

# Kaleidoscope: Cloud Micro-Elasticity via VM State Coloring

Roy Bryant\*, Alexey Tumanov†, Olga Irzak\*, Adin Scannell\*, Kaustubh Joshi§, Matti Hiltunen§, H. Andrés Lagar-Cavilla§, Eyal de Lara\*

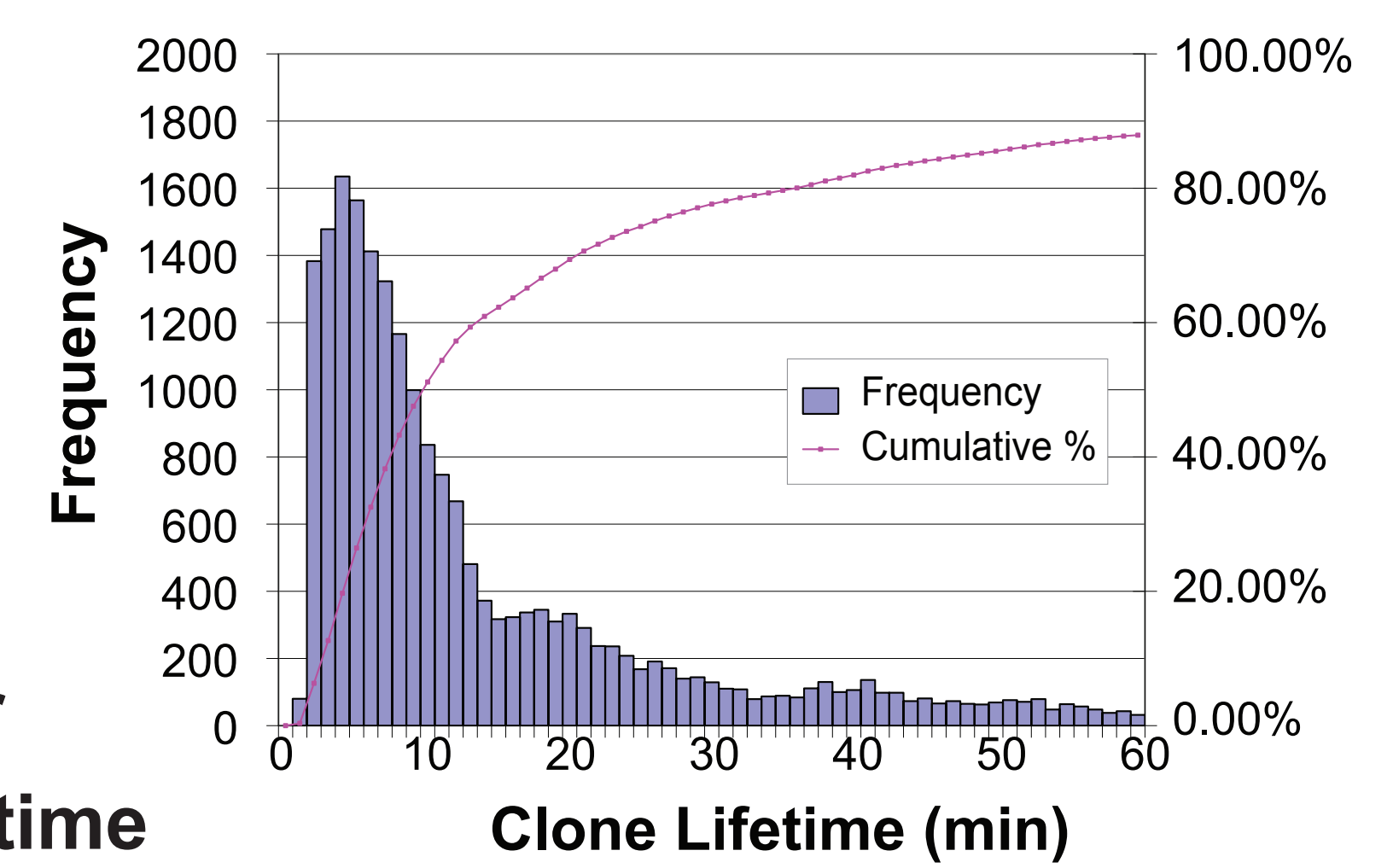
\*University of Toronto, †Carnegie Mellon University, §AT&T Labs Research

## Current Elasticity Models: Issues

- Long instantiation times
- Coarse granularity of footprint
- Network bandwidth & latency overhead
  - Common practice: dedicated migration bandwidth!
  - Wide-range variability in instantiation times
- Fresh workers → cold state
- Number of servers with runtime > month – doubled

## Motivation

- Strong need for elasticity
  - 15.3% of peak capacity – average usage
  - Wide-range short-term variability in workload
- Elastic workers – short lived
  - 85% of workers – live < 1hr
  - 10 min → mean worker lifetime

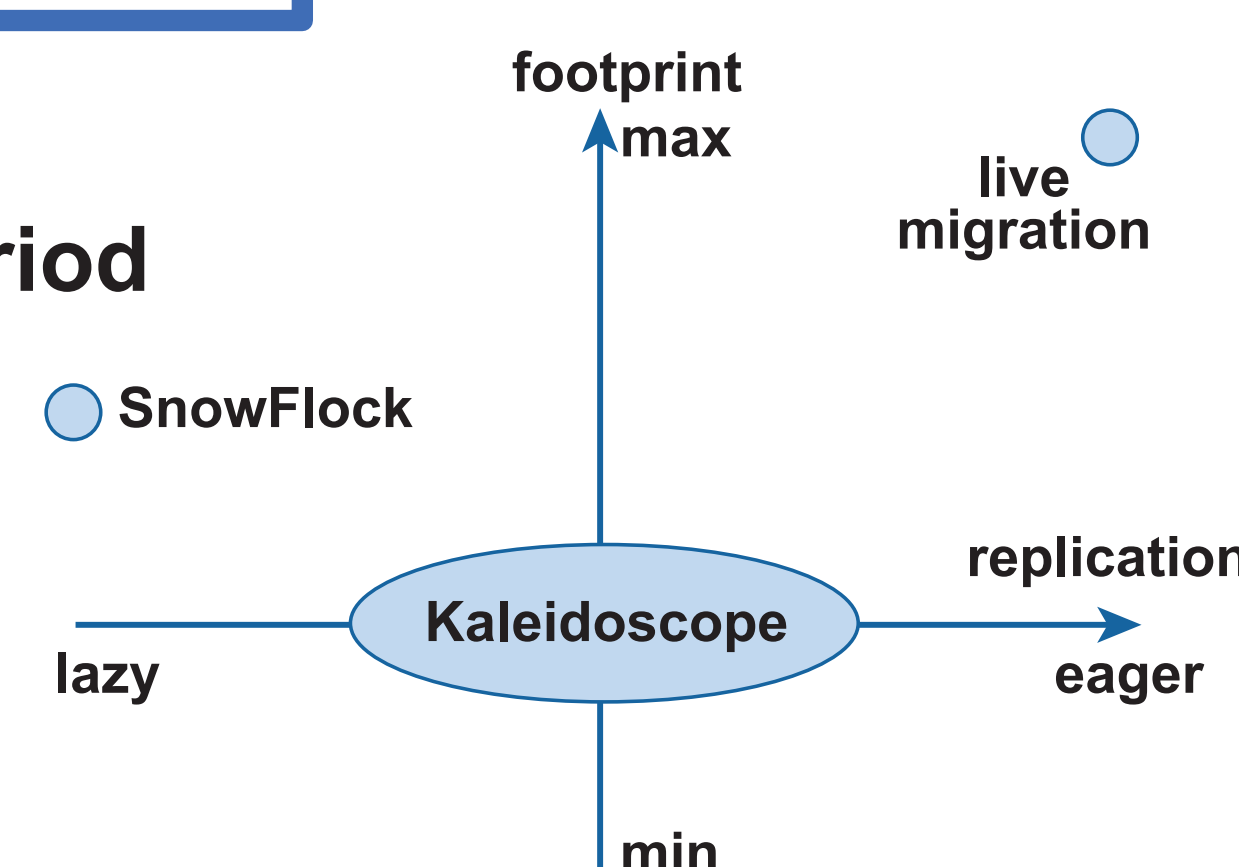


## Kaleidoscope Vision: Cloud $\mu$ -elasticity

- Allocate resources proportionally to the working set
- Stateful replication of VMs (inherit expensive warm state)
- Color-guided state propagation
  - proactive prefetch
  - reactive (on-demand)
- Continuum between full copy and minimal/on demand copy

## Kaleidoscope Design Space

- Fundamental tradeoff:
  - Instantiation time vs. warmup period
- Continuum between lazy and eager replication
- Continuum between minimal and maximal footprint



## Uses of Color

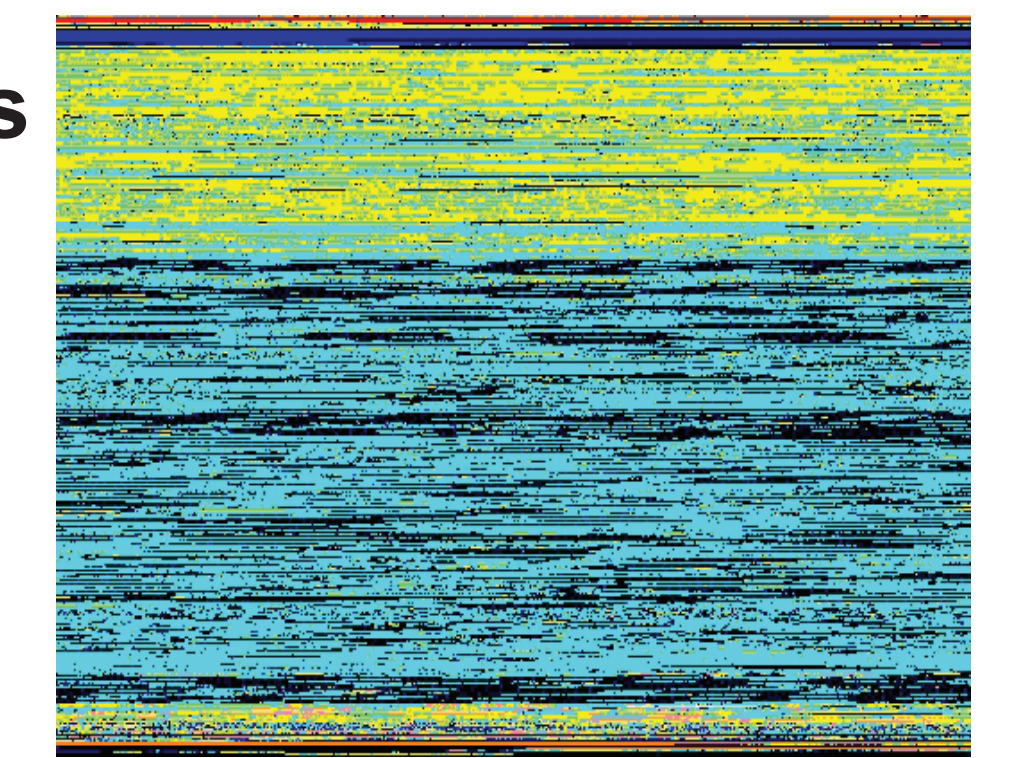
- State prefetch of consecutive pages in **color space**
  - Significant reduction in bandwidth consumption
- Per-color differentiation in the width of the prefetch window
  - *Insight*: reduced window size for executable state
  - User/kernel data windows : 3-4 times wider than executable
- Color-directed memory deduplication
  - Per-color probabilities of inter-VM page similarities
  - One-time, color guided calculation of hashes for the master memory state

## Kaleidoscope Contributions

1. VM memory state coloring
  - Semantically-aware state propagation
  - Architectural & introspective runtime memory state information
2. Time & space efficient implementation of coloring
3. Micro-elastic cloud server
  - Color-aware replication & sharing
  - Near-ideal post-clone QoS
  - Resource consumption proportional to demand
4. Real world datacenter savings

## VM Memory State Coloring

- Architecture-based coloring
  - Performed on translated page table pages undergoing mandatory canonicalization at clone time
  - Extracting the NX bit (executable)
  - Identifying the kernel/user-space split
- Introspective coloring
  - Performed on a "frozen" memory image of the master
  - File cache radix trees
  - Frame page structure: used/unused physical pages

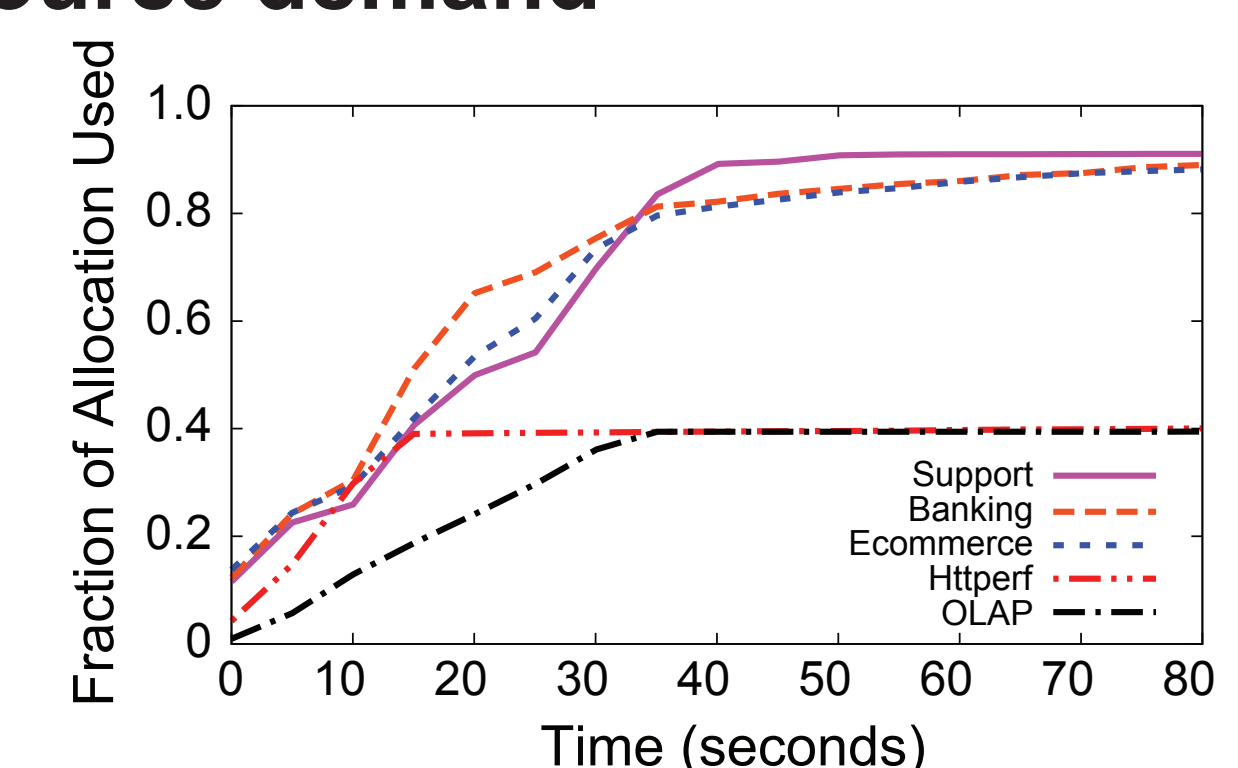


## Evaluation Categories

- Blocking time reduction
- Elastic footprint / resource preservation
- Scalability
- Runtime
- QoS

## Fractional Footprint

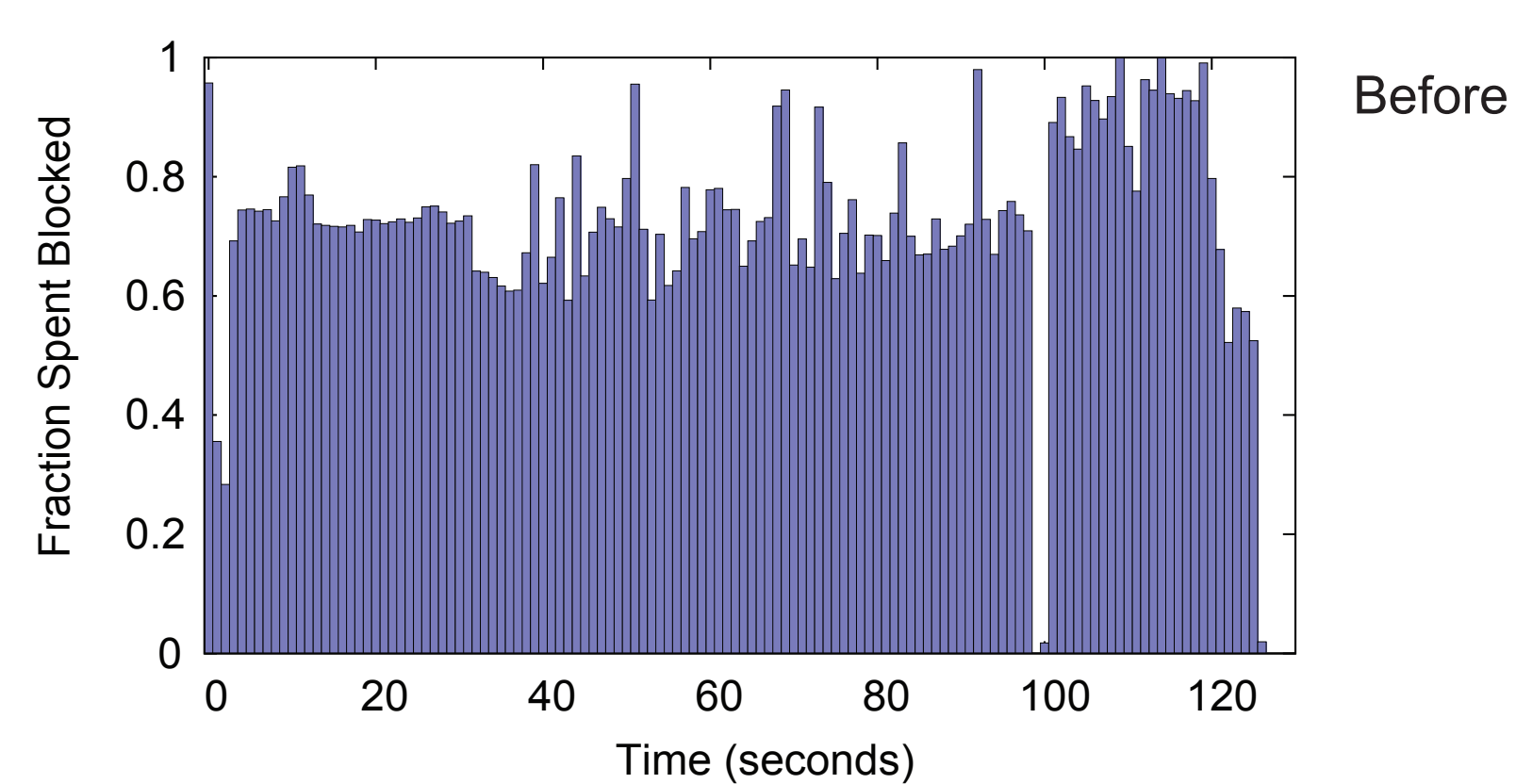
- Definition: ability to consume physical resources at sub-VM granularity, tracking effective resource demand
  - Physical host memory
  - Network bandwidth
- Description: Kaleidoscope workers allocate memory as necessary, an advantage when spikes are short lived.



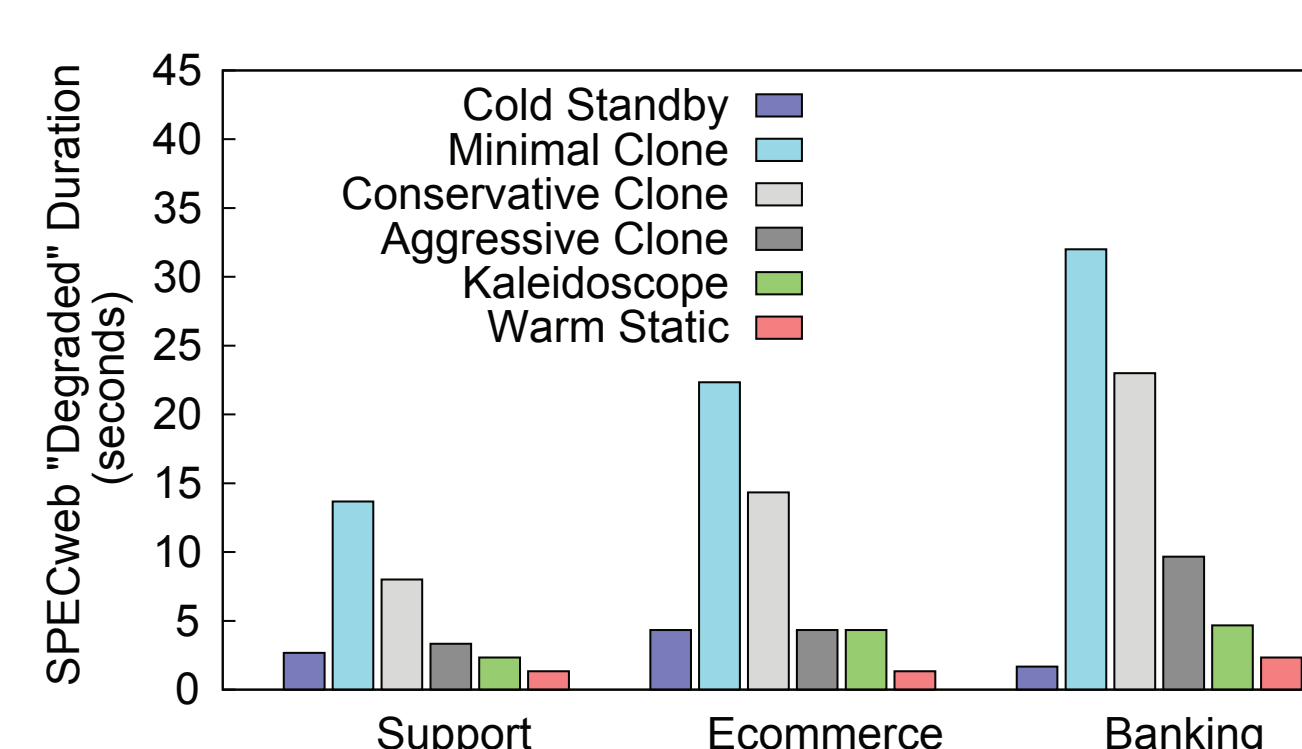
Traces in this figure include transferred state plus new allocations.

## Evaluation Results

### Reduction of page faults and blocking

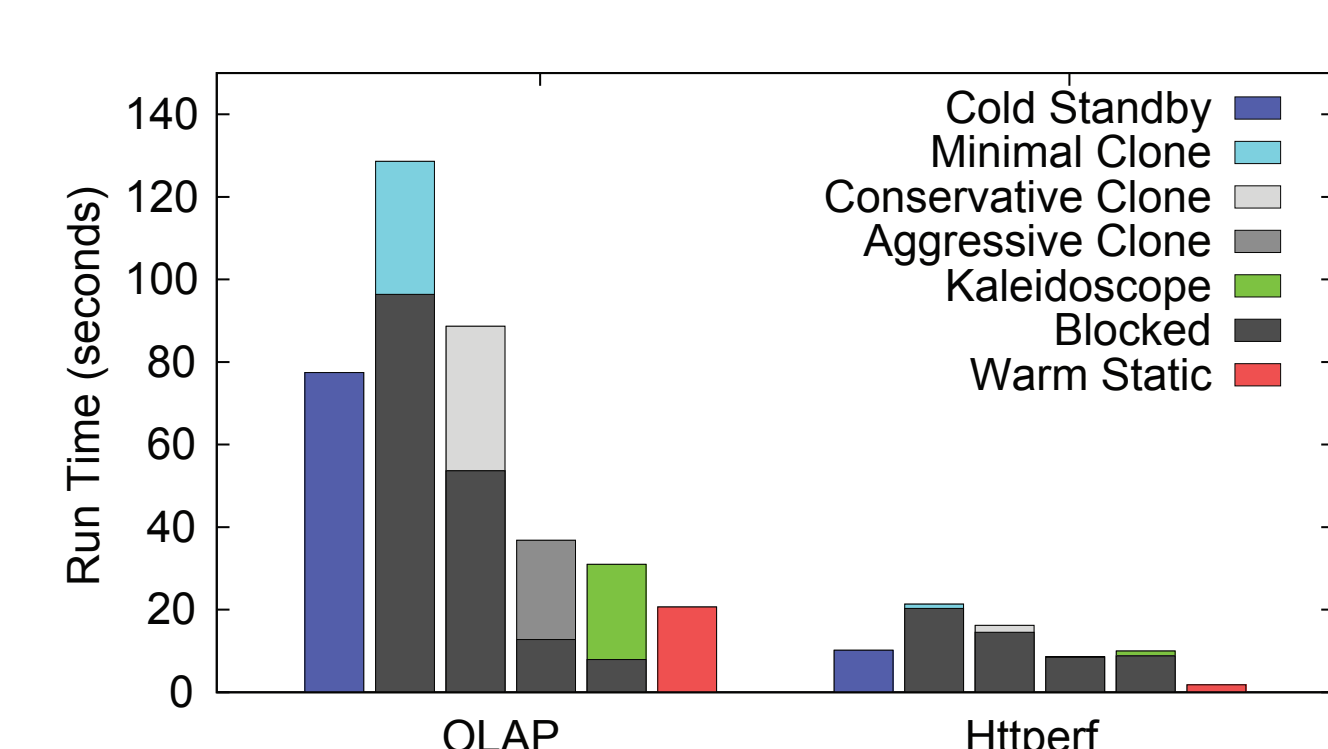


### Near-optimal QoS



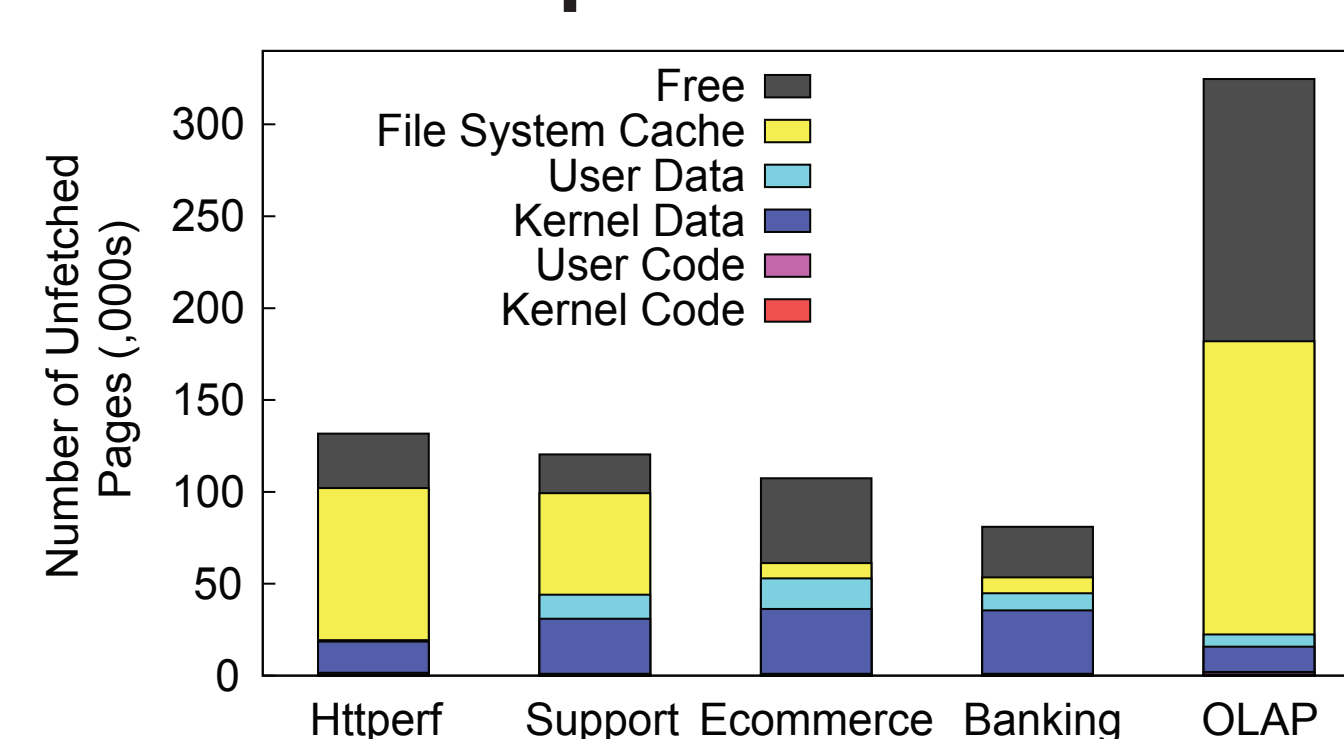
Prefetching reduces the length of time that prototypes fail to meet SPECweb's minimum acceptable QoS.

### Runtime



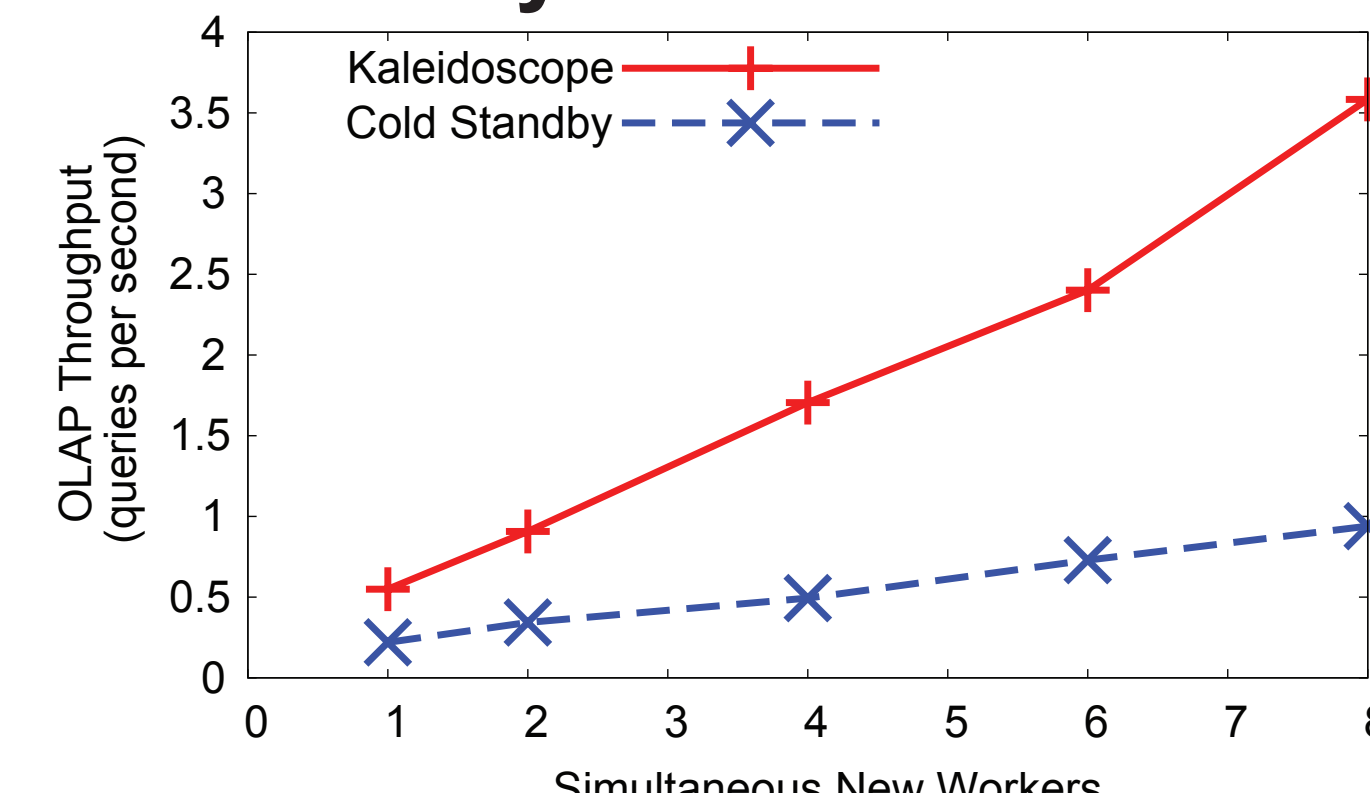
Runtime approaches near-optimal (warm static) as VCPU blocking and page faults are anticipated by color-directed prefetch and eliminated from the critical path.

### Resource preservation



Kaleidoscope workers avoid unnecessary pages. Unfetched pages save precious bandwidth and are mostly file system cache and a free list.

### Scalability



The database's OLAP throughput scales well with the number of simultaneous clones