The Multikernel
A new OS architecture for scalable multicore systems

Andrew Baumann

multikernel 一个新的可扩展多核系统的新的操作系统体系结构阅读报告：

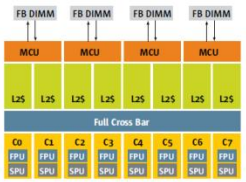
邓志会 2015210926
元东 2015210938

背景介绍

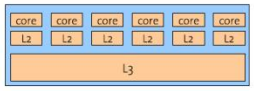
我们应该为未来的多核系统构建一个操作系统
对于多核可以扩展
异构和硬件多样性

系统多样性

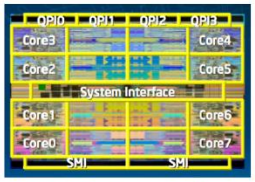
互连的问题



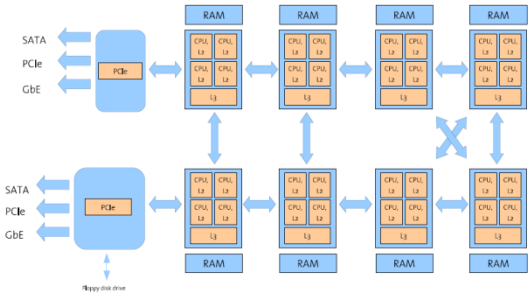
Sun Niagara T2

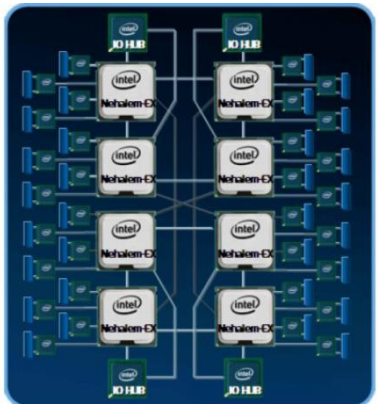


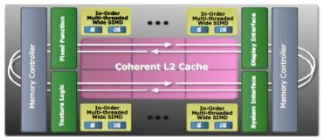
AMD Opteron (Istanbul)

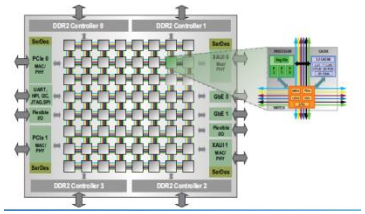


Intel Nehalem (Beckton)









背景介绍

现在开始重新思考默认的操作系统结构

在每个内核共享内存内核

有锁保护数据结构

其他都是一个设备

提议：将操作系统构造一个分布式系统

设计原则：

使得内核之间明确通信

使得操作系统结构硬件中间的

把状态视为复制的

概述

介绍

动机
 硬件多样性

多核模型
 设计原则
 模型

Barrelfish

评测（例子：Unmap）

概述

1.内核之间通信更明确

使用消息进行所有的通信(没有共享状态)

从内核之间通信机制分离系统结构

明确表示通信模型

自然支持异构的核

非一致互连(PCIe)

更好匹配了未来的硬件

通过廉价的外显消息传播

不是缓存一致性

允许分相操作

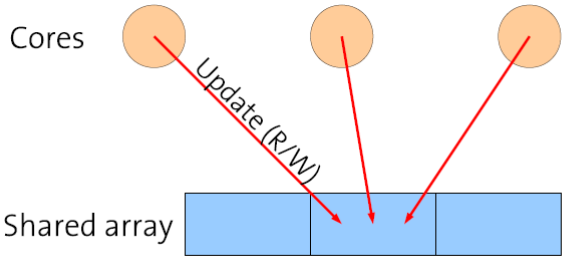
为了并发性分离请求和回复

我们可以推理它

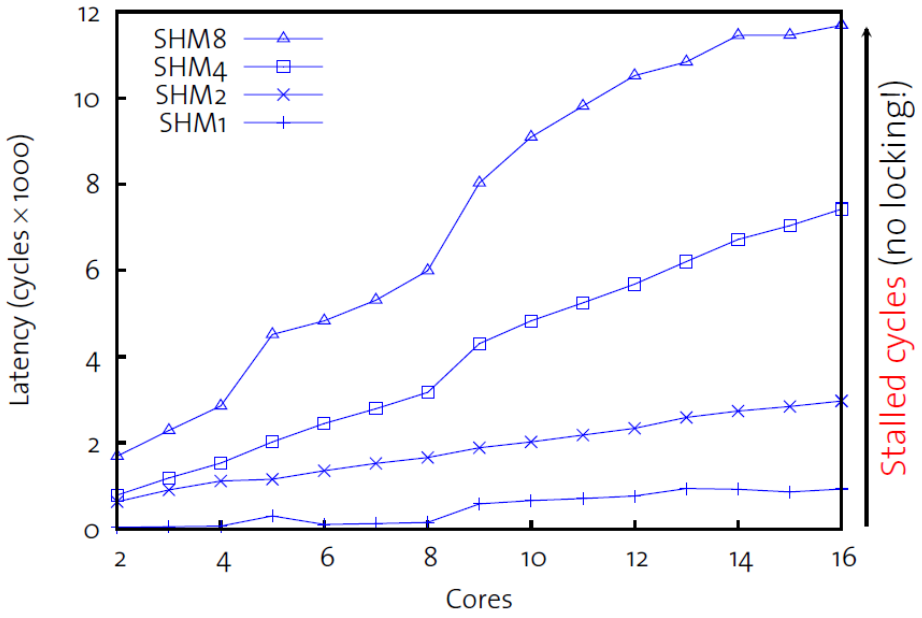
消息传递和共享内存实验

共享内存（数据移动到操作中）：

- 每个内核更新相同的内存区域(没有锁)
- 缓存一致性协议迁移到修改的缓存线
  - 处理器停滞而线路是牵制或者失效
  - 有限的互连往返延迟
  - 性能取决于数据大小(缓存线)和竞争(内核数量)

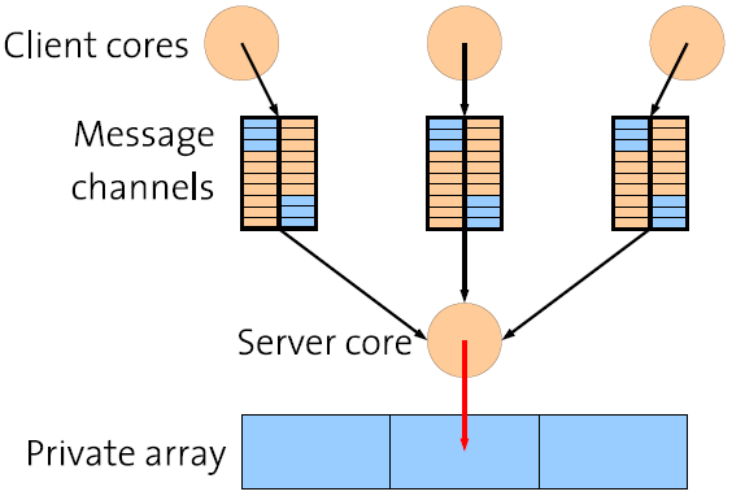


共享内存结果

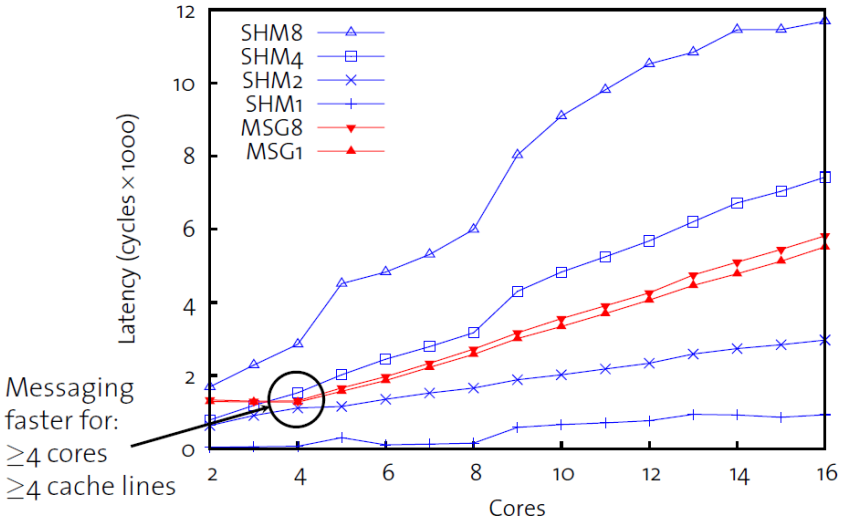


消息传递和共享内存实验

消息传递（操作移动到数据中）：
- 一个单一服务器内核更新内存位置
- 每个客户端内核发送RPCs到服务器
- 在单个缓存线上描述操作和结果
- 在等待一个响应的时候阻塞



消息传递和内存共享的折中



使得操作系统结构硬件中立

从硬件上分离操作系统结构
只有指定硬件部分
- 消息传递(专门高度优化)
- cpu/设备驱动

改变性能特点适应性
晚绑定协议和消息传递实现

将状态视作可复制的

潜在共享状态像本地副本一样访问

调度队列，过程控制块等

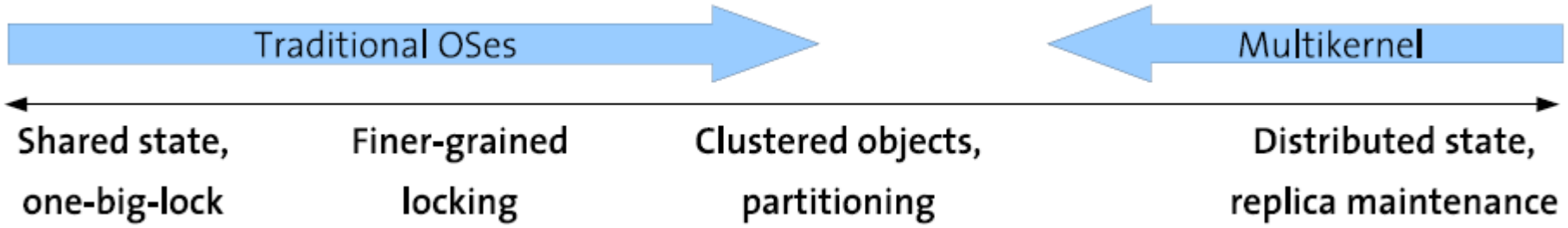
消息传递模型需要

自然支持不共享内存区域

自然支持运行内核集合的改变

HotPlug，电量管理

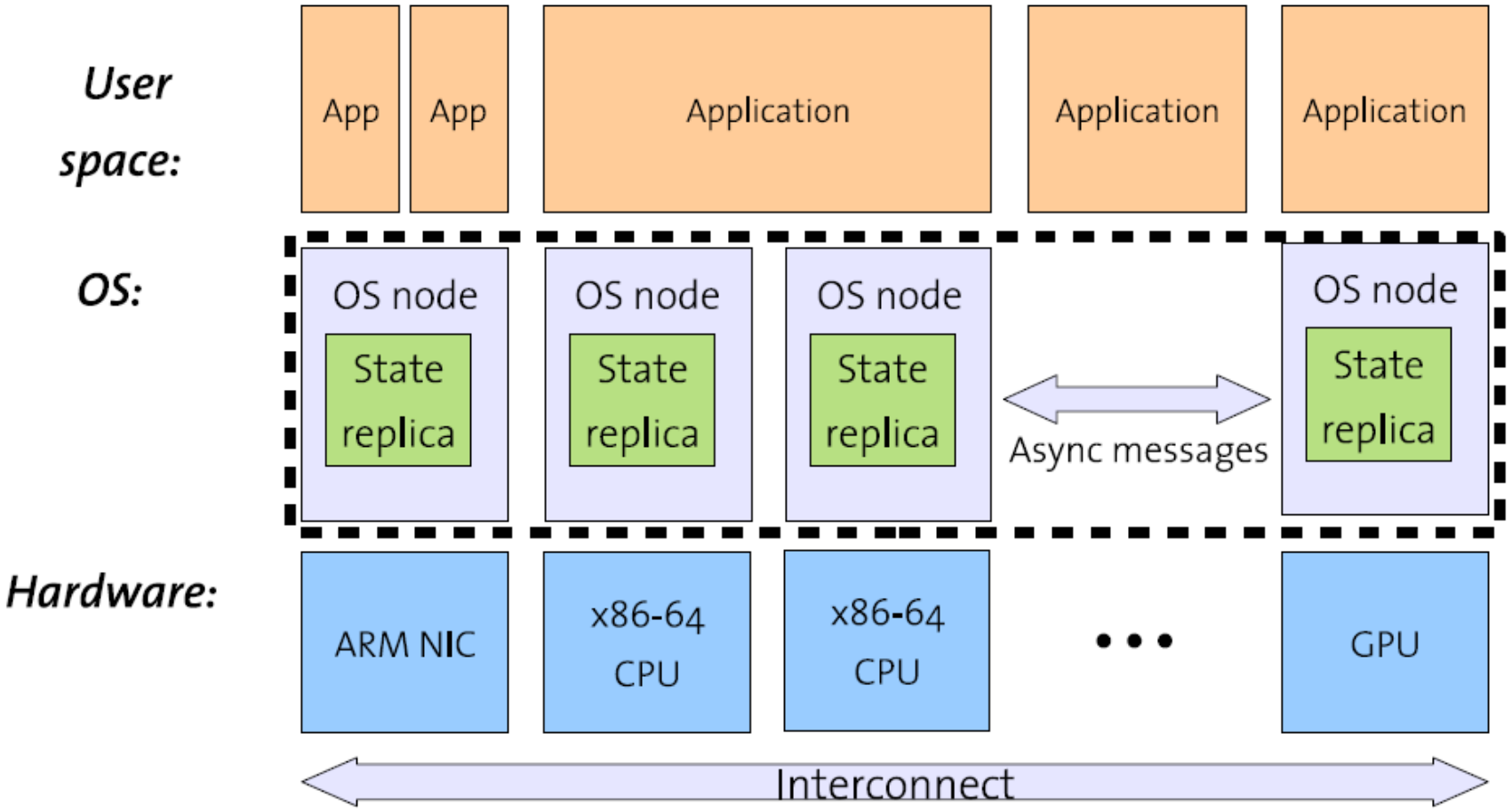
复制和默认共享



在以前的系统中用作优化的复制
- Tornado,K42 聚类对象
- Linux 只读数据，内核文本

在一个多核中，共享一个局部的优化
- 对于紧密耦合的线程或者内核共享复制
- 隐藏，局部
- 在在更快，在运行决定
- 基本的模型保持分相

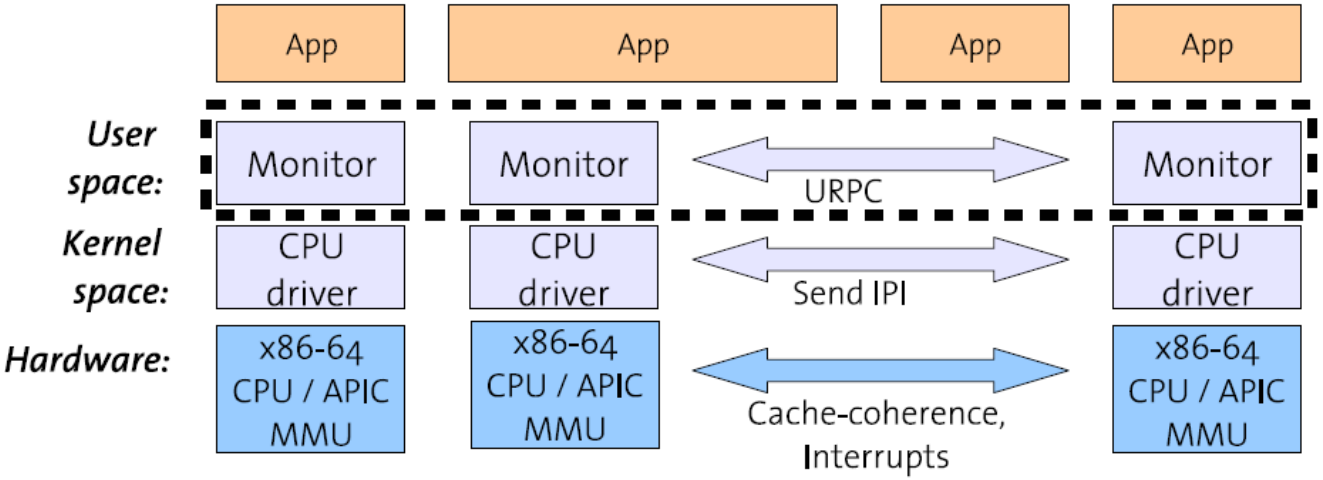
多核模型


User space:
OS:
Hardware:
App
App
Application
Application
Application
OS node
State replica
OS node
State replica
OS node
State replica
Async messages
OS node
State replica
ARM NIC
x86-64 CPU
x86-64 CPU
GPU
Interconnect

Barrelfish

从一开始实现多核
支持x86-64多处理器(ARM)
开源

监视器和CPU驱动



CPU 驱动 序列化处理陷阱和异常
监视器在全局状态促进局部操作
URPC 内核之间的在缓存一致的x86硬件上进行消息传输

评测结果

评测目标

好的基线性能

在现有硬件上与已存在系统可以比较

内核可扩展

对于不同硬件能够适应

利用消息传毒性能的能力

案例研究：Unmap(TLB)

用映射发送消息给每个内核，等待所有确认

Linux/Windows：

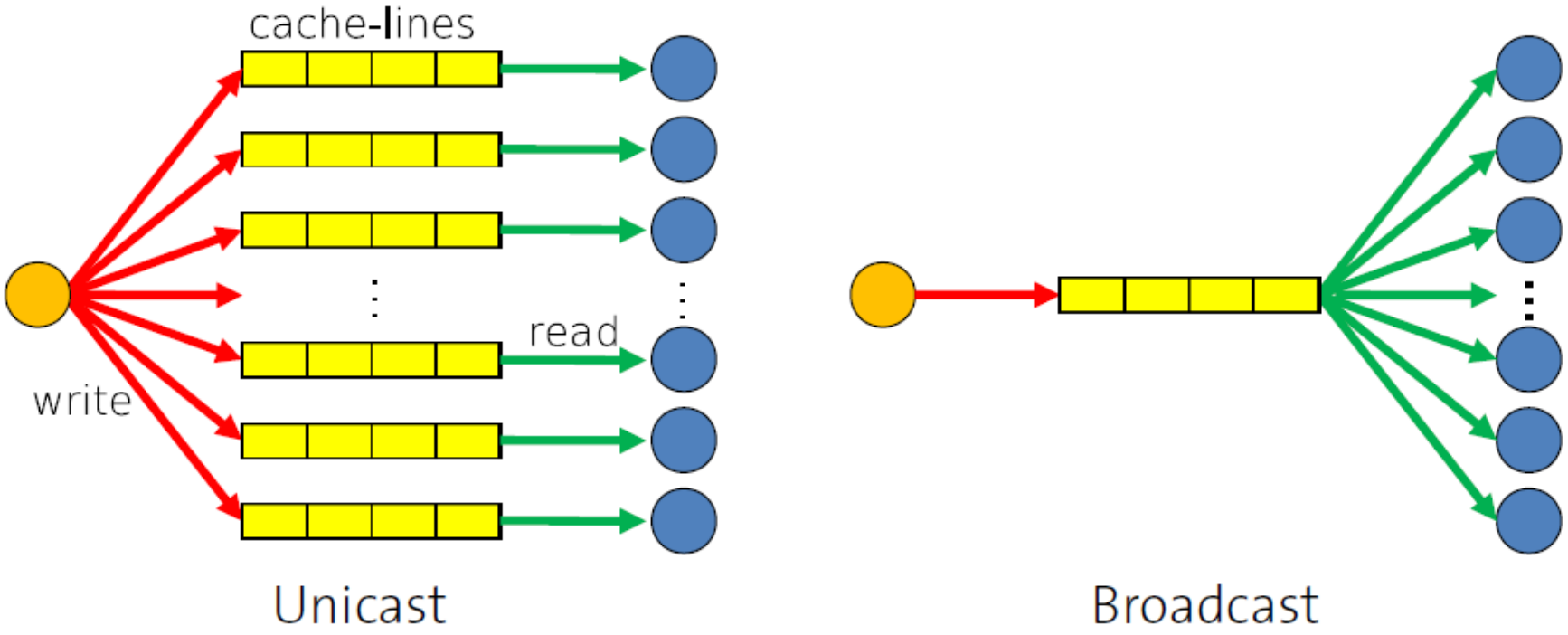
1.内核发送IPIs

2.共享内存计数/事件的锁相环

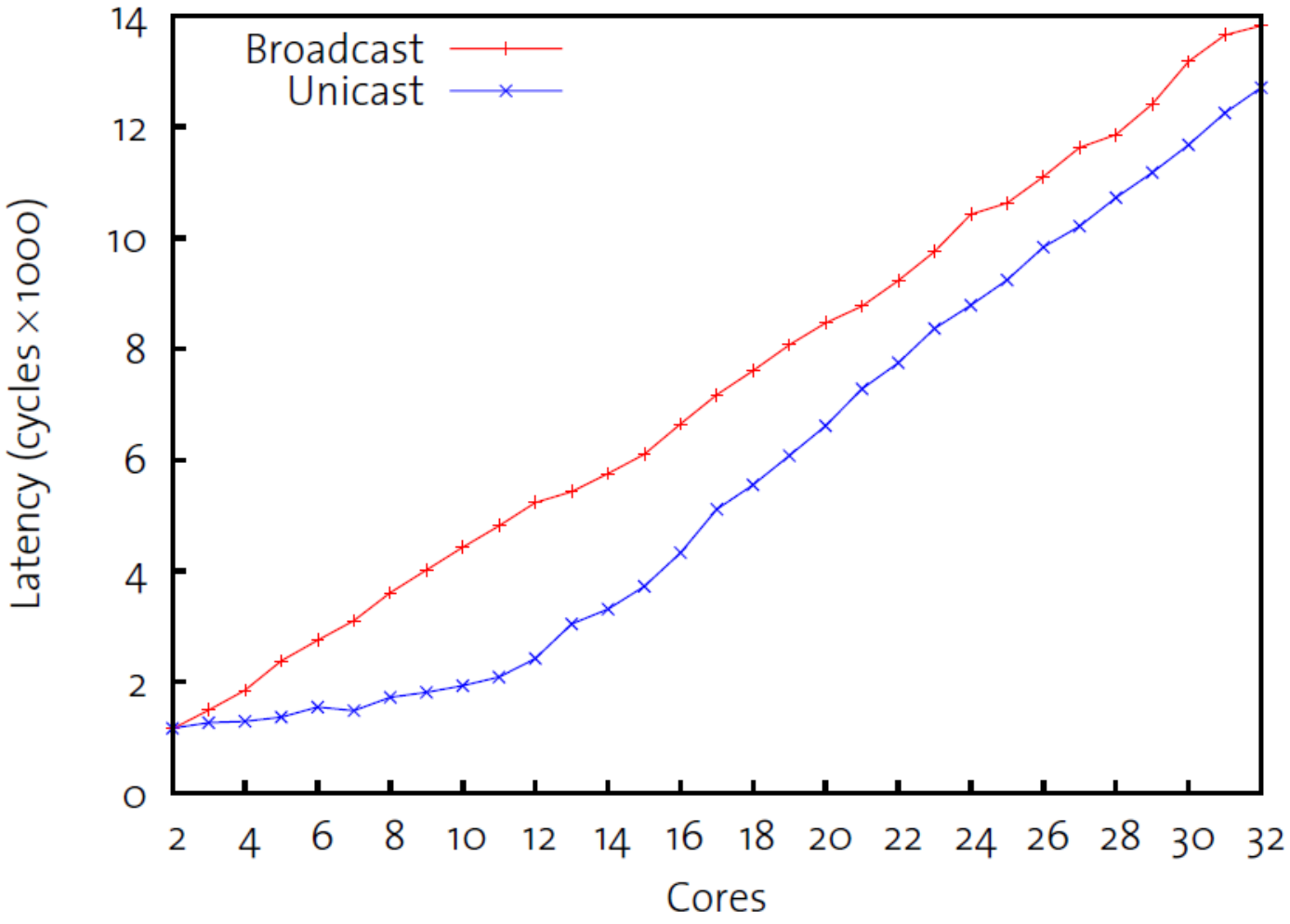
Barrelfish: 用户请求本地监视域；单相提交到远程内核

如何实现通信

Unmap 通信协议

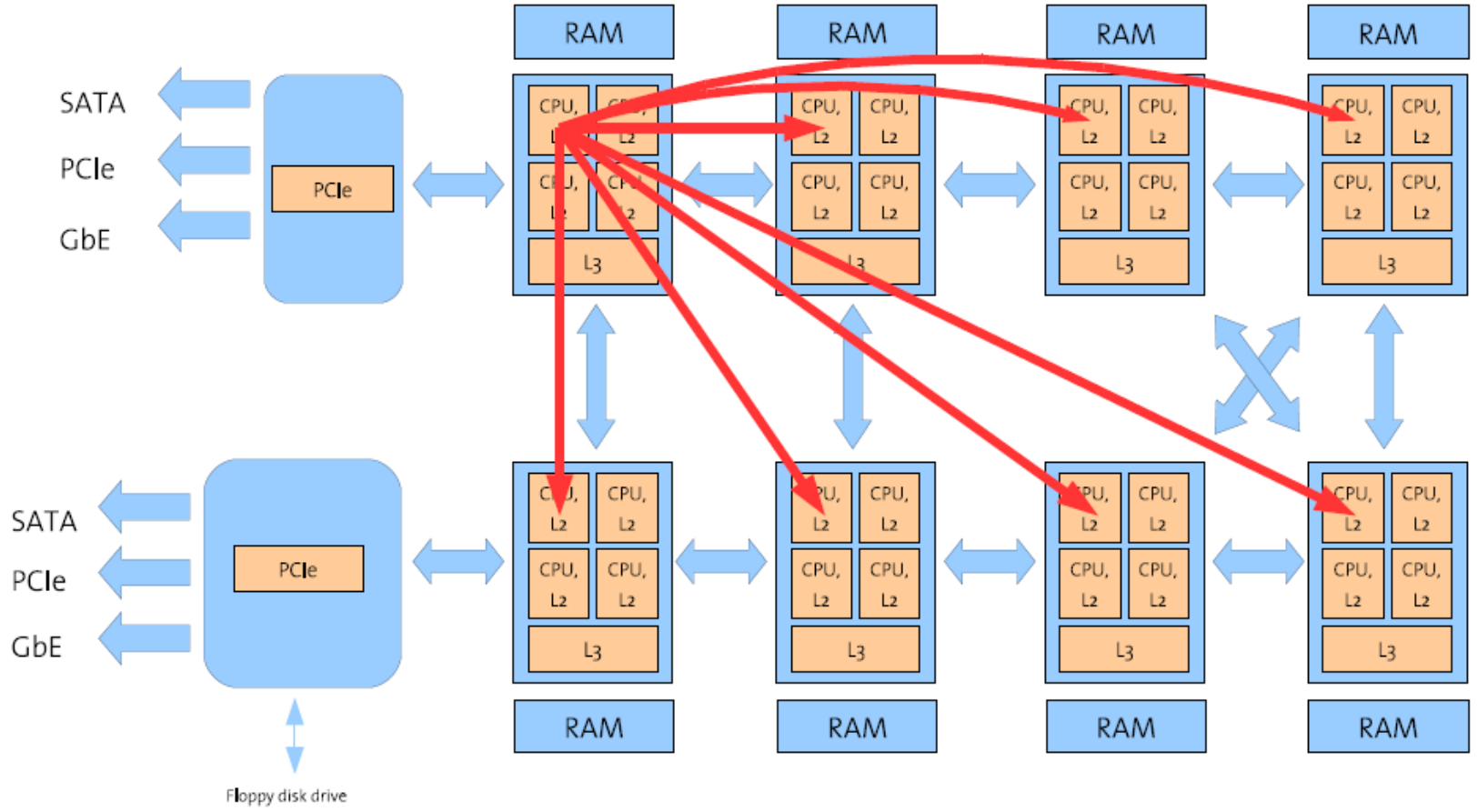


Unmap 通信协议


Broadcast
Unicast
Latency (cycles ×1000)
Cores
0
2
4
6
8
10
12
14
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32

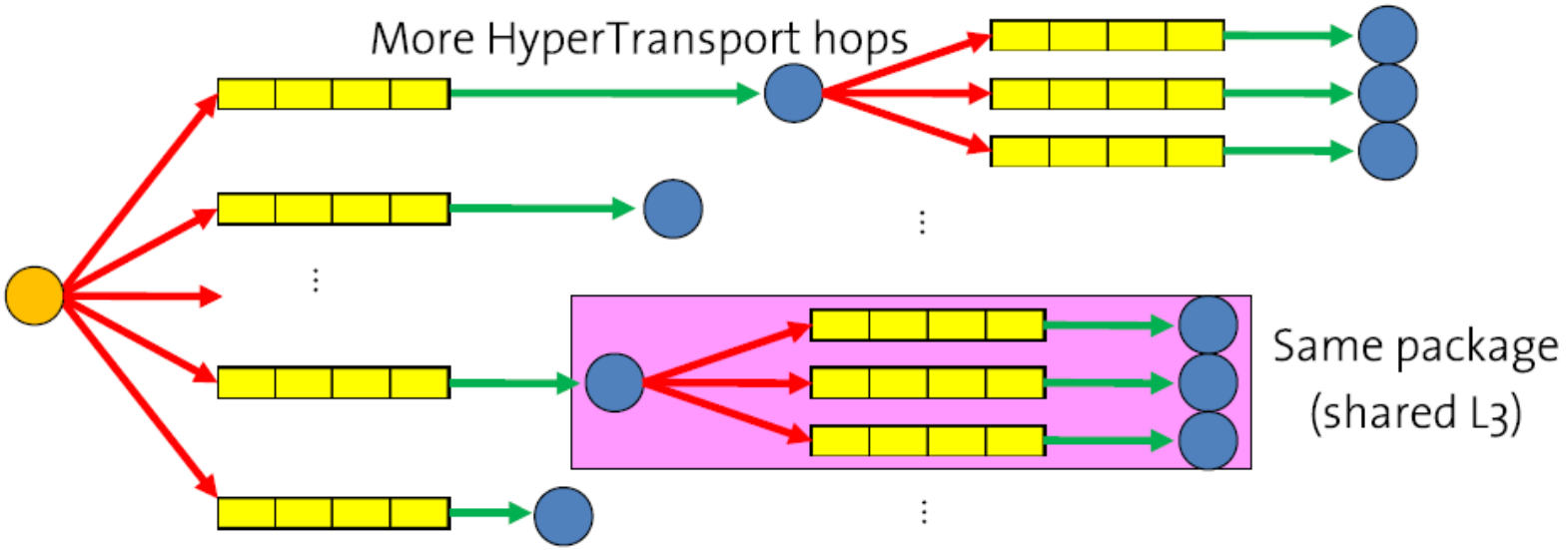
Unmap 通信协议

使用组播原因



Unmap 通信协议

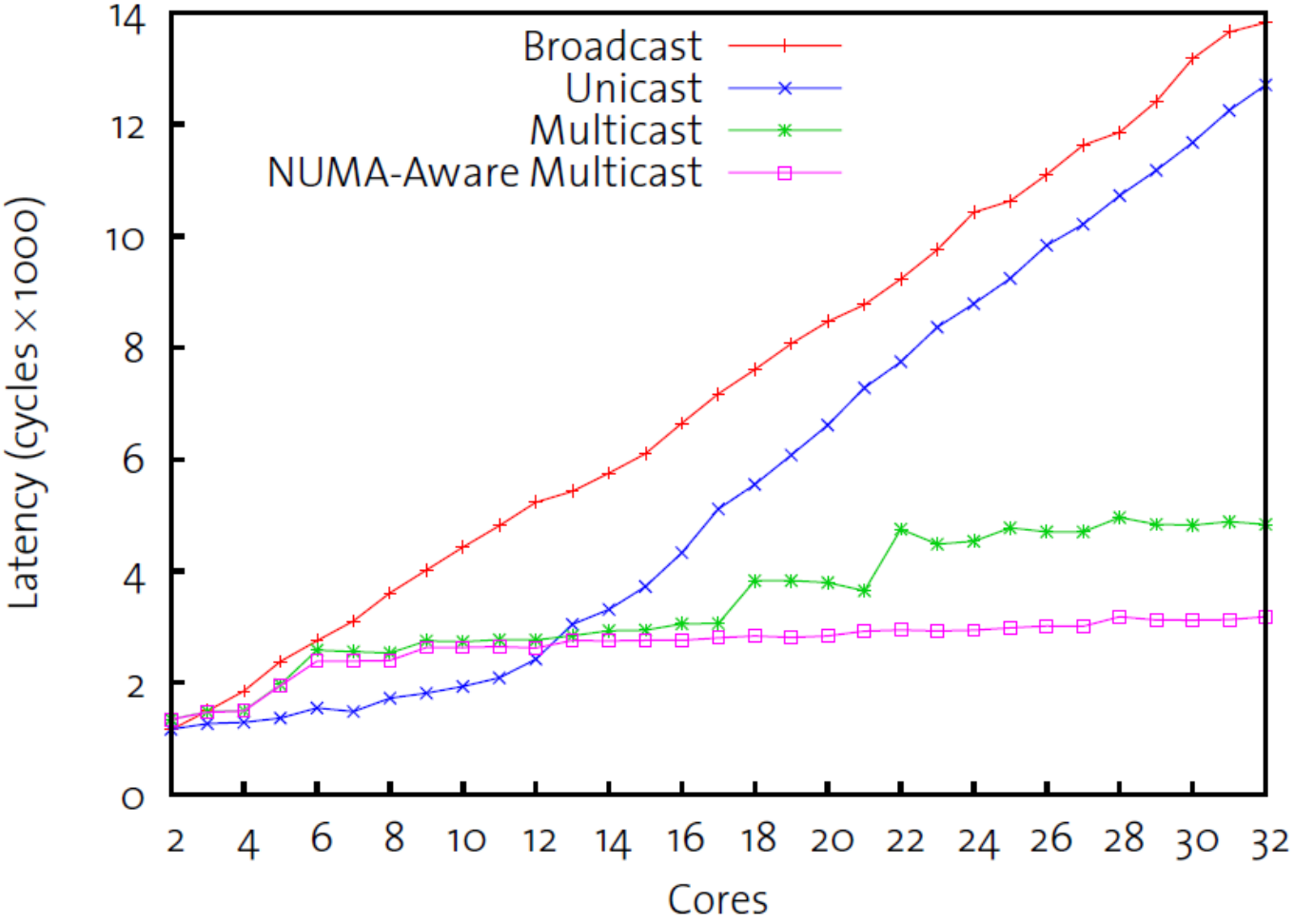
组播通信



- “NUMA-aware” multicast

Unmap 通信协议

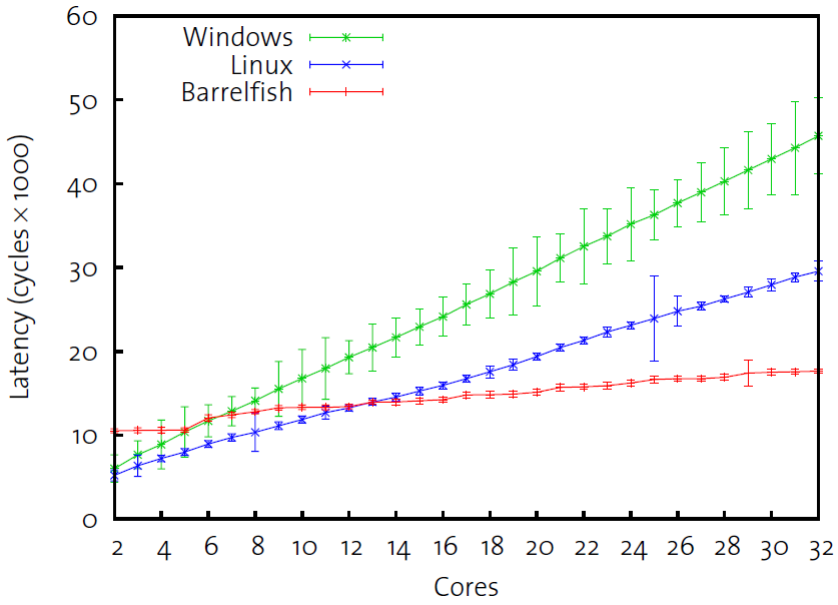
原始消息成本



系统知识库

构建组播树需要硬件知识
　内核映射到套接字
　通信延迟（在线测量）
更通常的，Barrelfish需要一种方式来推理不同的系统资源
使用约束逻辑设计来装配
系统知识库存储了丰富细节的硬件表示，执行在线推理
Prolog 查询用来构建组播路由树

Unmap延迟



总结

现代计算机都是固有的分布式系统
重新思考配对的操作系统

Multikernel：操作系统作为一个分布式系统的模型
1.显式通信，复制状态
2.硬件中立的操作系统结构

Barrelfish：具体的实施
在当前硬件上的合理性能
更好的扩展适应未来的硬件
很有前途的方案

谢谢