北京应用物理与计算数学研究所
Institute of Applied Physics and Computational Mathematics

高性能计算机运维实践与思考

罗红兵

2019年11月30日

主要内容

一、背景

二、近年来的关注点

- 系统部件亚健康的诊断
- 业务程序特征分析
- 并行程序性能诊断

三、展望

美国能源部超级计算机规划

DOE HPC Facilities Systems

Pre-Exascale Systems

2013 | 2016 | 2018 | 2020

Argonne
IBM BG/Q
Unclassified

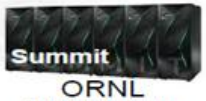
Argonne
Intel/Cray KNL
Unclassified

LBNL
Cray/Intel Xeon/KNL
Unclassified

ORNL
Cray/NVidia K20
Unclassified

LLNL
IBM BG/Q
Classified

LANL/SNL
Cray/Intel Xeon/KNL
Classified



ORNL
IBM/NVidia
P9/Volta
Unclassified

LLNL
IBM/NVidia
P9/Volta
Classified

NERSC-9

LBNL
TBD
Unclassified

Crossroads

LANL/SNL/2021
Solicit
Bids(18MW)
Classified

Exascale Systems

2021-2022

Argonne
Intel/Cray TBD
Unclassified

Frontier

ORNL/2021
Cray/1.5Eflops
Unclassified

El Capitan

LLNL/2022
Cray/1.5Eflops
Classified

Argonne NATIONAL LABORATORY
Lawrence Livermore National Laboratory
OAK RIDGE National Laboratory

关注的问题

- 对象
  - 大规模并行计算机
  - 大规模并行程序
- 问题
  - 计算机运行状态
  - 计算机的使用效能
    - 整体状况
    - 性能改进

主要内容

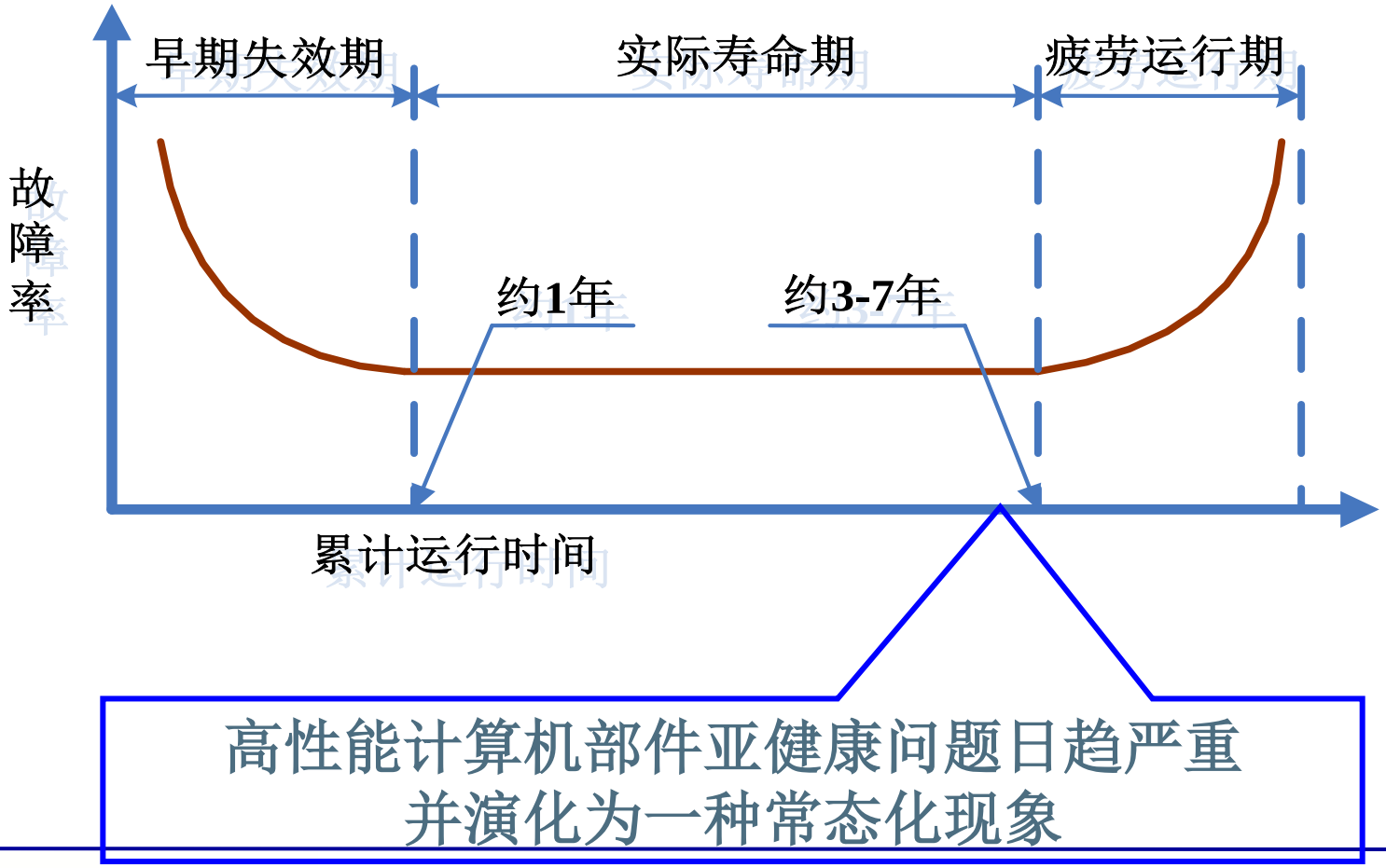
一、背景

二、近年来的关注点

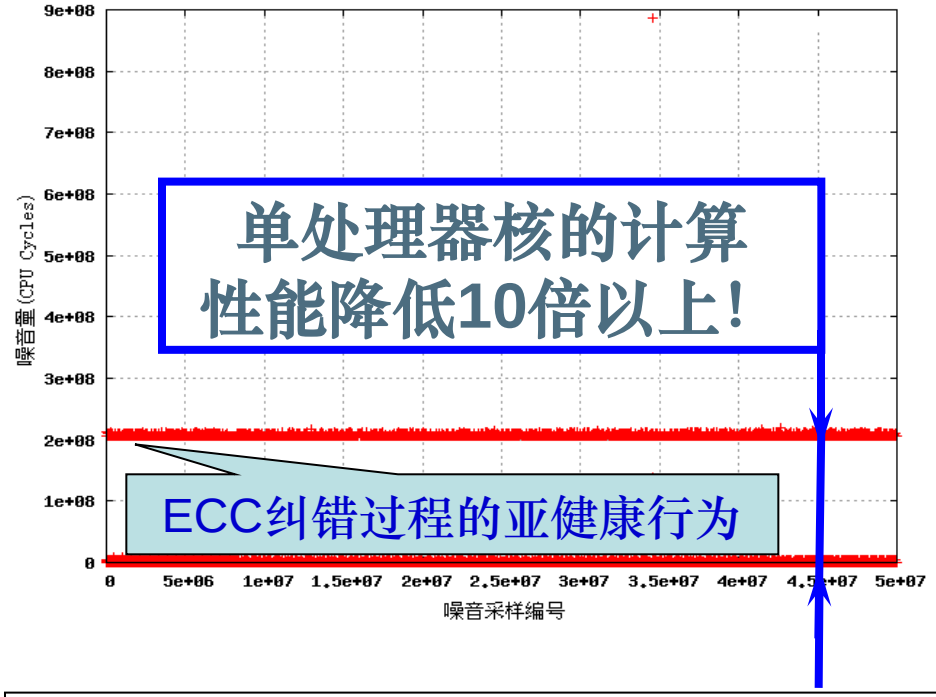
- 系统部件亚健康的诊断
- 业务程序特征分析
- 并行程序性能诊断

三、展望

关注点一：能力型计算机的亚健康现象



能力型计算机的亚健康现象



通信链路的带宽

- 正常的带宽
  - 点对点双向带宽10GB/s
- 亚健康带宽:
  - 点对点双向带宽<6GB/s

通信性能降低50%!

亚健康部件必然导致部件的实际输出性能下降，
从而导致数值模拟应用程序呈现**大幅度的性能下降!**

能力型计算机的亚健康现象

部件亚健康的主要特征

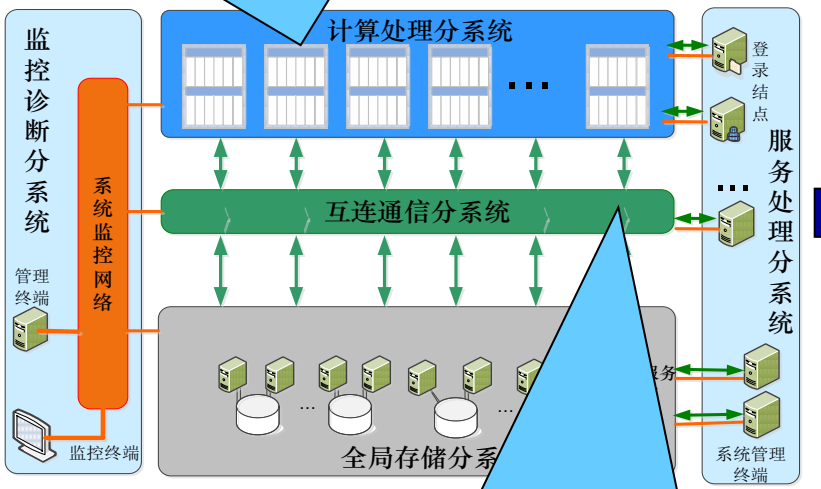
1. 系统的“正常”行为：并非“失效”
2. 以性能换取运行稳定性：部件的“降级运行”模式
3. 高负载情况下易被触发
4. 发生概率逐年升高：能力型计算机寿命的后半部
5. 无有效监控手段：监控信息不足、诊断困难！

部件亚健康的影响

1. 数值模拟应用的性能大幅下降
2. 严重影响大规模作业的成功率
   - 万核规模Lared-S成功率不足十分之一

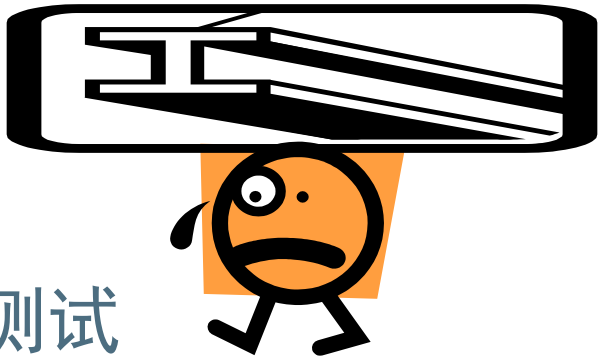
如何在海量部件中快速定位亚健康部件?

核心部件一：计算、访存
数万、上百万CPU核及内存条



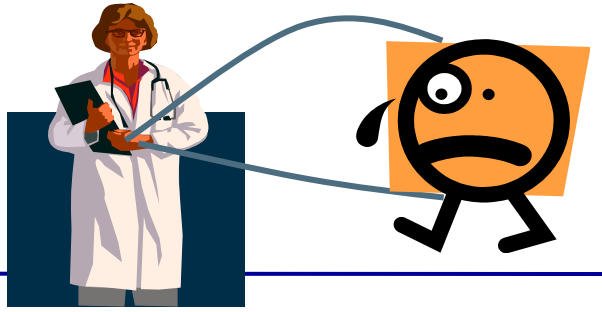
核心部件二：高速通信网络
数十万个交换机端口及光纤线路

方法一：实际负载



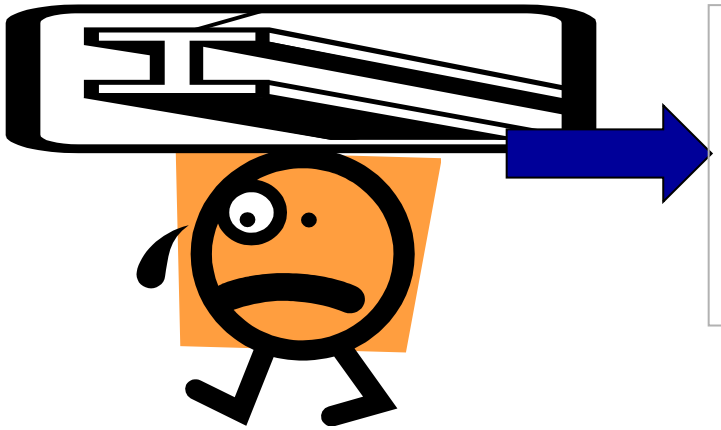
并行测试

方法二：离线观察



部件亚健康问题的诊断方法

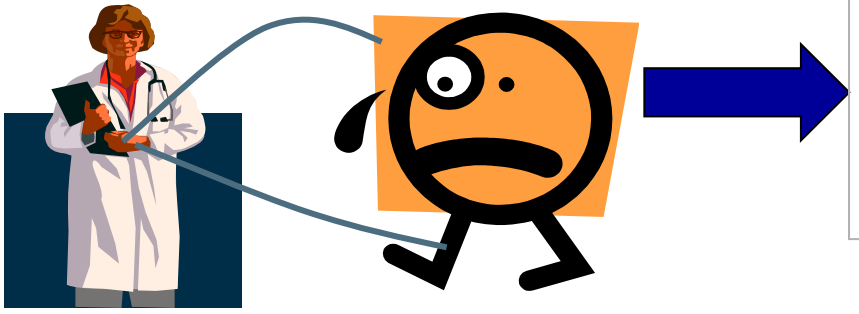
方法一：实际负载



1. 优点：结果直观
2. 缺点：独占测试
3. 难点：测试集的构建

   准确度 Vs. 开销

方法二：离线观察



1. 优点：不影响作业
2. 缺点：准确度
3. 难点：监控状态的选择

部件亚健康问题的诊断方法

计算部件亚健康诊断：采用方案一 （实际负载）

```
int  N,W;        /*W为负载大小;N为W负载的执行次数*/
while(count<N) {
      T1=rdtsc();          /* 记录当前时刻T1 */
      do_FWQ(W);           /*执行大小为W的固定计算负载*/
      T2=rdtsc();          /* 记录当前时刻T2 */
      td=td+T2-T1;        /*获得总计算时间*/
}
```

测试集：

```
#defin      W        a^=a+a; a^=a+a+a; \
                     a>>=b;  a>>=a+a; \
                     a^=a<<b; a^=a+b; \
                     a+=(a+b)&07;a^=n; \
                     b^=a;        a|=b;
```

时间开销：2毫秒

部件亚健康问题的诊断方法

访存部件亚健康诊断：采用方案一 （实际负载）

测试集：

```
for(j=0; j<step; j++) {
      for(i=j; i<num; i+=step) {
            dst[i] = src[i];
      }
}
```

num：数据集大小

- 至少为L3 Cache的两倍

Step：数据读写步长

- 至少为Cache line大小

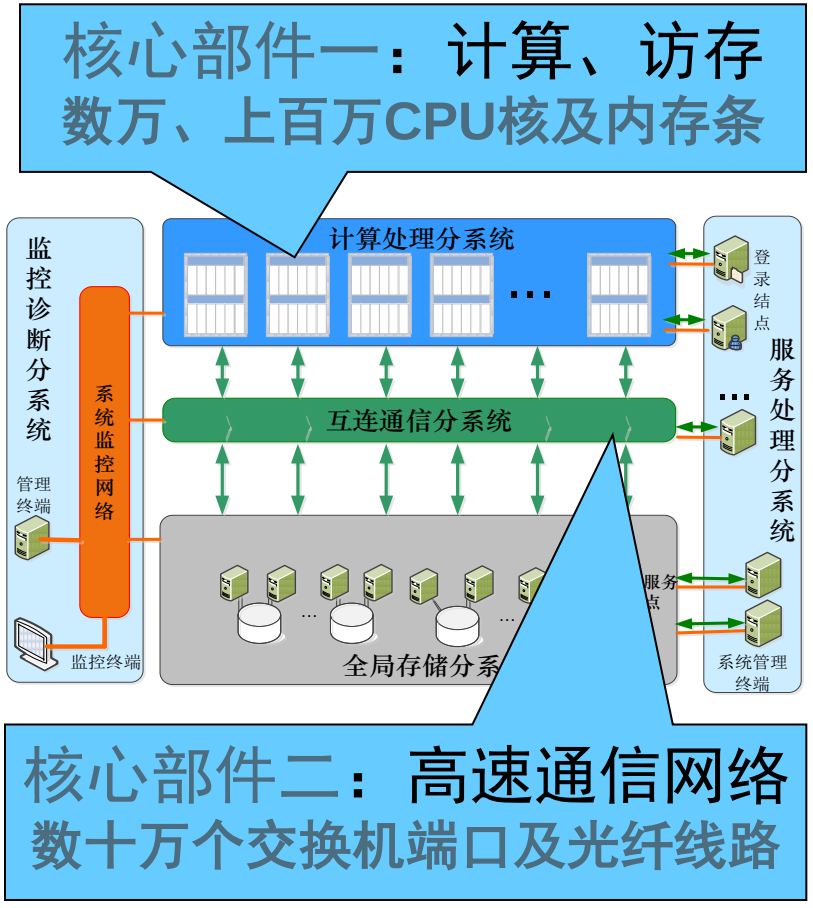
24M数据集
步长64字节
开销：15毫秒

低开销、准确度高的测试负载

计算负载+访存负载

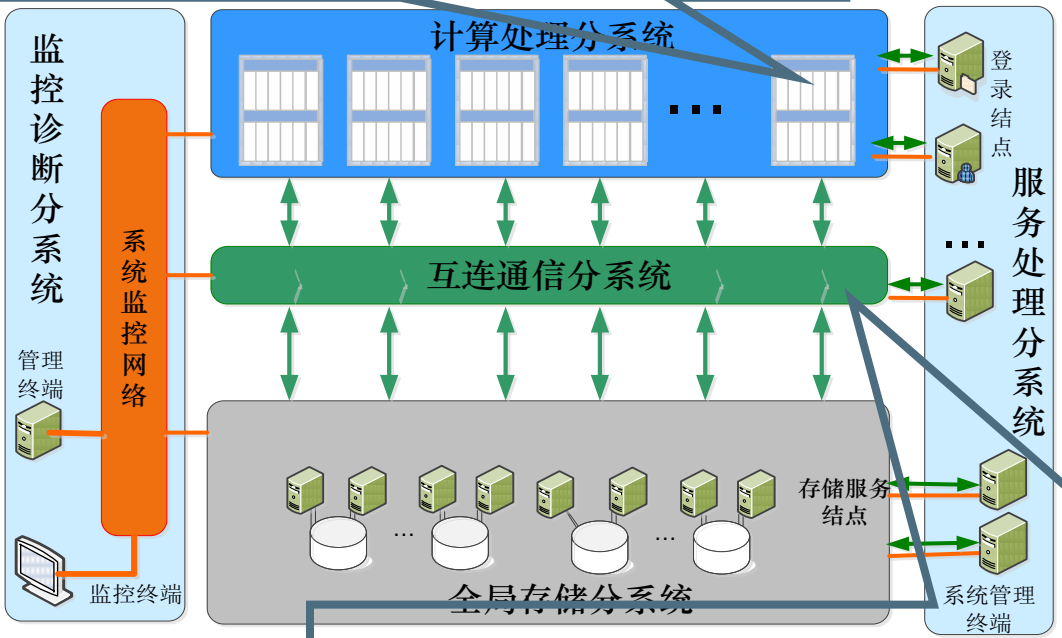
总开销仅17毫秒

√



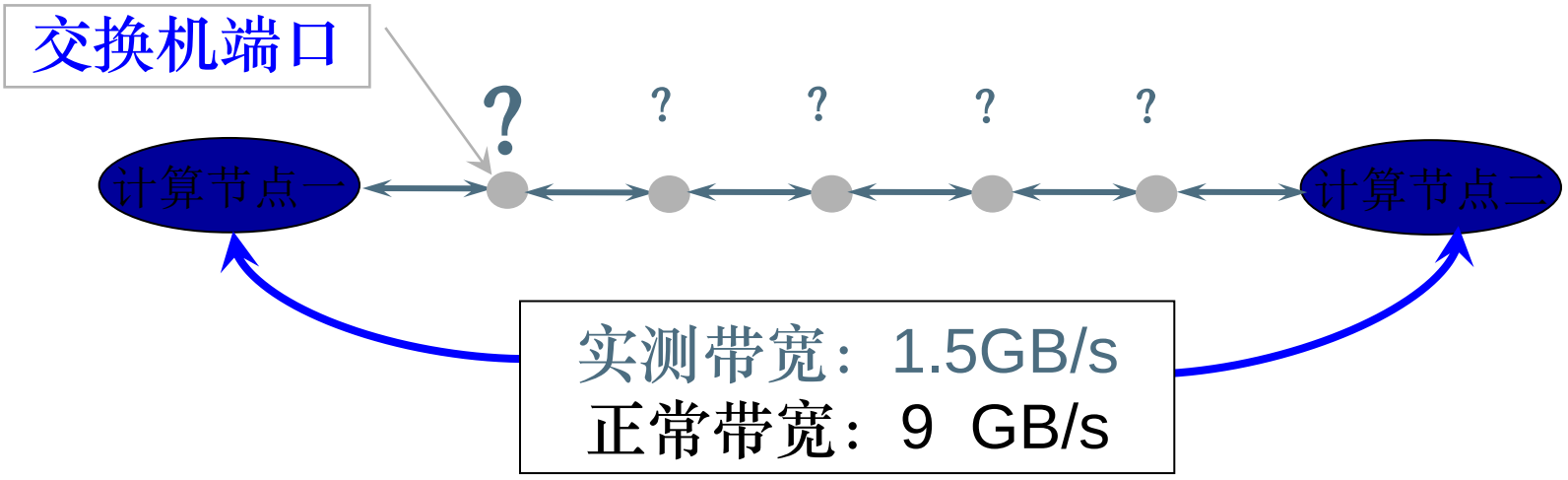
通信部件亚健康的诊断

计算、访存部件的亚健康：局部故障



通信部件的亚健康：全局故障

通信部件亚健康的诊断



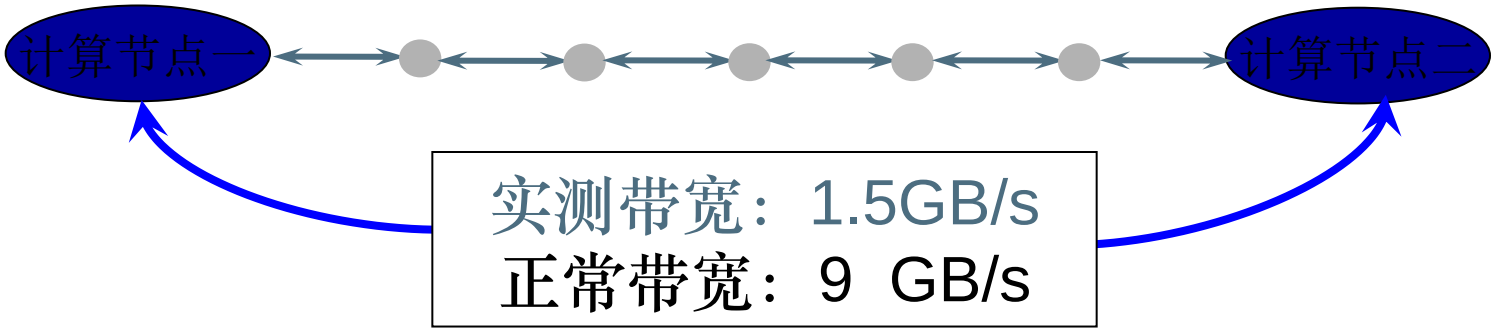
难点：

1：通信距离长，端到端的性能异常不能定位故障位置

2：通信网络存在共享，需独占测试，测试开销大

- 3000节点，需900万次测试，每次1秒：需2500小时

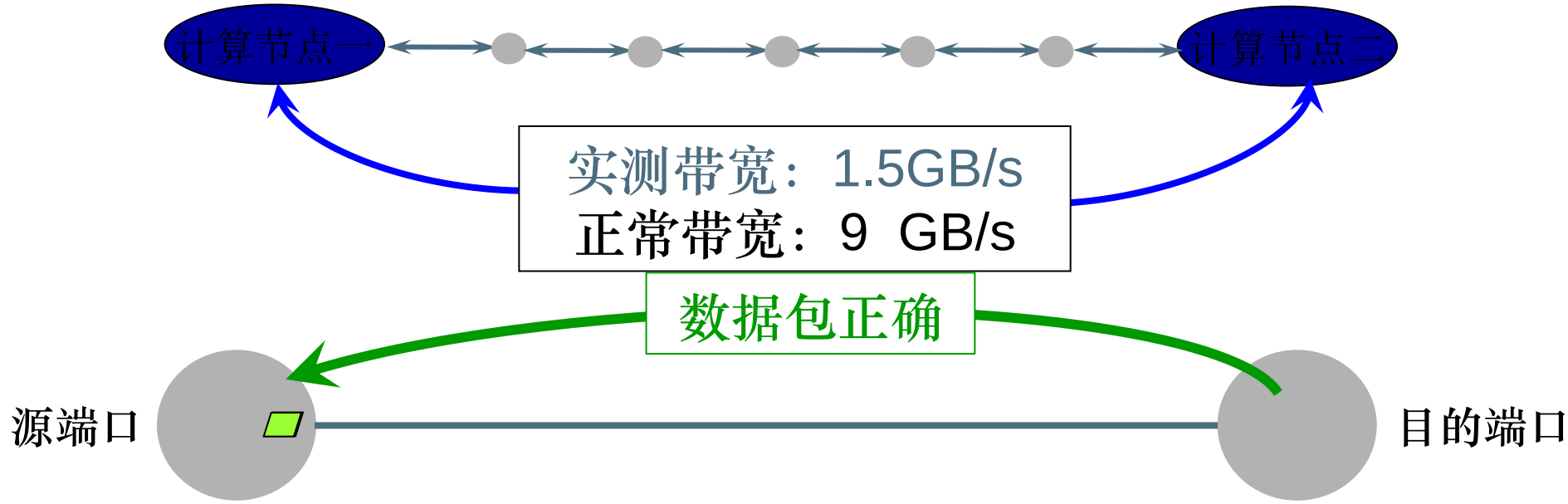
通信部件亚健康的诊断



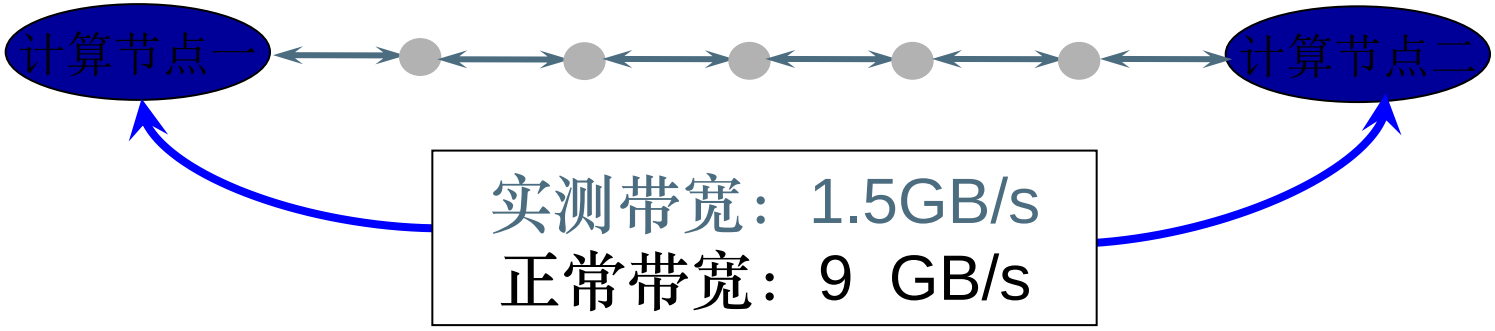
离线观测方法

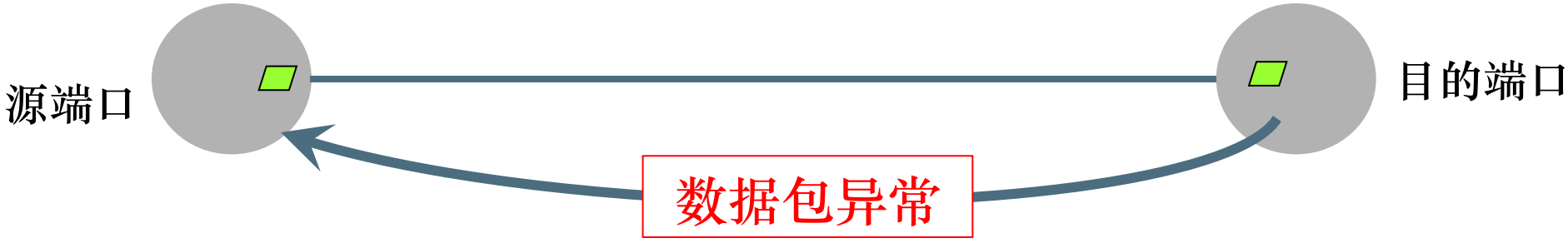
问题：如何构建体现通信部件健康状况的监控状态?

通信部件亚健康的诊断



通信部件亚健康的诊断

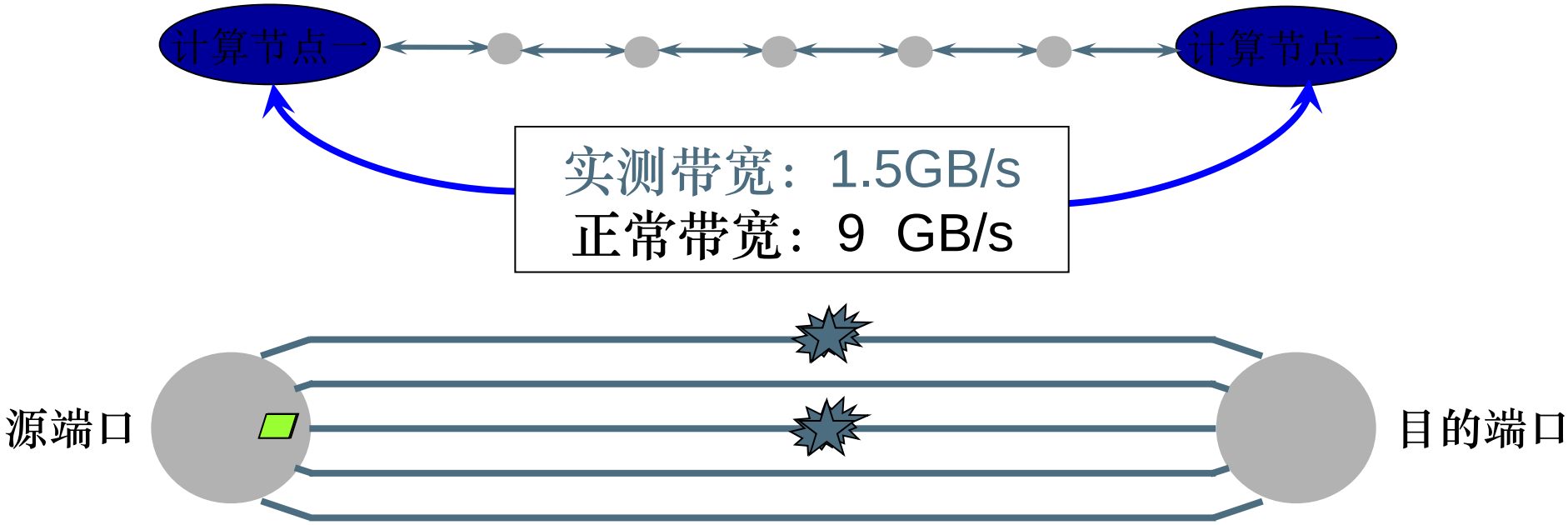




通信部件健康状况的监控状态（1）：端口重传次数

体现通信链路的通信质量

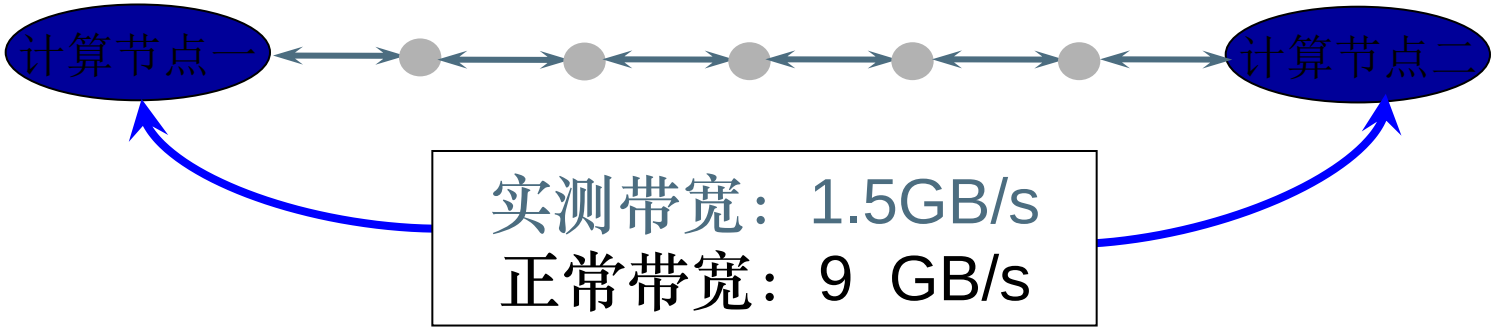
通信部件亚健康的诊断



通信部件健康状况的监控状态（2）：端口通道数量

体现通信链路的通信性能

通信部件亚健康的诊断



状态一：端口重传次数

状态二：端口通道数量

优点：

1：定位准确：精确到每一个“端口”

2：监控状态少：2个

3：监控开销小：全系统数分钟

4：对作业无干扰

应用效果

针对计算、访存部件的亚健康诊断：

已应用到银河、曙光、专项机等系统

- 数分钟内完成全系统计算节点的健康状况评估
- 已成为系统每天的检查工具

形成能力型计算机操作系统行为分析软件NoiseProfiler2.0版

针对通信系统的亚健康诊断：

1：应用在银河机、专项机：

- 通信部件亚健康的诊断开销：数天—>数小时—>数分钟

2：保障银河机在延寿期内的高效运行

- Lared-S：12000规模稳定运行超1天

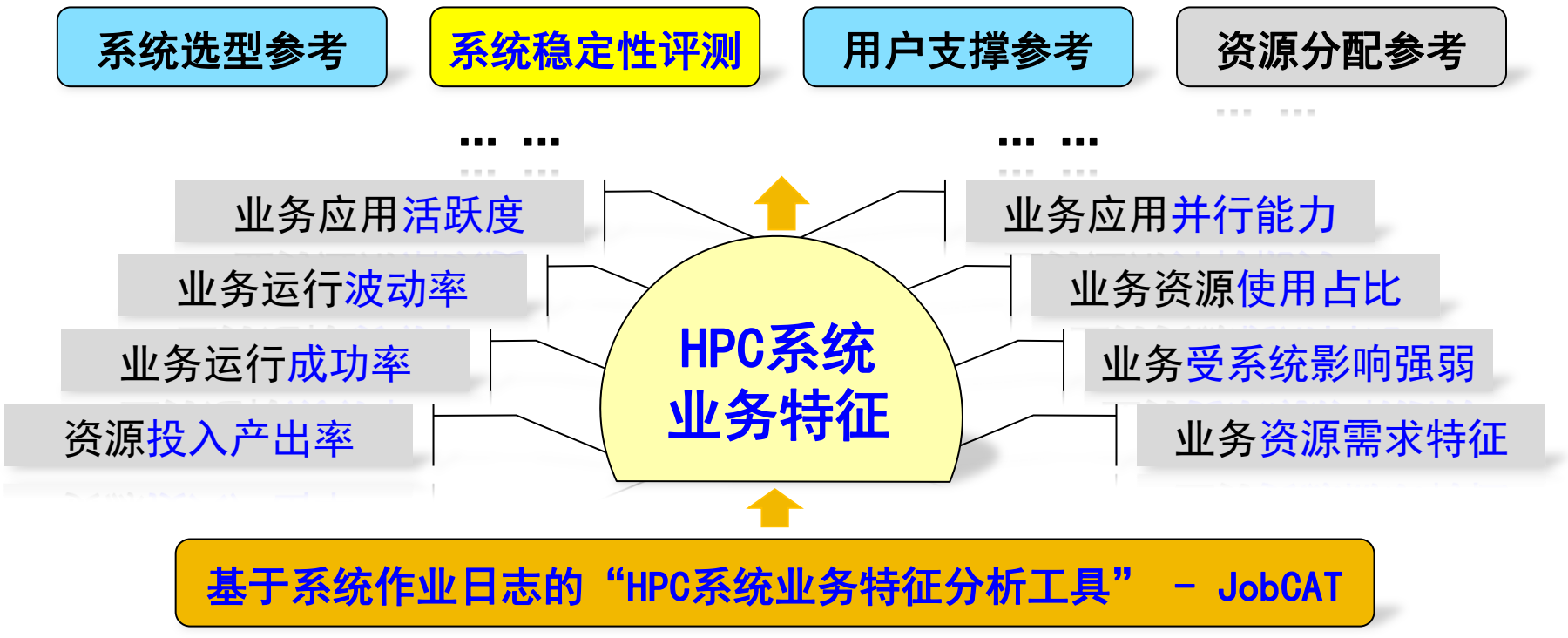
形成能力型计算机互联通信系统的监控与诊断软件Netview1.0版

关注点二：数值模拟作业的特征统计

数值模拟作业的特征统计工具jobCAT

- 实现高性能计算机上数值模拟作业特征的量化分析能力
- 快速甄别各科室在系统上实际使用的应用程序
- 定量分析各数值模拟程序的活跃度情况
- 为应用程序性能优化、系统稳定效能分析等提供支撑

需求：分析我所数值模拟业务特征，为各类需求提供基础数据支撑。



关键技术： 基于系统作业日志的“HPC系统业务特征分析工具”

■ 两个关键技术问题

```
[root@mn3%YH source]# yhacct --helpformat
Account          AdminComment     AllocCPUS        AllocGRES
AllocNodes       AllocTRES        AssocID          AveCPU
AveCPUFreq       AveDiskRead      AveDiskWrite     AvePages
AveRSS           AveVMSize        BlockID          Cluster
Comment          ConsumedEnergy   ConsumedEnergyRaw CPUTime
CPUTimeRAW       DerivedExitCode  Elapsed          ElapsedRaw
Eligible         End              ExitCode         GID
Group            JobID            JobIDRaw         JobName
Layout           MaxDiskRead      MaxDiskReadNode  MaxDiskReadTask
MaxDiskWrite     MaxDiskWriteNode MaxDiskWriteTask MaxPages
MaxPagesNode     MaxPagesTask     MaxRSS           MaxRSSNode
MaxRSSTask       MaxVMSize        MaxVMSizeNode    MaxVMSizeTask
MinCPU           MinCPUNode       MinCPUTask       NCPUS
NNodes           NodeList         QOS              Reservation
Partition        ReqCPUFreqMax    ReqCPUFreqMin    ReservationId
ReqCPUFreqMin    ReqMem           ReqNodes         Start
ReqGRES          ReservationId    Reservation      State
ReservationId    Start            ResvCPU          Submit
ResvCPURAW       SystemCPU        ResvCPURAW       Suspended
Suspended        Timelimit        Start            TotalCPU
UID              User             State            UserCPU
WCKeyID          UserCPU          Submit           WCKey
```

COMPLETED, TIMEOUT, NODE_FAIL, FAILED, CANCELLED

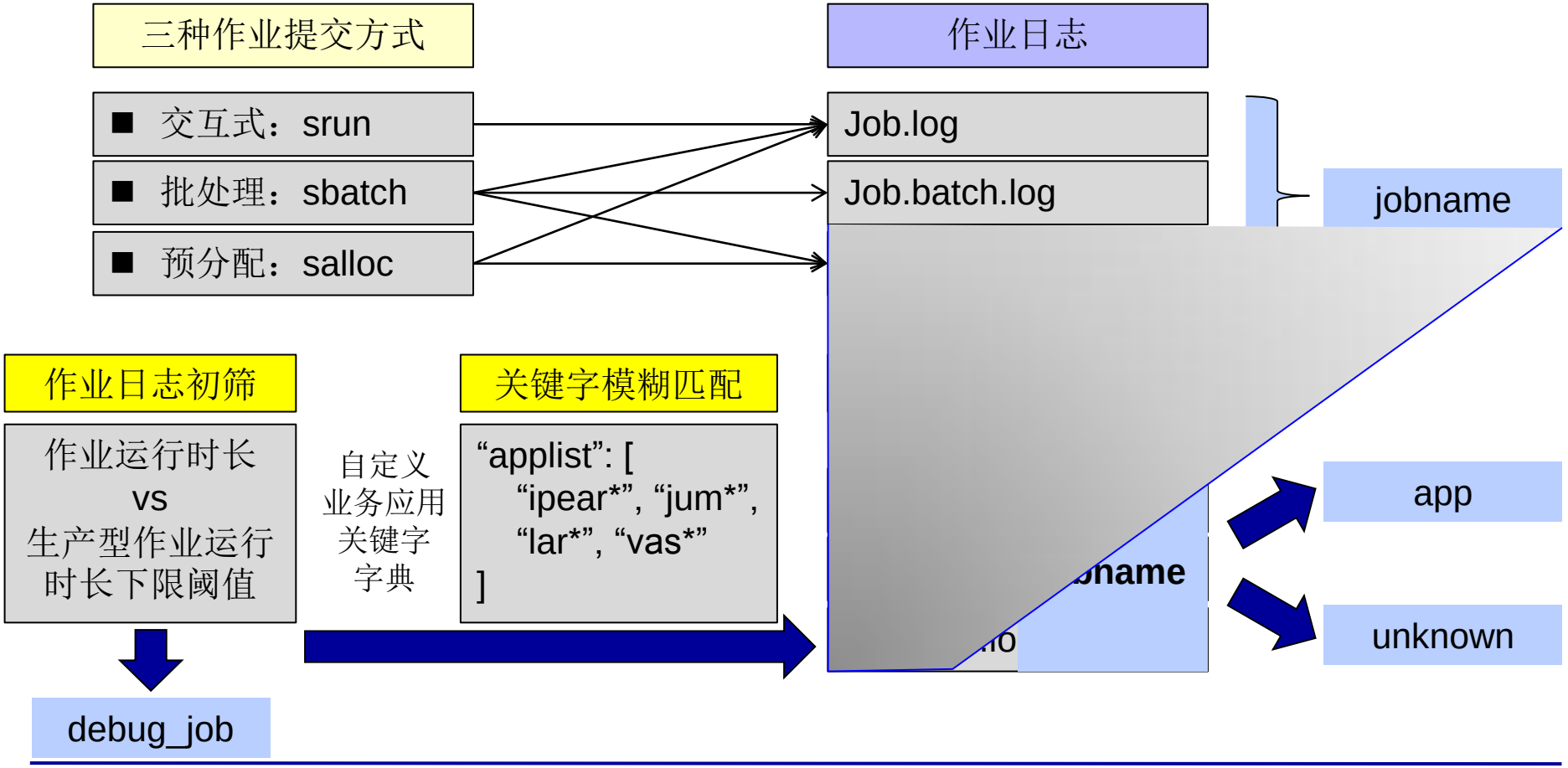
Slurm作业系统支持的日志数据记账字段（总计81 / 目前关注18）

■ **问题1：“日志-应用”关联识别问题：** 当前的作业日志系统不支持以业务应用名称索引，需要建立作业记账日志与业务应用程序的关联，识别日志对应的应用程序。

■ **问题2：作业状态校正及失效原因标定：** (1)作业的返回状态可能不是作业真正的返回状态，需要结合作业步的状态进行校正。(2)失效状态的作业没有给出导致失效的原因：用户-程序-系统

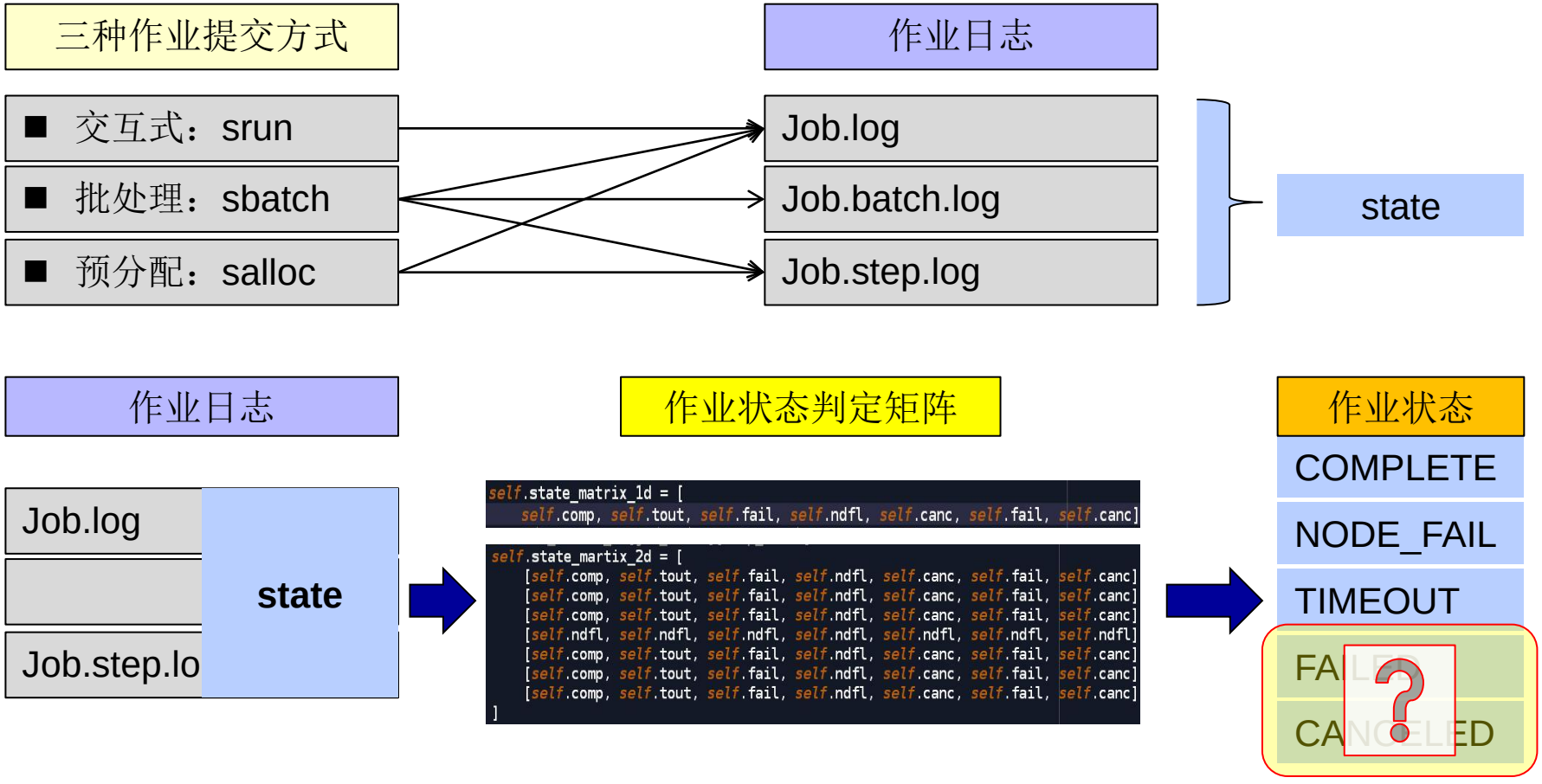
关键技术： 基于系统作业日志的“HPC系统业务特征分析工具”

■ 关键技术1：基于关键字模糊匹配的“日志-应用”关联识别技术



关键技术： 基于系统作业日志的“HPC系统业务特征分析工具”

■ 关键技术2：基于作业状态判定矩阵的作业状态校正技术



■ 数据预处理：每一类数据分10档，按档位计1-10分

$$F_u = \left| \frac{U_i - U_{\min}}{(U_{\max} - U_{\min}) \div 10} \right|$$

原始数据	29	27	25	24	17	9	7	7	…	4	3	3	3	2	2	2	1	1	1	1	1	1	1	1

MAX

$$\partial = \frac{U_{\max} - U_{min}}{10} = 2.8$$

MIN

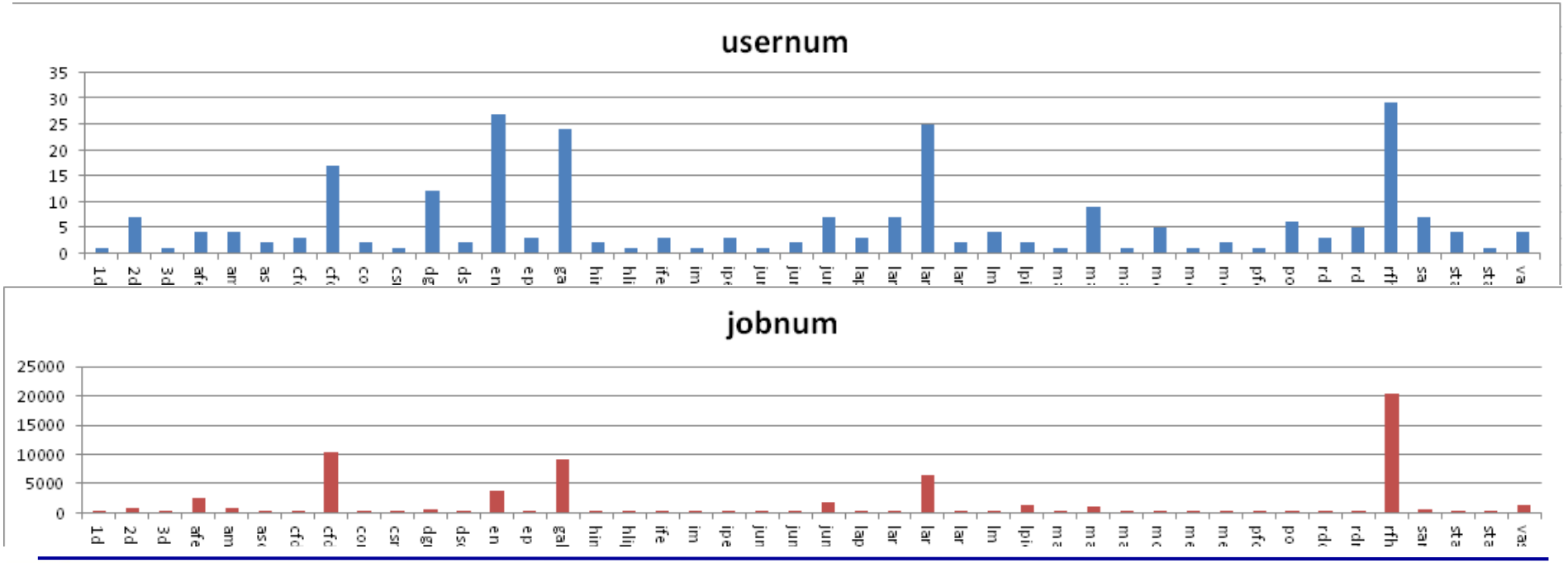
计分数据	10	10	9	9	6	3	3	3	…	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1

■ 业务应用活跃度（数据无量纲化预处理举例）

$$用户数量活跃度F_u = \left| \frac{U_i - U_{\min}}{(U_{\max} - U_{\min}) \div 10} \right|$$

$$作业数量活跃度F_j = \left| \frac{J_i - J_{\min}}{(J_{\max} - J_{\min}) \div 10} \right|$$

$$\Rightarrow \quad 业务应用活跃度F_{aa} = \sqrt{F_u \times F_j}$$



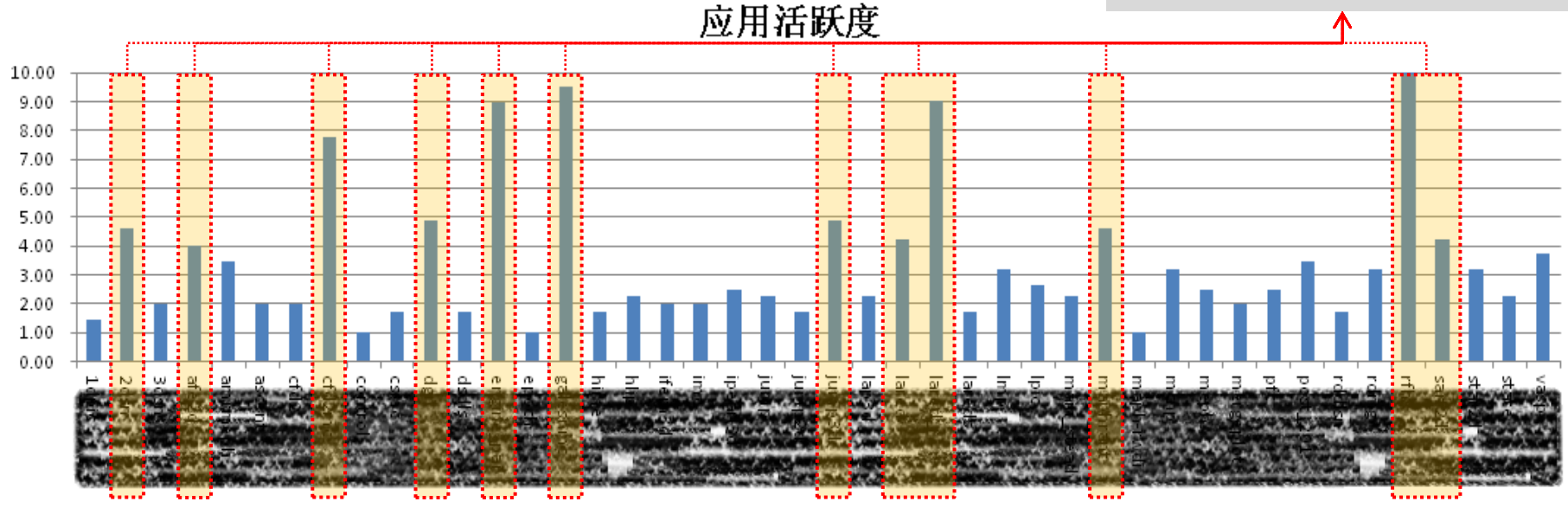
■ 业务应用活跃度（数据无量纲化预处理举例）

$$用户数量活跃度F_u = \left| \frac{U_i - U_{\min}}{(U_{\max} - U_{\min}) \div 10} \right|$$

$$作业数量活跃度F_j = \left| \frac{J_i - J_{\min}}{(J_{\max} - J_{\min}) \div 10} \right|$$

$$业务应用活跃度F_{aa} = \sqrt{F_u \times F_j}$$

活跃度 > 4的应用12个



研究成果： 系统业务特征分析

■ 系统业务应用特征分析评价矩阵

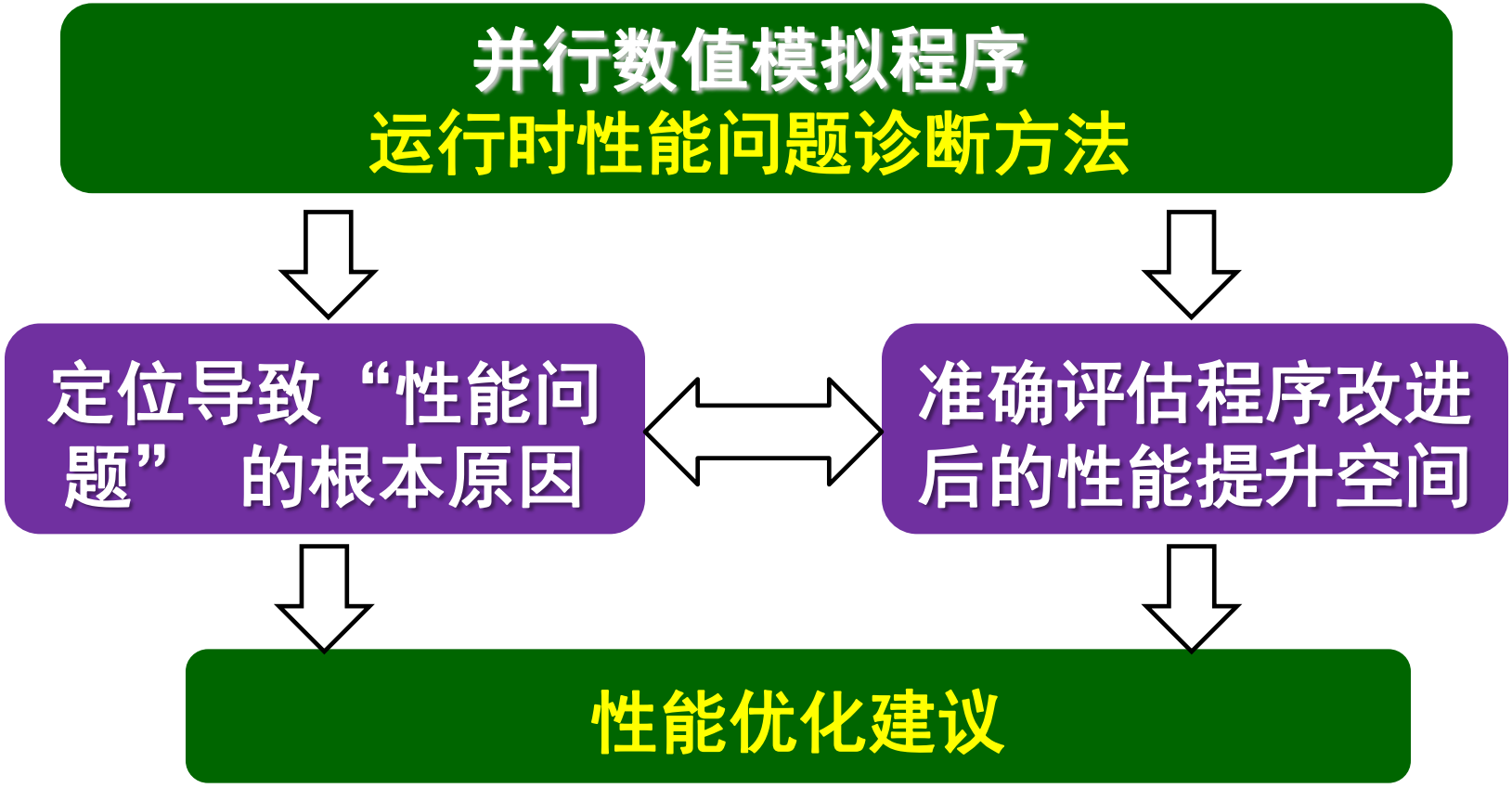
$$F_u = \left| \frac{U_i - U_{\min}}{(U_{\max} - U_{\min}) \div 10} \right|$$

所有应用的业务特征分析评价矩阵

业务程序	1d	2d	3d	aF	aa	as	cf	cf	co	cs	dg	ds	en	ep	ga	hl	hl	lf	la	lp	Ju	Ju
	[redacted]																					
用户活跃度	1	5	2	4	3	2	2	8	1	2	5	2	9	1	9	2	2	2	2	2	2	2
用户数量排行	1	3	1	2	2	1	1	6	1	1	4	1	10	1	9	1	1	1	1	1	1	1
作业数量排行	2	7	4	8	6	4	4	10	1	3	6	3	8	1	10	3	5	4	4	6	5	3
并行能力-典型规模	2	4	2	1	7	3	5	2	1	4	3	1	1	4	4	1	1	3	2	3	1	1
并行能力-最大规模	1	1	1	1	2	1	1	2	1	1	1	2	1	1	1	1	1	1	1	1	1	1
资源使用占比	1	1	1	1	1	1	1	4	1	1	1	1	1	1	4	1	1	1	1	1	1	1
运行成功率	10	1	7	10	5	4	5	4	10	9	3	8	9	8	4	4	9	4	3	4	4	5
资源产出率	10	1	5	10	4	6	3	2	4	10	2	8	5	4	2	4	7	4	3	1	2	5
时间产出率	10	1	5	10	4	4	3	2	10	10	2	8	4	6	2	4	8	2	3	4	2	5
典型规模运行波动率	1	3	2	6	3	2	8	5	5	2	5	5	10	1	7	1	5	5	8	6	5	8
典型规模运行时间长度	1	3	1	1	2	1	1	1	1	1	1	4	1	1	1	1	1	6	10	3	3	5
受系统影响强弱	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	2	2	4	1
性能评测代理指数	1	5	2	4	7	2	2	62	1	2	5	3	9	1	38	2	2	2	2	2	2	2
稳定评价指数	1	19	4	24	10	4	32	219	5	3	24	9	89	1	376	2	11	10	28	25	50	14
支撑需求指数	0	16	0	2	1	0	0	8	0	0	3	2	9	0	6	1	1	2	7	2	4	3

业务程序	Ju	la	la	la	la	la	lp	aa	aa	aa	ac	ac	ac	pr	po	rd	rd	rF	sa	st	st	va
用户活跃度	5	2	4	9	2	3	3	2	5	1	3	2	2	2	3	2	3	10	4	3	2	4
用户数量排行	3	1	3	9	1	2	1	1	3	1	2	1	1	1	2	1	2	10	3	2	1	2
作业数量排行	8	5	6	9	3	5	7	5	7	1	5	6	4	6	6	3	5	10	6	5	5	7
并行能力-典型规模	1	10	2	4	1	7	2	1	1	2	1	1	1	1	1	1	1	2	1	1	1	1
并行能力-最大规模	3	10	1	1	1	2	1	1	3	1	2	1	1	1	1	1	1	2	1	1	1	1
资源使用占比	1	1	1	3	1	1	1	1	1	1	1	1	1	1	1	1	1	10	1	1	1	1
运行成功率	6	6	2	2	4	6	1	8	3	2	5	8	1	4	9	7	7	4	9	8	7	7
资源产出率	3	6	1	1	2	4	3	3	2	3	4	10	3	5	9	2	6	2	8	5	1	7
时间产出率	4	7	1	2	1	8	2	5	3	7	3	10	2	4	9	2	5	3	9	7	1	8
典型规模运行波动率	8	3	2	6	1	2	3	10	4	1	7	3	4	6	2	4	9	7	5	5	4	4
典型规模运行时间长度	3	3	1	2	1	1	7	1	1	7	2	4	1	3	1	1	1	3	2	1	1	4
受系统影响强弱	2	2	1	1	8	3	1	1	1	1	2	3	1	1	1	1	1	1	1	1	1	2
性能评测代理指数	15	22	4	27	2	6	3	2	14	1	6	2	2	2	3	2	3	200	4	3	2	4
稳定评价指数	68	13	12	229	14	20	8	22	26	1	38	23	8	15	7	7	28	990	21	16	9	30
支撑需求指数	12	0	5	20	1	0	25	1	7	2	4	3	4	5	1	0	1	38	4	1	1	8

关注点三：数值模拟应用的“性能问题诊断”



诊断方法（1）：MPI通信等待问题的诊断

MPI并行开销诊断模型[2]

“MPI并行开销”诊断模型：

$$\cos t(q,c)=\frac{\sigma_t(q,c)}{\sum_{f\in C}\sigma_t(q,f)}\cdot w(p,i)$$

进程q上函数c的超时时间：

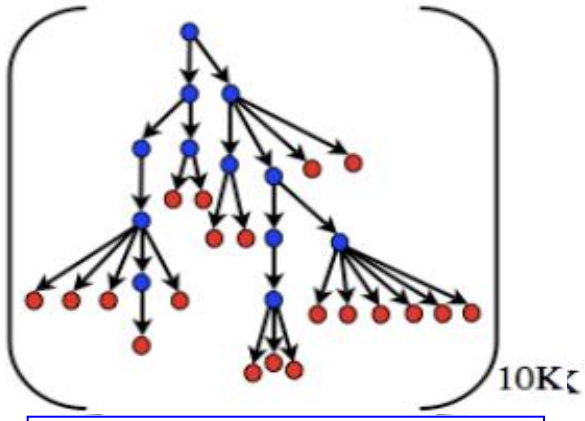
$$\sigma_t(q,c)=\begin{cases} t(q,c)-t(p,c) & if \quad t(q,c)>t(p,c) \\ 0 & otherwise \end{cases}$$

进程p上函数c的执行时间：

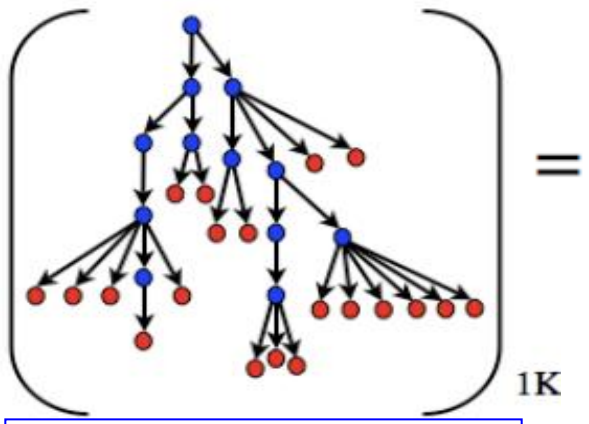
$$t(p,c)=Exit(p,c)-Enter(p,c)-w(p,c)$$

诊断方法一：(1)可确定导致“MPI并行开销”的关键函数;
(2)可预测“MPI并行开销”消除后可获得的最大性能提升空间。

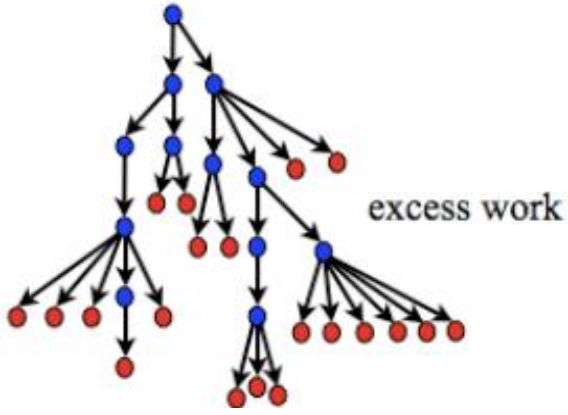
诊断方法（2）： 可扩展性能问题的诊断方法



万核并行程序callpath包含信息：
（1）函数调用关系（2）函数执行时间$T_{万核}$



千核并行程序callpath包含信息：
（1）函数调用关系（2）函数执行时间$T_{千核}$

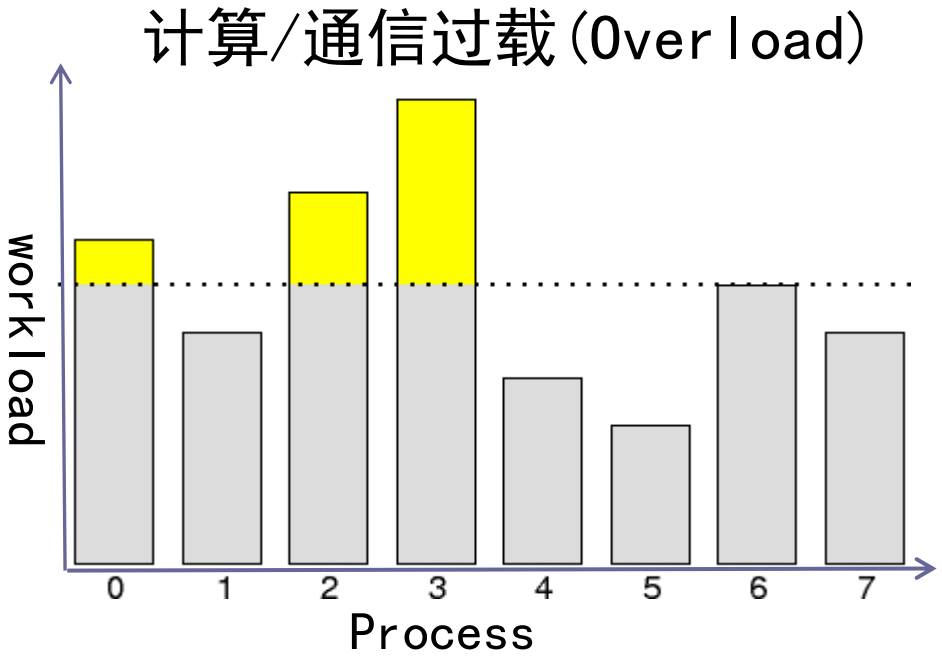


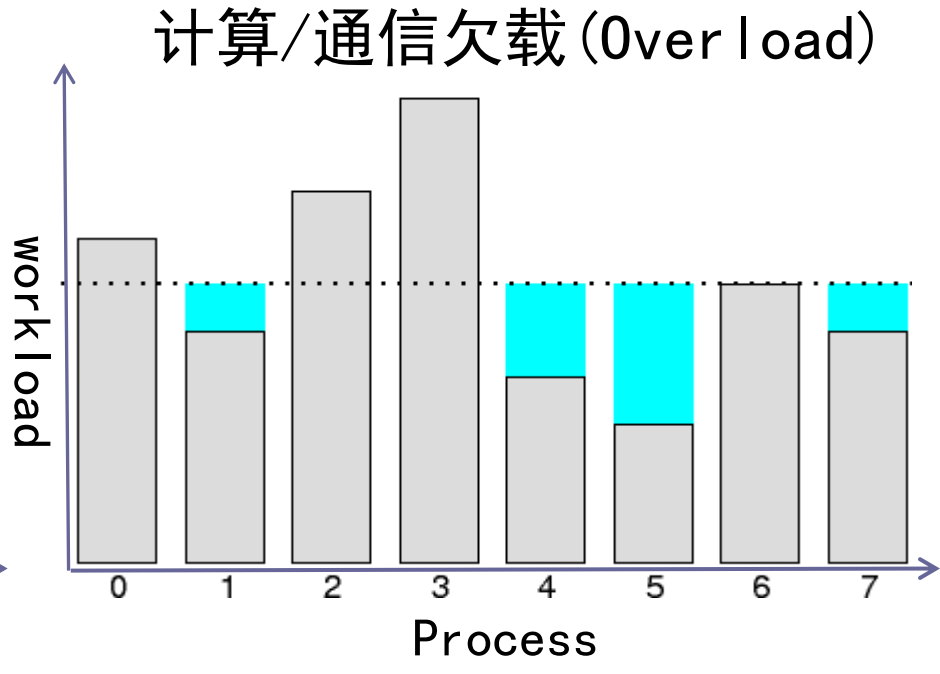
并行程序callpath包含信息：
（1）函数调用关系（2）函数扩展损失

每个函数的可扩展损失比例[1]

$$\left(\left(10*T_{f,10000}-T_{f,1000}\right)\Big/\left(10*\mathrm{T}_{\mathrm{main},10000}\right)\right)*100\%$$

诊断方法（3）：计算/通信负载平衡问题的诊断方法



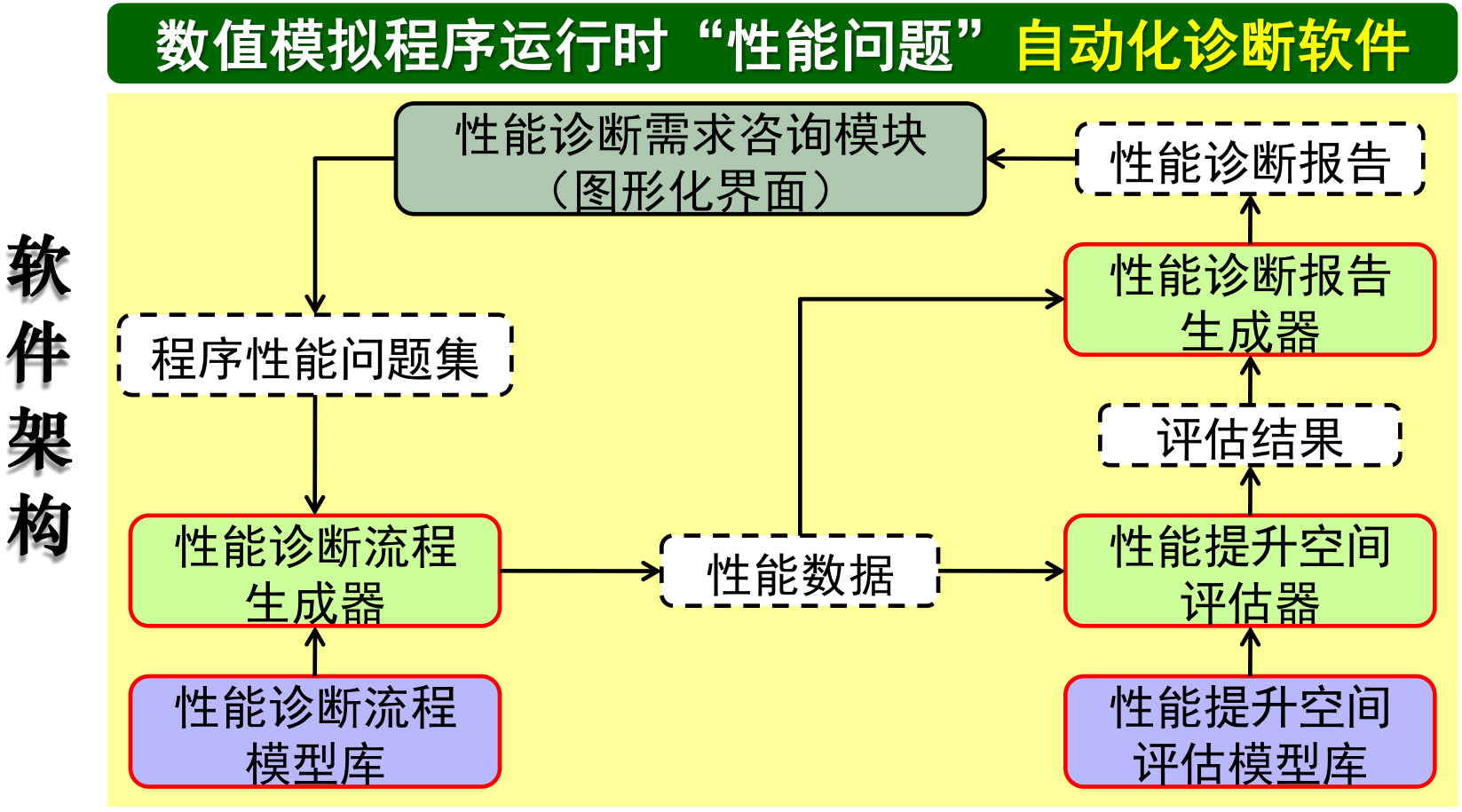


计算/通信负载均衡性度量指标:

$$I_t = \max_p \left(w_p - \frac{1}{P} \sum_{i=1}^{N} w_i \right)$$

注：w_i指某个函数在进程i上的计算量（如浮点数）或通信量（如通信字节数）

诊断软件架构


数值模拟程序运行时“性能问题”自动化诊断软件
软件架构
性能诊断需求咨询模块（图形化界面）
性能诊断报告
性能诊断报告生成器
程序性能问题集
评估结果
性能诊断流程生成器
性能数据
性能提升空间评估器
性能诊断流程模型库
性能提升空间评估模型库

展望

- 高性能计算机运维中遇到的问题多，涉及面宽，工作琐碎，加强实践中知识沉淀积累是我们下一步的方向

- 加强与国内高校和院所的合作，以实际应用需求为牵引，做高水平的研究，服务于高性能计算机运维

Thanks !