μTune: Auto-Tuned Threading for OLDI Microservices

Akshitha Sriraman, Thomas F. Wenisch
University of Michigan

On-Line Data Intensive Applications

Impact of Threading-Induced Overhead

Monoliths (>100 ms SLO) ➔ Microservices (sub-ms SLO)

Impact: Minor for monoliths & major for microservices

Example:

Threading Impact on Mid-Tier

Mid-Tier: Heavily impacted by threading
- Server & client
- Fans queries to many leaves
- RPC layer interactions dominate compute

Contributions

- A taxonomy of threading models
  - Structured understanding of threading implications
  - Reveals tail inflection points across load
  - Peak load-sustaining model is subpar at low load
- μTune:
  - Uses tail inflection insights to optimize tail latency
  - Tunes model & thread pool size across load
  - Simple interface: Abstracts threading from RPC code

Taxonomy of Threading Models

μTune: Automatic Load Adaptation

Abstracts threading boiler-plate code from RPC code

No single threading model works best at all loads

μTune Under Steady-State Load

μTune converges to best threading model and thread pool size to improve tail latency by up to 1.9x over static peak load-sustaining threading model with < 5% mean overhead

Acknowledgement