SCALING AN OPERATING SYSTEM TO MANY CORES USING A SYSTEM CALL LOG

CHINMAY KULKARNI

THE UNIVERSITY OF UTAH

vmware

BESPIN

> A multi-kernel OS designed for **many-core scalability**

> Bespin is **replicated**: machine runs multiple OS replicas

> System calls appended to **log**: Replicas consume log and stay in sync

OS SCALABILITY

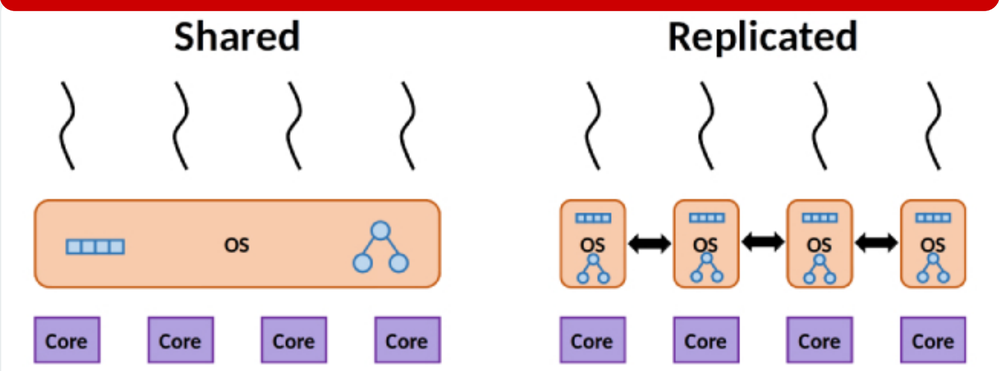
> Two broad approaches explored so far

> **Share** OS on all cores, scale each subsystem. Ex: rcu, mcs etc.

> **Replicate** OS on all cores, message to sync. Ex: barrelfish, fos

Two extremes with trade-offs

RESEARCH QUESTION



> New subsystem?

> Synchronization?

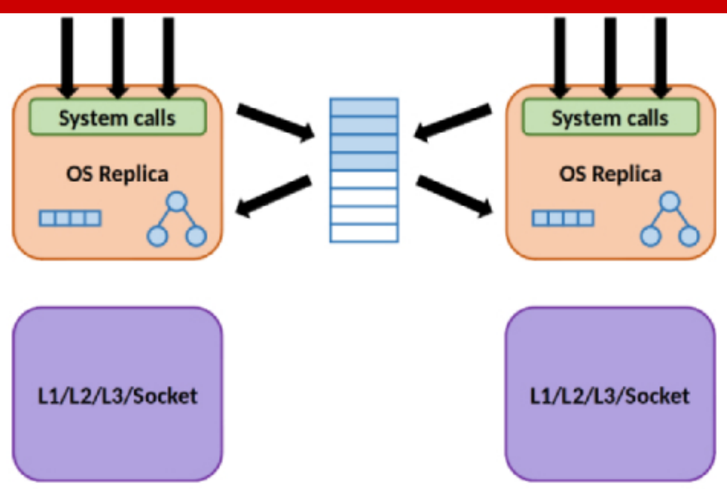
Better designs in-between?

BESPIN's APPROACH

> Inspired by node-replication
  > **Single-threaded** data structure
  > Replica per socket, log shared by replicas
  > Flat combining, Concurrent reads
  > Linearizable

> **Treat the OS as a data structure**
  > Replicate per **L1/L2/L3/Socket**

ARCHITECTURE



Share within, replicate across

IMPLICATIONS

> Trade memory for scalability

> Improve performance too?
Ex: Replicated page tables

> Simplify OS subsystems?
Ex: Journalling file-system

EVALUATION