• Modern servers are as powerful as distributed systems in the past
  ✓ CPU and storage devices are parallel, similar to distributed nodes
  ✓ Goal is to trade-off consistency and performance in a local store
  ✓ Use of stale data in different storage devices for better performance

StaleStore
• Local storage systems in any form that can trade-off consistency and performance
  (e.g. KV-store, filesystem, block store, DB, etc.)

Requirements:
1. Maintain multiple versions of data
   - Should have interface to access older versions
2. Aware of consistency semantics
   - Bounded Staleness, monotonic-reads, read-my-writes, etc.
3. Can give cost estimates for accessing each version
   - Considerations for data locations and storage conditions

Multi-Block Object Access in Yogurt
• Key-value stores, filesystems can store an object over multiple blocks
• Read should be served from a persistent snapshot: GetVersionRange()

GetCost Overhead
• Single disk log-structured store
• SSD flash translation layers
• Log-structured arrays
• Durable write caches that are fast for writes but slow for reads
• Deduplicated systems with read caches
• Fine-grained logging over a block-grained cache
• Systems storing differences from previous versions

Other Possible StaleStores
• Modern servers are similar to distributed systems
• Local storage systems can adopt weak consistency
  ✓ We define them as StaleStores
• Yogurt, a block level StaleStore
  ✓ Effectively trades-off consistency and performance
  ✓ Supports high level multi-block data constructs

Summary
Towards Weakly Consistent Local Storage Systems
Ji-Yong Shin1,2, Mahesh Balakrishnan2, Tudor Marian3, Jakub Szefer3 and Hakim Weatherspoon1
1 Cornell University, 2 Yale University, 3 Google