DIAL: Dynamic Interference-Aware Load Balancing

**OVERVIEW: Performance Interference**

VMs share underlying PM resources

VMs’ requests contend for resources

Performance interference

**PRIOR WORK vs OUR APPROACH**

Prior work: Provider-centric

DeepDive, TRACON, ILA, Paragon, Q-Clouds, etc.

1. Often slow and static
   - Placement, Migration
2. Application-oblivious
   - Limited visibility, SLO

Our work: User-centric

1. No host assistance
2. Complements provider
3. SLO-aware

**Key Challenge:** How to infer interference from within VM?

**CONTRIBUTION 1: Interference Model**

Total (fg+bg) host resource usage (%)

Observations:

- Non-linear increase in latencies (interference detection)
- Resource-specific curves (interference classification)
- Infer congestion from latency (interference estimation)

Modeling Results:

- Detection: $T_{90} > 5ms$ (5.7% false positive rate)
- Classification: Decision Trees (92.2% accurate)
- Inference: Queueing + Regression (93.9% accurate)

**CONTRIBUTION 2: Interference LB**

• We model each VM as an M/M/1 system

$$T_{90} \approx \frac{\sum_{i=1}^{n} p_i \cdot \ln(1 - a \cdot p_i)}{r_i \cdot \text{peak load}}$$

• Placement, Migration

**DIAL EVALUATION RESULTS**

AWS and OpenStack

CloudSuite and WikiBench

Tail latencies increase significantly under interference

**PRIOR WORK**

- VMware
- DeepDive
- TRACON
- ILA
- Paragon
- Q-Clouds

**CONTRIBUTION 2: Interference LB**

- No host assistance
- Complements provider
- SLO-aware

**DIAL EVALUATION RESULTS**

AWS and OpenStack

CloudSuite and WikiBench

- 90%ile response time

- detected

- classified

- estimated

- LB