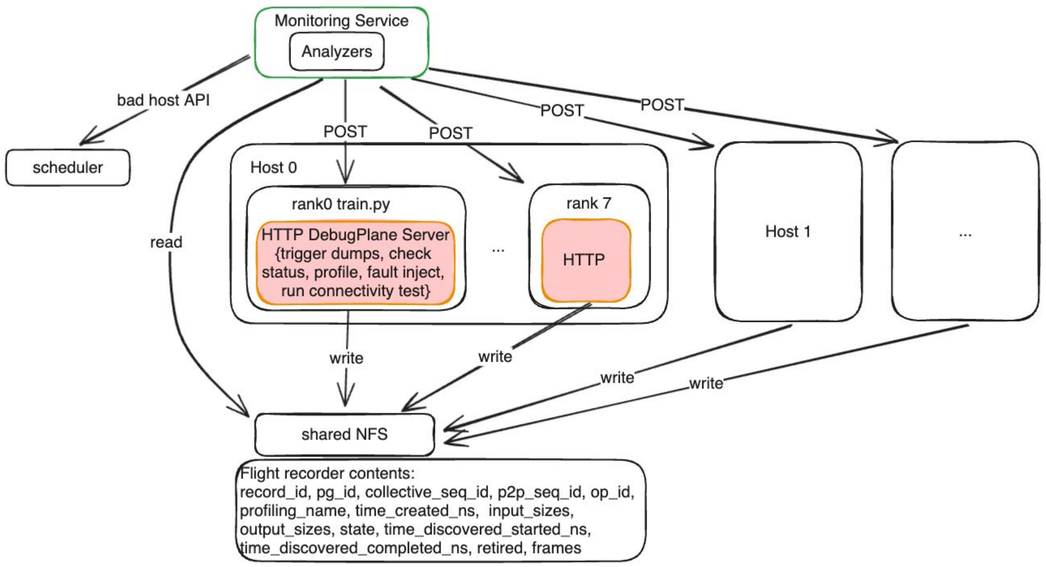
Fault Tolerance for Large Scale Training

Chirag Pandya and Tristan Rice, PyTorch Distributed, Meta

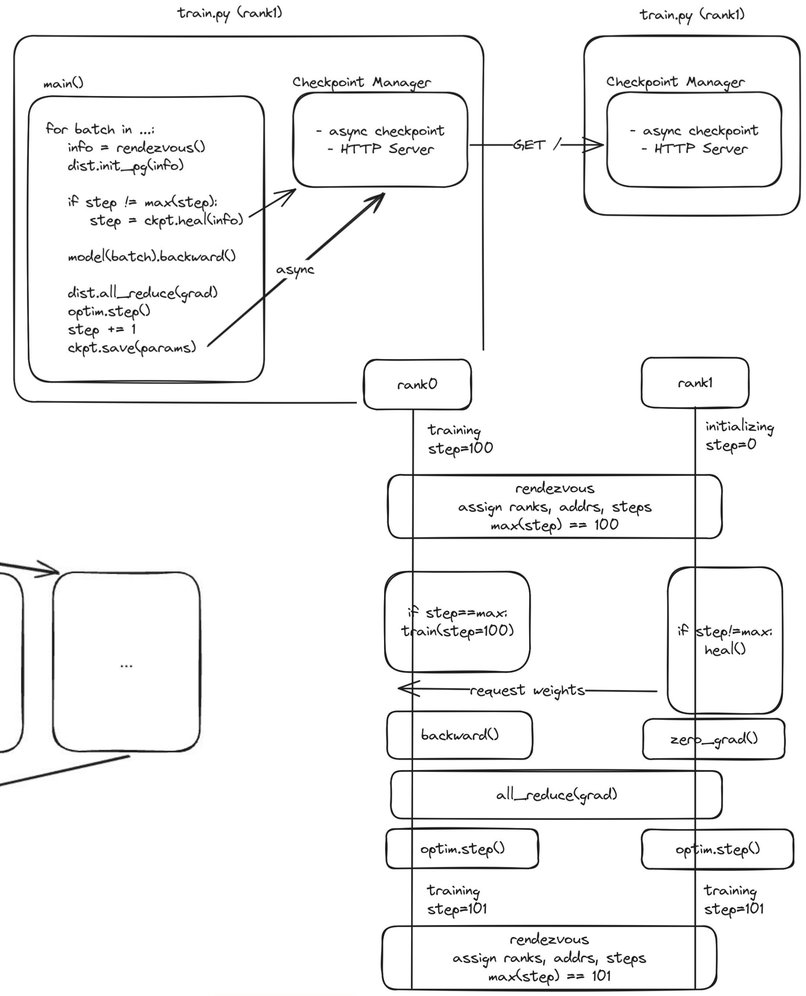
Monitoring / Error Detection

- **Flight Recorder** is a cutting-edge PyTorch feature that captures and logs records NCCL operations, providing valuable insights for identifying bad hosts and detecting errors including:
  - Deadlocks: processes blocked indefinitely, unable to proceed
  - Stragglers: slow/underperforming ranks
  - Mismatched collectives
  - Mismatched tensor sizes
  - NCCL timeouts and comm failures
- The **new WorkerServer** in PyTorch Distributed can be used to run a local HTTP server on every node with debug handlers for:
  - Retrieving flight recorder data
  - Probing NCCL debug state
  - Injecting failures
  - Running PyTorch and py-spy profilers.
- With flight recorder and WorkerServer, an independent monitoring service continuously polls the rank's HTTP server for diagnostic data and runs analysis
- Detailed flight recorder diagnostic data can optionally be written to shared NFS on timeout/failure for post-mortem analysis and for looking at trends



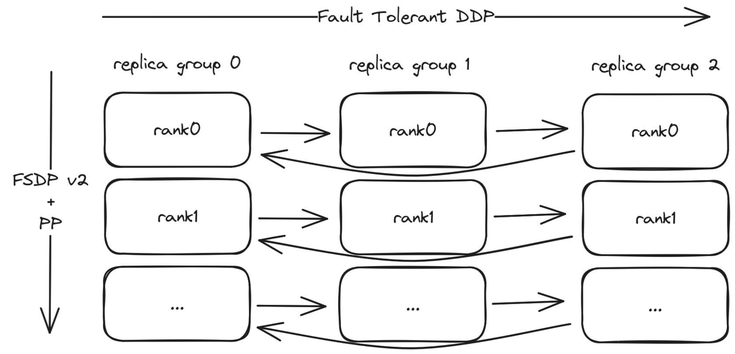
Live Checkpoint Recovery

- We're developing a novel way to **live recover from failures** by asynchronously saving checkpoints and serving them directly to newly joined and recovering workers.
- On worker start, the checkpoint is transferred via HTTP from an existing healthy worker.
- The weights are copied from the GPU in a non-blocking way during the forward pass using a separate CUDA stream.
- We use leader election to identify live workers and exchange step information to recover from failures.



Fault Tolerant HSDP

- We're developing a new fault tolerant version of **Hybrid Sharded Data Parallelism** that combines fault tolerant DDP with FSDP v2 and PP.
- On failure we use a customized comm library to gracefully handle errors and continue training on the next batch without downtime.
- This isolates failures to a single replica group and we can continue training with a subset of the original job.



Control Plane Hardening

- Torchelastic and TCPStore in PyTorch 2.4 just got **a major upgrade**! We've improved reliability, initialization time, and debuggability for large jobs.
- **Enhanced TCPStore libuv** backend with automatic retries, exponential backoff and better errors.
- Torchelastic static backend now **scales linearly** with the number of workers allowing it to work with the largest jobs.