

如何精打细算使用 GPU

主讲人：牛文灿

Content

目录

1. 引言（5分钟）
2. GPU浪费原因分析（15分钟）
3. 如何解决GPU浪费问题（30分钟）
4. 总结（5分钟）
5. 问答环节（5分钟）

Part 01

引言

■ AI，让数据中心支出大增

Before

服务器的平均售价约为 7,000 美元



After

H100 8 路服务器售价 30 万美元

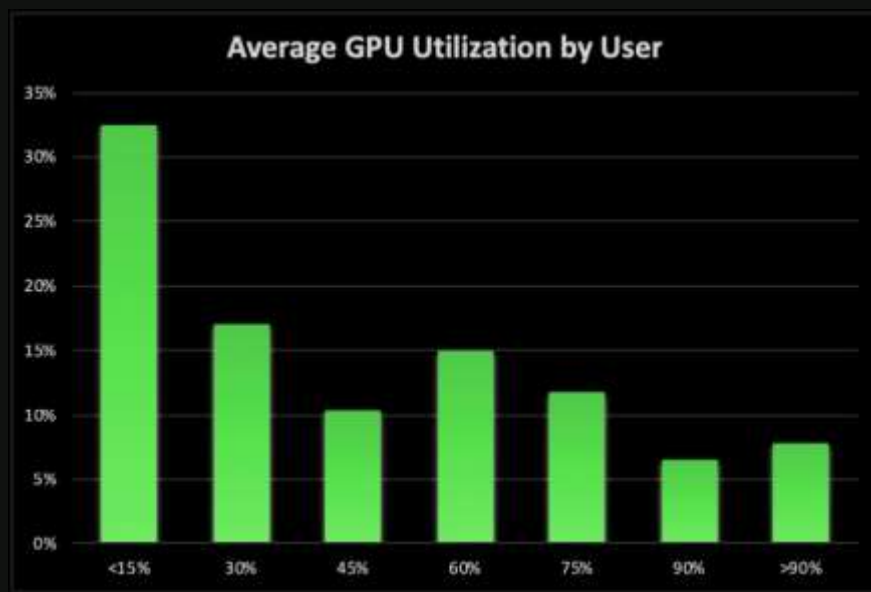
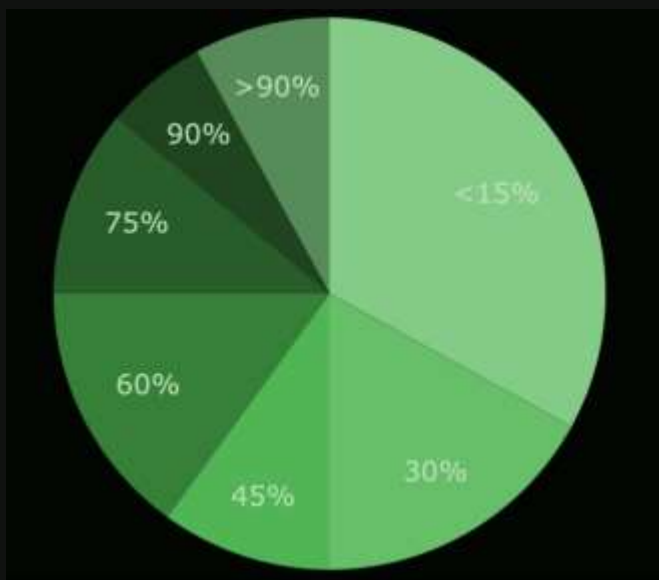
潜在成本

网络设备、存储、软件、人员技能提升

■ GPU利用率低调查分析

Average GPU Utilization

GPU utilization is under 30% for half of users (source: wandb)



最大化 GPU 利用率至关重要

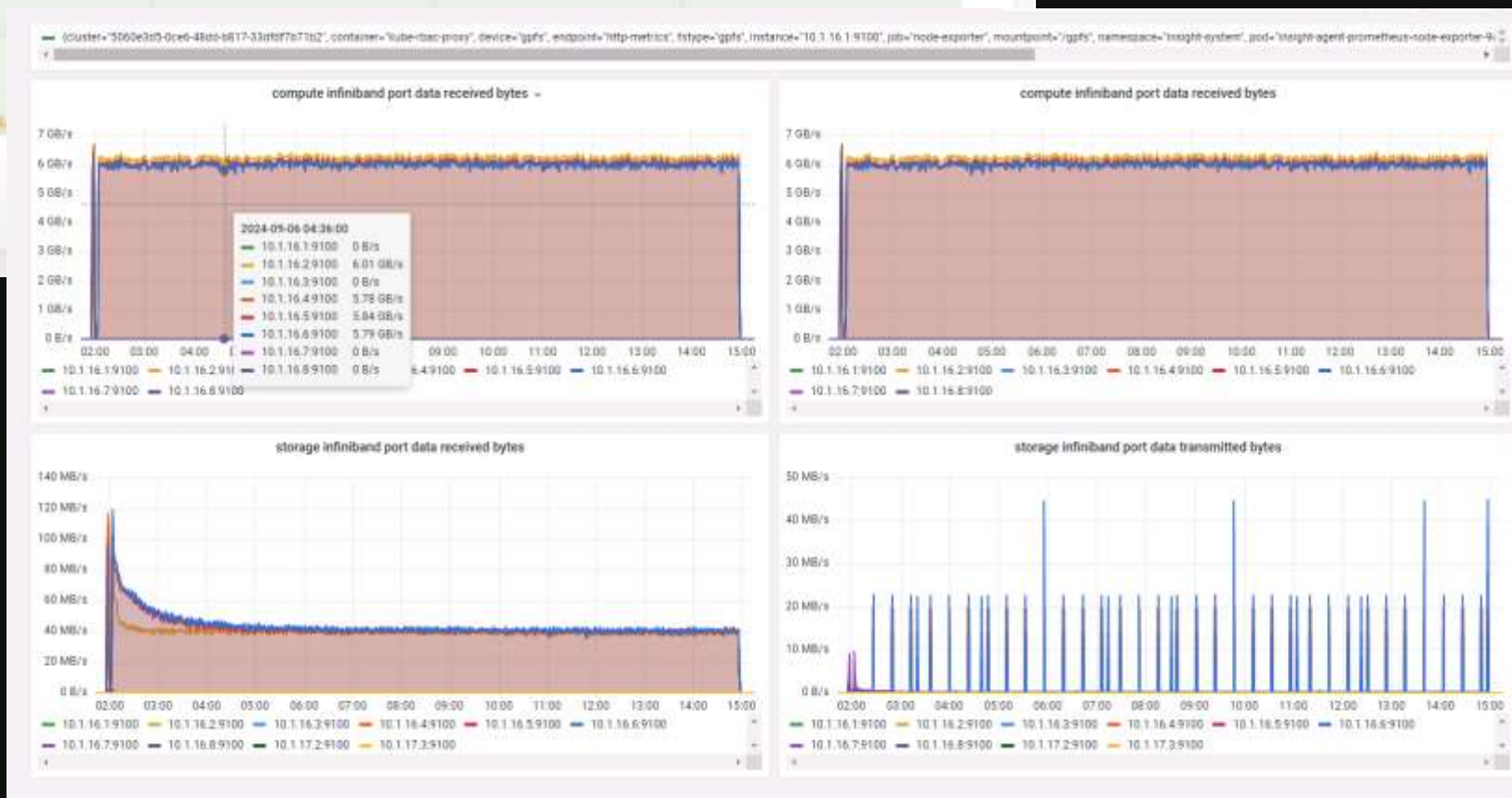
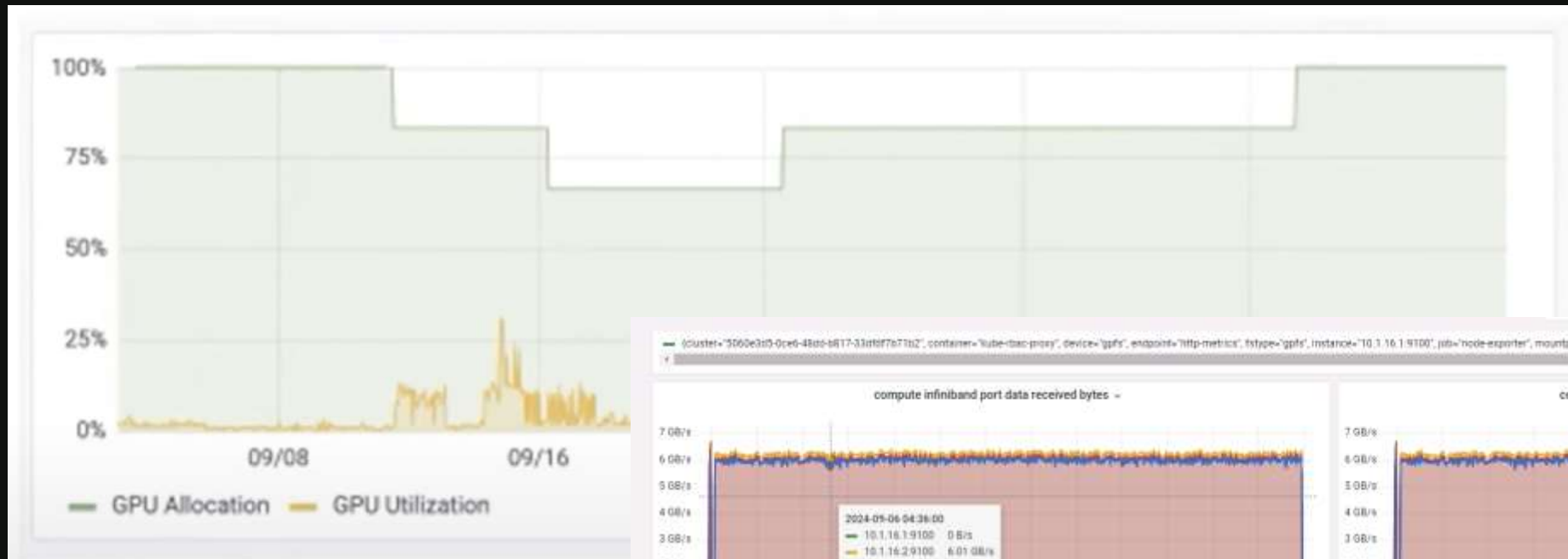
- 成本优化，GPU 是昂贵的投资
- 人工智能基础设施的投资回报率
- 最大限度地提高 GPU 利用率有助于可持续发展

根据 2024.4 对人工智能基础设施的调查，优化 GPU 利用率是一个主要问题。Wandb.ai 的另一份报告显示，近三分之一的利用率低于 15%，这是相当低的。

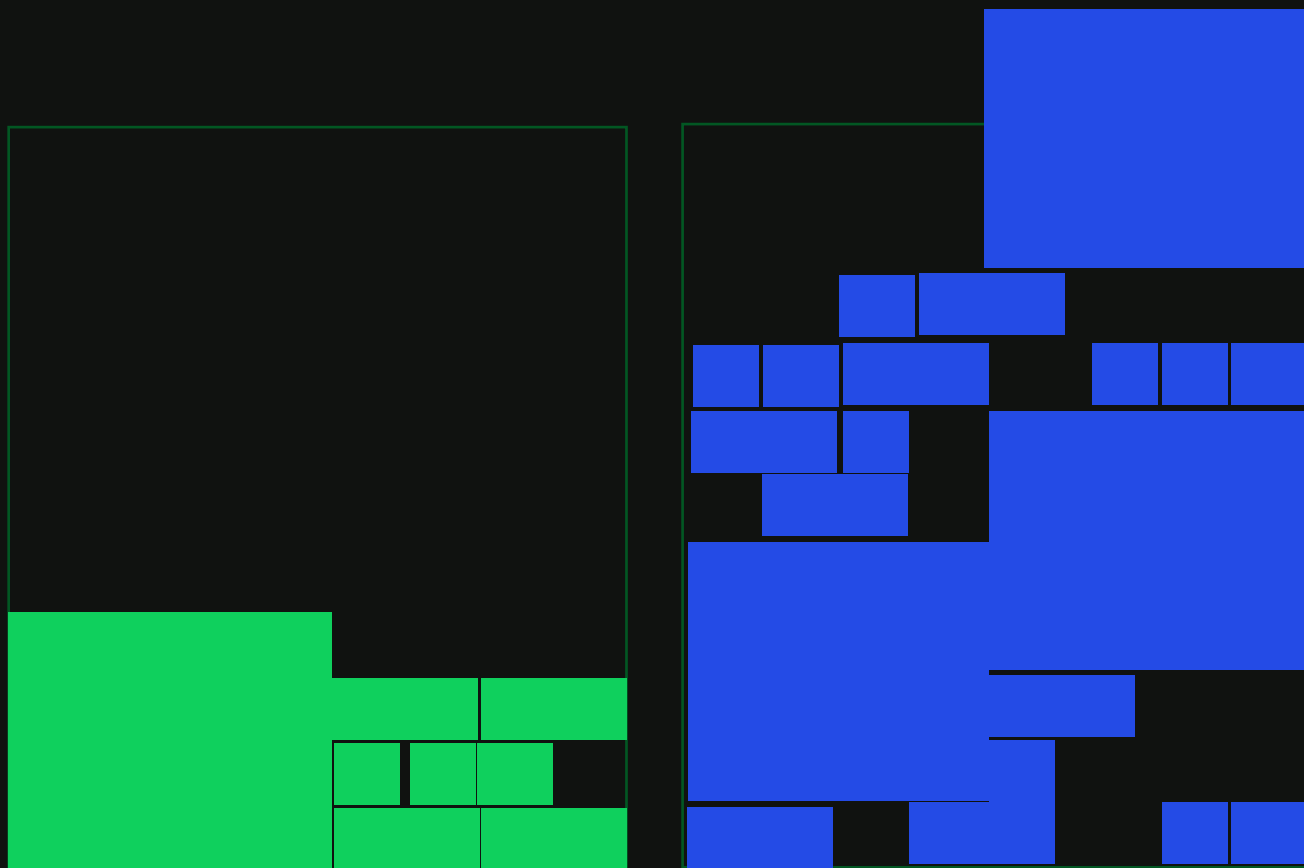
Part 02

GPU使用中存在浪费的原因

■ 浪费原因1 — GPU 独占，GPU 使用率低

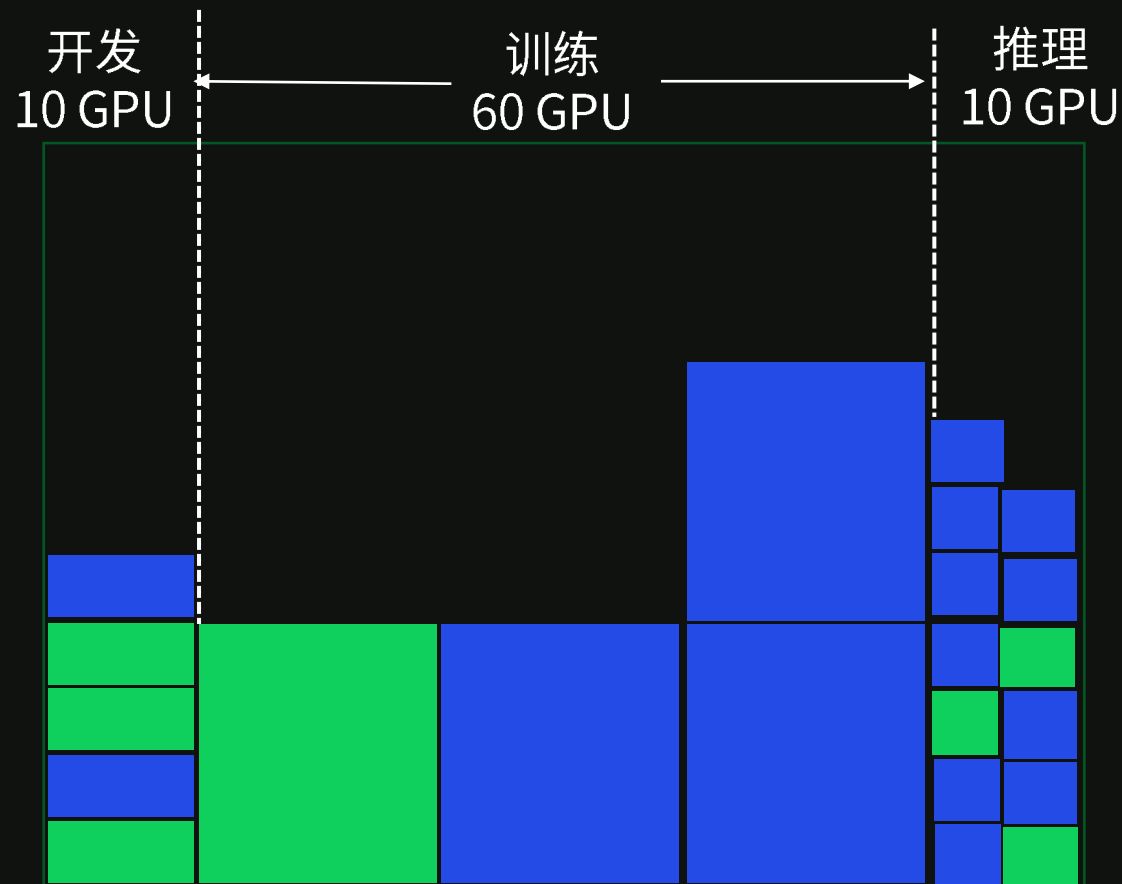


■ 浪费原因 2 — 资源孤岛



Team A: 40 个 GPU
开发 / 训练 / 推理

Team B: 40 个 GPU
开发 / 训练 / 推理



理想情况: GPU 池化

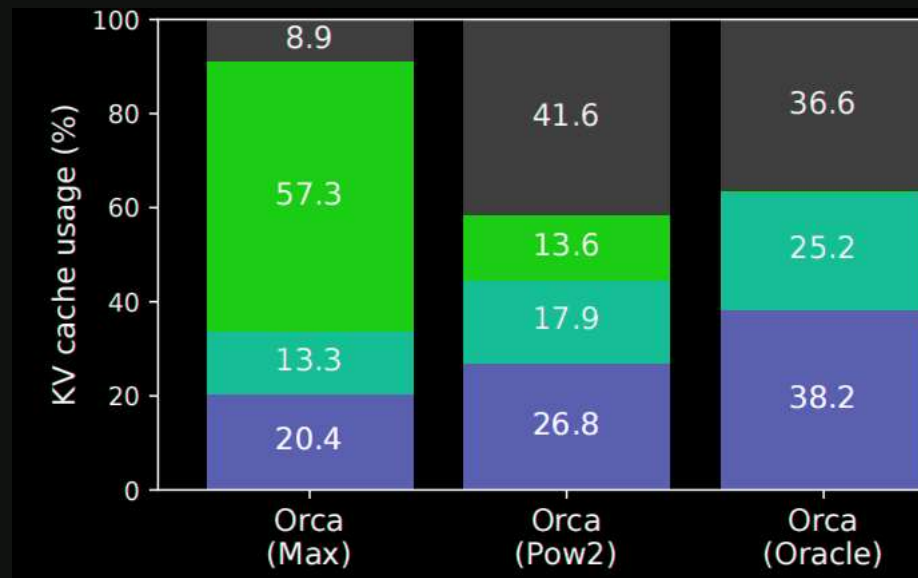
 AI工作负载

■ 浪费原因 3 — 模型部署和推理优化缺失



模型部署和管理挑战

《Efficient Memory Management for Large Language Model Serving with *PagedAttention*》



Average percentage of memory wastes in different LLM serving systems during the experiment

模型推理遇到显存管理挑战

■ 浪费原因 4 — 费控手段缺失

1. 资源利用率不透明

租户无法清晰地了解自己所使用的计算资源（GPU、CPU、内存、存储等）的具体消耗情况。

2. 成本难以精确计算

在没有精确的计量数据的情况下，很难准确计算出不同用户或不同应用的资源消耗成本。

3. 无法进行有效的资源分配

由于无法准确评估不同用户或不同应用对资源的需求，难以进行合理的资源分配，可能导致资源不足或过剩的情况。

4. 缺乏对资源使用情况的监控和预警


无法实时监控资源的使用情况，当资源出现瓶颈或异常时，无法及时发现并采取措施。

5. 无法实现按需付费

传统的使用模式往往是按固定周期付费，无法根据实际的资源消耗情况进行灵活的计费。



■ 浪费原因 5 — GPU 选型不当

Category	A100	H100	L40S	H200								
架构	Ampere	Ada Lovelace	Hopper	Hopper								
Release Year	2020	2022	2023	2024								
FP64 Performance	9.7 TFLOPS	34 TFLOPS	34 TFLOPS	34 TFLOPS								
FP64 (Tensor Core)	19.5 TFLOPS	67 TFLOPS	Ampere / 30-Series									
FP32 Performance	19.5 TFLOPS	67 TFLOPS	GeForce RTX 3090 Ti	GeForce RTX 3090	GeForce RTX 3080 Ti	GeForce RTX 3080	GeForce RTX 3070 Ti	GeForce RTX 3070	GeForce RTX 3060 Ti	GeForce RTX 3060	GeForce RTX 3050	
TF32 Performance	312 TFLOPS	990 TFLOPS	NVIDIA CUDA Cores	10752	10496	10240	8960 / 8704	6144	5888	4864	3584 / 2304	
BFLOAT16/FP16 Performance	624 TFLOPS	1,979 TFLOPS	Boost Clock (GHz)	1.86	1.70	1.67	1.71	1.77	1.73	1.67	1.78 / 1.76	
INT8 Performance	1,248 TOPS	3,958 TOPS	Memory Size	24 GB	24 GB	12 GB	12 GB / 10 GB	8 GB	8 GB	8 GB	12 GB / 8 GB	
GPU Memory	80 GB HBM2e	80 GB HBM3	Memory Type	GDDR6X	GDDR6X	GDDR6X	GDDR6X	GDDR6X	GDDR6	GDDR6	GDDR6	
Memory Bandwidth	2,039 GBps	864 GBps	Ada Lovelace / 40-Series									
Availability	Not applicable	NVIDIA	GeForce RTX 4090			GeForce RTX 4080		GeForce RTX 4070 Ti				
TDP	400W	700W	NVIDIA CUDA Cores			16384		9728		7680		
Form Factor	8.7 MI Gs 10 GB	8.7 MI Gs 12 GB	Boost Clock (GHz)			2.52		2.51		2.61		
Interconnect	NVLink: 600 GB/s	NVLink: 600 GB/s	Memory Size			24 GB		16 GB		12 GB		
NVIDIA HGX Partner	NVIDIA HGX A100 Partner and NVIDIA-Certified Systems	NVIDIA and NVLink System	Memory Type			GDDR6X		GDDR6X		GDDR6X		
			Max Display Resolution			4K at 240Hz or 8K at 60Hz with DSC		4K at 240Hz or 8K at 60Hz with DSC		4K at 240Hz or 8K at 60Hz with DSC and HDR		
AI Enterprise Support	Included	Add-on	Not applicable		Add-on							
CUDA Cores	6,912	16,896	18,176		Data not available		Powered By  DaoCloud					

Part 03

如何解决GPU浪费问题

- 调度
- 池化
- 共享
- 计量计费
- 场景选型

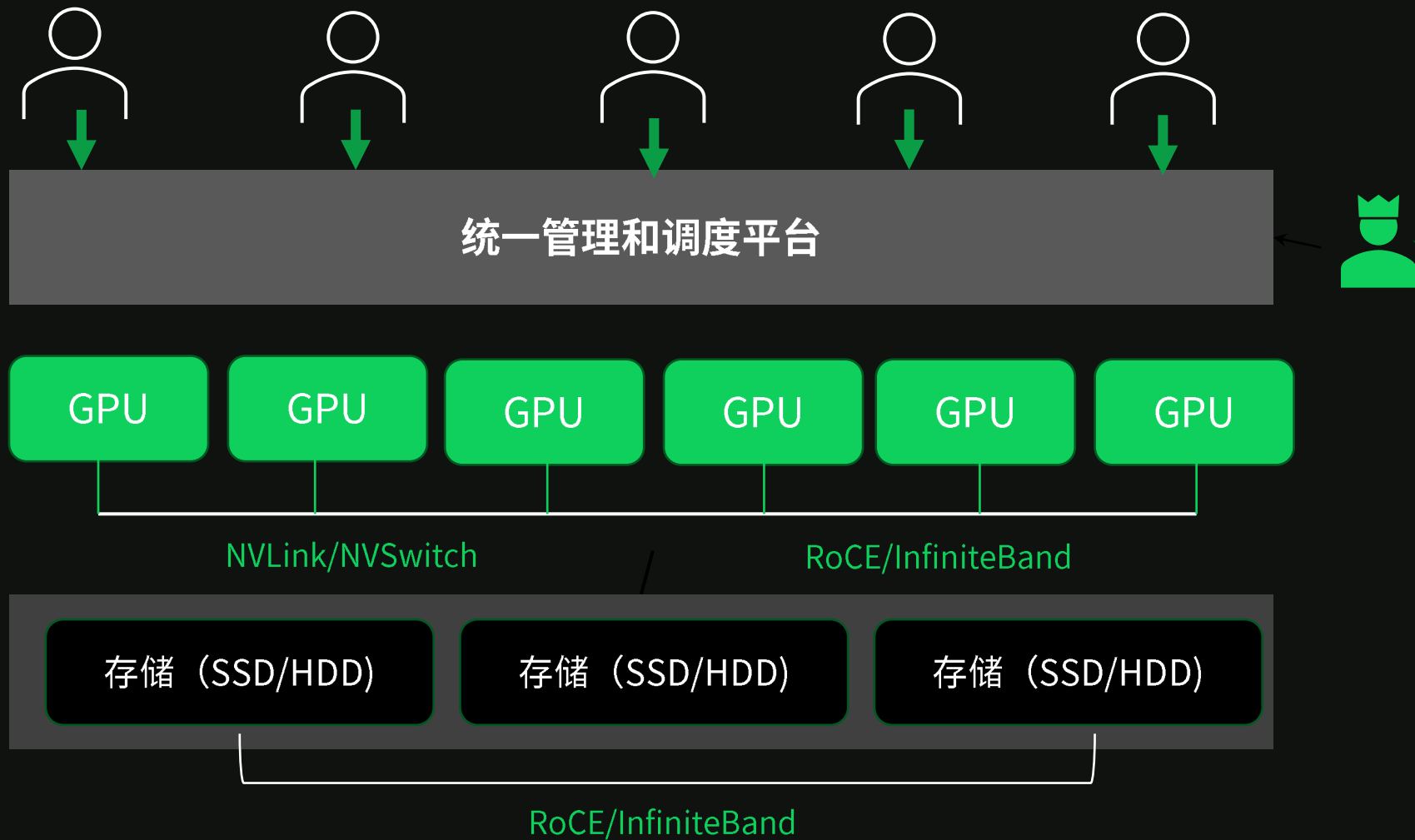
■ AI Workload 是怎么工作的



■ AI Workload 类型和 GPU 使用率

阶段	构建 (Build)	训练 (Train)	推理 (Inference)
1	开发与调试	训练与超参数优化	模型实时服务
2	交互式会话	远程执行	长期运行服务
3	短周期	长周期	计算短暂高峰
4	易用性重要	吞吐量极为重要	延迟和吞吐量重要
5	GPU 利用率低	GPU 利用率高	GPU 利用率低

■ 采用统一的管理调度平台，提升 GPU 利用率



集中式系统创建 GPU 共享池，集中式系统编排作业，同时监控使用情况和分配情况；根据预先定义的策略去分配资源给不同的用户和团队；共享基础设施可以带来硬件利用率和用户生产效率的巨大提升。

■ 基于 Kubernetes 统一调度平台，Nvidia 首选，DaoCloud 耕耘 10 年



Products Solutions Industries For You

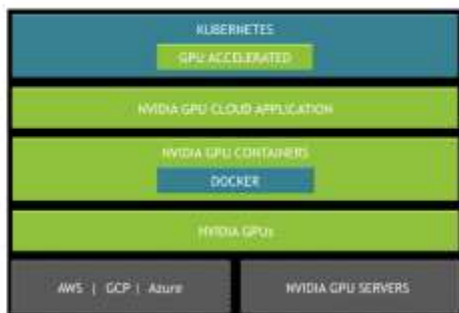
Shop Drivers Support



Why Kubernetes Runs Better on GPUs

Kubernetes includes support for GPUs, making it easy to configure and use GPU resources for accelerating workloads such as data science, machine learning, and deep learning. Device plug-ins enable pods access to specialized hardware features such as GPUs and expose them as schedulable resources.

With the increasing number of AI-powered applications and services and the broad availability of GPUs in public cloud, there's an increasing need for Kubernetes to be GPU-aware. NVIDIA has been steadily building its library of software to optimize GPUs to use in a container environment. For example, Kubernetes on NVIDIA GPUs enables multi-cloud GPU clusters to be scaled seamlessly with automated deployment, maintenance, scheduling, and operation of GPU-accelerated containers across multi-node clusters.



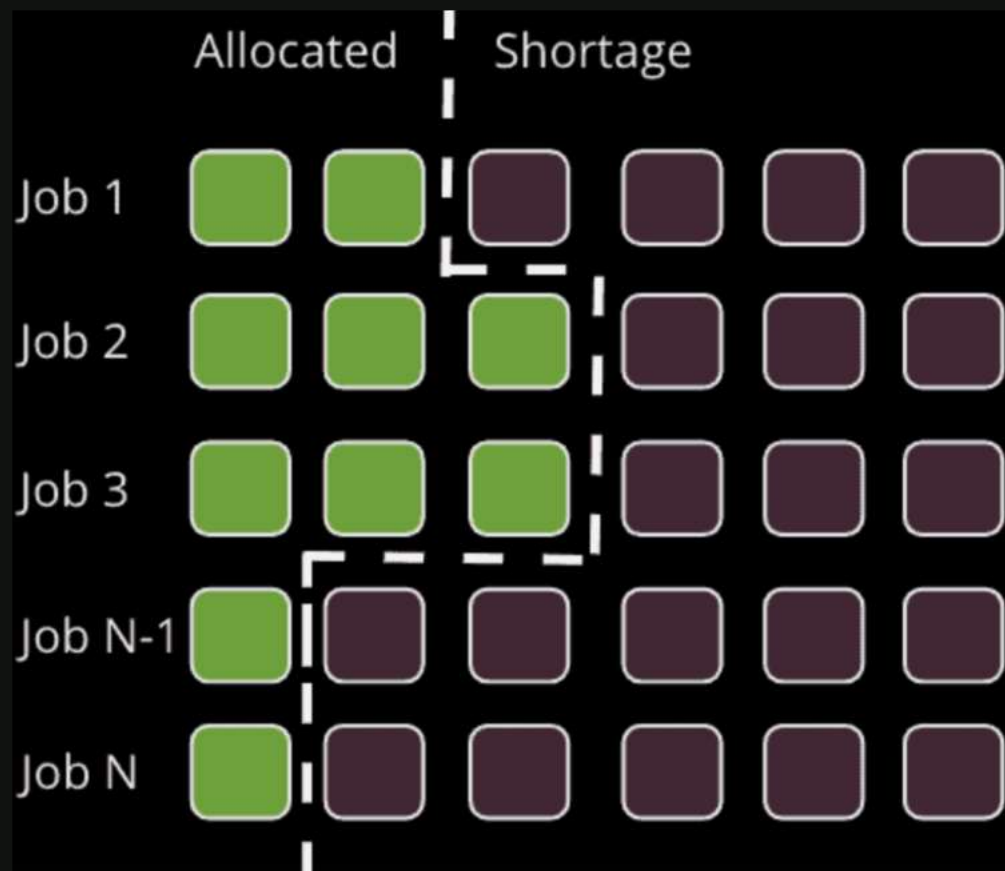
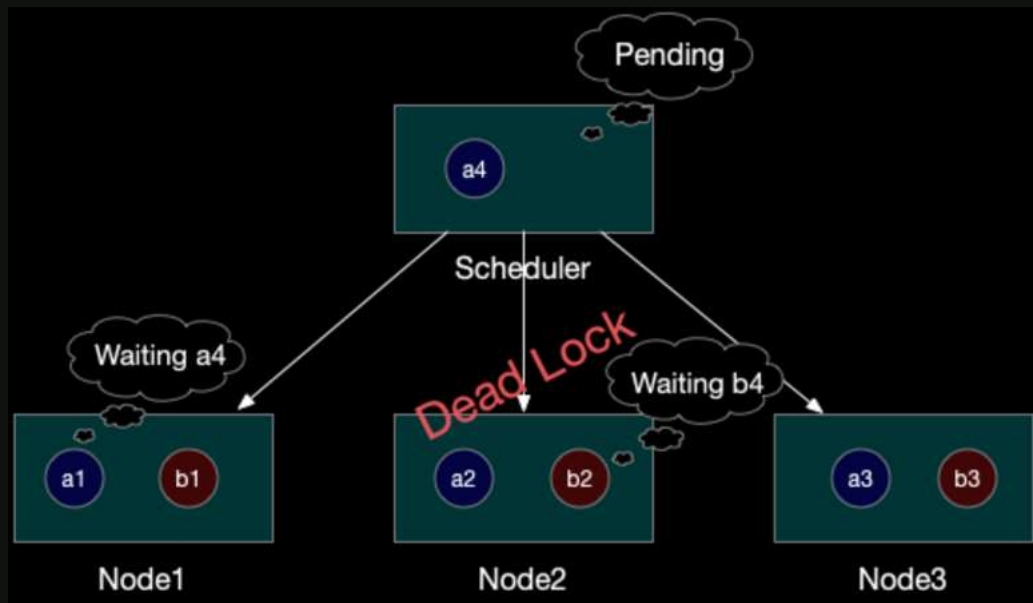
365 days CNCF Any company Any contributor

Show 10 entries Search

#	Company	Commits
0	*independent	12963
1	Google	3729
2	Red Hat	849
3	DaoCloud	580
4	Intel	402
5	PlanetScale	296
6	SUSE	280
7	Buoyant	277
8	NetEase	146

■ Kubernetes 调度 AI 应用面临的挑战

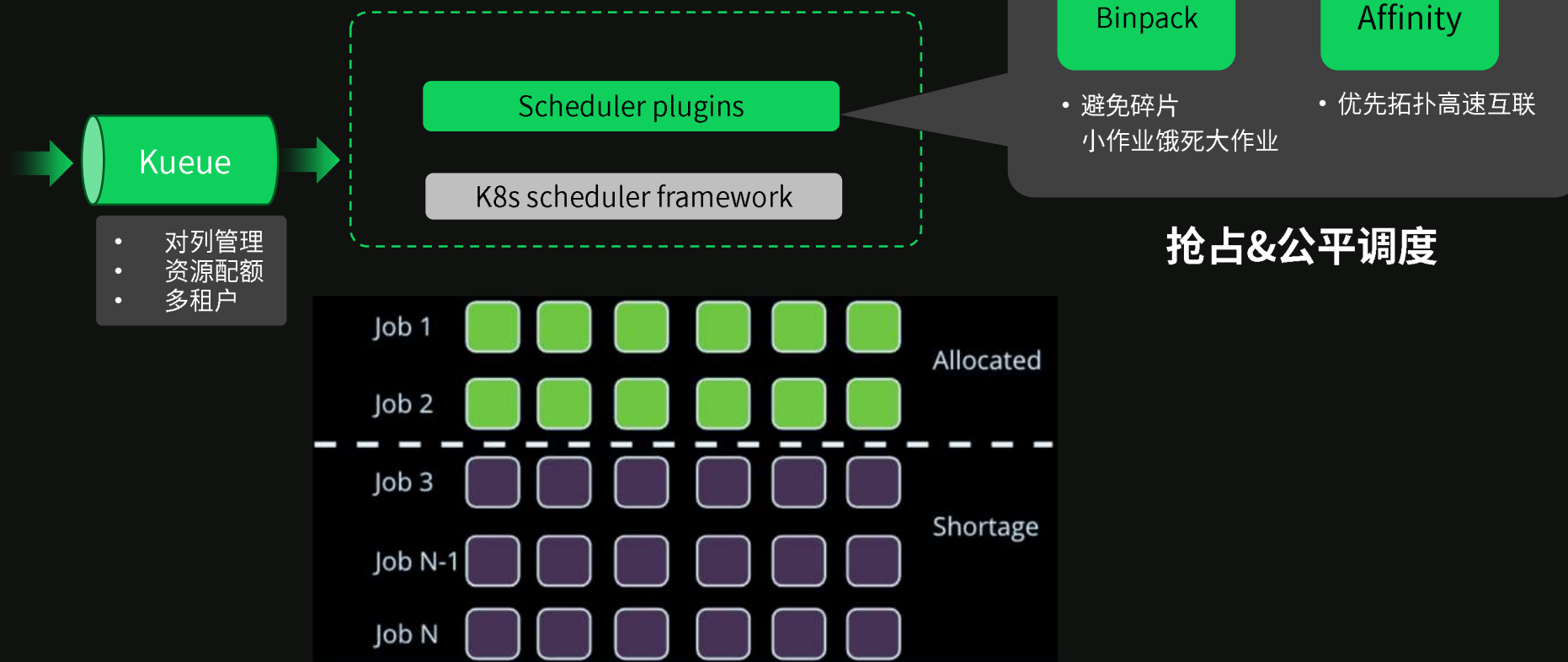
- Kubernetes 对无状态的应用支持很友好，但是有状态应用的支持比较一般
- kubernetes 缺少对批处理作业的支持，如 Job management, queueing, **All-or-nothing scheduling**
- 抢占与公平性 (Preemption & Fairness)



■ 如何解决原生 Kubernetes 挑战

灵活的算力调度，提升 GPU 利用率

- DaoCloud 与 Google 核心贡献的顶级项目 Kueue™
- 应对不同算力场景实现公平调度、亲和、组调度、紧凑等调度算法

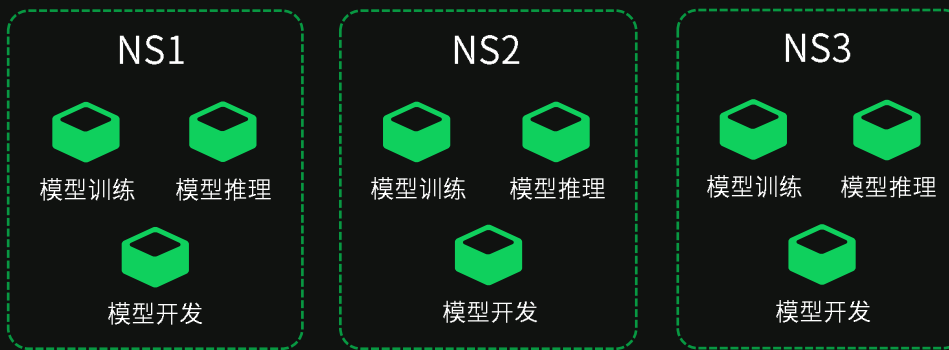


■ d.run 多种方式实现 GPU 共享

让单个任务可以使用更多的 GPU 资源而无需关心单机的 GPU 数量。
同时部署应用时可按照 1% 的算力颗粒度和 1MB 显存颗粒度极致压榨 GPU 资源，有效提高资源利用率，避免资源浪费。

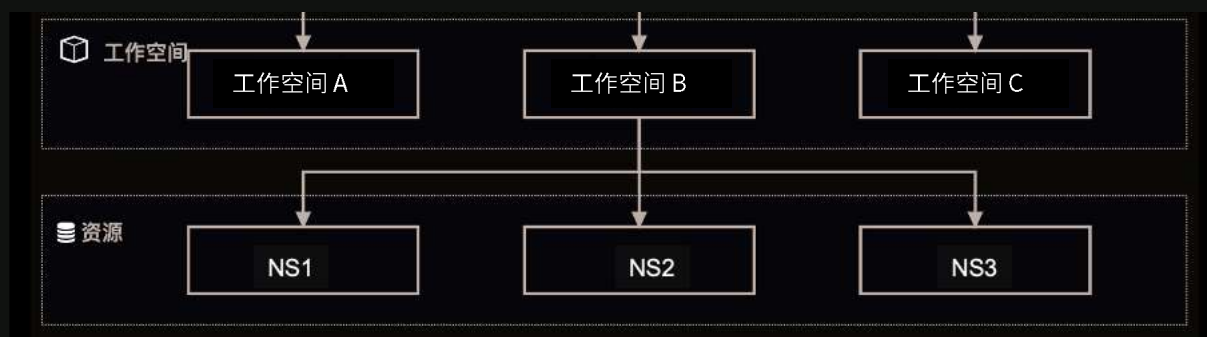
通过网络远程调用 GPU/vGPU 资源进行加速，本地无需 GPU 卡。

工作空间（算力/显存限额）

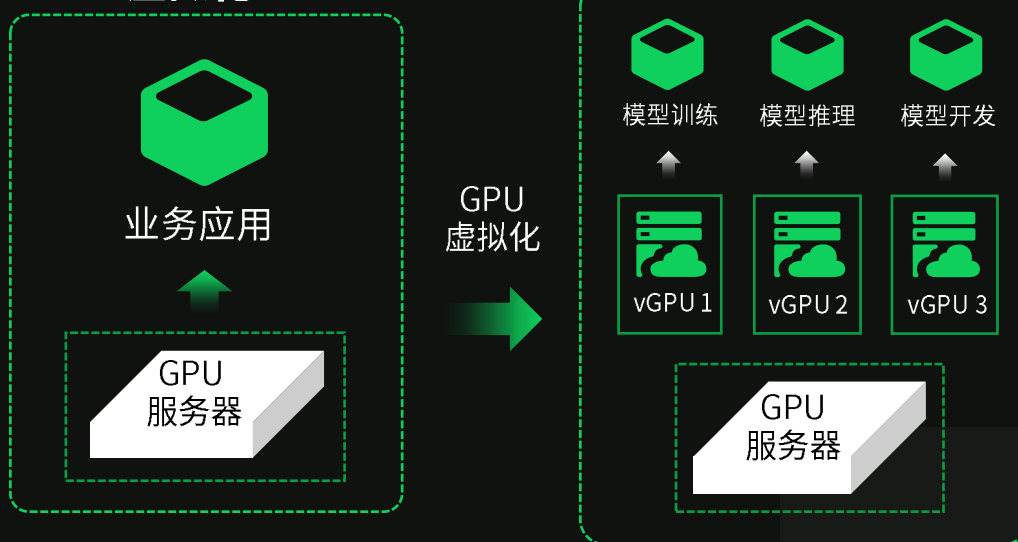


$\text{GPU (NS1+NS2...NSn)} \leq \text{GPU (工作空间 Workspace)}$

$\text{GPU (Pod1+Pod2...Podn)} \leq \text{GPU (命名空间 Namespace)}$

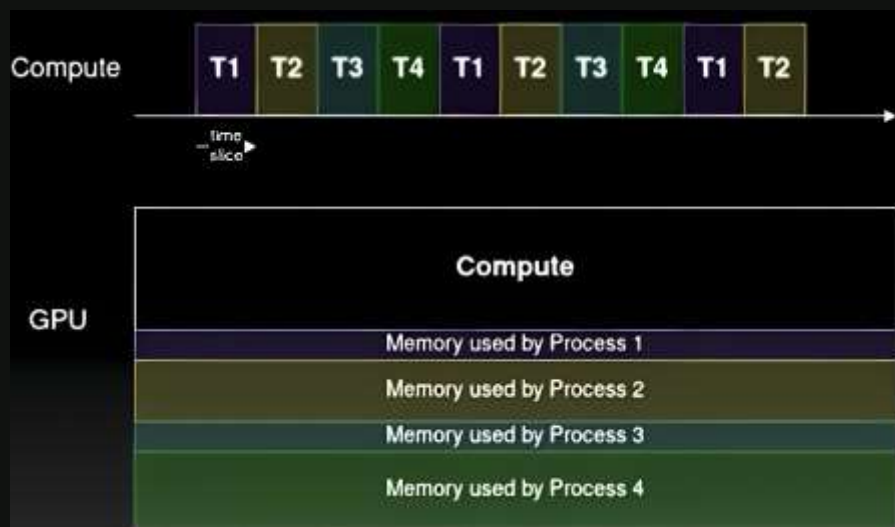


GPU 虚拟化



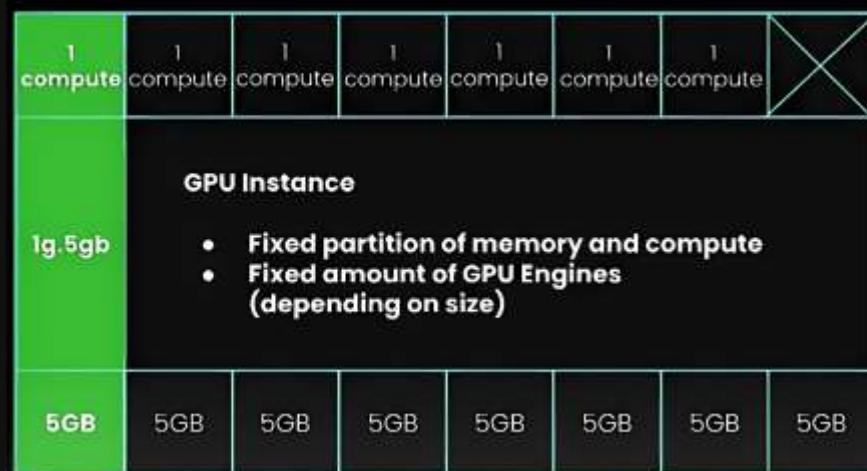
■ GPU 共享：MIG & vGPU

vGPU



	MIG	Time-Slicing
分区类型	Physical	Logical
最大分区	7	Unlimited
SM QoS	Yes	No
内存 QoS	Yes	No
错误隔离	Yes	No
配置更新	Requires Reboot	Dynamic
GPU 支持	H100, A100, A30	Most GPU

MIG



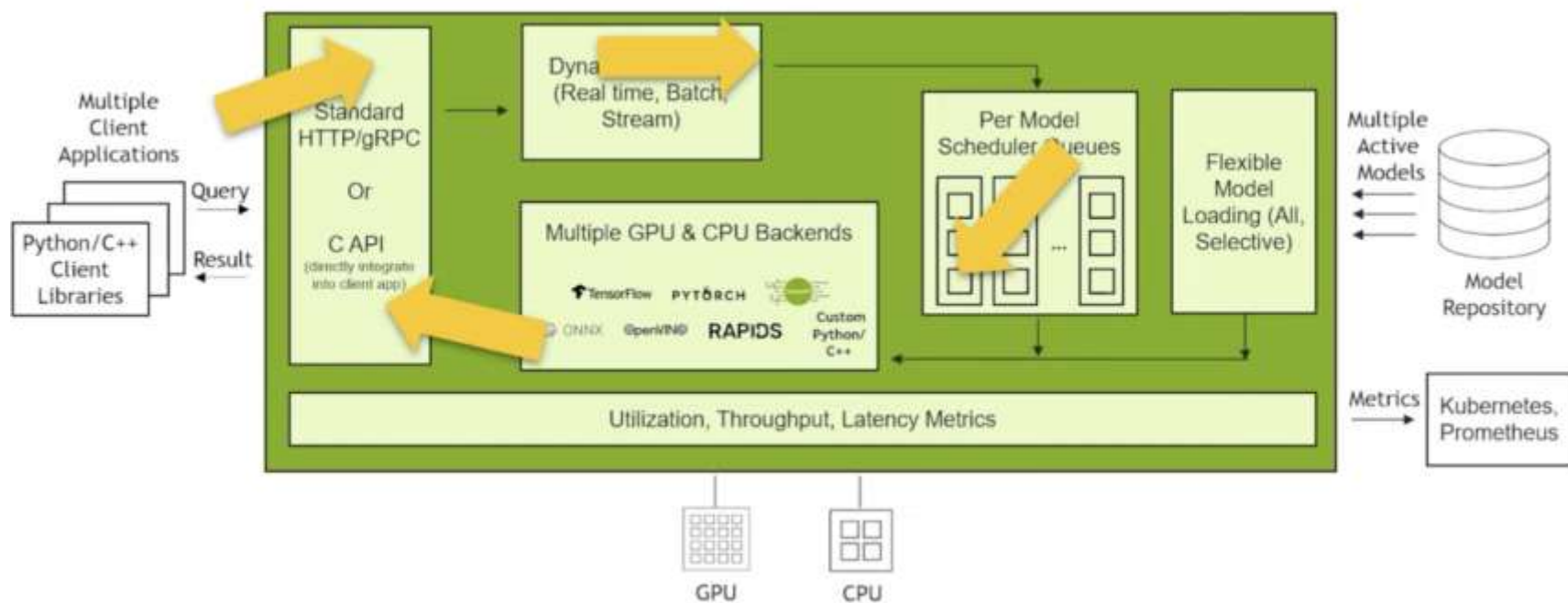
NVIDIA A100 (108SM 40GB)

- 8 x 5GB Memory Slices
- 7 Compute Slices (~1/7 total SM per slice*)

* Combining slices sometimes yields more SM

■ Triton: 优化模型推理部署和管理

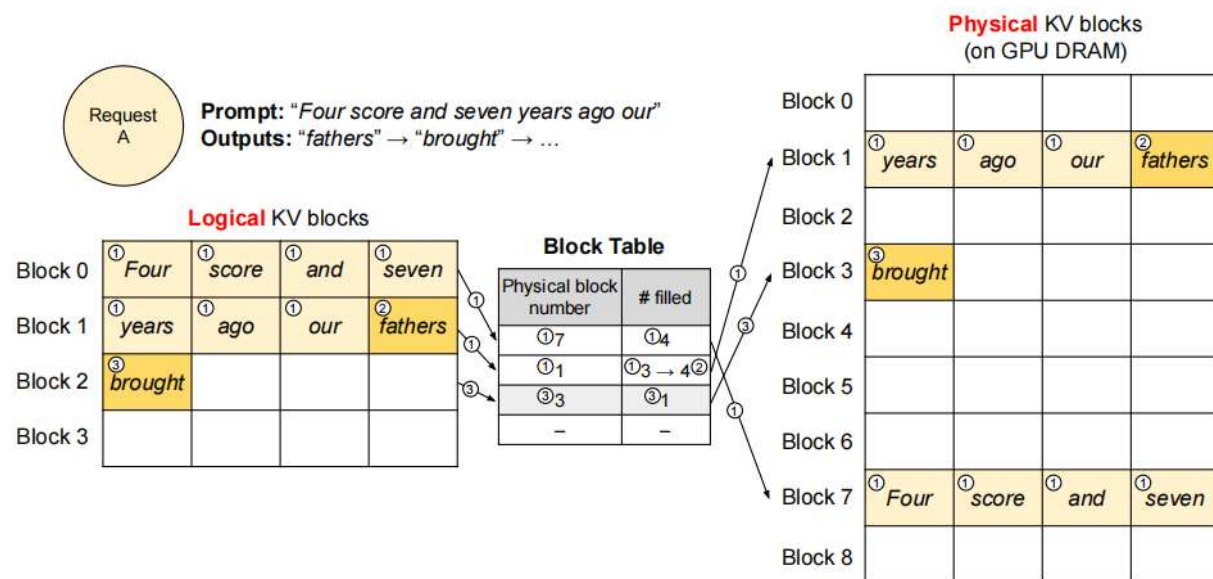
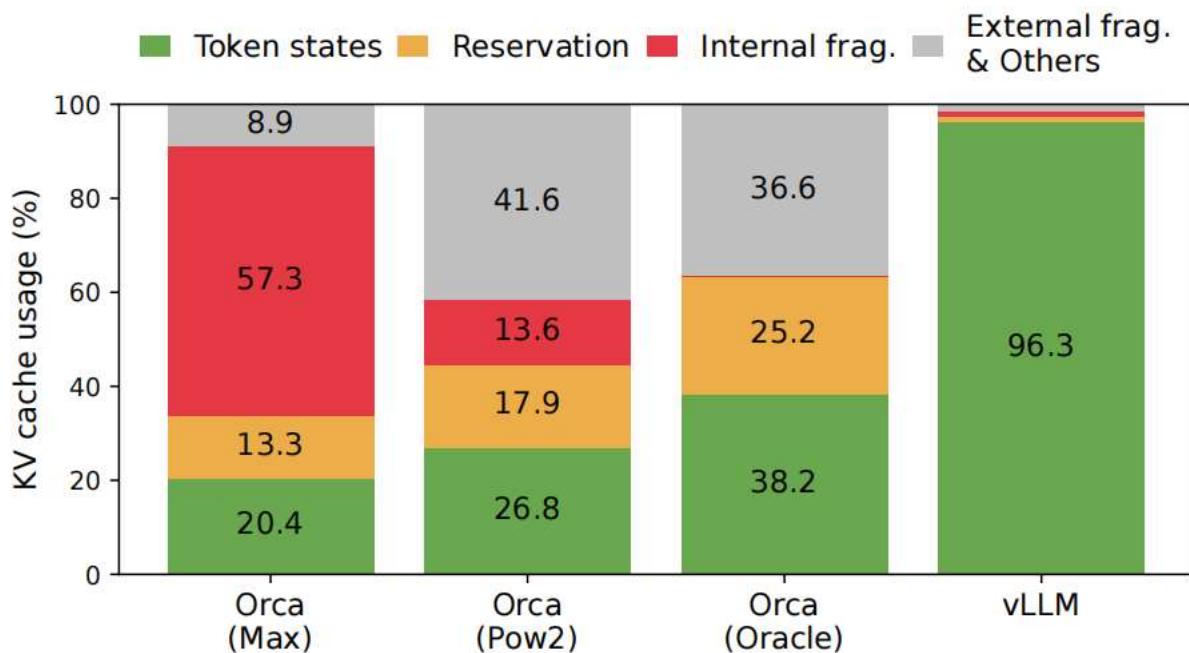
Triton Inference Server 旨在简化生产环境中机器学习模型的部署和管理，同时优化 GPU 和 CPU 资源的使用以实现高效的推理性能。



1. 框架支持多样性
2. 高效的资源利用
3. 模型管道和集成
4. 异步和同步推理支持
5. 动态批处理
6. 多租户和模型版本管理
7. 度量和监控
8. 易于集成

■ vLLM: 优化 GPU 显存使用，提升推理效率

vLLM 是一个专为大型语言模型（LLM）推理加速而设计的开源框架。它通过优化模型结构、推理算法等手段，显著提升了 LLM 的服务效率。



《Efficient Memory Management for Large Language Model Serving with *PagedAttention*》

■ d.run 模型部署和推理优化

AI 模型

ChatGLM

Gemma

Qwen

Llama

Baichuan

Phi

LLaVA

Phi-3-Vision

...

Huggingface

LLM

Vision

模型框架

Pytorch

Tensorflow

ONNX

Python

推理框架

Triton inference server

vLLM

■ 模型推理 - 表单化创建推理服务

← 创建推理服务

1

2

基本信息模型配置

基本配置

名称

dsq

2 - 63 个字符，必须由小写字母、数字字符或“-”组成，并且必须以字母开头及字母或数字字符结尾

部署位置

集群

gpu-cluster

命名空间

chuanjia

推理框架

Triton

队列

搜索

Triton

vLLM

创建队列

优先级

vLLM

选择模型

数据集

创建数据集

模型路径

← 创建推理服务

✓

2

基本信息模型配置

运行配置

托管引擎

PyTorch

输入参数

Data Type

FP16

dims

-1

输出参数

Data Type

FP16

dims

-1

自定义配置

不启用

安全配置

请求认证

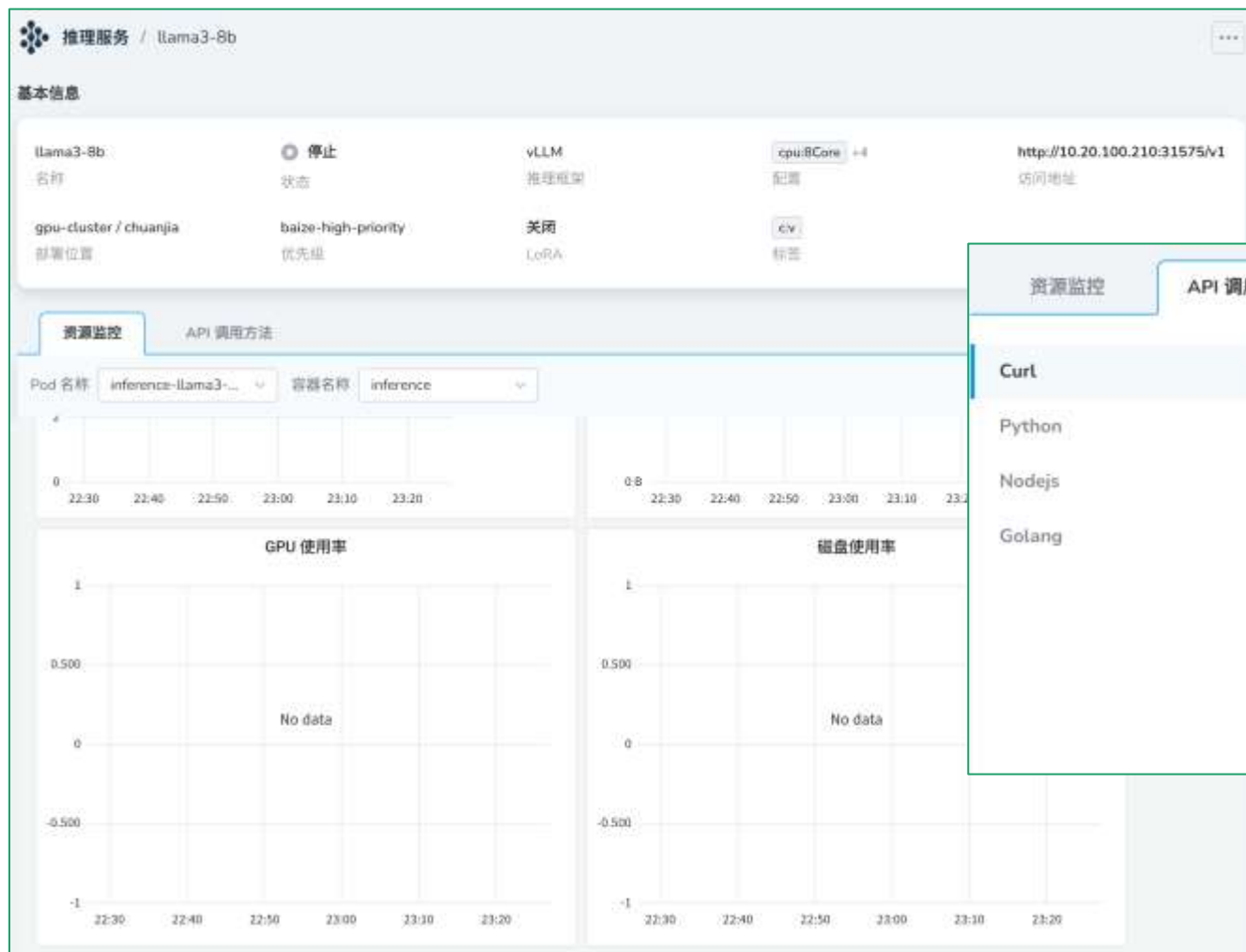
不启用

环境配置

关联环境

创建环境

■ 模型推理 - 推理服务详情



资源监控

API 调用方法

Curl

Python

Nodejs

Golang

支持一键复制代码，并自动替换模型的访问地址到代码块中。

```
curl 'http://10.20.100.210:31575/v1/chat/completions' \
-H "Content-Type: application/json" \
-d '{
  "model": "llama3-8b",
  "messages": [{"role": "user", "content": "Say this is a test!"}],
  "temperature": 0.7
}'
```

■ 通过计量计费，实现 GPU 精打细算使用

- 统计周期：所有报表数据均为按日聚合
- 报表管理
 - 集群、节点、容器组、工作空间、命名空间报表：按照不同维度统计资源（CPU、内存等）使用情况
 - 审计报表：按用户操作、资源操作维度统计操作次数、成功、失败次数
 - 告警报表：统计节点的告警数据，区分紧急、警告、提示级别
- 计量计费
 - 集群、节点、容器组、命名空间、工作空间计费报表：按照不同维度统计资源费用（CPU、内存等费用）
 - 计费配置：配置CPU、内存、存储、GPU等价格及货币单位
- 报表的搜索，开关及导出
 - 所有报表均可按照日期、对应报表字段进行搜索
 - 所有报表数据均可导出为 csv、excel 文件

■ 计量计费功能展示

The screenshot displays the DaoCloud management console. The left sidebar contains navigation menus for '运营管理' (Operation Management) and '计量计费' (Billing). The main area is titled '容器组报表' (Container Group Report). A table lists various container groups with columns for name, namespace, cluster, workspace, and usage metrics. A red box highlights the '导出' (Export) button and the '启用报表' (Enable Report) toggle. The export dropdown menu is open, showing options for CSV and Excel.

容器组报表

时间范围: 最近 1 天

搜索: 容器组名称 / 所属命名空间 / 所属集群 / 所属工作空间

导出: CSV, Excel

容器组名称	所属命名空间	所属集群	所属工作空间	100	0.01
aasd-0	demo	gpu-cluster	baize	100	0.01
analysis-rw-notebook-vscode-gpt2-t...	ruiw	gpu-cluster	ruiw	100	0.69
apitable-plus-697788d668-cctv6	chuanjia-donot-delete	kpanda-global-cluster	mcamel	0	0
app-deployment-demo-675d845cc9...	default	gpu-cluster	baize	0	0
app-deployment-demo-675d845cc9...	default	gpu-cluster	baize	0	0
asd-0	demo	gpu-cluster	baize	100	0.02
baize-agent-cluster-controller-54d8...	baize-system	gpu-cluster	-	0	0
baize-agent-cluster-controller-55f75...	baize-system	mxnet	-	0	0
baize-agent-cluster-controller-5968...	baize-system	rw-gpu-1	-	0	0
baize-agent-cluster-controller-5c96c...	baize-system	gpu-cluster	-	0	0

共 697 项

1 / 70

10 项

■ GPU性能影响因素

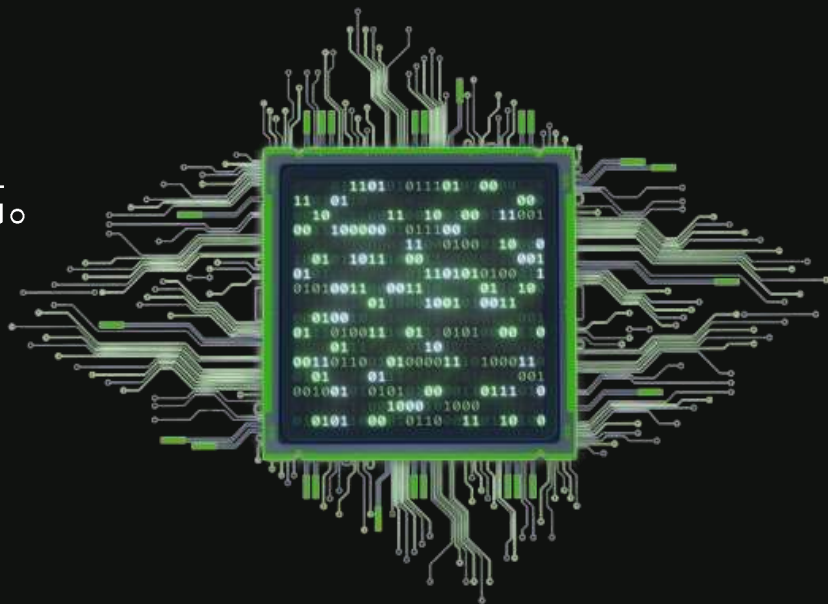
模型大小： 模型参数量越大，所需的显存就越大。

数据量： 数据量越大，训练时间越长，对 GPU 的计算能力要求越高。

预算： GPU 的价格差异较大，需要根据预算选择合适的型号。

性能： 训练速度、推理速度等。

性价比： 性能与价格的比值。



■ 选型建议

•高性能旗舰：

- NVIDIA H100/H800：性能最强的GPU，适合训练超大规模模型，但价格昂贵。
- NVIDIA A100：性能强大，性价比高，是目前最受欢迎的AI训练GPU之一。

•中高端：

- NVIDIA A40/A6000：性能较好，适合训练中等规模模型。
- NVIDIA RTX 3090/3080 Ti：游戏显卡，也可用于AI训练，性价比相对较高。

•入门级：

- NVIDIA RTX 2080 Ti/2080 Super：性能较入门级，适合小规模模型训练或研究。

选择建议：

- 1.确定模型和数据集： 首先明确要训练的模型和数据集的规模，以便评估所需的显存和计算能力。
- 2.评估预算： 根据预算范围，筛选出可能的GPU型号。
- 3.对比性能指标： 比较不同GPU的CUDA核心数、Tensor Core数、显存大小、带宽等指标。
- 4.考虑散热和功耗： 大型模型训练会产生大量热量，需要考虑GPU的散热能力和电源供应。
- 5.参考社区经验： 在选择GPU之前，可以参考其他用户的经验和评测。

■ 模型参数与模型大小关系

- 以 Transformer 结构可训练参数为例 $(12h^2) + Vh$
- 模型使用 FP16/BF16 方式保存和加载模型，那么占用 2b个 Byte
- 模型使用 FP32 方式保存和加载模型，那么占用 4b个 Byte

模型名称	模型层数 h	层数 l	$12h^2$	实际参数数量	模型大小 (FP16)
LLAMA-6B	4096	32	6442450944	6.7B	12 GB
LLAMA-13B	5120	40	12582912000	13.0B	23.4 GB
LLAMA-33B	6656	60	31897681920	32.5B	59.4 GB
LLAMA-65B	8192	80	64424509440	65.2B	120 GB

说明：

- 1 层Transformer的可训练参数量约为 $(12h^2+Vh)$ ：其中 h 为隐藏层维度，l 为层数，V 为词表大小。
- 模型使用 FP16/BF16 方式保存和加载到显存，那么占用 2 个Byte
- 模型使用 FP32 方式保存和加载到显存，那么占用 4 个Byte

■ 思路：从显存占用入手

模型训练

- 在一次训练迭代中，每个训练参数都对应 1 个梯度，2 个优化器状态（Adam）。设模型参数量为 ϕ (FP16)，那么梯度的参数量为 2ϕ (FP32)，Adam 优化器的参数量为 4ϕ (FP32)
- 大模型通常采用混合精度训练中，使用 BF16/FP16 进行前向运算，FP32 反向传播和更新；优化器更新时梯度参数时，使用 FP32 优化器状态。FP32 的梯度和优化器状态空间占用。

训练总内存 = 模型占用 + 梯度占用 + 优化器占用 + 激活占用 + 其他

$>8\phi$	ϕ	2	4	X	1.X
		ϕ	ϕ	ϕ	ϕ

模型推理

- 神经网络推理 (Inference) 阶段，没有优化器状态和梯度信息，也不需要保存中间中间值。因此推理阶段占用的内存要远远小于训练阶段。
- 推理阶段，占用显存大头是模型参数，一次推理过程中，每个可训练参数都对应1个梯度，设模型参数量为 ϕ ，使用float16来进行推理，模型参数占用的显存大概是 2ϕ bytes。

推理总内存 = 模型内存 + 其他内存

$>1.5\phi$	ϕ	$0.X\phi$
------------	--------	-----------



Category	A100	H100	L40S	H200
架构	Ampere	Ada Lovelace	Hopper	Hopper
Release Year	2020	2022	2023	2024
FP64 Performance	9.7 TFLOPS	34 TFLOPS	34 TFLOPS	34 TFLOPS
FP64 (Tensor Core)	19.5 TFLOPS	67		
FP32 Performance	19.5 TFLOPS	67		
TF32 Performance	312 TFLOPS	990		
BFLOAT16/FP16 Performance	624 TFLOPS	1,9		
INT8 Performance	1,248 TOPS	3,9		
GPU Memory	80 GB HBM2e	80		
Memory Bandwidth	2,039 GBps	864		
Availability	Not applicable	NV		
TDP	400W	700		
Form Factor	8.7 MI Gs 10 GB	8.7		
Interconnect	NVLink: 600 GB/s	NV		
NVIDIA HGX Partner	NVIDIA HGX A100 Partner and NVIDIA-Certified Systems	NV and Sys		
AI Enterprise Support	Included	Add-on	Not applicable	Add-on
CUDA Cores	6,912	16,896	18,176	Data not available

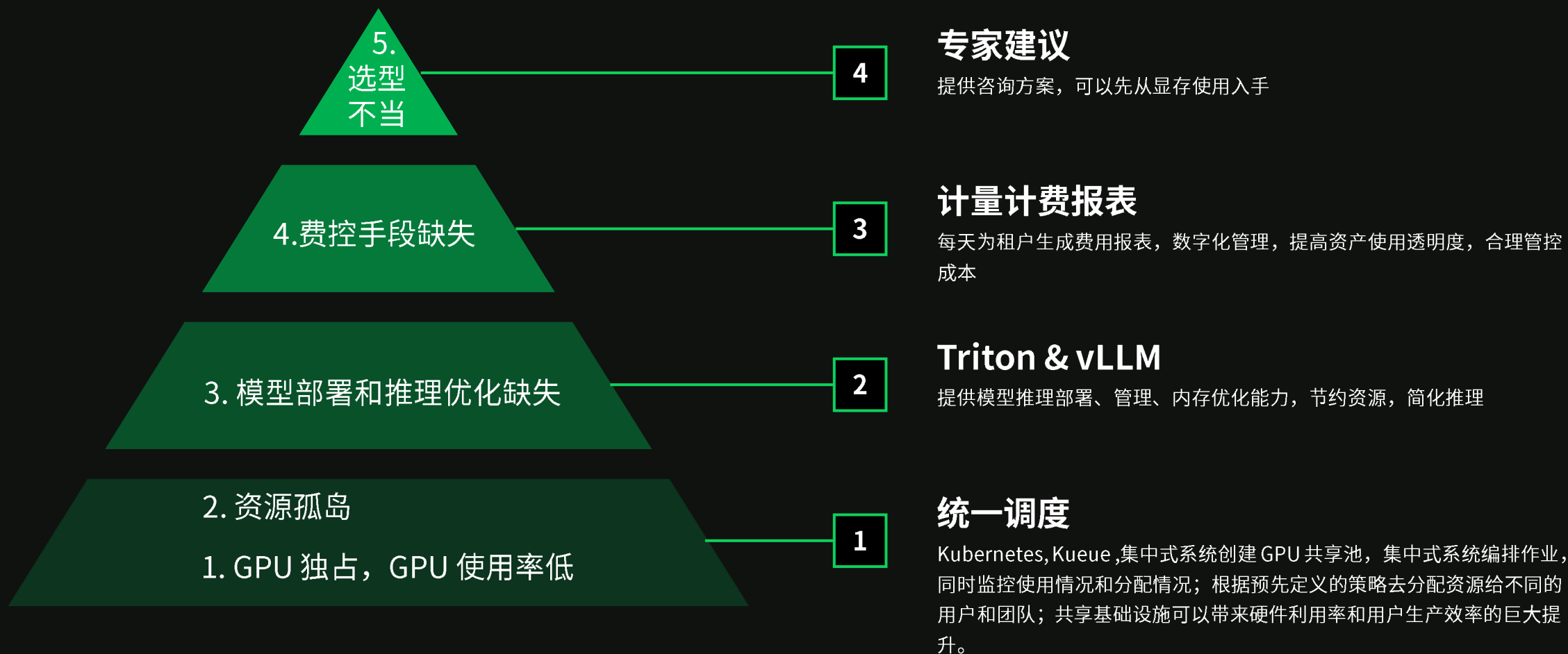
Ampere / 30-Series									
	GeForce RTX 3090 TI	GeForce RTX 3090	GeForce RTX 3080 TI	GeForce RTX 3080	GeForce RTX 3070 TI	GeForce RTX 3070	GeForce RTX 3060 TI	GeForce RTX 3060	GeForce RTX 3050
NVIDIA CUDA Cores	10752	10496	10240	8960 / 8704	6144	5888	4864	3584	2560 / 2304
Boost Clock (GHz)	1.86	1.70	1.67	1.71	1.77	1.73	1.67	1.78	1.78 / 1.76
Memory Size	24 GB	24 GB	12 GB	12 GB / 10 GB	8 GB	8 GB	8 GB	12 GB	8 GB
Memory Type	GDDR6X	GDDR6X	GDDR6X	GDDR6X	GDDR6X	GDDR6	GDDR6	GDDR6	GDDR6
Ada Lovelace / 40-Series									
	GeForce RTX 4090		GeForce RTX 4080		GeForce RTX 4070 TI				
NVIDIA CUDA Cores	16384		9728		7680				
Boost Clock (GHz)	2.52		2.51		2.61				
Memory Size	24 GB		16 GB		12 GB				
Memory Type	GDDR6X		GDDR6X		GDDR6X				
Max Display Resolution	4K at 240Hz or 8K at 60Hz with DSC		4K at 240Hz or 8K at 60Hz with DSC		4K at 240Hz or 8K at 60Hz with DSC				

Part 04

总结

■ 总结

DaoCloud d.run 方案，提供 4 大能力，提升 GPU 使用率





Part 05

问答环节

The background of the slide is a dark, swirling pattern of green and black. The green is a vibrant, slightly neon shade, and the black is a deep, dark charcoal. The pattern consists of fluid, organic shapes that swirl and flow, creating a sense of movement and depth. The overall effect is modern and tech-oriented.

Thanks.