Engineering Performance Using Control Theory

Theory and Practice

Joe Hellerstein
Google, Inc.
Seattle, Washington USA

April 13, 2010
Resource management of computing systems is complicated and getting more so

- Many interconnected resources
- Complex workloads
- Many sensors (metrics)
- Many effectors (e.g., priority, thread pool size, memory allocations, GC frequency)

Many other engineering disciplines have design problems with complicated non-linear and stochastic dynamics

- Aeronautical engineering: Maintain stable elevations
- Electrical engineering: Design low noise circuits
- Mechanical engineering: Control tools in manufacturing lines
- ...

All use control theory
Agenda

Theory
- Elements of a control system
- Control objectives
- Mathematical details (short)
- Performance engineering methodology

Practice
- DB2 utilities throttling
- DB2 automated memory tuning

Conclusions and what you can read to learn more
Theory
Example: Control of the IBM Lotus Domino Server

Architecture

Admin → Controller → MaxUsers → RPCs → Server

Desired RIS (RPCs in System) → Actual RIS

Block Diagram

Desired RIS → Controller → Server → Actual RIS

$r(k)$ + $e(k)$ → $u(k)$ → $y(k)$
Elements of a Control System

Components
- Target system: what is controlled - has Sensors (metrics) & Effectors (knobs)
- Controller: exercises control
- Transducer: translates measured outputs

Data
- Reference input: objective
- Control error: reference input minus measured output
- Control input: manipulates effectors to impact sensors
- Disturbance input: other factors that affect the target system
- Measured output: Sensor values
- Transduced output: result of manipulation
Closed Loop vs. Open Loop

**Closed Loop System**
- Adapts to change
- Simple system model
- Stable
- Fast settling
- Closed Loop Controller
- Server
- Sensor
- Target System

**Open Loop System**
- Reference RIS
- MaxUsers
- Open Loop Controller
- Server
- Sensor
- Target System
- Measured RIS
Types of Control Systems

Regulatory Control
- Manage to a reference value
- Ex: Service differentiation, resource management, constrained optimization

Disturbance Rejection
- Eliminate effect of a disturbance
- Ex: Regulating background work, controlling workload variations

Optimization
- Achieve the “best” value of outputs
- Ex: Minimize appl response times
The SASO Properties of Control Systems

Stability - Accuracy - Short Settling - Small Overshoot

Unstable System

\[ y_{ss}M_P \]

\[ e_{ss} \]

\[ r_{ss} \]
System Identification

MaxUsers $\rightarrow$ Server $\rightarrow$ Actual RIS $y(k)$

$u(k)$

Model of System Dynamics

$y(k) = a_1 y(k - 1) + b_1 u(k - 1)$

Transfer Function

$S(z) = \frac{b}{z - a}$

$a_1 = 0.913$

$b_1 = 0.055$

$R^2 = .97$
Control Design

\[ u(k) = u(k-1) + K_I e(k) \]

\[ F_R(z) = \text{Closed Loop Transfer Function} \]

\[ K_I = 0.1 \]

\[ K_I = 1 \]

\[ K_I = 5 \]
Methodology for Using Control Theory in Performance Engineering

1. Determine the objectives of the control system – type of control (e.g., disturbance rejection), goals in terms of SASO properties

2. Map the problem into the control system framework
   - Target system(s): Sensors, Effectors
   - Controller
   - Transducer (e.g., use calculated metrics?)
   - Reference value (policy)

3. System identification
   - (Simple) Mathematical description

4. Control design
   - Parameterize the controller

5. Assessment

![Diagram of the system with MaxUsers, Target System, Measured RIS, Desired RIS, Controller, Server, Sensor](image-url)
Practice

- Utilities Throttling (DB2 v8.2)

- Self-tuning memory management (DB2 v9.1)
The Utilities Throttling Problem

Utilities have a big impact on production performance.

**Administrative policy**

There should be no more than an $x\%$ performance degradation of production work as a result of executing administrative utilities.

Drops by >70%!!
Choosing an Effector

CPU Priority is ineffective for controlling BACKUP
Control Architecture

Online modeling provides a transducer that translates from Pages/sec (Y) to % Impact (M)

θ = (a, b)

Y* = a + b

M = \frac{Y* - Y}{Y*}

E = R - M
Assessments of Control System
System Description: DB/2 UDB – Self-Tuning Memory

Clients

UDB Server

Client Application
UDB Client Library

Async IO Prefetch Requests

Coordinators
Agent

Subagents

Buffer Pool(s)

Prefetchers

Page Cleaners

Disks

Clients

Log
Buffer

Logger

Deadlock
Detector

Write Log
Requests

Notifications

ShrMem+Sem, TCPIP, Named Pipes, NetBIOS, SNA, IPX/SPX

Logger

Deadlock
Detector

Scatter/Gather
I/Os

Parallel, Big-block,
Read Requests

Parallel, Page
Write Requests

System Description: DB/2 UDB – Self-Tuning Memory
Database Memory Management

Optimize performance by equalizing loads across the memory pools.

Challenges:
1. Hand tuning is difficult
2. Memory is constrained
3. Many memory pools
4. Workload varies
Multiple Input Multiple Output (MIMO) Control

\[
J = q \sum_{k=1}^{\infty} \sum_{i=1}^{N} e_i^2(k) + q \sum_{k=1}^{\infty} \sum_{i=1}^{N} e_{I,i}^2(k) + r \sum_{k=1}^{\infty} \sum_{i=1}^{N} u_i^2(k)
\]

Cost of load imbalance \hspace{1cm} \text{Cost of control}
Decision support workload
- 2 classes of long running transactions
- Resource requirements vary over time

Study effect on total query response time ($T_S$) in TPC-H.

Without controller

With controller

59% Reduction in Total RT
Conclusions

- Control systems consists of elements
  - Controller, target system, transducer, filter, adapter, …

- Control objectives for computing systems focus on
  - SASO: Stability, accuracy, settling time, overshoot

- Methodology for using control theory in performance engineering
  - Determine goals, map framework, system identification, control design, assessment

- These simple models and analyses have been used in performance engineering of DB2
  - Regulating the execution of administrative utilities
  - Self-tuning memory management
Bibliography – Control Theory Textbooks


