Framework for Consolidated Workload Adaptive Management

Marcin Jarząb, Krzysztof Zieliński
Department of Computer Science, AGH-UST
Kraków, Poland
Agenda

- Virtualization technologies,
- Model of Adaptive Systems,
- Simple algorithms for adaptive management of workloads,
- Solaris 10 Containers,
- Architecture of framework for adaptive management of Solaris containers,
- Practical aspects of using simple policies for management of workloads within Solaris 10 environment.
Consolidation through Virtualization

The big challenge for consolidating multiple applications into a single physical server is to provide mechanisms of control over the resources (CPU, memory portions or network bandwidth).

- **Container based** - there is only one underlying operating system kernel, which the containers enhance by providing distinct borders offering increased isolation between groups of processes (OpenVZ, Solaris Containers),

- **Paravirtualization** – provides a virtual machine and access to the native hardware, and thereby lets users run many instances of different OS’s (VMWare, XEN).

**Efficiency and correctness of the control strategy depends on many parameters, all of which must be very carefully identified.**

To automate such a task some adaptation techniques should be used e.g. *Control Theory, Fuzzy Logic, Decision Trees.*
Our effort was to design and implement a framework which integrates variety of resources and exposes them through well known interface to adaptive manager.
Solaris 10 Containers

Rich virtualization and resource management facilities:

- Two level *Fair Share Scheduler (Zones, Projects)*,
- *Dynamic Resource Pools*.

CPU entitlement of workload \( E_w = \frac{S_w}{\sum_i^N S_i} \) (1)

\( S_w \) – shares assigned to workload \( W \), \( S_i \) – shares assigned to active workload \( i=\{1,..,N\} \)

Source: "Consolidating Applications with Solaris Containers", Sun Microsystems Technical Whitepaper, November 2004
Workload controller implementation
Case study

- Open-loop AM workload manager, exploiting the FSS model, after transformation of equation (1):

\[ S_{w}^{t} = \left( U_{w} * \sum_{i \neq s} N_{w}^{i} S_{i}^{t} A_{i}^{t} \right) / \left( 1 - U_{w} \right) \]

where \( U_{w} \) is a target CPU usage (2)

Number of active workload is changing at time \( t \) according to activity state vector \( A_{i} = [A_{i}^{1}, ..., A_{i}^{N_{w}}] \), where \( A_{i}^{1} = 0 \) if \( W_{i} \) is not active and \( A_{i}^{1} = 1 \) if \( W_{i} \) is active, \( i = \{1, ..., N_{w}\} \)

- Closed-loop AM workload manager, which directly tunes Containers’ or Projects’ resource shares to achieve desired CPU allocation to the workload.

Proportional regulator

\[ S_{w}^{t+1} = S_{w}^{t} + K_{p} * e(t) \]

where \( e(t) = U_{w}^{t} - U_{w} \) (3)

Proportional-Integral regulator

\[ S_{w}^{t+1} = S_{w}^{t} + K_{p} * e(t) + K_{i} \sum_{i}^{t} e(t) \]

(4)

\( K_{p} \) and \( K_{i} \) coefficients are calculated using analysis of step response method

- Computer systems are non-linear (linear in some interval), response delay thus hybrid controllers using some rules should be used.
Adaptive Management of Virtualized Resources with JIMS

- Control loop consists of four basic steps: **Monitor, Analyze, Plan, Execute** which exploit knowledge collected during system activity,
- It requires exposition of virtualized resources using **Managed Element** interface which represents computer resources instrumented with sensors and effectors,
- It’s designed as a extension of JIMS (Java Infrastructure Monitoring System) platform implemented as a set of JMX MBeans,
- Each container have separate instances of **Effector** and **Sensor** MBeans exposed using JMX connectors (RMI, SOAP) making available them to a decision subsystem.

![Diagram of Adaptive Management of Virtualized Resources with JIMS](image)
Solaris 10 Management
Case study

Implementation based on hybrid controller
- P/PI algorithms,
- Jacobson algorithm
  \[ U_{est} = coef \times U_{t-1} + (1-coef)\times U_t \]
- Rules (irregular thread monitoring scheduling, CPU bound workloads)

**Target CPU usage** \( U_w = 70\% \)

**Constant disturbance**

**Variable disturbance**

**Comparison of the coefficient values for the P regulator**

**Comparison of P and PI regulators with and without Jacobson**

*Best results for* \( K_p = 7 \)

*Integral of squared error: P (102271), PI (100413) Jacobson coefficient: 0.4*
Summary

- Primary contribution of our work is a implementation of framework based on JMX technology for adaptive management of virtualized resources,
- Implemented framework was verified for a simple control policy,
- It opens a very wide area of research for control strategy selection which might use heuristic rules or fuzzy logic.