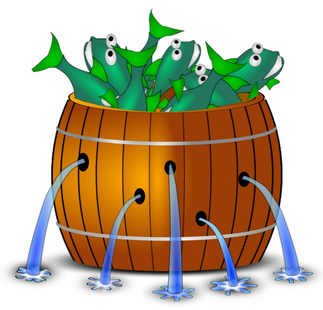
OS design for non-cache-coherent systems

Simon Peter, Jana Giceva, Pravin Shinde, Gustavo Alonso, Timothy Roscoe
Systems Group, Department of Computer Science, ETH Zurich

www.barrelfish.org

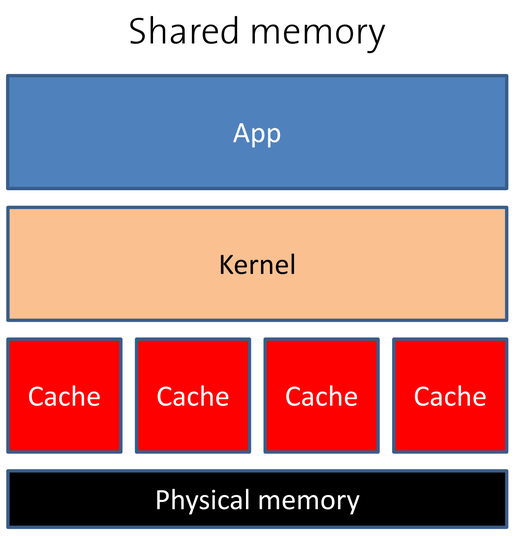


Motivation

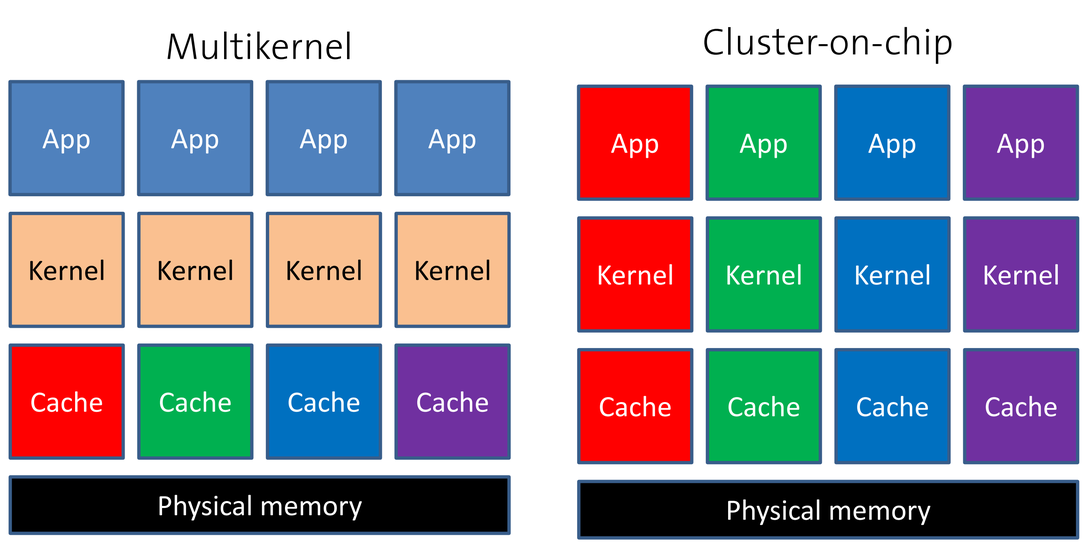
- **Hardware cache coherence might not be around forever**
  - Core counts increase
  - Cache coherence has to scale
- Research chips already experiment with non-coherent caches
  - **Intel's Single-Chip Cloud Computer**
  - Microsoft's Beehive processor
- **Today's shared-memory OSes do not run on non-coherent hardware**
  - Cache coherence required for shared-memory data structures

What's a good alternative OS design?

Today's design



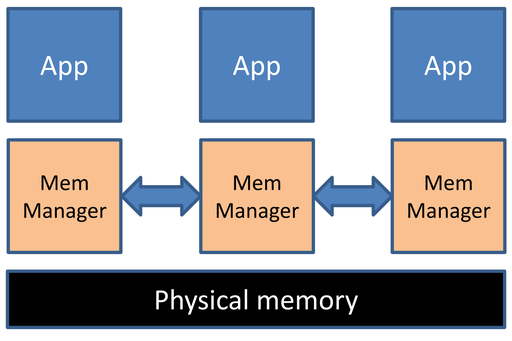
Alternatives for non-cache-coherent systems



- Refactor shared-memory OS
  - Insert explicit cache flush operations
  - Shared-memory with release semantics
  - Probably doesn't scale either
- Multikernel: **Distributed OS services**
  - Disciplined **sharing of resources**
  - Tight **resource coordination**
- Cluster-on-chip: Separate OS instance per core
  - Can re-use legacy OS
  - Have to partition shared resources

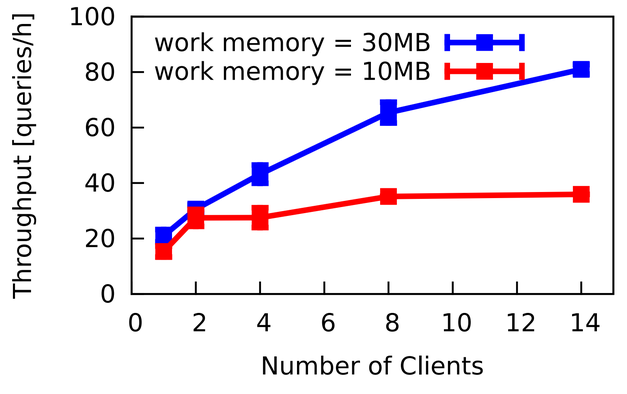
Distributed shared memory manager

- Manage core-local memory
- *Steal* free memory from other cores



Experimental results: Postgres Database Engine running TPC-H

- Multi-tenant scenario:
  - 1 database per core
- Bursty workload:
  - Changing memory requirements
- Cluster-on-chip: Partition limits available memory
- Multikernel: More clients at better performance

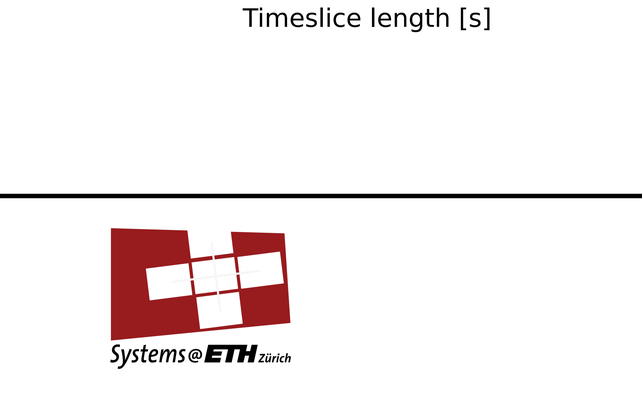


Shared filesystem buffer cache

- Central cache manager
  - Buffer lookup
  - Replacement policy
- Shared filesystem buffers
- Per-core buffer management

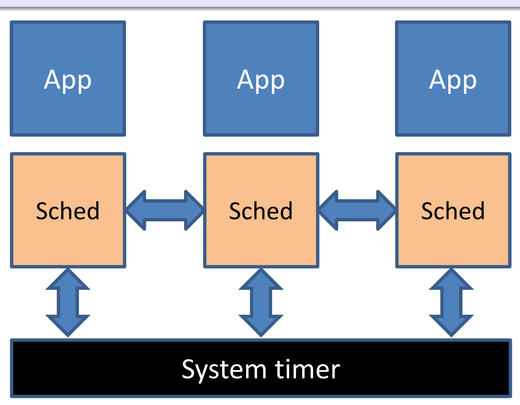
Experimental results: Parallel compile

- Parallel `git` compile
- Replay on shared NFS filesystem
- Cluster-on-chip: More misses to NFS
- Multikernel: Better scalability via shared cache



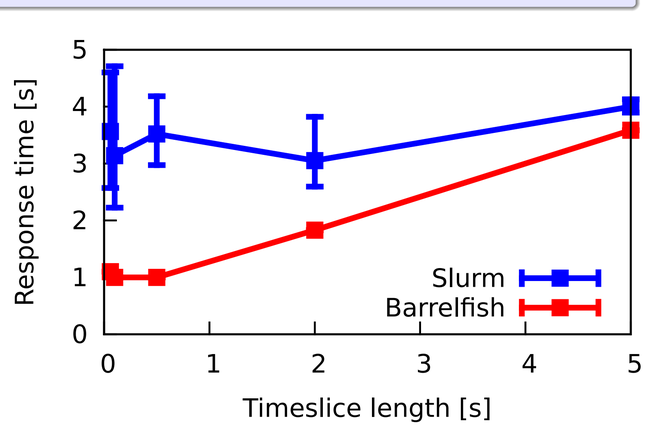
Coordinated CPU scheduling

- Per-core schedulers: Fully scalable
- Deterministic scheduling
- Globally shared system timer
- No dispatch synchronization on context switch



Experimental results: Parallel multi-tasking

- NPB benchmark & CPU stressors
- Request computation every 2 seconds
- Cluster-on-chip: Synchronization via user-level Middleware
- Multikernel: Better responsiveness through tight coordination



ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

inf | Informatik
Computer Science

Systems@ETH Zürich