Managed Virtualized Platforms: From Multicore Nodes to Distributed Cloud Infrastructures

Ada Gavrilovska
Karsten Schwan, Mukil Kesavan
Sanjay Kumar, Ripal Nathuji, Adit Ranadive

Center for Experimental Research in Computer Systems
Georgia Institute of Technology
Group Overview

www.cercs.gatech.edu/projects/virtualization/virt.htm

- Central themes:
  - Scalable, high-performance virtualization solutions
    - for IO, hypervisor structures, virtualized services...
  - Heterogeneous distributed environments
    - mobile and pervasive
    - heterogeneous manycore platforms (including accelerators such as GPUs, Cell, communication accelerators, ...)
    - datacenter and large-scale distributed compute clouds
  - Active resource management
    - continuous VM and platform monitoring
    - profile- and observation-driven analysis methods
    - dynamic inter- and intra-node resource allocation policies
- Work supported by or in collaboration with Intel, IBM, HP, Motorola, Cisco, OSIsoft, NVidia...
Presentation Overview

• Focus on management mechanisms in multicore environments
• Two case studies based on prototype implementations
  – CPU and IO coordination
    • Mukil Kesavan poster presentation
  – Performance and power utilization
• Compute cloud for critical enterprise workloads
Managed Virtualized Platforms

Challenges

• Quality of Service:
  – meet expected VM-level SLAs
    • SLA metric?
    • input from application/VM?
  – individual as well as sets of VMs

• Dynamism:
  – deal with bursty application/VM behavior
  – enable good resource utilization
    • static, worst-case allocation policies insufficient

• Coordination:
  – across multiple VMs’ and their policies for management of virtual resources
    • e.g., VMs’ OSs make conflicting decisions regarding platform power management
  – across different management layers
    • e.g., HP’s iLO management hardware and VMM CPU scheduler
  – allocation decisions regarding one resource type require adjustments to other resources
    • e.g., IO buffer size and CPU scheduling
Management Architecture

• Management brokers
  – make and enforce ‘localized’ management decisions
    • within VMs
    • VMM-level – CPU scheduling, allocation of memory or device resources, ..
    • at hardware level

• Management channels
  – enable inter-broker coordination through well-defined interfaces
  – event and shared memory based

• Management VMs
  – platform wide policies and cross-platform coordination
Management VM

- Interface to management ‘sensors’ and ‘actuators’
- Implements coordination and mediation across range of resource types and management layers
- Allows instantiation of user/administrator defined policies and resource management algorithms
Representing Platform Resources

- **Platform Units**
  - vector representing aggregate platform resources and properties
    - CPU, memory, IO, power budget, ...
    - reliability, trust, architecture type ...
- **Class of Service**
  - mapping of VM’s SLA to vector of resource requirements
  - continuously refined based on VM profile, specific input or runtime behavior
    - static CoS-level (Gold, Silver, Bronze) determines initial allocation and fluctuation limits
    - dynamically adjust runtime allocation within specific boundaries
- **Compensation Credits**
  - encourage VM’s participation in management processes
Resource Allocation Policies

- Enforced within platform level management VM
- External rules
  - static CoS specifications
  - well-understood exceptions
- VM inputs
  - management agents in VMs’ OSs or applications
    - e.g., platform power states
    - e.g., application agents leveraging WSDM standards
- Observation-based
  - black-box runtime monitoring of per VM resource utilization
  - support for range of algorithms, machine learning or statistical techniques...
- Profile-based
  - rely on offline analysis of VM behaviors for classes of workloads, correlation techniques, etc...
Example 1: Coordinated CPU and IO Management

- Even use of VMM-bypass capable devices introduce significant IO performance variability based on CPU resource allocation.
Example 1: Coordinated CPU and IO Management

• Testbed:
  – Multiple dual-socket quad-core x86 nodes
  – Interconnected via InfiniBand fabric to Xsigo VP780 I/O Director
  – Ethernet vnics exported to VMs; Ethernet – InfiniBand translation performed in control domain

• Management brokers:
  – CPU management through CPU credits
  – IO management through QoS limits to vnics enforced via Xsigo switch

• Workloads:
  – Gold: 80% CPU, 200Mbps; Silver: 60% CPU, 125Mbps; Bronze: 40%, 75Mbps
  – RUBiS: All 3 VMs Gold. Requests per second – the more the better.
  – Hadoop: Master VM Gold, Slave VMs Bronze. Execution time – the lower the better.
  – Iperf: Silver VMs. Throughput – the more the better.
  – Spec-h264ref: Gold VM. Execution time – the lower the better.
Example 1: Coordinated CPU and IO Management

• Algorithm:
  – Multiplicative Increase Subtractive Decrease with Reservations
    • based on TCP AIMD

  – Parameters determine
    • minimum resource guarantee
    • rate of change of resource allocation
    • inter-resource impact on allocations
    • experimentally derived
Example 1: Coordinated CPU and IO Management

- Allows fine tuning of resource allocations
- Maintains slack in reservation
- Monitoring frequency configurable
Example 1: Coordinated CPU and IO Management

- Ability to distribute resource based on VM importance

![Graphs showing resource distribution for RUBiS, Hadoop, Iperf, and SPEC-h264 with different scenarios: ACT Norm, CPU Oversub, 320:300, Pinning, and NW Oversub 1250:1000.]
Example 1: Coordinated CPU and IO Management

• Tradeoffs between resource consolidation opportunities vs. attainable performance
Example 2: Coordinated CPU and Power Management

- Same management architecture is used to perform power management decisions based on CPU utilization or SLA violation feedback provided by VM
Example 2: Coordinated CPU and Power Management

- Lack of coordination (left) triggers repeated oscillations in resource utilization
- Coordination reduces violations and helps determine migration thresholds
Example 2: Coordinated CPU and Power Management

- Experiment: High rate and low rate transaction VMs running on P4 platform
- Both VMs have same CoS value, therefore equal allocation of PUs (power) without runtime input
Example 2: Coordinated CPU and Power Management

- Coordinated power management (DVFS) + load management (migration) + CPU management (credit based soft scaling) -> cumulative reduction of 34% in power resources without SLA degradation of RUBis benchmark
ESX Integration

• Challenges
  – ESX 3.0.1
  – documentation of APIs, data structures...
  – not all components available as source code

• Status
  – coordinated CPU & IO management with same MISD-WR algorithm
  – used CPU scheduling and network token bucket shaping mechanisms in VMM kernel
    • Ethernet based interconnect
  – experimentation with synthetic workloads based on Iperf and SPEC-h264f results in same behavior as earlier results
Towards Distributed Clouds

• Build overlays between platform-level management domains
  – leverage our group’s high performance eventing middleware EVPath

• Placement of management logic
  – centralized at top level
  – distributed clustered hierarchies
  – localized at individual nodes for low-latency decisions

• Introduce statistical guarantees for allocation of shared resources:
  – e.g., guarantee bandwidth 150Mbps 95% of the time.
Critical Enterprise Cloud Computing System (CECCS)

- Enable efficient resource sharing by enterprise workloads with dynamic behaviors
- Infrastructure supported by IBM
  - CERCS Georgia Tech and OSU resources
  - additional GT locations
- Open testbed to facilitate research and education
  - generate workloads through classroom use and GT or external client applications
- Extend with additional capabilities
  - management of environmental facilities-level properties
Concluding Remarks

- Active management approach creates significant opportunities for consolidation and reduced resource utilization
- Coordination across different management layers and resource types leads to more adequate resource allocation decisions
- The same approach is being extended
  - to federate management decisions across administrative domains in distributed compute clouds
  - to coordinate across IT and facilities management techniques
  - to deal with increasingly more heterogeneous platforms
  - from different types of cores on individual manycore nodes, to different management and virtualization stacks across the distributed environment