Thinking About Workloads
CS 239
Experimental Methodologies for System Software
Peter Reiher
April 24, 2007

Examples in Choosing Workloads

- Ficus
- Time Warp
- DefCOM
- Conquest

Ficus

- A replicated file system
- Keeps multiple copies of files on different machines
  - Including possibly disconnected portable machines
  - Uses optimistic synchronization
- Benefits are availability and local performance
- Issues are overall performance and effects of conflicts

Basic Idea Behind Ficus

- Store replicas where users might need them
- Allow updates at any replica
- Propagate updates to other replicas
- Can lead to consistency problems
  - Detect them
  - Correct them (automatically, when possible)

A Little More About Ficus

- Update propagation
- Reconciliation
- Conflicts

Update Propagation

- Any update generates a "best-effort" propagation message
  - Generated on every write system call
  - Broadcast to all known replicas
  - Notification of change, not contents
- Receiving site can ignore or can request latest version of file from generating site
  - Only when no conflict

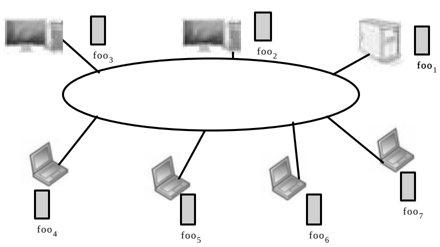
Reconciliation

- *Reconciliation* process ensures updates always get out eventually
- Runs periodically
- Operates between pair of replicas
  – Transfers data in one direction only
- Complex distributed algorithm
  – Proven to terminate correctly
- Performance might depend on order and pattern of reconciliations among replicas

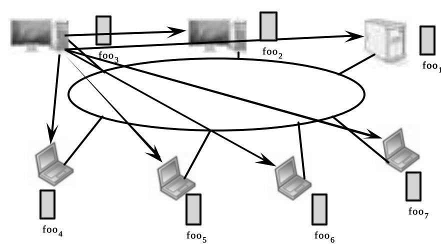
Conflicts

- Ficus allows updates to be applied at any replica
  – Including disconnected replicas
- What if two disconnected replicas each accept an update?
  – With no reconciliation or propagation between them
- The file experiences a conflict

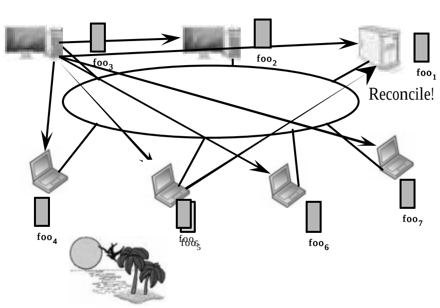
A Ficus Example



An Update Occurs

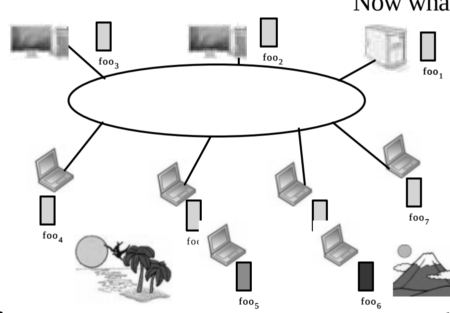


A Reconciliation Case



A Conflict Case

Now what?



Performance Issues for Ficus

- Is it really cheaper than remote access?
- What are the performance costs of maintaining multiple replicas?
- What are the costs of updates?
- How often do conflicting updates occur?
- How often is stale data read from a replica that hasn't gotten the latest update?
- Is there an availability benefit?

Designing a Workload For Ficus

- What kind of workload do we want?
- Where will we get it from?
- How will we use it to evaluate the important issues in Ficus?

Time Warp

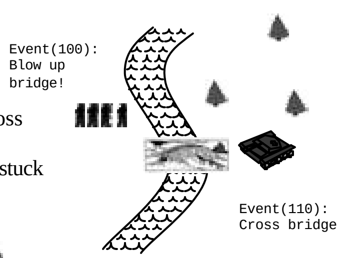
- Engine for running discrete event simulations in parallel
- Using "interesting" synchronization mechanisms
- Goal is essentially to run things faster
  - Than sequentially
  - Than competing parallel methods

Discrete Event Simulations

- Simulate a system by simulating individual events that comprise it
- Events scheduled/communicate via messages
- Parallelize by running multiple events simultaneously
- Key constraint is must get same results as if all events run sequentially
- Issues of proper event ordering vital

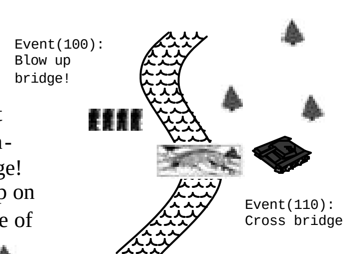
Illustrating the Problem

Attempt to cross bridge fails
So the tank's stuck on the far side



Illustrating the Problem, Con't

The tank just crossed a non-existent bridge!
And it ends up on the wrong side of the river



Basic Idea Behind Time Warp

- Be optimistic
- Run as many events in parallel as you can
  - Which could mean you run some out of order
- Detect out-of-order events, roll them back, and rerun them properly
  - Also rolling back all their side effects
  - Like scheduling other events

Performance Questions for Time Warp

- Can it speed up simulations? How much?
- How much benefit do you get from adding more hardware?
- Which internal optimizations are worthwhile?
- Can it run simulations faster than conservative methods?
- How do optimistic artifacts (like rollbacks) affect performance?

How Do We Design a Workload For Time Warp?

- What kind of workload do we want?
- Where will we get it from?
- How will we use it to evaluate the important issues in Time Warp?

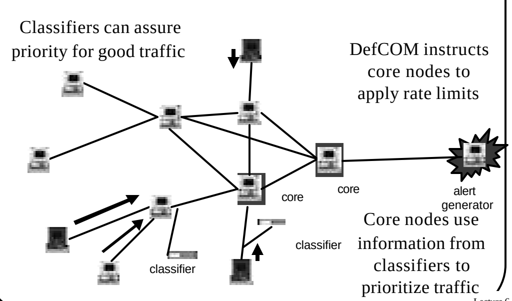
DefCOM

- A defensive system to counter distributed denial of service (DDoS) attacks
- Especially attacks based on high volumes of garbage traffic
  - Originating from many sources

DefCOM Defense Approach

- Addresses the core problem:
  - Too much traffic coming in, so get rid of some of it
  - A common idea in DDoS defense
- Vital to separate the sheep from the goats
- Mark traffic at network entrance points
- Apply rate limits in core
  - Using marks to filter preferentially
- Alerts of DDoS attacks mostly generated near targets

DefCOM



Performance Questions for DefCOM

- How well does DefCOM defend against attacks?
- Does DefCOM damage performance of normal traffic?
- Can all DefCOM components run fast enough for realistic cases?
- How much does partial deployment pattern matter?

How Do We Design a Workload For DefCOM?

- What kind of workload do we want?
- Where will we get it from?
- How will we use it to evaluate the important issues in DefCOM?

Conquest

- A file system meant to improve performance
- Key observation is that disks suck
  - Always have, but sucking harder every year
- Vast amounts of OS effort spent in hiding how badly disks suck

A Solution: Use Persistent RAM

- RAM that saves its state when power goes off
- Speed similar to regular RAM
- Battery-backed DRAM available today
- Flash RAM also common
  - But some bad characteristics
- Other forms of persistent RAM under development

Basic Idea of Conquest

- Use a few gigabytes of persistent RAM
- Store many files permanently in persistent RAM
  - Also store all metadata there
- Use disk only for big files
  - Mostly accessed sequentially
  - Which is OK for disks

Performance Questions for Conquest

- How much faster than pure disk?
- Can it perform better than just persistent caching?
- What about performance of big files?

How Do We Design a Workload For Conquest?

- What kind of workload do we want?
- Where will we get it from?
- How will we use it to evaluate the important issues in Conquest?