

打造高性能高可用搜索架构

爱奇艺搜索架构实践

陈爱云



促进软件开发领域知识与创新的传播



关注InfoQ官方信息
及时获取QCon软件开发者
大会演讲视频信息



扫码，获取限时优惠

ArchSummit
全球架构师峰会 2017 [深圳站]

2017年7月7-8日 深圳·华侨城洲际酒店

咨询热线：010-89880682

QCon

全球软件开发大会 [上海站]

2017年10月19-21日

咨询热线：010-64738142

个人介绍

- 2010年毕业于天津大学
- 2010加入PPTV，负责P2P下载
- 2013年加入爱奇艺，负责爱奇艺搜索架构，致力于提高搜索架构的性能和可用性

目录

- 1 爱奇艺搜索架构的演进
- 2 如何打造高可用搜索架构
- 3 挑战及未来计划

1

爱奇艺搜索架构的演进

爱奇艺搜索概况

- 始于2011年
- 服务爱奇艺6个业务，10个端
- 亿级别的搜索量和索引量
- 平均一个搜索请求有12次rpc调用，最多有44次rpc调用
- 平均搜索时长20ms

视频搜索难点和痛点

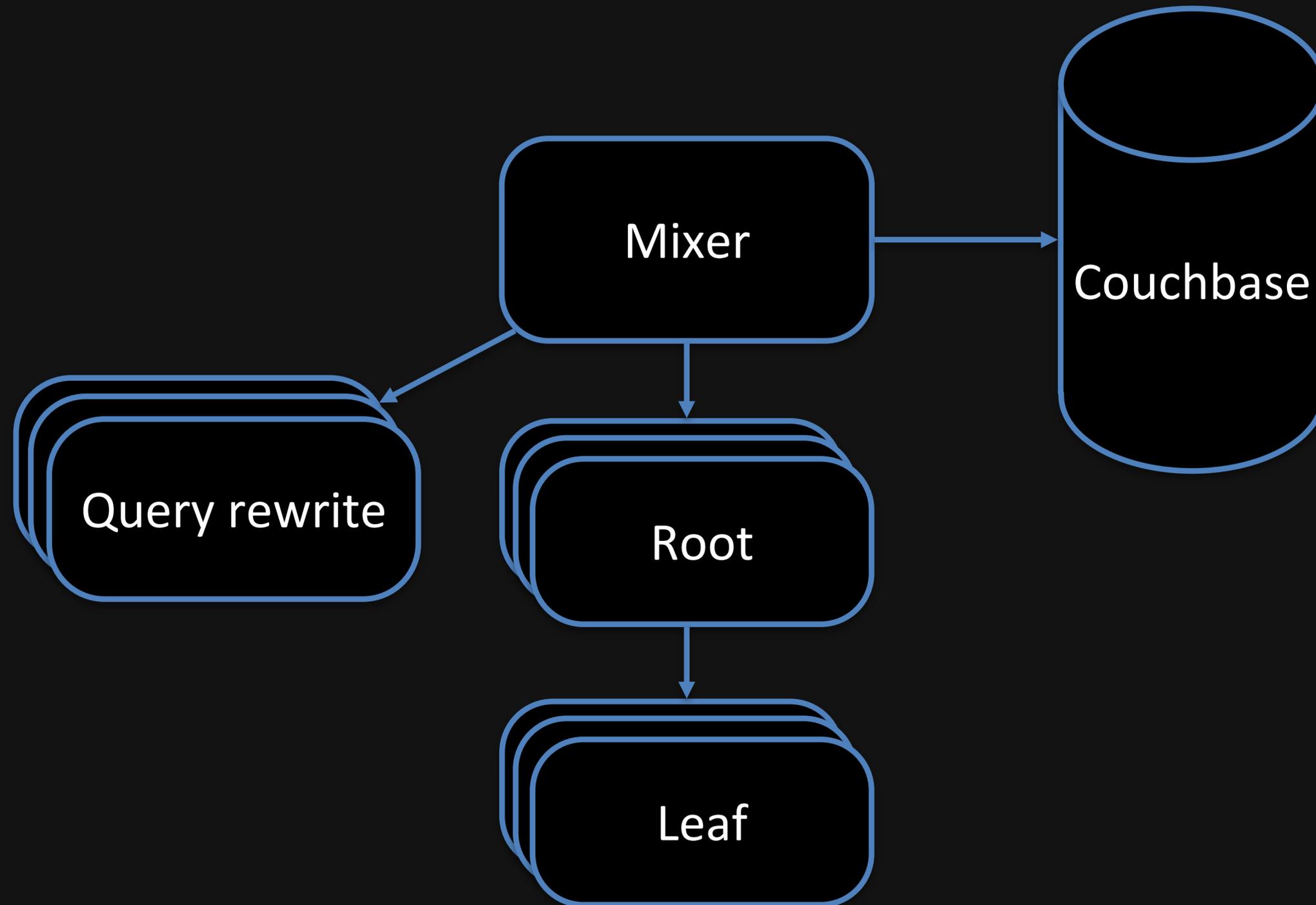
索引量大

低延迟

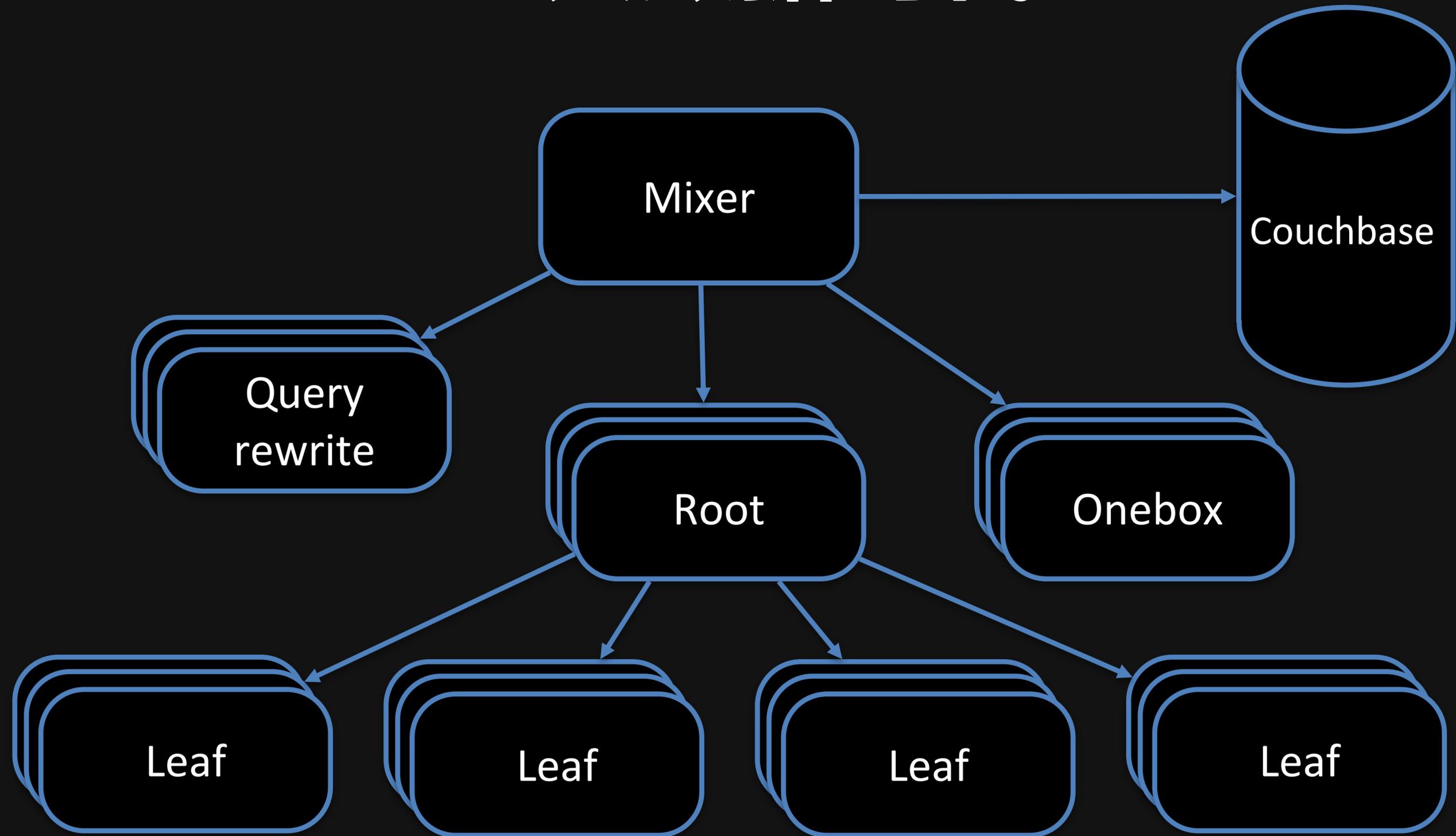
高并发

索引更新速度快

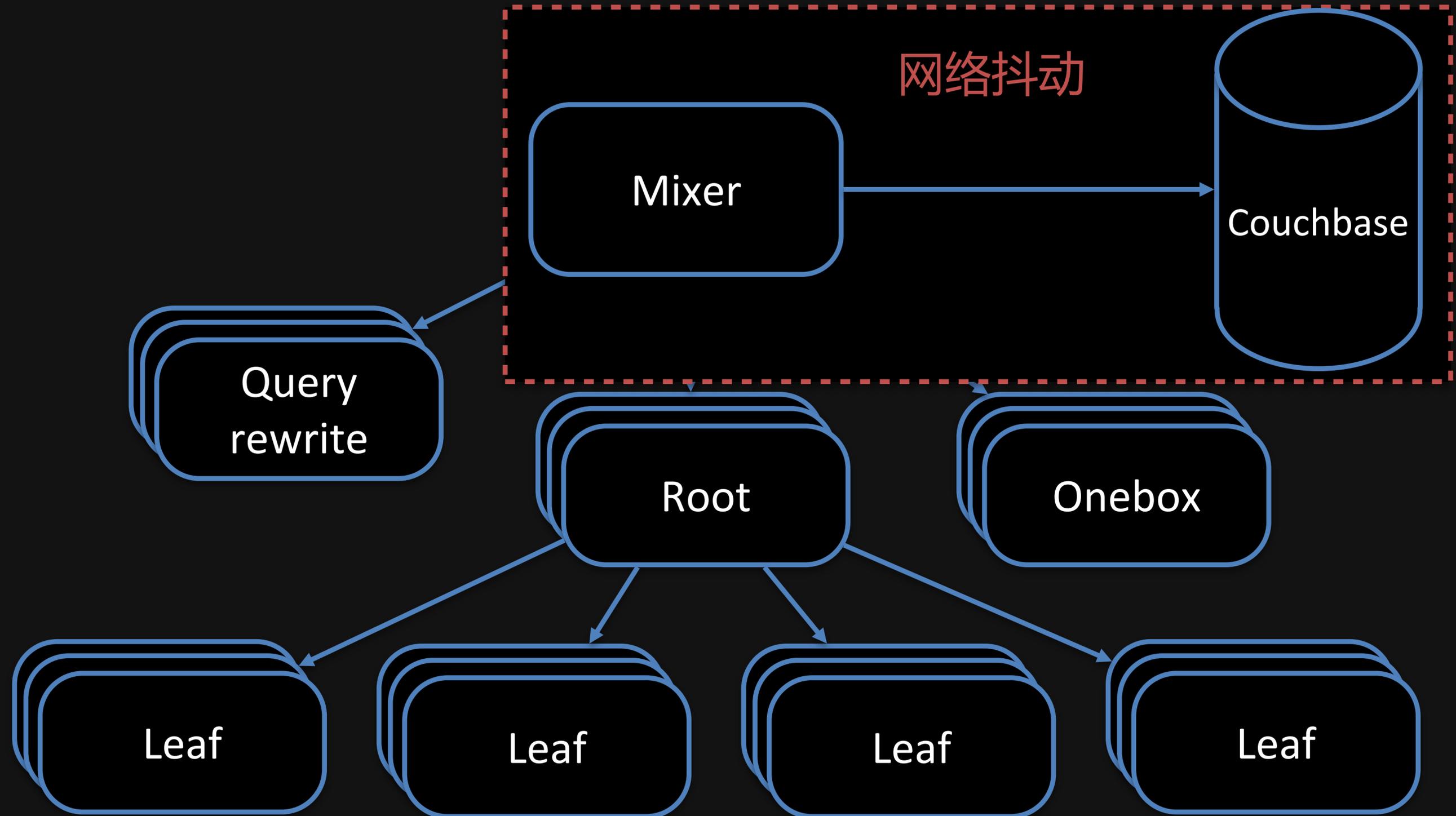
1.0 初期时代



2.0 大数据时代

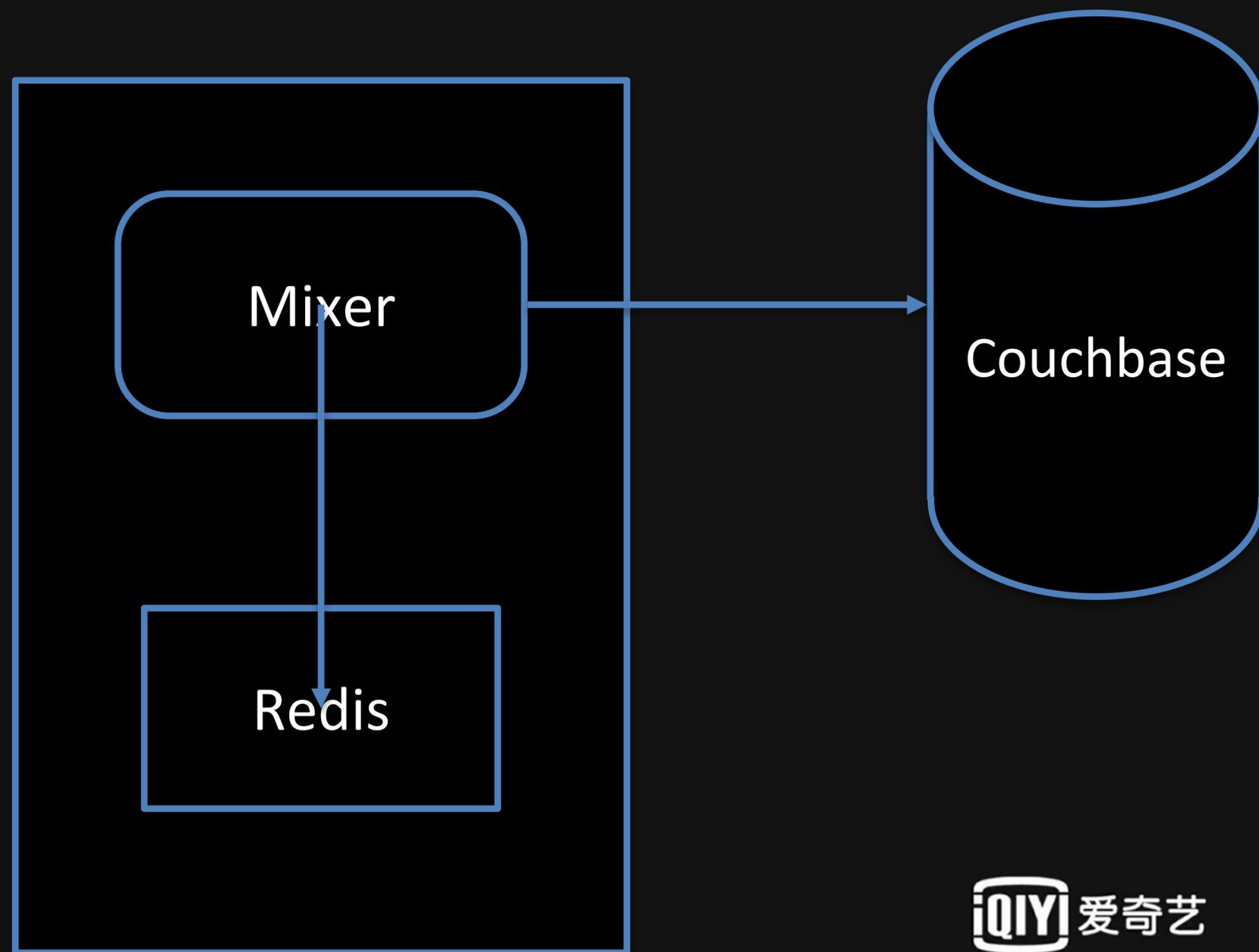


2.0 大数据时代



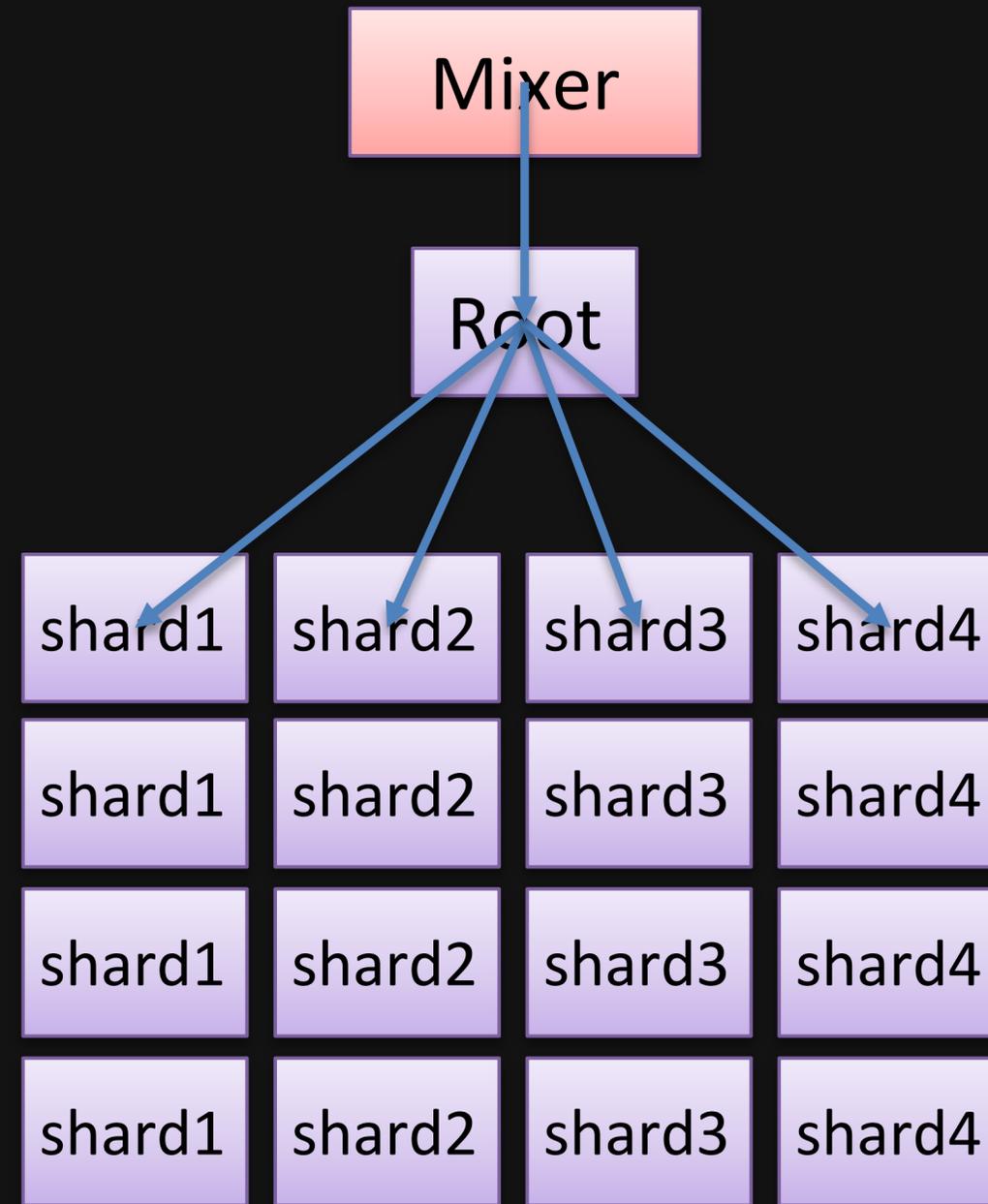
2.0 大数据时代——应对网络抖动

- 优先访问本机redis
- couchbase中的数据格式为：
key->value
- redis中的数据格式为：
key->value_timestamp



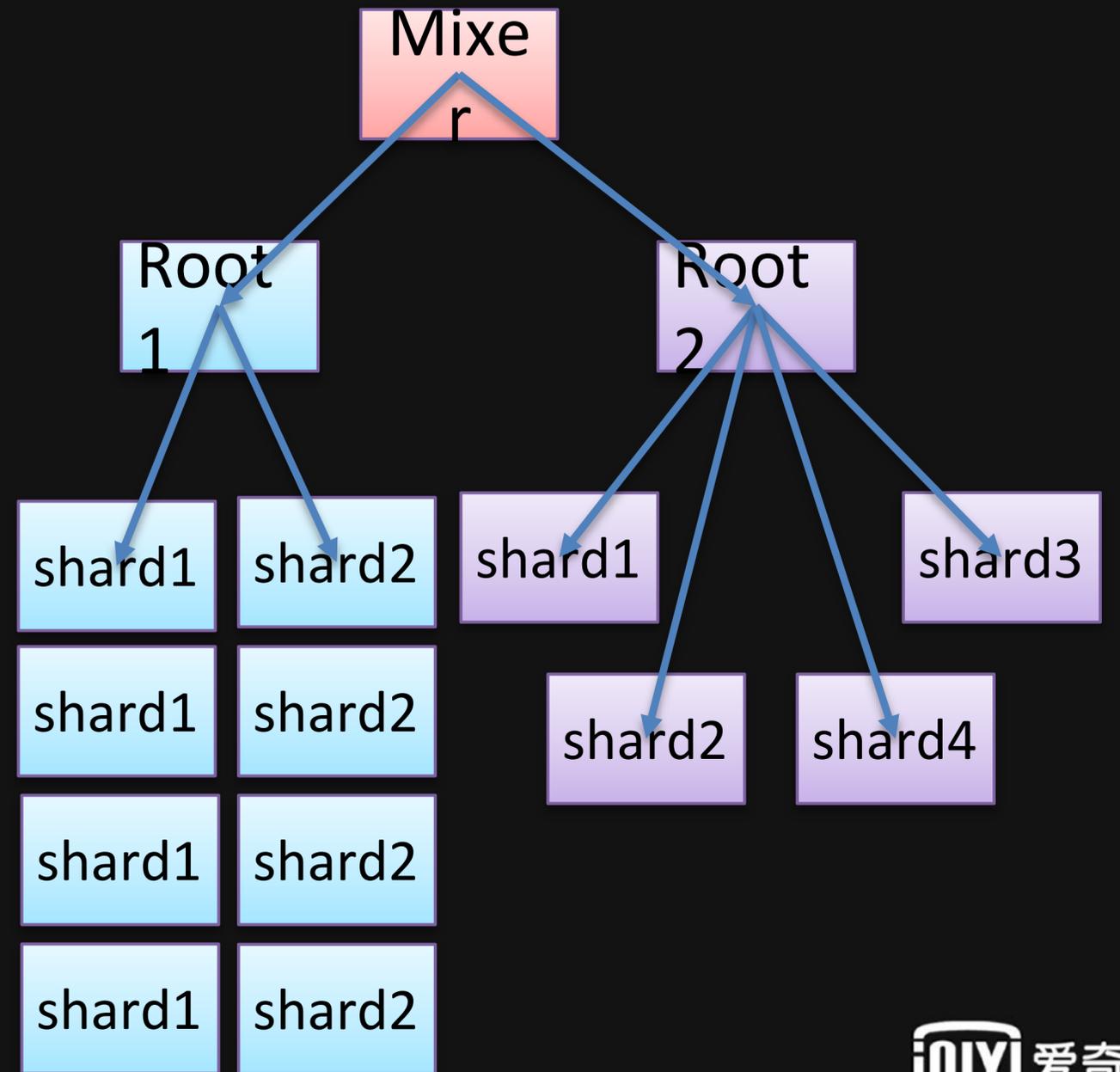
3.0 索引分级——过去

- 8亿doc，每台机器索引2亿，总共需要4列
- 4w qps，每台机器可以处理1w qps，总共需要4行

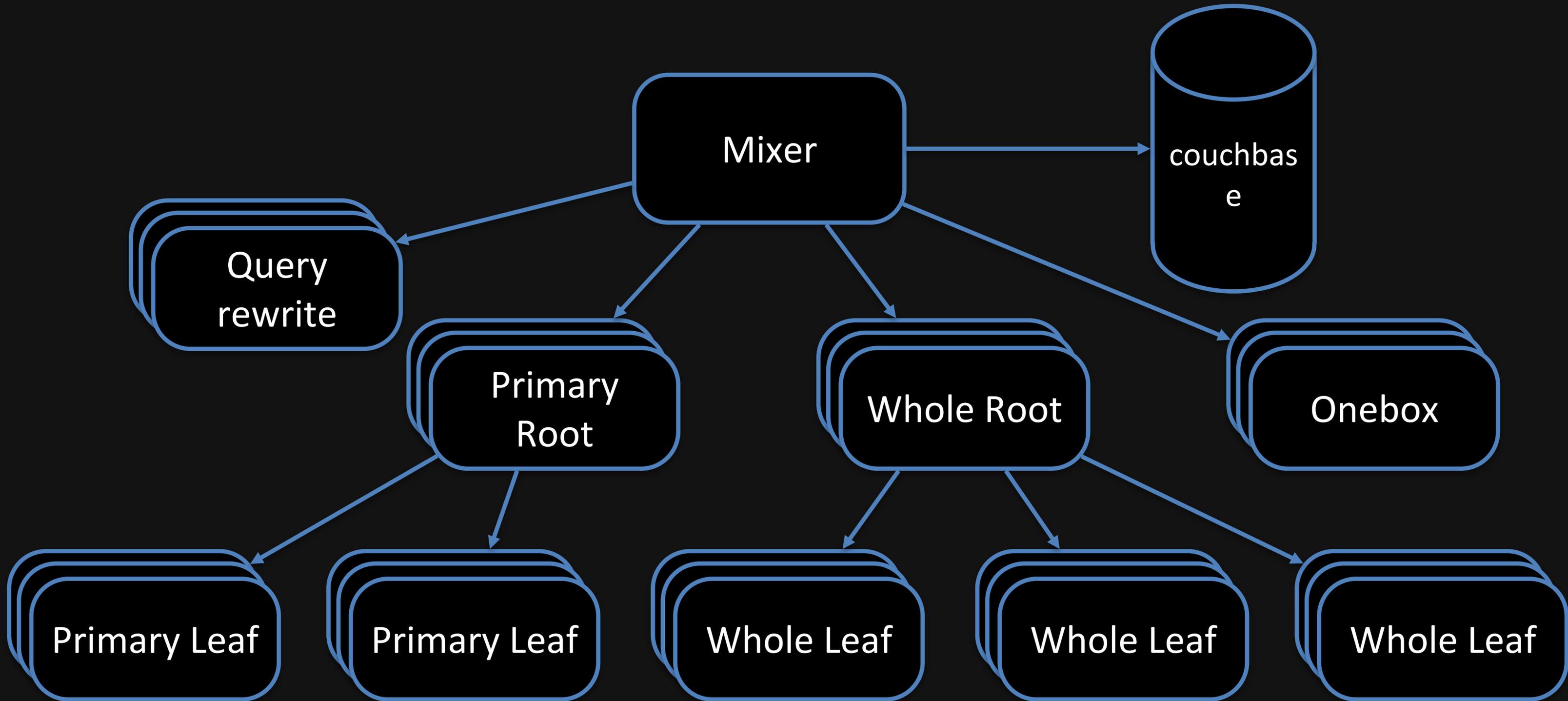


3.0 索引分级——现在

- 4亿热门doc需要2列
- 热门doc可以服务80%的请求
- 热门索引与全量索引比例为4 : 1
- 节省4台机器



3.0 索引分级——现在

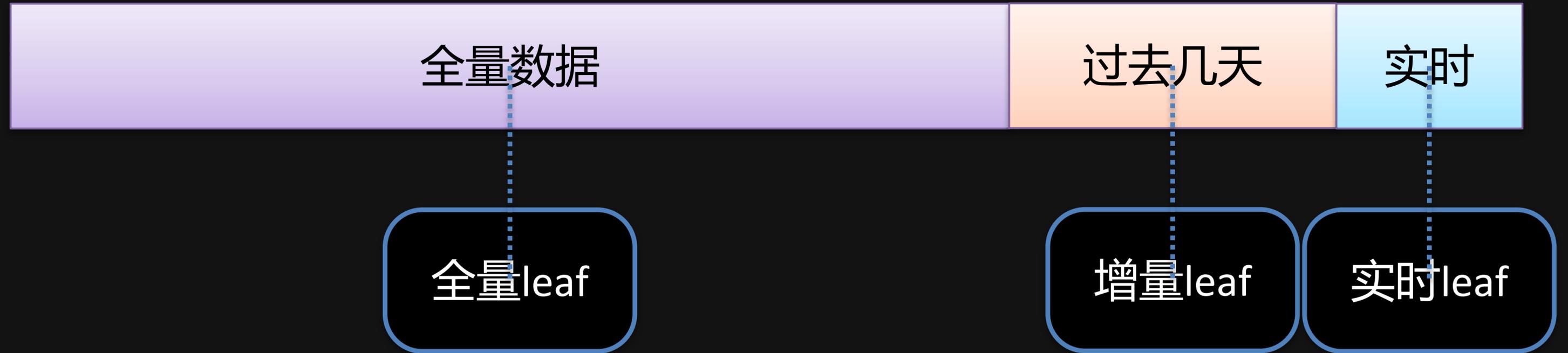


4.0 索引更新——过去



- 传输索引文件浪费带宽
- 加载索引文件浪费CPU和内存
- 对于前置任务的完成时间依赖较高

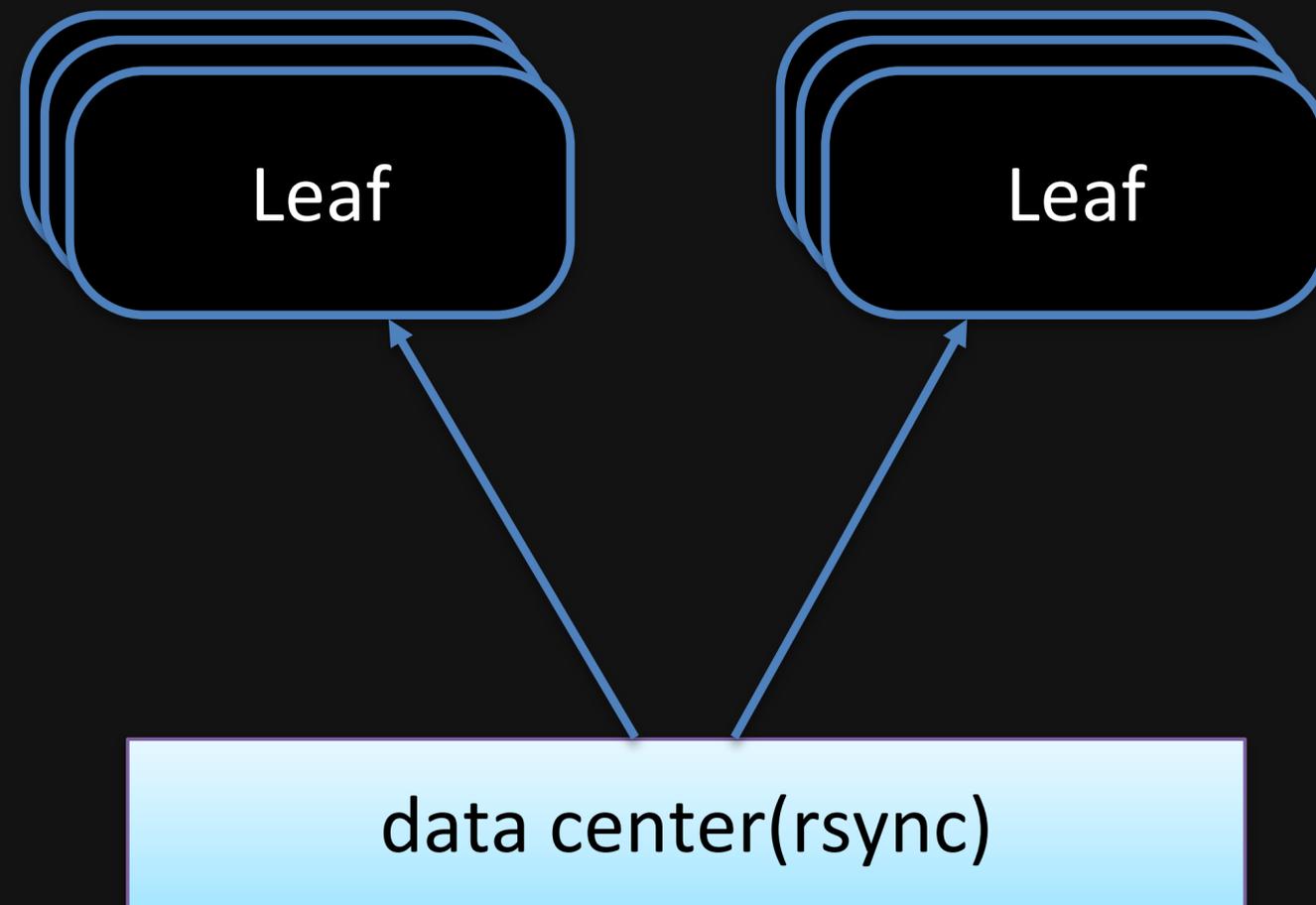
4.0 索引更新——现在



- 全量数据7天更新
- 增量leaf所需数据每天更新，数据量逐渐变大，直至下一次全量数据
- 全量数据的filter（需要过滤的doc）每天更新一次，包含全量数据生成好后有过更新的所有doc，这部分数据由增量leaf召回

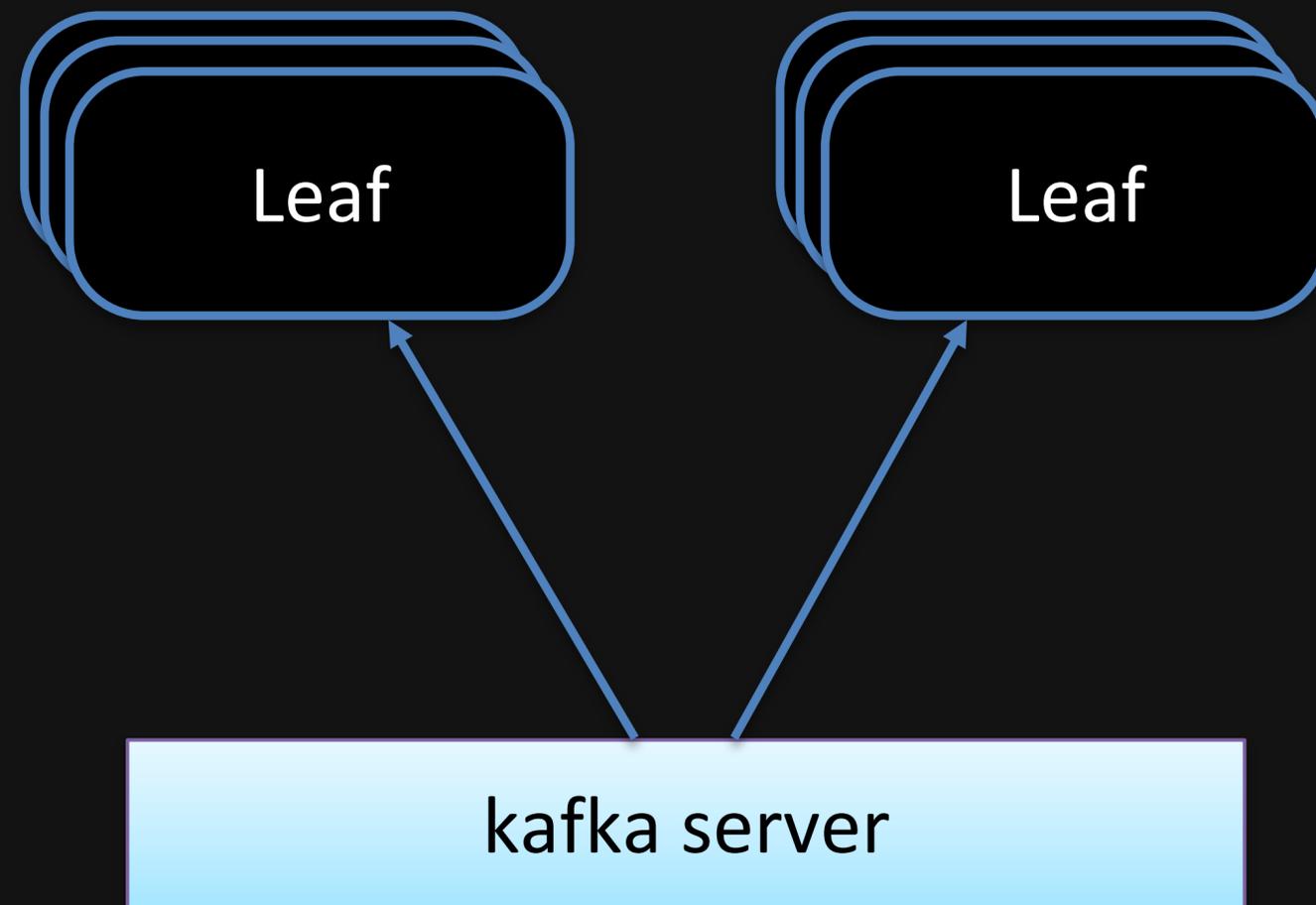
5.0 实时更新——过去

- 更新粒度大
- 更新延时大
- 追踪困难



5.0 实时更新——现在

- 支持消息级别的更新
- 更新可追踪
- 遇到的问题：不同partition的消息不能保证有序性



6.0 同步到异步

- 陈爱云定A餐厅的饭，等待□
- 陈爱云依次定A餐厅的饭，和B餐厅的汤，等待□
- 陈爱云同时定A餐厅的饭和B餐厅的汤，等待☹
- 陈爱云定A餐厅的饭和B餐厅的汤，定完继续工作，等待送餐员的电话□

6.0 同步到异步

- 团队内部开发的异步http框架libnet
 - 基于libev
- 团队内部开发的pbrpc异步框架
 - 基于libev和google protocol buffers
 - 单机qps可达67w
 - 支持服务注册和发现、负载均衡、流量控制、健康检查

性能优化

基于CPU的优化

改变数据结构
线程绑定CPU

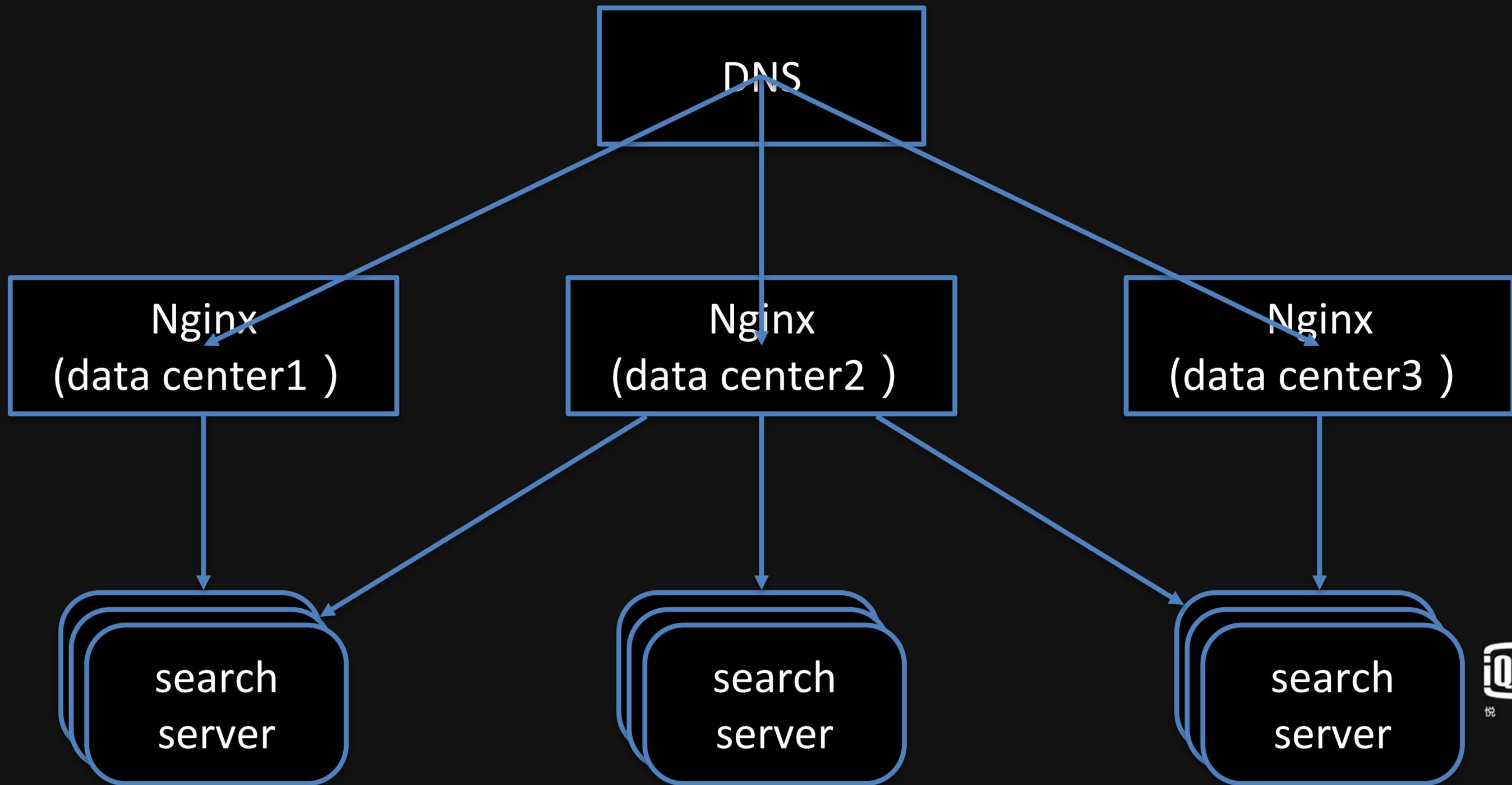
基于memory的优化

倒排压缩
正排压缩
正排数据放磁盘

2

如何打造高可用搜索架构

打造高可用的搜索架构——异地多活



打造高可用的搜索架构——降级

判断是否需要降级的标准

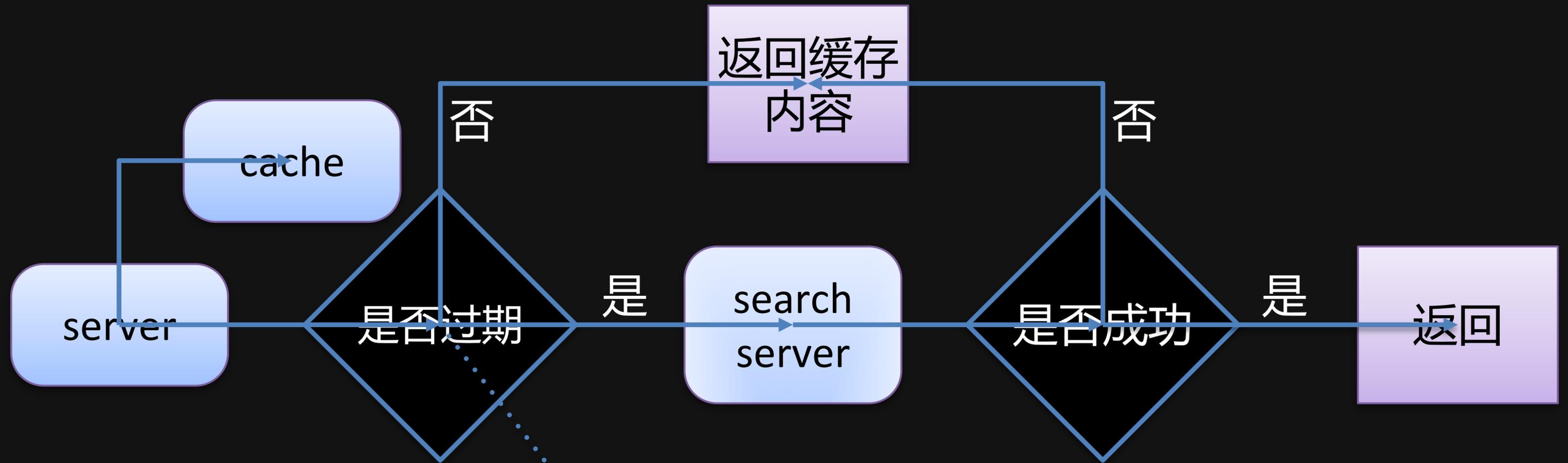
client

server的成功率
server的响应时间

server

task的平均处理时间
队列长度
本进程所消耗的CPU

降级——延长缓存过期时间



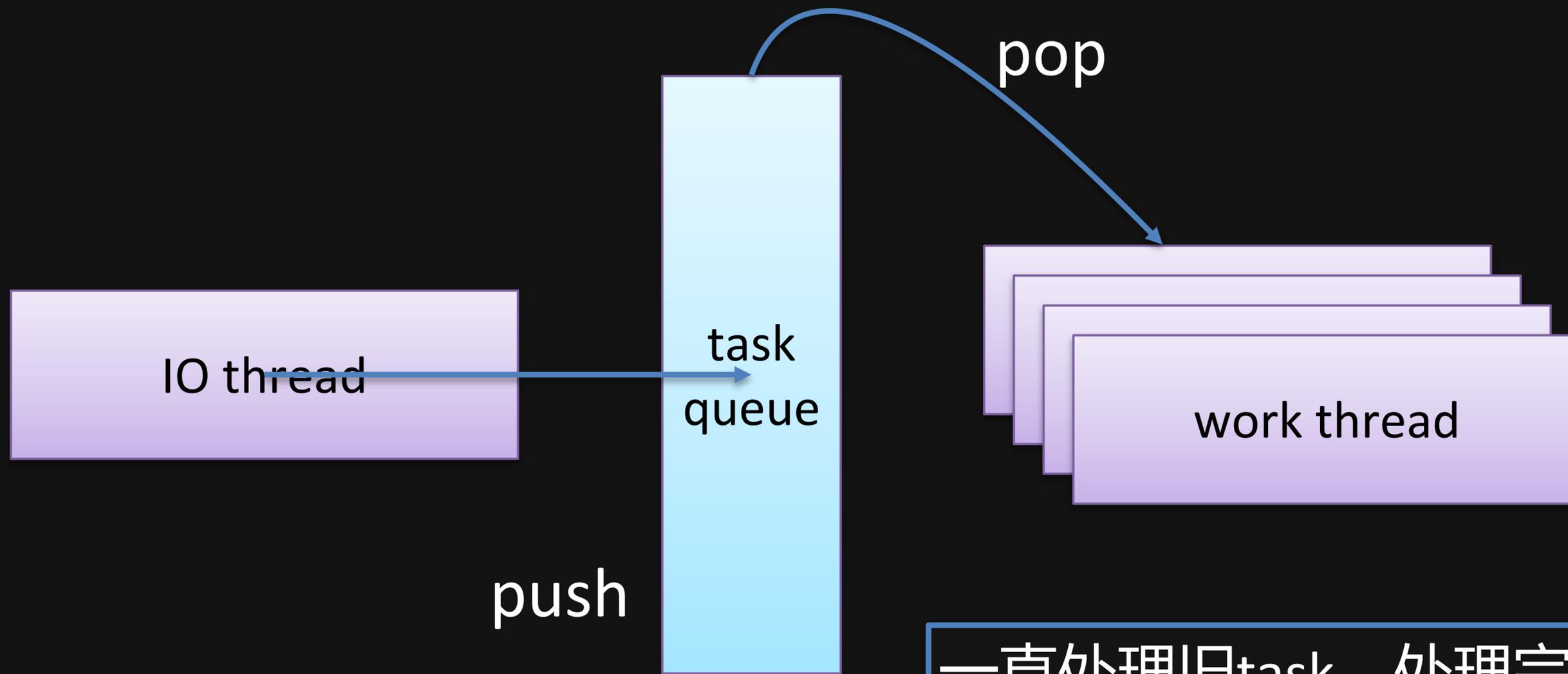
取缓存时，同时取出缓存写入时间 t_1 ； $t = \text{now} - t_1$

如果 t 小于阈值，则认为未过期，其中阈值随后端服务的稳定性变化，指数增长和回落

降级——降低计算复杂度

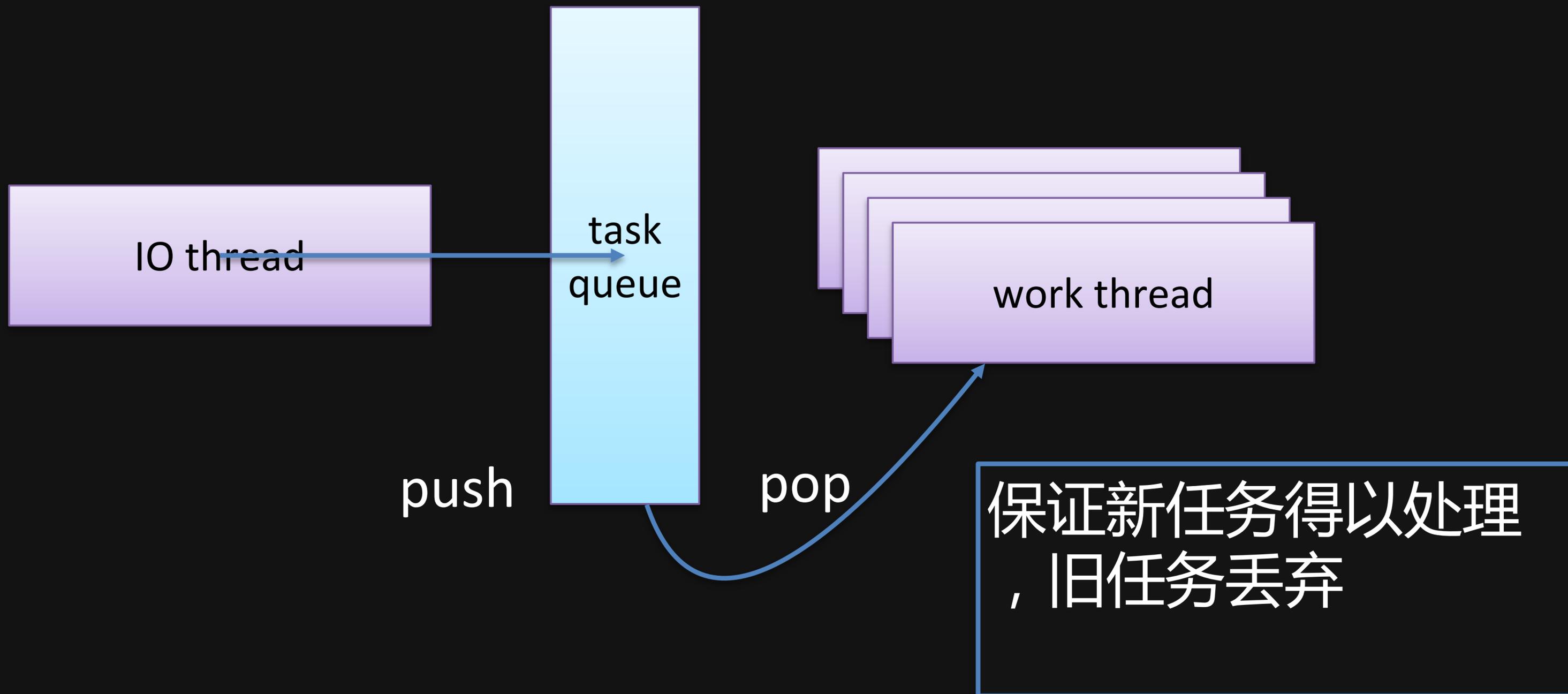
- 人工限制数据集的范围（专辑、短视频、站内、站外等）
- 人工&&自动去掉耗CPU的“锦上添花”的步骤（重排序）

降级——LIFO

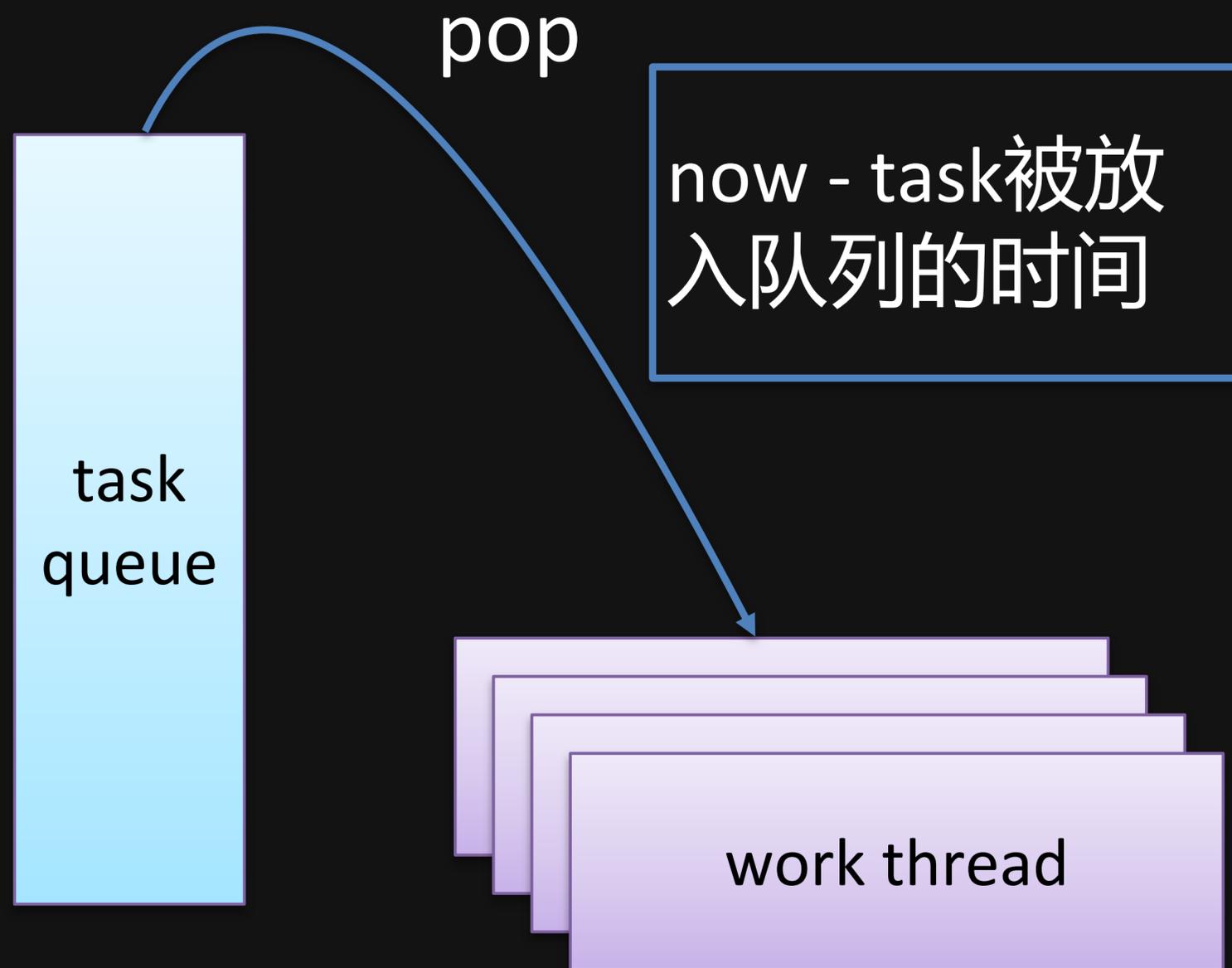


一直处理旧task，处理完后可能已经超时，极端情况下一直在做无用功

降级——LIFO



降级——缩短队列任务过期时间



- 为何要设置过期时间？
- 为何要动态调整过期时间？
- 如何动态调整？
 - 任务的平均响应时间
 - client设置的超时时间
 - 当前机器的CPU
 - task queue长度

打造高可用的搜索架构——限流（限流阈值）

连接数

网络
流量

总qps

CPU

分类qps
(分端
、分服
务)

内存

限流——动态修改限流阈值

client

server的成功率
server的响应时间

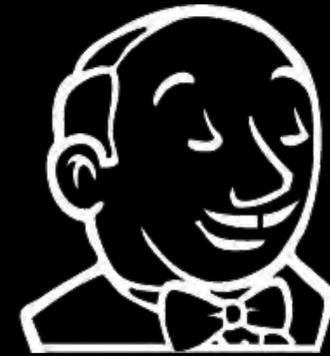
server

task的平均处理时间
队列长度
本进程所消耗的CPU

打造高可用的搜索架构——扩容

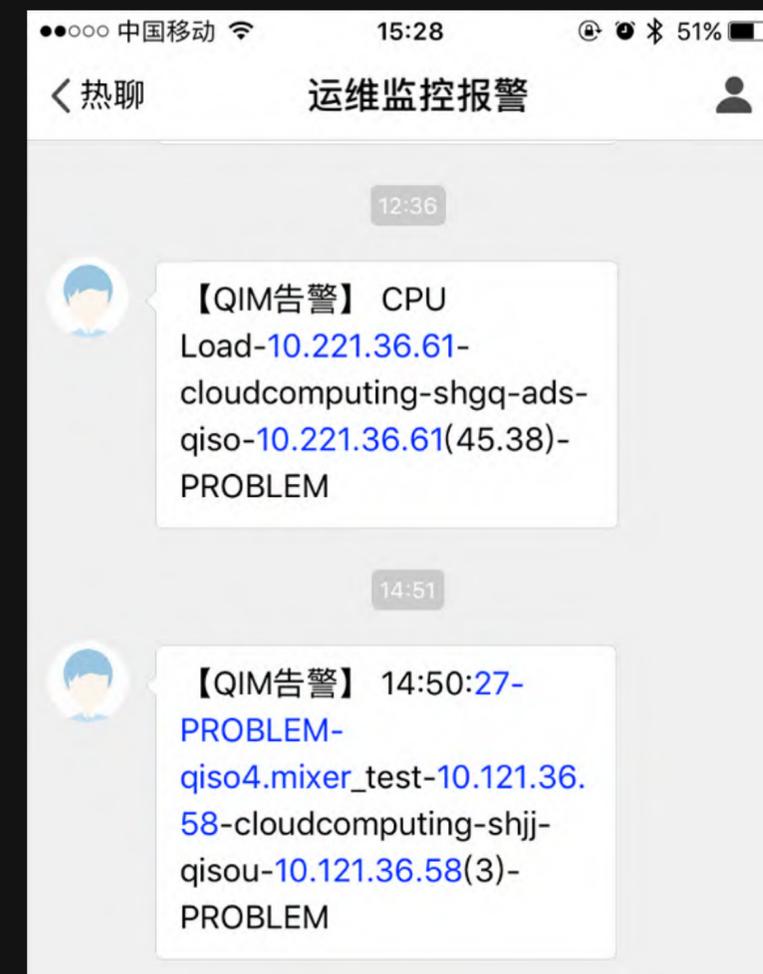
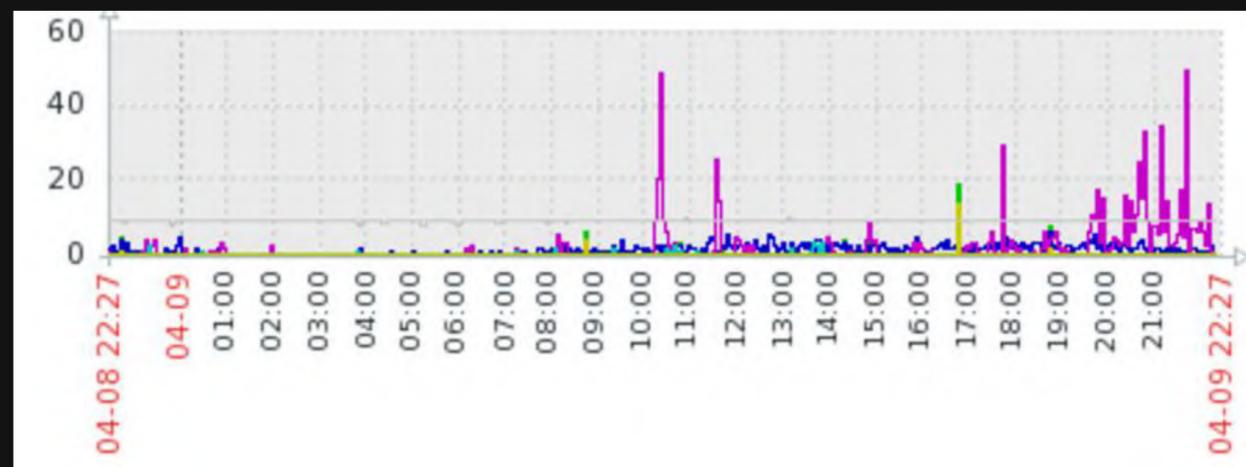
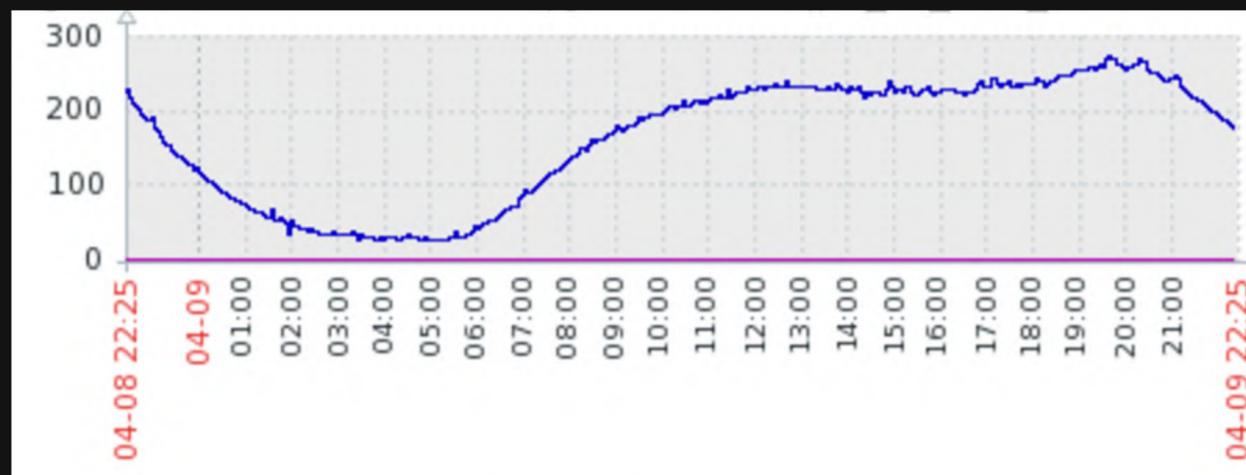


使用docker进行自动扩容



使用jenkins一键部署和扩容

打造高可用的搜索架构——监控系统



打造高可用的搜索架构——trace系统

根据唯一的event id列出各模块间的调用链

Type	Log
online_primary_leaf_shard3	I0416 12:23:11.149147 28560 [redacted] leaf_qps 人民的名义 6 10,ce4e91247f12548795c6ef8b3170650d,90,bkt_main,4,1,1135,eba3f1677f45e05ba1492271360.000000,12 0 0 12 0 0 8 4 0 0
online_real_onetree_leaf	I0416 12:23:11.145061 32246 [redacted] leaf_qps 人民的名义 6 10,ce4e91247f12548795c6ef8b3170650d,90,bkt_main,4,1,614,99c5692883eaa185e1492314880.000000,8 0 0 8 0 0 4 3 0 0
online_primary_leaf_shard2	I0416 12:23:11.175562 10030 [redacted] leaf_qps 人民的名义 6 10,ce4e91247f12548795c6ef8b3170650d,90,bkt_main,4,1,1152,4de8b44960ce313a1492212480.000000,41 0 0 41 0 0 29 11 0 0
online_primary_leaf_shard0	I0416 12:23:11.149381 38297 r [redacted] leaf_qps 人民的名义 6 10,ce4e91247f12548795c6ef8b3170650d,90,bkt_main,4,1,1120,284ad6457f4f2e79d1492207232.000000,13 0 0 12 0 0 7 4 0 0
online_primary_leaf_shard5	I0416 12:23:11.160593 32359 [redacted] leaf_qps 人民的名义 6 10,ce4e91247f12548795c6ef8b3170650d,90,bkt_main,4,1,1130,76ace30dace9865a1492206464.000000,26 0 0 26 0 0 18 7 0 0
online_primary_leaf_shard4	I0416 12:23:11.157480 2042 [redacted] leaf_qps 人民的名义 6 10,ce4e91247f12548795c6ef8b3170650d,90,bkt_main,4,1,1128,afda166240a9838d1492209664.000000,23 0 0 22 0 0 15 7 0 0
online_primary_leaf_shard6	I0416 12:23:11.146709 26392 r [redacted] leaf_qps 人民的名义 6 10,ce4e91247f12548795c6ef8b3170650d,90,bkt_main,4,1,1105,19279a7de114048d1492271360.000000,14 0 0 14 0 0 10 3 0 0

3

挑战及未来计划

挑战及未来计划

提供云
搜索服务

协程

保证服务100%
可用

性能的
极限提
高

more...

Q&A

谢谢



悦 享 品 质