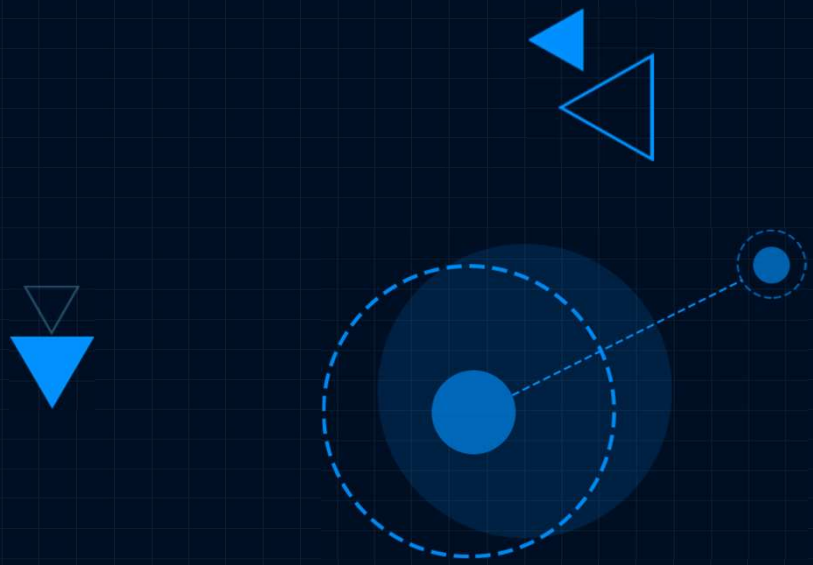
qGPU算力隔离与在离线混部

——GPU计算降本增效的终极方案



宋吉科

腾讯云异构计算研发负责人

目录

算力隔离：问题

问题

- 直通GPU无法多业务共享资源，利用率低、成本高
- vGPU 实例资源配置固定不灵活、虚拟机实例调度成本高，非进程级调度

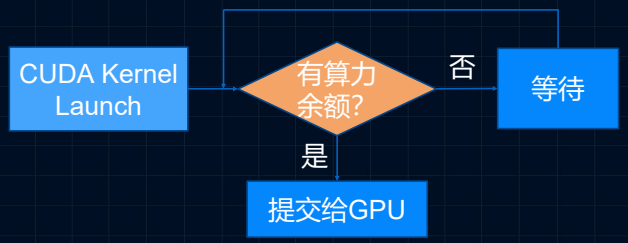
- 显存/算力/故障隔离性低，不同客户、任务之间存在资源的抢占和干扰
- GPU share等不支持QoS
- vCUDA需要侵入式修改CUDA库
- MPS存在故障隔离问题
- MIG只有高端GPU支持，灵活度低

算力隔离：困难

业界的状态

- GPU驱动的核心是一个~26M的二进制文件nv-kernel.o_binary
- 2009以来，国内外学术界和工业界，进行了大量的研究，尝试解决该价值巨大的GPU算力隔离（QoS）问题：
  - API拦截方案如vCUDA、rCUDA，其算力隔离是粗粒度的（coarse-grained QoS），且有入侵用户环境的副作用
  - ACM、IEEE每年针对该topic发表数十篇paper，大多集中在API层
  - 开源社区对英伟达软硬件、固件进行了大量研究，但未能形成完善的CUDA支持
    - Nouveau/Linux只支持OpenGL，不能支持CUDA
    - GDEV基于nouveau初步支持了CUDA，但代码老旧
  - 英伟达官方的GRID vGPU、MPS方案，支持细粒度算力隔离（fine-grained QoS），但分别带有缺陷：
    - vGPU依赖系统虚拟化，不适用于容器
    - MPS导致额外的故障传播，无法用于生产环境

API拦截的粗粒度隔离问题

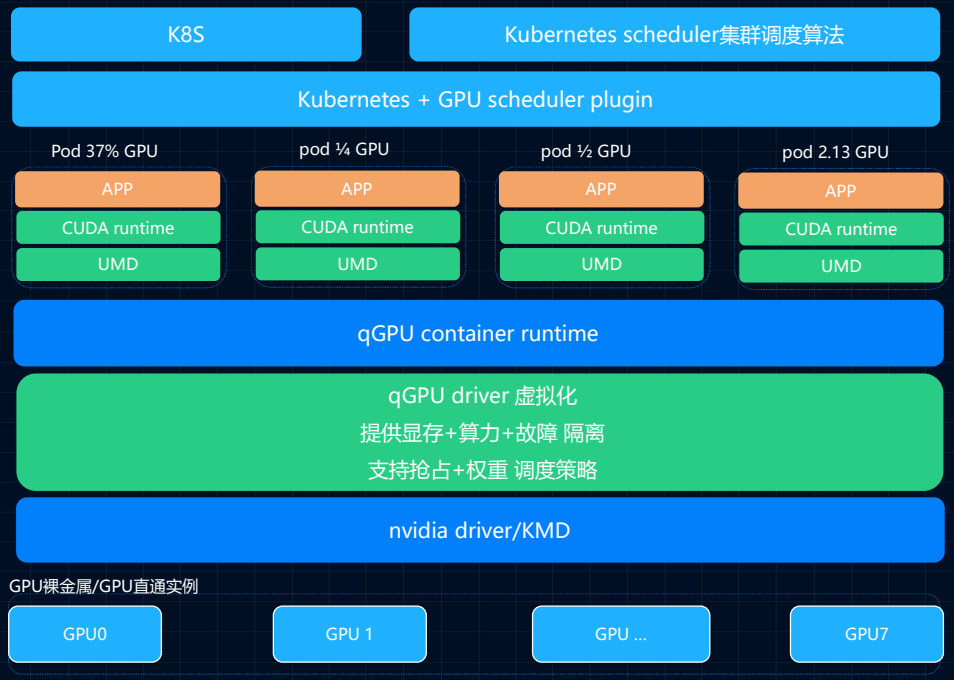


- Kernel执行时间不可预知，又无法中断
- API拦截方案，算力隔离只能以Kernel为粒度，存在问题：
  - 如果还有算力余额，可能一放过去就超份额了，造成其它业务抖动
  - 如果没有算力余额，只能秒级等待新的算力配额，造成本业务抖动

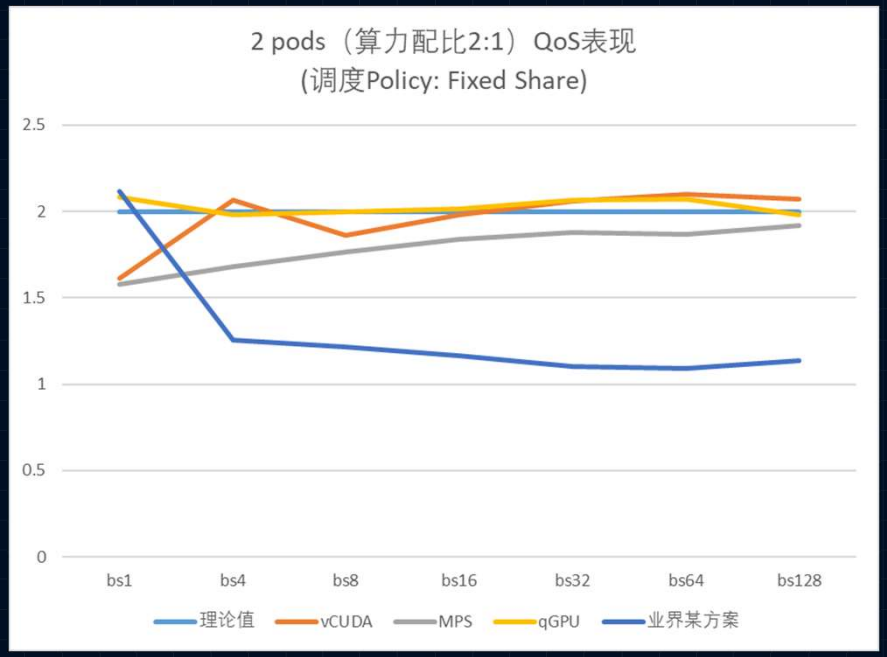
算力隔离：qGPU实现

qGPU（QoS GPU）特点

- 基于nvidia docker实现多POD共享，兼容K8S
- 创建POD时，仅需传入算力比例、显存大小，构建简单
- GPU显存、算力可自由划分
- 用户AI应用无需任何改动/重编，CUDA库无需替换，部署无缝迁移
- 显存、算力的精准隔离



算力隔离：qGPU效果



- 横轴为batch值，纵轴为运行中两个POD的实际算力比例
- batch较小时，GPU负载低，算力隔离效果或不明显；
- 随着batch增大，GPU负载增加，qGPU和MPS都趋近理论值2，vCUDA也相差不远，但缺乏算力隔离的方案则会趋近1。

综合对比

特性 \ 方案	qGPU	vCUDA	MPS	业界某方案
多POD共享GPU	Y	Y	Y	Y
灵活显存/算力配比	Y/Y	Y/Y	N/Y	Y/Y
用户环境无侵入、兼容性好	Y	N	Y	Y
显存隔离	Y	Y	N	Y
故障隔离	Y	Y	N	Y
精准算力隔离	Y	弱	Y	N
算力无损	Y	弱	Y	Y
业务不抖动	Y	弱	Y	Y

qGPU优势

- 全方位无缺点
- 解锁此前必须用pGPU整卡应对的场景： 云函数，小规格推理，同时有在线和离线任务等
- 以在离线混部场景为例，GPU利用率提升100%+

目录

qGPU调度技术详解

- Per-pGPU设置scheduling policy
- 支持三种调度策略：Best Effort、Fixed Share和Burst Share
- 支持两种调度类：在线、离线
  - 离线pods运行时，如果在线pod来了任务，实时捕捉到它
  - 实时抢占掉离线pods

Policy value	Policy name	含义解释
0	Best Effort	默认值。 各个 Pods 不限制算力，只要卡上有剩余算力就可使用。如果一共启动 N 个 Pods，每个 Pod 负载都很重，则最终结果就是每个Pod获得了 1/N 的算力
1	Fixed Share	每个 Pod 有固定的算力配额，无法超过该配额，即使 GPU 还有空闲算力、即使它用不完该配额。
2	Burst Share	每个 Pod 有保底的算力配额，但只要 GPU 还有空闲算力，就可被 Pod 使用。例如，当 GPU 有空闲算力时（没有分配给其他 Pod），Pod 可以使用超过它的配额的算力。注意，当它所占用的这部分空闲算力再次被分配出去时，Pod 会回退到它的算力配额。

目录

qGPU在离线混部技术探索

在离线 / pGPU调度策略	离线Pods	在线Pods
0 Best Effort	低优先级调度类	高优先级调度类
1 Fixed Share		
2 Burst Share		

- 同一pGPU，可以有N个离线Pods、M个在线Pods
- pGPU的policy设置为0、1、2之一
- 在线Pods空闲中，一旦有任务进来，实时捕捉到、并抢占掉正在执行的离线任务。

- 如果一个调度周期中，在线Pods没有任务，则离线pods按照Policy来分配GPU算力
- 如果一个调度周期中，在线Pods有任务，则qGPU优先调度在线pods
- 如果有多个在线Pods，则**在线Pods之间只支持Best Effort策略，其它policy的算力隔离需进一步研究。**

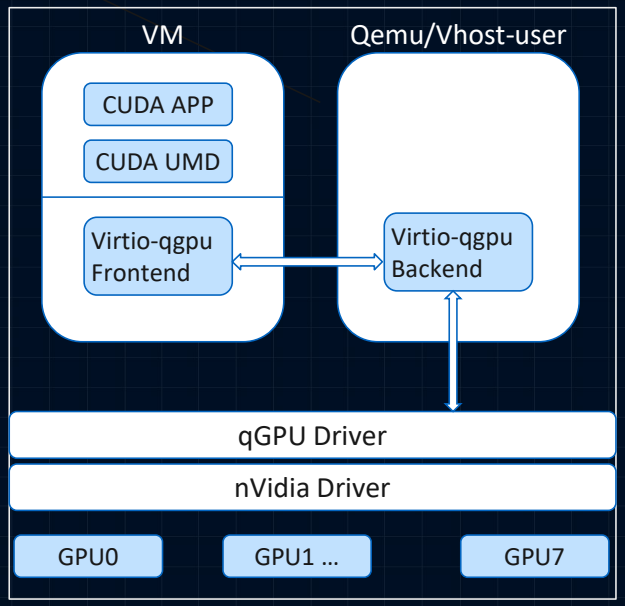
目录

再出发：更多的可能性 - 任意隔离

难度指数：★★★★★

- 同一pGPU，可以有N个离线Pods、M个在线Pods
- pGPU的policy设置为0、1、2之一
- 在线pods之间，也支持按照policy进行算力隔离

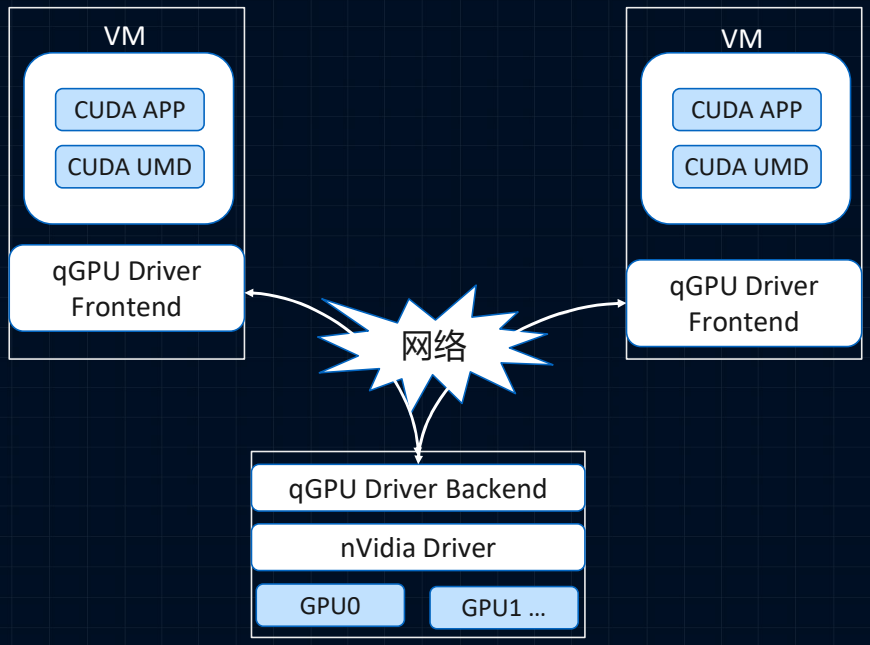
再出发：更多的可能性 - 系统虚拟化



难度：★★★★★

- VM中看到virtio-qgpu设备，UMD只和它交互
- Virtio-qgpu Backend实现在Qemu或独立的vhost-user进程中，它和qGPU Driver交互

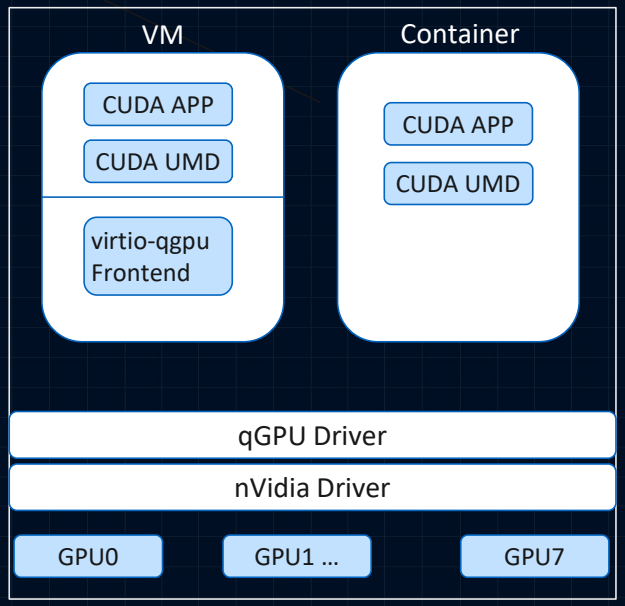
再出发：更多的可能性 - qGPU Remoting



难度：★★★★★

- 从qGPU Frontend所在的low-level，remoting到远端的GPU机器
- 为防止 "拦截层级越低、转发次数越多" 的bloating问题带来严重的latency，需要智能聚合low-level events
- 几乎不可能实现？

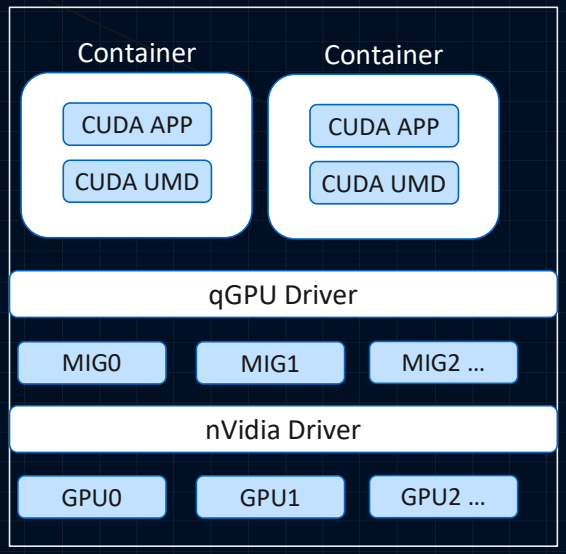
再出发：更多的可能性 - 热迁移



难度：★★★★★

- VM（Qemu-based）和Container（Criu-based）热迁移
- SYSMEM： DMA Dirty Tracking by IOMMU
- VIDMEM： Dirty Tracking by GMMU

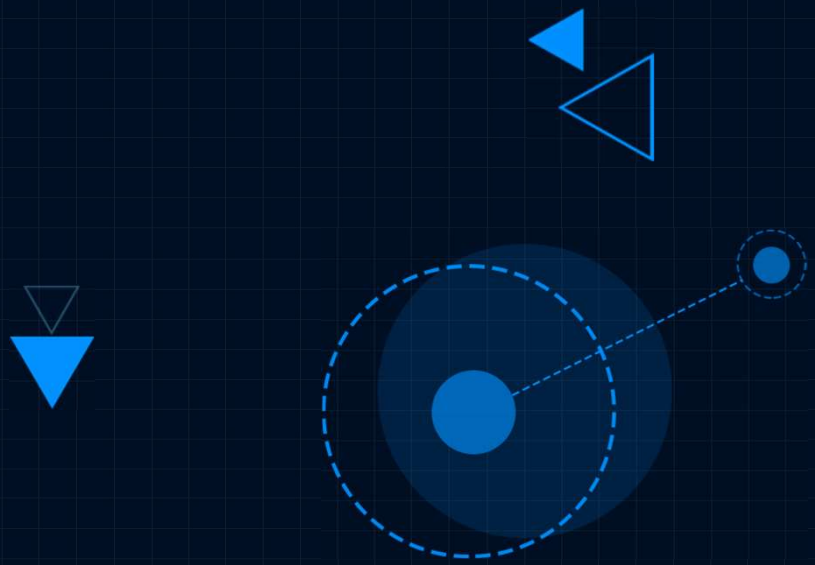
再出发：更多的可能性 – qGPU on MIG



难度：★★★★★

- qGPU支持MIG
- MIG提供空分的性能优势
- qGPU提供时分的灵活性

- "Imagination is more important than knowledge. " - Einstein
- "I would rather have questions that cannot be answered than answers that cannot be questioned. " - Feynman



Thanks!

欢迎体验&评测qGPU：

微信号：TKEPlatform