Mycroft: Tracing Dependencies in Collective Communication Towards Reliable LLM Training

Yangtao Deng*, **Lei Zhang***, Qinlong Wang, Xiaoyun Zhi, Xinlei Zhang, Zhuo Jiang, Haohan Xu, Lei Wang, Zuquan Song, Gaohong Liu, Yang Bai, Shuguang Wang, Wencong Xiao, Jianxi Ye, Minlan Yu, Hong Xu

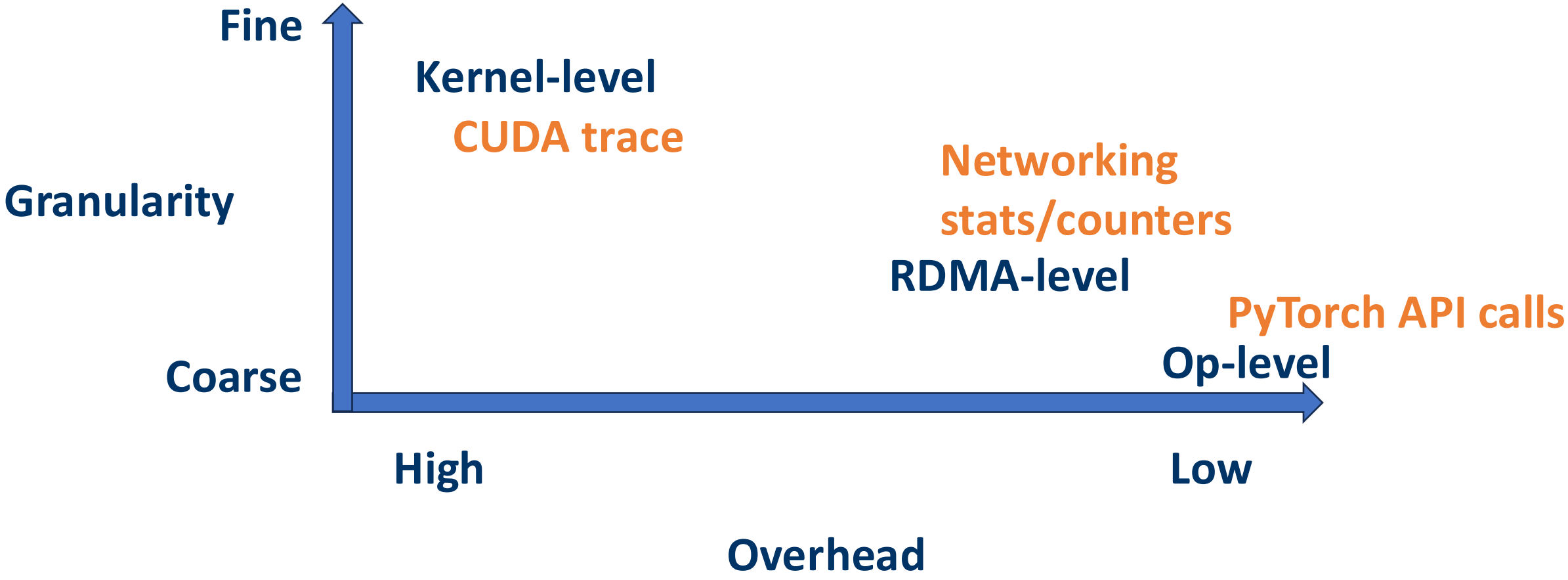
ByteDance

ByteDance | Seed

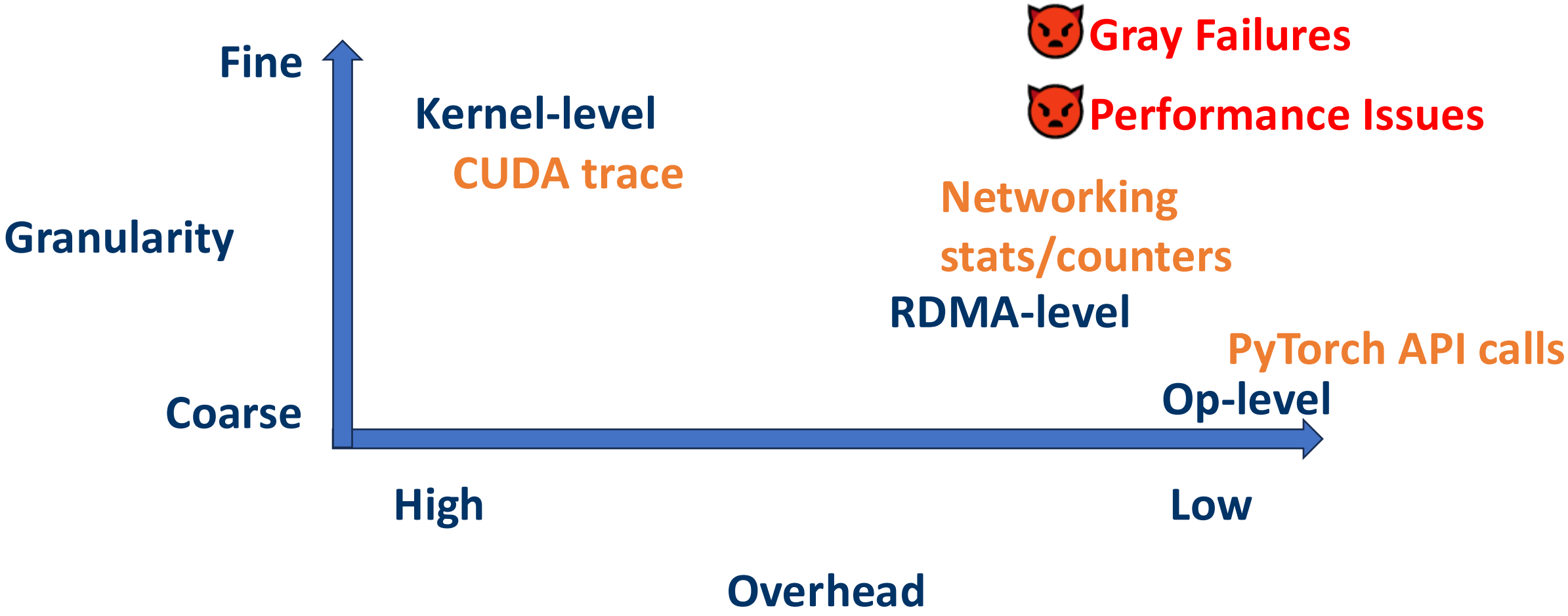
香港中文大學
The Chinese University of Hong Kong

HARVARD
UNIVERSITY

Today's Training Observability

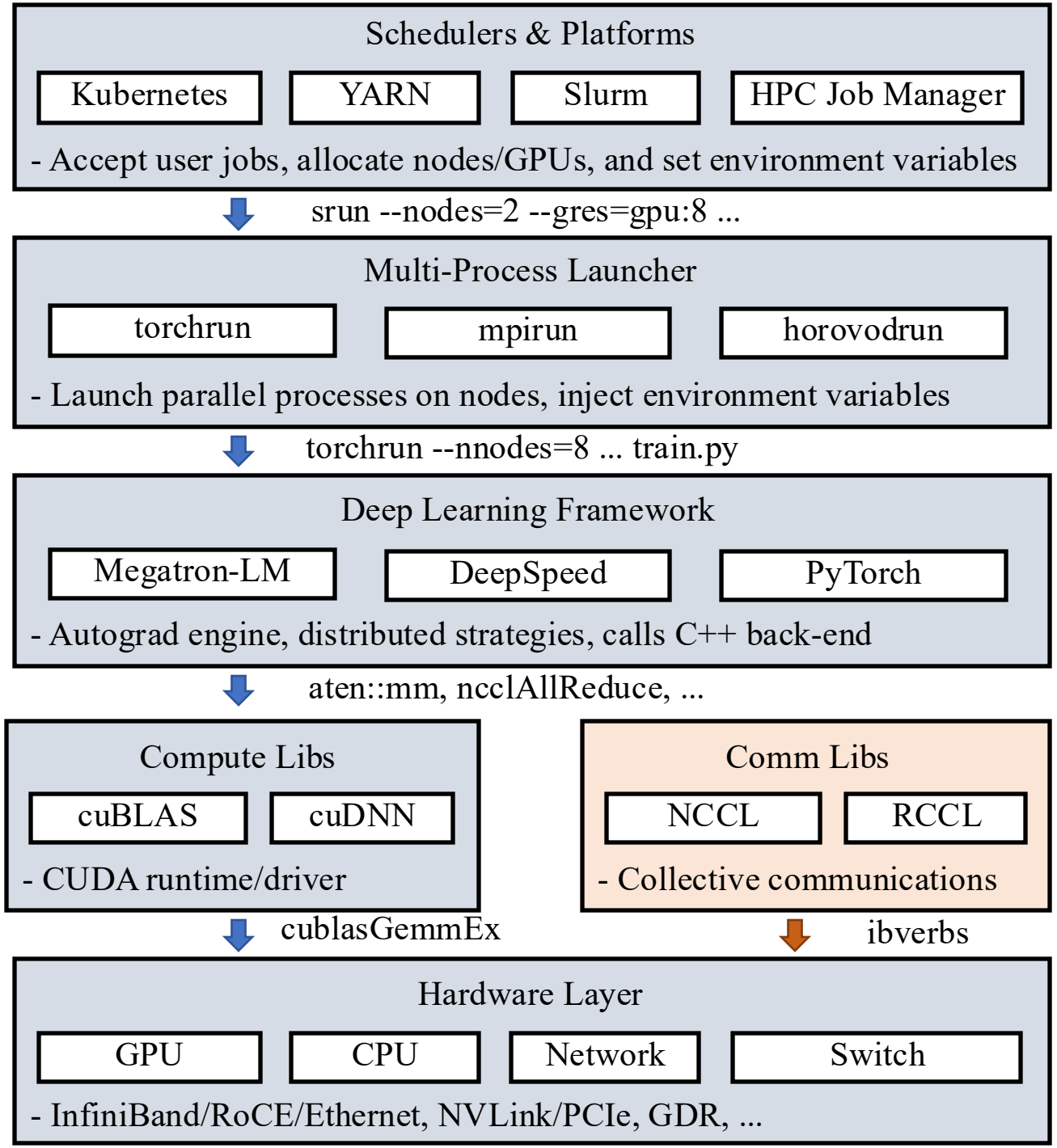


Debugging At Runtime, At Scale



Tracing Collective Communications

CCL observability was missing!

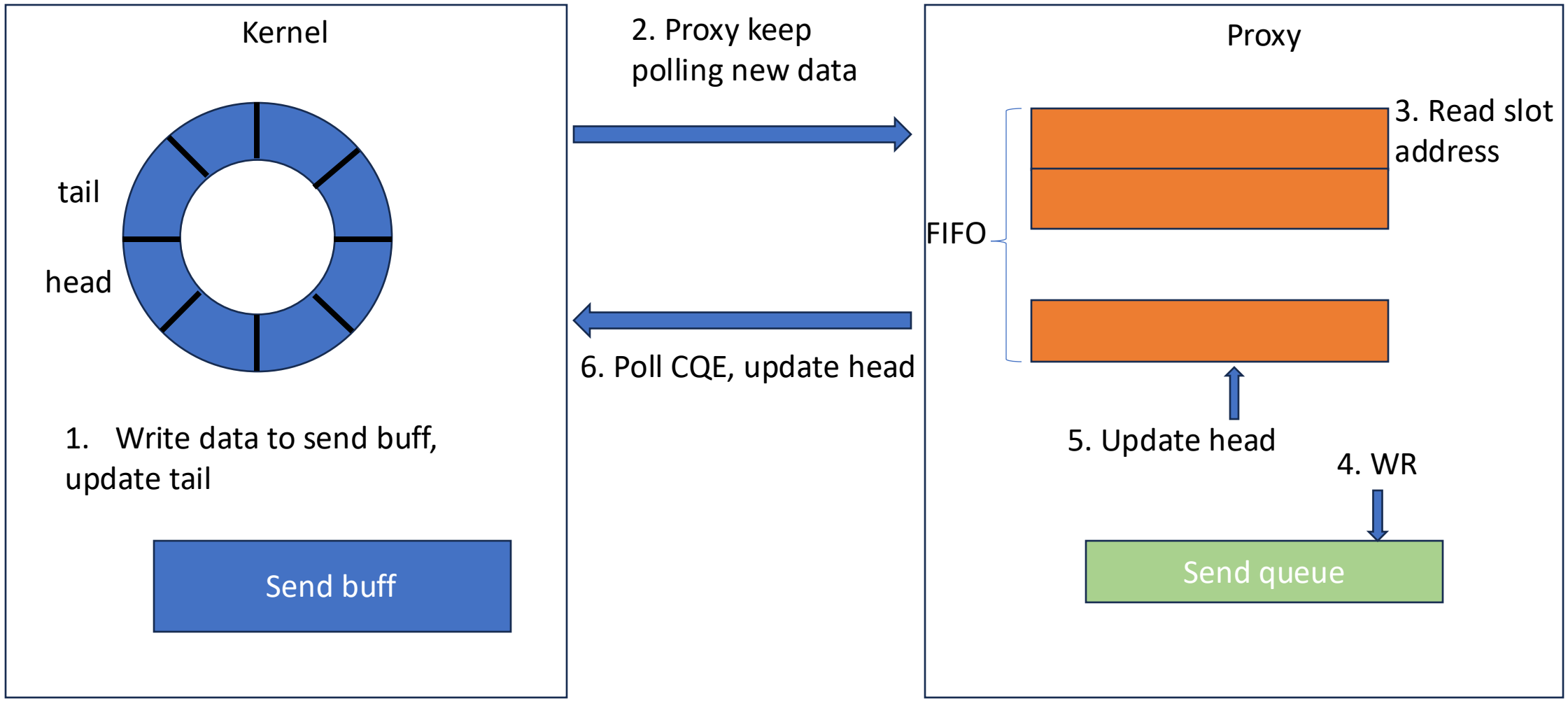


Why CCL Observability

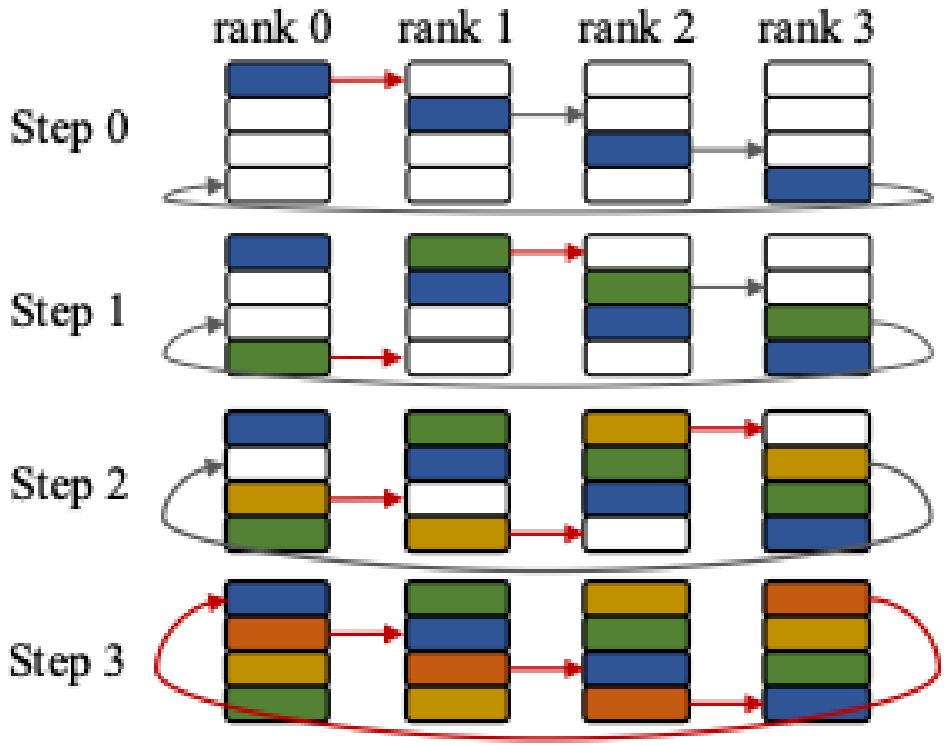
- CCLs are complex system software
  - Connect training framework and networking hardware
  - Involves control and data dependency with parallelisms

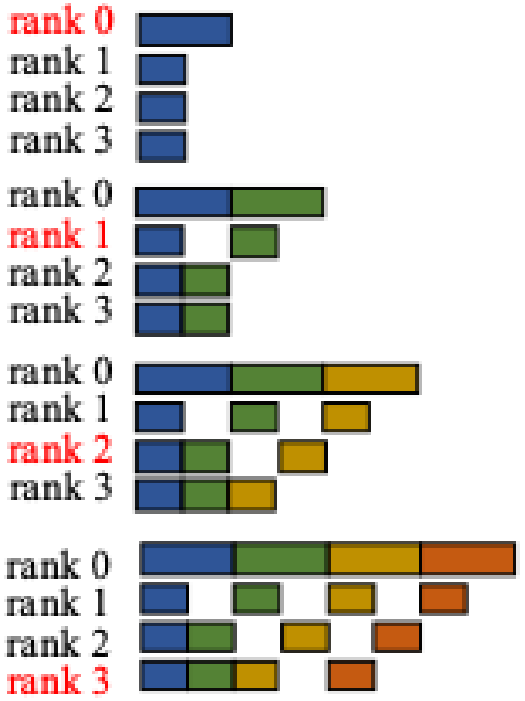
- Opportunity: construct a global state machine to represent such dependency

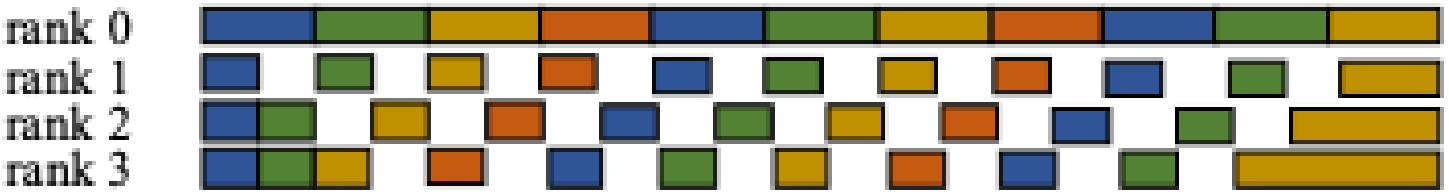
Intra-node Dependency



Inter-node Dependency

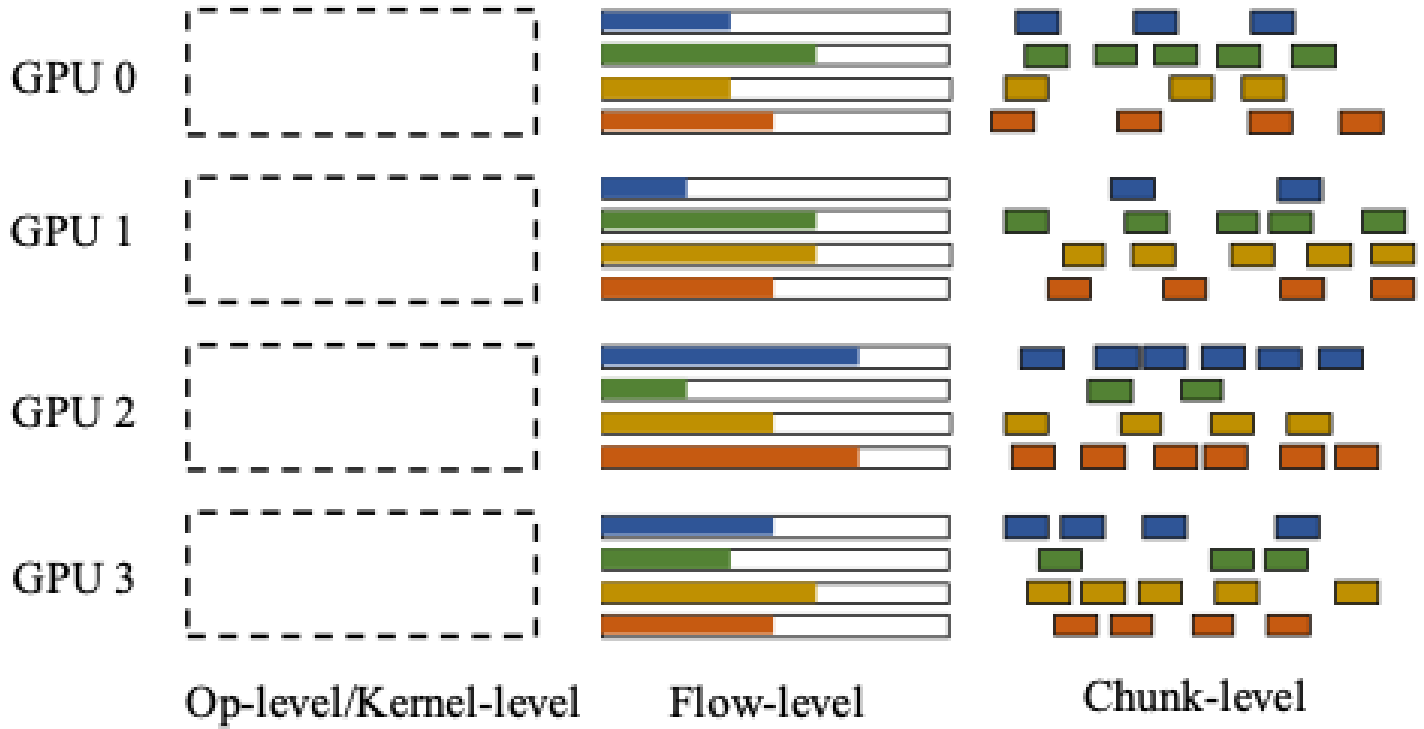






Tracing CCL Dependency

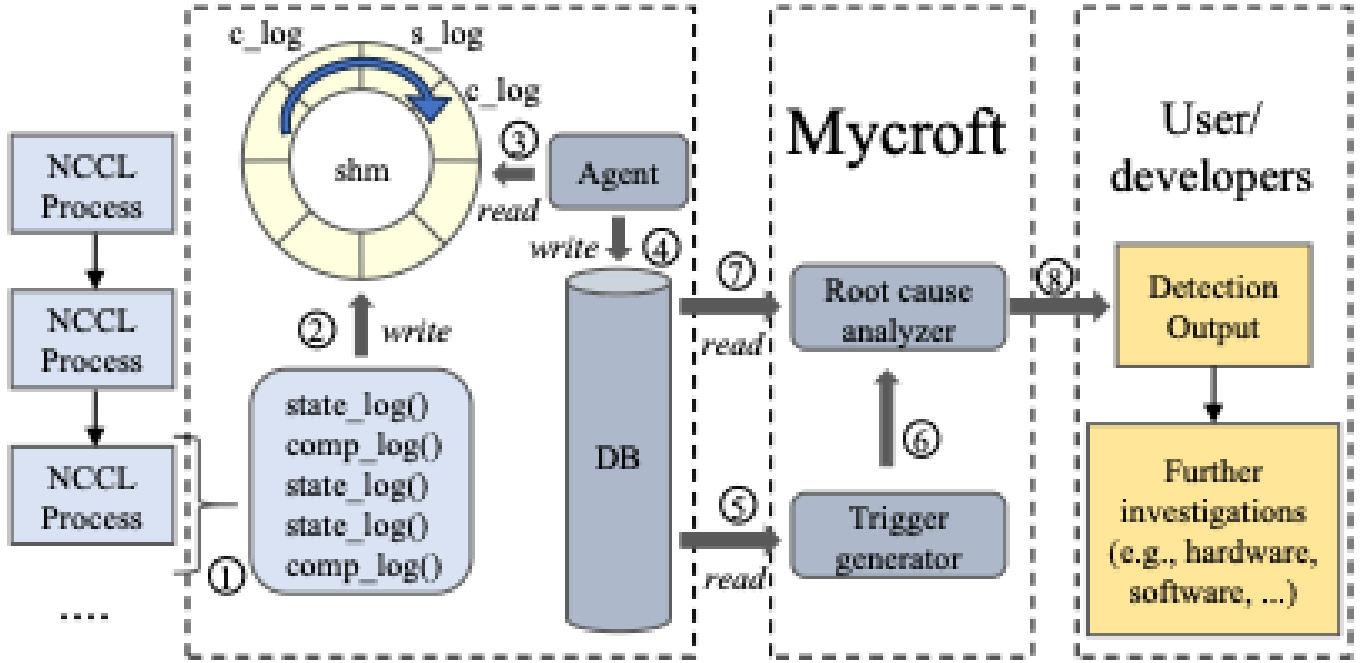
- Multiple levels
  - Chunk level
  - Flow level
- Fine-grained state machine
- Control & data dependency



Mycroft

- Lightweight, always-on runtime detection and root cause analysis

- Low overhead
  - Lightweight instrumentation
  - Trigger mechanism
  - Root cause analysis



Mycroft

- Instrumentation
  - Capture critical path of async coordination
  - Completion log and real-time state log
- Almost zero overhead to hosting CCL
  - Cheap memory management
  - Controlled trace data volume

Metadata	Operation	Chunk
• IP • Comm_id • GPU_id • NIC_id	• Timestamp • Op_name • Op_seq	• Stuck_time • Total_chunks • GPU_ready • RDMA_transmitted • RDMA_done

Mycroft

- Instrumentation

- Trigger Mechanism
  - Observation: reliability issues rapidly cascade to all nodes
  - Sample a subset of nodes is efficient enough
  - Faster than other existing (e.g. timeout) mechanisms

Mycroft

- Instrumentation

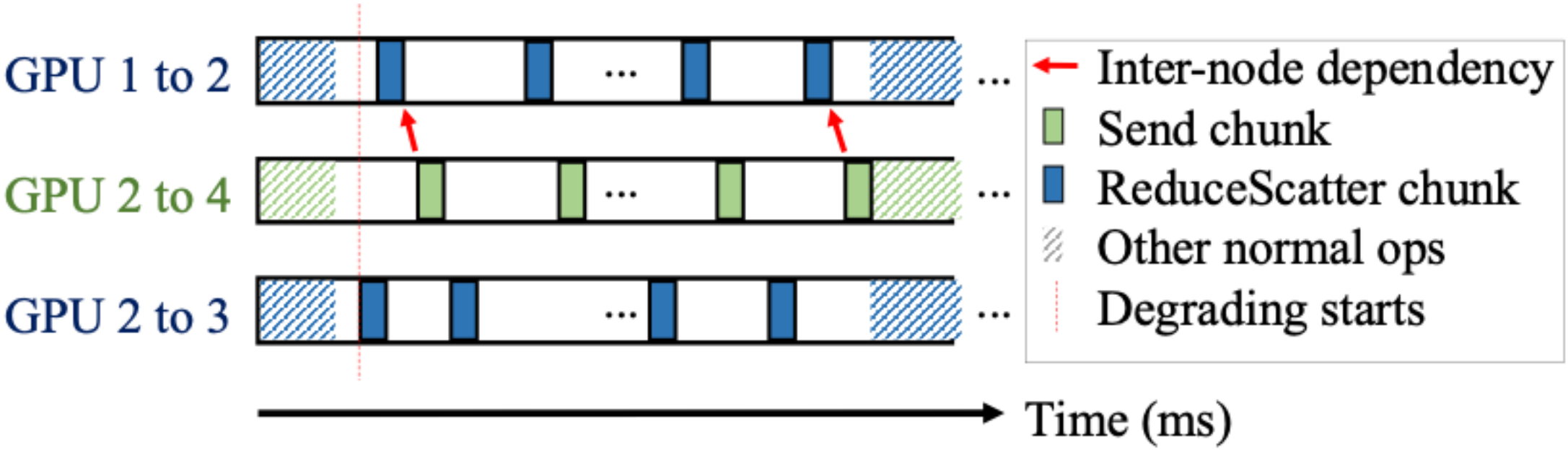
- Trigger Mechanism

- Debugging
  - Check collective communication rule violation
  - Zoom in root cause layers & system components

Level	Problem	Rule
Chunk-level	Failure	Each rank should transmit the same amount of data.
Chunk-level	Performance	Each component should finish within expected execution time.
Chunk-level	Performance	Each component should not block the downstream ones.
Flow-level	Failure	Each flow should complete.
Flow-level	Performance	Each flow should take similar execution time.
Flow-level	Performance	Each flow should start and end at similar time.

State	Condition	Local cause	Remote cause
Not started	①=②=③=0	Uninitialized	Blocked
Not transmitted	①>②	RDMA issue	Receiver not ready
Not delivered	②>③	RDMA issue	Receiver failed
GPU not ready	①=②=③>0	GPU issue	-

Debugging Example



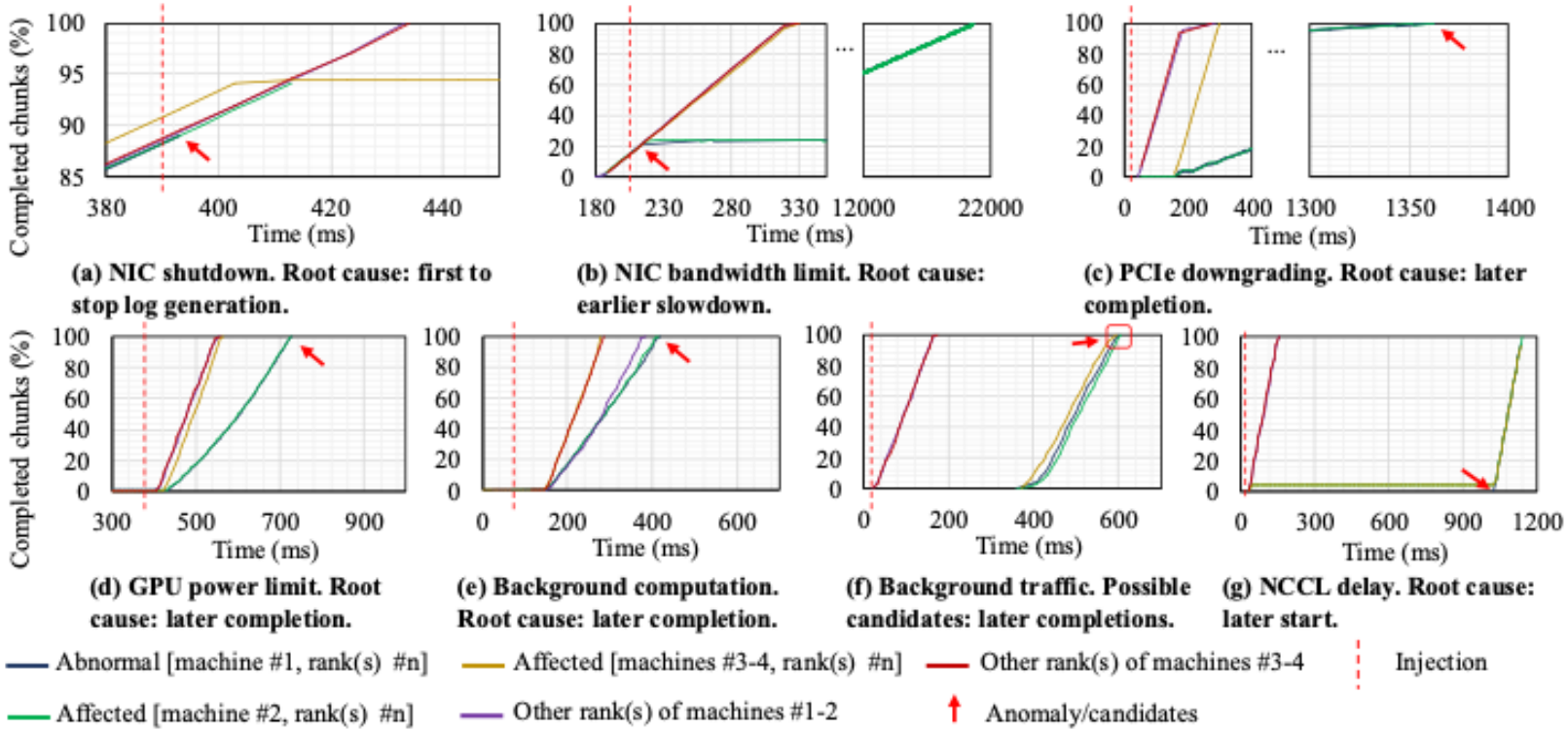
Evaluation

- Deployed in ByteDance’s production environment
  - No observable overhead on MFU

- Detection capability
- Performance & overhead
- Production performance

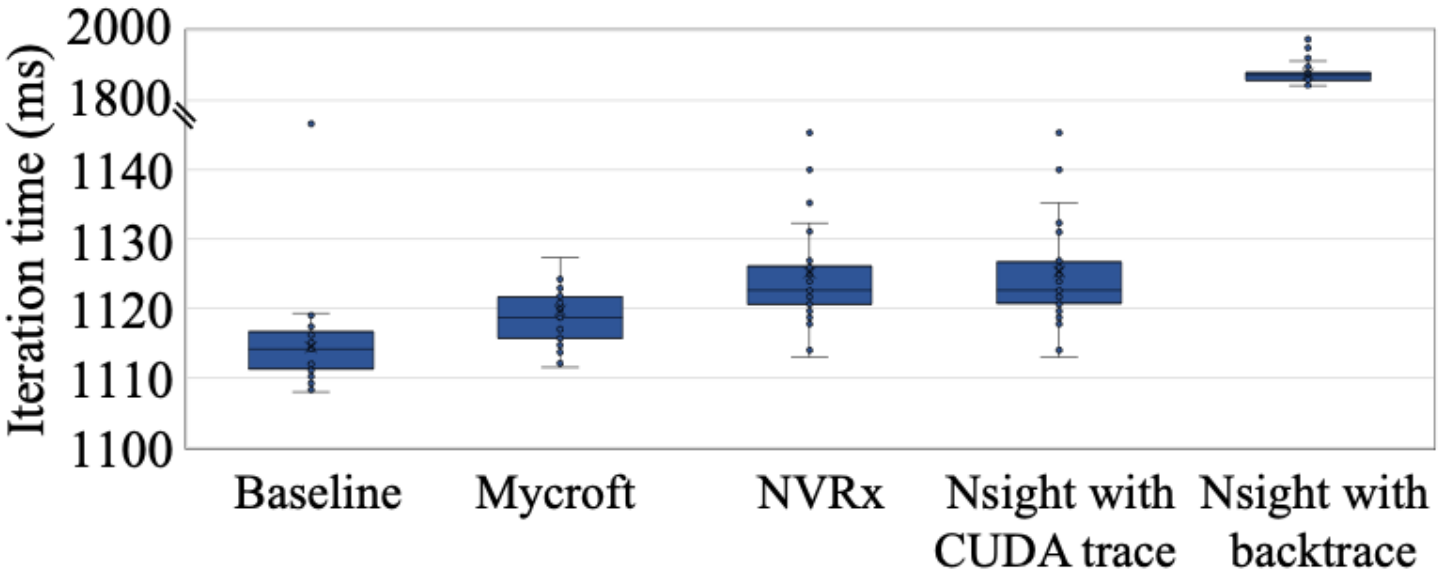
Capability

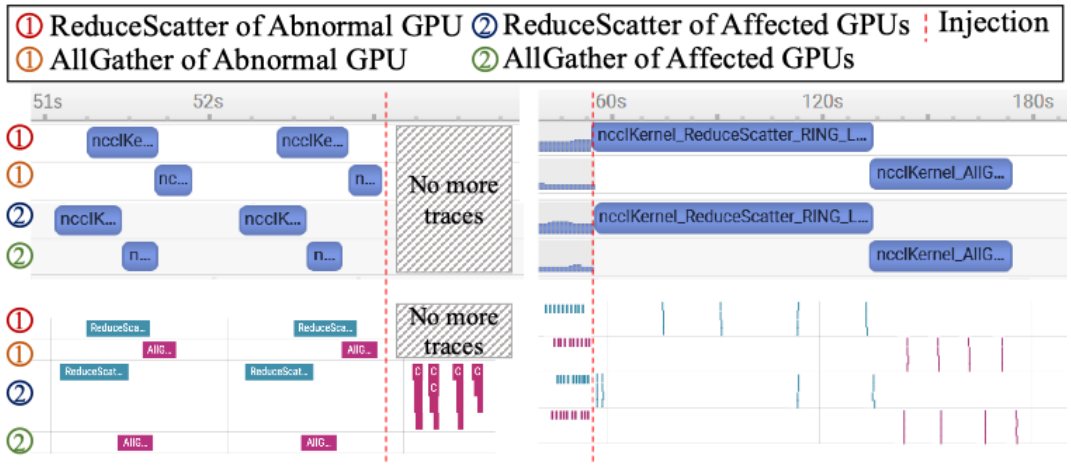
- Fault injection experiments



Overhead

- Almost zero overhead compared to barebone NCCL



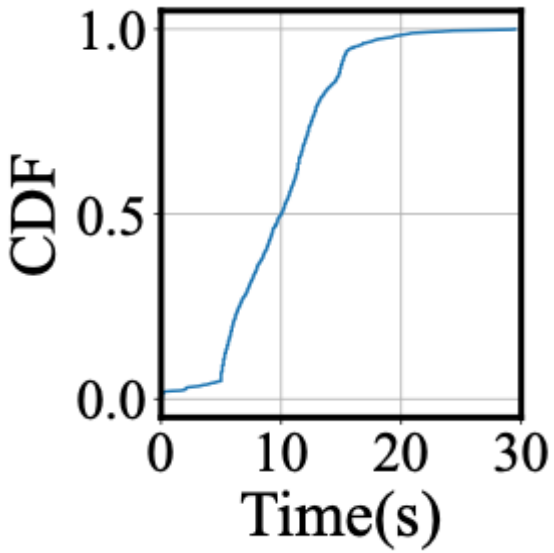


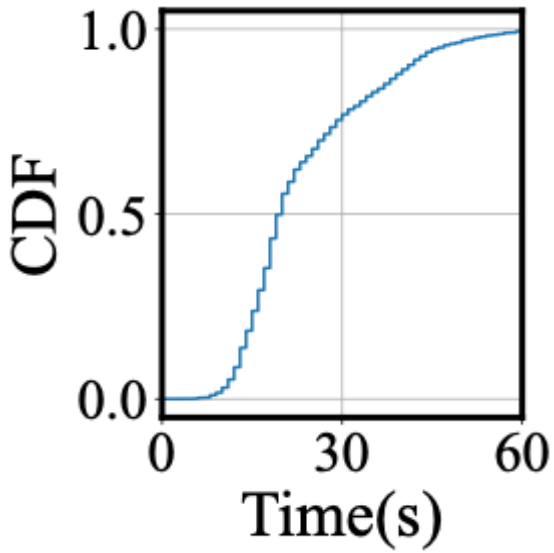
(a) NIC shutdown. Root cause: first to stop log generation.

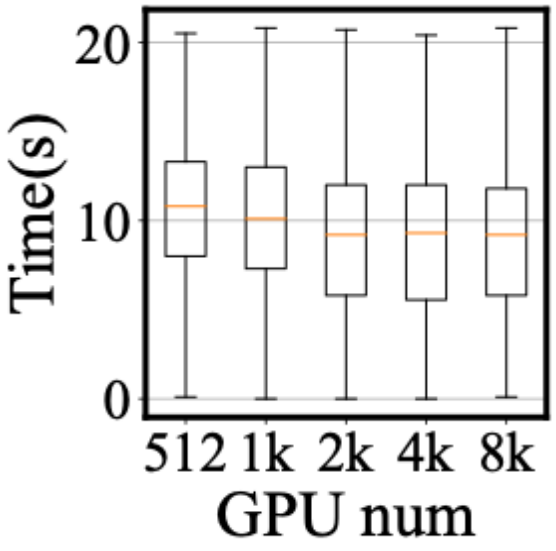
(b) NIC bandwidth limit. Root cause: earlier slowdown.

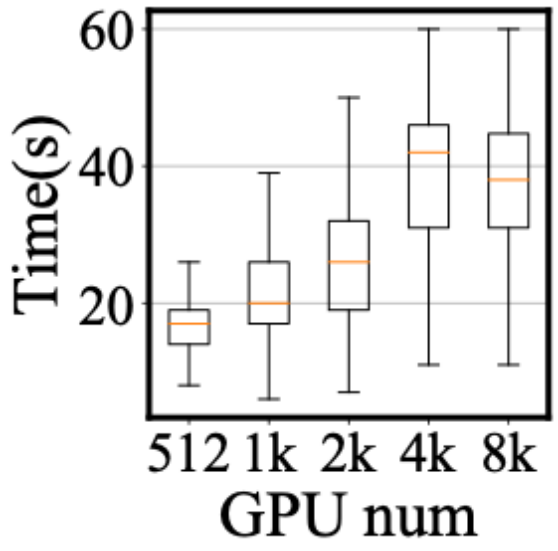
Production Performance

- Detect 90% problems in 15 seconds
- Reveal 60% root cause in 20 seconds









More in the paper

- Tracing data pipeline
- Case studies
- Discussions
- Integration with other debugging tools

Conclusion

- CCL observability was missing
- Opportunities
  - Consider CCL as a system software
  - Reveal dependencies to track control & data dependency
- Mycroft: Tracing CCL dependencies for reliability
- Capture real-world bugs in real time

Thank you