ModelArts

Lite Cluster User Guide

Issue 01

Date 2025-08-20

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: https://www.huaweicloud.com/intl/en-us/

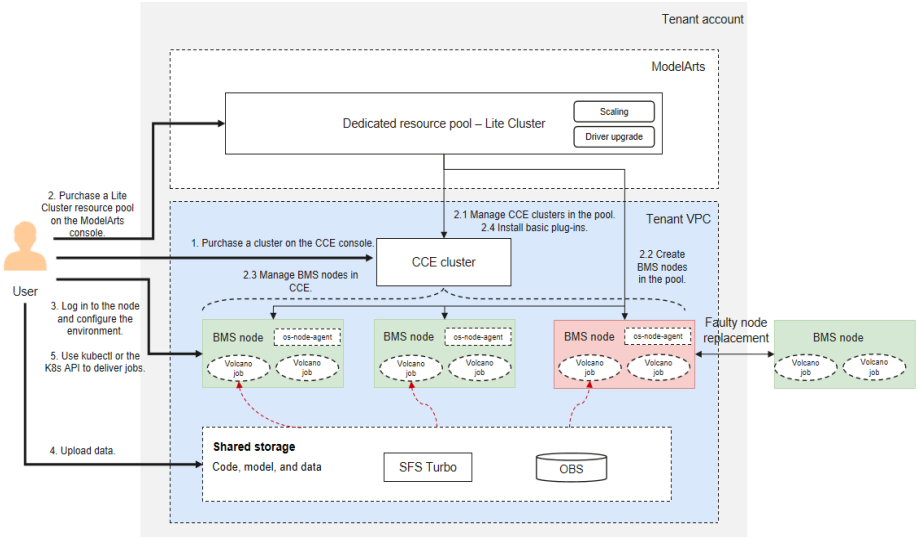
Contents

1 Before You Start

1.1 Using Lite Cluster

ModelArts Lite Cluster offers hosted Kubernetes clusters with pre-installed AI development and acceleration plug-ins. These elastic clusters allow you to access AI resources and tasks in a cloud-native environment. You can directly manage nodes and Kubernetes clusters within the resource pools. This document shows how to get started.

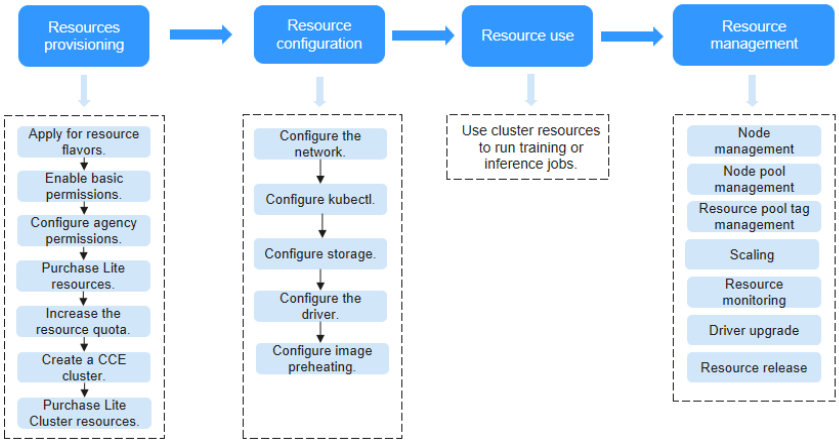
Figure 1-1 Resource pool architecture



This figure shows Lite Cluster architecture. To use Lite Cluster, start by purchasing a CCE cluster. Lite Cluster then manages resource nodes within this CCE cluster. After you purchase a Lite cluster on the ModelArts console, ModelArts manages the CCE cluster within a resource pool and creates compute nodes (BMSs/ECSs) based on the specifications you set. These nodes are then managed by CCE, and ModelArts installs necessary plug-ins (such as npuDriver and os-node-agent) in the CCE cluster. Once you have acquired a Lite Cluster resource pool, you can configure resources and upload data to the cloud storage service. When you

require cluster resources, you can use the kubectl tool or Kubernetes APIs to submit jobs. Additionally, ModelArts offers scaling and driver upgrade to streamline cluster resource management.

Figure 1-2 Usage process



To use Lite Cluster, follow these steps:

1. Resource subscription: Apply for the required specifications, configure permissions, and purchase Lite Cluster resources on the ModelArts console. For details, see **Enabling Lite Cluster Resources**.
2. Resource configuration: After acquiring resources, set up network, storage, and drivers. For details, see **Configuring Lite Cluster Resources**.
3. Resource usage: Once configured, use cluster resources for training and inference. For details, see **Using Lite Cluster Resources**.
4. Resource management: Lite Cluster provides scaling and driver upgrades. You can manage resources on the ModelArts console. For details, see **Managing Lite Cluster Resources**.

Table 1-1 Terms

Term	Description
Container	Containers, rooted in Linux, are lightweight virtualization technologies that isolate processes and resources. Docker popularized containers by making them portable across different machines. It simplifies the packaging of both applications and the applications' repository and dependencies. Even an OS file system can be packaged into a simple portable package that can be used on any other machine that runs Docker.
Kubernetes	Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications. To use Lite Cluster, familiarity with Kubernetes is essential. For details, see **Kubernetes Basics**.

Term	Description
CCE	Cloud Container Engine (CCE) is a Kubernetes cluster hosting service for enterprises. It manages containerized applications and offers scalable, high-performance solutions for deploying and managing cloud native applications.
BMS	Combining VM scalability with physical server performance, BMS provides dedicated cloud servers. These servers are designed to meet the demands of computing performance and data security for core databases, critical applications, high-performance computing (HPC), and big data.
ECS	Elastic Cloud Server (ECS) provides scalable, on-demand cloud servers for secure, flexible, and efficient application environments, ensuring reliable, uninterrupted services.
os-node-agent	The os-node-agent plug-in is installed by default on ModelArts Lite Kubernetes cluster nodes, allowing for node management. For example: • Driver upgrades: The plug-in downloads and updates or rolls back driver versions. • Fault detection: It periodically checks for node faults. • Metric collection: The plug-in gathers key monitoring data, such as GP and NPU usage, and sends it to AOM on the tenant side. • Node O&M: After authorization, the plug-in runs diagnosis scripts for fault identification and demarcation.

1.2 High-Risk Operations

When you perform operations on ModelArts Lite Cluster resources on the CCE, ECS, or BMS console, certain resource pool functions may be abnormal. The table below shows common risky operations.

Risky operations fall into three levels:

- High: Such operations may cause service failures, data loss, system maintenance failures, and system resource exhaustion.
- Medium: Such operations may cause security risks and reduce service reliability.
- Low: Such operations include high-risk operations other than those of a high or medium risk level.

Table 1-2 Operations and risks

Object	Operation	Risk	Severity	Solution
Cluster	Upgrade, modify, hibernate, or delete clusters.	These operations may impact basic ModelArts functions, including resource pool management, node management, scaling, and driver upgrades	High	These operations cannot be undone.
Node	Unsubscribe, remove, shut down, manage taints, or switch or reinstall OS.	These operations may impact basic ModelArts functions, including node management, scaling, driver upgrades, and data loss of local disks.	High	These operations cannot be undone.
	Modify a network security group.	These operations may impact basic ModelArts functions, including node management, scaling, and driver upgrades	Medium	If needed, revert to the original data.
Network	Modify or delete the CIDR block associated with a cluster.	These operations impact basic ModelArts functions, including node management, scaling, and driver upgrades	High	These operations cannot be undone.
Plug-in	Upgrade or uninstall the GPU-beta plug-in.	The GPU driver may be abnormal.	Medium	Roll back the version and reinstall the plug-in.
	Upgrade or uninstall the huawei-npu plug-in.	The NPU driver may be abnormal.	Medium	Roll back the version and reinstall the plug-in.
	Upgrade or uninstall the volcano plug-in.	Job scheduling may be abnormal.	Medium	Roll back the version and reinstall the plug-in.
	Uninstall the ICAgent plug-in.	Logging and monitoring may be abnormal.	Medium	Roll back the version and reinstall the plug-in.

Object	Operation	Risk	Severity	Solution
Helm	Upgrade, roll back, or uninstall os-node-agent.	Driver upgrades, fault detection, metric collection, and node O&M are abnormal.	High	Contact Huawei Cloud technical support to reinstall os-node-agent.
	Upgrade, roll back, or uninstall rdma-sriov-dev-plugin.	The use of RDMA NICs in containers may be affected.	High	Contact Huawei Cloud technical support to reinstall rdma-sriov-dev-plugin.

1.3 Software Versions Required by Different Models

A resource pool for elastic clusters can use either Elastic Bare Metal Servers (BMSs) or Elastic Cloud Servers (ECSs) as nodes. Each node model has its own operating system (OS) and compatible CCE cluster versions. This document outlines the necessary software versions for each model to simplify image creation and software upgrades.

CCE Cluster Maintenance Policy

The CCE cluster used by ModelArts Lite Cluster belongs to the user who has full control over the cluster.

- If your Lite Cluster uses the EOS CCE cluster, upgrade the cluster to the version recommended by ModelArts in time according to the lifecycle policy released by CCE.

  For details about how to upgrade a CCE cluster, see **Cluster Upgrade Overview**.

  For details about CCE cluster version policies, see **Cluster Version Release Notes**.
- If you have any technical problems related to CCE clusters in the Lite Cluster scenario, submit a service ticket to contact CCE technical support for troubleshooting.

Software Versions Required by BMSs

Table 1-3 BMS

Type	Card Type	RDMA Network Protocol	OS	Applicable Scope	Dependent Plug-in
NPU	ascend-snt9b	RoCE	• OS: EulerOS 2.10 64-bit (recommended) • Kernel version: 4.19.90-vhulk2211.3.0.h1543.eulerosv2r10.aarch64 • Architecture type: aarch64	• Cluster type: CCE Standard • Cluster version: v1.23 (v1.23.5-r0 or later), v1.25, v1.28, or v1.31 (recommended) • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: container tunnel network or VPC • Cluster forwarding mode: iptables or ipvs	• huawei-npu • volcano For details about the plug-in version mapping, see **Table 1-5**.
		RoCE	• OS: Huawei Cloud EulerOS 2.0 64-bit • Kernel version: 5.10.0-60.18.0.50.r865_35.hce2.aarch64 • Architecture type: aarch64	• Cluster type: CCE Turbo • Cluster version: v1.23, v1.25, v1.28, or v1.31 (recommended) • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: ENI • Cluster forwarding mode: iptables or ipvs	

Type	Card Type	RDMA Network Protocol	OS	Applicable Scope	Dependent Plug-in
	ascend-snt9	RoCE	• OS: EulerOS 2.8 64-bit • Kernel version: 4.19.36-vhulk1907.1.0.h619.eulerosv2r8.aarch64 • Architecture type: aarch64	• Cluster type: CCE Standard or Turbo • Cluster version: v1.23 (v1.23.5-r0 or later), v1.25, or v1.28 (recommended) • Cluster scale: 50, 200, 1,000, or 2,000 • Cluster network mode: container tunnel network, VPC, or ENI • Cluster forwarding mode: iptables or ipvs	

Type	Card Type	RDMA Network Protocol	OS	Applicable Scope	Dependent Plug-in
GPU	gp-ant8	RoCE	• OS: EulerOS 2.10 64-bit • Kernel version: 4.18.0-147.5.2.15.h1109.eulerosv2r10.x86_64 • Architecture type: x86	• Cluster type: CCE Standard or Turbo • Cluster version: v1.23, v1.25, v1.28, or v1.31 (recommended) • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: container tunnel network or VPC Distributed training only supports container tunnel network. • Cluster forwarding mode: iptables or ipvs	• gpu-beta • rdma-sriov-dev-plugin For details about the plug-in version mapping, see **Table 1-5**.

Type	Card Type	RDMA Network Protocol	OS	Applicable Scope	Dependent Plug-in
	gp-ant1	RoCE	• OS: EulerOS 2.10 64-bit • 4.18.0-147.5.2.15.h1109.eulerosv2r10.x86_64 • Architecture type: x86	• Cluster type: CCE Standard or Turbo • Cluster version: v1.23, v1.25, v1.28, or v1.31 (recommended) • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: container tunnel network or VPC Distributed training only supports container tunnel network. • Cluster forwarding mode: iptables or ipvs	

Type	Card Type	RDMA Network Protocol	OS	Applicable Scope	Dependent Plug-in
	gp-vnt1	RoCE|IB	• OS: EulerOS 2.9 64-bit (used only for p6 and p6s flavors in Shanghai 1) • Kernel version: 147.5.1.6.h1099.eulerosv2r9.x86_64 • Architecture type: x86	• Cluster type: CCE Standard • Cluster version: v1.23, v1.25, v1.28, or v1.31 (recommended) • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: container tunnel network or VPC Distributed training only supports container tunnel network. • Cluster forwarding mode: iptables or ipvs	
			• OS: EulerOS 2.9 64-bit (recommended) • Kernel version: 4.18.0-147.5.1.6.h841.eulerosv2r9.x86_64 • Architecture type: x86		

- Remote direct memory access (RDMA) is a direct memory access from the memory of one computer into that of another without involving either one's operating system.
- RDMA over Converged Ethernet (RoCE) is a network protocol which allows RDMA over an Ethernet network.
- InfiniBand (IB) is a computer networking communications standard used in high-performance computing. It is used for data interconnect both among and within computers.

Software Versions Required by ECSs

Table 1-4 ECS

Type	Card Type	OS	Applicable Scope	Dependent Plug-in
NPU	ascend-snt3p-300i	• OS: Huawei Cloud EulerOS 2.0 64-bit • Architecture type: x86 or Arm	• Cluster type: CCE Standard • Cluster version: v1.23 (v1.23.5-r0 or later), v1.25, v1.28, or v1.31 (recommended) • Cluster scale: 50, 200, 1,000, or 2,000 • Cluster network mode: container tunnel network or VPC • Cluster forwarding mode: iptables or ipvs	• huawei-npu • volcano For details about the plug-in version mapping, see **Table 1-5**.
		• OS: EulerOS 2.9 • Architecture type: x86	• Cluster type: CCE Standard or Turbo • Cluster version: v1.23 (v1.23.5-r0 or later), v1.25, or v1.28 (recommended) • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: container tunnel network, VPC, or ENI • Cluster forwarding mode: iptables or ipvs	

Type	Card Type	OS	Applicable Scope	Dependent Plug-in
	ascend-snt3	• OS: EulerOS 2.5 • Architecture type: x86	• Cluster type: CCE Standard • Cluster version: v1.23 or v1.25 • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: container tunnel network or VPC • Cluster forwarding mode: iptables or ipvs	
		• OS: EulerOS 2.8 • Architecture type: Arm	• Cluster type: CCE Standard • Cluster version: v1.23 or v1.25 • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: container tunnel network or VPC Cluster forwarding mode: iptables or ipvs	
GPU	gp-vnt1	• OS: EulerOS 2.9 • Architecture type: x86	• Cluster type: CCE Standard • Cluster version: v1.23, v1.25, v1.28, or v1.31 (recommended) • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: container tunnel network or VPC • Cluster forwarding mode: iptables or ipvs	• gpu-beta • rdma-sriov-dev-plugin For details about the plug-in version mapping, see **Table 1-5**.

Type	Card Type	OS	Applicable Scope	Dependent Plug-in
	gp-ant03	• OS: EulerOS 2.9 • Architecture type: x86	• Cluster type: CCE Standard • Cluster version: v1.23, v1.25, v1.28, or v1.31 (recommended) • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: container tunnel network or VPC • Cluster forwarding mode: iptables or ipvs	
	gp-ant1-pcie40	• OS: EulerOS 2.9 • Architecture type: x86	• Cluster type: CCE Standard • Cluster version: v1.23, v1.25, v1.28, or v1.31 (recommended) • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: container tunnel network or VPC • Cluster forwarding mode: iptables or ipvs	

Type	Card Type	OS	Applicable Scope	Dependent Plug-in
	gp-tnt004	• OS: EulerOS 2.9 • Architecture type: x86	• Cluster type: CCE Standard • Cluster version: v1.23, v1.25, v1.28, or v1.31 (recommended) • Cluster scale: 50, 200, 1000, or 2000 • Cluster network mode: container tunnel network or VPC • Cluster forwarding mode: iptables or ipvs	

Mapping Between Drivers, Plug-in Versions, and CCE Cluster Versions

Table 1-5 Mapping between driver and CCE cluster versions

Type	Driver	Driver Version	Matched CCE Cluster Version	Applicable Scope	Plug-in Function
npuDriver	npu-driver	7.1.0.9.220-23.0.6 (recommended) 7.1.0.7.220-23.0.5 7.1.0.5.220-23.0.3	None	NPU (snt9b)	Upgrades and rolls back NPU drivers.
gpuDriver	gpu-driver	515.65.01 (recommended) 510.47.03 470.182.03 470.57.02	None	GPU	Upgrades and rolls back GPU drivers. The plug-in depends on the gpu-beta version.

Table 1-6 Mapping between plug-in versions and CCE cluster versions

Plug-in	Plug-in Version	Matched CCE Cluster Version	Applicable Scope	Plug-in Function
gpu-beta	2.7.63 (recommended)	v1.(28|31).*	GPU	Allows containers to use GPU devices.
	2.6.4	v1.28.*		
	2.0.48	v1.(23|25).*		
huawei-npu	2.1.53 (recommended)	v1.(23|25|28|31).*	NPU	Allows containers to use Huawei NPU devices.
	2.1.22	v1.(23|25|28).*		
volcano	1.16.8 (recommended)	v1.(23|25|28|31).*	NPU	Kubernetes-based batch processing platform.
	1.15.8	v1.(23|25|28).*		
os-node-agent	7.0.0	None	None	OS plug-in, which is used for fault detection.
icagent	default	The matched CCE version is installed by default.	None	CCE basic component, which is used for logging and monitoring.

2 Enabling Lite Cluster Resources

ModelArts Lite Cluster is a dedicated resource pool in Huawei Cloud ModelArts. Targeting at Kubernetes resource users, it provides hosted Kubernetes clusters, pre-installs mainstream AI development plug-ins and Huawei-developed acceleration plug-ins, and provides users with AI native resources and tasks in cloud native mode. You can directly perform operations on nodes and Kubernetes clusters in the resource pool, applicable to scenarios where cloud native resources are required.

Differences between ModelArts Lite Cluster and ModelArts Standard resource pools:

- ModelArts Lite resource pool: You can directly perform operations on nodes and Kubernetes clusters, applicable to scenarios where cloud native resources are required.
- ModelArts Standard resource pool: Provides upper-layer capabilities such as training, inference, and development environments, suitable for one-stop development.

This section describes how to enable a Lite Cluster resource pool.

Billing

- After Lite Cluster resources are enabled, compute resources are charged. For Lite Cluster resource pools, only the yearly/monthly billing mode is supported. For details, see **Table 2-1**.

Table 2-1 Billing items

Billing Item		Description	Billing Mode	Billing Formula
Compute resource	Dedicated resource pool	Usage of compute resources. For details, see **ModelArts Pricing Details**.	Yearly/Monthly	Specification unit price x Number of compute nodes x Purchase duration

- When purchasing a Lite Cluster resource pool, you need to select a CCE cluster. For details about CCE pricing, see **CCE Price Calculator**.

Process

The following figure shows the process of enabling cluster resources.

Figure 2-1 Process

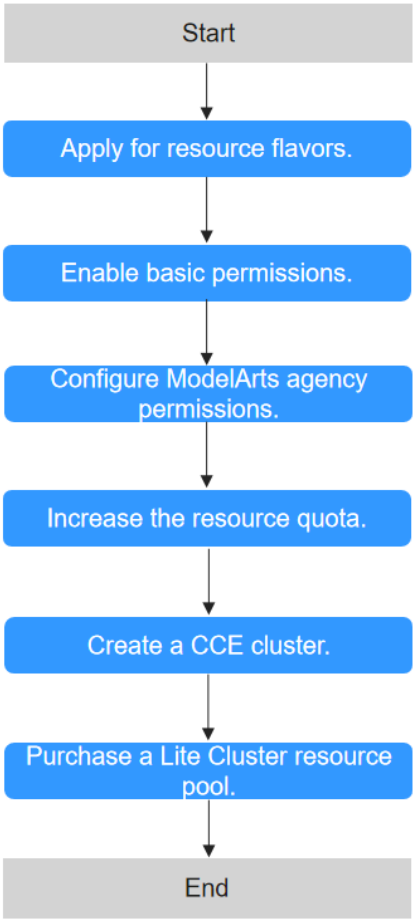


Table 2-2 Enabling cluster resources

Step	Description
Preparations	Before enabling resources, ensure that you have completed related preparations, including applying for required specifications and configuring permissions.
Purchasing a Lite Cluster Resource Pool	Purchase Lite Cluster resources on the ModelArts console.

Preparations

Before enabling resources, ensure that you have completed related preparations, including applying for required specifications and configuring permissions.

Step1 Enabling Resource Specifications

Contact your account manager to request restricted specifications (such as modelarts.bm.npu.arm.8snt9b3.d) in advance. They will enable the specifications within one to three working days. If there is no account manager, submit a service ticket.

Step 2: Enabling Basic Permissions

Log in to the administrator account and grant the target IAM account basic permissions to use resource pools.

Step 1 Log in to the **IAM console**.

Step 2 In the navigation pane on the left, choose **User Groups**, and then click **Create User Group** in the upper right corner.

Step 3 Set the user group name and click **OK**.

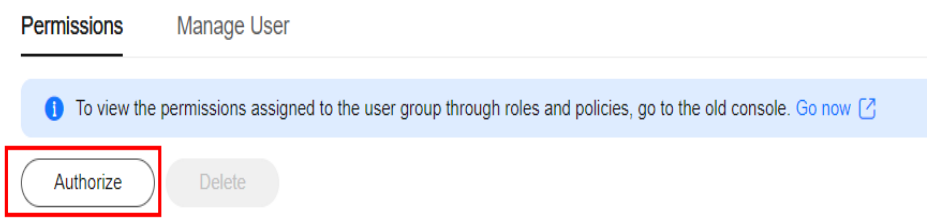
The name can contain only letters, digits, spaces, hyphens (-), and underscores (_).

Step 4 To add a user to a user group to assign permissions, locate the user group in the list, and click **Manage User** in the **Operation** column.

Step 5 Click the user group name to access its details page.

Step 6 In the **Permissions** tab, click **Authorize**.

Figure 2-2 Permissions



Step 7 Search for **ModelArtsFullAccessPolicy** and select it.

Figure 2-3 ModelArtsFullAccessPolicy

Repeat this step to add the following permissions:

- ModelArts FullAccess
- CTS Administrator
- CCE Administrator
- BMS FullAccess
- IMS FullAccess
- DEW KeypairReadOnlyAccess
- VPC FullAccess
- ECS FullAccess
- SFS Turbo FullAccess
- OBS Administrator
- AOM FullAccess
- TMS FullAccess
- BSS Administrator

Step 8 Click **Next** and set **Scope** to **All resources**.

Step 9 Click **OK**.

After the permissions are assigned, click the user group name to access its details page, and click the **Permissions** tab to view the assigned permissions.

----End

Step 3 Creating an Agency on ModelArts

- Creating an agency

  Create an agency on ModelArts to authorize access to other cloud services.

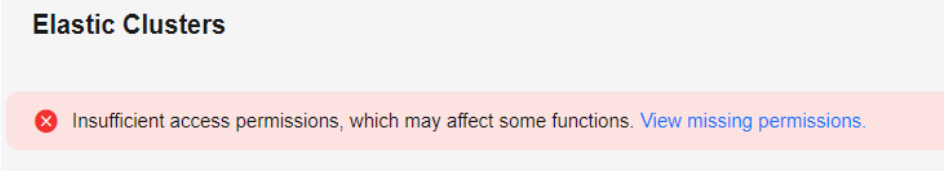
  Log in to the **ModelArts console**. In the navigation pane on the left, choose **Permission Management**. On the displayed page, click **Add Authorization**. For details, see **Adding Authorization**.

- Updating an agency

  Update the permissions for your existing ModelArts agency.

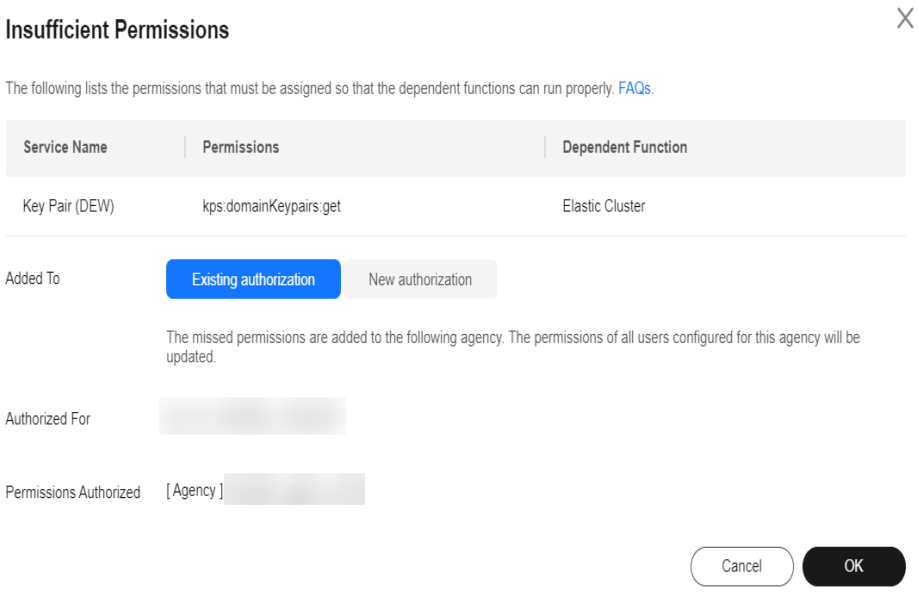
  a. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**. Check whether a message is displayed, indicating missing authorization.

Figure 2-4 Missing permissions



b. Click **View missing permissions** to update the agency if needed. Set **Added To** to **Existing authorization** and click **OK**.

Figure 2-5 Adding authorization



Step 4 Applying for a Higher Resource Quota

To run clusters, you will need more resources than Huawei Cloud's default quotas provided. This includes more ECS instances, memory, CPU cores, and EVS disk space. You will need to request a higher quota to meet these needs. To obtain the quota scheme, submit a service ticket or contact the customer manager.

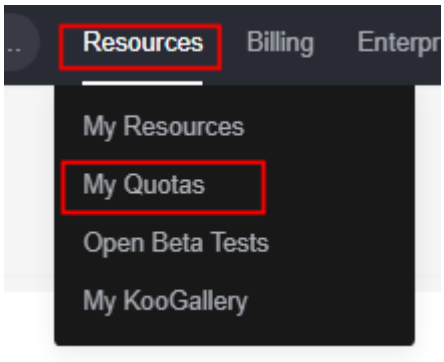
Increase the quota before purchasing and provisioning the resource, ensuring it exceeds the resource's requirements.

To run AI workloads in resource pools, you will need more resources than Huawei Cloud's default quotas provided. This includes more ECS instances, memory, CPU cores, and EVS disk space. To access these extra resources, request a higher quota. Confirm the solution with the customer manager, then apply for a higher resource quota by following these steps.

Step 1 Log in to the **Huawei Cloud console**.

Step 2 Hover over **Resources** from the top menu bar and choose **My Quotas**.

Figure 2-6 My Quotas



Step 3 On the **Quotas** page, click **Increase Quota** in the upper right corner and submit a service ticket.

Request the required number of ECS instances, CPU cores, RAM capacity (memory size), and EVS disk capacity. Contact your customer manager for quota details.

Figure 2-7 ECS resource type

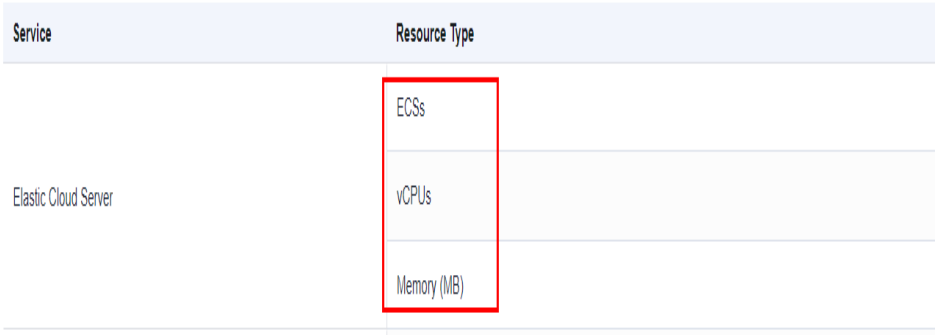
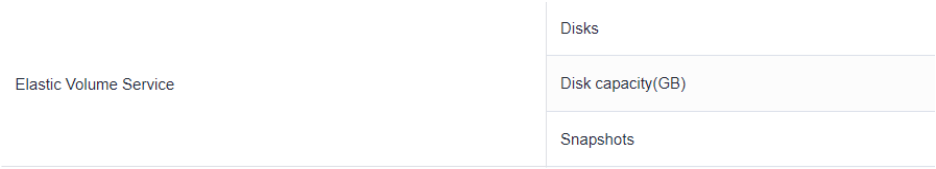


Figure 2-8 EVS resource type



----End

Step 5 Buying a CCE Cluster

Lite Cluster resource pools depend on CCE clusters to provide a container-based environment. CCE clusters provide Lite resource pools with required computing, storage, and network resources. Therefore, you need to select CCE clusters when purchasing Lite Cluster resource pools.

If no CCE cluster is available, purchase one by referring to **Buying a CCE Standard/Turbo Cluster**. For details about the cluster version, see **Software Versions Required by Different Models**.

CCE clusters of versions 1.23, 1.25, 1.28, and 1.31 are supported. CCE clusters of v1.28 and v1.31 can be created on the console or using APIs. CCE clusters of v1.23 and v1.25 can be created using APIs. For details about how to create CCE clusters of different versions, see **Kubernetes Version Policy**.

If you already have a CCE cluster but its version is earlier than 1.23, you shall upgrade it to 1.28 by referring to **Cluster Upgrade Overview**.

Create a Lite Cluster resource pool only when the CCE cluster is running.

Purchasing a Lite Cluster Resource Pool

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.
2. Click **Buy Lite Cluster**. On the displayed page, configure parameters according to the following table.

Table 2-3 Parameters

Parameter	Sub-Parameter	Description
Billing Mode	Yearly/ Monthly	Yearly/Monthly is a prepaid billing mode in which your subscription is billed based on the required duration. This mode is more cost-effective when the usage duration is predictable. For Lite Cluster resources, only the yearly/monthly billing mode is supported.
Cluster Flavor	Pool Name	The system provides an editable name. Only lowercase letters, digits, and hyphens (-) are allowed. The value must start with a lowercase letter and cannot end with a hyphen (-).
	Select CCE Cluster	Choose an existing CCE cluster from the drop-down list. Click **Create Cluster** on the right to create a cluster if none is available. For details about the required cluster version, see **Software Versions Required by Different Models**. Create a Lite Cluster resource pool only when the CCE cluster is running. CCE clusters of versions 1.23, 1.25, 1.28, and 1.31 are supported. CCE clusters of v1.28 and v1.31 can be created on the console or using APIs. CCE clusters of v1.23 and v1.25 can be created using APIs. For details about how to create CCE clusters of different versions, see **Kubernetes Version Policy**. If you already have the CCE cluster but its version is earlier than 1.23, you shall upgrade it to 1.28 by referring to **Cluster Upgrade Overview**.

Parameter	Sub-Parameter	Description
Default Specifications	CPU Architecture	The CPU architecture refers to the command set and design specifications for CPUs. The CPU supports x86 and Arm64 architectures, as well as heterogeneous scheduling for x86 and Arm64. Set this parameter as required. • x86: applies to most general-purpose computing scenarios and supports a wide software ecosystem. • Arm64: applies to specific optimization scenarios, such as mobile applications and embedded systems, features low power consumption.
	Instance Specifications Type	Choose CPU, GPU, or Ascend chips as needed. • CPU: general-purpose compute architecture, features low computing performance, is suitable for lightweight general tasks. • GPU: parallel compute architecture, features high computing performance, is suitable for parallel tasks and scenarios such as deep learning training and image processing, and supports multi-PU distributed training. • Ascend: dedicated AI architecture, features extremely high computing performance, is suitable for AI tasks and scenarios such as AI model training and inference acceleration, and supports multi-node distributed deployment.
	Instance Specifications	Select required specifications. Due to system loss, the available resources are fewer than specified. After a resource pool is created, view the available resources in the **Nodes** tab on the details page.
	AZ	Select **Automatic** or **Manual** as required. An AZ is a physical region where resources use independent power supplies and networks. AZs are physically isolated but interconnected over an intranet. • **Automatic**: AZs are automatically allocated. • **Manual**: Specify AZs for resource pool instances. To ensure system disaster recovery, deploy all instances in the same AZ. You can set the number of instances in an AZ.

Parameter	Sub-Parameter	Description
	Instances	Select the number of instances (nodes) in the Lite Cluster resource pool. A larger number indicates higher computing performance. If **AZ** is set to **Manual**, you do not need to configure **Instances**. Do not create more than 30 instances at a time. Otherwise, the creation may fail due to traffic limiting. You can purchase instances by rack for certain specifications. The total number of instances is the number of racks multiplied by the number of instances per rack. Purchasing a full rack allows you to isolate tasks physically, preventing communication conflicts and maintaining linear computing performance as task scale increases. All instances in a rack must be created or deleted together.

Parameter	Sub-Parameter	Description
	Advanced Specifications Configuration	By enabling this, you can configure the following parameters: • **Operating system**: Operating system of the instances. • **Container Engine**: Container engines, one of the most important components of Kubernetes, manage the lifecycle of images and containers. The kubelet interacts with a container runtime through the Container Runtime Interface (CRI). **Docker** and **Containerd** are supported. For details about the differences between Containerd and Docker, see **Container Engines**. When creating a resource pool, you can choose a container engine. Alternatively, you can change the container engine on the scaling page after the resource pool is created. The CCE cluster version determines the available container engines. If it is earlier than 1.23, only Docker is supported. If it is 1.27 or later, only containerd is supported. For all other versions, both containerd and Docker are options. • **Node Pool Name**: Customize the name of the new node pool. Once a resource pool is created, the name of an existing node pool in the resource pool cannot be changed. • **Virtual Private Cloud**: VPC network where the CCE cluster is located and cannot be changed. • **Node subnet**: Choose a subnet within the same VPC. New nodes will be created using this subnet. • **Associated Security Group**: Specifies the security group used by nodes created in the node pool. A maximum of four security groups can be selected. Traffic needs to pass through certain ports in the node security group to ensure node communications. If no security group is associated, the cluster's default rules are applied. • **Resource Tag**: Add resource tags to classify resources. You can also modify the tags on the resource pool details page after the resource pool is created. • **Kubernetes Labels**: Add key/value pairs that are attached to Kubernetes objects, such as Pods. A maximum of 20 labels can be added. Labels can be used to distinguish nodes. With workload affinity settings, container pods can be scheduled to a specified node. • **taint**: This parameter is left blank by default. Configure anti-affinity by adding taints to nodes, with a maximum of 20 taints per node. • **Post-installation Command**: Enter the script command, which cannot include Chinese characters. The Base64-

Parameter	Sub-Parameter	Description
		encoded script must be transferred. The encoded script should not exceed 2,048 characters. The script will be executed after Kubernetes software is installed, which does not affect the installation. Do not run the **reboot** command in the post-installation script to restart the system immediately. To restart the system, run the **shutdown -r 1** command to restart with a delay of one minute.

Parameter	Sub-Parameter	Description
	Storage Configuration	By enabling this, you can configure the following parameters: • **System Disk**: Select the type and size of the system disk. The system disk can be a local disk or an EVS disk (including common SSD, high I/O, and ultra-high I/O). System disks of some specifications support only local disks. • **Container Disk**: Select the type, size, and quantity of the container disk. The storage type of container disks of some specifications can be manually set. You can select local disks or EVS disks. • **Advanced Container Disk Configuration**: You can specify the disk space, container engine space size, and write mode. – **Container Engine Space Size**: The default container engine space size is **50 GiB**. You can specify the container engine space or set the space to unlimited. The default and minimum values are 50 GiB. The maximum value depends on the specifications, and can be found in the console prompt. – **Write Mode**: For some specifications, the write mode of container disks can be set to linear or striped. A linear logical volume integrates one or more physical volumes. Data is written to the next physical volume when the previous one is used up. A striped logical volume stripes data into blocks of the same size and stores them in multiple physical volumes in sequence. This allows data to be concurrently read and written. A storage pool consisting of striped volumes cannot be scaled out. • **Data Disk**: Some specifications support common data disks. Multiple data disks can be mounted to a resource pool. The type, size, and quantity of data disks can be set. The maximum number of disks varies depending on the flavor. For example, a maximum of four data disks can be mounted to a 300I Duo node. • **Advanced Data Disk Configuration**: For some specifications, you can set the data disk mounting mode. The details are as follows: – **Default**: The EVS disks are directly attached to the resource pool without any additional processing, such as partitioning. – **Specified Directory**: You can set the path and write mode, which can be linear or striped.

Parameter	Sub-Parameter	Description
		– **Local PV**: You can set the PV write mode, which can be linear or striped. In this example, the write mode of all data disks is set. – **Ephemeral Volume**: You can set the EV write mode, which can be linear or striped. In this example, the write mode of all data disks is set.
Add	-	Add multiple specifications as needed. Restrictions: ● Selecting multiple same specifications allows you to specify a node pool name by clicking **Advanced Configuration**. Only one node pool name can be left unspecified. ● If you select multiple CPU architectures, x86 and Arm heterogeneous scheduling is supported. ● When selecting multiple GPU or NPU specifications, distributed training speed is impacted because different specifications' parameter network planes are not connected. For distributed training, it is recommended that you choose only one GPU or NPU specification. ● You can add up to 10 specifications to a resource pool.

Parameter	Sub-Parameter	Description
Plug-in Configuration	Select Plug-in	ModelArts offers several plug-ins to help you expand resource pool functions as needed. Click **Select Plug-in**. In the displayed dialog box, select plug-ins you want to add and click **OK**. Click **View Details** to view plug-in functions and version updates. The following plug-ins are added by default: • **ModelArts Node Agent**: a node problem detector that monitors cluster node exceptions and interconnects with third-party monitoring platforms. It is a daemon that runs on each node to collect node problems from different daemon processes. • **ModelArts Metric Collector**: default built-in plug-in, which runs as a node daemon to collect monitoring metrics of nodes and jobs and report the metrics to AOM. • **AI Suite – Ascend NPU (ModelArts Device Plugin)**: a management plug-in that supports Huawei NPU devices in containers It is automatically installed when **Instance Specifications Type** is set to **Ascend**. • **Volcano Scheduler**: a batch scheduling platform based on Kubernetes. It provides a series of features required by machine learning, deep learning, bioinformatics, genomics, and other big data applications, as a powerful supplement to Kubernetes capabilities.
Resource scheduling and allocation	Custom Driver	This function is disabled by default. Some GPU and Ascend resource pools allow custom driver installation. The driver is automatically installed in the cluster by default. Enable this function only if you need to specify the driver version. Determine the required driver version and choose the matching driver when buying Lite Cluster resources.
	GPU / Ascend Driver	This parameter is displayed if **Custom Driver** is enabled. You can select a GPU or Ascend driver. The value depends on the driver you choose. For details about the required gpu-driver version, see **Software Versions Required by Different Models**.
Advanced Config	Cluster Description	Enter a description.

Parameter	Sub-Parameter	Description
	User-defined Node Name Prefix	Choose whether to enable this function to add a node name prefix. • After a prefix is added, a node name consists of a prefix and a random number. • The value can contain 1 to 64 characters. • The prefix starts with a lowercase letter and only contains lowercase letters and digits. It is separated from the node name by a hyphen (-), for example, **node-com**.
	Tags	Click **Add Tag** to configure tags for the Lite resource pool so that resources can be managed by tag. The tag information can be predefined in Tag Management Service (TMS). You can also set tag information in the **Tags** tab of the details page after the Lite resource pool is created.
Management	Login Credential	Choose a cluster login mode, **Password** or **Key pair**. • **Password**: The default username is **root**, and you can set a password. • **Key pair**: Select an existing key pair or click **Create Key Pair** to create one.
Required Duration	N/A	Select the time length for which you want to use the resource pool. This parameter is mandatory only when the **Yearly/Monthly** billing mode is selected. Auto-renewal is disabled by default. If you enable this function, the resource pool will be automatically renewed upon expiration. The fees generated by auto-renewal will be deducted from your account balance. For details, see **Auto-Renewal**. If the subscription is monthly, the subscription is automatically renewed for one month. If the subscription is yearly, the subscription is automatically renewed for one year.

3. Click **Buy Now** and confirm the specifications. Confirm the information and click **Submit**.
   - After a resource pool is created, its status changes to **Running**. Only when the number of available nodes is greater than 0, tasks can be delivered to this resource pool.
   - Hover over **Creating** to view the details about the creation process. Click **View Details** to go the operation record page.
   - You can view the task records of the resource pool by clicking **Records** in the upper right corner of the Lite resource pool list.

Figure 2-9 Operation records

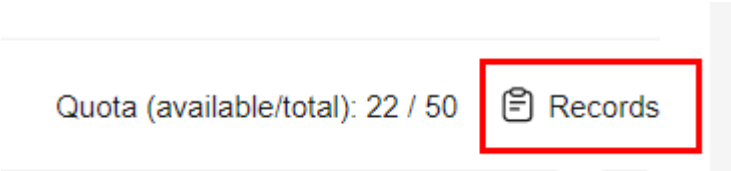
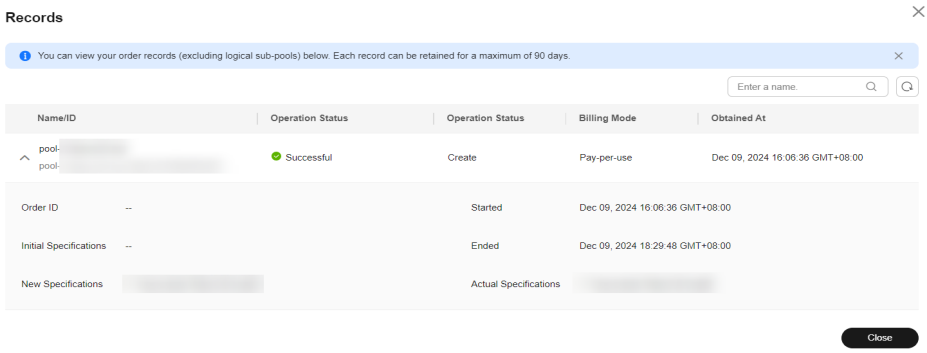
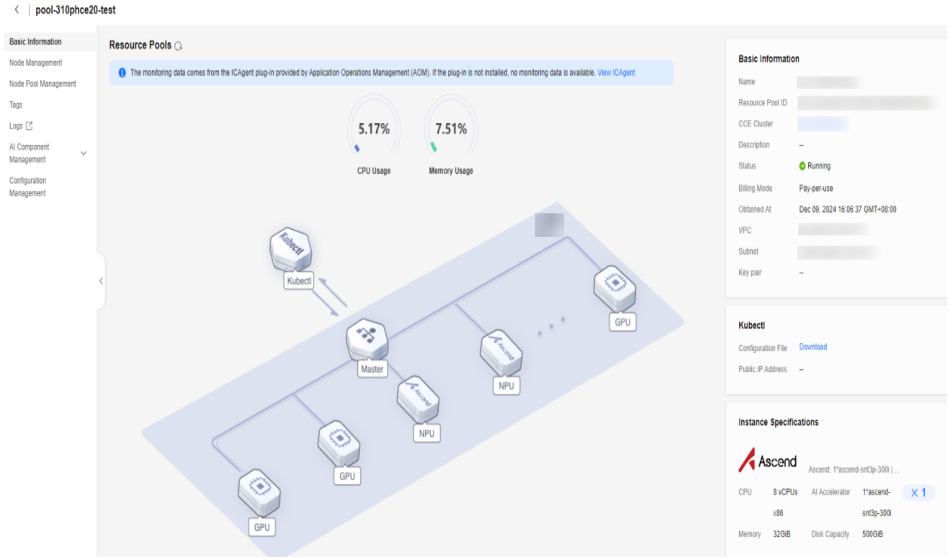


Figure 2-10 Viewing operation records



After a resource pool is created, its status changes to **Running**. Click the cluster resource name to go to the resource details page. Check whether the purchased specifications are correct.

Figure 2-11 Viewing resource details



3 Configuring Lite Cluster Resources

3.1 Configuring the Lite Cluster Environment

Configure the Lite Cluster environment by following this section, which applies to the accelerator card environment setup.

Prerequisites

- You have purchased and enabled cluster resources. For details, see **Enabling Lite Cluster Resources**.
- To configure and use a cluster, you need to have a solid understanding of **Kubernetes Basics**, as well as basic knowledge of networks, storage, and images.

Configuration Process

Figure 3-1 Flowchart

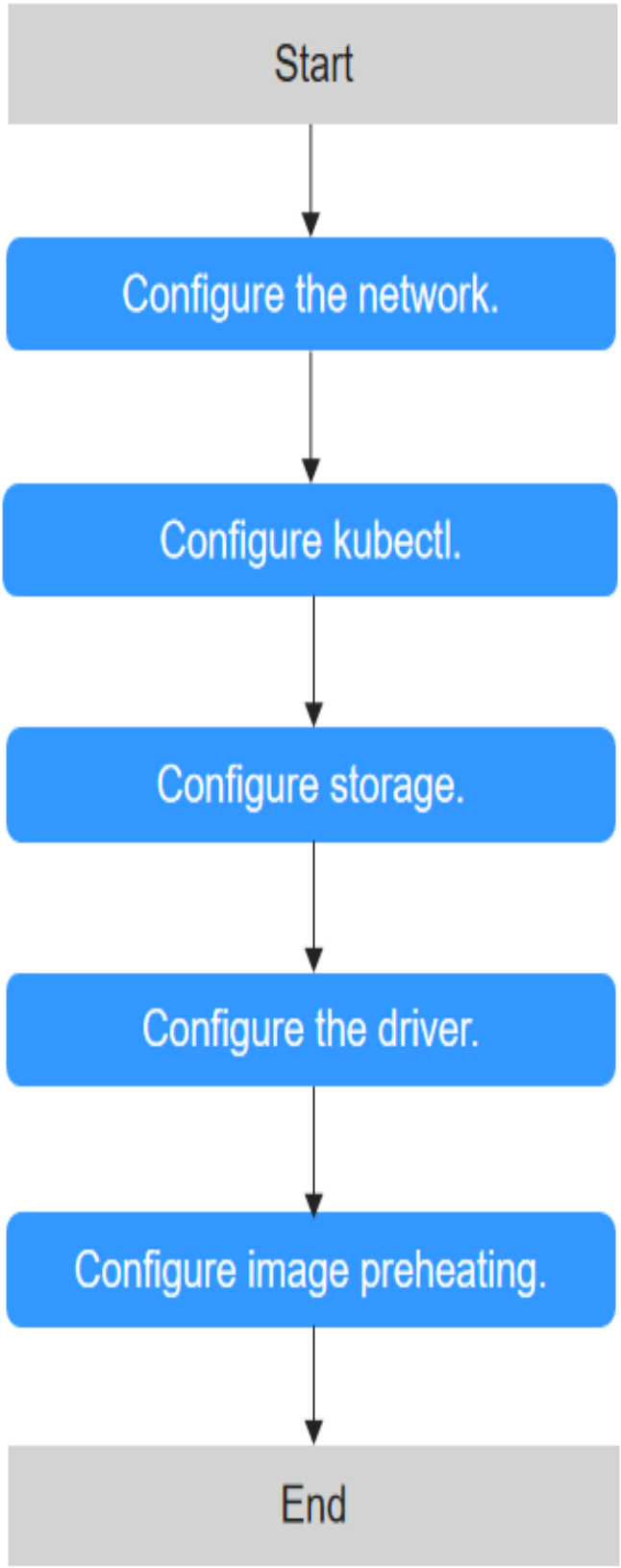


Table 3-1 Configuration process

Step	Task	Description
1	**Configuring the Lite Cluster Network**	After purchasing a resource pool, create an elastic IP (EIP) and configure the network. Once the network is set up, you can access cluster resources through the EIP.
2	**Configuring kubectl**	With kubectl configured, you can use the command line tool to manage your Kubernetes clusters by running kubectl commands.
3	**Configuring Lite Cluster Storage**	The available storage space is determined by dockerBaseSize when no external storage is mounted. However, the accessible storage space is limited. It is recommended that you mount external storage to overcome this limitation. You can mount storage to a container in various methods. The recommended method depends on the scenario, and you can choose one that meets your service needs.
4	**(Optional) Configuring the Driver**	Configure the corresponding driver to ensure proper use of GP/Ascend resources in nodes within a dedicated resource pool. If no custom driver is configured and the default driver does not meet service requirements, upgrade the default driver to the required version.
5	**(Optional) Configuring Image Pre-provisioning**	Lite Cluster resource pools enable image pre-provisioning, which pulls images from nodes in the pools beforehand, accelerating image pulling during inference and large-scale distributed training.

Quick Configuration of Lite Cluster Resources

This section shows how to configure Lite Cluster resources quickly to log in to nodes and view accelerator cards, then complete a training job. Before you start, you need to purchase resources. For details, see **Enabling Lite Cluster Resources**.

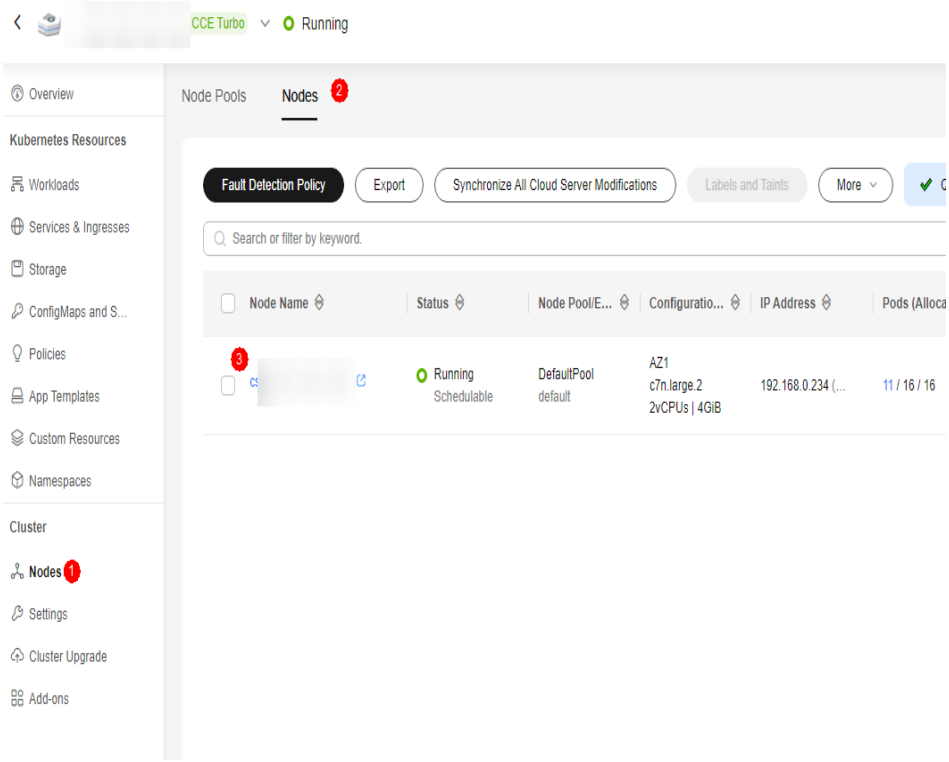
Step 1 Log in to a node.

(Recommended) Method 1: Binding an EIP

Bind an EIP to the node and use Bash tools such as Xshell and MobaXterm to log in to the node.

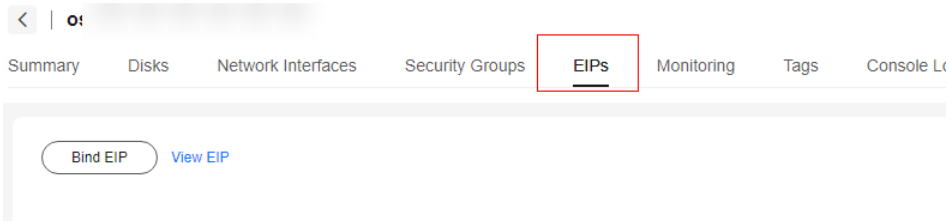
1. Log in to the CCE console.
2. On the CCE cluster details page, click **Nodes**. In the **Nodes** tab, click the name of the target node to go to the ECS page.

Figure 3-2 Node management



3. Bind an EIP.
   Choose or create one.

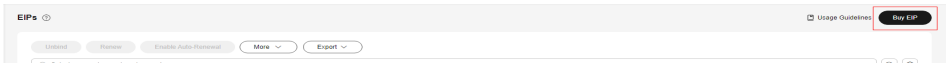
Figure 3-3 EIP



Click **Buy EIP**.

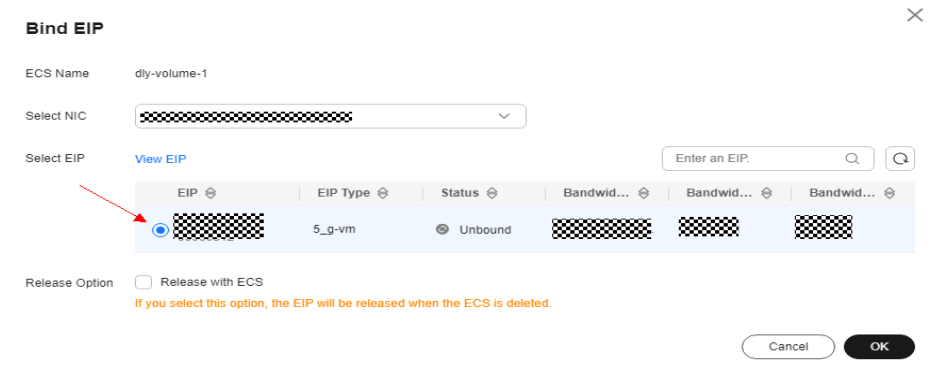
Figure 3-4 Binding an EIP

Figure 3-5 Buying an EIP



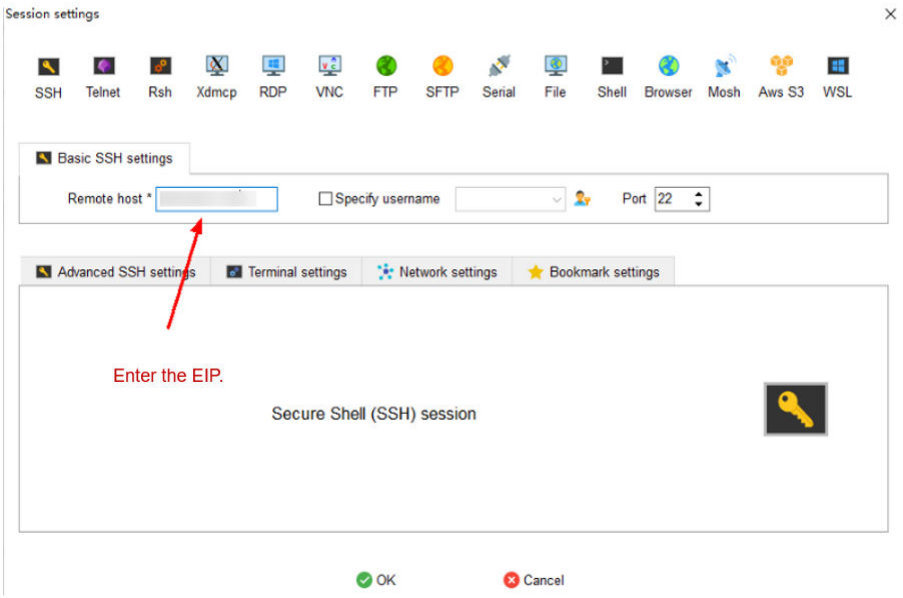
Refresh the list on the ECS page after completing the purchase. Select the new EIP and click **OK**.

Figure 3-6 Binding an EIP



4. Log in to the node using MobaXterm or Xshell. To log in using MobaXterm, enter the EIP.

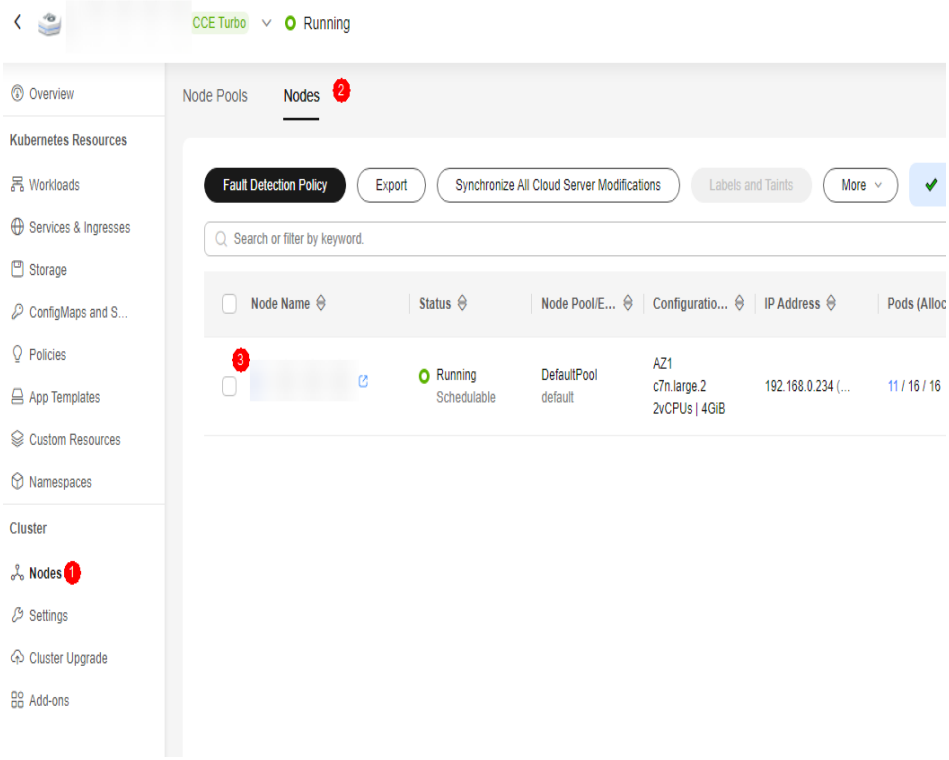
Figure 3-7 Logging in to a node



Method 2: Using Huawei Cloud Remote Login

1. Log in to the CCE console.
2. On the CCE cluster details page, click **Nodes**. In the **Nodes** tab, click the name of the target node to go to the ECS page.

Figure 3-8 Node management



3. Click **Remote Login**. In the displayed dialog box, click **Log In**.

   Figure 3-9 Remote login

4. After setting parameters such as the password in CloudShell, click **Connect** to log in to the node. For details about CloudShell, see **Logging In to a Linux ECS Using CloudShell**.

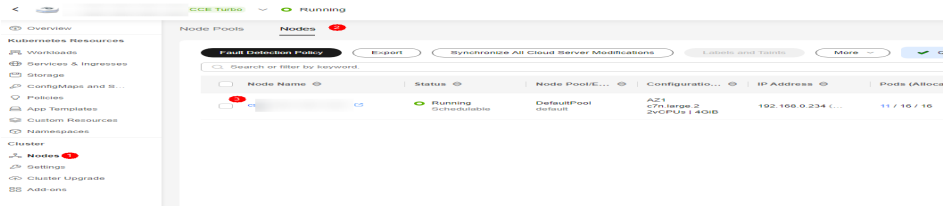
Step 2 Configure the kubectl tool.

Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.

Click the new dedicated resource pool to access its details page. Click the CCE cluster to access its details page.

On the CCE cluster details page, locate **Connection Information** in the cluster information.

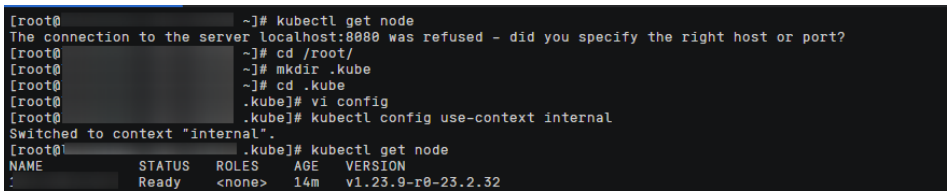
Figure 3-10 Connection Information



Use kubectl.

- To use kubectl through the intranet, install it on a node within the same VPC as the cluster. Click **Configure** next to **kubectl** to use the kubectl tool.

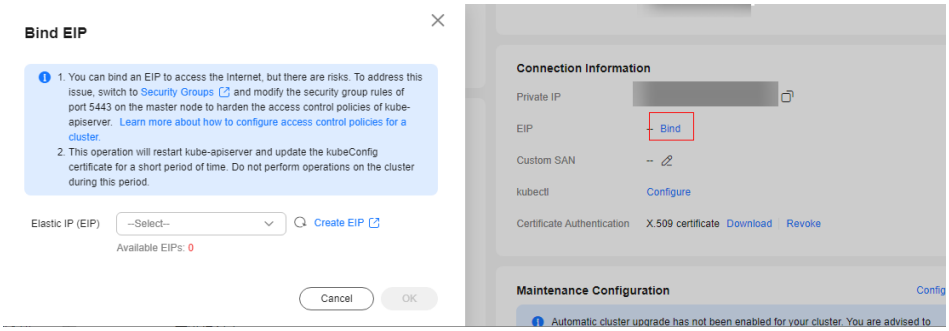
  Figure 3-11 Using kubectl through the intranet

- To use kubectl through an EIP, install it on any node that associated with the EIP.

  To bind an EIP, click **Bind** next to **EIP**.

Figure 3-12 Binding an EIP



Select an EIP and click **OK**. If no EIP is available, click **Create EIP** to create one.

After the binding is complete, click **Configure** next to **kubectl** and use kubectl as prompted.

Step 3 Start a task using **docker run**.

After an Snt9B cluster is managed by a CCE cluster, a container will be installed for running. The following uses Docker as an example. This is for test only. You can directly start the container for test without creating a deployment or volcano job. The BERT NLP model is used in the training test cases.

1. Pull the image. The test image is **bert_pretrain_mindspore:v1**, which contains the test data and code.

```
docker pull swr.cn-southwest-2.myhuaweicloud.com/os-public-repo/bert_pretrain_mindspore:v1
docker tag swr.cn-southwest-2.myhuaweicloud.com/os-public-repo/bert_pretrain_mindspore:v1
bert_pretrain_mindspore:v1
```

2. Start the container.

```
docker run -tid --privileged=true \
-u 0 \
-v /dev/shm:/dev/shm \
--device=/dev/davinci0 \
--device=/dev/davinci1 \
--device=/dev/davinci2 \
--device=/dev/davinci3 \
--device=/dev/davinci4 \
--device=/dev/davinci5 \
--device=/dev/davinci6 \
--device=/dev/davinci7 \
--device=/dev/davinci_manager \
--device=/dev/devmm_svm \
--device=/dev/hisi_hdc \
-v /usr/local/Ascend/driver:/usr/local/Ascend/driver  \
-v /usr/local/sbin/npu-smi:/usr/local/sbin/npu-smi \
-v /etc/hccn.conf:/etc/hccn.conf \
bert_pretrain_mindspore:v1 \
bash
```

Parameters:

- --privileged=true //Privileged container, which can access all devices connected to the host.
- -u 0 //root user
- -v /dev/shm:/dev/shm //Prevents the training task from failing due to insufficient shared memory.
- --device=/dev/davinci0 //NPU card device

- --device=/dev/davinci1 //NPU card device
- --device=/dev/davinci2 //NPU card device
- --device=/dev/davinci3 //NPU card device
- --device=/dev/davinci4 //NPU card device
- --device=/dev/davinci5 //NPU card device
- --device=/dev/davinci6 //NPU card device
- --device=/dev/davinci7 //NPU card device
- --device=/dev/davinci_manager //Da Vinci-related management device
- --device=/dev/devmm_svm //Management device
- --device=/dev/hisi_hdc //Management device
- -v /usr/local/Ascend/driver:/usr/local/Ascend/driver //NPU card driver mounting
- -v /usr/local/sbin/npu-smi:/usr/local/sbin/npu-smi //npu-smi tool mounting
- -v /etc/hccn.conf:/etc/hccn.conf //hccn.conf configuration mounting

3. Access the container and view the card information.

```
docker exec -it xxxxxxx bash    //Access the container. Replace xxxxxxx with the container ID.
npu-smi info    //View card information.
```

Figure 3-13 Viewing NPU information

```
[root@3c799939827b bert]# npu-smi info
+------------------------------------------------------------------------------------------------+
| npu-smi 23.0.rc2                 Version: 23.0.rc2.2.b030                                      |
+---------------------------+---------------+----------------------+-----------------------------+
| NPU   Name                | Health        | Power(W)    Temp(C)          Hugepages-Usage(page)|
| Chip                      | Bus-Id        | AICore(%)   Memory-Usage(MB) HBM-Usage(MB)        |
+===========================+===============+======================+=============================+
| 0     910B1               | OK            | 93.1        46               0    / 0            |
| 0                         | 0000:C1:00.0  | 0           0    / 0         4313 / 65536         |
+---------------------------+---------------+----------------------+-----------------------------+
| 1     910B1               | OK            | 93.5        48               0    / 0            |
| 0                         | 0000:01:00.0  | 0           0    / 0         4313 / 65536         |
+---------------------------+---------------+----------------------+-----------------------------+
| 2     910B1               | OK            | 93.0        46               0    / 0            |
| 0                         | 0000:C2:00.0  | 0           0    / 0         4314 / 65536         |
+---------------------------+---------------+----------------------+-----------------------------+
| 3     910B1               | OK            | 93.1        47               0    / 0            |
| 0                         | 0000:02:00.0  | 0           0    / 0         4339 / 65536         |
+---------------------------+---------------+----------------------+-----------------------------+
| 4     910B1               | OK            | 93.3        48               0    / 0            |
| 0                         | 0000:81:00.0  | 0           0    / 0         4313 / 65536         |
+---------------------------+---------------+----------------------+-----------------------------+
| 5     910B1               | OK            | 94.8        48               0    / 0            |
| 0                         | 0000:41:00.0  | 0           0    / 0         4181 / 65536         |
+---------------------------+---------------+----------------------+-----------------------------+
| 6     910B1               | OK            | 93.3        49               0    / 0            |
| 0                         | 0000:82:00.0  | 0           0    / 0         4180 / 65536         |
+---------------------------+---------------+----------------------+-----------------------------+
| 7     910B1               | OK            | 93.2        48               0    / 0            |
| 0                         | 0000:42:00.0  | 0           0    / 0         4180 / 65536         |
+===========================+===============+======================+=============================+
+---------------------------+---------------+----------------------------------------------------+
| NPU     Chip              | Process id    | Process name             | Process memory(MB)      |
+===========================+===============+====================================================+
| No running processes found in NPU 0                                                            |
+===========================+===============+====================================================+
| No running processes found in NPU 1                                                            |
+===========================+===============+====================================================+
| No running processes found in NPU 2                                                            |
+===========================+===============+====================================================+
| No running processes found in NPU 3                                                            |
+===========================+===============+====================================================+
| No running processes found in NPU 4                                                            |
+===========================+===============+====================================================+
| No running processes found in NPU 5                                                            |
+===========================+===============+====================================================+
| No running processes found in NPU 6                                                            |
+===========================+===============+====================================================+
| No running processes found in NPU 7                                                            |
+===========================+===============+====================================================+
```


4. Start the training task:

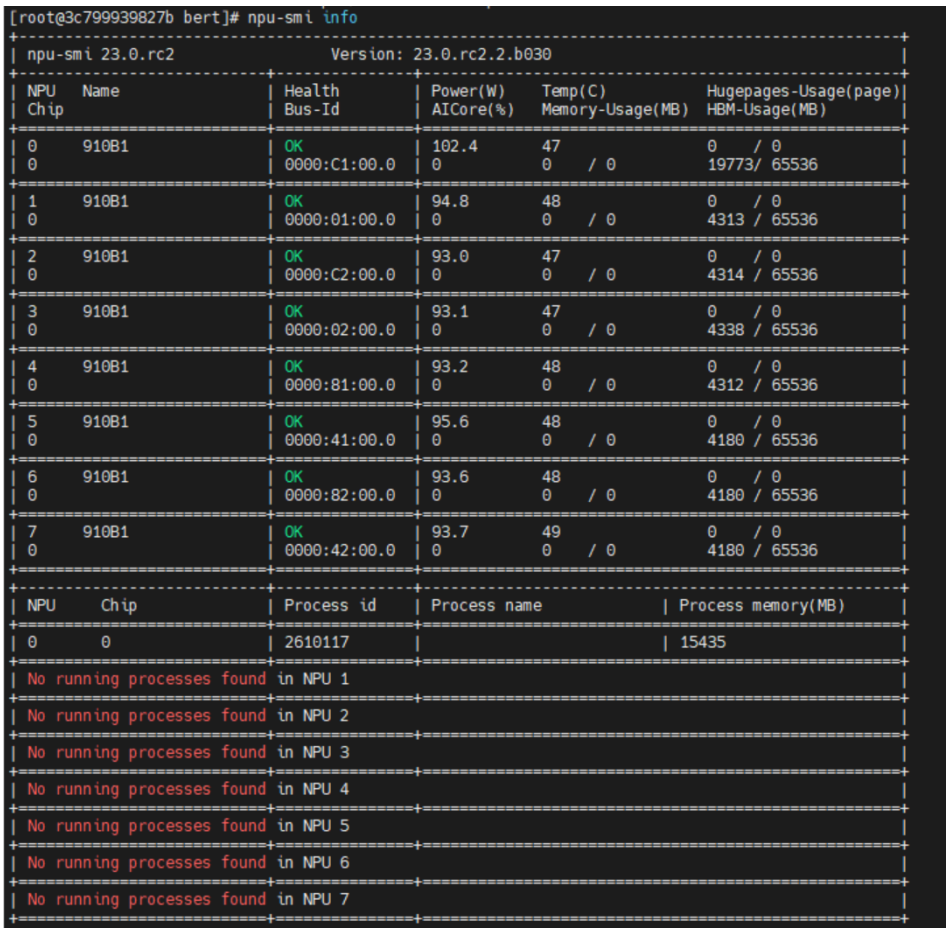
```
cd /home/ma-user/modelarts/user-job-dir/code/bert/
export MS_ENABLE_GE=1
export MS_GE_TRAIN=1
bash scripts/run_standalone_pretrain_ascend.sh 0 1 /home/ma-user/modelarts/user-job-dir/data/cn-news-128-1f-mind/
```

Figure 3-14 Training process

Check the card usage. The card 0 is in use, as expected.

```
npu-smi info    //View card information.
```

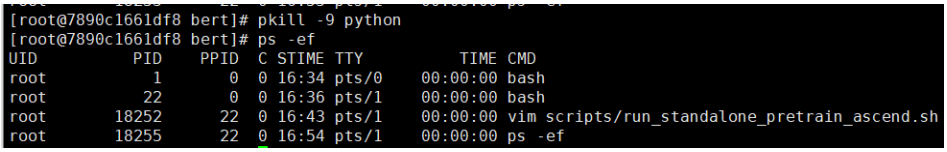
Figure 3-15 Viewing NPU information



The training task takes about two hours to complete and then automatically stops. To stop a training task, run the commands below:

```
pkill -9 python
ps -ef
```


Figure 3-16 Stopping the training process



----End

3.2 Configuring the Lite Cluster Network

Elastic IP (EIP) provides independent public IP addresses and bandwidth for Internet access.

After purchasing a Lite Cluster resource pool, create an EIP and configure the network. Once the network is set up, you can access Lite Cluster resources through the EIP.

Apply for an EIP and bind it to a Lite Cluster to enable the ECS to access the Internet.

Billing

After an EIP is bound to a Lite Cluster, bandwidth fees may be generated. For details, see **EIP Billing Overview**.

Prerequisites

- You have purchased and enabled Lite Cluster resources. For details, see **Enabling Lite Cluster Resources**.
- You have obtained the EIP to be bound. For details, see **Assigning an EIP**.

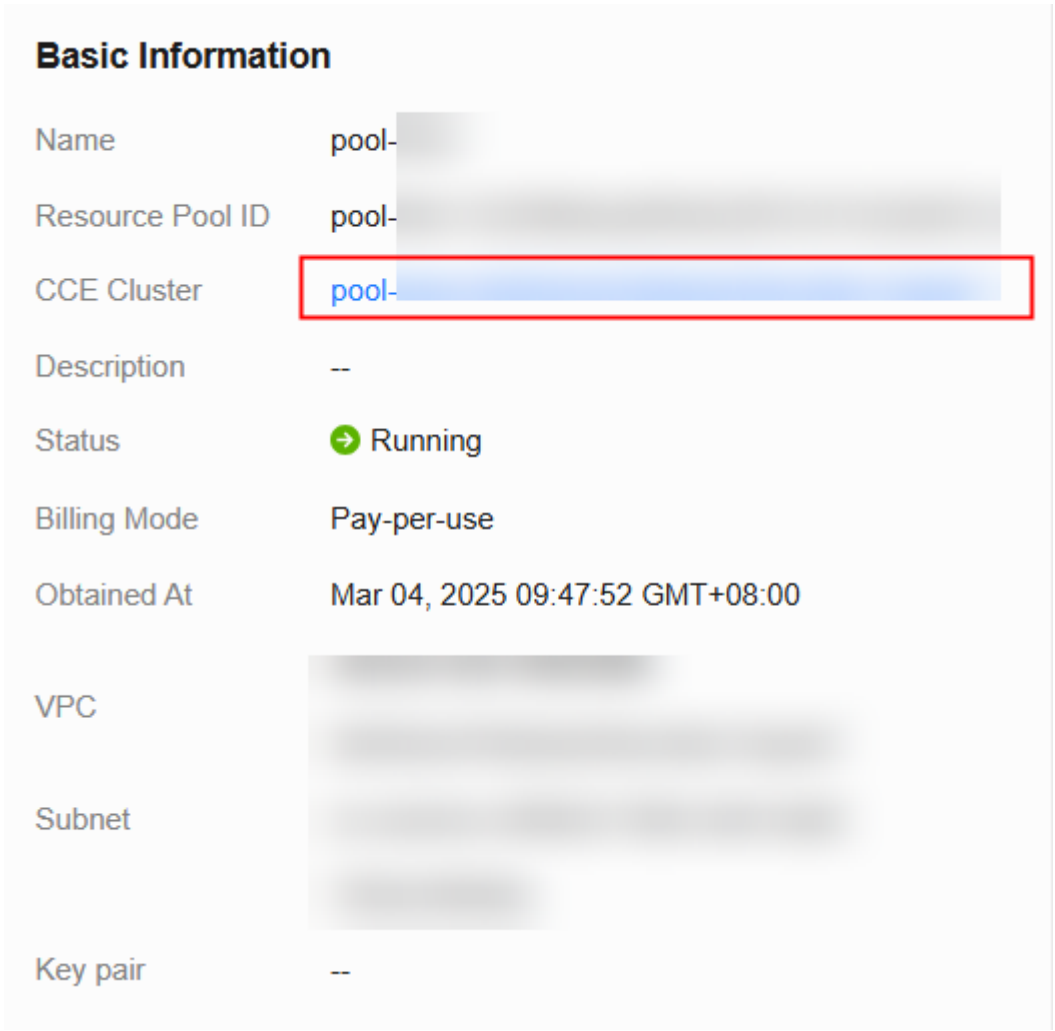
Binding an EIP to Configure Lite Cluster Network

Step 1 Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.

Step 2 Click the target Lite Cluster name to access its details page.

Step 3 In the **Basic Information** tab, click the CCE cluster name to access the CCE console.

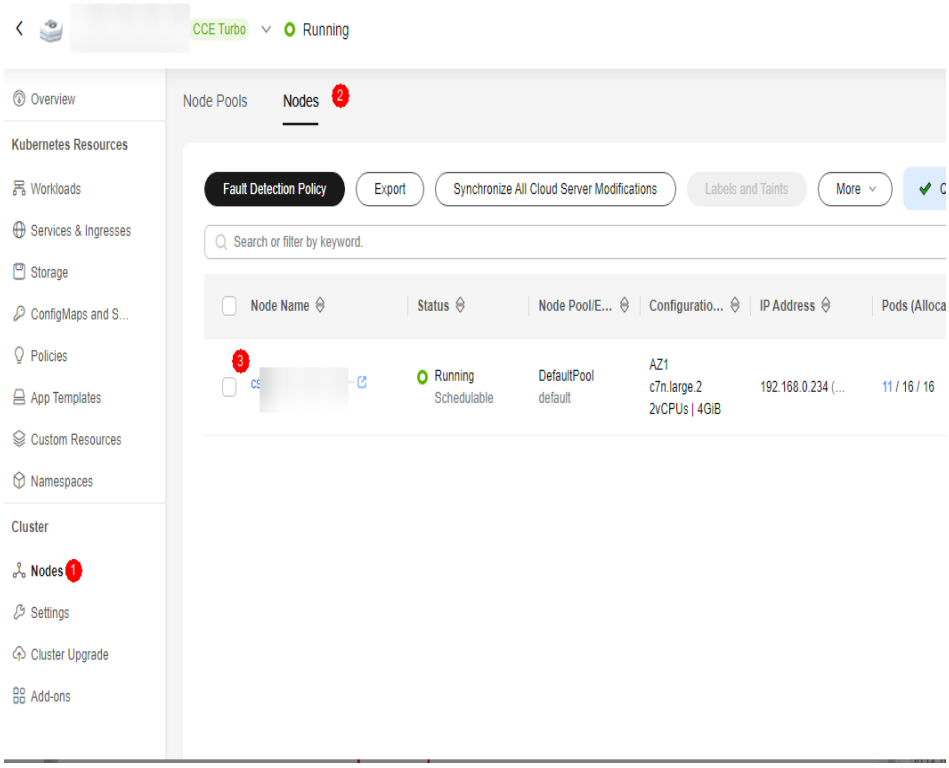
Figure 3-17 Lite Cluster resource pool basic information



Step 4 Locate the CCE cluster selected during Lite Cluster purchase, click its name to access the details page.

Step 5 In the navigation pane on the left, choose **Nodes**, and then switch to the **Nodes** tab. Click the node to be logged in to. The ECS details page is displayed.

Figure 3-18 Node management



Step 6 On the ECS details page, switch to the **EIPs** tab.

Step 7 Click **Bind EIP**, select an unbound EIP, and click **OK**.

Figure 3-19 EIP

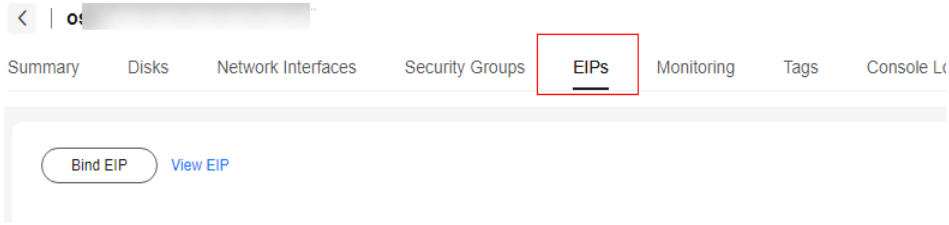
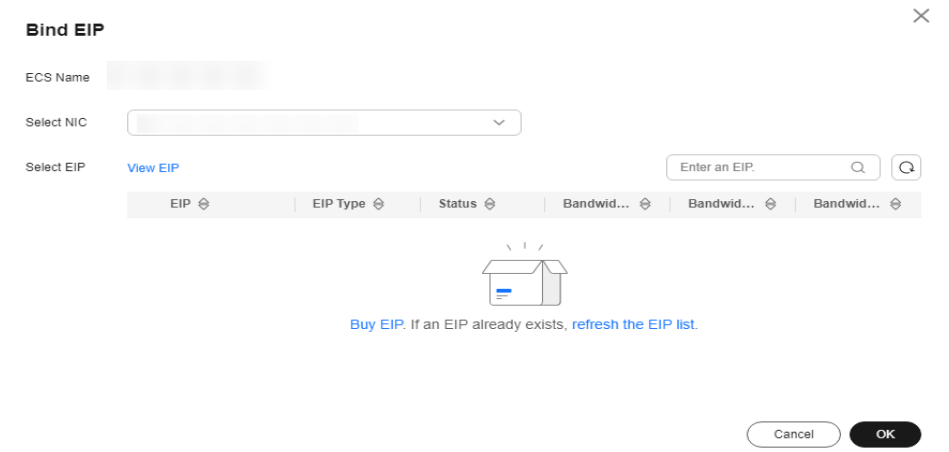


Figure 3-20 Binding an EIP



If no EIP is available, purchase one. To do so, click Buy EIP.

Refresh the list on the ECS page after completing the purchase. Select the created EIP and click **OK**.

For details about how to buy an EIP, see **Assigning an EIP**.

Figure 3-21 Purchasing an EIP

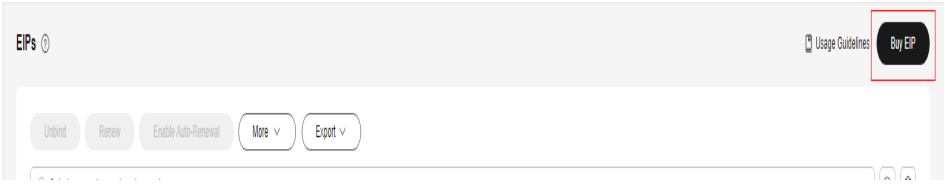
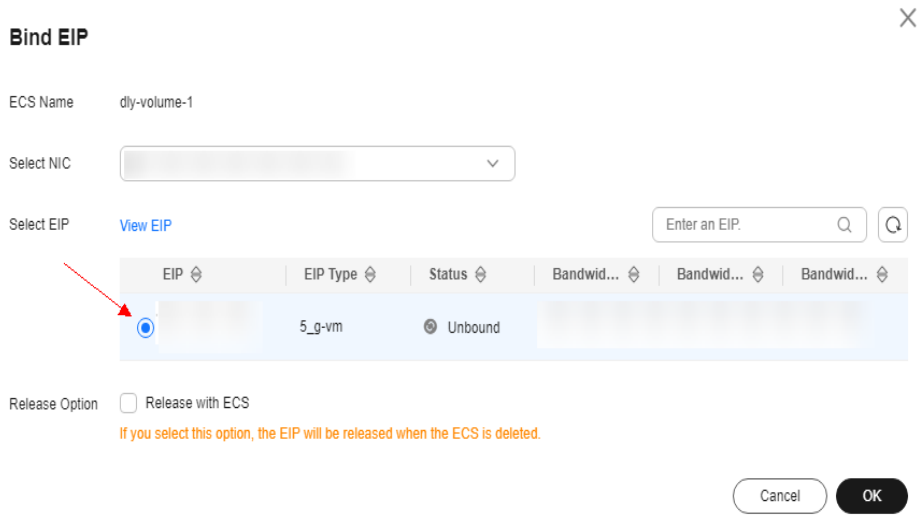


Figure 3-22 Binding an EIP



Step 8 Access cluster resources remotely using SSH with a password or key pair.

- To use a key pair, see **Logging In to a Linux ECS Using an SSH Key Pair**.
- To use a key pair, see **Logging In to a Linux ECS Using an SSH Password**.

----End

Follow-Up Operation

Configuring kubectl: With kubectl configured, you can use the CLI tool to manage your Kubernetes clusters by running **kubectl** commands.

3.3 Configuring kubectl

kubectl is a CLI tool provided by Kubernetes, enabling you to manage cluster resources, view cluster status, deploy applications, and debug issues through the CLI.

After kubectl is configured, you can connect it to a Lite Cluster resource pool for easy resource management by running **kubectl** commands.

To access a Kubernetes cluster of Lite Cluster through kubectl, use either of the following methods:

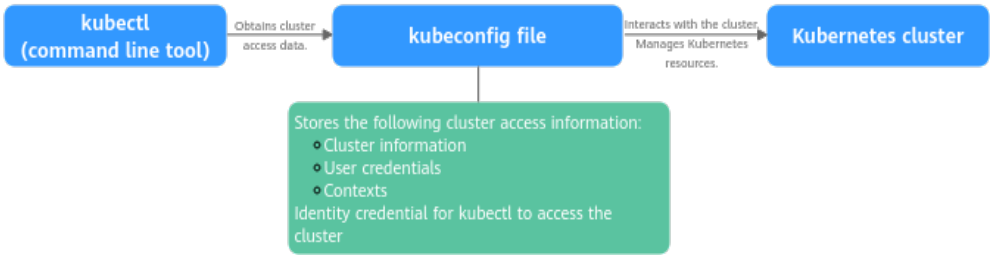
- Intranet access: Clients access the cluster's API server via an intranet IP address, keeping data traffic internal and enhancing security.
- Public access: The client's API server exposes a public API, allowing clients to access the Kubernetes cluster over the Internet.

This section describes how to configure kubectl for a Lite Cluster.

How It Works

kubectl retrieves cluster information from a kubeconfig file and communicates with the Kubernetes API server. The **kubeconfig** file is the identity credential for kubectl to access the Kubernetes cluster. It contains the API server address, user authentication credentials, and other configuration details. With these details, kubectl can interact with the Kubernetes cluster to perform management tasks.

Figure 3-23 Using kubectl to access a cluster



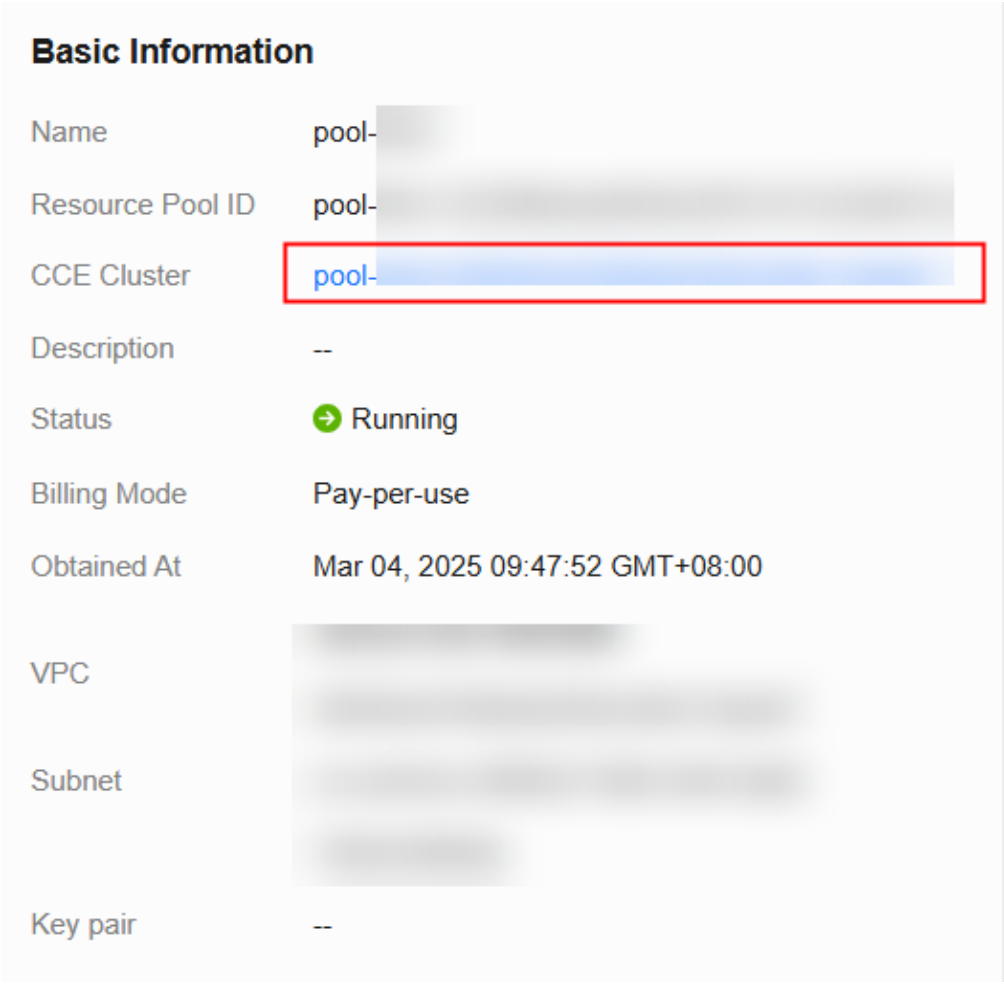
Prerequisites

- You have purchased and enabled Lite Cluster resources. For details, see **Enabling Lite Cluster Resources**.
- If you use kubectl through the VPC intranet, ensure that the client and Lite Cluster are in the same VPC.
- If you use kubectl through the public network, obtain the EIP to be bound in advance. For details, see **Assigning EIP**.

Configuring Kubectl for Lite Cluster

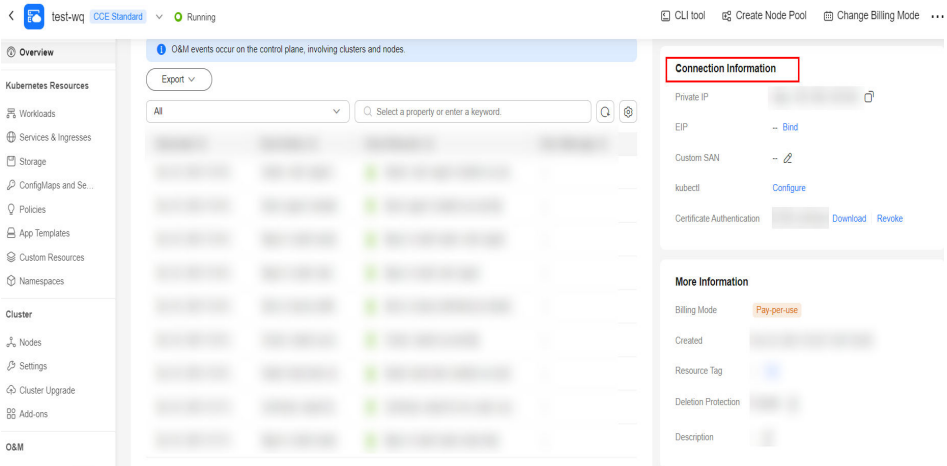
1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.
2. Click the created Lite Cluster dedicated resource pool to access its details page.

Figure 3-24 Basic information



3. Click the CCE cluster to access its details page. From there, locate **Connection Information** in the cluster information.

Figure 3-25 Connection Information



4. Configure the kubectl tool.
   - To use kubectl through the intranet, install it on a node within the same VPC as the cluster.

Click **Configure** next to kubectl and perform operations as prompted. For details, see **Configure kubectl**.

Figure 3-26 Cluster connection information

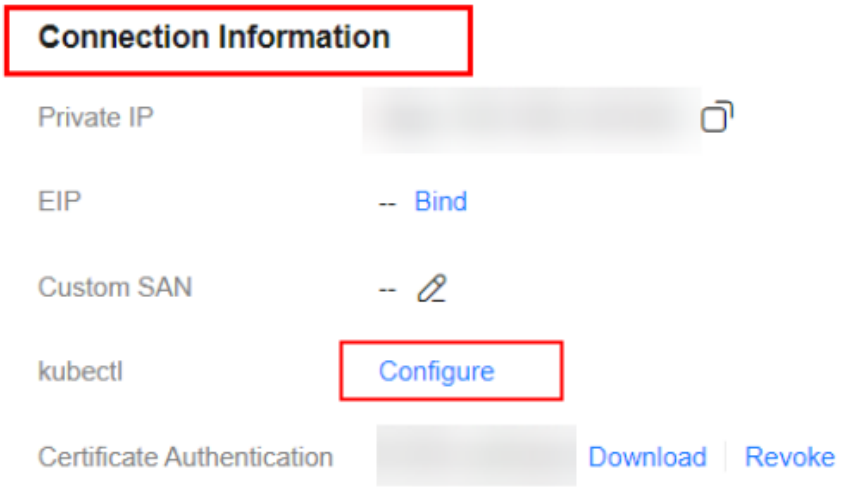
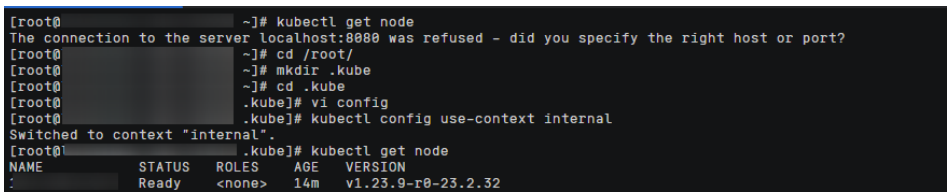


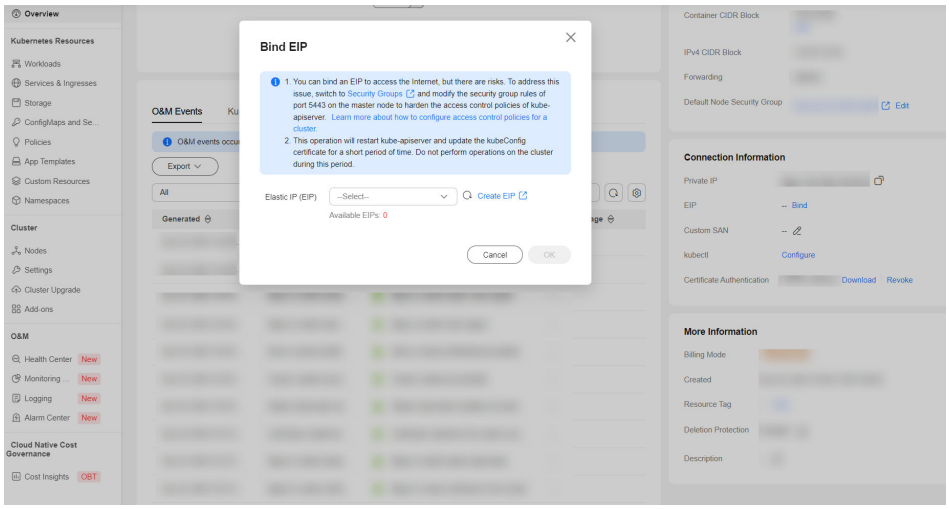
Figure 3-27 Using kubectl through the intranet



- To use kubectl through an EIP, install it on any client that associated with the EIP. First, you need to bind an EIP.

  i. Click **Bind** next to the **EIP** field.

  Figure 3-28 Binding an EIP

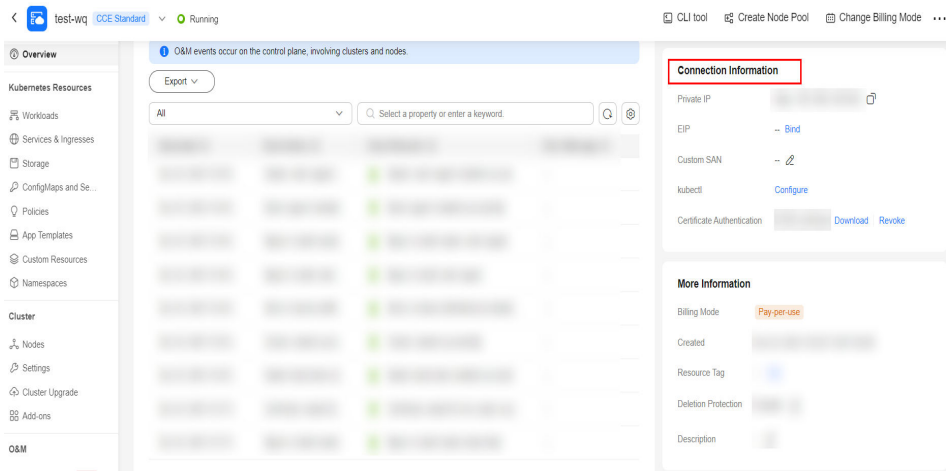
  

  ii. Select an existing EIP or click **Create EIP** to go to the EIP console and create one. For details, see **Assigning an EIP**.

iii. After the EIP is bound, locate **Connection Information** in the cluster information and click **Configure** next to **kubectl**.

iv. Perform operations as prompted. For details, see **Configure kubectl**.

Figure 3-29 Configuring kubectl



5. Run the command below on the client where kubectl is installed. If the cluster node is displayed, kubectl is configured.

```
kubectl get node
```

Follow-Up Operation

Configuring Lite Cluster Storage: The available storage space is determined by dockerBaseSize when no external storage is mounted. However, the accessible storage space is limited. It is recommended that you mount external storage to overcome this limitation. You can mount storage to a container in various methods. The recommended method depends on the scenario, and you can choose one that meets your service needs.

3.4 Configuring Lite Cluster Storage

The available storage space is determined by dockerBaseSize when no external storage is mounted. However, the accessible storage space is limited. It is recommended that you mount external storage to overcome this limitation.

You can mount storage to a container in various methods. The recommended method depends on the scenario. **Table 3-2** lists the details. For details about container storage, see **Storage Basics**. You can learn about data disk space allocation by referring to **Space Allocation of a Data Disk** and adjust the data disk size as required.

Table 3-2 Different methods of mounting storage to a container

Method	Scenario	Description	Operation Reference
EmptyDir	Training cache	Kubernetes ephemeral volumes, which are created and deleted together with Pods following the Pod lifecycle.	**Using a Temporary Path**
HostPath	This method is suitable for: 1. Containerized workload logs that need to be saved permanently 2. Containerized workloads that need to access internal data structure of the Docker engine in the host	Node storage. Multiple containers may share the storage, causing write conflicts. Deleting a Pod does not clear its storage.	**hostPath**
OBS	Training dataset storage	Object storage. The OBS SDKs are used to download sample data. Due to the large storage capacity being far from nodes, direct training speed is slow. To improve this, data is typically pulled to a local cache before training.	• **Using an Existing OBS Bucket Through a Static PV** • **Using an OBS Bucket Through a Dynamic PV**
SFS Turbo	Massive amounts of small files	• POSIX file system • Shared or interconnected VPC between the file system and resource pool • High costs	• **Using an Existing SFS Turbo File System Through a Static PV** • Dynamic mounting: not supported

Method	Scenario	Description	Operation Reference
SFS	Persistent storage for frequent reads and writes	This method applies to cost-sensitive workloads which require large-capacity scalability, such as media processing, content management, big data analytics, and workload analysis. SFS Capacity-Oriented file systems are not suitable for services with massive amounts of small files.	• **Using an Existing SFS File System Through a Static PV** • **Using an SFS File System Through a Dynamic PV**
EVS	Persistent storage	Each volume can be mounted to only one node. The storage size depends on the size of the EVS disk.	• **Using an Existing EVS Disk Through a Static PV** • **Using an EVS Disk Through a Dynamic PV**

3.5 (Optional) Configuring the Driver

Configure the corresponding driver to ensure proper use of GPU/Ascend resources in nodes within a dedicated resource pool to meet service requirements.

Lite Cluster supports two driver configuration methods:

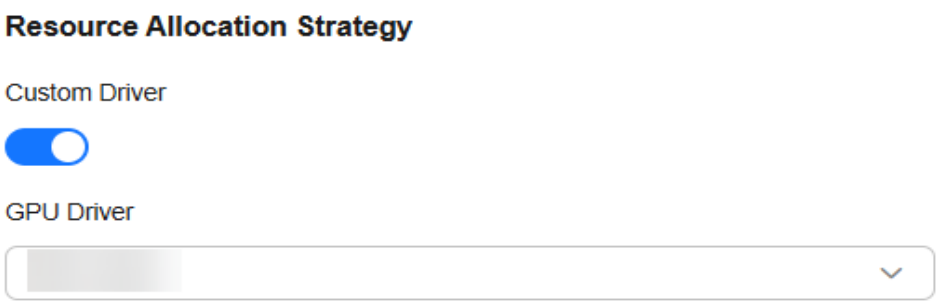
- **Method 1: Configuring a Custom Driver When Purchasing a Resource Pool**: Some GPU and Ascend resource pools allow custom drivers. Enable **Custom Driver** and select the required driver version.
- **Method 2: Upgrading the Existing Resource Pool Driver**: If no custom driver is configured and the default driver does not meet service requirements, upgrade the default driver to the required version.

Method 1: Configuring a Custom Driver When Purchasing a Resource Pool

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Standard Cluster** under **Resource Management**.
2. On the **Standard Cluster** page, click **Buy Standard Cluster**, and configure the parameters on the displayed page.

   Some GPU and Ascend resource pools allow custom driver installation. When configuring resource allocation, enable **Custom Driver**. Select the required GPU/Ascend driver from the drop-down list. For details about gpu-driver mapping versions, see **Software Versions Required by Different Models**.

Figure 3-30 GPU/Ascend driver



For details about parameters, see **Enabling Lite Cluster Resources**.

3. Click **Buy Now** and confirm the specifications. Confirm the information and click **Submit**.

Method 2: Upgrading the Existing Resource Pool Driver

If no custom driver is configured and the default driver does not meet service requirements, upgrade the default driver to the required version.

- The target Lite Cluster resource pool must be running and contains GPU or Ascend resources.
- To perform the upgrade, you need to restart the node, which is recommended to be performed during off-peak hours to avoid affecting running tasks. You can view the node usage on the **Node Management** page of the resource pool details page.

WARNING

Upgrading the driver will restart the node, which may result in the loss of any customized configurations made on the host.

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**. In the resource pool list, locate the target resource pool, and choose ··· > **Upgrade Driver**.

   Alternatively, click the resource pool name in the list to access its details page. In the navigation pane on the left, choose **Node Pool Management**. Locate the target node pool and choose **More** > **Upgrade Driver** in the **Operation** column.

2. In the displayed dialog box, you can view the driver type, number of instances, current version, target version, upgrade mode, upgrade scope, and rolling switch of the Lite Cluster resource pool.

   Set the parameters by referring to **Table 5-7**.

3. Click **OK** to start the driver upgrade.

   In the resource pool list, locate the target resource pool, and choose ··· > **Upgrade Driver**. On the displayed page, check whether the current version is the target version. If yes, the driver is upgraded.

For details, see **Upgrading the Lite Cluster Resource Pool Driver**.

Follow-Up Operation

(Optional) Configuring Image Pre-provisioning: Lite Cluster resource pools enable image pre-provisioning, which pulls images from nodes in the pools beforehand, accelerating image pulling during inference and large-scale distributed training.

3.6 (Optional) Configuring Image Pre-provisioning

Image pre-provisioning is the process of loading required images on compute nodes in advance. This can improve image loading efficiency and reduce the training job startup time.

Lite Cluster resource pools enable image pre-provisioning, which pulls images from nodes in the pools beforehand, accelerating image pulling during inference and large-scale distributed training.

This section describes how to configure image pre-provisioning on Lite Cluster.

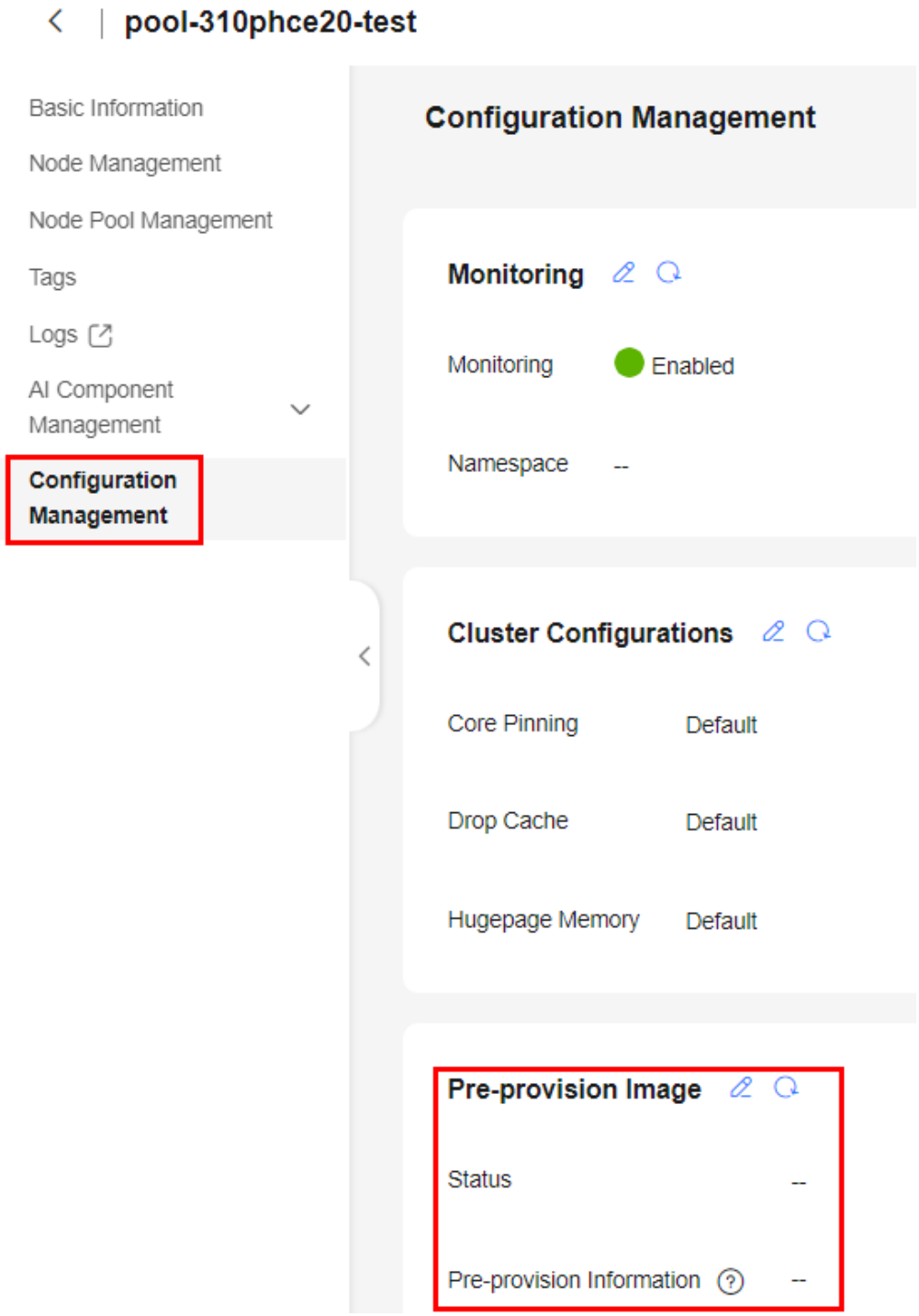
Prerequisites

- You have purchased and enabled Lite Cluster resources. For details, see **Enabling Lite Cluster Resources**.
- To obtain the image source for image pre-provisioning, you need to grant SWR operation permissions to ModelArts, so that ModelArts can use the dependent services and perform resource operations on your behalf. For details, see **Configuring Agency Authorization for ModelArts with One Click**.
- If a custom image is used for image pre-provisioning, upload the created custom image to SWR. For details, see **Pushing an Image**

Configuring Image Pre-provisioning for Lite Cluster

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.
2. Click the name of a resource pool to access its details page.
3. Click **Configuration Management** on the left.

Figure 3-31 Configuration Management



4. In **Pre-provision Image**, click and configure parameters.

Table 3-3 Parameters

Parameter	Description
Image Source	Select **Preset** or **Custom**. • **Preset**: Select an image on SWR or a shared image. • **Custom**: Enter an image path. You need to upload the created custom image to SWR in advance. For details, see **Pushing an Image**.
Image Key	To pre-provision an image that you do not have permissions on, you will need to add an image key. Once enabled, select the namespace and key. For details about how to create a key, see **Creating a Secret**. The key type must be kubernetes.io/dockerconfigjson. To create a key, refer to the tenant's SWR login command for the repository address, username, and password. **Figure 3-35** shows a temporary login command. To obtain a long-term valid login command, click **Learn how to obtain a long-term login command**. To add multiple keys, click the plus sign (+).
Add	To add multiple images, click this button.

Figure 3-32 Pre-provisioning a preset image

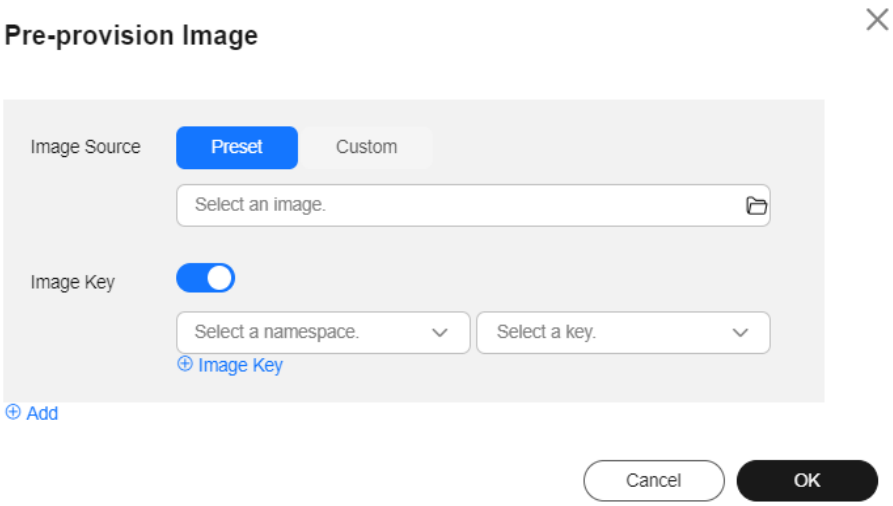


Figure 3-33 Selecting a preset image

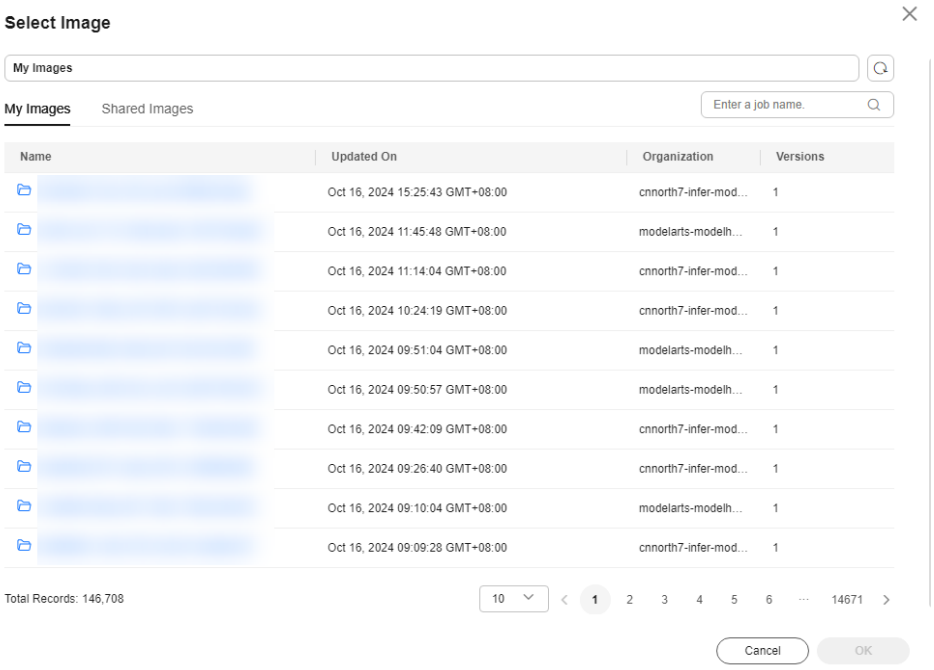


Figure 3-34 Pre-provisioning a custom image

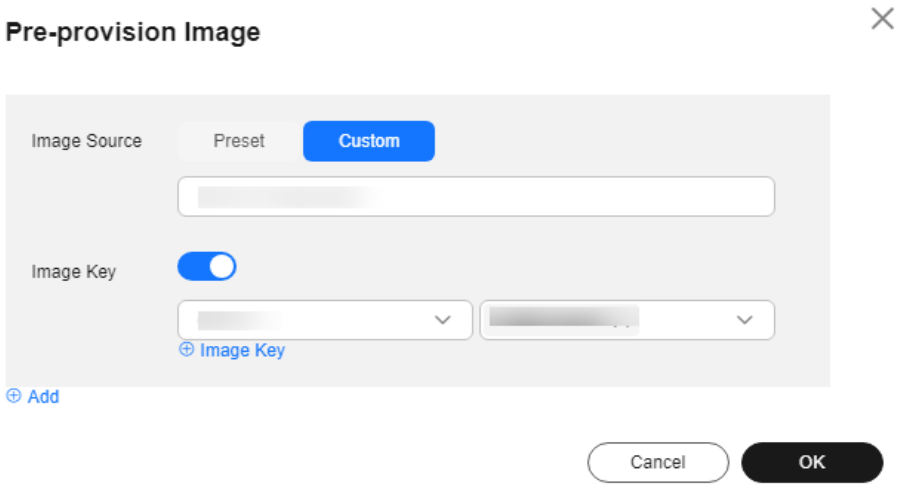
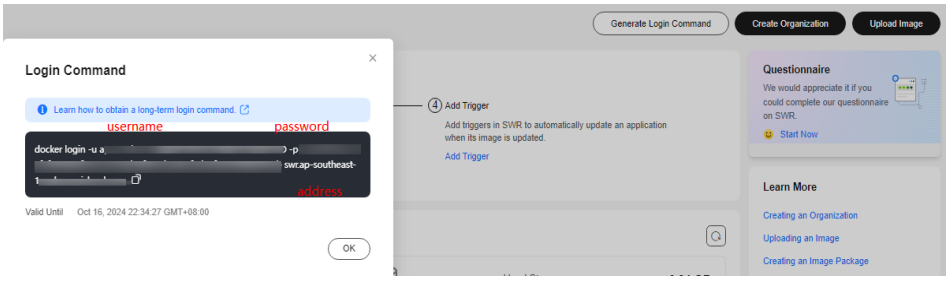


Figure 3-35 Login Command



5. Click **OK**. Then, you can see the information about the image that is pre-provisioned.

   If pre-provisioning an image failed, check whether the image path and key are correct.

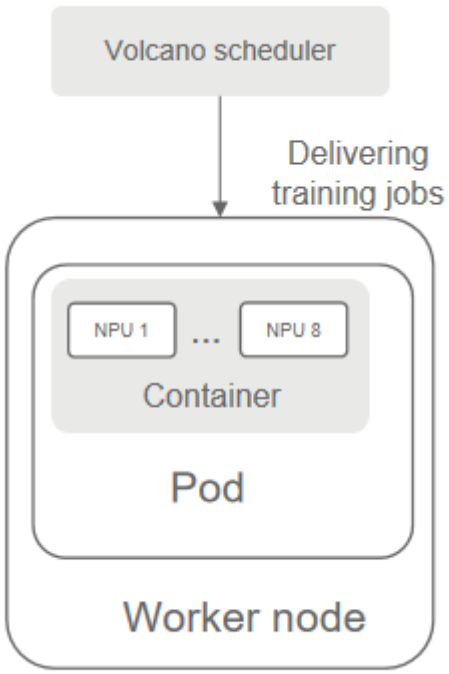
4 Using Lite Cluster Resources

4.1 Using Snt9B for Distributed Training in a Lite Cluster Resource Pool

Description

This case guides you through distributed training on Snt9B. By default, Lite Cluster resource pools come with the volcano scheduler, which delivers training jobs to clusters in volcano job mode. The BERT NLP model is used in the training test cases.

Figure 4-1 Delivering training jobs



Procedure

Step 1 Pull the image. The test image is bert_pretrain_mindspore:v1, which contains the test data and code.

```
docker pull swr.cn-southwest-2.myhuaweicloud.com/os-public-repo/bert_pretrain_mindspore:v1
docker tag swr.cn-southwest-2.myhuaweicloud.com/os-public-repo/bert_pretrain_mindspore:v1
bert_pretrain_mindspore:v1
```

Step 2 Create the **config.yaml** file on the host.

Configure Pods using this file. For debugging, start a Pod with the **sleep** command. Alternatively, replace the command with the boot command for your job (for example, **python train.py**). The job will run once the container starts.

The file content is as follows:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: configmap1980-yourvcjobname     #The prefix is configmap1980-, followed by the vcjob name.
  namespace: default                    #Namespace, which is optional and must be in the same namespace as
vcjob.
  labels:
    ring-controller.cce: ascend-1980   # Retain the default settings.
data:            # The data content remains unchanged. After the initialization is complete, the data content is
automatically modified by the Volcano plug-in.
  jobstart_hccl.json: |
    {
        "status":"initializing"
    }
---
apiVersion: batch.volcano.sh/v1alpha1   # The value cannot be changed. The volcano API must be used.
kind: Job                               # Only the job type is supported at present.
metadata:
  name: yourvcjobname                   # Job name, which must be the same as that in configmap.
namespace: default             # The value must be the same as that of ConfigMap.
  labels:
    ring-controller.cce: ascend-1980        # Retain the default settings.
    fault-scheduling: "force"
spec:
  minAvailable: 1                       # The value of minAvailable is 1 in a single-node scenario and N in an N-
node distributed scenario.
schedulerName: volcano          # Retain the default settings. Use the Volcano scheduler to schedule jobs.
  policies:
    - event: PodEvicted
      action: RestartJob
  plugins:
    configmap1980:
    - --rank-table-version=v2          # Retain the default settings. The ranktable file of the v2 version is
generated.
    env: []
    svc:
    - --publish-not-ready-addresses=true
  maxRetry: 3
  queue: default
  tasks:
  - name: "yourvcjobname-1"
    replicas: 1                         # The value of replicas is 1 in a single-node scenario and N in an N-node
scenario. The number of NPUs in the requests field is 8 in an N-node scenario.
    template:
      metadata:
        labels:
          app: mindspore
ring-controller.cce: ascend-1980       # Retain the default value. The value must be the same as the label in
ConfigMap and cannot be changed.
      spec:
        affinity:
```

```
        podAntiAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            - labelSelector:
                matchExpressions:
                  - key: volcano.sh/job-name
                    operator: In
                    values:
                      - yourvcjobname
              topologyKey: kubernetes.io/hostname
      containers:
- image: bert_pretrain_mindspore:v1        # Training framework image path, which can be modified.
        imagePullPolicy: IfNotPresent
        name: mindspore
        env:
        - name: name                        # The value must be the same as that of Jobname.
          valueFrom:
            fieldRef:
              fieldPath: metadata.name
        - name: ip                          # IP address of the physical node, which is used to identify the
node where the pod is running
          valueFrom:
            fieldRef:
              fieldPath: status.hostIP
        - name: framework
          value: "MindSpore"
        command:
        - "sleep"
        - "10000000000000000000"
        resources:
          requests:
            huawei.com/ascend-1980: "1"               # Number of required PUs. The keys remain the same.
Number of required NPUs. The maximum value is 16. You can add lines below to configure resources such
as memory and CPU.
          limits:
            huawei.com/ascend-1980: "1"               # Limit the number of PUs. The keys remain the same.
The value must be consistent with that in requests.
        volumeMounts:
        - name: ascend-driver             # Mount driver. Retain the settings.
          mountPath: /usr/local/Ascend/driver
        - name: ascend-add-ons            # Mount driver. Retain the settings.
          mountPath: /usr/local/Ascend/add-ons
        - name: localtime
          mountPath: /etc/localtime
        - name: hccn                      #  HCCN configuration of the driver. Retain the settings.
          mountPath: /etc/hccn.conf
        - name: npu-smi                   #npu-smi
          mountPath: /usr/local/sbin/npu-smi
      nodeSelector:
        accelerator/huawei-npu: ascend-1980
      volumes:
      - name: ascend-driver
        hostPath:
          path: /usr/local/Ascend/driver
      - name: ascend-add-ons
        hostPath:
          path: /usr/local/Ascend/add-ons
      - name: localtime
        hostPath:
          path: /etc/localtime                # Configure the Docker time.
      - name: hccn
        hostPath:
          path: /etc/hccn.conf
      - name: npu-smi
        hostPath:
          path: /usr/local/sbin/npu-smi
      restartPolicy: OnFailure
```

Step 3 Create a pod based on the **config.yaml** file.

```
kubectl apply -f config.yaml
```


Step 4 Run the following command to check the pod startup status. If **1/1 running** is displayed, the startup is successful.

```
kubectl get pod -A
```

Step 5 Go to the container, replace {pod_name} with your pod name (displayed by the **get pod** command), and replace {namespace} with your namespace (default).

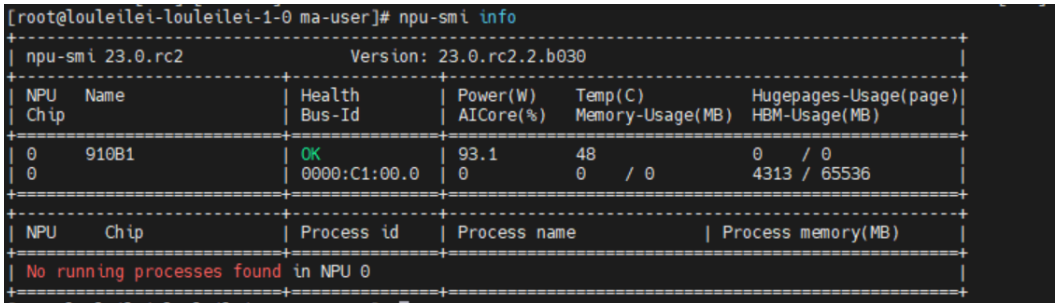
```
kubectl exec -it {pod_name} bash -n {namespace}
```

Step 6 Run the following command to view the NPU information:

```
npu-smi info
```

Kubernetes allocates resources to pods according to the number of NPUs specified in the **config.yaml** file. As illustrated in the figure below, only one NPU is displayed in the container, reflecting the single NPU configuration. This confirms that the configuration is effective.

Figure 4-2 Viewing NPU information



Step 7 Change the number of NPUs in the pod. In this example, distributed training is used. The number of required NPUs is changed to 8.

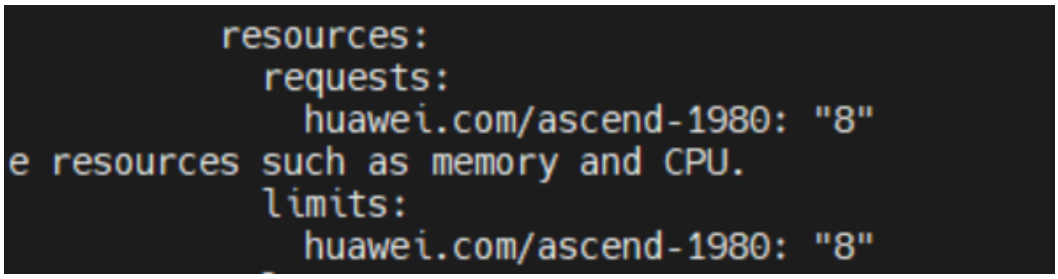
Delete the created pod.

```
kubectl delete -f config.yaml
```

Change the values of **limit** and **request** in the **config.yaml** file to 8.

```
vi config.yaml
```

Figure 4-3 Modify the number of NPUs



Re-create a pod.

```
kubectl apply -f config.yaml
```

Go to the container and view the NPU information. Replace {pod_name} with your pod name and {namespace} with your namespace (default).

```
kubectl exec -it {pod_name} bash -n {namespace}
npu-smi info
```


As shown in the following figure, 8 NPUs are used and the pod is successfully configured.

Figure 4-4 Viewing NPU information

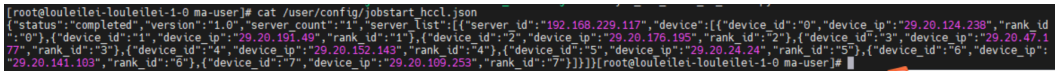
```
[root@os-node-created-ljknq ~]# kubectl get pod
NAME                                READY   STATUS    RESTARTS        AGE
maos-node-agent-gqrvs               2/2     Running   32 (3d2h ago)   3d4h
yourvcjobname-yourvcjobname-1-0     1/1     Running   0               52s
[root@os-node-created-ljknq ~]# kubectl exec -it yourvcjobname-yourvcjobname-1-0 bash -n default
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use kubectl exec [POD] -- [COMMAND]
 instead.
[root@yourvcjobname-yourvcjobname-1-0 ma-user]# npu-smi info
+------------------------------------------------------------------------------------------------+
| npu-smi 23.0.rc2                 Version: 23.0.rc2.2                                           |
+---------------------------+---------------+----------------------------------------------------+
| NPU   Name                | Health        | Power(W)    Temp(C)           Hugepages-Usage(page)|
| Chip                      | Bus-Id        | AICore(%)   Memory-Usage(MB)  HBM-Usage(MB)        |
+===========================+===============+====================================================+
| 0     910B4               | OK            | 83.7        48                0    / 0             |
| 0                         | 0000:C1:00.0  | 0           0    / 0          3151 / 32768         |
+===========================+===============+====================================================+
| 1     910B4               | OK            | 83.8        48                0    / 0             |
| 0                         | 0000:01:00.0  | 0           0    / 0          3148 / 32768         |
+===========================+===============+====================================================+
| 2     910B4               | OK            | 83.8        45                0    / 0             |
| 0                         | 0000:C2:00.0  | 0           0    / 0          3149 / 32768         |
+===========================+===============+====================================================+
| 3     910B4               | OK            | 87.6        48                0    / 0             |
| 0                         | 0000:02:00.0  | 0           0    / 0          3147 / 32768         |
+===========================+===============+====================================================+
| 4     910B4               | OK            | 83.8        45                0    / 0             |
| 0                         | 0000:81:00.0  | 0           0    / 0          3148 / 32768         |
+===========================+===============+====================================================+
| 5     910B4               | OK            | 83.8        46                0    / 0             |
| 0                         | 0000:41:00.0  | 0           0    / 0          3148 / 32768         |
+===========================+===============+====================================================+
| 6     910B4               | OK            | 83.7        46                0    / 0             |
| 0                         | 0000:82:00.0  | 0           0    / 0          3147 / 32768         |
+===========================+===============+====================================================+
| 7     910B4               | OK            | 92.6        49                0    / 0             |
| 0                         | 0000:42:00.0  | 0           0    / 0          3148 / 32768         |
+===========================+===============+====================================================+

+---------------------------+---------------+----------------------------------------------------+
| NPU     Chip              | Process id    | Process name             | Process memory(MB)      |
+===========================+===============+====================================================+
```

Step 8 Run the following command to view the inter-NPU communication configuration file:

```
cat /user/config/jobstart_hccl.json
```

During multi-NPU training, the **rank_table_file** configuration file is essential for inter-NPU communication. This file is automatically generated and provides the file address once the pod is initiated. It takes a period of time to generate the **/user/config/jobstart_hccl.json** and **/user/config/jobstart_hccl.json** configuration files. The service process can generate the inter-NPU communication information only after the status field in **/user/config/jobstart_hccl.json** is **completed**. The process is shown in the figure below.

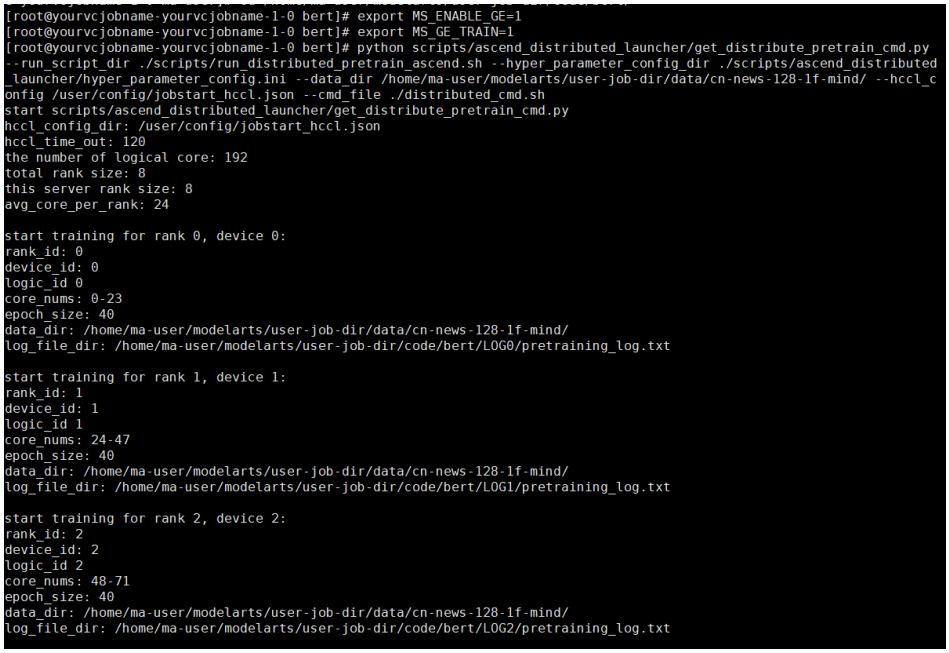
Figure 4-5 Inter-NPU communication configuration file



Step 9 Start a training job.

```
cd /home/ma-user/modelarts/user-job-dir/code/bert/
export MS_ENABLE_GE=1
export MS_GE_TRAIN=1
python scripts/ascend_distributed_launcher/get_distribute_pretrain_cmd.py --run_script_dir ./scripts/
run_distributed_pretrain_ascend.sh --hyper_parameter_config_dir ./scripts/ascend_distributed_launcher/
hyper_parameter_config.ini --data_dir /home/ma-user/modelarts/user-job-dir/data/cn-news-128-1f-mind/ --
hccl_config /user/config/jobstart_hccl.json --cmd_file ./distributed_cmd.sh
bash scripts/run_distributed_pretrain_ascend.sh /home/ma-user/modelarts/user-job-dir/data/cn-
news-128-1f-mind/ /user/config/jobstart_hccl.json
```

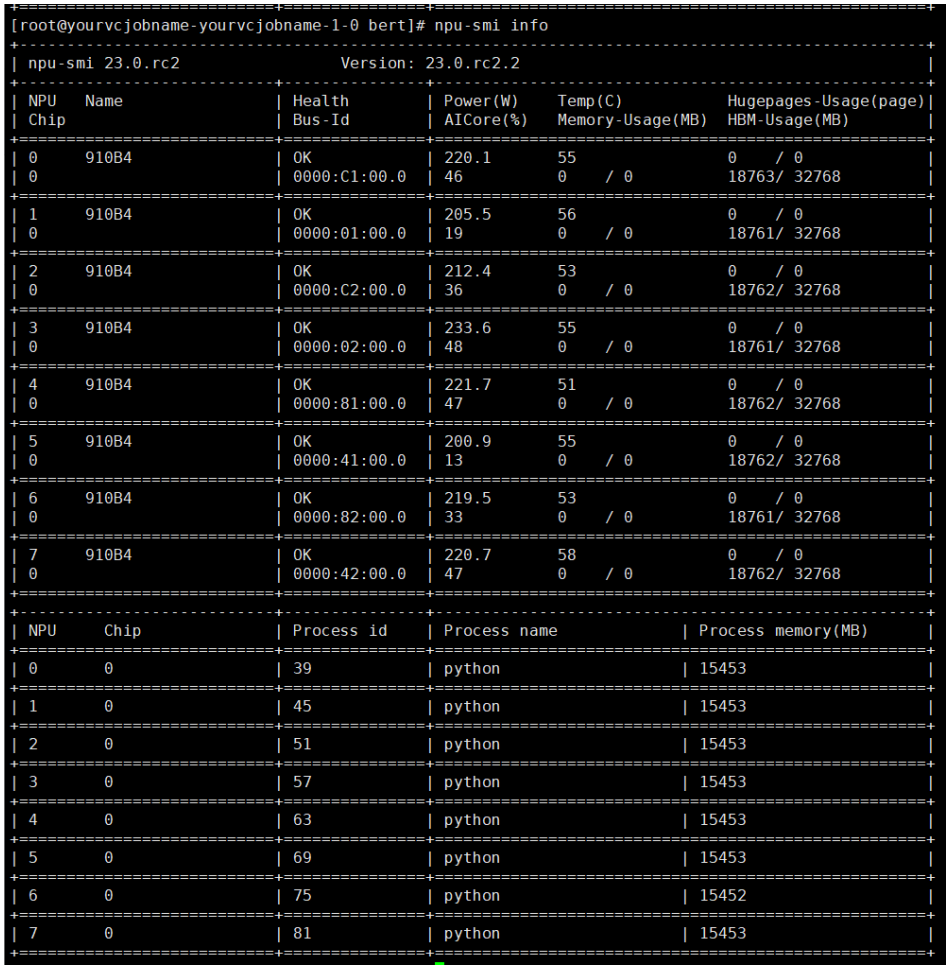

Figure 4-6 Starting a training job



It takes some time to load a training job. After several minutes, run the following command to view the NPU information. As shown in the following figure, all the eight NPUs are occupied, indicating that the training task is in progress.

```
npu-smi info
```


Figure 4-7 Viewing NPU information

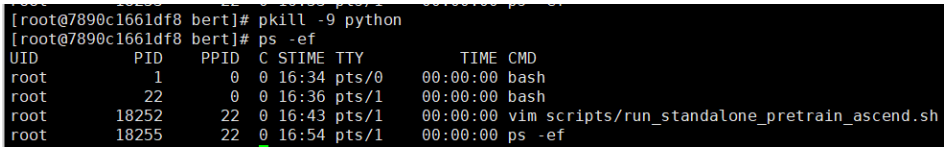


```
[root@yourvcjobname-yourvcjobname-1-0 bert]# npu-smi info
+------------------------------------------------------------------------------------------------+
| npu-smi 23.0.rc2                 Version: 23.0.rc2.2                                           |
+---------------------------+---------------+----------------------------------------------------+
| NPU   Name                | Health        | Power(W)    Temp(C)           Hugepages-Usage(page)|
| Chip                      | Bus-Id        | AICore(%)   Memory-Usage(MB)  HBM-Usage(MB)        |
+===========================+===============+====================================================+
| 0     910B4               | OK            | 220.1       55                0    / 0             |
| 0                         | 0000:C1:00.0  | 46          0    / 0          18763/ 32768         |
+===========================+===============+====================================================+
| 1     910B4               | OK            | 205.5       56                0    / 0             |
| 0                         | 0000:01:00.0  | 19          0    / 0          18761/ 32768         |
+===========================+===============+====================================================+
| 2     910B4               | OK            | 212.4       53                0    / 0             |
| 0                         | 0000:C2:00.0  | 36          0    / 0          18762/ 32768         |
+===========================+===============+====================================================+
| 3     910B4               | OK            | 233.6       55                0    / 0             |
| 0                         | 0000:02:00.0  | 48          0    / 0          18761/ 32768         |
+===========================+===============+====================================================+
| 4     910B4               | OK            | 221.7       51                0    / 0             |
| 0                         | 0000:81:00.0  | 47          0    / 0          18762/ 32768         |
+===========================+===============+====================================================+
| 5     910B4               | OK            | 200.9       55                0    / 0             |
| 0                         | 0000:41:00.0  | 13          0    / 0          18762/ 32768         |
+===========================+===============+====================================================+
| 6     910B4               | OK            | 219.5       53                0    / 0             |
| 0                         | 0000:82:00.0  | 33          0    / 0          18761/ 32768         |
+===========================+===============+====================================================+
| 7     910B4               | OK            | 220.7       58                0    / 0             |
| 0                         | 0000:42:00.0  | 47          0    / 0          18762/ 32768         |
+===========================+===============+====================================================+
+---------------------------+---------------+----------------------------------------------------+
| NPU     Chip              | Process id    | Process name             | Process memory(MB)      |
+===========================+===============+==========================+=========================+
| 0       0                 | 39            | python                   | 15453                   |
+===========================+===============+==========================+=========================+
| 1       0                 | 45            | python                   | 15453                   |
+===========================+===============+==========================+=========================+
| 2       0                 | 51            | python                   | 15453                   |
+===========================+===============+==========================+=========================+
| 3       0                 | 57            | python                   | 15453                   |
+===========================+===============+==========================+=========================+
| 4       0                 | 63            | python                   | 15453                   |
+===========================+===============+==========================+=========================+
| 5       0                 | 69            | python                   | 15453                   |
+===========================+===============+==========================+=========================+
| 6       0                 | 75            | python                   | 15452                   |
+===========================+===============+==========================+=========================+
| 7       0                 | 81            | python                   | 15453                   |
+===========================+===============+==========================+=========================+
```

To stop a training task, run the commands below:

```
pkill -9 python
ps -ef
```

Figure 4-8 Stopping the training process



```
[root@7890c1661df8 bert]# pkill -9 python
[root@7890c1661df8 bert]# ps -ef
UID          PID    PPID  C STIME TTY          TIME CMD
root           1       0  0 16:34 pts/0    00:00:00 bash
root          22       0  0 16:36 pts/1    00:00:00 bash
root       18252      22  0 16:43 pts/1    00:00:00 vim scripts/run_standalone_pretrain_ascend.sh
root       18255      22  0 16:54 pts/1    00:00:00 ps -ef
```

NOTE

Set **limit** and **request** to proper values to restrict the number of CPUs and memory size. A single Snt9B node is equipped with eight Snt9B PUs and 192 vCPUs 1,536 GB. Properly plan the CPU and memory allocations to avoid task failures due to insufficient CPU and memory limits.

----End

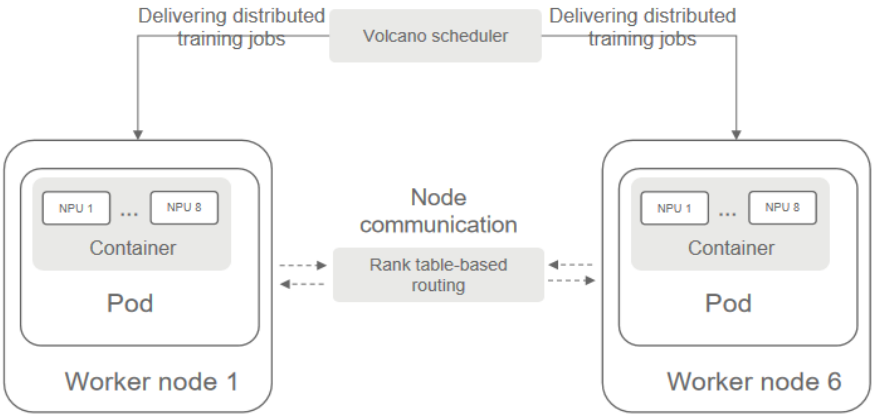
4.2 Performing PyTorch NPU Distributed Training In a ModelArts Lite Resource Pool Using Ranktable-based Route Planning

Description

The ranktable route planning is a communication optimization capability used in distributed parallel training. When NPUs are used, network route affinity planning can be performed for communication paths between nodes based on the actual switch topology, improving the communication speed between nodes.

This case describes how to complete a PyTorch NPU distributed training task in ModelArts Lite using ranktable route planning. By default, training tasks are delivered to the Lite resource pool cluster in Volcano job mode.

Figure 4-9 Job delivering



Constraints

- This function is available only in CN Southwest-Guiyang1. If you want to use it in another region, contact technical support.
- The Volcano plug-ins of 1.10.12 or later must be installed in the CCE cluster corresponding to the ModelArts Lite resource pool. For details about how to install and upgrade a Volcano scheduler, see **Volcano Scheduler**. Only Huawei Cloud Volcano plug-ins support route acceleration.
- Python 3.7 or 3.9 must be used for training. Otherwise, the ranktable route cannot be used for accelerating.
- There must be at least three task nodes in a training job. Otherwise, the ranktable route will be skipped. Use ranktable route in large model scenarios, that is, there are 512 cards or more.
- The script execution directory cannot be a shared directory. Otherwise, the ranktable route will fail.

- To use ranktable route is to change the rank number. Therefore, the rank in codes must be unified. Otherwise, the training will be abnormal.

Procedure

Step 1 Enable the cabinet plug-in of the CCE cluster corresponding to the ModelArts Lite resource pool.

1. In the ModelArts Lite dedicated resource pool list, click the resource pool name to view its details.
2. On the displayed page, click the CCE cluster.
3. In the navigation pane on the left, choose **Add-ons**, and search for **Volcano Scheduler**.
4. Click **Edit** and check whether **{"name":"cabinet"}** exists in the **plugins** parameter.
   - If **{"name":"cabinet"}** exists, go to **Step 2**.
   - If **{"name":"cabinet"}** does not exist, add it to the **plugins** parameter in the advanced settings, and click **Install**.

Step 2 Modify the **torch_npu** training startup script.

NOTICE

You can only run the **torch.distributed.launch/run** command to start up the script. Otherwise, the ranktable route cannot be used for accelerating.

During PyTorch training, you need to set **NODE_RANK** to the value of the environment variable **RANK_AFTER_ACC**. The following shows an example of a training startup script (*xxx*_**train.sh**): **MASTER_ADDR** and **NODE_RANK** must retain these values.

```
#!/bin/bash

# MASTER_ADDR
MASTER_ADDR="${MA_VJ_NAME}-${MA_TASK_NAME}-${MA_MASTER_INDEX:-0}.${MA_VJ_NAME}"
NODE_RANK="${RANK_AFTER_ACC:-$VC_TASK_INDEX}"
NNODES="$MA_NUM_HOSTS"
NGPUS_PER_NODE="$MA_NUM_GPUS"
# self-define, it can be changed to >=10000 port
MASTER_PORT="39888"

# replace ${MA_JOB_DIR}/code/torch_ddp.py to the actual training script
PYTHON_SCRIPT=${MA_JOB_DIR}/code/torch_ddp.py
PYTHON_ARGS=""

# set hccl timeout time in seconds
export HCCL_CONNECT_TIMEOUT=1800

# replace ${ANACONDA_DIR}/envs/${ENV_NAME}/bin/python to the actual python
CMD="${ANACONDA_DIR}/envs/${ENV_NAME}/bin/python -m torch.distributed.launch \
    --nnodes=$NNODES \
    --node_rank=$NODE_RANK \
    --nproc_per_node=$NGPUS_PER_NODE \
    --master_addr $MASTER_ADDR \
    --master_port=$MASTER_PORT \
    $PYTHON_SCRIPT \
    $PYTHON_ARGS
"
```



```
echo $CMD
$CMD
```

Step 3 Create the **config.yaml** file on the host.

The **config.yaml** file is used to configure pods. The following shows a code example. ***xxxx_train.sh*** indicates the modified training startup script in **Step 2**.

```
apiVersion: batch.volcano.sh/v1alpha1
kind: Job
metadata:
  name: yourvcjobname                # Job name
  namespace: default           # Namespace
  labels:
    ring-controller.cce: ascend-1980   # Retain the default settings.
    fault-scheduling: "force"
spec:
  minAvailable: 6                    # Number of nodes used for distributed training
  schedulerName: volcano               # Retain the default settings.
  policies:
    - event: PodEvicted
      action: RestartJob
  plugins:
    configmap1980:
    - --rank-table-version=v2          # Retain the default settings. The ranktable file of the v2 version is generated.
    env: []
    svc:
    - --publish-not-ready-addresses=true   # Retain the default settings. It is used for the communication between pods. Certain required environment variables are generated.
  maxRetry: 1
  queue: default
  tasks:
  - name: "worker"  # Retain the default settings.
    replicas: 6                        # Number of tasks, which is the number of nodes in PyTorch. Set this to the value of ***minAvailable***.
      template:
        metadata:
          annotations:
          cabinet: "cabinet"  # Retain the default settings. Enable tor-topo delivery.
          labels:
          app: pytorch-npu   # Tag
          ring-controller.cce: ascend-1980  # Retain the default settings.
        spec:
          affinity:
            podAntiAffinity:
              requiredDuringSchedulingIgnoredDuringExecution:
                - labelSelector:
                    matchExpressions:
                      - key: volcano.sh/job-name
                        operator: In
                        values:
                        - yourvcjobname   # Job name
                  topologyKey: kubernetes.io/hostname
          containers:
        - image:  swr.xxxxxx.com/xxxx/custom_pytorch_npu:v1            # Image address
            imagePullPolicy: IfNotPresent
          name: pytorch-npu          # Container name
            env:
          - name: OPEN_SCRIPT_ADDRESS     # Open script address. Set ***region-id*** based on the actual-life scenario, for example, ***cn-southwest-2***.
                value: "https://mtest-bucket.obs.{region-id}.myhuaweicloud.com/acc/rank"
              - name: NAME
                valueFrom:
                  fieldRef:
                    fieldPath: metadata.name
          - name: MA_CURRENT_HOST_IP                    # Retain the default settings. This indicates the IP address of the node where the current pod is deployed.
                valueFrom:
                  fieldRef:
```



```
                fieldPath: status.hostIP
            - name: MA_NUM_GPUS     # Number of NPUs used by each pod
              value: "8"
          - name: MA_NUM_HOSTS   # Number of nodes used in the distributed training. Set this to the value of minAvailable.
              value: "6"
- name: MA_VJ_NAME              # Name of the volcano job.
              valueFrom:
                fieldRef:
                  fieldPath: metadata.annotations['volcano.sh/job-name']
            - name: MA_TASK_NAME # Name of the task pod.
              valueFrom:
                fieldRef:
                  fieldPath: metadata.annotations['volcano.sh/task-spec']
          command:
            - /bin/bash
            - -c
- Replace "wget ${OPEN_SCRIPT_ADDRESS}/bootstrap.sh -q && bash bootstrap.sh; export RANK_AFTER_ACC=${VC_TASK_INDEX}; rank_acc=$(cat /tmp/RANK_AFTER_ACC 2>/dev/null); [ -n \"${rank_acc}\" ] && export RANK_AFTER_ACC=${rank_acc};export MA_MASTER_INDEX=$(cat /tmp/MASTER_INDEX 2>/dev/null || echo 0); bash xxxx_train.sh"         # Replace xxxx_train.sh with the actual training script path.
          resources:
            requests:
              huawei.com/ascend-1980: "8"               # Number of PUs required by each node, which is the value of MA_NUM_GPUS. The keys remain the same.
            limits:
              huawei.com/ascend-1980: "8"               # Maximum number of PUs on each node, which is the value of MA_NUM_GPUS. The keys remain the same.
          volumeMounts:
        - name: ascend-driver             #Mount driver. Retain the settings.
            mountPath: /usr/local/Ascend/driver
        - name: ascend-add-ons          #Mount driver. Retain the settings.
            mountPath: /usr/local/Ascend/add-ons
          - name: localtime
            mountPath: /etc/localtime
        - name: hccn                        # HCCN configuration of the driver. Retain the settings.
            mountPath: /etc/hccn.conf
          - name: npu-smi
            mountPath: /usr/local/sbin/npu-smi
        nodeSelector:
          accelerator/huawei-npu: ascend-1980
        volumes:
          - name: ascend-driver
            hostPath:
              path: /usr/local/Ascend/driver
          - name: ascend-add-ons
            hostPath:
              path: /usr/local/Ascend/add-ons
          - name: localtime
            hostPath:
              path: /etc/localtime
          - name: hccn
            hostPath:
              path: /etc/hccn.conf
          - name: npu-smi
            hostPath:
              path: /usr/local/sbin/npu-smi
        restartPolicy: OnFailure
```

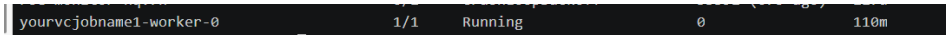
Step 4 Run the following command to create and start the pod based on **config.yaml**. After the container is started, the training job is automatically executed.

```
kubectl apply -f config.yaml
```

Step 5 Run the following command to check the pod startup status. If **1/1 running** is displayed, the startup is successful.

```
kubectl get pod
```


Figure 4-10 Command output of successful startup

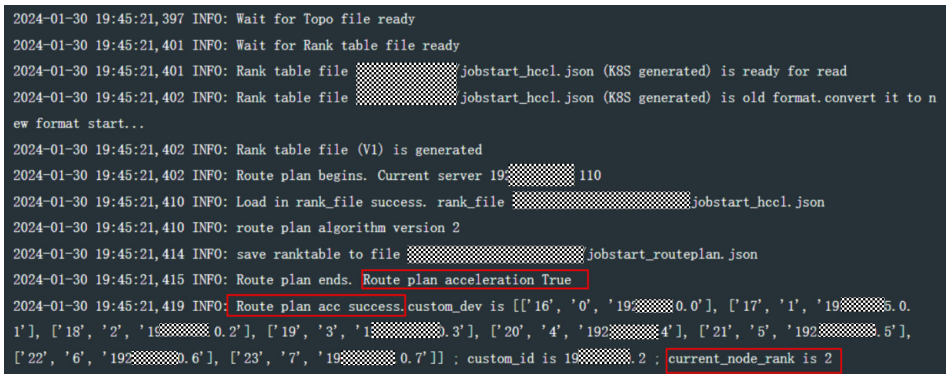


Step 6 Run the following command to view the logs. If **Figure 4-11** is displayed, the route is executed.

```
kubectl logs {pod-name}
```

Replace *{pod-name}* with the actual pod name, which can be obtained from the output in **Step 5**.

Figure 4-11 Command output of executed dynamic route



NOTE

- Dynamic routing can be executed only if there are at least three training nodes in a training task.
- If the execution fails, rectify the fault by referring to **Troubleshooting: ranktable Route Optimization Fails**.

----End

Troubleshooting: ranktable Route Optimization Fails

Symptom

There is error information in the container logs.

Possible Causes

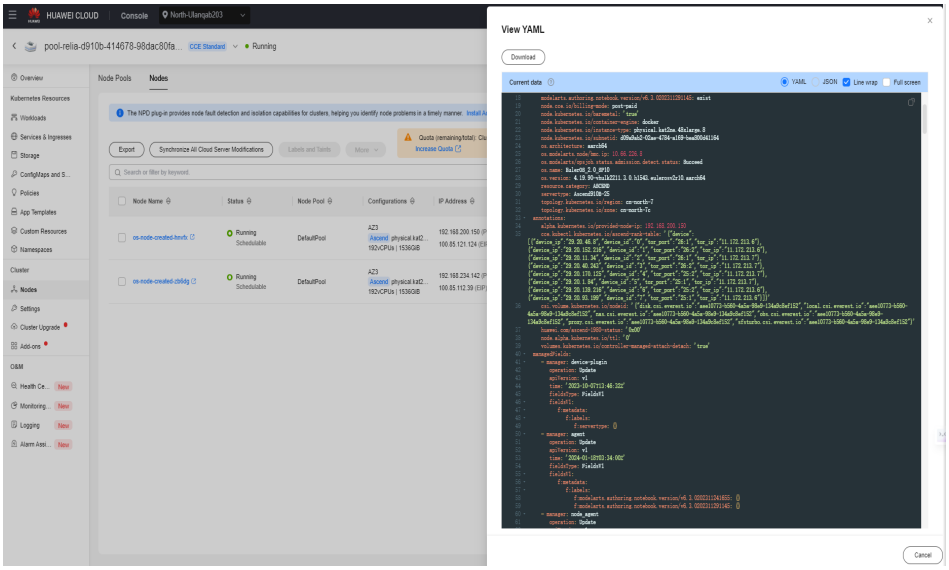
The cluster node does not deliver the **topo** file and **ranktable** file.

Procedure

1. In the ModelArts Lite dedicated resource pool list, click the resource pool name to view its details.
2. On the displayed page, click the CCE cluster.
3. In the navigation pane on the left, choose **Nodes**, and go to the **Nodes** tab.
4. In the node list, locate the target node, and choose **More** > **View YAML** in the **Operation** column.
5. Check whether the **cce.kubectl.kubernetes.io/ascend-rank-table** field in the **yaml** file has a value.

As shown in the following figure, if there is a value, delivering the **topo** file and **ranktable** file has been enabled on the node. Otherwise, contact technical support.

Figure 4-12 Viewing the **YAML** file of a node

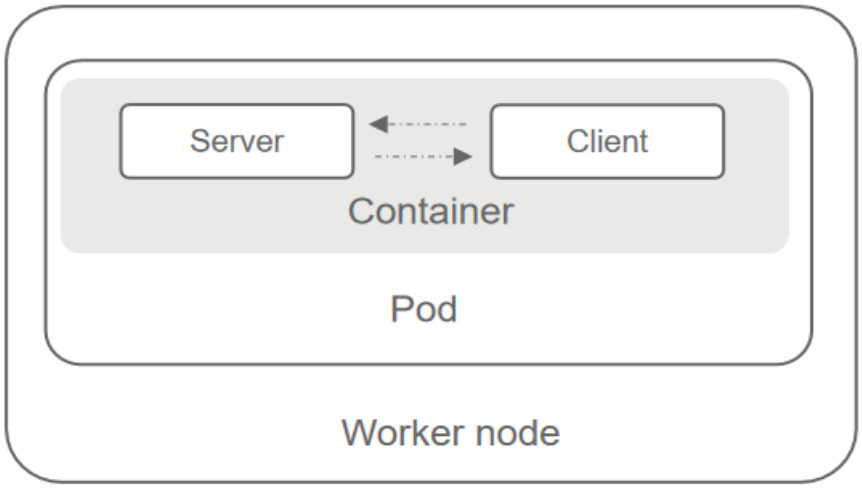


4.3 Using Snt9B for Inference in a Lite Cluster Resource Pool

Description

This case outlines the process of using the Deployment mechanism to deploy a real-time inference service in the Snt9B environment. Create a pod to host the service, log in to the pod container to deploy the real-time service, and create a terminal as the client to access the service to test its functions.

Figure 4-13 Task diagram



Procedure

Step 1 Pulls the image. The test image is **bert_pretrain_mindspore:v1**, which contains the test data and code.

```
docker pull swr.cn-southwest-2.myhuaweicloud.com/os-public-repo/bert_pretrain_mindspore:v1
docker tag swr.cn-southwest-2.myhuaweicloud.com/os-public-repo/bert_pretrain_mindspore:v1
bert_pretrain_mindspore:v1
```

Step 2 Create the **config.yaml** file on the host.

Configure pods using this file. For debugging, start a pod using the **sleep** command. Alternatively, replace the command with the boot command for your job (for example, **python inference.py**). The job will run once the container starts.

The file content is as follows:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: yourapp
  labels:
     app: infers
spec:
  replicas: 1
  selector:
    matchLabels:
      app: infers
  template:
    metadata:
      labels:
        app: infers
    spec:
      schedulerName: volcano
      nodeSelector:
        accelerator/huawei-npu: ascend-1980
      containers:
      - image: bert_pretrain_mindspore:v1                # Inference image name
        imagePullPolicy: IfNotPresent
        name: mindspore
        command:
        - "sleep"
        - "1000000000000000000"
        resources:
          requests:
            huawei.com/ascend-1980: "1"            # Number of required PUs. The keys remain the same. Number of required NPUs. The maximum value is 16. You can add lines below to configure resources such as memory and CPU.
          limits:
            huawei.com/ascend-1980: "1"            # Limit the number of PUs. The keys remain the same. The value must be consistent with that in requests.
        volumeMounts:
          - name: ascend-driver              # Mount driver. Retain the settings.
          mountPath: /usr/local/Ascend/driver
          - name: ascend-add-ons           # Mount driver. Retain the settings.
          mountPath: /usr/local/Ascend/add-ons
          - name: hccn                        # HCCN configuration of the driver. Retain the settings.
          mountPath: /etc/hccn.conf
        - name: npu-smi                          #npu-smi
          mountPath: /usr/local/sbin/npu-smi
        - name: localtime                    #The container time must be the same as the host time.
          mountPath: /etc/localtime
      volumes:
      - name: ascend-driver
        hostPath:
          path: /usr/local/Ascend/driver
      - name: ascend-add-ons
```



```
      hostPath:
        path: /usr/local/Ascend/add-ons
    - name: hccn
      hostPath:
        path: /etc/hccn.conf
    - name: npu-smi
      hostPath:
        path: /usr/local/sbin/npu-smi
    - name: localtime
      hostPath:
        path: /etc/localtime
```

Step 3 Create a pod based on the **config.yaml** file.

```
kubectl apply -f config.yaml
```

Step 4 Run the following command to check the pod startup status. If **1/1 running** is displayed, the startup is successful.

```
kubectl get pod -A
```

Step 5 Go to the container, replace {pod_name} with your pod name (displayed by the **get pod** command), and replace {namespace} with your namespace (default).

```
kubectl exec -it {pod_name} bash -n {namespace}
```

Step 6 Activate the conda mode.

```
su - ma-user //Switch the user identity.
conda activate MindSpore //Activate the MindSpore environment.
```

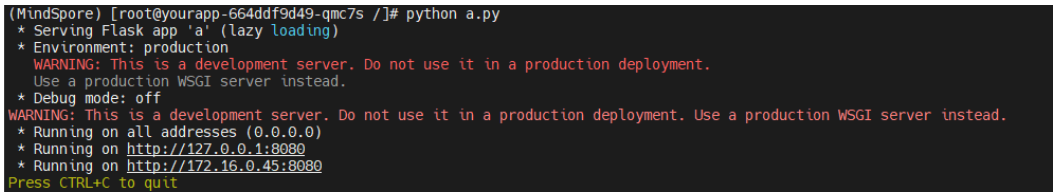
Step 7 Create test code **test.py**.

```
from flask import Flask, request
import json
app = Flask(__name__)

@app.route('/greet', methods=['POST'])
def say_hello_func():
    print("----------- in hello func ----------")
    data = json.loads(request.get_data(as_text=True))
    print(data)
    username = data['name']
    rsp_msg = 'Hello, {}!'.format(username)
    return json.dumps({"response":rsp_msg}, indent=4)

@app.route('/goodbye', methods=['GET'])
def say_goodbye_func():
    print("----------- in goodbye func ----------")
    return '\nGoodbye!\n'


@app.route('/', methods=['POST'])
def default_func():
    print("----------- in default func ----------")
    data = json.loads(request.get_data(as_text=True))
    return '\n called default func !\n {} \n'.format(str(data))

# host must be "0.0.0.0", port must be 8080
if __name__ == '__main__':
    app.run(host="0.0.0.0", port=8080)
```

Execute the code. After the code is executed, a real-time service is deployed. The container is the server.

```
python test.py
```

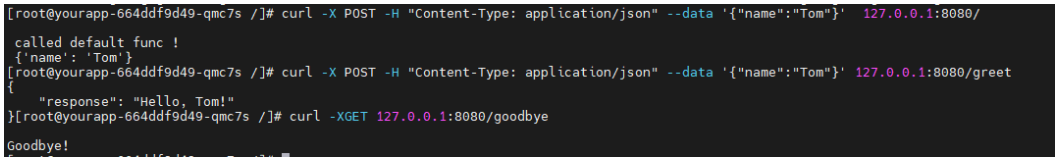

Figure 4-14 Deploying a real-time service



Step 8 Open a terminal in XShell and access the container (client) by referring to steps 5 to 7. Run the following commands to test the functions of the three APIs of the custom image. If the following information is displayed, the service is successfully invoked.

```
curl -X POST -H "Content-Type: application/json" --data '{"name":"Tom"}'  127.0.0.1:8080/
curl -X POST -H "Content-Type: application/json" --data '{"name":"Tom"}' 127.0.0.1:8080/greet
curl -X GET 127.0.0.1:8080/goodbye
```

Figure 4-15 Accessing a real-time service



NOTE

Set **limit** and **request** to proper values to restrict the number of CPUs and memory size. A single Snt9B node is equipped with eight Snt9B PUs and 192 vCPUs 1,536 GB. Properly plan the CPU and memory allocations to avoid task failures due to insufficient CPU and memory limits.

----End

4.4 Using Ascend FaultDiag to Diagnose Logs in the ModelArts Lite Cluster Resource Pool

Description

This section describes how to use Ascend FaultDiag to diagnose logs in the ModelArts Lite environment, including log collection, log cleaning, and fault diagnosis.

The log data is collected **by node** and will be cleaned in the log directory of each node. **The cleaning results will be summarized** for fault diagnosis. For example, for a task running on a cluster with eight nodes and 64 cards, logs need to be collected on the eight nodes respectively. The collected logs are stored in eight directories from **worker-0** to **worker-7**. Then, the logs are cleaned in the eight directories respectively. The cleaning results of each directory are stored in directories from **output/worker-0** to **output/worker-7**. Finally, fault diagnosis is performed in the **output** directory, where you can obtain the diagnosis result.

To download Ascend FaultDiag, see **Ascend Community**.

Step 1: Collecting Logs

You need to collect the following types of logs: user training screen printing logs, host OS logs (host logs), device logs, CANN logs, host resource information, and NPU network port resource information.

- User training logs: log information output to the standard output (screen) during training sessions. Set environment variables to enable logging to the screen.
- Host OS logs: logs generated by user processes on the host during the execution of training jobs.
- Device logs: AI CPU and HCCP logs generated on the device when user processes run on the host. These logs are sent back to the host.
- CANN logs: runtime information from the Compute Architecture for Neural Networks (CANN) module within the Ascend computing architecture. They are useful for diagnosing issues during model conversion, such as errors like "Convert graph to om failed."
- Host resource information: statistics on resources used by AI applications or services running on the host.
- NPU network port resource information: statistics on resources used by AI applications or services running on the host.

If the log data has been output and dumped during training, for example, stored in OBS, and meets the file name and path requirements in **Constraints**, skip this step and go to **step 2**.

- **Constraints**
  - The CANN logs must be stored in the **process_log** folder, for example, **worker-0/.../process_log/**.
  - The device logs must be stored in the **device_log** folder, for example, **worker-0/.../device_log/**.
  - The host resource information and NPU network port resource information must be stored in the **environment_check** folder, for example, **worker-0/.../environment_check/**.
  - Logs collected on a single node are stored in a single **worker** directory. The total file size must be smaller than 5 GB and there cannot be more than one million files. Otherwise, the log cleaning efficiency will be affected.
  - There is no size limit on the user training screen printing logs. By default, only the last 100 KB logs are read.
  - A single CANN log file must be smaller than 20 MB.
  - The NPU status monitoring indicator file, NPU network port monitoring indicator file, and host resource information file must be smaller than 512 MB.
  - The host logs must be **messages** logs in the **/var/log** directory. The maximum size of a single file to be dumped must be less than 512 MB.

Step 2: Cleaning Logs

The collected logs should be organized by node path. For example, the logs collected on the **worker-0** node must be stored in the **worker-0** directory. Check

whether the **device_log**, **process_log**, and **environment_check** subdirectories exist in the directory and whether the names are correct.

1. **Data mounting**

   If the collected logs are stored on OBS, mount the log data in OBS using the **rclone tool**.

   a. Download and install rclone.

   b. Configure the credentials required for accessing OBS.

   ```
   # Hard-coded or plaintext AK/SK is risky. For security, encrypt your AK/SK and store them in the configuration file or environment variables.
   # In this example, the AK/SK is stored in environment variables for identity authentication. Before running this example, set environment variables HUAWEICLOUD_SDK_AK and HUAWEICLOUD_SDK_SK.
   export AWS_ACCESS_KEY=${HUAWEICLOUD_SDK_AK}
   export AWS_SECRET_KEY=${HUAWEICLOUD_SDK_AK}
   export AWS_SESSION_TOKEN=${TOKEN}
   ```

   c. Fill the rclone configuration file **rclone.conf**.

   ```
   [rclone]
   type = s3
   provider = huaweiOBS
   env_auth = true
   acl = private
   ```

   d. Run the **lsd** command to view the directory in the log path and check whether the configuration is successful.

   ```
   rclone lsd rclone:/${obs_bucket_name}/${path_to_logs} --config=${path_to_rclone.config} --s3-endpoint=${obs_endpoint} –no-check-certificate
   ```

   e. The following shows a log directory example. The task only has one node **worker-0**.

   ```
   [root@test-7f5655594b4-mvzb8 modelarts-ascend-brain]# /opt/rclone/rclone lsd rclone:/${obs_bucket_input}/${obs_path_input}/ \
   > --config=/opt/rclone/rclone.conf --s3-endpoint=${s3_endpoint} --no-check-certificate
              0 2000-01-01 08:00:00        -1 worker-0
   ```

   f. Run the **mount** command to mount the log directory to the local host.

   ```
   rclone mount rclone:/${obs_bucket_name}/${path_to_logs} /${path_to_local_dir} --config=${path_to_rclone.config} --s3-endpoint=${obs_endpoint} –no-check-certificate
   ```

2. **Node log cleaning**

   Specify the log path of a single node as the input and the log cleaning storage path as output. The output path must be empty. Run the **ascend-fd parse** command to clean the logs of each node.

   ```
   ascend-fd parse -i ${path_to_worker_logs} -o ${path_to_parse_output}
   ```

   ```
   [root@[redacted]]# ascend-fd parse -i worker-0/ -o log-output/worker-0/
   The parse job starts. Please wait.
   These job ['NODE_ANOMALY', 'KNOWLEDGE_GRAPH', 'ROOT_CLUSTER', 'NET_CONGESTION'] succeeded.
   The parse job is complete.
   ```

   The cleaning result is similar to the log. **Different nodes need to be stored in different worker directories**.

Step 3: Fault Diagnosis

There is a limit on the maximum number of processes (1,024 by default) in the Linux system. Therefore, there should be no more than 128 servers (1,024 cards) in a cluster. If the number of servers exceeds the limit, run the **ulimit -n ${num}** command to adjust the upper limit of the file descriptor. The value of **${num}** should be greater than the number of cards, for example, for a cluster with 6,000 cards, set the value to **8192**.

To perform fault diagnosis, you need to specify the path for storing the cleaning results of all nodes. The output path must be empty. Run the **ascend-fd diag** command to perform fault diagnosis.


```
ascend-fd diag -i ${path_to_parse_outputs} -o ${path_to_diag_output}
```

The diagnosis results are displayed in the following two ways.

- Command output
- The **diag_report.json** file generated in the **${path_to_diag_output}/fault_diag_result** directory

4.5 Mounting an SFS Turbo File System to a Lite Cluster

Scenario

If the inadequate disk space of the local server cannot meet your growing service requirements, you can mount the data disk to a Lite Cluster resource pool to dynamically allocate storage resources, meeting your requirements for flexible scheduling, efficient utilization, and secure access.

Huawei Cloud Scalable File Service Turbo (SFS Turbo) features on-demand, high-performance network-attached storage (NAS), ensuring large-scale data reads and writes. It allows multiple compute nodes to share the same file system. Mounting SFS Turbo to a Lite Cluster enhances data access performance, enables data sharing across multiple nodes, improves resource utilization, and boosts task collaboration efficiency, making it ideal for high-performance computing and distributed training scenarios.

This document uses IPv4 as an example to describe how to mount an SFS Turbo file system to a ModelArts Lite Cluster and how to mount the SFS Turbo file system to the host and Kubernetes container. Ensure that the Lite Cluster and the SFS Turbo file system to be mounted are in the same VPC subnet.

Precautions

You need to properly plan the resource pool and SFS Turbo CIDR block to avoid conflicts between ModelArts Lite Cluster nodes and SFS Turbo file systems. To be more specific:

If the SFS Turbo file system uses a VPC CIDR block starting with 192.168.*x.x*, the resource pool nodes must not use the 172.16.0.0/16 CIDR block.

If the SFS Turbo file system uses a VPC CIDR block starting with 172.*x.x.x*, the resource pool nodes must not use the 192.168.0.0/16 CIDR block.

If the SFS Turbo file system uses a VPC CIDR block starting with 10.*x.x.x*, the resource pool nodes must not use the 172.16.0.0/16 CIDR block.

Billing

- After Lite Cluster resources are enabled, compute resources are charged. For Lite Cluster resource pools, only the yearly/monthly billing mode is supported. For details, see **Table 4-1**.

  Table 4-1 Billing items

Billing Item		Description	Billing Mode	Billing Formula
Compute resource	Dedicated resource pool	Usage of compute resources. For details, see **ModelArts Pricing Details**.	Yearly/Monthly	Specification unit price x Number of compute nodes x Purchase duration

- When purchasing a Lite Cluster resource pool, you need to select a CCE cluster. For details about CCE pricing, see **CCE Price Calculator**.
- The SFS Turbo file system is charged based on the storage capacity and duration you selected during purchase. For details, see **SFS Billing**.

Step 1: Creating a VPC

To mount an SFS Turbo file system to a Lite Cluster, you need to create a VPC subnet and add the Lite Cluster and the SFS Turbo file system to the subnet.

1. Go to the page for **creating a VPC**.
2. On the **Create VPC** page, set the parameters as prompted.

   For details about certain parameters in this case, see **Table 4-2**. For more parameters, see **Creating a VPC with a Subnet**.

Figure 4-16 Creating a VPC and subnet

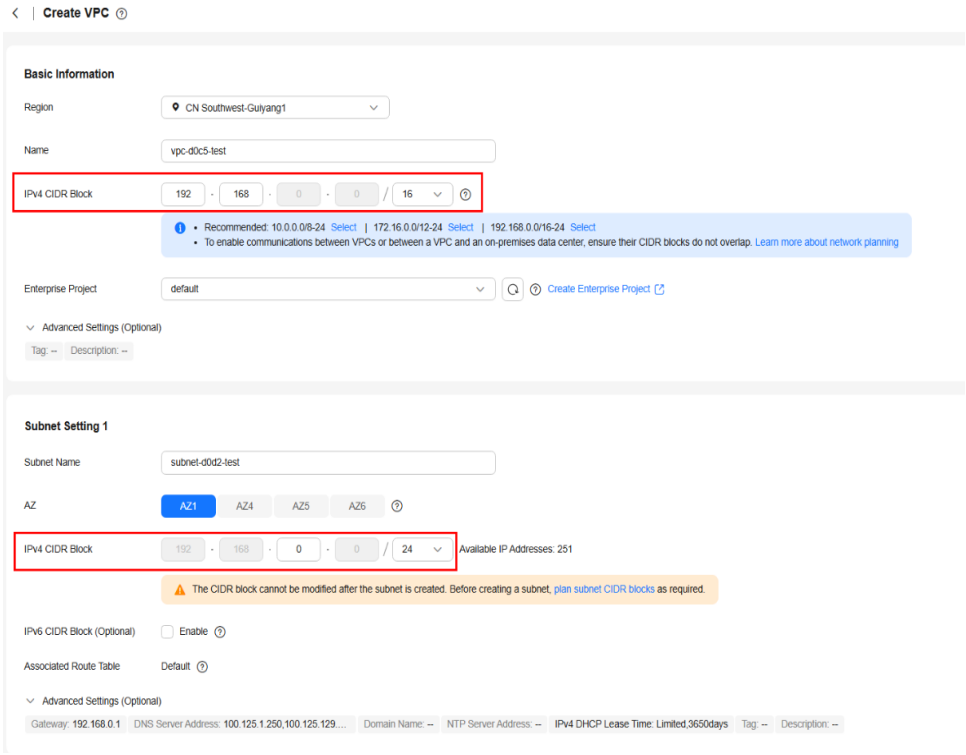


Table 4-2 VPC parameters

Parameter	Description	Example Value
IPv4 CIDR Block	When SFS Turbo is mounted to Lite Cluster, you are recommended to use the private IPv4 addresses specified in **RFC 1918** as the VPC CIDR block. To be more specific: • 10.0.0.0/8–24: The IP address ranges from 10.0.0.0 to 10.255.255.255, and the mask ranges from 8 to 24. • 172.16.0.0/12–24: The IP address ranges from 172.16.0.0 to 172.31.255.255, and the mask ranges from 12 to 24. • 192.168.0.0/16–24: The IP address ranges from 192.168.0.0 to 192.168.255.255, and the mask ranges from 16 to 24.	192.168.0.0/16

Table 4-3 Subnet parameters

Parameter	Description	Example Value
CIDR Block	This parameter is displayed only in regions where IPv4/IPv6 dual-stack is not supported. Set the IPv4 CIDR block of the subnet. For details, see section "IPv4 CIDR Block".	192.168.0.0/24

Parameter	Description	Example Value
IPv4 CIDR Block	This parameter is displayed only in regions where IPv4/IPv6 dual stack is supported. A subnet is a unique CIDR block with a range of IP addresses in a VPC. Comply with the following principles when planning subnets: • Planning CIDR block size: After a subnet is created, the CIDR block cannot be changed. You need to plan the CIDR block in advance based on the number of IP addresses required by your service. – The subnet CIDR block cannot be too small. Ensure that the number of available IP addresses in the subnet meets service requirements. The first and last three addresses in a subnet CIDR block are reserved for system use. For example, in subnet 10.0.0.0/24, 10.0.0.1 is the gateway address, 10.0.0.253 is the system interface address, 10.0.0.254 is used by DHCP, and 10.0.0.255 is the	192.168.0.0/24

Parameter	Description	Example Value
	broadcast address. – The subnet CIDR block cannot be too large, either. If you use a CIDR block that is too large, you may not have enough CIDR blocks available for new subnets, which can be a problem when you want to create subnets. • Avoiding subnet CIDR block conflicts: If you need to connect two VPCs or connect a VPC to an on-premises data center, there cannot be any CIDR block conflicts. If the subnet CIDR blocks at both ends of the network conflict, create a subnet. For details, see **Creating a Subnet for an Existing VPC**. A subnet mask can be between the netmask of its VPC CIDR block and /29 netmask. If a VPC CIDR block is 10.0.0.0/16, its subnet mask can be between 16 and 29. For details about VPC subnet planning, see **VPC and Subnet Planning**.	

3. Click **Create Now**.
   Return to the VPC list and view the created VPC.

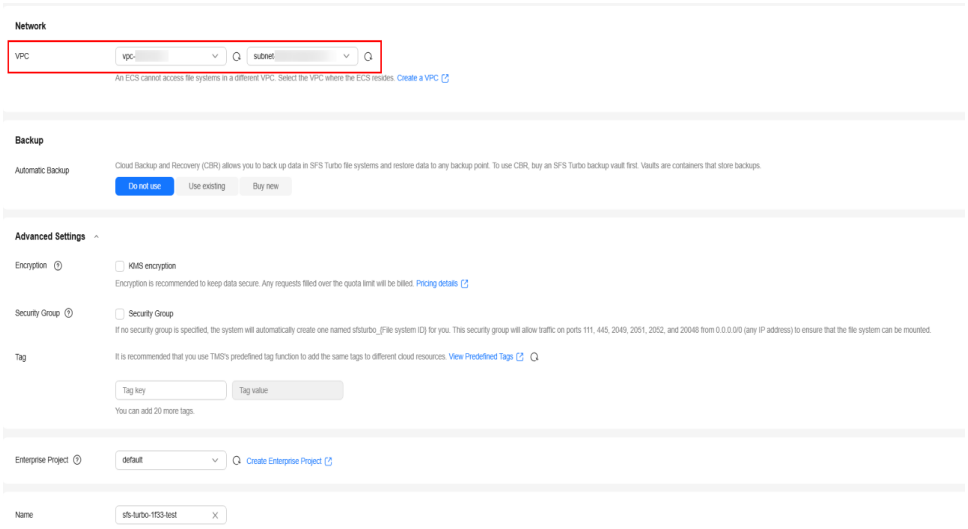
Step 2: Creating an SFS Turbo File System

Create an SFS Turbo file system to be mounted and use the VPC created in **Step 1: Creating a VPC**.

1. Log in to the **SFS console**. In the navigation pane on the left, choose **SFS Turbo**. Click **Create File System** in the upper right corner.
2. Configure the parameters, as shown in **Figure 4-17**.

   For **VPC**, select the VPC and subnet created in **Step 1: Creating a VPC**. For details about the parameters, see **Creating an SFS Turbo File System**.

   Figure 4-17 Creating an SFS Turbo file system



3. Click **Create Now**.
4. Confirm the file system information and click **Submit**.
5. Return to the file system list page to view the file system status.

Step 3: Creating a CCE Cluster

Lite Cluster resource pools depend on CCE clusters to provide a container-based environment. CCE clusters provide Lite resource pools with required computing, storage, and network resources. Therefore, you need to select CCE clusters when purchasing Lite Cluster resource pools.

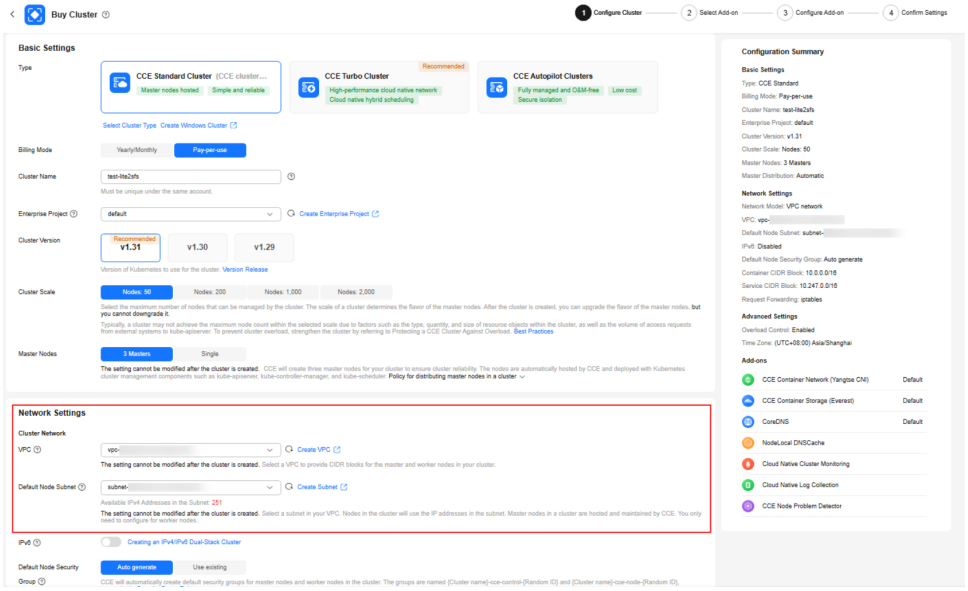
If no CCE cluster is available, purchase one by referring to **Buying a CCE Standard/Turbo Cluster**. For details about the cluster version, see **Software Versions Required by Different Models**.

1. Log in to the **CCE console**.
   - If no cluster has been created, click **Buy Cluster** on the wizard page.
   - If you already have a CCE cluster, choose **Clusters** in the navigation pane and click **Create Cluster** in the upper right corner.
2. Configure the cluster basic information.
   - VPC: Select the VPC created in **Step 1: Creating a VPC**.

- Default node subnet: Select the subnet created in **Step 1: Creating a VPC**.
- Enable IPv6: This document uses IPv4 as an example. Therefore, IPv6 is not enabled.

For details about the parameters, see **Buying a CCE Standard/Turbo Cluster**.

Figure 4-18 Buying a CCE cluster



3. Click **Next: Confirm Settings**, check the displayed cluster resource list, and click **Submit**.

   It takes about 5 to 10 minutes to create a cluster. You can click **Back to Cluster List** to perform other operations on the cluster or click **Go to Cluster Events** to view the cluster details.

Step 4: Creating a Lite Cluster

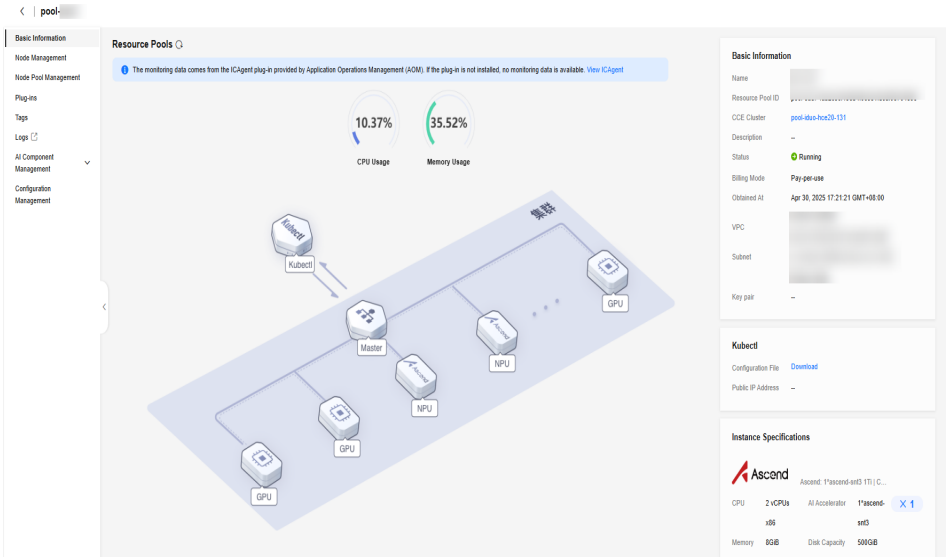
Create a Lite Cluster using the CCE cluster created in **Step 3: Creating a CCE Cluster**.

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.
2. Click **Buy Lite Cluster**. On the displayed page, configure parameters.

   Select the CCE cluster created in **Step 3: Creating a CCE Cluster**. For details about the parameters, see **Table 2-3**.
3. Click **Buy Now** and confirm the specifications. Confirm the information and click **Submit**.

   After a resource pool is created, its status changes to **Running**. Click the cluster resource name to access its details page. Check whether the purchased specifications are correct.

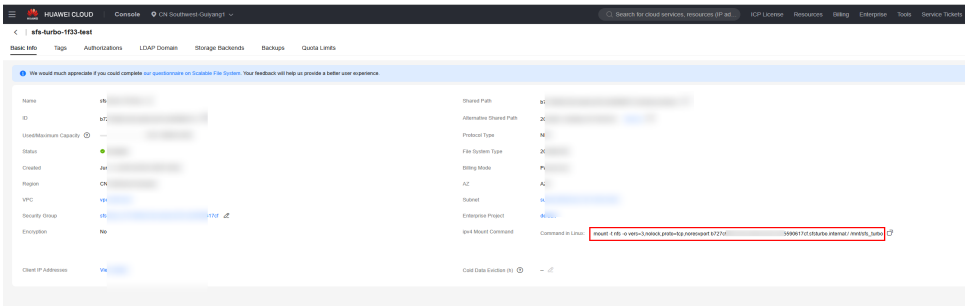
Figure 4-19 Viewing resource details



Step 5: Mounting an SFS Turbo File System to a Lite Cluster Node

1. Log in to the SFS console. In the navigation pane on the left, choose **SFS Turbo**. Click the SFS Turbo file system created in **Step 2: Creating an SFS Turbo File System** to access its details page, and copy the Linux mounting command.

   Figure 4-20 Copying the Linux mounting command

2. Log in to the Lite Cluster node using a bash tool such as Xshell or MobaXterm, and run the following command to create the folder to be mounted:

   ```
   mkdir /mnt/sfs_turbo
   ```

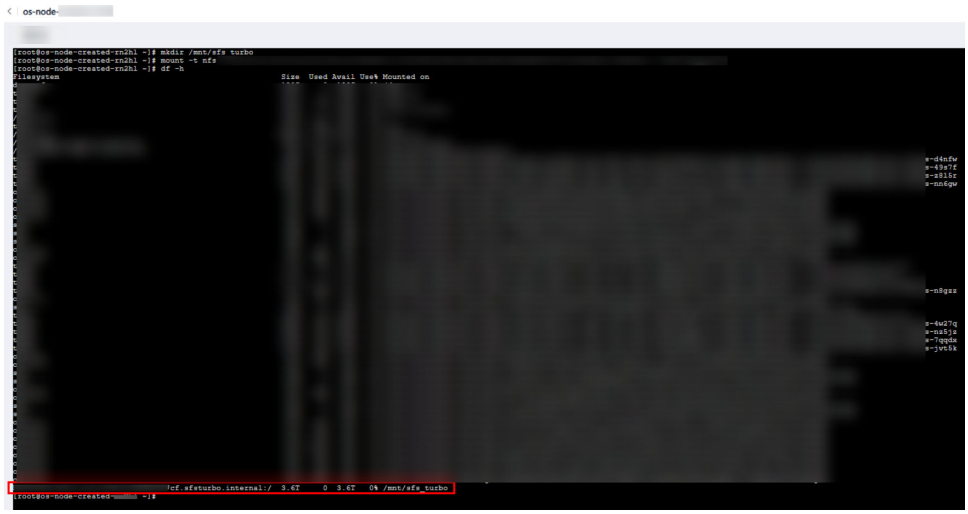
3. Run the Linux mounting command copied in the previous step.

   ```
   mount -t nfs -o vers=3,nolock,proto=tcp,noresvport xxxx.sfsturbo.internal:/ /mnt/sfs_turbo
   ```

   Run the **df -h** command to view the SFS Turbo file system mounting information.

   As shown in the following figure, the SFS Turbo file system is mounted to the **/mnt/sfs_turbo** directory on the node.

Figure 4-21 Viewing mounting information



Step 6: Mounting the SFS Turbo File System to the Workload in the Lite Cluster Kubernetes Cluster

You can mount the SFS Turbo file system to the workload using Kubernetes NFS.

1. Log in to the SFS console. In the navigation pane on the left, choose **SFS Turbo**. Click the SFS Turbo file system created in **Step 2: Creating an SFS Turbo File System** to access its details page, and copy the shared path.

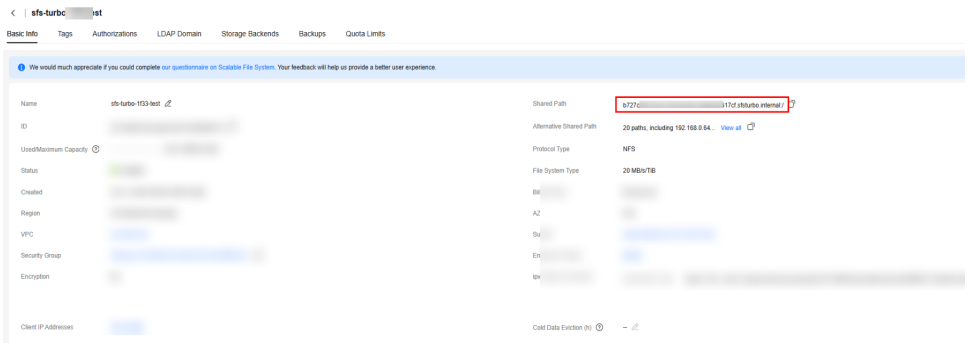
   ```
   xxxxx.sfsturbo.internal:/
   ```

   The path divides the two parameters required for Kubernetes NFS mounting with a colon (:).

   xxxxx.sfsturbo.internal is the NFS mounting parameter **nfs.server** of the workload.

   / is the NFS mounting parameter **nfs.path** of the workload.

   Figure 4-22 Copying a shared path

2. Log in to the Lite Cluster node using a bash tool, such as Xshell or MobaXterm, and create the **dep.yaml** file. The file content is as follows:

   ```
   kind: Deployment
   apiVersion: apps/v1
   metadata:
     name: testlite2sfsturbo
   ```

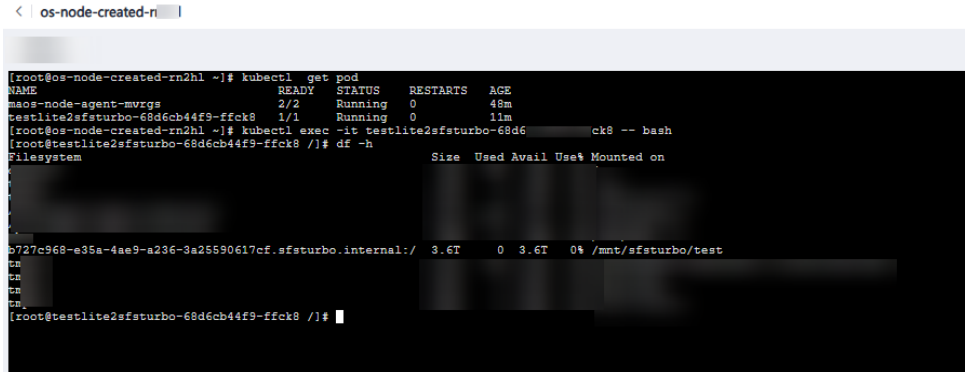


```
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: testlite2sfsturbo
      version: v1
  template:
    metadata:
      labels:
        app: testlite2sfsturbo
        version: v1
    spec:
      volumes:
        - name: nfs0
          nfs:
            server: xxxxx.sfsturbo.internal ## Enter the server information in the shared path of the SFS
Turbo file system.
            path: /
      containers:
        - name: pod0
          image: swr.cn-southwest-2.myhuaweicloud.com/hwofficial/everest:2.4.134 ## Image path
          command:
            - /bin/bash
            - '-c'
            - while true; do echo hello; sleep 10; done
          env:
            - name: PAAS_APP_NAME
              value: testlite2sfsturbo
            - name: PAAS_NAMESPACE
              value: default
            - name: PAAS_PROJECT_ID
              value: xxxx
          resources:
            limits:
              cpu: 250m
              memory: 2000Mi
            requests:
              cpu: 250m
              memory: 2000Mi
          volumeMounts:
            - name: nfs0
              mountPath: /mnt/sfsturbo/test
          imagePullPolicy: IfNotPresent
```

3. Run the **kubectl apply -f dep.yaml** command to create a workload.
4. Run the **kubectl get pod** command to check whether the pod container group is started.
5. Run the **kubectl exec -it {pod_name} -- bash** command to log in to the container.
6. Run the **df -h** command.

   As shown in the following figure, the SFS Turbo file system is mounted to the **/mnt/sfs_turbo/test** directory in the Kubernetes container.

Figure 4-23 Viewing mounting information



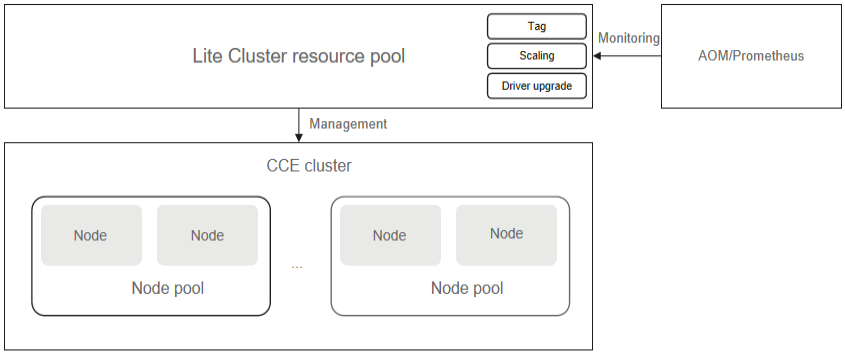
5 Managing Lite Cluster Resources

5.1 Managing Lite Cluster Resources

On the ModelArts console, you can manage created resources. You can click a resource pool name to access its details page and perform the following operations:

- **Managing Lite Cluster Resource Pools**: ModelArts allows you to manage resource pools, including renewing subscriptions, enabling or modifying auto-renewal, scaling resources, and upgrading drivers.
- **Managing Lite Cluster Node Pools**: To help you better manage nodes in a Kubernetes cluster, ModelArts provides node pools. A node pool is a group of nodes with the same configuration in a cluster. You can create, update, or delete node pools.
- **Managing Lite Cluster Nodes**: A node is a fundamental component of a container cluster. You can replace, delete, or reset a node within a resource pool. You can also delete, unsubscribe from, or renew nodes in batches.
- **Resizing a Lite Cluster Resource Pool**: The demand for resources in a Cluster resource pool may change due to the changes of AI development services. In this case, you can resize your resource pool in ModelArts.
- **Upgrading the Lite Cluster Resource Pool Driver**: If GP or Ascend resources are used in a resource pool, you may need to customize GP or Ascend drivers. ModelArts allows you to upgrade GP and Ascend drivers of your dedicated resource pools.
- **Monitoring Lite Cluster Resources**: ModelArts leverages AOM and Prometheus to monitor resources, providing insights into resource usage.
- **Releasing Lite Cluster Resources**: You can release Lite Cluster resources that are no longer used.

Figure 5-1 Lite Cluster resource management



5.2 Managing Lite Cluster Resource Pools

Renewal Management of Lite Cluster Resource Pools

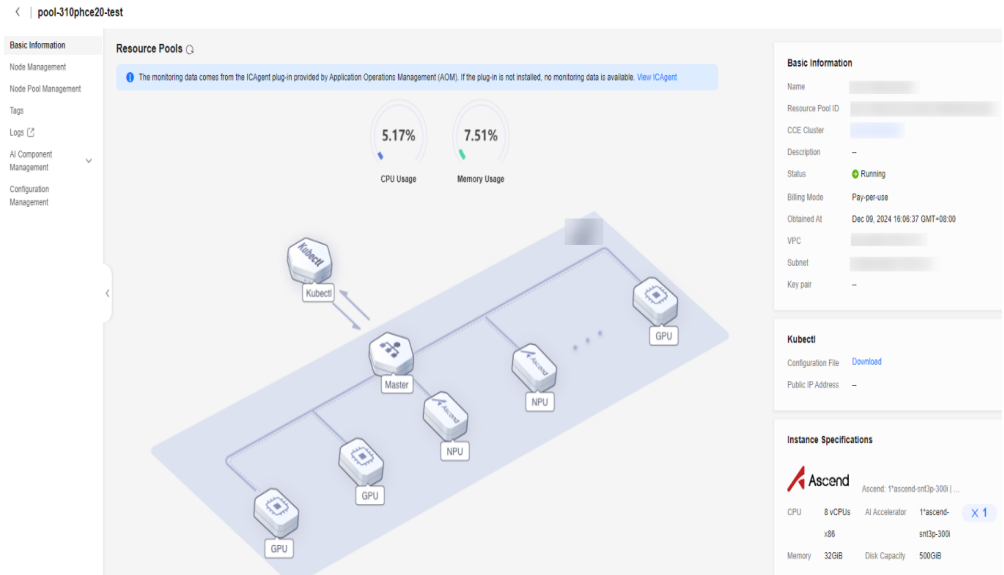
For yearly/monthly Lite Cluster resource pools, you can renew them, enable auto-renewal, and modify auto-renewal. The fees generated by auto-renewal will be deducted from your account balance. For details, see **Auto-Renewal**.

Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.

Viewing Basic Information About a Lite Cluster Resource Pool

Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**. On the displayed page, click a resource pool name to view its details.

Figure 5-2 Viewing basic information about a Lite Cluster resource pool



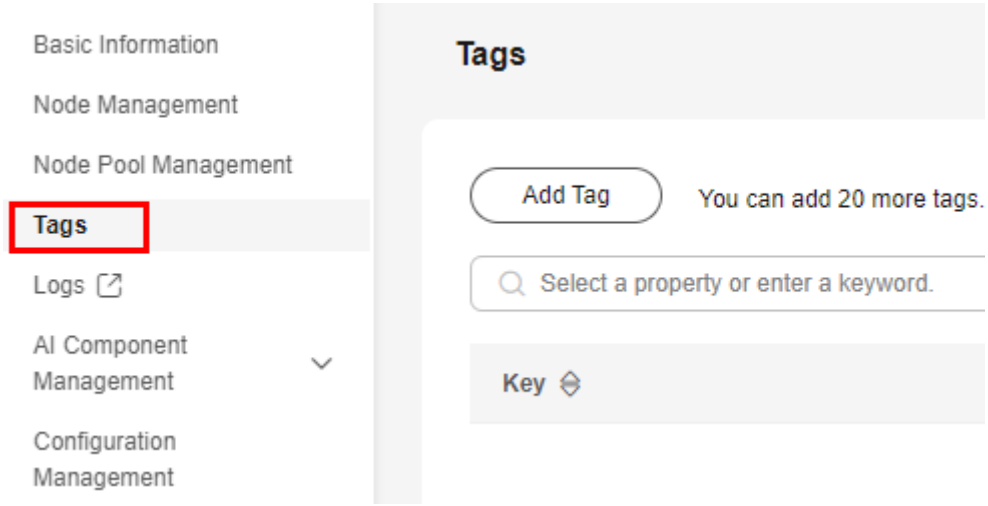
Managing Lite Cluster Resource Pool Tags

You can add tags to a resource pool for quick search.

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.
2. In the Lite resource pool list, click the name of the target resource pool to view its details.
3. On the resource pool details page, click the **Tags** tab to view the tag information.

   Tags can be added, modified, and deleted. For details about how to use tags, see **Using TMS Tags to Manage Resources by Group**.

Figure 5-3 Tags



NOTE

You can add up to 20 tags.

Configuration Management of Lite Cluster Resource Pools

On the resource pool details page, click **Configuration Management**. From there, you can modify the namespace to be monitored, cluster configuration, and image pre-provisioning information.

- Click next to monitoring to enable or disable monitoring and set the namespace to be monitored. For details about how to use monitoring, see **Viewing Lite Cluster Metrics Using Prometheus**.
- Click next to cluster configuration to set core binding, dropcache, and hugepage memory parameters. If no value is set, the default value from the resource pool image will be used.
  - **Core Pinning**: If CPU pinning is enabled, workload pods exclusively use CPUs to improve performance (such as training and inference performance) and reduce the scheduling delay. This function is ideal for scenarios that are sensitive to CPU caching and scheduling delay. If CPU binding is disabled, exclusive CPUs will not be allocated to workload pods. Disable this function if you want a large pool of shareable CPUs.

You can also disable core binding and use **taskset** to flexibly bind cores in service containers.

- **Dropcache**: After this function is enabled, Linux cache clearing is enabled. This function can improve application performance in most scenarios. Clearing the cache can potentially lead to container startup failure or a degradation in system performance, as the system will need to reload data from the disk into memory. If this function is disabled, cache clearing is disabled.
- **Hugepage Memory**: When enabled, Transparent Huge Page (THP) is used. This memory management technique boosts system performance by increasing the memory page size. THP dynamically allocates huge page memory, simplifying its management. Enabling huge page memory can enhance application performance in most cases. However, it may trigger node restarts due to the soft lockup mechanism. If disabled, huge page memory is not used.

- Click for image pre-provisioning to set the image source, add an image key, and configure image pre-provisioning. For details, see **(Optional) Configuring Image Pre-provisioning**.

More Operations

For more operations, see the following:

- Managing node pools: **Managing Lite Cluster Node Pools**
- Managing nodes: **Managing Lite Cluster Nodes**
- Resizing Lite Cluster resource pools: **Resizing a Lite Cluster Resource Pool**
- Upgrading the driver of a Lite Cluster resource pool: **Upgrading the Lite Cluster Resource Pool Driver**
- Upgrading the driver of a Lite Cluster resource pool node: **Upgrading the Driver of a Lite Cluster Resource Pool Node**

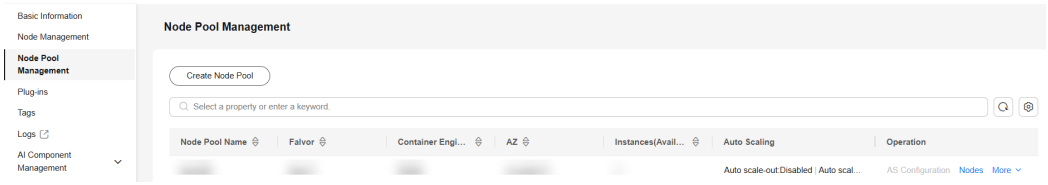
5.3 Managing Lite Cluster Node Pools

To help you better manage nodes in a Kubernetes cluster, ModelArts provides node pools. A node pool consists of one or more nodes, allowing you to set up a group of nodes with specific configurations.

Accessing the Node Pool Management Page

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.
2. On the displayed page, click the Lite Cluster name to access its details page.
3. In the navigation pane on the left, choose **Node Pool Management**. You can create, update, and delete node pools.

Figure 5-4 Node pool management



Creating a Node Pool

1. If you need more node pools, click **Create Node Pool** to create one. Configure the parameters by referring to **Table 5-1**.

   In CN East 2, each Lite Cluster can contain a maximum of 15 node pools. In CN Southwest Guiyang1, each Lite Cluster can contain a maximum of 50 node pools. In other regions, each Lite Cluster can contain a maximum of 10 node pools.

Table 5-1 Node pool parameters

Parameter	Description
Node Pool Name	Enter a custom node pool name. Only lowercase letters, digits, and hyphens (-) are allowed. The value must start with a lowercase letter and cannot end with a hyphen (-) or **-default**.
Instance Specifications	Choose **CPU**, **GPU**, or **Ascend** as needed. • **CPU**: general-purpose compute architecture, features low computing performance, is suitable for lightweight general tasks. • **GPU**: parallel compute architecture, features high computing performance, is suitable for parallel tasks and scenarios such as deep learning training and image processing, and supports multi-PU distributed training. • **Ascend**: dedicated AI architecture, features extremely high computing performance, is suitable for AI tasks and scenarios such as AI model training and inference acceleration, and supports multi-node distributed deployment.
Operating System	Specify the OS of the instance.

Parameter	Description
AZ Allocation	Select **Automatic** or **Manual** as required. An AZ is a physical region where resources use independent power supplies and networks. AZs are physically isolated but interconnected over an intranet. • **Automatic**: AZs are automatically allocated. • **Manual**: Specify AZs for resource pool instances. To ensure system disaster recovery (DR), deploy all instances in the same AZ. You can set the number of instances in an AZ.
Target Instances	Set the number of nodes in the node pool. More nodes indicate higher computing performance. If **AZ** is set to **Manual**, you do not need to configure **Instances**. Do not create more than 30 instances at a time. Otherwise, the creation may fail due to traffic limiting. The total number of instances cannot exceed the cluster scale of the node pool. If the cluster scale of the node pool is set to the default value, the total number of instances cannot exceed 50. For details, see the console. You can purchase instances by rack for certain specifications. The total number of instances is the number of racks multiplied by the number of instances per rack. Purchasing a full rack allows you to isolate tasks physically, preventing communication conflicts and maintaining linear computing performance as task scale increases. All instances in a rack must be created or deleted together.
Virtual Private Cloud	The VPC to which the cluster belongs by default, which cannot be changed.
Kubernetes Label	Add key/value pairs that are attached to Kubernetes objects, such as Pods. A maximum of 20 labels can be added. Labels can be used to distinguish nodes. With workload affinity settings, container pods can be scheduled to a specified node.
Taint	This parameter is left blank by default. Configure anti-affinity by adding taints to nodes, with a maximum of 20 taints per node.

Parameter	Description
Container Engine	Container engine, one of the most important components of Kubernetes, manages the lifecycle of images and containers. The Kubelet interacts with a container runtime through the Container Runtime Interface (CRI). **Docker** and **Containerd** are supported. For details about the differences between Containerd and Docker, see **Container Engines**. The CCE cluster version determines the available container engines. If it is earlier than 1.23, only Docker is supported. If it is 1.27 or later, only Containerd is supported. For all other versions, both Containerd and Docker are supported.
Node subnet	Choose a subnet within the same VPC. This subnet will be used to create node pools.
Associate Security Group	Security group used by the nodes created in the node pool. A maximum of four security groups can be selected. Traffic needs to pass through certain ports in the node security group to ensure node communications. If no security group is associated, the cluster's default rules are applied.
Resource Tag	You can add resource tags to classify resources.
Post-installation Command	Enter the script command, which cannot include Chinese characters. The Base64-encoded script must be transferred. The encoded script should not exceed 2,048 characters. The script will be executed after Kubernetes software is installed, which does not affect the installation. Do not run the **reboot** command in the post-installation script to restart the system immediately. To restart the system, run the **shutdown -r 1** command to restart with a delay of one minute.

Parameter	Description
Node Billing Mode	When you add nodes, you can enable this function to specify the billing mode or validity period for them. If this parameter is not specified, the billing information is the same as that of the resource pool by default. For example, you can create pay-per-use nodes in a yearly/monthly resource pool. If the billing mode is not specified, the new nodes share the same billing mode with the resource pool. If yearly/monthly nodes are to be created, select whether to enable auto-renewal. If auto-renewal is enabled, the nodes to be created will be automatically renewed upon expiration. For a yearly/monthly node pool, if the billing mode of the nodes to be created is also yearly/monthly, the billing period of the new nodes cannot be later than that of the original node pool. For example, if the original yearly/monthly node pool is about to expire in six months, the nodes to be added cannot be billed later than six months.

2. Confirm the configurations. Hover the cursor over the fees to confirm the details. Then, click **Confirm**.
3. In the displayed dialog box, confirm whether to enable auto-renewal for the new nodes, and click **OK**.

   You can view the created node pool on the node pool management page.

Configuring Auto Scaling for a Node Pool

Nodes in a node pool can be automatically added or removed based on the pod scheduling status and resource usage. Multiple scaling modes, such as multi-AZ, multi-instance specifications, metric triggering, and periodic triggering, are supported to meet different node scaling scenarios.

To use auto scaling for a node pool, you need to install the cluster elastic engine plug-in first. For details, see **Cluster Autoscaler**.

Viewing the Node List

To view information about nodes in a node pool, click **Nodes** in the **Operation** column to view the node name, specifications, and AZ.

Updating a Node Pool

1. Locate the target node pool and choose **More** > **Modify Configuration** in the **Operation** column. For details about the parameters, see **Table 5-1**.

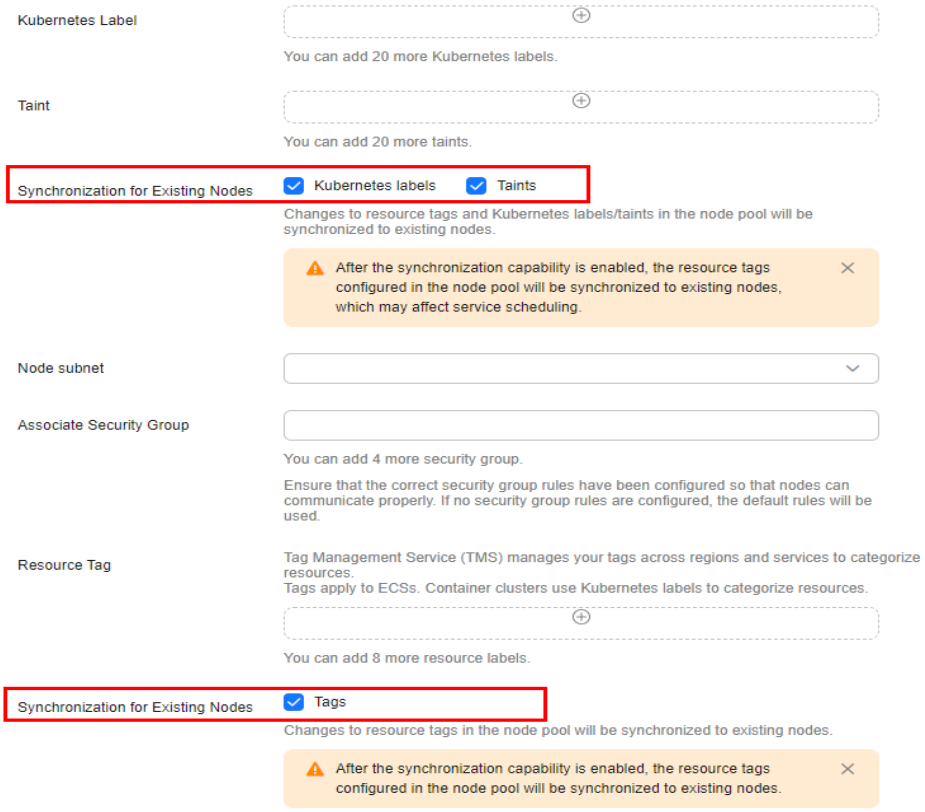
   Note the following:

   - The total number of instances cannot exceed the cluster scale of the node pool. If the cluster scale of the node pool is set to the default value, the total number of instances cannot exceed 50. For details, see the console.

- When you update the node pool configuration, the advanced configuration takes effect only for new nodes. **Synchronization for Existing Nodes** (labels and taints) and **Synchronization for Existing Nodes** (labels) can be modified synchronously for existing nodes (by selecting the check boxes).

  The updated resource tag information in the node pool is synchronized to its nodes.

Figure 5-5 Updating a node pool



2. Confirm the configurations. Hover the cursor over the fees to confirm the details. Then, click **Confirm**.
3. In the displayed dialog box, confirm whether to enable auto-renewal for the new nodes, and click **OK**.

   You can view the updated node pool on the node pool management page.

Upgrading a Lite Cluster Resource Pool Driver

If there are GPU/Ascend resources in a Lite Cluster resource pool node, and the node performance cannot meet your requirements, you can upgrade the driver to resolve known issues, improve performance, or support new functions, ensuring resource pool performance and compatibility.

To upgrade the GPU or Ascend driver of the Lite Cluster resource pool, choose **More** > **Upgrade Driver** in the **Operation** column. For details, see **Upgrading the Lite Cluster Resource Pool Driver**.

Deleting a Node Pool

If there are multiple node pools, you can delete one. To do so, click **Delete** in the **Operation** column. Confirm the associated resources and jobs that will be affected, click **Delete**, enter **DELETE**, and click **OK**.

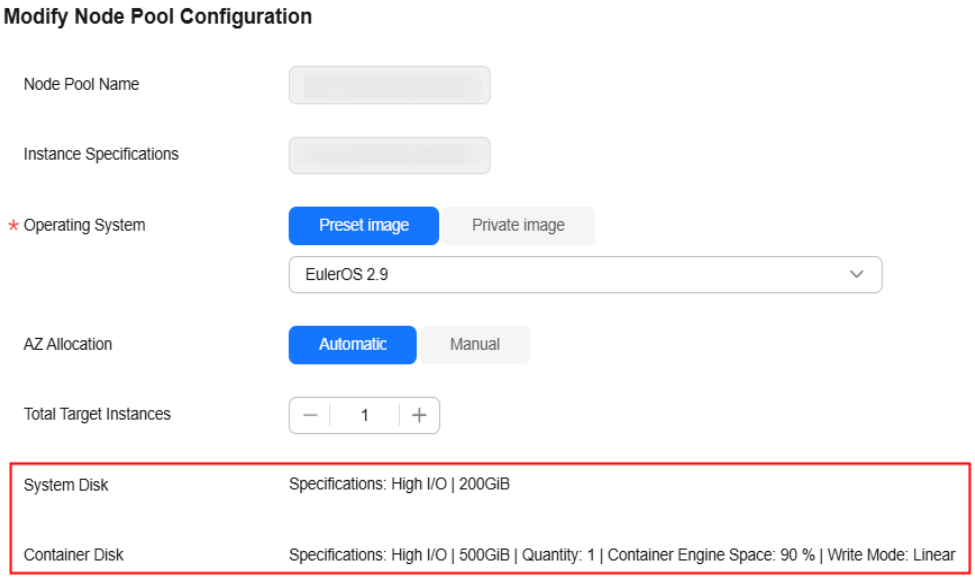
For yearly/monthly nodes that are not unsubscribed from or released, click **Go Now** to access the resource pool details page. For details, see **Deleting, Unsubscribing from, or Releasing a Node**.

Each resource pool must have at least one node pool. If there is only one node pool in a resource pool, it cannot be deleted.

Viewing the Storage Configuration of a Node Pool

On the **Modify Node Pool Configuration** page, you can view details like disk type, size, quantity, write mode, and container engine space size for system, container, or data disks.

Figure 5-6 Modifying node pool configurations



Additionally, on the Lite resource pool scaling page, you can view the storage configuration of its node pools.

Searching for a Node Pool

In the search box on the node pool management page, you can search for node pools by keyword, such as the node pool name, specifications, container engine space size, or AZ.

Specifying Node Pool Information to Be Displayed

On the node pool management page, click  in the upper right corner to customize the information to be displayed in the node pool list.

5.4 Managing Lite Cluster Nodes

Nodes are fundamental components of a container cluster. On the resource pool details page, click the **Nodes** tab to replace, delete, reset, or renew nodes. When you hover over a node name, the resource ID is displayed. You can use the resource ID to query bills or billing information of yearly/monthly resources in the Billing Center.

Deleting, Unsubscribing from, or Releasing a Node

- To release a single node from a pay-per-use resource pool, find the target node and click **Delete** in the **Operation** column, enter **DELETE** in the displayed dialog box, and click **OK**.

  To delete nodes in batches, select the target nodes and click **Delete** above the node list, enter **DELETE** in the displayed dialog box, and click **OK**.
- For a yearly/monthly resource pool whose resources are not expired, click **Unsubscribe** in the **Operation** column. You can unsubscribe from nodes in batches.
- For a yearly/monthly resource pool whose resources are expired (in the grace period), click **Release** in the **Operation** column. Nodes in the grace period cannot be released in batches.

  If the delete button is available for a yearly/monthly node, the node is an inventory node, click **Delete**.

NOTE

- Before deleting, unsubscribing from, or releasing a node, ensure that there are no running jobs on this node. Otherwise, the jobs will be interrupted.
- Delete, unsubscribe from, or release abnormal nodes in a resource pool and add new ones for substitution.
- If there is only one node, it cannot be deleted, unsubscribed from, or released.

Enabling/Disabling the Deletion Lock

To prevent nodes from being deleted or unsubscribed by mistake, you can enable the deletion lock. Once enabled, the nodes cannot be deleted or unsubscribed unless the lock is disabled.

NOTE

- The deletion lock can be enabled only for the nodes in the resource pool.
- If the deletion lock is enabled, only node deletion and unsubscription are restricted. Other operations, such as node replacement, node restart, and node reset, work properly. Moreover, the resource pool that contains the nodes with deletion lock enabled can be deleted.

- Enabling deletion lock: Locate the target node and choose **More** > **Enable Deletion Lock** in the **Operation** column. In the displayed dialog box, confirm the information, enter **YES** in the text box, and click **OK**.

  To enable deletion lock for multiple nodes in batches, select the target nodes and choose **More** > **Enable Deletion Lock** above the node list.

- Disabling deletion lock: Locate the target node and choose **More** > **Disable Deletion Lock** in the **Operation** column. In the displayed dialog box, confirm the information, enter **YES** in the text box, and click **OK**.

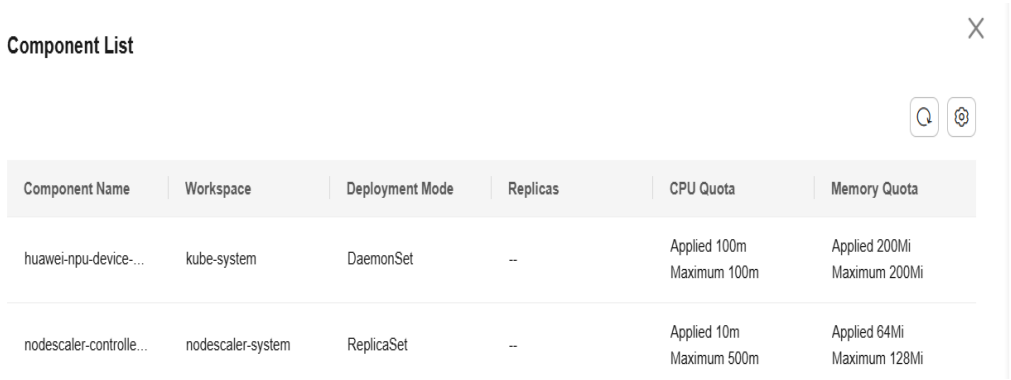
  To disable deletion lock for multiple nodes in batches, select the target nodes and choose **More** > **Disable Deletion Lock** above the node list.

Querying Plug-in Component

On the resource pool details page, choose **Node Management** from the navigation pane to view the plug-in usage of the current node.

To view the instance usage of the plug-in, locate the target node and choose **More** > **Query Plug-in Component** in the **Operation** column.

Figure 5-7 Instances



Renewing a Subscription, Enabling Auto-Renewal, or Modifying Auto-Renewal

For yearly/monthly nodes, you can renew them, enable auto-renewal, and modify auto-renewal in the **Nodes** tab. You can also perform batch operations on nodes.

The fees generated by auto-renewal will be deducted from your account balance. For details, see **Auto-Renewal**.

Resetting a node

In the **Nodes** tab, locate the node you want to reset. Click **Reset** in the **Operation** column to reset a node. You can also select the check boxes of multiple nodes and choose **More** > **Reset** above the node list to reset multiple nodes.

Configure the parameters.

Table 5-2 Parameters

Name	Description
Operating System	Choose a supported OS from the drop-down list.

Name	Description
Configuration Mode	Select a configuration mode for resetting the node. • **By node percentage**: the maximum percentage of nodes that can be reset at a time • **By instance quantity**: the maximum number of nodes that can be reset at a time
Driver Version	Specify the driver version of the nodes to reset from the drop-down list.

Check the node reset records on the **Records** page. If a node is being reset, its status is **Resetting**. After the reset is complete, the node status changes to **Available**. Resetting a node will not be charged.

NOTE

- Resetting a node will impact the operation of related services. During the reset process, the local disk and the Kubernetes tag on the node will be cleared. Proceed with caution when performing this operation.
- Only nodes in the **Available** state can be reset.
- A single node can be in only one reset task at a time. Multiple reset tasks cannot be delivered to the same node at a time.
- If there are any nodes in the **Replacing** state in the operation records, nodes in the resource pool cannot be reset.
- When the driver of a resource pool is being upgraded, nodes in this resource pool cannot be reset.
- For GPU and NPU specifications, after the node is reset, the driver of the node may be upgraded. Wait patiently.

Authorizing O&M on the Event Center Page

To view the faulty nodes reported by the ModelArts O&M platform, log in to the ModelArts console. In the navigation pane on the left, choose **Event Center**. The planned events of the faulty nodes are displayed, including the basic information, event type, event status, and event description. You can either redeploy the nodes or authorize Huawei technical support to perform O&M operations.

- **Authorization conditions**

  Table 5-3 lists the event types and event status of the authorization operations that can be performed on the faulty node.

  Table 5-3 Authorization conditions

Event Type	Event Status	Authorization Operations
System maintenance	To be authorized	Authorization and redeployment

Event Type	Event Status	Authorization Operations
Local disk recovery	To be authorized	Authorization and redeployment After the local disk is recovered, you can restore the partition by **resetting the node**. **WARNING** After authorization, recovering the local disk will cause local disk data loss. Therefore, migrate services and back up data before authorization.
Restarting a node	To be authorized	Authorization
O&M authorization	To be authorized	Authorization
Supernode maintenance	To be authorized	Authorization
Supernode redeployment	To be authorized	Redeployment Redeployment of supernodes must be performed within physical supernodes. When supernodes are sold out, redeployment is not supported, and the authorization button becomes unavailable.
Supernode local disk recovery	To be authorized	Authorization **WARNING** After authorization, recovering the local disk will cause local disk data loss. Therefore, migrate services and back up data before authorization.

- **Authorization**

  If the faulty nodes meet the requirements listed in **Table 5-3**, you can authorize Huawei technical support to perform O&M on the faulty nodes.

  To do so, log in to the ModelArts console. In the navigation pane on the left, choose **Event Center**. Locate the target node and click **Authorize** in the **Operation** column. In the displayed dialog box, click **OK**.

  If the planned event does not meet the requirements listed in **Table 5-3**, the **Authorize** button becomes unavailable.

  After the O&M, Huawei technical support will disable the authorization. No further operations are required.

- **Redeployment**

  If the faulty nodes meet the redeployment requirements listed in **Table 5-3**, you can authorize Huawei technical support to redeploy the faulty nodes.

  After the O&M, Huawei technical support will disable the authorization. No further operations are required.

> **WARNING**
>
> Redeploying nodes can restore them quickly, but local disk data will be lost. Therefore, migrate services and back up data before redeployment.

a. To redeploy a node, log in to the ModelArts console. In the navigation pane on the left, choose **Event Center** under **Resource Management**, locate the node, and click **Redeploy** in the **Operation** column.

   If the planned event does not meet the requirements listed in **Table 5-3**, the **Redeploy** button becomes unavailable.

b. Check whether **Forcible redeployment** is selected, enter **YES** in the text box, and click **OK**.

   Redeployment depends on the node status. If the node is unavailable, redeployment cannot be completed. However, you can select **Forcible redeployment** to forcibly redeploy the node.

   > **WARNING**
   >
   > Forcible redeployment resets the node, deleting all data on both its local and cloud disks. Exercise caution when performing this operation.

Restarting a Node

Locate the target node and choose **More** > **Reboot** in the **Operation** column. You can also select node names and click **Reboot** above the node list to restart nodes in batches. Restarting a node will affect running services.

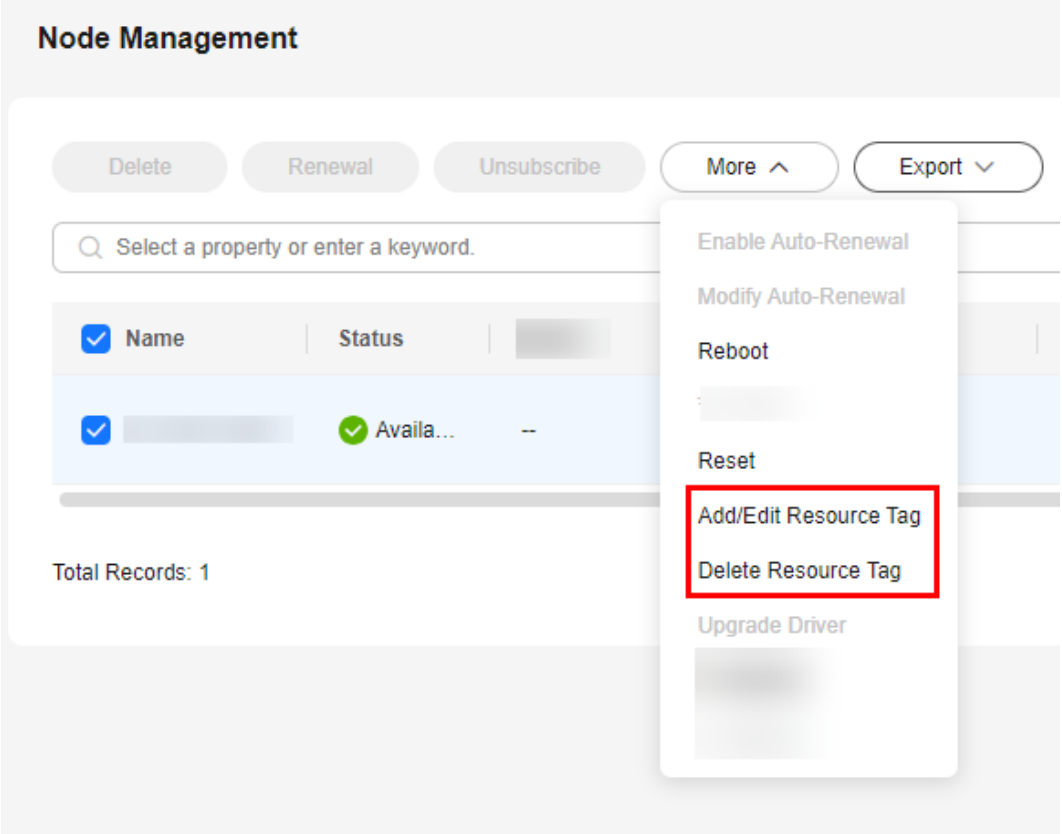
Adding, Editing, or Deleting Resource Tags

Use resource tags for easy billing management.

To edit the resource tags of a single node, locate the target node and choose **More** > **Edit Resource Tag** in the **Operation** column.

You can also select node names and choose **More** > **Add/Edit Resource Tag** or **Delete Resource Tag** above the node list to manage tags in batches.

Figure 5-8 Adding, editing, or deleting resource tags



Exporting Node Data

You can export the node information of a Lite resource pool as an Excel file.

Select the target nodes, choose **Export** > **Export All Data to XLSX** or **Export** > **Export Part Data to XLSX** above the node list, and click in the browser to view the exported Excel file.

Upgrading a Driver

You can upgrade the driver version of a single node in a Lite resource pool or upgrade the driver versions of multiple nodes in batches. For details, see **Upgrading the Driver of a Lite Cluster Resource Pool Node**.

Searching for a Node

In the search box on the node management page, you can search for nodes by node name, status, batch, driver version, driver status, IP address, node pool, or resource tag.

Specifying Node Information to Display

On the node management page, click in the upper right corner to customize the information to display in the node list.

5.5 Resizing a Lite Cluster Resource Pool

Scenario

The demand for resources in a Lite Cluster resource pool may change due to the changes of services. In this case, you can resize your resource pool as needed.

You can add instances to or delete instances from a resource pool for resizing. The instances to be added or deleted must share the same specifications with that of existing instances. This helps you adjust the resource pool scale. For better resource usage, you can delete abnormal or idle nodes from a resource pool. For details, see **Deleting, Unsubscribing from, or Releasing a Node**.

WARNING

- Before scaling in a resource pool, ensure that there are no services running in the pool. Alternatively, go to the resource pool details page, delete the idle nodes where no services are running to scale in the pool.
- When you reduce the number of instances, nodes with deletion lock enabled may be deleted, interrupting running services. This operation cannot be rolled back. Therefore, you are advised not to delete such nodes. However, if really needed, you can delete these nodes by referring to **Deleting, Unsubscribing from, or Releasing a Node**.

Billing

When you increase the number of instances, compute resources will be billed. For details, see **ModelArts Pricing Details**.

You can specify the billing mode of new nodes in a resource pool during scaling. For example, you can create pay-per-use nodes in a yearly/monthly resource pool. If you do not specify this parameter, the billing mode of the created nodes is the same as that of the resource pool. For details, see **Table 5-4**.

Table 5-4 Billing items

Billing Item		Description	Billing Mode	Billing Formula
Compute resource	Dedicated resource pool	Usage of compute resources. For details, see **ModelArts Pricing Details**.	Yearly/Monthly	Specification unit price x Number of compute nodes x Purchase duration

Prerequisites

You have **enabled a Lite Cluster resource pool**.

Constraints

- Only Lite Cluster resource pools in the **Running** state can be resized.
- If there is only one instance node in the Lite Cluster resource pool, scaling in cannot be performed. Therefore, keep at least one node.
- Yearly/Monthly resource pools support only scale-out.

Resizing a Lite Cluster Resource Pool

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.
2. Locate the target resource pool and click **Adjust Capacity** in the **Operation** column. For a yearly/monthly resource pool, only **Scale Out** is displayed. To scale in the resource pool, go to its details page and **unsubscribe from the nodes**.
3. On the **Scale In/Out Dedicated Resource Pool** page, set the parameters by referring to **Table 5-5**.

Table 5-5 Parameters

Parameter	Description
Instance Specifications Type	Instance specifications type of the target Lite Cluster resource pool, which cannot be modified.
Specifications	Specifications of the target Lite Cluster resource pool, which cannot be modified.
Instances in Total	Number of instances in the target Lite Cluster resource pool, which cannot be modified.
AZ Allocation	AZ allocation of nodes after scaling. You can select **Automatic** or **Manual**. • If you select **Automatic**, nodes are randomly allocated to AZs after the scaling. • If you select **Manual**, you can allocate nodes to specified AZs. By default, the value of **Target Instances** indicates how many instances there will be in the AZ. For example: – If there are three instances, and **Target Instances** is set to **5**, the instances in the AZ will be scaled up to 5. – If there are three instances, and **Target Instances** is set to **2**, the instances in the AZ will be scaled in to 2.

Parameter	Description
Container Engine Space Limit	When you scale out a resource pool, and the value of **Target Instances** is greater than that of **Instances in Total**, you can set the container engine space size of the new node to a specified value or unlimited. This operation will cause inconsistencies in **dockerBaseSize** of nodes within the resource pool. As a result, some tasks may run differently on different nodes. **The container engine space size cannot be changed for existing nodes.**
Container Engine Space Size	You can set a specified size by setting **Container Engine Space Limit** to **Manual**.
Target Instances	You can set this parameter for scaling based on service requirements. • Scale out: The value of **Target Instances** is greater than that of **Instances in Total**. • Scale in: The value of **Target Instances** is smaller than that of **Instances in Total**. If **AZ Allocation** is set to **Manual**, you do not need to set this parameter. **Target Instances** indicates how many instances can there be in the target AZ. When you purchase a resource pool, the nodes for certain specifications can be purchased by rack. When you resize the resource pool, the instances are also added or deleted by rack. You can choose to purchase nodes by rack when creating a resource pool, which cannot be modified when resizing a resource pool. Adjust the rack quantity to change the number of target instances.
Node Pool Name	Target Lite cluster resource pool name, which cannot be modified.
Container Engine	Container engine, one of the most important components of Kubernetes, manages the lifecycle of images and containers. The Kubelet interacts with a container runtime through the Container Runtime Interface (CRI). Containerd has a shorter call chain, fewer components, and lower resource requirements, making it more stable. For details about the differences between Containerd and Docker, see **Container Engines**. The CCE cluster version determines the available container engines. If it is earlier than 1.23, only Docker is supported. If it is 1.27 or later, only containerd is supported. For all other versions, both containerd and Docker are options.
Operating system	Choose an OS version from the drop-down list.

4. Configure the node billing mode. When adding nodes, you can enable **Node Billing Mode** to change the billing mode, set the required duration, and enable auto-renewal. For example, you can create pay-per-use nodes in a yearly/monthly resource pool. If the billing mode is not specified, the new nodes share the same billing mode with the resource pool.
5. Click **Submit** and then **OK**.

   On the **Lite Cluster** page, check whether the number of nodes in the resource pool is the value of **Target Instances**.

Related Operations

- **Managing Lite Cluster Nodes**: If there are abnormal or idle nodes that need to be removed from a resource pool, access the resource pool details page, and delete the target nodes (in batches). You can also replace, reset, and renew nodes in the resource pool.
- **Managing Lite Cluster Nodes**: If there are GPU/Ascend resources in the dedicated resource pool, you can upgrade the GPU/Ascend driver based on service requirements.

5.6 Upgrading the Lite Cluster Resource Pool Driver

Scenario

If there are GPU/Ascend resources in a Lite Cluster resource pool node, and the node performance cannot meet your requirements, you can upgrade the driver to resolve known issues, improve performance, or support new functions, ensuring resource pool performance and compatibility.

ModelArts allows you to upgrade the GPU/Ascend driver of a Lite Cluster resource pool on the ModelArts console as required.

Secure Upgrade and Forcible Upgrade

There are two driver upgrade modes: secure upgrade and forcible upgrade. The following table describes the comparisons.

Table 5-6 Secure upgrade and forcible upgrade

Item	Secure Upgrade	Forcible Upgrade
Introduction	Upgrade the driver when the node is idle, which does not affect running tasks. The smooth upgrade reduces the impact on services. After the upgrade starts, the nodes will be isolated (new jobs cannot be delivered). Only after the existing jobs on the node are complete will the upgrade be performed. This may take a rather long time as existing jobs must be completed first.	Running tasks on the node will be ignored and the driver will be directly upgraded. This upgrade mode is fast as you do not need to wait until the node is idle.
Scenario	Non-urgent and gradual upgrade	Urgent and fast upgrade
Precautions	The upgrade period is rather long as nodes must be idle first. Before the upgrade, plan the node idle time to reduce the impact on services.	Running tasks may be interrupted or fail. Exercise caution when using this mode.

Notes and Constraints

- The target Lite Cluster resource pool must be running and contains GPU or Ascend resources.
- To perform the upgrade, you need to restart the node, which is recommended to be performed during off-peak hours to avoid affecting running tasks. You can view the node usage on the **Node Management** page of the resource pool details page.

WARNING

Upgrading the driver will restart the node, which may result in the loss of any customized configurations made on the host.

Upgrading the GPU/Ascend Driver in a Lite Cluster Resource Pool

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**. In the resource pool list, locate the target resource pool, and choose ··· > **Upgrade Driver**.

   Alternatively, click the resource pool name in the list to access its details page. In the navigation pane on the left, choose **Node Pool Management**. Locate the target node pool and choose **More** > **Upgrade Driver** in the **Operation** column.

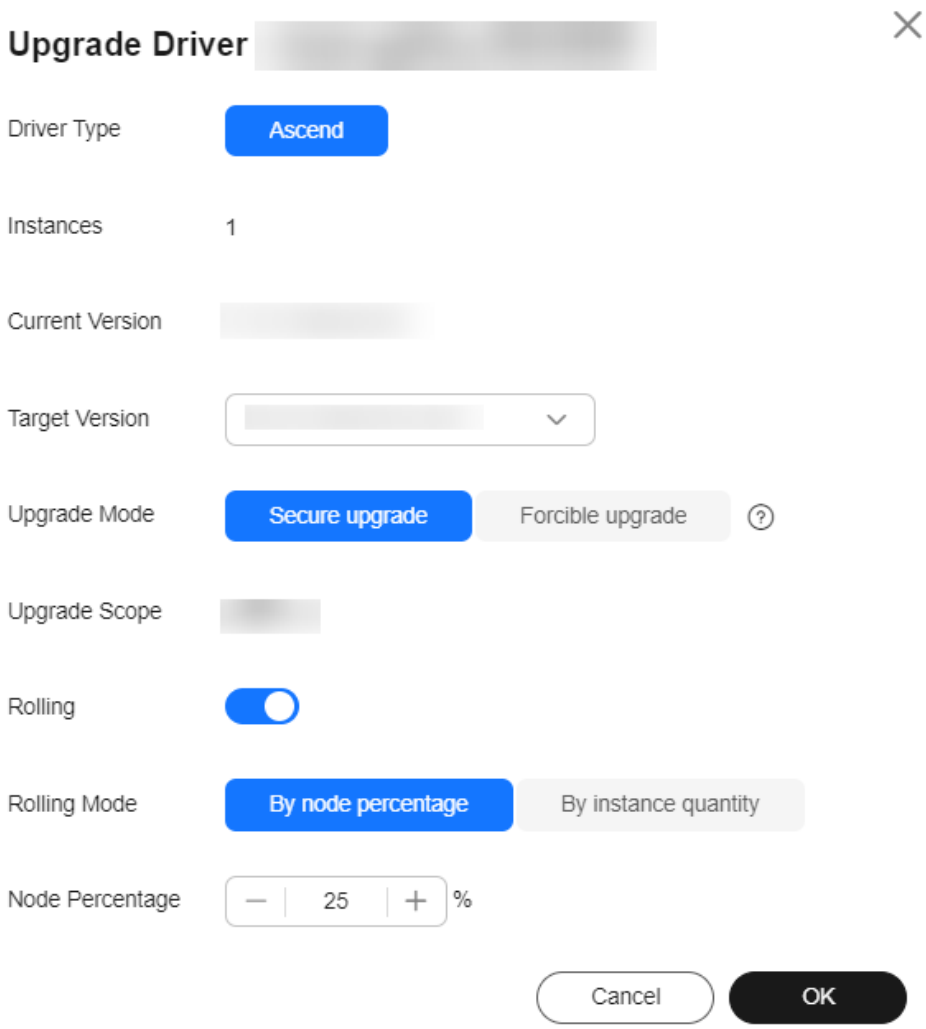
2. In the displayed dialog box, you can view the driver type, number of instances, current version, target version, upgrade mode, upgrade scope, and rolling switch of the Lite Cluster resource pool. Set the parameters by referring to **Table 5-7**.

Table 5-7 Parameters

Parameter	Description
Target Version	Choose the target version from the drop-down list. The driver of the added nodes may not be that of the existing nodes. Select the current driver version for **Target Version**. After the upgrade, all nodes will be upgraded to the same version
Upgrade Mode	Select **Secure upgrade** or **Forcible upgrade**. For details about the differences, see **Secure Upgrade and Forcible Upgrade**. • **Secure upgrade**: Perform the upgrade when no job is running on the node. The upgrade may take a long time. • **Forcible upgrade**: Ignore the running jobs and perform the upgrade directly. This may cause the running jobs to fail.
Rolling	Once enabled, you can upgrade the driver in rolling mode. Rolling upgrade is a gradual instance replacement method that applies to scenarios where service continuity is required. Instances are upgraded in batches to ensure that some instances are running properly during the upgrade, reducing the downtime. Nodes with abnormal drivers will be upgraded during a rolling upgrade, just like other nodes.

Parameter	Description
Rolling Mode	Currently, **By node percentage** and **By instance quantity** are supported. • **By node percentage**: The number of instances to be upgraded is the percentage multiplied by the total number of instances in the resource pool. • **By instance quantity**: The number of instances to be upgraded is the value of this parameter. For different upgrade modes, the policies for upgrading nodes are different. • If **Secure upgrade** is selected, the instances without services are upgraded. To check whether a node has any service, go to the resource pool details page. In the **Nodes** tab, check whether all GPUs and Ascend chips are available. If yes, the node has no services. • If **Forcible upgrade** is selected, random instances are upgraded.
Node Percentage	Set this parameter if **Rolling Mode** is set to **By node percentage**. The number of instances to be upgraded in each batch = The value of **By node percentage** x Total number of instances in the resource pool.
Instances	Set this parameter if **Rolling Mode** is set to **By instance quantity**.

Figure 5-9 Upgrading a driver



3. Click **OK** to start the driver upgrade.

   In the resource pool list, locate the target resource pool, and choose ··· > **Upgrade Driver**. On the displayed page, check whether the current version is the target version. If yes, the driver is upgraded.

5.7 Upgrading the Driver of a Lite Cluster Resource Pool Node

Scenario

If there are GPU/Ascend resources in a Lite Cluster resource pool node, and the node performance cannot meet your requirements, you can upgrade the driver to resolve known issues, improve performance, or support new functions, ensuring resource pool performance and compatibility.

ModelArts allows you to upgrade the GPU/Ascend driver of a Lite Cluster resource pool on the ModelArts console as required.

Notes and Constraints

The target node driver must be running, and the resource pool contains GPU or Ascend resources.

Procedure

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.
2. Go to the resource pool details page. On the **Node Management** page, locate the node whose driver needs to be upgraded and choose **More** > **Upgrade Driver** in the **Operation** column.
3. In the displayed dialog box, select the target version.
4. Click **OK** to upgrade the node driver.

   On the resource pool details page, choose **Node Management** from the navigation pane. Locate the target resource pool and click **More** in the **Operation** column. If the **Upgrade Driver** button is unavailable, the driver has been upgraded.

5.8 Monitoring Lite Cluster Resources

5.8.1 Viewing Lite Cluster Metrics on AOM

ModelArts Lite Cluster regularly collects data on key resource usage for each node in a resource pool, including GPUs, NPUs, CPUs, and memory, and sends this information to AOM. You can view standard metrics on AOM or create custom metrics and send them to AOM for reporting.

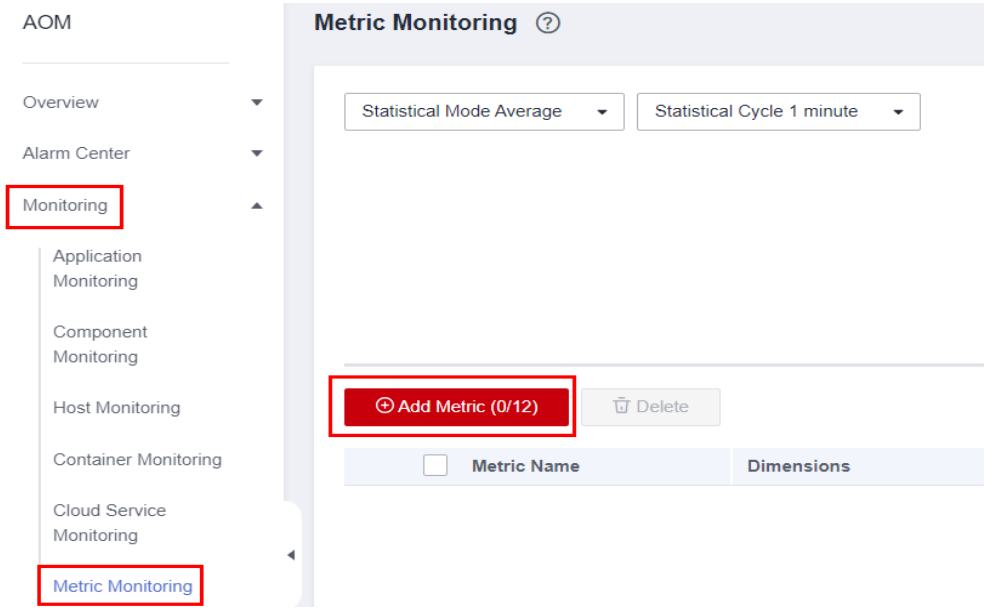
Additionally, you can install Prometheus on ModelArts Lite Cluster to collect metrics. For details, see **Viewing Lite Cluster Metrics Using Prometheus**.

This section describes how to view Lite Cluster metrics on AOM.

Viewing Existing Metrics on AOM

1. Log in to the **console** and search for **AOM** to go to the AOM console.
2. Choose **Monitoring** > **Metric Monitoring**. On the **Metric Monitoring** page that is displayed, click **Add Metric**.

Figure 5-10 Example



3. Add a metric for query.
   - **Add By**: Select **Dimension**.
   - **Metric Name**: Click **Custom Metrics** and select the desired ones for query. For details, see **Table 5-8** and **Table 5-9**.
   - Dimension: Enter the tag of the metric.
4. Click **Confirm**. The metric information is displayed.

Reporting Custom Metrics to AOM

ModelArts allows you to run commands to save custom metrics to AOM.

Constraints

- ModelArts invokes the commands or HTTP APIs specified in the custom configuration every 10 seconds to retrieve metric data.
- The size of the metric data text returned by these commands or HTTP APIs must not exceed 8 KB.

Collecting Custom Metric Data Using Commands

The following is an example of the YAML file for creating a pod for collecting custom metrics:

```
apiVersion: v1
kind: Pod
metadata:
  name: my-task
  annotations:
ei.huaweicloud.com/metrics: '{"customMetrics":[{"containerName":"my-task","exec":{"command":["cat","/metrics/task.prom"]}}]}'  # Replace the containerName and command parameters based on the container from which metric data is obtained and the command used to obtain metric data.
spec:
  containers:
  - name: my-task
image: my-task-image:latest   # Replace it with the actual image.
```


To collect custom metrics, you can either run them alongside your service workload in the same container or use a separate sidecar container for this purpose. This keeps the service workload's resources unchanged.

Data Format of Custom Metrics

The format of custom metrics data must comply with the open metrics specifications. That is, the format of each metric must be:

```
<Metric name>{<Tag name>=<Tag value>, ...} <Sampled value>[Millisecond timestamp]
```

The following is an example (the comment starts with #, which is optional):

```
# HELP http_requests_total The total number of HTTP requests.
# TYPE http_requests_total gauge
html_http_requests_total{method="post",code="200"} 1656 1686660980680
html_http_requests_total{method="post",code="400"} 2 1686660980681
```

Container-Level Metrics

Table 5-8 Container metrics

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
CPU	CPU Usage	ma_container_cpu_util	CPU usage of a measured object	%	0% – 100%	Raw data > 95% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	Used CPU Cores	ma_container_cpu_used_core	Number of CPU cores used by a measured object	Core	≥ 0	N/A	N/A	N/A
	Total CPU Cores	ma_container_cpu_limit_core	Total number of CPU cores that have been applied for a measured object	Core	≥1	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	CPU Memory Usage	ma_container_gpu_mem_util	Percentage of the used GPU memory to the total GPU memory	%	0% – 100%	Raw data > 95% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
Memory	Total Physical Memory	ma_container_memory_capacity_megabytes	Total physical memory that has been applied for a measured object	MB	≥ 0	N/A	N/A	N/A
	Physical Memory Usage	ma_container_memory_util	Percentage of the used physical memory to the total physical memory	%	0% – 100%	Raw data > 95% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	Used Physical Memory	ma_container_memory_used_megabytes	Physical memory that has been used by a measured object (**container_memory_working_set_bytes** in the current working set) (Memory usage in a working set = Active anonymous page and cache, and file-baked page ≤ **container_memory_usage_bytes**)	MB	≥ 0	N/A	N/A	N/A
Storage	Disk Read Rate	ma_container_disk_read_kilobytes	Volume of data read from a disk per second	KB/s	≥ 0	N/A	N/A	N/A
	Disk Write Rate	ma_container_disk_write_kilobytes	Volume of data written into a disk per second	KB/s	≥ 0	N/A	N/A	N/A
GPU memory	Total GPU Memory	ma_container_gpu_mem_total_megabytes	Total GPU memory of a training job	MB	>0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	GPU Memory Usage	ma_container_gpu_mem_util	Percentage of the used GPU memory to the total GPU memory	%	0% – 100%	N/A	N/A	N/A
	Used GPU Memory	ma_container_gpu_mem_used_megabytes	GPU memory used by a measured object	MB	≥ 0	N/A	N/A	N/A
	Idle GPU Memory	ma_container_gpu_mem_free_megabytes	Idle GPU memory of a measured object	MB	≥ 0	N/A	N/A	N/A
GPU	GPU Usage	ma_container_gpu_util	GPU usage of a measured object	%	0% – 100%	Raw data > 95% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	GPU Memory Bandwidth Usage	ma_container_gpu_mem_copy_util	GPU memory bandwidth usage of a measured object For example, the maximum memory bandwidth of GPU Vnt1 is 900 GB/s. If the current memory bandwidth is 450 GB/s, the memory bandwidth usage is 50%.	%	0%–100%	N/A	N/A	N/A
	GPU Encoder Usage	ma_container_gpu_enc_util	GPU encoder usage of a measured object	%	%	N/A	N/A	N/A
	GPU Decoder Usage	ma_container_gpu_dec_util	GPU decoder usage of a measured object	%	%	N/A	N/A	N/A
	GPU Temperature	DCGM_FI_DEV_GPU_TEMP	GPU temperature	°C	Natural number	N/A	N/A	N/A
	GPU Power	DCGM_FI_DEV_POWER_USAGE	GPU power	Watt (W)	>0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	GPU Memory Temperature	DCGM_FI_DEV_MEMORY_TEMP	GPU memory temperature	°C	Natural number	N/A	N/A	N/A
Network I/O	Downlink rate	ma_container_network_receive_bytes	Inbound traffic rate of a measured object	Bytes/s	≥ 0	N/A	N/A	N/A
	Packet receive rate	ma_container_network_receive_packets	Number of data packets received by a NIC per second	Packets/s	≥ 0	N/A	N/A	N/A
	Downlink Error Rate	ma_container_network_receive_error_packets	Number of error packets received by a NIC per second	Packets/s	≥ 0	Raw data > 1 for two consecutive periods	Critical	Packet loss on the network. Submit a service ticket and contact the O&M support to locate the fault.
	Uplink rate	ma_container_network_transmit_bytes	Outbound traffic rate of a measured object	Bytes/s	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	Uplink Error Rate	ma_container_network_transmit_error_packets	Number of error packets sent by a NIC per second	Packets/s	≥ 0	Raw data > 1 for two consecutive periods	Critical	Packet loss on the network. Submit a service ticket and contact the O&M support to locate the fault.
	Packet send rate	ma_container_network_transmit_packets	Number of data packets sent by a NIC per second	Packets/s	≥ 0	N/A	N/A	N/A
NPU	NPU Usage	ma_container_npu_util	NPU usage of a measured object (To be replaced by **ma_container_npu_ai_core_util**)	%	0% – 100%	Raw data > 95% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NPU Memory Usage	ma_container_npu_memory_util	Percentage of the used NPU memory to the total NPU memory (To be replaced by **ma_container_npu_ddr_memory_util** for snt3 series, and **ma_container_npu_hbm_util** for snt9 series)	%	0% – 100%	Raw data > 98% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	Used NPU Memory	ma_container_npu_memory_used_megabytes	NPU memory used by a measured object (To be replaced by **ma_container_npu_ddr_memory_usage_bytes** for snt3 series, and **ma_container_npu_hbm_usage_bytes** for snt9 series)	≥ 0	MB	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	Total NPU Memory	ma_container_npu_memory_total_megabytes	Total NPU memory of a measured object (To be replaced by **ma_container_npu_ddr_memory_bytes** for snt3 series, and **ma_container_npu_hbm_bytes** for snt9 series)	>0	MB	N/A	N/A	N/A
	Overall NPU Usage	ma_container_npu_general_util	The Ascend NPU usage, which covers both AI Cores and Vector Cores. (supported by driver version 24.1.RC2 or later)	%	0% – 100%	N/A	N/A	N/A
	Successful NPU Operator Retransmissions	ma_container_npu_operator_retry_success_cnt	Number of successful NPU operator retransmissions (supported by A3 Ascend HDK 24.1.RC3.3 or later)	Number	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	Failed NPU Operator Retransmissions	ma_container_npu_operator_retry_fail_cnt	Number of failed NPU operator retransmissions (supported by A3 Ascend HDK 24.1.RC3.3 or later)	Number	≥ 0	N/A	N/A	N/A
	Number of Times that an NPU Is Used for Communication	ma_container_npu_borrow_comms_cnt	Number of times that an NPU is used for communication. The more the NPU is used, the lower the transmission efficiency is. (supported by A3 Ascend HDK 24.1.RC3.3 or later)	Number	≥ 0	N/A	N/A	N/A
AI Processor	AI Processor Error Codes	ma_container_npu_ai_core_error_code	Error codes of Ascend AI processors	N/A	N/A	Raw data > 0 for three consecutive periods	Critical	Abnormal card. Submit a service ticket and contact the O&M support.

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	AI Processor Health Status	ma_container_npu_ai_core_health_status	Health status of Ascend AI processors	N/A	• **1**: healthy • **0**: unhealthy	Raw data > 0 for two consecutive periods	Critical	Abnormal card. Submit a service ticket and contact the O&M support.
	AI Processor Power Consumption	ma_container_npu_ai_core_power_usage_watts	Power consumption of Ascend AI processors (processor power consumption for snt9 and snt3, and card power consumption for snt3P)	Watt (W)	>0	N/A	N/A	N/A
	AI Processor Temperature	ma_container_npu_ai_core_temperature_celsius	Temperature of Ascend AI processors	°C	Natural number	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	AI Core Usage	ma_container_npu_ai_core_util	Ascend AI Core usage, which is the ratio of busy to idle cube operators executed on the NPU	%	0% – 100%	Raw data > 95% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	AI Vector Core Usage	ma_container_npu_vector_core_util	Ascend AI Vector Core usage, which is the ratio of busy to idle vector operators executed on the NPU.	%	0% – 100%	Raw data > 95% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	Overall NPU Usage	ma_container_npu_general_util	The Ascend NPU usage, which covers both AI Cores and Vector Cores. (supported by driver version 24.1.RC2 or later)	%	0% – 100%	N/A	N/A	
	AI Core Clock Frequency	ma_container_npu_ai_core_frequency_hertz	AI core clock frequency of Ascend AI processors	Hertz (Hz)	>0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	AI Processor Voltage	ma_container_npu_ai_core_voltage_volts	Voltage of Ascend AI processors	Volt (V)	Natural number	N/A	N/A	N/A
	AI Processor DDR Memory	ma_container_npu_ddr_memory_bytes	Total DDR memory capacity of Ascend AI processors	Byte	>0	N/A	N/A	N/A
	AI Processor DDR Usage	ma_container_npu_ddr_memory_usage_bytes	DDR memory usage of Ascend AI processors	Byte	>0	N/A	N/A	N/A
	AI Processor DDR Memory Utilization	ma_container_npu_ddr_memory_util	DDR memory utilization of Ascend AI processors	%	0% – 100%	Raw data > 95% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	AI Processor HBM Memory	ma_container_npu_hbm_bytes	Total HBM memory of Ascend AI processors (dedicated for Ascend snt9 processors)	Byte	>0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	AI Processor HBM Memory Usage	ma_container_npu_hbm_usage_bytes	HBM memory usage of Ascend AI processors (dedicated for Ascend snt9 processors)	Byte	>0	N/A	N/A	N/A
	AI Processor HBM Memory Utilization	ma_container_npu_hbm_util	HBM memory utilization of Ascend AI processors (dedicated for Ascend snt9 processors)	%	0%–100%	Raw data > 95% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	AI Processor HBM Memory Bandwidth Utilization	ma_container_npu_hbm_bandwidth_util	HBM memory bandwidth utilization of Ascend AI processors (dedicated for Ascend snt9 AI processors)	%	0%–100%	Raw data > 95% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	AI Processor HBM Memory Clock Frequency	ma_container_npu_hbm_frequency_hertz	HBM memory clock frequency of Ascend AI processors (dedicated for Ascend snt9 processors)	Hertz (Hz)	>0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	AI Processor HBM Memory Temperature	ma_container_npu_hbm_temperature_celsius	HBM memory temperature of Ascend AI processors (dedicated for Ascend snt9 processors)	°C	Natural number	N/A	N/A	N/A
	AI CPU Utilization	ma_container_npu_ai_cpu_util	AI CPU utilization of Ascend AI processors	%	0%–100%	N/A	N/A	N/A
	AI Processor Control CPU Utilization	ma_container_npu_ctrl_cpu_util	Control CPU utilization of Ascend AI processors	%	0%–100%	N/A	N/A	N/A

Node-Level Metrics

Table 5-9 Node metric

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
CPU	Total CPU Cores	ma_node_cpu_limit_core	Total number of CPU cores that have been applied for a measured object	Core	≥1	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	Used CPU Cores	ma_node_cpu_used_core	Number of CPU cores used by a measured object	Core	≥ 0	N/A	N/A	N/A
	CPU Usage	ma_node_cpu_util	CPU usage of a measured object	%	0%–100%	Raw data > 95% for two consecutive periods	Major	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	CPU I/O Wait Time	ma_node_cpu_iowait_counter	Disk I/O wait time accumulated since system startup	jiffies	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
Memory	Physical Memory Usage	ma_node_memory_util	Percentage of the used physical memory to the total physical memory	%	0%–100%	Raw data > 95% for two consecutive periods	Major	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	Total Physical Memory	ma_node_memory_total_megabytes	Total physical memory that has been applied for a measured object	MB	≥ 0	N/A	N/A	N/A
Network I/O	Downlink Rate (BPS)	ma_node_network_receive_rate_bytes_seconds	Inbound traffic rate of a measured object	Bytes/s	≥ 0	N/A	N/A	N/A
	Uplink Rate (BPS)	ma_node_network_transmit_rate_bytes_seconds	Outbound traffic rate of a measured object	Bytes/s	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
Storage	Disk Read Rate	ma_node_disk_read_rate_kilobytes_seconds	Volume of data read from a disk per second (Only data disks used by containers are collected.)	KB/s	≥ 0	N/A	N/A	N/A
	Disk Write Rate	ma_node_disk_write_rate_kilobytes_seconds	Volume of data written into a disk per second (Only data disks used by containers are collected.)	KB/s	≥ 0	N/A	N/A	N/A
	Total Cache	ma_node_cache_space_capacity_megabytes	Total cache of the Kubernetes space	MB	≥ 0	N/A	N/A	N/A
	Used Cache	ma_node_cache_space_used_capacity_megabytes	Used cache of the Kubernetes space	MB	≥ 0	N/A	N/A	N/A
	Cache Usage	ma_node_cache_space_used_percent	Cache usage of the Kubernetes space	%	≥ 0	Raw data > 90% for two consecutive periods	Critical	Check the disk in a timely manner to avoid affecting services. Clear invalid data on compute nodes.

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	Total Container Space	ma_node_container_space_capacity_megabytes	Total container space	MB	≥ 0	N/A	N/A	N/A
	Used Container Space	ma_node_container_space_used_capacity_megabytes	Used container space	MB	≥ 0	N/A	N/A	N/A
	Container Space Usage	ma_node_container_space_used_percent	Space usage of a container	%	≥ 0	Raw data > 90% for two consecutive periods	Critical	Check the disk in a timely manner to avoid affecting services. Clear invalid data on compute nodes.
GPU	GPU Usage	ma_node_gpu_util	GPU usage of a measured object	%	0%–100%	N/A	N/A	N/A
	Total GPU Memory	ma_node_gpu_mem_total_megabytes	Total GPU memory of a measured object	MB	>0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	GPU Memory Usage	ma_node_gpu_mem_util	Percentage of the used GPU memory to the total GPU memory	%	0%–100%	Raw data > 97% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	Used GPU Memory	ma_node_gpu_mem_used_megabytes	GPU memory used by a measured object	MB	≥ 0	N/A	N/A	N/A
	Idle GPU Memory	ma_node_gpu_mem_free_megabytes	Idle GPU memory of a measured object	MB	> 0	N/A	N/A	N/A
	Tasks on a Shared GPU	node_gpu_share_job_count	Number of tasks running on a shared GPU	Number	≥ 0	N/A	N/A	N/A
	GPU Temperature	DCGM_FI_DEV_GPU_TEMP	GPU temperature	°C	Natural number	N/A	N/A	N/A
	GPU Power	DCGM_FI_DEV_POWER_USAGE	GPU power	Watt (W)	>0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	GPU Memory Temperature	DCGM_FI_DEV_MEMORY_TEMP	GPU memory temperature	°C	Natural number	N/A	N/A	N/A
NPU	NPU Usage	ma_node_npu_util	NPU usage of a measured object (To be replaced by **ma_node_npu_ai_core_util**)	%	0%–100%	N/A	N/A	N/A
	Overall NPU Usage	ma_node_npu_general_util	The Ascend NPU usage, which covers both AI Cores and Vector Cores. (supported by driver version 24.1.RC2 or later)	%	0%–100%	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NPU Memory Usage	ma_node_npu_memory_util	Percentage of the used NPU memory to the total NPU memory (To be replaced by **ma_node_npu_ddr_memory_util** for snt3 series, and **ma_node_npu_hbm_util** for snt9 series)	%	0%–100%	Raw data > 97% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	Used NPU Memory	ma_node_npu_memory_used_megabytes	NPU memory used by a measured object (To be replaced by **ma_node_npu_ddr_memory_usage_bytes** for snt3 series, and **ma_node_npu_hbm_usage_bytes** for snt9 series)	MB	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	Total NPU Memory	ma_node_npu_memory_total_megabytes	Total NPU memory of a measured object (To be replaced by **ma_node_npu_ddr_memory_bytes** for snt3 series, and **ma_node_npu_hbm_bytes** for snt9 series)	MB	>0	N/A	N/A	N/A
	AI Processor Error Codes	ma_node_npu_ai_core_error_code	Error codes of Ascend AI processors	N/A	N/A	N/A	N/A	N/A
	AI Processor Health Status	ma_node_npu_ai_core_health_status	Health status of Ascend AI processors	N/A	• **1**: healthy • **0**: unhealthy	The value is **0** for two consecutive periods.	Critical	Submit a service ticket.

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	AI Processor Power Consumption	ma_node_npu_ai_core_power_usage_watts	Power consumption of Ascend AI processors (processor power consumption for snt9 and snt3, and card power consumption for snt3P)	Watt (W)	>0	N/A	N/A	N/A
	AI Processor Temperature	ma_node_npu_ai_core_temperature_celsius	Temperature of Ascend AI processors	°C	Natural number	N/A	N/A	N/A
	AI Processor Fan Speed	ma_node_npu_fan_speed_rpm	Fan speed of the Ascend series AI processors	RPM	Natural number	N/A	N/A	N/A
	AI Core Usage	ma_node_npu_ai_core_util	Ascend AI Core usage, which is the ratio of busy to idle cube operators executed on the NPU.	%	0%–100%	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	AI Vector Core Usage	ma_node_npu_vector_core_util	Ascend AI Vector Core usage, which is the ratio of busy to idle vector operators executed on the NPU.	%	0%–100%	N/A	N/A	N/A
	Overall NPU Usage	ma_node_npu_general_util	The Ascend NPU usage, which covers both AI Cores and Vector Cores. (supported by driver version 24.1.RC2 or later)	%	0%–100%	N/A	N/A	N/A
	AI Core Clock Frequency	ma_node_npu_ai_core_frequency_hertz	AI core clock frequency of Ascend AI processors	Hertz (Hz)	>0	N/A	N/A	N/A
	AI Processor Voltage	ma_node_npu_ai_core_voltage_volts	Voltage of Ascend AI processors	Volt (V)	Natural number	N/A	N/A	N/A
	AI Processor DDR Memory	ma_node_npu_ddr_memory_bytes	Total DDR memory capacity of Ascend AI processors	Byte	>0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	AI Processor DDR Usage	ma_node_npu_ddr_memory_usage_bytes	DDR memory usage of Ascend AI processors	Byte	>0	N/A	N/A	N/A
	AI Processor DDR Memory Utilization	ma_node_npu_ddr_memory_util	DDR memory utilization of Ascend AI processors	%	0%–100%	Raw data > 90% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	AI Processor HBM Memory	ma_node_npu_hbm_bytes	Total HBM memory of Ascend AI processors (dedicated for Ascend snt9 processors)	Byte	>0	N/A	N/A	N/A
	AI Processor HBM Memory Usage	ma_node_npu_hbm_usage_bytes	HBM memory usage of Ascend AI processors (dedicated for Ascend snt9 processors)	Byte	>0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	AI Processor HBM Memory Utilization	ma_node_npu_hbm_util	HBM memory utilization of Ascend AI processors (dedicated for Ascend snt9 processors)	%	0%–100%	Raw data > 97% for two consecutive periods	Suggestion	Check if the service resource usage meets the expectation. If the service is normal, no action is required.
	AI Processor HBM Memory Bandwidth Utilization	ma_node_npu_hbm_bandwidth_util	HBM memory bandwidth utilization of Ascend AI processors (dedicated for Ascend snt9 processors)	%	0%–100%	N/A	N/A	N/A
	AI Processor HBM Memory Clock Frequency	ma_node_npu_hbm_frequency_hertz	HBM memory clock frequency of Ascend AI processors (dedicated for Ascend snt9 processors)	Hertz (Hz)	>0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	AI Processor HBM Memory Temperature	ma_node_npu_hbm_temperature_celsius	HBM memory temperature of Ascend AI processors (dedicated for Ascend snt9 processors)	°C	Natural number	N/A	N/A	N/A
	AI CPU Utilization	ma_node_npu_ai_cpu_util	AI CPU utilization of Ascend AI processors	%	0%–100%	N/A	N/A	N/A
	AI Processor Control CPU Utilization	ma_node_npu_ctrl_cpu_util	Control CPU utilization of Ascend AI processors	%	0%–100%	N/A	N/A	N/A
	AI Vector Core Usage	ma_node_npu_vector_core_util	Ascend AI Vector Core usage, which is the ratio of busy to idle vector operators executed on the NPU	%	0%–100%	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NPU Macro Packet Retransmissions	ma_node_npu_macro_retry_cnt	Number of NPU Macro packet retransmissions within a detection period (10 seconds) (supported by A3 24.1.RC2 or later)	Number	≥ 0	N/A	N/A	N/A
	Packets Received by NPU Macro	ma_node_npu_macro_rx_cnt	Number of packets received by NPU Macro within a detection period (10 seconds)	Number	≥ 0	N/A	N/A	N/A
	Invalid Packets Received by NPU Macro	ma_node_npu_macro_crc_error_cnt	Number of invalid CRC packets received by NPU Macro within a detection period (10 seconds)	Number	≥ 0	N/A	N/A	N/A
	NPU Macro BER	ma_node_npu_macro_crc_error_rate	The percentage of invalid CRC packets received by NPU Macro within a detection period	%	0%–100%	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	Successful NPU Operator Retransmissions	ma_node_npu_operator_retry_success_cnt	Number of successful NPU operator retransmissions (supported by A3 Ascend HDK 24.1.RC3.3 or later)	Number	≥ 0	N/A	N/A	N/A
	Failed NPU Operator Retransmissions	ma_node_npu_operator_retry_fail_cnt	Number of failed NPU operator retransmissions (supported by A3 Ascend HDK 24.1.RC3.3 or later)	Number	≥ 0	N/A	N/A	N/A
	Number of Times that an NPU Is Used for Communication	ma_node_npu_borrow_comms_cnt	Number of times that an NPU is used for communication. The more the NPU is used, the lower the transmission efficiency is (supported by A3Ascend HDK 24.1.RC3.3 or later).	Number	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
Node Compute PU Allocation Rate	Total number of compute PUs on a node	ma_node_total_card	Number of compute (GPU or NPU) PUs on a node	Number	≥ 0	N/A	N/A	N/A
	Number of allocated compute PUs on a node	ma_node_allocate_card	Number of compute (GPU or NPU) PUs on a node that have been allocated to containers.	Number	≥ 0	N/A	N/A	N/A
	Node compute PU allocation rate	ma_node_allocate_card_util	Ratio of the number of compute PUs allocated to containers to the total number of compute PUs.	Number	≥ 0	N/A	N/A	N/A
NPU HCCS Links (A3 Specifications)	Available NPU L/Cs	ma_npu_hccs_avail_credit	Number of available L/Cs, which is used to measure the capability of continuing to receive data (supported by driver version 25.1.RC1 or later).	Number	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	Total TX Bandwidth of HCCS Links	ma_node_npu_hccs_total_txbw_per_second	Total bandwidth of data sent by the NPU (supported by driver version 24.1.RC3.5 or later)	GB/s	≥ 0	N/A	N/A	N/A
	Total RX Bandwidth of HCCS Links	ma_node_npu_hccs_total_rxbw_per_second	Total bandwidth of data received by the NPU (supported by driver version 24.1.RC3.5 or later)	GB/s	≥ 0	N/A	N/A	N/A
	HCCS Link TX Bandwidth Details	ma_node_npu_hccs_txbw_per_second	Bandwidth sent by the NPU and each switch chip (supported by driver version 24.1.RC3.5 or later)	GB/s	≥ 0	N/A	N/A	N/A
	HCCS Link RX Bandwidth Details	ma_node_npu_hccs_rxbw_per_second	Bandwidth received by the NPU and each switch chip (supported by driver version 24.1.RC3.5 or later)	GB/s	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
InfiniBand or RoCE network	Total Amount of Data Received by a NIC	ma_node_infiniband_port_received_data_bytes_total	The total number of data octets, divided by 4, (counting in double words, 32 bits), received on all VLs from the port.	Double words (32 bits)	≥ 0	N/A	N/A	N/A
	Total Amount of Data Sent by a NIC	ma_node_infiniband_port_transmitted_data_bytes_total	The total number of data octets, divided by 4, (counting in double words, 32 bits), transmitted on all VLs from the port.	Double words (32 bits)	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
NFS mounting status	NFS Getattr Congestion Time	ma_node_mountstats_getattr_backlog_wait	Getattr is an NFS operation that retrieves the attributes of a file or directory, such as size, permissions, owner, etc. Backlog wait is the time that the NFS requests have to wait in the backlog queue before being sent to the NFS server. It indicates the congestion on the NFS client side. A high backlog wait can cause poor NFS performance and slow system response times.	ms	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NFS Getattr Round Trip Time	ma_node_mountstats_getattr_rtt	Getattr is an NFS operation that retrieves the attributes of a file or directory, such as size, permissions, owner, etc. RTT stands for Round Trip Time and it is the time from when the kernel RPC client sends the RPC request to the time it receives the reply34. RTT includes network transit time and server execution time. RTT is a good measurement for NFS latency. A high RTT can indicate network or server issues.	ms	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NFS Access Congestion Time	ma_node_mountstats_access_backlog_wait	Access is an NFS operation that checks the access permissions of a file or directory for a given user. Backlog wait is the time that the NFS requests have to wait in the backlog queue before being sent to the NFS server. It indicates the congestion on the NFS client side. A high backlog wait can cause poor NFS performance and slow system response times.	ms	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NFS Access Round Trip Time	ma_node_mountstats_access_rtt	Access is an NFS operation that checks the access permissions of a file or directory for a given user. RTT stands for Round Trip Time and it is the time from when the kernel RPC client sends the RPC request to the time it receives the reply34. RTT includes network transit time and server execution time. RTT is a good measurement for NFS latency. A high RTT can indicate network or server issues.	ms	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NFS Lookup Congestion Time	ma_node_mountstats_lookup_backlog_wait	Lookup is an NFS operation that resolves a file name in a directory to a file handle. Backlog wait is the time that the NFS requests have to wait in the backlog queue before being sent to the NFS server. It indicates the congestion on the NFS client side. A high backlog wait can cause poor NFS performance and slow system response times.	ms	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NFS Lookup Round Trip Time	ma_node_mountstats_lookup_rtt	Lookup is an NFS operation that resolves a file name in a directory to a file handle. RTT stands for Round Trip Time and it is the time from when the kernel RPC client sends the RPC request to the time it receives the reply34. RTT includes network transit time and server execution time. RTT is a good measurement for NFS latency. A high RTT can indicate network or server issues.	ms	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NFS Read Congestion Time	ma_node_mountstats_read_backlog_wait	Read is an NFS operation that reads data from a file. Backlog wait is the time that the NFS requests have to wait in the backlog queue before being sent to the NFS server. It indicates the congestion on the NFS client side. A high backlog wait can cause poor NFS performance and slow system response times.	ms	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NFS Read Round Trip Time	ma_node_mountstats_read_rtt	Read is an NFS operation that reads data from a file. RTT stands for Round Trip Time and it is the time from when the kernel RPC client sends the RPC request to the time it receives the reply34. RTT includes network transit time and server execution time. RTT is a good measurement for NFS latency. A high RTT can indicate network or server issues.	ms	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NFS Write Congestion Time	ma_node_mountstats_write_backlog_wait	Write is an NFS operation that writes data to a file. Backlog wait is the time that the NFS requests have to wait in the backlog queue before being sent to the NFS server. It indicates the congestion on the NFS client side. A high backlog wait can cause poor NFS performance and slow system response times.	ms	≥ 0	N/A	N/A	N/A

Classification	Name	Metric	Description	Unit	Value Range	Alarm Threshold	Alarm Severity	Solution
	NFS Write Round Trip Time	ma_node_mountstats_write_rtt	Write is an NFS operation that writes data to a file. RTT stands for Round Trip Time and it is the time from when the kernel RPC client sends the RPC request to the time it receives the reply34. RTT includes network transit time and server execution time. RTT is a good measurement for NFS latency. A high RTT can indicate network or server issues.	ms	≥ 0	N/A	N/A	N/A

Label Metrics

Table 5-10 Metric names

Classification	Metric	Description
Container metrics	pod_name	Name of the pod to which the container belongs

Classification	Metric	Description
	pod_id	ID of the pod to which the container belongs
	node_ip	IP address of the node to which the container belongs
	container_id	Container ID
	cluster_id	Cluster ID
	cluster_name	Cluster name
	container_name	Name of the container
	namespace	Namespace where the POD created by the user is located.
	app_kind	The value is obtained from the **kind** field in the first **ownerReferences**.
	app_id	The value is obtained from the **uid** field in the first **ownerReferences**.
	app_name	The value is obtained from the **name** field in the first **ownerReferences**.
	npu_id	Ascend card ID, for example, **davinci0** (to be discarded)
	device_id	Physical ID of Ascend AI processors
	device_type	Type of Ascend AI processors
	pool_id	ID of a resource pool corresponding to a physical dedicated resource pool
	pool_name	Name of a resource pool corresponding to a physical dedicated resource pool
	gpu_uuid	UUID of the GPU used by the container
	gpu_index	Index of the GPU used by the container
	gpu_type	Type of the GPU used by the container
Node metrics	cluster_id	ID of the CCE cluster to which the node belongs
	node_ip	IP address of the node
	host_name	Hostname of a node
	pool_id	ID of a resource pool corresponding to a physical dedicated resource pool
	project_id	Project ID of the user in a physical dedicated resource pool

Classification	Metric	Description
	npu_id	Ascend card ID, for example, **davinci0** (to be discarded)
	device_id	Physical ID of Ascend AI processors
	device_type	Type of Ascend AI processors
	gpu_uuid	UUID of a node GPU
	gpu_index	Index of a node GPU
	gpu_type	Type of a node GPU
	device_name	Device name of an InfiniBand or RoCE network NIC
	port	Port number of the IB NIC
	physical_state	Status of each port on the IB NIC
	firmware_version	Firmware version of the InfiniBand NIC
	filesystem	NFS-mounted file system
	mount_point	NFS mount point
Diagnosis	cluster_id	ID of the CCE cluster to which the node with the GPU equipped belongs
	node_ip	IP address of the node where the GPU resides
	pool_id	ID of a resource pool corresponding to a physical dedicated resource pool
	project_id	Project ID of the user in a physical dedicated resource pool
	gpu_uuid	GPU UUID
	gpu_index	Index of a node GPU
	gpu_type	Type of a node GPU
	device_name	Device name of an InfiniBand or RoCE network NIC
	port	Port number of the IB NIC
	physical_state	Status of each port on the IB NIC
	firmware_version	Firmware version of the InfiniBand NIC

5.8.2 Viewing Lite Cluster Metrics Using Prometheus

Prometheus is an open-source monitoring tool. ModelArts supports the Exporter function, enabling you to use third-party monitoring systems like Prometheus to obtain metric data collected by ModelArts.

This section describes how to view Lite Cluster metrics using Prometheus.

Notes and Constraints

- You must enable the monitoring function on the configuration management page of the ModelArts Lite cluster resource pool details page.
- After this function is enabled, third-party components compatible with the Prometheus metric format can obtain the metric data collected by ModelArts through API **http://***<Node IP address>***:***<Port number>***/metrics**.
- Before enabling this function, you need to confirm the port number. It can be any number within the range of 10120 to 10139. Ensure that the selected port number is not being used by any other applications on each node.

Interconnecting Prometheus with ModelArts in Kubernetes

1. Use kubectl to connect to the target cluster. For details, see **Accessing a Cluster Using kubectl**.
2. Configure Kubernetes access authorization.

   Use any text editor to create the **prometheus-rbac-setup.yml** file. The content of the YAML file is as follows:

   NOTE

   This YAML file defines the role (ClusterRole) for Prometheus and assigns the necessary access permissions. Additionally, it creates the account (ServiceAccount) for Prometheus and binds this account to the role (ClusterRoleBinding).

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: prometheus
rules:
- apiGroups: [""]
  resources:
  - pods
  verbs: ["get", "list", "watch"]
- nonResourceURLs: ["/metrics"]
  verbs: ["get"]
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: prometheus
  namespace: default
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: prometheus
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: prometheus
subjects:
- kind: ServiceAccount
```



```
  name: prometheus
  namespace: default
```

3. Create RBAC resources:

```
$ kubectl create -f prometheus-rbac-setup.yml
clusterrole "prometheus" created
serviceaccount "prometheus" created
clusterrolebinding "prometheus" created
```

4. Use any text editor to create the **prometheus-config.yml** file with the following content. This YAML file manages Prometheus configurations. When Prometheus is deployed, these configurations can be used by containers through file system mounting.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: prometheus-config
data:
  prometheus.yml: |
    global:
      scrape_interval: 10s
    scrape_configs:
    - job_name: 'modelarts'
      tls_config:
        ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
      bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
      kubernetes_sd_configs:
      - role: pod
      relabel_configs:
        - source_labels: [__meta_kubernetes_pod_name]    # Collect metric data from the POD whose name starts with a fixed string. If the ModelArts Node Agent version is earlier than 7.2.0, the pod prefix is maos-node-agent-. Otherwise, the pod prefix is modelarts-metric-collector.
          action: keep
          regex: ^(maos-node-agent-|modelarts-metric-collector).+
        - source_labels: [__address__]    # Specifies the IP address and port number for obtaining metric data. __address__:9390 specifies the IP address of the POD, which is also the node IP address.
          action: replace
          regex: '(.*)'
          target_label: __address__
          replacement: "${1}:10120"
```

5. Create ConfigMap resources:

```
$ kubectl create -f prometheus-config.yml
configmap "prometheus-config" created
```

6. Use any text editor to create the **prometheus-deployment.yml** file. The content is as follows:

   NOTE

   This YAML file is used to deploy Prometheus. It grants the permissions of the created account (ServiceAccount) to Prometheus and mounts the created ConfigMap resource to the **/etc/prometheus** directory of the Prometheus container as a file system. The **--config.file=/etc/prometheus/prometheus.yml** parameter specifies the configuration file used by **/bin/prometheus**.

```
apiVersion: v1
kind: "Service"
metadata:
  name: prometheus
  labels:
    name: prometheus
spec:
  ports:
  - name: prometheus
    protocol: TCP
    port: 9090
    targetPort: 9090
  selector:
    app: prometheus
```



```
  type: NodePort
---
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    name: prometheus
  name: prometheus
spec:
  replicas: 1
  selector:
    matchLabels:
      app: prometheus
  template:
    metadata:
      labels:
        app: prometheus
    spec:
      hostNetwork: true
      serviceAccountName: prometheus
      serviceAccount: prometheus
      containers:
      - name: prometheus
        image: prom/prometheus:latest
        imagePullPolicy: IfNotPresent
        command:
        - "/bin/prometheus"
        args:
        - "--config.file=/etc/prometheus/prometheus.yml"
        ports:
        - containerPort: 9090
          protocol: TCP
        volumeMounts:
        - mountPath: "/etc/prometheus"
          name: prometheus-config
      volumes:
      - name: prometheus-config
        configMap:
          name: prometheus-config
```

7. Create a Prometheus instance and check the creation result:

```
$ kubectl create -f prometheus-deployment.yml
service "prometheus" created
deployment "prometheus" created

$ kubectl get pods
NAME                          READY     STATUS        RESTARTS   AGE
prometheus-55f655696d-wjqcl   1/1       Running       0          5s

$ kubectl get svc
NAME            TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
kubernetes      ClusterIP   10.96.0.1        <none>        443/TCP          131d
prometheus      NodePort    10.101.255.236   <none>        9090:32584/TCP   42s
```

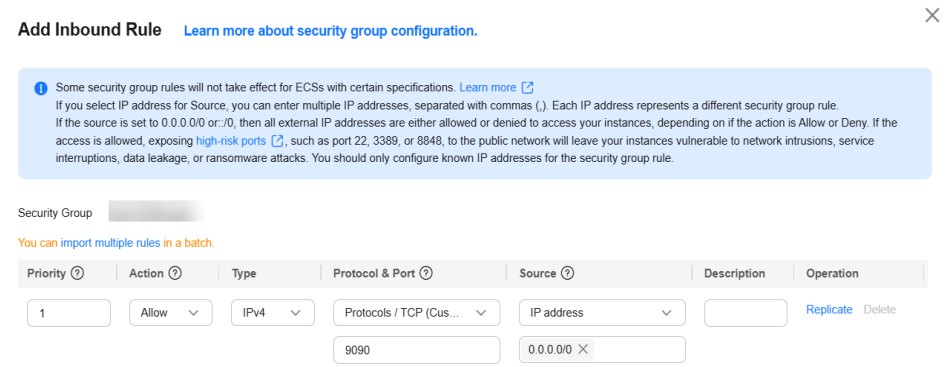
Viewing Metric Data Collected by Prometheus

1. On the CCE console, bind an EIP to the node where Prometheus is deployed. Enable the security group configuration for the node and add an inbound rule to allow external access to port 9090.

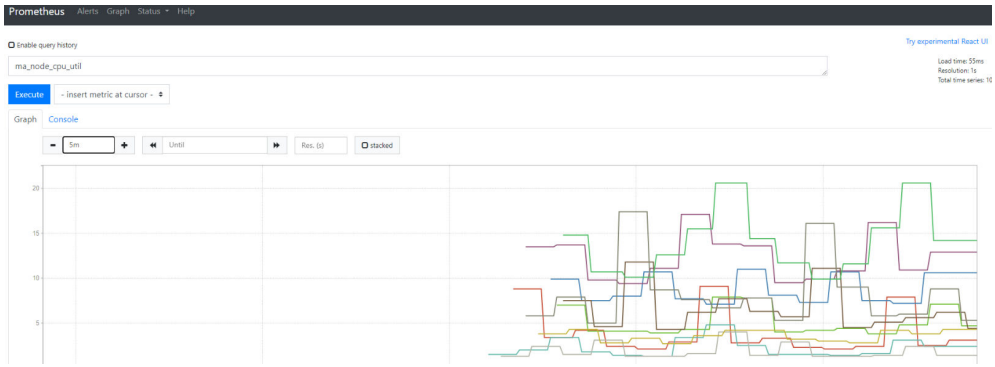
   NOTE

   If you use Grafana to interconnect with Prometheus for report creation, you can deploy Grafana within the cluster. In this scenario, there is no need to bind a public IP address to Prometheus or configure a security group for it. Instead, you only need to bind a public IP address to Grafana and configure its security group.

Figure 5-11 Adding an inbound rule



2. Enter **http://<*EIP*>:9090** in the address box of the browser. The Prometheus monitoring page is displayed. Click **Graph** and enter a metric name in the text box to view the metric data collected by Prometheus.



5.9 Releasing Lite Cluster Resources

You can release Lite Cluster resources that are no longer used. For details about how to stop billing, see **Stopping Billing**.

WARNING

- Deleting a resource pool will delete the associated resources as well. The operation cannot be undone. The yearly/monthly nodes in the resource pool need to be unsubscribed from or released separately.
- When you delete a Lite Cluster resource pool, the associated disks are also removed, and any data on those disks will be permanently deleted and cannot be recovered.
- If there are nodes with deletion lock enabled in the resource pool, such nodes will be deleted when the resource pool is deleted, which interrupts running services. Exercise caution when performing this operation as it cannot be rolled back and ensure that key services are not affected.

Unsubscribing from a Yearly/Monthly Lite Cluster Resource Pool

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.

2. In the resource pool list, choose ··· > **Unsubscribe** in the **Operation** column.
3. Confirm the target resources and select the reason for unsubscription.
4. Confirm the information and select **After being unsubscribed from, the resource not in the recycle bin will be deleted immediately and cannot be restored. I have backed up data or no longer need the data**.
5. Click **Unsubscribe** and confirm the resources.
6. Click **Unsubscribe** again to finish the subscription.

6 Lite Cluster Plug-in Management

6.1 Overview

ModelArts offers several plug-ins to help you expand resource pool functions as needed.

Default Plug-ins

The plug-ins are installed by default when you create a dedicated resource pool.

> ⚠ CAUTION
>
> Plug-ins installed by default in a resource pool cannot be uninstalled.

Table 6-1 Default plug-ins

Plug-in	Description
Node Fault Detection (ModelArts Node Agent)	ModelArts Node Agent is a plug-in for monitoring cluster node exceptions, also, a component for connecting to third-party monitoring platforms. It is a daemon that runs on each node to collect node problems from different daemon processes.
AI Suite (ModelArts Device Plugin)	The CCE AI suite, Ascend NPU, is a device management plug-in that supports Huawei NPUs in containers. When you **enable Lite Cluster resources**, this plug-in is automatically downloaded only when the instance specification type is set to Ascend.
Volcano Scheduler	Volcano is a batch scheduling platform based on Kubernetes. It provides a series of features required by machine learning, deep learning, bioinformatics, genomics, and other big data applications, as a powerful supplement to Kubernetes capabilities.

Installing the Plug-in Manually

You can install plug-ins to extend resource pool functions as required.

Table 6-2 Default plug-ins

Plug-in	Description
Cluster Autoscaler	Cluster Autoscaler is a plug-in for elastic scaling of ModelArts resource pools in a cluster. It can be used to scale in or out node pools based on user-defined rules.

Plug-in Lifecycle

Status	Status Attribute	Description
Installing	Intermediate	The plug-in is being deployed. If all instances cannot be scheduled due to incorrect plug-in configuration or insufficient resources, the system sets the plug-in status to **Unavailable** 10 minutes later.
Running	Stable	The plug-in is running, all plug-in instances are deployed, and the plug-in can be used properly.
Upgrading	Intermediate	The plug-in is being upgraded.
Unavailable	Stable	The plug-in is abnormal and cannot be used. You can click the status to view the failure cause.
Deleting	Intermediate	The plug-in is being deleted. If this state stays for a long time, an exception occurred.

Searching for a Plug-in on the Plug-in Square

The **ModelArts** Plug-in Square provides various plug-ins. You can view the plug-in details and install them to a specified resource pool as needed.

Table 6-3 Supported operations on the Plug-in Square

Operation	Description	Procedure
Searching for and viewing a plug-in	Search for and view a plug-in.	Log in to the **ModelArts console**. In the navigation pane on the left, choose **Add-ons**. Choose a resource type from the drop-down list to filter plug-ins, or enter a keyword in the search box to search for a plug-in.
Viewing plug-in details	View the plug-in details, including the plug-in introduction and component list.	1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Add-ons**. 2. Click the plug-in name to view its details.

Operation	Description	Procedure
Installing a plug-in	Certain plug-ins can be manually installed.	1. Log in to the ModelArts console. In the navigation pane on the left, choose **Add-ons**. 1. Locate the target plug-in and click **Install**. 2. In the displayed dialog box, select the resource type of the plug-in to be installed. For some plug-ins, you also need to select a plug-in version. Set the information and click **Next**. • **Dedicated cluster**: Install the plug-in to a resource pool. The supported resource pool types vary depending on the plug-in. See the supported types on the GUI accordingly. • **Dedicated node**: Install the plug-in to a specific node in the resource pool. Perform operations and run commands as prompted. 3. Configure related parameters. The configurations vary depending on the plug-in. For details, see section"Plug-ins".

Viewing the Lite Cluster Plug-in on the Resource Pool Details Page

In the **Plug-ins** tab of the resource pool details page, perform the operations described in **Table 6-4**.

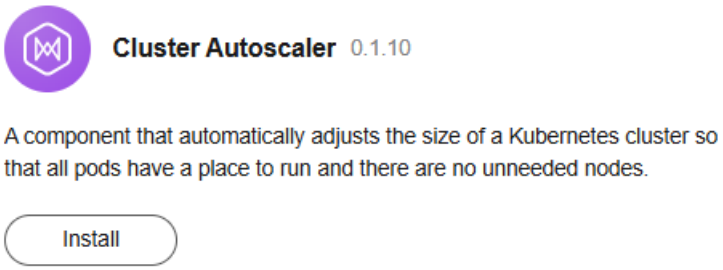
Table 6-4 Related operations

Operation	Description	Procedure
Querying the plug-ins	View all plug-ins of a resource pool. On this page, you can view plug-in details, install, upgrade, and uninstall plug-ins, and manage plug-ins in a centralized manner.	1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**. 2. Click the resource pool name to access its details page. 3. In the navigation pane on the left, choose **Plug-ins**.
Viewing plug-in details	View the plug-in details, including the plug-in introduction and component list.	1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**. 2. Click the resource pool name to access its details page. 3. In the navigation pane on the left, choose **Plug-ins**. 4. Click the plug-in name to view its details.
Plug-ins installed by default	When you create a resource pool, certain plug-ins are installed by default.	**Enabling Lite Cluster Resources**

Operation	Description	Procedure
Installing the plug-in manually	Install the specified plug-in in the resource pool.	Method 1: Install the plug-in when you enable Lite Cluster resources. For details, see **Enabling Lite Cluster Resources**. Method 2: 1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**. 2. Click the resource pool name to access its details page. 3. In the navigation pane on the left, choose **Plug-ins**. 4. Locate the plug-in to be installed and click **Install**, as shown in **Figure 6-1**. 5. In the displayed dialog box, configure the parameters. Currently, Lite Cluster allows you to manually install the elastic cluster engine plug-in. For details about the parameters, see **Table 6-11**.
Editing a plug-in	Edit plug-in parameters.	1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**. 2. Click the resource pool to access its details page. 3. In the navigation pane on the left, choose **Plug-ins**. 4. Locate the plug-in to be edited in the list and click **Edit**. The configurations vary depending on plug-ins. For details, see "Plug-ins". Only the following plug-in versions can be edited: • ModelArts Node Agent 7.2.0 or later • AI suite (Ascend NPU) 2.1.53 or later • Volcano Scheduler 1.17.11 or later • Cluster Autoscaler 0.1.13 or later 5. Click **OK**.

Operation	Description	Procedure
Upgrading a plug-in	Upgrade the plug-in to the latest version.	1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**. 2. Click the resource pool to access its details page. 3. In the navigation pane on the left, choose **Plug-ins**. 4. Locate the plug-in to be upgraded in the list and click **Upgrade**. Currently, Lite Cluster allows you to manually install the elastic cluster engine plug-in. For details about the parameters, see **Table 6-11**. 5. Click **OK**. **CAUTION** • Plug-ins are deployed based on Helm templates. To modify or upgrade plug-ins, you need to use the plug-in list on the ModelArts console or the open plug-in management APIs. Do not manually modify related resources on the CCE server. Otherwise, exceptions may occur, for example, parameter settings may be lost or overwritten after the upgrade. • During the plug-in upgrade, some functions of the resource pool may be affected. You should check the status and version compatibility of all external dependencies before the upgrade and reserve enough time for the upgrade. For details about the impact, see the plug-in description.
Uninstalling a plug-in	Uninstall a plug-in from the resource pool. This operation cannot be undone.	1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**. 2. Click the resource pool to access its details page. 3. In the navigation pane on the left, choose **Plug-ins**. 4. Locate the plug-in to be uninstalled in the list and click **Uninstall**. 5. In the displayed dialog box, enter **DELETE** and click **OK**.

Figure 6-1 Installing a plug-in



FAQ

1. If the plug-in must be installed is unavailable or is being installed or deleted for a long time, you can click the resource pool name to view the basic information. In the CCE cluster area, click the CCE cluster in the resource pool.

   Go to the plug-in center, locate the target plug-in, and click it to view the details. In the instance list, click the abnormal instance and check the exception cause.
2. If an optional plug-in is unavailable or is being installed or deleted for a long time, uninstall the plug-in and reinstall it. If the plug-in is still unavailable after the re-installation, view the exception details by referring to the previous step.
3. If the fault persists, contact ModelArts technical personnel.

6.2 Node Fault Detection (ModelArts Node Agent)

Description

ModelArts Node Agent is a plug-in for monitoring cluster node exceptions, also, a component for connecting to third-party monitoring platforms. It is installed in each Kubernetes resource pool by default. It is a daemon running on each node and can collect node problems from different daemon processes.

Installing a Plug-in

Certain plug-ins are automatically installed when you create a dedicated resource pool. Currently, you cannot upgrade the plug-ins yourself.

Components

Table 6-5 Plug-in component

Component	Description	Resource Type
maos-node-agent	Monitor cluster node exceptions and interconnect with third-party monitoring platforms.	DeamonSet

Change History

Table 6-6 Release history

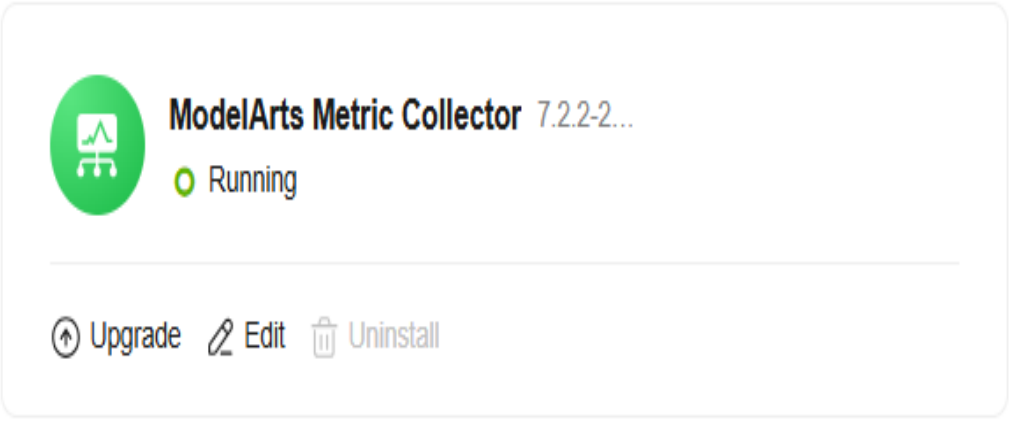
Plug-in Version	New Feature
7.2.0	ModelArts Node Agent is supported to detect faults and monitor cluster node exceptions.
6.8.0	ModelArts Node Agent is supported to detect faults and monitor cluster node exceptions.
6.7.0	ModelArts Node Agent is supported to detect faults and monitor cluster node exceptions.
6.6.0	ModelArts Node Agent is supported to detect faults and monitor cluster node exceptions.

6.3 ModelArts Metric Collector

Description

ModelArts Metric Collector, a default built-in plug-in of ModelArts, runs as a node daemon to collect node and job metrics and report them to AOM. For details about the metrics, see **Viewing Lite Cluster Metrics on AOM**.

Figure 6-2 ModelArts Metric Collector



Constraints

- The plug-in is automatically installed during resource pool creation and cannot be uninstalled.
- This plug-in is automatically installed if ModelArts Node Agent is upgraded to the latest version for an existing resource pool.
- During the plug-in upgrade, the pod for metric collection restarts. As a result, metrics may not be reported for a short period of time. Exercise caution when performing the operation.

Components

Component	Description	Resource Type
modelarts-metric-collector	Node and container metrics collection	DaemonSet

Parameters

Parameter	Description
Standby Node Metric Reporting	Whether the standby node of a dedicated pool reports metrics. The default value is **false**.
Enable Exporter	Third-party monitoring systems such as Prometheus can obtain metrics collected by ModelArts. If this function is disabled, third-party monitoring systems such as Prometheus cannot collect metrics. This function is enabled by default. Dedicated pool: Enable this function if you want to use inference job metrics for scaling.

Parameter	Description
Report Metrics to a Custom Common Prometheus Instance on AOM	By default, metrics are reported to the **Prometheus_AOM_Default** instance of AOM. If this function is enabled, metrics are reported to the custom Prometheus common instance, as shown in **Figure 6-3**. If this function is disabled, metrics are reported to the default Prometheus instance, that is, the **Prometheus_AOM_Default** instance, as shown in **Figure 6-4**.

Figure 6-3 Custom Prometheus common instance

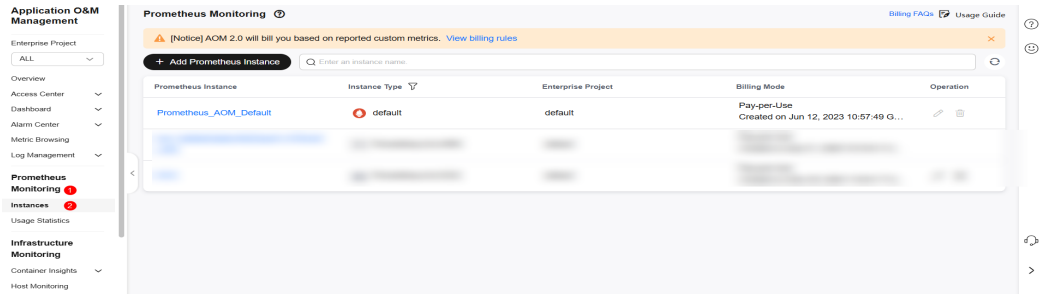
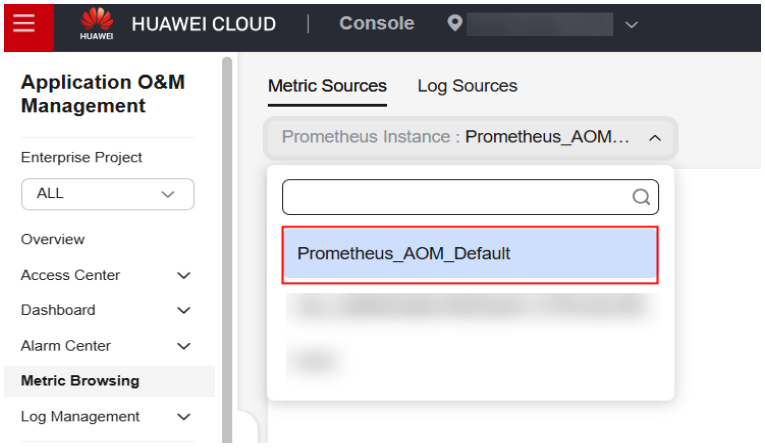


Figure 6-4 Prometheus_AOM_Default instance



6.4 AI Suite (ModelArts Device Plugin)

Description

The AI suite, ModelArts Device Plugin, is a device management plug-in that supports Huawei NPUs in containers.

Constraints

When you **create a dedicated resource pool**, this plug-in is automatically downloaded only when the instance specification type is set to Ascend.

Components

Table 6-7 Plug-in components

Component	Description	Resource Type
modelarts-device-plugin	Allows containers to use ModelArts Device Plugin devices.	Daemon Set

Change History

Table 6-8 Release history

Plug-in Version	New Feature
7.2.0	Supported Kubernetes v1.31. Supported A2 single-PU multi-pods.
7.0.1	Supported Kubernetes v1.31.
7.0.0	Supported Kubernetes v1.31.
2.1.53	Fixed security vulnerabilities.
2.1.46	Kubernetes v1.31 is supported.
2.1.23	Fixed some issues.
2.1.22	• Fixed some page display issues. • Supernode information can be obtained. • NPU topology information can be reported. • Fixed log printing issues.
2.1.5	• CCE v1.29 clusters are supported. • Added silent fault codes.
1.2.14	Supported NPU monitoring.
1.2.5	Supported automatic installation of NPU drivers.

6.5 Volcano Scheduler

Description

Volcano is a batch scheduling platform based on Kubernetes. It provides a series of features required by machine learning, deep learning, bioinformatics, genomics, and other big data applications, as a powerful supplement to Kubernetes capabilities.

Volcano provides general computing capabilities such as high-performance job scheduling, heterogeneous chip management, and job running management. It accesses the computing frameworks for various industries such as AI, big data, gene, and rendering and schedules up to 1,000 pods per second for end users, greatly improving scheduling efficiency and resource utilization.

Volcano provides job scheduling, job management, and queue management for computing applications. Its main features are as follows:

- Diverse computing frameworks: CRD provides common APIs for batch computing tasks. With various plug-ins and advanced job lifecycle management, computing frameworks such as TensorFlow, MPI, and Spark can run on Kubernetes in containers.
- Advanced scheduling: Advanced scheduling capabilities are provided for batch computing and high-performance computing scenarios, including group scheduling, priority preemption, packing, resource reservation, and task topology.
- Queue management: Queues can be effectively managed for scheduling jobs. Complex job scheduling can be implemented based on queue priorities or through multi-level queues.

Volcano has been open-sourced in GitHub at **https://github.com/volcano-sh/volcano**.

Constraints

When upgrading the plug-in, exercise caution when you downgrade a later version to an earlier version, as this may cause job scheduling failures.

Installing the Plug-in

Certain plug-ins are automatically installed when you enable Lite Cluster resources. For details, see **Enabling Lite Cluster Resources**.

Components

Table 6-9 Plug-in components

Component	Description	Resource Type
volcano-scheduler	Schedule pods.	Deployment
volcano-controller	Synchronize CRDs.	Deployment
volcano-admission	Webhook server, which verifies and modifies resources such as pods and jobs	Deployment

Change History

Table 6-10 Release history

Plug-in Version	New Feature
1.17.11	• Optimized the cabinet affinity and packing capabilities. • Optimized the Ascend NPU preemption capability. • Supported Kubernetes v1.32. • Supported topology affinity scheduling of Ascend high-density models.
1.16.8	• Optimized the resource scheduling capability of supernodes. • Kubernetes v1.31 is supported.
1.15.8	Supported Ascend NPU dual-die affinity scheduling.
1.15.6	Resources can be oversubscribed based on pod profiling.
1.13.5	• Supported scale-in of customized resources based on node priorities. • Optimized the association between preemption and node scale-out.
1.12.18	• Adapted to CCE 1.29 clusters. • The preemption function is enabled by default.
1.12.1	Optimized application auto scaling performance.
1.11.9	• Optimized sorting capability of NPU rank table. • Supported priority-based scheduling in autoscaling scenarios.
1.10.10	Fixed the issue that the local PV plug-in fails to calculate the number of pods pre-bound to the node.
1.10.7	Fixed the issue that the local PV plug-in fails to calculate the number of pods pre-bound to the node.
1.7.1	Supported clusters v1.25.

6.6 Cluster Autoscaler

Description

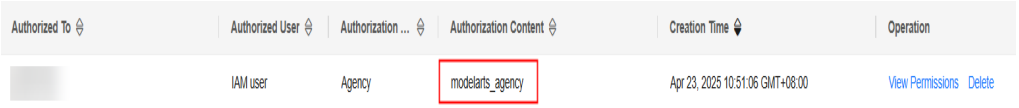
Cluster Autoscaler is a plug-in for elastic scaling of ModelArts resource pools in a cluster. It can be used to scale in or out node pools based on user-defined rules.

Constraints

- This plug-in is supported only for nodes in a pay-per-use or yearly/monthly Lite Cluster resource pool.
- If the resource specifications are sold out or the underlying capacity is insufficient, the scale-out will fail.
- This plug-in is not supported for Lite Cluster resource pools purchased by rack.
- This plug-in uses the global agency permissions to perform operations on resource pools. If the global agency includes blacklist policies related to resource pool operations, delete the blacklist first. To do so, perform the following steps:

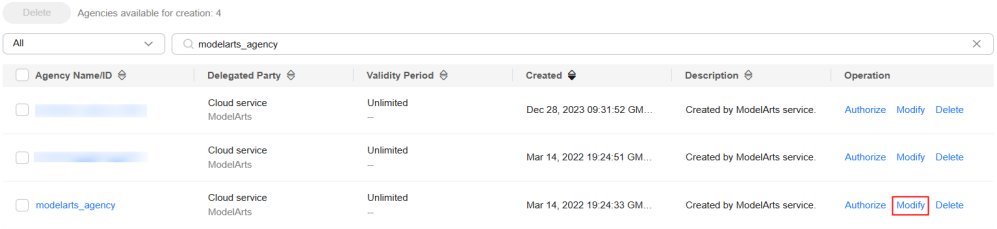
  a. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Permission Management**. Locate the target authorization and obtain the content in the **Authorization Content** column, which is the name of the agency granted to the current user.

  Figure 6-5 Authorization content

  b. Go to the **IAM console**. In the navigation pane on the left, choose **Agencies**. Locate the obtained agency and click **Modify** in the **Operation** column.

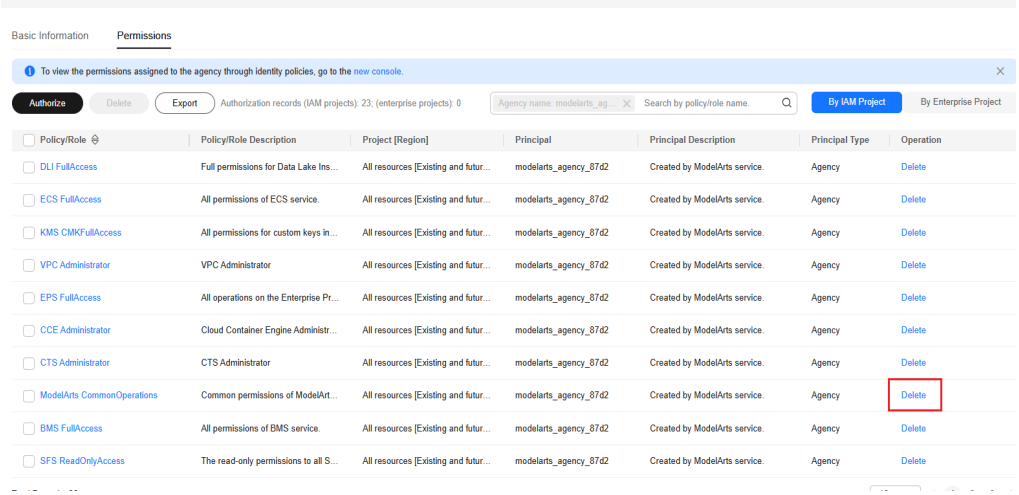
  Figure 6-6 IAM agency

  c. Click the **Permissions** tab.

  d. Locate **ModelArts CommonOperations** and click **Delete** in the **Operation** column. In the displayed dialog box, click **OK**.

Figure 6-7 Deleting the ModelArts CommonOperations permission



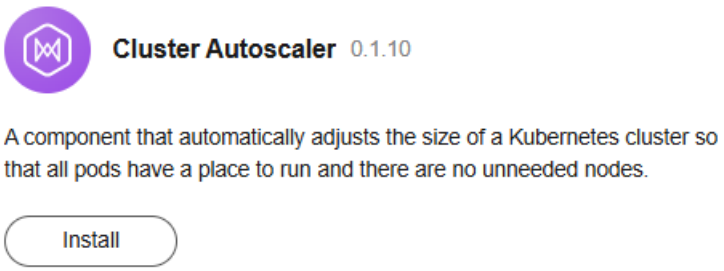
Installing a Plug-in

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Lite Cluster** under **Resource Management**.
2. Click the resource pool name to access its details page.
3. In the navigation pane on the left, choose **Plug-ins**.
4. Locate the plug-in to be installed and click **Install**.

   NOTE

   If Cluster Autoscaler is not manually installed in a newly created resource pool, however, it is displayed as installed in the plug-in list. This indicates that Cluster Autoscaler has been installed in the CCE cluster used by the newly created resource pool. In this case, uninstall Cluster Autoscaler and reinstall it.

   Figure 6-8 Installing a plug-in

5. In the displayed dialog box, configure the parameters.

   The following table lists the related parameters.

Table 6-11 Cluster Autoscaler parameters

Parameter	Sub-Parameter	Description
Specifications	Plug-in Version	Specify the version of Cluster Autoscaler to be deployed.
	Plug-in Specifications	Specify the specifications of the plug-in to be deployed. You can select preset specifications or customize one.

6. Read "Usage Notes" and select **I have read and understand the preceding information**.
7. Click **OK**.

Configuring Auto Scaling Policies for a Node Pool

After you install Cluster Autoscaler, you need to configure auto scaling policies for node pools.

Only pay-per-use nodes can be added.

WARNING

Configured nodes may be deleted and cannot be restored if automatic scale-in is performed. Exercise caution.

1. On the resource pool details page, choose **Node Pool Management** from the left.
2. Locate the target node pool and click **AS Configuration** in the **Operation** column.
3. In the displayed dialog box, configure the node pool scaling policy.
   - **Auto Scale-Out**

     If this function is enabled, the node pool can be automatically scaled out. Each node pool can have a maximum of six scale-out rules.

Table 6-12 Auto scale-out parameters

Parameter	Description
Custom Scale-out Rules	Click **Add Rule**. In the dialog box displayed, set the following parameters: Set **Rule Type** to **Period** or **Metric Trigger**. Each node pool can have a maximum of six scale-out rules, including five periodic scale-out rules and one metric-triggered scale-out rule. A periodic scale-out rule cannot be added repeatedly. You can add only one metric-triggered scale-out rule. For details, see **Table 6-13**.
Max. Nodes	The node pool will not be scaled out if the number of nodes hits the configured maximum. If the number of nodes in a node pool plus the expected number of nodes to be added exceeds the upper limit, the scale-out will not be triggered. This is to ensure the atomicity of scale-out.

Table 6-13 Scale-out rule types

Rule Type	Description
Periodic	Automatically adds nodes to the node pool in a specified period of time, optimizing resource allocation and reducing costs. • **Trigger Time**: Specify a time as required. This time indicates the local time of where the node is deployed. • **New Nodes**: Set the number of nodes to be added to a node pool during elastic scaling.

Rule Type	Description
Metric Trigger	Dynamically adds nodes to the node pool based on the NPU usage, improving task execution efficiency. • **Trigger**: Currently, only NPU usage-triggered scale-out is supported. When the NPU usage of a node is low, the system may migrate tasks to the node or adjust the number of nodes to better match the requirements. NPU usage = Resource requested by the pod in the node pool/Allocatable resources of the node pod (Node Allocatable) The value must be greater than the scale-in percentage configured in Autoscaler. • Action – **Customization**: Customize the number of nodes to be added during auto scaling. – **Automatic calculation**: When the trigger condition is met, nodes are automatically added and the usage is restored to a value lower than the threshold. Number of nodes to be added = Resource requested by the pod in the node pool/ (Allocatable resources of a single node x Number of target nodes) - Current number of nodes + 1

– **Auto Scale-In**

Once enabled, the system checks the resource status of the entire cluster. If it confirms that workload pods can be scheduled and run properly, it automatically chooses nodes for scale-in.

Table 6-14 Auto scale-in parameters

Parameter	Description
Min. Nodes	The node pool will not be scaled in if the number of nodes hits the configured minimum. You must set the minimum number of nodes (minCount) in the node pool for scale-in. Otherwise, the auto scale-in will fail.
Cooldown Period (Min)	The cooldown period for starting scale-in evaluation again after auto scale-out is triggered

4. Click **OK**.

Configuring Metric-Triggered Auto Scaling

When you configure an auto scaling policy for a node pool, if metric **ma_node_pool_allocate_card_util** is used as the scaling policy, you need to complete the following configurations.

1. Install the cloud native plug-in, select local storage, and enable custom metric collection. For details, see **Creating an HPA Policy with Custom Metrics**.
2. Create external APIServices and use kubectl apply to apply the configurations to the Kubernetes cluster.

   a. Log in to the **CCE console**. Go to the shell page of the cluster by clicking the CLI tool in the upper right corner.

   b. Create the e**xternal.yaml** file and save the YAML content below to the file.

   c. Run the **kubectl apply -f external.yaml** command.

```
apiVersion: apiregistration.k8s.io/v1
kind: APIService
metadata:
  labels:
    app: external-metrics-apiserver
    release: cceaddon-prometheus
  name: v1beta1.external.metrics.k8s.io
spec:
  group: external.metrics.k8s.io
  groupPriorityMinimum: 100
  insecureSkipTLSVerify: true
  service:
    name: custom-metrics-apiserver
    namespace: monitoring
    port: 443
  version: v1beta1
  versionPriority: 100
```

3. Add custom external metrics of the Prometheus plug-in. For details, see **Step 3: Modify the Configuration File**.

   a. Log in to the **CCE console** and click the cluster name to access its details page. In the navigation pane on the left, choose **ConfigMaps and Secrets** and switch to the **monitoring** namespace.

   b. Update **user-adapter-config**. You can modify the **rules** field in **user-adapter-config** to convert the metrics exposed by Prometheus to metrics that can be associated with HPA.

      Add the following example rules:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-adapter-config
  namespace: monitoring
data:
  config.yaml: |
    rules: []
    ...
#The following content is the added content.
    externalRules:
    - seriesQuery: '{__name__="ma_node_allocate_card_util",pool_id!=""}'
      metricsQuery: avg(<<.Series>>{<<.LabelMatchers>>}) by (pool_id,node_pool)
      resources:
        overrides:
          pool_id:
            resource: namespace
      name:
        as: ma_node_pool_allocate_card_util
```


4. On the CCE console, choose **Clusters** from the navigation pane.
5. Click the cluster name. Then, in the navigation pane on the left, choose **Workload**. Switch to the **monitoring** namespace. Locate the **custom-metrics-apiserver** instance and choose **More** > **Redeploy** next to the workload.
6. After the redeployment is complete, you can use the CLI tool on the CCE console to view the current metric values. The command is shown below. In the command, **pool_id** indicates the resource pool ID, and **node_pool** indicates the node pool name. When querying the default node pool, leave this parameter blank.

```
kubectl get --raw /apis/external.metrics.k8s.io/v1beta1/namespaces/{{pool_id}}/
ma_node_pool_allocate_card_util?labelSelector=node_pool={{node_pool_name}}
```

Components

Table 6-15 Nodescaler component of Cluster Autoscaler

Component	Description	Resource Type
nodescaler-controller-manager	Manage auto scaling of resource pools.	Deployment

Related Operations

For details, see **Viewing the Lite Cluster Plug-in on the Resource Pool Details Page**.

Change History

Table 6-16 Release history

Plug-in Version	New Feature
0.1.20	Supported auto scale-out at a scheduled time, scale-out based on the NPU allocation rate, and auto scale-in based on the load of idle nodes.

