航天一院研发中心
暑期实践自我介绍

孙 骁

2024年5月23日

目录

个人简介

- 姓名：孙骁
- 性别：男
- 民族：汉族
- 政治面貌：共青团员
- 出生年月：2000.08
- 籍贯：黑龙江省大庆市
- 手机号码：18145939706
- 邮箱：demerzelsun@163.com

2018.08-2022.06 哈尔滨工业大学 计算机科学与技术 本科

- 平均学分绩91.3，专业排名25/146，CET-6: 542，推荐免试攻读研究生
- 哈尔滨工业大学百佳毕业论文

2022.08-2024.06 哈尔滨工业大学 网络空间安全 硕士研究生

- 学位课成绩 85.10，综合排名32/266，特等奖学金
- 主干课程：高级逆向分析、云安全技术、高级数据库系统、社交网络分析、深度学习技术、高级算法设计与分析、矩阵分析
- 研究方向：云原生环境下任务混合部署和调度、资源分配方法
- 导师：张伟哲，教授，博士生导师，国家高层次人才计划、CCF杰出会员、IEEE和ACM高级会员。哈尔滨工业大学计算学部网络空间安全学院副院长、鹏城实验室新型网络部副主任、国家自然科学基金信息学部评审会专家

交流访学经历

2021.02-2021.07　哈工大（深圳）

- 首次校区间交流交换，计算学部共**9**人前往
- 传播哈工大精神，促进校区间交流

奖项荣誉

2019.12 校优秀学生
2019.05 校优秀团员
2021.05 校优秀团员
2020.12 校优秀学生干部
2021.12 校优秀学生干部
2023.05 校优秀团干部标兵(前1‰)
2023.10 校优秀学生标兵(前3‰)

2018年秋季学期一等人民奖学金
2019年春季学期三等人民奖学金
2019年秋季学期二等人民奖学金
2019.07 大一年度项目计划一等奖
2022.06 哈尔滨工业大学百佳毕业论文
2023.09 哈尔滨工业大学一等研究生奖学金
2024.09 哈尔滨工业大学特等研究生奖学金

2021.02 哈尔滨工业大学（深圳）交流访学
2020.10 全国大学生数学建模竞赛黑龙江省二等奖
2023.03 杭州蚂蚁科技集团股份有限公司校园大使
2024.03 杭州蚂蚁科技集团股份有限公司校园大使

获校级以上荣誉称号

（含奖学金）10余项

目录

云原生技术——虚拟化时代的新型软件架构

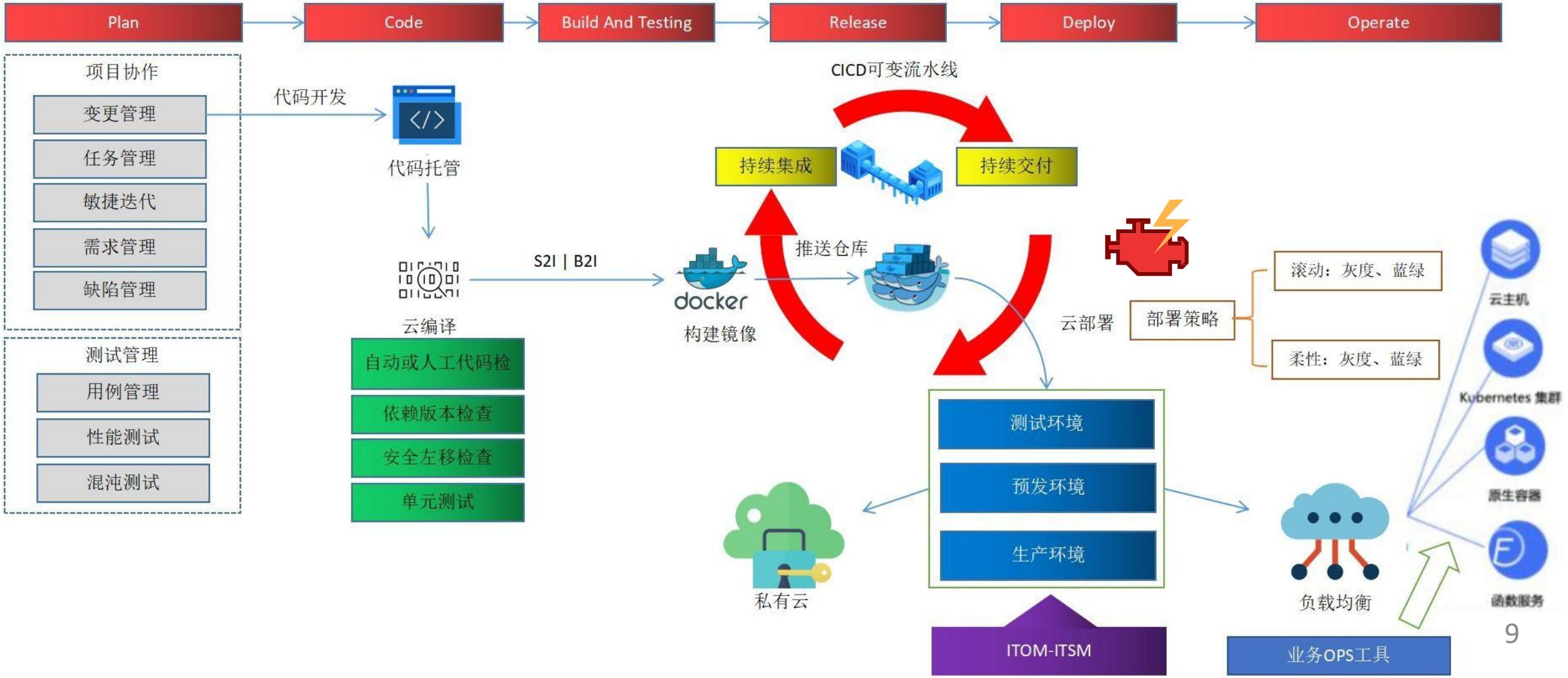
云原生技术是一种通过云计算服务交付应用程序的方法。

- **云原生技术（Cloud Native Technology）**
  - 包括容器化、微服务架构、DevOps、持续集成/持续部署(CI/CD)等关键技术
- **云原生技术关键组件：**
  - 容器：例如 Docker，用于封装应用及其依赖。
  - 编排工具：例如 Kubernetes，用于自动化部署、扩展和管理容器化应用。
  - 服务网格：例如 Istio，用于管理微服务间的通信。

2020年，国家发展改革委、中央网信办印发《关于推进“上云用数赋智”行动 培育新经济发展实施方案》，强调加快数字产业化和产业数字化，培育新经济发展，扎实推进国家数字经济创新发展试验区建设，构建新动能主导经济发展的新格局，助力构建现代化产业体系，实现经济高质量发展，

标准化能力-工程能效-DevOps-研发平台

- 提高协作效率和质量(关注的不是部署效率、而是协作研发效率)
- 减少变更带来的风险：制品一致性、配置一致性、环境一致性
- 打通研发与运维：谁定义谁负责，平台提供业务OPS工具，减少研发和运维视角割裂带来的高成本
- 可以作为研发人员的唯一工作台，将底层技术收敛到统一平台上，减少研发管理成本



标准化能力-分布式系统核心-容器化集群管理

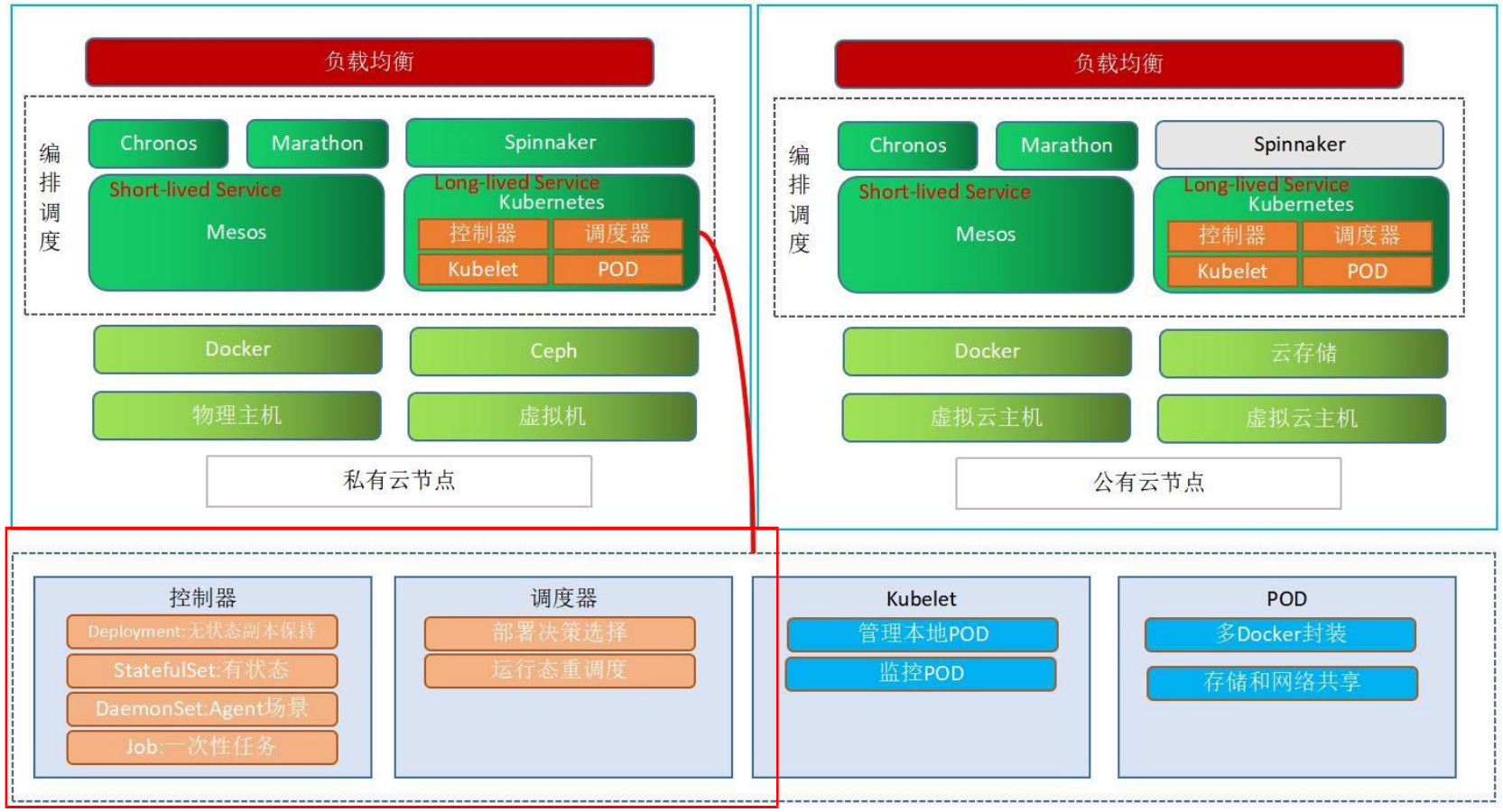
服务编排基本原理：

- 以度量为基础，以NodeSelector算法来决定在哪儿部署容器服务
- 运行时以期望与实际的差别进行动态调整到期望的状态

高效稳定应用资源供给

向上提供抽象化IT运营视角

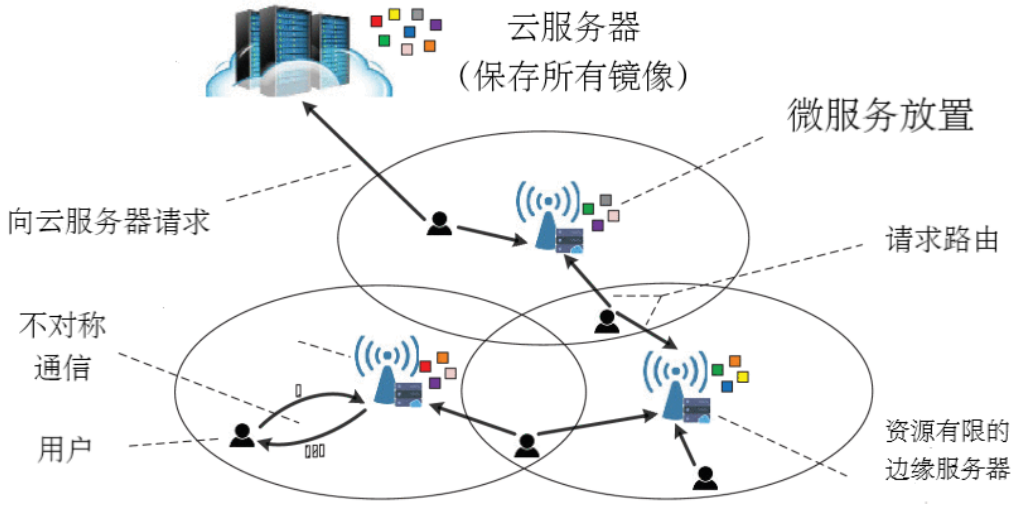
架构

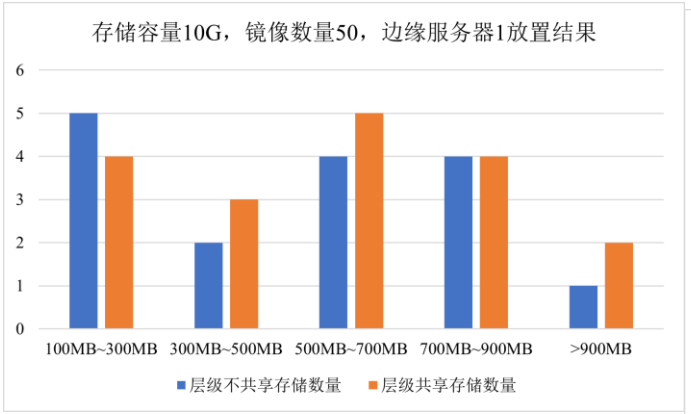


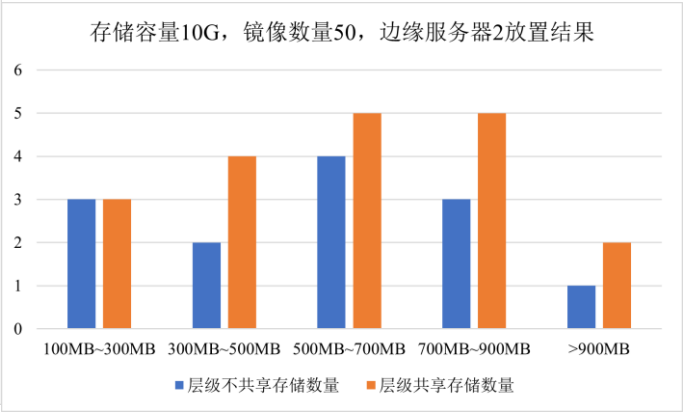
云原生底座=控制器+调度器的组合+Docker=根据环境的变化而动+基于封装一致性的大规模分发

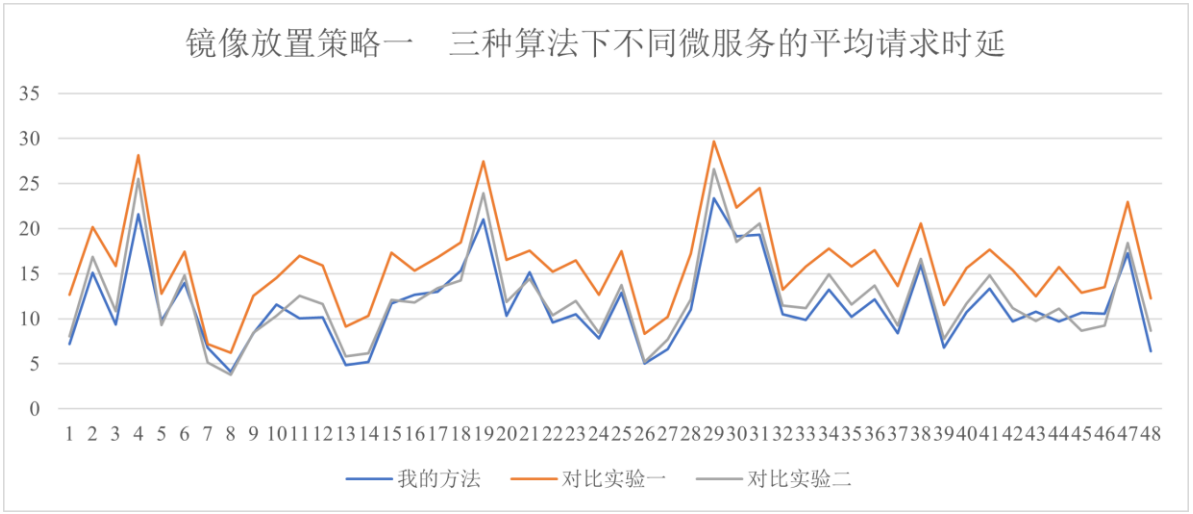
基于随机近似算法的服务缓存和任务请求路由方法

边缘计算 KubeEdge环境下实现请求路由平均时间最小化。









- 在存储容量相同时，层级共享机制下能存放更多的镜像；
- 我们的算法和基于贪心的任务路由算法会优于基于镜像拉取的算法。

Minimizing Service Latency through Image-based Microservice Caching and Randomized Request Routing in Mobile Edge Computing，IOTJ (CCF-C) 在投一作

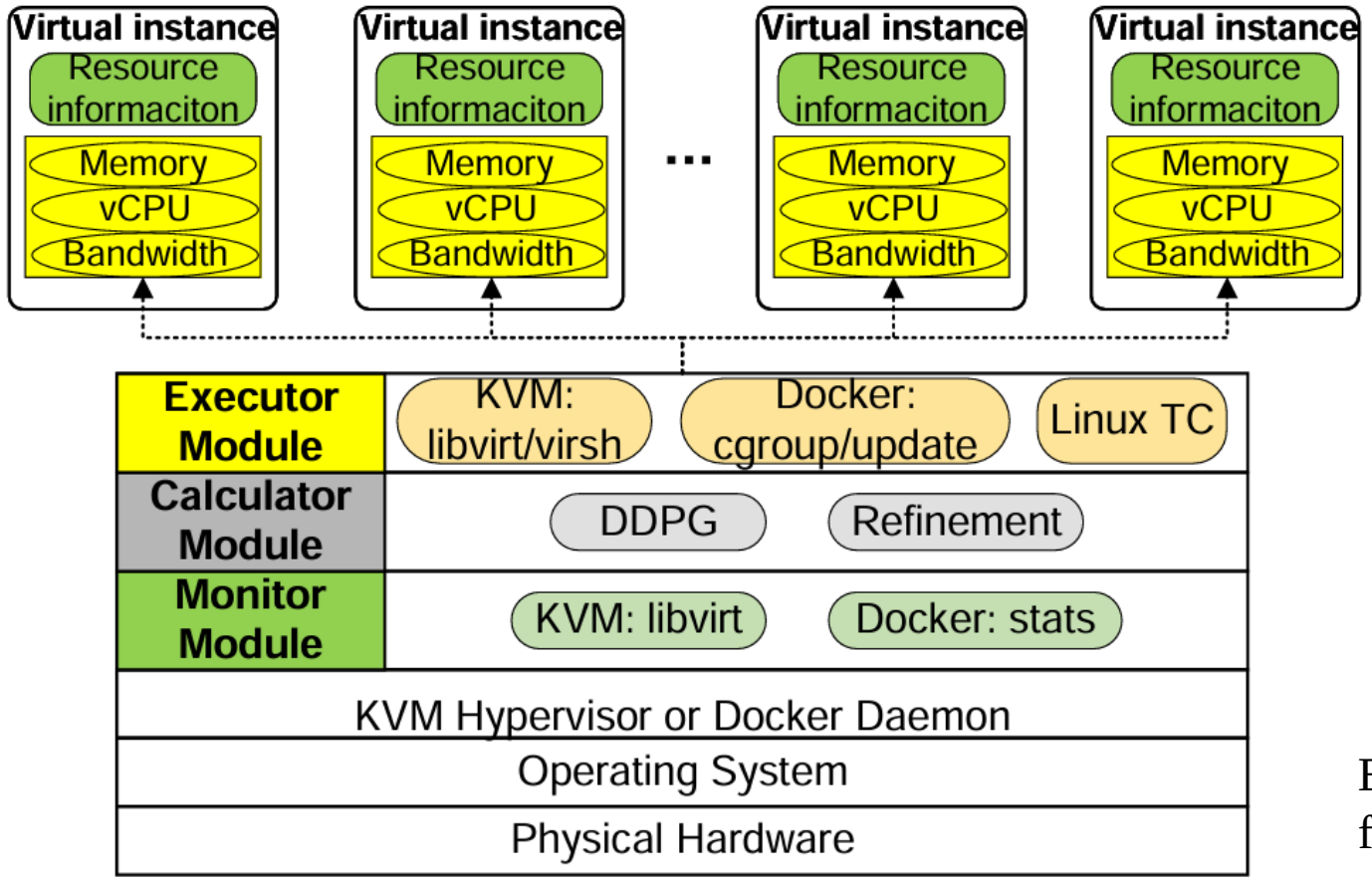
哈尔滨工业大学2022届3824名本科生

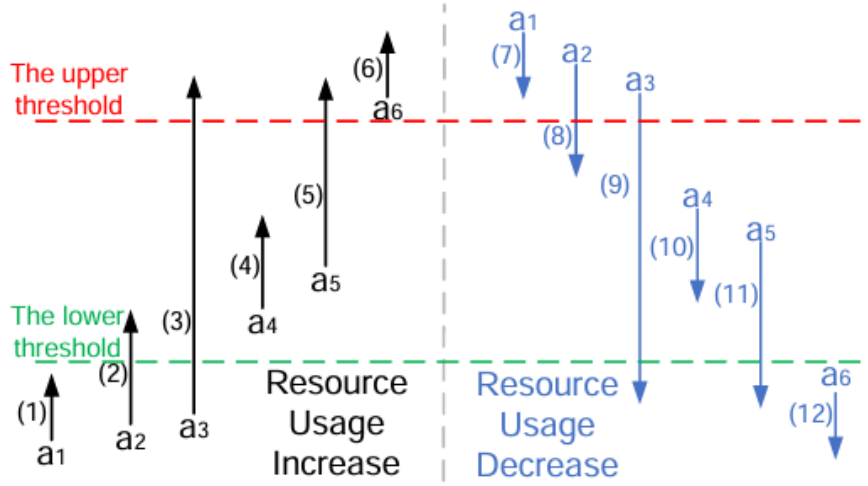
经研究中心、学部、学校层层选拔

荣获百佳毕业论文

基于深度强化学习A2C算法的虚拟机或容器的参数自动调优方法

通过获取kvm虚拟机或Docker容器的状态，自适应动态调节虚拟实例的内存、vCPU核数、带宽大小，
提高系统资源利用率和虚拟实例运行性能。





针对不同情况的资源变化需求做出自动化决策

EVRM: Elastic Virtual Resource Management Framework for Cloud Virtual Instances，FGCS (CCF-A) 在投三作

云原生在离线DNN任务混合部署方法

深圳市高等学校稳定支持计划项目， GXWD20231129102636001

本人研究课题

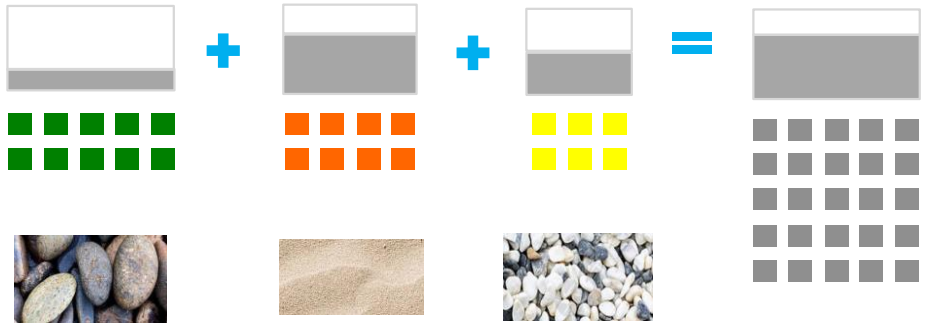
混部定义：

1，基于节点粒度的定义： **多个容器部署在一个节点上**，其中：

多个容器： 包括多个在线容器，多个离线容器，多个在离线容器三种形态，不仅仅是在离线容器；

一个节点：能够运行容器的最小单位，包括物理机、ECS等；

2，基于集群粒度的定义： 多种应用在一个集群内自动部署，通过预测分析应用特性，业务间错峰填谷；



混部的核心技术：

1，节点粒度：

a、容器隔离技术，包括依赖内核的namespace、RunC等；

b、单机调度技术：包括单机的负载感知，容器策略和阈值设置，容器弹性控制、cpushare；

c、中心调度技术：支持单机粒度的负载反馈、调度策略、调度性能等；

d、**资源的差异化SLO技术**： 包括差异化SLO 设定，优先级设定，基于差异化SLO 的单机和中心调度配合；

2，集群粒度：基于节点粒度之上实现；

a，中心调度技术：高性能调度、资源视图、多负载的调度协同、GPU拓扑感知等；

b，单机调度技术：任务的高频起停控制，单机多环境、硬件适配、cpu归一化等；

c，**k8s 生态框架优化和配套能力建设**，包括集群规模突破、稳定性改进、运维配套能力、界面和接口能力、业务对接等；

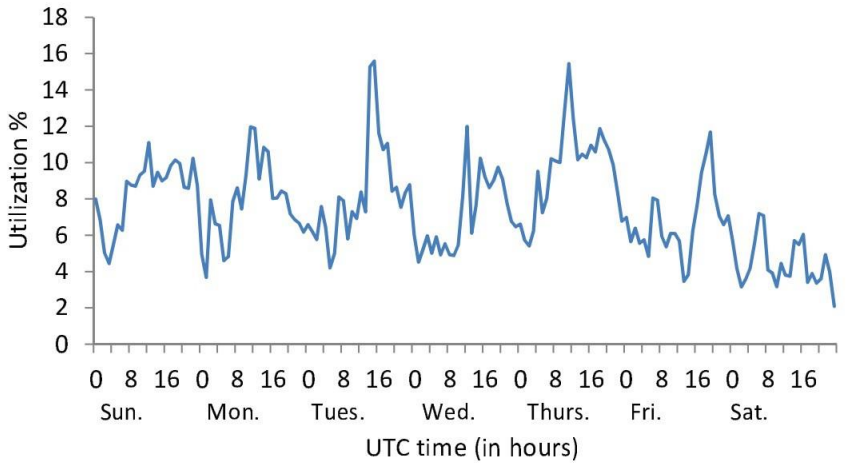
混部的目标和方案

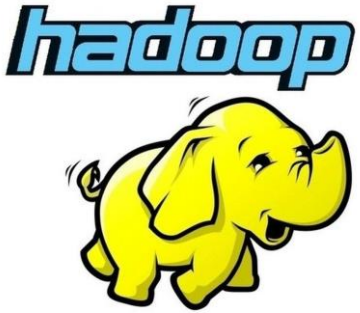
混部的目标：

- 对数据中心**资源利用效率**的不懈追求
- 埃森哲报告显示 2011 年公有云数据中心的机器利用率平均不到 10%，意味着集群的资源成本极高
- 另一方面大数据技术的发展迅速，计算作业对资源的需求越来越大。

如何混部：

- **大数据云原生**上云已成必然趋势
  - 据 Pepperdata 2021年12月调查报告，大数据平台开始向云原生技术迁移。超过77%的受访者预计到2021年底，其50%的大数据应用将迁移到 Kubernetes 平台。
- Google 公开数据显示其数据中心利用率 30%~40% @2011，50%～60% @2019
- 在线服务类型应用 + 批处理类型任务





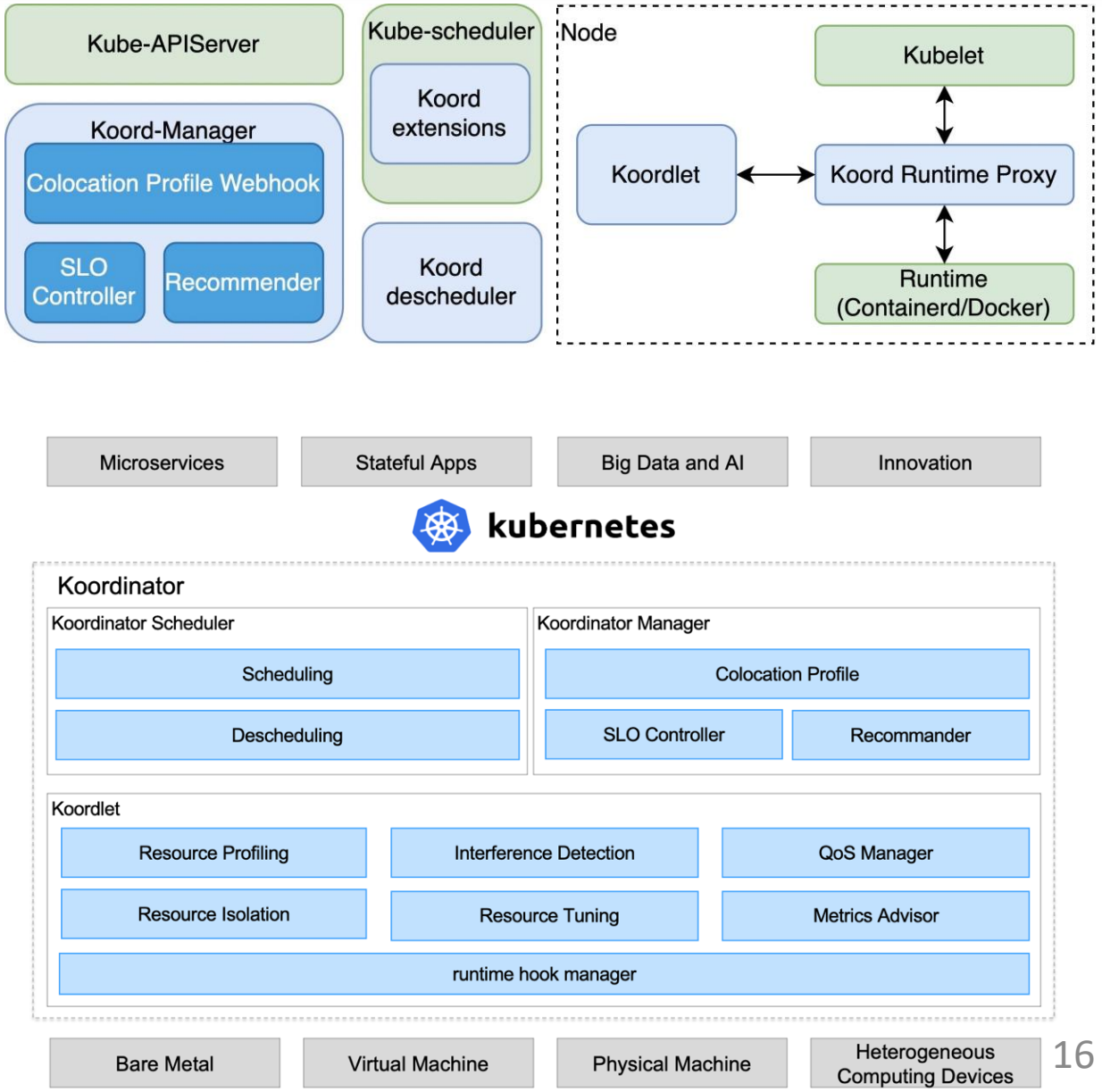
混部系统架构

中心管控

- Koord-Manager
  - SLO-Controller
    - 提供**资源超卖**、混部SLO 管理、精细化调度增强等核心管控能力
  - Recommender：围绕**资源画像**为应用提供相关的弹性能力
  - Colocation Profile Webhook
    - 为应用提供**一键接入**的能力，自动注入相关优先级、QoS 配置
- Koord extensions for Scheduler：面向**混部场景的调度能力**增强
- Koord descheduler：提供灵活可扩展的**重调度**机制

单机资源管理

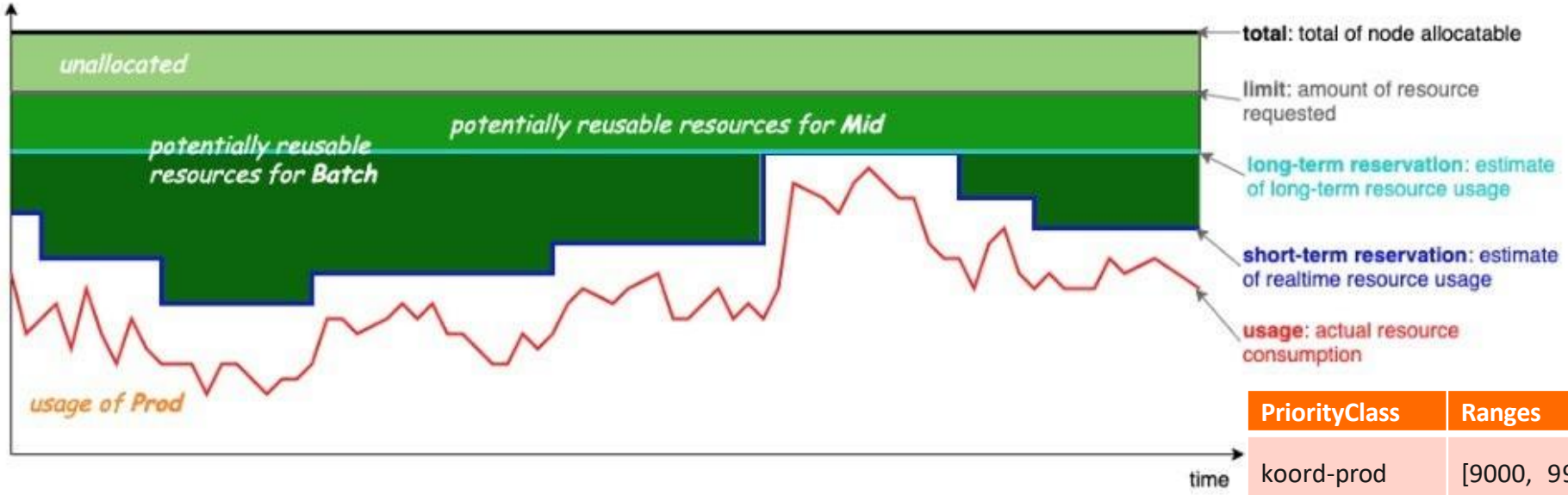
- Koord Runtime Proxy
  - 作为Kubelet 和Runtime 之间的**代理**，满足不同场景的资源管理需求
  - 提供插件化的**注册框架**，提供相关资源参数的注入机制
- Koordlet
  - 细粒度的**容器指标采集**，包括资源消耗、容器进程性能等
  - 面向不同QoS 等级Pod 的**干扰检测和调节**策略能力
  - 提供一系列的 **Runtime Proxy 插件**，支持精细化的QoS 参数注入



任务优先级划分

资源优先级

- 分为**Product、Mid、Batch、Free** 四个等级，Pod 需要指定申请的资源优先级，scheduler 会基于总量和分配量做调度。
- 各优先级的资源总量会受高优先级资源的 request 和usage 影响，例如**已申请但未使用**的Product 资源会以 Batch 优先级再次分配。
- 资源**优先级策略的激进/保守**，决定了集群资源的超卖容量，与资源稳定性的高低成反相关
- Koordinator 将各优先级资源以标准的 **extend-resource** 形式更新在 Node 信息中



应用优先级

- 以K8s **标准的 PriorityClass** 进行了扩展定义，代表 Pod 申请资源的优先级
- 同时提供了 Pod 级别的**子优先级**（sub-priority），用于调度器层面的精细化控制（排队，抢占等）
- Pod 不能跨优先级申请资源（例如不允许 Batch Pod 申请Prod 资源）
- 多优先级资源超卖情况下，当单机资源紧张时，低优先级 Pod 会被压制或驱逐

PriorityClass	Ranges	Description
koord-prod	[9000，9999]	典型的在在线应用，特征是对响应时间较为敏感
koord-mid	[7000，7099]	流式计算，近线计算，AI类型任务
koord-batch	[5000，5999]	离线批处理任务，例如大数据作业、OLAP 类查询
koord-free	[3000，3999]	低优先级的离线任务，常用于研发人员的测试场景

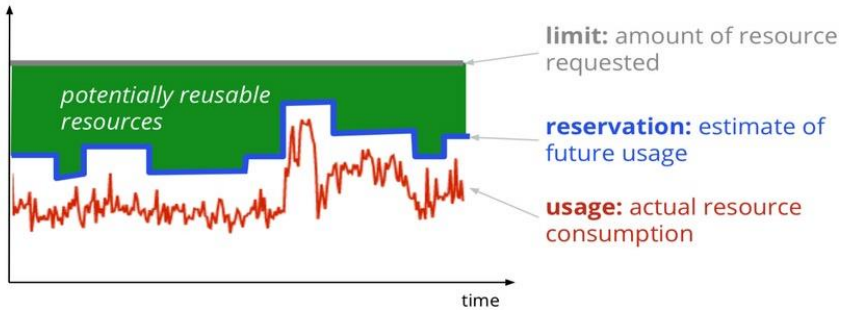
服务质量（Quality of Service）

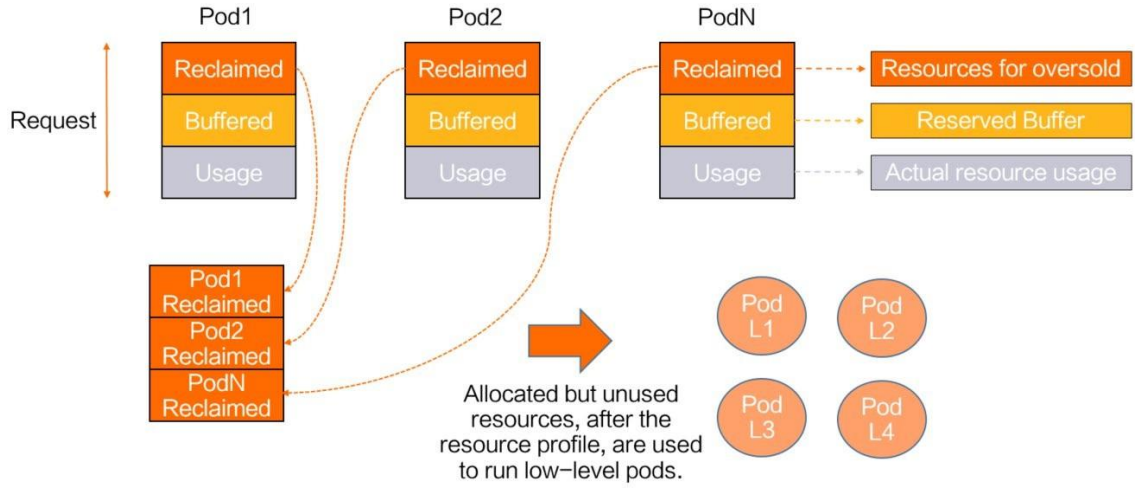
- 整体分为 System（系统级服务），Latency Sensitive（延迟敏感性的在线服务），Best Effort（资源消耗型的离线应用）三类
- 根据应用性能敏感程度的差异，Latency Sensitive 又细分为 LSE，LSR 和LS
- 影响 Pod 在单机的**运行质量**，主要表现为使用的隔离参数不同，单机资源紧张时会优先满足高等级 QoS 的需求
- 我们将 QoS 模型在 Pod Annotation 级别进行了扩展定义

QoS	Description	Kubernetes QoS
System	系统级服务	--
LS (Latency Sensitive)	延迟敏感型应用，与其他 LS 类型应用共享 CPU 资源，弹性较好，对于突发的资源需求可以比较灵活的得到满足。	Guaranteed/Burstable
LSE (Latency Sensitive Exclusive)	敏感程度极高，要求独占 CPU 资源，不与其他任何 Pod 共享CPU 核心。	Guaranteed
LSR (Latency Sensitive Reserved)	敏感程度介于 LS 和LSE 之间，轻易不与其他任何 Pod 共享CPU 核心。 与LS 类型相比，LSR 对CPU 的确定性要求更高；与 LSE 类型相比，在自身资源空闲的前提下，LSR 允许BE 类型Pod 临时使用。	Guaranteed
BE (Best Effort)	敏感程度较低的资源消耗型 Pod，在于以上类型应用混部时可以接受压制甚至驱逐，以避免对敏感型应用产生干扰。	BestEffort

资源超发 - Resource Overcommitment

Efficiency Resource reclamation



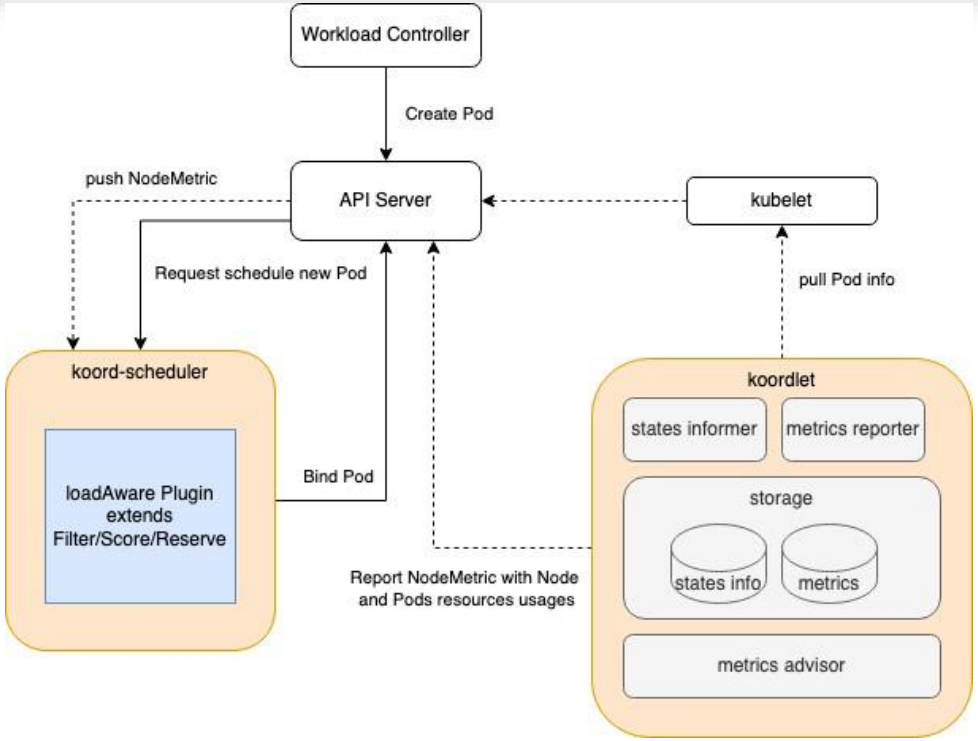


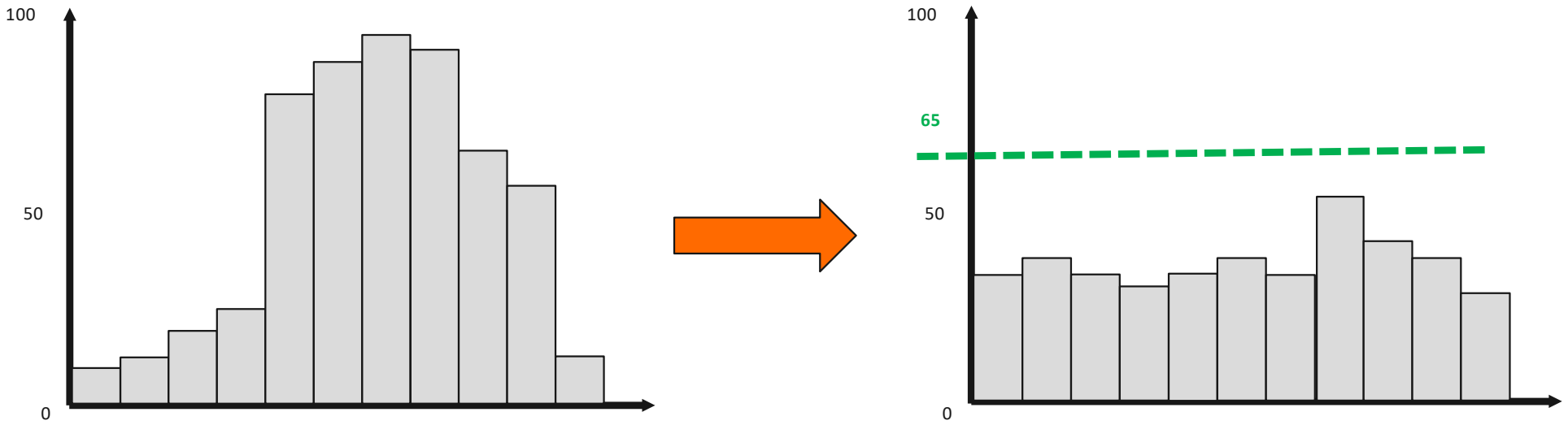

```
# node info
allocatable:
   bach-cpu: 50k #  milli-core 、
   bach-memory:  50Gi
```

```
# pod info
annotations:
    resource-limit: {cpu:"5k"}
-   resources:
     requests
         bach-cpu: 5k #  milli-core
         bach-memory:  5Gi
```

负载均衡调度 - Load-Aware Scheduling

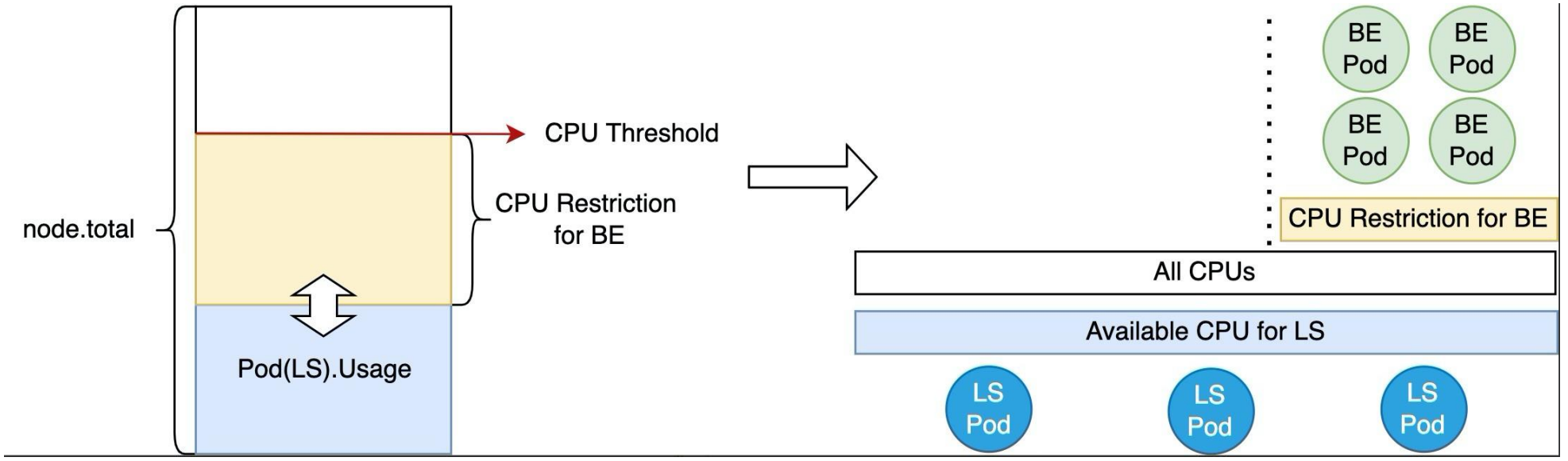
- **资源利用率均衡度影响运行时质量**
  - 负载热点和负载不均衡，影响工作负载的整体运行效果
  - 在线应用和离线任务之间冲突严重
- **可配置的调度插件控制集群利用率**
  - 自定义多维度资源和权重
  - 过滤负载过高的节点，选择利用率更低的节点
  - 基于资源预估和窗口机制，应对并发调度和冷节点场景





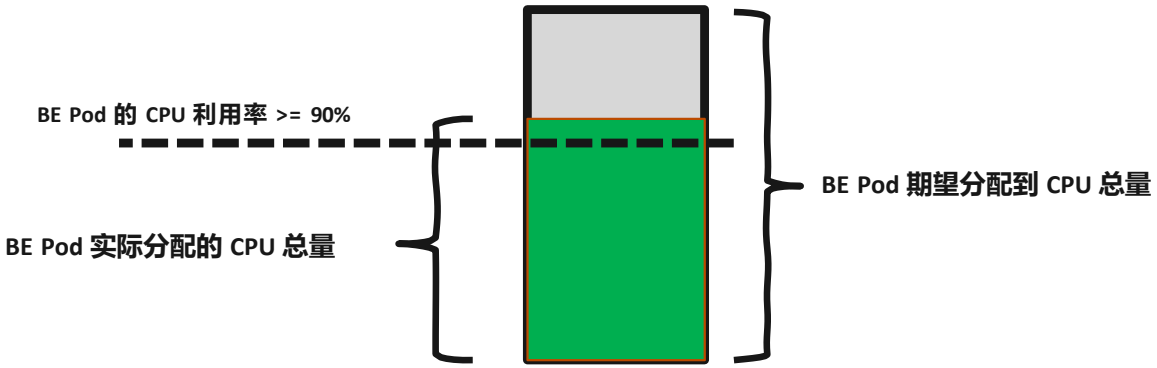
单机 QoS 增强 – CPU Suppress

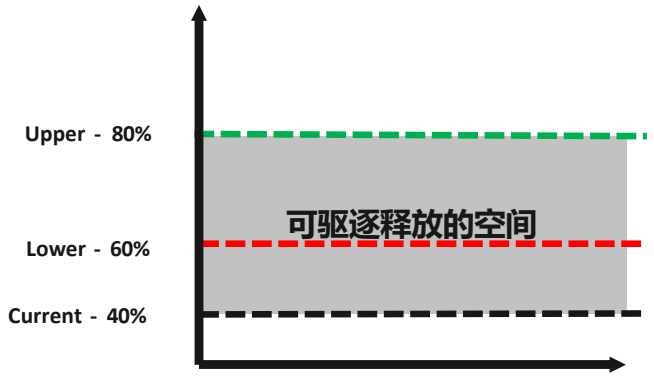
- BE Pod 的可用CPU 取决于LS Pod 的负载
- LS Pod 的负载升高时Y， BE Pod 的可用CPU 总量下降t，反之则增加



单机 QoS 增强 – 基于资源满足度的驱逐

- **CPU Suppress 有可能会频繁的压制 BE Pod**
  - BE Pod 的性能得不到满足
  - BE Pod 在被压制时持有内核全局锁等特殊资源，频繁压制可能导致优先级反转，影响 LS Pod 的性能
- **CPU 满足度**：实际分配的 CPU 总量与期望分配的CPU 总量的比值

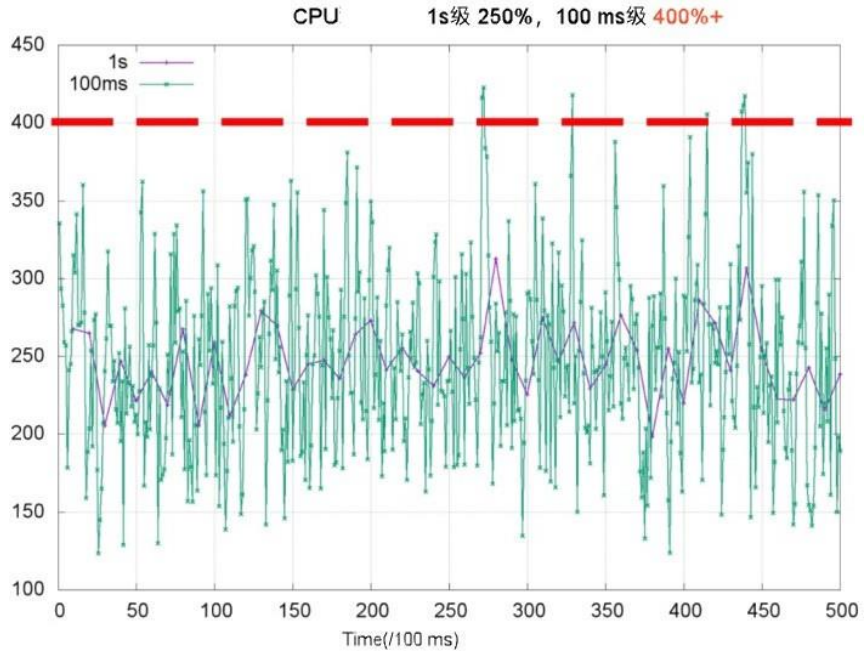


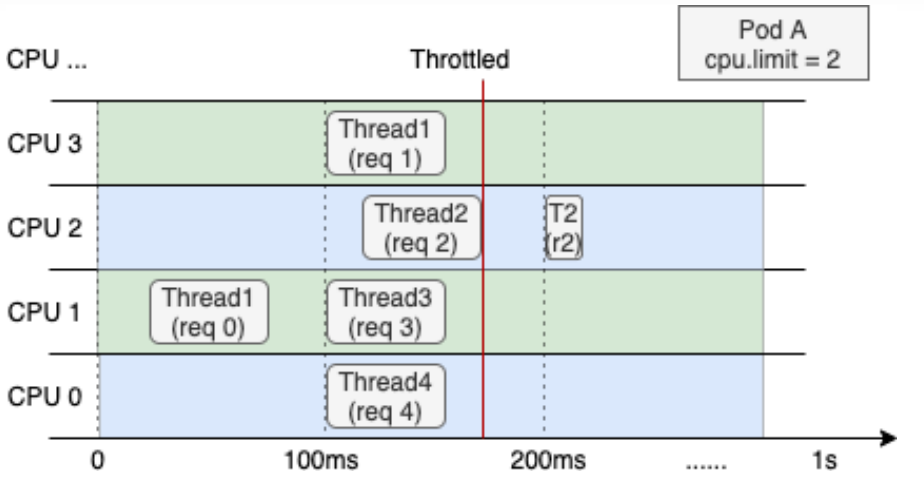


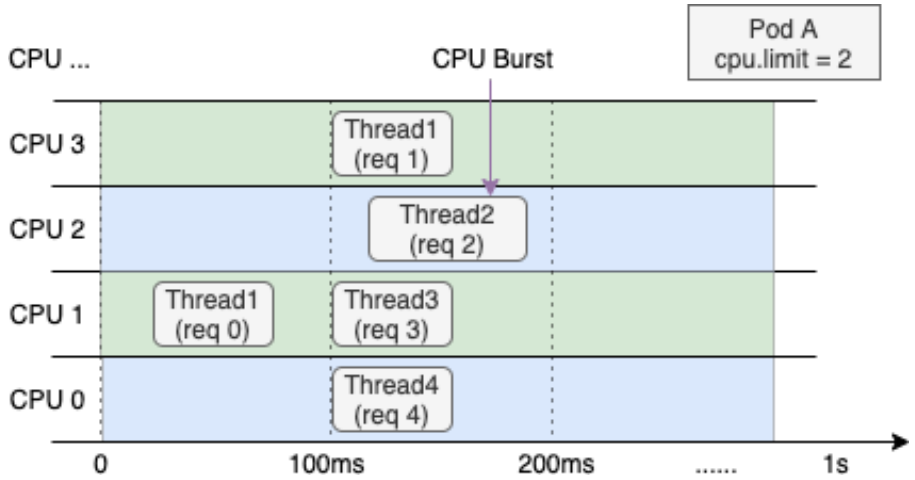
- **当 BE Pod 组的 CPU 得不到满足且 CPU 利用率超过阈值时驱逐**
  - 把节点上所有 BE Pod 作为一组统一决策
  - 同时参考 BE Pod 组最近一段时间的 CPU 利用率（Avg/Current）
  - 优先级越低的优先被驱逐，同优先级的利用率越高的优先被驱逐
  - 驱逐后会进入冷却期，防止过度驱逐

单机 QoS 增强 – CPU Burst

- **细粒度观察 CPU 利用率并不稳定**
  - CPU 突发和压制是常态
- **CPU Burst**
  - 空余的CPU 资源将会被积累
  - 需要时使用这部分被积累的 CPU 资源







单机 QoS 增强 – Group Identity

- **Group Identity 特性是一种以 cgroup 组为单位实现调度特殊优先级的手段**
  - LS Pod 的任务唤醒延迟最小化
  - BE Pod 的任务唤醒不会对 LS Pod 造成性能影响
  - BE Pod 的任务不会通过 SMT 调度器共享物理核而对 LS Pod 造成性能影响

- **Group Identity 通过 cpu cgroup 接口 cpu.bvt_warp_ns 配置**

QoS	值	含义	说明
-	0	该 cgroup 中任务为普通系统任务	默认优先级
BE	-1	该 cgroup 中的任务为离线任务	优先级最低
-	1	该 cgroup 中的任务为在线任务	优先级高
LSR/LS	2	该 cgroup 中的任务为在线任务，并具有 SMT 驱逐能力	优先级最高

请各位专家老师批评指正