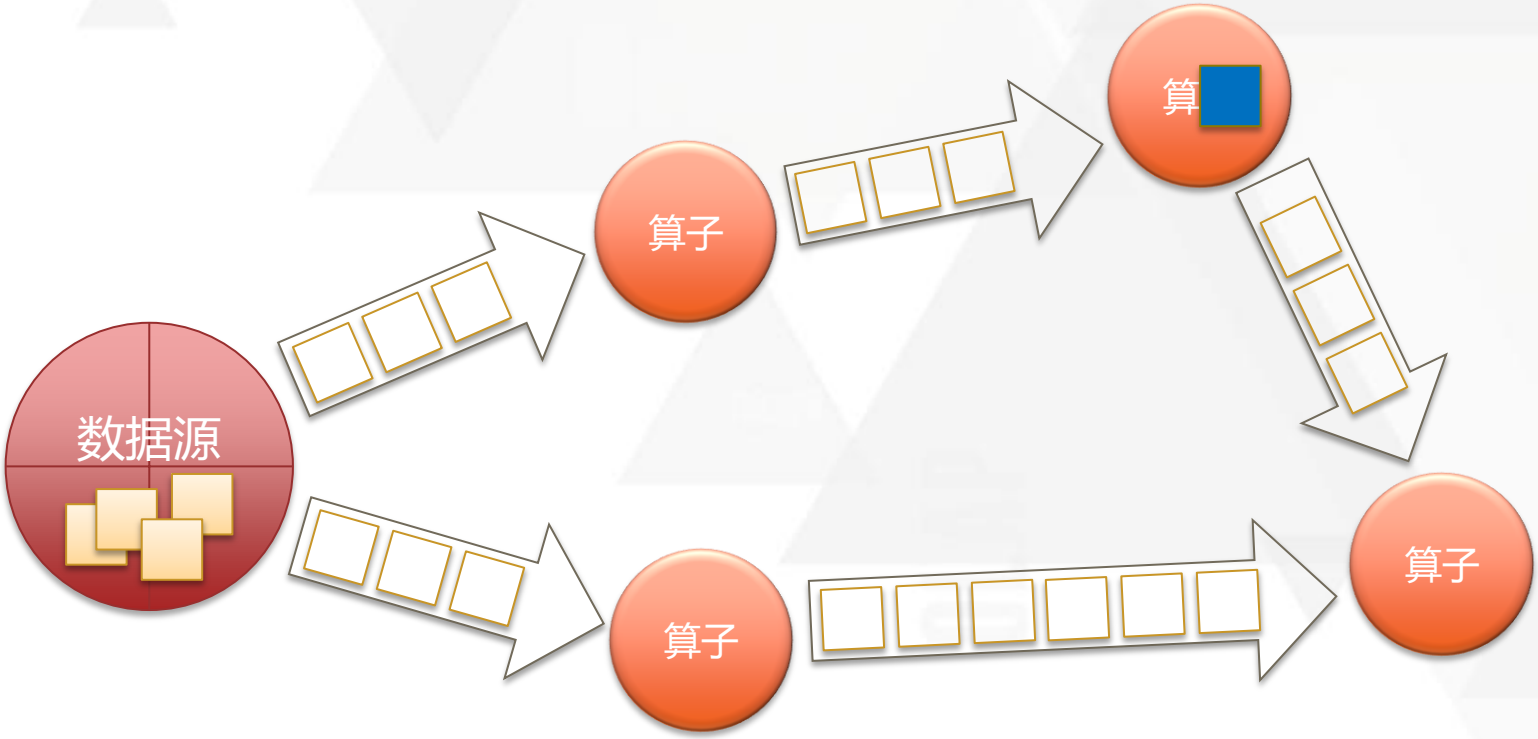
分布式流处理技术

禹晓辉
山东大学计算机科学与技术学院

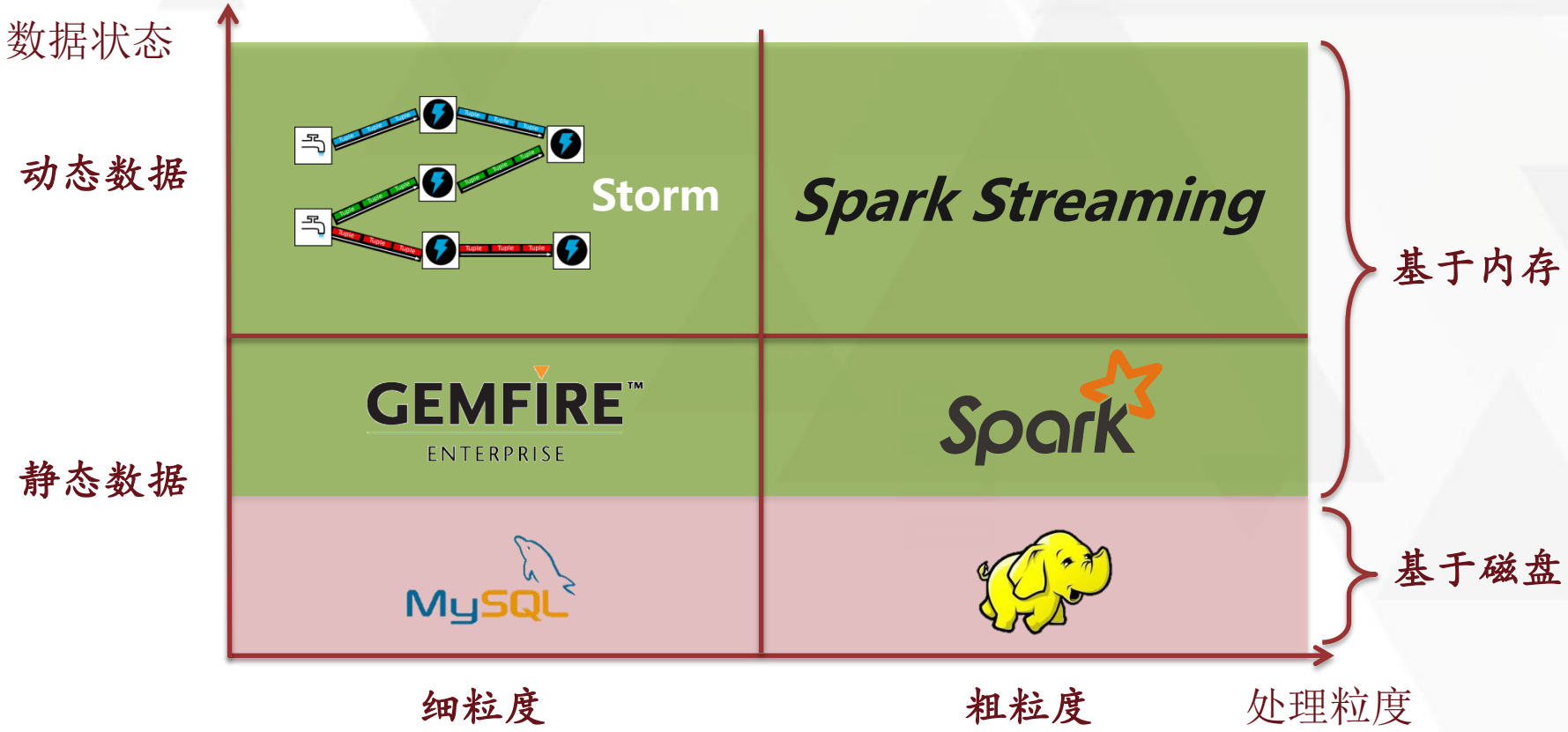
流处理



内容

- 大数据处理模式
- 流处理技术发展
- 分布式流处理系统剖析
- 分布式流处理应用实例

大数据处理模式


数据状态
动态数据
静态数据
Storm
Spark Streaming
GEMFIRE™
ENTERPRISE
Spark
MySQL
基于内存
基于磁盘
细粒度
粗粒度
处理粒度

内容

- 大数据处理模式
- 流处理技术发展
- 分布式流处理系统剖析
- 分布式流处理应用实例

流处理技术发展

	数据流管理系统	分布式化	分布式流处理系统
	Aurora	Medusa	S4
实时数据库	STREAM	Flux	Storm
主动数据库	TelegraphCQ	Borealis	Samza
信息过滤系统	StreamBase	……	……
	……		

20世纪末 **21世纪初** **2010年至今** 时间

内容

- 大数据处理模式
- 流处理技术发展
- 分布式流处理系统剖析
- 分布式流处理应用实例

分布式流处理系统剖析

分布式流处理系统剖析 - 数据模型

处理单元

连续不断的序列

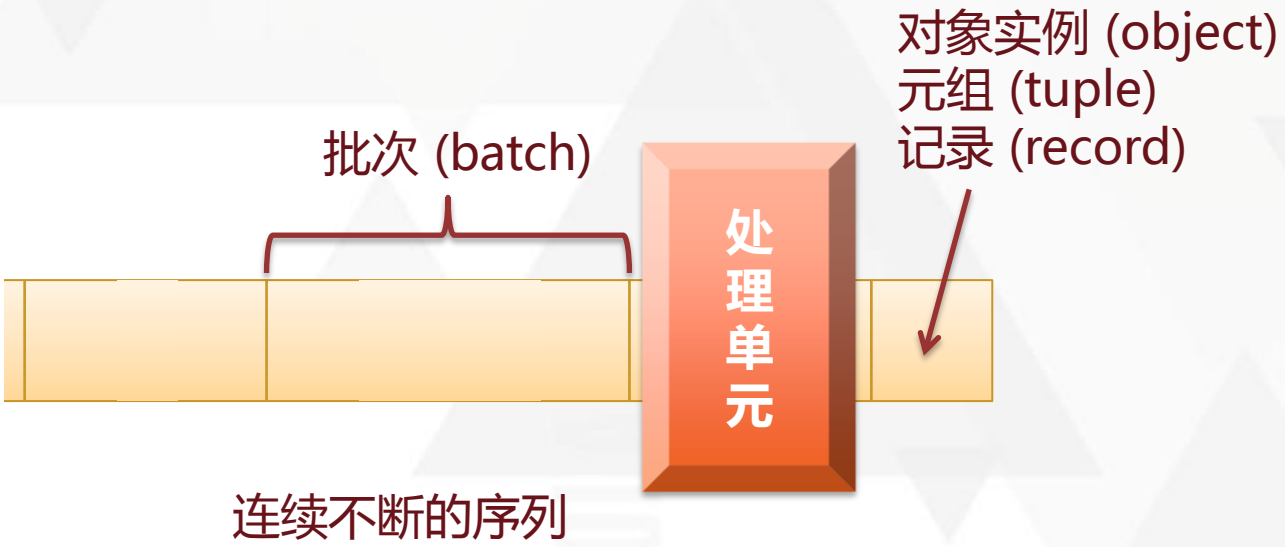
分布式流处理系统剖析 - 数据模型

批次模型

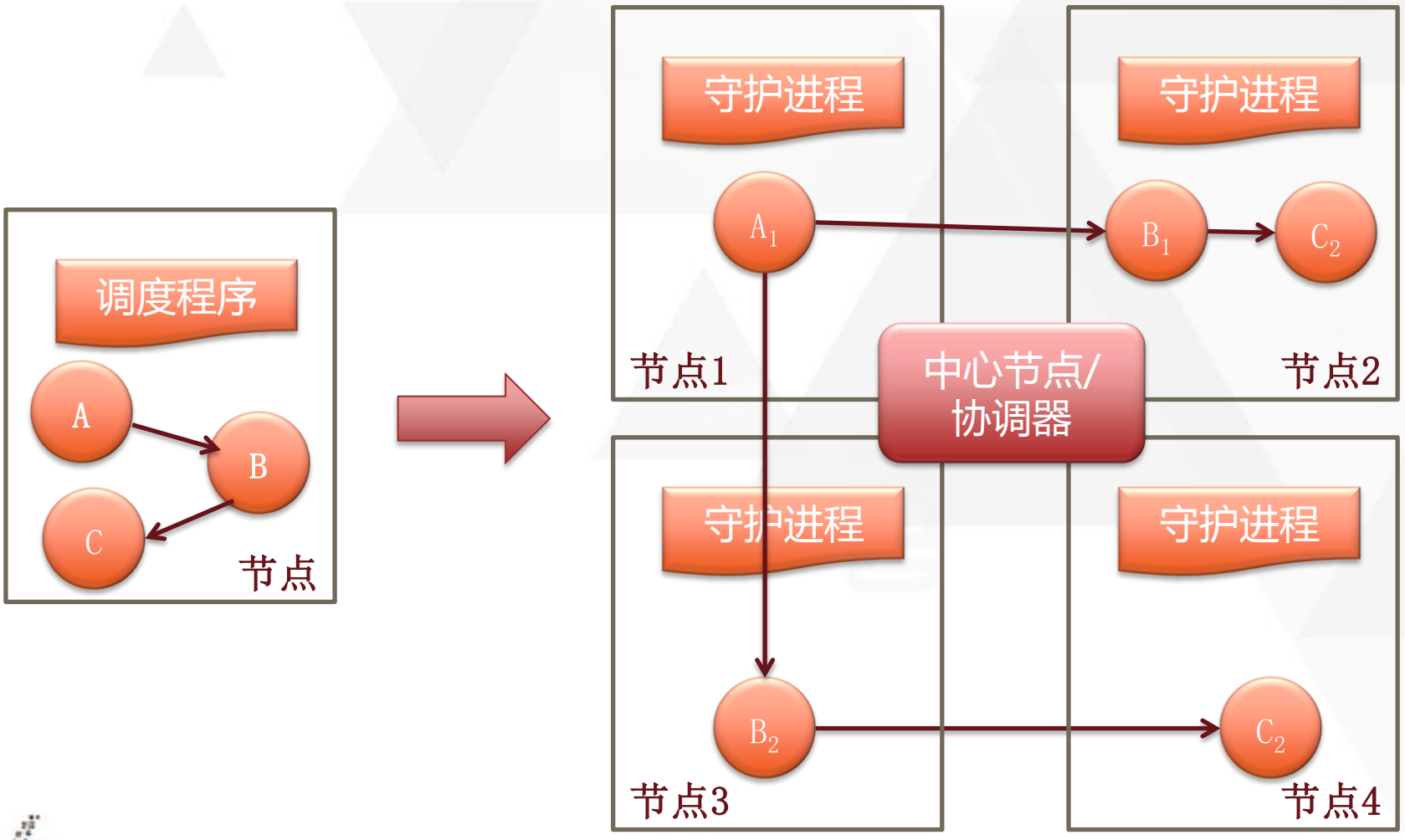
减少传输成本

降低容错难度

增加处理延迟

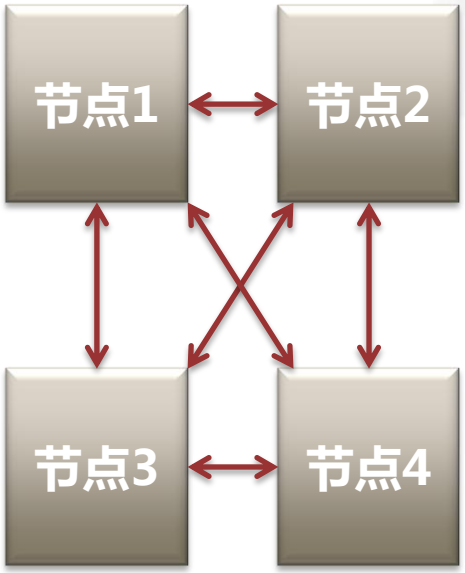


分布式流处理系统剖析 - 系统架构



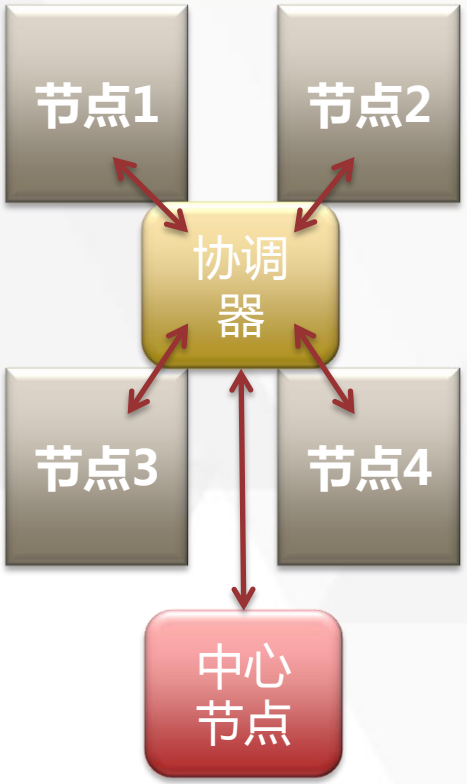
分布式流处理系统剖析 - 系统架构

P2P



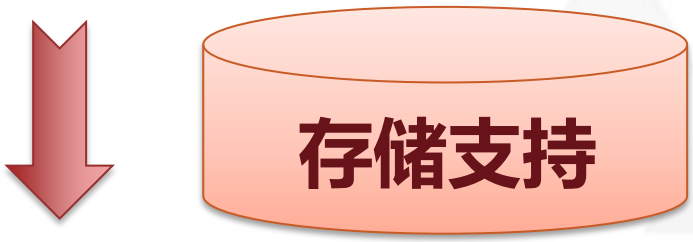
中心化

弱中心化



分布式流处理系统剖析 - 存储管理

“一过性” + 0故障概率 = 可靠保障



“可重复性” + ε故障概率 = 可靠保障

分布式流处理系统剖析 - 存储管理

> accumulated state for each transaction (like the partial counts).
>
> Finally, another thing to note is that transactional topologies require a source queue that can replay an exact batch of messages. Technologies like Kestrel can't do this. Apache Kafka is a perfect fit for this kind of spout, and storm-kafka contains a transactional spout implementation for Kafka.

log.flush.interval.messages	Long.MaxValue
log.flush.scheduler.interval.ms	Long.MaxValue
log.flush.interval.ms	Long.MaxValue

专(tuī)注(xiè)核(zé)心(rèn)

分布式流处理系统设计 - 存储管理

数据	说明	处理方式	可能改进
元数据	节点状态、任务信息、负载情况……	ZooKeeper	null
原始数据	系统接入的数据，如：句子	上游组件（消息队列）	集成可靠存储
衍生数据	计算产生的中间或最终结果，如：句子中某词出现的频数	内存（最终结果可能写外部数据库）	持久化接口（共享存储）

分布式流处理系统剖析 - 语义保障

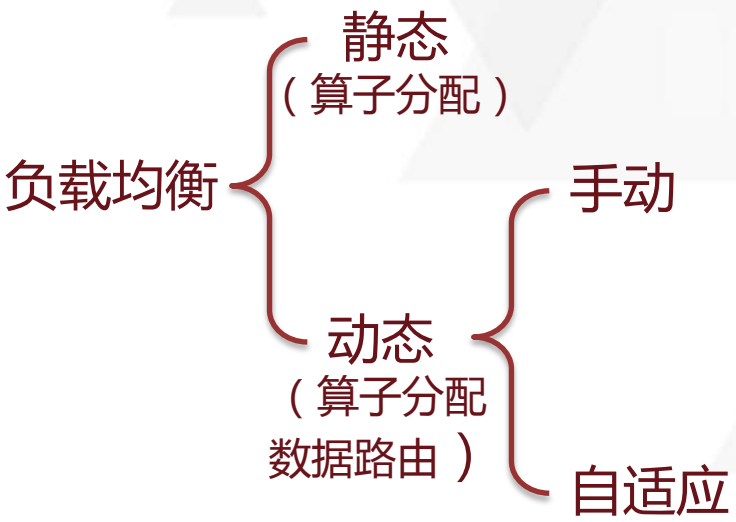
语义	应用场景	实现方式
至多一次	粗略log分析、温度报警	不重发
至少一次	一切幂等操作	原始数据可重复
精确一次	数目敏感应用、金融相关	基于至少一次，记录log

分布式流处理系统剖析 - 语义保障

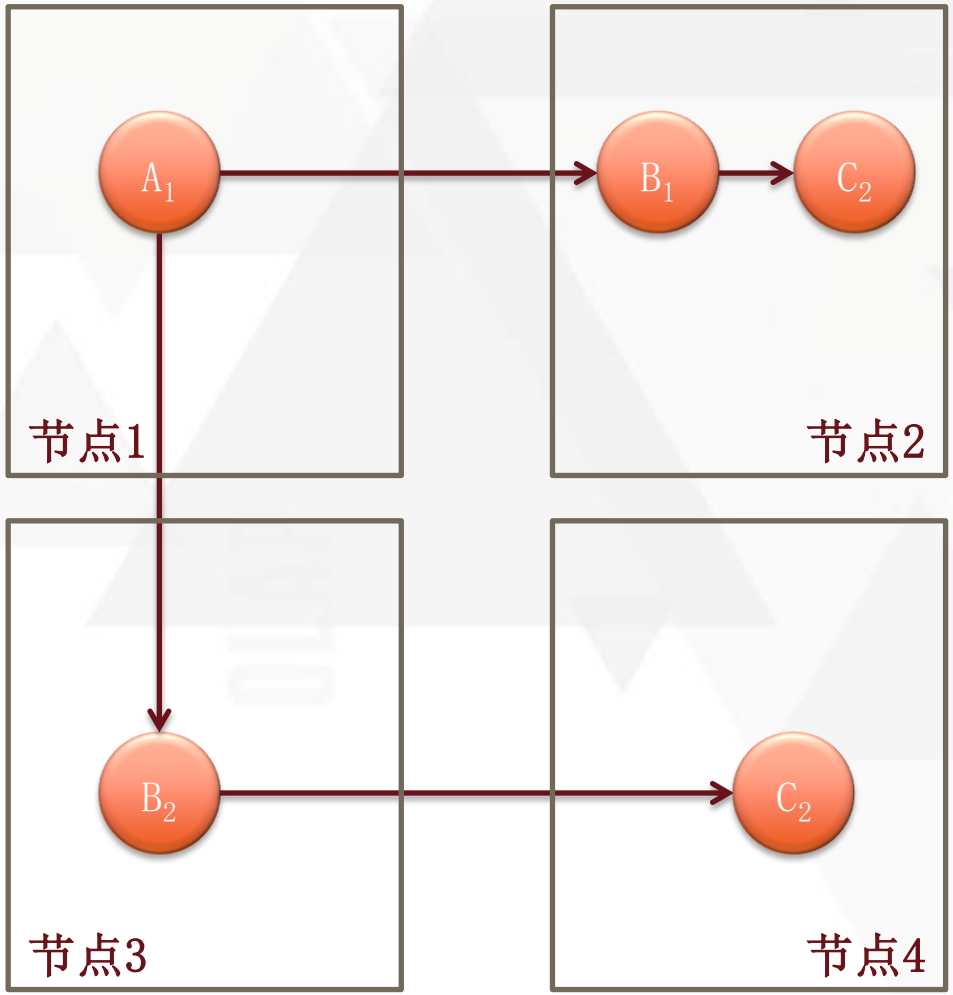
1、输出结果　2、记log

3、通知上游不要重发

分布式流处理系统剖析 - 负载控制



算子分配、数据路由算法

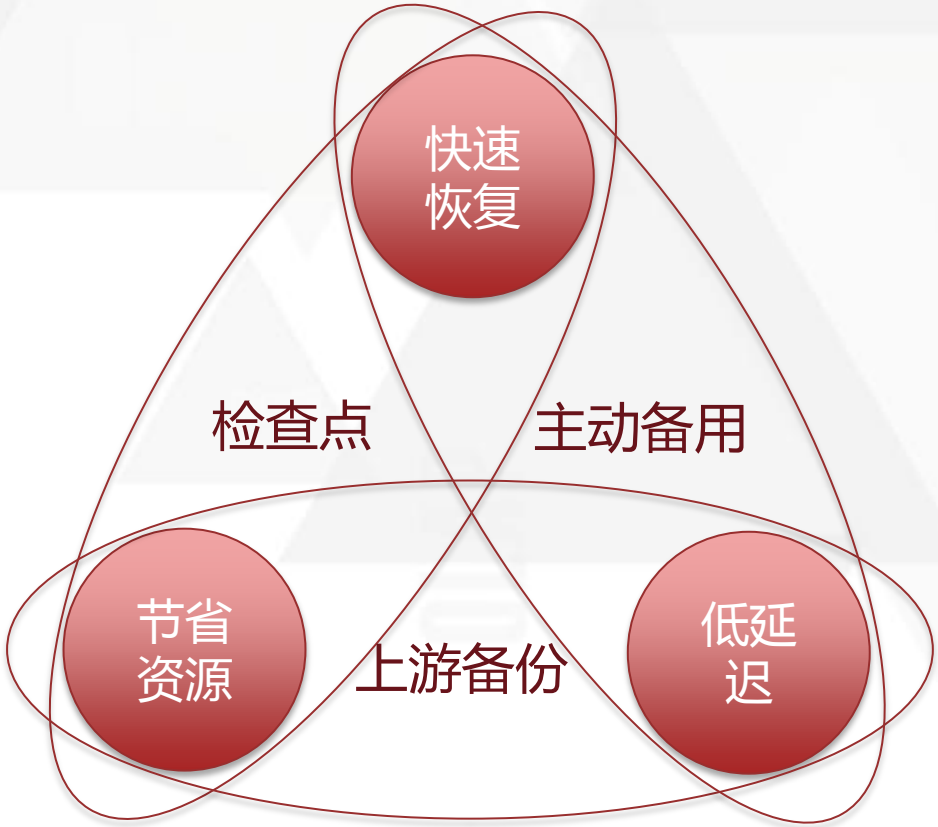


分布式流处理系统剖析 - 系统容错

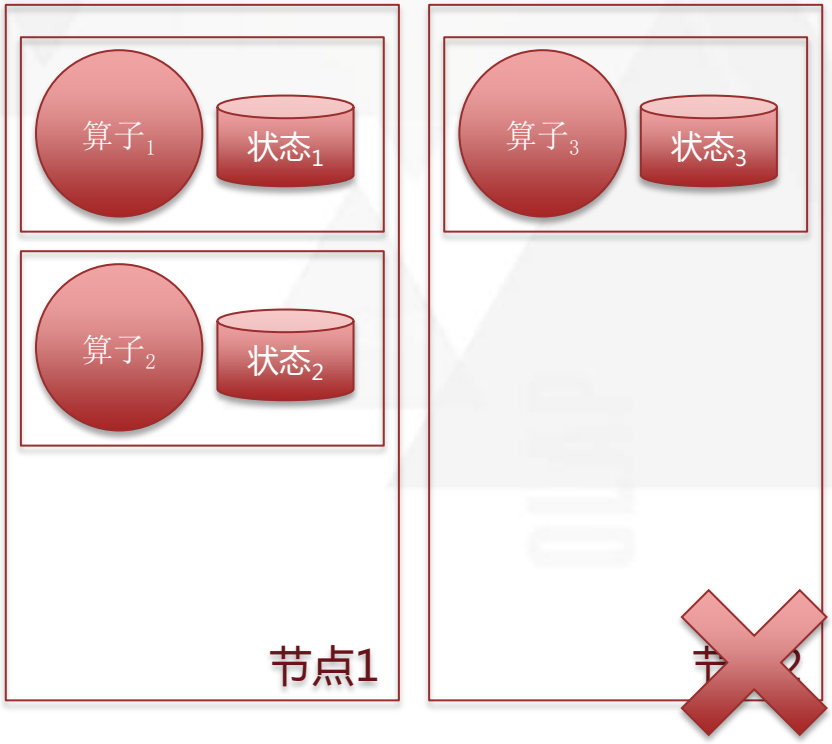
恢复级别	至多一次	至少一次	精确一次
精确恢复	是	是	是
回滚恢复	可能	可能	可能
有损恢复	可能	***否***	***否***

分布式流处理系统剖析 - 系统容错

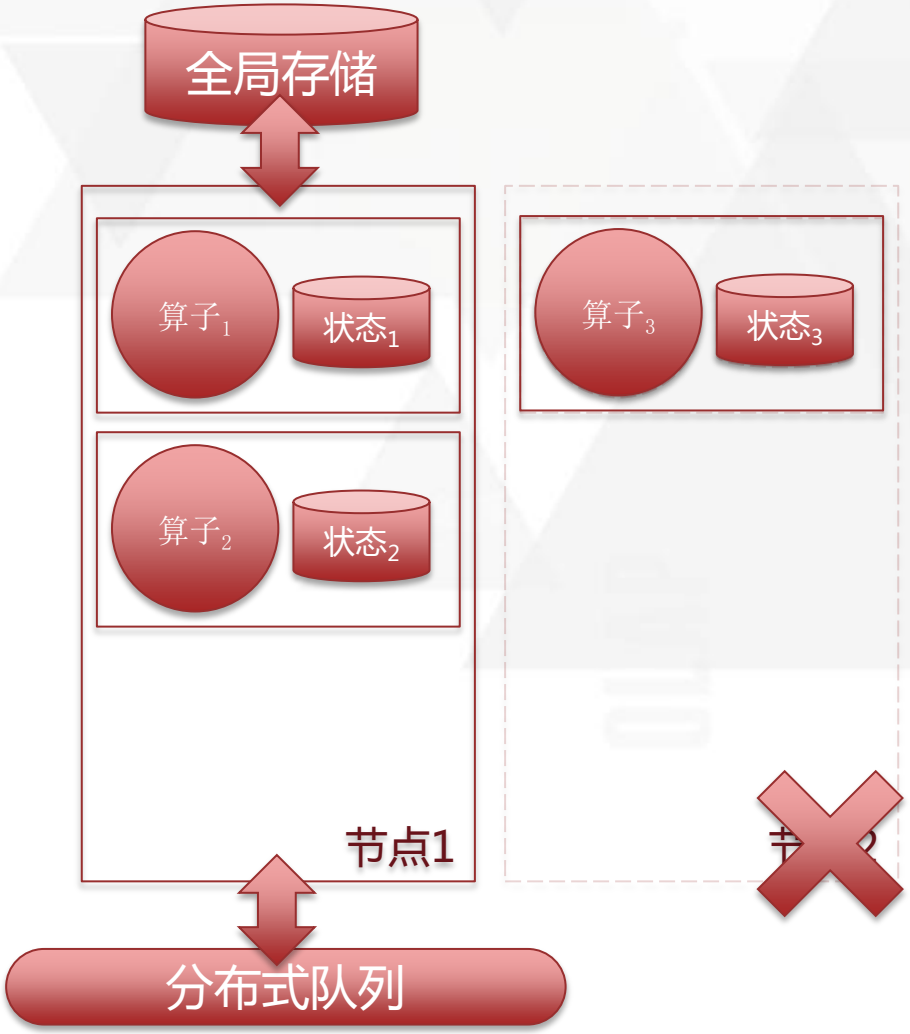
冗余



分布式流处理系统剖析 - 系统容错



分布式流处理系统剖析 - 系统容错



分布式流处理系统剖析 - 其他问题

- 高可用性（HA）
  - 高延迟
  - 过度频繁负载调度
  - 语义保障失误造成崩溃
  - 故障恢复时间过长
- 语言
  - 硬编码 ➔ Query/Manipulation Language
  - RDD Transformation

内容

- 大数据处理模式
- 流处理技术发展
- 分布式流处理系统剖析
- 分布式流处理应用实例

分布式流处理应用实例

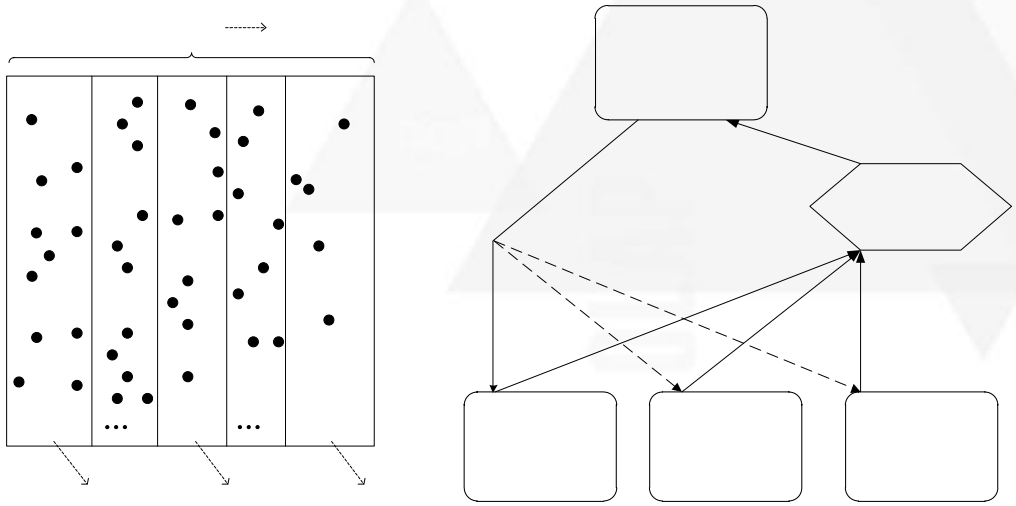
- 针对“**海量**”、“**高速**”数据进行较**复杂**处理，**低延迟**
  - 分布式时空K近邻搜索
  - 频繁伴随模式发现
  - 实时微博搜索
  - 流处理+批处理：TariDB

分布式时空K近邻搜索

- 问题
  - 海量的时空数据和大规模的并发搜索
- 难点
  - 集中式的K近邻搜索算法难以应对时空大数据

分布式时空K近邻搜索

基于主从分布式模型，通过建立分布式的动态Strip索引结构（DSI），实现对海量数据的分布式实时索引；设计分布式搜索算法（DKNN），通过最多两次迭代计算，得到准确的K近邻搜索结果。



Z. Yu, X. Yu, Y. Liu, K. Q. Pu. *Scalable Distributed Processing of K Nearest Neighbor Queries over Moving Objects*. In TKDE, 2014.

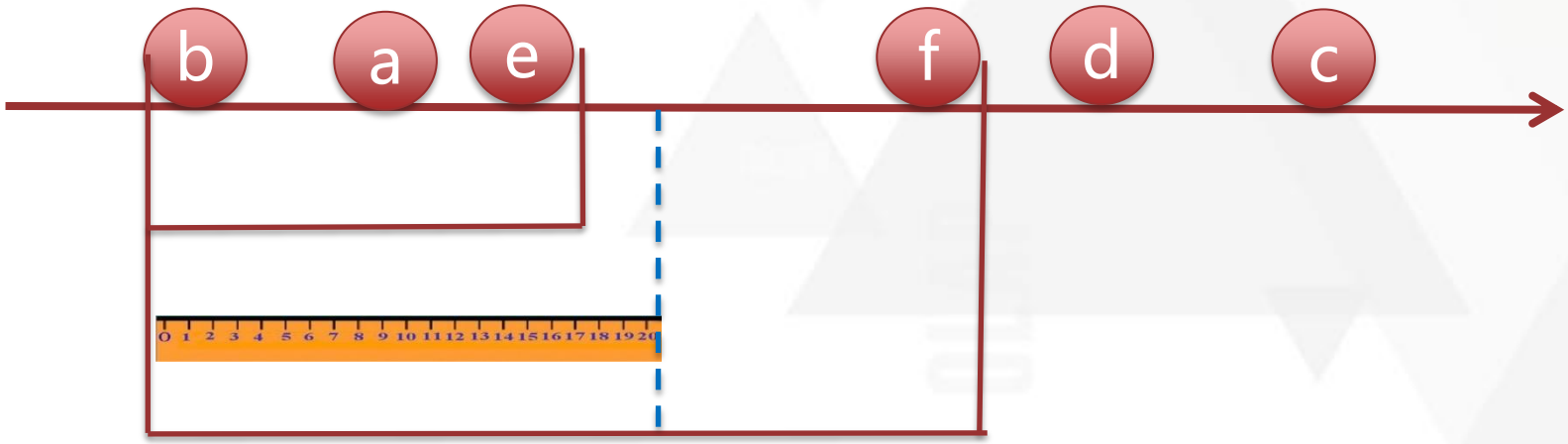
频繁伴随模式发现

- 问题：
  - 一组对象较短时间内在某个数据流连续出现
  - 该组对象之后一段时间内在多个数据流上以同样的方式出现
  - 实时发现多个数据流所有的频繁伴随模式
- 难点：
  - 涉及多个流数据复杂关系的比较分析
  - 流数据快速到达且连续变化，需要实时返回结果

频繁伴随模式发现

将流数据划分成Segment，将问题进行简化。

建立索引Seg-tree高效索引segment，并支持频繁伴随模式发现



Z. Yu, X. Yu, Y. Liu, W. Li, and J. Pei. *Mining Frequent Co-occurrence Patterns across Multiple Data Streams*. In EDBT 2015.

实时微博搜索

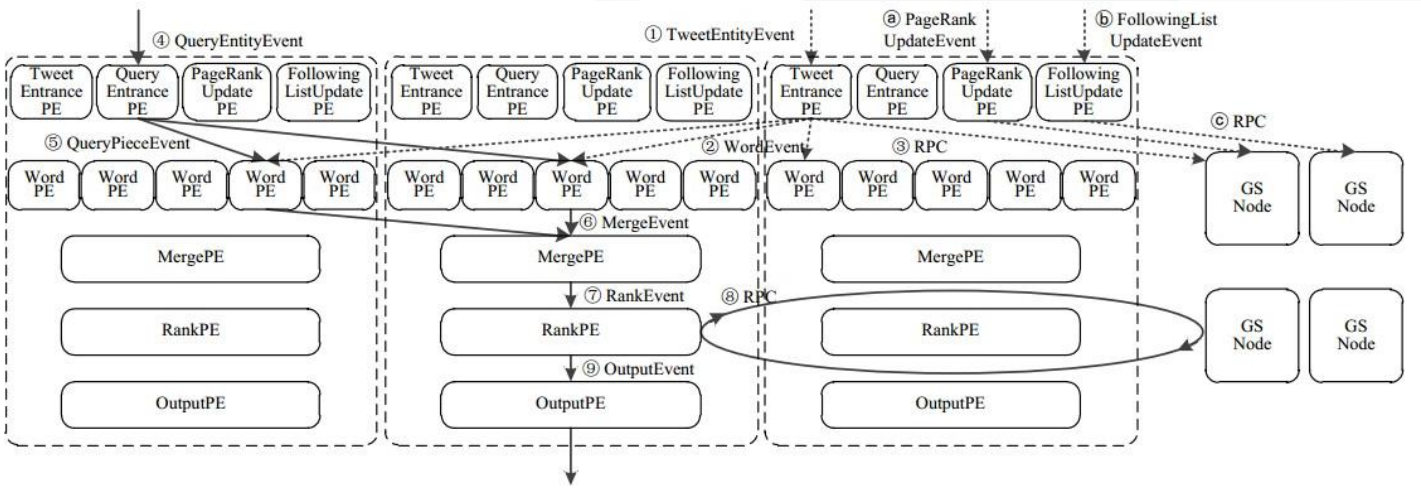
- 问题
  - 微博发布后，应立即可搜
  - 支持各种结果排序策略
- 难点
  - 微博数据呈现海量高速的特点
    - Twitter峰值25000条/秒 (2011年)
  - 基于内存的高可用

实时微博搜索

基于S4构建分布式微博搜索系统

1.通过同步冗余节点的方式实现热备

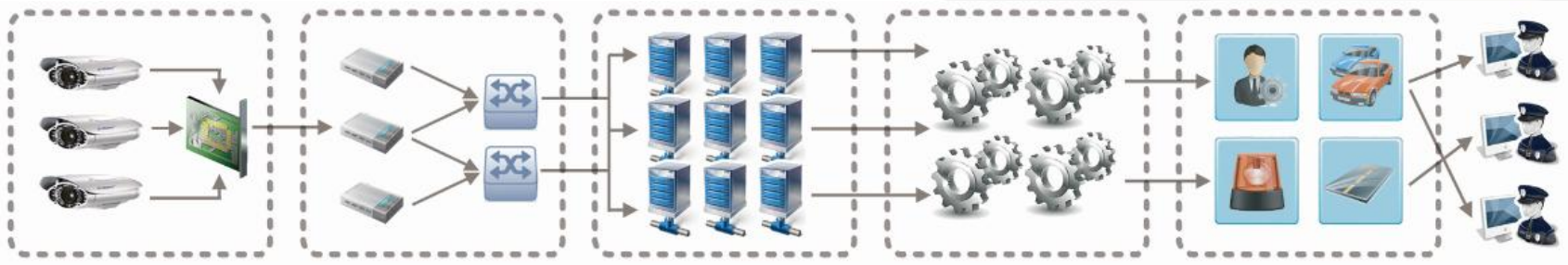
2.利用全局存储解决信息共享问题



L. Lin, X. Yu, N. Koudas. *Pollux: Towards Scalable Distributed Real-time Search on Microblogs.* In EDBT 2013.

流处理+批处理：TariDB

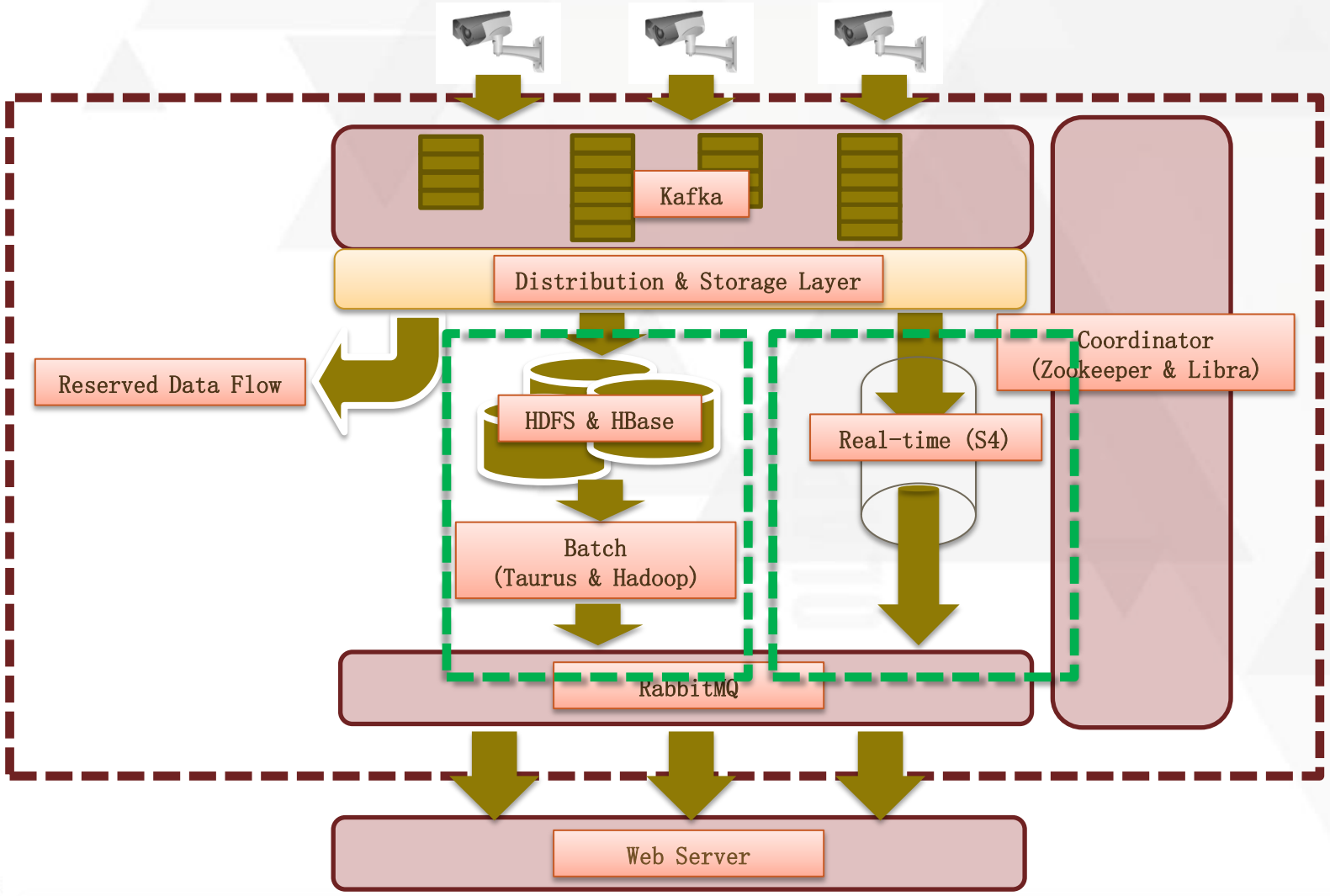
流处理+批处理：TariDB



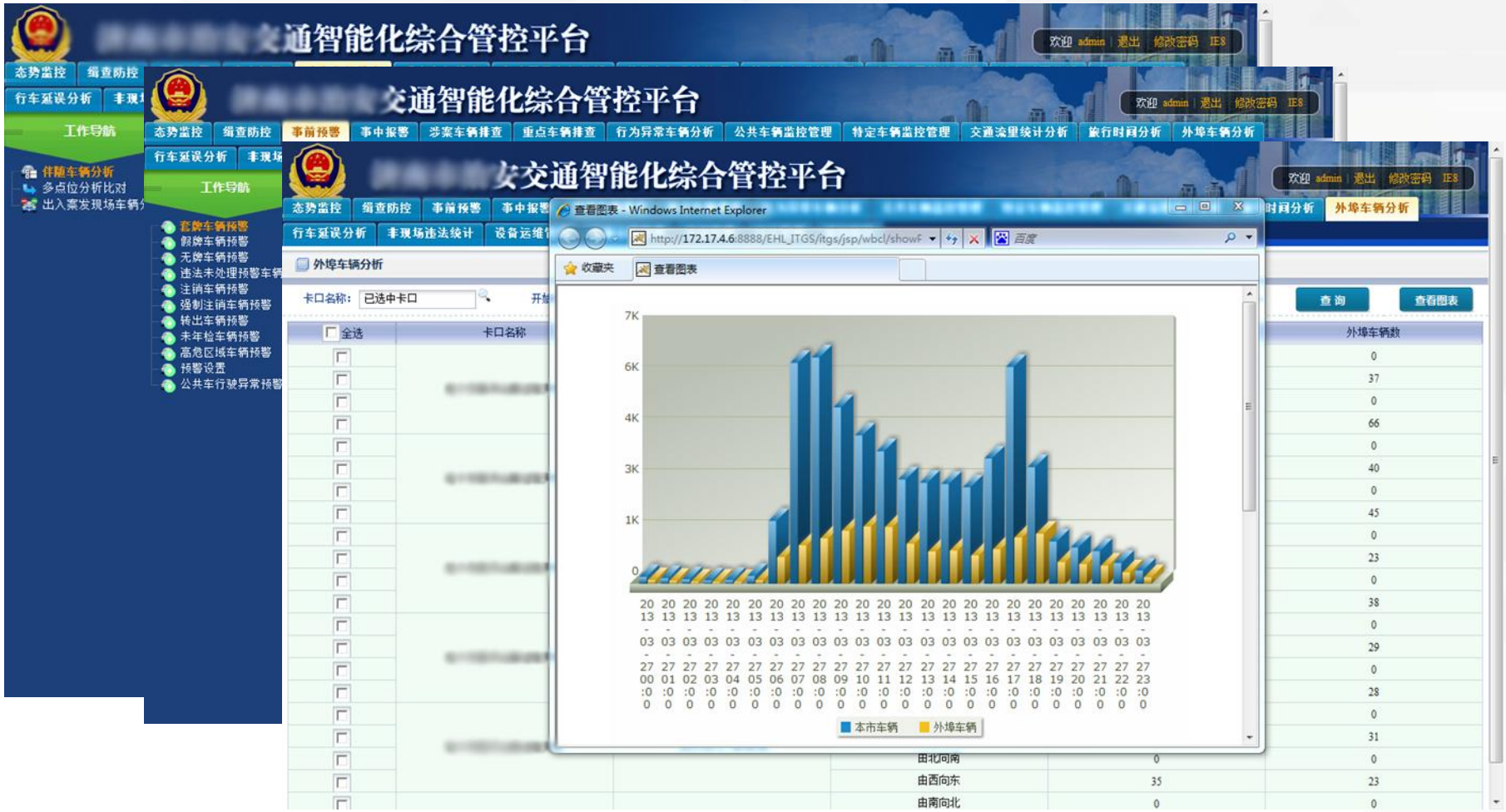
捕获识别　　网络传输

基于TariDB的处理平台

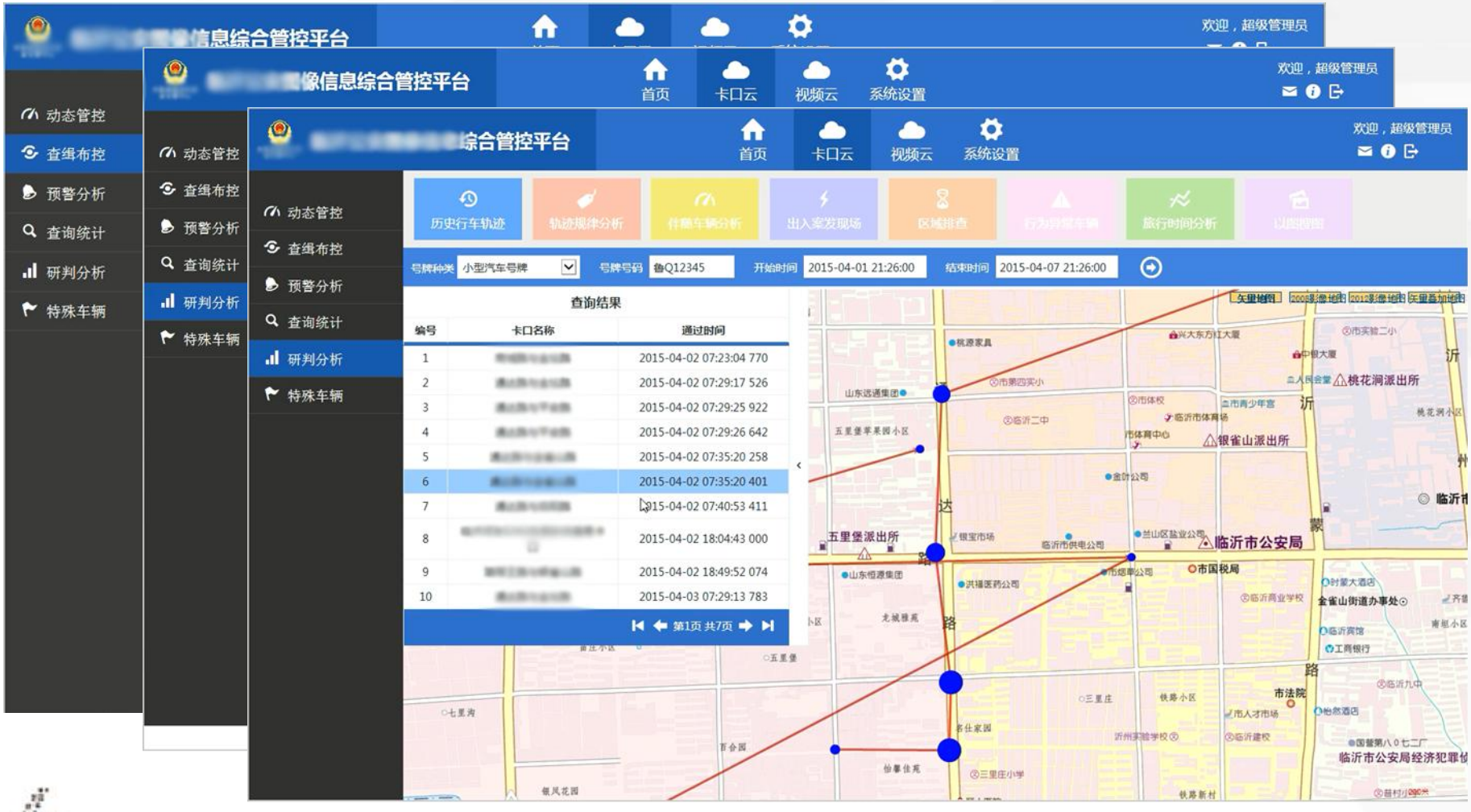
流处理+批处理：TariDB



流处理+批处理：TariDB



流处理+批处理：TariDB



总结

- 大数据处理模式 – 基于内存面向动态数据的细粒度处理
- 流处理技术发展 – 报警 ➔ 数据流管理系统 ➔ 分布式
- 分布式流处理系统剖析
  - 数据模型 – 源源不断的单条或批量数据
  - 系统架构 – 去中心化、中心化、弱中心化结构
  - 存储管理 – 抛弃存储 ➔ 集成可靠存储
  - 语义保障 – 至少一次，精确一次
  - 负载控制 – 静态、（非）自适应动态负载均衡
  - 系统容错 – 额外开销、恢复速度、节约资源
- 分布式流处理应用实例 – 海量、高速、复杂、低延迟

分布式流处理技术

THANKS