Storage-Aware Caching:
Revisiting Caching for Heterogeneous Systems

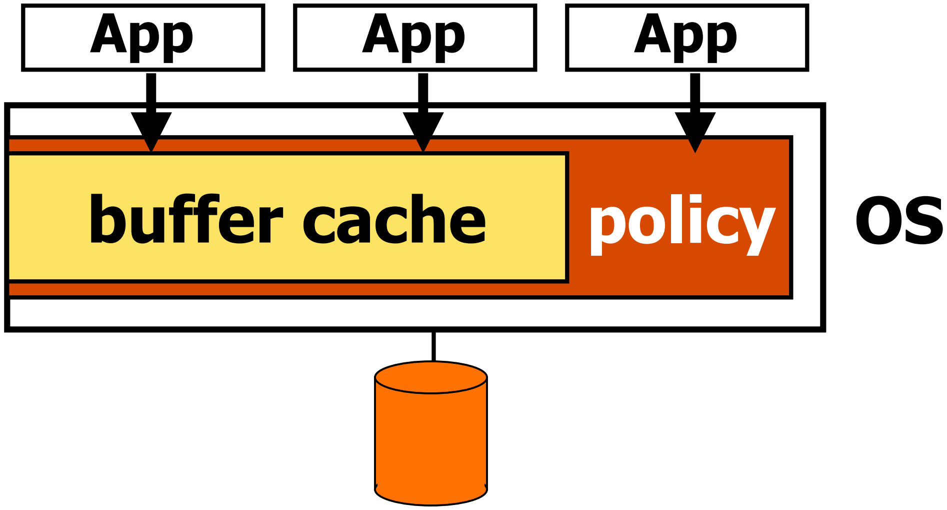
Brian Forney

Andrea Arpaci-Dusseau **Remzi Arpaci-Dusseau**

Wisconsin Network Disks

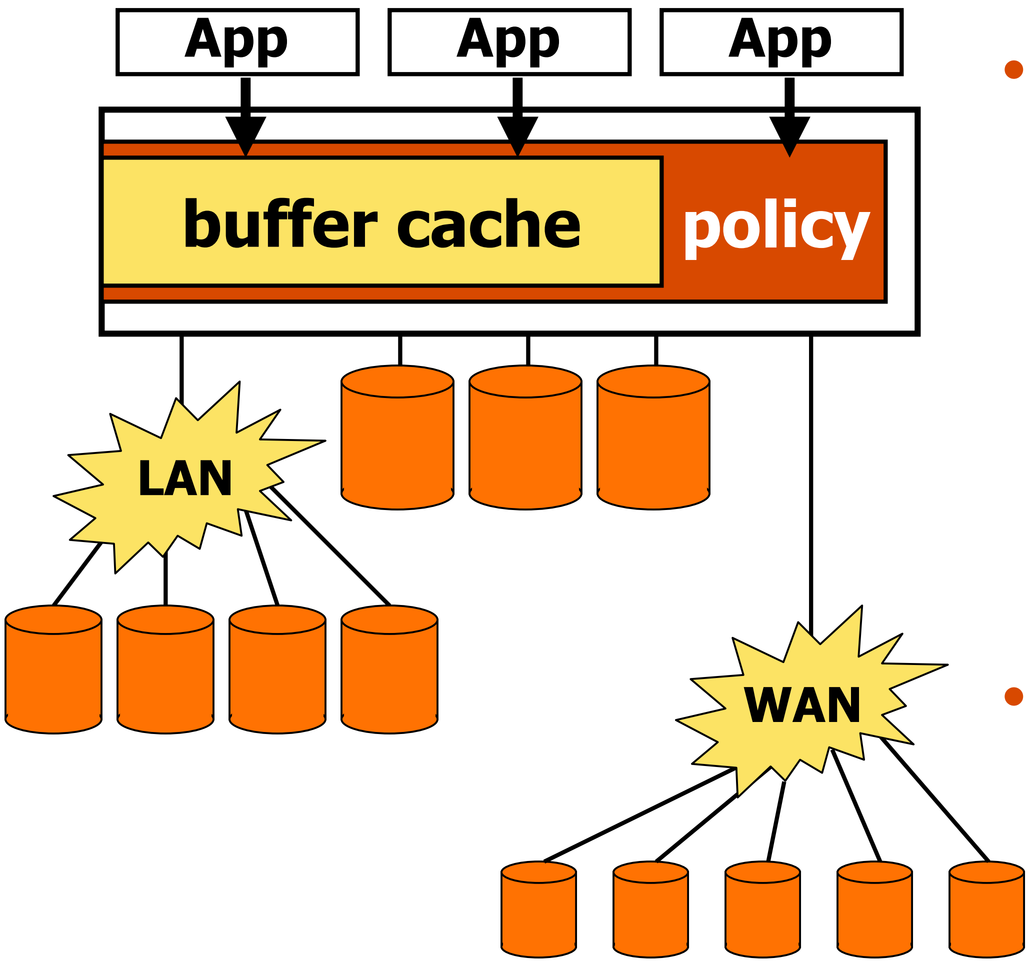
University of Wisconsin at Madison

Circa 1970



- Intense research period in policies
- Wide variety developed; many used today
  - Examples: Clock, LRU
- Simple storage environment
  - Focus: workload
  - Assumption: consistent retrieval cost

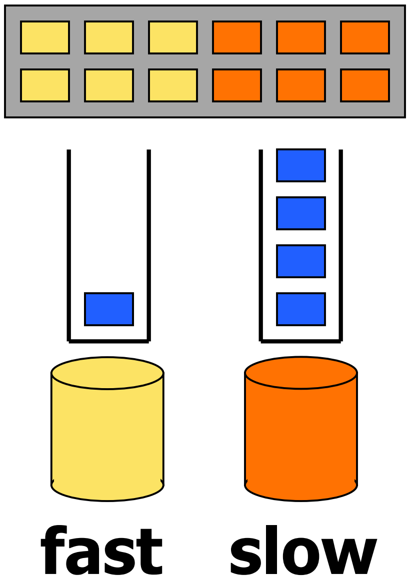
Today



- Rich storage environment
  - Devices attached in many ways
  - More devices
  - Increased device sophistication
- Mismatch: Need to reevaluate

Problem illustration

- Uniform workload
- Two disks
- LRU policy



- Slow disk is bottleneck
- Problem: Policy is oblivious
  - Does not filter well

General solution

- Integrate workload and device performance
  - Balance *work* across devices
  - Work: cumulative delay
- Cannot throw out:
  - Existing non-cost aware policy research
  - Existing caching software

Our solution: Overview

- Generic partitioning framework
  - Old idea
  - One-to-one mapping: device ⇔ partition
  - Each partition has cost-oblivious policy
  - Adjust partition sizes
- Advantages
  - Aggregates performance information
  - Easily and quickly adapts to workload and device performance changes
  - Integrates well with existing software
- Key: How to pick the partition sizes

Outline

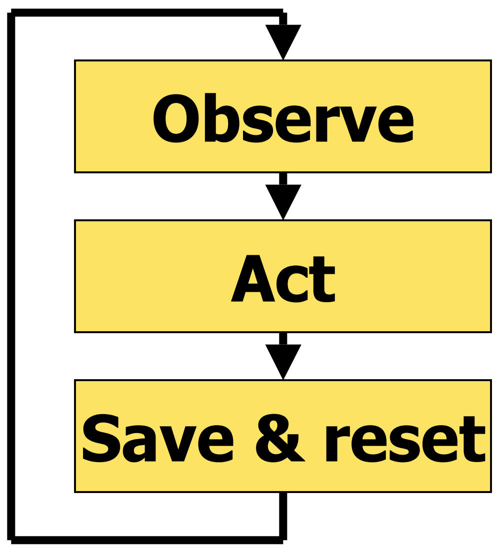
- ~~Motivation~~
- ~~Solution overview~~
- Taxonomy
- Dynamic partitioning algorithm
- Evaluation
- Summary

Partitioning algorithms

- Static
  - Pro: Simple
  - Con: Wasteful
- Dynamic
  - Adapts to workload
    - Hotspots
    - Access pattern changes
  - Handles device performance faults
- Used dynamic

Our algorithm: Overview

1. **Observe:** Determine per-device cumulative delay
2. **Act:** Repartition cache
3. **Save & reset:** Clear last W requests



Algorithm: Observe

- Want accurate system balance view
- Record per-device cumulative delay for last W completed disk requests
  - At client
  - Includes network time

Algorithm: Act

- Categorize each partition
  - Page consumers
    - Cumulative delay above threshold → possible bottleneck
  - Page suppliers
    - Cumulative delay below mean → lose pages without decreasing performance
  - Neither
- Always have page suppliers if there are page consumers

Page consumers

- How many pages? Depends on state:
  - Warming
    - Cumulative delay increasing
    - Aggressively add pages: reduce queuing
  - Warm
    - Cumulative delay constant
    - Conservatively add pages
  - Cooling
    - Cumulative delay decreasing
    - Do nothing; naturally decreases

Dynamic partitioning

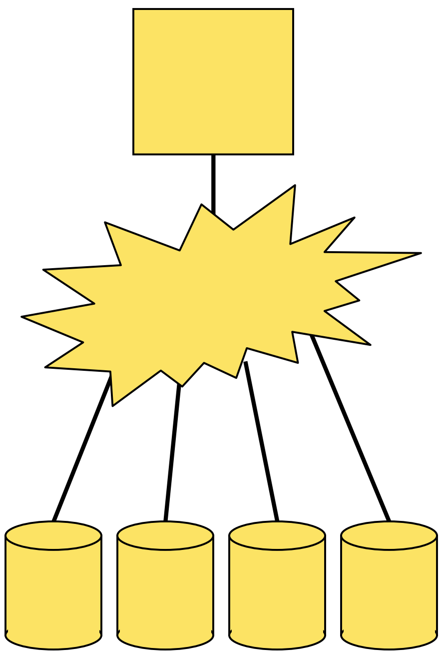
- Eager
  - Immediately change partition sizes
  - Pro: Matches observation
  - Con: Some pages temporarily unused
- Lazy
  - Change partition sizes on demand
  - Pro: Easier to implement
  - Con: May cause over correction

Outline

- ~~Motivation~~
- ~~Solution overview~~
- ~~Taxonomy~~
- ~~Dynamic partitioning algorithm~~
- Evaluation
- Summary

Evaluation methodology

- Simulator



Caching, RAID-0 client

LogGP network
- With endpoint contention

Disks
- 16 IBM 9LZX
- First-order model: queuing, seek time & rotational time

- Workloads: synthetic and web

Evaluation methodology

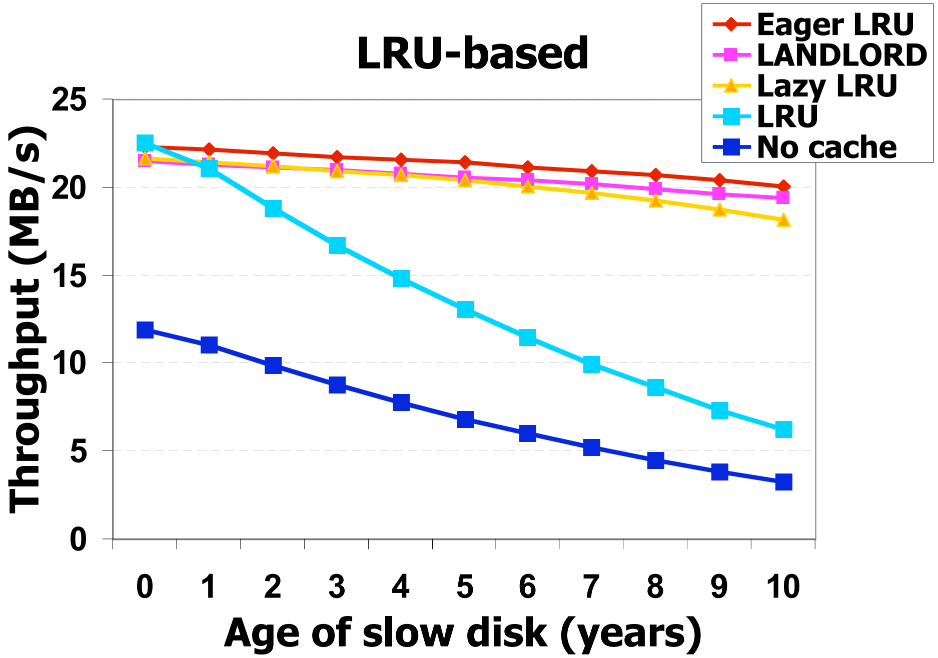
- Introduced performance heterogeneity
  - Disk aging
    - Used current technology trends
      - Seek and rotation: 10% decrease/year
      - Bandwidth: 40% increase/year
    - Scenarios
      - Single disk degradation: Single disk, multiple ages
      - Incremental upgrades: Multiple disks, two ages
  - Fault injection
    - Understand dynamic device performance change and device sharing effects
- Talk only shows single disk degradation

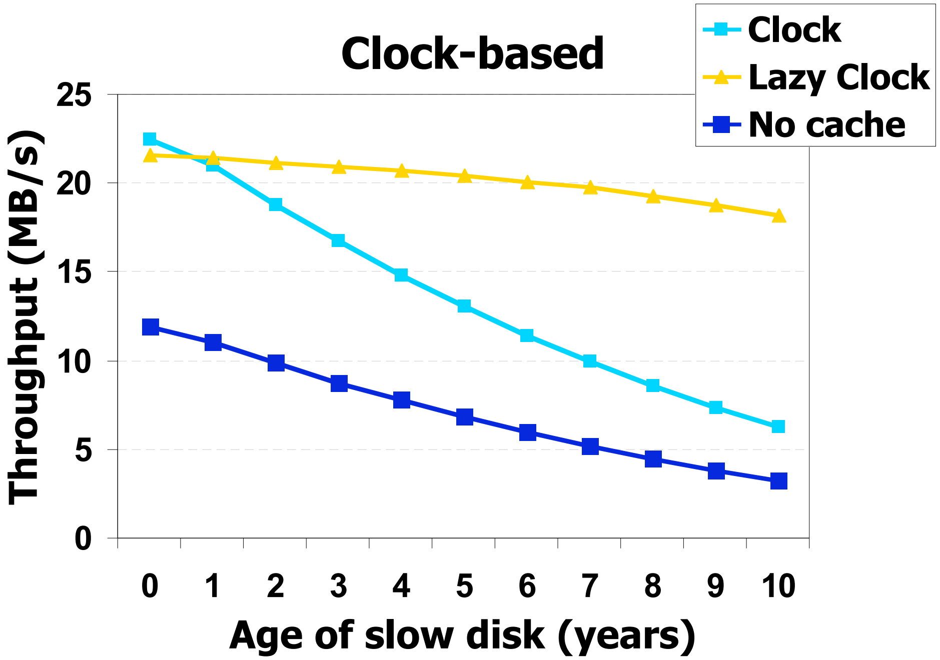
Evaluated policies

- Cost-oblivious: LRU, Clock
- Storage-aware: Eager LRU, Lazy LRU, Lazy Clock (Clock-Lottery)
- Comparison: LANDLORD
  - Cost-aware, non-partitioned LRU
  - Same as web caching algorithm
  - Integration problems with modern OSes

Synthetic

- Workload: Read requests, exponentially distributed around 34 KB, uniform load across disks

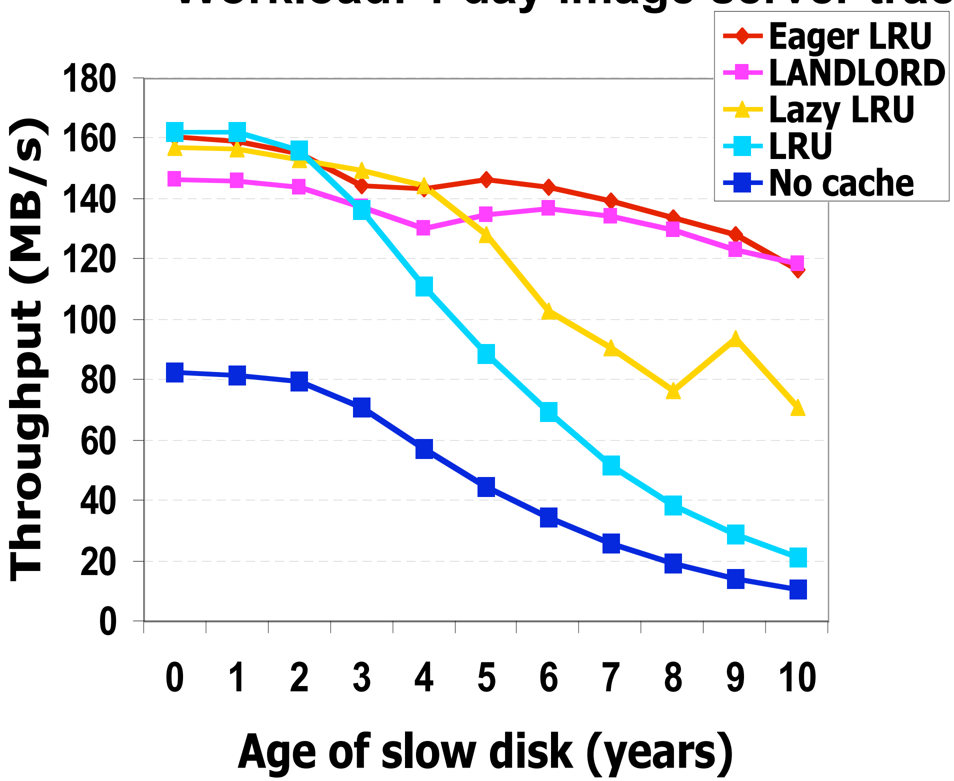


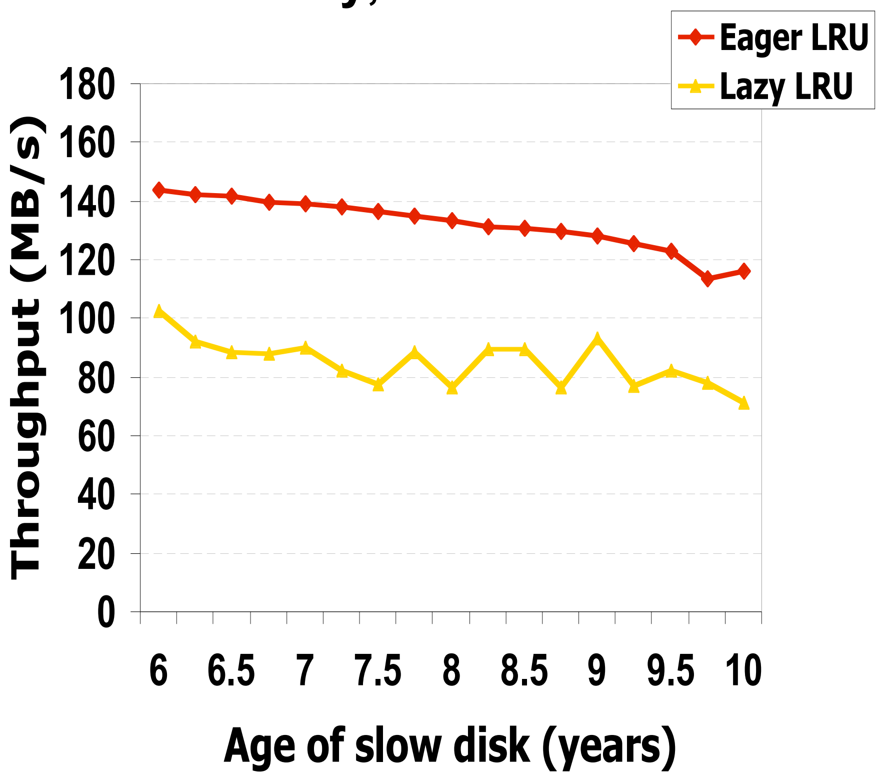


- A single slow disk greatly impacts performance.
- Eager LRU, Lazy LRU, Lazy Clock, and LANDLORD robust as slow disk performance degrades

Web

- Workload: 1 day image server trace at UC Berkeley, reads & writes





- Eager LRU and LANDLORD are the most robust.

Summary

- Problem: Mismatch between storage environment and cache policy
  - Current buffer cache policies lack device information
- Policies need to include storage environment information
- Our solution: Generic partitioning framework
  - Aggregates performance information
  - Adapts quickly
  - Allows for use of existing policies

Questions?

More information at www.cs.wisc.edu/wind

Future work

- Implementation in Linux
- Cost-benefit algorithm
- Study of integration with prefetching and layout

Problems with LANDLORD

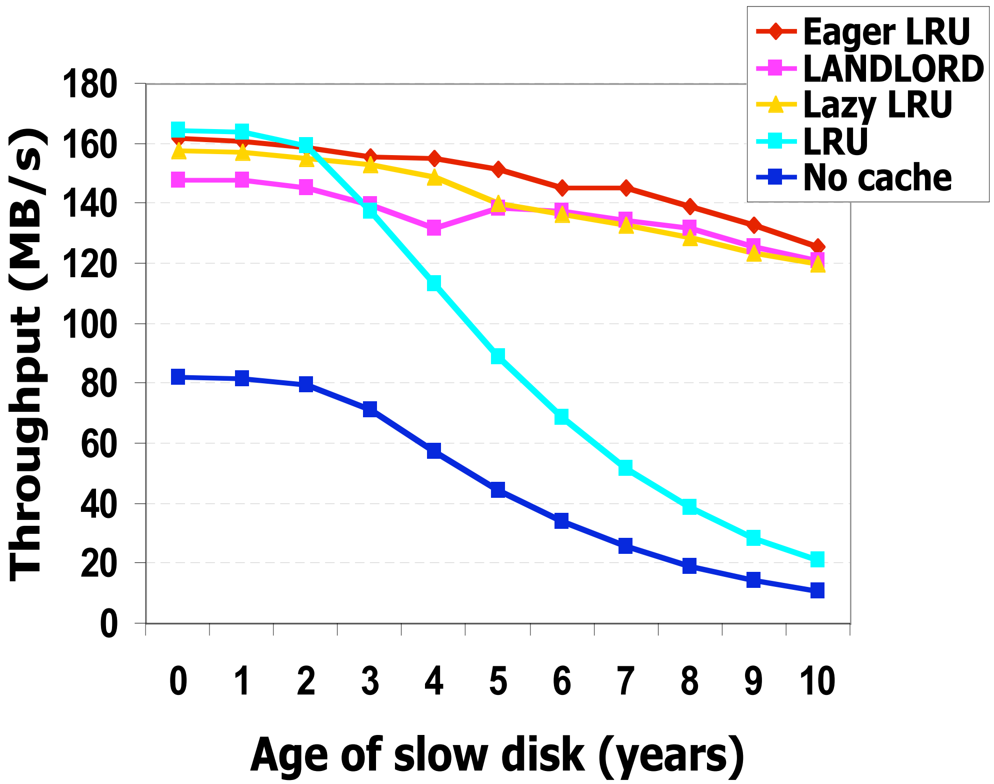
- Does not mesh well with unified buffer caches (assumes LRU)
- LRU-based: not always desirable
  - Example: Databases
- Suffers from a “memory effect”
  - Can be much slower to adapt

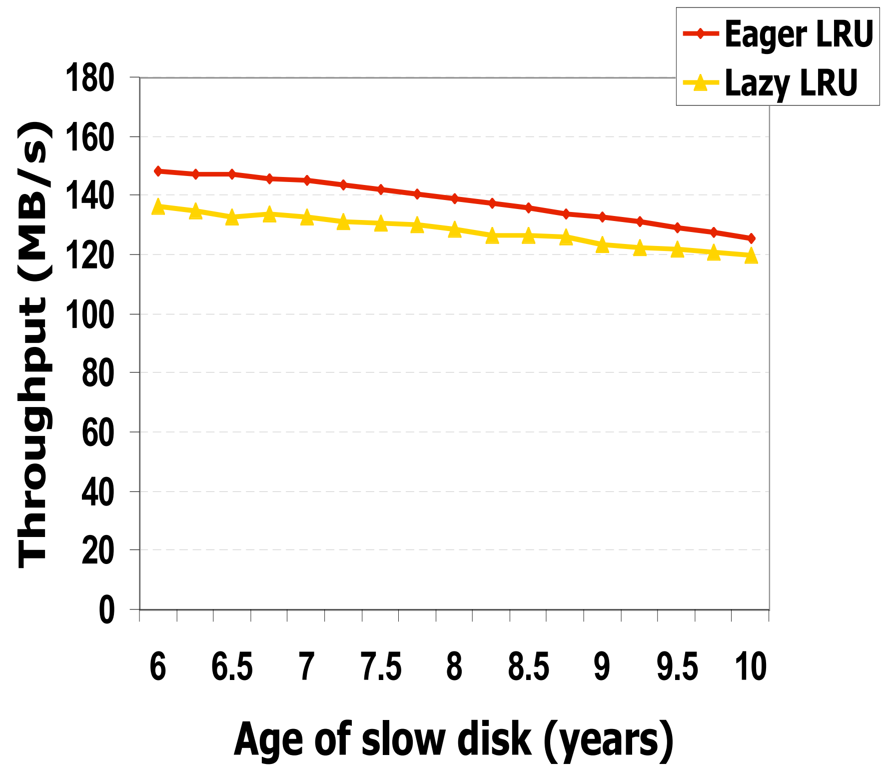
Disk aging of an IBM 9LZX

Age (years)	Bandwidth (MB/s)	Avg. seek time (ms)	Avg. rotational delay (ms)
0	20.0	5.30	3.00
1	14.3	5.89	3.33
2	10.2	6.54	3.69
3	7.29	7.27	4.11
4	5.21	8.08	4.56
5	3.72	8.98	5.07
6	2.66	9.97	5.63
7	1.90	11.1	6.26
8	1.36	12.3	6.96
9	0.97	13.7	7.73
10	0.69	15.2	8.59

Web without writes

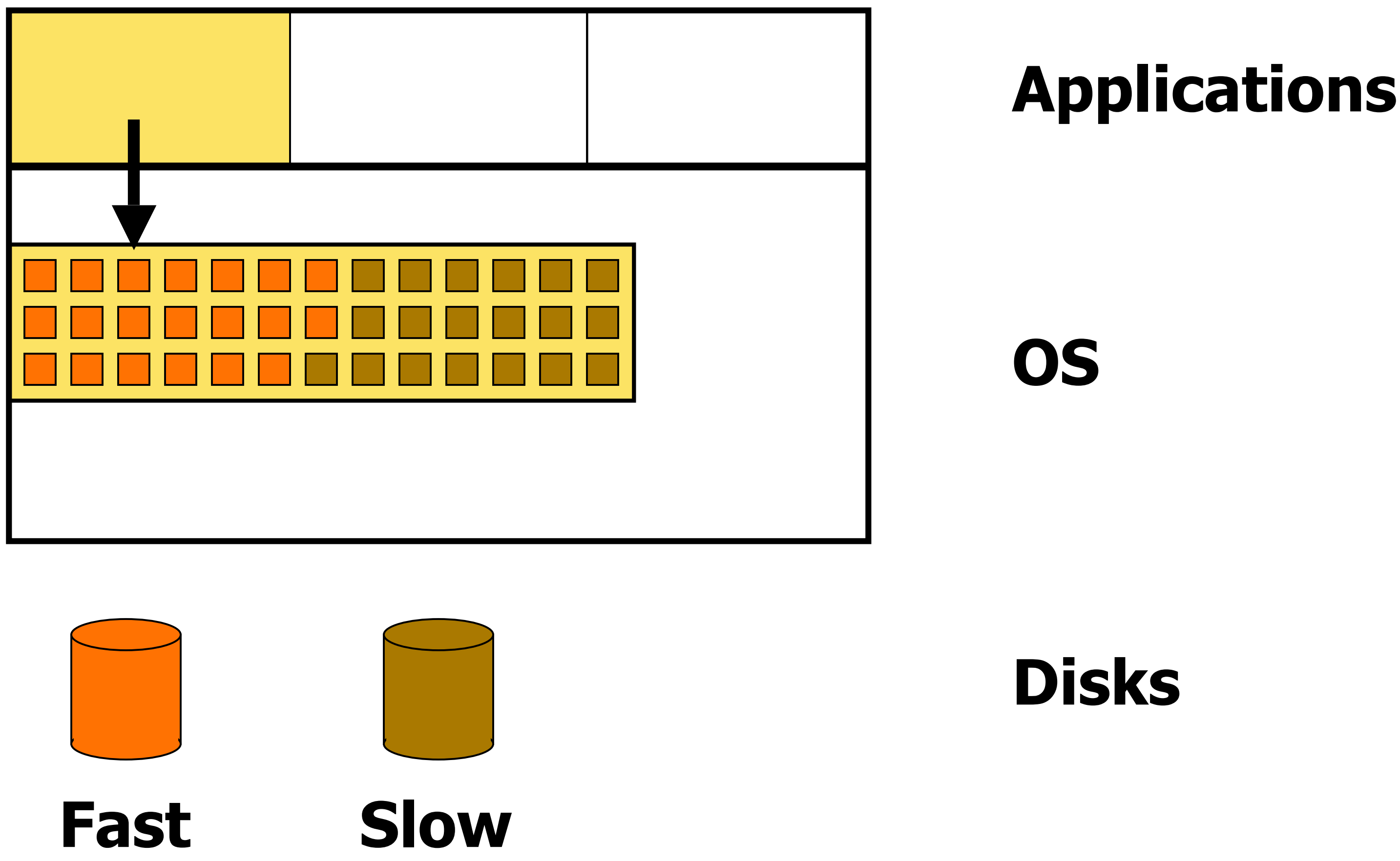
- Workload: Web workload where writes are replaced with reads



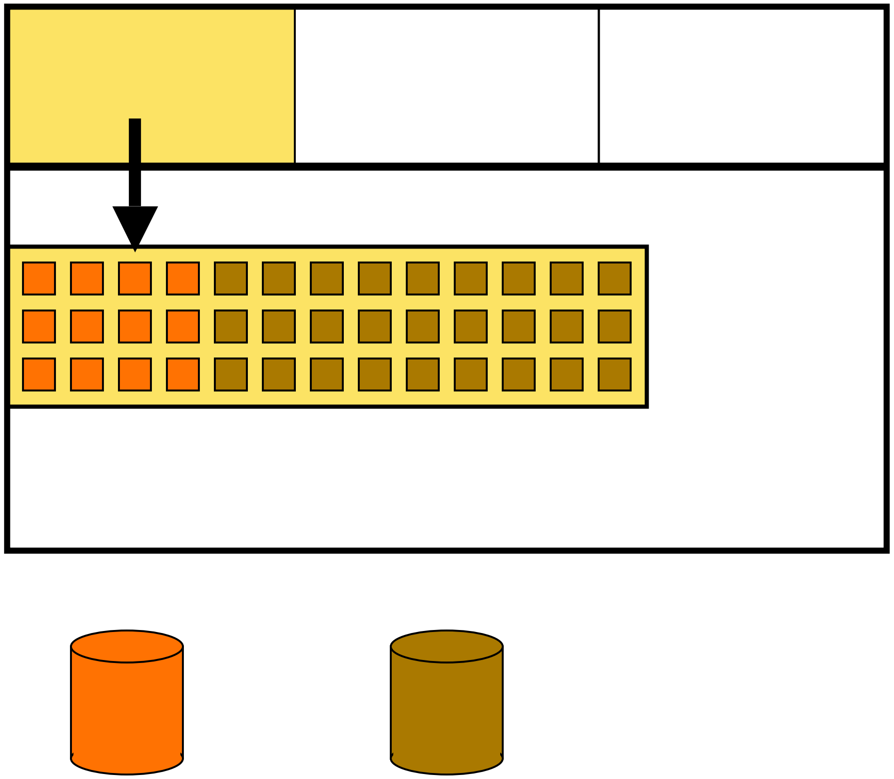


- Eager LRU and LANDLORD are the most robust.

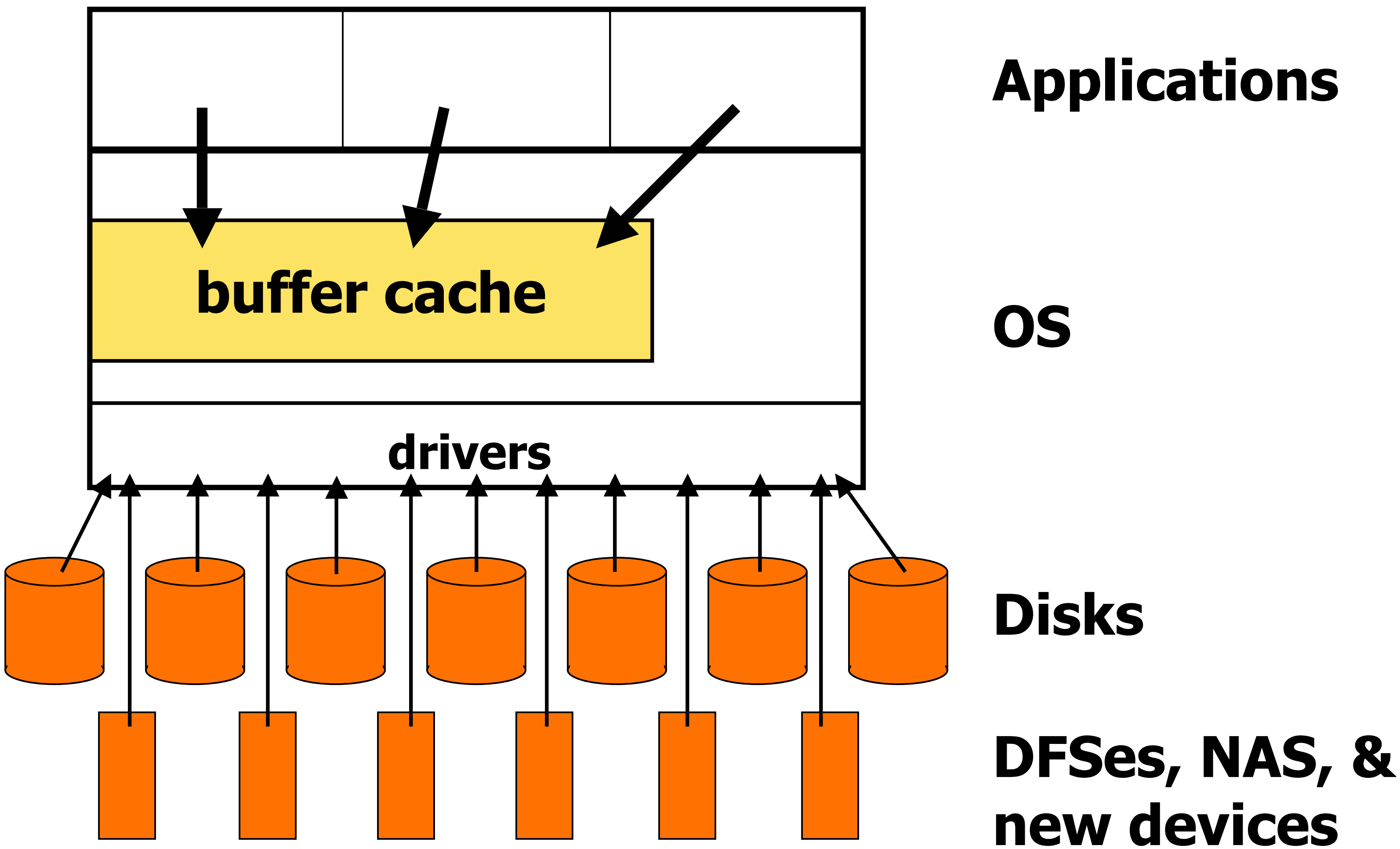
Problem illustration



Problem illustration

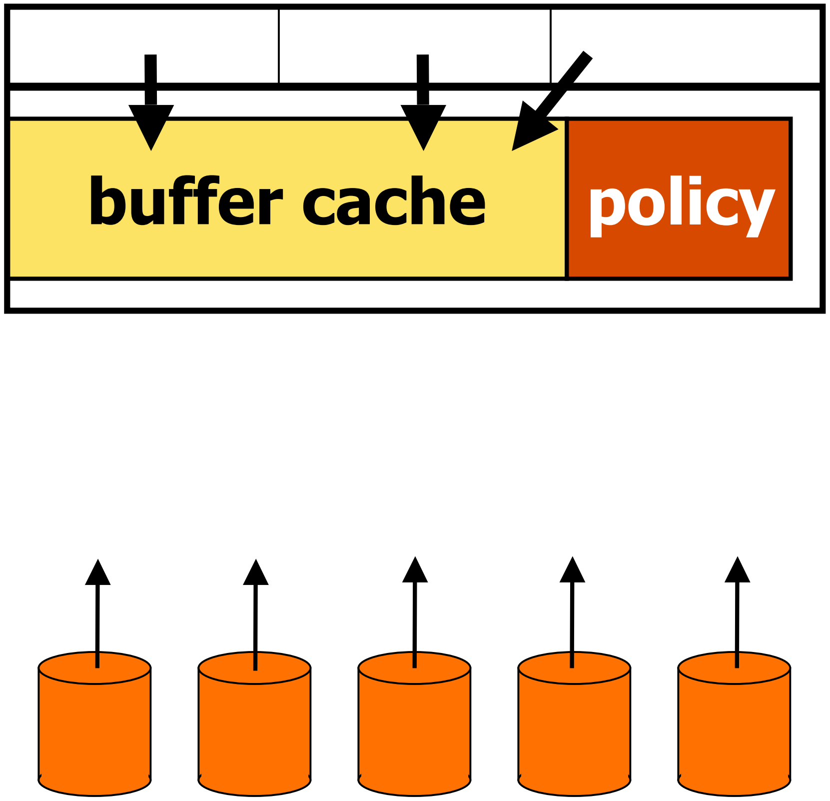


Lack of information



Solution: overview

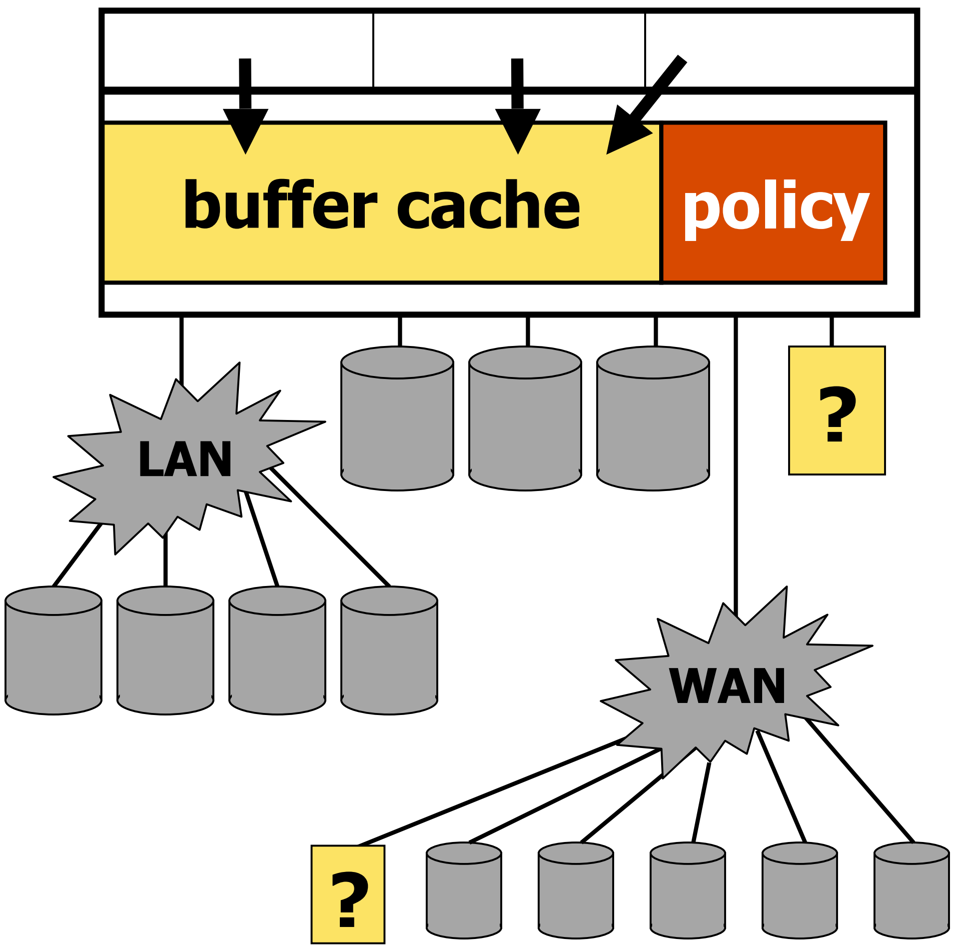
- Partition the cache
  - One per device
  - Cost-oblivious policies (e.g. LRU) in partitions
  - Aggregate device perf.
- Dynamically reallocate pages



Forms of dynamic partitioning

- Eager
  - Change sizes

Tomorrow



- New devices
- New paradigms
- Increasingly rich storage environment
- Mismatch: Reevaluation needed

WiND project

- Wisconsin Network Disks
- Building manageable distributed storage
- Focus: Local-area networked storage
- Issues similar in wide-area