Karlsruhe Institute of Technology

Checkpointing

Jonas Eppelt, Matthias Schnepf, Günter Quast | August 28, 2023

ETP

Checkpointing Project

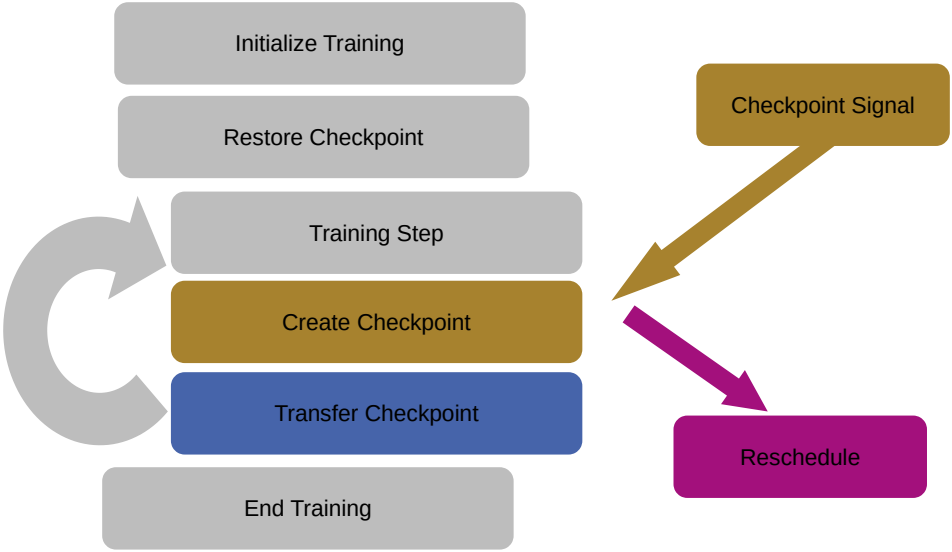
Motivation: Long-running ML Trainings

- not blocking resources for several days
- get intermediate output
- resistance to eviction/crashes of resources
- flexible usage of opportunistic resources $\rightarrow$ enables sustainability

Goals

- get it as approved as a service task (ST)
- develop self-checkpointing (training) jobs
- enable induced checkpointing by host (ST)
- provide examples for ETP, TOpAS, NAF (ST), KEK, Horeka
- provide convenience python class (ST)

Basic Training Program with checkpointing



How to Checkpoint: The ingredients

When to checkpoint:

- regularly (e.g. every 10 epochs)
- upon receiving a signal (SIGTERM, SIGINT, user-defined)

Where to store the checkpoint:

- shared filesystem (e.g. ETP)
- remote storage (e.g. xrootd)
- batch system (e.g. htcondor's transfer_checkpoint_files)

How to restart the job:

- manually (resubmit to the batch system, restart local process)
- batch system (htcondor's checkpoint_exit_code)
- Workflow Management (e.g. luigi)

Python class: almost 2 lines of code

```python
from checkpointer.checkpointer import Checkpointer

checkpointer = Checkpointer(
    # define checkpoint file
     local_checkpoint_file=Path("checkpoint.pt"),
    # define a function, that saves the checkpoint
    checkpoint_function=lambda path, model: torch.save(model.state_dict(),
        path),
    # define a function, that restores the checkpoint
    restore_function=lambda path: model.load_state_dict(torch.load(path)["
        model_state_dict"]),
# Reload checkpoint and give default value if there is none
model = checkpointer.restore(model)
# envoke the checkpoint_function
checkpointer.checkpoint(i)
```

When to checkpoint:

Regularly Checkpointing

```
checkpointer = Checkpointer(
    ...
    # define a frequency
    checkpoint_every = 100)
...
for i in range(epochs):
    train(train_dataloader, model
        , loss_fn, optimizer)
    test(test_dataloader, model,
        loss_fn)
    checkpointer.step(i, model)
```

Induced Checkpointing

- python program sets traps on SIGTERM and SIGINT automatically
- To run on a batch system change the executable:

```
# push program to the
    background
python3 counter.py &
pid=$! # get pid
# trap SIGINT & SIGTERM
trap "kill -15 $pid" 15
trap "kill -10 $pid" 10
# wait for program to exit
wait $pid
```

How to restart the job: HTCondor

- use the traps from induced checkpointing
- define a checkpoint_exit_code in the jdl file:

```
...
# define an exit code on which condor copies the checkpoint
checkpoint_exit_code = 85
# only if also checkpoint transfer via htcondor
transfer_checkpoint_files = checkpoint.txt
when_to_transfer_output = ON_EXIT_OR_EVICT
```

- give the same code to the checkpointer:

```
checkpointer = Checkpointer(
    ...
    checkpoint_exit_code = 85 # default
    # only if also checkpoint transfer via htcondor
    checkpoint_transfer_mode="htcondor" # automatically infers
        local_checkpoint_file from _CONDOR_JOB_AD
)
```

How to restart the job: Luigi

- Use checkpointer normally in the task.
- need to configure the luigi scheduler:

```
[scheduler]
retry_count=100
retry_delay=5
[worker]
keep_alive=true
max_reschedules=100
```

Where to store the checkpoints

Shared Filesystem

- set checkpoint_transfer_mode

```
checkpointer = Checkpointer(
    ...
    checkpoint_transfer_mode = "
        shared",
    checkpoint_transfer_target =
        Path("/work/jeppelt/
        checkpoint.txt")
)
```

- First, the checkpoint is written locally.
- Then it is transferred to the target.

Remote storage

- set checkpoint_transfer_mode

```
checkpointer = Checkpointer(
    ...
    checkpoint_transfer_mode = "
        xrootd",
    checkpoint_transfer_target =
        "/pnfs/...",
    xrootd_server_name = "root
        ://..." )
```

- First, the checkpoint is written locally.
- Then it is transferred to the target.
- Don't forget your proxy!

Checkpointing with LSF@KEKcc

Checkpointing in LSF

- has kernel-level checkpoints by default.
- works via additional scripts ‘echkpnt‘ and ‘erestart‘.
- is on application level only available in C and Fortran.

What is working so far

- transferring checkpoints to shared/remote filesystems $\rightarrow$ python programs can be checkpointed

Conclusion

Road to service Task

- see full examples in gitlab ✓
- provide examples with submitfiles for HTcondor systems (ETP, NAF) ✓
- demonstrate integration with workflow management system (b2luigi) ✓
- bring python prototype to real world application
  - used in my own project ✓
  - integrate in Flavourtagger pipeline

More Improvements

- provide example for KEK
- Bring to MonteCarlo Jobs
- enable integration to highlevel ML-frameworks (keras, lightning)
- think about benchmarking?