DLRover

蚂蚁智能大规模分布式训练系统

开源地址： https://github.com/intelligent-machine-learning/dlrover

DLRover
Deep Learning Rover

目录

CONTENTS

DLRover
Deep Learning Rover

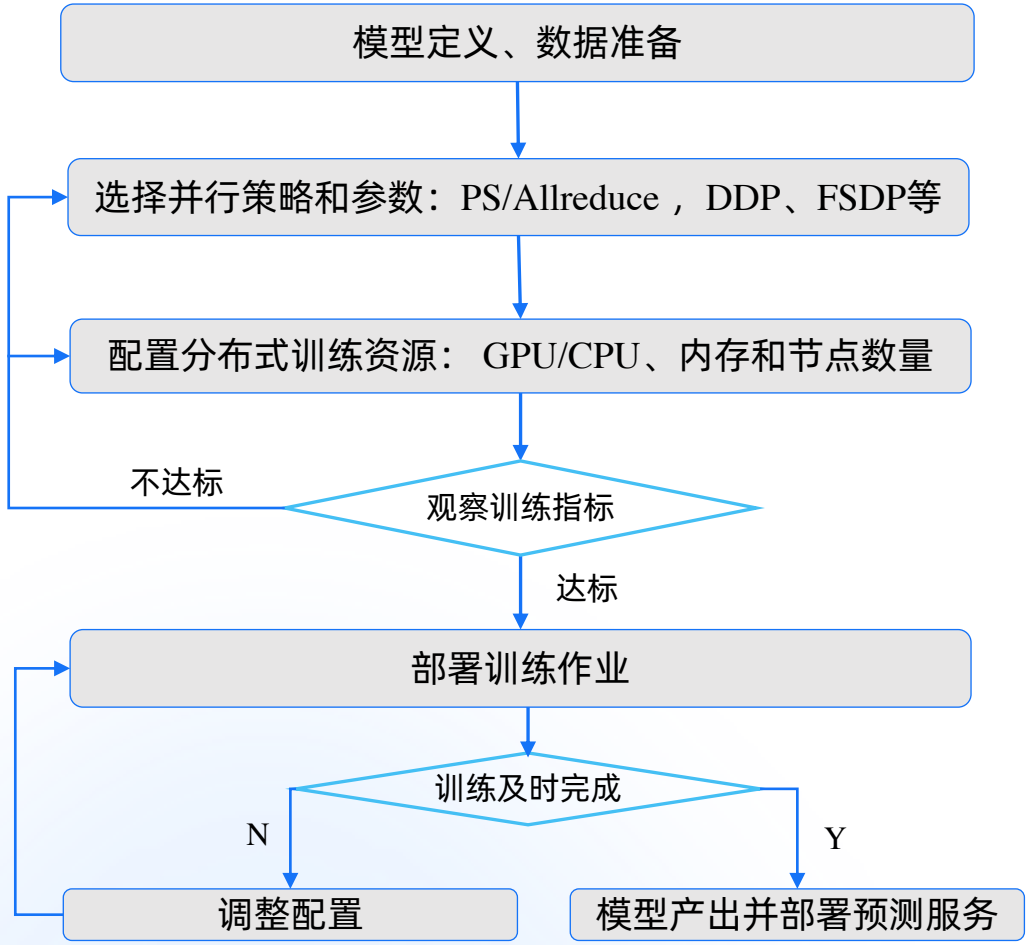
01. 项目愿景

深度学习领域的探月车，推进分布式训练的自动化

大规模分布式训练面临的挑战

DLRover
Deep Learning Rover

大模型分布式训练开发流程



大模型分布式训练开发的难点

不同模型训练配置不同的并行策略：

- 并行策略多，DP、TP、PP等。
- 高效的 FSDP 配置难，配置选项多，模型分片难。

资源配置不合理：

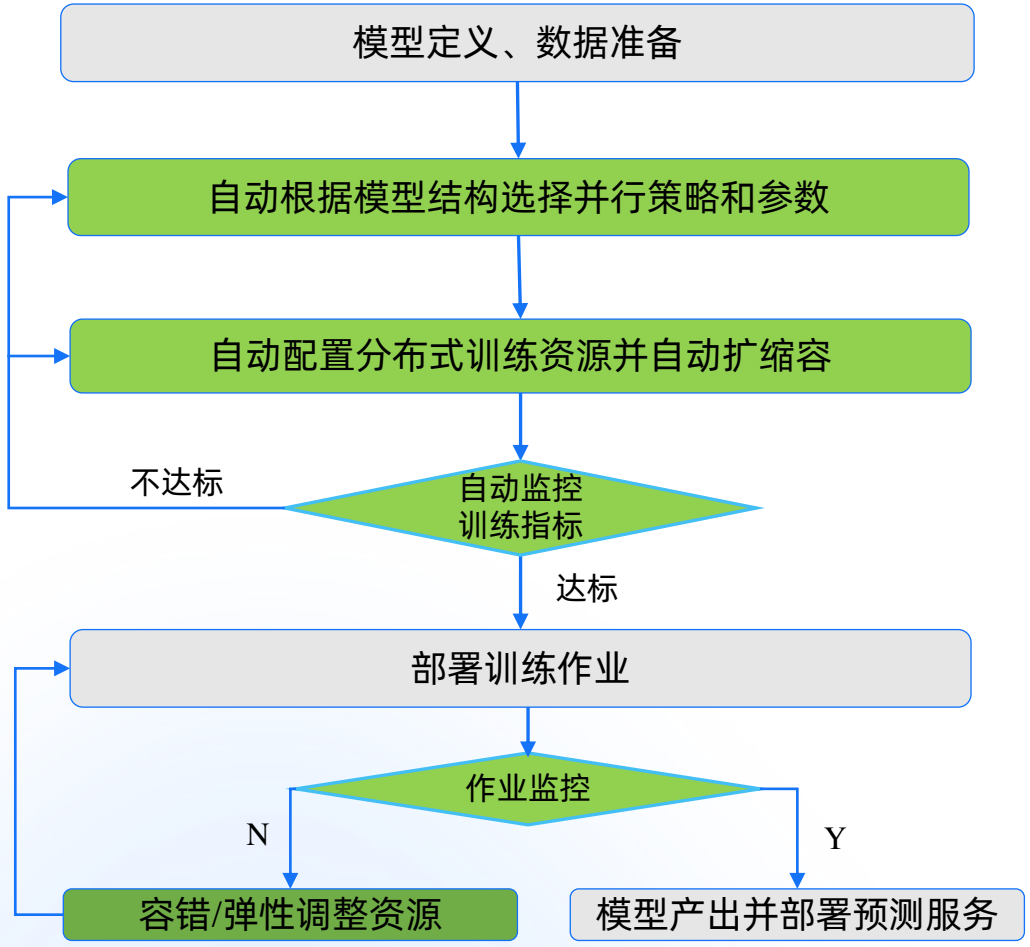
- 资源配置后不可更改，需要反复实验训练最优配置。
- 资源配置过少，资源成为瓶颈导致训练速度慢。
- 资源配置过多，训练资源利用率低下，训练成本高。

稳定性不足：

- 节点出错导致训练中断，A100卡的故障率在 2%-5%.
- 节点故障排查不及时，作业重启然后失败。
- 节点可能被更高优的作业抢占导致训练挂起。
- 资源不够导致长时间等待，不能动态利用集群空闲资源。

DLRover 的目标

大模型分布式训练开发流程



DLRover 智能训练服务

目标：用户提供模型定义和数据，DLRover 自适应地分布式训练模型。

自动并行：

- 分析模型结构选择合适的并行方式。
- 根据训练运行的profiling 动态调整训练配置。

自动资源配置：

- 训练启动前根据模型、数据和场景自动预估训练资源。
- 训练过程中采集资源负载和训练性能来动态预估资源。
- 训练过程中通过自动扩缩容来动态调整训练资源。

提高训练的稳定性：

- 自动容错，训练可以在部分节点故障后继续进行。
- 自动故障检测，及时发现故障机并将其隔离。
- 弹性训练，集群有空闲资源时可以增加训练数量。
- 弹性训练，节点资源被抢占后能缩减训练节点数量。

DLRover
Deep Learning Rover

02. 系统设计

DLRover 智能分布式训练系统架构

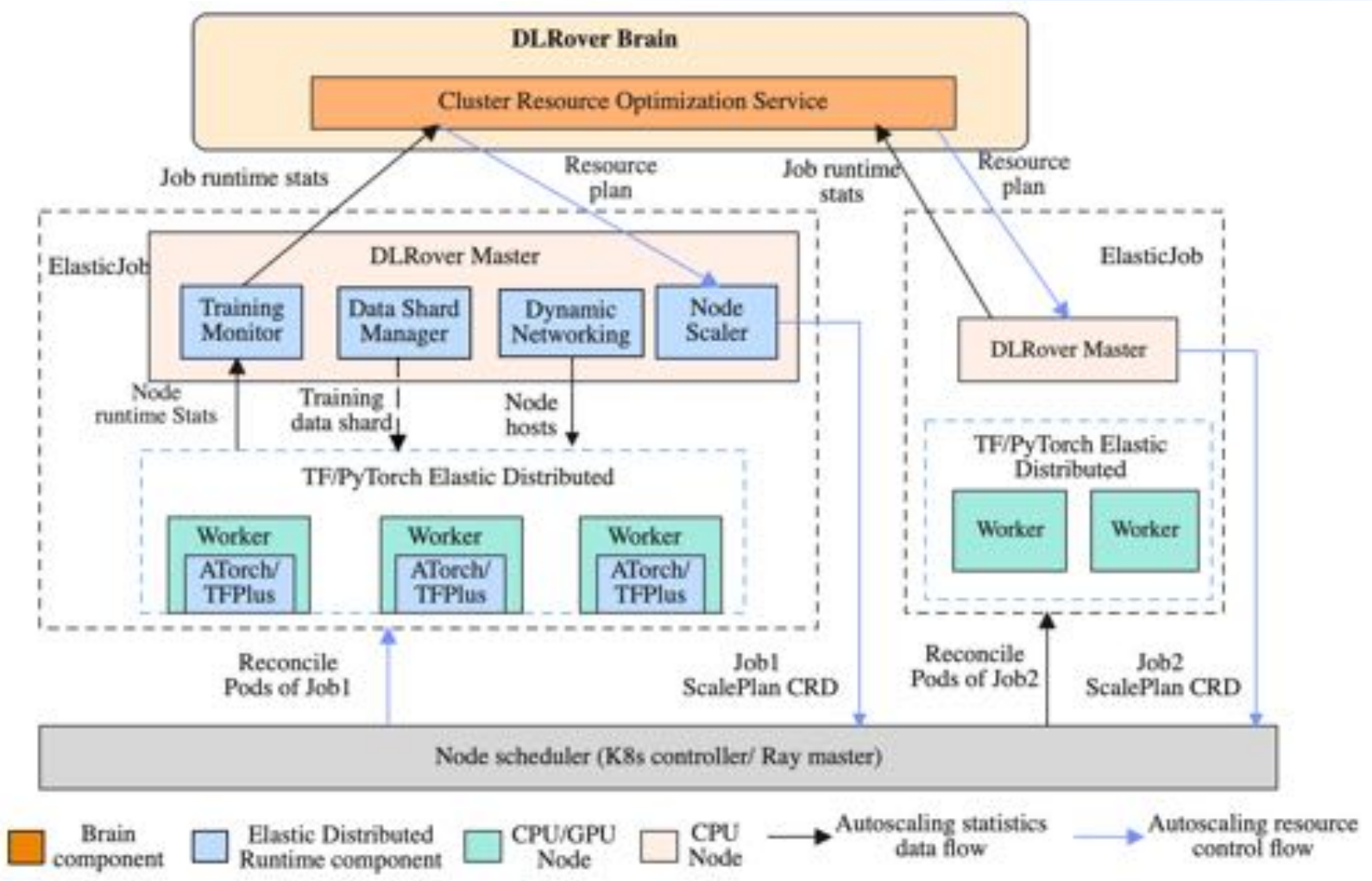
DLRover 系统设计

DLRover 架构分为三层：

DLRover Brain：
- 集群资源优化算法策略。
- Job 资源优化算法策略。
- 集群与作业状态监控。

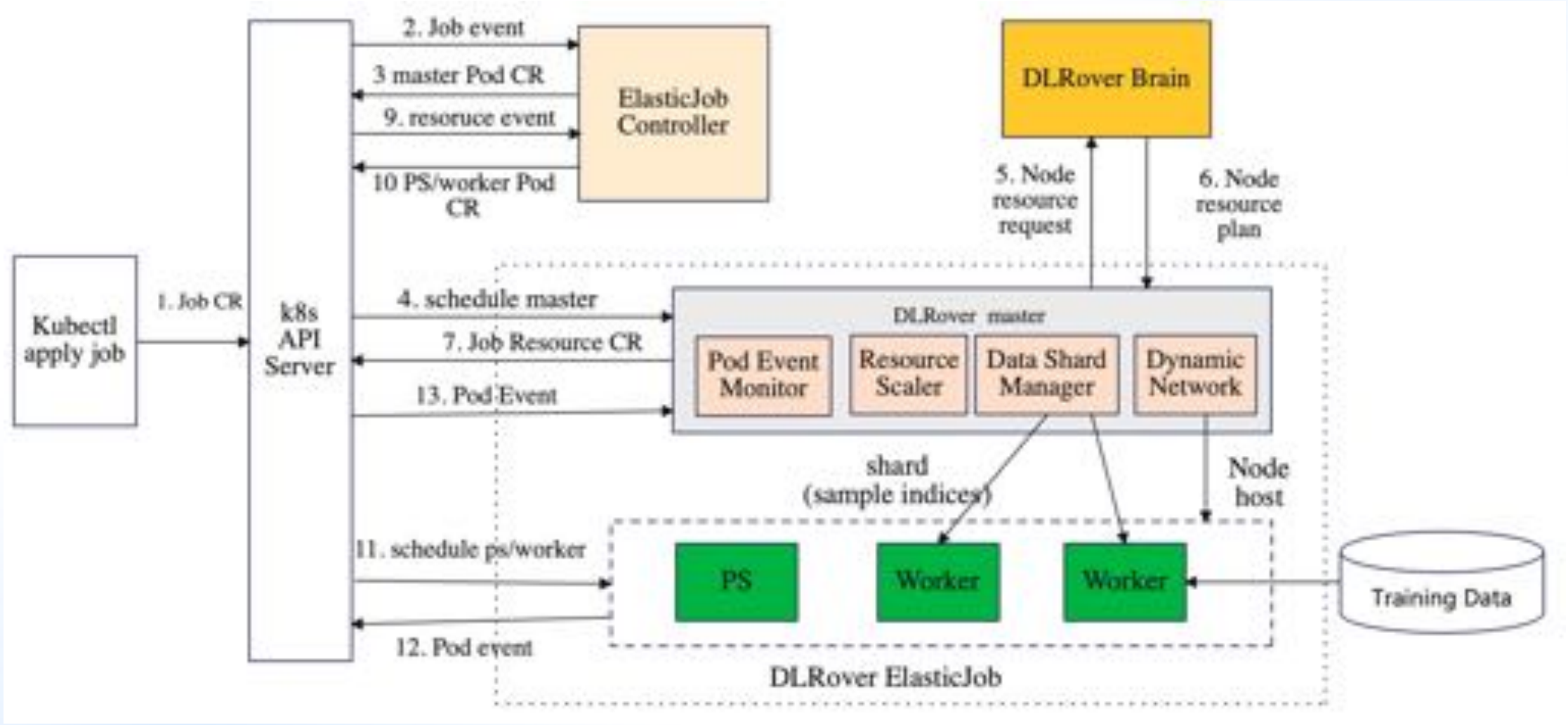
训练节点 Job Master：
- 节点状态和训练性能监控。
- 容错，自动故障检测与恢复。
- 动态扩缩容，增减作业节点数量。
- 动态组网管理，编排所有节点地址。
- 训练样本动态分发。

训练框架加速包 Atorch/TFPlus：
- 动态组网，动态伸缩训练节点。
- 训练优化，IO优化、存储优化等。
- 分布式训练自动并行策略。



DLRover 系统设计

K8s 上提交一个 DLRover 弹性训练作业ElasticJob 的流程：



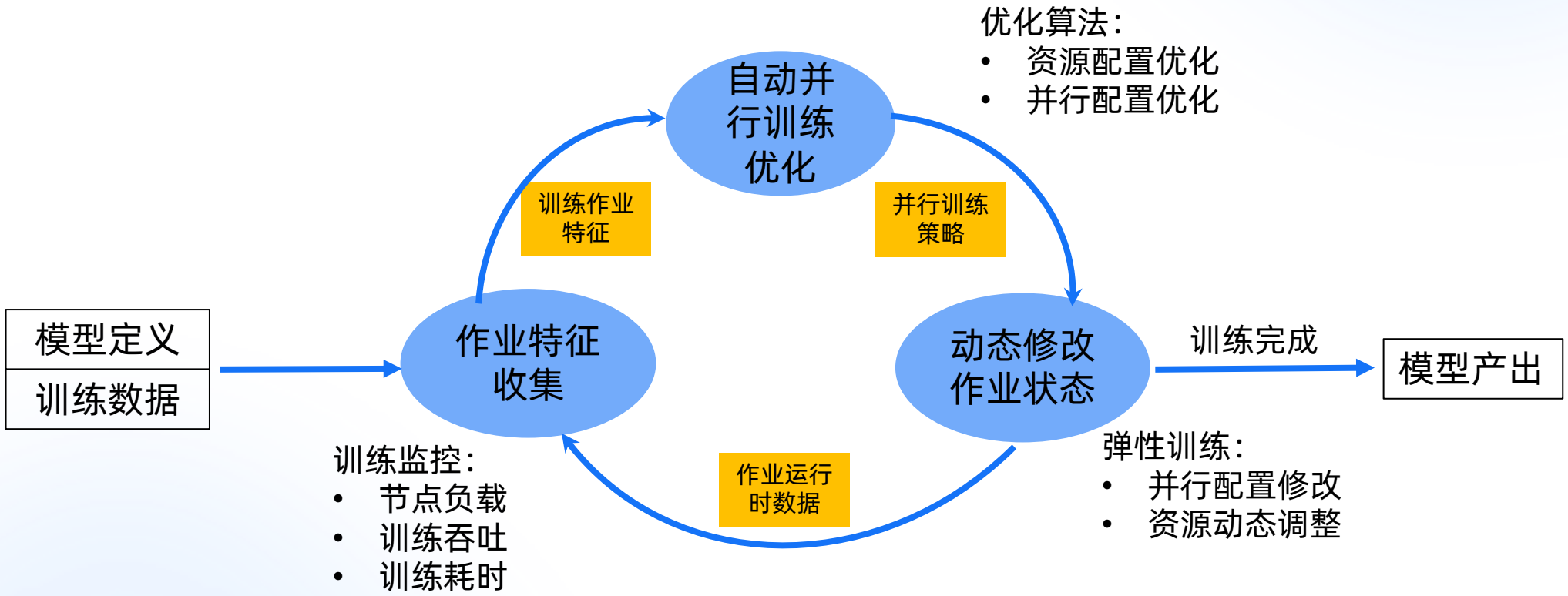
DLRover
Deep Learning Rover

03. 智能运行训练

基于运行时优化策略的智能训练系统

DLRover 智能运行训练

思路：DLRover 采取运行时动态优化方式，在不停止训练作业的情况下动态优化分布式训练。



DLRover 动态优化分布式训练

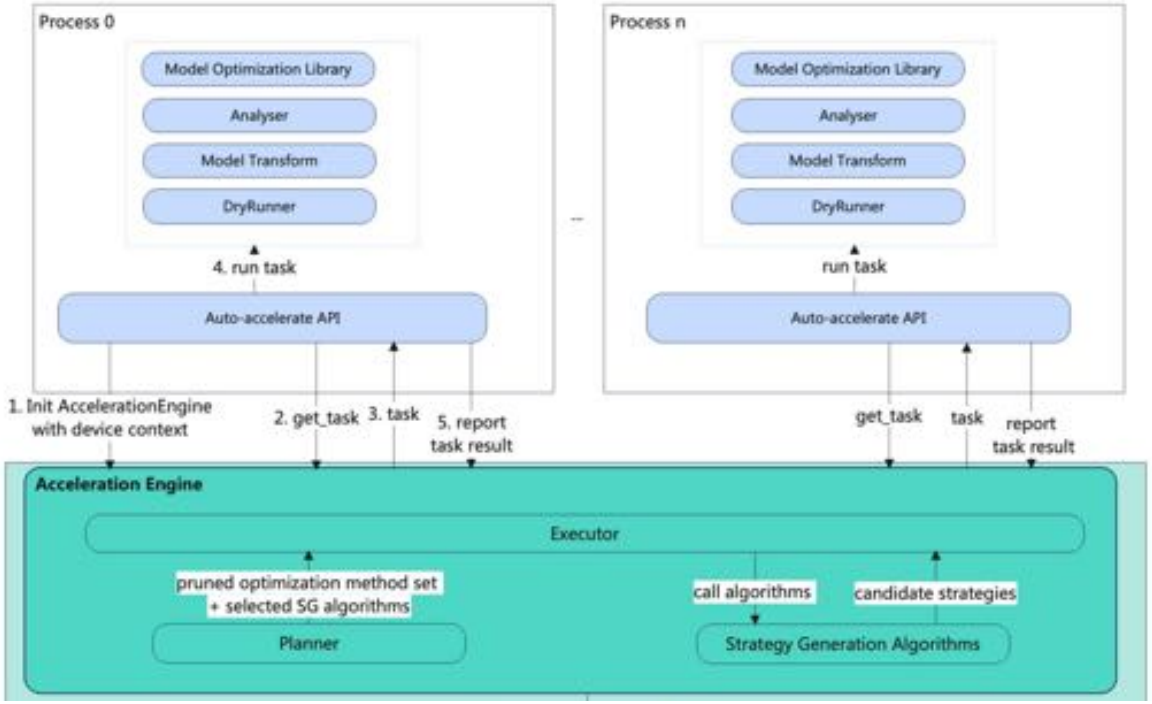
DLRover 智能运行训练

训练框架层的自动化-- PyTorch 模型的自动并行训练框架 Atorch

auto_accelerate：用户提供单机模型定义，Atorch 自动搜索多种优化策略寻找最优的分布式优化策略。自动寻优分布式配置。，例如：

- DP/TP/PP 的并行度和配置。
- FSDP 的显存配置。
- 混合精度配置。

```
status, result, best_strategy = auto_accelerate(
    model,
    torch.optim.AdamW,
    train_dataset,
    loss_func=my_loss_func,
    prepare_input=prepare_input,
    model_input_format="unpack_dict",
    optim_args={"lr": args.learning_rate},
    optim_param_func=partial(optim_param_func, args=args),
    dataloader_args=dataloader_args,
)
```

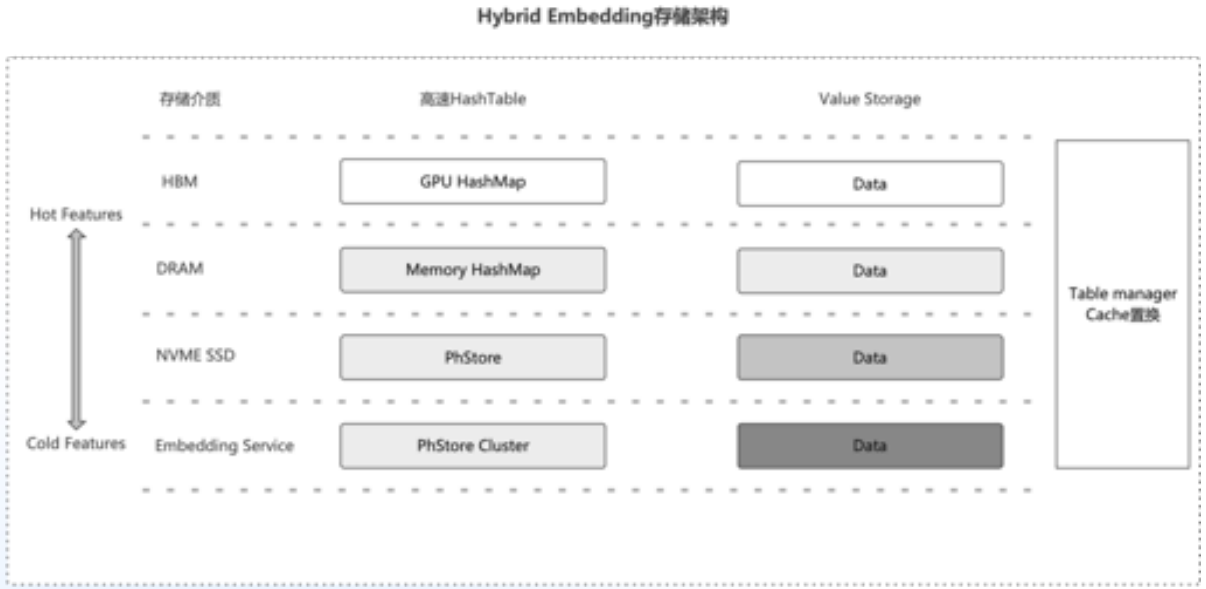


DLRover 智能运行训练

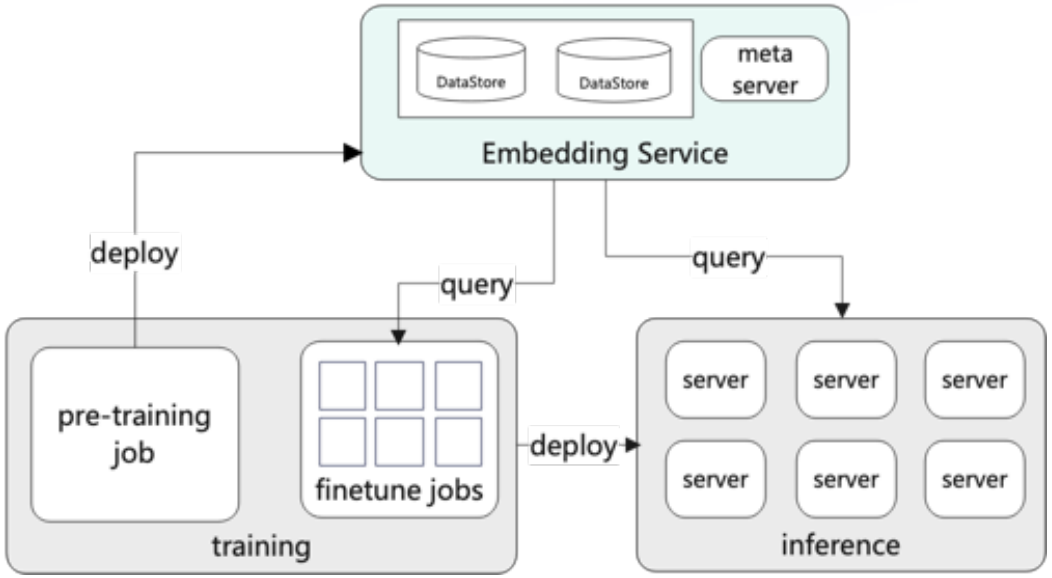
DLRover
Deep Learning Rover

训练框架层的自动化--TF 搜推广模型embedding 的自动优化库 TFplus

Hybrid Embedding：根据 item 频次自动选择存储介质，保证训练速度的前提下减少 DRAM 的使用。



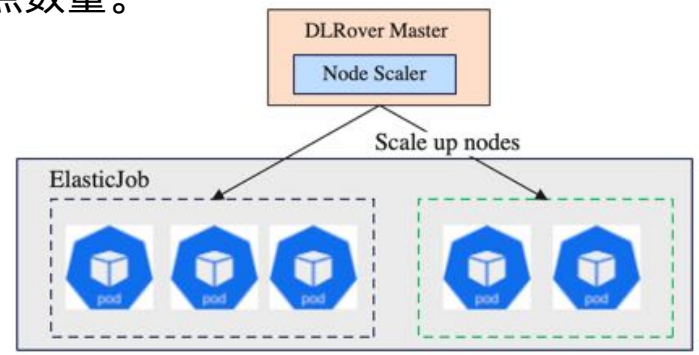
Embedding Service：独立的服务嵌入到模型中，供模型在线训练和推理使用，避免数据冗余和资源浪费。



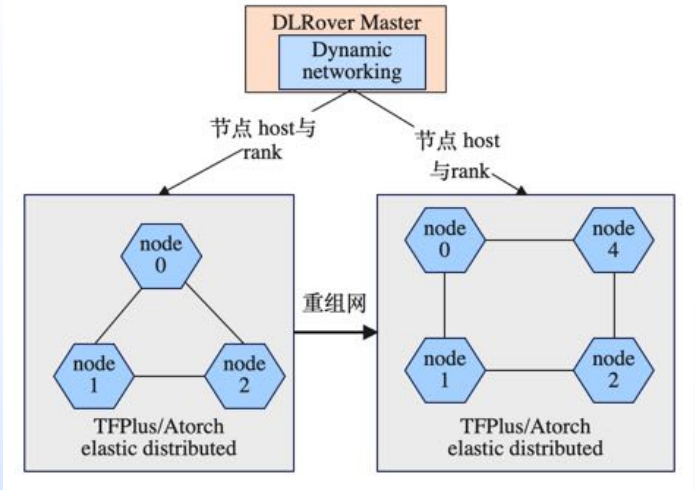
DLRover 智能运行训练

训练调度的自动化--弹性训练

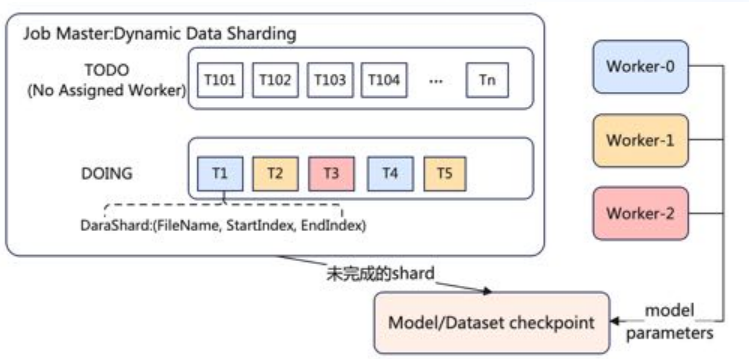
弹性调度：作业运行时，Job master可以动态伸缩节点数量。



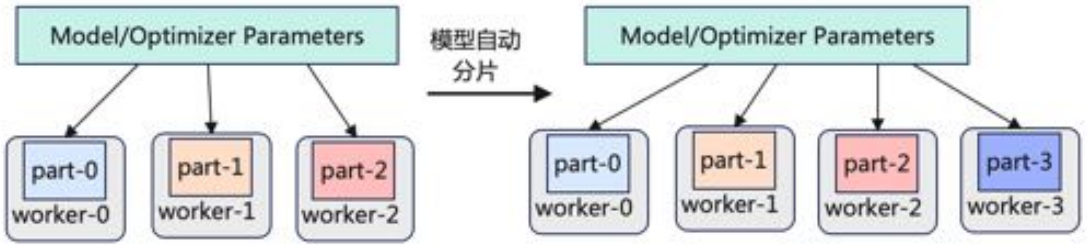
动态组网：节点数量变化后， Job master 可以协助TFplus/Atorch 能重新组成训练网络。



训练状态恢复：调整训练配置后，Tfplus/Atorch 可以快速回复模型参数和数据消费进度。



自动模型分片：节点数量变化后，TFplus/Atorch 可以自动将模型根据节点数量分片。



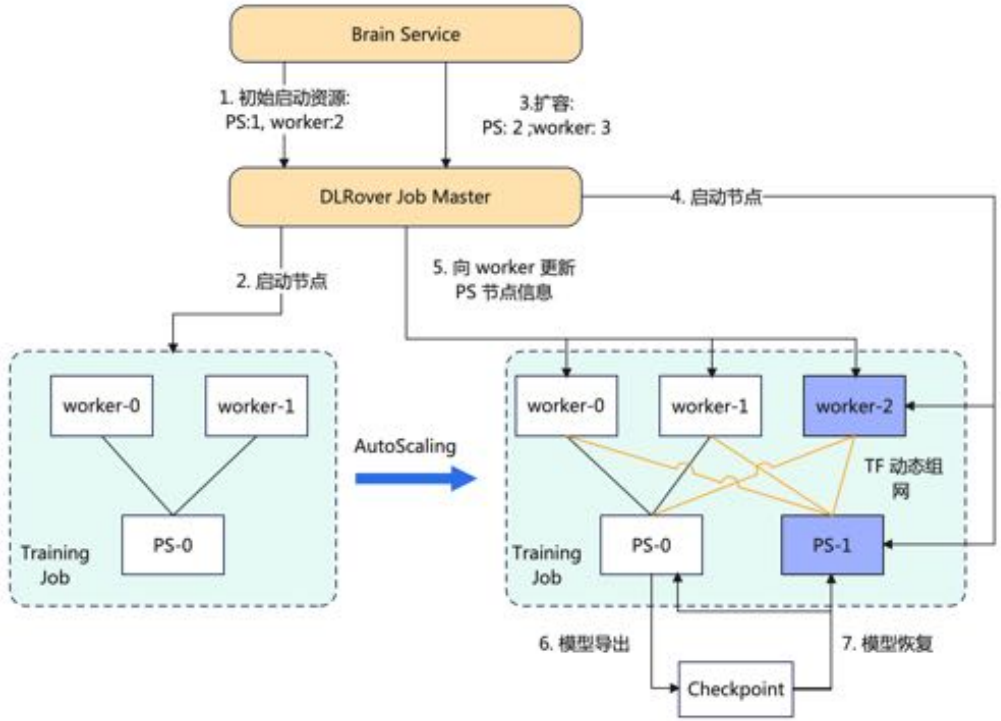
TFPlus：支持 embedding 的自动模型分片。
Atorch：支持 FSDP 并行策略的自动模型分片。

DLRover 智能运行训练

训练调度自动化--自动资源配置与扩缩容

提交作业时用户无需指定节点资源，DLRover 根据用户和模型画像自动配置初始资源，且在训练过程中自动伸缩节点数量，提升训练速度和资源利用率。

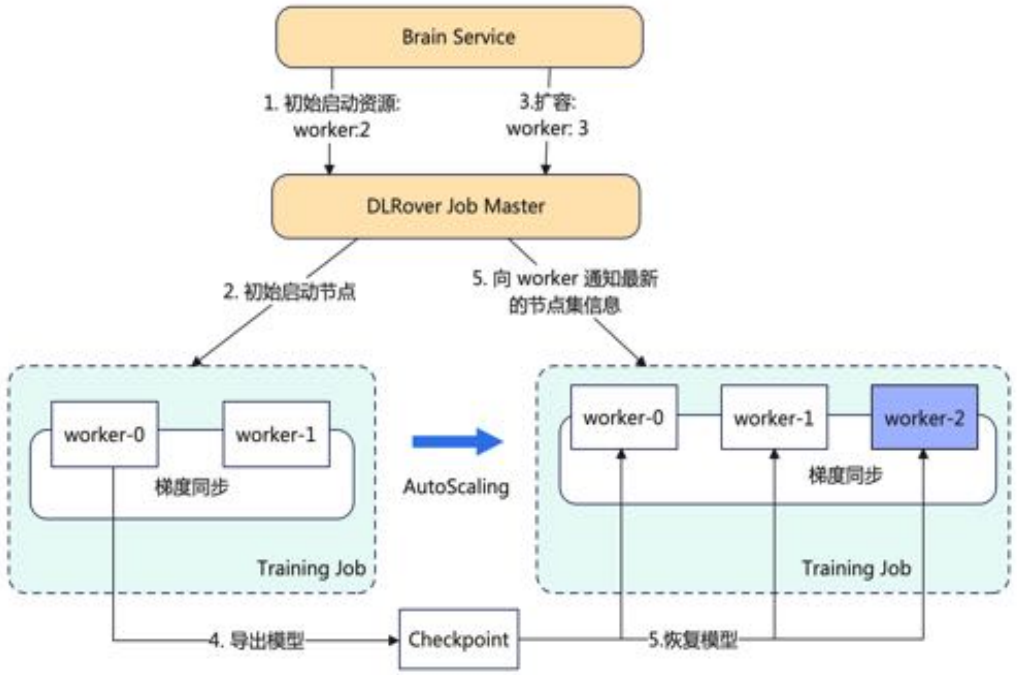
TensorFlow 的 PS 架构的自动扩缩容



✓ Worker的扩缩容无需保存 checkpoint。

✓ PS 的扩缩容前，master通知 PS 导出checkpoint。

PyTorch 的 allreduce 架构的自动扩缩容



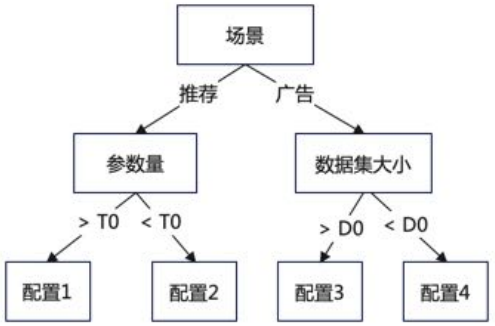
✓ 扩缩容前，master 通知 worker 到处 checkpoint。

DLRover 智能运行训练

DLRover Deep Learning Rover

训练调度自动化--自动资源配置与扩缩容算法：

启动阶段 → 运行初始阶段 → 运行阶段



$$\text{maximize } R$$

$$\text{subject to}: \sum_{i=0}^{N_w} w_i + \sum_{j=0}^{N_{ps}} s_i < C_{total}$$

$$N_w \cdot \hat{s} < \sum_{j=0}^{N_{ps}} s_i$$

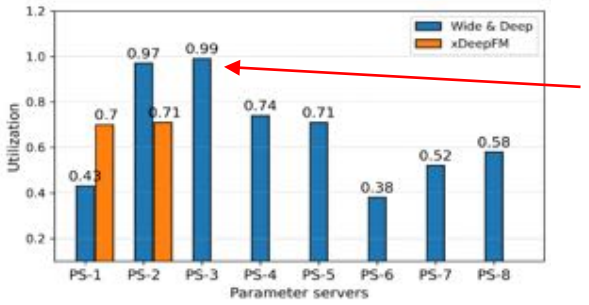
$$w_i < C_{max}, s_j < C_{max}$$

$$N_w \in Z^+, N_{ps} \in Z^+, w_i \in Z^+, s_j \in Z^+$$

调整后的速度增幅　worker增加的数量

$$s_{k,d} = \frac{\Delta R / \Delta k}{r_k}$$

调整的节点平均吞吐



输入：用户信息、数据集信息和模型信息。
算法：基于用户和模型画像预测初始资源。
✓ 历史作业建立训练作业画像数据。

输入：节点负载、资源上限。
算法：利用运筹规划资源配置。
✓ 目标最大化训练吞吐。

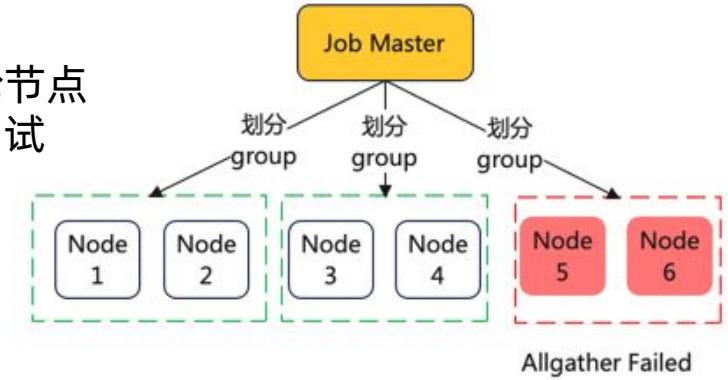
输入：节点负载、训练吞吐。
算法：启发式资源动态优化
✓ 增加节点数量观察加速比。
✓ PS热点增加 CPU 资源。

DLRover 智能运行训练

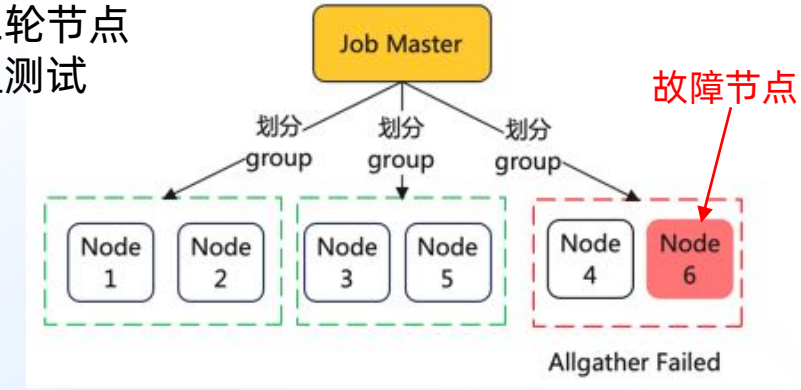
训练容错自动化：自动节点故障检测，保障节点出错后训练能快速恢复。

自动故障机检测：

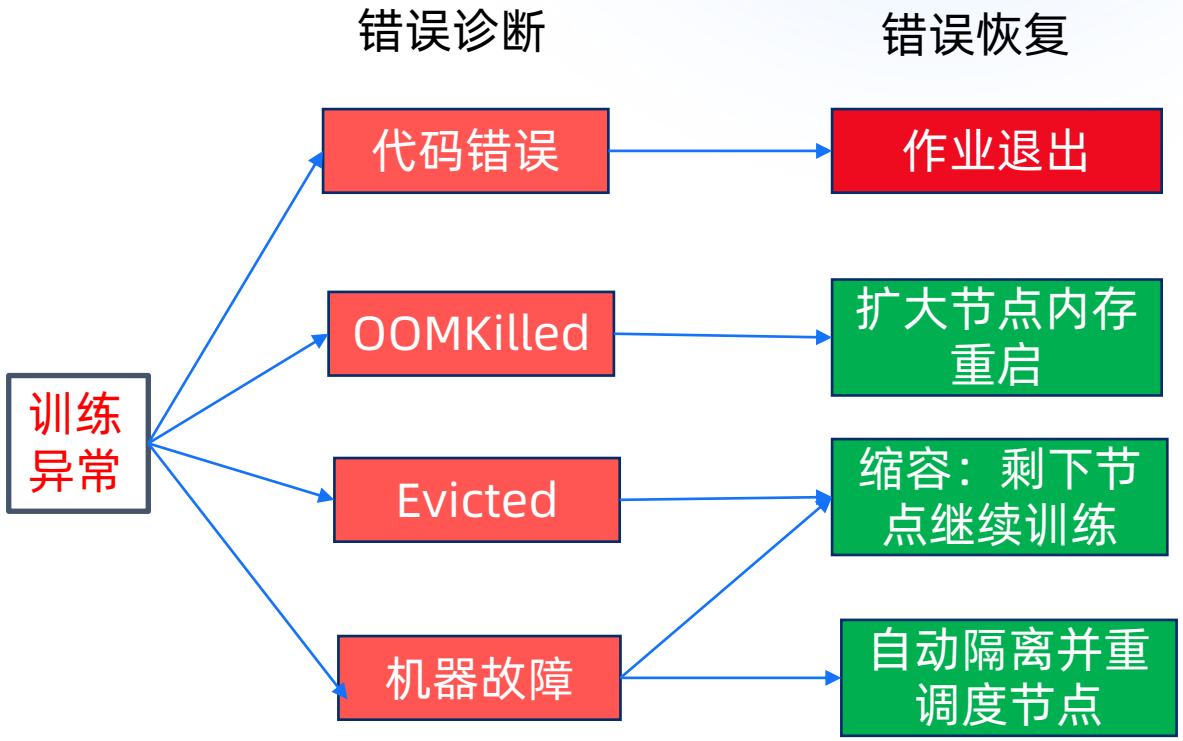
第一轮节点分组测试



第二轮节点分组测试

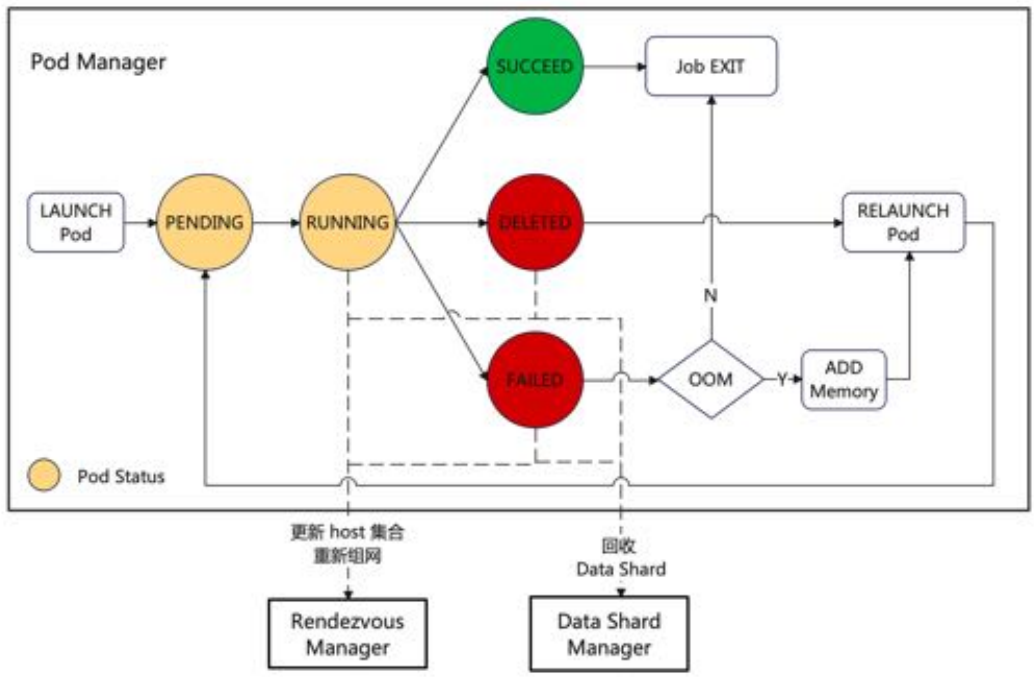


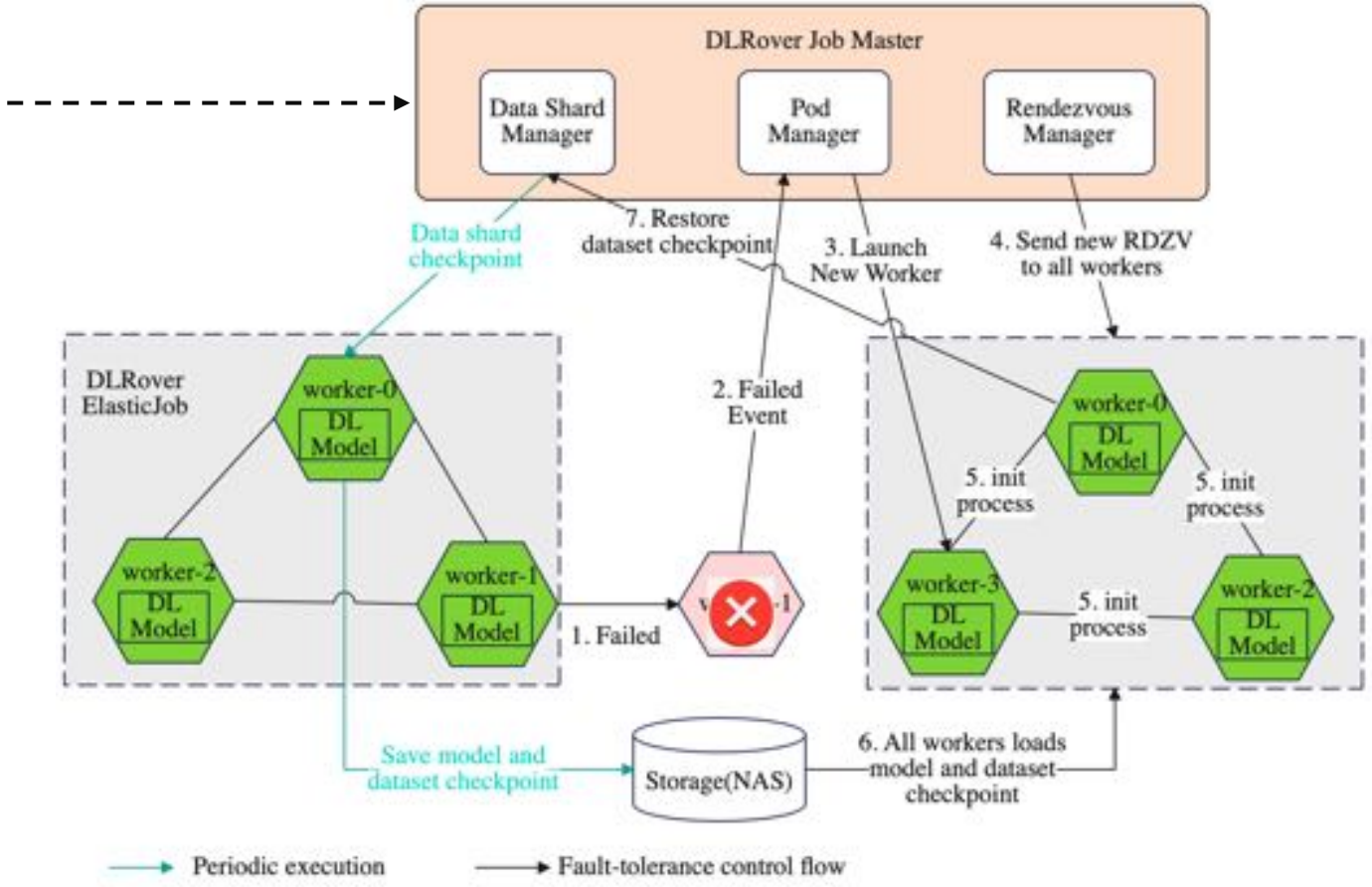
训练自动容错：



DLRover 智能运行训练

训练容错自动化：DLRover ElasticJob 的容错流程。





Job Master 容错事件处理流程

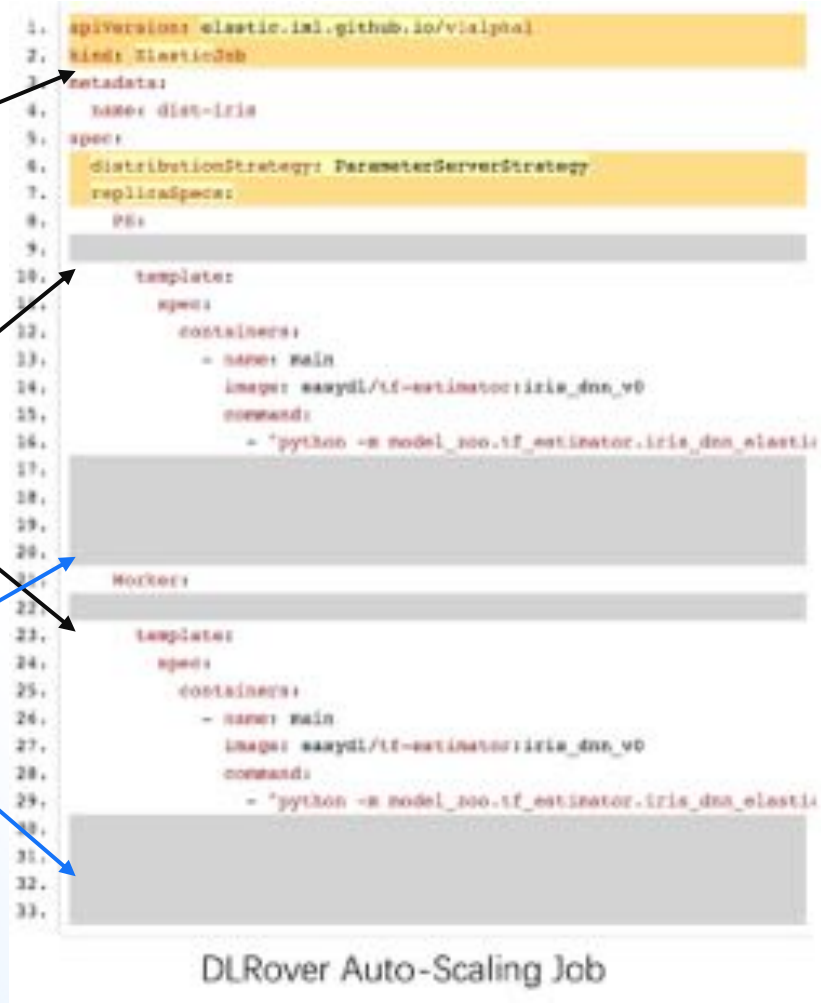
DLRover 智能运行训练

DLRover 自动资源配置与调整作业示例

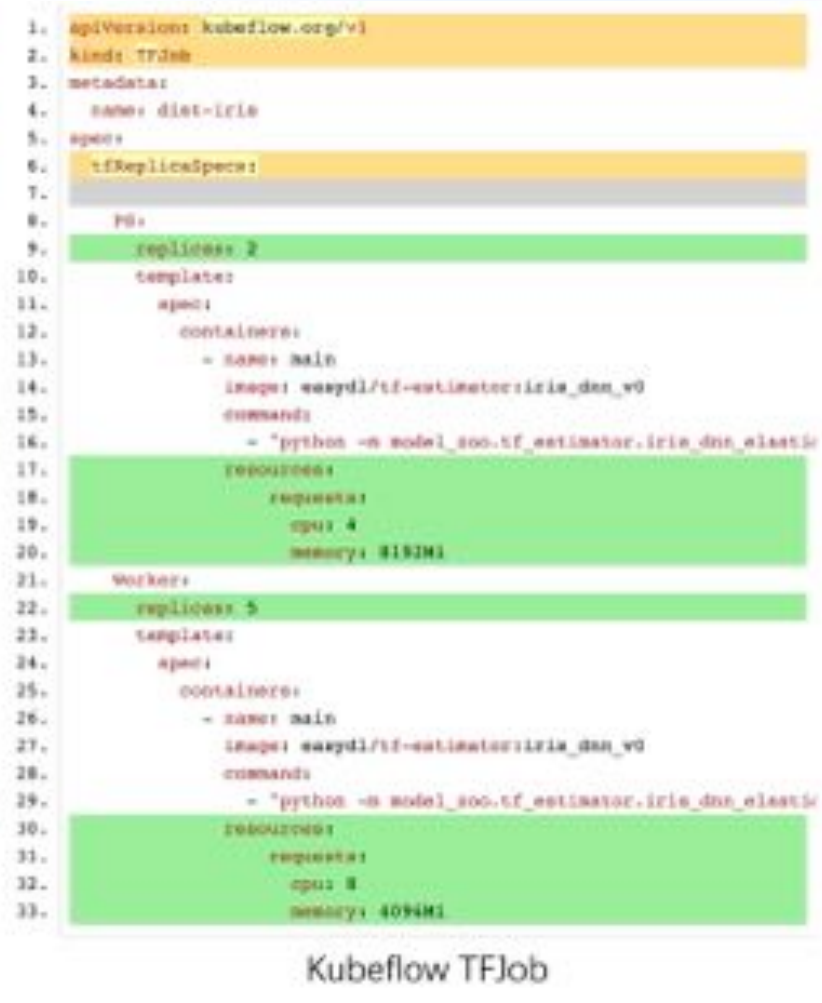
k8s 部署 DLRover ElasticJob CRD 后提交 ElasticJob yaml。

无需指定节点数量

无需指定节点资源规模



DLRover Auto-Scaling Job



Kubeflow TFJob

DLRover 智能运行训练

DLRover 自动资源配置与调整作业示例

模型：xDeepFM，数据集 CRITEO
集群：阿里云 ACK

```
NAME                                    READY   STATUS    RESTARTS   AGE
dlrover-auto-scale-edljob-chief-0       1/1     Running   0          32s
dlrover-auto-scale-edljob-ps-0          1/1     Running   0          32s
elasticjob-torch-mnist-dlrover-master   1/1     Running   0          39s
```

速度 30 steps/s

自动扩容

```
NAME                                    READY   STATUS    RESTARTS   AGE
dlrover-auto-scale-edljob-chief-0       1/1     Running   0          6m17s
dlrover-auto-scale-edljob-ps-0          1/1     Running   0          6m17s
dlrover-auto-scale-edljob-worker-0      1/1     Running   0          3m19s
dlrover-auto-scale-edljob-worker-1      1/1     Running   0          3m19s
dlrover-auto-scale-edljob-worker-2      1/1     Running   0          3m19s
```

速度 100 steps/s

DLRover 智能运行训练

DLRover 容错作业示例：

1. 作业启动

```
NAME                                        READY   STATUS    RESTARTS   AGE
elasticjob-torch-mnist-dlrover-master       1/1     Running   0          26s
torch-mnist-edljob-worker-0                 1/1     Running   0          29s
torch-mnist-edljob-worker-1                 1/1     Running   0          32s
```

2. 杀掉worker-1

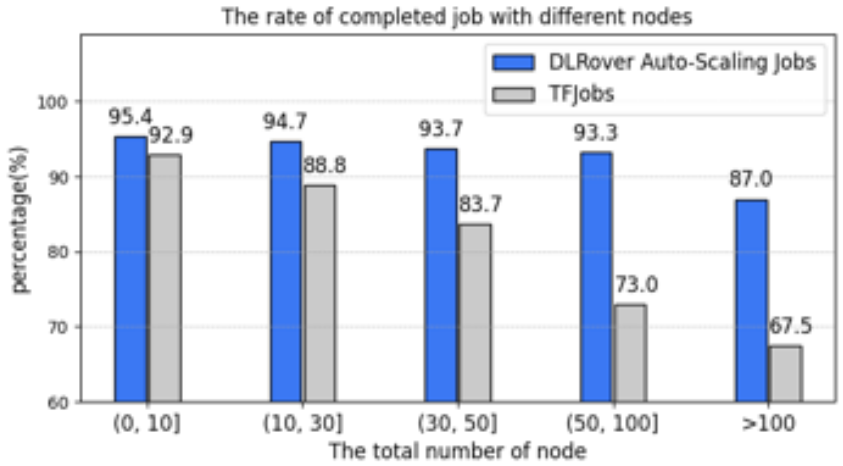
```
NAME                                        READY   STATUS    RESTARTS   AGE
elasticjob-torch-mnist-dlrover-master       1/1     Running   0          1m12s
torch-mnist-edljob-worker-0                 1/1     Running   0          1m15s
```

3. 恢复 worker-1

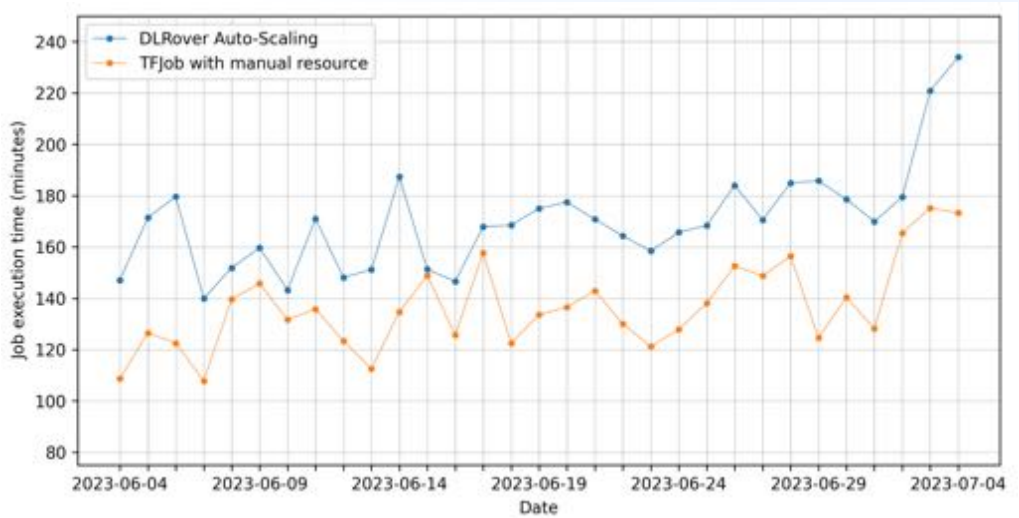
```
NAME                                        READY   STATUS    RESTARTS   AGE
elasticjob-torch-mnist-dlrover-master       1/1     Running   0          1m52s
torch-mnist-edljob-worker-0                 1/1     Running   0          1m55s
torch-mnist-edljob-worker-1                 1/1     Running   0          32s
```

DLRover 智能运行训练

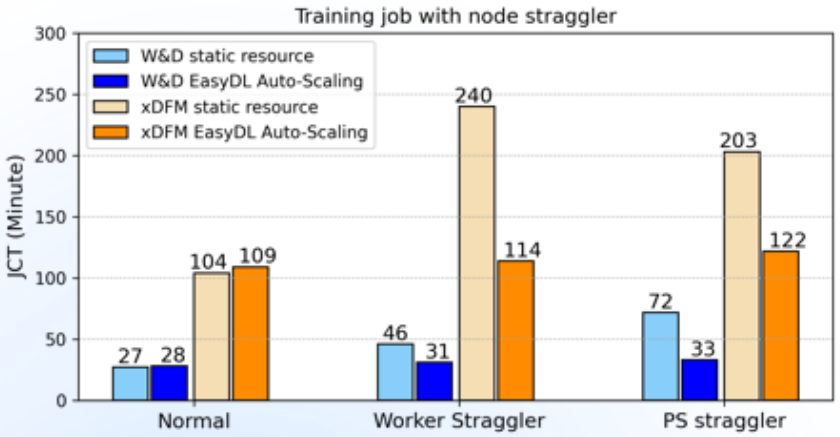
蚂蚁混布集群上，DLRover 训练作业的应用效果：



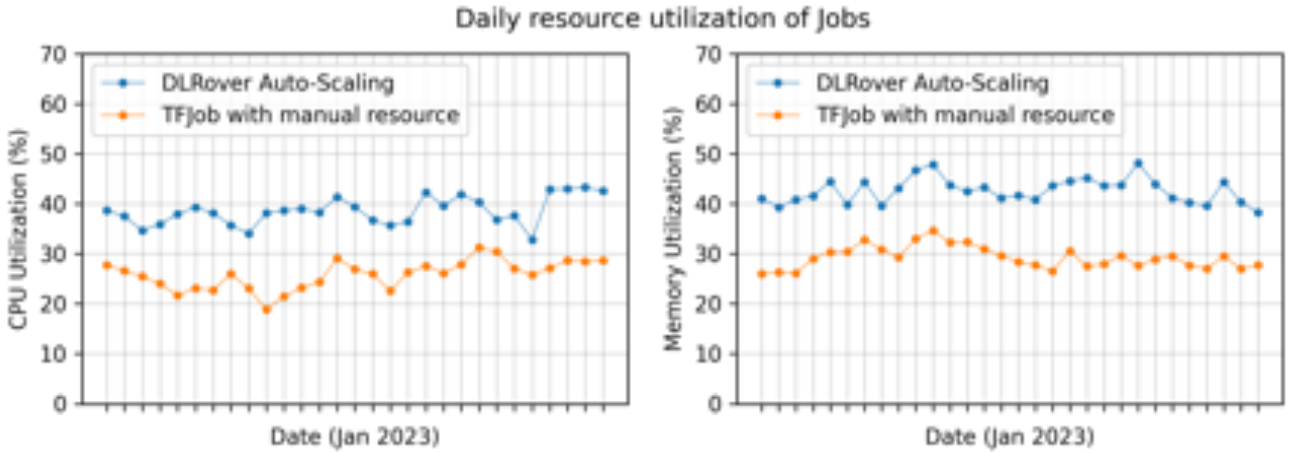
提升作业成功率



缩短训练时间



缓解慢节点拖累训练



提升集群资源利用率

DLRover
Deep Learning Rover

04. 未来规划

提 升 训 练 自 动 化 与 智 能 化

DLRover 未来规划

继续推进分布式训练的自动化与智能化，提高分布式训练效率，降低分布式训练门槛。

训练自动调优：

- 训练超参的自动化调整：自动调整 learning rate/ batch size 防止 loss 跑飞。
- FSDP 的自动调优：探索算法自动根据 GPU 节点数、显存来调整 FSDP 的配置从而达到最优的训练性能。
- 自动并行训练：根据模型自动的指定合适的并行策略，充分利用硬件资源提升训练效率。

智能高效的训练容错：

- 智能错误检测：及时且快速地检测故障机，降低机器或者网络故障导致的训练停止时间。
- 高效的训练恢复：更快地导出和加载 checkpoint，从而支持更加高频的checkpoint，缩短训练回滚间隔。

智能高效的弹性训练：

- 弹性训练覆盖更多分布式策略：支持 FSDP、TP 等分布式训练模式的弹性扩缩容。
- 自动资源配置：为大模型训练作业提供自动资源配置策略。
- 智能扩缩容训练：根据集群负载自动伸缩训练作业规模，充分利用集群资源提升训练性能。

已发布版本：

- V0.1.0: k8s/ray 上 TensorFlow PS 的弹性容错和自动扩缩容。
- V0.2.0: k8s 上 PyTorch AllReduce 的故障检测和弹性容错。

THANKS

王勤龙