Octopus-Man

QoS-Driven Task Management for Heterogeneous Multicore in Warehouse Scale Computers

Vinicius Petrucci (UFBA)*, Michael Laurenzano, John Doherty, Yunqi Zhang (UMich), Daniel Mossé (PITT), Jason Mars, Lingjia Tang (UMich)

** Work done while the author was a post-doc at UMich*

UFBA

The 21st IEEE International Symposium on High Performance Computer Architecture (HPCA)
February 2015 — Bay Area, CA

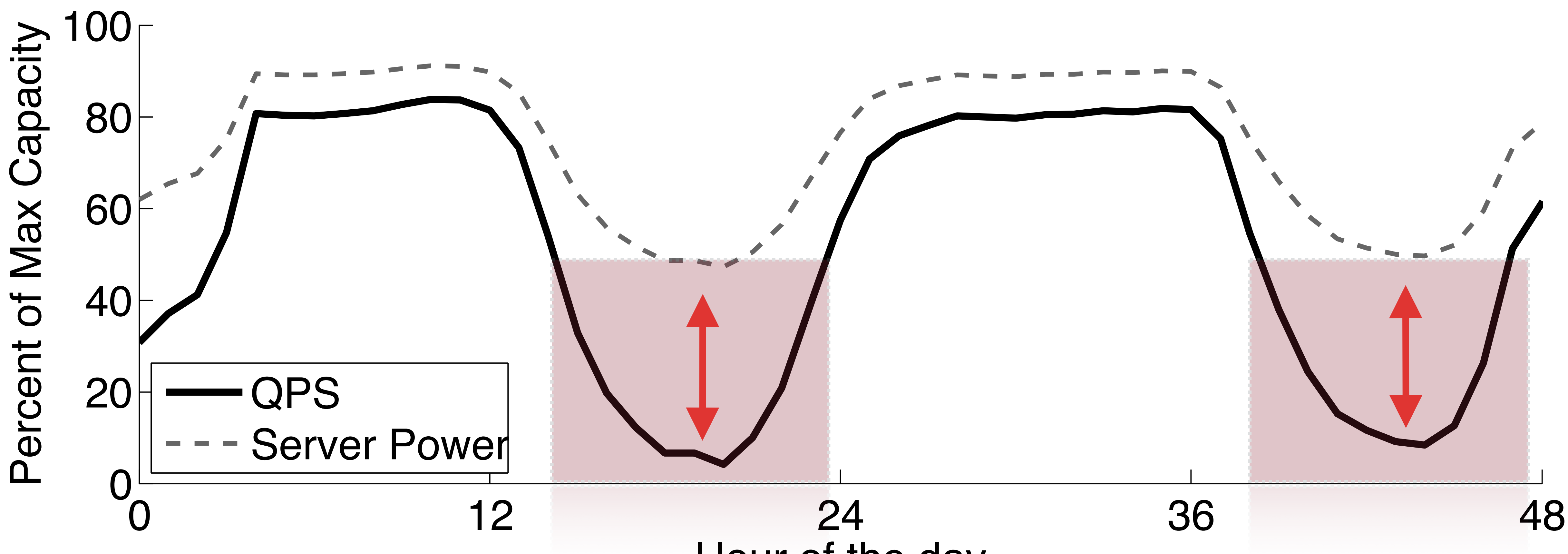
Warehouse Scale Computers (WSC)

Computation shifting to the "cloud"

Google data center in Douglas County, Georgia

Typical WSC workload

Load fluctuation and power consumption of Web-search running on Google servers *



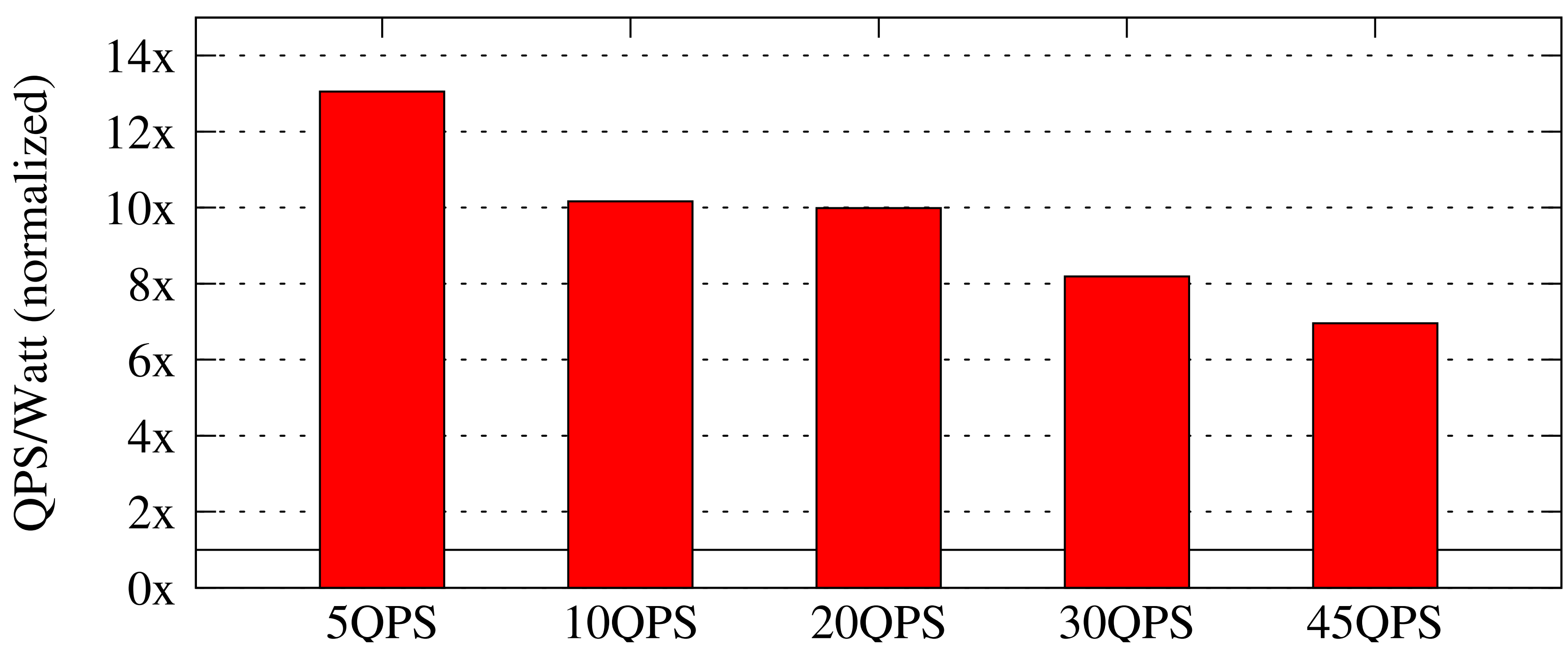
(QPS = Queries Per Second)

** Meisner et al. Power management of online data-intensive services. ISCA 2011*

Energy consumption is not proportional to the amount of computation!

Opportunity: heterogeneous multicores

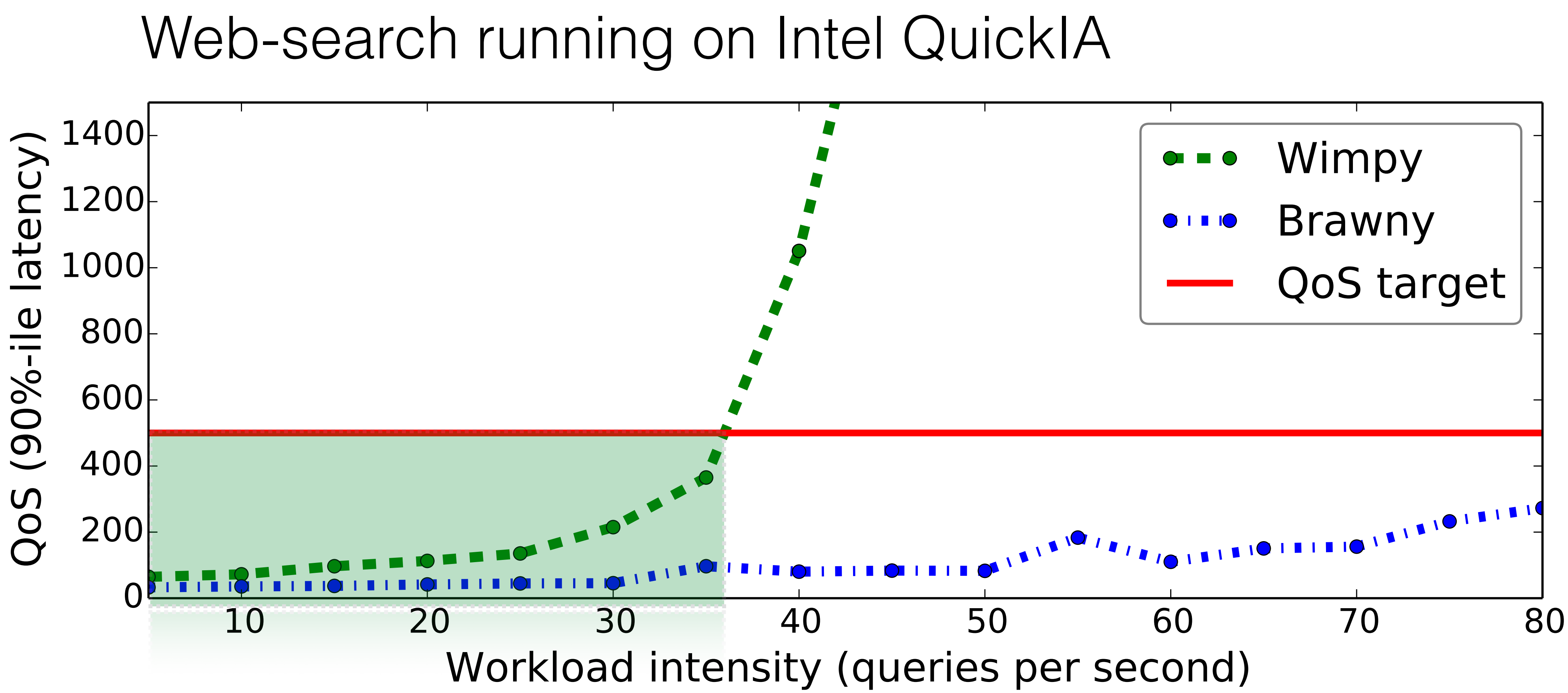
- Heterogeneous multicore (Wimpy + Brawny cores)
  - Power efficiency improvement
- **Real system** evaluation on Intel QuickIA (Atom + Xeon)



Wimpy cores can be 7-13x more power-efficient than Brawny cores

Opportunity: heterogeneous multicores

- What about performance (e.g., tail latency)?

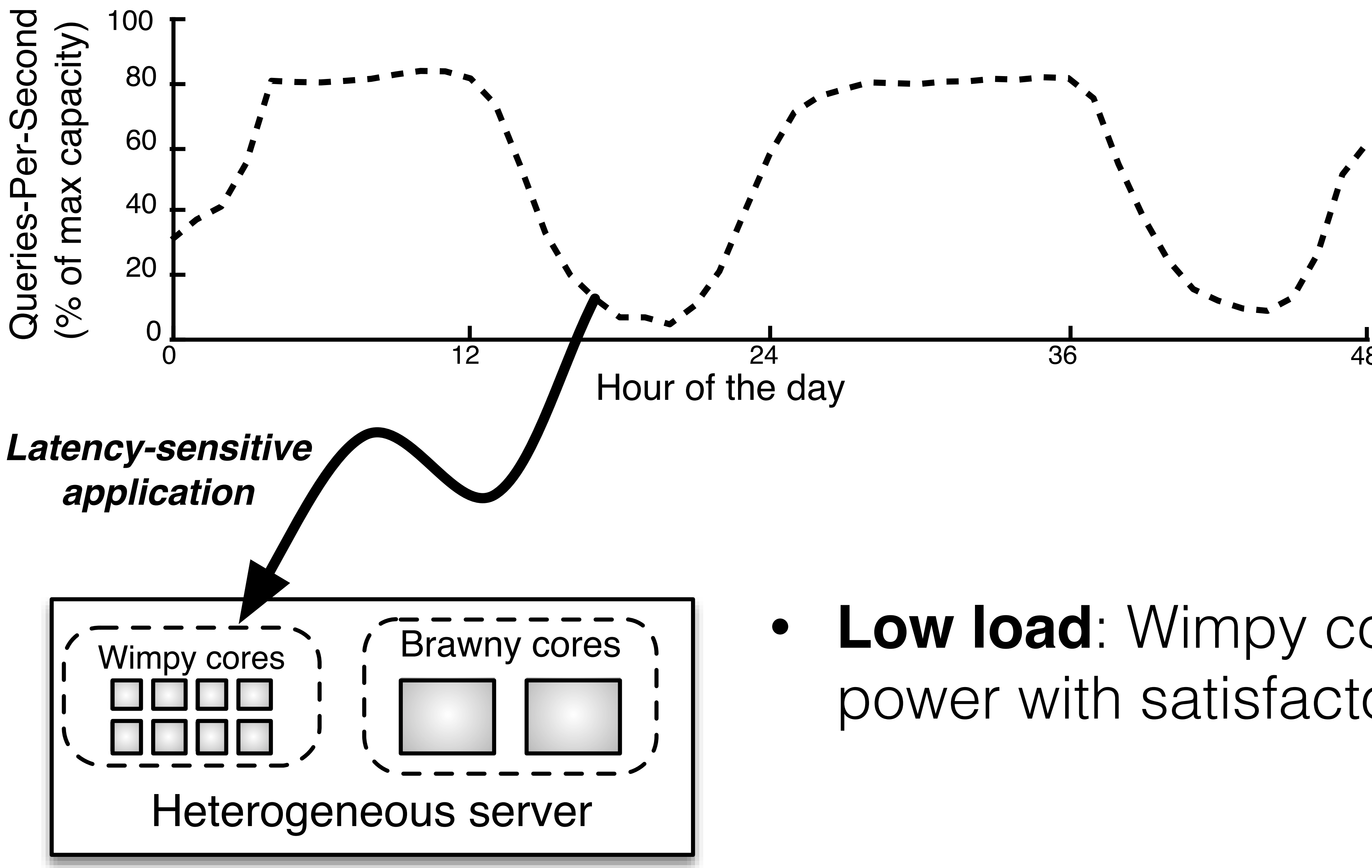


Brawny cores achieve lower latency at all load levels

But wimpy cores can still meet the QoS at low load using much less power!

Opportunity: heterogeneous multicores

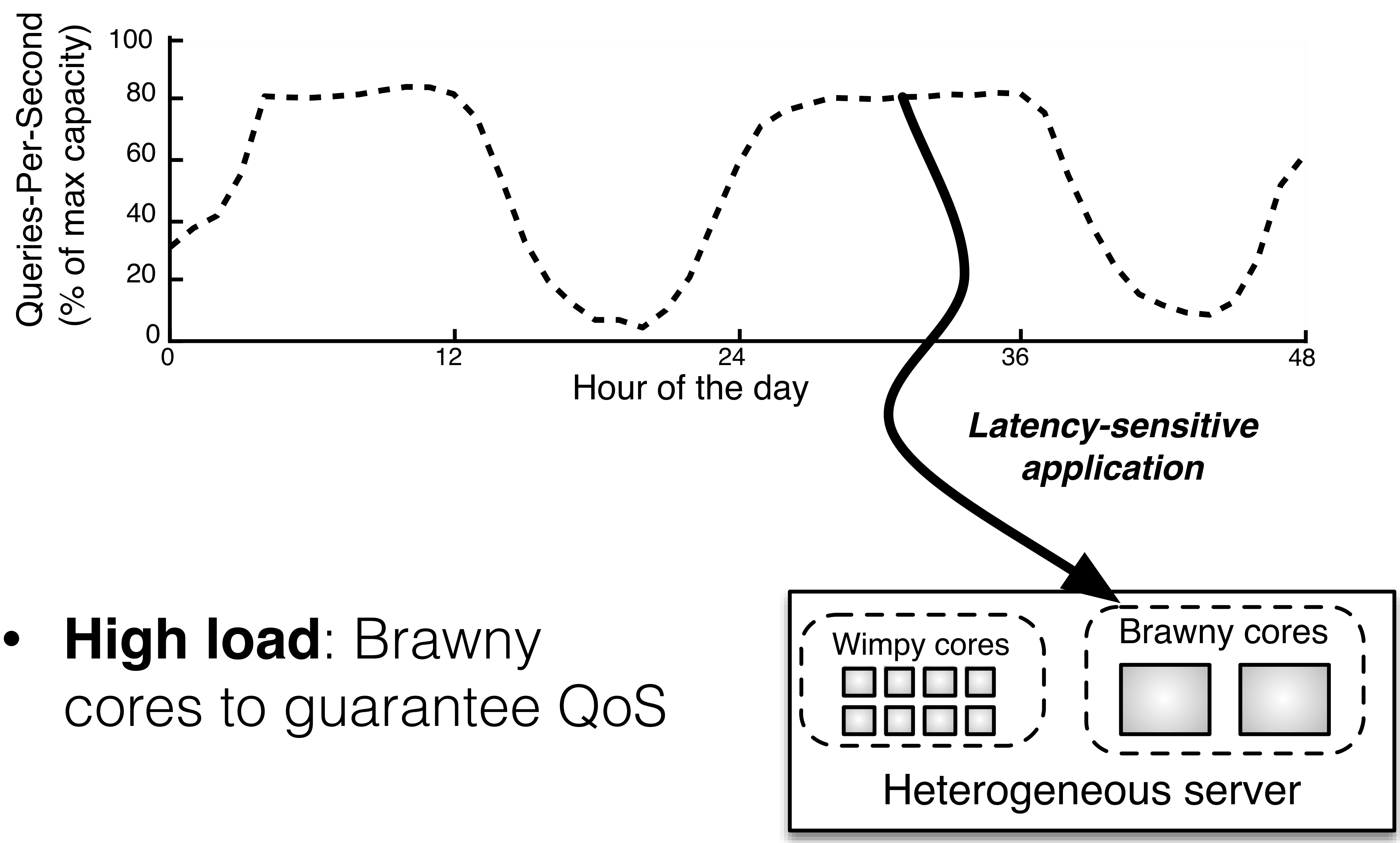
Insight: Exploit *load fluctuation* to improve energy efficiency and meet QoS



- **Low load**: Wimpy cores to reduce power with satisfactory QoS

Opportunity: heterogeneous multicores

Insight: Exploit *load fluctuation* to improve energy efficiency and meet QoS



- **High load**: Brawny cores to guarantee QoS

Octopus-Man: Goal

- To **guarantee quality of service** (e.g., bounding tail latency) while **maximizing energy efficiency**

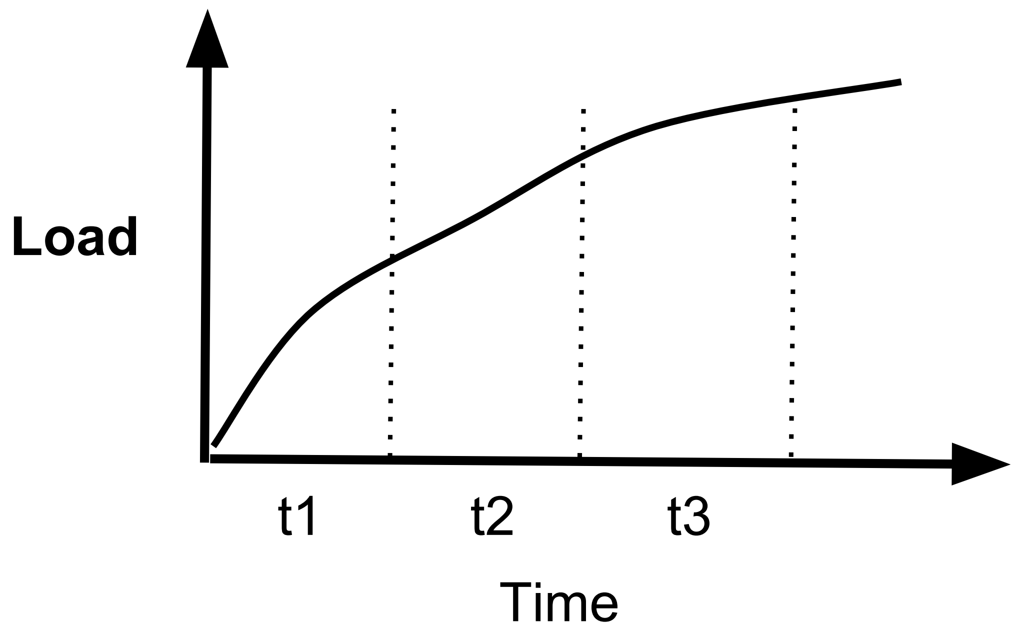
... but this is **not** a trivial task!

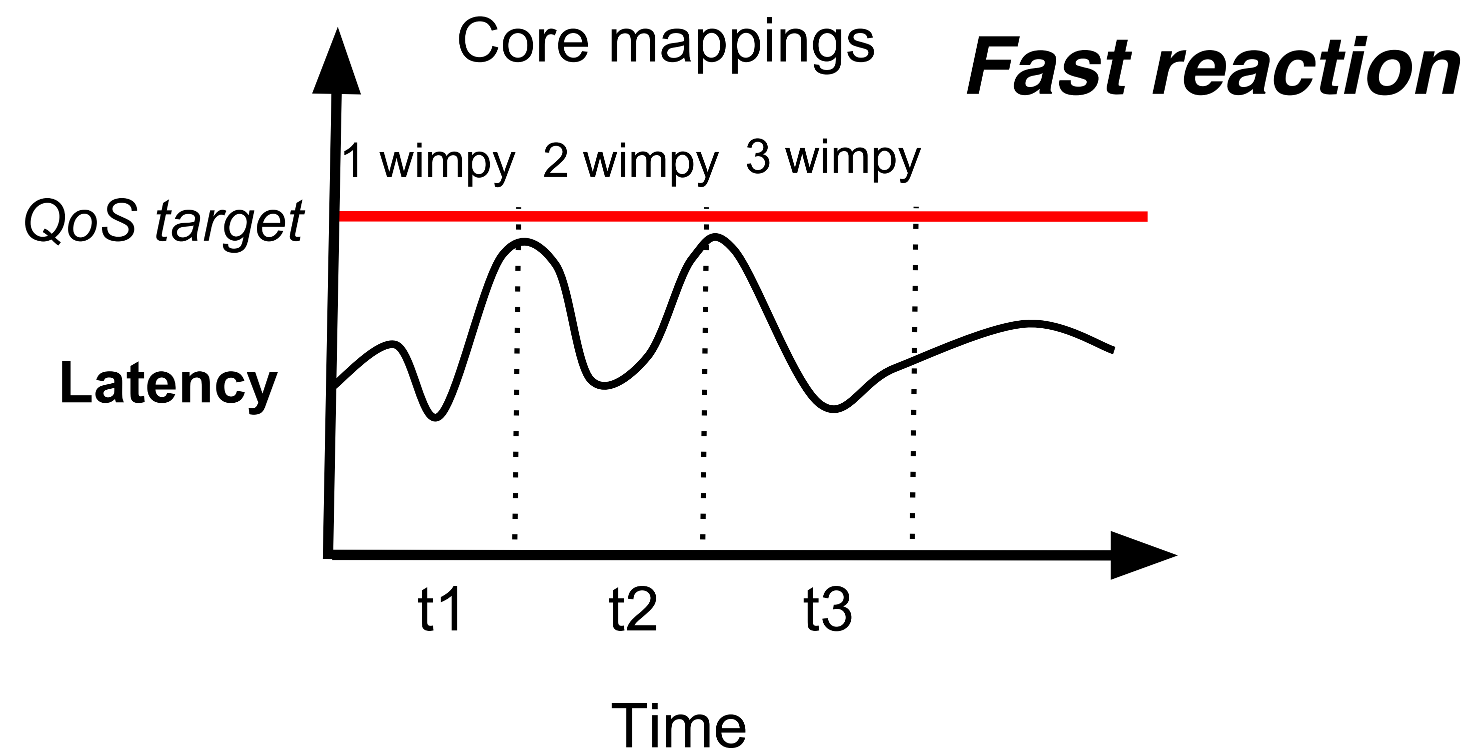
Naive design of tasking mapping/migrations on heterogenous multicore can cause ***significant QoS violations***

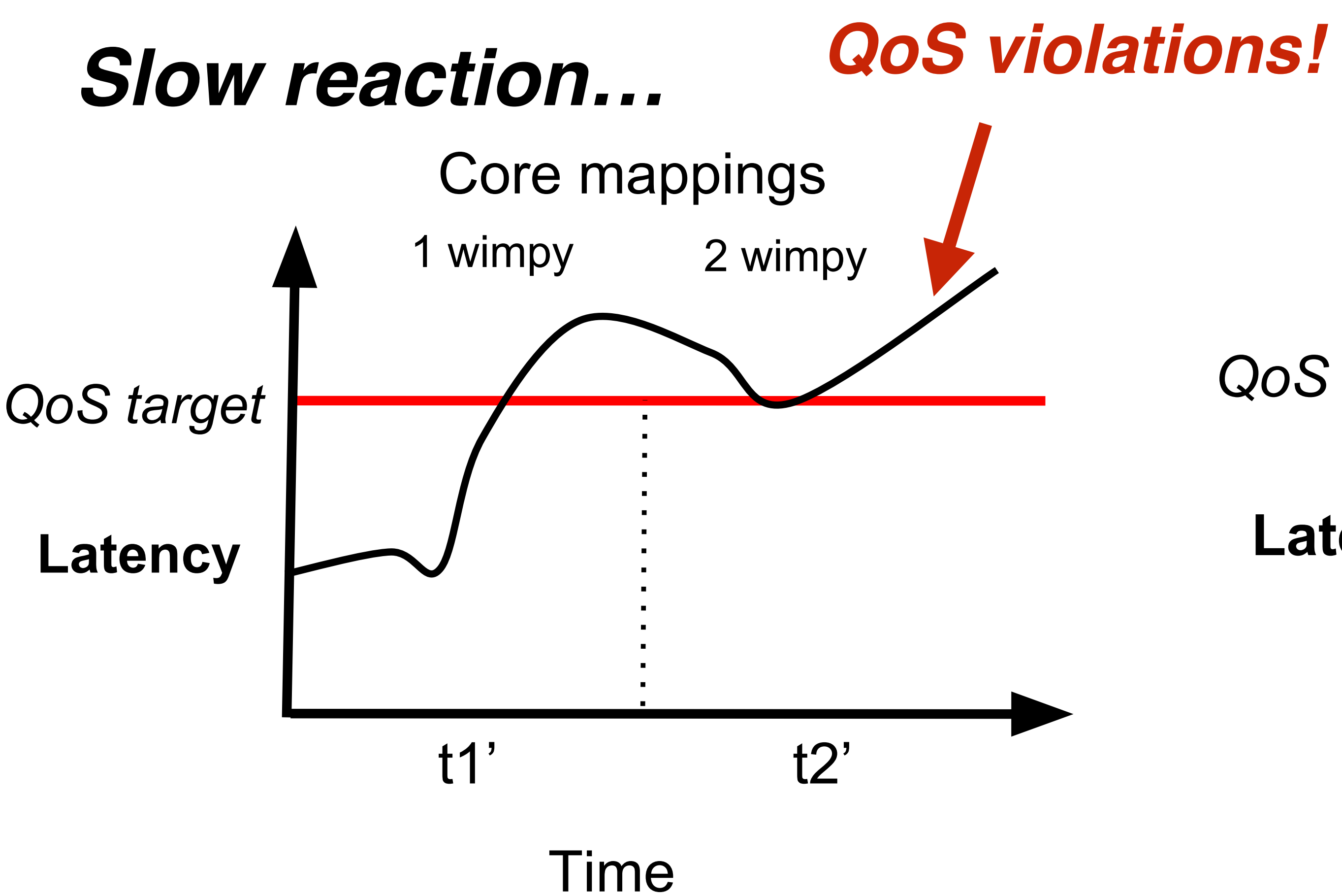
Octopus-Man: Challenges

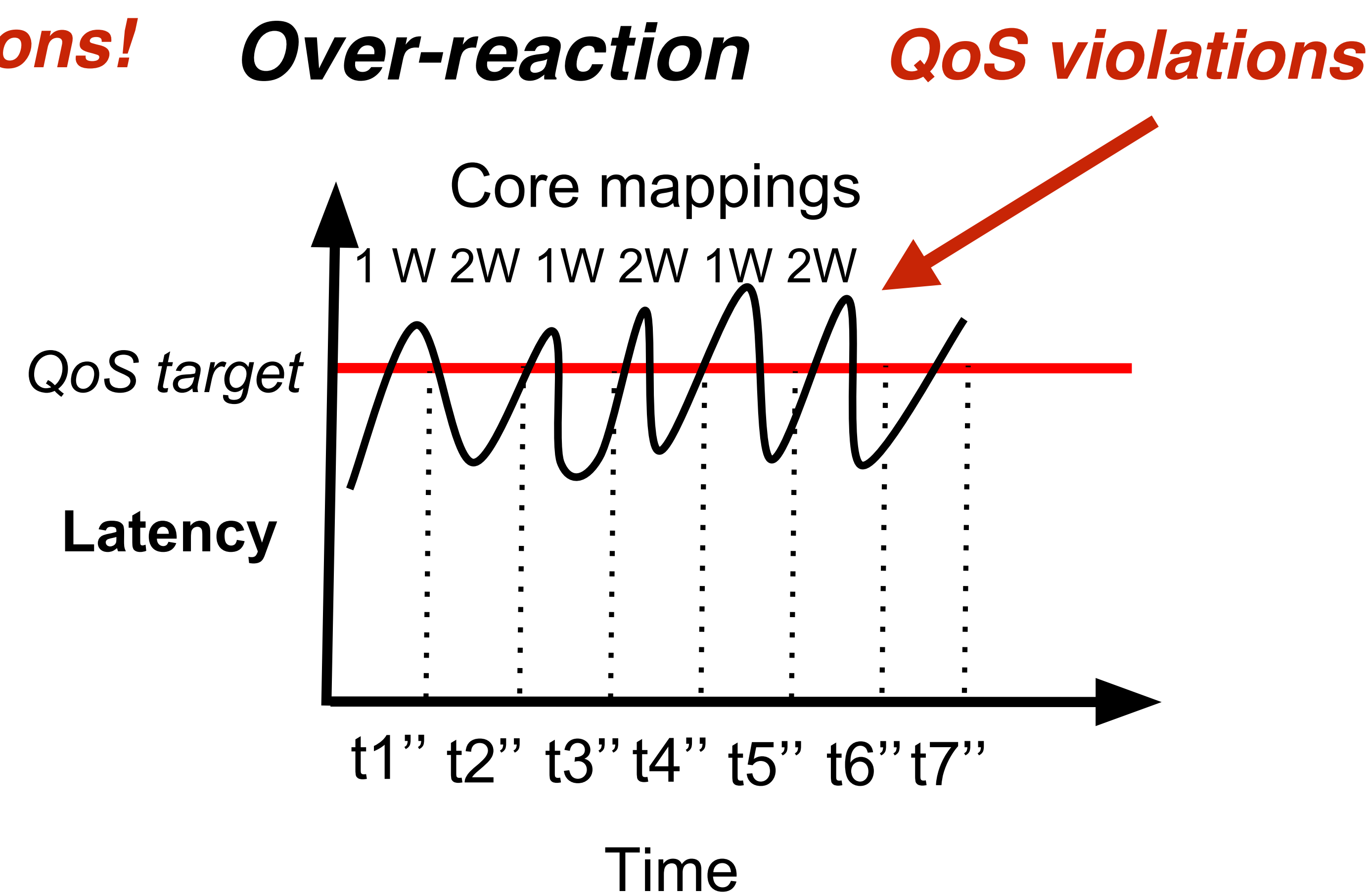
- Tension between **responsiveness** and **stability**
  - **Responsiveness**
    - *react quickly* to capture load fluctuations and migrate tasks accordingly to meet QoS
  - **Stability**
    - do not *over-react* because it can cause oscillatory behavior and hurt the QoS

Responsiveness and stability

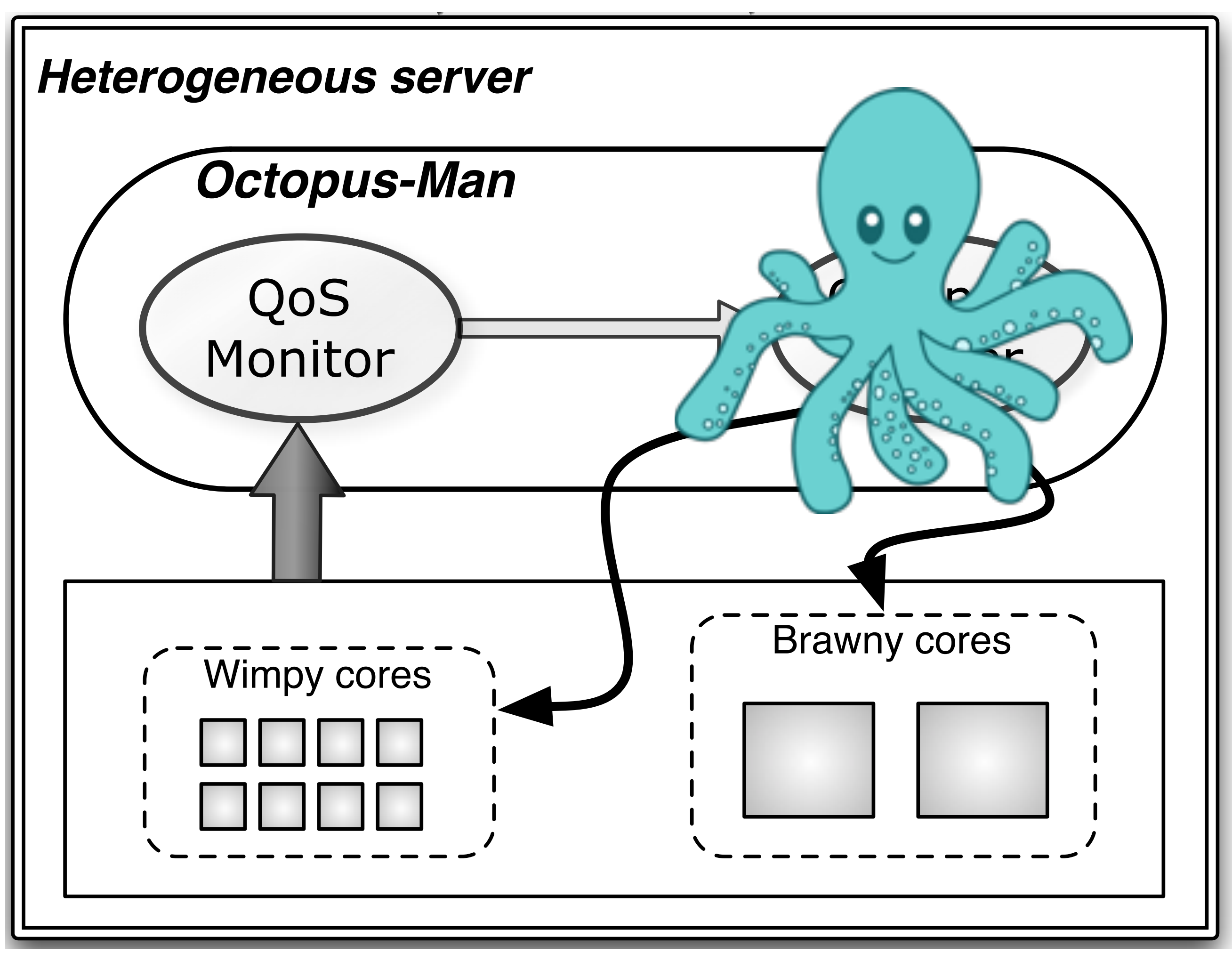








Octopus-Man: Solution



- **Octopus-Man monitor**
  - Application-level latency monitoring
- **Octopus-Man Mapper**
  - Task-to-core management for QoS guarantee and energy efficiency

Octopus-Man Mapper: Designs

1) PID control system

- **pros**: well-known control methodology
- **cons**: parameter tuning via extensive offline app profiling

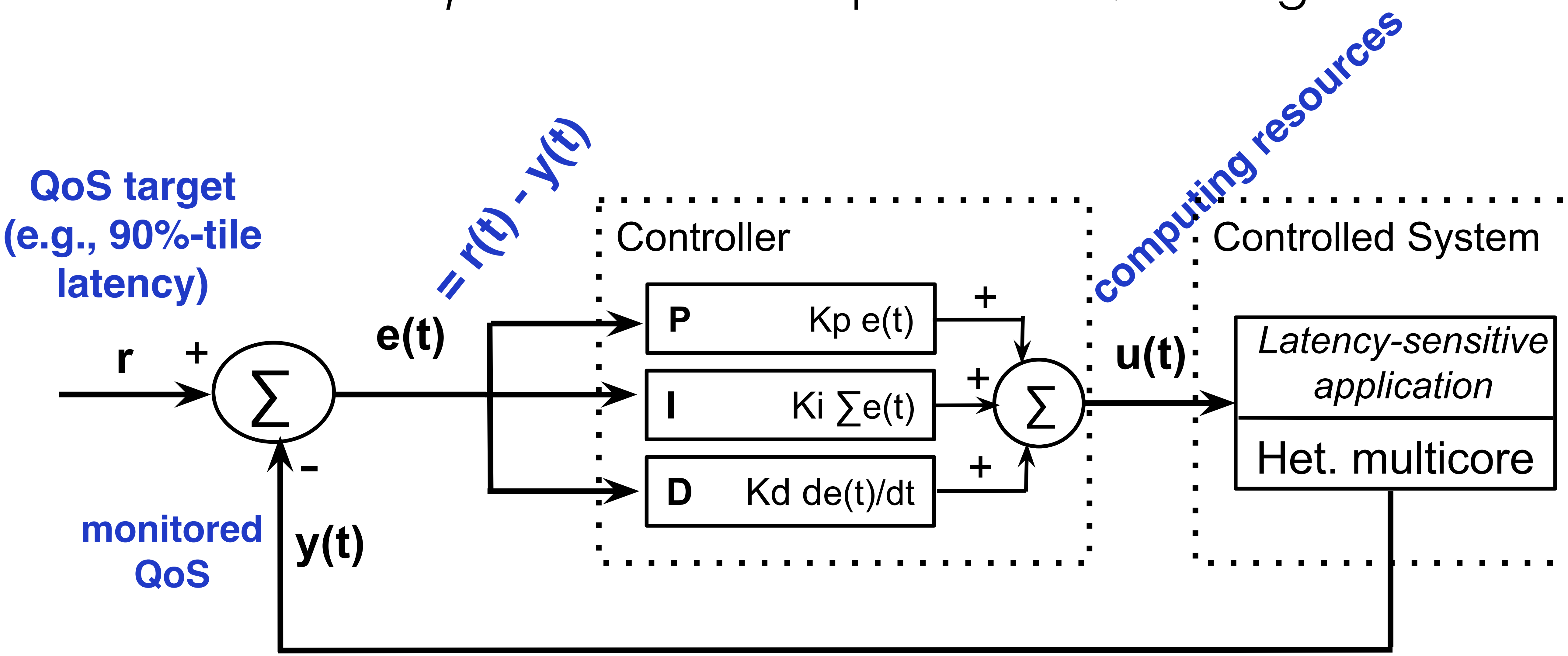
2) Deadzone-based control system

- **pros**: simple online scheme based on QoS thresholds
- **cons**: sensitive to threshold parameter selection

- Can either effectively provide high QoS while maximizing energy efficiency?
  - Responsiveness and Stability

Design 1: PID control system

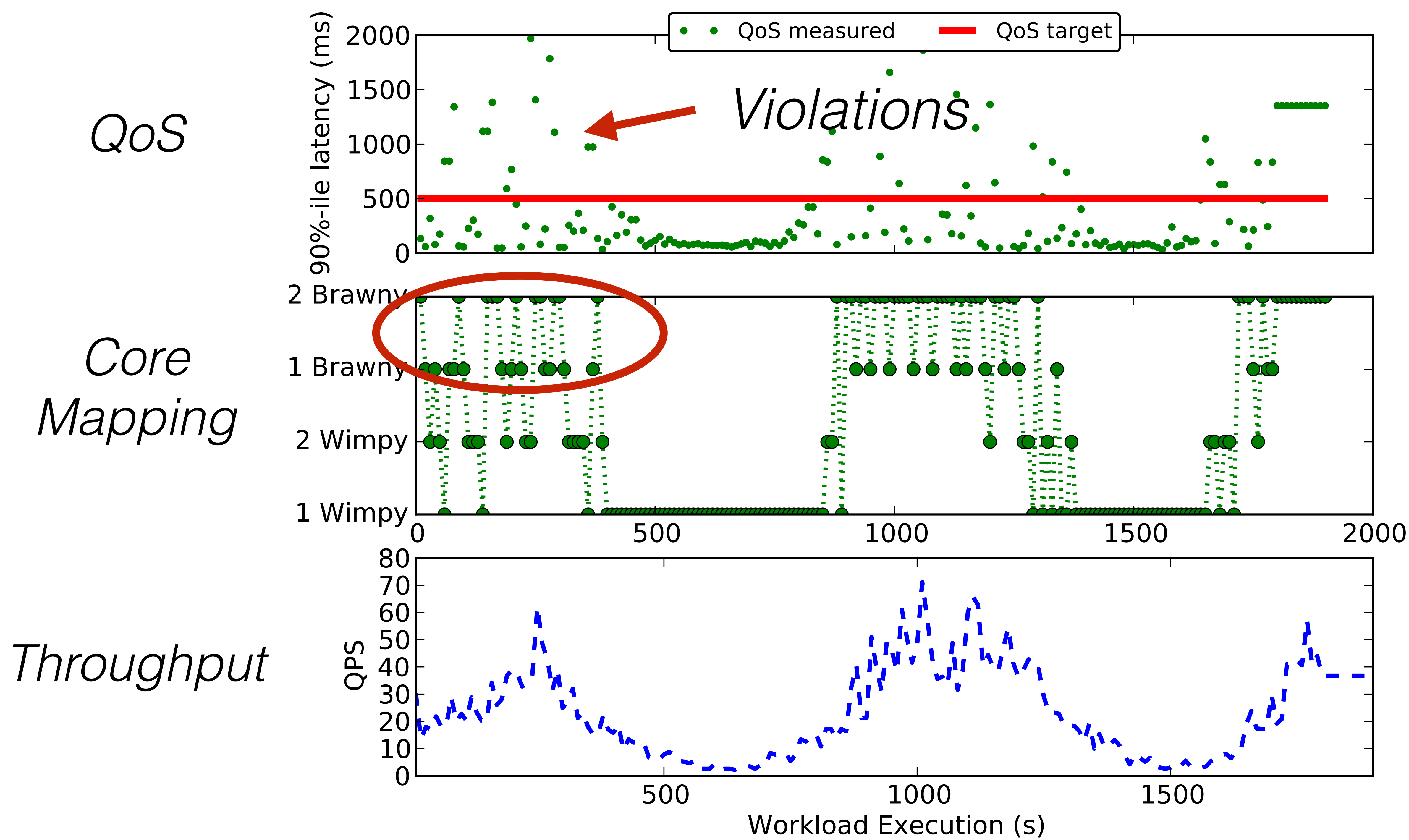
GOAL: To keep the **controlled system** running *as close as possible* to its specified QoS target



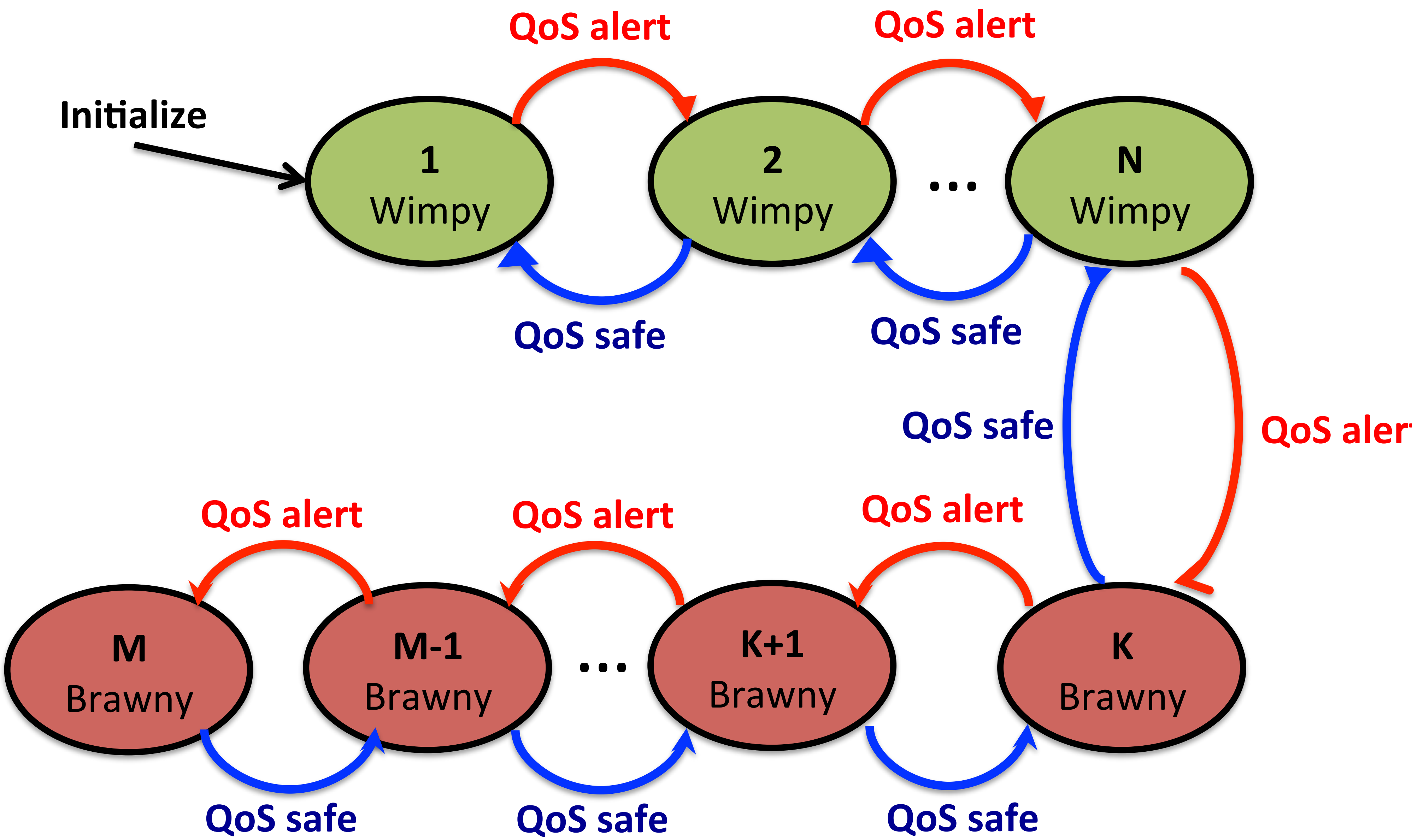
PID Control Mapping

- Task-to-core mapping
  - Mapping from the continuous PID output to a discrete task-core mapping
- Parameter selection/tuning
  - Classical control system method, root locus (Hellerstein et al. 2004), is used to determine **Kp, Ki, Kd** parameter
    - Responsiveness and stability

PID control: web-search



Design 2: Deadzone State Machine



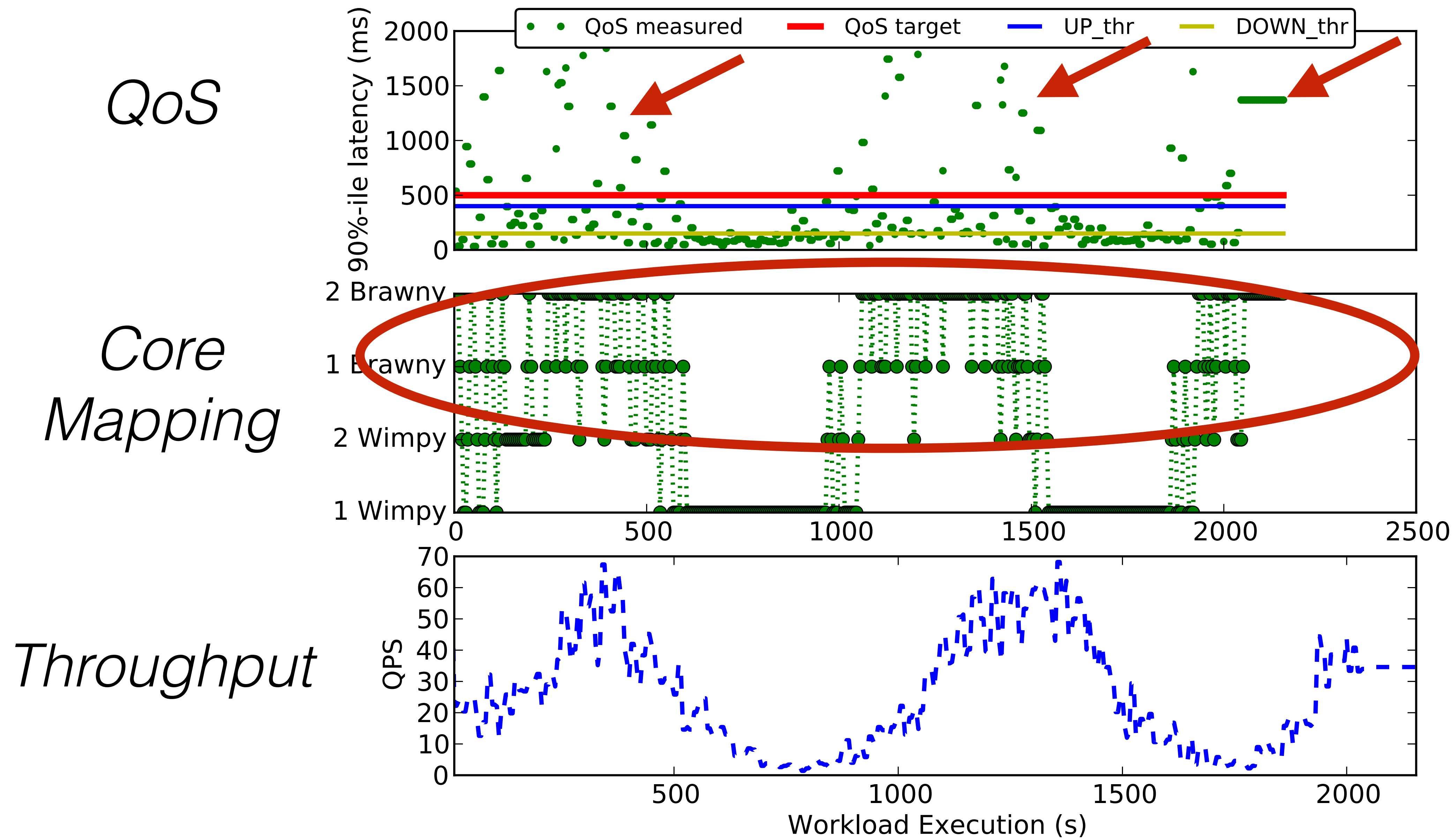
QoS alert: QoS variable > QoS target * UP_THR

QoS safe: QoS variable < QoS target * DOWN_THR

The deadzone thresholds impact the stability of the mapping algorithm!

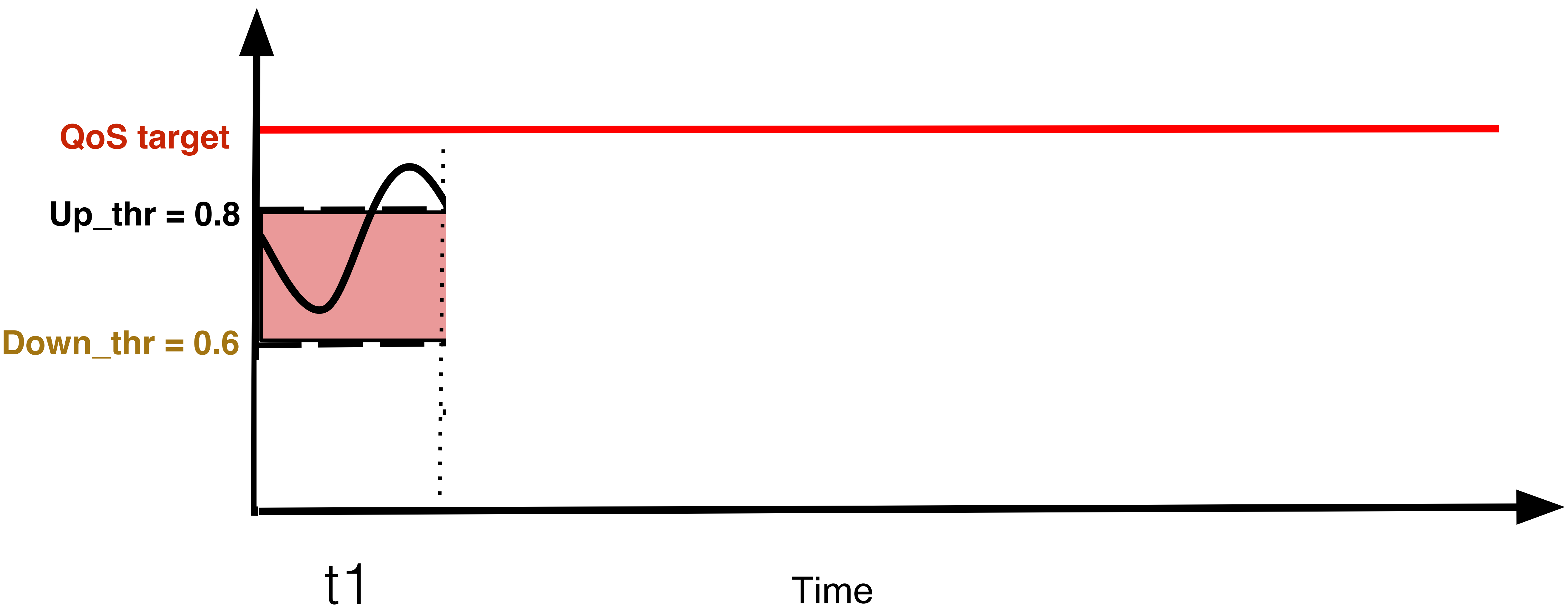
Stability: selecting deadzone parameters

Web-search execution with UP thr=0.8, DOWN thr=0.3

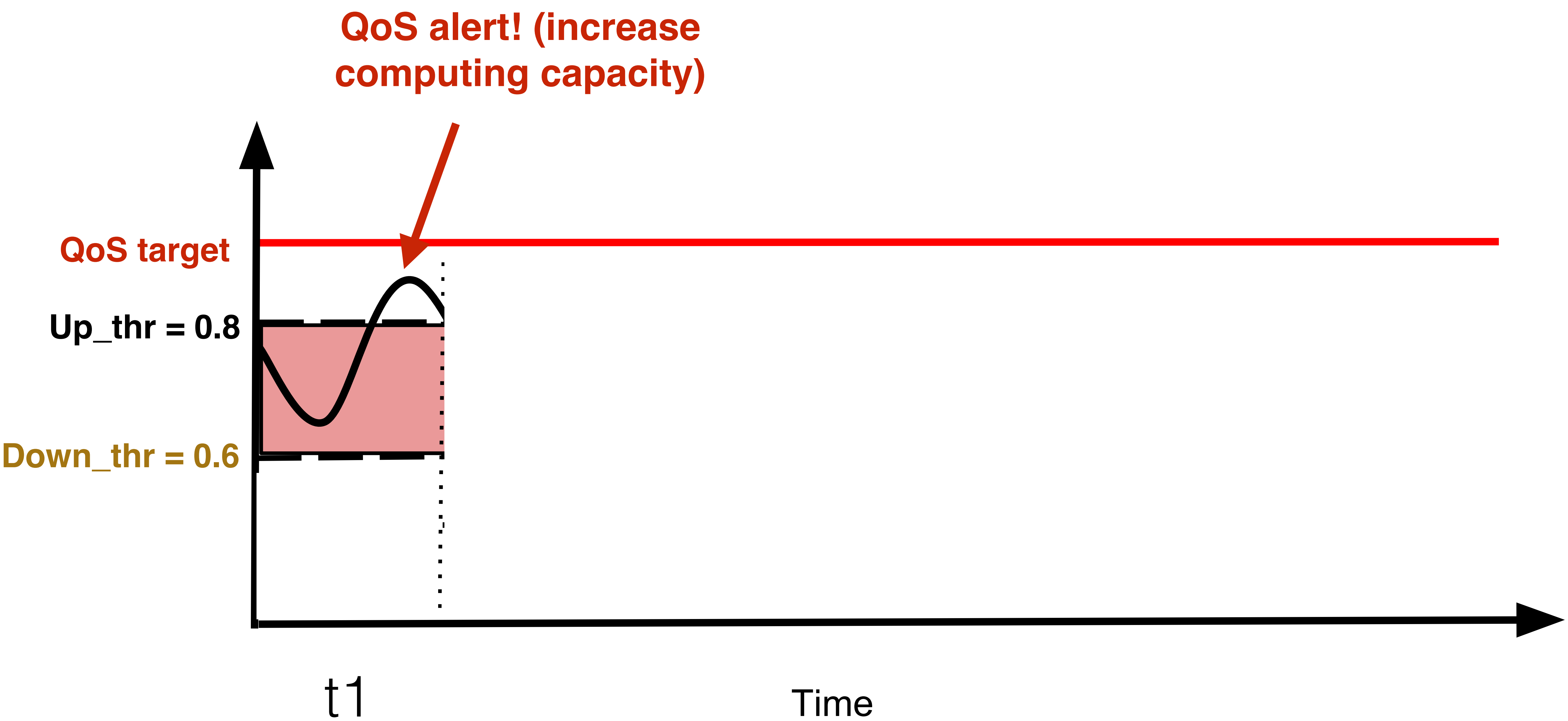


High QoS violations occur due to oscillatory behavior!

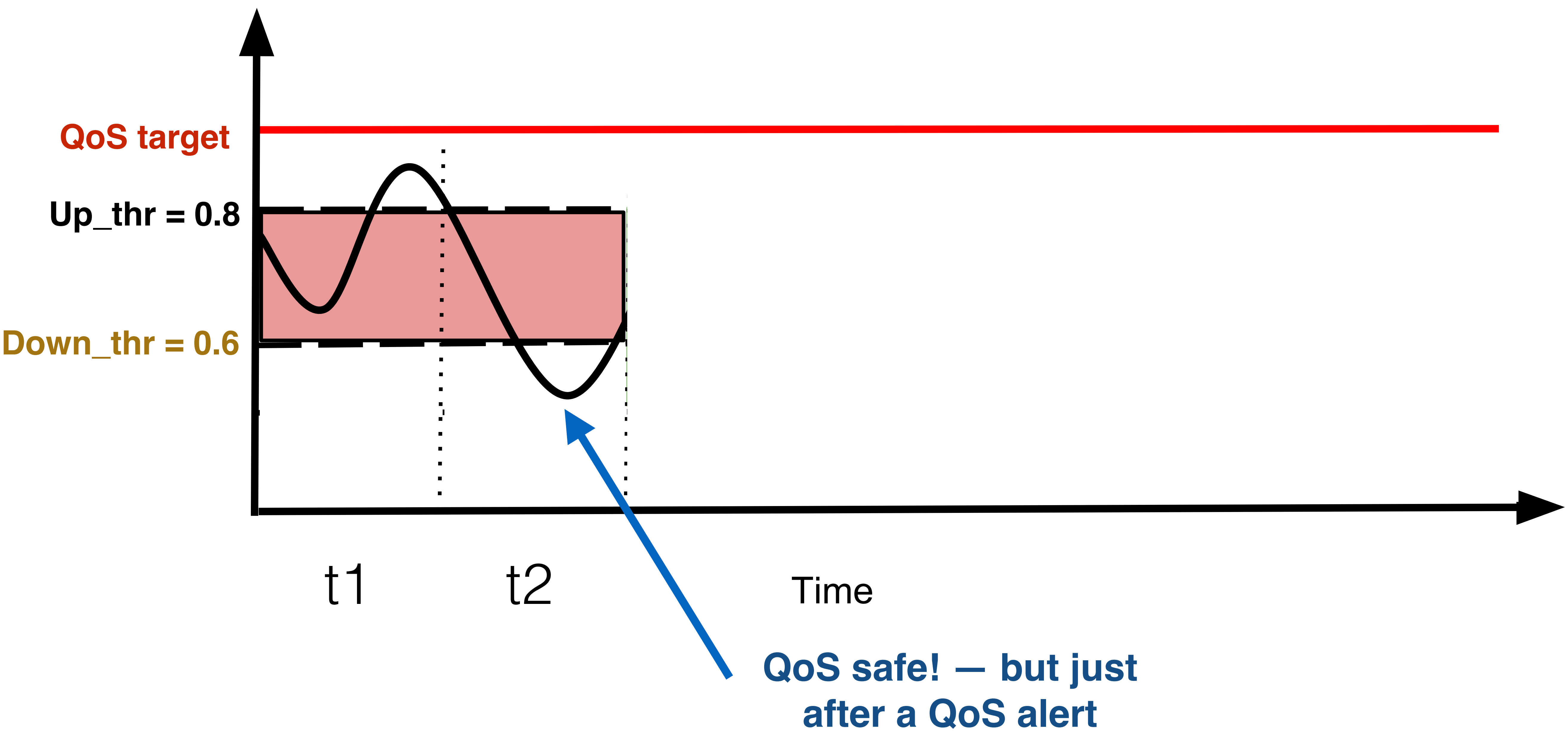
Solution: Dynamic deadzone selection



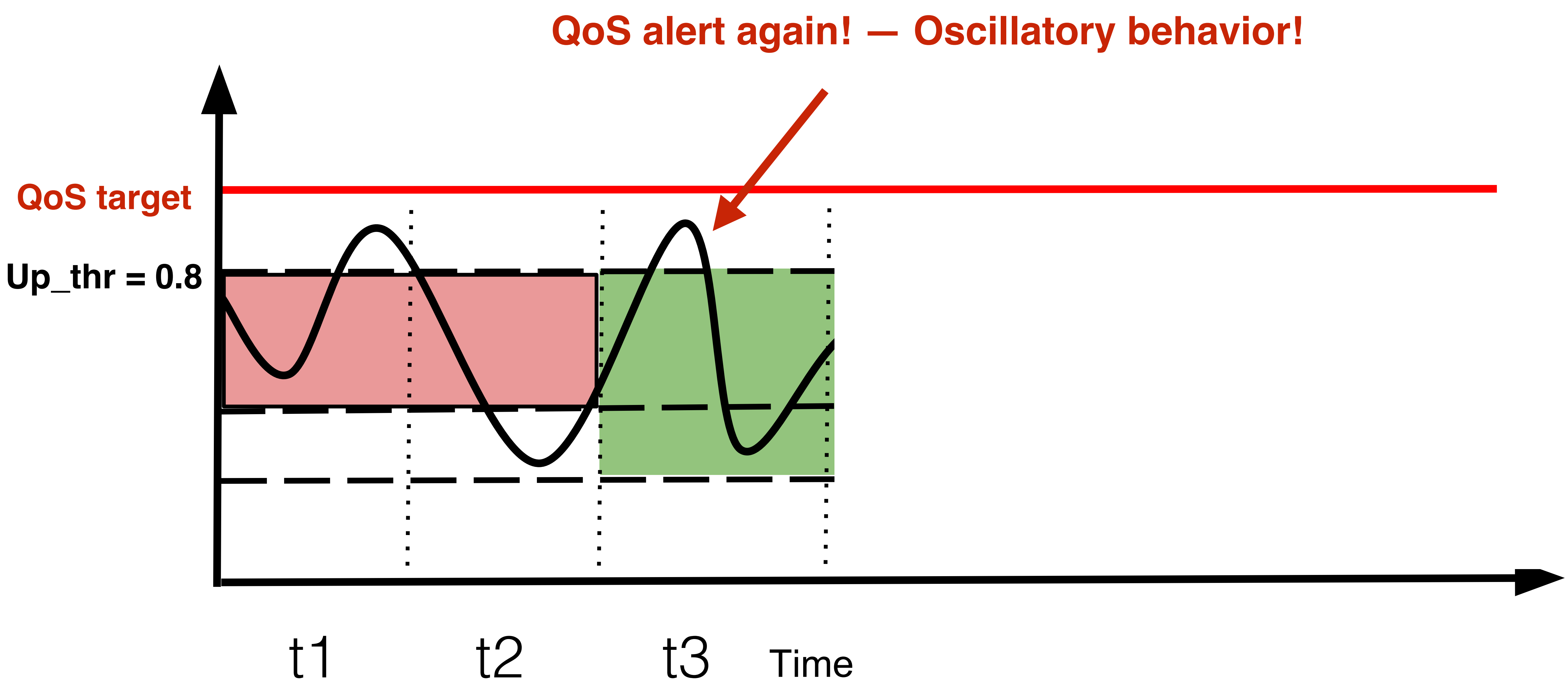
Solution: Dynamic deadzone selection



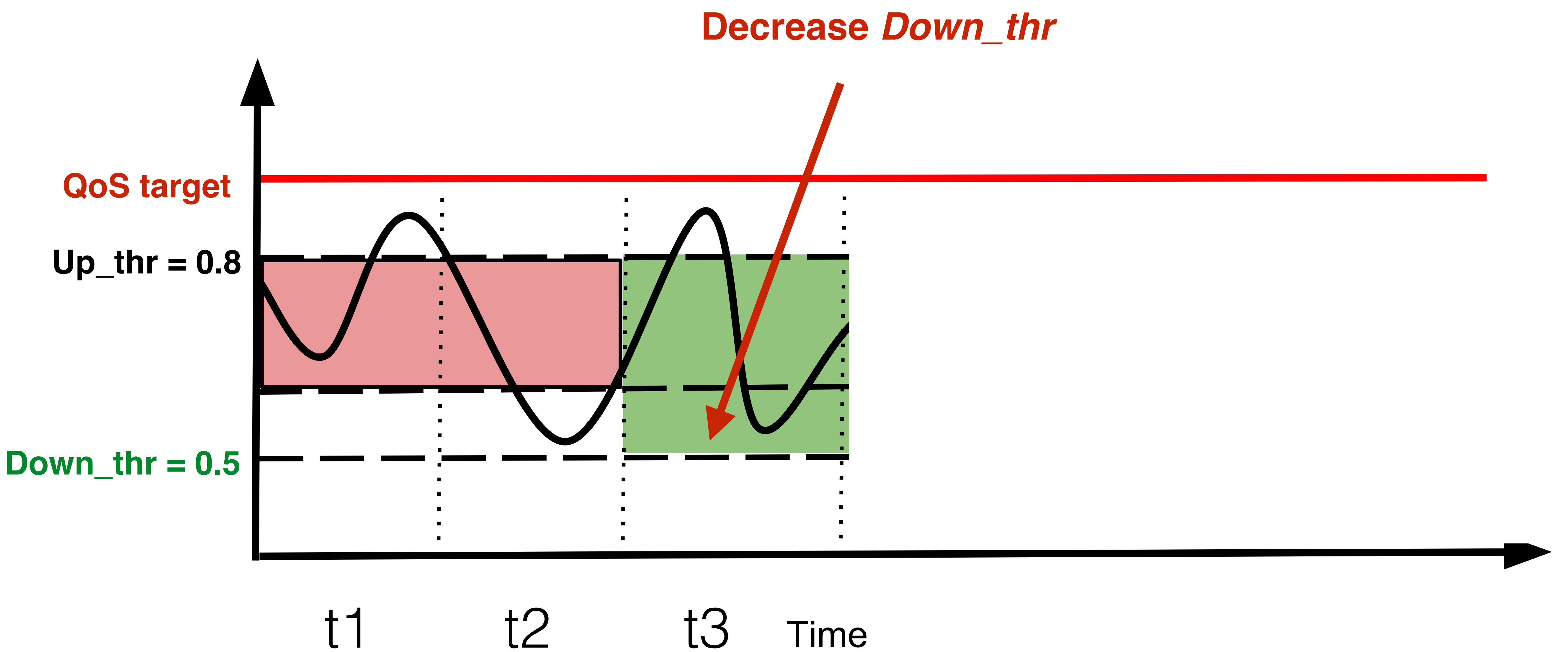
Solution: Dynamic deadzone selection



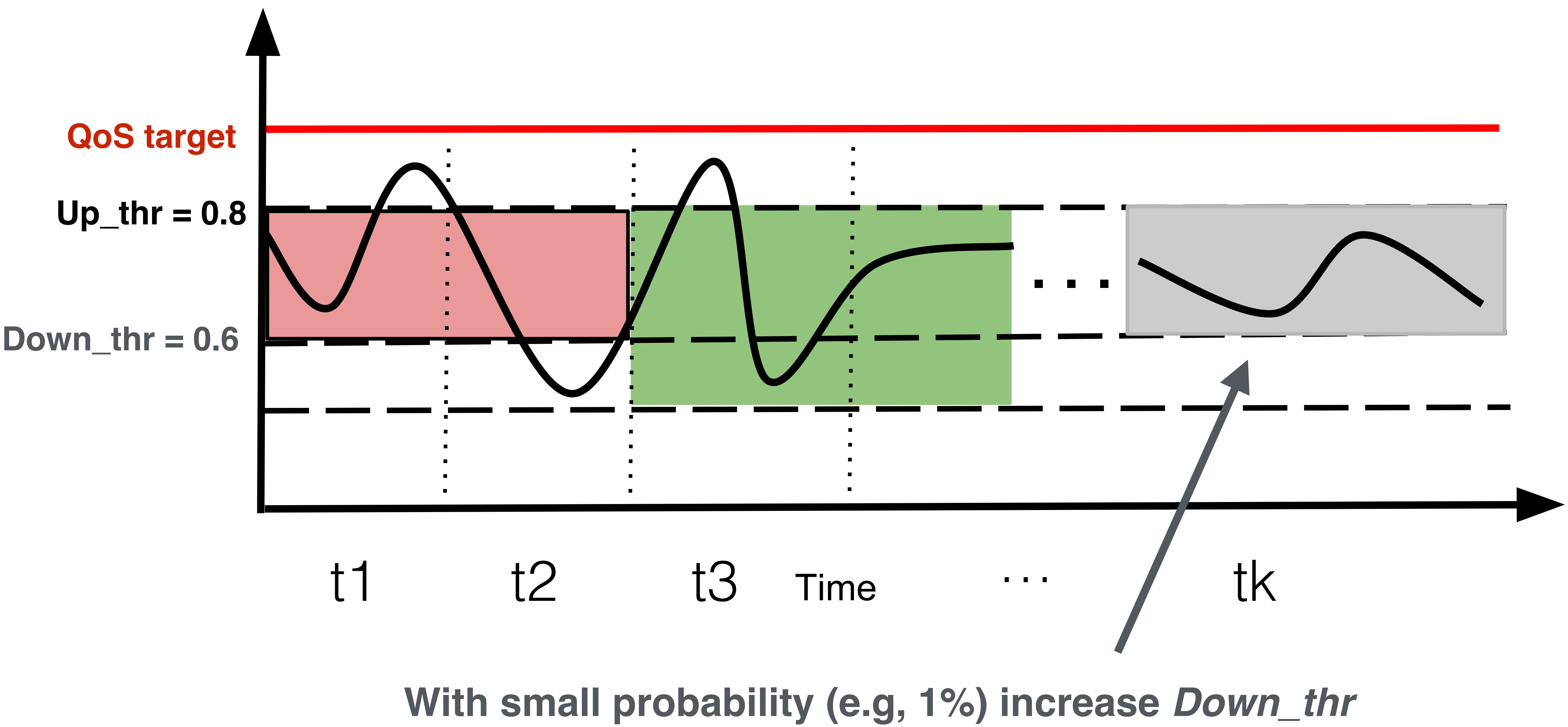
Solution: Dynamic deadzone selection



Solution: Dynamic deadzone selection



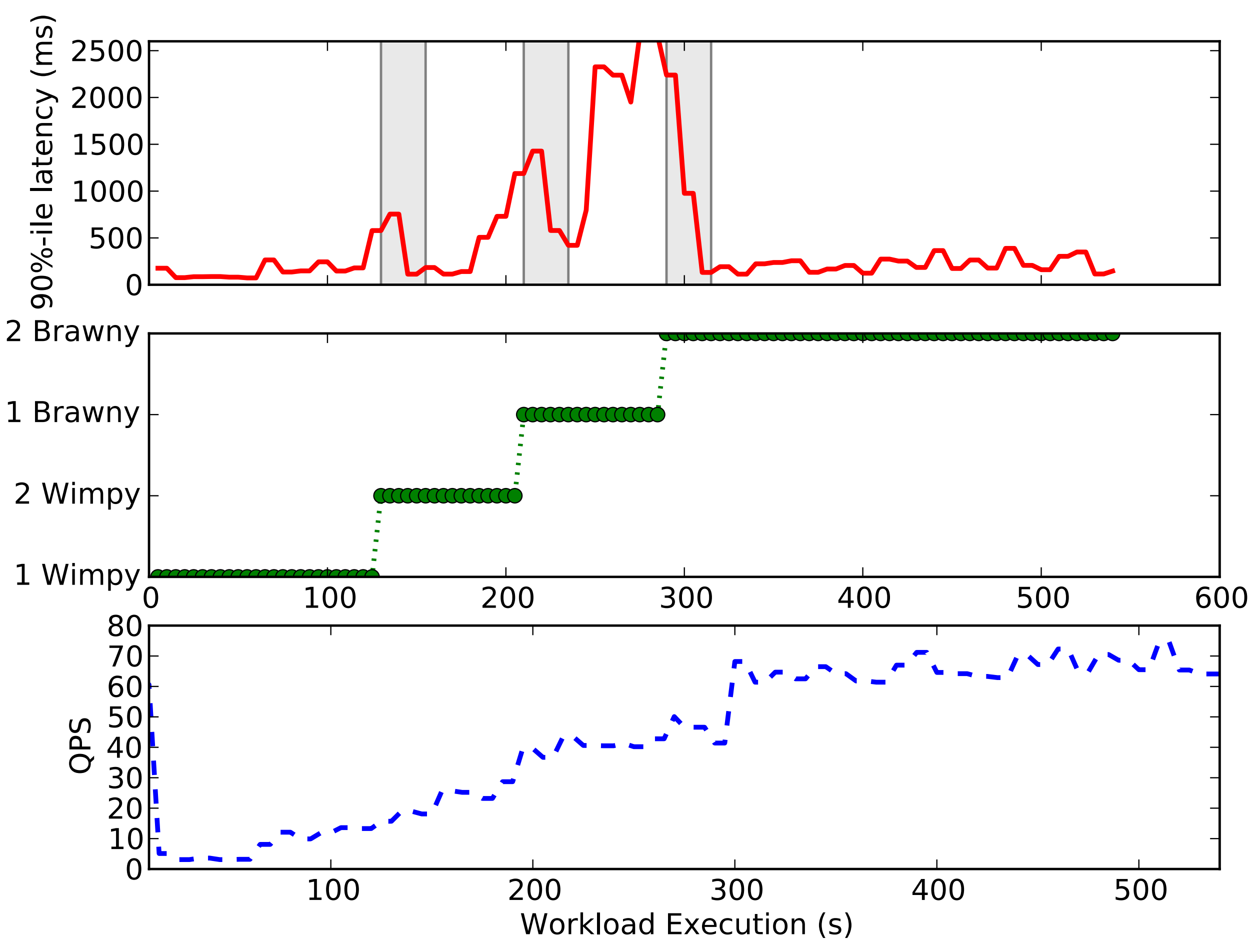
Solution: Dynamic deadzone selection



Stability: Dealing with settling time

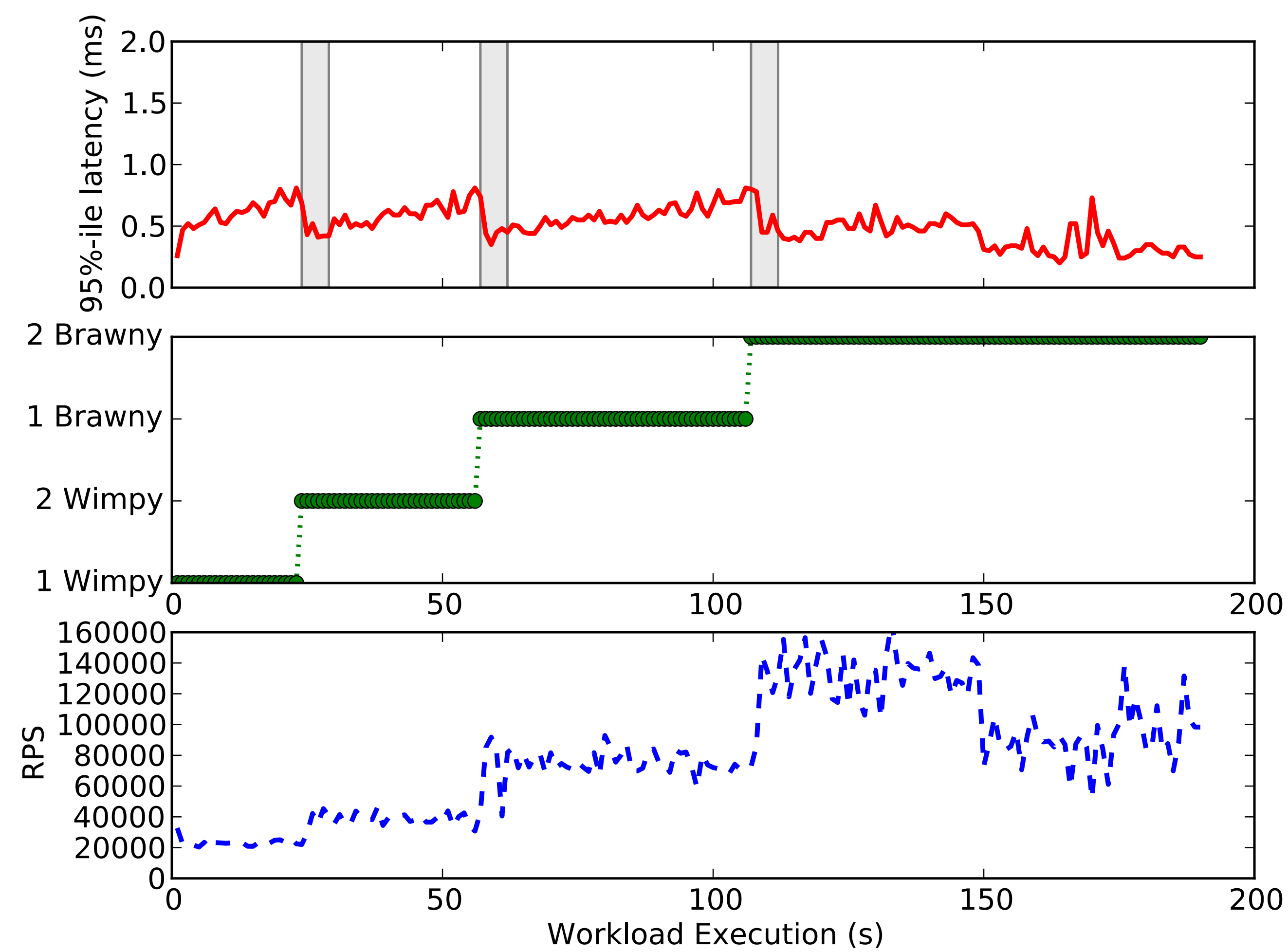
Websearch

QoS target = 500ms (90%-ile)



Memcached

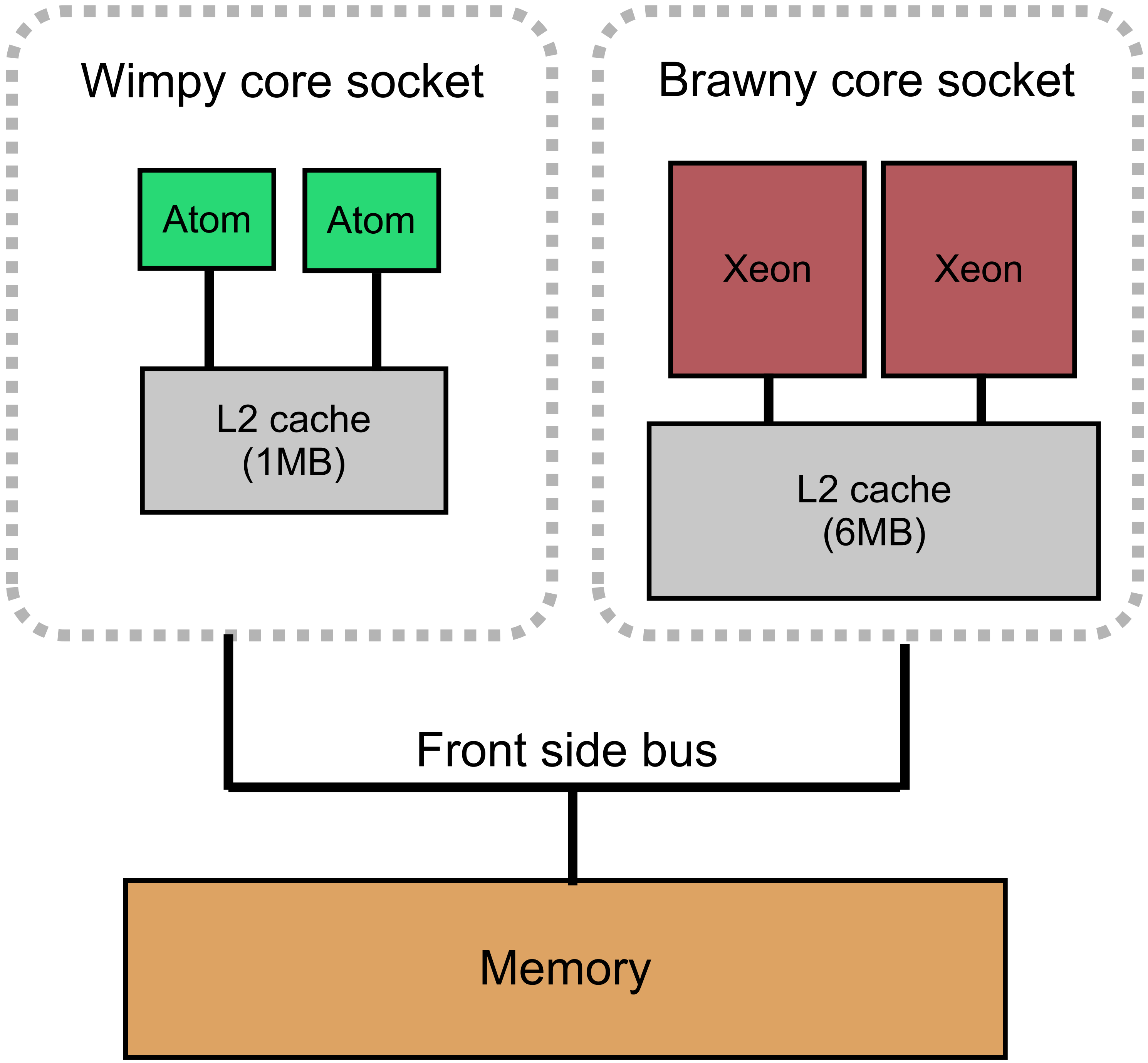
QoS target = 1ms (95%-ile)



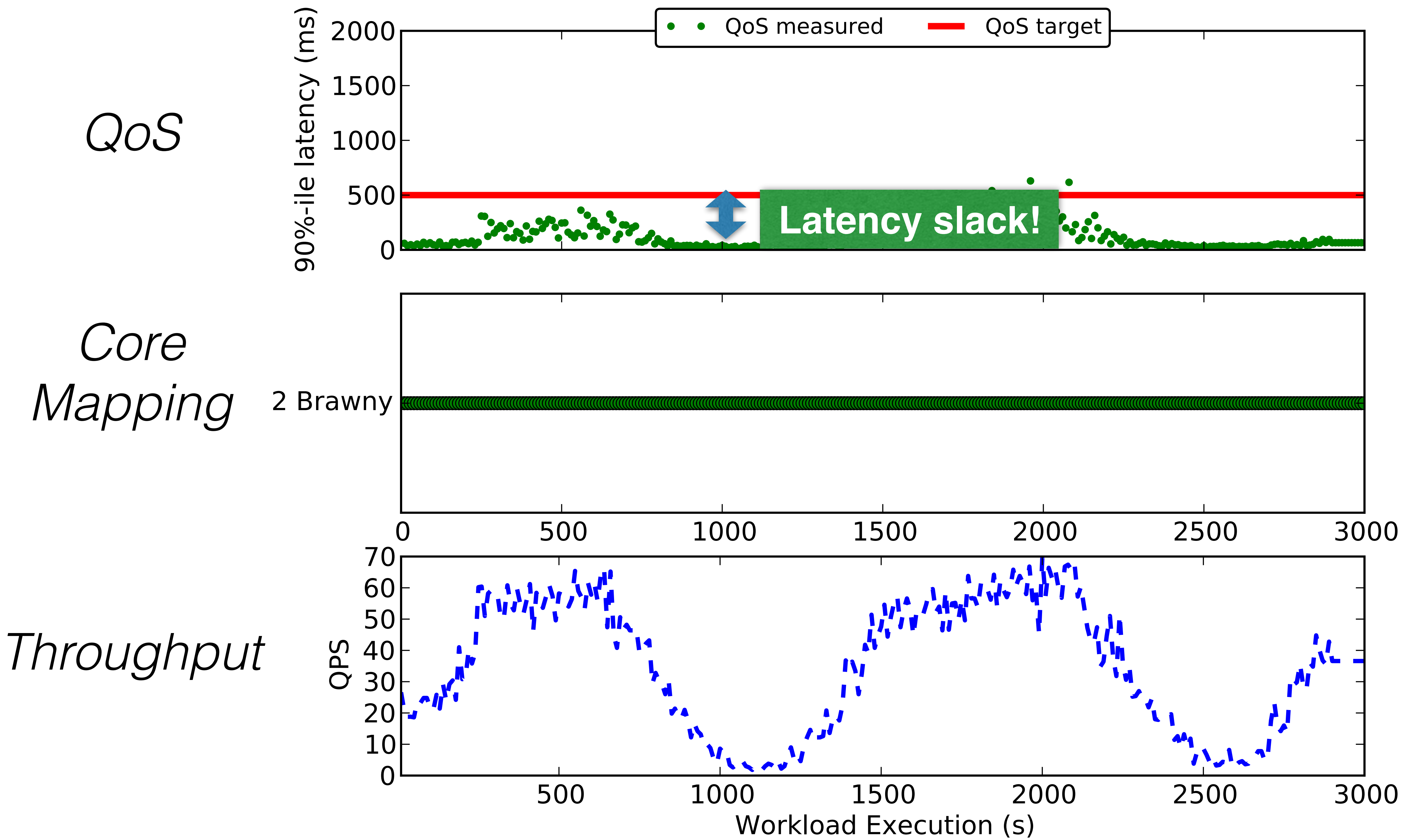
Do not reconfigure the system during the course of task migration (gray area)!

Evaluation

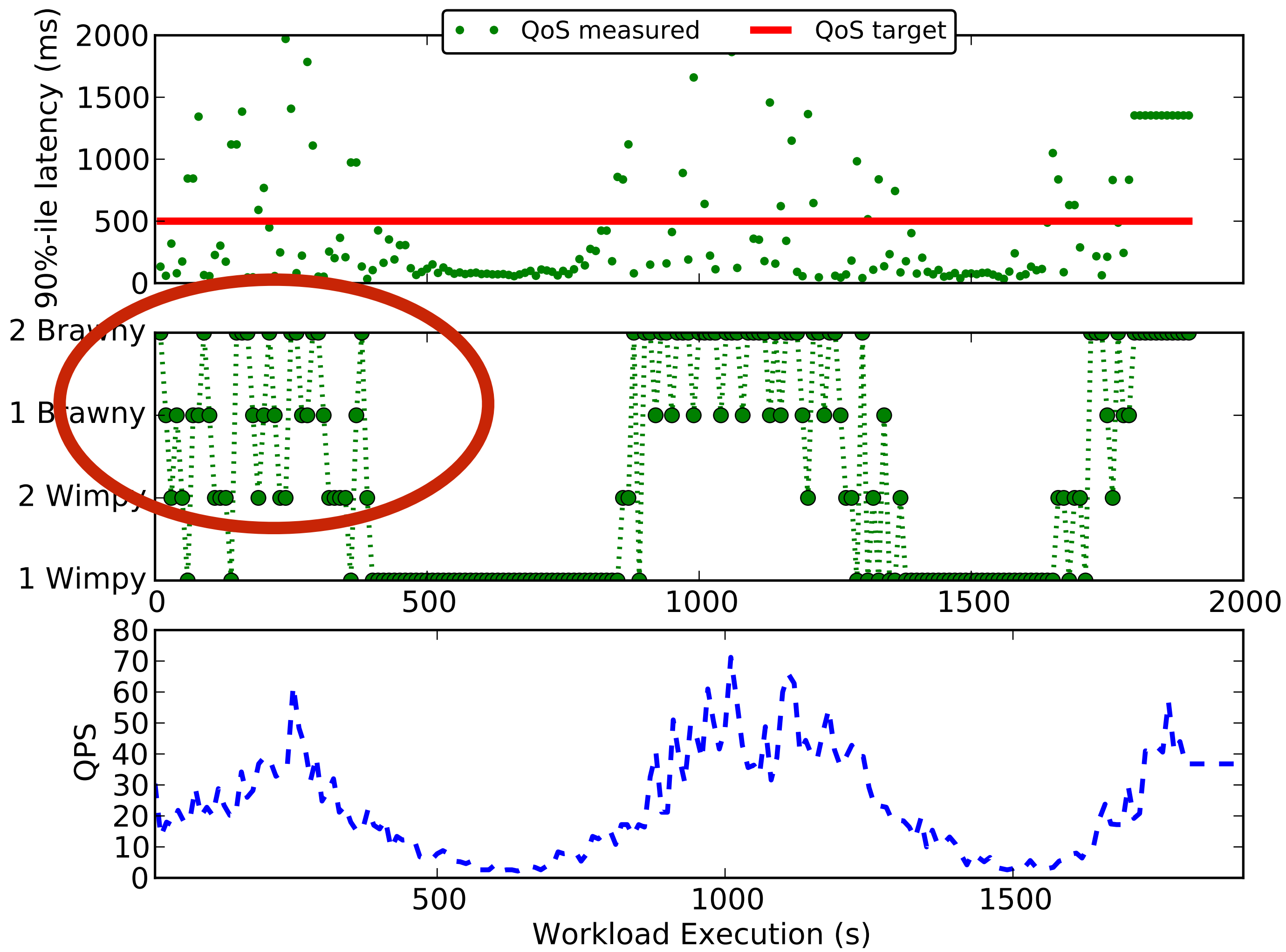
Experimental Platform: Intel QuickIA

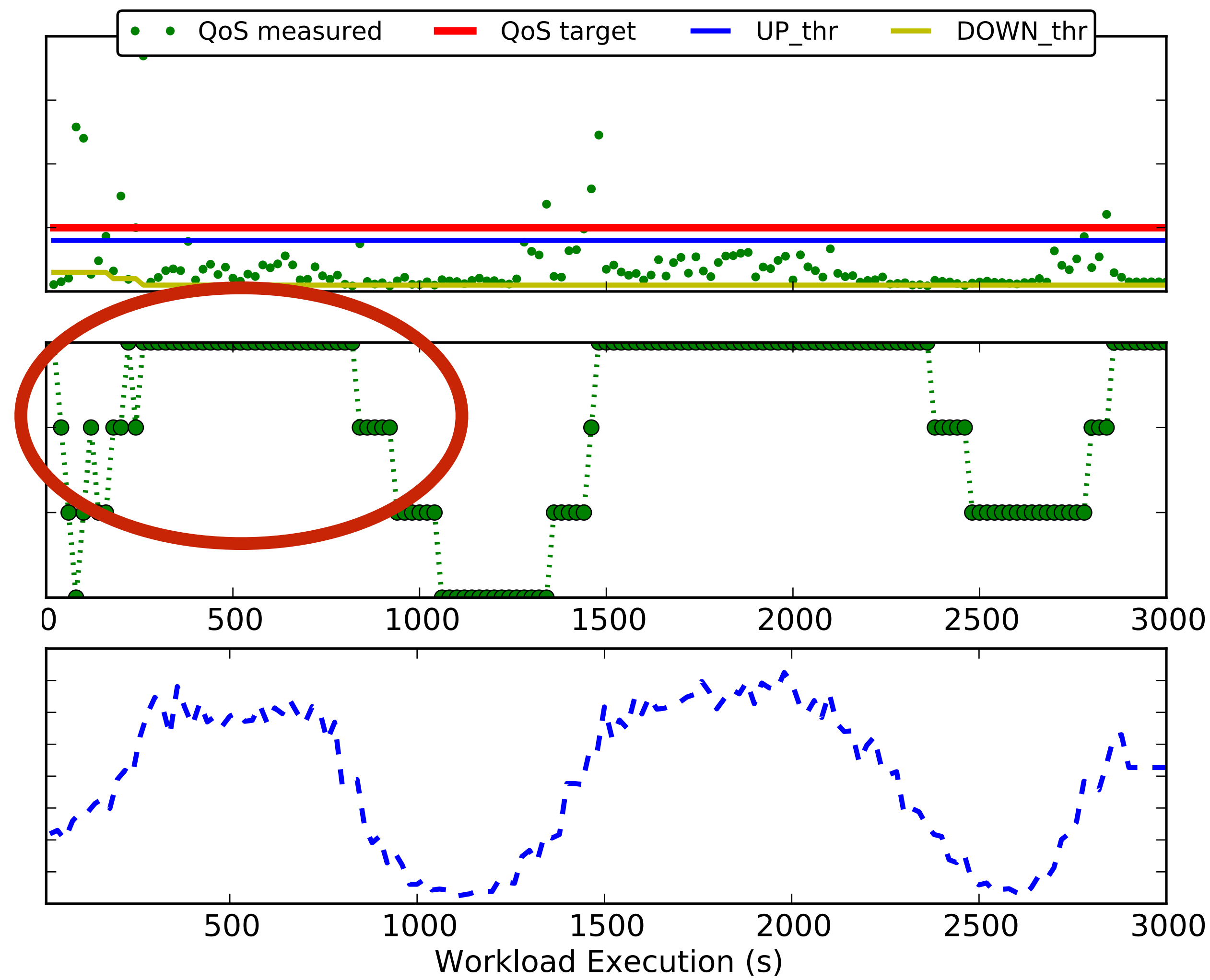


All-brawny (Static) baseline: Web-search



PID vs Deadzone: web-search

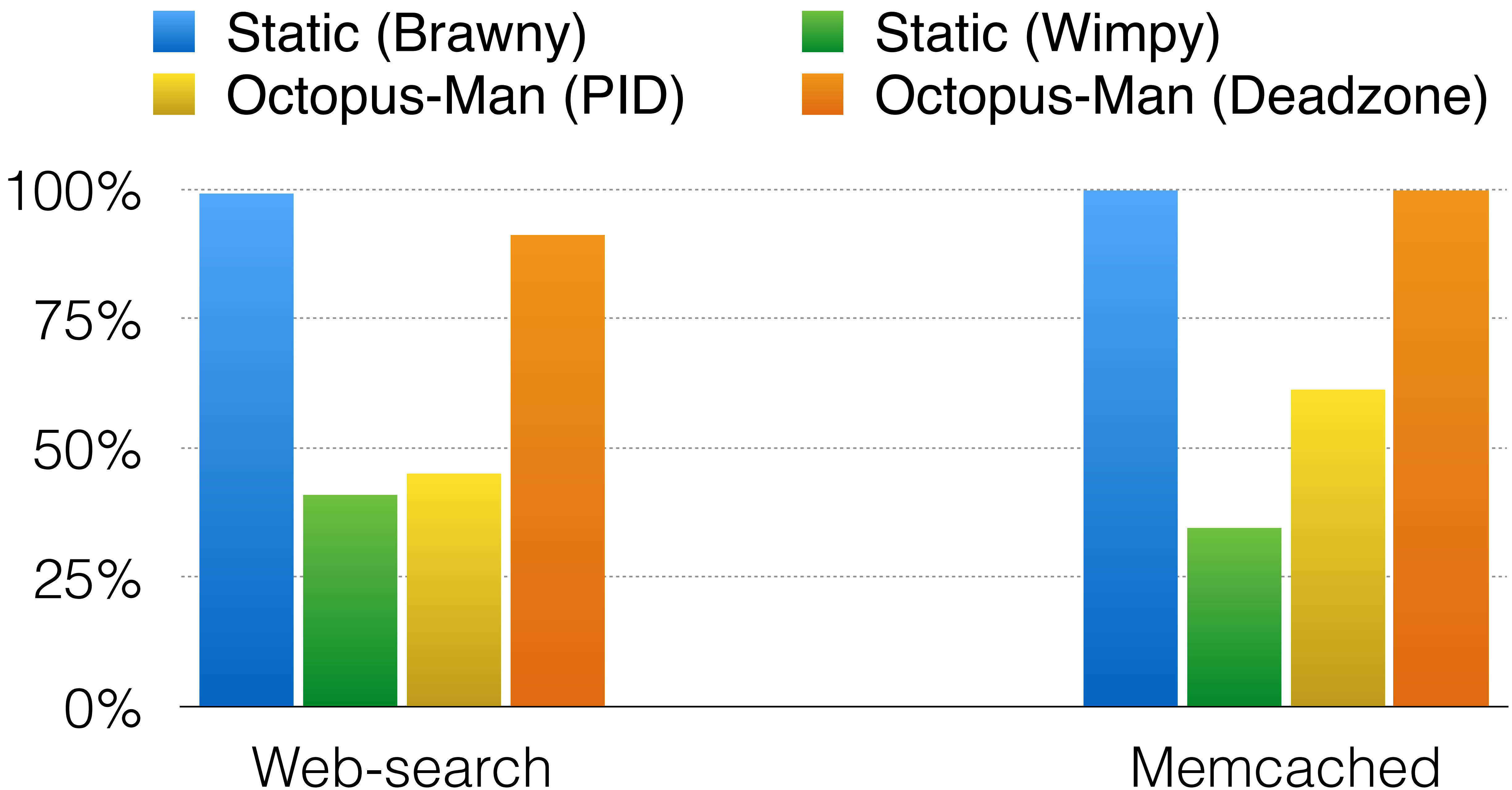




PID control

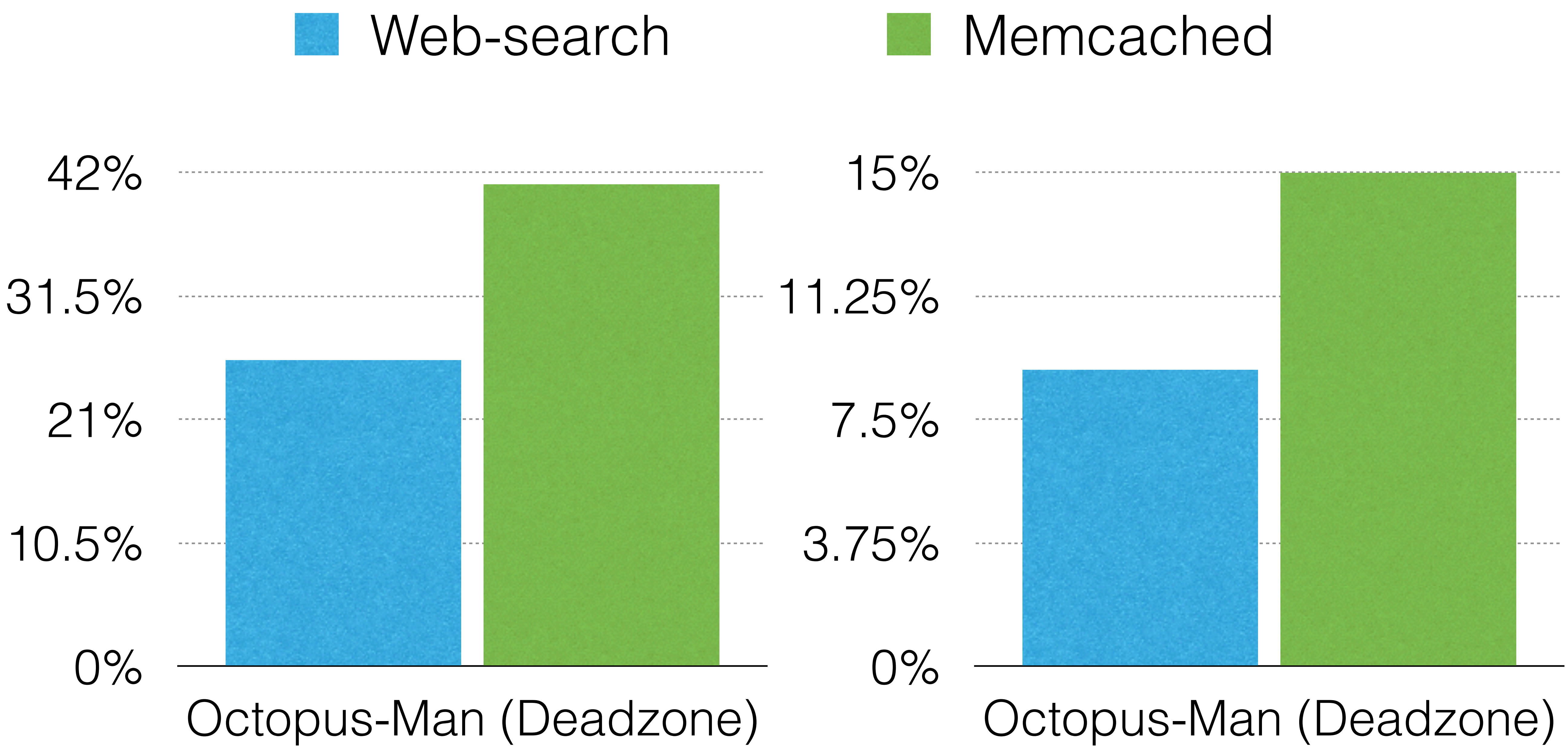
Deadzone control (adaptive threshold)

QoS results



Static (Brawny)
Static (Wimpy)
Octopus-Man (PID)
Octopus-Man (Deadzone)
100%
75%
50%
25%
0%
Web-search
Memcached

Energy reduction



CPU

Full-system

Conclusion

- Octopus-Man: task management solution exploring heterogeneous multicores
    - challenges addressed on *responsiveness* and *stability*
- Evaluation on real heterogeneous platform (Intel QuickIA)
    - Web-search and Memcached workloads
- Energy improvement of up to **41%** (CPU) and **15%** (full-system) over all-brawny homogeneous multicores
    - Batch processing throughput improvement of **34%** (mean) and **50%** (max)

Thanks!