Towards Energy Proportionality for Large-Scale Latency-Critical Workloads

David Lo†*, Liqun Cheng*, Rama Govindaraju*, Luiz André Barroso*, Christos Kozyrakis†

† *Stanford University* * *Google Inc.*

Stanford MAST

Motivation

- **Energy proportionality**
  - Servers are far less energy efficient at low and medium utilizations
  - Servers are underutilized due to diurnal load patterns
- **Large-scale latency-critical workloads**
  - Web search, social networking, etc.
  - Strict guarantees on tail latency and workload complexity precludes previous power management techniques

Executive Summary

- Energy waste is caused by overachieving on performance
- **Solution: Match power to Service Level Objective (SLO)**
  - End-to-end SLO latency monitoring
  - Fine-grain power saving mechanism (i.e. RAPL)
- Built dynamic controller for large-scale latency-critical workloads
  - 20-30% power savings on production Google search without SLO violations

Outline

- Energy proportionality vs. latency-critical workloads
- Recovering energy proportionality: iso-latency
- PEGASUS: QoS aware dynamic controller

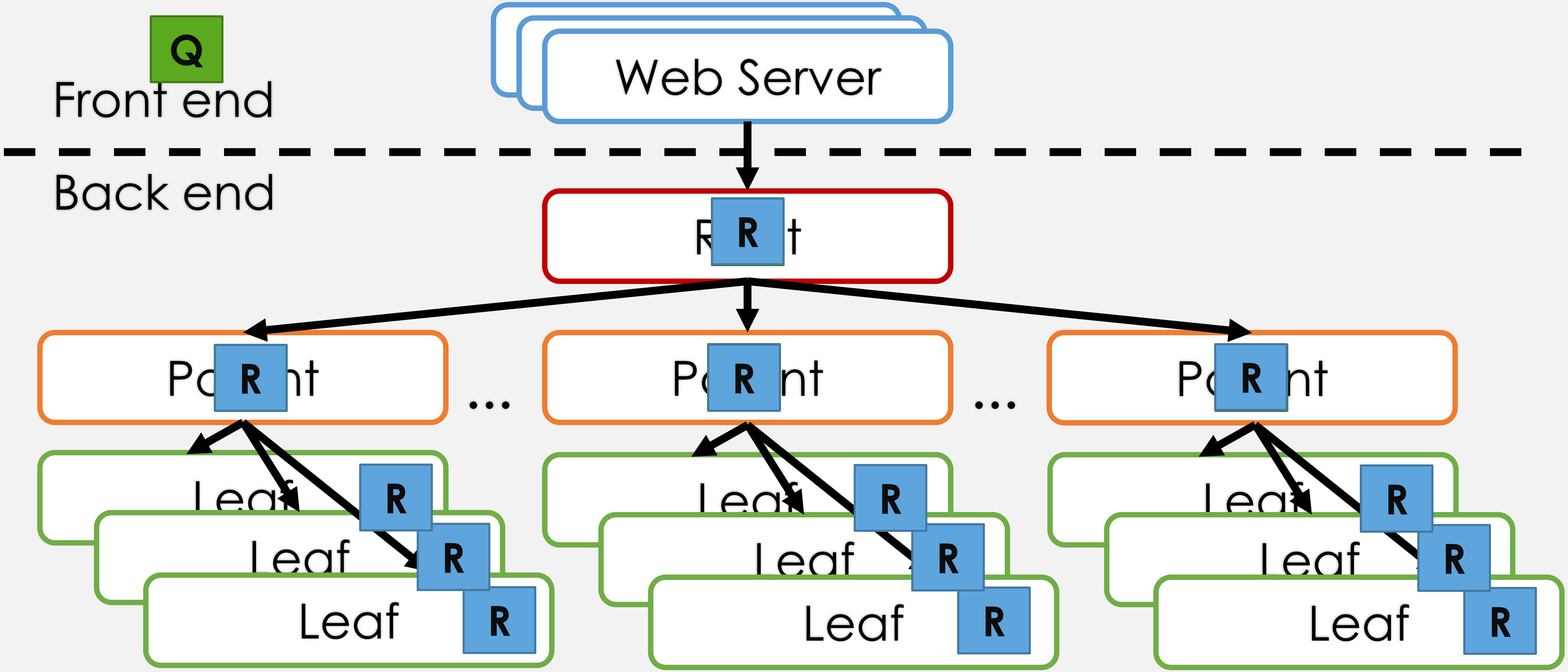
Energy proportionality vs. latency-critical workloads

The case for latency-aware fine-grain power management

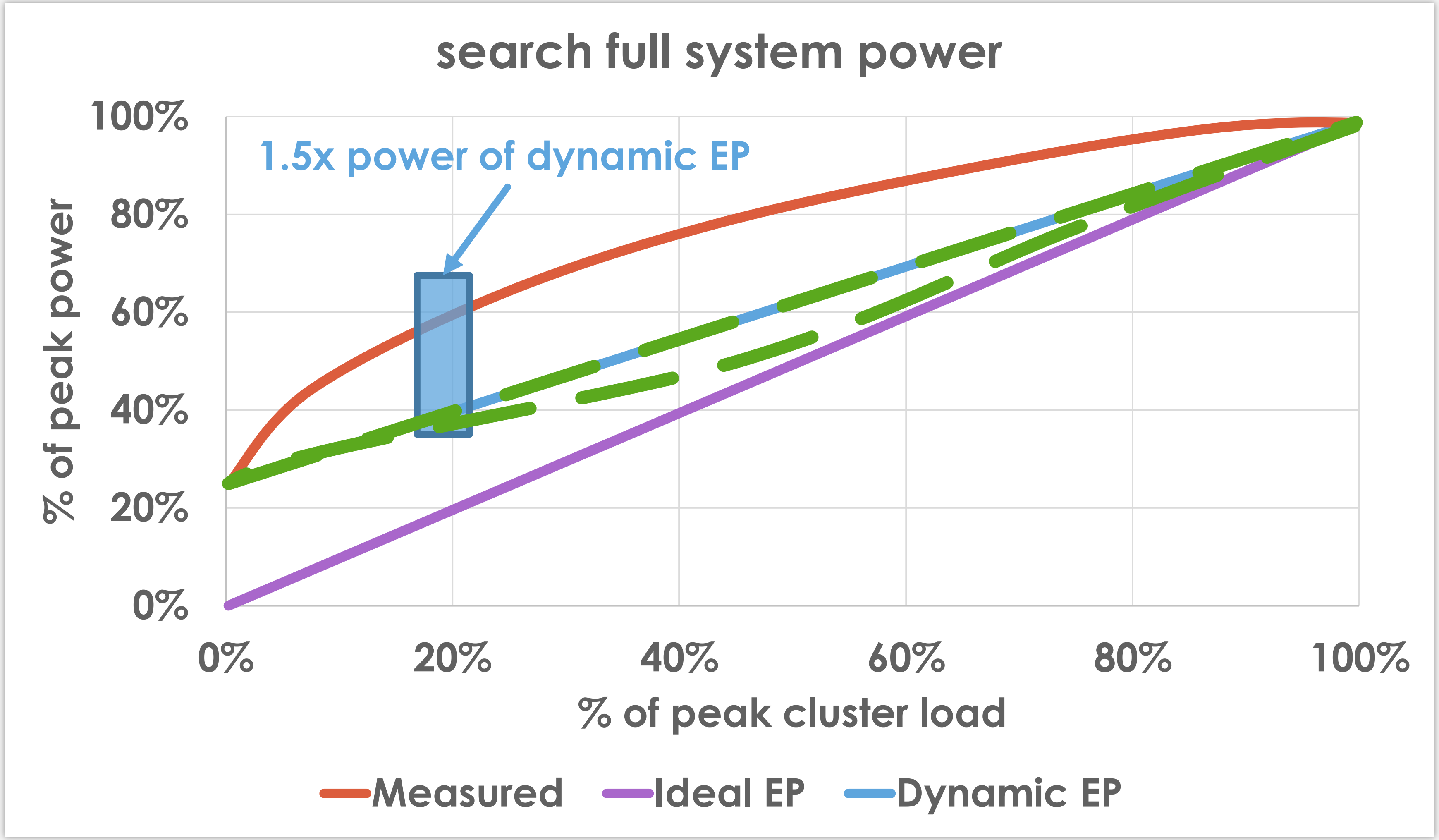
OLDI workloads

- On-line Data Intensive (**OLDI**) workloads are user-facing workloads that mine massive datasets across many servers
  - Strict Service Level Objectives (**SLO**): e.g. 99%-ile tail latency is 5ms
  - High fan-out with large distributed state
  - Extremely challenging to perform power management
- Workload we evaluate on:
  - **search**: Query serving portion of production Google search

Search topology

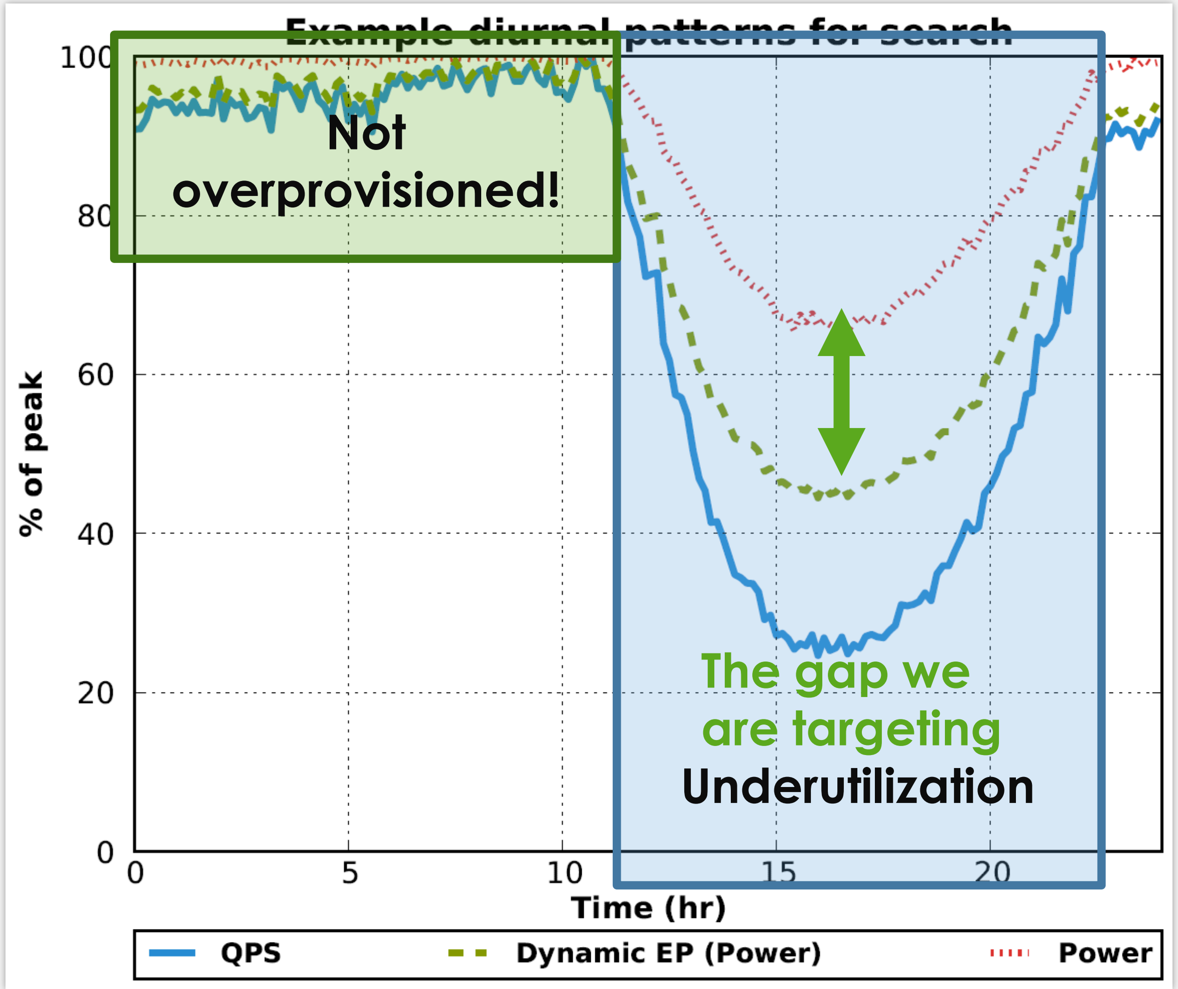


The challenge of energy proportionality



The need for energy proportionality

- Diurnal variation in cluster load and power for search across a 24 hour period
- Cluster not fully utilized half the time
- Gap between measured power and EP curves represent potential savings

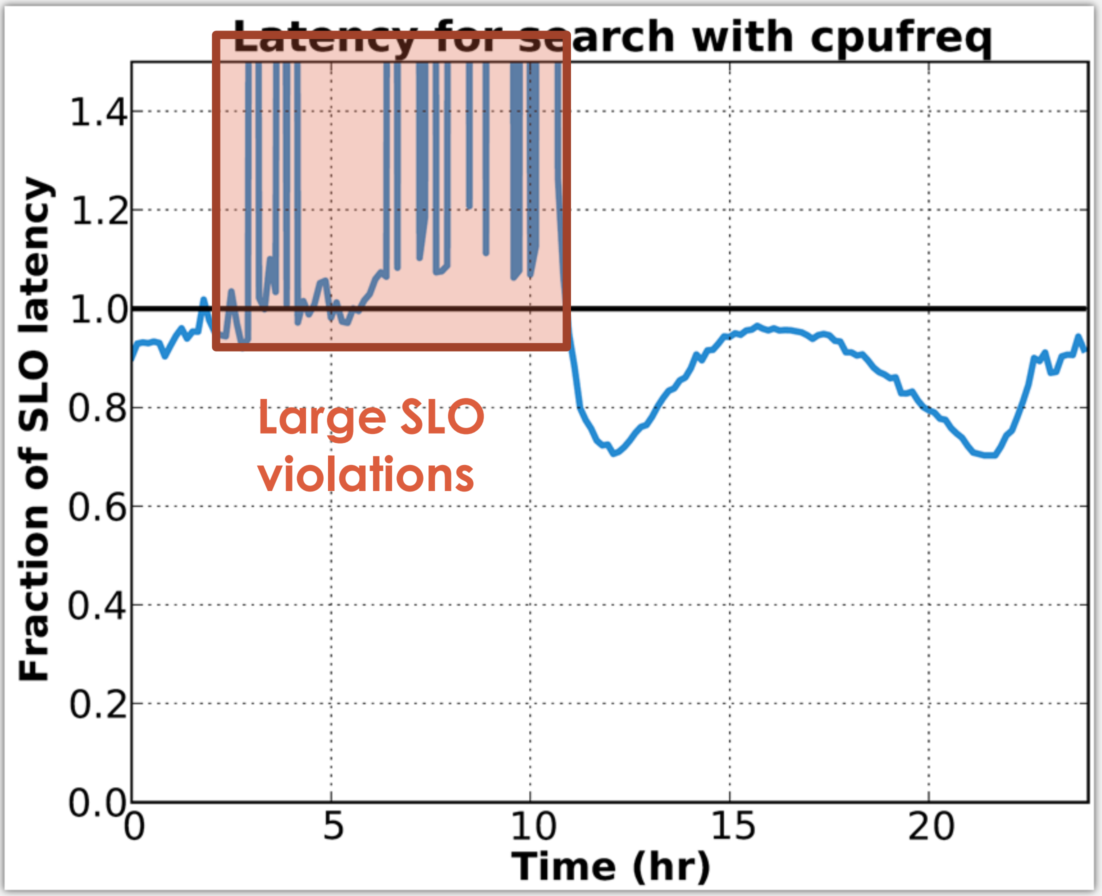


Previous cluster-level power management

- ~~Consolidate load on fewer servers during low utilization~~
  - **Issue:** state of OLDI applications cannot fit on fewer servers
- ~~Use very low power idle modes~~
  - **Issue:** OLDI request rate is always too high, e.g. >1k requests/sec
- ~~Batch requests to form long enough idle periods~~
  - **Issue:** OLDI applications cannot tolerate msec exit times and batching delays

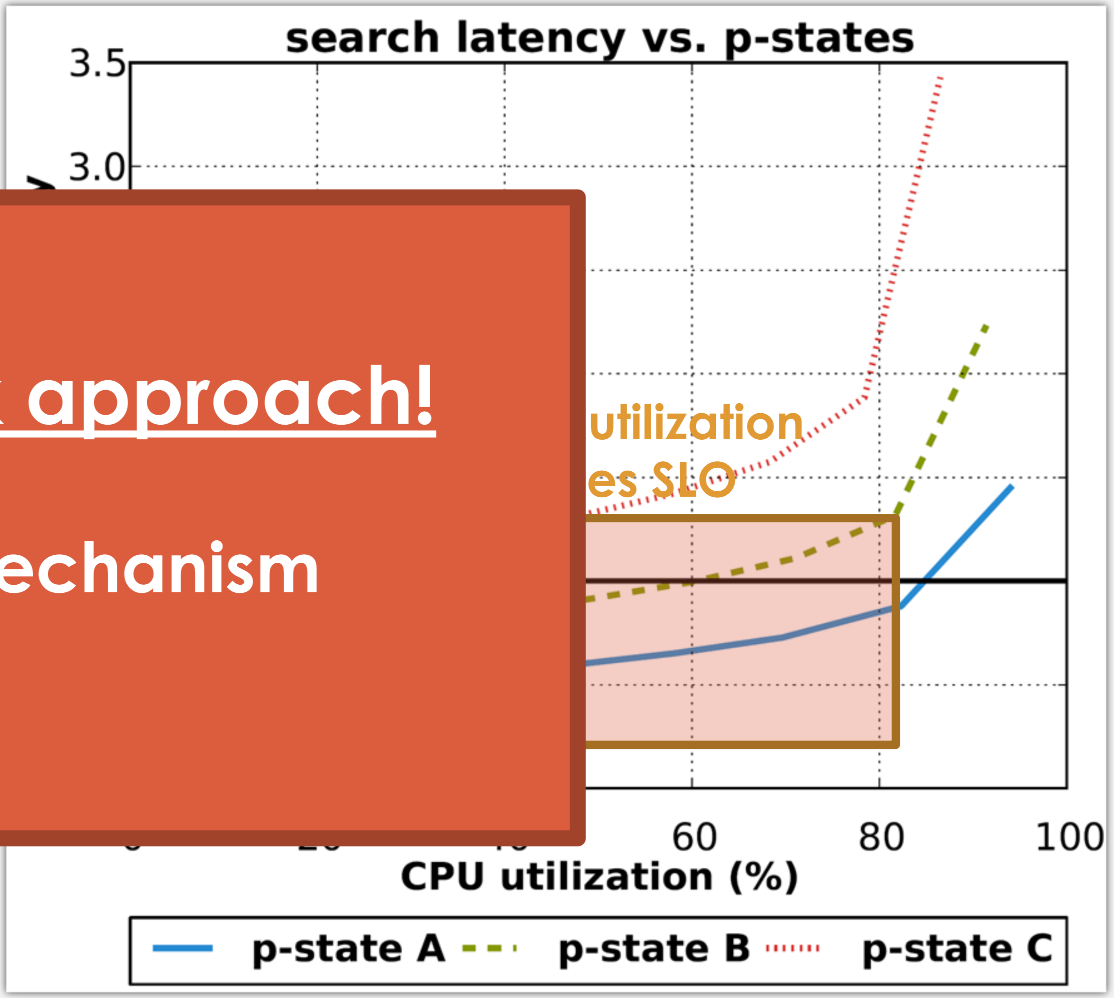
Previous machine-level power management

- CPU utilization based DVFS
  - Changes p-states based on CPU utilization
- **Issue**: causes SLO violations



Weakness of current DVFS schemes

- CPU utilization is a poor proxy for workload latency
- **To meet SLO latency-aw**



Need to rethink approach!

- New policy
- New control mechanism
- New controller

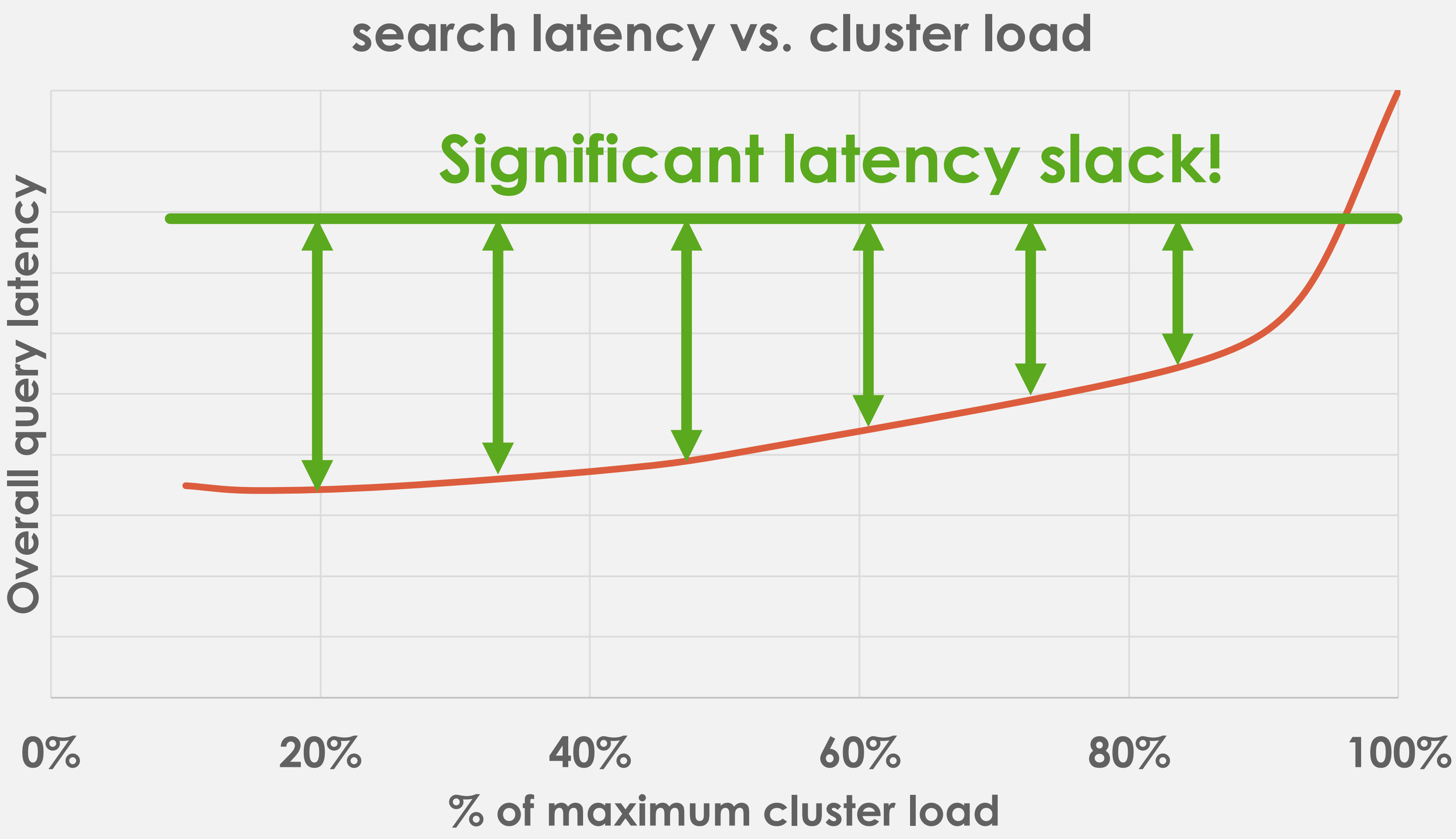
Recovering energy proportionality: iso-latency

Trading end-to-end latency slack for immense power savings

Motivating assumption

- **Beating the end-to-end SLO is no better than meeting it**
  - The end-user only cares if the web page takes a long time to load
  - If the page loads in 0.25sec vs. 0.50sec, user does not notice

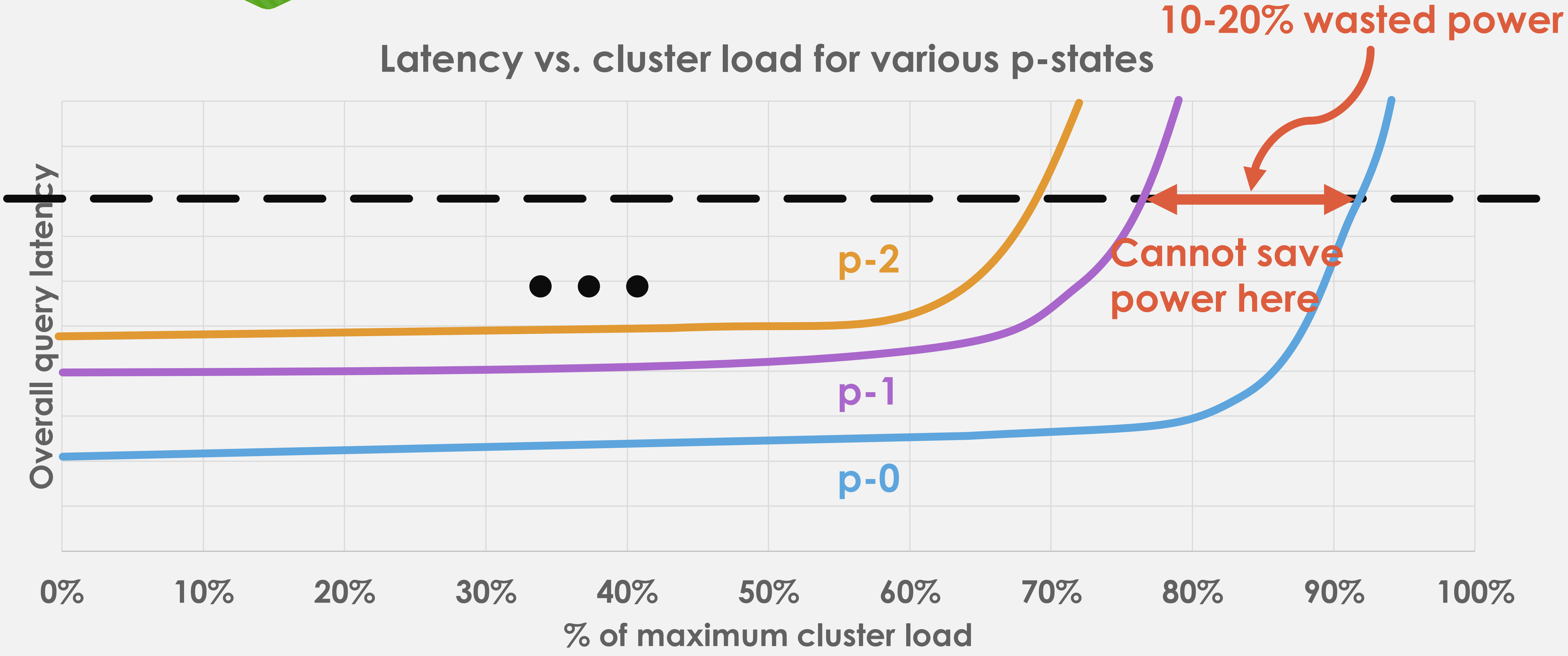
Latency opportunities



Iso-latency power management

- **Key idea:** Trade end-to-end latency slack for power savings
- Use power management mechanisms to keep the workload performing just well enough to avoid SLO violations
  - **Need end-to-end latency feedback from workload**
    - Most OLDI workloads have ways of measuring this
  - **Need fine-grained power management mechanisms**

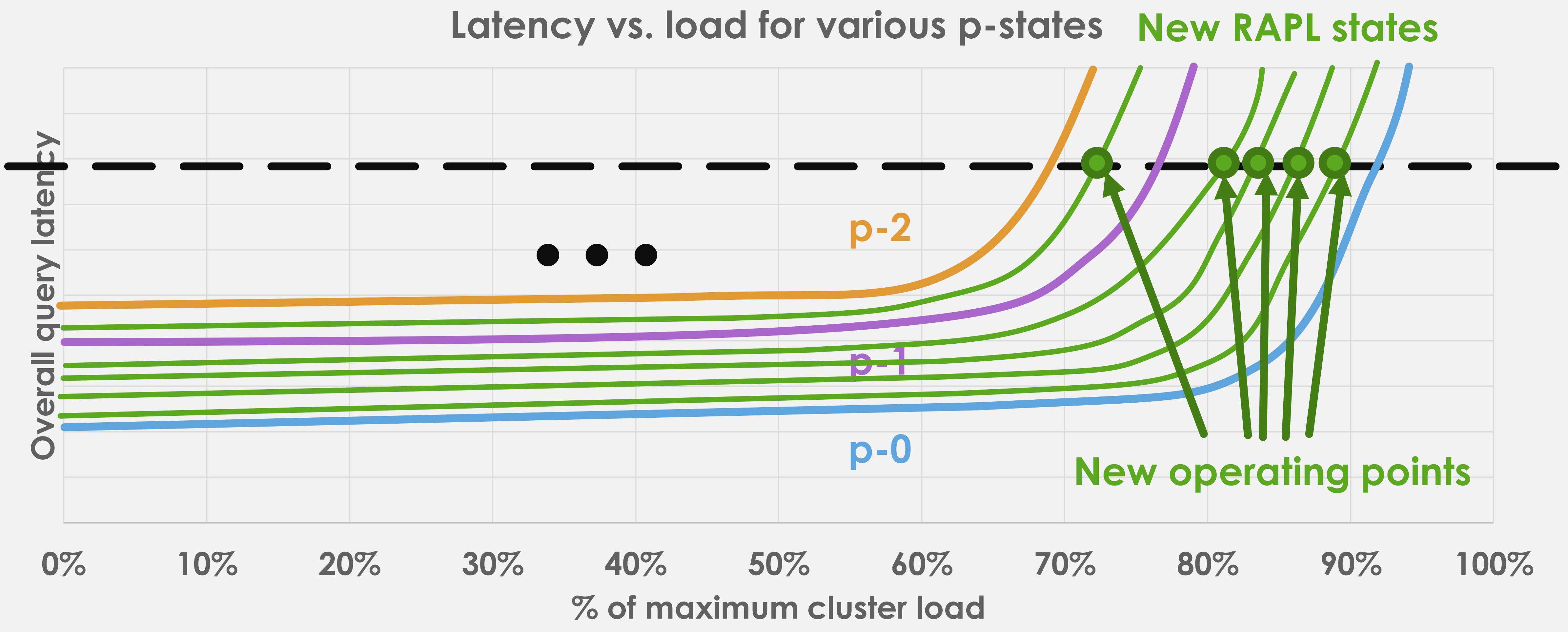
Problem: p-states are not fine grained



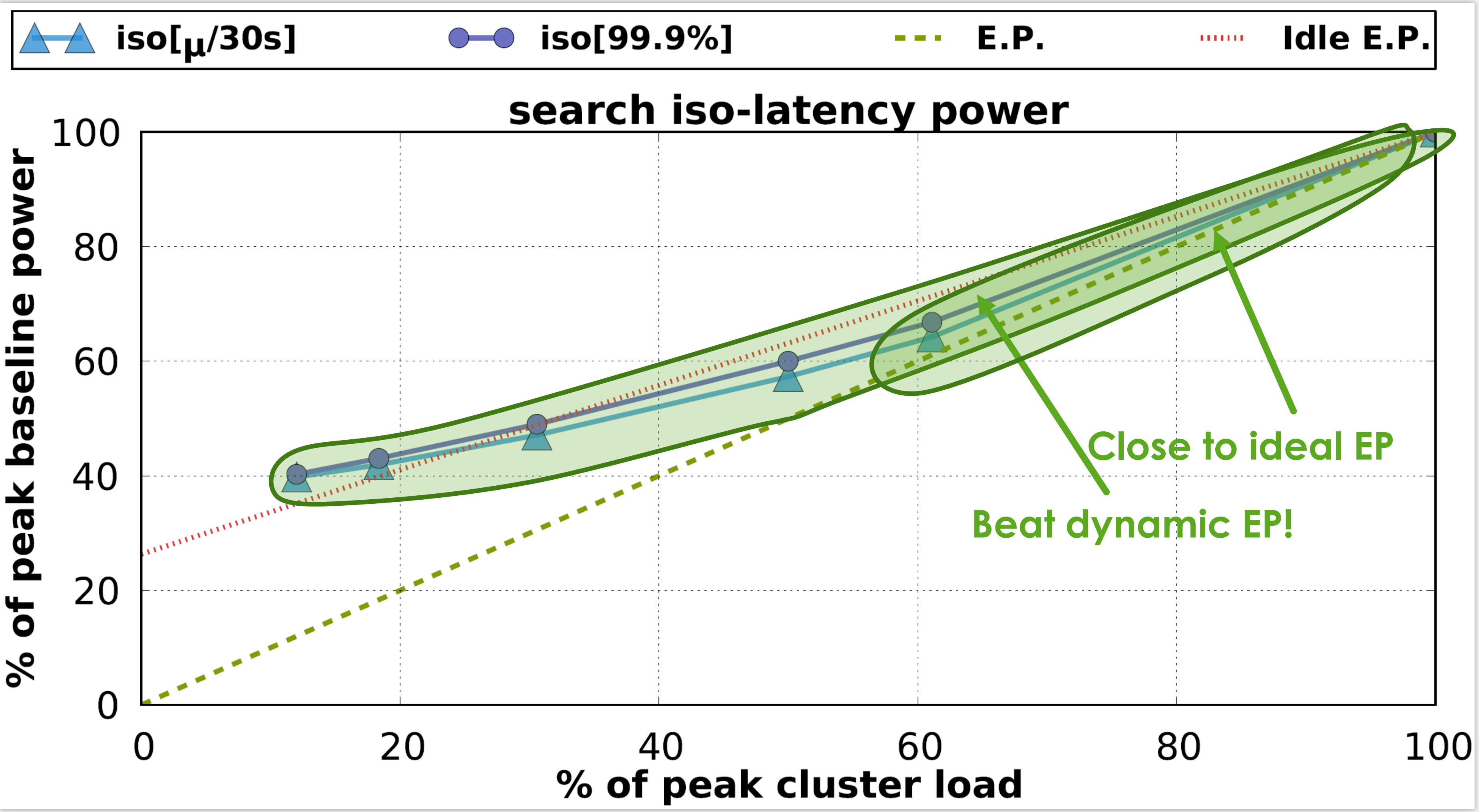
Solution: RAPL

- RAPL: **R**unning **A**verage **P**ower **L**imit
- **Fine-grained**: power limit increments as small as 0.125W
- **Fast**: <1ms delay to apply new limit
- **Effective**: Dynamic Voltage Frequency Scaling (DVFS) behind the scenes to meet the power limit
  - More fine-grained than p-states
  - Can even modulate between multiples of base clock frequencies

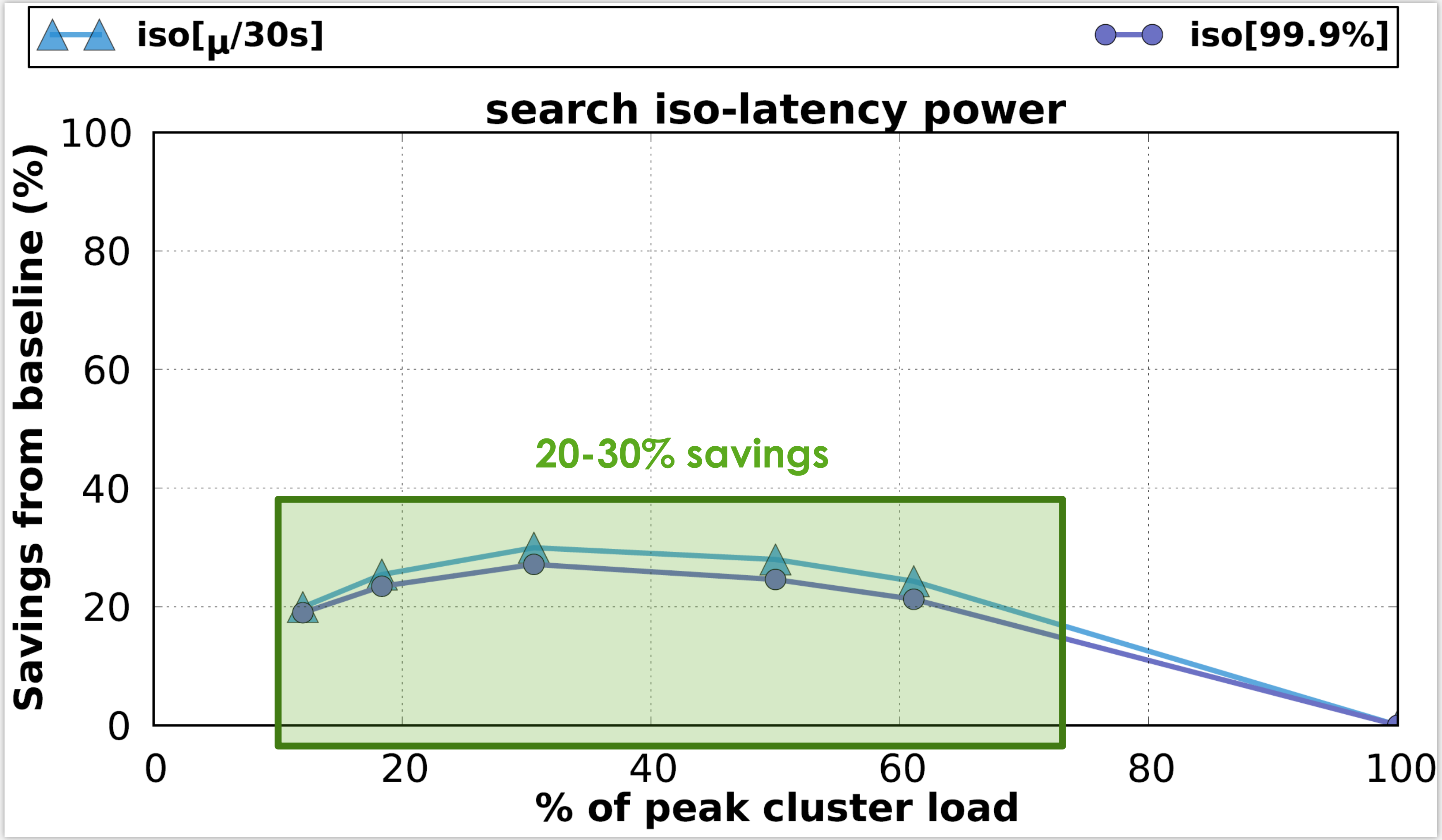
Advantages of fine-grain control



Iso-latency potential: power



Iso-latency potential: power savings

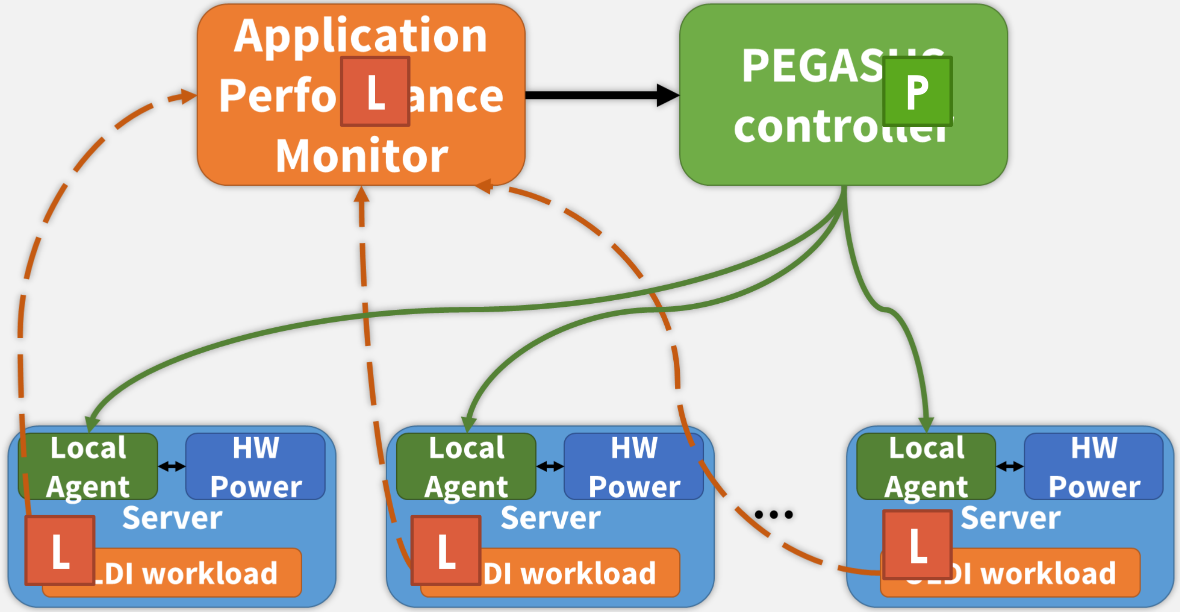


Power and **E**nergy **G**ains **A**utomatically **S**aved from **U**nderutilized **S**ystems

QoS aware dynamic controller

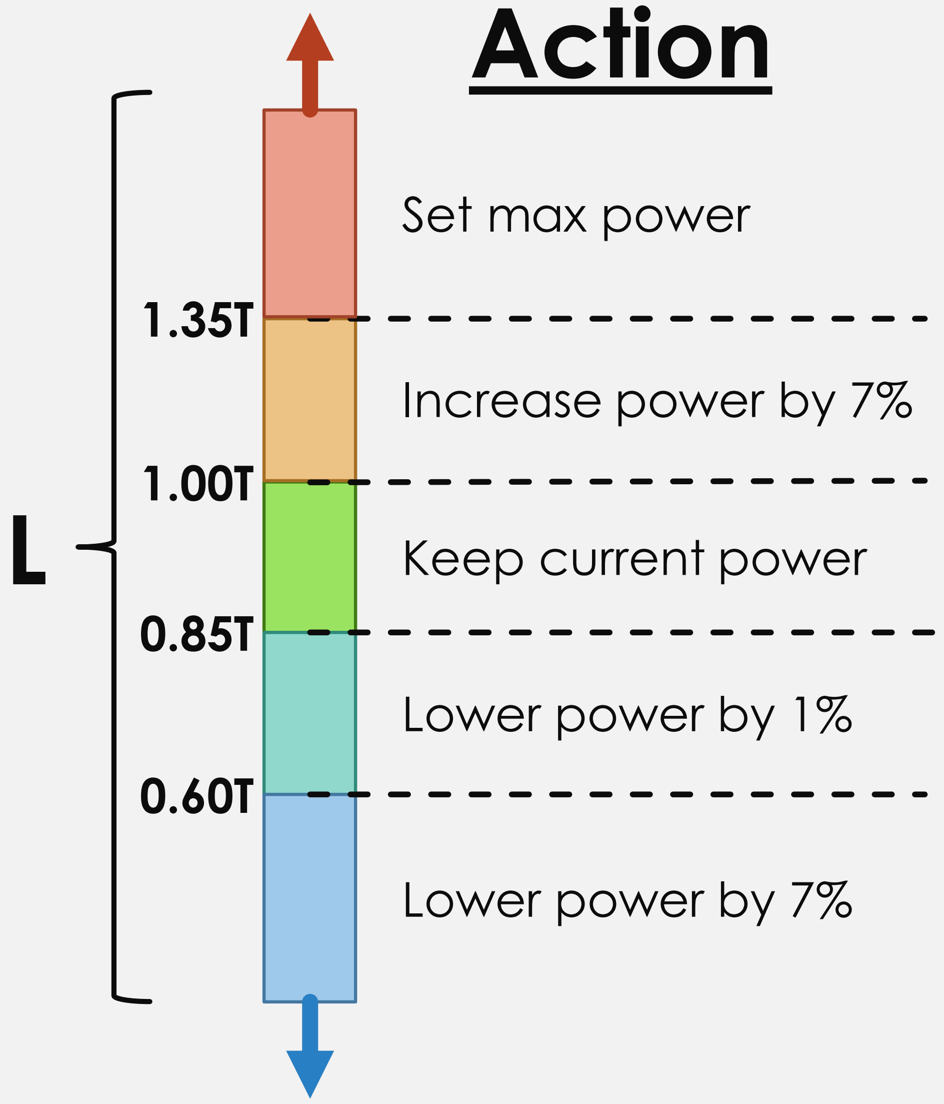
PEGASUS description

- Real-time dynamic controller for **iso-latency**
- Use RAPL as knob for power
- Measures latency slack and sets **uniform** power limit across all servers
- Power is set by workload specific policy



Example PEGASUS policy for search

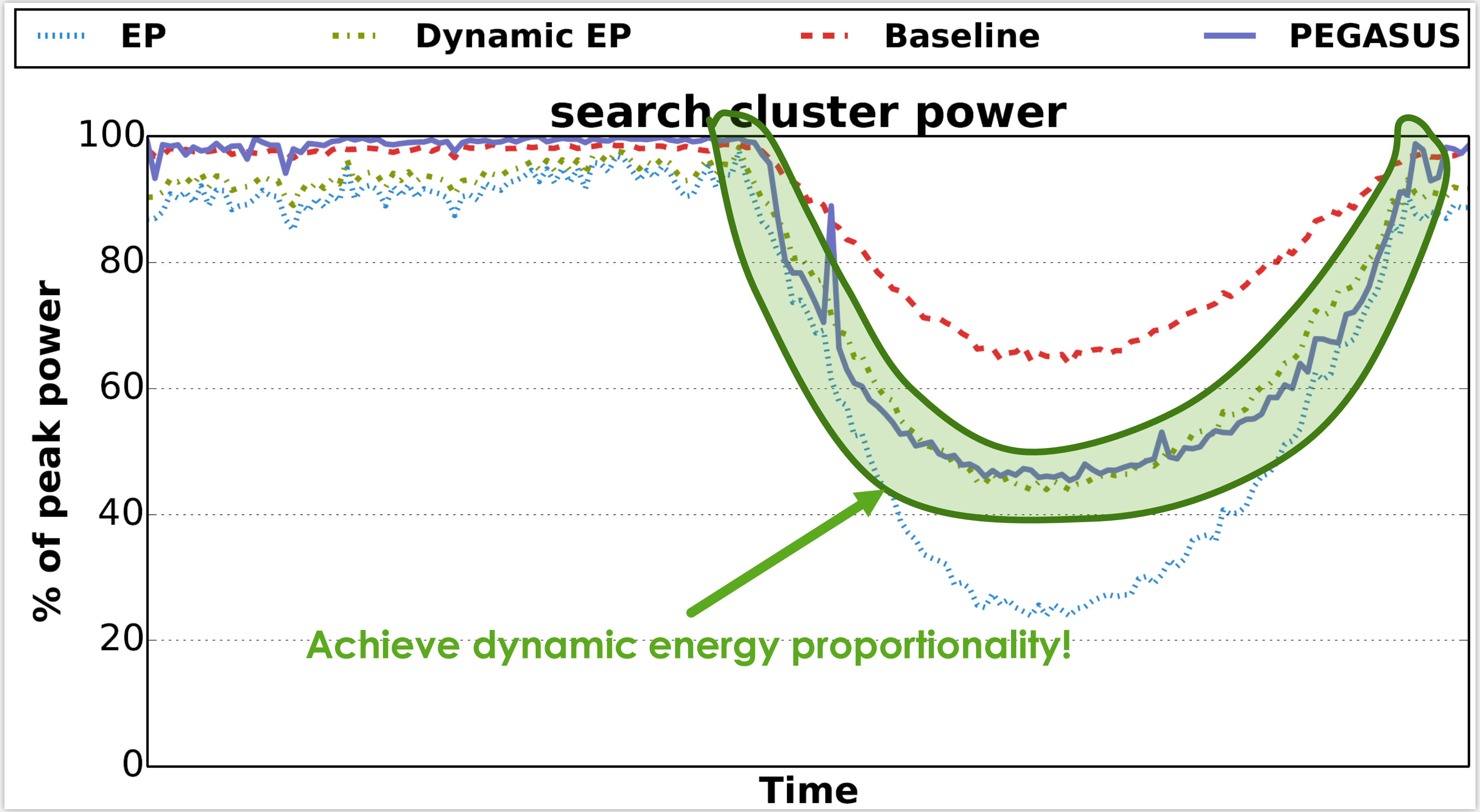
- **L** = Measured instant latency
- **T** = SLO target
- Use instant latency for quick corrections
- Violating SLO latency triggers fail-safe
- Constants determined through empirical optimization



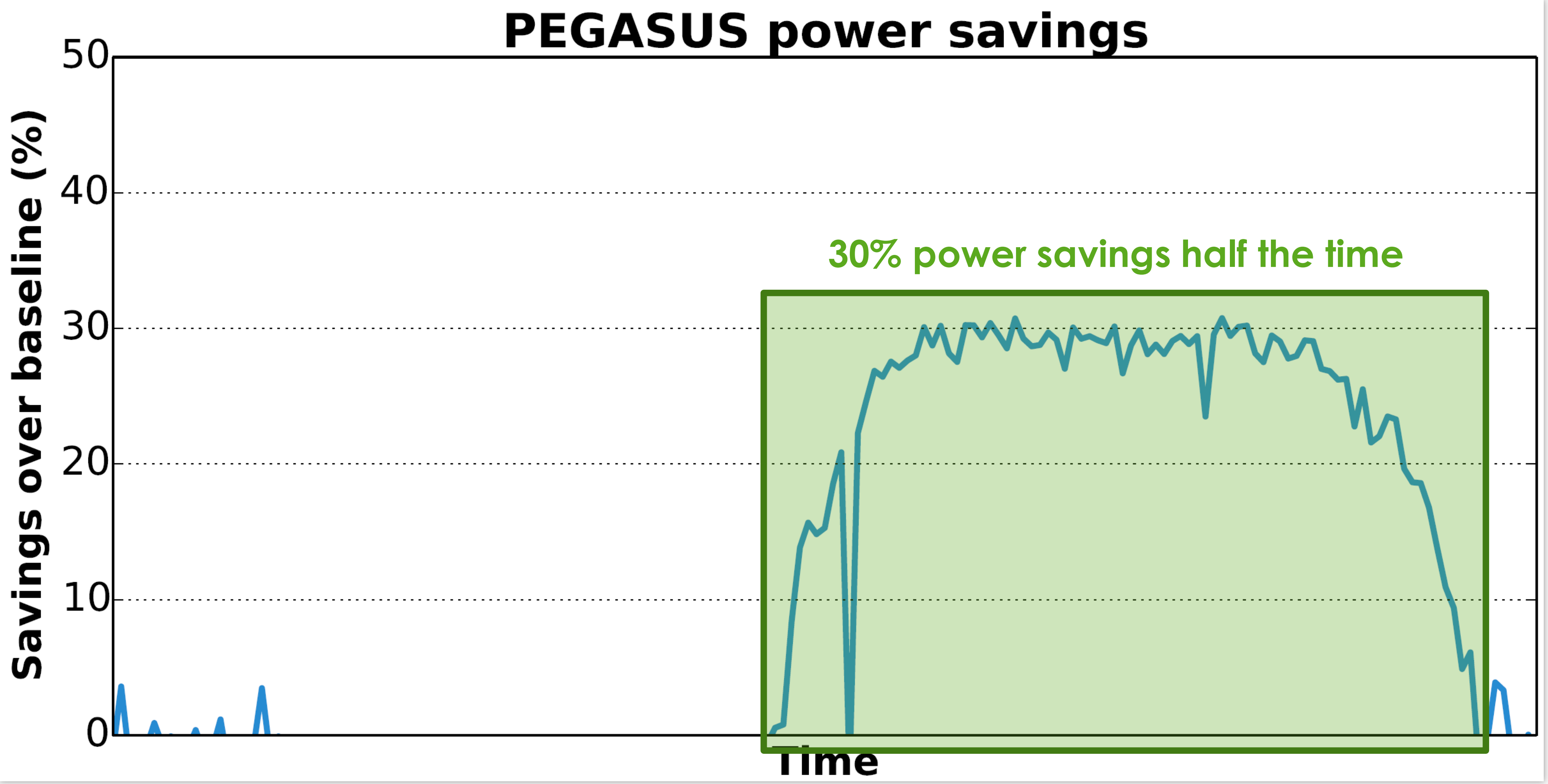
Evaluation methodology

- Workload parameters
  - **SLO metric**: 30 second average latency
  - Traffic pattern and user queries derived from anonymized search logs
  - Index derived from production search index
- Evaluate on several cluster sizes
  - **Small**: tens of machines, use full 24hr trace
  - **Production**: thousands of machines, use 12hr portion
- Measure full cluster power and SLO latency

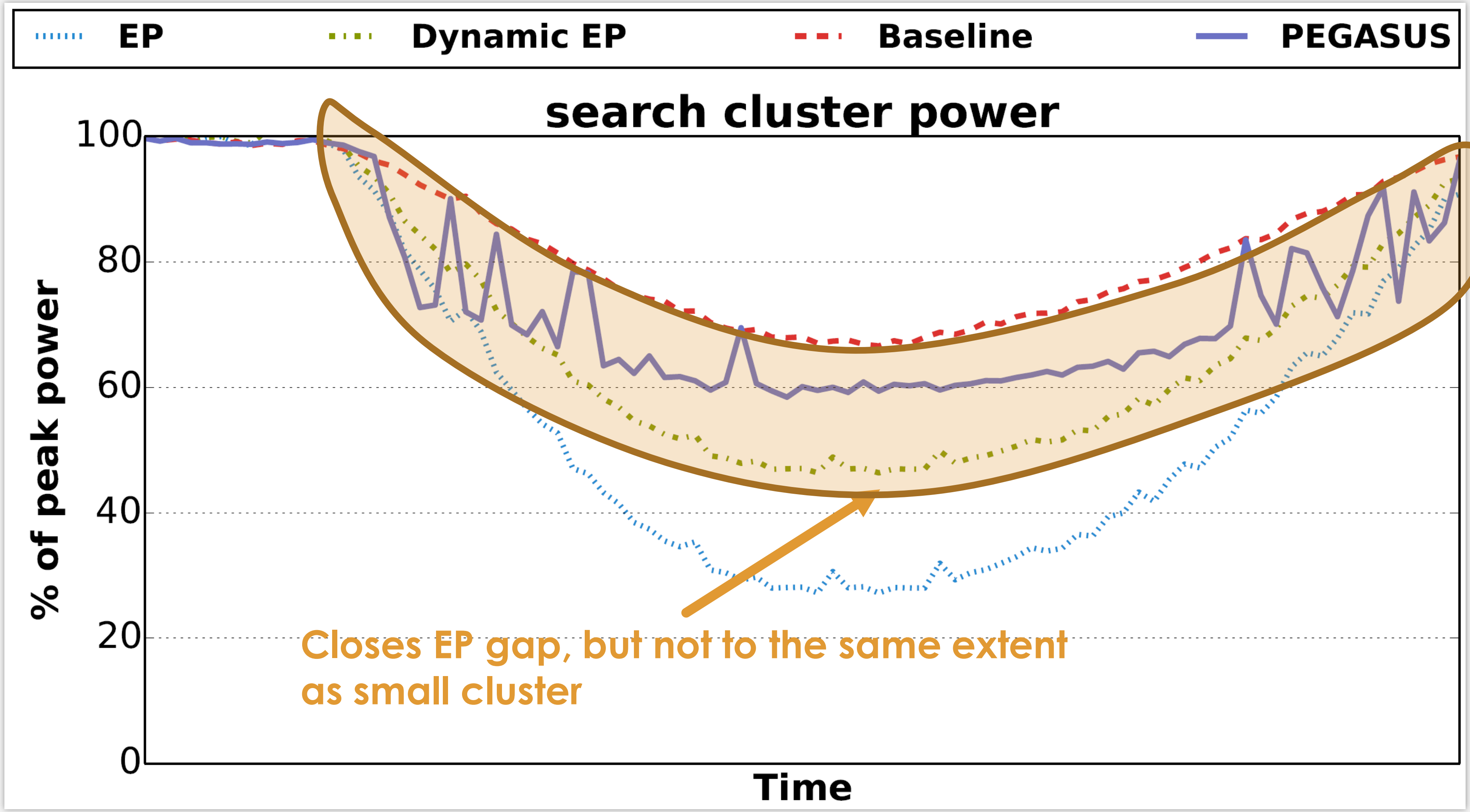
Small cluster results: power over time



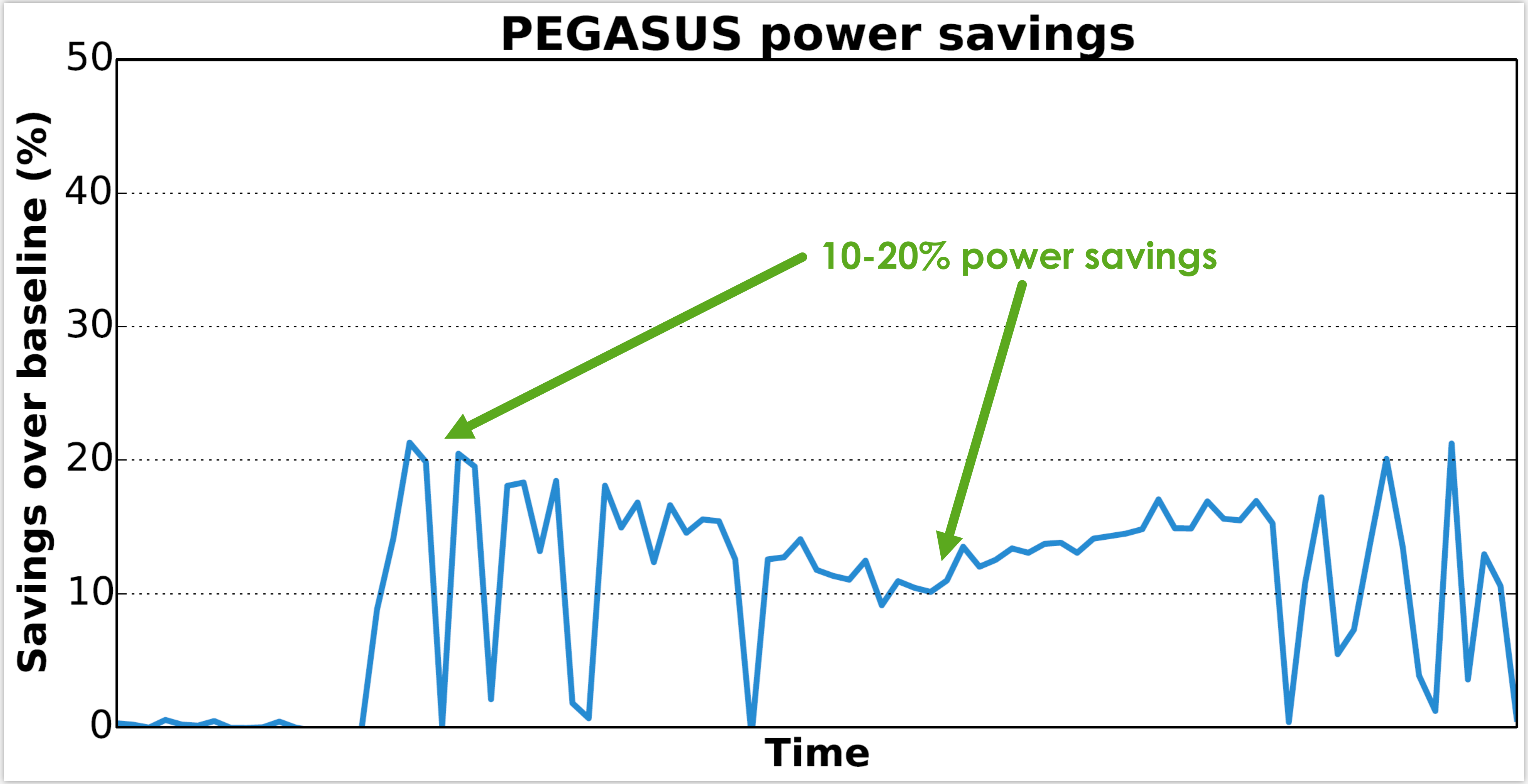
Small cluster results: power comparison



Production cluster results: power over time



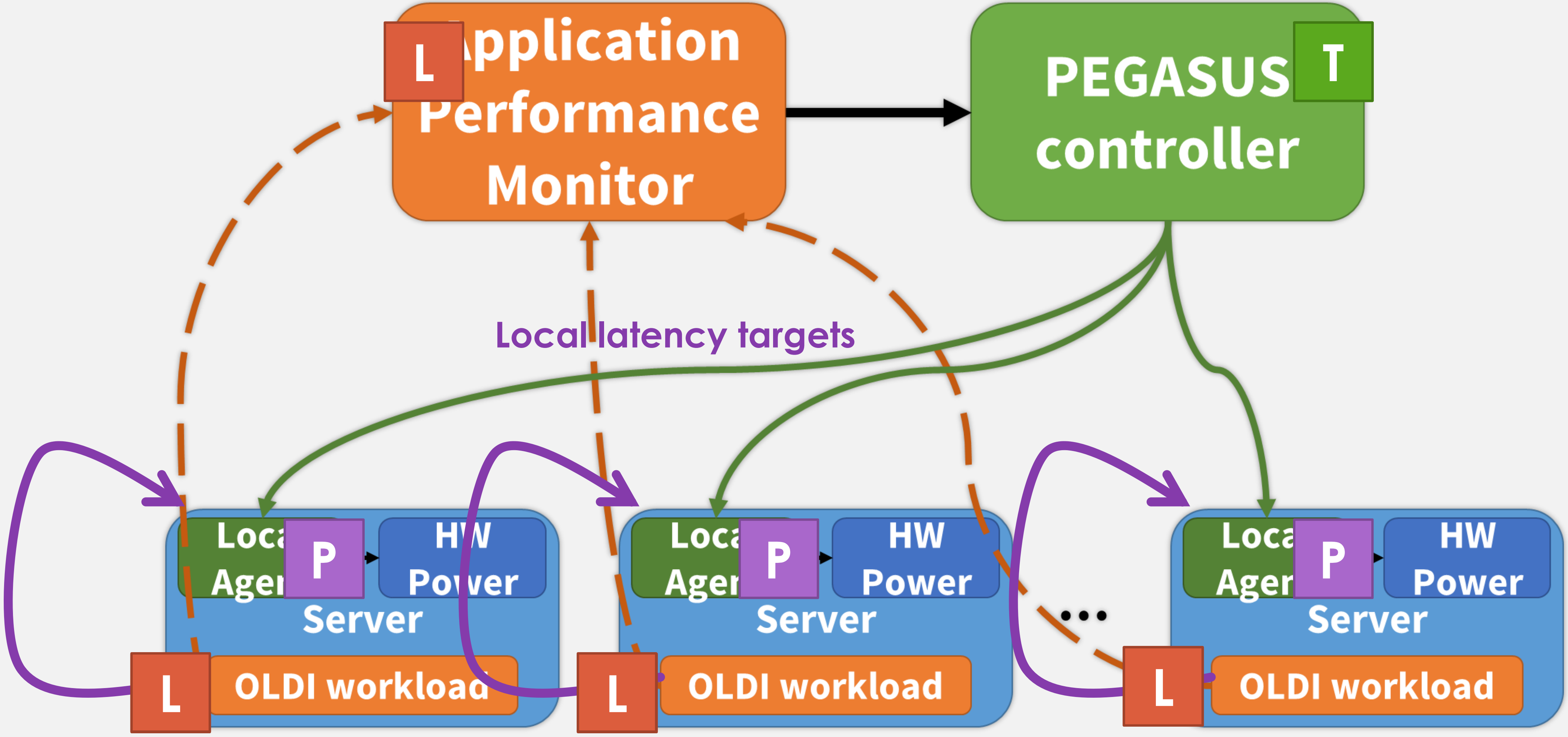
Production cluster results: power comparison



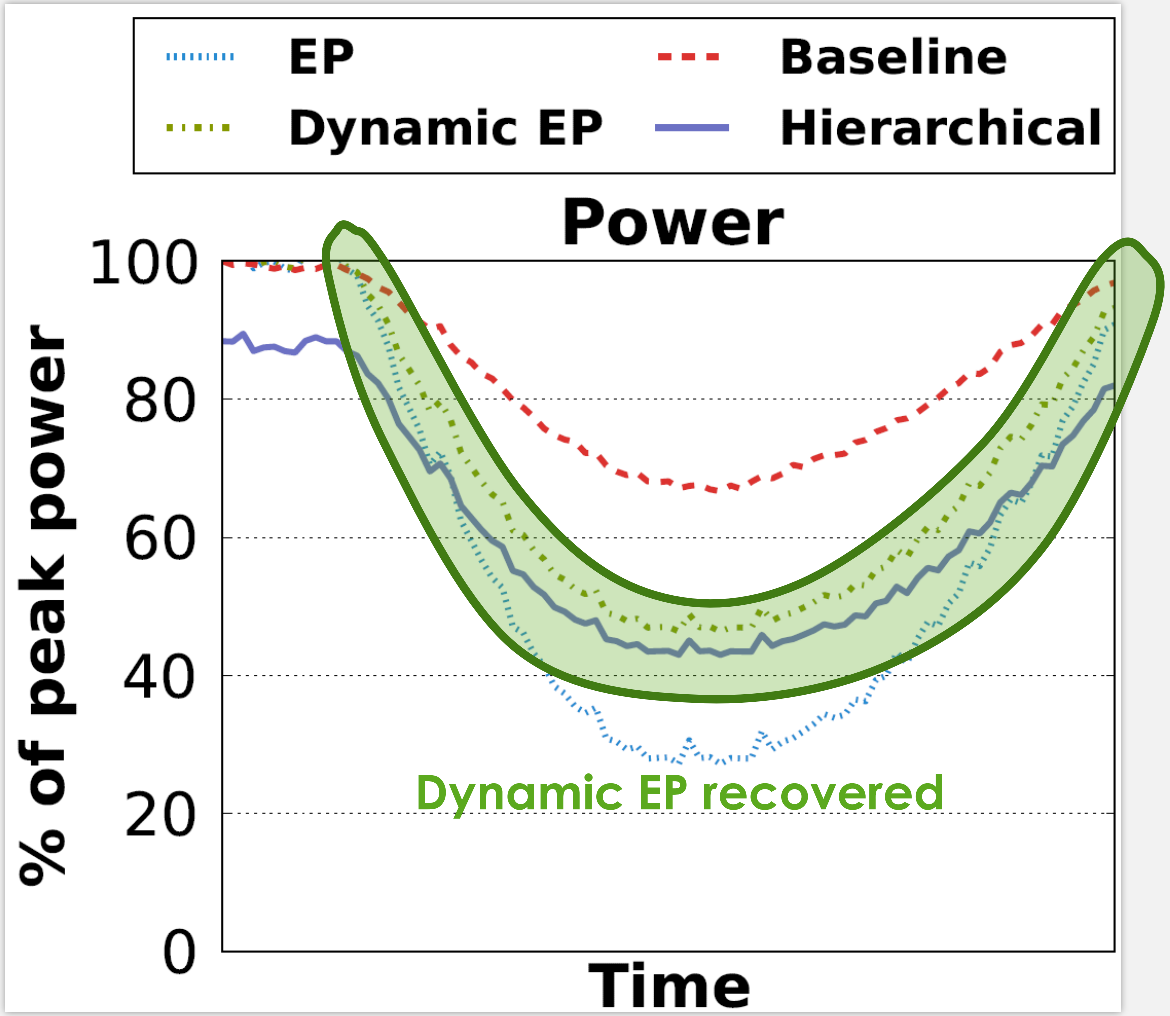
Improving PEGASUS scalability

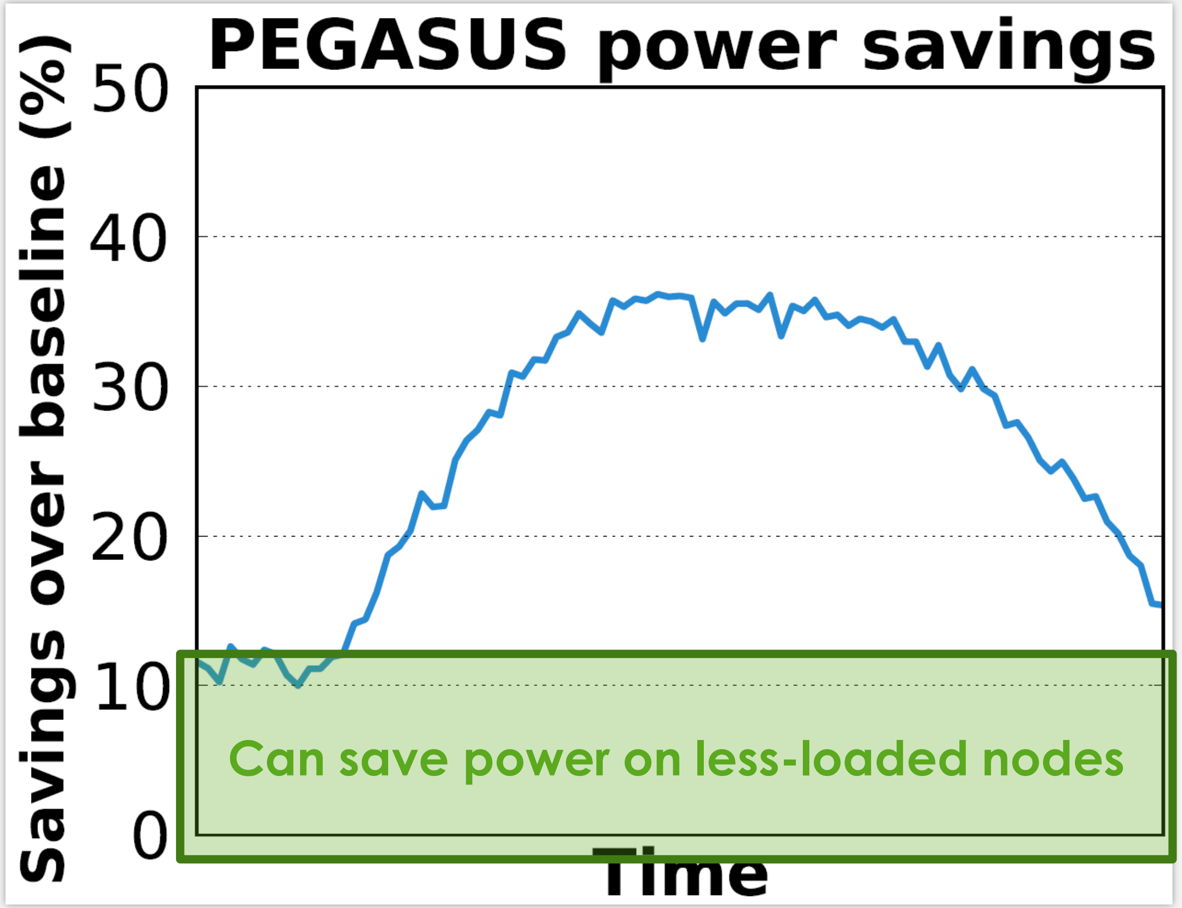
- Production cluster sees **“tail at scale”** for server utilization
  - At peak load, 0.2% nodes at 100% load while 50% nodes at <85% load
  - Caused by popular queries hitting a few shards
  - **Issue:** Hot nodes set lower bound on power limits for everyone
- **Idea**: hierarchical control
  - **Global:** sets latency targets instead of power limits
  - **Local:** decides amount of power needed to meet target latency

Hierarchical PEGASUS design



Estimated hierarchical PEGASUS results





Conclusion

- Halfway there to fully energy proportional systems
- **Iso-latency:** Use SLO metrics and fine-grain power control
  - Save up to 30% power
  - Meet/exceed energy proportionality targets
- PEGASUS achieves **iso-latency** benefits
  - Up to 20% savings on production cluster
  - Be aware of tail at scale effects