PR #2355 完整报告

radixark/miles

[fix] fix the bugs/outdated commands in `.sh` scripts and the corresponding snapshots

合并时间：2026-08-12 10:12

原文链接：http://prhub.com.cn/radixark/miles/pull/2355

执行摘要

- 一句话：修复启动脚本批量 bug 并同步快照与测试 harness
- 推荐动作：值得精读。三个看点：一是 before_ray_job_submit 钩子如何在共享执行器中处理“集群就绪后”的准备工作，这是对多节点训练启动时序的通用解法；二是 os.environ.get(name, default) 默认值急切求值这个隐蔽坑；三是测试 harness 保存 / 恢复环境以消除顺序依赖的通用做法。建议与 #2354、#2356 连起来读，理解整个脚本清理与重写脉络，并关注 #2356 的 argv 对比是否真的发现移植漂移。

功能与动机

PR body 开门见山：这批 launcher 每一个都“cannot do what it says”——声称能做某事却实际跑不通。更关键的是，快照套件（#1899/#1901）把坏掉的输出固化成了期望输出，而不是标记错误：“The snapshot suite added in #1899/#1901 recorded the broken output rather than flagging it, so the fixes are visible here as snapshot diffs — one diff per fix, nothing else.” 清理必须先于接下来的 Python 重写（#2356），因为 argv 对比必须基于已经能正常工作的 launcher：“the argv comparison in the next PR is against launchers that already work, so a difference there means the port drifted, not that it inherited a bug.”

实现拆解

1. shell launcher 直接错误修复

逐个点名修复：run-qwen3-next-80B-A3B.sh 的 --runtime-env-json 在 NCCL_NVLS_ENABLE 后漏了逗号，ray 收到无效 JSON；run-gpt-oss-20b-bf16.sh 读取 ${HAS_NVLINK} 却从不探测 NVLink，导致向 NCCL 传入空值；run-qwen3-4B-base-sft.sh、run-qwen3-235B-A22B-sft.sh、run-gpt-oss-20b-bf16.sh 展开从未声明的 ${EVAL_ARGS[@]}，脚本一执行就退出；run-kimi-k25.sh 传入 miles 中已不存在的 --filter-zero-reward-samples；run-glm4.5-355B-A32B.sh 把 rollout 数据 dump 到个人绝对路径，并把 NVLink 探测结果写进从未读取的变量。

2. Python launcher 静默失效修复

run_glm47_flash.py 的 --eval-interval 被注释掉，而该 flag 默认是 None，导致它构建的 eval 块从未运行，本 PR 恢复 --eval-interval 20。run_qwen3_30b_a3b.py 与 amd/run_qwen3_30b_a3b.py 在调用 U.execute_train 时漏传 config=args，导致

cuda_core_dump 与 extra_env_vars 被静默丢弃，本 PR 补回。run_glm45_355b_a32b.py 的 hardware 默认值是 H100——恰好是 _execute_train 断言拒绝的值，快照 harness 不得不覆盖它才能运行脚本，现改为 GB200 后该覆盖被移除。

3. ray 生命周期重构（最大变更）

run_deepseek.py 与 run_deepseek_v32.py 的链式 prepare 流程原本是：_prepare_download → _prepare_bf16_ckpt → _prepare_megatron_ckpt → _prepare_cp → _execute_train。但 _prepare_megatron_ckpt 与 _prepare_cp 通过 exec_command_multi_node fan out，其第一步就是 ray.init(address="auto")——在干净主机上会死在 launcher 自己执行 ray start --head 之前；手动起集群也不行，因为 execute_train 的 preamble 先 ray stop --force 再拉全新 head，造成多节点转换之后跟单节点训练错配。修复方式是为 U.execute_train 增加 before_ray_job_submit 回调参数，在 ray 集群就绪、ray job submit 之前执行这些步骤：run_deepseek.py 的 train 通过 functools.partial 注入 _prepare_ray_dependent，run_deepseek_v32.py 的 full_train 注入 _prepare_megatron_ckpt。

4. 共享执行器与测试 harness 修复

miles/utils/external_utils/command_utils.py 中 os.environ.get("NCCL_NVLS_ENABLE", str(int(check_has_nvlink()))) 的默认值会被急切求值：即使调用方已显式设置该变量，每次启动仍会 shell out 到 nvidia-smi 探测 NVLink，在 ROCm 机器上毫无意义。改为 get() or 探测后只在未设置时才探测。tests/fast/launch_scripts/py_harness.py 的 call_entrypoint 现在保存并在 finally 中恢复 os.environ，解决 run_inkling.py 导出的 MODEL_ARGS_NUM_LAYERS 泄漏到后续测试的问题（未修复时先跑 test_py_launch_scripts.py 再跑 test_model_args.py 会失败 8 个快照）。

5. 配套同步与测试

26 个快照 diff 每个对应一个修复并逐行核对只含预期变更；docs/models/deepseek/deepseek.md 恢复了被 #2391 替换的 V3 页面（run_deepseek.py 仍是 deepseek-v3 唯一 launcher），并补充 external-ray 流程说明（MILES_SCRIPT_EXTERNAL_RAY=1）；tests/e2e 的 mimo MTP-only-grad 用例标记 FIXME；顺手补上了 #2300 新增 dashboard 参数导致的快照失配。验证：pytest tests/manual/launch_scripts tests/fast/launch_scripts tests/fast/utils/test_command_utils.py 全量 594 通过，随机顺序下 524 也通过。

关键文件：

- scripts/run_deepseek.py（模块 启动脚本；类别 source；类型 core-logic；符号 _execute_train, _prepare_ray_dependent, train）：DeepSeek-V3 唯一 launcher。引入 before_ray_job_submit 钩子并将 _prepare_megatron_ckpt/_prepare_cp 移入该回调，修复多节点 checkpoint 转换因 ray 集群未就绪而必然失败的问题，是本次最大的一处重构。
- scripts/run_deepseek_v32.py（模块 启动脚本；类别 source；类型 core-logic；符号 _execute_train, full_train）：DeepSeek-V3.2 launcher，与 run_deepseek.py 同构修复：full_train 通过 partial 把 _prepare_megatron_ckpt 注入 before_ray_job_submit，移除原链式调用。

- miles/utils/external_utils/command_utils.py（模块 命令工具；类别 source；类型 core-logic；符号 execute_train）：共享执行器 execute_train 是所有 python launcher 的公共出口。修复 NCCL_NVLS_ENABLE 默认值急切求值导致的每次启动都探测 NVLink 的副作用。
- tests/fast/launch_scripts/py_harness.py（模块 测试夹具；类别 test；类型 test-coverage；符号 call_entrypoint）：快照 harness 修复：call_entrypoint 在调用前后保存并恢复 os.environ，消除 launcher 导出变量对后续测试的泄漏，这是随机顺序下测试全绿的关键。
- scripts/run_glm45_355b_a32b.py（模块 启动脚本；类别 source；类型 core-logic；符号 ScriptArgs）：hardware 默认值从 H100 改为 GB200，消除与 _execute_train 断言的冲突，使快照 harness 不再需要覆盖该参数，也修正了脚本默认跑不起来的现状。
- scripts/run_qwen3_30b_a3b.py（模块 启动脚本；类别 source；类型 core-logic；符号 execute）：补回 U.execute_train 的 config=args，恢复 cuda_core_dump 与 extra_env_vars 的传递，否则这些功能被静默丢弃。
- scripts/amd/run_qwen3_30b_a3b.py（模块 启动脚本；类别 source；类型 core-logic；符号 execute）：AMD 版本的同一个 bug：漏传 config=args，与主版本同步修复。
- scripts/run_glm47_flash.py（模块 启动脚本；类别 source；类型 core-logic；符号 execute）：恢复被注释的 --eval-interval 20。该 flag 默认是 None，注释导致 eval 块从未运行，训练流程每次都跳过评估。
- scripts/run-qwen3-4B_4xgpu.sh（模块 启动脚本；类别 other；类型 core-logic）：WANDB_ARGS 改为条件构造并对 ${WANDB_KEY} 加引号（未设置时不再让 --wandb-key 吞掉下一个 flag），同时补 --num-gpus-per-node 4 修正 4 卡共置配置。
- scripts/run-gpt-oss-20b-bf16.sh（模块 启动脚本；类别 other；类型 core-logic）：补上从未存在的 NVLink 探测逻辑（此前读取未初始化变量导致 NCCL_NVLS_ENABLE 为空），并删除未声明的 ${EVAL_ARGS[@]} 展开。

关键符号：_execute_train, _prepare_ray_dependent, full_train, train, execute_train, call_entrypoint, execute

关键源码片段

scripts/run_deepseek.py

DeepSeek-V3 唯一 launcher。引入 before_ray_job_submit 钩子并将 _prepare_megatron_ckpt/_prepare_cp 移入该回调，修复多节点 checkpoint 转换因 ray 集群未就绪而必然失败的问题，是本次最大的一处重构。

```
# scripts/run_deepseek.py
# 核心修复：把依赖 ray 集群的 checkpoint 转换 / 拷贝步骤推迟到集群就绪之后。
# 旧流程在 train 命令里顺序执行 _prepare_megatron_ckpt / _prepare_cp，
# 这两个函数通过 exec_command_multi_node fan out，第一步就是
# ray.init(address="auto")，在干净主机上会先于 launcher 自己的
# ray start --head 崩溃；即便手动起集群，execute_train 的 preamble 也会
# 先 ray stop --force 再拉全新 head，导致多节点转换与单节点训练错配。

def _execute_train(args: ScriptArgs, before_ray_job_submit=None):
```

```
    # 此处省略 ckpt_args / rollout_args / optimizer_args 等参数串组装
    train_args = (
        f"{ckpt_args} "
        f"{rollout_args} "
        f"{optimizer_args} "
        f"{grpo_args} "
        f"{U.get_default_wandb_args(__file__, run_id=args.run_id)} "
        f"{perf_args} "
        f"{eval_args} "
        f"{sglang_args} "
        f"{misc_args} "
        f"{args.extra_args} "
    )
    U.execute_train(
        train_args=train_args,
        config=args,
        num_gpus_per_node=args.num_gpus_per_node,
        megatron_model_type=args.megatron_model_type,
        extra_env_vars={**sglang_extra_env_vars},
        megatron_path=args.megatron_path,
        # 新增钩子：execute_train 在 ray start --head 成功之后、
        # ray job submit 之前回调，保证多节点步骤拿到可用集群地址
        before_ray_job_submit=before_ray_job_submit,
    )


@app.command()
@U.dataclass_cli
def train(args: ScriptArgs):
    _prepare_download(args)
    _prepare_bf16_ckpt(args)
    # ray 依赖的步骤不再直接链式调用，而是作为回调注入
    _execute_train(args, before_ray_job_submit=partial(_prepare_ray_dependent, args))


def _prepare_ray_dependent(args: ScriptArgs):
    """ray 集群就绪后才执行的多节点准备工作。"""
    _prepare_megatron_ckpt(args)
    _prepare_cp(args)
```

miles/utils/external_utils/command_utils.py

共享执行器 execute_train 是所有 python launcher 的公共出口。修复 NCCL_NVLS_ENABLE 默认值急切求值导致的每次启动都探测 NVLink 的副作用。

```
# miles/utils/external_utils/command_utils.py
# NCCL_NVLS_ENABLE 默认值修复。
# os.environ.get(name, default) 的 default 表达式是急切求值的：
# 即便调用者已显式设置 NCCL_NVLS_ENABLE，check_has_nvlink() 仍会被执行，
# 每次启动都 shell out 到 nvidia-smi 做探测，在 ROCm 上无意义。
```

```
# 改为 get() or 探测 后，只有环境变量未设置（或为空串）时才探测。

runtime_env_vars = {
    "PYTHONUNBUFFERED": "1",
    # 显式设置时直接采用，未设置时才探测 NVLink
    "NCCL_NVLS_ENABLE": os.environ.get("NCCL_NVLS_ENABLE") or str(int(check_has_nvlink())),
    **{
        k: os.environ[k]
        for k in ("NCCL_SOCKET_IFNAME", "GLOO_SOCKET_IFNAME", "NCCL_DEBUG", "NCCL_
        DEBUG_FILE")
        if k in os.environ
    },
    # 其余 runtime env 键（MASTER_ADDR、CUDA_COREDUMP 系列等）从略
    **extra_env_vars,
    **_parse_extra_env_vars(config.extra_env_vars),
}
runtime_env_json = json.dumps({"env_vars": runtime_env_vars})
```

tests/fast/launch_scripts/py_harness.py

快照 harness 修复：call_entrypoint 在调用前后保存并恢复 os.environ，消除 launcher 导出变量对后续测试的泄漏，这是随机顺序下测试全绿的关键。

```
# tests/fast/launch_scripts/py_harness.py
# 修复环境变量泄漏：launcher 可能导出自己的配置，如 run_inkling.py
# 会设置 MODEL_ARGS_NUM_LAYERS 供其剪枝变体使用，旧 harness 从不恢复，
# 导致后续所有快照记录依赖“前一个运行的是哪个 launcher”。
# 复现：先运行 test_py_launch_scripts.py 再运行 test_model_args.py
# 会使 8 个 model-args 快照失败；pytest-randomly 下任何顺序都可能触发。

def call_entrypoint(module: ModuleType, name: str, overrides: dict[str, object], sandbox: Path) -
> None:
    entrypoint = getattr(module, name)
    first = next(iter(inspect.signature(entrypoint).parameters.values()), None)
    saved_env = dict(os.environ)
    try:
        with host_filesystem_frozen(sandbox):
            if first is not None and first.name == "args":
                entrypoint(module.ScriptArgs(**overrides))
            else:
                entrypoint(**overrides)
    finally:
        # 泄漏的开关会让后续记录依赖 launcher 执行顺序，这里强制还原
        os.environ.clear()
        os.environ.update(saved_env)
```

评论区精华

该 PR 审查过程零评论：guapisolo 直接 APPROVED，没有展开任何讨论。设计权衡的论证全部沉淀在 20 个 commit 的 message 中，最有信息量的几条：d654bad 解释了为何“手动起好

集群再跑 launcher”也不可行——execute_train 的 preamble 会先 ray stop --force 再拉起全新单节点 head，导致多节点转换后跟单节点训练错配；97b136e 指出 os.environ.get 的默认参数被急切求值，即使调用者已设置 NCCL_NVLS_ENABLE 也会无条件 shell out 到 nvidia-smi，包括 ROCm 机器；3a9c722 给出了环境泄漏的复现路径（先跑 test_py_launch_scripts.py 再跑 test_model_args.py 会失败 8 个快照，pytest-randomly 下任何顺序都可能触发）；21a1f43 则体现作者的注释哲学——理由属于 commit message，不属于调用点上方的四行注释。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 核心启动路径变更：before_ray_job_submit 重构改变了 run_deepseek.py / run_deepseek_v32.py 的执行顺序，checkpoint 转换从“launcher 直连”变为“execute_train 集群就绪后回调”；若 ray 启动失败，转换步骤不会执行，多节点用户必须按新文档使用外部集群模式（MILES_SCRIPT_EXTERNAL_RAY=1）并在 launcher 之前自行组集群。
  2. 默认行为变化：run_glm45_355b_a32b.py 的 hardware 默认值从 H100 改为 GB200，未显式传参的 H100 用户会拿到不同配置；run-qwen3-4B_4xgpu.sh 未设置 WANDB_KEY 时不再启用 wandb，依赖默认 wandb 的用户需显式设置；NCCL_NVLS_ENABLE 从“总是探测”变为“尊重显式设置”，空字符串环境变量仍会触发探测。
  3. 快照批量更新：26 个快照 diff 虽逐行核对，但快照测试的特性是“固化当前行为”，若某处修复本身有误，快照会同步固化错误；且 tests/manual 不在 CI 发现路径（#2279），回归可能漏网。
  4. 审查风险：0 条 review 评论、单一 approve，多脚本行为变更缺少第二轮审查视角。 - 影响：影响范围：scripts/ 下 10 余个 launcher、共享执行器 miles/utils/external_utils/command_utils.py、测试 harness 与 26 个快照、docs/models/deepseek 文档。用户侧收益：DeepSeek-V3 多节点训练从“启动即失败”变为可用，GLM-4.7 flash 评估恢复，AMD Qwen3-30B 恢复 cuda_core_dump 与 extra_env_vars 支持，4 卡共置的 Qwen3-4B 脚本不再因缺 --num-gpus-per-node 而错配资源。测试侧收益：harness 修复消除了测试顺序敏感性，随机顺序下 524 个测试通过，CI 更稳定。团队侧收益：为 #2356 的 Python 重写提供了干净基线，快照体系从此能真正发现回归而非固化错误。影响程度中等偏上，集中在启动脚本与快照测试体系。 - 风险标记：核心启动路径变更，快照批量更新，默认参数行为变化，无 review 讨论，外部 ray 集群依赖

关联脉络

- PR #2354 Delete the glm4-9B, mimo-7B, moonlight-16B and deepseek-r1 launch scripts: 本 PR 的 base 分支（yueming/script-delete），先删除废弃模型启动脚本；本 PR 在其上修复剩余脚本 bug，二者构成脚本清理三部曲的前两环。

- PR #2356 Python rewrite of the remaining launchers: 链式三部曲的第三环（Python 重写）；本 PR 先让 launcher 真正可用，为 argv 对比提供干净基线，避免移植时继承既有 bug。
- PR #1899 Add command-snapshot suite for launch scripts: 引入启动脚本快照套件，固化了坏输出；本 PR 的每个修复都以一条快照 diff 呈现，是修复清单的验证依据。
- PR #1901 Extend launch-script snapshot coverage: 与 #1899 配套的快照套件扩展，同样固化了坏行为；本 PR 批量更新其快照。
- PR #2391 docs: replace DeepSeek V3/R1 page with a DeepSeek-V3.2 recipe: 该 PR 用 V3.2 配方页替换了 deepseek.md；本 PR 恢复 V3 页面，因为 run_deepseek.py 仍是 deepseek-v3 模型类型唯一的 launcher，否则该 launcher 没有任何文档。
- PR #2300 feat: add dashboard args to the quick-start launcher: #2300 给 run-qwen3-4B.sh 新增 --use-miles-dashboard 和 --dump-details 但未重新生成快照；本 PR 补上该快照差异，并顺带暴露了 tests/manual 不被 CI 覆盖的问题（#2279）。