Continuous Performance Analysis of Fault-Tolerant Virtual Machines

The Concept of an Execution Platform

Boguslaw Jablkowski
Olaf Spinczyk
Department of Computer Science
TU Dortmund
Overview

• Motivation
• Architecture
• System View
• Models and Realization Techniques
  – Real-Time Calculus
  – Performance Model
  – Fault & Dependability Model
• Scenario
• Algorithmic Challenges
• Conclusion and Outlook
Motivation

• Integration of distributed large-scale cyber-physical systems:
  – decoupling of software from dedicated hardware (vendor lock-in)
  – capital and maintenance costs reduction (energy, administration)
  – ease of system extension (new functionalities, new protocols)

• Context: Electric power systems*
  – Example: Switchgear

*DFG research unit FOR1511, *Protection and Control Systems for Reliable and Secure Operation of Electrical Transmission Systems*
Virtualization

- Physical Host
  - Hypervisor
    - Domain 0 privileged
      - VM
        - Management & Monitoring System (MMS)
    - VM
      - User Software
        - Guest-OS
    - VM
      - User Software
        - Guest-OS
  - Hardware
  - App.
  - App.
  - App.
Overview

• Motivation
• Architecture
• System View
• Models and Realization Techniques
  – Real-Time Calculus
  – Performance Model
  – Fault & Dependability Model
• Scenario
• Algorithmic Challenges
• Conclusion and Outlook
Architecture

- Hardware
  - x86 platform

- Hypervisor:
  - Xen 4.1.2
Architecture

- Management & Monitoring System:
  - Distributed (on every Server)
  - Timing and content analysis of VMs
  - Proactive mechanisms
  - Management of VMs

- Virtual machine:
  - User software (application)
  - Tailored operating system
Architecture

- Management & Monitoring System (MMS)
- Domain 0 (privileged)
- Hypervisor
- Hardware
- Physical Host
- User Software
- Guest-OS
- User Software
- Guest-OS
- VM
Architecture

- Communication interfaces:
  - Inter-node
  - Intra-node
Overview

• Motivation
• Architecture
• **System View**
• Models and Realization Techniques
  - Real-Time Calculus
  - Performance Model
  - Fault & Dependability Model
• Scenario
• Algorithmic Challenges
• Conclusion and Outlook
System View

- Determining system data:
  - Environment
  - Hardware
  - Applications
- Static planning and scheduling

- Monitoring
- Resolving hypothetical faults:
  - Searching the configuration space and determining candidates for new system configuration
  - Performance analysis
  - Saving feasible configurations

- Migrating of VMs
- Replication of VMs
- (Re)booting of VMs
- Switching to backup hosts

Proactive mechanisms

Fault or maintenance

Adaptation mechanisms
Overview

• Motivation
• Architecture
• System View
• Models and Realization Techniques
  – Real-Time Calculus
  – Performance Model
  – Fault & Dependability Model
• Scenario
• Algorithmic Challenges
• Conclusion and Outlook
Real-Time Calculus

- Formal performance analysis technique for distributed real-time systems
- Computes bounds for non-functional system properties:
  - Execution delays
  - Buffer utilization
  - Network utilization
- Event streams and hardware capacities represented as functions
- Based on max-min-plus algebra
- Bounds can be mathematically proven correct
Performance Model

• Unifies information about the system:
  • Environment (events)
  • Hardware (capacities)
  • Applications (deadlines)

• Needed for:
  • Planning and scheduling
  • Proactive mechanisms
Performance Model – Modeling Abstraction Level

- Modeling abstraction level:
  - Entire physical host as Server
  - Entire virtual machine as Node
  - Other possibility:
    - Hierarchical scheduling
Fault and Dependability Models

Fault assumption:
• Overwriting of another processes data
• Blocking the CPU

Dependability mechanisms:
1. Standard execution of a VM
Fault and Dependability Models

Fault assumption:
• Overwriting of another processes data
• Blocking the CPU
• Corrupted function output of a single VM

Dependability mechanisms:
1. Standard execution of a VM
2. Redundant execution on a single physical host
Fault and Dependability Models

Fault assumption:
- Overwriting of another processes data
- Blocking the CPU
- Corrupted function output of a single VM
- Failure of a physical host

Dependability mechanisms:
1. Standard execution of a VM
2. Redundant execution on a single physical host
3. Redundant execution on multiple physical hosts and/or backup on multiple hosts
Overview

• Motivation
• Architecture
• System View
• Models and Realization Techniques
  – Real-Time Calculus
  – Performance Model
  – Fault & Dependability Model
• Scenario
• Algorithmic Challenges
• Conclusion and Outlook
Server Failure Scenario

Scenario flow:
• System is working properly
• Failure of server one
• Backup VM’’1 takes over
Server Failure Scenario

Scenario flow:
• VM1, VM’1 and VM2 have to be rescheduled
• Two challenges:
  • Candidates for new system configuration?
Server Failure Scenario

Scenario flow:
• VM1, VM’1 and VM2 have to be rescheduled
• Two challenges:
  • Candidates for new system configuration?
    • Suppose this candidate
  • Real-time capabilities of the new system? Proven bounds?
Overview

- Motivation
- Architecture
- System View
- Models and Realization Techniques
  - Real-Time Calculus
  - Performance Model
  - Fault & Dependability Model
- Scenario
- Algorithmic Challenges
- Conclusion and Outlook
Algorithmic Challenges

Searching for new system configurations:
  • Modeling the scheduling problem as an optimization problem
  • Optimization technique:
    • Genetic programming (other methods possible)
  • Problem representation:
    • Includes already correctly scheduled parts of the system
  • Fitness function:
    • Data dependability
    • Deadlines fulfillment

Continuous performance analysis and validation:
  • Two possible validation approaches:
    1. Compute candidates first and validate them with real-time calculus
    2. Use real-time calculus in the fitness function
Overview

• Motivation
• Architecture
• System View
• Models and Realization Techniques
  – Real-Time Calculus
  – Performance Model
  – Fault & Dependability Model
• Scenario
• Algorithmic Challenges
• Conclusion and Outlook
Conclusion and Outlook

- Relevant and complex undertaking
- Architecture involves a wide range of techniques:
  - Virtualization
  - Tailored operating systems
  - Fault tolerance
  - Scheduling/optimization techniques
  - Formal performance analysis

Further validation and evaluation of the architecture in order to guarantee fault tolerance and real-time capabilities.

Thank You