Hot Topics in Operating Systems

CS239
Advanced Operating Systems
December 10, 1998
Dr. Richard Guy

Outline

- Mobile computing, proxies, and agents
- Prediction, introspection, and self-organization
- SOSP `97
- OSDI `99

Mobile Computing, Proxies, and Agents

- What are the special challenges of mobile computing?
- What are proxies and agents?
- How can they help mobile computers?
- Research issues

Mobile Computing

- Portable computers are becoming increasingly popular
- Portables computers are inherently mobile
  - They compute while moving
  - They communicate from various places

Systems Issues for Mobile Computers

- Power management
- Handling disconnection
- Handling limited connectivity
- Handling communications while mobile

Power Management Issues

- Batteries have limited lives
- The lifetime of batteries isn't increasing much
- The amount of power you're willing to carry isn't enough
  - For many purposes

How To Save Power

Well, use less

- Shut down things you don't need
- Burn the power elsewhere

Shutting Things Down

What can be profitably shut down?

- Screen
- Disk
- Other I/O devices
- Memory
- Processor

Disk Shutdown Strategies

Disks burn lots of power while spinning

So spin disk down when you don't need it

The problem - spinning a disk down and spinning it back up takes more power

In practice, aggressive choices work best

E.g., spin down with 1 second idle time

Burning the Power Elsewhere

Since portables don't have much power, make someone who does do your work

- Migrate big user tasks
- Run system-related tasks elsewhere

Handling Disconnections

Most portable computers will eventually have to run with zero connectivity

Best if they run well when disconnected

How do we make that happen?

Make it look to the user and applications like it is connected

Making Disconnected Computers Appear Connected

Make sure everything the user needs is local

If it normally was remote to the portable, pull it onto the machine

Caching, replication, hoarding

Handling Limited Connectivity

E.g., modem or wireless links

You have some bandwidth, but a lot less than usual

What should the system do?

- Limitation of transmissions
- Spooling of transmission
- Prioritization

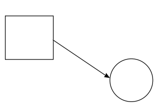
Handling Communications While Mobile

What if someone's sending you a lot of data?

But you're moving fast

How can you make sure all of your data gets delivered?

Communicating With a Mobile Machine

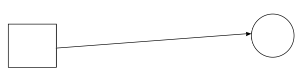


Communicating With a Mobile Machine

CS 239, Fall 1998

Lecture 15
Page 16

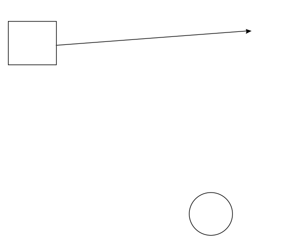
Communicating With a Mobile Machine



CS 239, Fall 1998

Lecture 15
Page 17

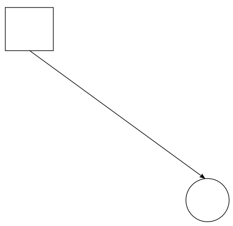
Communicating With a Mobile Machine



CS 239, Fall 1998

Lecture 15
Page 18

Communicating With a Mobile Machine



Methods of Handling Rapid Movement

- Send data before it's needed (buffer)
- Track movement of machine and predict where to send future data
- Send data to more than one place

What Are Proxies and Agents?

Programs working on your behalf

Why do I need those?

More precisely, what's different about them from any other program?

They operate semi-autonomously at remote sites

The Proxy Model of Distributed Computing

The distributed system is a big, bad world

Rather than have the user (or his OS) worry about all the problems,

Assign handling of certain problems to proxies working on his behalf

Examples of Proxies

- Handling limited bandwidth over a network
- Handling changes in network characteristics
- Buffering data for a disconnected computer
- Taking care of security needs

An Example of A Proxy

A proxy that knows how to adapt HTTP transmissions for low bandwidth

Developed at Berkeley

A general-purpose proxy that calls *distillers* to adapt particular types of data

Details of the Berkeley Proxy Solution

All WWW traffic goes through proxy

Proxy is knowledgeable about its client machine and the network

Proxy sits at boundary between good and bad connectivity

Proxy calls proper distiller for each piece of data delivered to client

Berkeley Distillers

Programs that can adapt a piece of data to match the capabilities of the machine and network

E.g., if you have a B/W display and little bandwidth, change color to grayscale

Must calculate what things will cost and what is possible

Difficult Issues in Proxy Research

- Application specific or not?
- Must the application be aware of the proxy?
- May proxies interact with each other?
- Where can proxies live?
- How do proxies get there?

More Proxy Research Issues

- If instantiated dynamically, how do you start up a proxy?
- What do you do when conditions change?
- How does mobile routing match up with proxies?
- In what ways may the proxy alter the data?

Yet More Proxy Research Issues

- What about security?
  - How much do I trust my proxy?
  - How much do I trust someone else's proxy?
  - What if I don't even know the proxy is there?

A Generalized Proxy Approach

Under development here at UCLA

Proxies/agents can live at any node

Agents assume no application-awareness

Several agents at different sites can cooperate to support one data stream

The collection of agents must be able to adapt

Handling Agent Generality

Build an agent framework

Capable of

- Setting up agents
- Establishing strings of agents
- Monitoring and responding to changes
- Accepting new agents

Prediction, Introspection, and Self-Organization

Operating systems "out of the box" are tuned for general purpose work

Since the seller doesn't know what you'll do with it

Most users are not capable of tuning systems

But can the system tune itself?

Related Motivations

- Self-organization for system administration
- Automatic response to changing system conditions
- Handling distributed systems

Some Sample Areas for Predictive Methods

- File system optimization
- Page caching algorithms
- Predictive hoarding
- Prefetching web pages

File System Optimization

The game in file system performance is to minimize physical movement in the disk

By trying to keep related disk blocks together

Previous FS Optimization Strategies

- Spacial locality
  - Keep blocks of same file, files in same directory together on disk
  - E.g., FFS
- Temporal locality
  - Keep blocks written at the same time close together
  - E.g., LFS

A Possibly Better Optimization Method

Neither of these are exactly what you want

What you really want is *referential locality*, and these are mere approximations

Referential locality - keep blocks that are referenced together close together

How To Achieve Referential Locality

Figure out what blocks are referenced together

Re-organize the disk so that they are stored close together

Sounds simple, right?

But . . .

Potential Problems in Obtaining Referential Locality

- Gathering the data
- Properly understanding the data
- Minimizing cost of data gathering
- Trading off costs of disk re-organization vs. minimization of access time

SOSP `97

- Performance and Correctness
- Kernels and OS Structure
- Network and Data Services
- Security; Multiprocessing Support
- Scheduling for Multimedia
- File Systems & I/O; Mobile Systems

Performance and Correctness

- Continuous Profiling: Where Have All the Cycles Gone?
- System Support for Automated Profiling and Optimization
- Eraser: A Dynamic Data Race Detector for Multi-Threaded Programs

Kernels and OS Structure

- The Flux OSKit: A Substrate for Kernel and Language Research
- Application Performance and Flexibility on Exokernel Systems
- The Performance of u-Kernel-Based Systems

Network and Data Services

- Cluster-Based Scalable Network Services
- Free Transactions with Rio Vista
- HAC: Hybrid Adaptive Caching for Distributed Storage Systems

Security

- Extensible Security Architecture for Java
- A Decentralized Model for Information Flow Control

Multiprocessing Support

- Disco: Running Commodity Operating Systems on Scalable Multiprocessors
- Towards Transparent and Efficient Software Distributed Shared Memory
- Cashmere-2L: Software Coherent Shared Memory on a Clustered Remote-Write Network

Scheduling for Multimedia

- The Design, Implementation and Evaluation of SMART: A scheduler for Multimedia Applications
- CPU Reservations and Time Constraints: Efficient, Predictable Scheduling of Independent Activities
- Distributed Schedule Management in the Tiger Video Fileserver

File Systems and I/O

- Frangipani: A Scalable Distributed Filesystem
- Improving the Performance of Log-Structured Filesystems with Adaptive Methods
- Exploiting the Non-Determinism and Asynchrony of Set Iterators to Reduce Aggregate File I/O Latency

Mobile Systems

- Automated Hoarding for Mobile Computers
- Agile Application-Aware Adaptation for Mobility
- Flexible Update Propagation for Weakly Consistent Replication

OSDI `99

- I/O
- Resource Management
- Kernels
- Real-Time
- Distributed Systems
- Virtual Memory
- Filesystems

I/O

- Automatic I/O Hint Generation through Speculative Execution
- IO-Lite: A Unified I/O Buffering and Caching System
- Virtual Log-Based File Systems for a Programmable Disk

Resource Management

- Resource Containers: A New Facility for Resource Mangement in Servers
- Defending Against Denial of Service Attacks in Scout
- Self-Paging in the Nemesis OS

Kernels

- Tornado: Maximizing Locality and Concurrency in a Shared Memory Multiprocessor Operating System
- Interface and Execution Models in the Fluke Kernel
- Fine-Grained Dynamic Instrumentation of Commodity OS Kernels

Real-Time

- ETI Resource Distributor: Guaranteed Resource Allocation and Scheduling in Multimedia Systems
- A Feedback-Driven Proportion Allocator for Real-Rate Scheduling
- A Comparison of Windows NT4.0 & 98 Driver Model Latency Performance

Distributed Systems

- Practical Byzantine Fault Tolerance
- The Coign Automatic Distributed Partitioning System

Virtual Memory

- Tapeworm: High-level Abstractions of Shared Accesses
- MultiView and Millipage--Fine-grain Sharing in Page-based DSMs
- Optimizing the Idle Task and Other MMU Tricks

Filesystems

- Logical vs Physical File System Backup
- The Design of a Multicast-Based Distributed File System
- Integrating Content-based Access Mechanisms with Hierarchical File Systems

Summary

- There's LOTS of fun things to think about in operating systems… both in single-site models and distributed system models
- OS-related folks here at UCLA include Muntz, Popek, Reiher, Guy, Bagrodia, Gerla, Zhang, Potkonjak