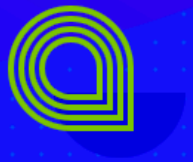
饿了么+程炎岭

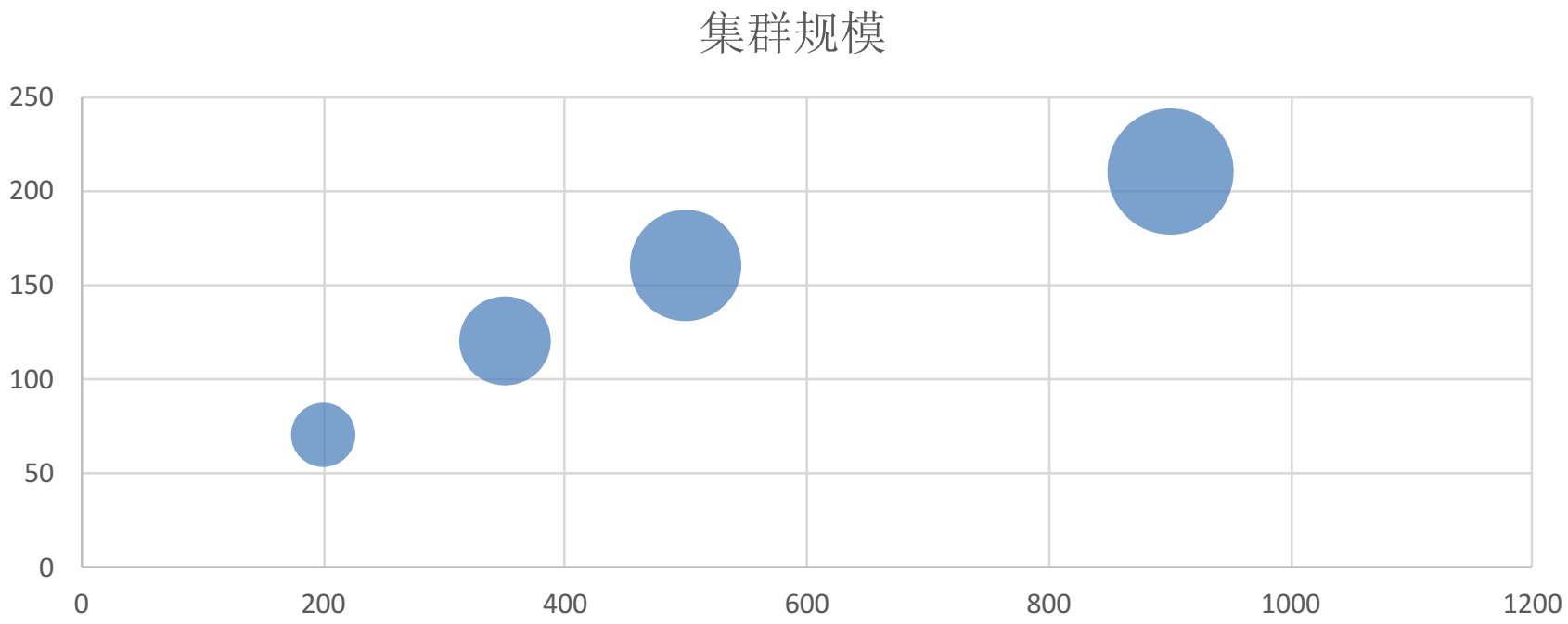
饿了么Redis Cluster集群化演进



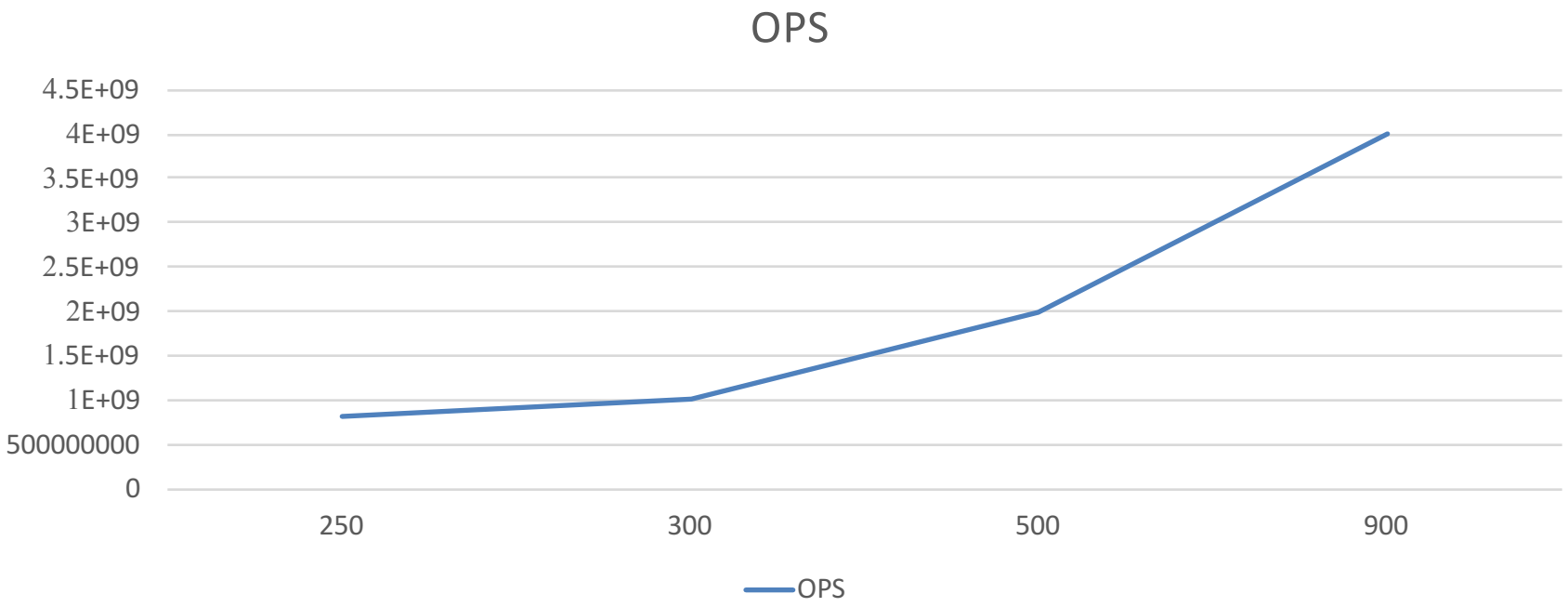
背景和数据

- 2015-2016饿了么的业务爆炸式增长
- 订单量从几十w到峰值900w/天
- 服务化治理诉求
- 应用性能和稳定诉求
- 人肉运维成本诉求
- 未来的PAAS平台池化准备

订单量和集群规模



高峰期请求


OPS
4.5E+09
4E+09
3.5E+09
3E+09
2.5E+09
2E+09
1.5E+09
1E+09
500000000
0
250
300
500
900
OPS

使用场景

- 用户端
- 商户端
- 物流配送
- 搜索排序，热卖，画像等
- 各类cache，计数器，分布式锁等

问题很多

- 单点
- Db/cache混用
- 共用一个redis，大key阻塞整个实例
- 经常需要扩容，扩容很痛苦
- 使用复杂
- 基础环境非标，存在各种配置
- 监控缺失
- 不能统一配置

治理
2016-05

自研选型

- Redis vs redis Cluster
- Twenproxy vs Codis vs Redis Cluster
- Redis Cluster + proxy

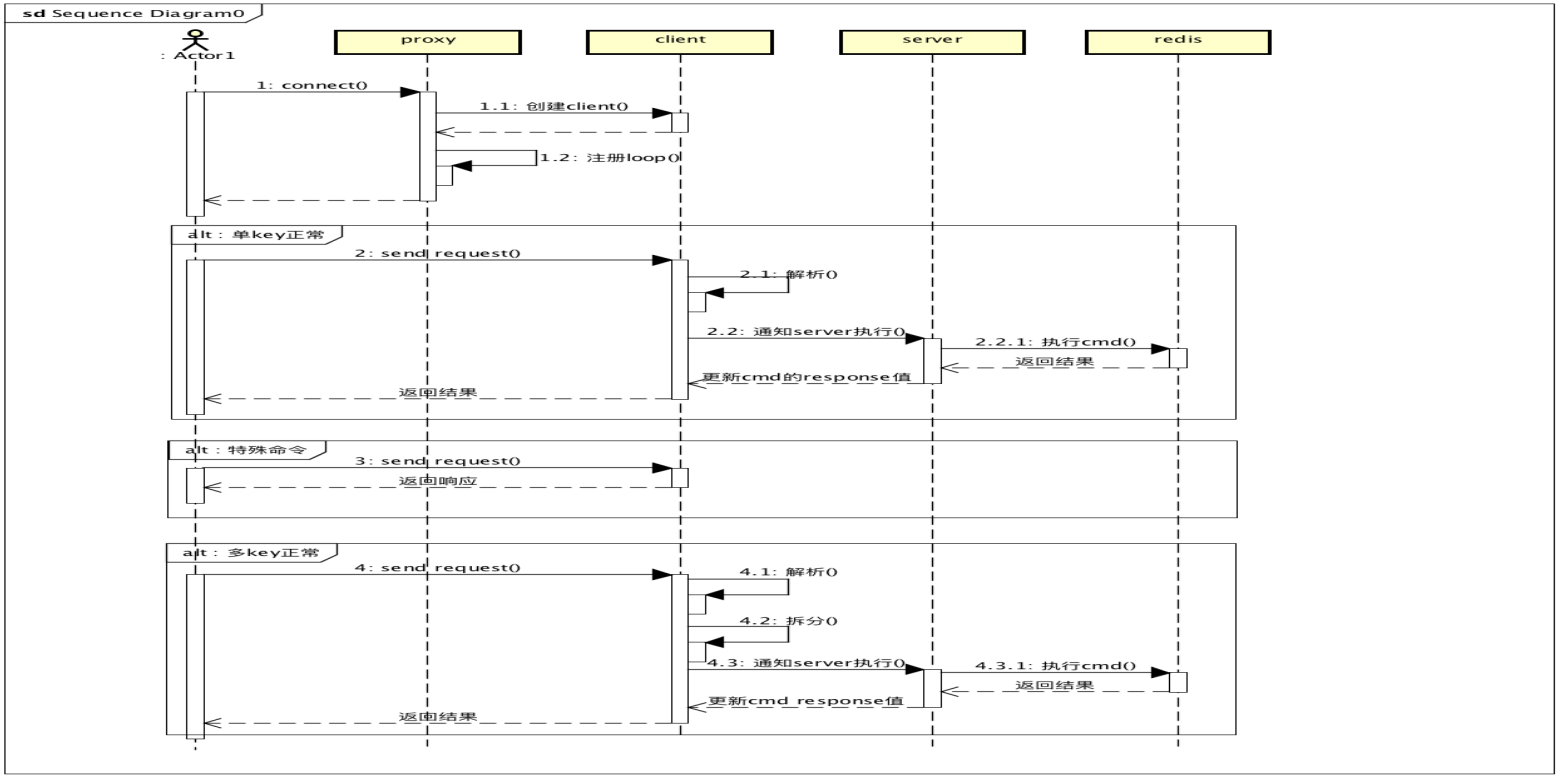
Redis Cluster 的优点

- 自带迁移功能
- Fast，高可用
- 支持在线分片
- 丰富集群管理命令

Redis Cluster 的缺点

- Client 实现复杂，需要缓存slot mapping关系并及时更新
- Client不成熟导致提高开发难度
- 存储和分布式逻辑耦合，节点太多时节点之间的检测占大量网卡带宽
- 3.0.6版本前，只能单个key迁移，并且同时只允许一个slot处于迁移状态

Corvus 时序设计


sd Sequence Diagram0
: Actor1
proxy
client
server
redis
1: connect()
1.1: 创建client()
1.2: 注册loop()
alt : 单key正常
2: send request()
2.1: 解析()
2.2: 通知server执行()
2.2.1: 执行cmd()
返回结果
更新cmd的response值
返回结果
alt : 特殊命令
3: send request()
返回响应
alt : 多key正常
4: send request()
4.1: 解析()
4.2: 拆分()
4.3: 通知server执行()
4.3.1: 执行cmd()
返回结果
更新cmd response值
返回结果
powered by Astah

Corvus

- 封装了redis cluster 协议，提供redis 协议
- 扩容缩容应用无感知
- 实时缓存了slots mapping
- Multiple Thread
- Lightweight
- Reuseport support

Corvus commands

- Pipeline support
- Modified commands
- Restricted commands
- Unsupported commands

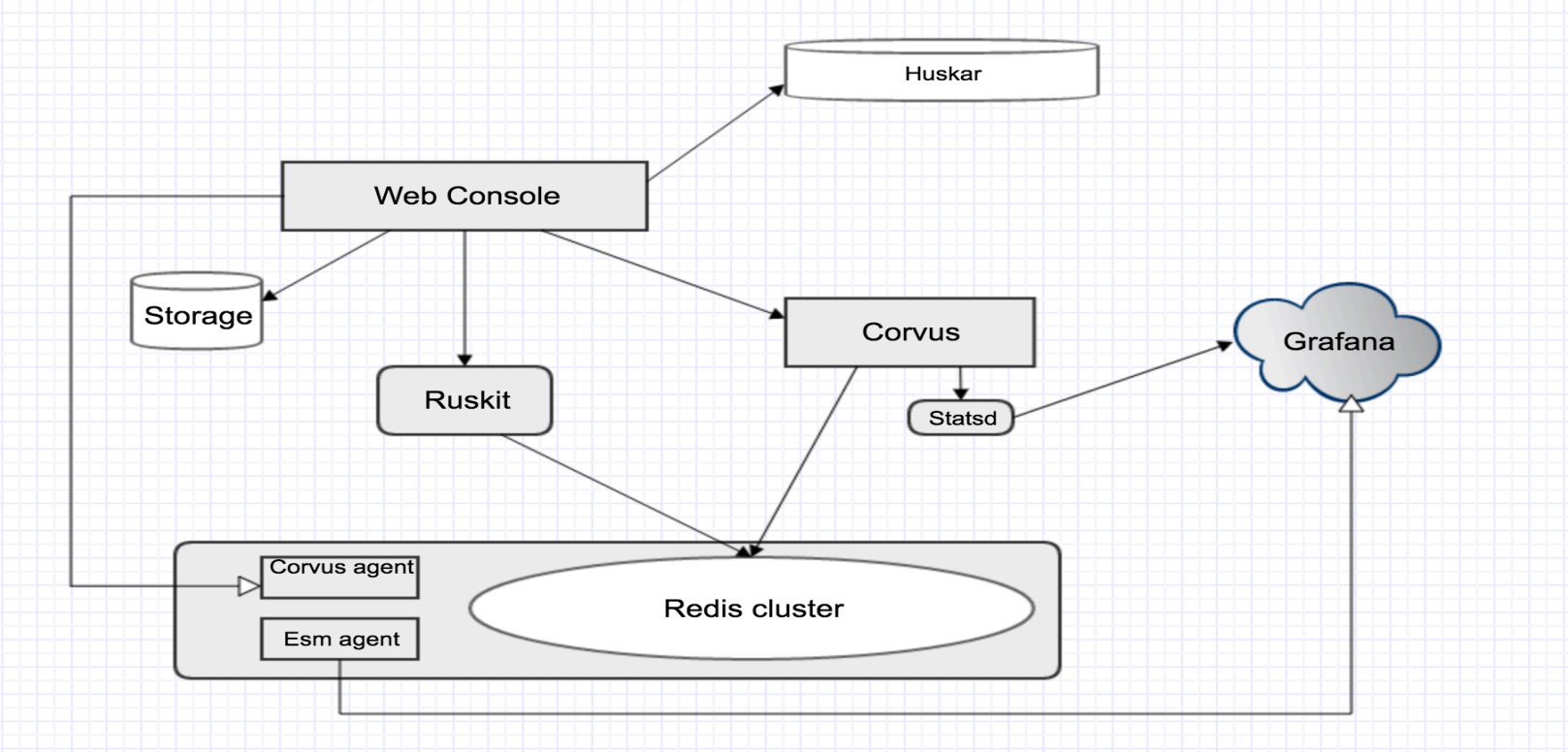
Corvus performance

server	OS	CPU	core	mem	net
Corvus(0.2.4)	Centos 7.1 3.10.0-229.el7.x86_64	Intel(R) Xeon(R) CPU E5-2620 v3	24	32G	Bond 4 1G*2
Redis(3.0.3)	Centos 7.1 3.10.0-229.el7.x86_64	Intel(R) Xeon(R) CPU E5-2620 v3	24	32G	Bond 4 1G*2

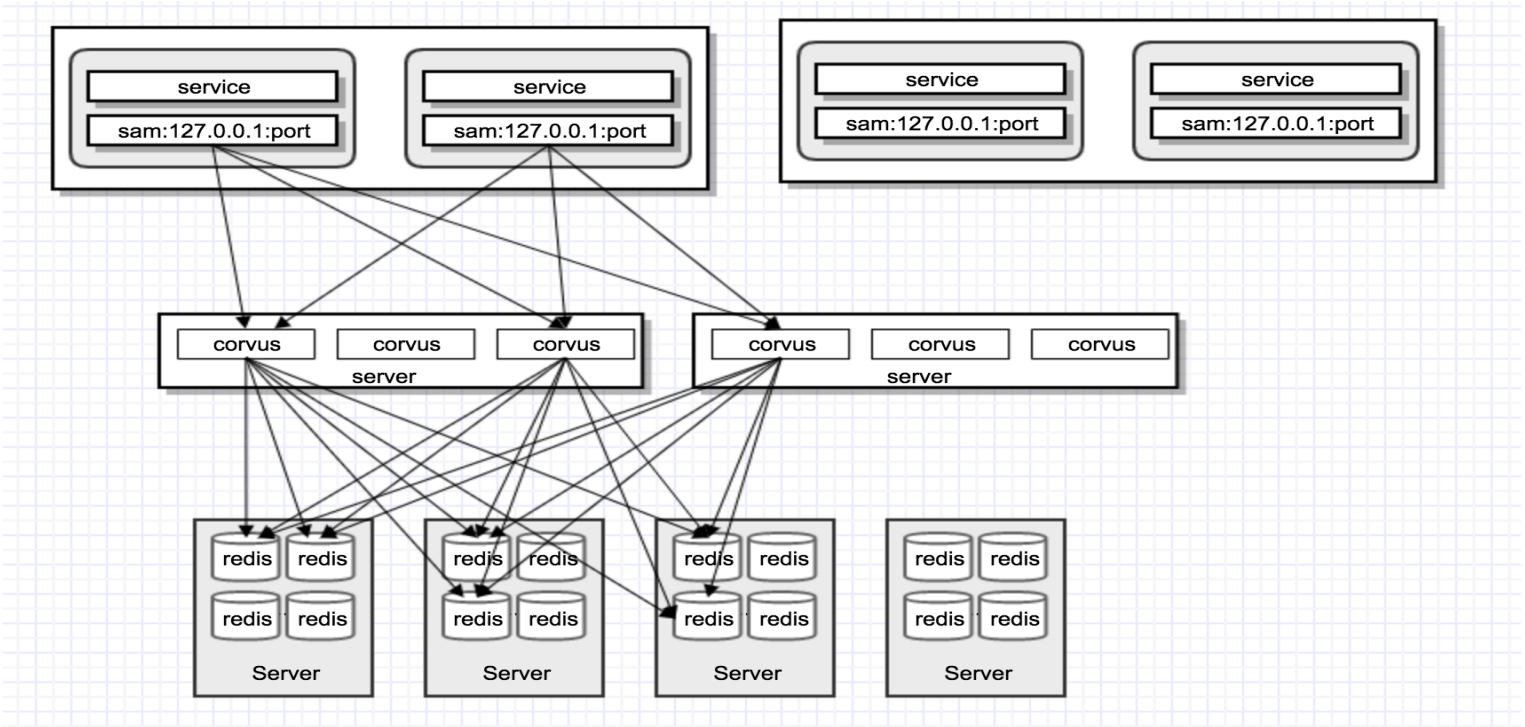
redis-benchmark -h 10.0.16.72 -p 8802 -n 90000000 -r 90000000 -t set -P 1000 -c 100
redis-benchmark -h 10.0.16.72 -p 8802 -n 90000000 -r 90000000 -t get -P 1000 -c 100

Set/get	Corvus Cpu util	Corvus cpu kernel	Corvus cpu user	Redis cpu util	Redis node cpu	Completed command
set	94.7%	19.4	4.2	33%	43%	1427k/s
get	94.9%	19.8	4.1	5%	18%	1434k/s

Corvus 逻辑架构


Huskar
Web Console
Storage
Corvus
Grafana
Ruskit
Statsd
Corvus agent
Esm agent
Redis cluster

Corvus 物理部署


service
sam:127.0.0.1:port
service
sam:127.0.0.1:port
service
sam:127.0.0.1:port
service
sam:127.0.0.1:port
corvus
corvus
corvus
server
corvus
corvus
corvus
server
redis
redis
redis
redis
Server
redis
redis
redis
redis
Server
redis
redis
redis
redis
Server
redis
redis
redis
redis
Server

Redis 运维

- 移交数据库团队运维
- 统一redis 物理部署架构
- 标准化redis/corvus 服务器以及单机容量
- 设定redis 开发设计规范
- 设定redis资源申请流程，做好容量评估
- 完善监控，跟进全链路压测
- 隔离关键集群，分专用资源和公共资源
- 升级万兆网卡

Redis 开发约定

- Key 格式约定：object-type:id:field。用":"分隔域，用"."作为单词间的连接
- 禁止大k-v
- 只用作cache, Key TTL设置
- 做好容量评估
- 设定集群上限
- 接入统一为本地goproxy

Redis 改造

- Redis 迁移cluster
- 完善system、redis、corvus监控
- 拆分、扩容
- 非标机器腾挪下线
- 大key、热节点、大节点扫描
- 网卡Bond，多队列绑定
- 修改jedis源码，加入matrix埋点上报etrace

经验总结

- Redis 更适合cache，cluster下更不合适做持久化存储
- 一致性保证，不做读写分离(尽管corvus支持)
- 单实例不宜过大，节点槽位要均衡
- 发现问题，复盘，积累经验
- 不跨机房部署，性能优先
- 异地多活通过订阅消息更新缓存
- 大部分的故障都是由于hot-key/big-key
- 关心流量，必要时立刻升级万兆

遗留问题

- 还不够自动化
- 版本升级问题
- 大集群Reshard扩容慢
- Sam->corvus->redis链路分析
- 多活跨机房写缓存同步
- AOF/RDB
- 冷数据
- 集群自愈

池化

- 公共池，专用池，资源buffer
- 资源申请只需关注容量以及使用方式，无需关心机器
- 新申请以及扩容缩容按组(group)操作
- 运维操作自动化

开发自助

- 开发自助申请redis 资源
- 填入demo key/value，自动计算容量
- 状态，告警推送
- 各类文档齐全

THANKS