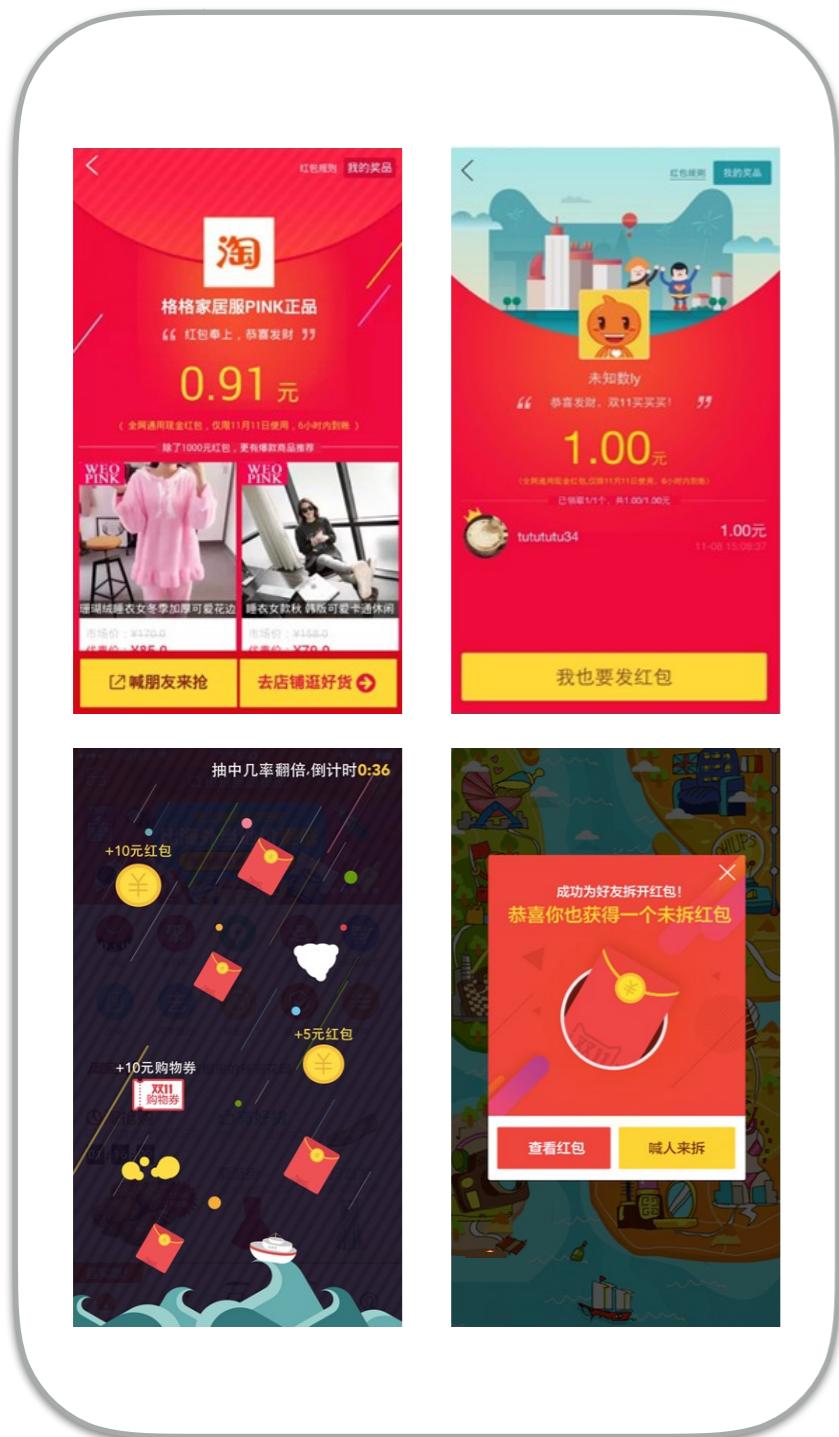




# 支付宝红包 ——双11的挑战与应对

蚂蚁金服-支付核算技术部  
邱硕

# 活动创建



红包发放



查询红包



红包渲染、支付

# 大纲

---

- 链路梳理
- 性能优化
- 容量评估
- 业务量评估
- 系统保护



# 链路梳理



## 上下游链路

- 跨机房调用
- 公共组件访问量
- 接口上游入口
- 下游系统**热点**
- 重复操作

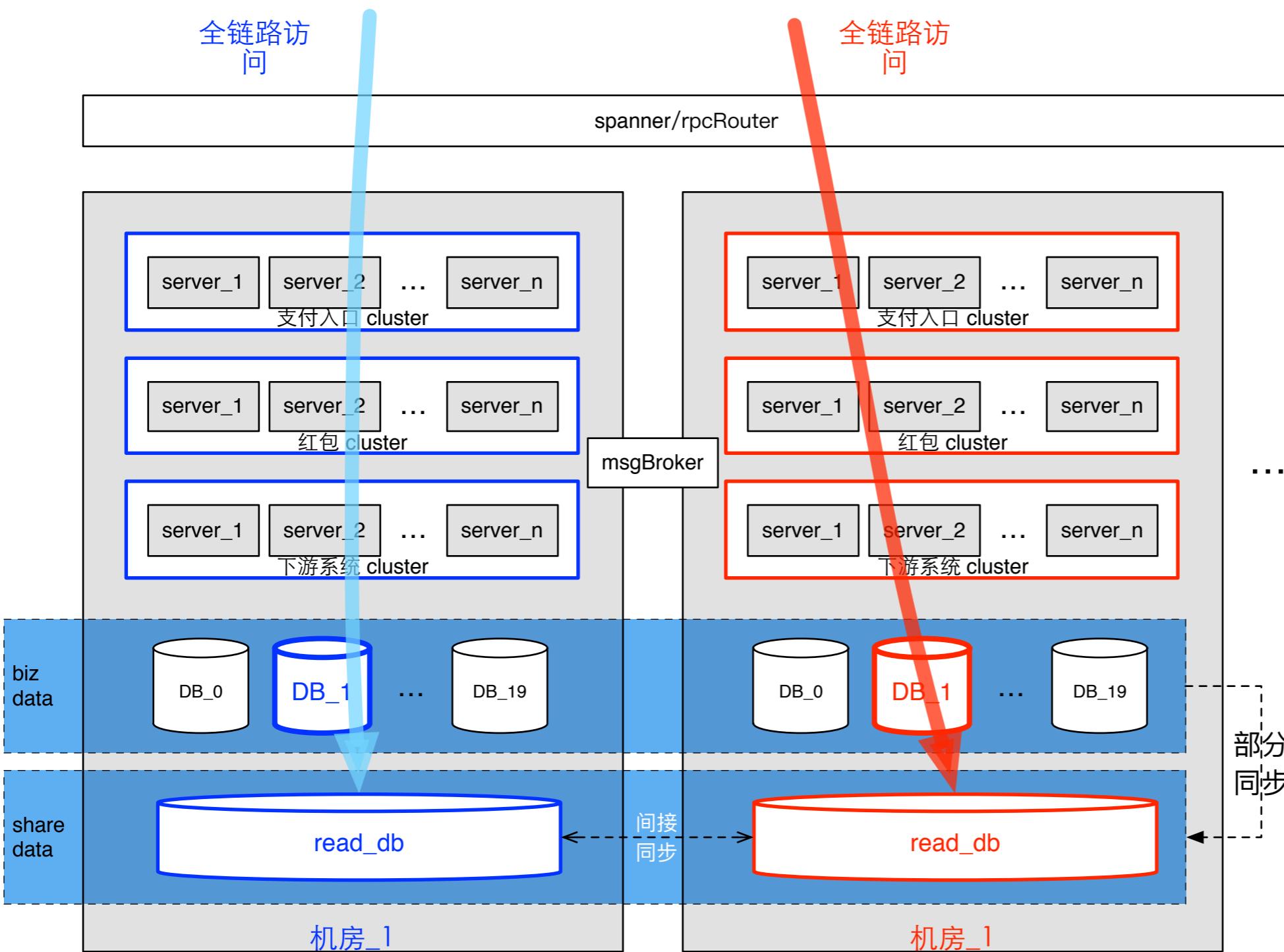
## 红包内部热点

- 关键链路性能缺陷
- 连接池/线程池依赖关系
- 存储分片**热点**
- 数据库记录**热点**



# 机房级拆分

**本地化**: 一个机房包含一次业务请求的全部数据和服务器



**全站同分布**: 不同业务请求根据数据分布路由到不同机房；一次业务请求 (e.g. 使用红包进行支付) 全链路的所有系统数据 (有状态主体) 数据分布相同

**共享数据副本**: 无法同分布的共享类数据读写分离，每个机房本地访问副本

连接池瓶颈缓解  
低响应的异地部署



发放(领用)  
红包

预算控制单点



支付前渲染  
可用红包

支付规则管理  
模板数据跨库



支付时  
红包抵扣

单笔支付多红包

# 预算控制单点——乐观性锁



```
start transaction  
lock budget record and get its value  
(业务层余额判断余额足够);  
subtract budget record by Δ  
insert business orders...  
commit
```

加锁时间

单预算吞吐量：  
50 tps → **800** tps

缩短加锁时间

```
start transaction;  
lock budget record and get its value;  
<业务层余额判断余额不足>;  
rollback;
```

加锁时间

```
start transaction;  
insert business orders...  
subtract budget record by Δ  
when value is sufficient  
commit
```

加锁  
时间

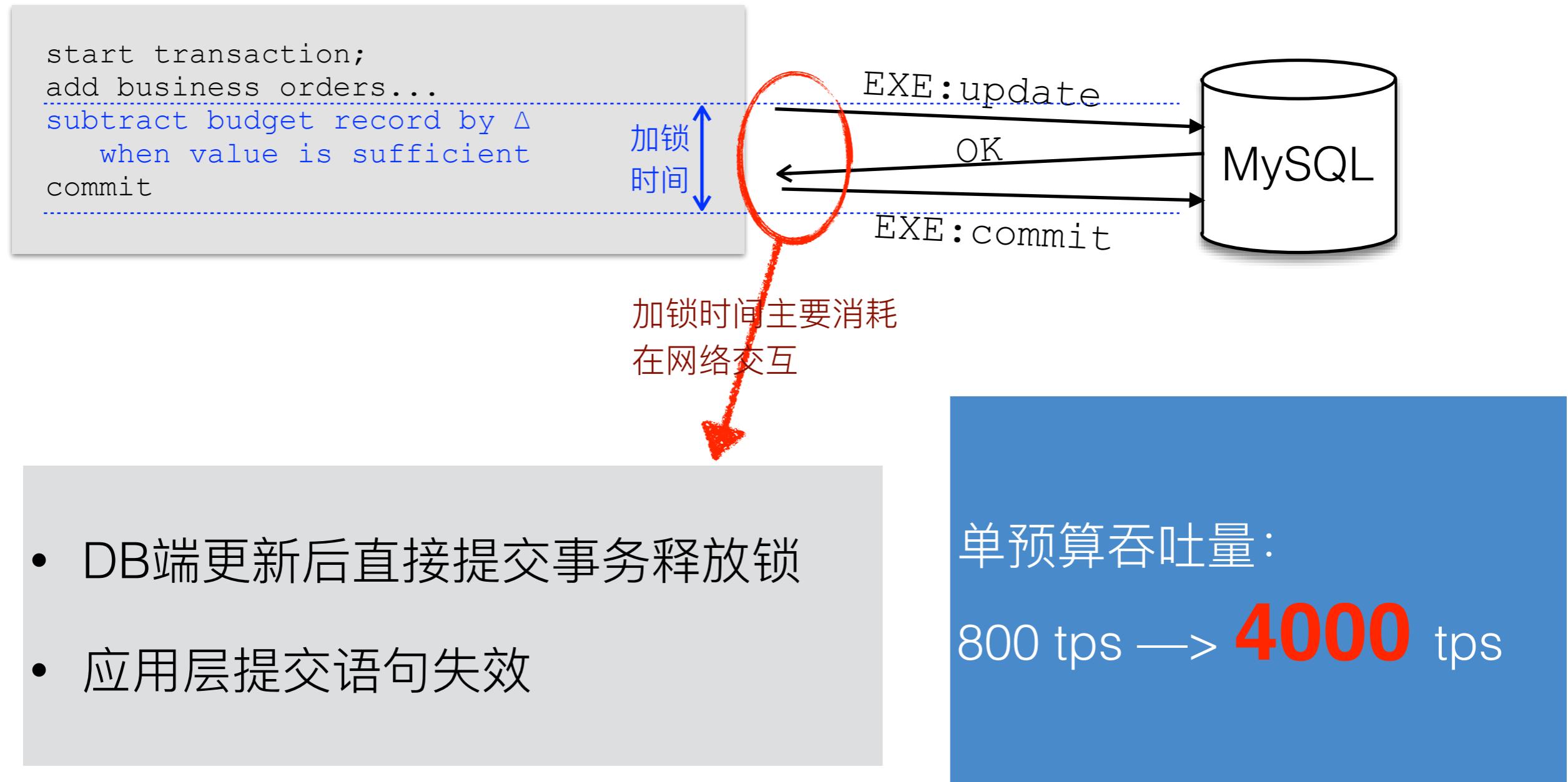
缩短加锁时间

```
start transaction  
insert business orders...  
subtract budget record by Δ  
when value is sufficient  
rollback;
```

更新  
失败  
不会  
加锁

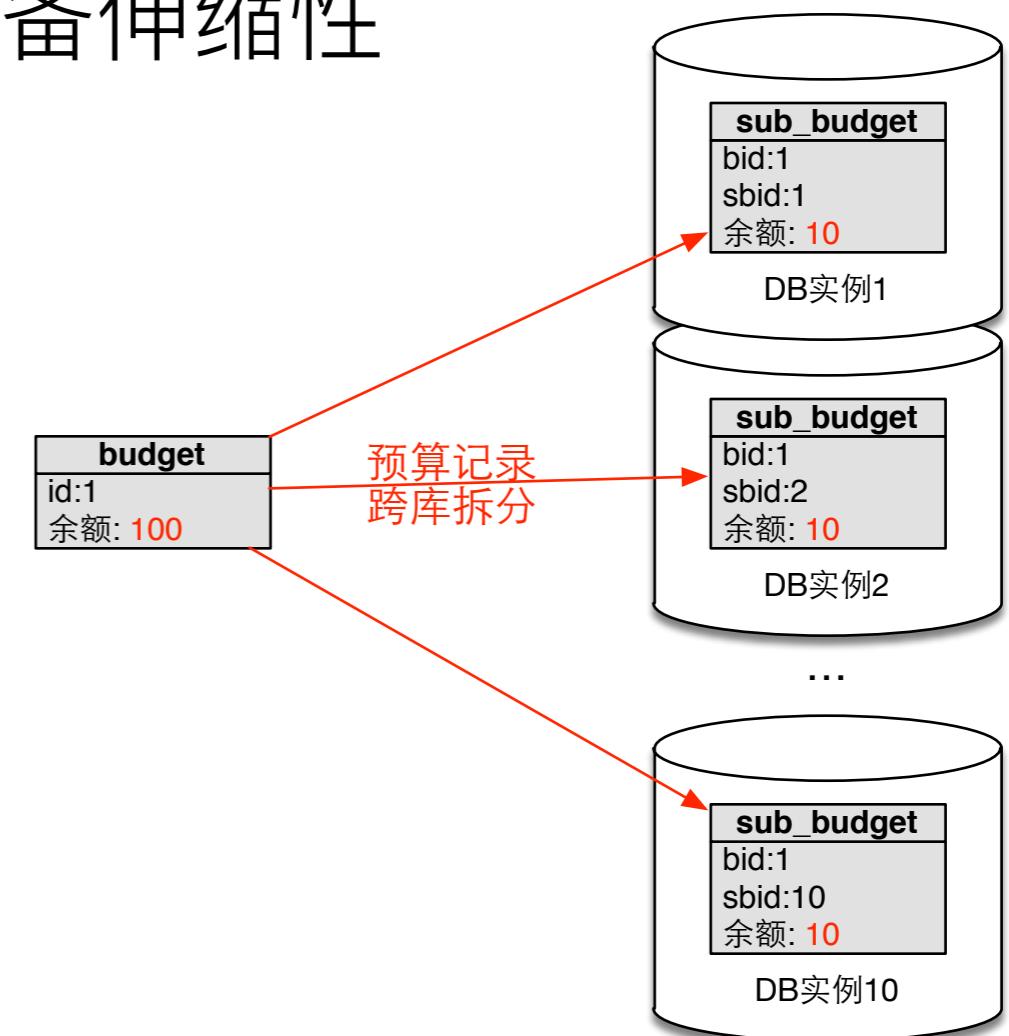
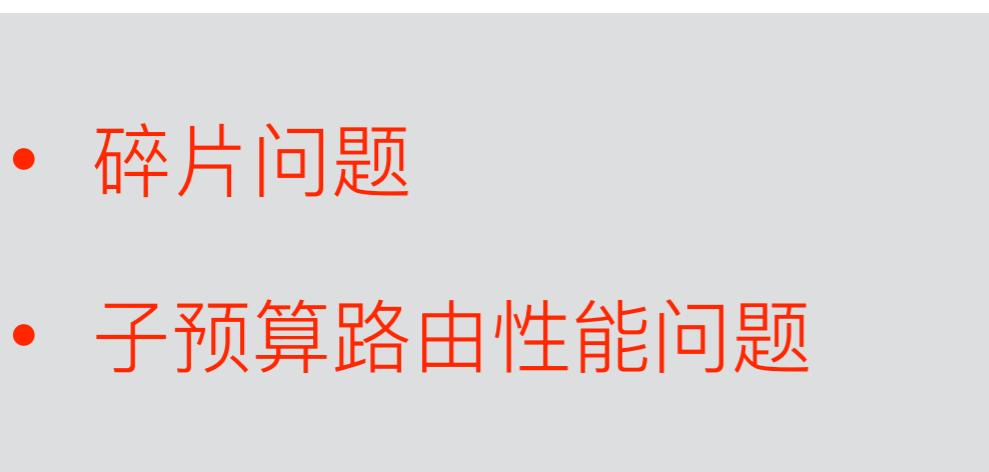
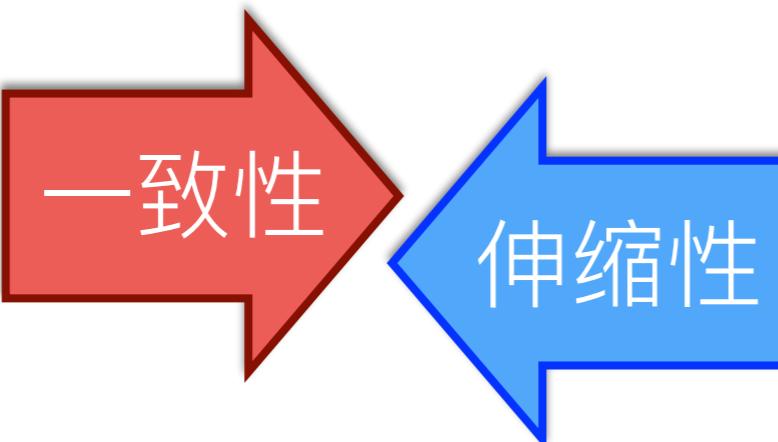


# 预算控制单点——MySQL patch

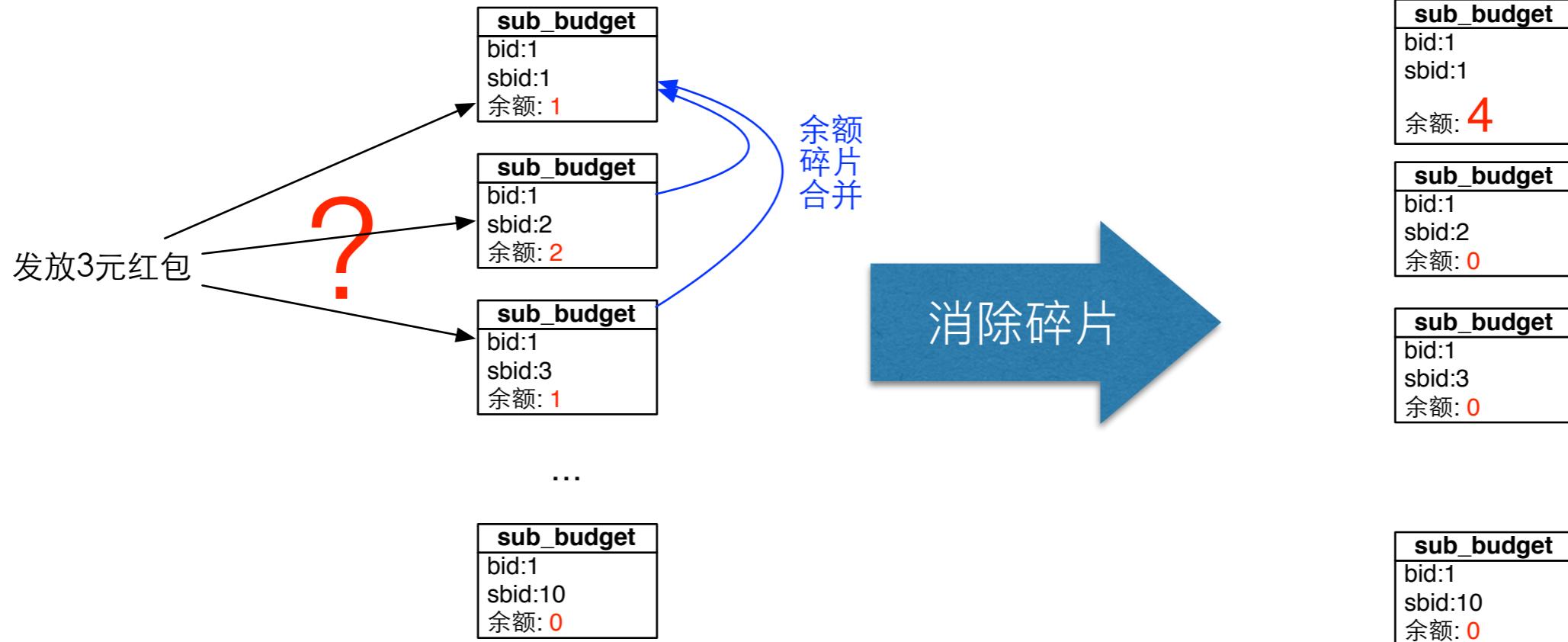


# 预算控制单点——预算拆分

- 预算水平拆分，子预算集合具备伸缩性
- 子预算和DB物理库映射



# 碎片问题



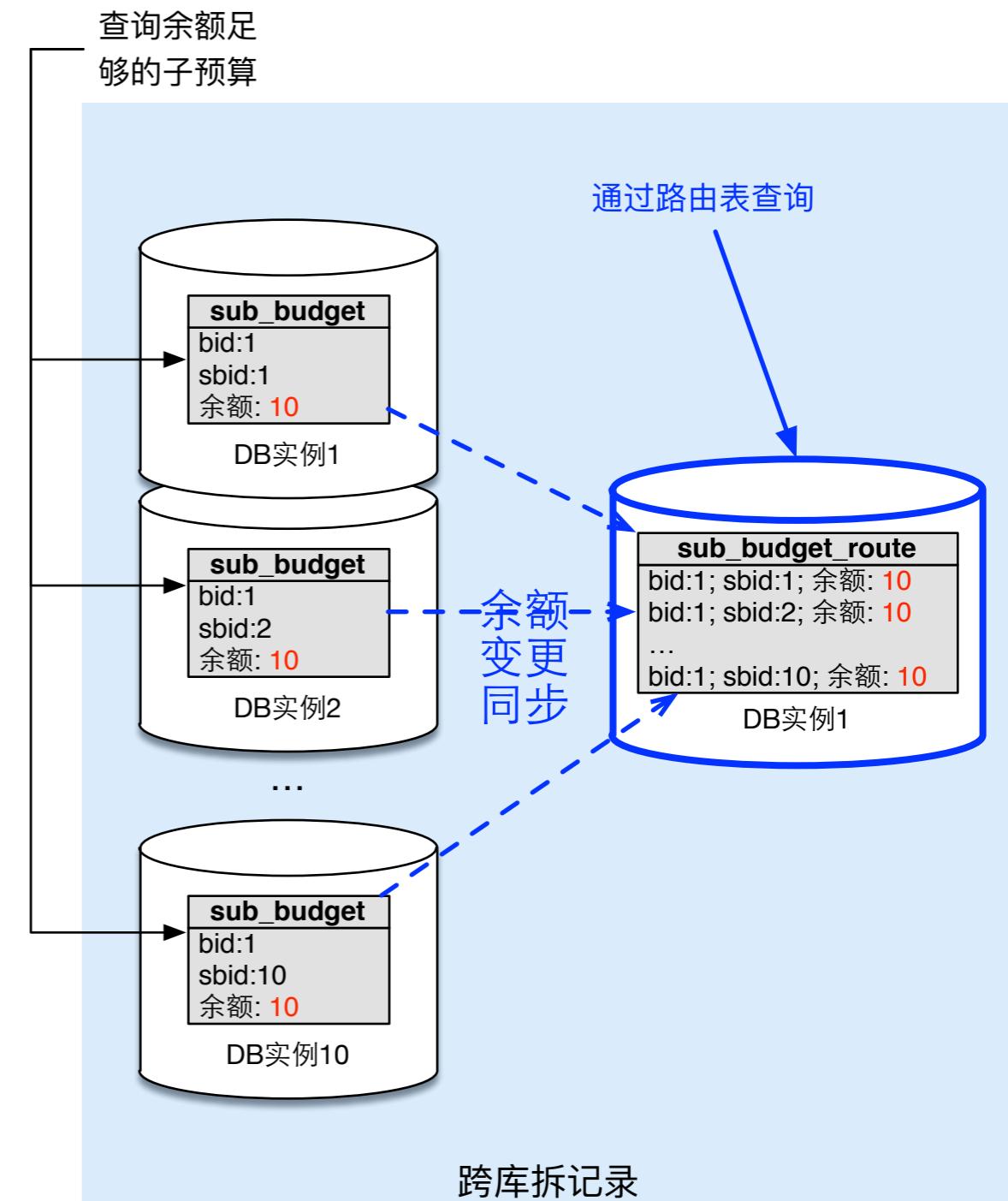
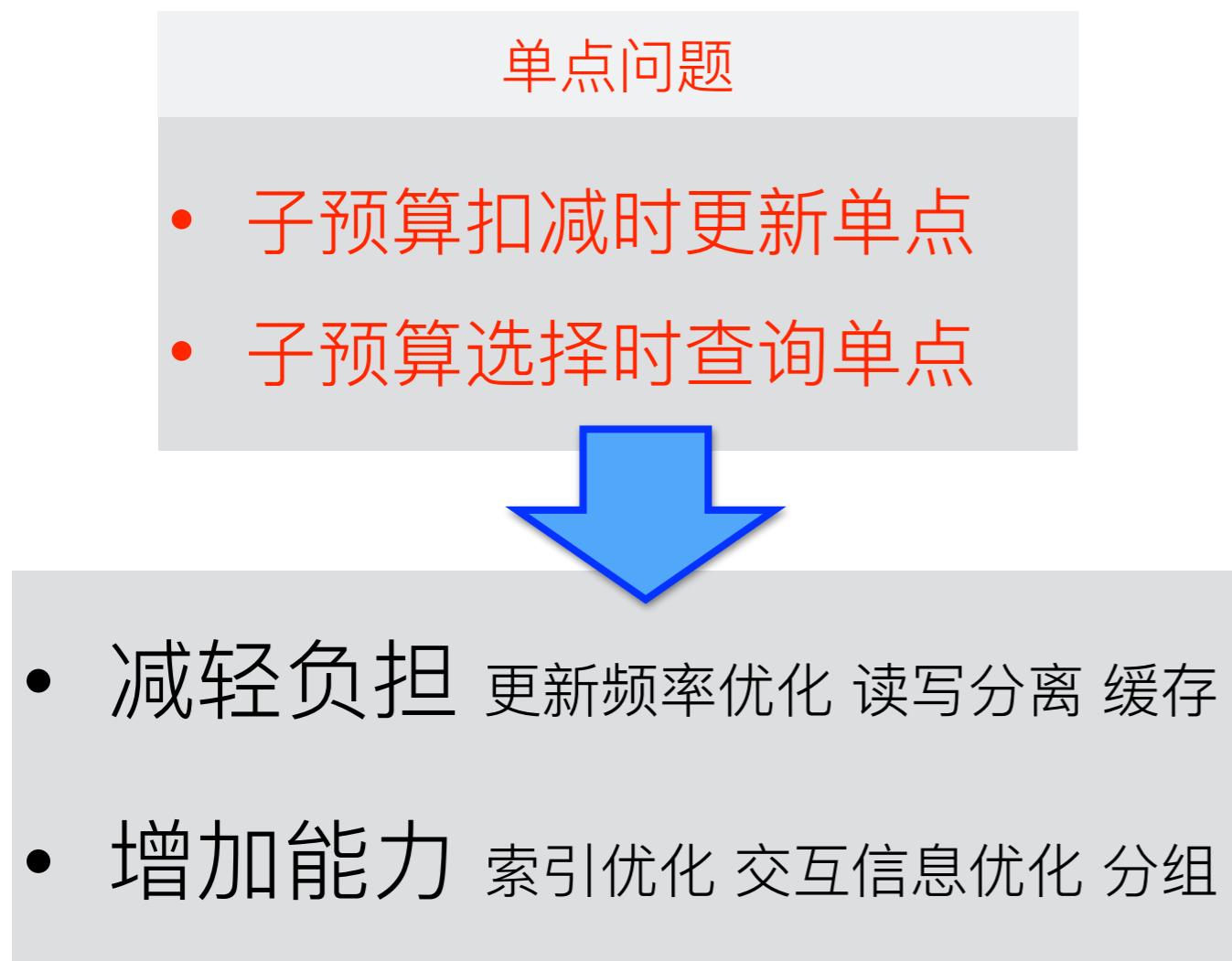
- 扣减发生后异步触发合并
- 分布式事务一致性?

- 后台定时任务补偿机制
- 合并订单唯一性控制



# 子预算的路由

- 跨库查询到路由表查询

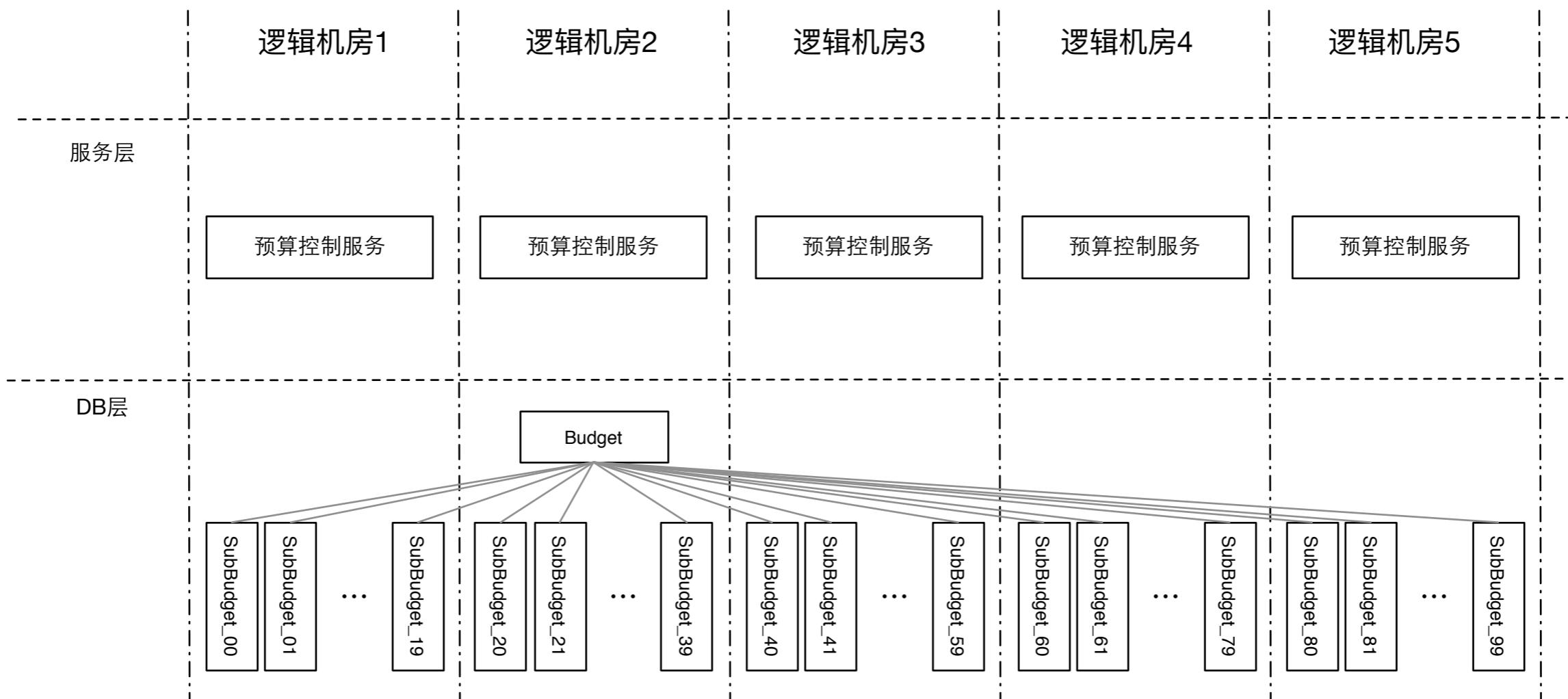


# 子预算的路由

- 跨库查询到路由表查询

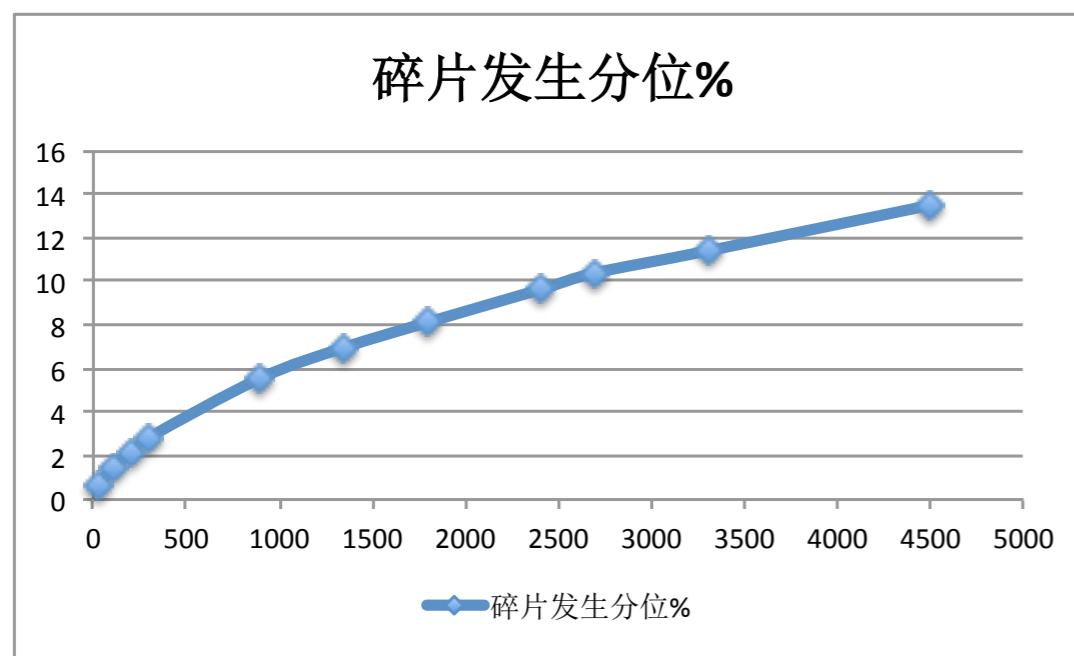


# 预算拆分——解决跨机房问题



# 突破50,000 tps

- 路由表查询单点
- 碎片分位上限



去DB的预算控制-挑战

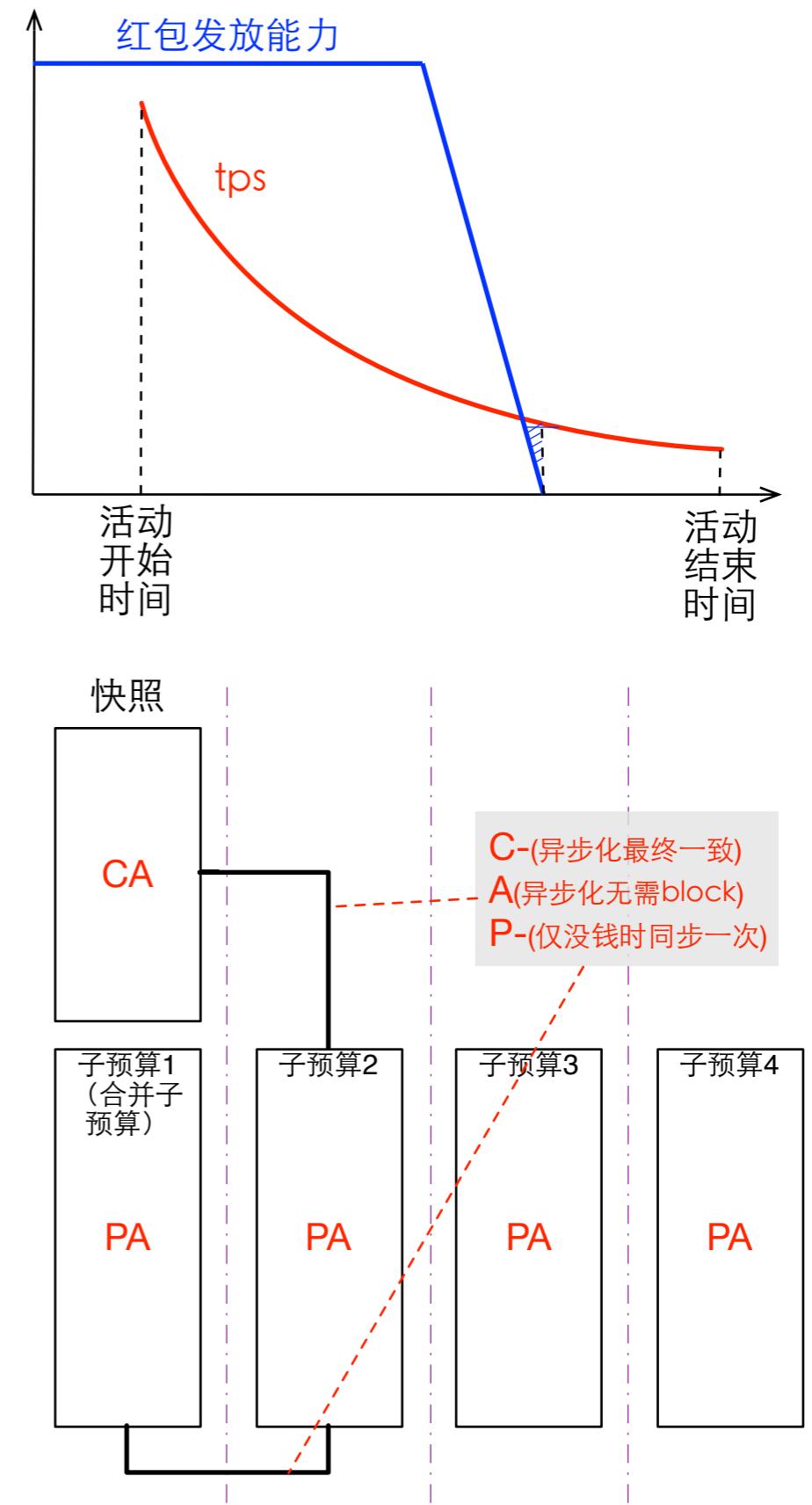
服务治理 SOA环境下有状态节点如何治理?

宕机一致性 无DB事务的一致性如何保证?

目前已实现：**500,000+** tps

# CAP解释

- **Consistency**
  - C1: 有预算时可以发红包 (弃)
  - C2: 没预算时不能发红包 (保)
- **Availability**
  - 每次发红包的预算控制在短时间内返回
- **Partition-Tolerance**
  - 子预算间相互无感知





发放(领用)  
红包

预算控制单点



支付前渲染  
可用红包

支付规则管理  
模板数据跨库



支付时  
红包抵扣

单笔支付多红包

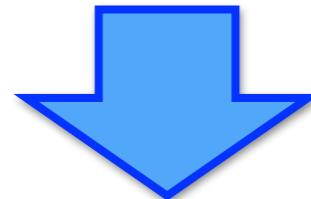
# 支付规则管理



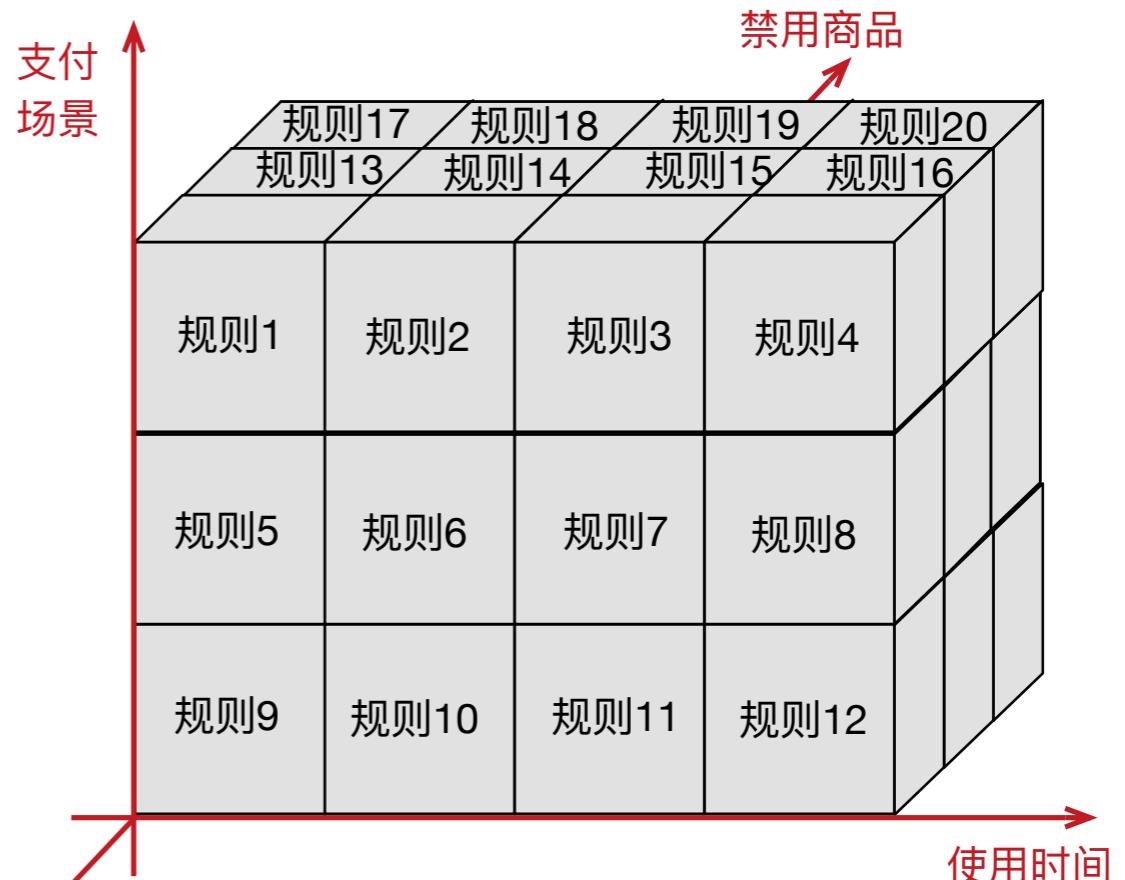
规则数量 多维度数量大

渲染响应 访问每个红包的规则

$$N_1 \times N_2 \times N_3$$



$$N_1' + N_2'$$



- 枚举值冗余进红包记录
- 各个维度拆分存储
- 主要枚举值固化 压缩值域



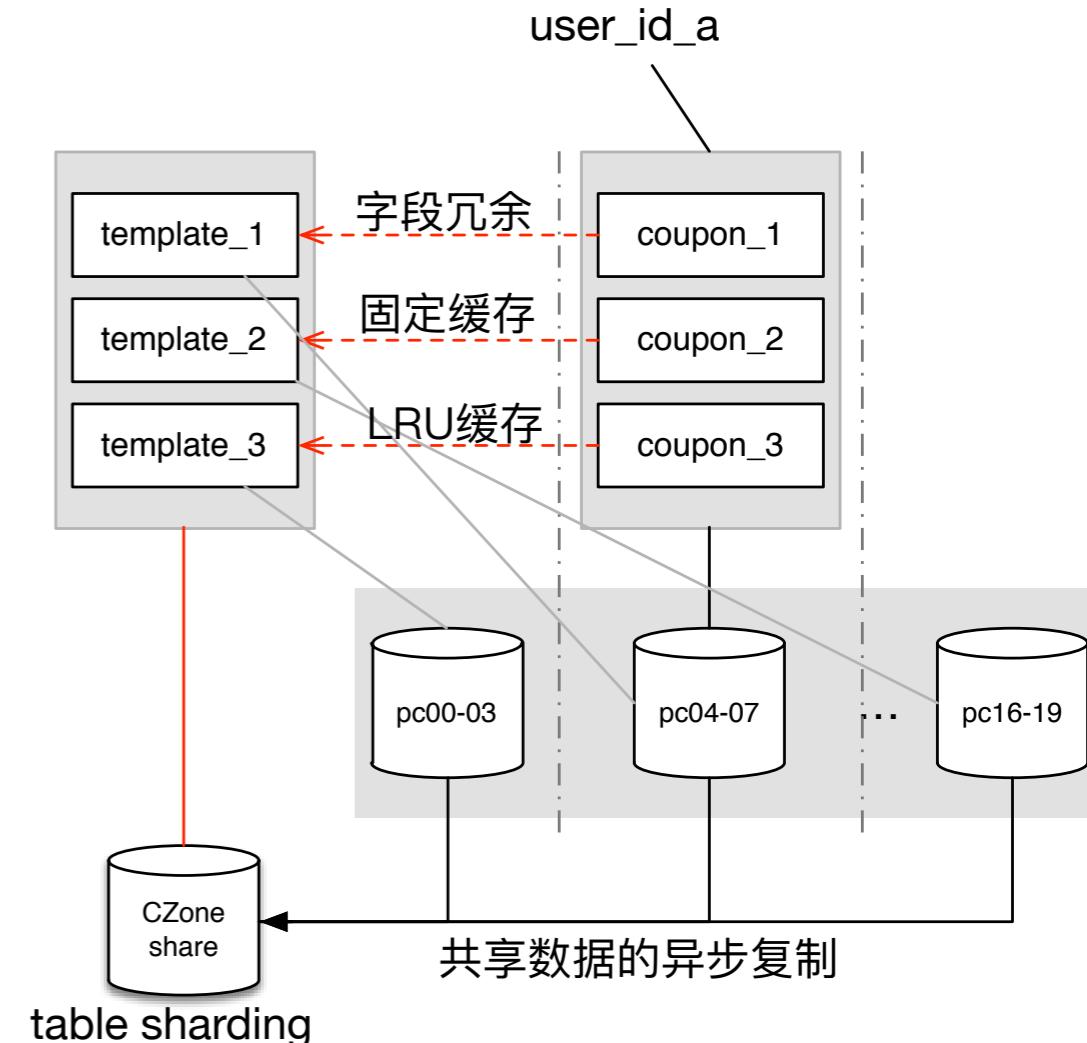
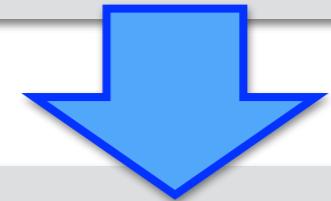
**小量规则  
利用缓存  
消除DB操作**



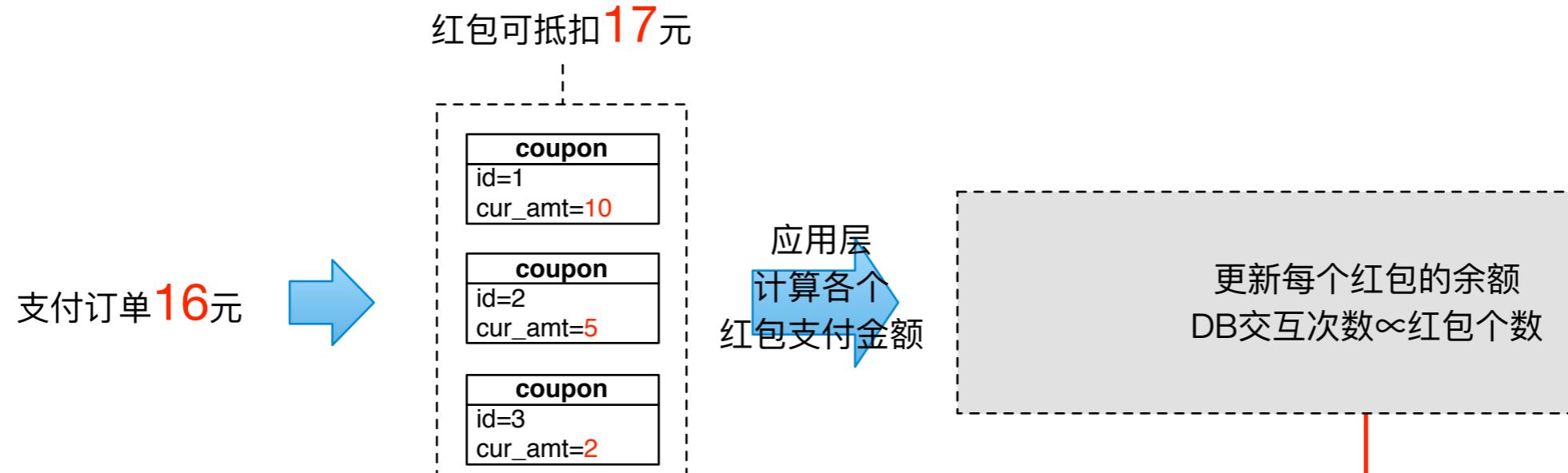
# 分级缓存



- 数据跨库分布
  - 红包按照用户分库、模板分布在共享库
  - 一次渲染**多次共享库访问**
- 字段冗余
  - C2C模板：数量预估百万级+、信息简单
- 固定内存缓存
  - 平台模板：数量可枚举、信息复杂、可变
- LRU内存缓存
  - 商家模板：数量预估十万级、大商户集中



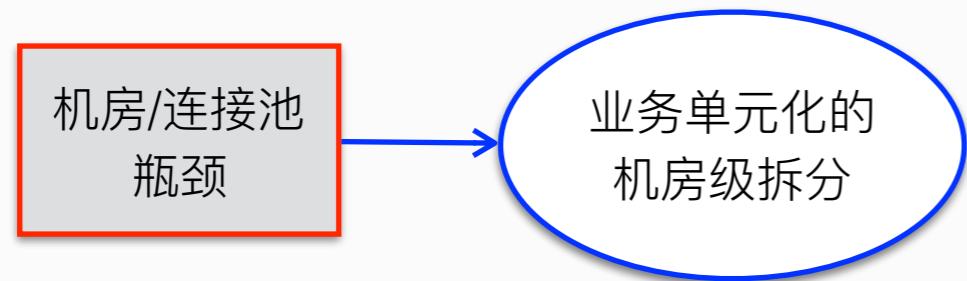
# SQL合并



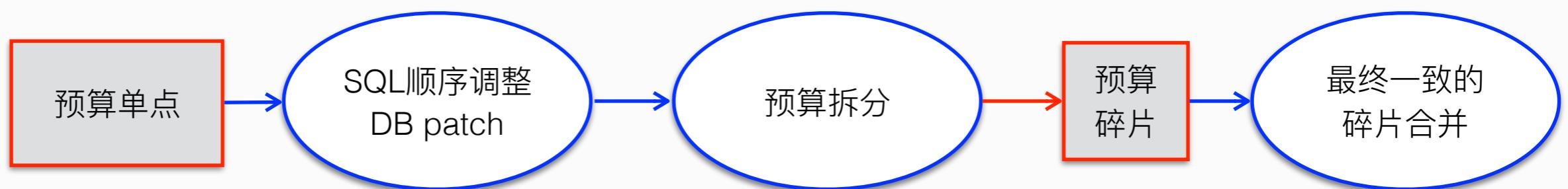
- 红包余额更新
  - 订单额较小时仅末位红包用不完
  - 订单额通常大于可用红包额：全用完
- SQL交互
  - $O(n) \rightarrow O(1)$

# 性能风险 应对总结

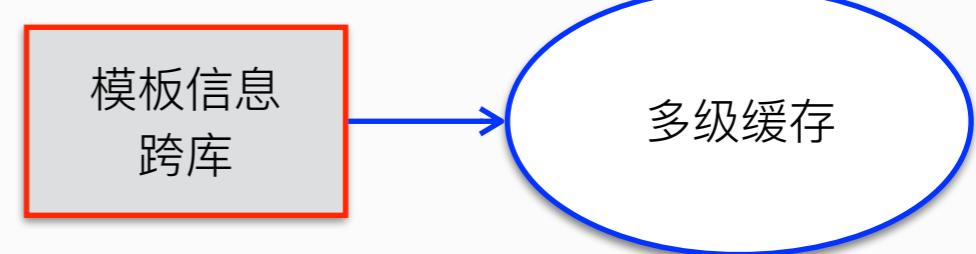
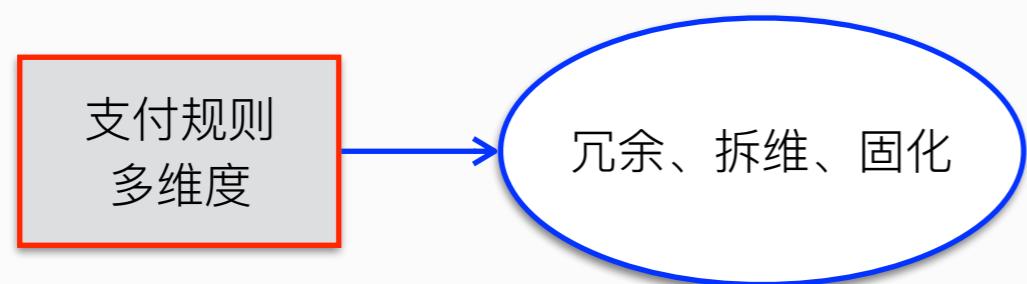
总体



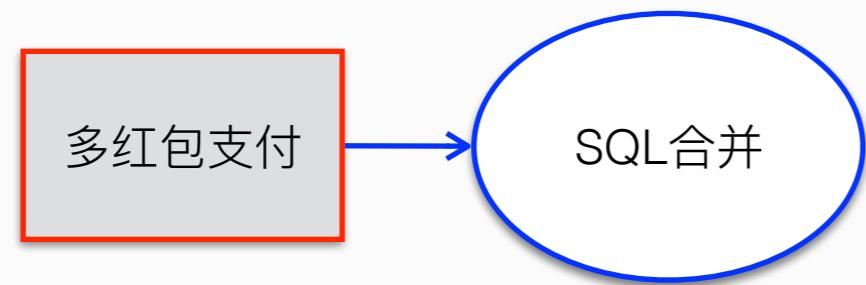
发放



渲染



支付



发放能力(单活动): 50 -> **50,000 tps**  
支付能力: **14** 倍



# 容量评估



- 反映系统真实能力
- 为关键业务配置合适的资源
- 为关键资源的保护方案提供依据



# 真实能力

- 基于线上环境压测
- 数据隔离
- 调用隔离
- 监控隔离

## 服务调用

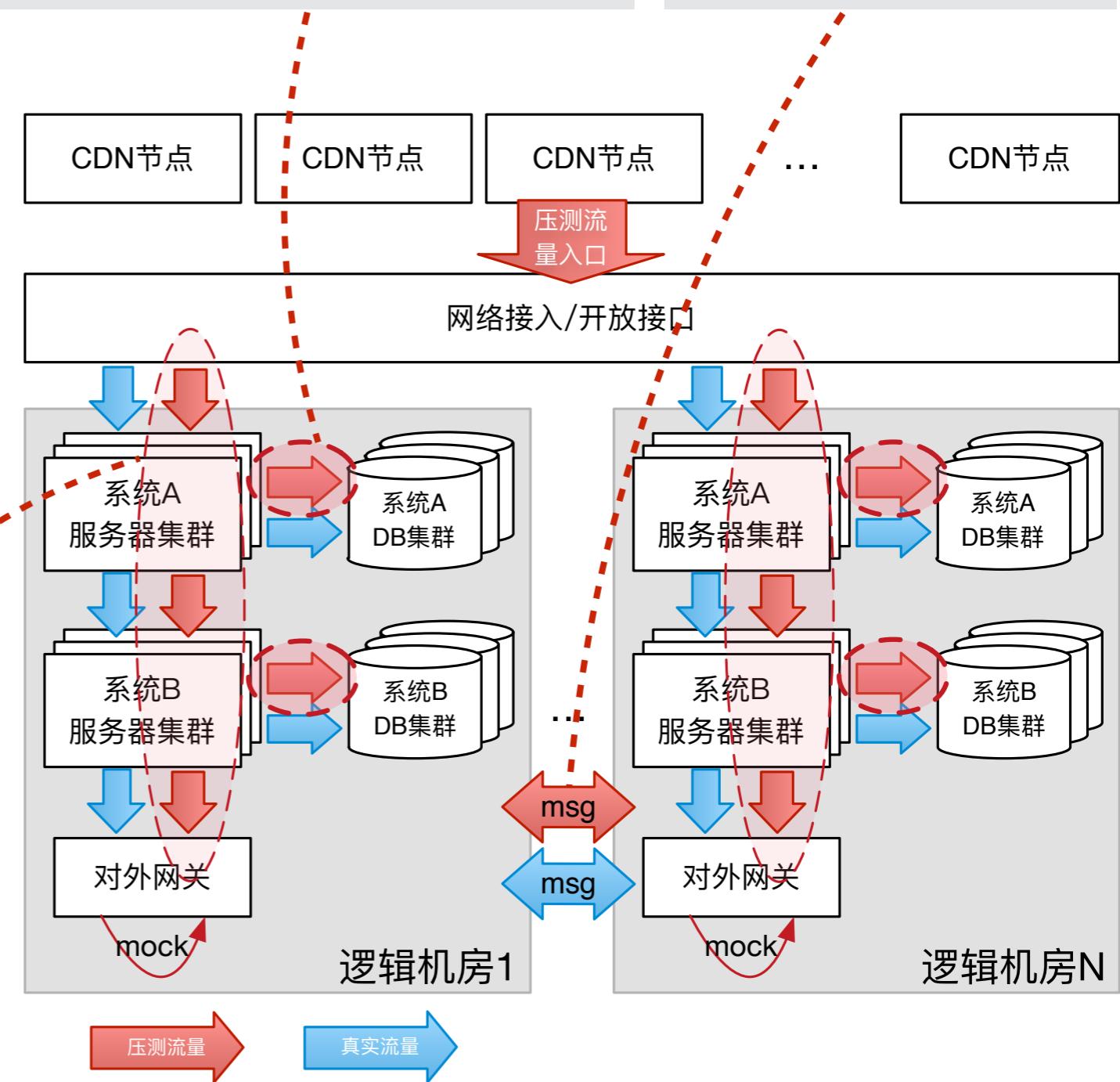
- RPC调用染色
- 跨系统上下文透传
- 用户数据隔离

## DB访问

- 数据表隔离
- 数据值隔离
- DB分表路由规则隔离

## 异步消息

- 消息数据隔离

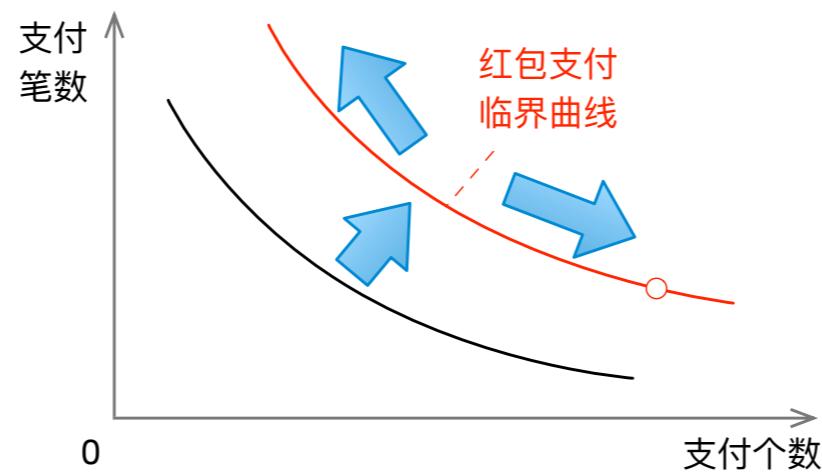


## 内部资源配置比

- 关键业务固定负载
- 服务器
- 数据库连接
- 数据库实例
- 分布式缓存
- 下游系统

## 业务自身消耗

- 临界资源固定分配
- 关键业务：组合
- 非关键业务：资源沙盘



## 数据模拟

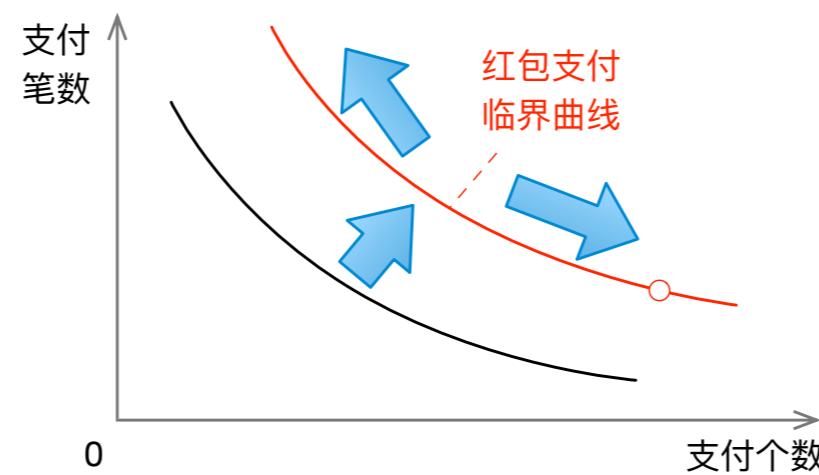
- 活动类型分布
- 热点发放活动分布
- 红包个数分布
- 支付规则分布
- DB存量数据
- 背景噪声
- ...

## 内部资源配置

- 关键业务固定负载
- 服务器
- 数据库连接
- 数据库实例
- 分布式缓存
- 下游系统

## 业务自身消耗

- 临界资源固定分配
- 关键业务：组合
- 非关键业务：资源沙盘



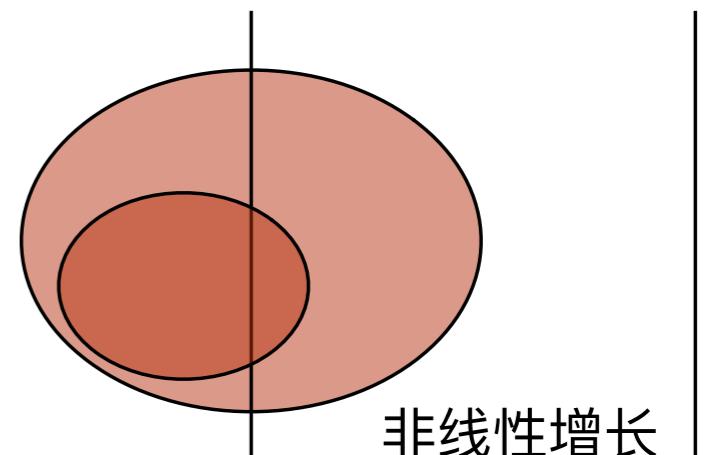
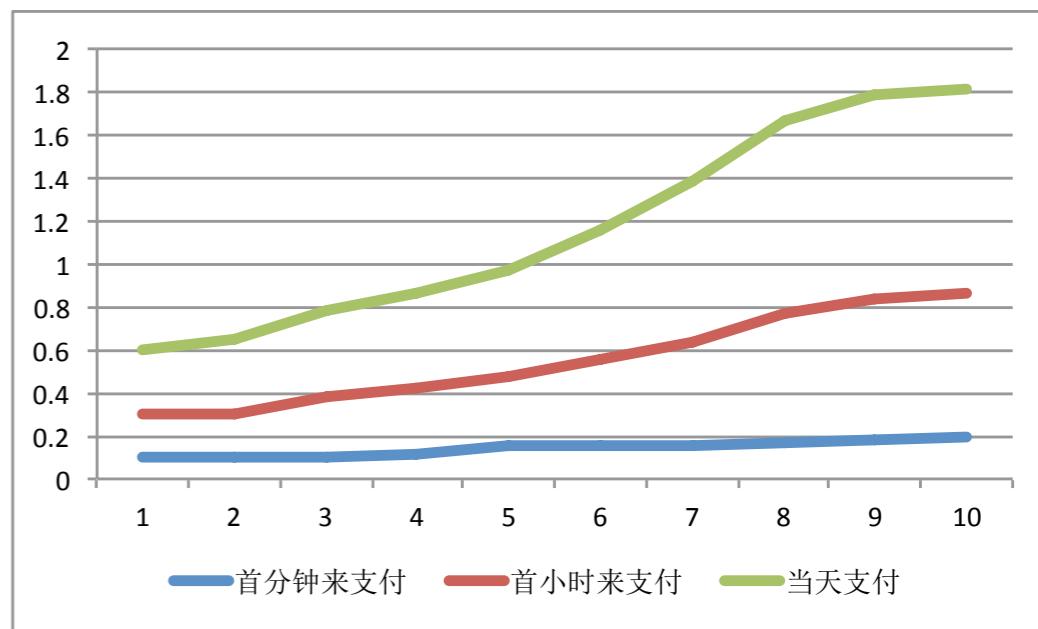
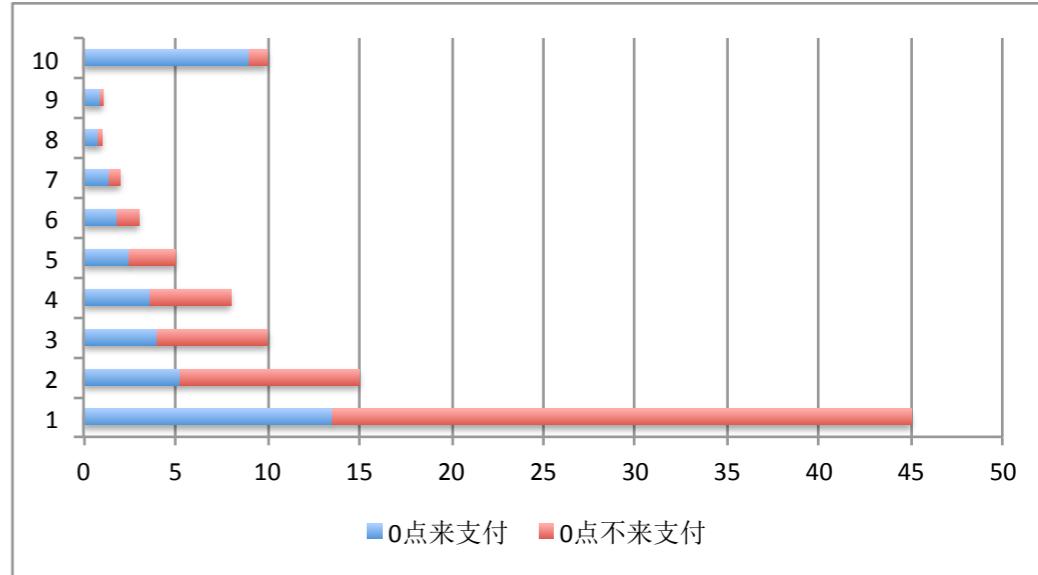
## 数据模拟

- 活动类型分布
- 热点发放活动分布
- 红包个数分布
- 支付规则分布
- DB存量数据
- 背景噪声
- ...

# 大纲

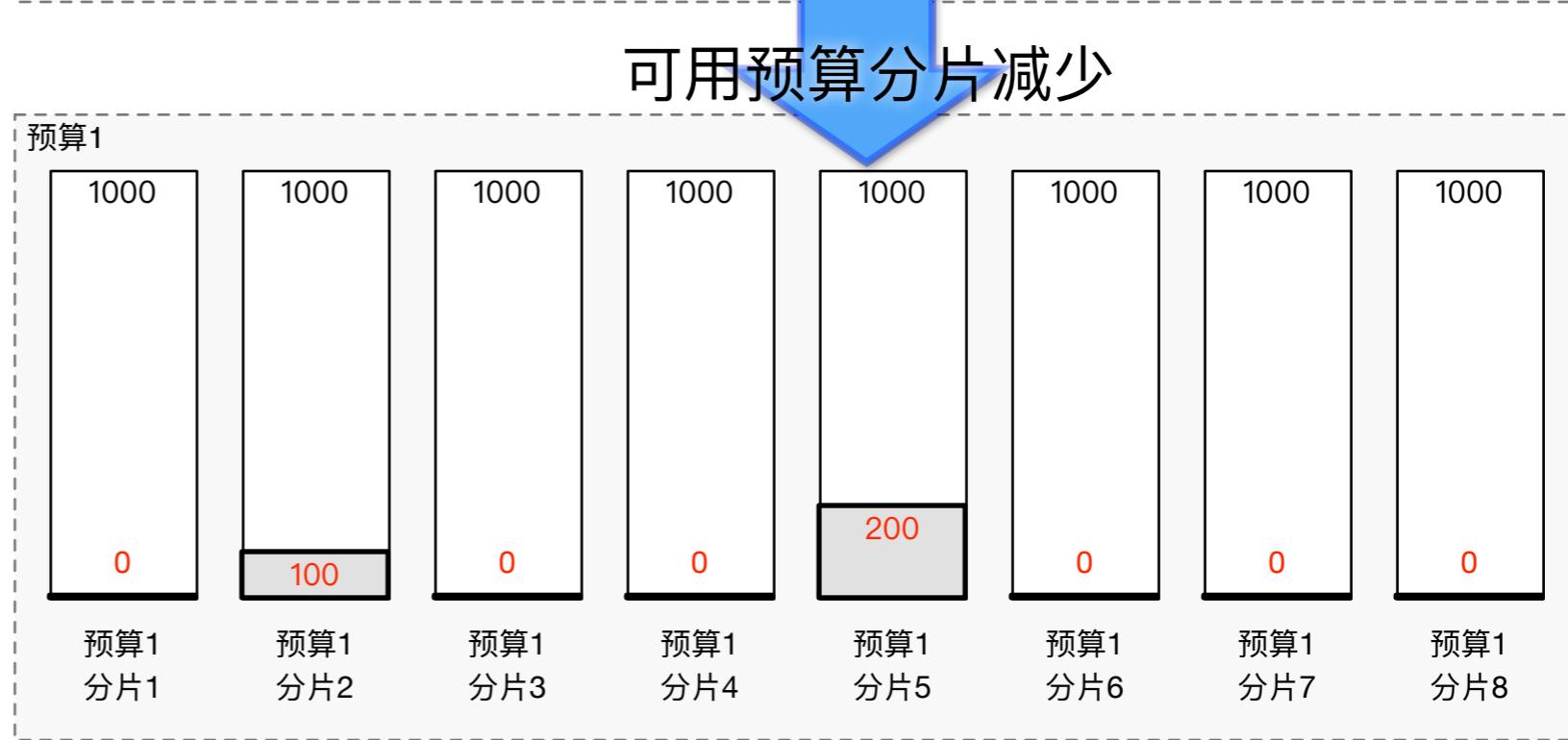
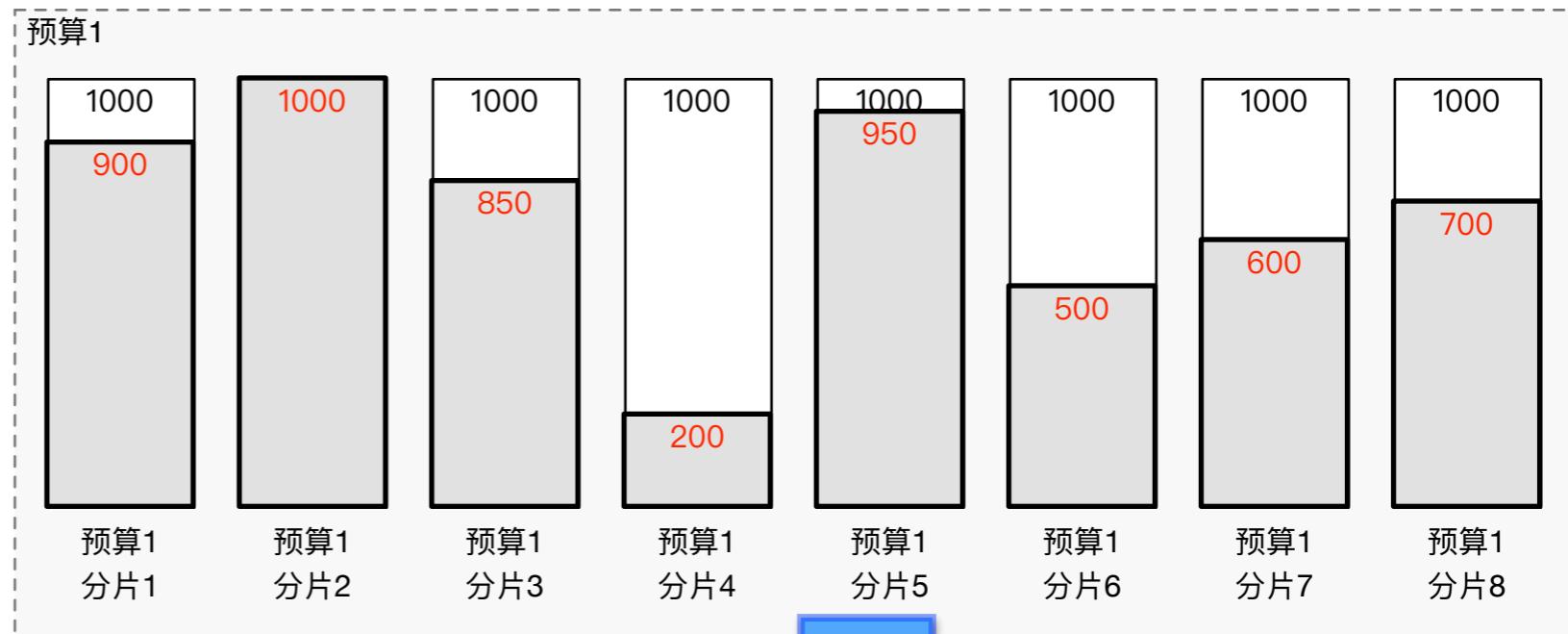
- 链路梳理
- 性能优化
- 容量评估
- 业务量评估
- 系统保护

- 发放
  - 活动收集
  - 各类活动经验曲线
- 支付
  - 笔数、个数预估
  - 往年曲线参照
  - 每日跟踪
  - 红包支付占总支付比
- 查询
  - 关键场景线性预估



# 过载保护——红包发放

- 接口限流
  - 最简单的实现
  - 单个接口单机最大tps
- 预算限流
  - 单个预算单机最大tps
  - 缓存单位周期的每个预算hit-count
- 预算分片限流
  - 单个分片级的hit-count限流



# 过载保护——支付

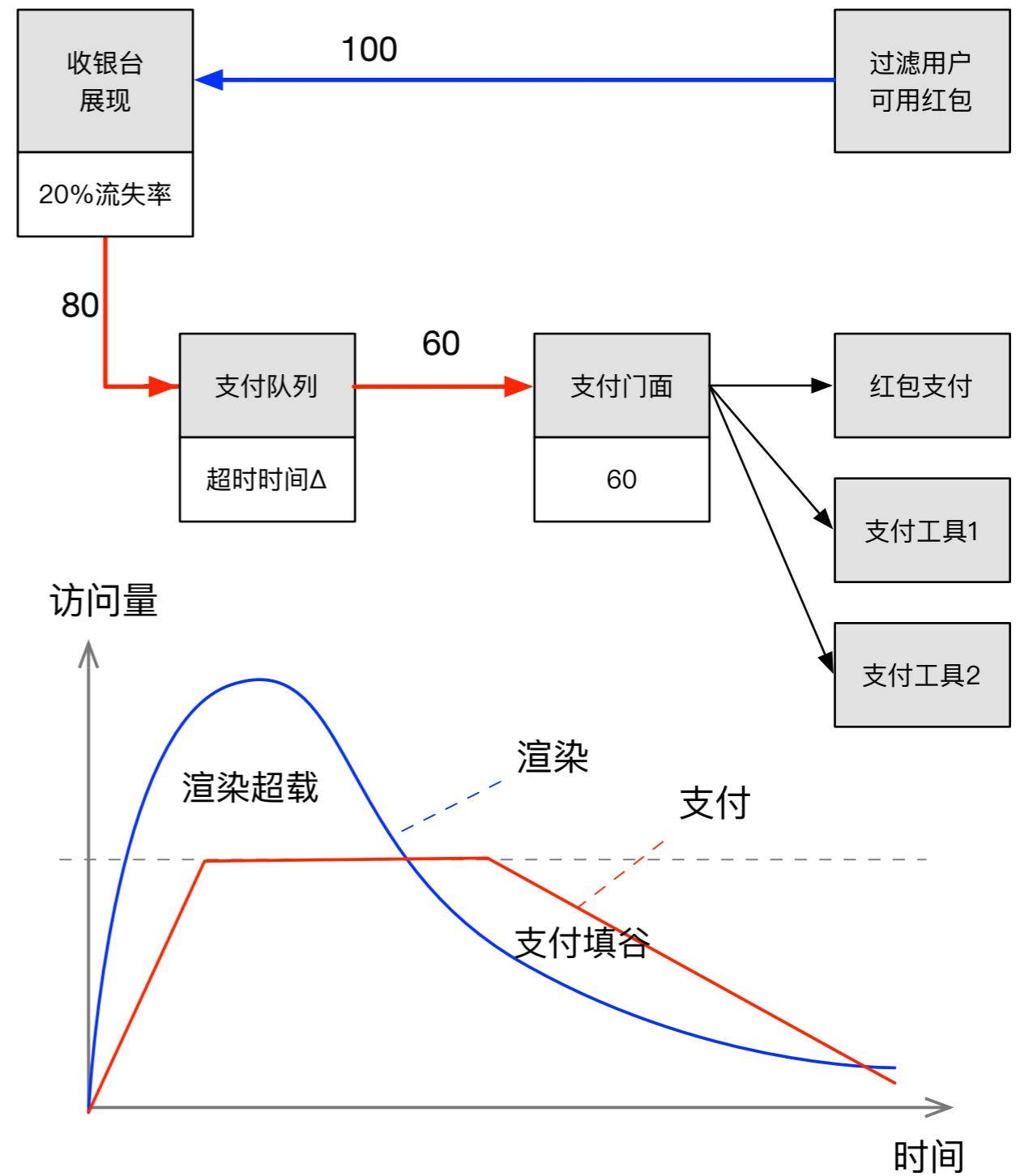
- 接口限流

拦截量过大

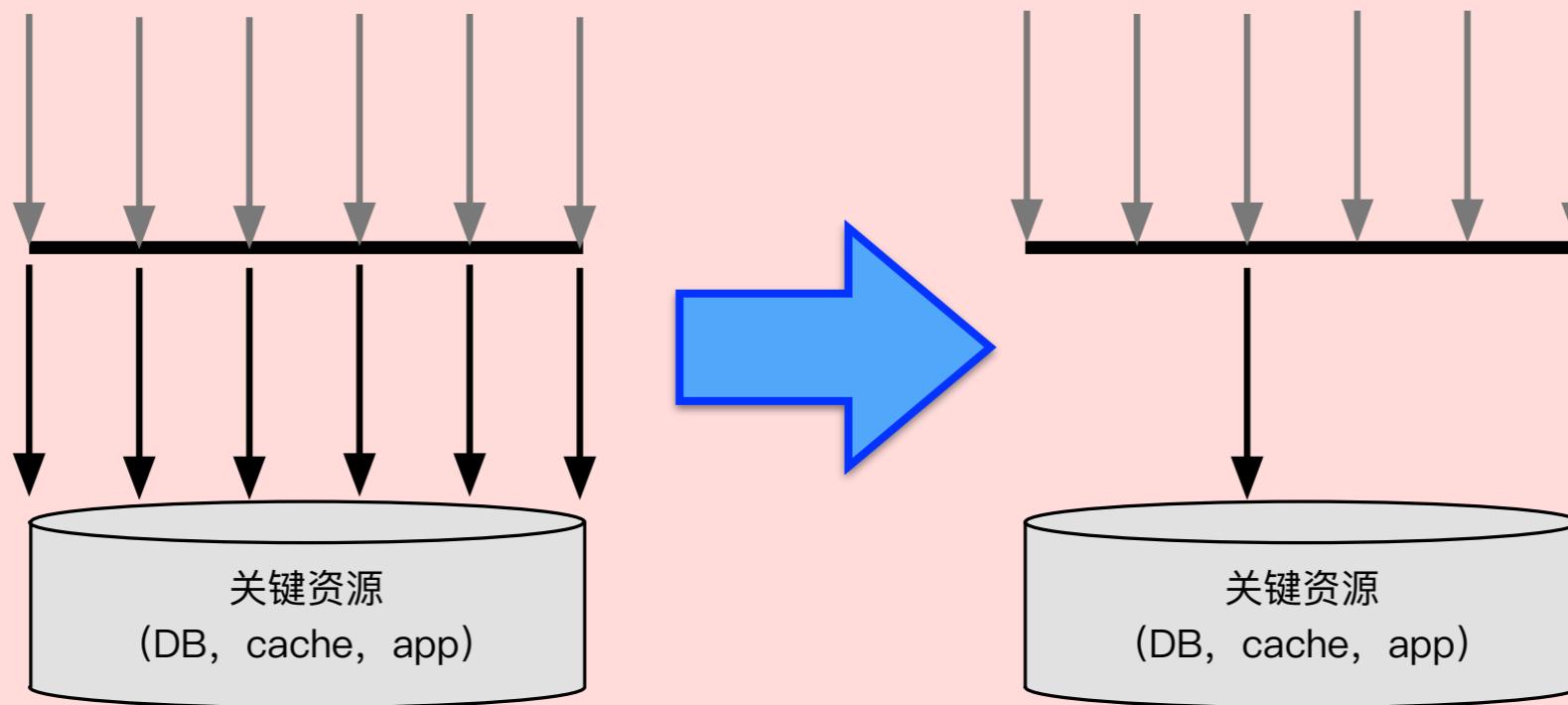
- 队列阻塞：利用支付等待间隙

木桶效应

- 队列分离：根据红包能力专用队列



# 过载保护——突发热点



- 缓存未预热状态的峰值



# 降级非关键业务

---

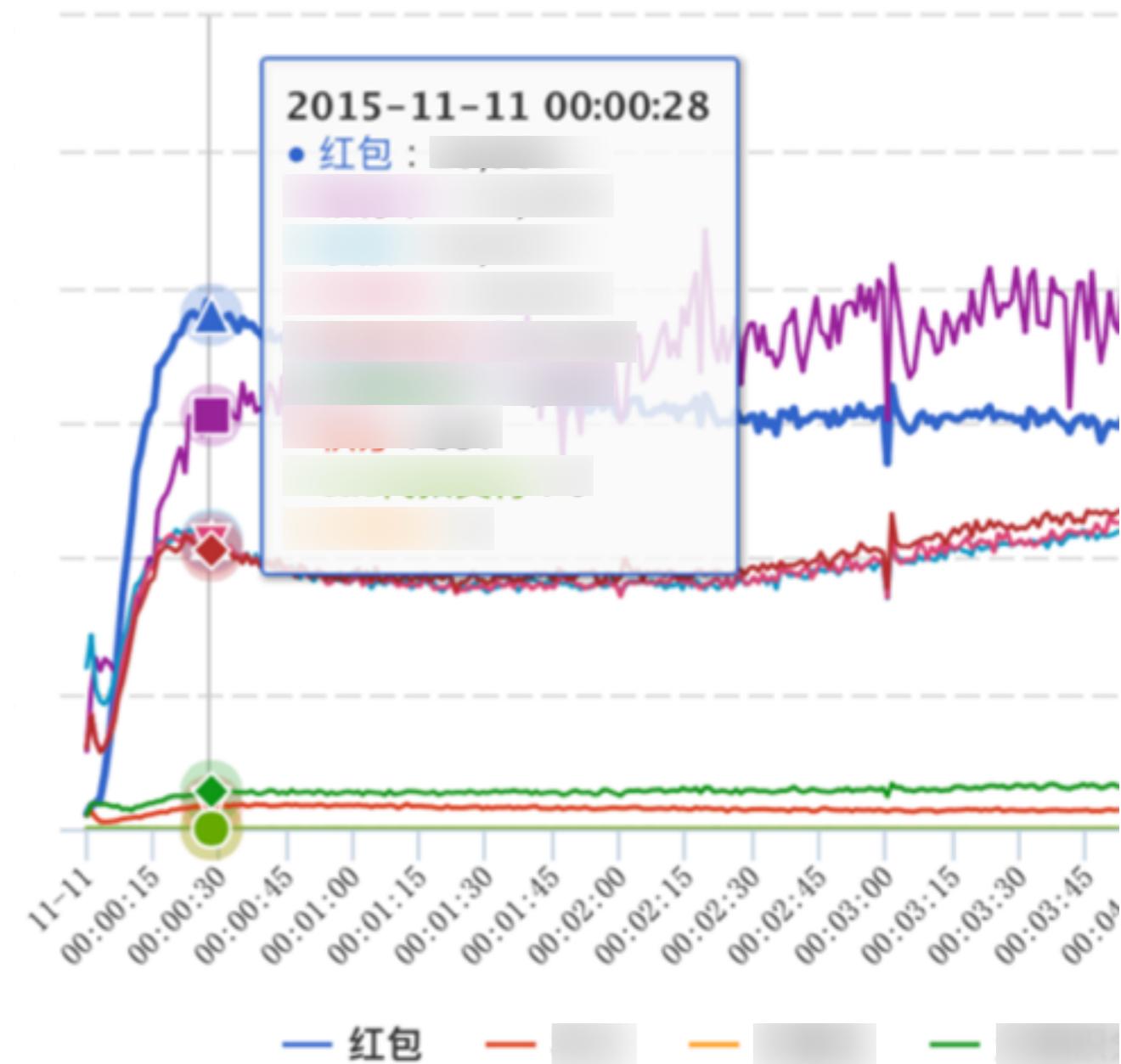


- 非关键业务资源消耗量
- 峰值时刻资源配给
- 非关键业务的组合限流



- 链路梳理
  - 上下游链路；红包内部链路
- 性能优化
  - 适配机房拆分；发放预算问题
  - 多级别缓存；支付规则；支付语句合并
- 容量评估
  - 线上；分层配比；业务资源消耗；数据模拟
- 业务量预估
  - 发放；支付(笔数个数预估)；查询
- 过载保护
  - 发放；支付；非关键业务；突发热点

# 迎接双11零点



# 下一步

---

- **性能** 服务器/DB资源不变 容量提升一个量级
- **容量预估** 根据历史数据统计特征 估算大促峰值时 各种组合的概率
- **伸缩性** 应用层实现**非迁移**的DB伸缩容
- **故障定位** 资源消耗的跟踪 链路数据流跟踪



# Thanks!

蚂蚁金服 成都研发中心

