Flink Runtime 核心架构

高赟 · 阿里巴巴 / 高级开发工程师

Apache Flink Community China

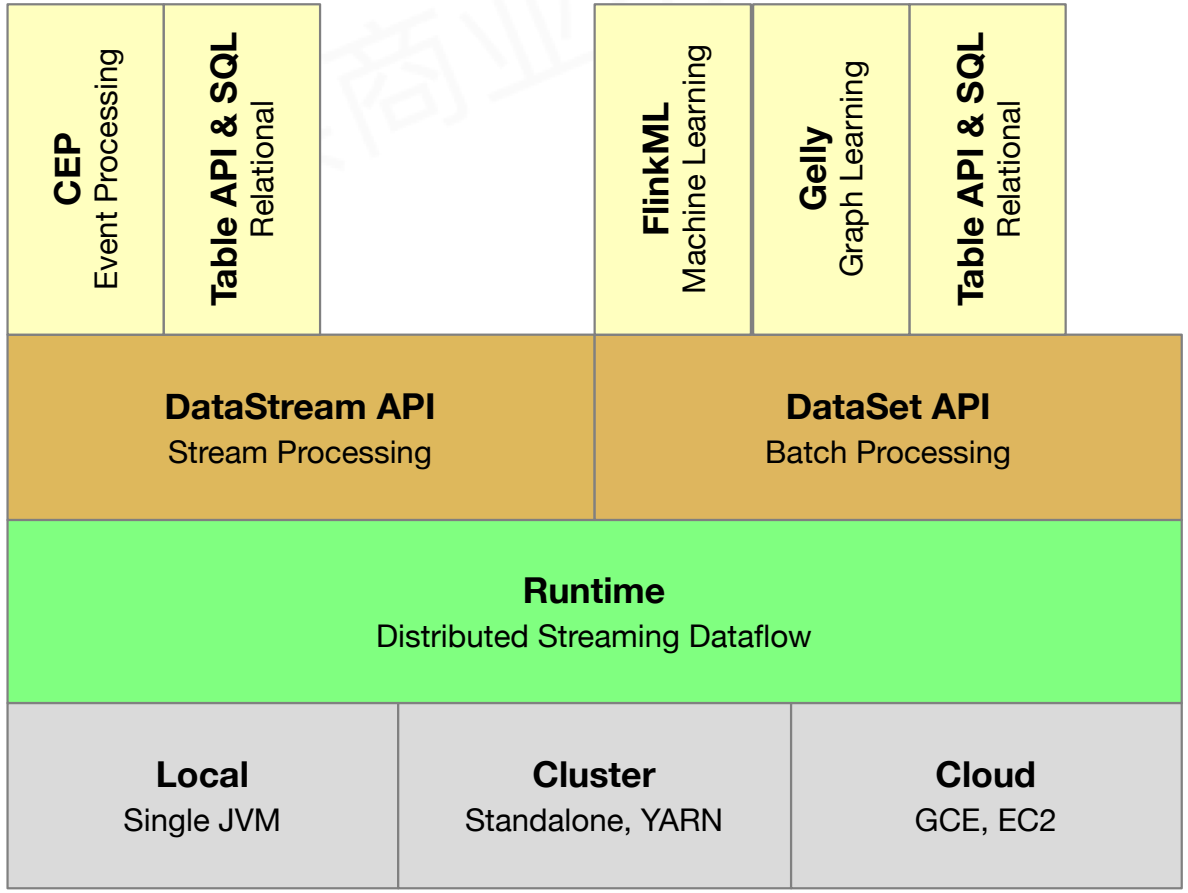
CONTENT

目录 >>

01

整体架构

Flink整体架构



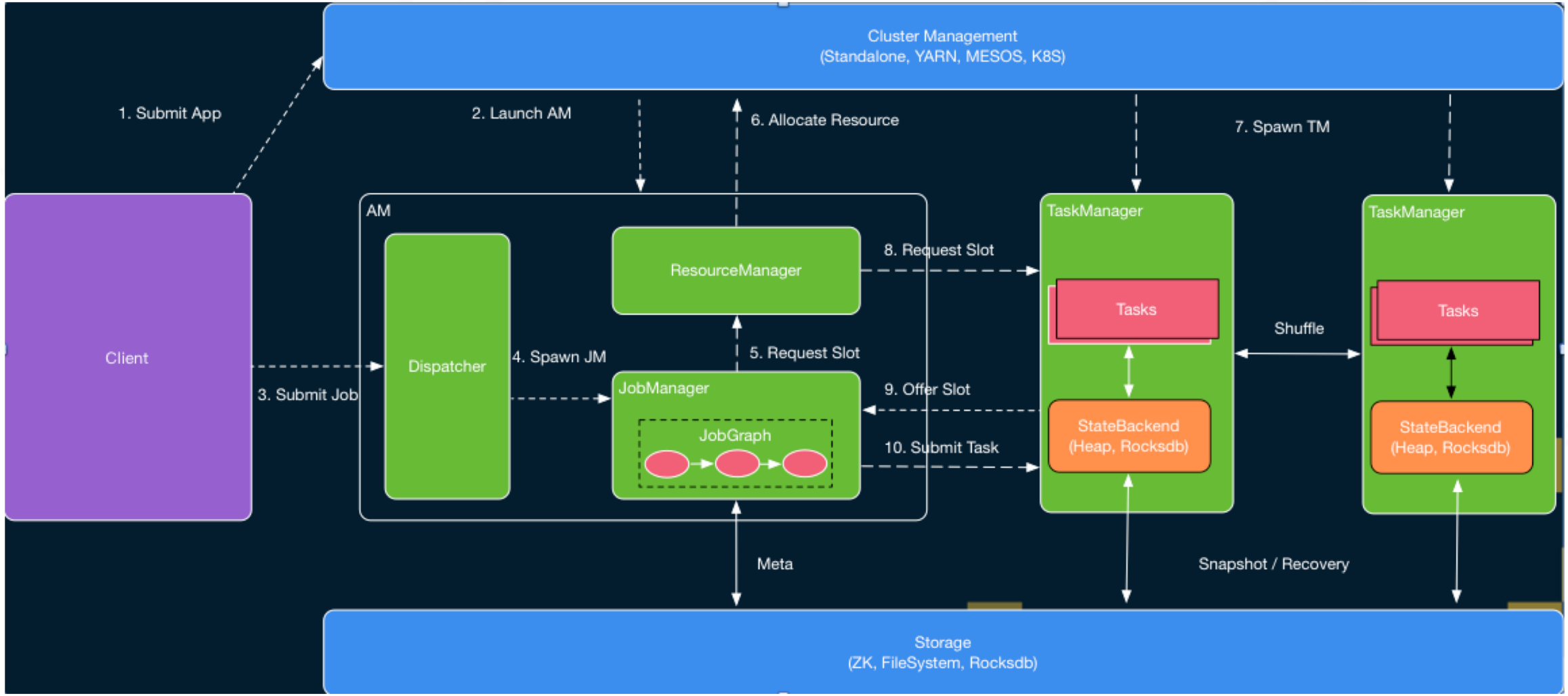
物理资源层

Runtime统一执行引擎

API层

High-level API层

Runtime层总体架构



运行模式

Per-Job

- 独享Dispatcher与Resource Manager
- 按需要申请资源（即TaskExecutor）
- 适合执行时间较长的大作业

Session

- 共享Dispatcher和Resource Manager
- 共享资源（即TaskExecutor）
- 适合规模小，执行时间短的作业

02

资源管理与作业调度

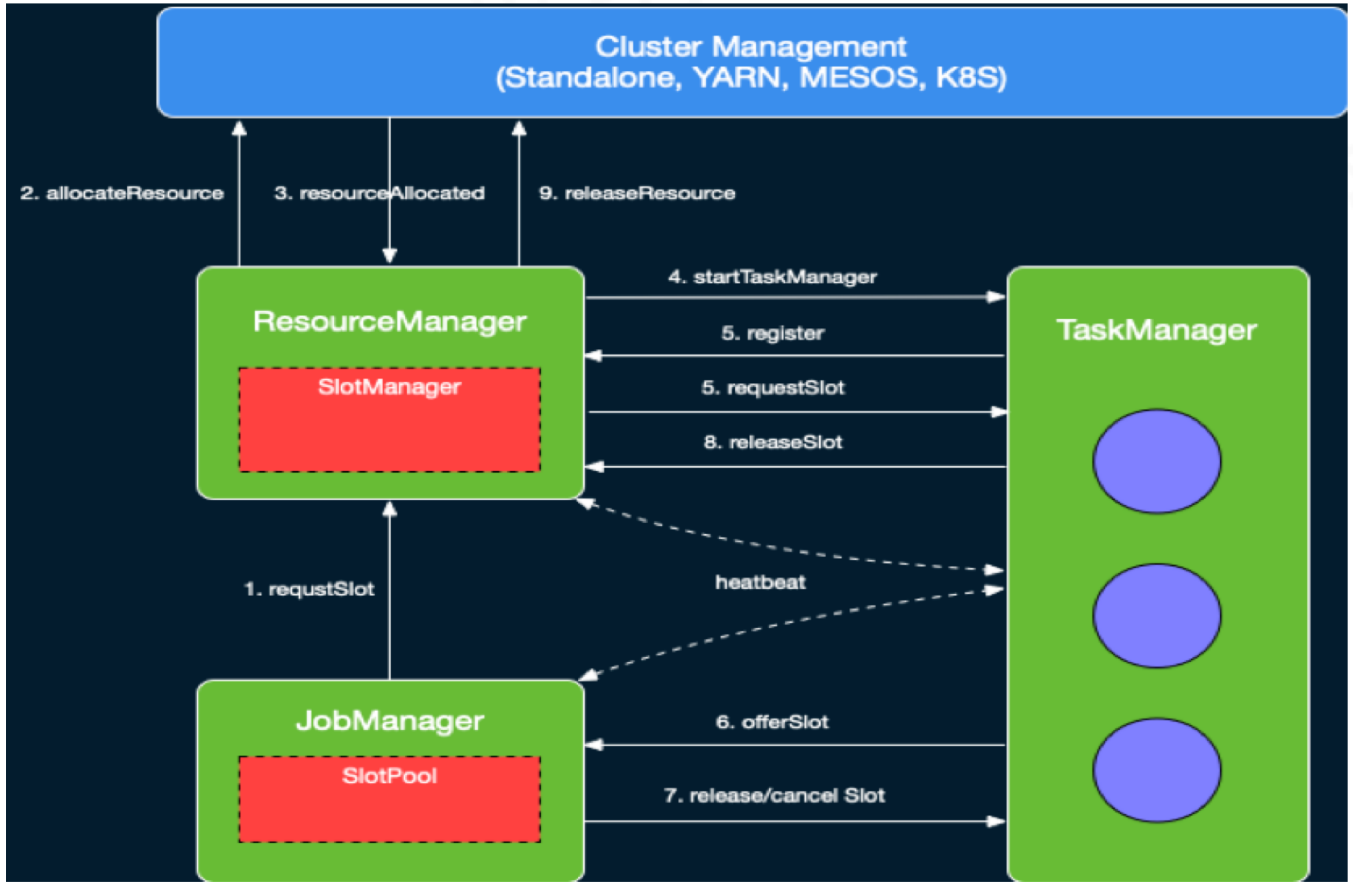
资源管理与任务调度

- 调度：任务与资源匹配

- 资源：Slot
  - 大小（未启用）
  - Location
  - 每个TaskManager包含一个或多个Slot

- 任务
  - 大小（未启用）
  - CoLocation Constraint

Slot管理



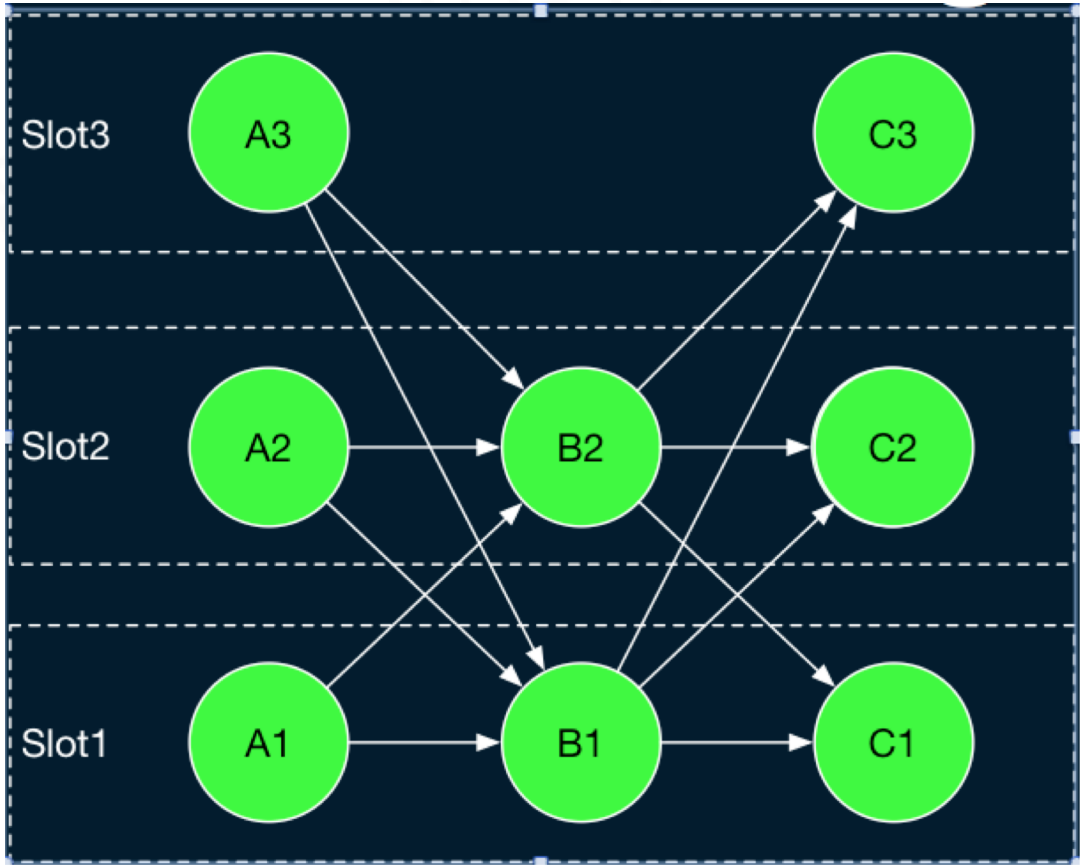
- ResourceManager
  - Slot Manager
  - 管理Slot状态
  - 分配Slot资源
- TaskExecutor
  - 实际持有Slot资源
- JobMaster
  - Slot资源的申请者

Slot划分

- 社区版未启用Slot大小
  - TM启动固定个数的Slot
  - 默认每个Task占用一个Slot
    - 可以通过Slot Sharing在单个Slot中

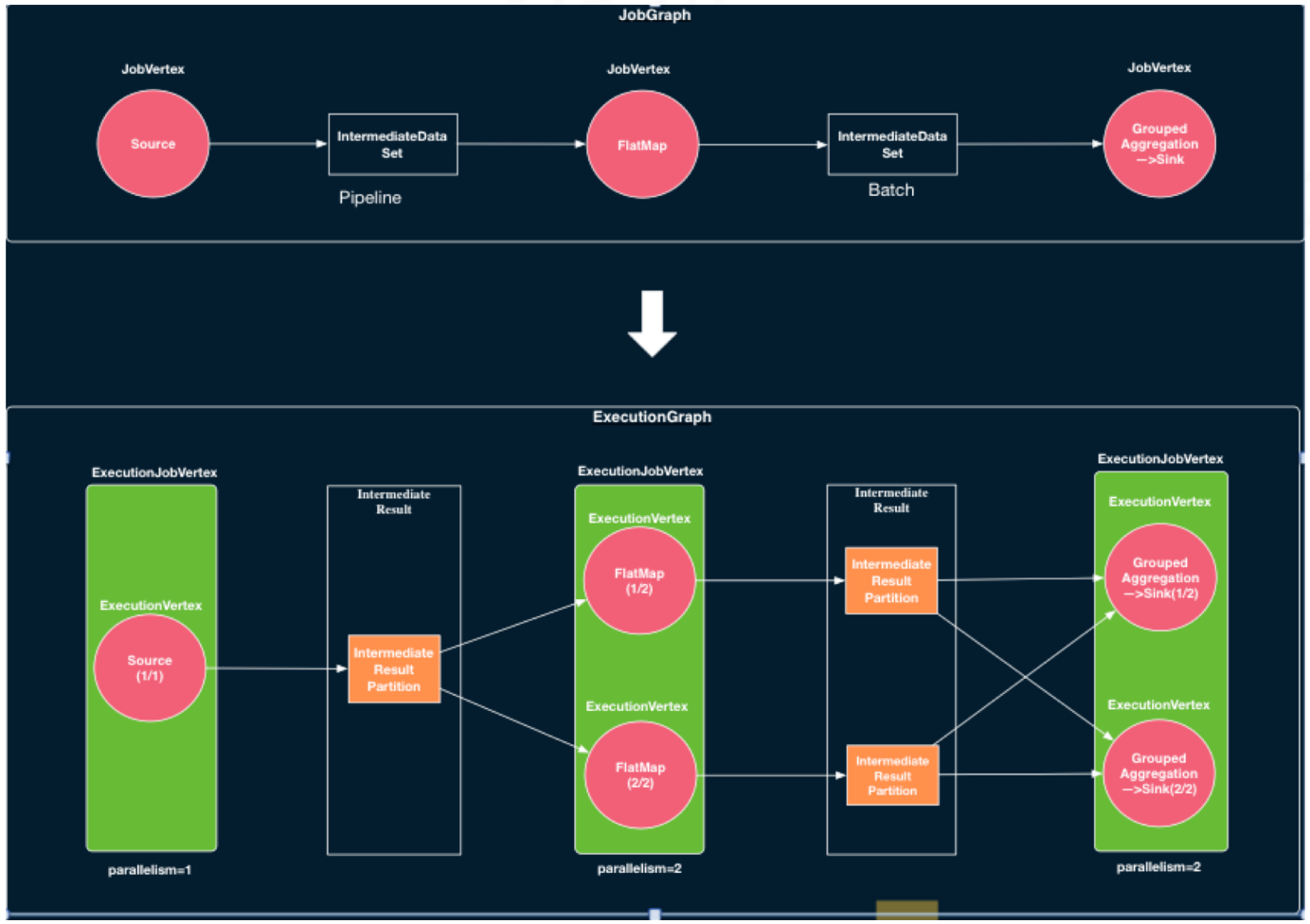
- Blink开源版
  - 增加Slot资源严格匹配
  - ResourceProfile: (CPU, Heap, …)
  - Per-Job模式：根据Task资源申请合适的Slot资源
  - Session模式：从预先申请好的资源中扣除

Slot Sharing

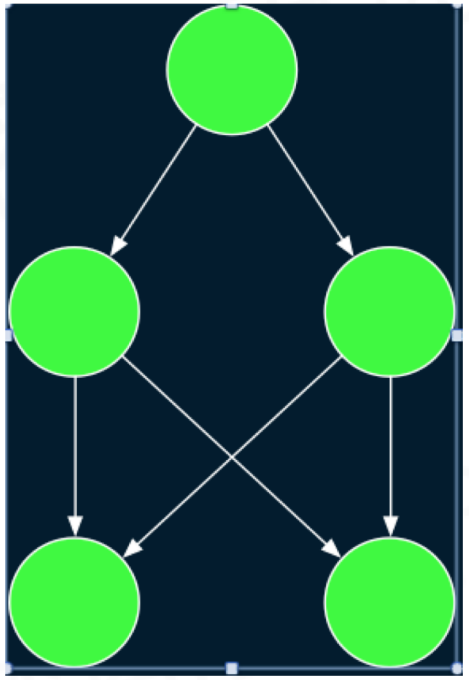


- 单个Slot中部署多个Task
  - 同一Vertex的多个Task不能共享Slot

作业DAG图结构

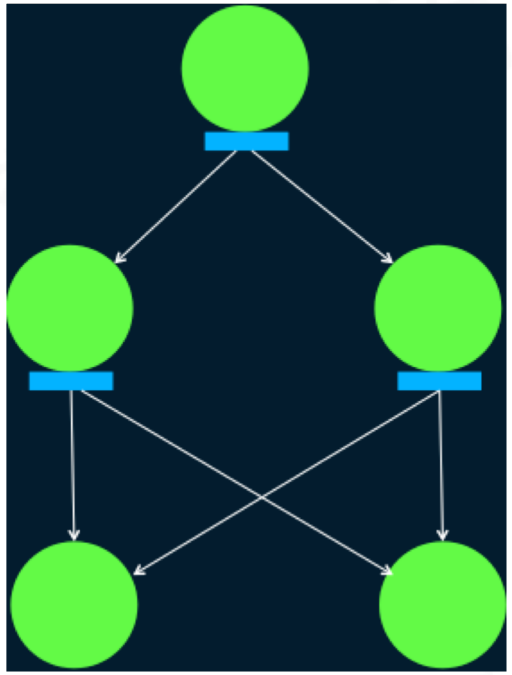


- JobGraph
  - 客户端提交的作业DAG图结构
  - 逻辑结构，不考虑并发
- ExecutionGraph
  - JobMaster实际维护的数据结构
  - 物理结构，考虑并发



Eager调度

- 适用于流作业
- 一次性调度所有的Task



LAZY_FROM_SOURCE

- 适用于批作业
- 上游作业执行完成后，调度下游的作业

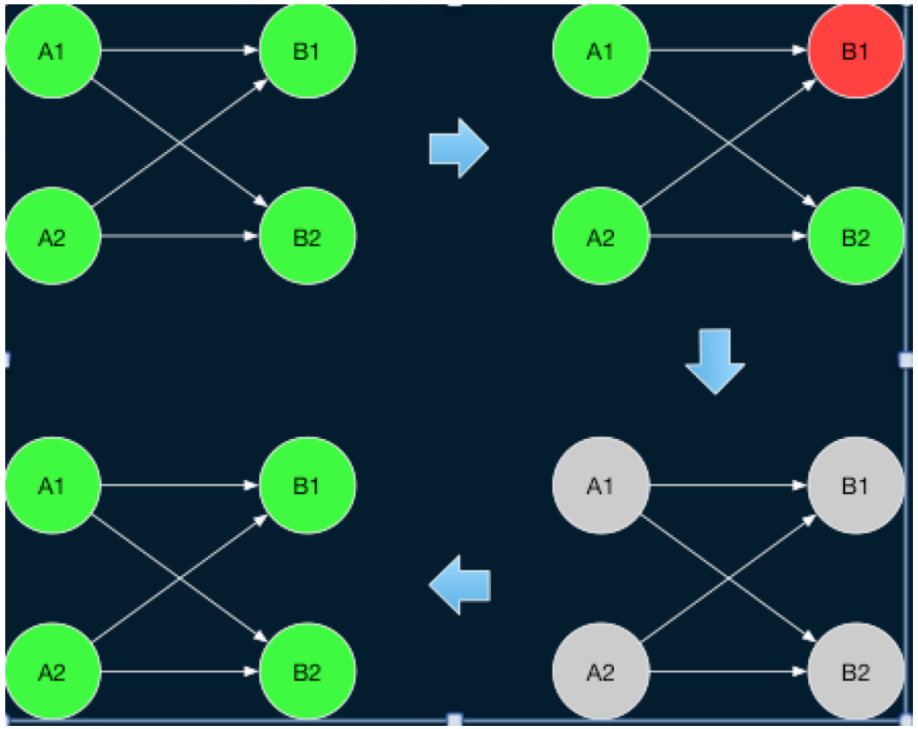
03

错误恢复

- Task Failover
  - 单个Task执行失败或TM出错退出等
  - 可以有多种不同的恢复策略

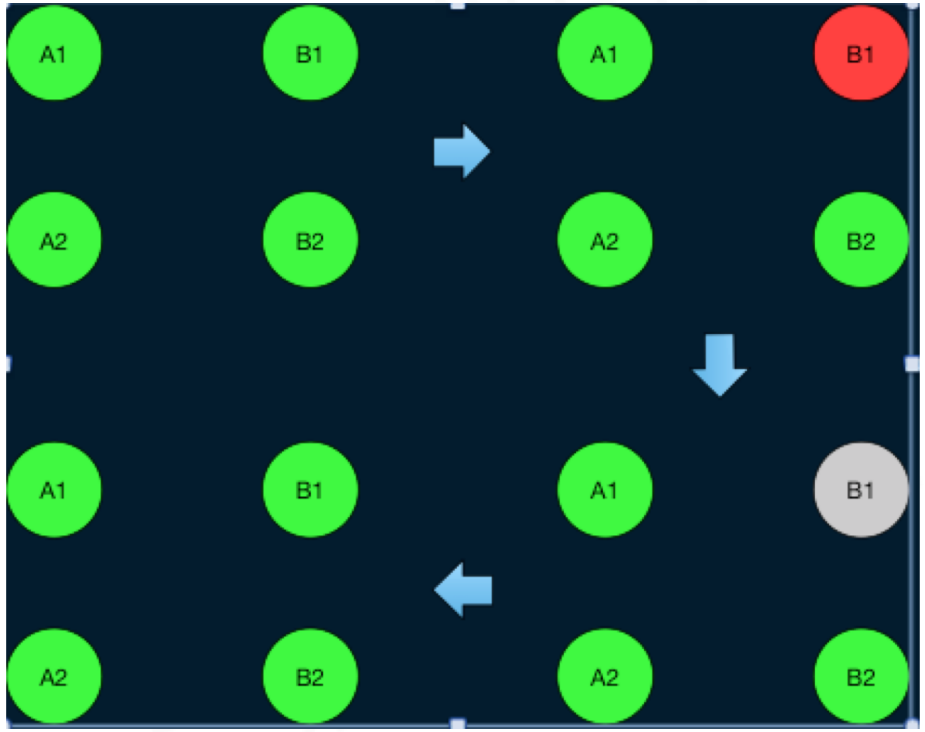
- Master Failover
  - AM执行失败

Task Failover: Restart-all



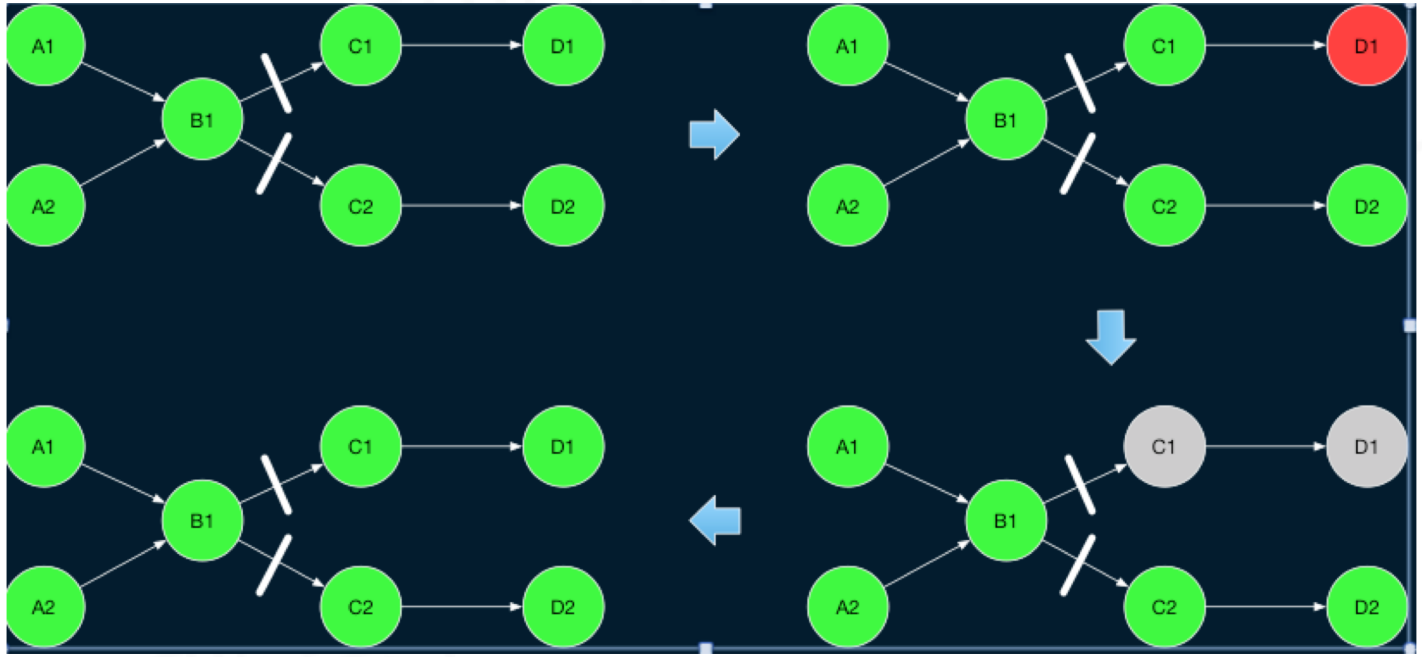
- 重启所有Task
  - 从上次的Checkpoint开始重新执行

Task Failover: Restart–individual



- 只重启出错的Task
  - 只能用于Task间无连接的情况
  - 应用极为有限

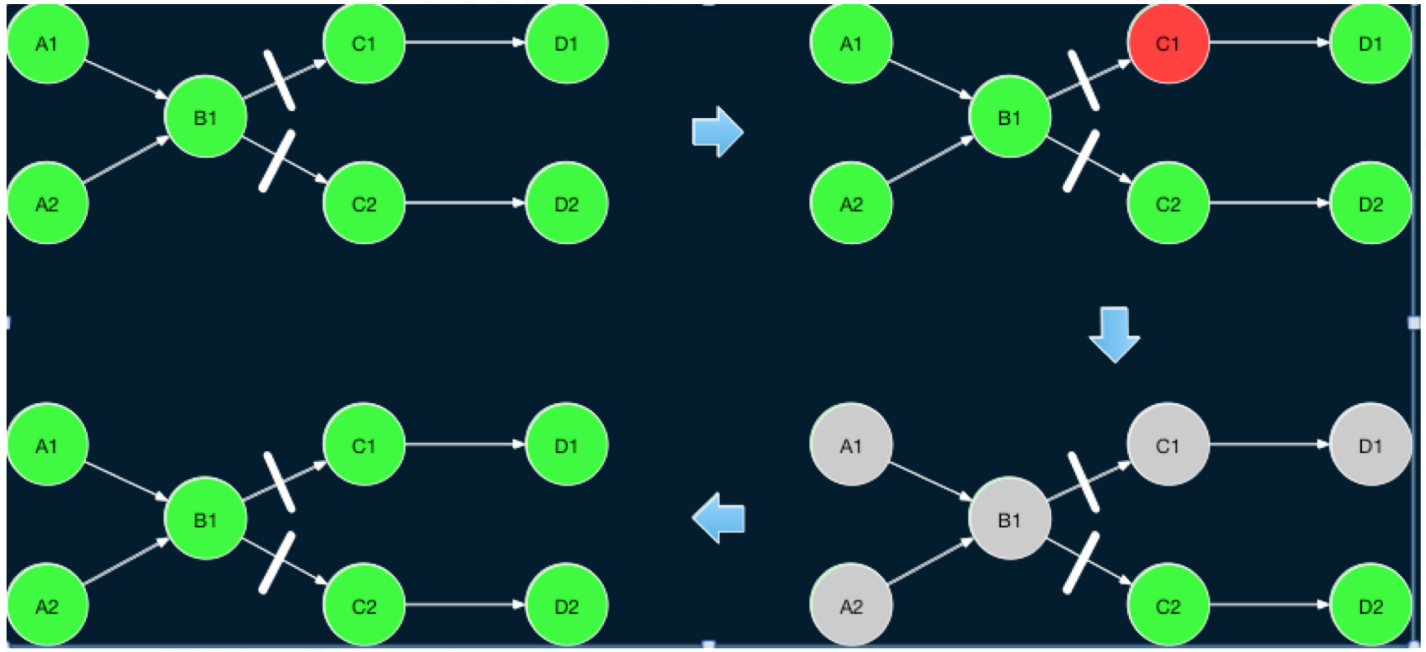
Task Failover: Restart-Region



- 重启Pipeline Region
  - Blocking数据落盘，可以直接读取
  - 逻辑上仅需重启通过Pipeline边关联的Task

- 两种错误类型
  - 作业自身执行失败
  - 作业读取上游数据失败

Task Failover: Restart-Region



- 重启Pipeline Region
  - Blocking数据落盘，可以直接读取
  - 逻辑上仅需重启通过Pipeline边关联的Task
- 两种错误类型
  - 作业自身执行失败
  - 作业读取上游数据失败

- Master Failover
  - 多个Master通过ZK进行选主
  - 目前Master Failover要求全图重启

04

未来展望

- 完善的资源管理
- 统一Stream和Batch
- 更灵活的调度策略
- Master Failover优化

THANKS

Flink China社区大群

扫一扫群二维码，立刻加入该群。