操作系统调度器的演进

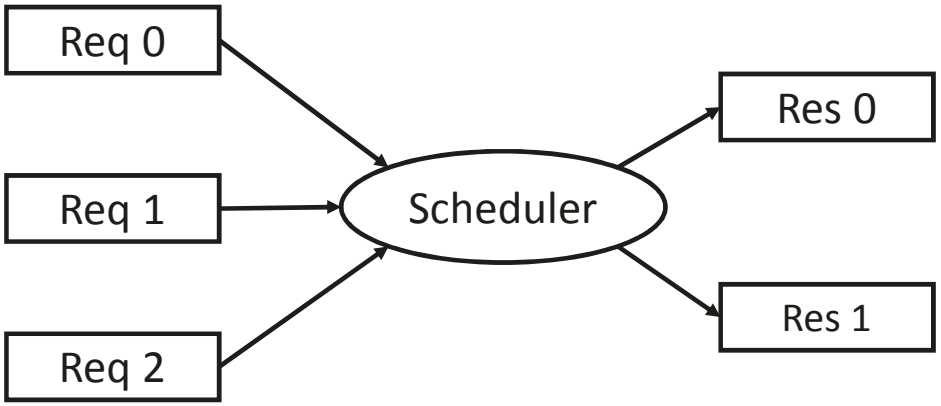
华为OS内核实验室
王飞

Security Level:

HUAWEI

目录

调度的本质：Req > Res



Task Scheduler	Task -> scheduler -> CPU
K8s:	Pod -> scheduler -> Node
Hypervisor:	Guest OS -> scheduler -> Host OS
Memory Manage:	VA –> scheduler -> PA
Golang：	G + P -> scheduler -> M
Io：	Request -> scheduler -> io

Network, nginx, spark, DPDK, etc.

调度需求 — 演进

大型机

高吞吐

个人PC

低时延

移动设备

低功耗

HUAWEI

调度需求 — 矛盾性

- 调度开销导致高吞吐和低时延存在矛盾
  - 直接影响：中断处理、调度器选取任务、硬件上下文保存恢复
  - 间接影响：刷TLB、程序局部性、CPU pipeline 刷新
- 高性能与低功耗之间天然的存在矛盾
- 在不同需求之间做取舍、平衡成为调度的主要目标

调度算法回顾

- FCFS：先到先服务；利用率↑、 吞吐率↑、 公平↑、 响应时间↓
- SJF：短作业优先；响应时间↑、 饿死↓
- Round Robin：轮询；响应时间↑、 公平↑、 周转时间↓、 吞吐率↓
- Priority Based：优先级调度；高优先级（io）响应+周转↑、 饿死↓
- Dynamic Priority Based：动态优先级；公平↑
- Lottery Scheduling: 彩票调度；公平↑

调度算法回顾 (cont.)

- 无论是ULE的penalty，MuQss的virtual deadline，还是CFS的virtual runtime，都继承了以上思想。
- 比如CFS的share语义和彩票调度中的share语义。

$$priority = f(score)$$

$$score = penalty + niceness$$

$$penalty = \begin{cases} \frac{m}{s/r}; & s > r \\ \frac{m}{r/s} + m; & otherwise \end{cases} \quad m = 50$$

$$thread\ is \begin{cases} interactive & score < 30 \\ batch & otherwise \end{cases}$$

virtual deadline = jiffies + (prio_ratio[task_static_prio] * quantum)

$$\text{prio\_ratio} \begin{cases} 1 & 0 \\ \text{prio\_ratio}[0] * 1.1^{(n-1)} & > 0 \end{cases}$$

vruntime += t * (weight based on nice of process)

调度算法回顾 (cont.)

- 除了任务选取算法，其他实现同样重要。
  - rq：global，per-core，per-eneity，per-cluster ……
  - load：task number，nice weight，pelt，walt ……
  - quantum：固定，不固定 ……
  - Load balance：显式，隐式，push，pull ……
  - Thread：user thread，kernel thread，1:1，N:1，N:M，coroutines ……
  - 资源隔离：cpuset，cpu cgroup，quota ……
  - ……
- 实现和实现之间没有好坏之分，只有对应某场景的适合与不适合
- 举个例子：设计一个面向终端领域的调度器（EAS有改进，但不完善）

面向终端调度设计

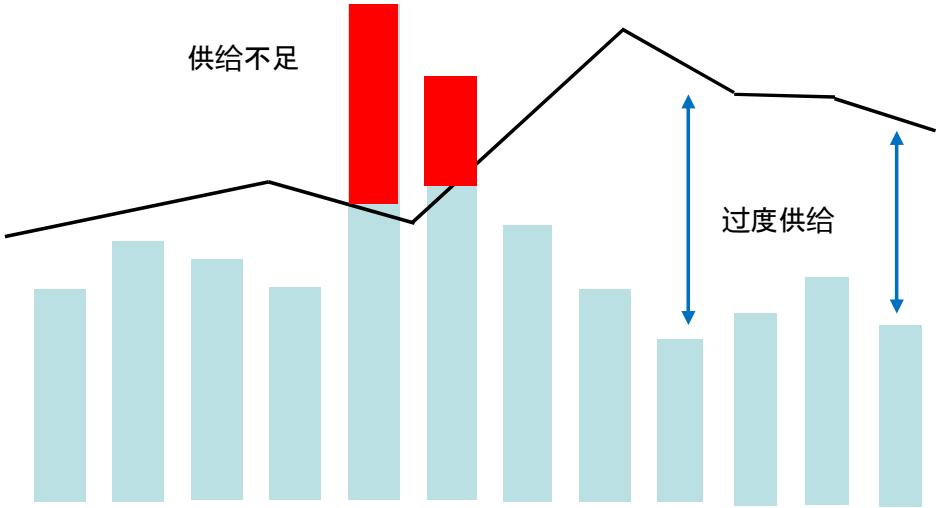
- 需求排序：
  - 首先也是最重要的是不能“卡”
    - 交互时，有能力维持恒定60fps
    - 大量操作场景越快越好。启动应用，load游戏等
  - 满足上条基础上，随时随地保持整体功耗最低
    - 除了有利续航，另外一个意义是留head room
  - 相对公平
    - 避免全局资源，全局锁等引起的优先级反转
    - 同时运行过多任务是系统灾难
    - 任务分组的准确性很关键

面向终端调度设计 (cont.)

- jitter：严格的全局优先级调度，关键任务必能抢占非关键任务
  - scheduler latency ： share -> priority　ms –> us
  - cpu cgroup：share分配不合理导致任务饿死
  - cpuset：大量后台导致cluster整体频率被拉上去
  - load balance：关键任务的LB 和 schedtune的 prefer_idle 是两个概念，时间粒度
  - dos cpu：throtting or demote
  - global rq VS per-cpu rq

面向终端调度设计 (cont.)

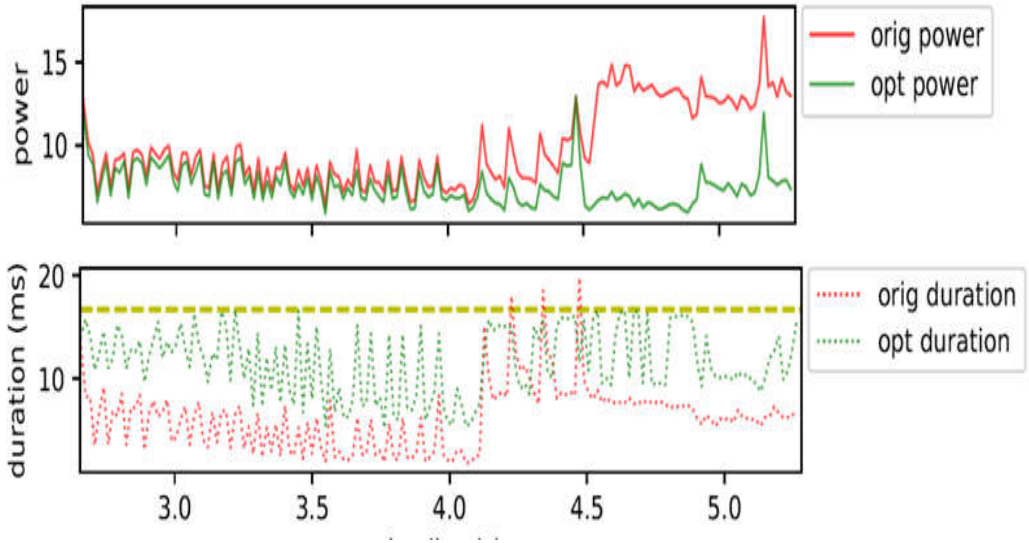
- capacity：准确的Load prediction
  - load tracking ： 历史发生
  - capacity分配：此刻决定
  - running：之后运行
  - ？？
- how
  - hint: 各种boost、各种powerhint ……
  - scenario last value not windows last value
  - Information: high level, low level
  - phase analysis
  - deadline: vtime load



HUAWEI

面向终端调度设计 (cont.)

- capacity：基于scenario load ，建立QoS模型。让调度器知道分配多少capacity能满足task group的deadline
  - pelt VS walt VS scenario load
  - 为什么需要QoS？ 蓝光视频 VS 标清视频 (性能、功耗)
  - task group：group load，木桶理论，group schedtune boost
  - deadline： aware capacity， EDF scheduler -> EDF capacity allocation。



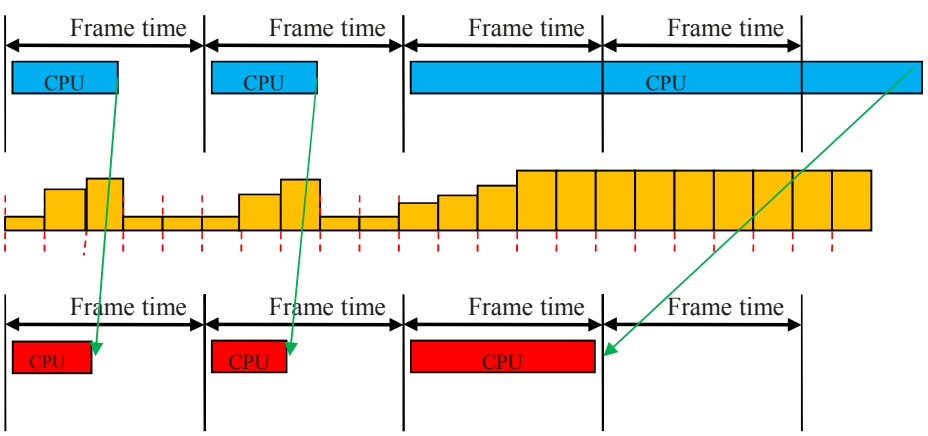
依据满足帧绘制deadline条件下的调优，某场景下：

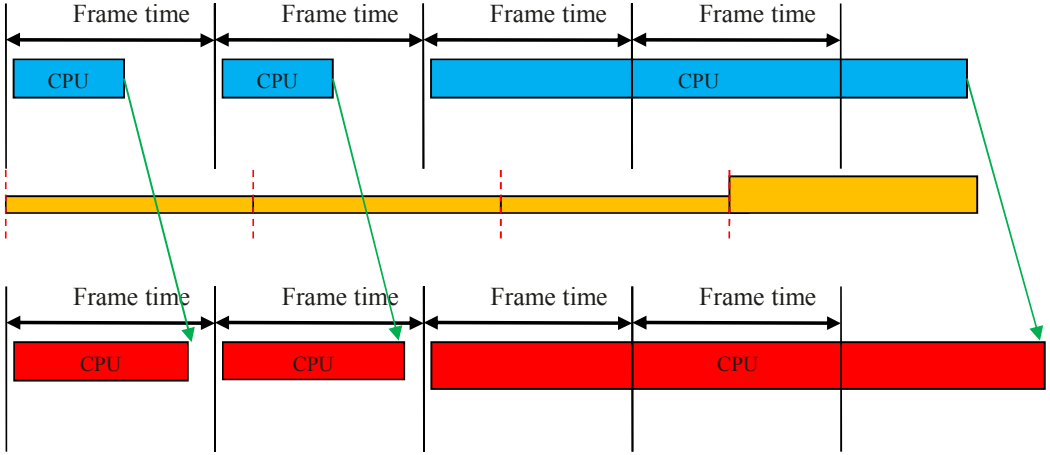
- 平均帧时长增加**113%**（无丢帧）
- 功耗降低**20%**。

HUAWEI

面向终端调度设计 (cont.)

- 单调（激进、保守）的capacity分配参数，无法保证性能和功耗的平衡。

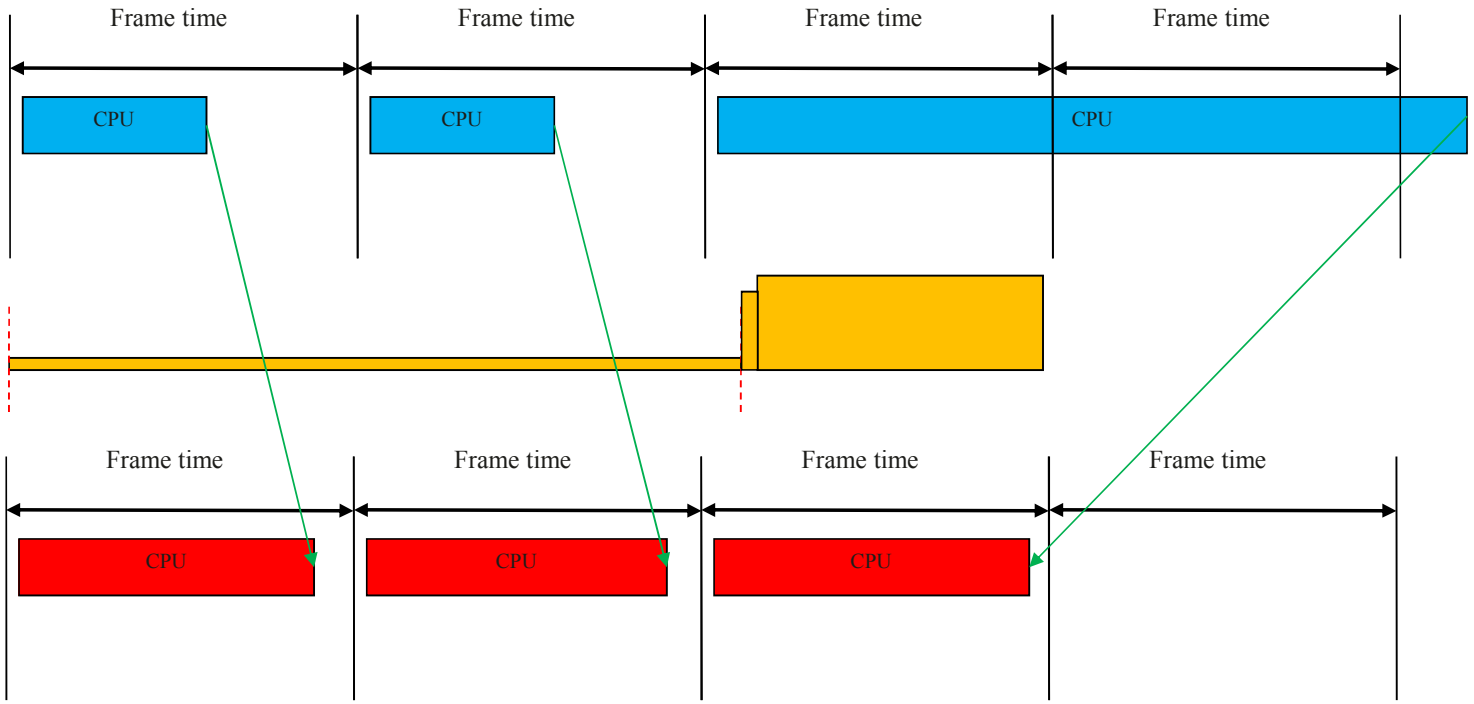




HUAWEI

面向终端调度设计 (cont.)

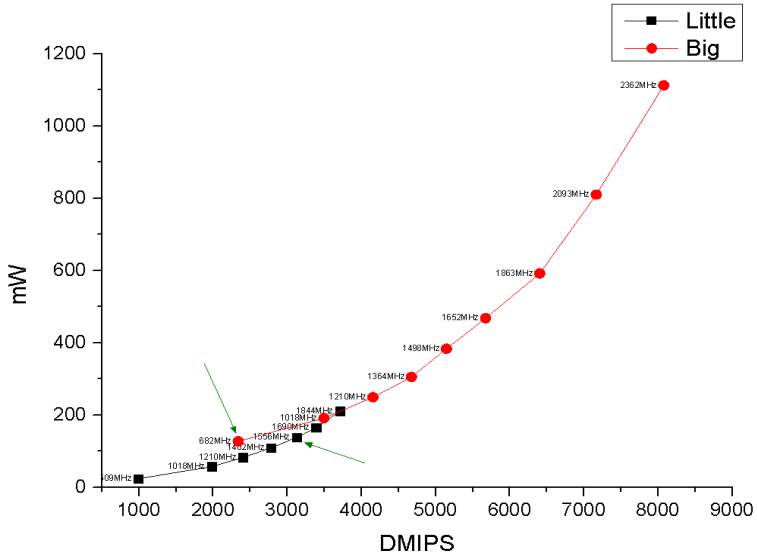
- aware capacity的deadline功率分配 + 负载预测 使性能功耗平衡成为可能。

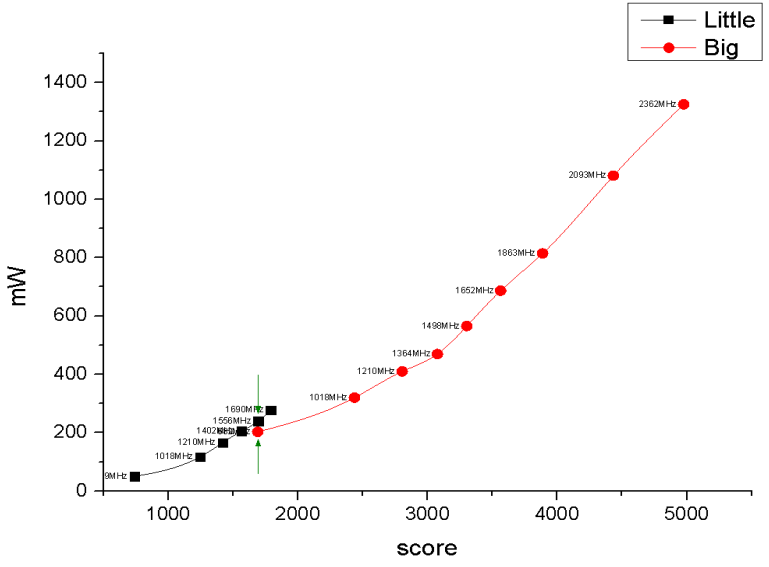


HUAWEI

面向终端调度设计 (cont.)

- energy：依据任务运行pattern的动态能耗模型。
  - 100% Running benchmark != capacity
  - 充分考虑程序周期特征，微架构影响所带来的pipeline stall



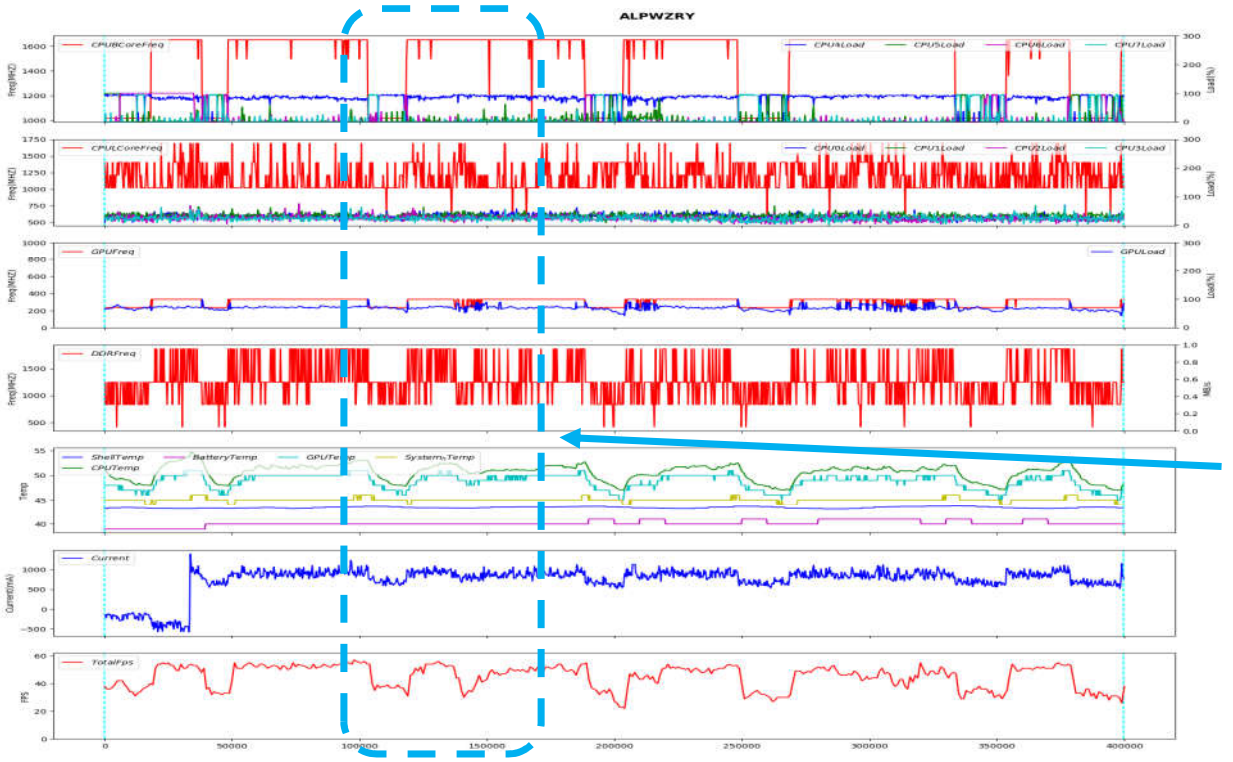


某ARM64开发板

HUAWEI

面向终端调度设计 (cont.)

- energy：统一管理所有功耗特性：task migration，cpufreq，cpuidle，devfreq，cache partition，thermal control，IPA ……
  - IPA、温控控什么？QoS
  - cpufreq or ddrfreq？
  - cpufreq or task placement



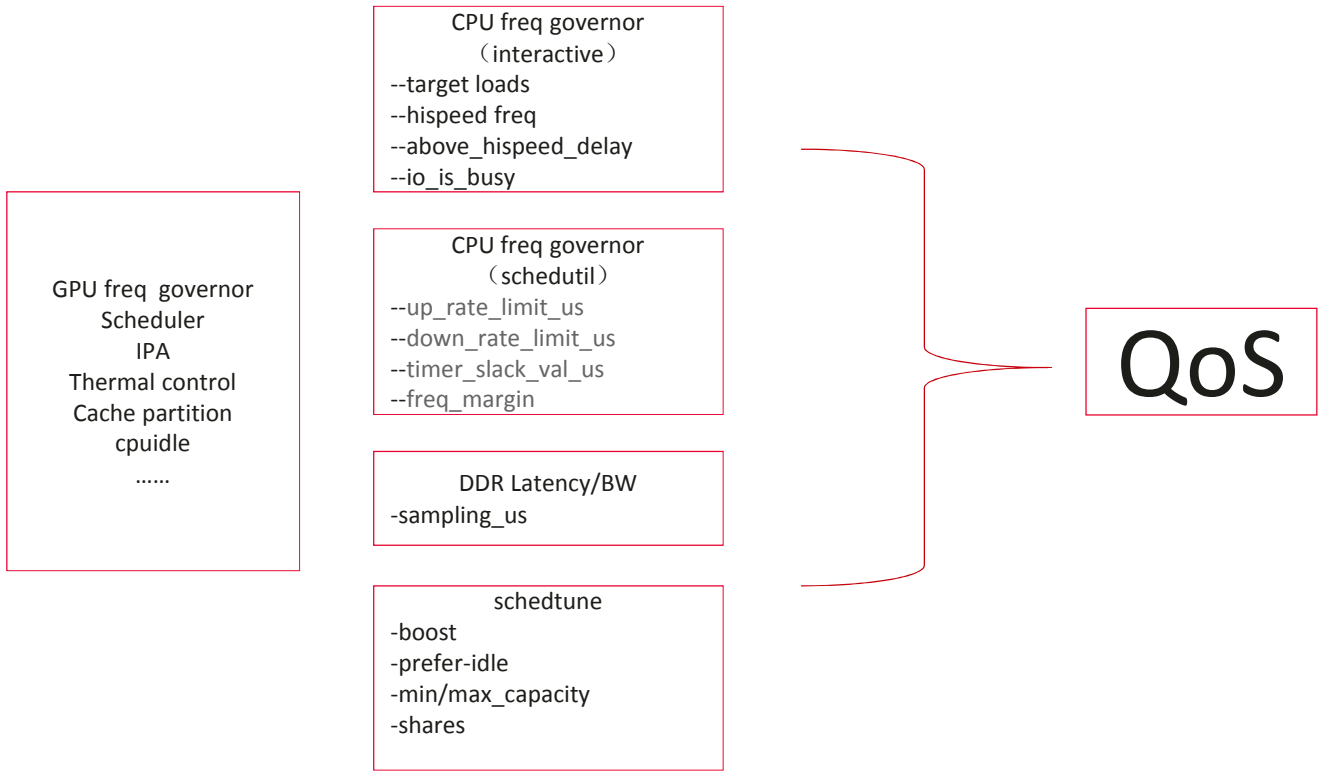
HUAWEI

面向终端调度设计 (cont.)

- 相对公平：适度的资源隔离：irq，cache，memory，io，network……
  - irq：隔离、线程化
  - cache：MPAM
  - memroy：allocate（静态、动态）、bandwidth（MPAM）
  - 锁依赖传递
  - cgroup
  - ……

面向终端调度设计 (cont.)

- 简单、直观的对上调优接口，调度器成为整机计算资源分配中枢。



HUAWEI

面向终端调度设计 (cont.)

- 进一步优化
  - 更深层次的软硬结合
    - 能耗模型卸载到芯片中
    - 更精细化的clock、电源管理
  - AI 调优
    - AI 获取任务关联性，精确分组
    - AI 预测负载
    - AI 场景识别，下发参数
  - ……

面向终端调度设计 (cont.)

- 基于CFS的优化效果：
  - 响应时延降低 25.7%
  - 时延波动率下降55.6%
  - 能效比提升20%

写在最后

- 调度器的战争中没有绝对赢家，只有适合不适合：
  - 面向终端调度器设计 （phone，pad，notebook……）：interactive，energy
  - 面向确定性时延调度器设计（航空，mdc，工控……）：deterministic，mcs
  - 面向计算调度器设计（服务器，集群）：schedule level，distributed ……

- 整机性能功耗问题是系统性问题
  - IPC（ Instruction Per Cycle ）
  - 软件膨胀
  - 功耗墙
  - 并行化能力（inflate、obtain-view）
  - 硬件瓶颈

欢迎加入华为OS内核实验室

华为OS内核实验室：华为端、管、云核心的OS软件基础设施

OS Kernel Lab

- Linux内核（ARM/x86/异构等）的技术研发与创新
- 低时延、高安全、高可靠、高智能的下一代OS内核技术的研究和成果转化

招聘岗位

下一代操作系统研究员/高级工程师

形式化技术研究员/高级工程师

Linux内核架构师/高级工程师

工作地

杭州、北京、上海、南京、深圳

简历投递

Tel：13738037695

Email：

leijitang@Huawei.com

HUAWEI

Thank you.

把数字世界带入每个人、每个家庭、
每个组织，构建万物互联的智能世界。

Bring digital to every person, home, and organization for a fully connected, intelligent world.