On the Applicability of Virtualization in an Industrial Environment

VisorHPC 2017

January 24, 2017, Stockholm, Sweden

Josef Weidendorfer
Tilman Küstner
Carsten Trinitis

Chair for Computer Architecture

TECHNISCHE UNIVERSITÄT MÜNCHEN

Andreas Blaszcyk
Patrik Kaufmann
Marcus Johansson

ABB Corporate Research

Context

Compute systems for product design in industry

- allow to simulate different designs
- need large systems & parallelization
- are essential today
- example at ABB: electric field simulation
  - in-house parallel PVM/MPI code POLOPT
  - since early 1990s, still in production use
  - cooperation with TUM: optimization/porting
  - different geo-located systems & engineers' workstations
  - maintained for Linux/Windows, Client/Server etc.

Motivation

High maintenance / development efforts

- change to 3rd-party simulation codes?
- unify runtime environment by virtualization
  - remove dependence to target environments
  - only one version to maintain, easy deployment:
    64-bit x86, Linux, single MPI implementation
  - multiple virtualization solutions required on different target systems

Objectives

For each different target system:
what is the best virtualization solution?

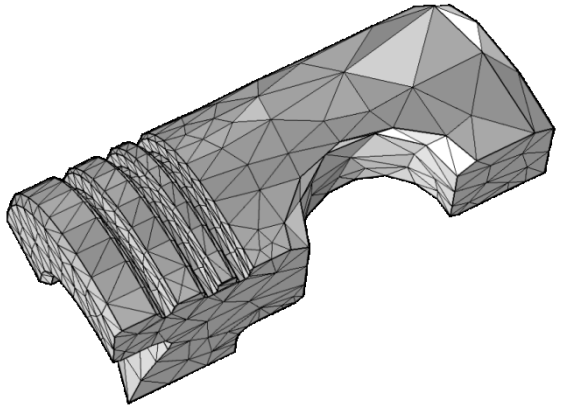
- importance of performance
  (on cluster: essential, on workstation: nice)
- permissions required
  (on cluster: root not available, on WS: fine)
- setup costs: software/adaptation scripts
  (once / regularly e.g. for security updates)

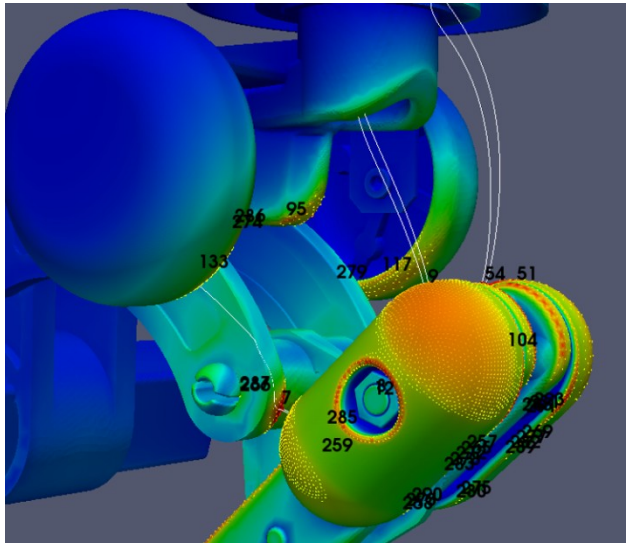
Outline

- POLOPT
- Virtualization Solutions
- Proposed Solution
- Measurements

POLOPT

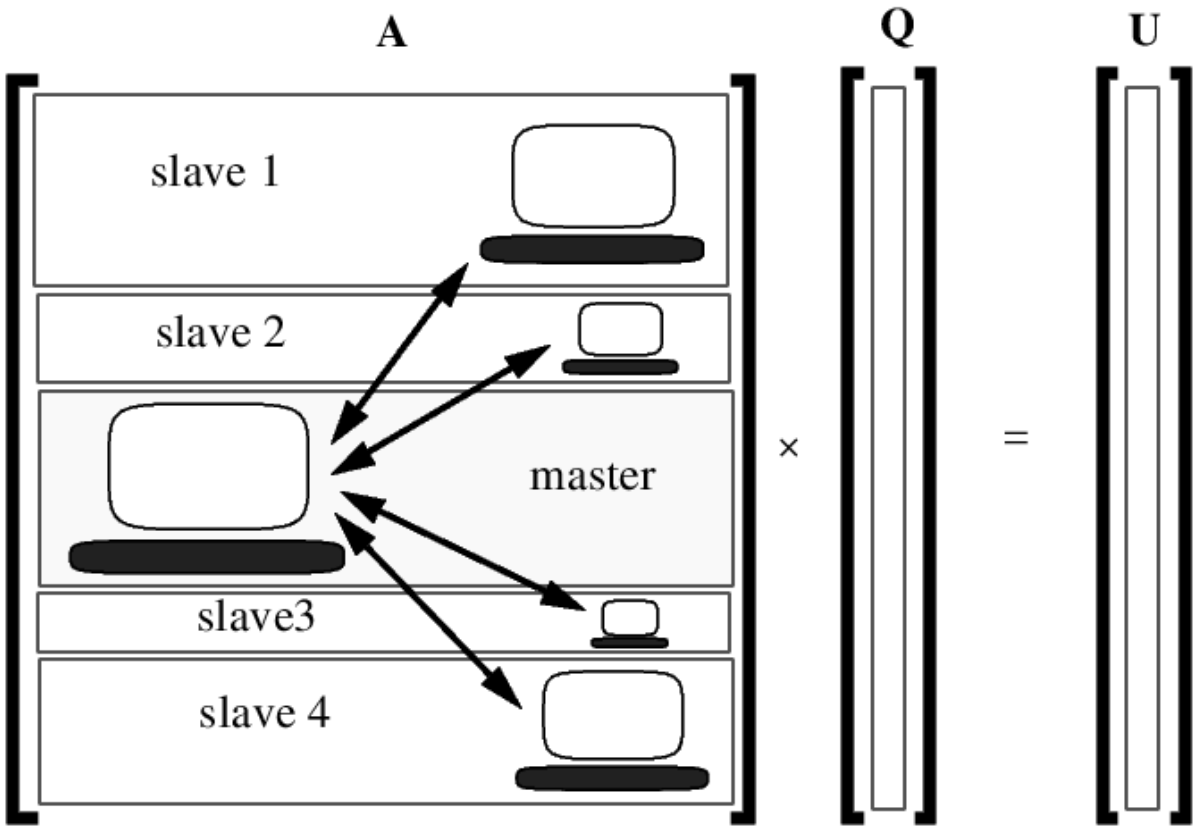
- Boundary Element Method (BEM)
- Unknowns (N): Electric charges on surface elements
- Two phases
  - matrix setup
    (size N^2, complexity: N^2)
  - iterative solver: GMRES
    (complexity: #iter * N^2)
  - both phases relevant
  - mostly streaming over data





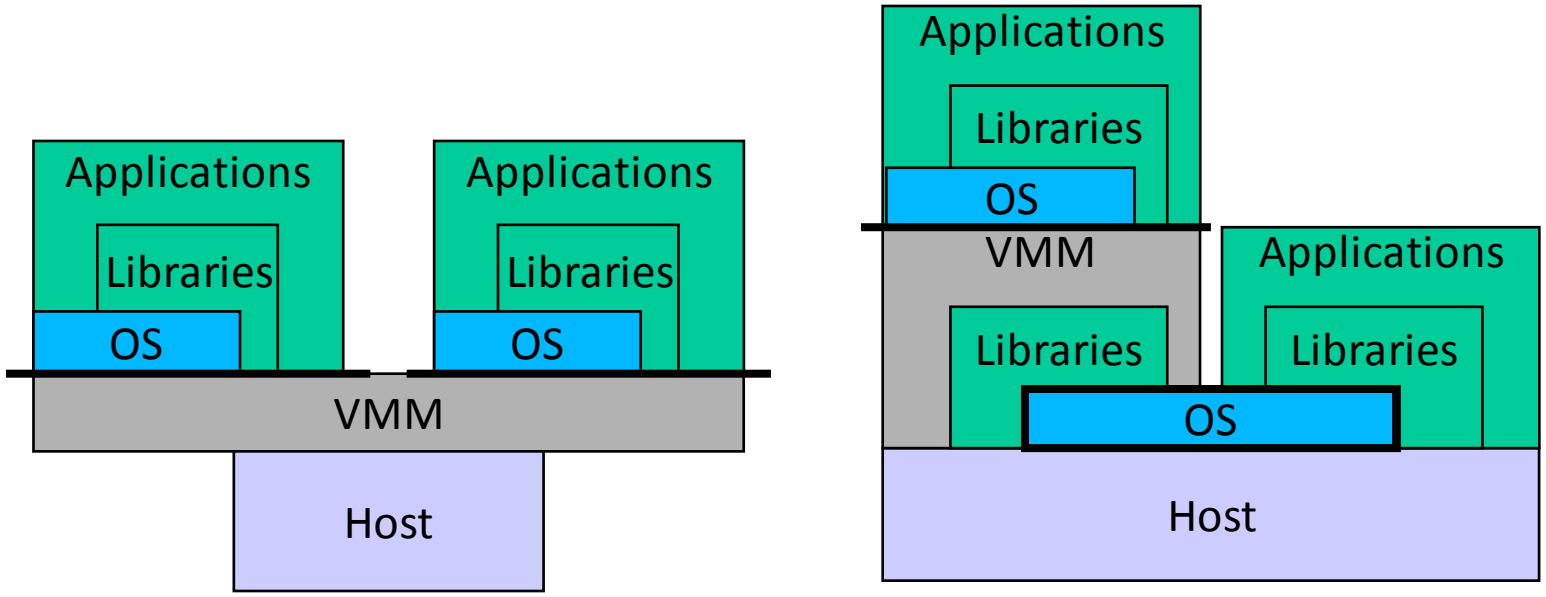
Parallelization

- Fix partitioning of matrix for parallelization
- Load balanced (static)
- Master also participates in computation



Virtualization Solutions

- System-VMs (VMWare, Xen, VirtualBox, KVM)



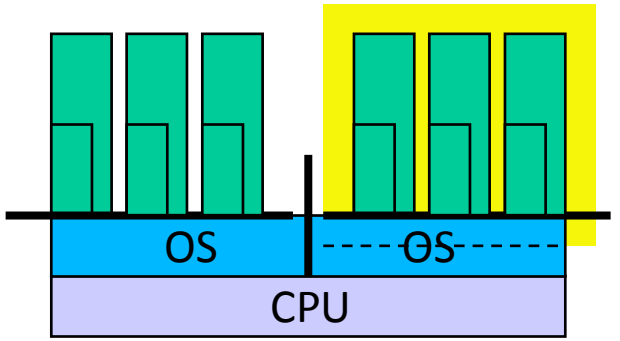
Native / Type-I

Hosted / Type-II

- different guest vs. host OS, often used in cloud
- typically non-negligible overhead (5-10 %)

Virtualization Solutions (2)

- OS-level virtualization = container (Solaris Zones, Jails, Docker)
  - indirection for OS-controlled resources (file system, net,…)
  - no inherent overhead (but: TCP)
  - needs root for creation (Linux: CGroups)
- lightweight: only chroot (preferred by LRZ)
  - Singularity: SUID binary, drops priveledges after chroot
  - PRoot: chroot emulation via PTrace hooks (slow!)

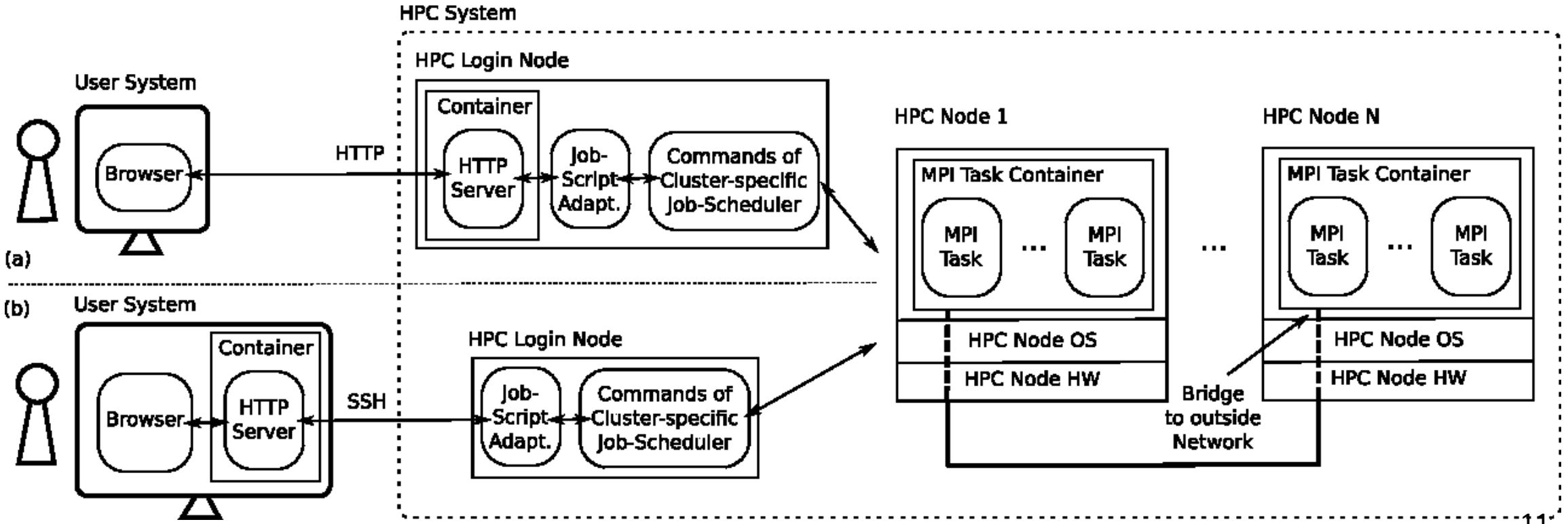


Our Solution

- identical container image for **all** target platforms
- web user interface for
  - input / output
  - progress report

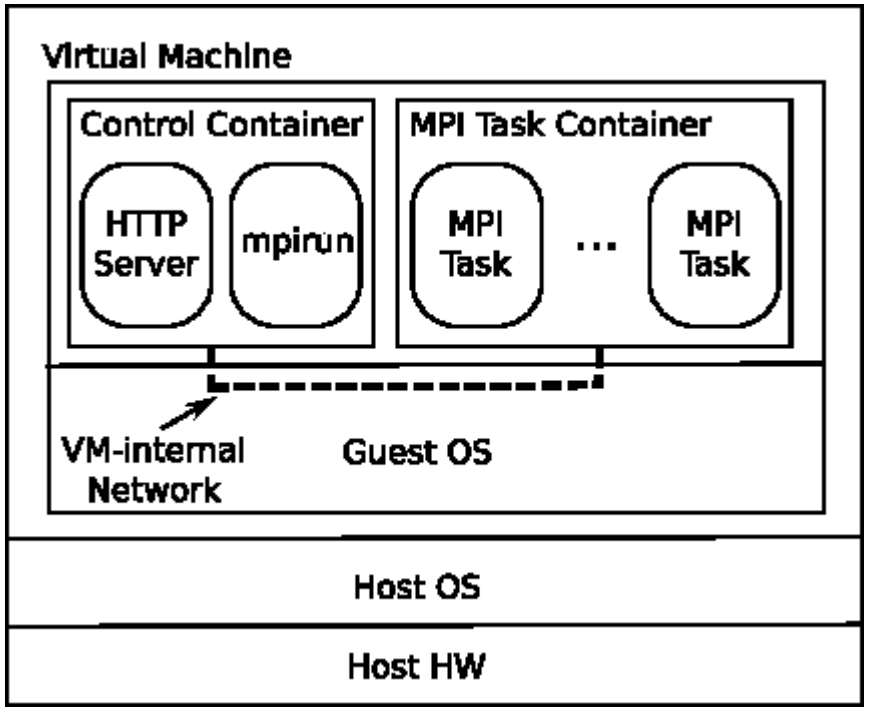
Target Platform: HPC Cluster

- use regular job-scheduling system
- dedicated containers: setup/cleanup part of job script



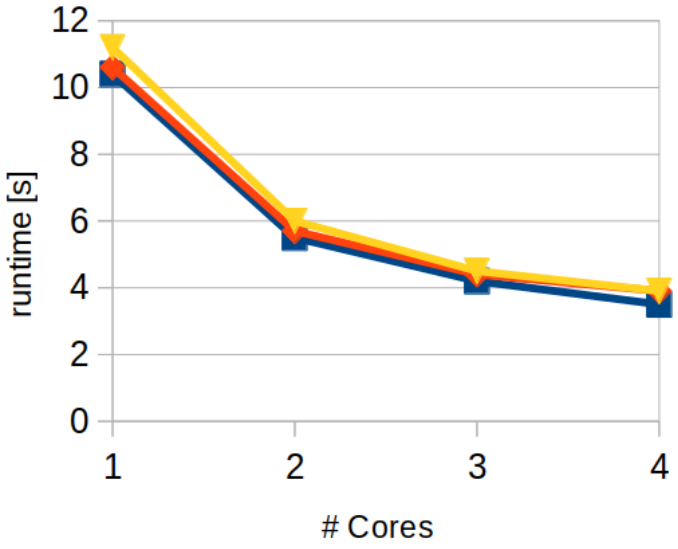
Target Platform: Workstation

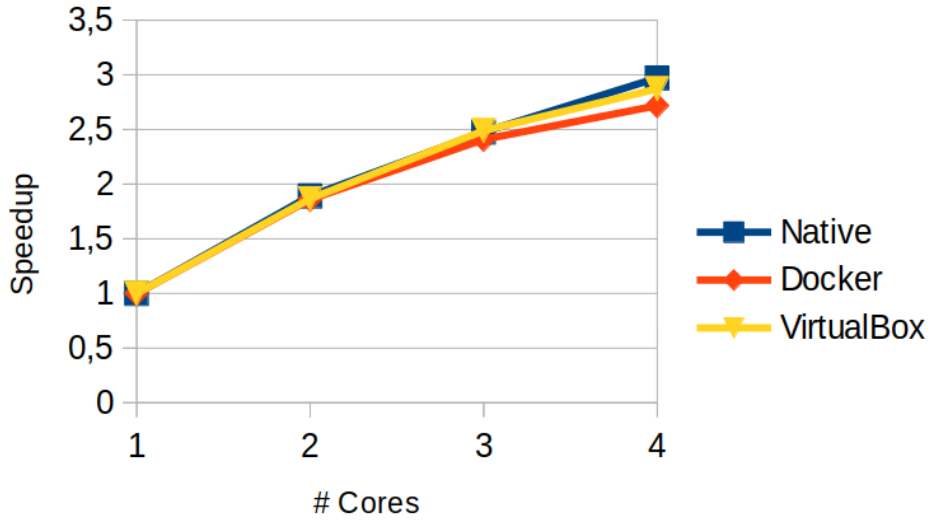
- VM + Container
  - host can be Windows, slowdown acceptable



First Measurements: Native vs. Docker vs. VM

- quad-core Xeon i5-3450 (Ivy Bridge): 3.1GHz, 6MB L3, VTX
- container based on Debian 8, Docker 1.12 (here: multiple containers)
- POLOPT: Intel compiler 17.0.0, Intel MPI 2017
- Small model (~ 1700 unknowns)





Conclusion

Virtualization

- helps reducing costs by allowing single target
- reduces maintenance cost of in-house code
- eases deployment to different target systems
- enables future commercial cloud usage
  (needs risk analysis for sensitive data)

Outlook

- measurements for multi-node