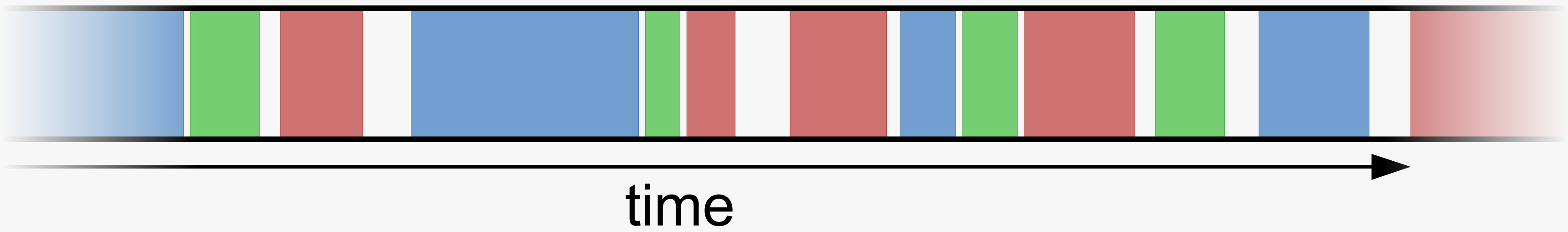
LoGA: Low-Overhead GPU Accounting Using Events

Jens Kehne, Stanislav Spassov, Marius Hillenbrand, Marc Rittinghaus, Frank Bellosa
10th ACM International Systems and Storage Conference

Operating Systems Group, Karlsruhe Institute of Technology (KIT)

GPU Sharing

- GPUs increasingly popular in computing
- Not every application saturates a GPU

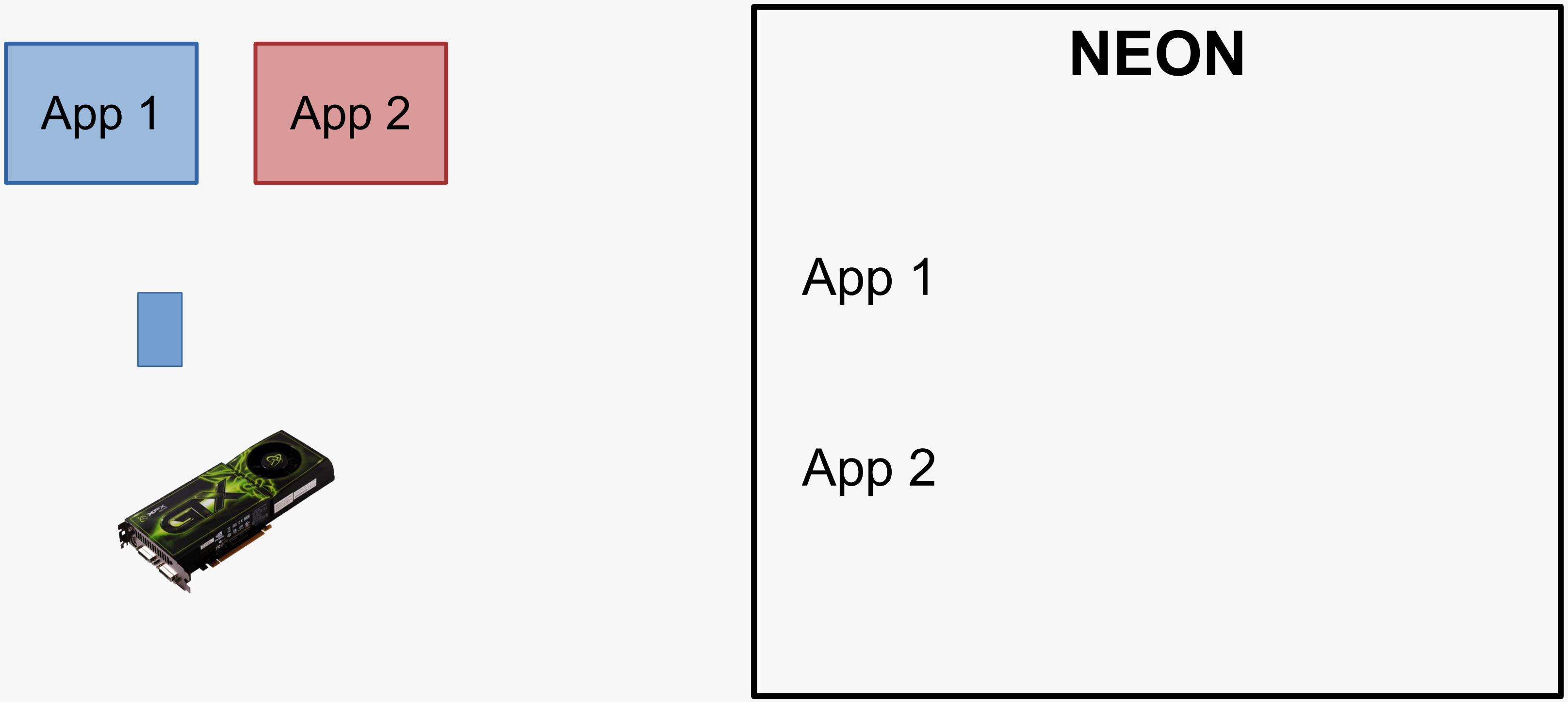


- Move GPUs to the cloud
  - Sharing increases cost-efficiency

- Problem: Need fairness
- Software scheduling is inefficient

NEON (University of Rochester, ASPLOS '14)

- Applies fair queuing to GPUs



NEON (University of Rochester, ASPLOS '14)

- Applies fair queuing to GPUs

App 1

App 2

NEON

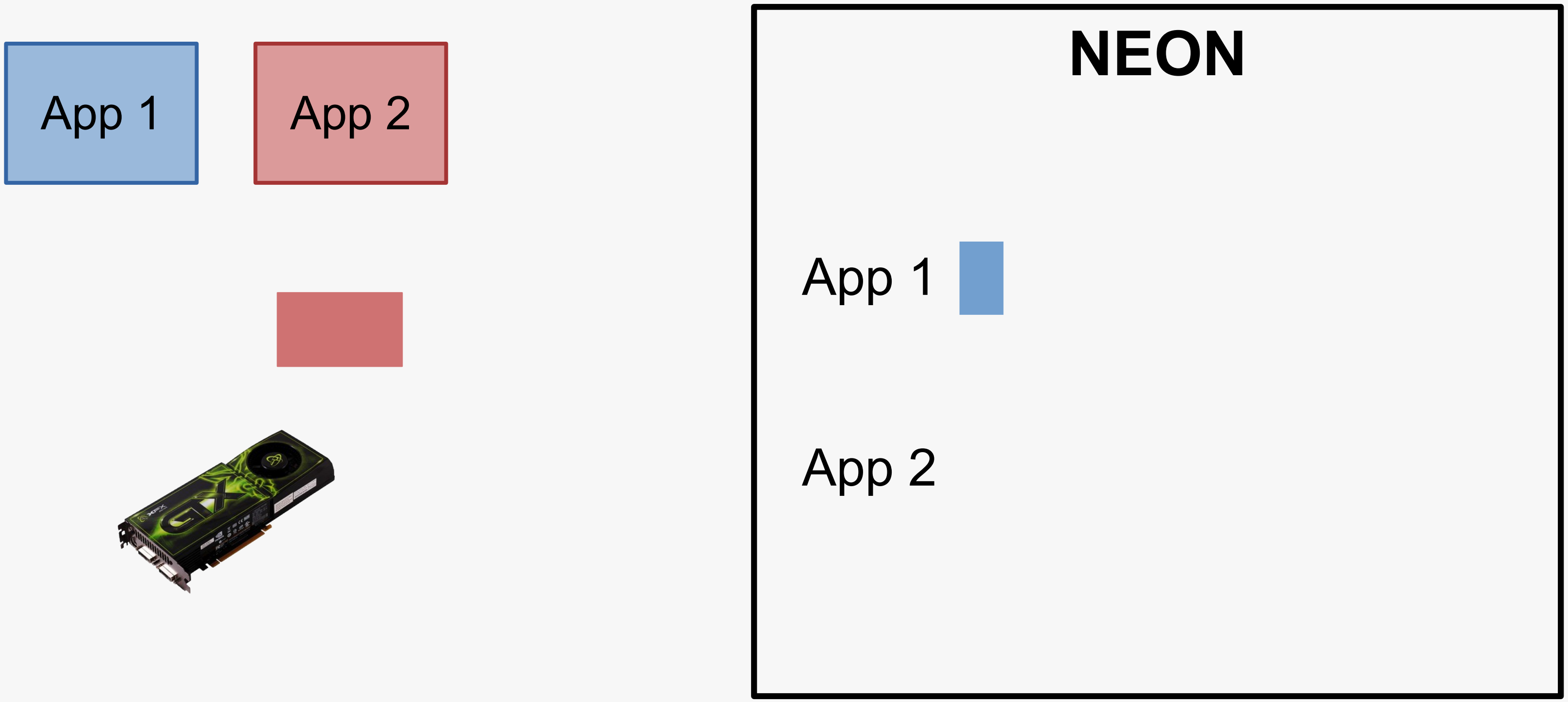
App 1

App 2

NEON (University of Rochester, ASPLOS '14)

Karlsruhe Institute of Technology

- Applies fair queuing to GPUs

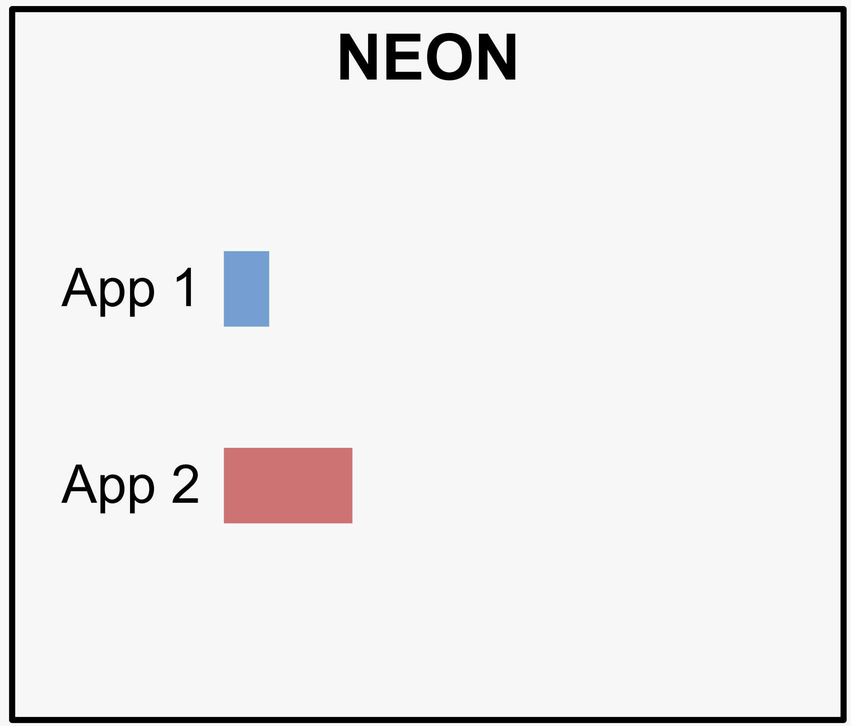


NEON (University of Rochester, ASPLOS '14)

- Applies fair queuing to GPUs

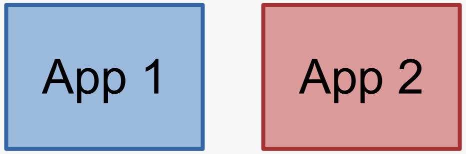
App 1

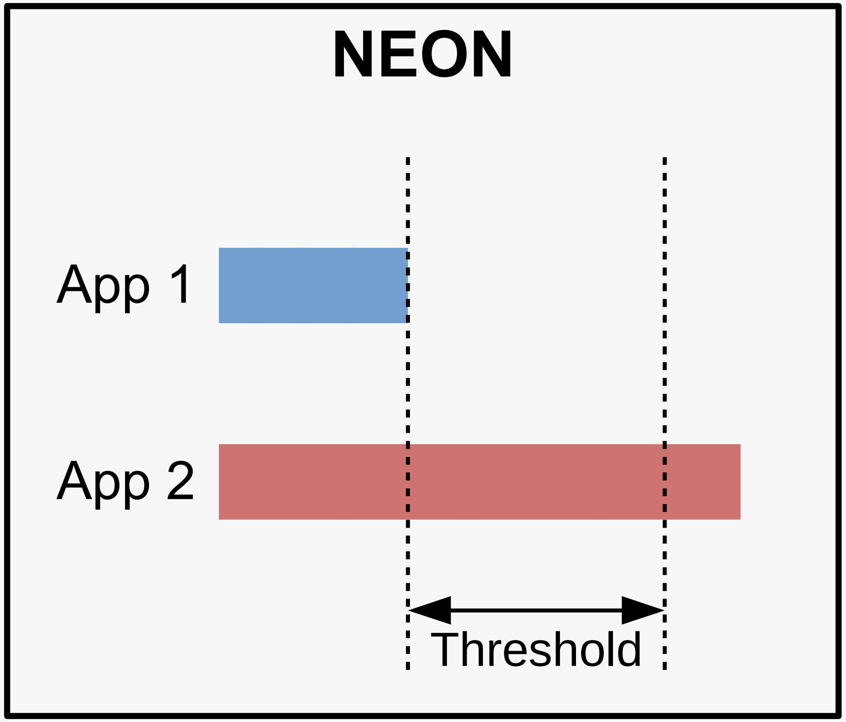
App 2



NEON (University of Rochester, ASPLOS '14)

- Applies fair queuing to GPUs



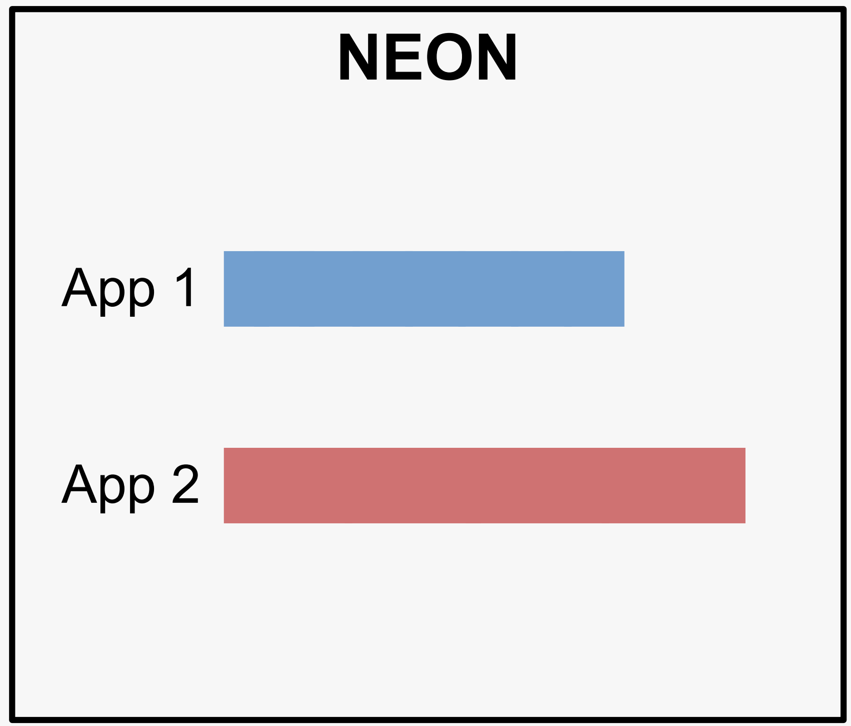


NEON (University of Rochester, ASPLOS '14)

- Applies fair queuing to GPUs

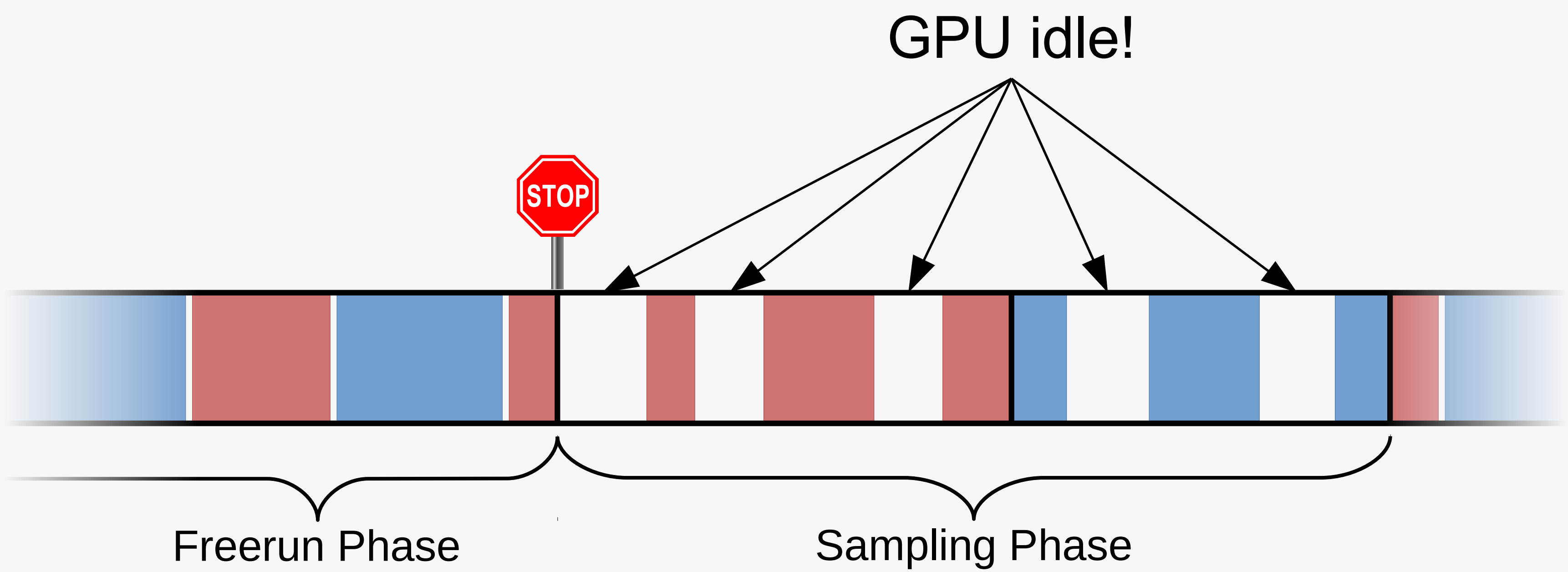
App 1

App 2



NEON: Accounting Problem

- NEON's accounting disables GPU access



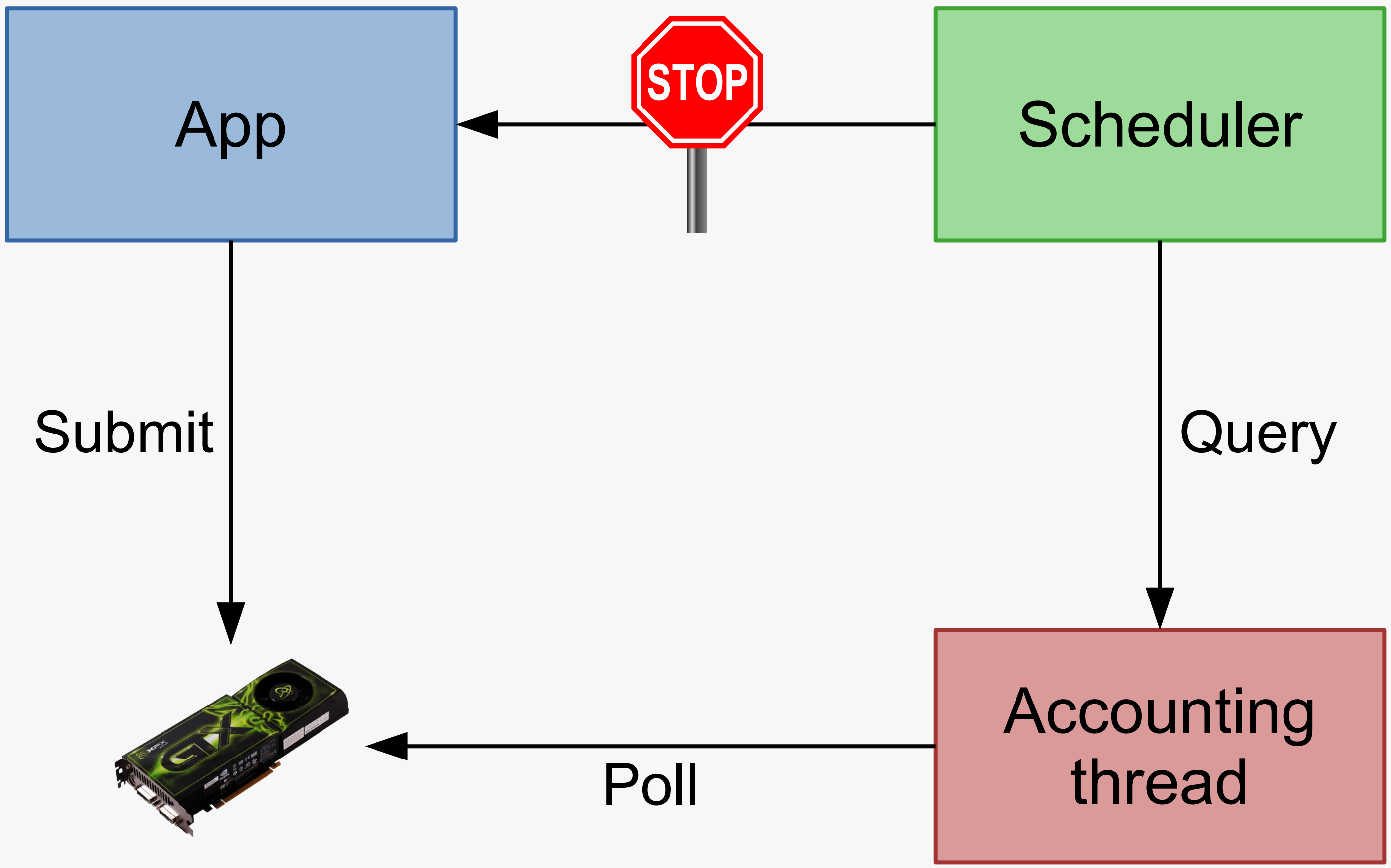
➔ High accounting overhead if application does not saturate the GPU

Idea

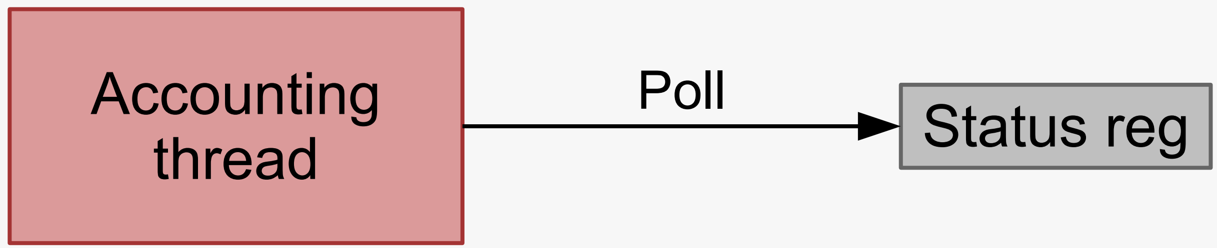
- GPUs have lots of status registers for the device driver
  - Leak information about GPU‘s internal state
- Reading has no effect on GPU driver or running application

- Idea:
  - Poll status registers
  - Infer GPU-internal context switches

Design



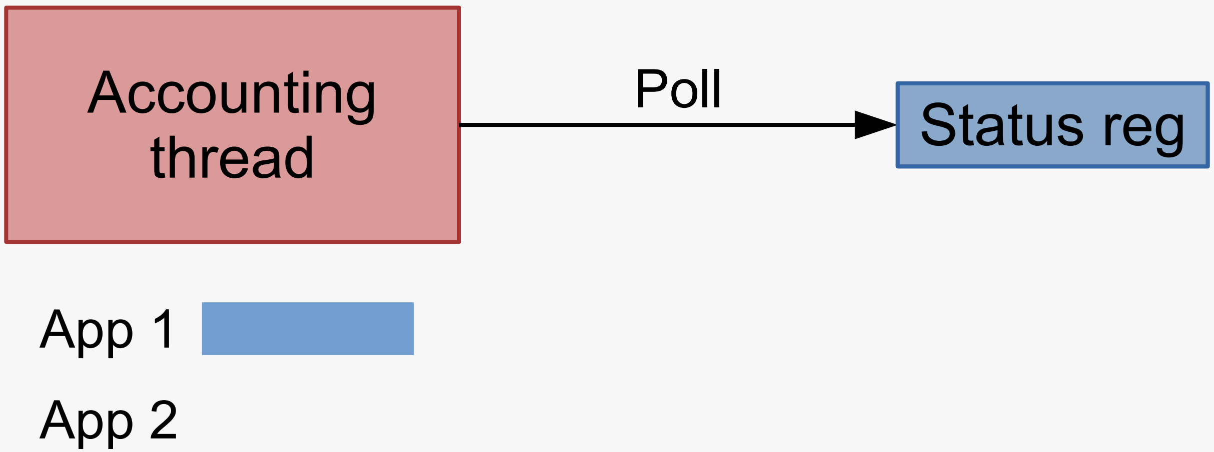
Accounting Thread



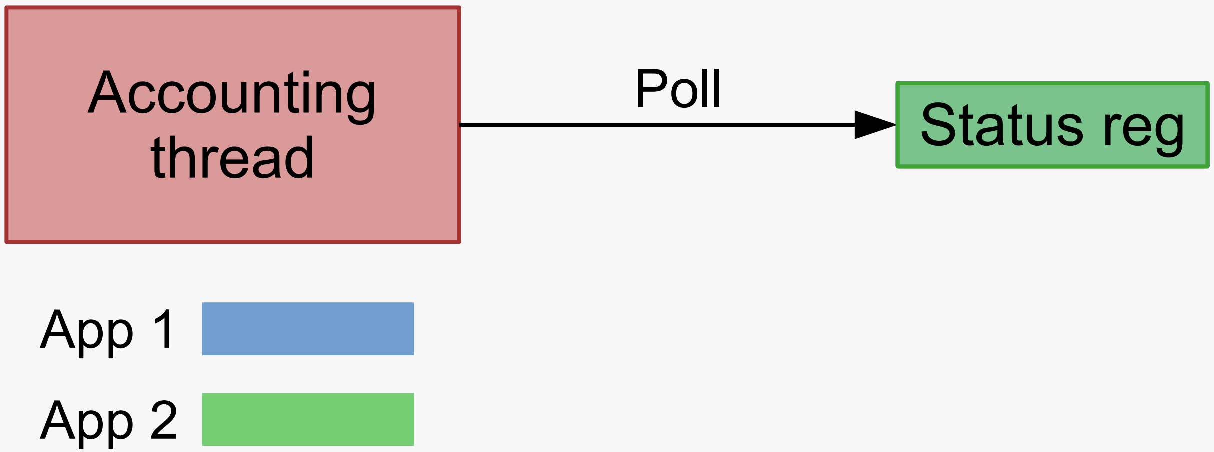
App 1

App 2

Accounting Thread



Accounting Thread



- Poll frequency must be faster than kernel length
- High CPU load ➔ Poll periodically

Scheduling

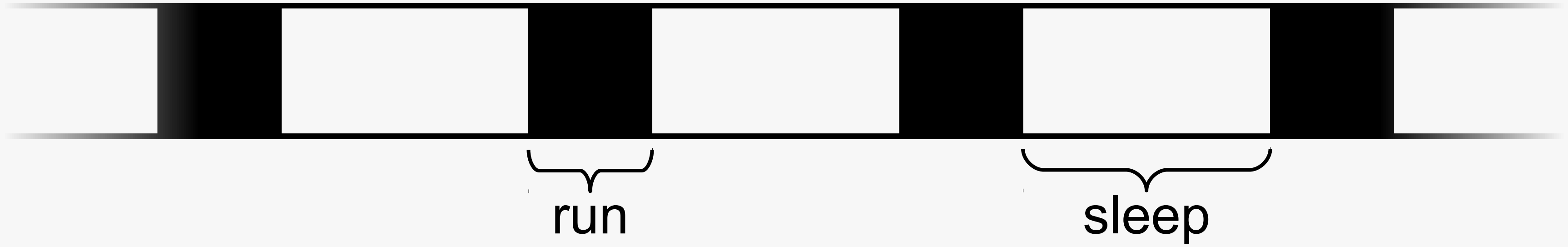
- Scheduling thread queries accounting thread
- Updates fair queuing counters
- Stops applications if necessary

- Different metric than NEON
  - NEON: Average kernel length
  - LoGA: Total GPU time consumed

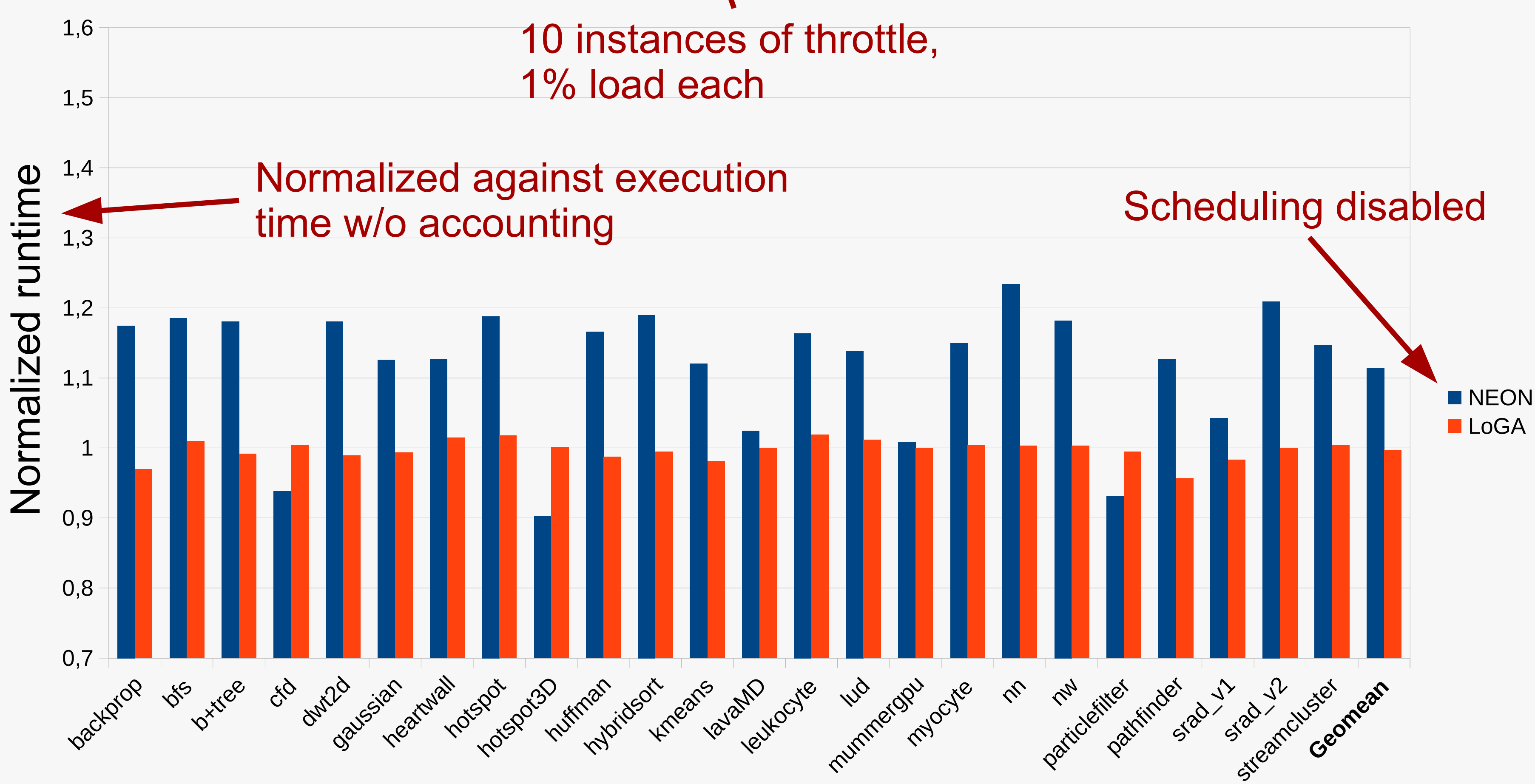
Benchmark Scenario

- Accounting overhead
- Accuracy of accounting (→ scheduling quality)

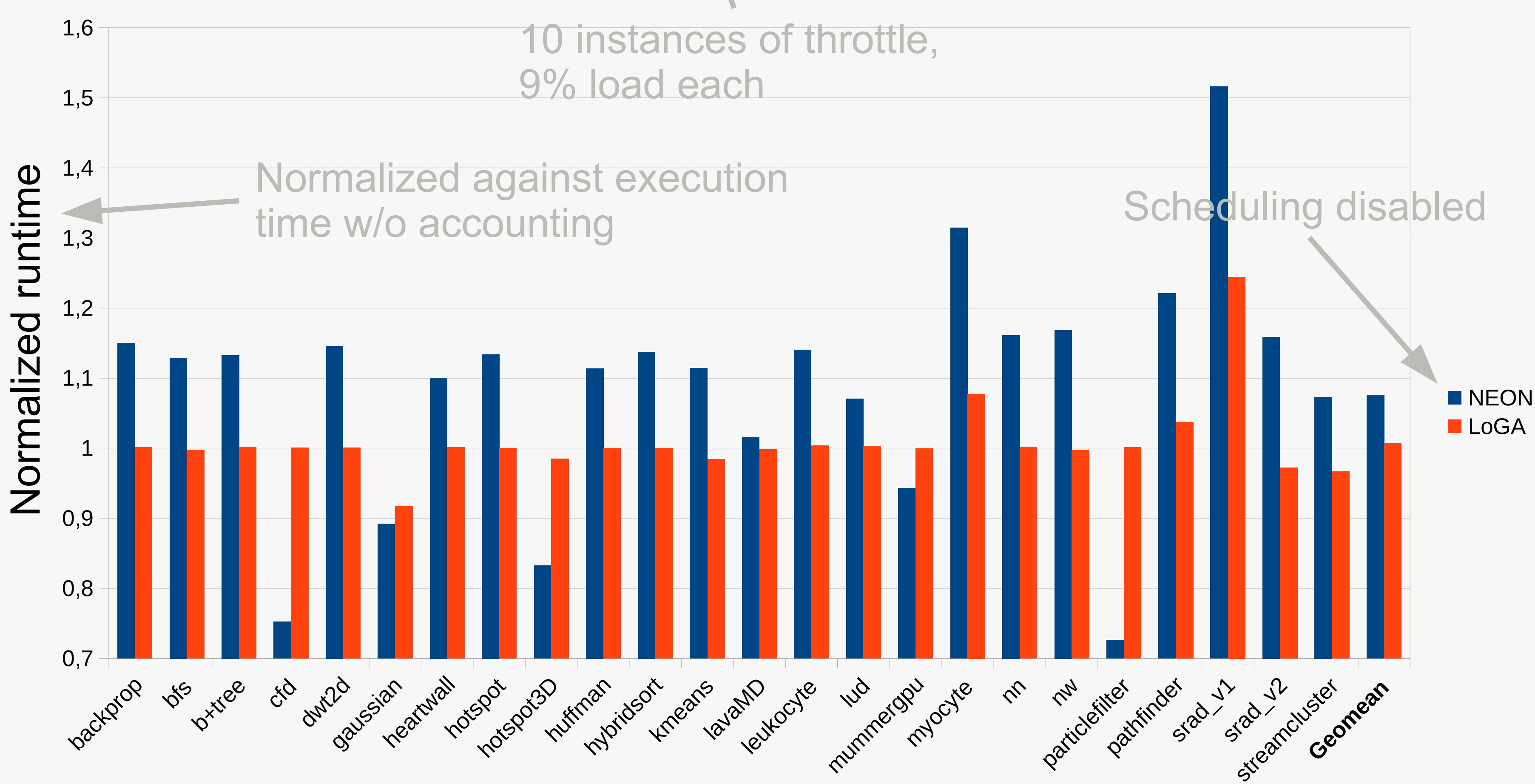
- Competing workload: Throttle
  - Creates well-defined GPU load



Accounting Overhead (10% load)



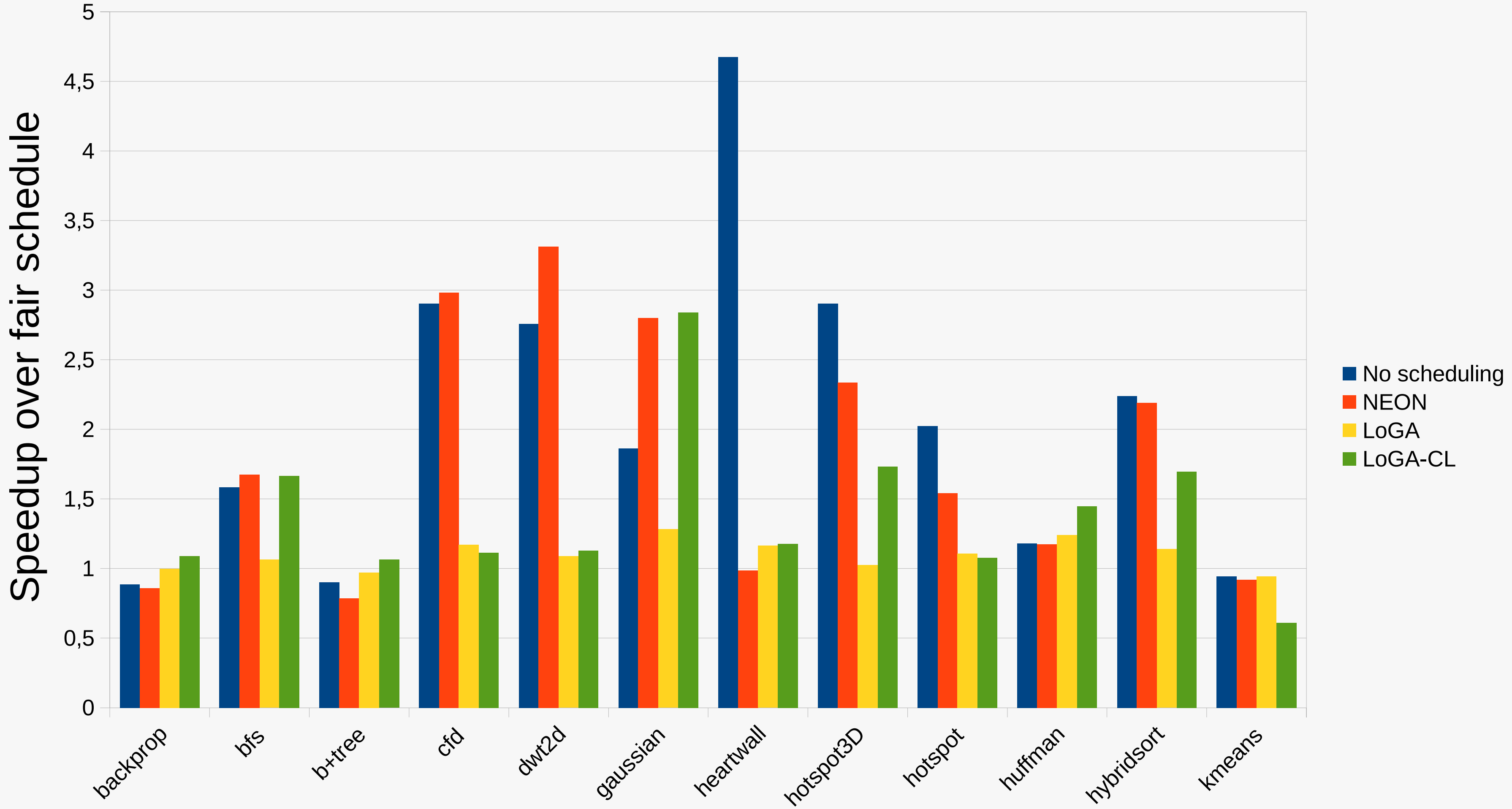
Accounting Overhead (90% load)



Fairness

- Goal: LoGA should not reduce fairness
- Problem: Which application runtime is fair?
- Measure total GPU time for each application
- Calculate optimal scheduling
  - Next slide: Speedup over fair schedule

Fairness: Results (4x throttle, 20% load each)



Conclusion

- Sharing beneficial if applications do not saturate the GPU
- Scheduler interference reduces sharing

- LoGA accounts GPU usage without overhead
  - Poll GPU status registers, detect context switches
  - Time between context switches as input for fair queuing

Finding Registers

- Envytools project has documented some registers
  - Unfortunately, far from complete
- Trial and error:
  - Run workload with known behavior
  - Dump register values
  - See which registers correlate with workload behavior
- Two registers identified:
  - ID of currently running GPU context
  - Activity status of entire GPU