SAND2010-3174C

Using Cloud Constructs and Predictive Analysis to Enable Pre-Failure Process Migration in HPC Systems

Brandt, Chen, De Sapio, Gentile,

Mayo, Pebay, Thompson, and Wong

Sandia National Laboratories

This work was supported by the United States Department of Energy, Office of Defense Programs. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

Outline

- Motivation
  - Current resource allocation models
  - How resource utilization monitoring and dynamic configuration can improve performance
- System
  - Virtualization for HPC (MPI) Apps
  - Resource Characterizations
  - Resource Manager
  - Orchestration
- Examples
  - Resource health degradation
  - Resource contention

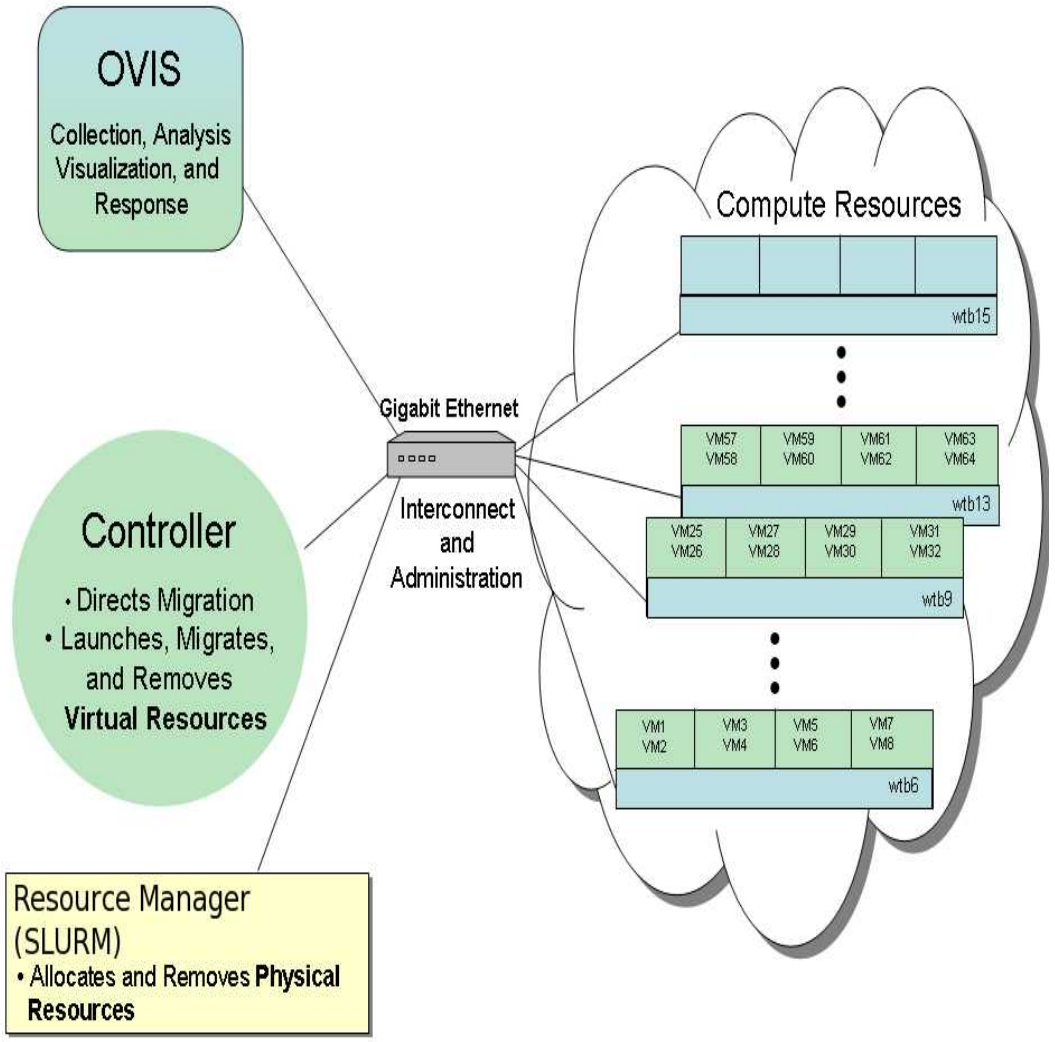
Motivation

- Future architectures will have vastly increased numbers of processing elements per node and more shared subsystem resources (e.g. cache, main memory bus, network)
  - Increased likelihood of contention (e.g. memory and communication buses) within an applications processes and among multiple applications
- Dynamic reconfiguration could enable increased system performance
  - Progress in the face of resource failure

Components and Enabling Technologies

- Scalable monitoring and analysis system
  - Provides real-time fine-grained collection and analysis of hardware state information
  - Provides facility for obtaining meaningful real-time, low-level resource utilization information
- Low overhead virtualization technologies
  - Provide simple process level mobility that can enable dynamic resource re-balancing in response to application requirements, system state, and/or failure prediction
- Advanced resource manager/scheduler
  - Manage and allocate both physical and virtual resources
- Controller
  - Orchestrate allocation, evaluation, and re-balancing of resources with respect to the applications running on a system, their resource requirements, and priorities

Prototype HPC System for Pre-failure Process Migration



- Resource monitoring and characterization – OVIS
- Resource Manager – SLURM
- Controller
  - Dynamic mapping of virtual to physical resources
  - Orchestrates and controls launch and migration of virtual resources
  - Requests and releases physical resource allocations
- Compute Resources
  - Base physical infrastructure on which virtual resources are hosted and applications run
- Gigabit Ethernet interconnect
  - Communication infrastructure for both application and system administration

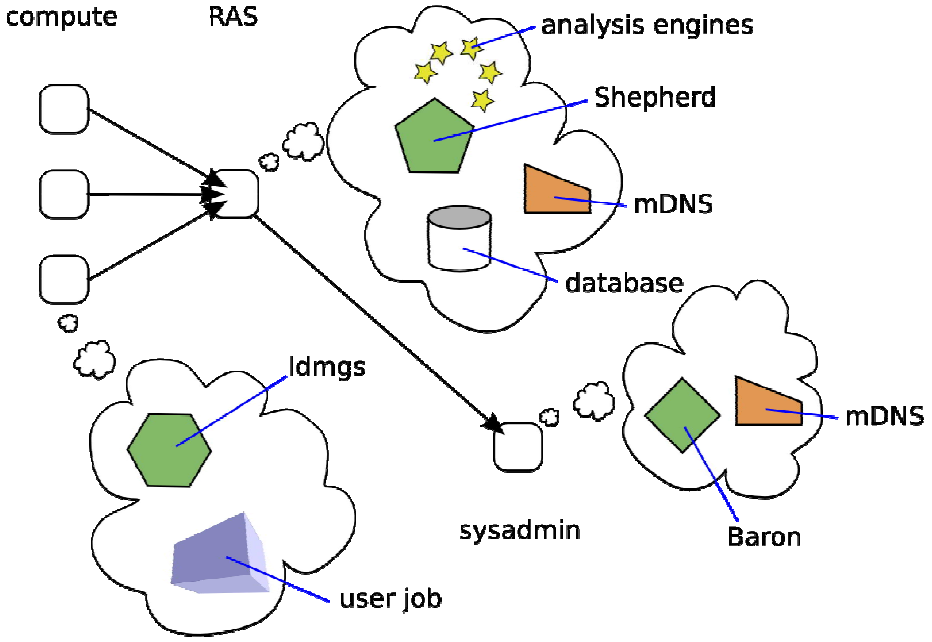
Running HPC (MPI) Applications in a Virtualized Environment

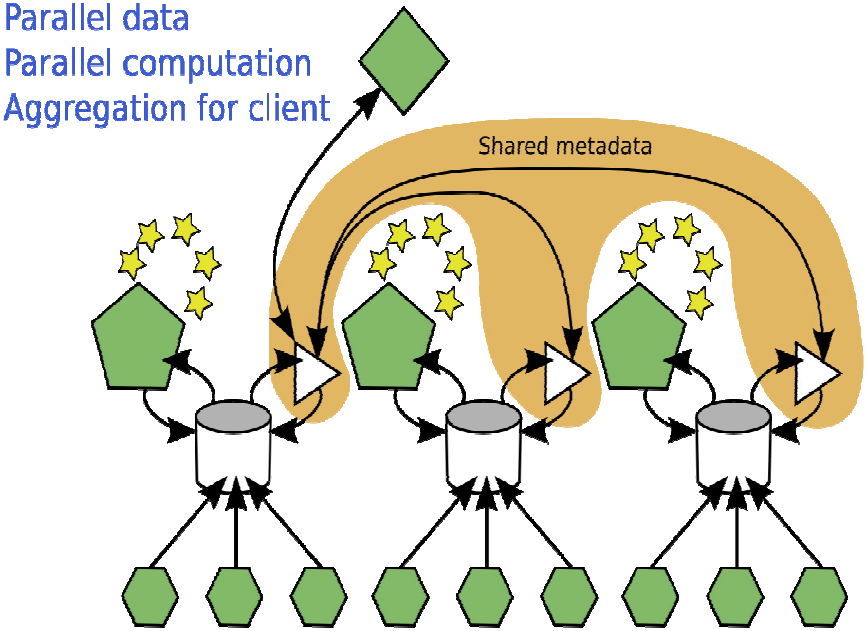
- Overhead
  - Claims of from zero to 3% depending on which flavor is being used and tuning
  - We ignore this aspect in this work as studies show it is currently viable and getting better
- Provisioning and Mobility for MPI Applications
  - MPI_Barrier wrapper as an easy way to ensure against in-flight message loss
    - Only Migrate at a barrier
  - Dr. Panda has included facility for addressing this in the context of Xen Para-virtualization in MVAPICH

Dynamic Reconfiguration using Live Migration

- Migration of virtual machine while application processes are still running
  - Migration time bounded by Gigabit Ethernet interconnect speed
- Use simple MPI_Barrier wrapper to coordinate migration with application
  - Only perform migration at MPI_Barrier
  - Checks for flags written by the Controller
  - Coordination across all MPI ranks so no in-flight messages lost
  - Minimize changing process state

Scalable Resource Monitoring and Analysis Architecture (OVIS)





Actionable Pre-failure Resource Characterizations

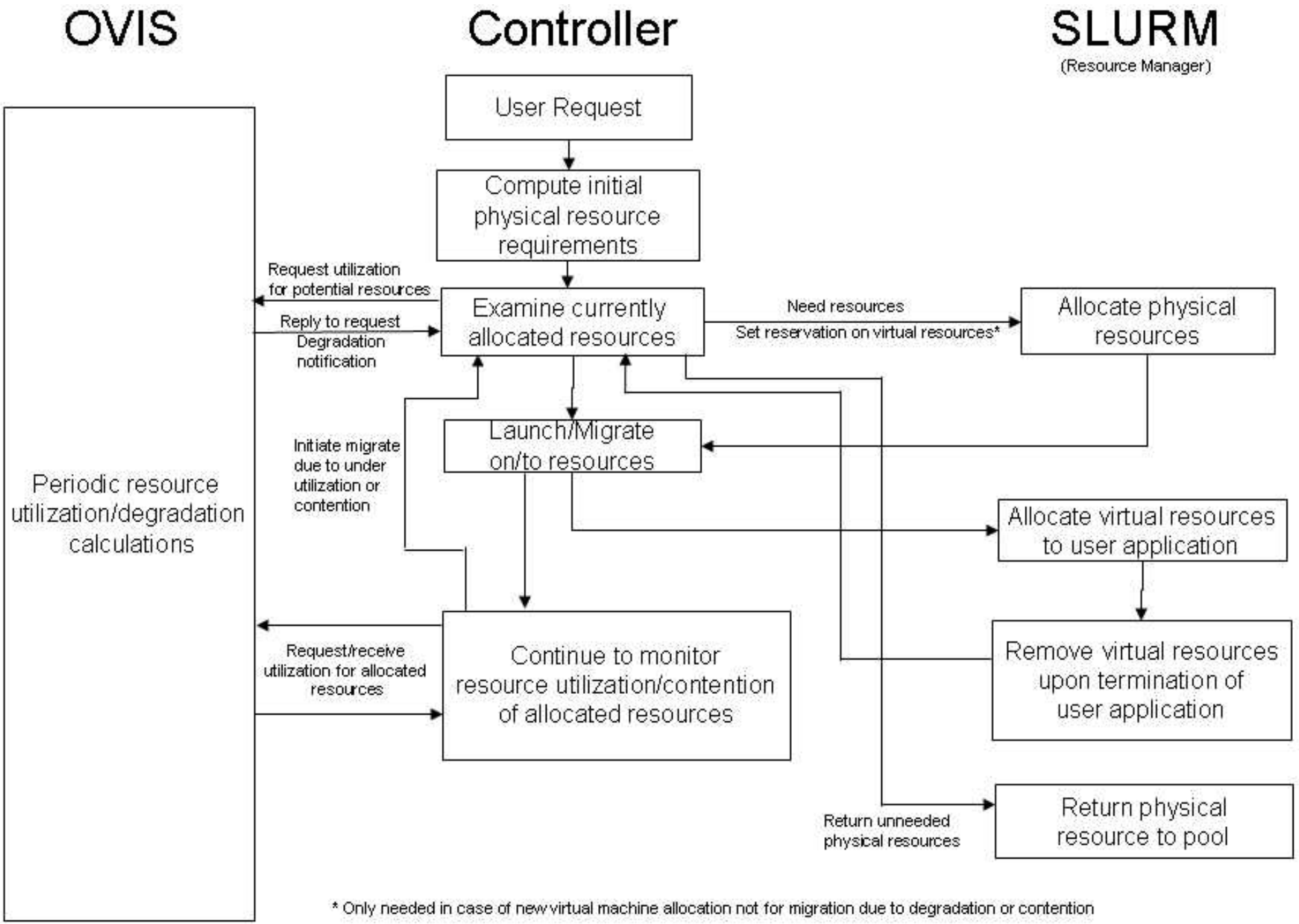
- Understand dynamic application resource characteristics
- Provide run-time profiling to detect performance issues and state change (e.g. increasing active memory utilization)
- Identify possible pre-failure conditions
  - CPU utilization, memory (including caches), internal buses, network
- Identify anomalous behavior
  - Relative to other resources hosting similar processes in similar environment
- Determine efficacy of reconfiguration: frequency, overhead, impact

Resource Manager

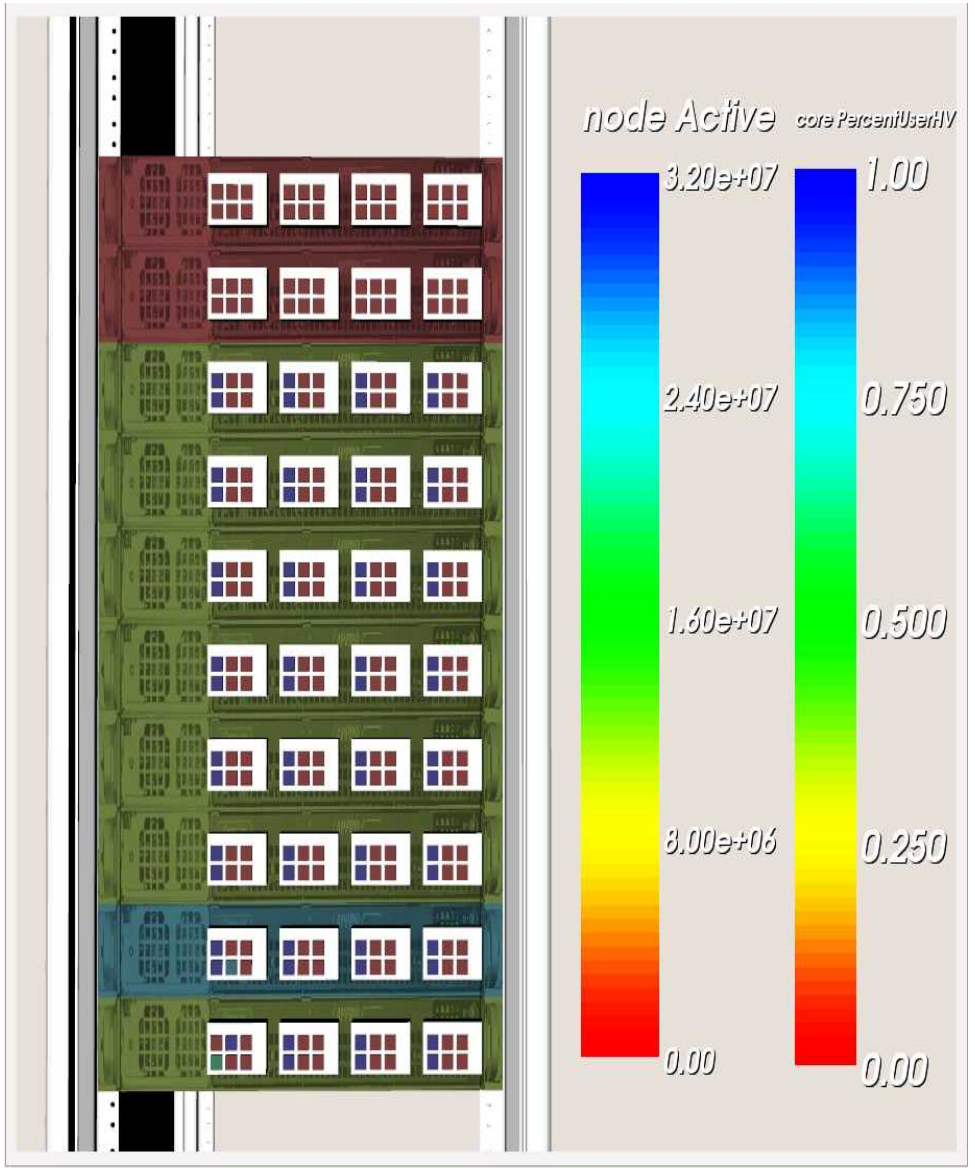
SLURM – Simple Linux Utility for Resource Management

- Can maintain separate resource partitions
  - Used separate **physical** and **virtual** resource partitions
- Supports resource allocation on a per-node, per-socket, and per-core granularity
  - Requested physical allocations on a **per-node** granularity
- Does not innately support virtual to physical resource mappings
- Does not currently support additional resource allocations to jobs

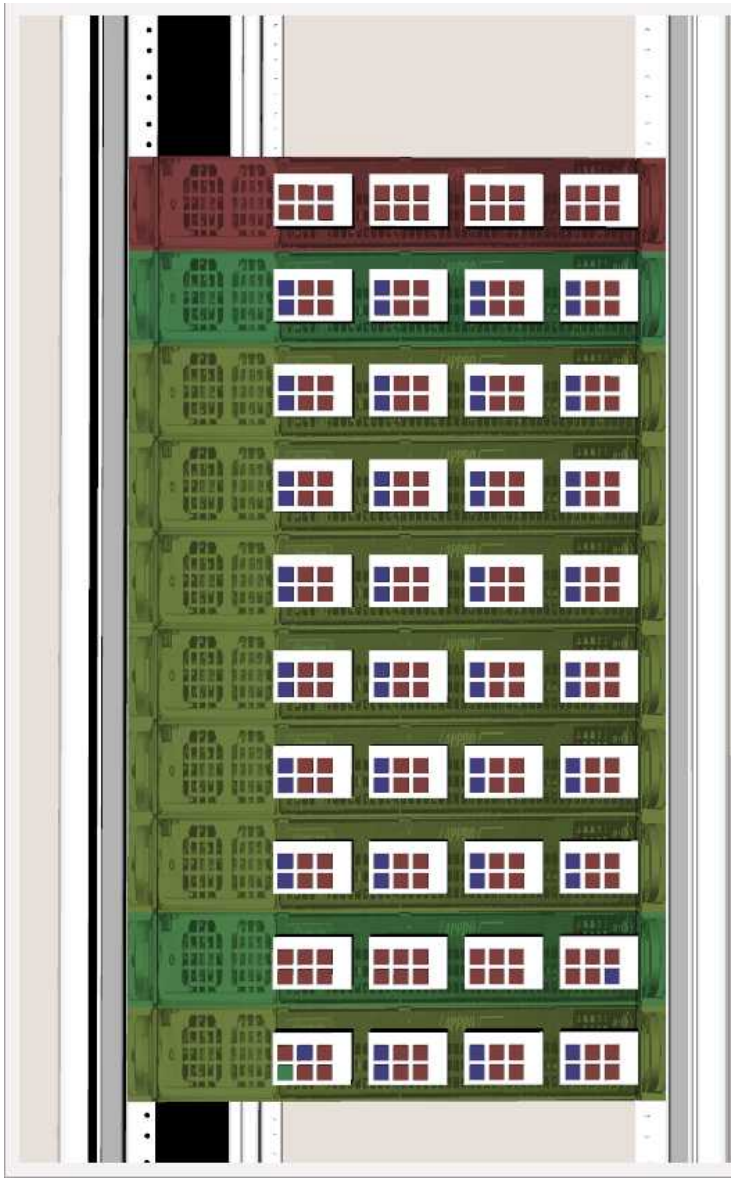
Orchestrating System Interactions



Example: Health Degradation

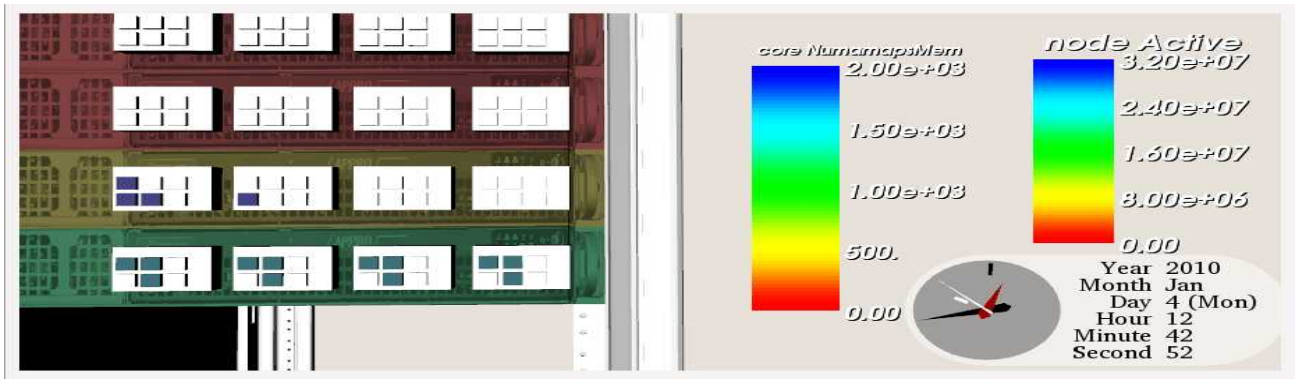


Anomalous condition detected



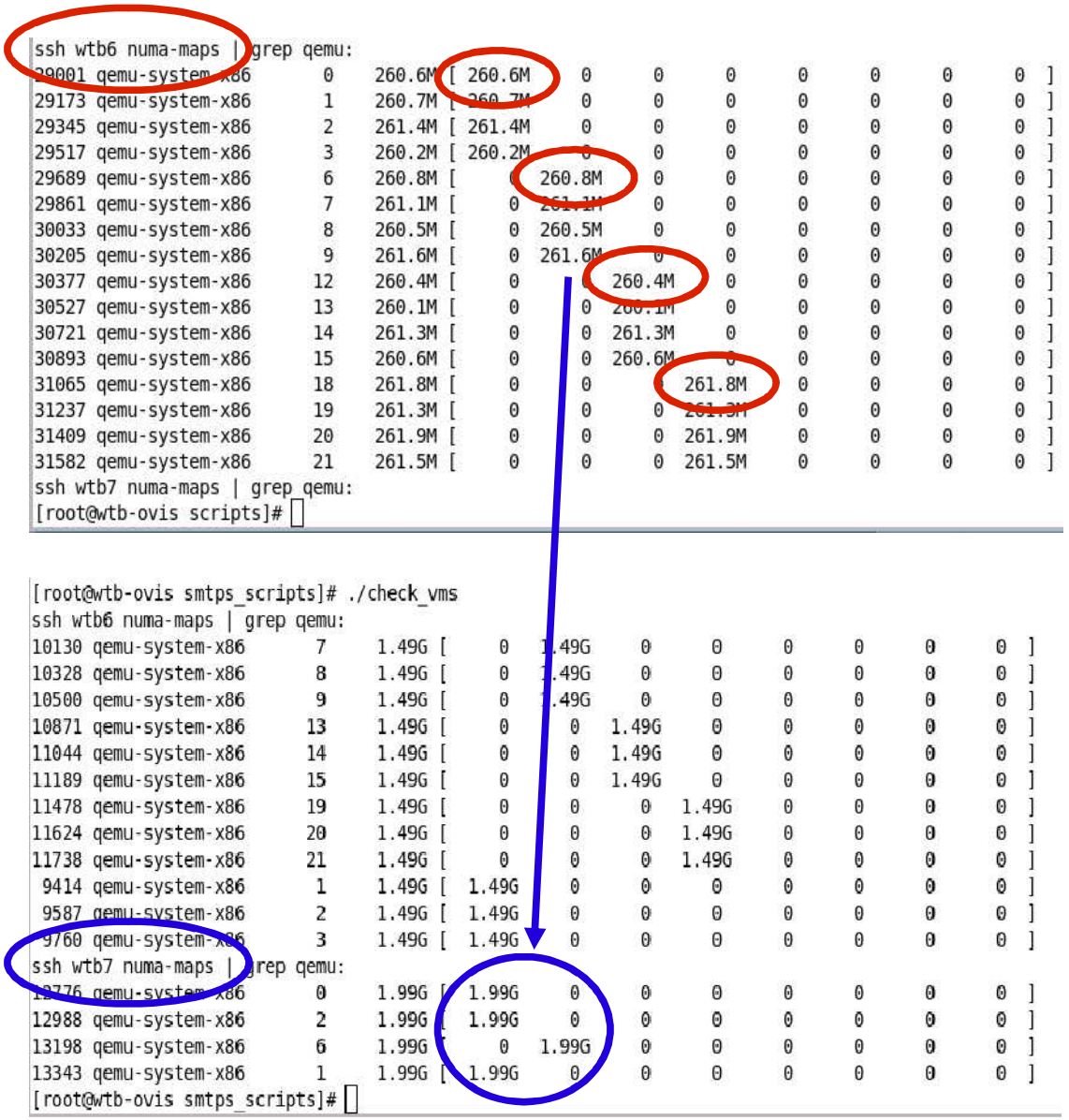
Anomalous condition mitigated via dynamic resource allocation and migration

Contention Detection Triggers Live Migration



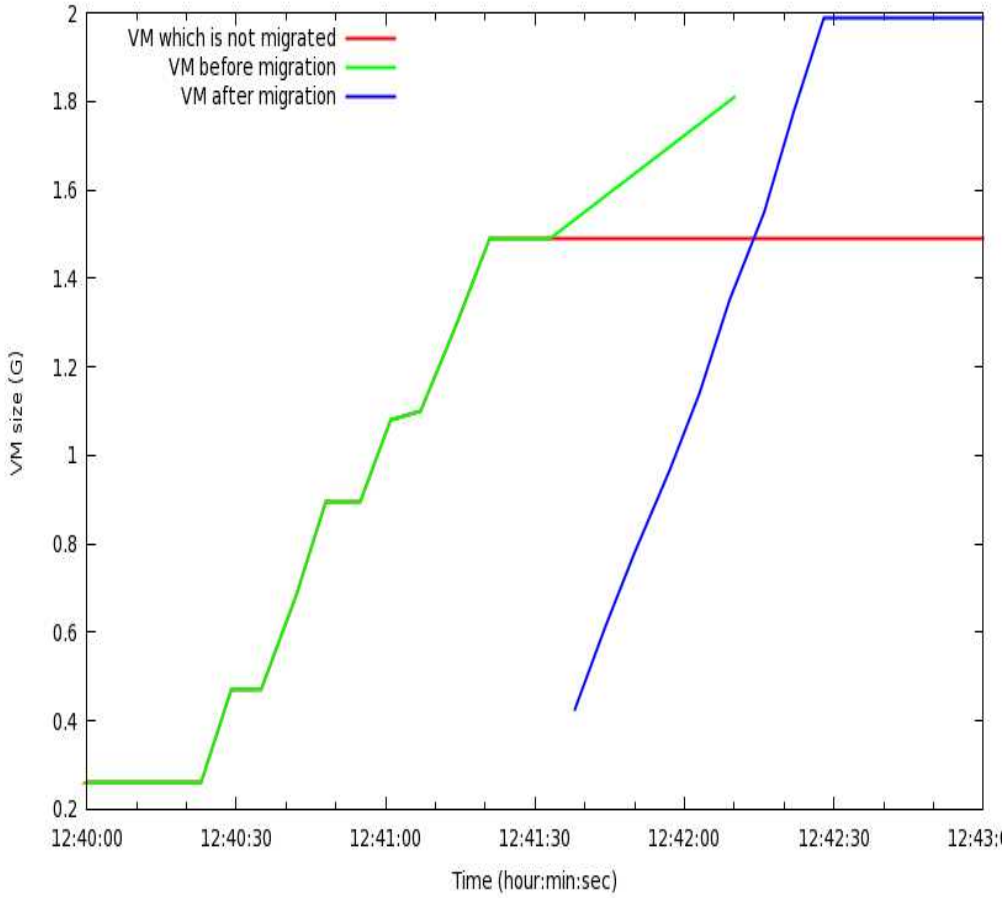
- 1VM/Core, 4VM/Socket
- Bind to 8GB of memory associated with socket
- Migration triggered by run-time detection of impending memory oversubscription
- Triggering (top)
- Migration occurring (middle)
- Migration complete (bottom)

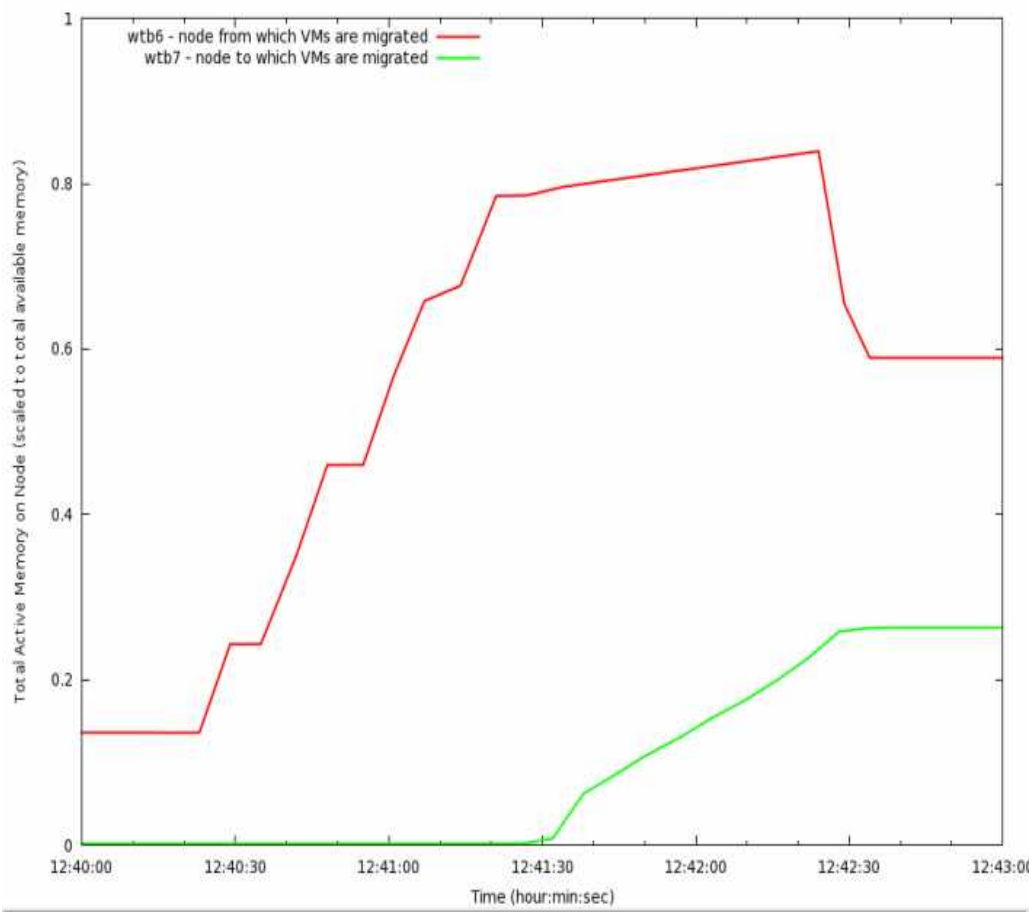
Example HPC Usage Model Leveraging VM Characteristics



- Container size at launch is minimum (top)
- After migration is maximum (bottom)
- Intentional oversubscription of resources
  - Start application on fewer resources and adjust as needs change and resources become available

Memory Utilization Monitoring





Future Work

- Determine resource characterizations of interest
- Decision making – When to migrate and where in environment with many applications running and sharing resources
- Efficacy of response – don't want to migrate too often, don't want to migrate to resources that will result in cascading bad effects
- Robustness of integrated system – Scalability and stability

Questions?

Resource View Considerations in Virtualized Environments

- Resource utilization looks different from the host than from the VM
  - Host view shows actual CPU utilization ----->
  - VM view shows actual memory footprint of application (described later)