数字转型 架构演进

SACC

2019 中国系统架构师大会

SYSTEM ARCHITECT CONFERENCE CHINA 2019

IT168.com ChinaUnix.net ITPUB

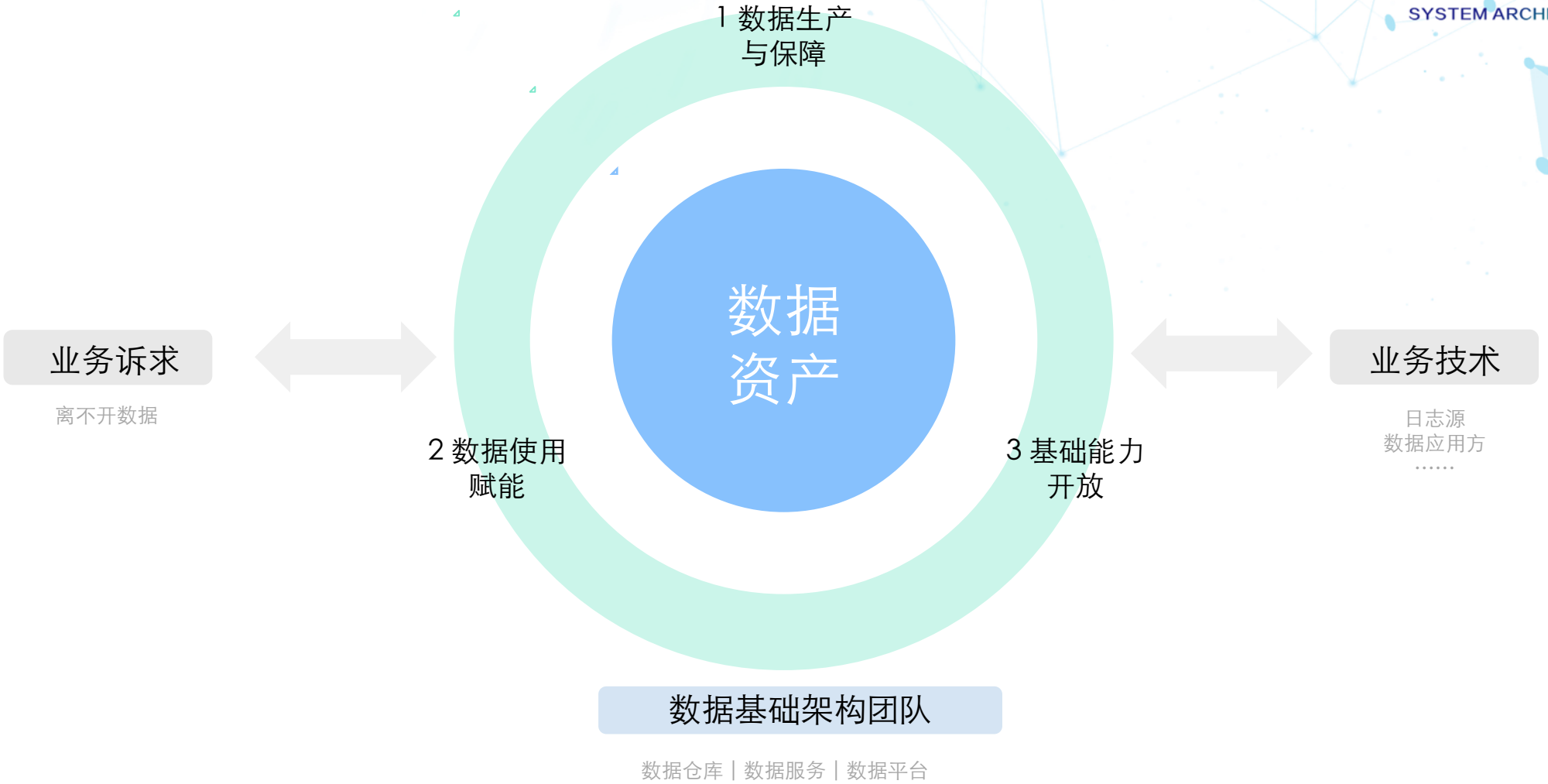
2019年10月31-11月2日 | 北京海淀永泰福朋喜来登酒店

陌陌大数据平台
在 SLA 驱动下的演进实践

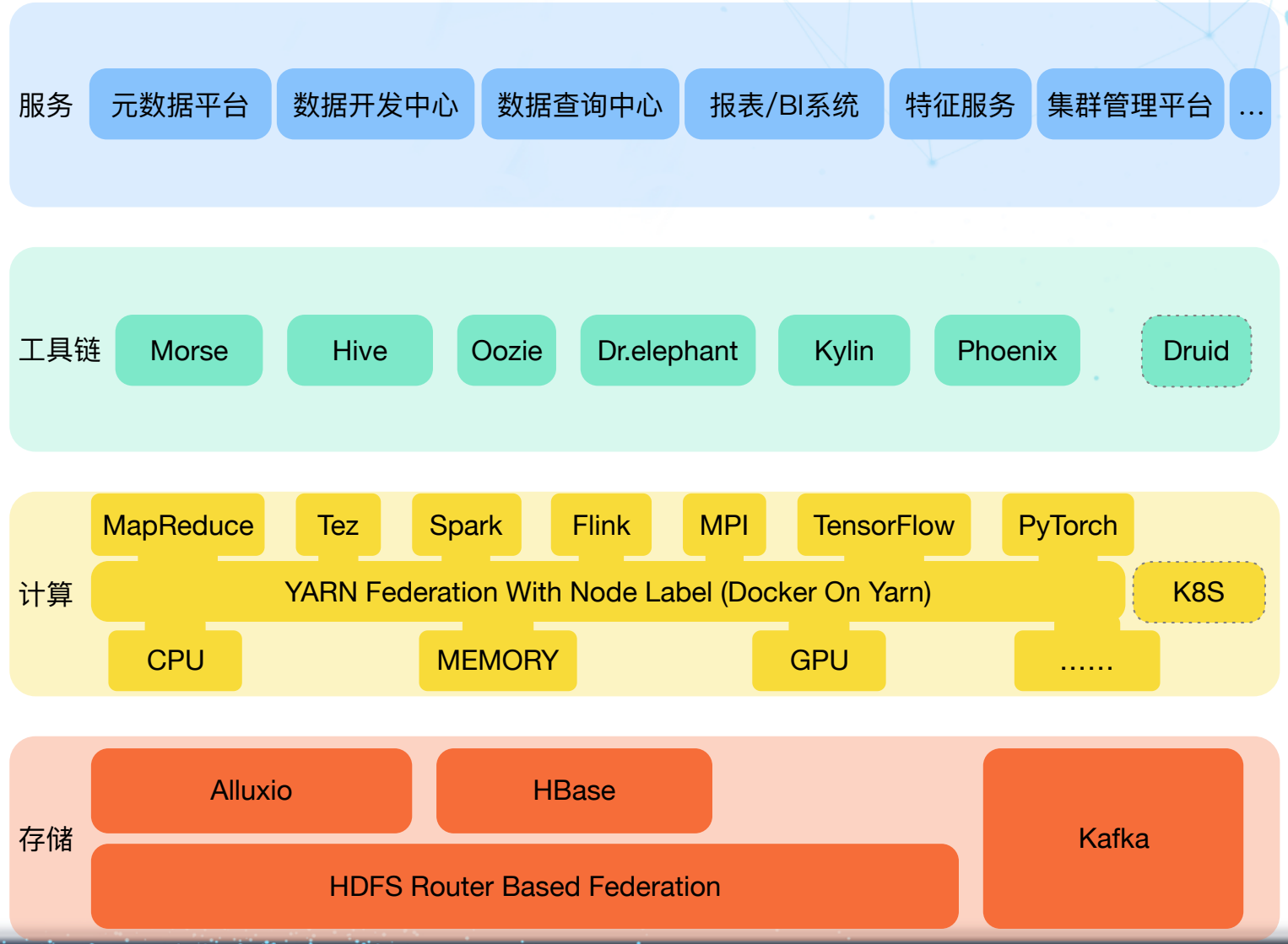
jin.xiaoye@immomo.com
2019.11.02

团队介绍

-



团队技术栈



SLA 问题

➢ SLA :

- 科学量化
  合理业务对数据生产稳定性的诉求
- 持续提升
  系统迭代演进的客观衡量标准

✓ 我们的SLA定义:

- 核心数据产出时间
  基础用户属性 | 活跃情况 | 订单集成
- 数据就绪率
  基础数据 | 报表数据 | 生产导出数据
- 数据就绪达标率
  月, 季度 | SLA 稳定性

➢ 保障 SLA 的矛盾:

- 主要矛盾: 高速增长的业务诉求 VS 有限增长的平台算力
- 次要矛盾: 平台算力与服务复杂性 VS 服务运维管理能力

1.0 阶段

需求规模
(作业量: 300～6000)
算力规模
(集群节点: 20～400)

简单快速堆机器
满足需求高增长

2.0 阶段

需求规模
(作业量: 10000)
算力规模
(集群节点: 600)

单机房容量
与单集群性能
限制算力增长

3.0 阶段

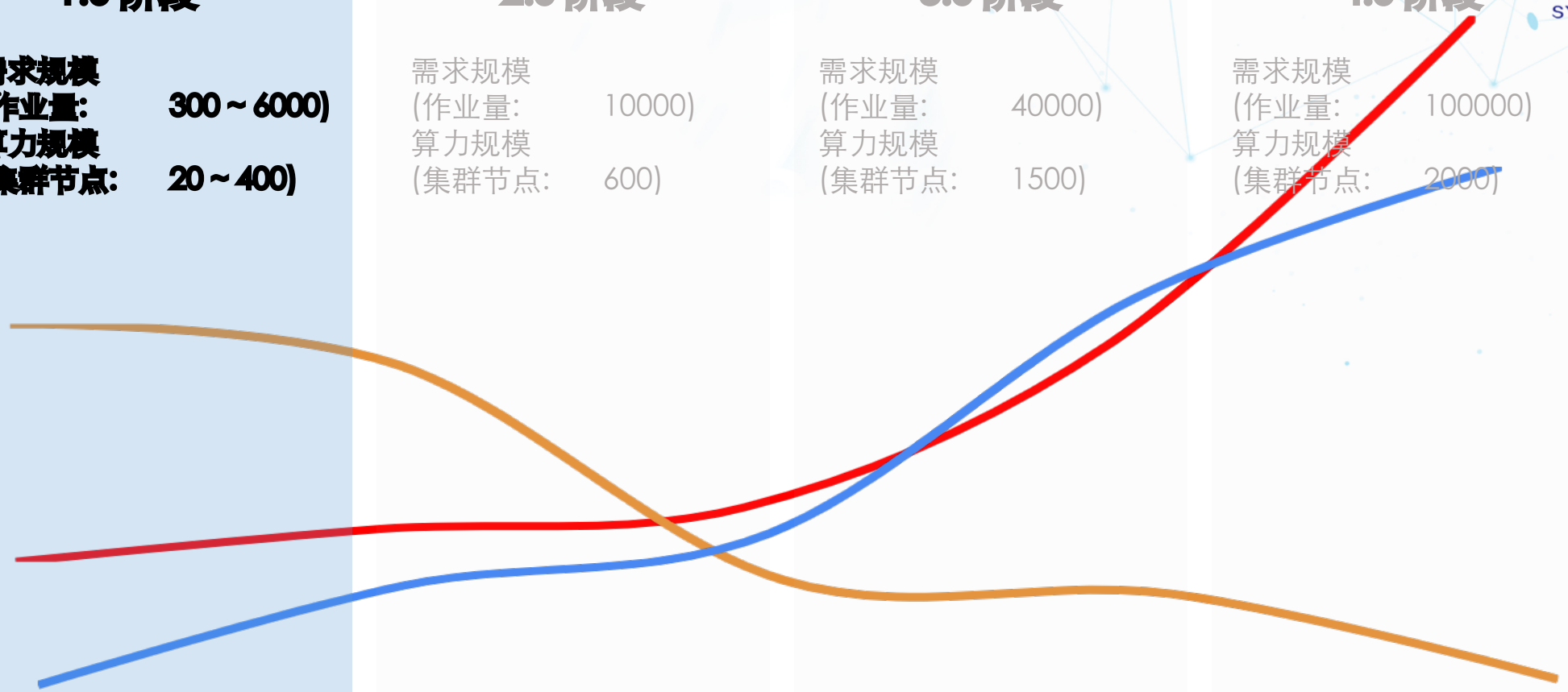
需求规模
(作业量: 40000)
算力规模
(集群节点: 1500)

机房与集群级
水平扩展能力
满足算力增长

4.0 阶段

需求规模
(作业量: 100000)
算力规模
(集群节点: 2000)

多机房多集群复杂性
引入运维稳定性风险



需求增量(作业数量)
算力水平(集群节点)
SLA 表征(异常风险)

1.0 阶段：2014 ~ 2016

- 主要矛盾：高速增长的业务诉求与平台算力 VS 手段简单的集群管理

- 业务快速增长：
  - 数据仓库对外提供服务
    基础用户属性 | 活跃情况 | 订单集成
  - 直播业务高速增长
    基础数据 | 报表数据 | 生产导出数据

✓ 解决思路：

- 服务扩容加机器 计算节点 | 存储节点	算力扩展：水平扩容算力
- 需求优化 数据流优化 | 作业参数优化	算力挖掘：关键任务难以加机器解决
- 工具与稳定性优 重点环节工具化	稳定性：手段简单，临时解决

1.0 阶段：算力挖掘 – 特定任务优化

✓ 特定任务节点单独优化:

- 数据模型与计算流优化
  中间表提取 | JOIN 顺序调整
- 系统参数调优
  并行度 | 数据倾斜优化

✓ 优化收益总结:

- 业务层改造成本低，但收益明显
  部分用例 1人日完成 | 收益 90 分钟 -> 50 分钟
- 边际效应小
  同一个作业难于持续获得收益
- 优化难于广泛应用，约束条件较多
  case by case 解决 | Hive 本身的 CBO 受限于版本 BUG

1.0 阶段：稳定性保证 – 工具化

✓ 重点环节工具化:

- 数据集成(DUMP)
  自动重试 | 资源分配 | 数据校验
- ETL 管理 (COORD)
  结构规范 | 上下线 Review | 屏蔽统一调度系统配置
- 数据导出(PUMP)
  降低人为错误 | 快速恢复

1.0 阶段

需求规模
(作业量: 300 ~ 6000)
算力规模
(集群节点: 20 ~ 400)

简单快速堆机器
满足需求高增长

2.0 阶段

需求规模
(作业量: 10000)
算力规模
(集群节点: 600)

单机房容量
与单集群性能
限制算力增长

3.0 阶段

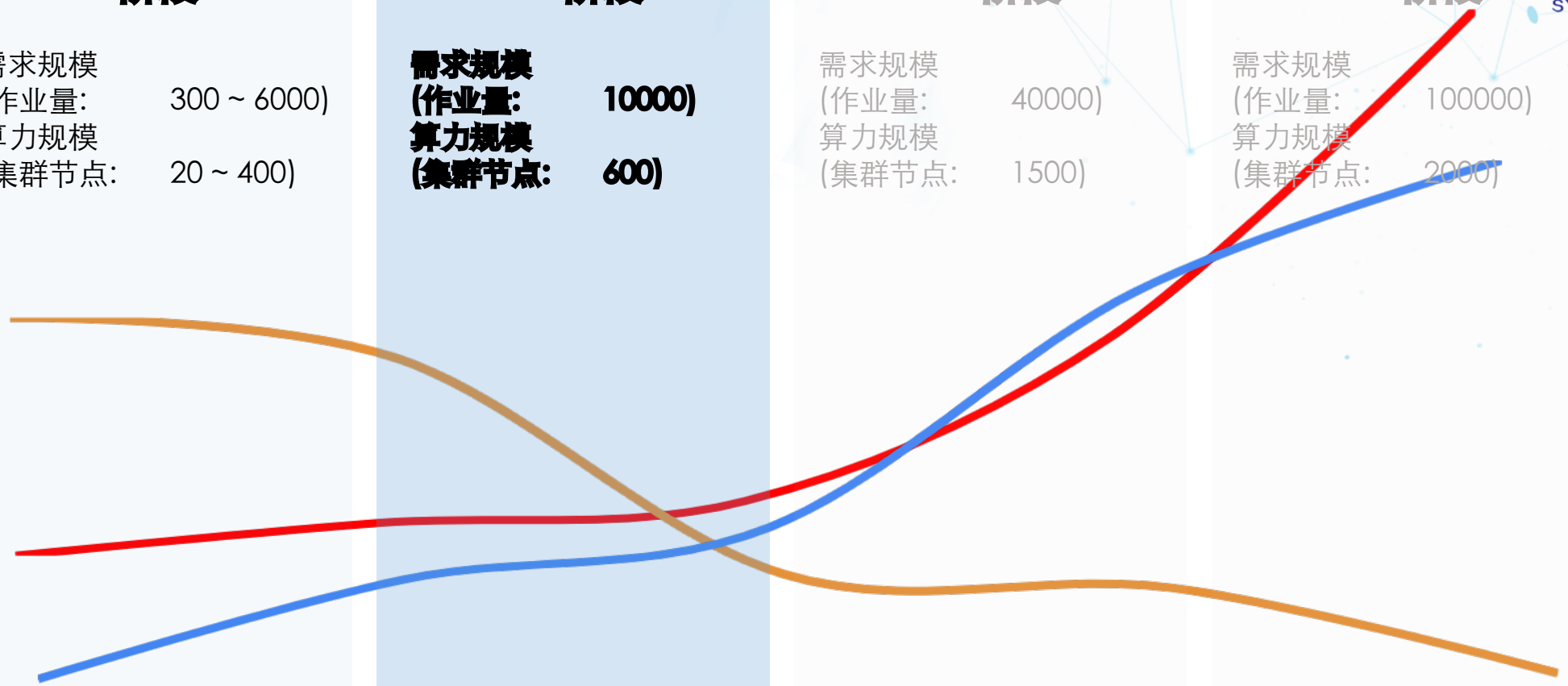
需求规模
(作业量: 40000)
算力规模
(集群节点: 1500)

机房与集群级
水平扩展能力
满足算力增长

4.0 阶段

需求规模
(作业量: 100000)
算力规模
(集群节点: 2000)

多机房多集群复杂性
引入运维稳定性风险



需求增量(作业数量)
算力水平(集群节点)
SLA 表征(异常风险)

2.0 阶段：2017

- 主要矛盾：业务诉求持续增长 VS 平台算力增长遇到瓶颈

- 原 IDC 增长瓶颈无法扩容：
  - 服务不能长时间停服迁移
    存储服务 | 计算服务 | 流式任务
  - 可用现成运维工具不足
    服务监控指标采集不足 | 资源管理混乱

- 单集群出现性能压力 SLA 不退化面临挑战：
  - HDFS NN 无法垂直扩容
    Memory : 192 G
  - HDFS NN 启动耗时增加
    故障影响范围广 | 影响时间长

✓ 解决思路：

- 引擎升级(MR->Tez)
  整体作业计算效率提升
- 流式数据集成
  保证 SLA 在迁移与优化时间内不退化

算力挖掘：满足 1个季度需求增长

- 机房平滑迁移
  数据量大 | 时间紧

算力扩展：满足算力持续水平增长

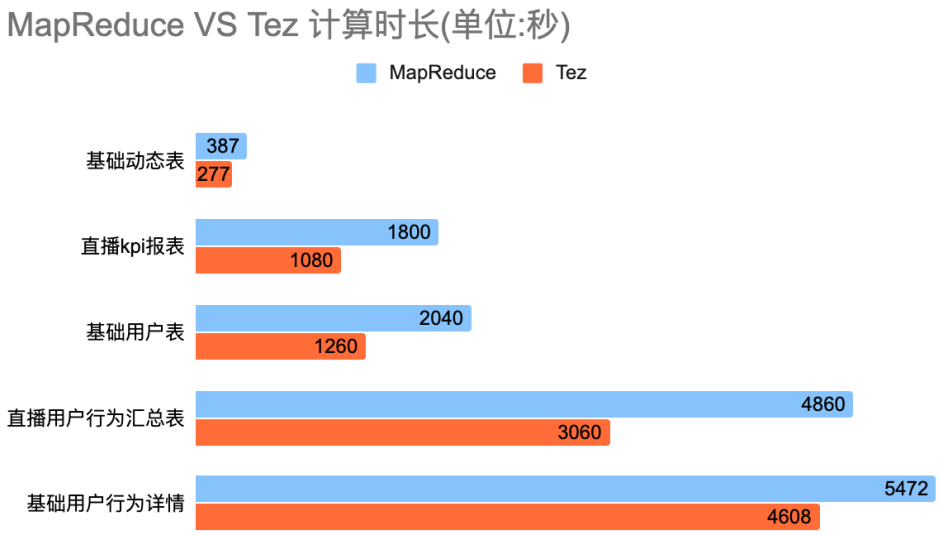
- NN 启动优化
  启动模拟 | 参数优化 | 逻辑优化

稳定性：扩容后的集群出现挑战

2.0 阶段：算力挖掘 – 引擎升级

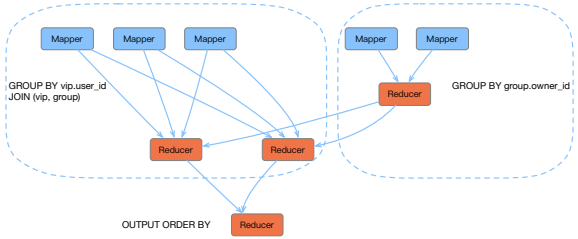
✓ 计算引擎升级 (MR -> Tez):

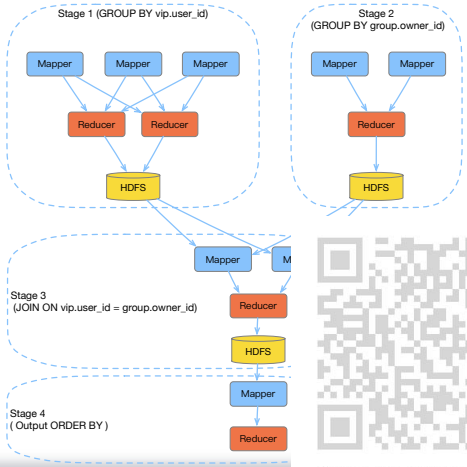
- 同任务引擎切换可以获得计算速度提升 20% ~ 30%
  计算 DAG 越复杂, 迭代轮数越多提升越明显



✓ Tez VS MR 收益来源:

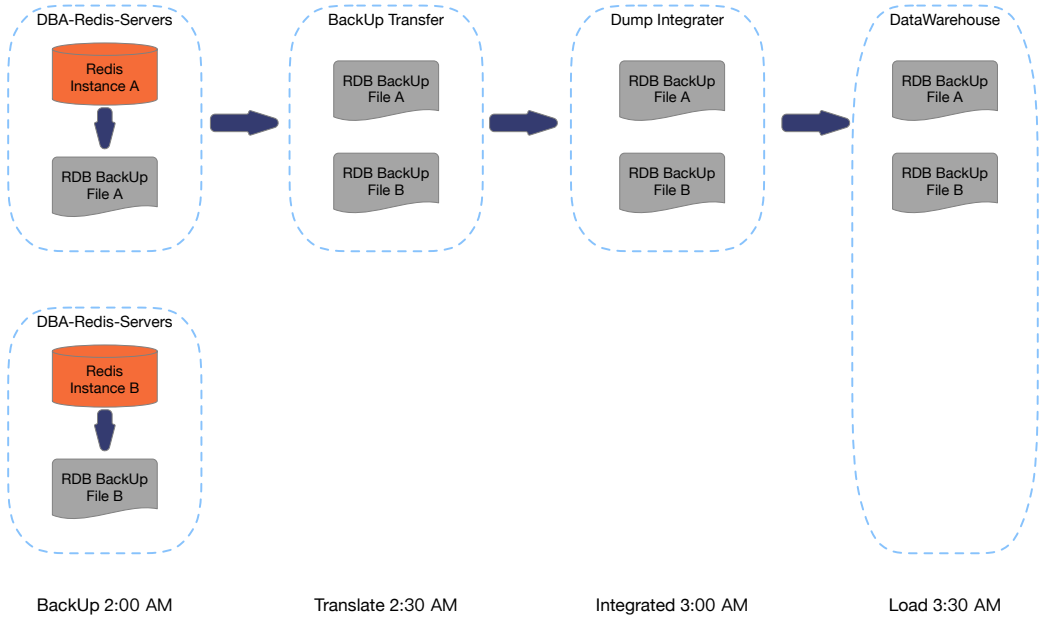
- 计算作业直接翻译成 DAG VS 多个由 Map-Reduce 组成的 Stage
  更可控的 FailOver 节点
- 无 Stage 之间 HDFS 交互开销
  Shuffle 后续可插拔优化
- 动态 Container 申请
  更灵活的资源利用效率

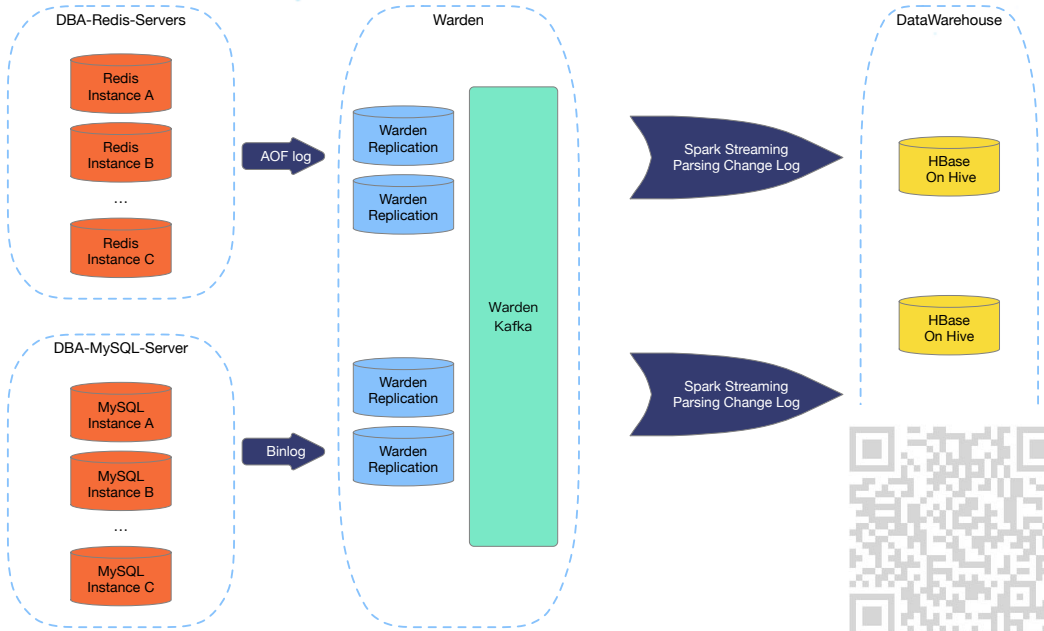




2.0 阶段：算力挖掘 – 流式集成

- ✓ **解决的问题**: 数据集成完成时间 (3:00 -> 00:15)
- **技术重点:**
- 数据库 Change Log 向 HBase 数据操作转化
- Change Log 的出错后回放修复
- 避免数据丢失的一致性保证





2.0 阶段：算力扩展 – 集群迁移

✓ 迁移的关键点:

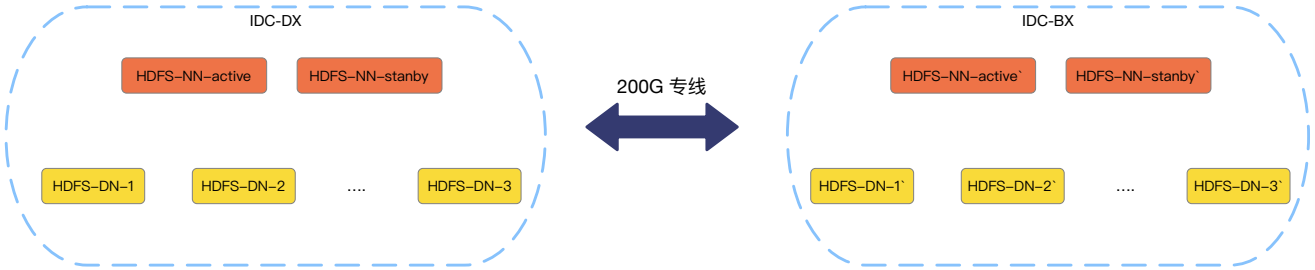
- 目标: DataNode 迁移过程不停服
- 难点: 节点多(600) | 数据量大(20P) | 时间有限(60天)

✓ 直接 Decommission 的问题:

- 耗时无法接受
  场景下单节点耗时20小时 | 并行节点迁移下耗时更长
- Block 安置策略可能落回旧机房
  单个 Block 多次迁移造成浪费

✓ 迁移方案(Fast Migrating):

- DataNode 增加 Freezing 状态
  该状态下节点可读不可写
- 点对点镜像复制 DataNode 工具(Rsync 数据同步)
  时间缩短到 8小时每机柜 | 整个机柜并行迁移



2.0 阶段：算力扩展 – 迁移遇到的问题

✓ HDFS动态拓扑:

- **遇到问题:** DN 镜像节点注册后大量报 Block Under-Replicated 触发文件块复制
- **原因:** HDFS 机柜拓扑启动时加载 | 新节点所属机柜信息缺失 | 触发副本放置策略(本机 | 机柜 | 其他机柜)
- **解决方法:** Dynamic Topology CLI 命令行动态加载拓扑配置

✓ 其他问题:

- 迁移流量造成 Flume 写 HDFS 文件关闭超时出现租约未释放现象
- NameNode 内存紧张情况下迁移启动时间较长甚至异常退出
- 迁移流量在交换机层面合理分配，切换网卡bond充分利用网络结构提升迁移效率

2.0 阶段：稳定性保证

✓ 资源拆分隔离：

- YARN 队列拆分
  当时运维手段简单
- HDFS Quota 拆分
  固定拆分

2.0 阶段：稳定性保证 – 单集群性能优化

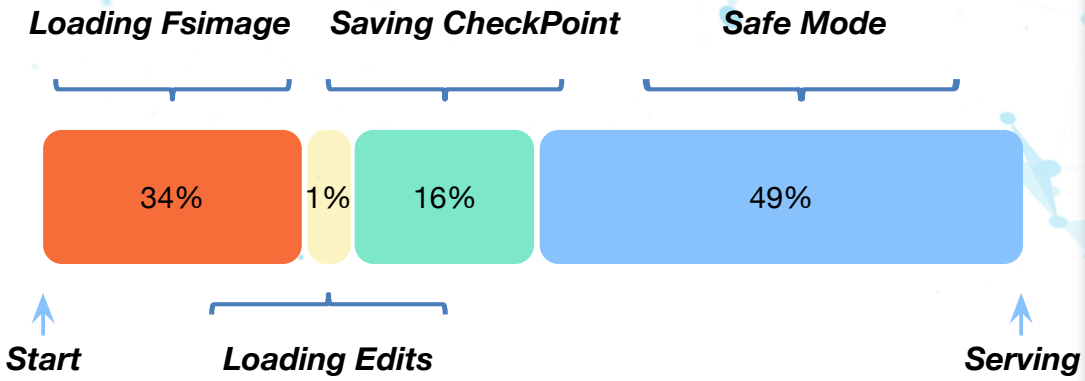
✓ HDFS启动耗时问题

◘ 挑战一：集群规模大，NN启动耗时较大

- 1500 DN | 6亿 Block | 10亿 INODE，重启耗时 60分钟 (多轮 Full GC)
- 运维管理低效，出现故障后恢复成本高

◘ 挑战二：测试环境规模太小，生产环境又不能拿来做验证

- 海量 DN 块汇报不好复现
- 生产 NN 无法 Debug

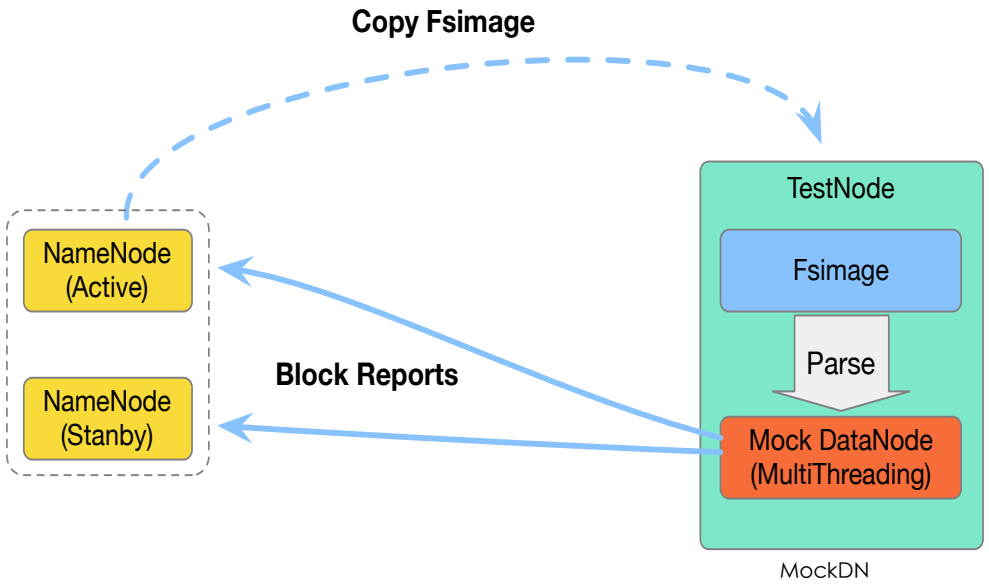


能不能模拟出生产环境?

2.0 阶段：稳定性保证 – HDFS NN 启动优化

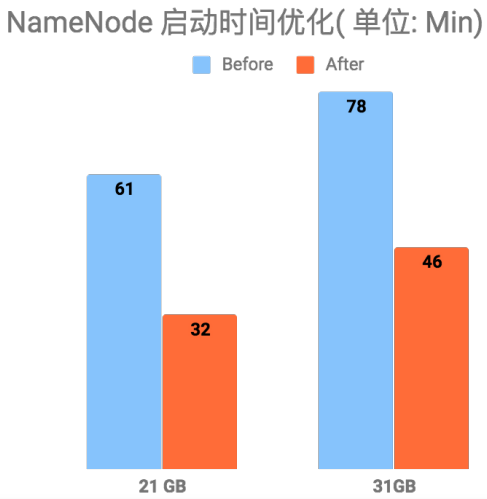
» 阶段一: 测试 NN 启动， MockDN 模拟产生块汇报

- 解析 FSimage
  分配每个DN归属 Block 信息 (id, length, timestamp)
- 多线程模拟 DataNode
  单机可以模拟 1500+ DN



» 阶段二: 性能优化

- 参数调优
  -JVM 参数调优
  - NN参数调优
- 多线程模拟 DataNode
  - 块汇报优化 (限速, 合并 IBR , 独立线程 RPC Queue)
  - 锁优化 (IBR 合并后的写事务合并)， 减少锁争用
  - Protobuf 协议优化提高序列化和反序列化效率
  - 统一调度类调度 SendHearBeat 和 Block Report



时间降低 40%（同生产配置）

1.0 阶段

需求规模
(作业量: 300 ~ 6000)
算力规模
(集群节点: 20 ~ 400)

简单快速堆机器
满足需求高增长

2.0 阶段

需求规模
(作业量: 10000)
算力规模
(集群节点: 600)

单机房容量
与单集群性能
限制算力增长

3.0 阶段

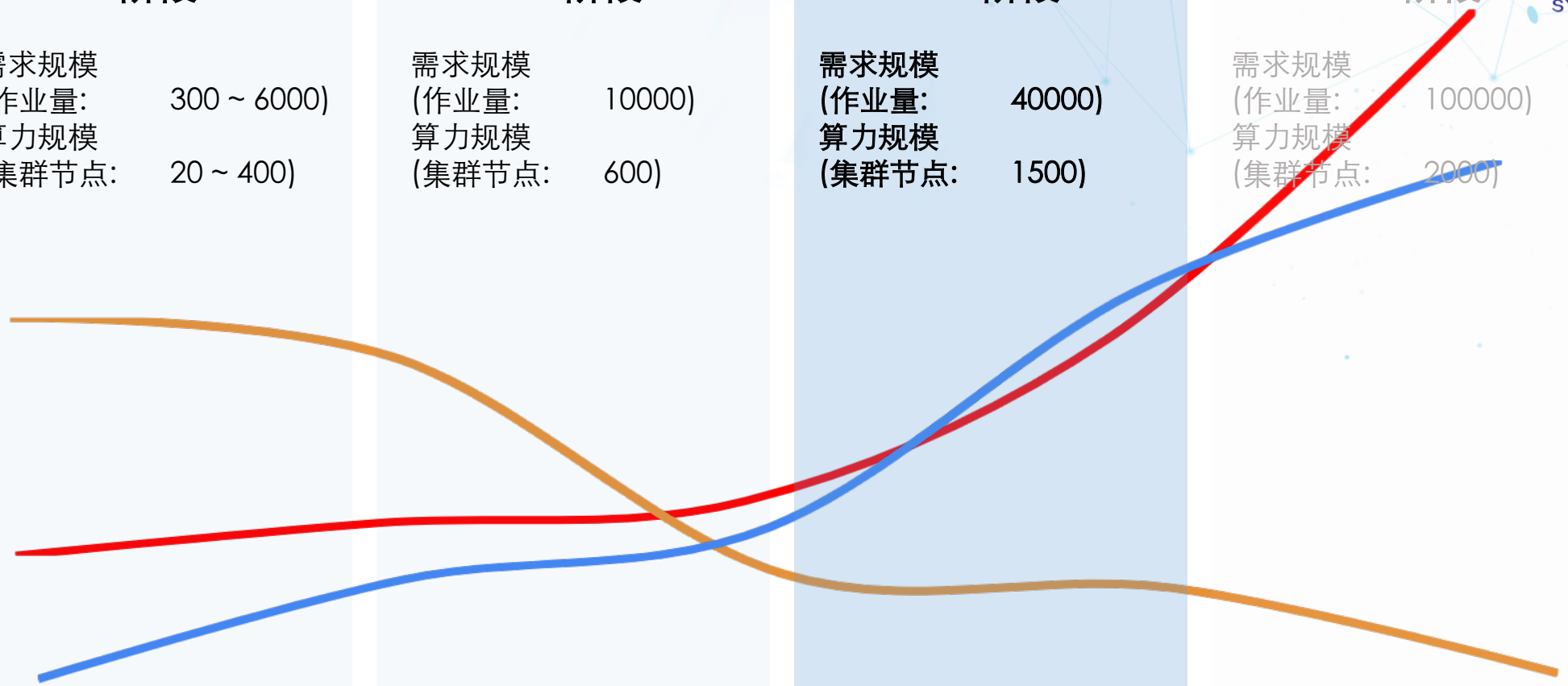
需求规模
(作业量: 40000)
算力规模
(集群节点: 1500)

机房与集群级
水平扩展能力
满足算力增长

4.0 阶段

需求规模
(作业量: 100000)
算力规模
(集群节点: 2000)

多机房多集群复杂性
引入运维稳定性风险



需求增量(作业数量)
算力水平(集群节点)
SLA 表征(异常风险)

3.0 阶段：2018

- 主要矛盾：需求场景更多复杂度上升 VS 算力受限于单集群红线与机房瓶颈

- 单集群瓶颈逼近：
  - NN 优化收益成本增加
    内存容量 | RPC 队列容量 | RPC 响应速度
  - SLA 故障隔离域问题
    不同业务之间 SLA 不同 | 降低业务方互相影响

- 单机房依旧存在瓶颈(2000)：
  - 业务场景更加丰富
    需求持续增长多业务线与团队使用数据

✓ 解决思路：

- 分治拆分 算力扩展：满足算力持续水平增长
  HDFS Federation | YARN Resource Model
- 计算优化 算力挖掘：新技术解决更多问题
  Kylin 降低需求固化模型 | Alluxio 内存文件加速

3.0 阶段：算力扩展 – HDFS Federation

✓ 目标:

- 水平扩容 NN 支撑更大的业务场景
- 故障隔离域
- 更近细化的参数调优

✓ 关键点:

- NN 单集群红线 (文件数 6亿 文件块数 10亿)
- 切分后的数据迁移会有大量的存储与网络开销

FastMV ： 并行创建 Block 硬链接避免真正的文件拷贝|

	Federation(New)			
	Ns1	Ns2	Ns3	Ns4
FileNum (Million)	276	296	67	38
BlockNum (Million)	422	296	40	43
Memory (GB)	117	108	18	15

3.0 阶段：算力挖掘 – OLAP场景应用

✓ Kylin 应用:

- 业务场景稳定的报表类需求支持
  维度固定 | 稳定需求 | 降低开发成本

✓ 解决的问题:

- 抽象自动 Build 组件打通 ETL 调度系统
  依赖检查 | 自动重试 | 失败与产出延迟报警
- 集成公司可视化报表系统
  拖拽维度筛选 | 透视表构建

3.0 阶段：算力挖掘 – Alluxio 内存文件系统

✓ 陌陌场景

» 场景1：大数据生态的存储访问缓存层，加速 Ad-hoc 查询以及ETL生产

» 场景2：分布式计算的中间存储引擎，优化Shuffle、SavePoint等IO性能

✓ 集群部署

单机配置	部署模式
CPU：32 核 内存：72G（总196 G） 硬盘：无HDD层	Alluxio Worker 混合部署 HDFS DataNode & YARN NodeManager & SparkExecutor

- *根据场景需求 HDD 层交给 HDFS，并避免 Alluxio 内部内存与硬盘间数据换入换出*
- *各服务组件分别使用自身 Locality 能力保证*

✓ 整体收益

- 较原生方案实际生产情况，时间开销优化 3~5 倍
  - 不同输入数据量级下性能提升程度不同
  - 例如，小任务场景下由于 Spark 自身内存缓存管理机制 Alluxio优化不明显

1.0 阶段

需求规模
(作业量: 300 ~ 6000)
算力规模
(集群节点: 20 ~ 400)

简单快速堆机器
满足需求高增长

2.0 阶段

需求规模
(作业量: 10000)
算力规模
(集群节点: 600)

单机房容量
与单集群性能
限制算力增长

3.0 阶段

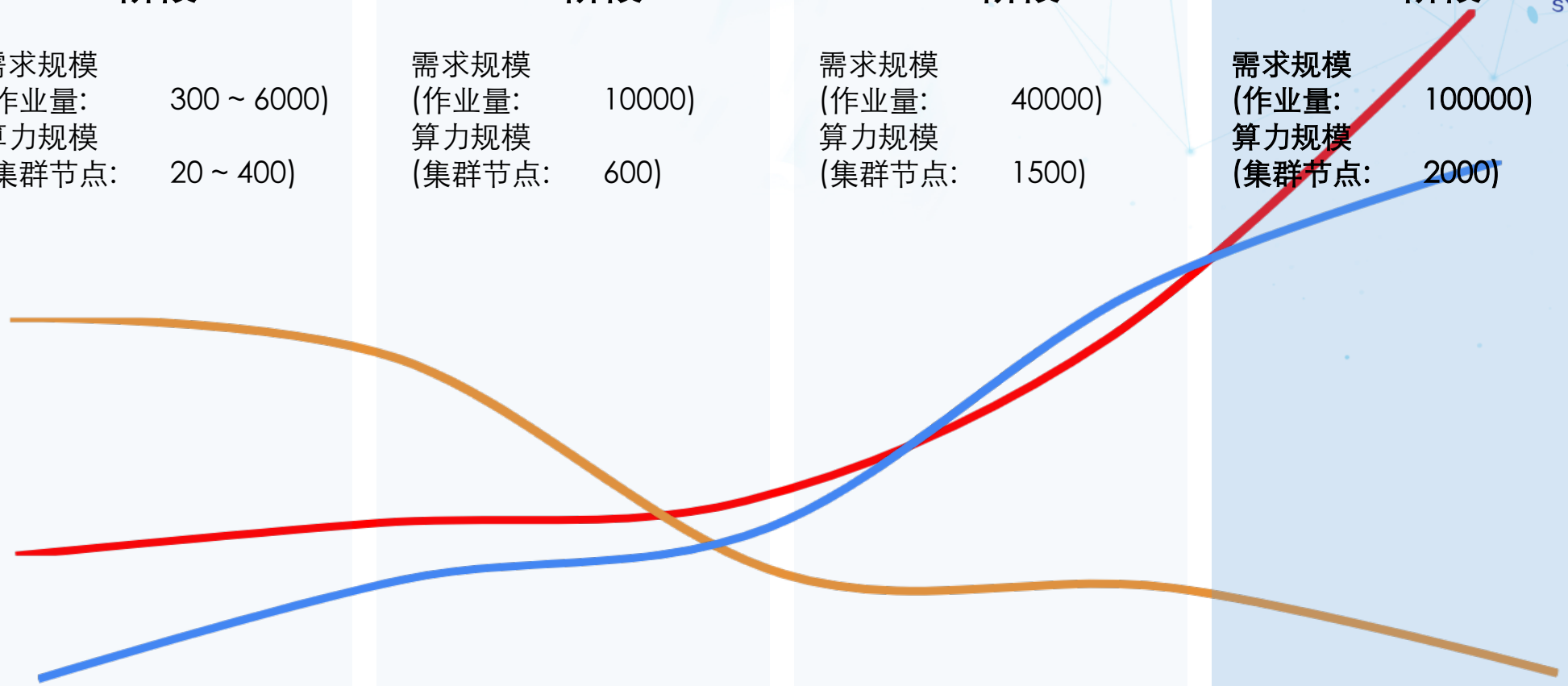
需求规模
(作业量: 40000)
算力规模
(集群节点: 1500)

机房与集群级
水平扩展能力
满足算力增长

4.0 阶段

需求规模
(作业量: 100000)
算力规模
(集群节点: 2000)

多机房多集群复杂性
引入运维稳定性风险



需求增量(作业数量)
算力水平(集群节点)
SLA 表征(异常风险)

4.0 阶段：2019

- 主要矛盾：更高的服务标准与运维复杂度 VS 团队成员人力资源有限之间的矛盾

- 多集群多机房精细化管理
  - 轻客户端用户无感知升级迁移
  - 统一资源管理

- 集群算力的更充分利用：
  - 在离线峰谷资源错配?

✓ 解决思路：

- DPW 自研服务运维平台
  基础用户属性 | 活跃情况 | 订单集成
- 多机房多集群统一管理
  HDFS Router | YARN Router

稳定性：
多机房多集群下运维复杂性上升

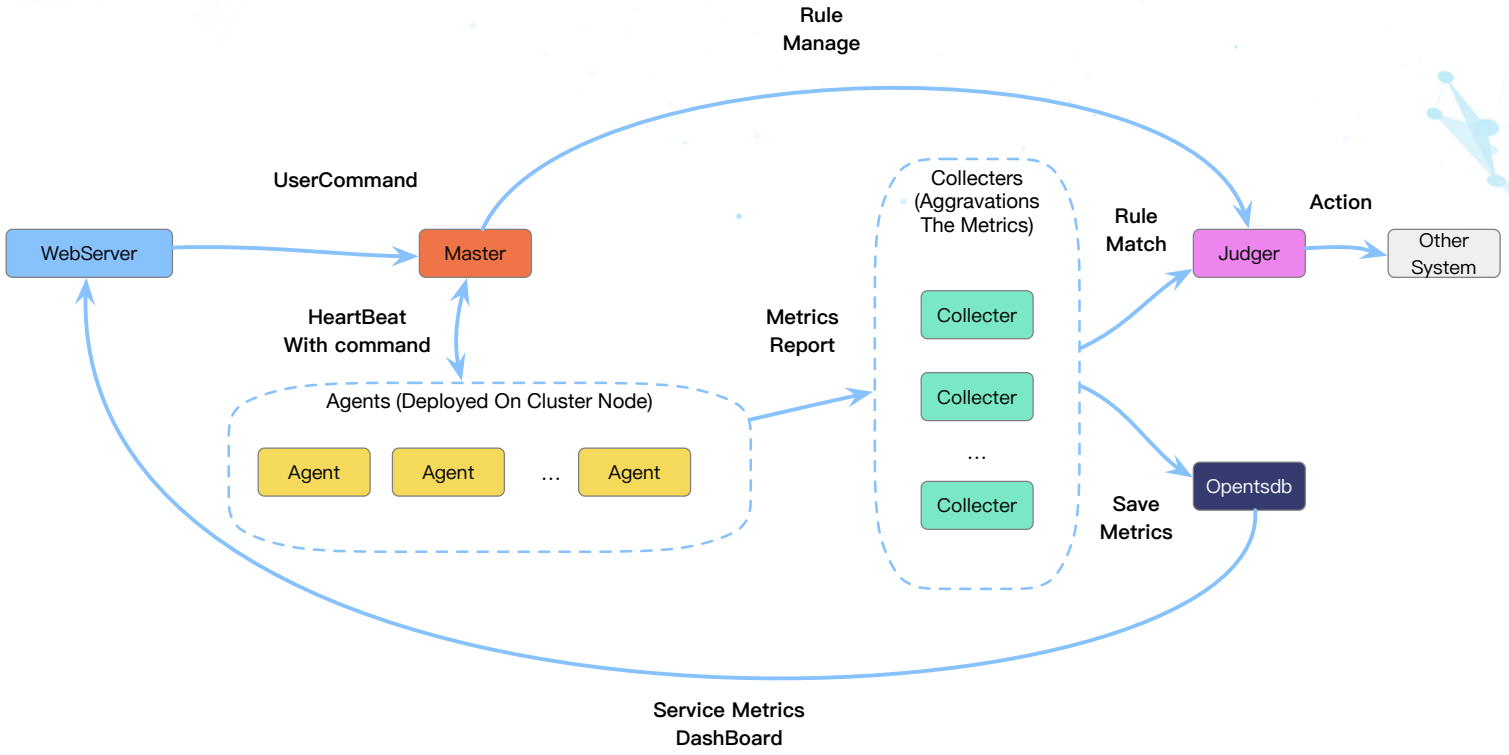
- Tez -> Spark 引擎升级 & 优化
  引擎升级 | 子查询复用 | Spark AE
- 流式计算
  计算负载打散 | 实时批处理错峰

算力挖掘：持续优化提高算力效率

4.0 阶段：稳定性保证 – DPW 集群管理平台

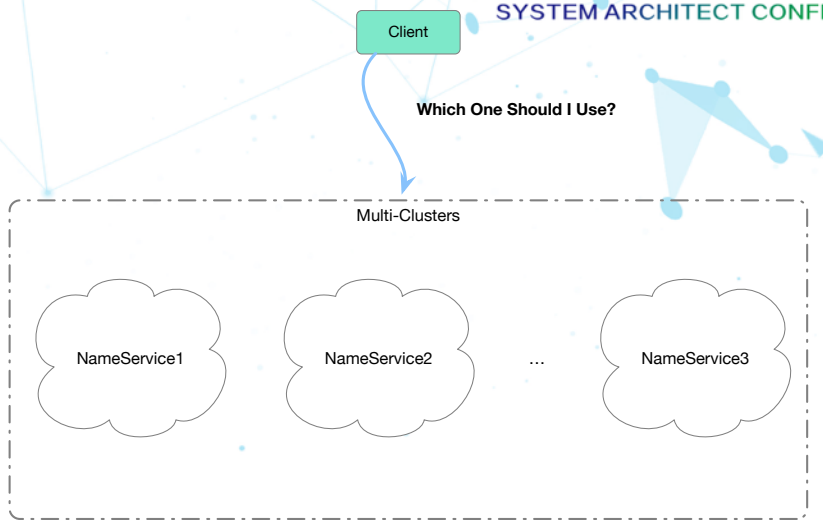
✓ 解决问题:

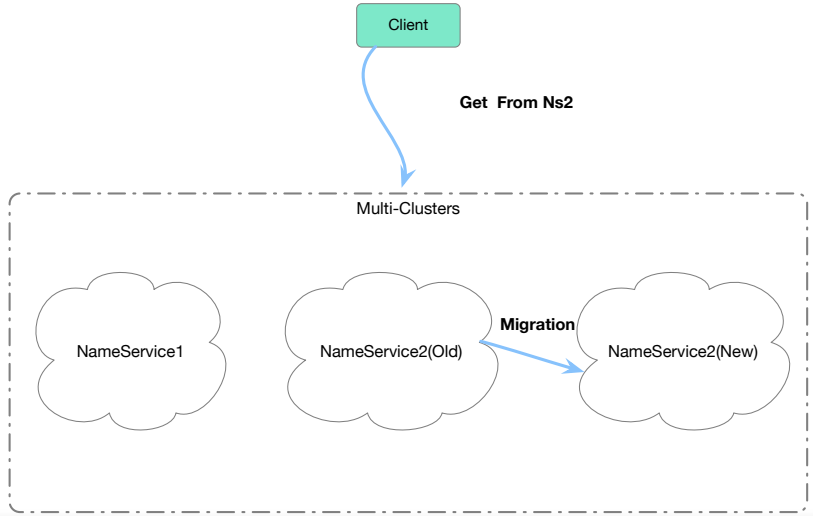
- 区分多服务与多集群日常运维管理操作
  服务启停 | 滚动升级 | 集群扩容 | 节点上下线
- 异构化配置管理支持
  基于标签异构 | 配置版本管理 | 快速配置回退
- 集群服务节点元信息管理
  服务分布 | 业务分布信息
- 服务指标采集仪表盘与报警规则
  HDFS| YARN | HBase | Kafka etc .



4.0 阶段：稳定性保证 – 多机房 & 多集群管理

- 问题一：用户不清楚自己属于哪个集群
  - HDFS 9（Federation： 6） | YARN 4 |
  - 统一 Schema URI | 统一 ACLS
- 问题二：集群升级与迁移，需要业务方更新客户端配置
  - 业务方需求： 透明无感知集群运维

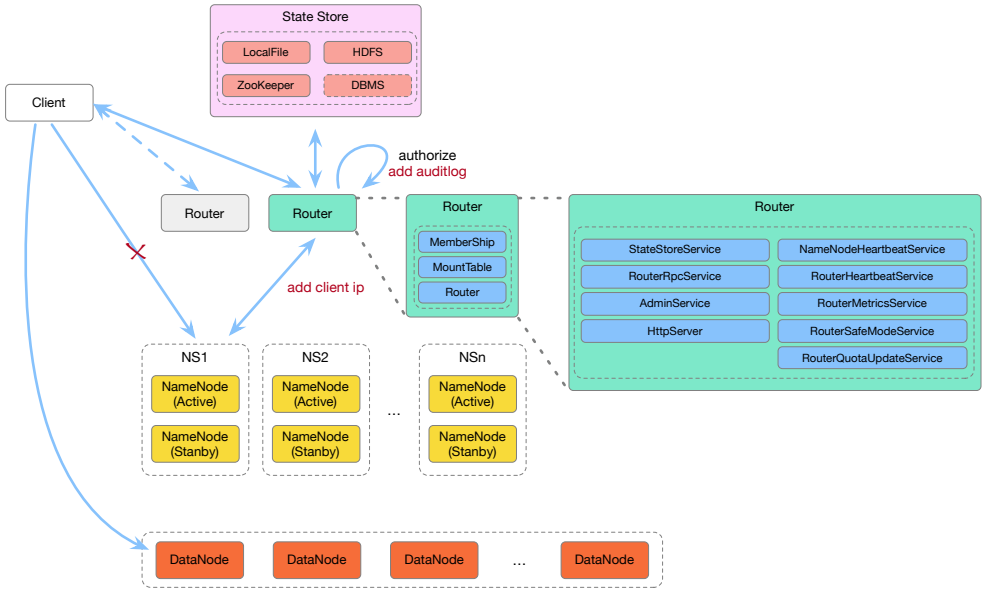




4.0 阶段：稳定性保证 – HDFS Router & YARN Router

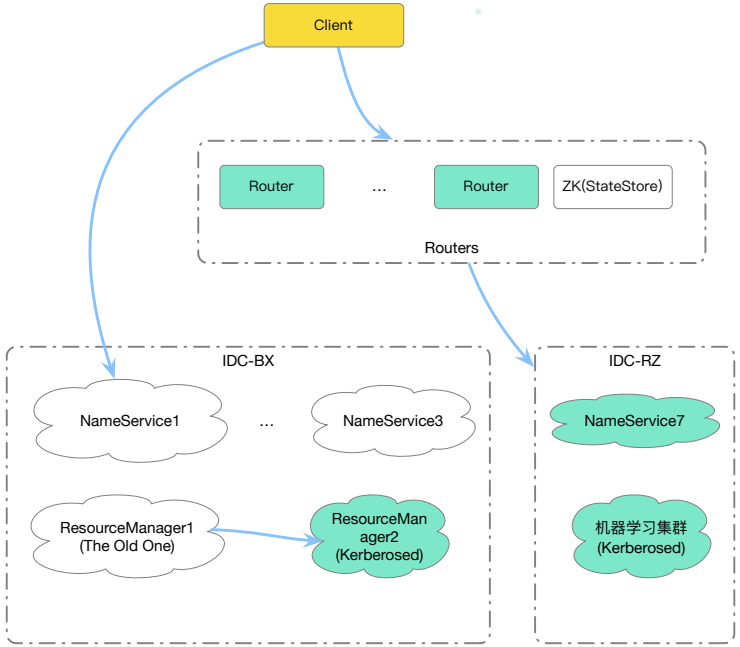
✓ HDFS Router:

- 无状态的访问组服务，兼容 NN 访问协议(HA, Fail Over)
- 支持多种外部状态存储，配置访问策略(Local File, ZooKeeper, DBMS, HDFS)



✓ YARN Router With Kerberos:

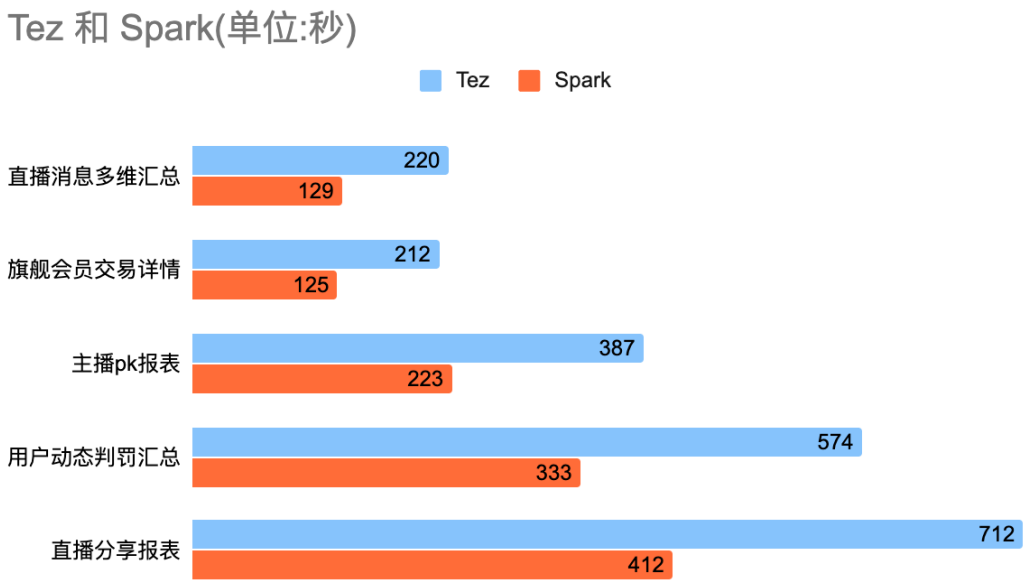
- YARN-6539 | YARN-9811



4.0 阶段：算力挖掘 – 引擎升级

✓ 计算引擎升级 (Tez -> Spark):

- 同任务引擎切换参数调优可获得计算平均 35%
  使用默认参数迁移优化不明显，需要针对作业调参



✓ Spark VS Tez 收益来源:

- 资源重用
  Executor 一次申请多轮 Stage 可复用
- 计算流程数据本地化
  Spark 首轮无本地化但后续 Stage 数据本地化 VS Tez相反
- Codegen
  Spark 由 SQL 生成原生代码 | Tez 根据生成 pb 文件反序列化生成
- 更丰富的自动优化策略
  COUNT(DISTINCT) 自动改写 | EXIST/IN 自动优化 | 字节比对 ETC.

4.0 阶段：算力挖掘 – 子查询复用 & Spark AE

✓ Spark 子查询复用:

- 集群存在很多相同子查询重复计算较多
  旧查询复制修改 | 仓库建模不合理让业务方频繁 JOIN 基础表

	算子Fragment占比	作业占比
复用片段次数 > 5	7.88 %	16.98%
子查询深度 > 2	3.57 %	8.22%

✓ 解决的方案:

- 子查询缓存
  阿里的 Relation Cache| Hive 3.0 Materialized View | Calcite
- 数据查询中心提供子查询集市
  增加子查询重复占比 | 统一计算口径
- 结果：Ad-hoc 作业平均时间降低 30%，查询时长90分位数降低 25%

✓ Spark AE:

- 资源动态调整(INTEL 方案)
  SPIP | Runtime Skewed Join

✓ 解决的方案:

- 优化 Spark 查询引擎性能
  Shuffle 合并多线程 | BroadcastJoin
- 合理分配各查询任务资源利用率
  分Stage独立申请计算资源
- 结果：复杂性查询能带来就计算时间 10%

4.0 阶段：算力挖掘 – 流式计算

✓ 解决问题:

- 负载在时间维度分摊
  计算引擎升级 (MR -> Tez)
- 数据时效性提升业务价值（时间就是金钱）
  实时推荐与模型在线训练 | 活动效果及时评估策略优化
- 避免数据飞线
  EventDrived Arch | Reatime DataWarehouse

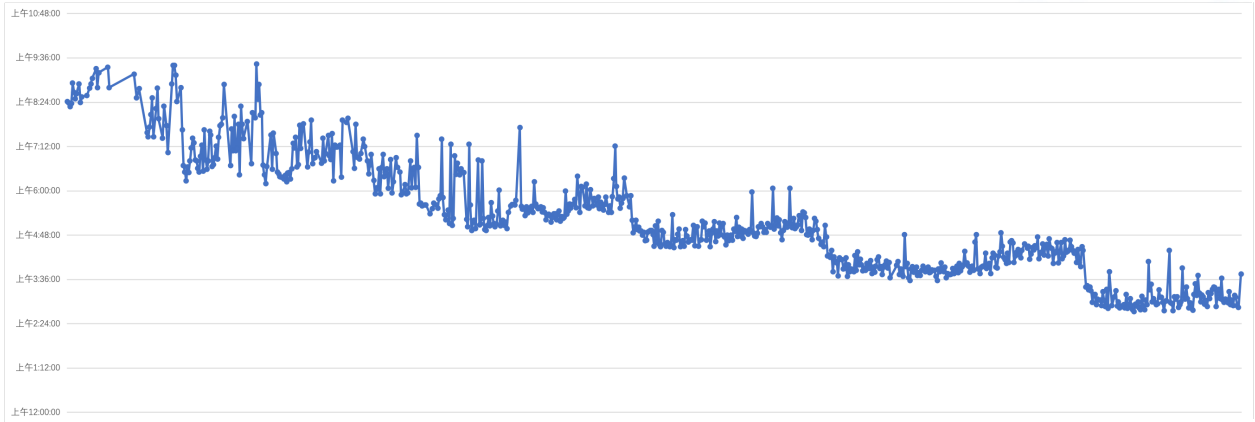
✓ 主要技术点:

- Flink SQL 化支持
  开发中心 | Schema DDL 管理 | Hive MetaStore 集成
- Kafka Manager
  自动生成迁移计划 | TImeMachine | Traffic Control
- Flink 优化
  Auto Rescaling (FLIP-53 | FLIP-56 | FLINK-12002)

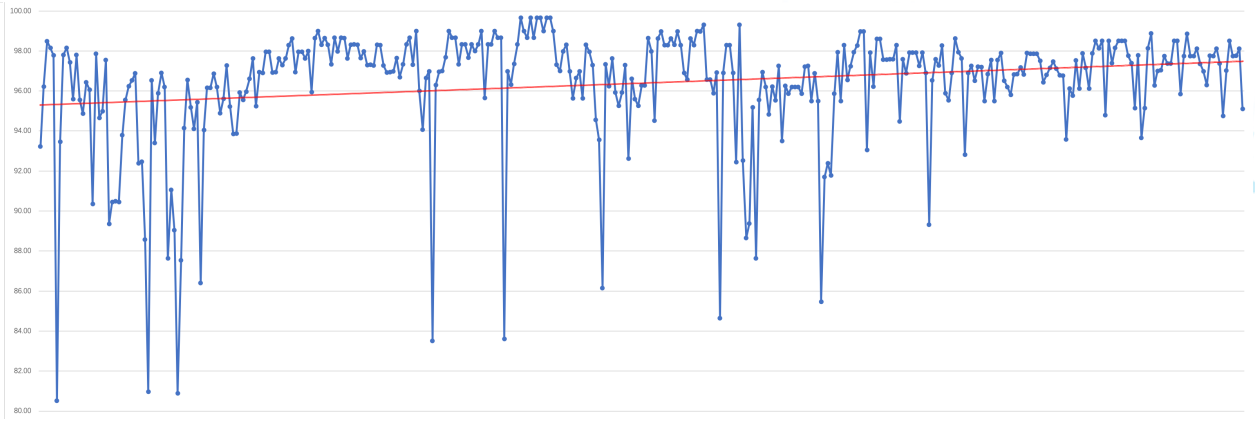
总结(a)

		1.0 2014 ~ 2016	2.0 2017	3.0 2018	4.0 2019
需求规模	用户组数	5	10	40	80
	作业量级	6000	1W	4W	10W
算力规模	节点规模	400	600	1500	2000
	部署机房	1	1	1	2
	集群数量	1	1	存储:4 | 计算:2	存储:9 | 计算:4
挑战		业务快速增长	IDC与单集群性能瓶颈	多IDC多集群架构改造	精细化透明化管理
解决思路		快速加机器 与 特定作业优化	机房迁移算力扩张 与 引擎升级算力挖潜	分而治之更大程度水平扩展	分久必合 与 流式消峰填谷

总结(b)



图a: 核心数据产出时间(三年)



图b: 数据就绪率(两年)

陌陌数据基础架构演进

1.0 阶段 2014 ~ 2016	2.0 阶段 2017	3.0 阶段 2018	4.0 阶段 2019
自给自足满足数仓分析	工具化自助查询数据	数据与算力开放	数据服务化

欢迎对我们做的事感兴趣的同学加入我们!

Thanks