新型架构案例与实践

目录

1. 个人

2.数据同步

3.任务分配

4.接口保护

5. 集群检测

6. 容量模型

个人介绍

•个人信息：

•2008年07月参加工作 工作7.5年。
•热爱Linux内核，在分布式系统 机器学习 并行计算上有一定经验。

•主要经历：

- 华为/亚信

  Linux内核分布式企业存储系统开发
  中国电信海量业务支撑平台设计开发

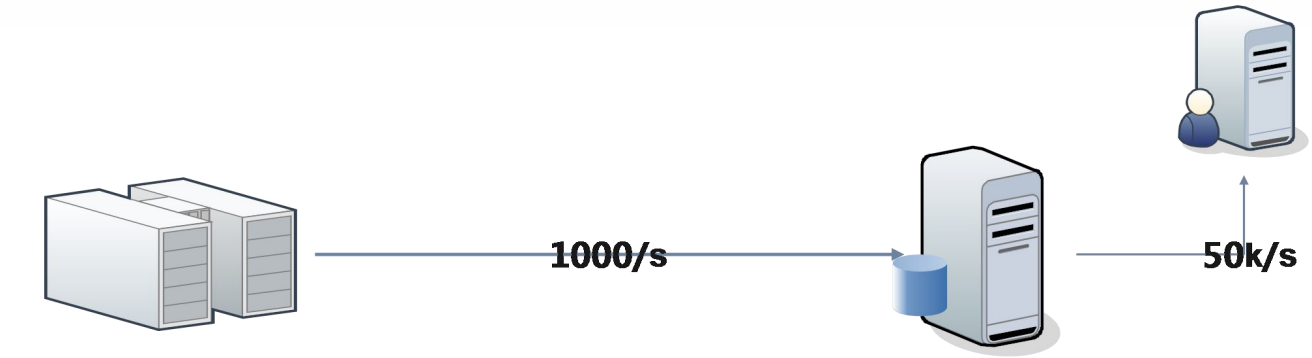
- 百度

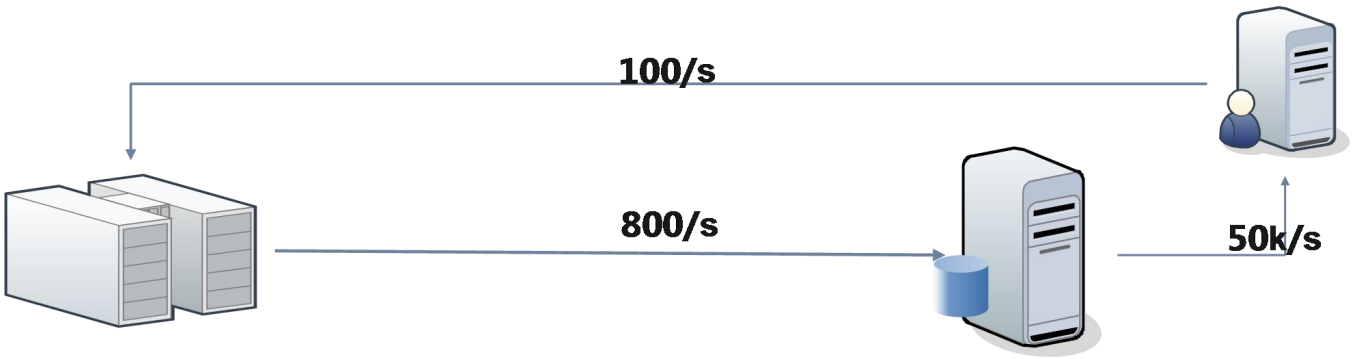
  大规模机器学习平台及算法开发

•2012. 腾讯

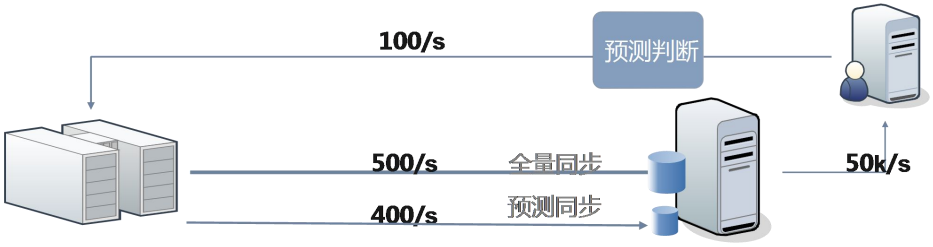
•主导项目： QQ群广告系/旋风下载技术2.0/ QQ公众号 后台负责人

Q：40亿源数据， 接口 1000/s的QPS
副本和源如何保持实时性？


1000/s
50k/s


100/s
800/s
50k/s

人数 X_a
分类 X_b
人数历史变化；资料历史变化 X_c
X_{c1} X_{c2}
。。。



解决方法：

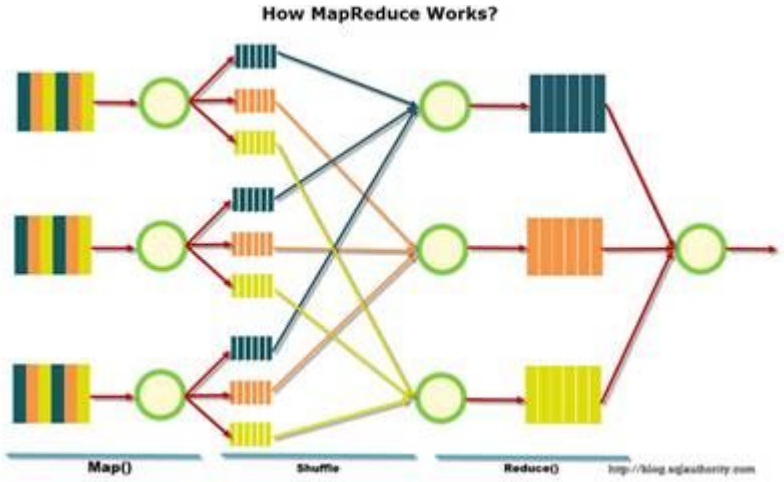
$$P(Y=1 \mid x) = \pi(x) = \frac{1}{1+e^{-g(x)}}$$

- 对于 P(Y=1) 较高的群，我们通过类似行刷新的方式，每隔1小时就可以刷新一次，大概在10分钟内。
- 同时在计费阶段，使用同样的机制对每个匹配中的群进行人数变化,进行预测计算，保证计费准确。

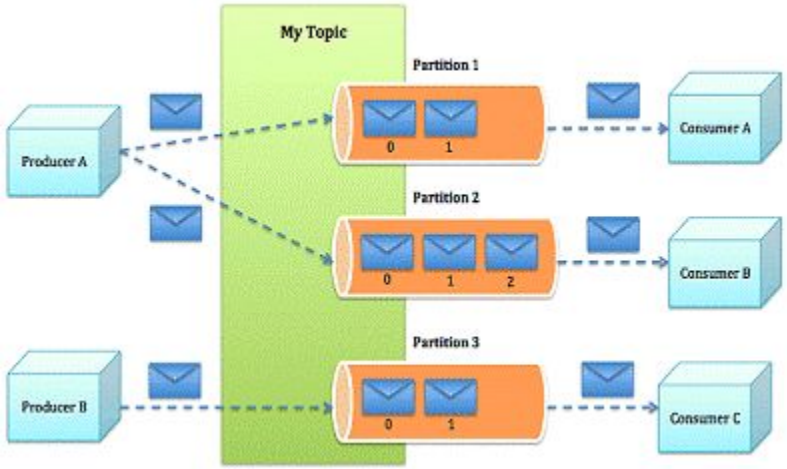
我们把原来千万需要同步数据，在业务可容忍的有损情况下，
通过预测，保证准确度始终在90%以上，将系统同步的代价缩小了一个数量级。

Q：如何 路由？

无主并行化选择:



Plan A

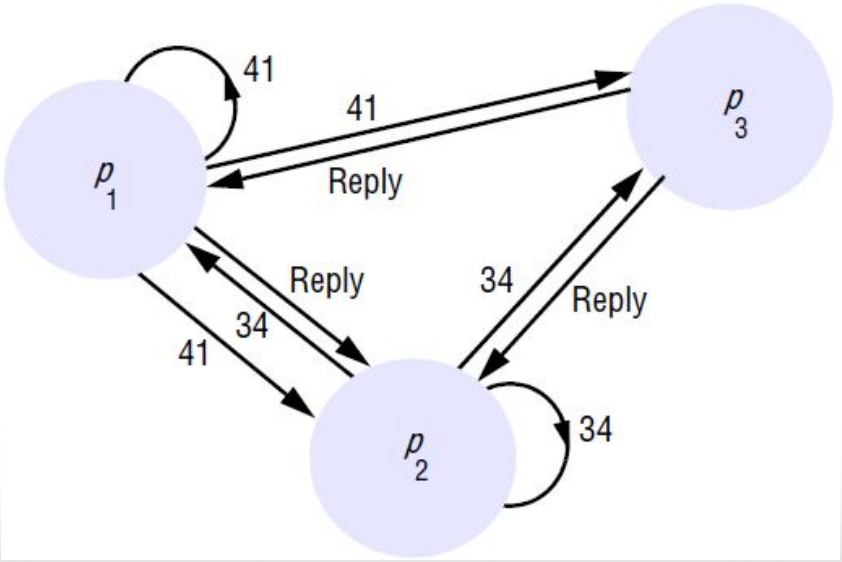


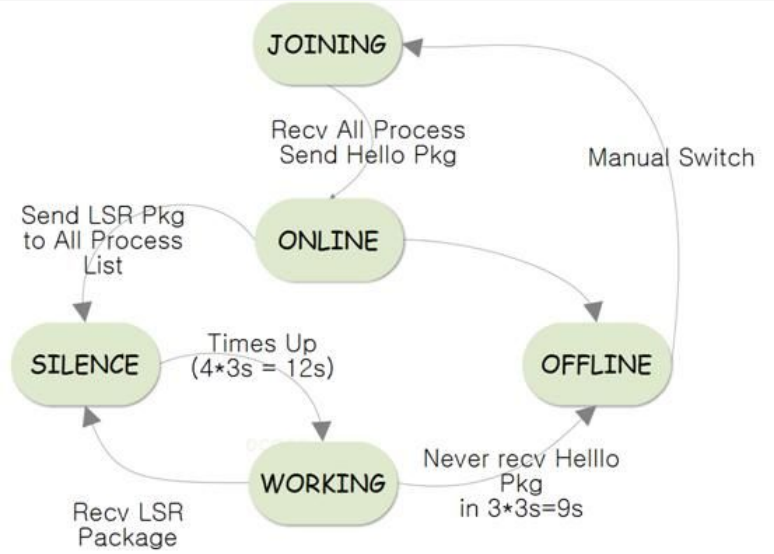
Plan B

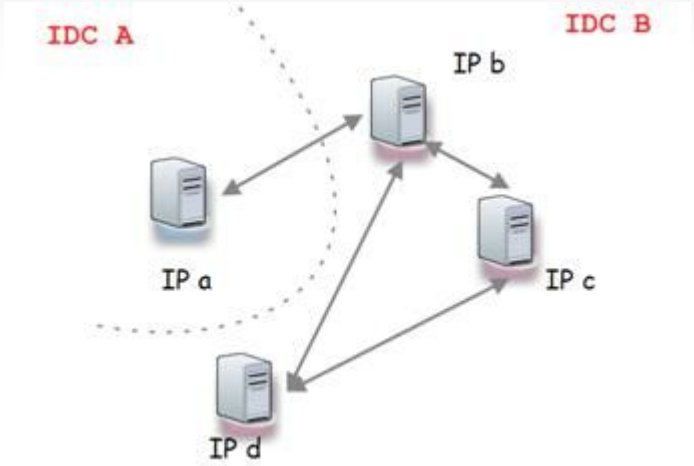
Algorithms for mutual exclusion

- The central server algorithm
- A ring-based algorithm
- An algorithm using multicast and logical clocks
- Maekawa's voting algorithm

Goal : Fault tolerance







作业进程表:

每个机器作业进程中，都维护着这个作业表。 都表明当前系统中 机器的列表，对于WORKING状态的机器会有一个MOD字段，表面当前正在并行处理的任务 ， 按照 WORKING状态机器总数 取模的余数。

静默期
于是我们就需要有一个静默期，也就是SILENCE状态， 在这个静默期间， 任何作业集群中的机器， 都不可以进行任务的获取和处理操所有作。 一个作业集群中的机器 ， 当收到LSR包之后 ， 就开始进入静默状态 。

挑战:

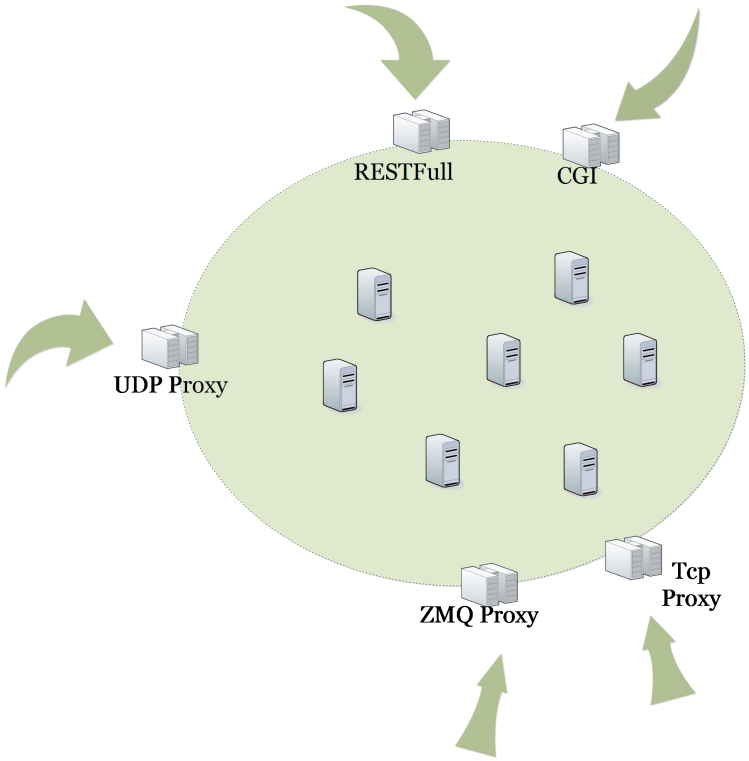
一定需要选举算法?
无主一定是DHT?
时钟同步?

解决:

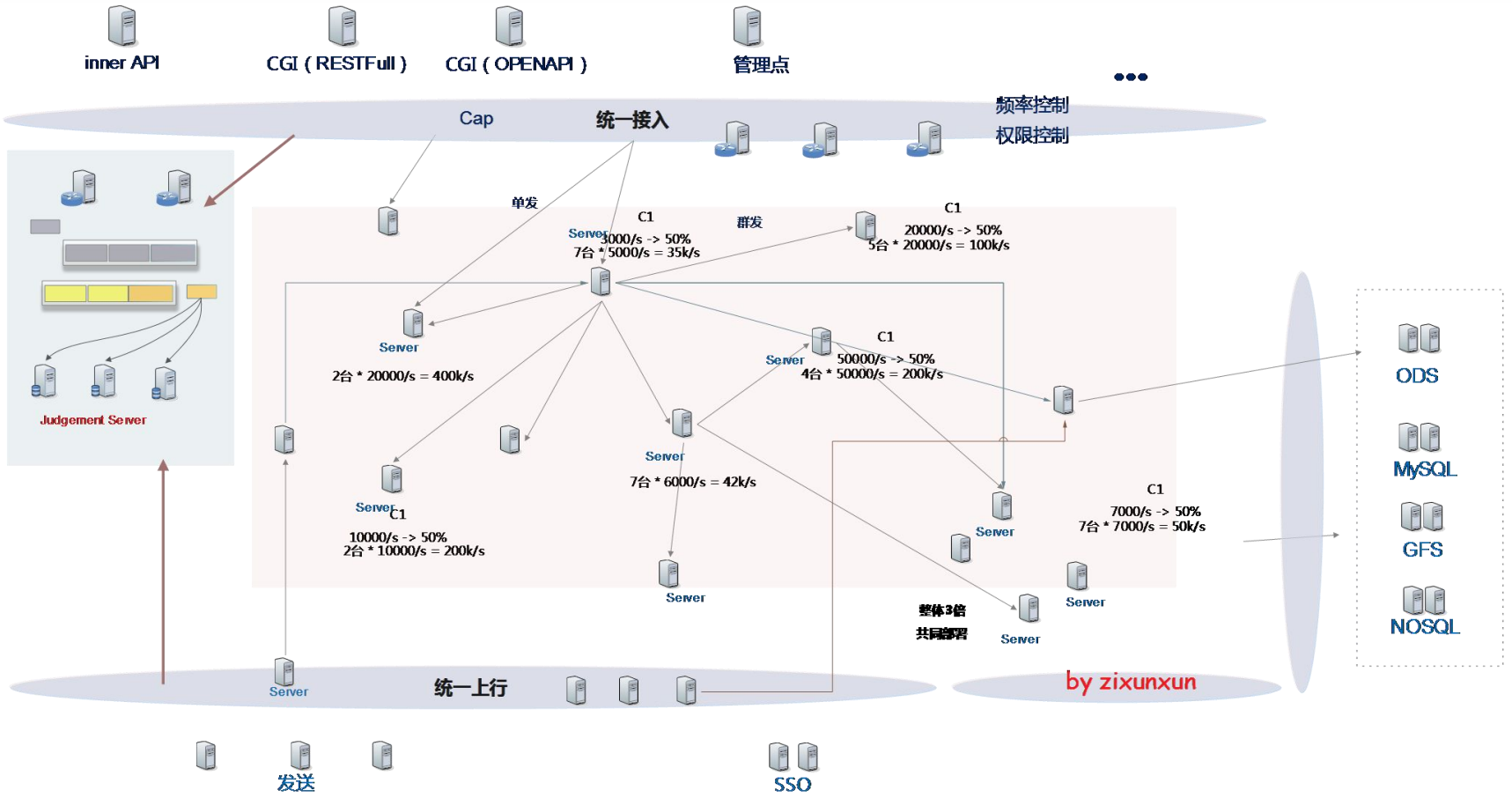
可靠组通讯协议
视图同步
可证明的最大时延

Van Leeuwen等人[LSUZ87]已经证明，可以通过交换更多消息，减少时间复杂度。进行k次扩散过程（使消息复杂度达到$O(k \cdot |E|)$，在$O(p_0^{1/k} D)$个脉冲内就可以找到最小标识。

Q：如何 接口保护？



接口保护


inner API
CGI (RESTFull)
CGI (OPENAPI)
管理点
Cap
统一接入
频率控制
权限控制
Judgement Server
单发
群发
C1
3000/s -> 50%
7台 * 5000/s = 35k/s
C1
20000/s -> 50%
5台 * 20000/s = 100k/s
Server
2台 * 20000/s = 400k/s
C1
50000/s -> 50%
4台 * 50000/s = 200k/s
Server
7台 * 6000/s = 42k/s
C1
10000/s -> 50%
2台 * 10000/s = 200k/s
C1
7000/s -> 50%
7台 * 7000/s = 50k/s
整体3倍
共同部署
ODS
MySQL
GFS
NOSQL
统一上行
by zixunxun
发送
SSO

（毫）秒级控制的难点：

1 主机时间不同步

2 网络延迟问题

3 IDC分区问题

（毫）秒级控制的难点：

1 主机时间不同步

解决： 记录各个上报服务的时间偏移

2 网络延迟问题

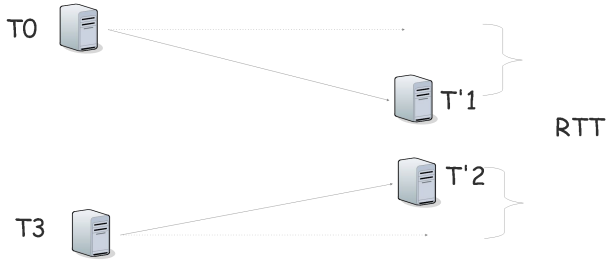
解决： 通过平滑预测函数 ， 得到当前时钟预测值

3 IDC分区问题

解决： 地域统计+ 园区IDC统计

（毫）秒级控制的难点：

主机时间不同步



解决： 记录各个上报服务的时间偏移

T_0：*agent*发送时间

T_1 : *server*接受时间

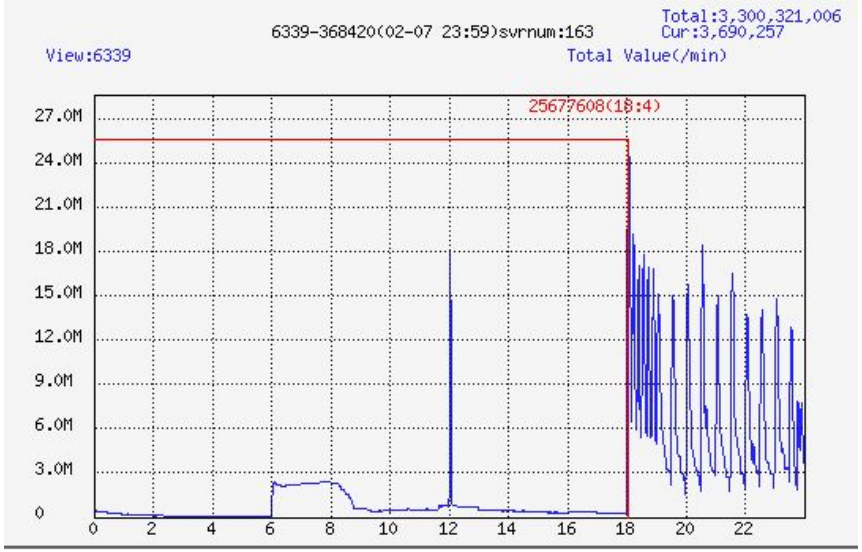
T_2 : *server*发送时间

T_3：*agent*接受时间

$$\Delta T = |(T_1 - T_0) - \frac{((T_3 - T_0) - (T_2 - T_1))}{2}|$$

（毫）秒级控制的难点：

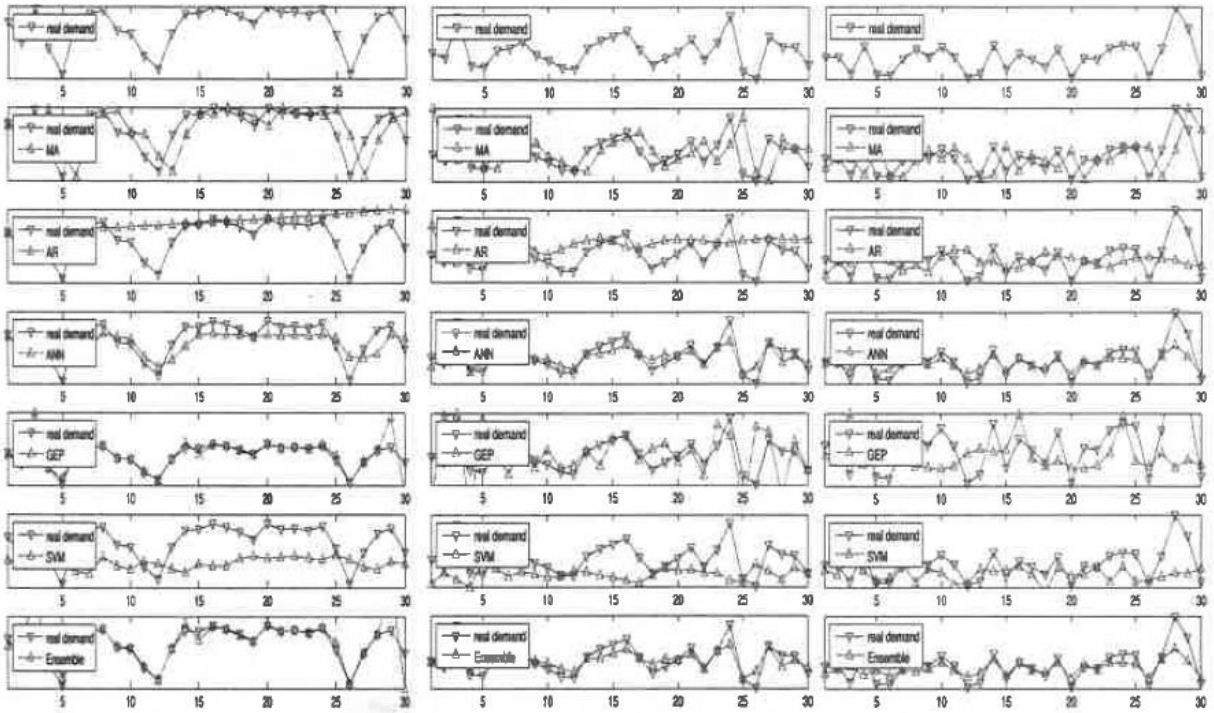
2 网络延迟问题



解决： 通过平滑预测函数 ， 得到当前时钟预测值

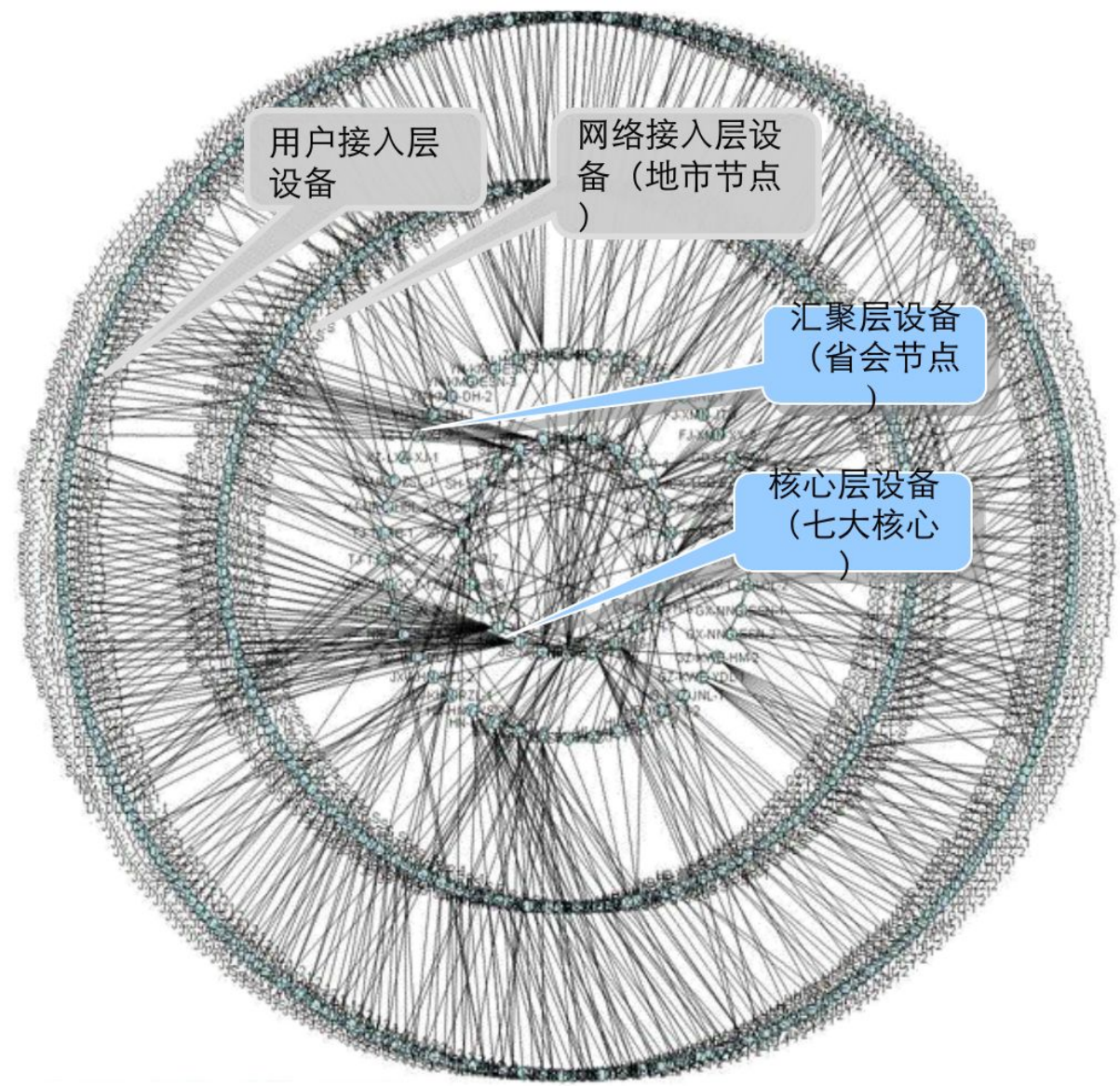
（毫）秒级控制的难点：

滑动平均
自回归模型
神经网络
基因表达式编程
SVM回归机
集成学习

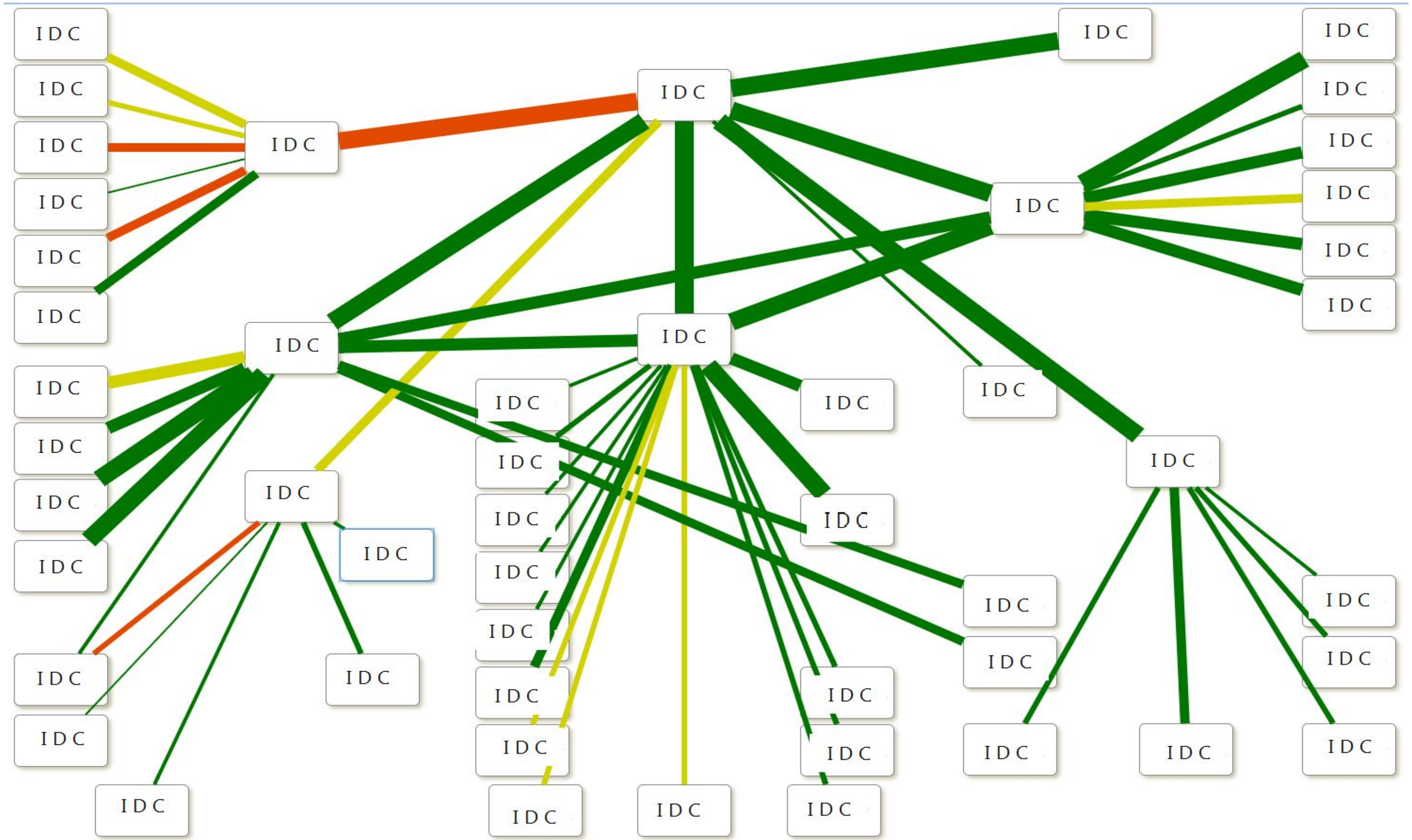


Q：分布式集群中如何找到链路最好的种子节点

集群检测



集群检测

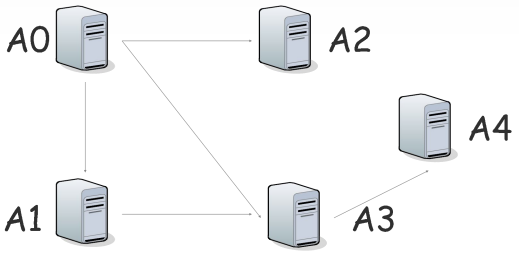


1 我们将传输值矩阵的每一列的所有行上的元素数值进行相加， 如果所求的值，等于0，那么说明 该节点不和其他任何节点相连。这时候输出这些节点，把下标 i放到输出节点序号集合O 中。

2 找出值最高的节点，输出该节点 ，输出这一列的下标 标记为k，放到输出节点序号集合 O 中。

3 用M[k]去和 所有的M[i]的所有元素， 一一对应的进行 fcal(M[k][n],M[i][n])操作。获得新的矩阵 M’ 。对新的矩阵M’ 同样M’ [i] 的表示矩阵的每一列，继续求列所有元素的和。 按值的大小排序，输出除了已经在输出节点序号集合O，值最大的列的下标，同样将其放到输出节点序号集合O中。

4 然后反复步骤3直到所有的矩阵元素的值都为0 。



节点	0	1	2	3	4
0	0	1	1	1	0
1	1	0	1	1	0
2	1	1	0	0	0
3	1	1	0	0	0
4	0	0	0	0	0

	A0	A1	A2	A3	A4
A0	0	1	1	1	0
A1	1	0	0	1	0
A2	1	0	0	0	0
A3	1	1	0	0	1
A4	0	0	0	1	0

3 2 1 3 1

A0

	A0	A1	A2	A3	A4
A0	0	1	1	1	0
A1	0	0	0	0	0
A2	0	0	0	0	0
A3	0	0	0	0	0
A4	0	0	0	1	0

0 1 1 2 0

A0 A3

	A0	A1	A2	A3	A4
A0	0	0	0	0	0
A1	0	0	0	0	0
A2	0	0	0	0	0
A3	0	0	0	0	0
A4	0	0	0	0	0

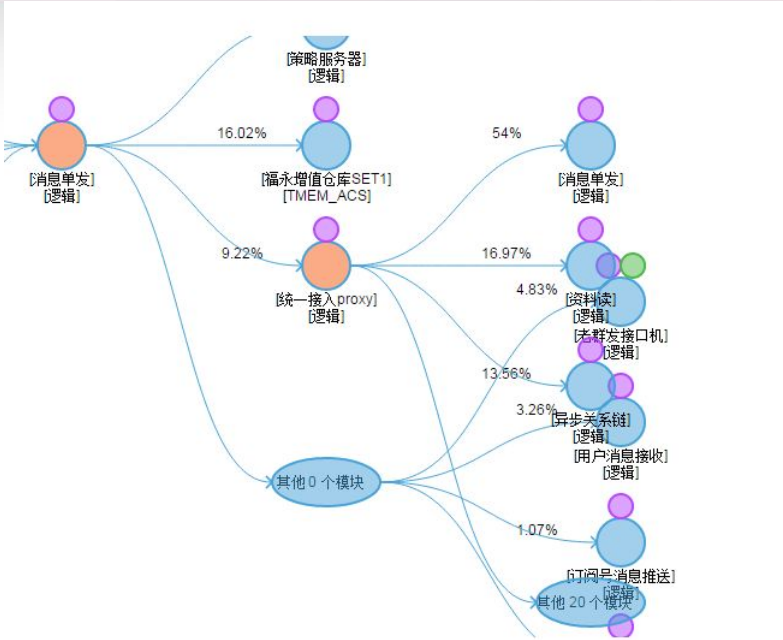
0 0 0 0 0

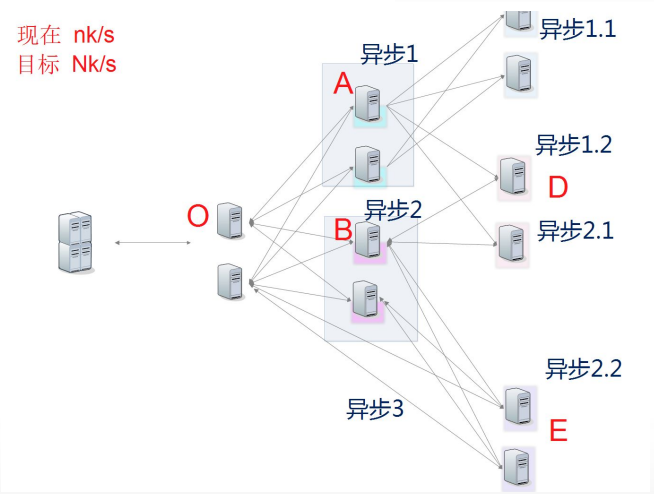
A0 A3

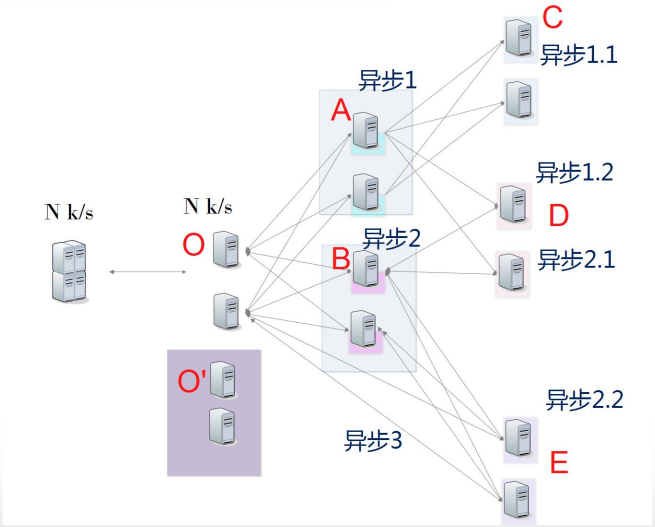
Q：如何评估容量模型和扩容？

容量模型

通用模型


[策略服务器]
[逻辑]
16.02%
[福永增值仓库SET1]
[TMEM_ACS]
54%
[消息单发]
[逻辑]
[消息单发]
[逻辑]
9.22%
[统一接入proxy]
[逻辑]
16.97%
4.83%
[资料读]
[逻辑]
[无群发接口机]
[逻辑]
13.56%
3.26%
[异步关系链]
[逻辑]
[用户消息接收]
[逻辑]
其他 0 个模块
1.07%
[订阅号消息推送]
[逻辑]
其他 20 个模块


现在 nk/s
目标 Nk/s
异步1
A
异步1.1
异步1.2
D
O
异步2
B
异步2.1
异步2.2
异步3
E


N k/s
N k/s
C
异步1.1
异步1
A
异步1.2
D
O
异步2
B
异步2.1
O'
异步2.2
异步3
E

容量模型

通用模型

$$\int_a^b f_{request}(x)dx = \lim_{n\to\infty}\sum_{i=0}^{\infty}\frac{b-a}{n}f_{request}(t_i)$$

$$\int_{a+t}^{b+t} f_{response}(x)dx = \lim_{n\to\infty}\sum_{i=0}^{\infty}\frac{b-a}{n}f_{response}(t_i)$$

$$1s \Rightarrow L$$

$$1s = \sum T_{Brust}$$

$$J \Rightarrow T_{Brust}$$

$$All_{except} = J \bullet {}^{1}\!/_{T_{Brust}}$$

$$L = All_{reality} = J \bullet {}^{1}\!/_{(T_{Brust} + 2T_{rtt/2} + T_1 + T_2 + T..)}$$

$$T_1 = f(Num_1, {}^{1}\!/_{P_1}, T_{rtt})$$

$$Num_i = f({}^{J}\!/_{NodeNum}, k)$$

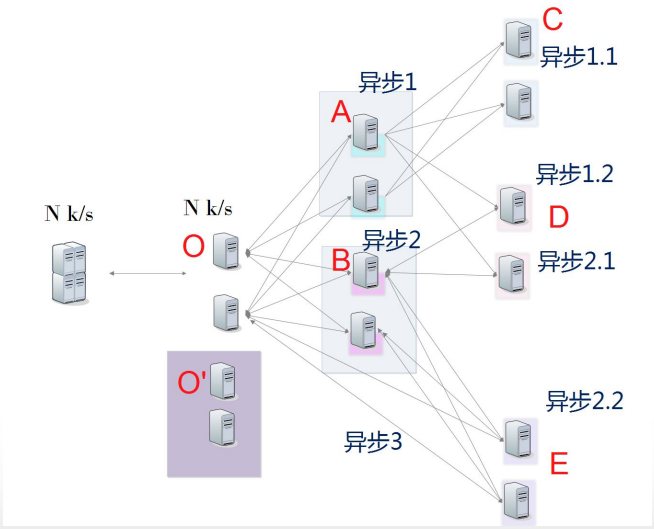
$$Num_i \propto \mathrm{k}$$

$$\sum_i^k Num_i = J$$

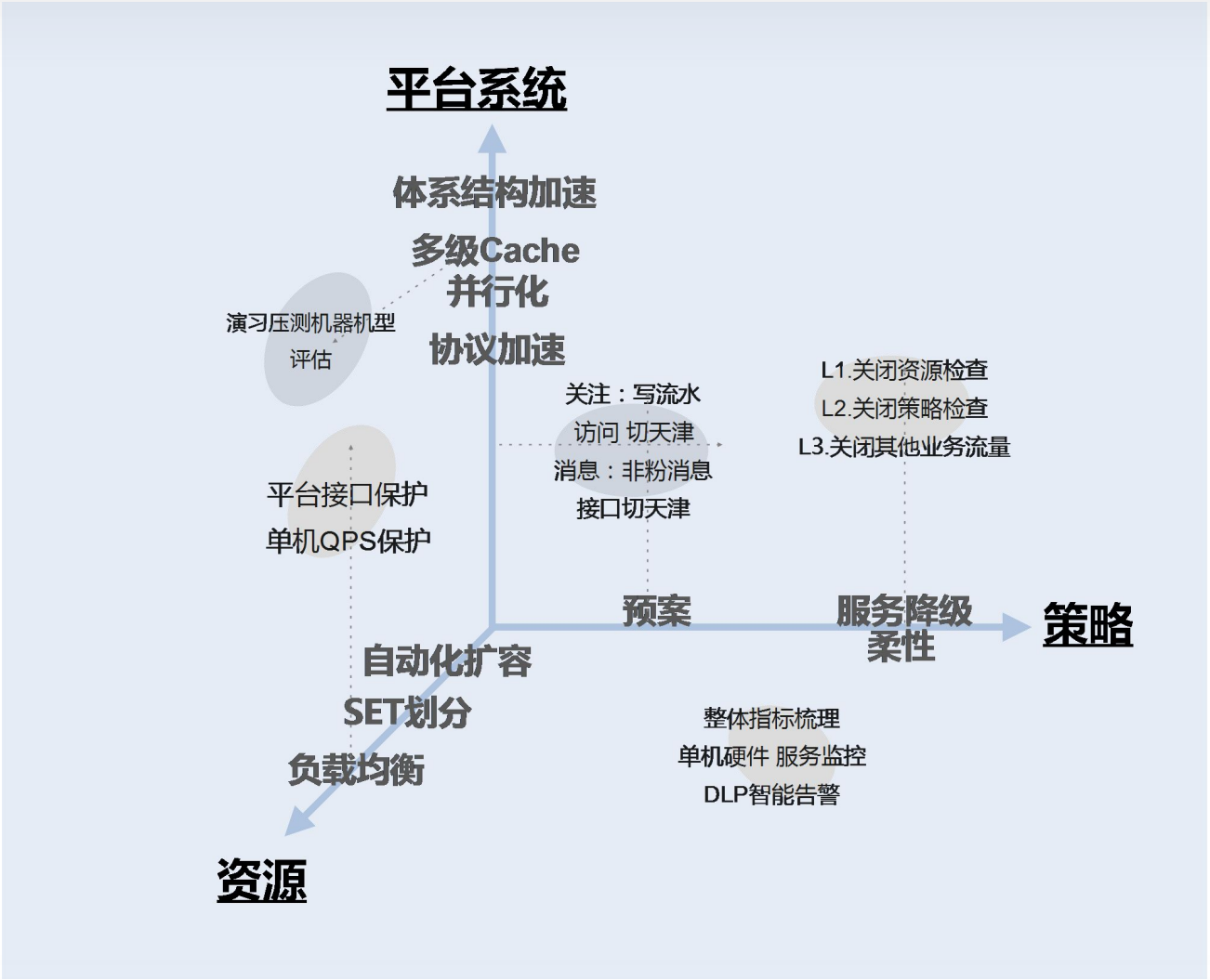
接口压力上不去？ 应该扩容哪个服务

Get 基本压力下接口耗时视图;

do（还有叶子||流量没到目标）
增加流量;
分析叶子节点耗时，按需扩容
i f(上层流量上不去 && 下层耗时没有增加）
叶子节点上移
else
下一个叶子节点
loop;



容量模型


平台系统
体系结构加速
多级Cache
并行化
协议加速
演习压测机器机型
评估
关注：写流水
访问 切天津
消息：非粉消息
接口切天津
L1.关闭资源检查
L2.关闭策略检查
L3.关闭其他业务流量
平台接口保护
单机QPS保护
预案
服务降级
柔性
策略
自动化扩容
SET划分
负载均衡
整体指标梳理
单机硬件 服务监控
DLP智能告警
资源

- 谢谢大家


互联网应用架构实战峰会
CSDN