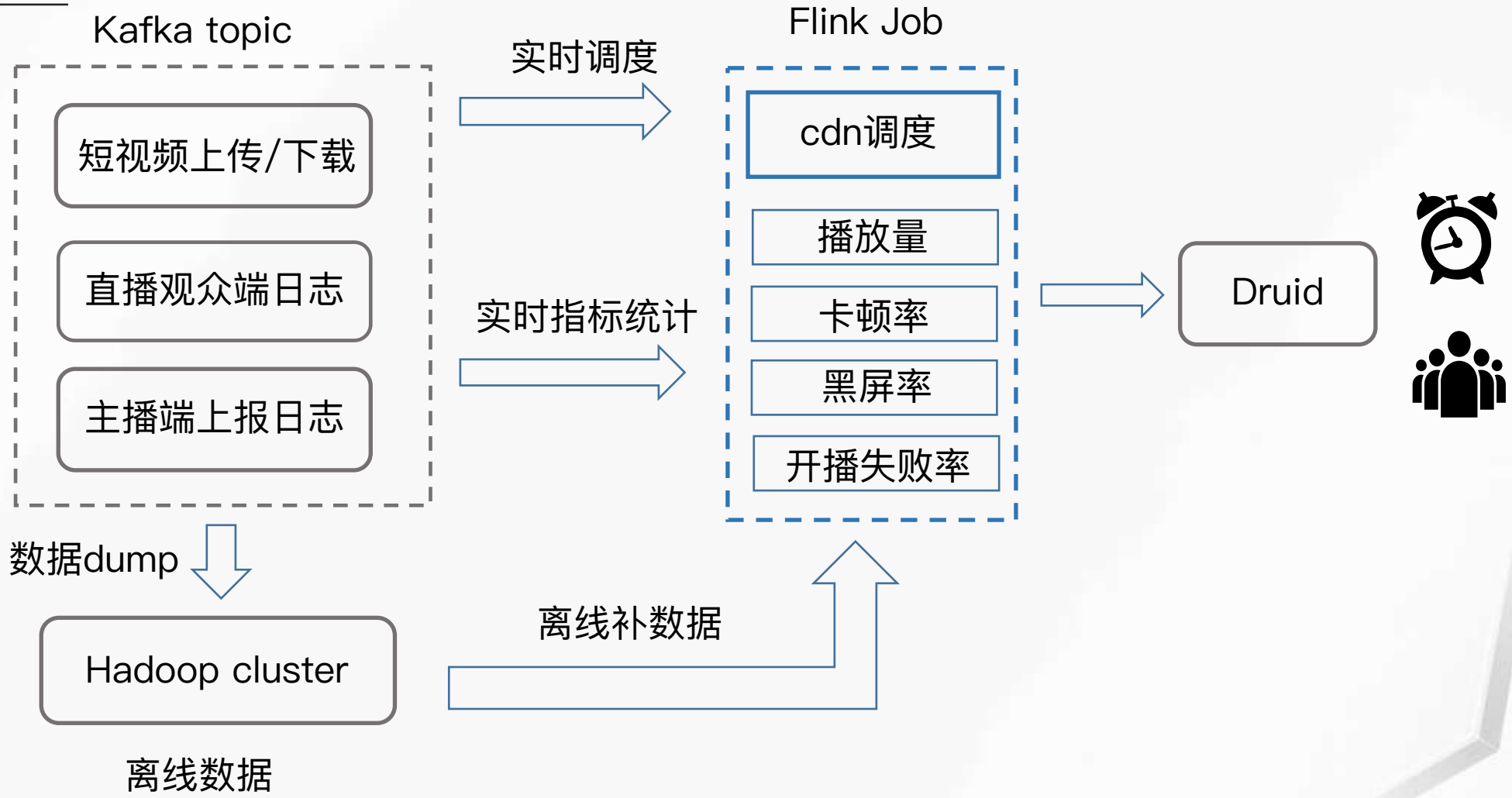
- 机器选型：计算型，大内存、单块hdd盘
- 单机器 4–5 container，使用一块hdd盘
- 高峰磁盘压力：disk uitl 90%，150Mb/s



优化：

- Rocksdb 参数调优：减少compaction IO 量
- Rocksdb配置套餐：新增large state 套餐
- 框架支持RocksdbBackend自定义各种rocksdb参数配置
- 未来计划：state考虑共享存储的方式

视频质量监控调度应用



短视频、直播质量监控和调度

稳定性改进–数据源控速

视频应用 历史数据读取问题：

- 作业DAG复杂：多个数据源读取数据
- 读取历史数据：作业失败从较早状态恢复
- 速度不可控：不同Source并发读数据速度不同
- Window类算子state堆积：作业性能变差、恢复失败
- 临时解决办法：重置groupid，从最新开始消费

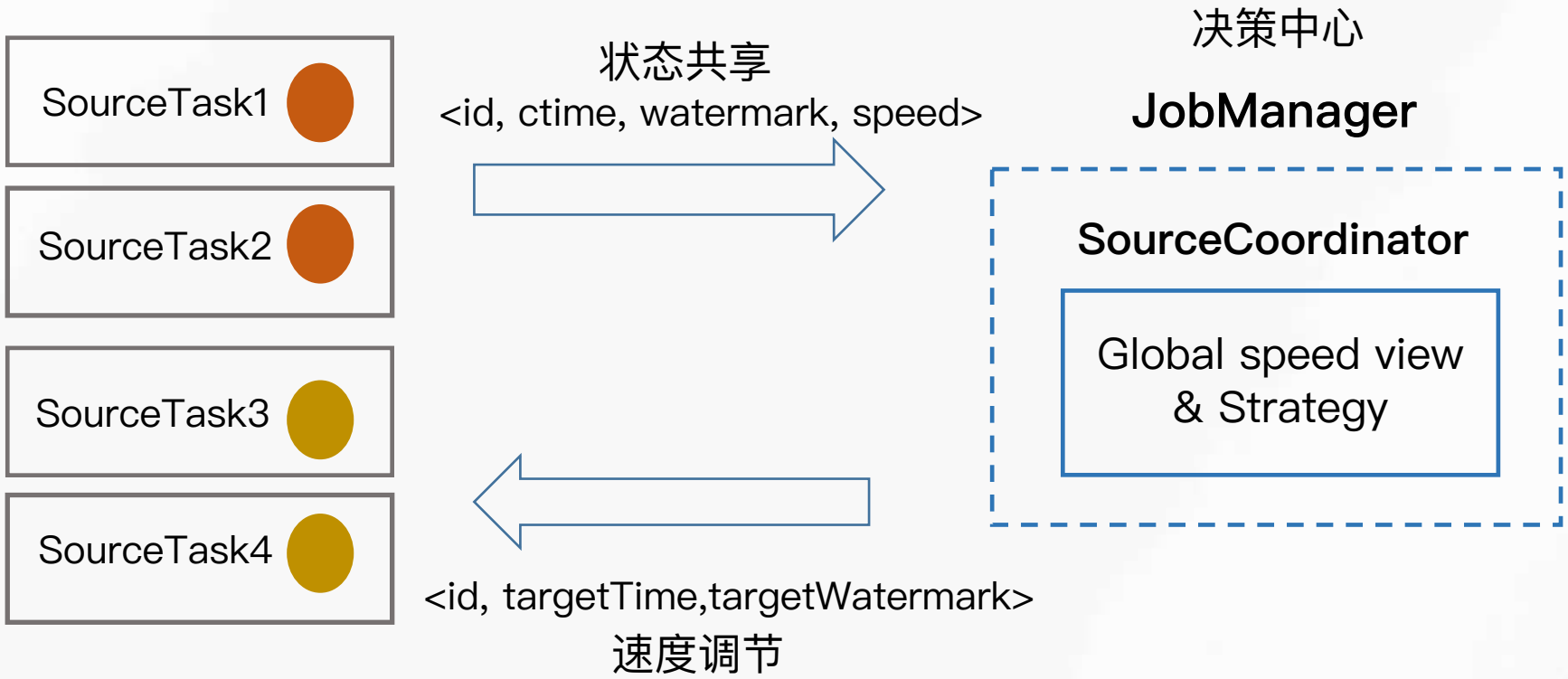
- 离线补数据：从不同hdfs文件读取，读取数据不可控

目的：

- 从源头控制多个Source并发读取速度

稳定性改进–数据源控速

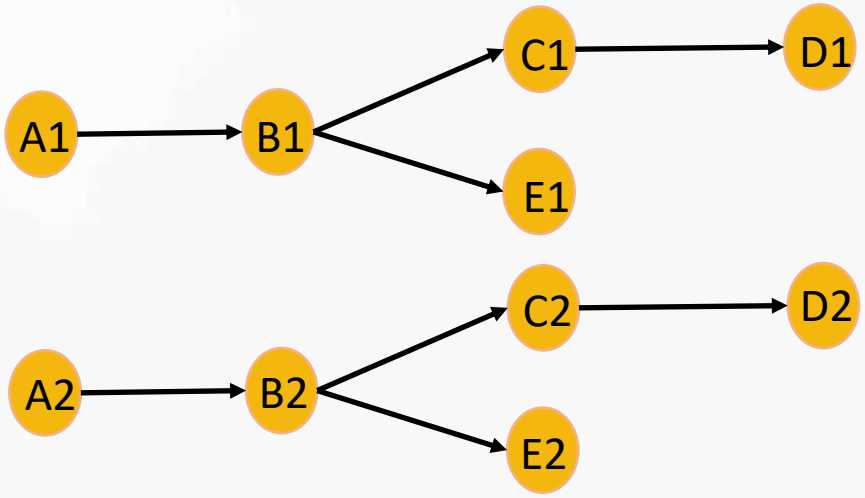
Source控速策略



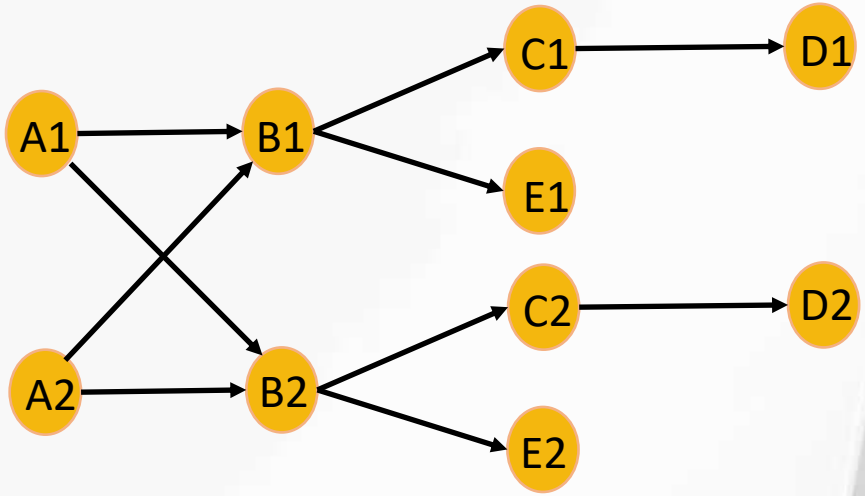
稳定性改进–数据源控速

Source控速策略

A1,A2相互独立，无需对齐



A1,A2非独立
下游节点接受A1，A2发送过来数据

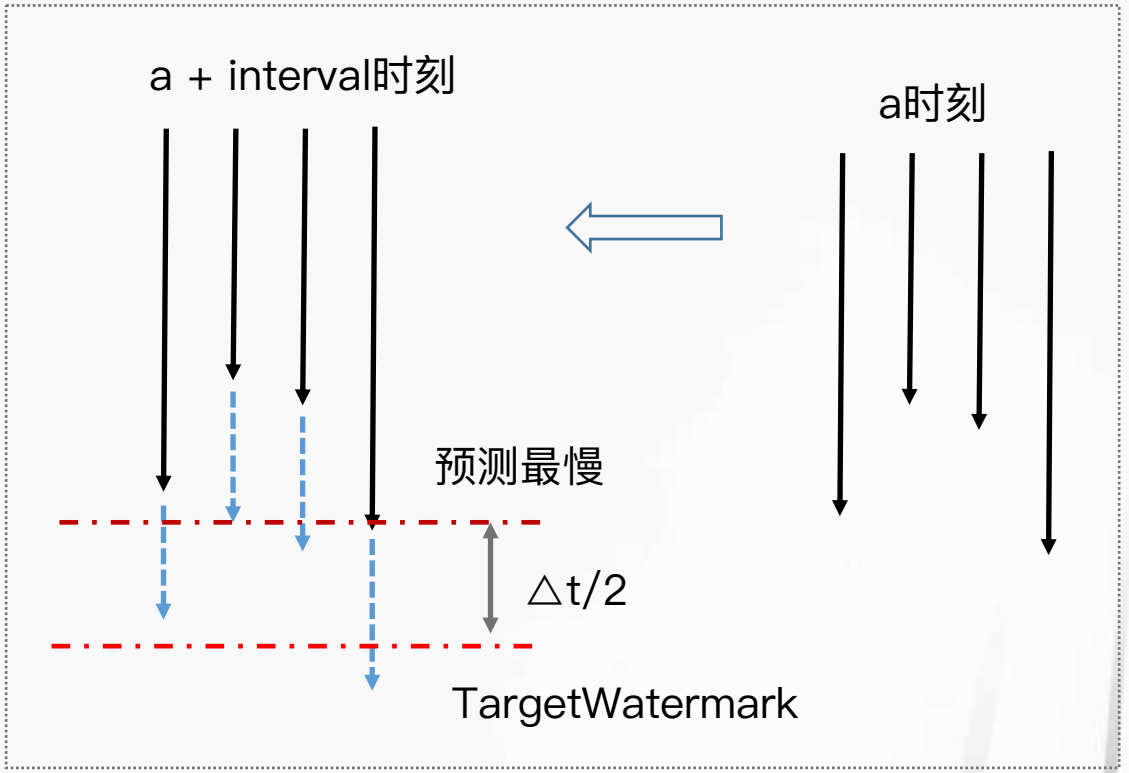


dag子图 source源之间不互相影响

稳定性改进–数据源控速

Source控速策略

- 共享状态
  - sourceTask定期汇报：默认10s
  - 汇报内容：<id, clocktime, watermark, speed>



- 协调中心：SourceCoordinator
  - 限速阈值： 最快并发watermark – 最慢并发watermark > △t（默认5min）
  - 全局预测：各并发targetWatermark= base + speed * time
  - 全局决策： targetWatermark = 预测最慢watermark + △t/2
  - 限速信息： <targetTime, targetWatermark>

稳定性改进–数据源控速

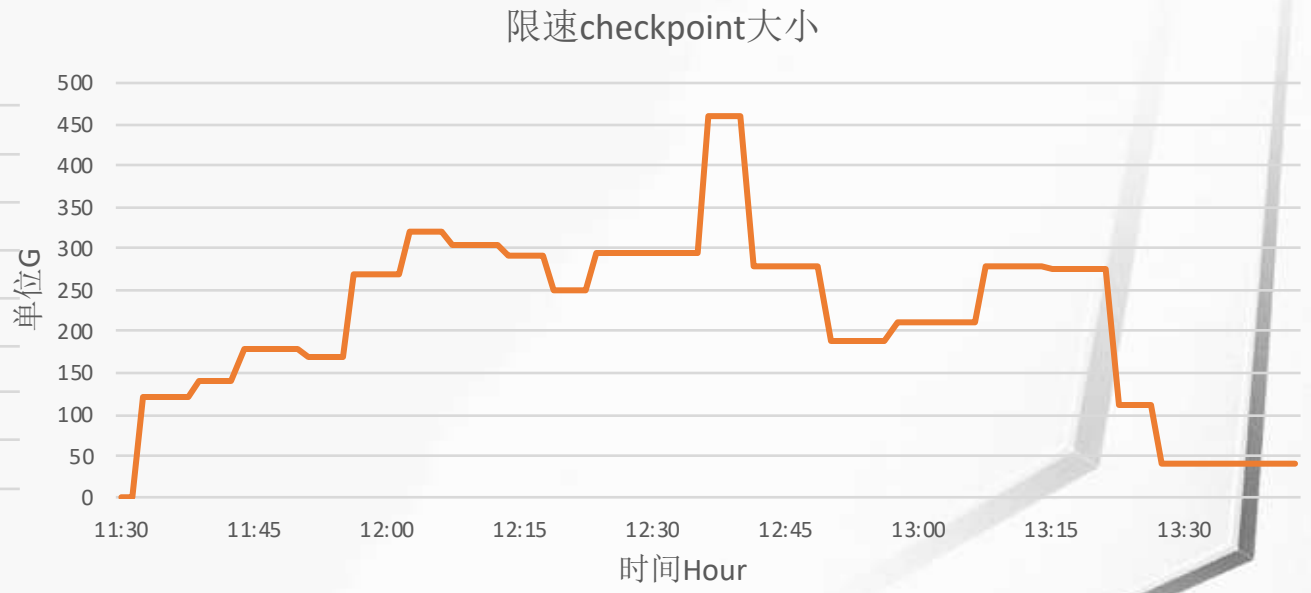
Source限速策略

- SourceTask：限速控制
  - 获取到限速信息：<targetTime, targetWatermark>
  - kafkaFetcher获取数据时，根据限速信息check当前进度，是否需要等待。
- 其他考虑
  - 时间属性：EventTime
  - 开关控制：是否开启source限速策略
  - dag子图 source源之间不互相影响
  - 是否会影响checkpoint barrier下发
  - 数据源发送速度不恒定，watermark突变情况

稳定性改进–数据源控速

Source控速结果

- 线上作业，kafka消费From earliest(2 days ago)
  - 不限速：state持续增大、挂掉
  - 限速：state可控、平稳追上最新数据



稳定性改进

JobManager 稳定性

◘ JobManager内存

- NameNode压力大，complete checkpoint路径删除慢
- Checkpoint path 在内存中堆积
- 删除策略：每删除一个文件，list目录判空，为空删除目录
- 优化删除逻辑：直接调用hdfs delete(path, false) 语义保持一致

Checkpoint 目录结构

```
chkdir/jobid/chk-id/file-1
                    file-2
                    …
                    file-20000
```

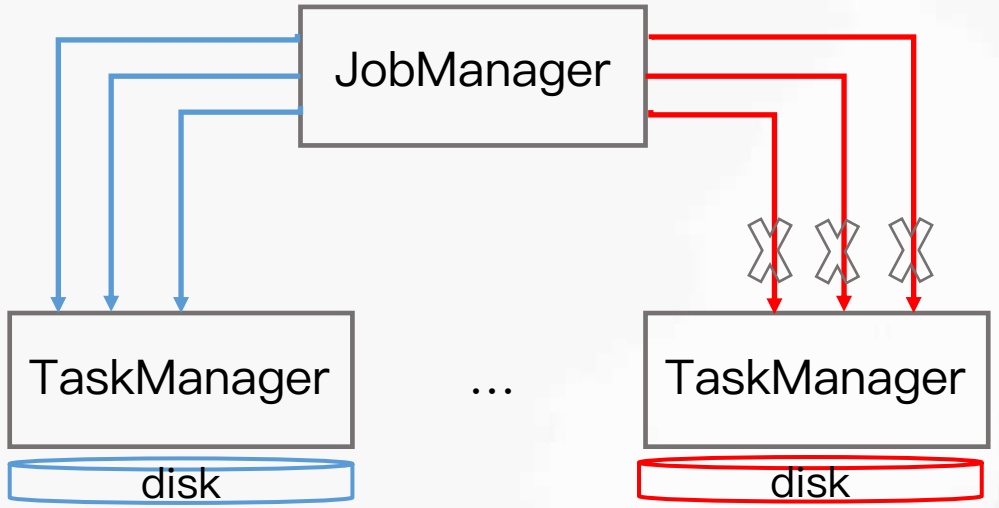
◘ JobManager GC慢

- Cgroup上线，影响调度、web ui卡
- jobmanager g1 gc过程变慢
- AppMaster 申请cpu个数硬编码为1
- 解决：AppMaster 申请cpu个数可参数化配置

稳定性改进

机器故障造成作业频繁失败

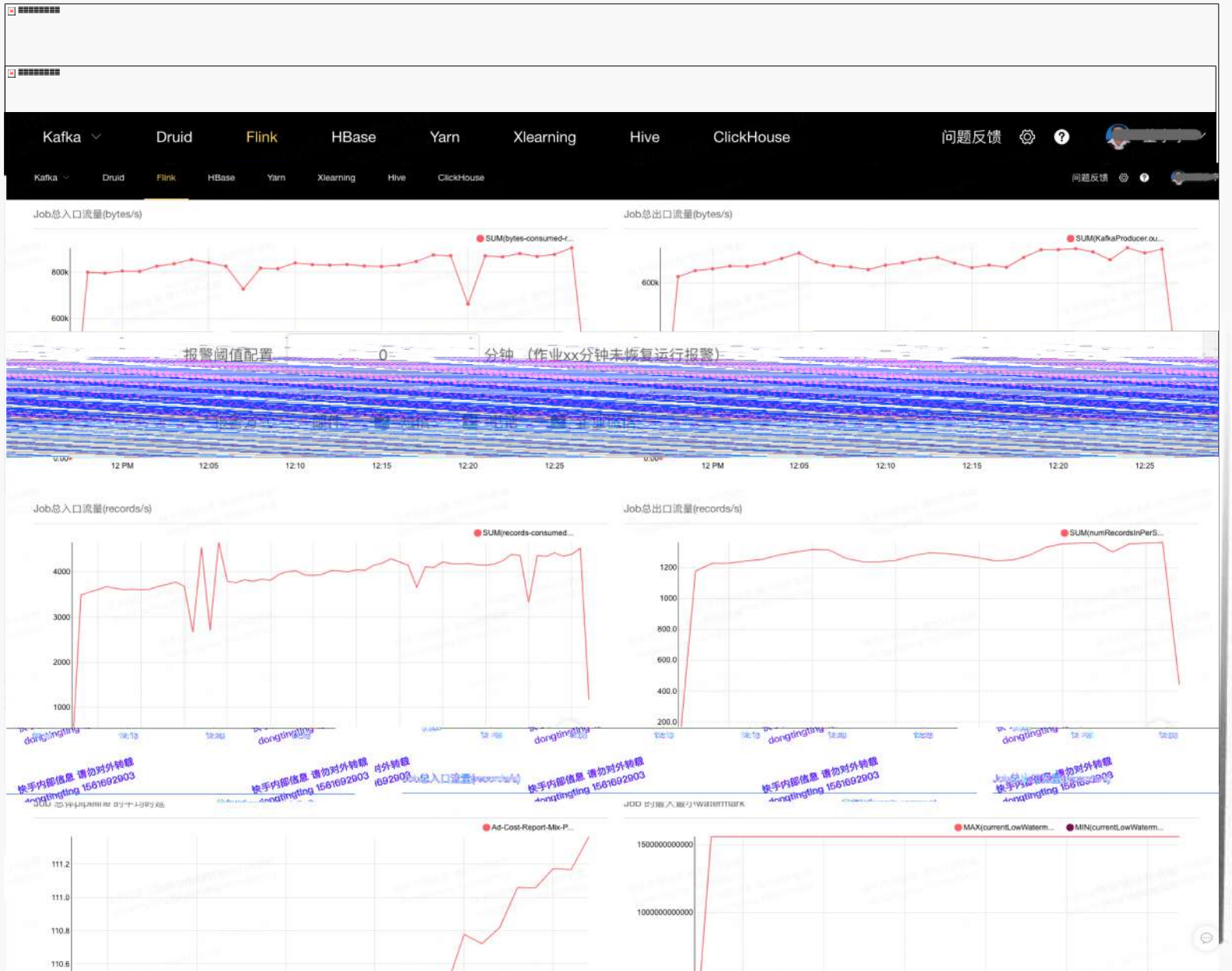
- 磁盘问题导致作业持续调度失败
  - TaskManager不感知磁盘健康状况
- 作业频繁调度到有问题的机器
  - TaskManager在某台机器上频繁出core



- 问题解决
  - TaskManager增加DiskChecker磁盘健康检查
  - 软黑名单：尽量不调度到某些机器

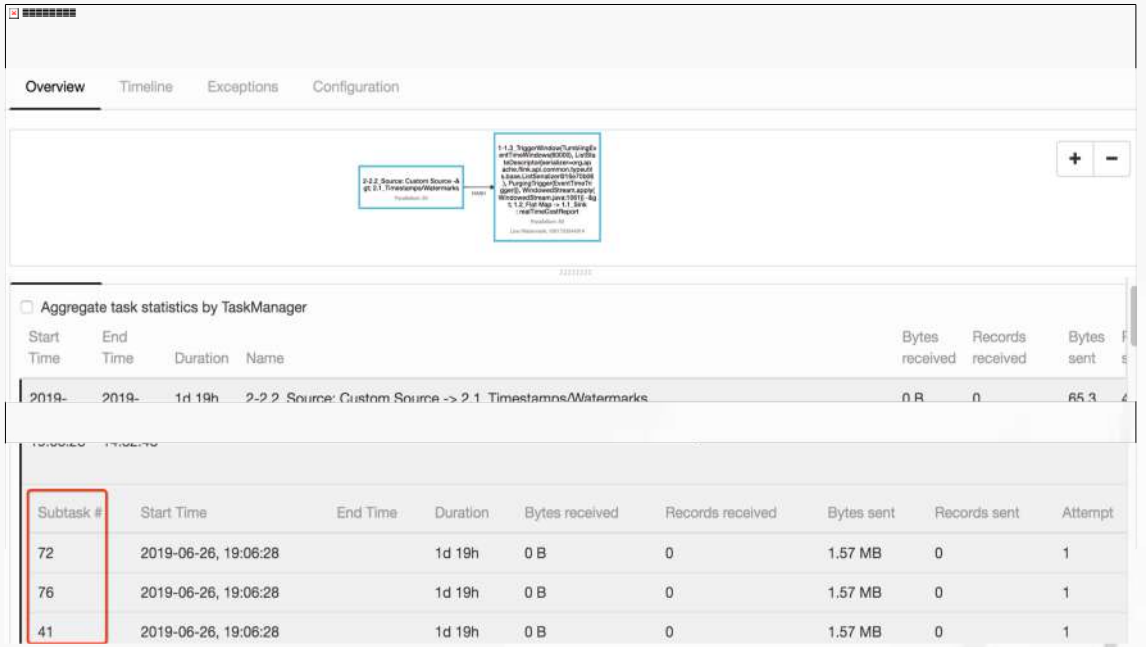
平台化建设

- 平台建设
  - 青藤作业托管平台
  - 作业操作：提交、停止
  - 作业管理：报警、自动拉起
  - 作业metirc查看



平台化建设

- 问题定位流程优化
  - 所有metric入druid+superset分析
  - Web ui 支持实时打印jstack
  - Web dag为各Vertex增加序号
  - Subtask信息中增加各并发subtaskId
  - 异常信息增加机器宕机信息提示
  - 新增多种metirc



Apache Flink

03

未来计划

Apache Flink

未来计划

Flink SQL

- 减少用户开发成本
- 实时数仓

资源优化

- 合理的资源申请
- 提升资源利用率

Apache Flink

THANKS

【2群】Apache Flink C...

扫一扫群二维码，立刻加入该群。

快手数据架构