Review of

Barrelfish

Andrew Baumann et al.

What is the problem?

It is no longer acceptable to tune a general-purpose OS design for a particular hardware model because optimizations become obsolete after a few years when new hardware arrives.

Summary

In this work, the authors employ the ideas of distributed systems to create a new OS model, the multi-kernel, that uses explicit message passing between cores for communication as compared to a shared memory space thus reducing latency in communication. They implement a prototype multi-kernel, Barrelfish, and measure its performance against traditional operating systems.

Key Insights

- As hardware architectures keep evolving and the number of cores keep increasing, optimizations are being implemented for increasing performance of kernels that utilize shared-memory, but there is no common solution for improving performance of multi-kernel systems.
- Pipelining and batching messages encoding remote operations allows for a single core to achieve greater throughput than otherwise.

Strengths

- Using messages to access or update state rapidly becomes more efficient than shared-memory access as the number of cache-lines involved increases.
- Replicating data structures can improve system scalability by reducing the load on the system interconnect, contention for memory, and overhead for synchronization.

Weaknesses

- This paper does not show great advantages over the traditional OS.
- High workloads may result in queuing and slowdown.

Summary of Key Results

- Barrelfish has comparable performance to traditional OSes but greatly improves scalability.
- For IP loopback, Barrelfish uses URPC to connect two user-space IP stacks such that it can achieve the equivalent functionality for point-to-point links.

Open Questions

- What changes need to be made to application layer software?
- Is the scalability increase worth the effort required to port all OS models to a multi-kernel model?