Outline

- Control-theoretic Framework
- Service delay control on Web servers
- On-line data migration in storage servers
- ControlWare: adaptive QoS control middleware
Online Data Migration in Storage Systems

- Enterprise storage servers need to move data
  - System expansion
  - Application changes
- **Always-on**: e-business, global data centers
  - **Online** data migration
State of Practice

E-mail server; DB…

Need to bound impact on applications!

Slow I/O’s!!!

SAN

New device

Script

Migration plan

Submover HP-UX LVM

Storage system

data migration

storage devices
The Problem

- Execute a given migration plan on-line
- Challenges
  - Keep data consistent
  - Bound impact on application performance
  - Complete migration quickly
Adaptive solution

- Feedback control loop: adapts migration speed based on application I/O latency
  - Enforce latency contract: Bounded average I/O latency
  - Complete migration in shortest time allowed by contract

- Standard control-theoretic design
  - Systematic methodology
  - Robust, analytically proven performance

- Handle different workloads and devices
Aqueduct

E-mail server; DB...

SAN I/Os

Storage system

data migration

storage devices

Aqueduct migration executor

Monitor

{L_i(k)}

Controller

R_m(k)

Actuator

Submover

HP-UX LVM

Application Latency Contract

Migration plan
Monitor

- Measure applications’ average I/O latency of each store in the last sampling window
  - Current implementation: trace replayer directly monitors I/O latencies
  - Can interface with performance monitoring tools (HP Openview)
**Actuator**

- Fine-grained control of migration speed using HP-UX LVM
  - Divide store into small (32 MB) substores (LVs)
  - **Submover** moves substore using LVM silvering

- **Actuator** enforces a submove rate by sleeping

```plaintext
<table>
<thead>
<tr>
<th>submv</th>
<th>sleep</th>
<th>submv</th>
<th>sleep</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sampling Window</td>
<td>Sampling Window</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

- 1 submv/sw
- 2 submv/sw
Controller

- Compute error for each store $i$
  \[ E_i(k) = P \times LC_i - L_i(k) \]
  $0 < P < 1$: safety margin, related to burstiness
  $k$: represents the $k^{th}$ sampling window

- Compute worst error
  \[ E_{\text{min}}(k) = \min\{E_i(k)\} \]

- Integral controller computes new submove rate:
  \[ R_m(k) = R_m(k-1) + K \times E_{\text{min}}(k) \]
  Control gain $K$: aggressiveness of rate change
Tuning controller parameters

Approximate linear model

\[ VL(k+1) - VL(k) = G(R_m(k) - R_m(k-1)) \]

System profiling: Estimate \( G \)

Construct transfer function

Control Analysis
Compute \( K \)

Victim latency \( VL(k) \): highest average latency among all stores in the \( k^{th} \) sampling window

Process gain \( G \): impact of submove rate on victim latency.

- Stability
- Tracking: \( VL(k) = P*LC \) in steady state
- Settling time
## Experimental setup

<table>
<thead>
<tr>
<th>Aqueduct</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>HP-UX 11 &amp; LVM</td>
<td></td>
</tr>
<tr>
<td>HP 9000-N4000 Server</td>
<td>8 440MHz processors</td>
</tr>
</tbody>
</table>

- **Openmail I/O Trace**
- **Fibre Channel**
- **FC-60 disk array**
  (1.05 TB, 5 RAID5 Logical Units)
  - $LU_0$:
    - emails
    - metadata
  - $LU_{\text{new}}$:
    - emails
    - metadata

- **Enterprise-scale storage server**
Experiments

- Baselines: no sleeping between (sub)moves
  - Whole-store: move one store at a time
  - Sub-store: move one substore at a time

- Constant: steady Poisson streams
  - Replace Logical Unit; migrate three 640-MB stores.

- Openmail: trace of an enterprise e-mail server running HP Openmail
  - Add Logical Unit; migrate a 1854 MB store and a 96 MB store
Measure G \rightarrow Tune K

Process gain G: the slope of the curves
Control gain K

Constant: $K = 1.09$
Openmail: $K = 0.36$

\[ y = 1.12x + 7.55 \quad \text{R}^2 = 0.99 \]
\[ y = 1.41x + 5.80 \quad \text{R}^2 = 0.98 \]
Openmail: victim latency

Average Victim Latency (ms)

- Aqueduct
- Sub-store
- Whole-store

LC
0.8*LC
Openmail: latency

Aqueduct uniformly better than baselines, but …
Openmail: latency & submove rate

- Load highest on new LU towards end of migration
- By design, submove rate must be 1 or higher ➔ controller is working correctly
Openmail: average latency

![Average latency graph]

- **Graph Legend:**
  - Aqueduct
  - Sub-store
  - Whole-store

- **Labels:**
  - X-axis: big0, big1, tiny0, tiny1
  - Y-axis: Average Latency (ms)

- **Annotations:**
  - LC
Openmail: latency CDF

CDF

request latency (us)

Whole-store
Aqueduct
Sub-store
No Migration

91%
76%
Related work

- Simpler versions of the problem
  - Take (parts of) system offline
  - Migrate data in “quiet periods”
- Silvering in Logical Volume Manager [HP-UX LVM, VxVM]: maintain data consistency, no QoS guarantees
- Proportional I/O scheduling: hard to handle unpredictability
- MS Manners: no guarantee to important tasks
- Control-theory-based: distributed visual tracking, Web servers, e-mail server, database, real-time processor scheduling ...
Summary

- Migration must be executed adaptively
- Aqueduct is neither overly aggressive
  - Average I/O latency reduced by 76%
  - Contract violation ratio reduced by 78%
- nor overly conservative
  - Average victim latency 15% lower than latency contract

- Future
  - More detailed sensitivity analysis
  - Self-tuning controller
  - Multi-dimensional QoS contracts
References


Outline

- Control-theoretic Framework
- Service delay control on Web servers
- On-line data migration in storage servers
- ControlWare: adaptive QoS control middleware
Adaptive QoS Control Framework

QoS Guarantee

QoS Mapping

Control Loop Architecture

Controllers

System Identification

Dynamic Model

Controller Design

Dynamic Response Specs

guarantee
ControlWare
Isolate programmers from control-theoretic concerns

QoS contract

QoS Mapper
Control Loop Composition
System ID
Controller Design
Software QoS Control Loops

ControlWare Library
Controllers
Monitors
Actuators

SoftBus
ControlWare: Reference

- Case studies on Squid Web cache and Apache
Control-theoretic QoS Framework

- Map QoS guarantees to feedback control loops
- Establish difference equation models for computing systems via system identification
- Build practical QoS control systems
  - Apache Web server.
  - Enterprise storage server.
  - Avionics image transmission.
- Develop middleware for deploying QoS control
  - FCS/nORB, FC-ORB: Distributed real-time embedded systems.
  - ControlWare: Internet servers.