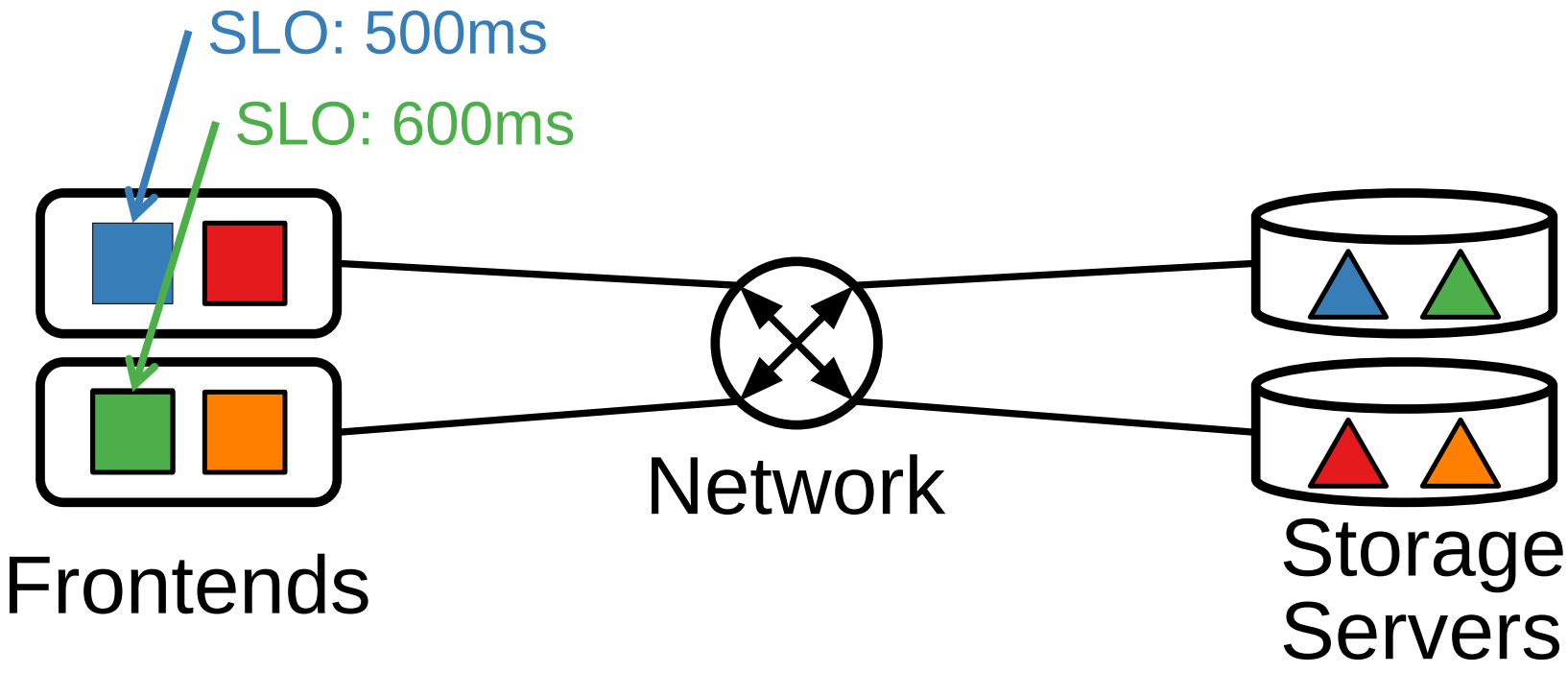
PriorityMeister: Tail Latency QoS for Shared Networked Storage

Timothy Zhu*

Alexey Tumanov* Michael A. Kozuch†
Mor Harchol-Balter* Greg Ganger*

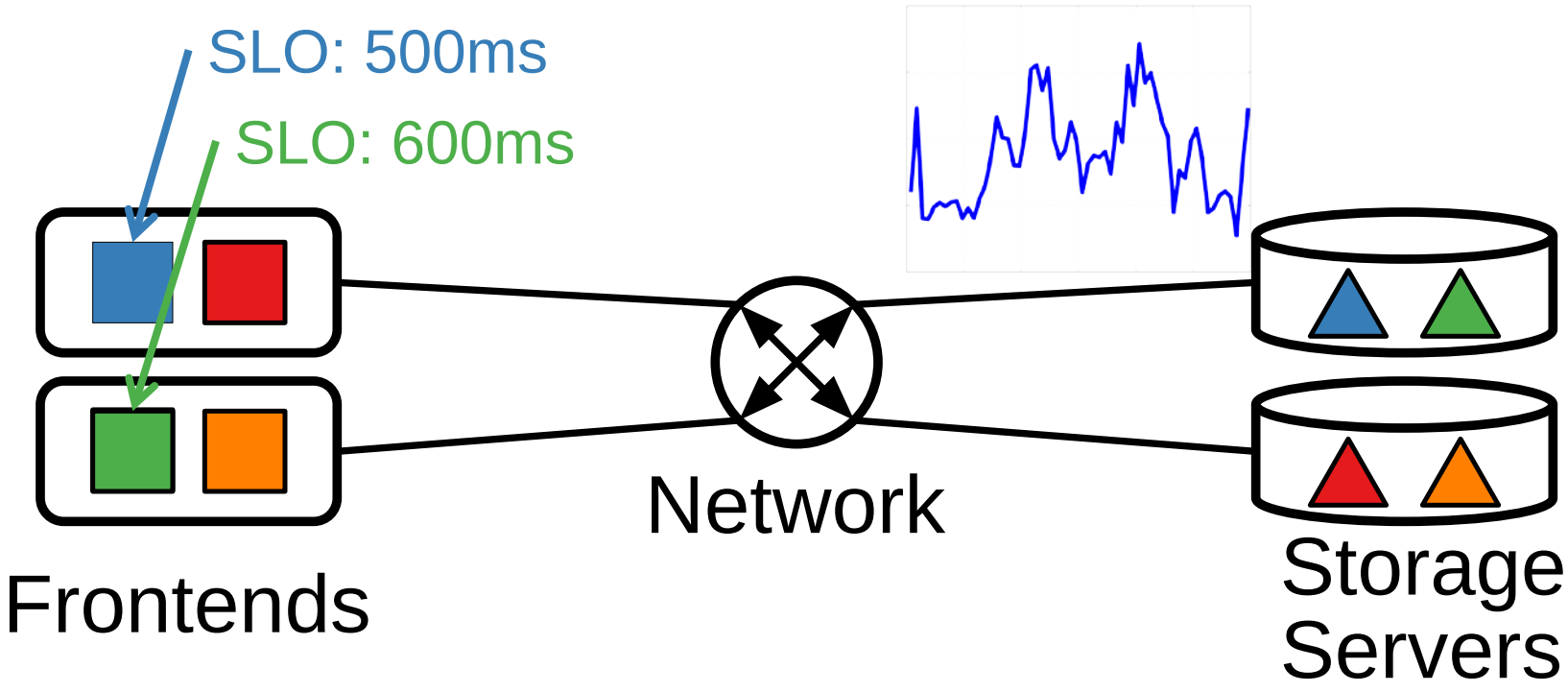
Carnegie Mellon University* Intel†

Problem statement



Goal: meet per-workload tail latency SLOs

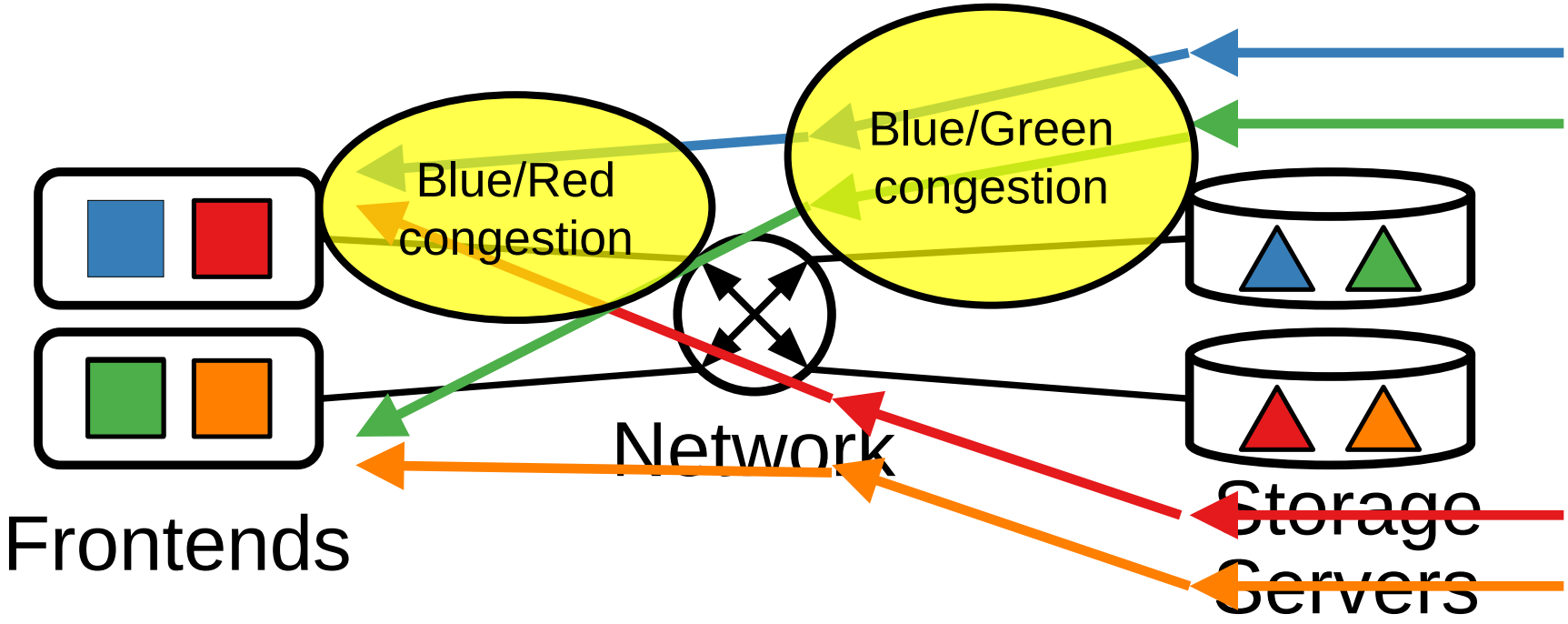
Challenge – burstiness



Goal: meet per-workload tail latency SLOs

- Bursts cause queueing for workloads sharing the system

Challenge – end-to-end performance



Goal: meet per-workload tail latency SLOs

- Bursts cause queueing for workloads sharing the system
- Latency is affected by each of the resources
- Workloads congest at different resources

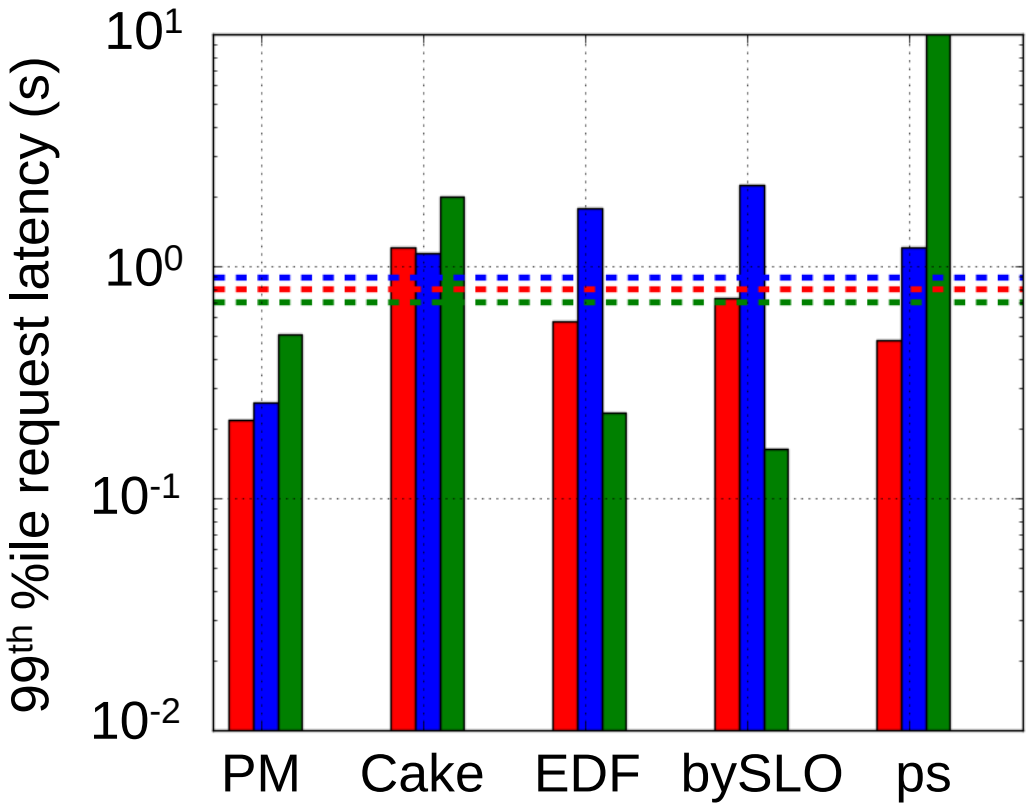
Solution – priority & rate limiting

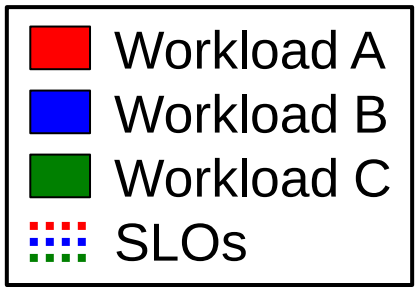
- Priority
  - Purpose: reduce latency for workloads that care most
  - Simple mechanism, applies to storage & network

- Rate limiting
  - Purpose: prevents starvation of low priority workloads
  - Characterizes limits of workload behavior

Automatically assign priority & rate limits to meet SLOs

PM meets tail latency SLOs





Scheduling policies:

- PM: PriorityMeister
- Cake: reactive feedback-control
- EDF: earliest deadline first
- bySLO: prioritize by SLO
- ps: proportional sharing

PM accounts for workload behavior to better meet SLOs