My Ongoing Research

CS 256/456

Dept. of Computer Science, University of Rochester

Non-intrusive Soft Error Monitoring

- Joint work with Xin Li and Michael Huang (ECE)
- Slides are shamelessly stolen from Xin
- Soft error – AKA transient error
  - Errors not due to defect of HW/SW
  - Usually caused by particles hitting the transistors: cosmic rays and radioactive materials

Past Results

- Realized in 60s or 70s
  - Atomic bomb tests
  - Spacecraft anomalies
- An example of results on error rate
  - [O'Gorman et al. 1996] – "Terrestrial Cosmic Rays and Soft Errors"
    - 864 4KB modules 4671 hours
    - result: about 1 error every 19 years on each memory module
- Impact on software systems
  - [Messer et al. 2004]; [Lu et al. 2004]
  - Only 2% of the soft errors will be manifested as problems at application level

Our Research

- Goal: Monitor memory soft errors on production computer systems (non-intrusiveness)
  - No tampering with the system (OS in particular)
  - Requiring no privilege to install
  - Minimal performance impact on existing applications
- Approach: Recruit memory not used by other applications and monitor any changes (completely at user-level)
  - Nominal free memory vs. practical free memory
  - Take away memory gradually when system performance is not affected
  - Give back quickly (e.g., exponential back-off) when the system performance is affected
- Problem: How to tell the system performance is affected?
  - Checking page fault rates
  - Checking CPU idleness

Our Results (so far)

- Running on 200 geographically distributed machines for two months
  - No error has been found!
  - Reason: busy machines and ECC
- Accelerated tests with heat gun (for cooking cakes)
- Test 1 – ECC enabled
  - Bluesmoke:
    - Single-bit errors: 1200 Multi-bit errors: 5
  - Our program:
    - Single-bit errors: 0 Multi-bit errors: 9
- Test 2 – ECC disabled
  - Our program
    - Single-bit errors: 200

Comprehensive Anomaly Characterization and Performance Debugging

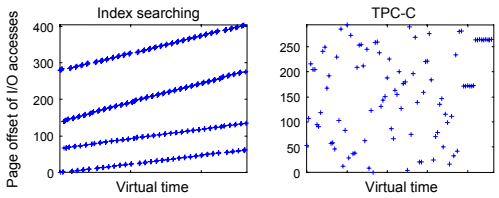
- Joint work with Chuanpeng Li, Christopher Stewart, Ming Zhong, Tom O'Neil
- Implementations of complex systems contain "problems"
  - over-simplification, mishandling of special cases, … …
  - these problems degrade the system performance; make system behavior unpredictable
- Complex systems (like the OS)
  - many system features and configuration settings
  - dynamic workload behaviors
  - problems manifest under special conditions
- Identifying those performance problems under wide ranges of runtime conditions are desirable but challenging

Bird's Eye View of Our Work

- Construct models that report expected system behavior
  - "simple": modeling system components following their design algorithms
  - "comprehensive": considering wide ranges of system configuration and workload conditions
- Anomaly characterization
  - Discover anomalies (discrepancies between model expectation and actual behavior)
  - Characterize them and attribute them to possible causes
- Final usage
  - debugging
  - avoiding anomaly-inducing conditions

Disk I/O-Bound Online Servers

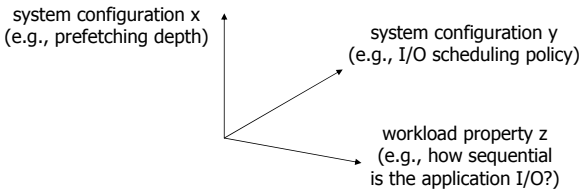
- Server processing access large disk-resident data
- Examples:
  - Web servers serving large Web data
  - index searching
  - database-driven server systems



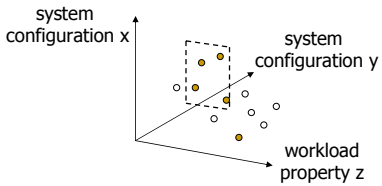
- Relevant OS feature:
  - I/O prefetching, I/O scheduling, file system, memory caching

Anomaly Sampling

- We choose a set of system configurations and workload properties for anomaly sampling
- Sample parameters are chosen from a parameter space



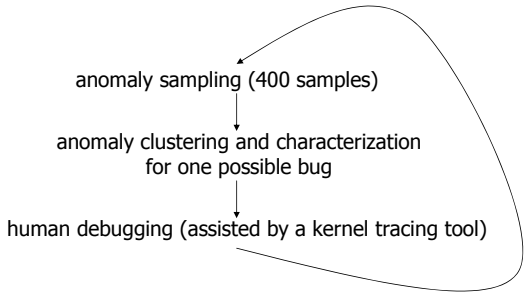
Anomaly Clustering



- Anomalous settings may be due to multiple causes (bugs)
  - hard to make observation out of all anomalous settings
  - desirable to cluster anomalous settings into groups likely attributed to individual causes
    - we perform hyper-rectangle clustering
- Produce succinct characterization desirable for human understanding and analysis

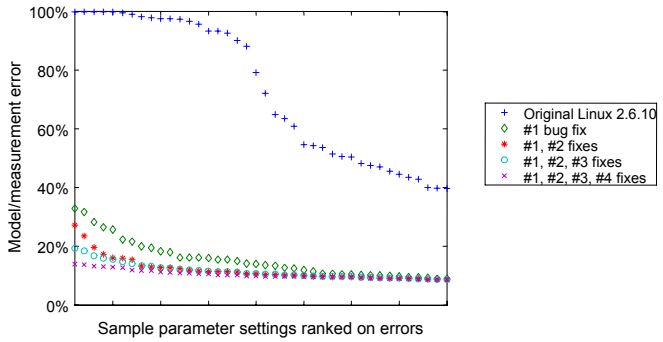
Experimental Setup

- Linux 2.6.10 kernel



Result – Top Anomaly Errors

Error defined as: $1 - \frac{\text{Measured throughput}}{\text{Expected throughput}}$



Result – Anomaly #1

Workload property	
concurrency:	128 and above
Stream length:	256KB and above
System configuration	
Prefetching:	enabled

- The cause
  - when the disk queue is "congested", prefetching is cancelled
  - however, prefetching sometimes include synchronously requested data, which is resubmitted as single-page "makeup" I/O
- Solutions
  - do not cancel prefetching that includes synchronously requested data
  - or block reads when the disk queue is "congested"

Result – Anomaly #2, #3, #4

- Anomaly #2
  - concerning the anticipatory I/O scheduler
  - uses average seek distance of past requests to estimate seek time
- Anomaly #3
  - concerning the elevator I/O scheduler
  - always search from block address 0 for next request after "reset"
- Anomaly #4
  - concerning the anticipatory I/O scheduler
  - a large I/O operation is often split into small disk requests, anticipation timer is started after the first disk request returns

Result – Overall Predictability