大模型时代星环科技在云原生 AI 基础设施构建的实践

杨超
星环科技资深开发工程师
蔚滢璐
星环科技高级开发工程师

01 基于容器云的星环大语言模型构建

02 异构 GPU 池化管理

03 高速网络管理

04 实践案例

基于容器云的星环大语言模型构建

金融大模型『**无涯**』 和大数据分析大模型『**求索**』

金融大模型 Infinity 无涯

结合星环完整数据生命周期技术的金融行业解决方案

构建六类大模型基础因子集

支撑复合因子策略体系

贯通宏观行业和大类资产分析逻辑

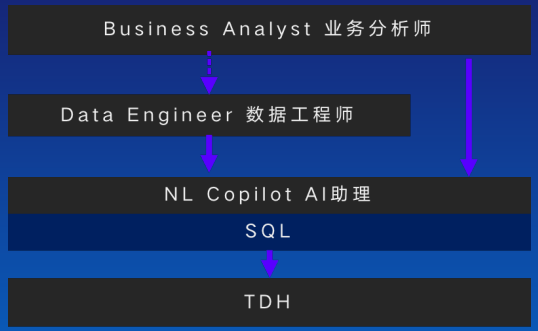
实现事件复盘分析与推演

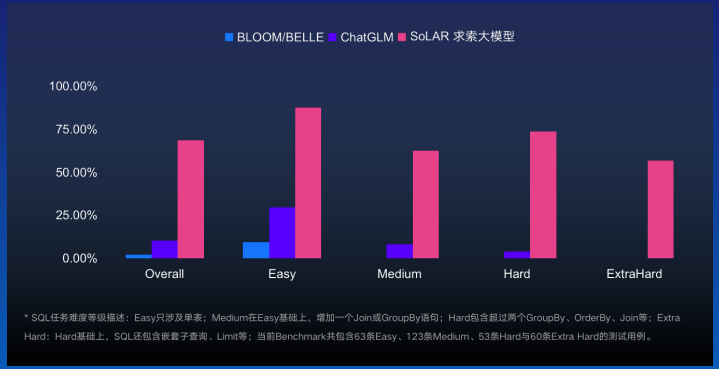
准确理解和合理分析的能力

针对金融行业

大数据分析大模型 SoLar 求索

数据库查询平民化，让非专业用户在无需掌握数据库编程语言的前提下，通过自然语言自由地按需查询数据





大数据分析大模型 SoLar 求索

数据库查询平民化，让非专业用户在无需掌握数据库编程语言的前提下，通过自然语言自由地按需查询数据

场景：熟人网络交叉销售数据分析

提问：

待分析用户满足下列两个条件：

1、是12个月内注册的新用户

2、超过3个朋友买过A基金

现在查询这些用户的用户ID，以及其在最近3个月内贷款总额。

```
select d.cust_id, sum(d.amount) as amount
  from loan d
  join (select a.cust_id as cust_id
          from customer a
          join $cypher<< '
            use graph kg_relation_friend_with_schema; \
            match (c:Person)-[s:FRIEND]->(f:Person)  \
                return
                toInteger(c.cust_id) as cust_id,  \
                toInteger(f.cust_id) as friend_id' >> as b
            on a.cust_id = b.cust_id
          join fund c
            on b.friend_id = c.cust_id
         where c.fund_id = "A"
           and a.register_date > ADD_MONTHS(SYSTIMESTAMP, -12)
      group by a.cust_id
        having count(distinct b.friend_id) > 3) e
    on d.cust_id = e.cust_id
 where d.loan_date >= ADD_MONTHS(SYSTIMESTAMP, -3)
 group by d.cust_id;
```

星环大模型应用开发产品全景图

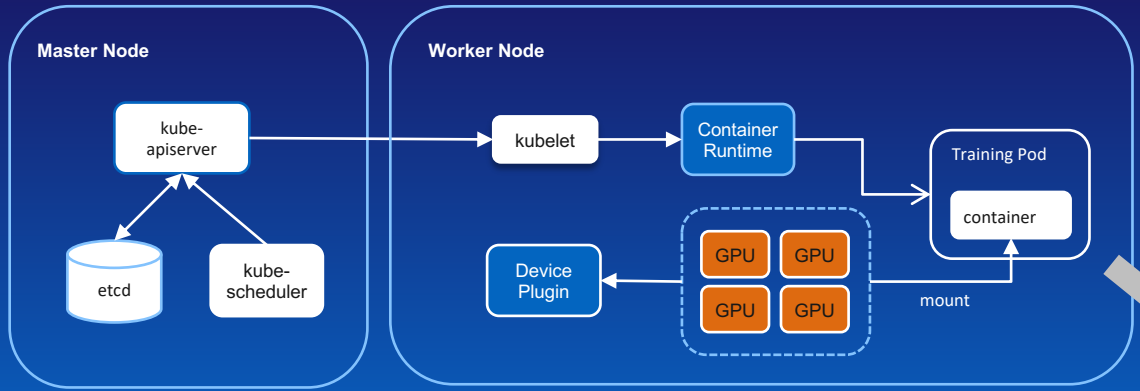
异构 GPU 池化管理

容器云平台 GPU 资源的纳管、使用、调度和监控

GPU 设备虚拟化方案

方案 1： GPU 直通	方案 2： GPU 受控的直通	方案 3： GPU 直通 + API 拦截
可以直接在虚拟机、容器中使用； 性能损耗小；	Nvidia GRID vGPU，Intel GVT-g 技术 Asend VNPU	拦截发送向用户态 driver 或内核态 driver 的 API 调用实现虚拟化
仅支持 1:1 使用场景	需要有内核态的 driver 支持	用户态 driver API 可自定义； 内核态 driver API 拦截用户程序无感知；

□ □ GPU □ □ □ □

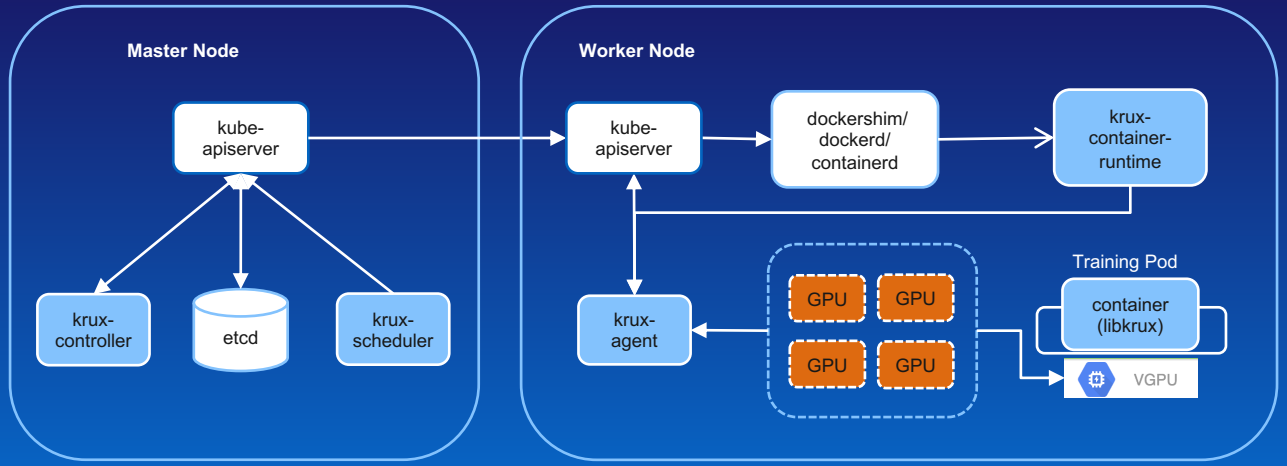


- GPU 设备通过 Device Plugin 接入 kubelet
- 工作负载以个数为单位申请 GPU 资源
- 调度成功后，kubelet 通过容器运行时启动业务容器，并挂载相应的 GPU 设备

```
apiVersion: v1
kind: Pod
metadata:
    name: train-pod
spec:
    containers:
    - name: tf-container
      image: tensorflow:2.4.1-gpu
      resources:
        limits:
          nvidia.com/gpu: 1
```

开源方案的不足

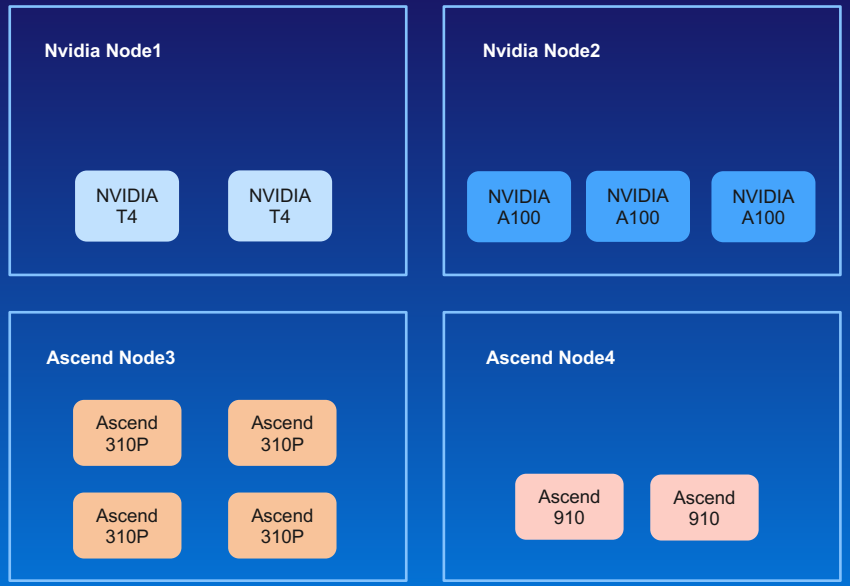
□□□□ GPU □□□□ —— Krux



- **krux-controller:** GPU 资源控制组件，面向业务场景，如 GPU 分组管理等；
- **krux-scheduler:** 基于 Scheduler Framework v2，以插件形式集成到星环自研调度器 venus-scheduler，扩展 GPU 调度策略；
- **krux-agent:** DaemonSet部署，负责工作节点的 GPU 资源探测、监控和管理；
- **krux-container-runtime:** 用于注入容器环境变量和配置文件；
- **libkrux:** 劫持 CUDA API，实现算力和现存切割和限制

星环异构 GPU 池化能力

□□ GPU □□□



□□ □□□ (Nvidia)□ □□ (Ascend)□□□□□□□□□□□□

- Nvidia □□□□ Kepler □□□□□□□□□□
- Ascend □□□□□□□□□ 310□ 310P□ 910□□□

krux-agent□ kubernetes CRD □□□□□□□□□□ GPU□□

```
Name: node1-gpu-0
Node: node1
Device Path: /dev/nv
Model: Tesla T4
Vendor: Nvidia
Memory: 15GiB
Allocated:
  Memory: 3GiB
  UtilPercent: 30%
```

```
Name: node2-gpu-0
Node: node1
Device Path: /dev/nvidia0
Model: NVIDIA A100
Vendor: Nvidia
Memory: 80GiB
Allocated:
  Memory: 3GiB
  UtilPercent: 30%
```

```
Name: node4-npu-0
Node: node1
Device Path: /dev/nvidia0
Model: NVIDIA A100
Vendor: Nvidia
Core: 32
Memory: 15GiB
Allocated:
  Memory: 3GiB
  Core: 4
```

多模式的任务运行策略

独占模式

容器级别的独占使用

支持某个容器独占使用一块或多块 GPU。
推荐用于模型的训练任务。

共享模式

多容器有限制的共享使用

支持多个容器共享使用一块 GPU。
提供算力与显存两个维度的限制。
推荐用于模型的推理任务。

无限制模式

多容器无限制的共享使用

支持多个容器无共享使用一块或多块 GPU，不进行资源限制。
推荐用于模型的开发阶段。

Nvidia GPU 的多模式支持

- transwawrp.io/vgpu-core 配置为 100 及 100 的整数倍，表示为 GPU **独占模式**，该容器可以独自使用一到多块 GPU 设备的算力；
- transwarp.io/vgpu-core 配置为 1-99，表示为 GPU **共享模式**，可以和其它容器共用某块 GPU 设备的算力；
- transwarp.io/vgpu-memory 配置仅在共享模式下生效，限制了容器使用某块 GPU 的显存大小
- transwarp.io/vgpu-core 配置为 100 及 100 的整数倍，同时在 annotation 中声明 **无限制模式**，可以多个容器自由使用一到多块 GPU 设备的算力；

```
apiVersion: v1
kind: Pod
metadata:
  name: gpu-pod
spec:
  containers:
  - name: tf-container
    image: tensorflow/tensorflow:2.4.1-gpu
    resources:
      limits:
        transwarp.io/vgpu-core: 50
        transwarp.io/vgpu-memory: 10Gi
```

Ascend NPU 的多模式支持

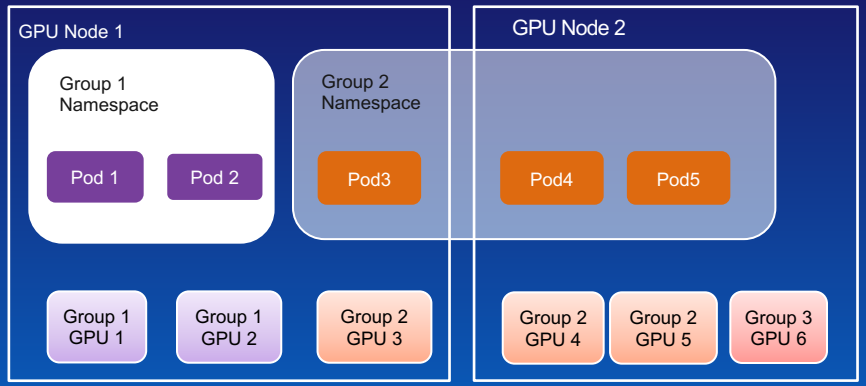
- transwawrp.io/npu-core 配置为 8 及 8 的整数倍，表示为 **独占模式**，该容器可以独自使用一到多块 NPU 设备的算力；
- transwarp.io/npu-core 配置为 1、2、4，表示为 **共享模式**，可以和其它容器共用某块 GPU 设备的算力；
- transwarp.io/npu-core 配置为 0，表示为 **无限制模式**；
- 昇腾不同类型的 NPU 使用姿势有差别，需要配合指定 model 使用；

```
apiVersion: v1
kind: Pod
metadata:
  name: train-npu-pod
  annotations:
    gpu.scheduling.transwarp.io/models: “Ascend 310P”
spec:
  containers:
  - name: npu-container
    image: ascend-tensorflow:22.0.0-centos7
    resources:
      limits:
        transwarp.io/npu-core: 4
```

不同模式下的调度策略

单节点共享任务	单机多卡任务	多机多卡任务
可用资源检查 (GPUResourceFit)	加速网络拓扑感知（ NVLink） (NetworkTopologyAwareness)	加速网络拓扑感知（InfiniBand/NVSwitch） (NetworkTopologyAwareness)
最小碎片优先 (MinFragment)	训练数据本地性感知 (DataLocalityAwareness)	组调度 (CoScheduling)
最少容器数量优先 (MinContainerNumber)	GPU 分组选择 (GPUGroupSelector)	调度资源队列 (ResourceQueue)

GPU Group □ □ □ □ □ □ □



- GPU □ □ □ □ □
  - □ □ □ □ □ □ □ □ GPU □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ GPU □ □ □ □ □ □ □ default □ □
  - □ □ □ □ □ □ □ GPU □ □ □ □ □ □ □ □ □
- GPUGroupSelector □ □ □ □
  - □ □ □ □ □ □ □ pod □ □ □ □ □ □ GPU□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ default □ □

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: test-gpugroup
  namespace: test
spec:
  replicas: 2
  template:
    metadata:
      annotations:
        gpu.scheduling.transwarp.io/group: gpugroup-sample
    spec:
      ...
```

```
apiVersion:
resources.transwarp.io/v1alpha1
kind: GPUGroup
metadata:
name: gpugroup-sample
spec:
  gpus:
  - GPU 1
  - GPU 2
```

多维度指标监控

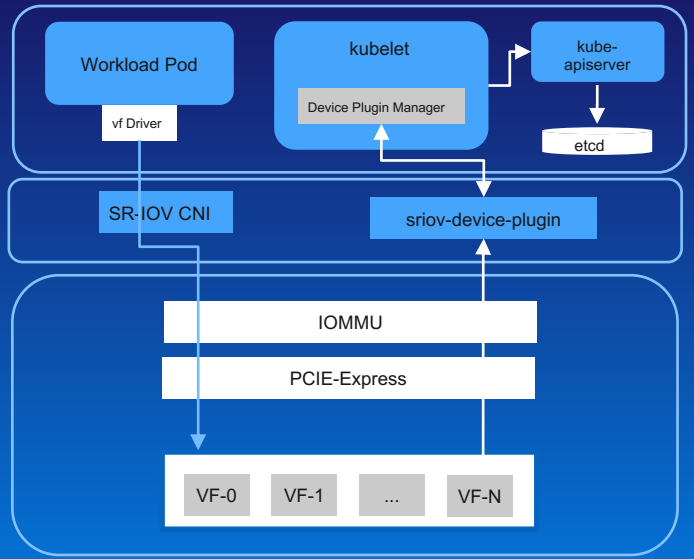
支持单个 Pod 维度的 GPU 资源监控：
可以查看某个 Pod 对于 GPU 的使用情况；

支持单个 Node 维度的 GPU 资源监控：
可以查看某节点每块 GPU 的使用情况；

高速网络管理

容器云平台 InfiniBand 网络资源管理和使用

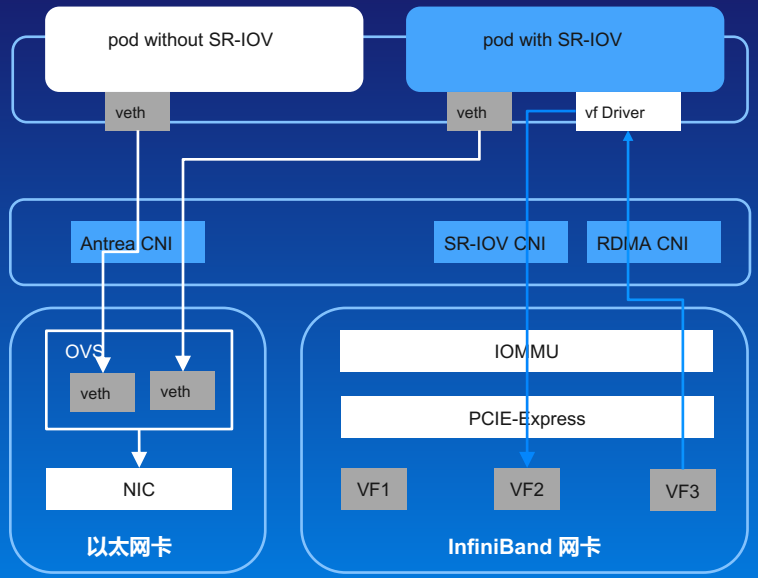
基于 SR-IOV 的 InfiniBand 网络管理



- 支持 InfiniBand 管理组件的一键部署
- 支持vf 的切割和配置，并保障配置的持久化
- 支持 InfiniBand 网卡探测和状态监控

```
{
  "cpu": "6540",
  "ephemeral-storage": "203911331681",
  "hugepages-1Gi": "0",
  "hugepages-2Mi": "0",
  "intel.com/mlnx_sriov_rdma": "127",
  "memory": "14635360320Ki",
  "pods": "110",
  "transwarp.io/vgpu-core": "200",
  "transwarp.io/vgpu-memory": "160Gi"
}
```

Pod 使用高速网络

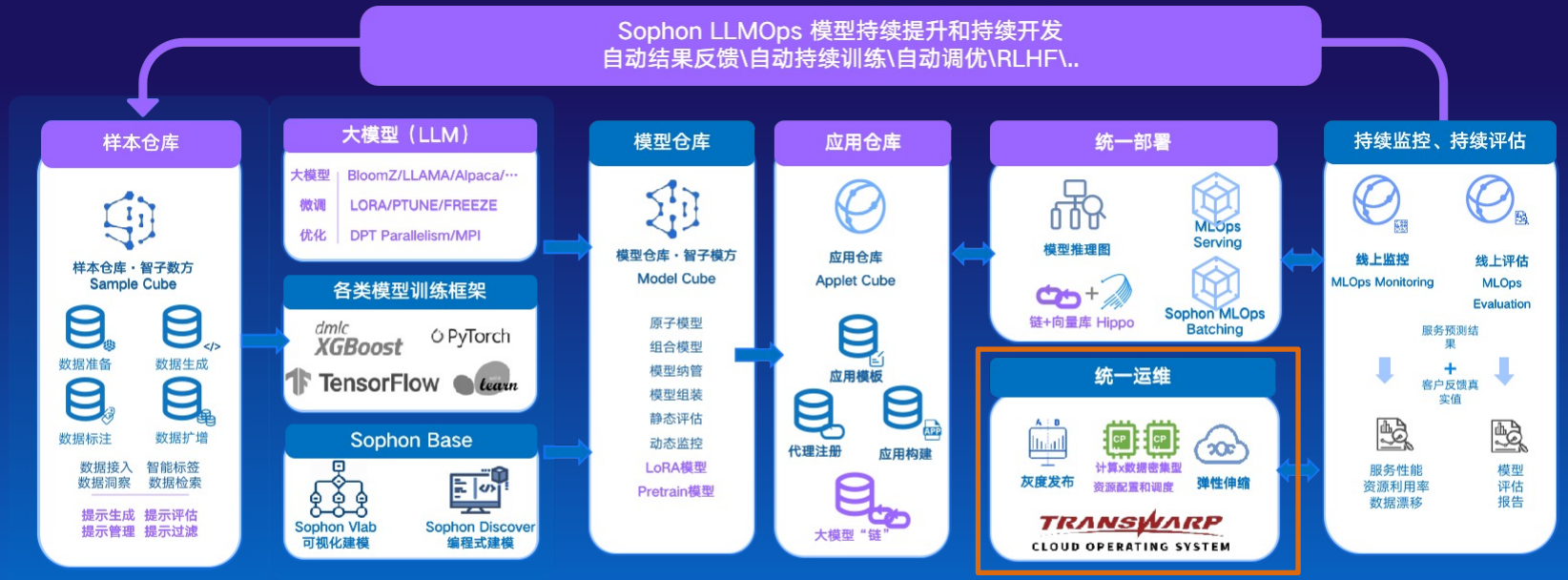


- 支持 Pod 通过第二块网卡申请 virtual function 资源

```
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
  name: performance-testly-ib
spec:
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: ib-secondarynetwork
    spec:
        resources:
          limits:
            intel.com/mlnx_sriov_rdma: "1"
          requests:
            intel.com/mlnx_sriov_rdma: "1"
```

实践案例

星环 Sophon LLMOps 工具链



- 模型易获得
- 样本易管理

- 模型易管理
- 原型易开发

- 效果易迭代
- 系统易运维

Thanks