The Wilson HPC Cluster

Dalibor Djukanovic

Institut für Kernphysik
Johannes Gutenberg-Universität, Mainz

Technical Seminar Desy Zeuthen
03.11.2009

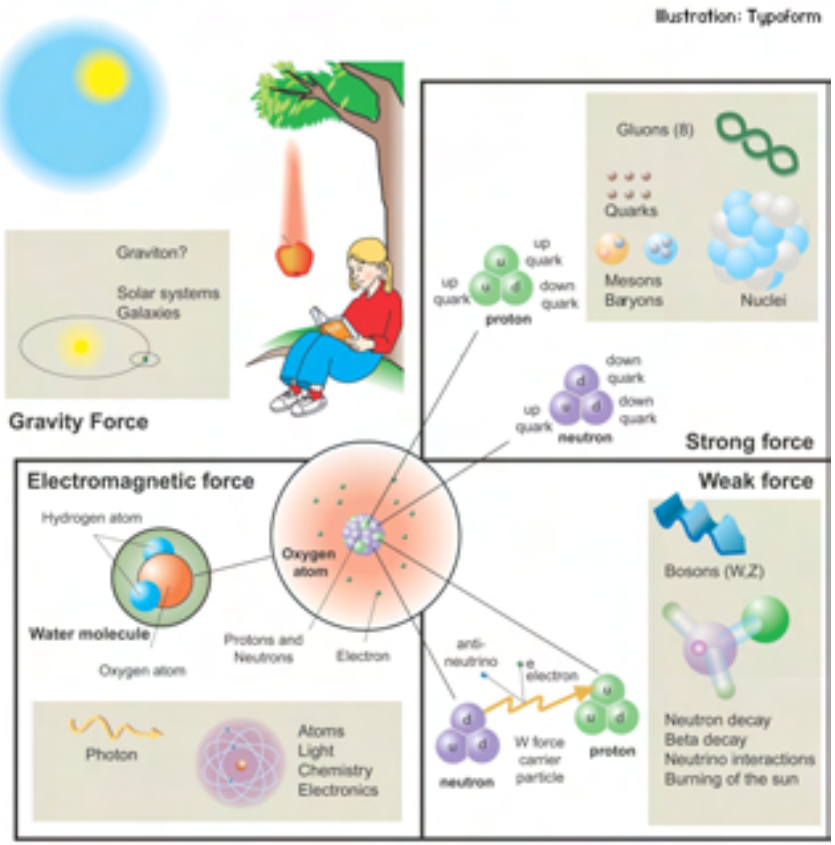
Mainz Group

- Head of group: Prof. Hartmut Wittig
- Members:
  3 postdocs
  5 Ph. D. students
  1 diploma student
- IT department:
  Head of group: Dr. Klaus Werner Krygier
  2 system administrators
  1 techincal staff member
  2 electronics engineers
  1 system admistrator dedicated to Cluster

Outline

Outline

Physics



- 4 fundamental forces
- Strong force is strongest
- Described by Quantum Chromodynamics (QCD)

Quantum Chromodynamics (QCD)

- Nuclear Physics @ low energies

$$\mathcal{L}_{\mathrm{QCD}} = -\frac{1}{4} F^a_{\mu\nu} F^{a\,\mu\nu} + \sum_{f=u,d,s,\ldots} \overline{\Psi}_f \left\{ i\gamma^\mu \left(\partial_\mu - igA^a_\mu T^a \right) - m_f \right\} \Psi_f$$

- Looks simple, but it is hard to solve
- Only way to solve this equation $\Rightarrow$ numerical simulation
- Discretize Spacetime $\Rightarrow$ Put the theory on a lattice
- Lattice QCD $\Rightarrow$ mostly Monte Carlo simulations
  $\Rightarrow$ this is why we need Wilson
- But still what is Wilson?

Wilson

- Kenneth G. Wilson: Nobel prize in physics 1982
- One of the founders of Lattice QCD
- We use Wilson fermions

Our own Wilson

- Dedicated platform for lattice QCD simulations
- Cluster delivered by Hewlett-Packard
- Installation October 2008 $\rightarrow$ Total cost: 1.098.000 €

Pictures

We bascially started from scratch

- Preparation of the server room
- Installation of a proper cooling
- HP provided ready to go Installation

Pictures

Pictures

Air Condition

First time a theorist needed a Riga crane!

Pictures

Pictures

Sometimes it was close:

Finally

Outline

Originally we had

- 250 DL 160 G5 Servers:
  Intel Xeon E5462 @ 2.8 GHz
- Problems with stability:
  Nodes crashed with kernel panic
  No obvious reason
  Not specific to node
  Uncorrelated to runtime
- Finally the complete System was replaced
  $\Rightarrow$ some parts are olden than other:
  Power supplies
  IB HCA

Now we have

Wilson & Lilly

- We actually have 2 Clusters:
  - Wilson (the big one):
    280 dual socket quad core compute nodes
    Infiniband interconnect
  - Lilly (Little Willy):
    22 dual socket quad core compute nodes
    Infiniband interconnect

This talk . . .

I will concentrate on Wilson, since Lilly is really just the little sister.

Masternode

- Hewlett-Packard Proliant DL 385 G5 node:
  - 2 U rack mount server
  - 2 AMD 2356 QuadCore processors @ 2.3 GHz
  - Memory: 16 GB
  - 500 GB local SAS storage
  - 4 MSA 60 storagesystems (12 disks each):
    2×500 GB disks @ Raid $5 + 1$ hot spare
    2×1 TB disks @ Raid $6 + 1$ hot spare
  - Red Hat Enterprise Linux 5.1

Used for

- NFS server
- Management & Monitoring
- Login node, job submission

Compute Nodes

Wilson = 280 compute nodes
Lilly = 22 compute nodes

- Hewlett-Packard Proliant DL 165 G5 nodes:
  - 1 U rack mount server
  - 2 AMD 2356 QuadCore processors @ 2.3 GHz
    $\Rightarrow$ 2416 cores
  - Memory: 1 GB/Core
    $\Rightarrow$ 2.42 TB total
  - Infiniband HCA: Mellanox ConnectX MT25418
    4x DDR $\rightarrow$ 20 Gb/s
  - 2 Hotswap 147 GB hard disk with Raid 1
    $\Rightarrow$ 44.4 TB local storage
  - Red Hat Enterprise Linux 5.1

Cooling

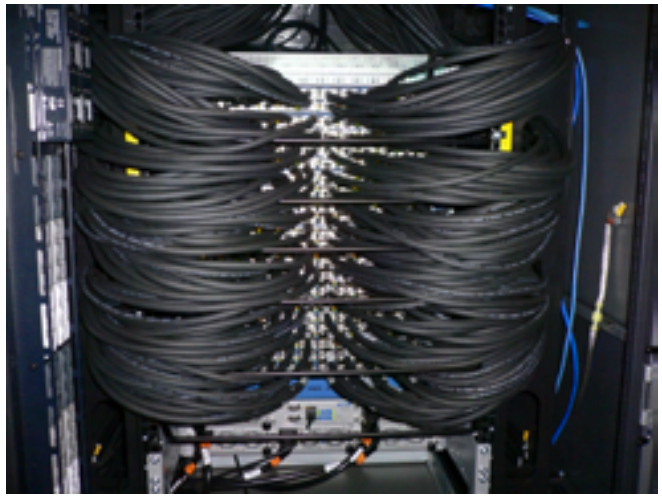
Cooling Capacity per Rack

Each rack hosts 40 compute nodes $\approx$ 12 kW/rack

Problematic using conventional air cooled racks

- 7 water cooled racks:
  HP MCS (Modular Cooling System) G1
- Maximum Cooling Capacity 35 kW
- Doors can be opened for maintenance
- Doors open automatically in case of cooling failure

- Infiniband:
  - The Switch:
    Voltaire ISR 2012 288 Port IB Switch
    Voltaire ISR9024D-8032 24 Port IB Switch



- Ethernet:
  - 2 seperate Networks:
    1 GBit Network: login, storage connection ...
    1 100 MBit Network: serial console, virtual kvm, ...
  - Each rack has one switch for both networks
  - Bonded Uplink to master node

Summary of the system

- 1 air cooled rack
- 36 TB gross storage
- 288 port IB switch
- 7 water cooled racks
- 280 compute nodes

Experience with the hardware

Most problematic components:

- Hard disks in compute nodes (exchange 1 hd every 5 weeks)
- RAM: Usually a pile up of correctable ECC errors, but also uncorrectable ECC's
- System fans: Non redundant, i.e. if a fan is reported as missing system shuts down
  This happens frequently even though the fan is ok

In general

Hardware is robust and downtime due to hardware errors on the compute node side is negligible.

Software

- Compilers:
  Openmpi, Mvapich, Mvapich2, GNU Compilers
- Almost all programs use Message Passing Interface (MPI)
- Batchsystem:
  Wilson: "Nearest neighbour shouting"
  Lilly: Torque, Maui

Big benefit of the machine

No restrictions to runtime and resources!

Outline

Cluster Management Utility (CMU) - Java program providing:

- Quick overview of Cluster Status
- Sensorinformation, e.g. cpu load, memory used, ...
- Several management tasks:
  - Power on/off compute nodes
  - Open serial console
  - Backup and clone nodes
  - Single and multiple window parallel shell
  - Simple alerting

CMU


CMU V4.1 – Running
File
Monitoring
Cluster Configuration
Options
Help
Network Entity
CMU Cluster
Lilly
Rack1
Rack2
Rack3
Rack4
Rack5
Rack6
Rack7
global cluster view
Pies
cpuload
%
26562
av 88
min
0
max
10502
Part. state summary
Node state:
Normal
Unknown
Warning
Critical

OS Installation

Installation of the OS on compute nodes:

- CMU provides cloning mechanism
- Install one node by hand and backup hard disk ⇒ golden image
- Distribute the golden image over the cluster ⇒ cloning
- Installation Steps:
  - Network boot the nodes to be cloned (1 master serves 1 slave)
  - Copy the golden image
  - Perform node specific configuration, e.g. IP-address
- Each cloned node serves as a master for another node
  ⇒ Cascading installation
- Masters and slave stay within a given network entity
  ⇒ reduction of network traffic

OS Installation

Installation of the OS on compute nodes:

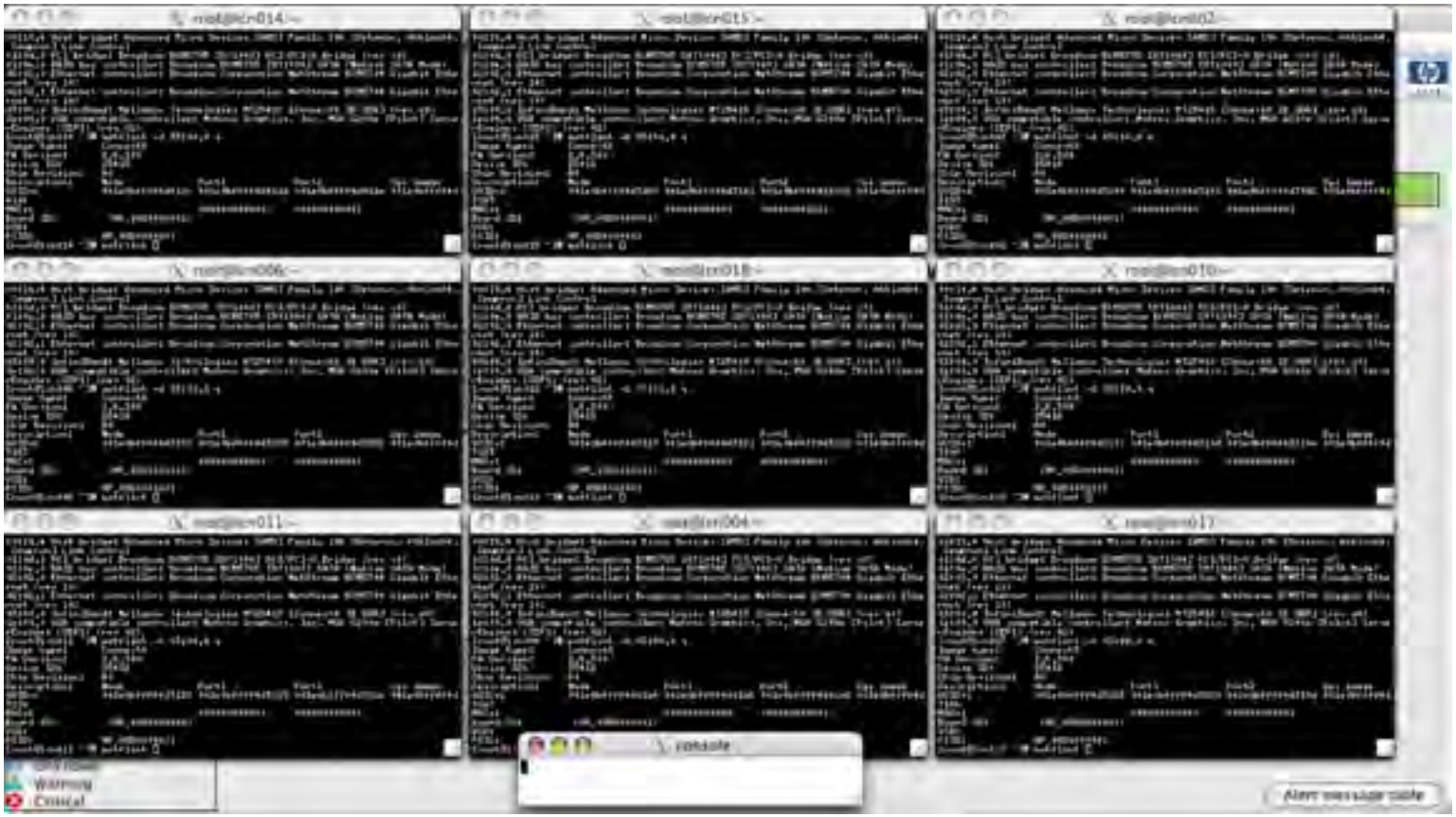
- CMU provides cloning mechanism
- Install one node by hand and backup hard disk ⇒ golden image
- Distribute the golden image over the cluster ⇒ cloning
- Installation Steps:
  - Network boot the nodes to be cloned (1 master serves 1 slave)
  - Copy the golden image
  - Perform node specific configuration, e.g. IP-address
- Each cloned node serves as a master for another node ⇒ Cascading installation

Complete Cluster

Installation takes 1 hour with a 3 GB image.

Multiple Window parallel shell

Useful for interactive multiple shell access, e.g. firmware update of IB HCA



What do we use?

Intelligent Platform Management Interface (IPMI)

- Remote control of servers:
  Shutdown
  power off/on
  boot order
- Collection of out of band information:
  CPU temperatures
  Fan speeds

Simple Network Management Protocol (SNMP)

Challenge

We have 280 nodes
$\Rightarrow$ scalable solution needed

For monitoring we have

- Purchased hardware ⇒ usually Simple Network Management Protocol (SNMP), with vendor specific MIB
- Tailor made hardware ⇒ custom made readout in hardware and software

Purchased components giving sensor information via SNMP

- Water cooled racks
- UPS
- Switches
- Cooling system, i.e. chillers

Self-made

- Circuit breaker: Emergency power off, remote control via AVR micro controler
- Temperature and humidity sensors: attached to embedded systems, via serial to usb

Hardware for monitoring

Master node provides

- Centralized syslog server: syslog-ng
- SNMP-trap receiver

Additionally we have

- 2 virtual machines :
  - One machine runs conserver, archiving the console output of all machines
  - One machine constantly checks the air intake temperature of water cooled racks.
  - Initiate shutdown of cluster based on thresholds

What do we do with the information

We collect a lot of information and one quickly looses the overview!

Monitoring and Eventhandling

We want

- Framework to easily collect and archive data in form of rrd databases
- Graphical representation of data and other information via web interface
- Eventhandler for several user defined thresholds

We use

- Nagios
- Ganglia
- Cacti

Nagios

- Nagios is an extensible monitoring framework
- Nagios performs scheduled checks of all defined hosts and services, reporting the current state (Missing, Ok, Warning, Critical)
- Eventhandlers based on state change or actual state
- Nice escalation feature for notifications
- Another nice thing: Flapping detection

Example Lilly

Nagios checks rack temperatures of the Lilly Rack:
Temperature at least 2 consecutive times $> T_{\text{crit}}$ shutdown.

Ganglia

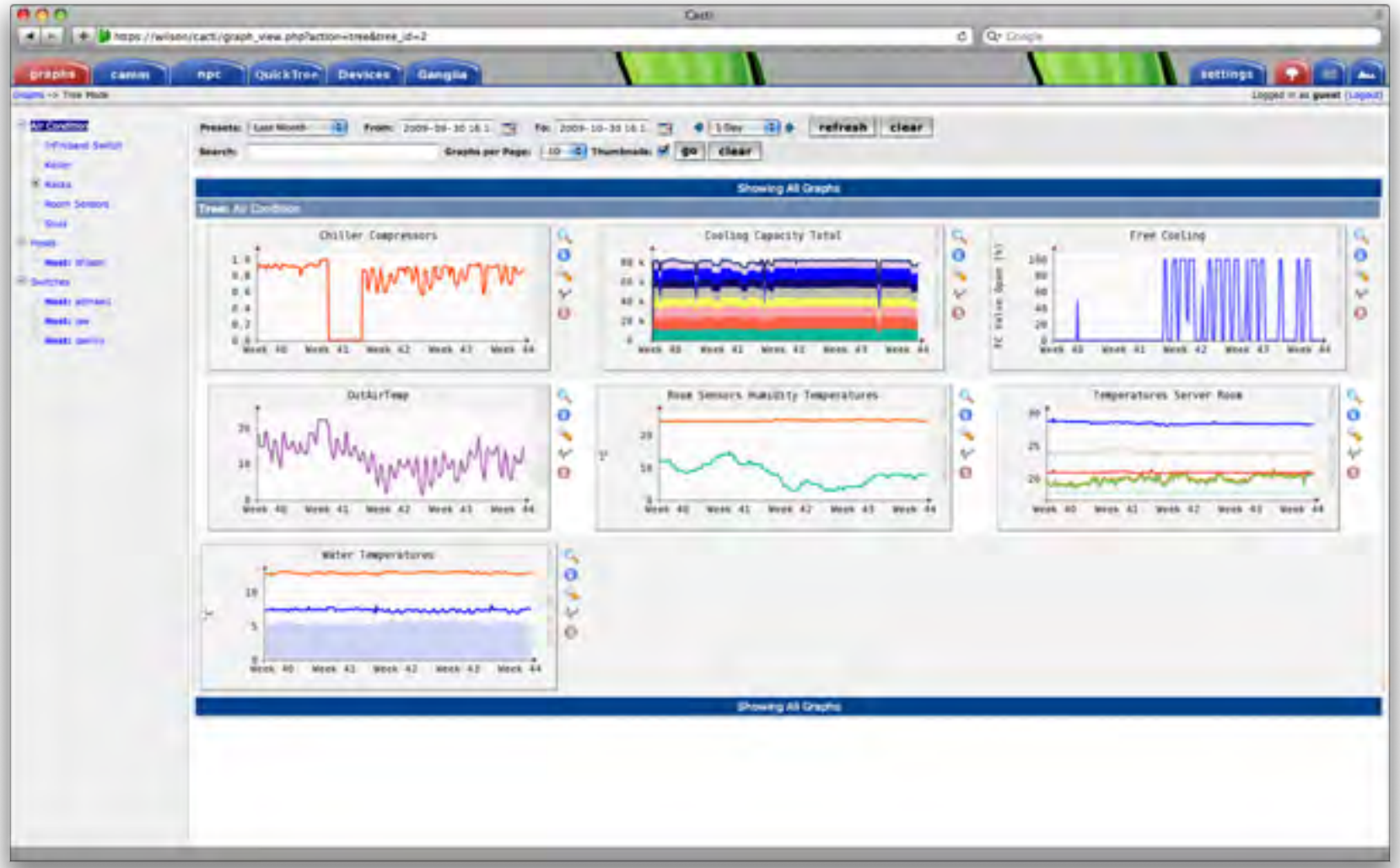
- Ganglia is a distributed monitoring system
- Daemon based datagatherer (gmond)
- Server collecting all information related to a cluster in rrd databases (gmetad)
- Nice feature:
  Spoofing mechanism $\Rightarrow$ inject data to the datastream of a certain node
  We collect out of band information using the spoofing mechanism

Cacti

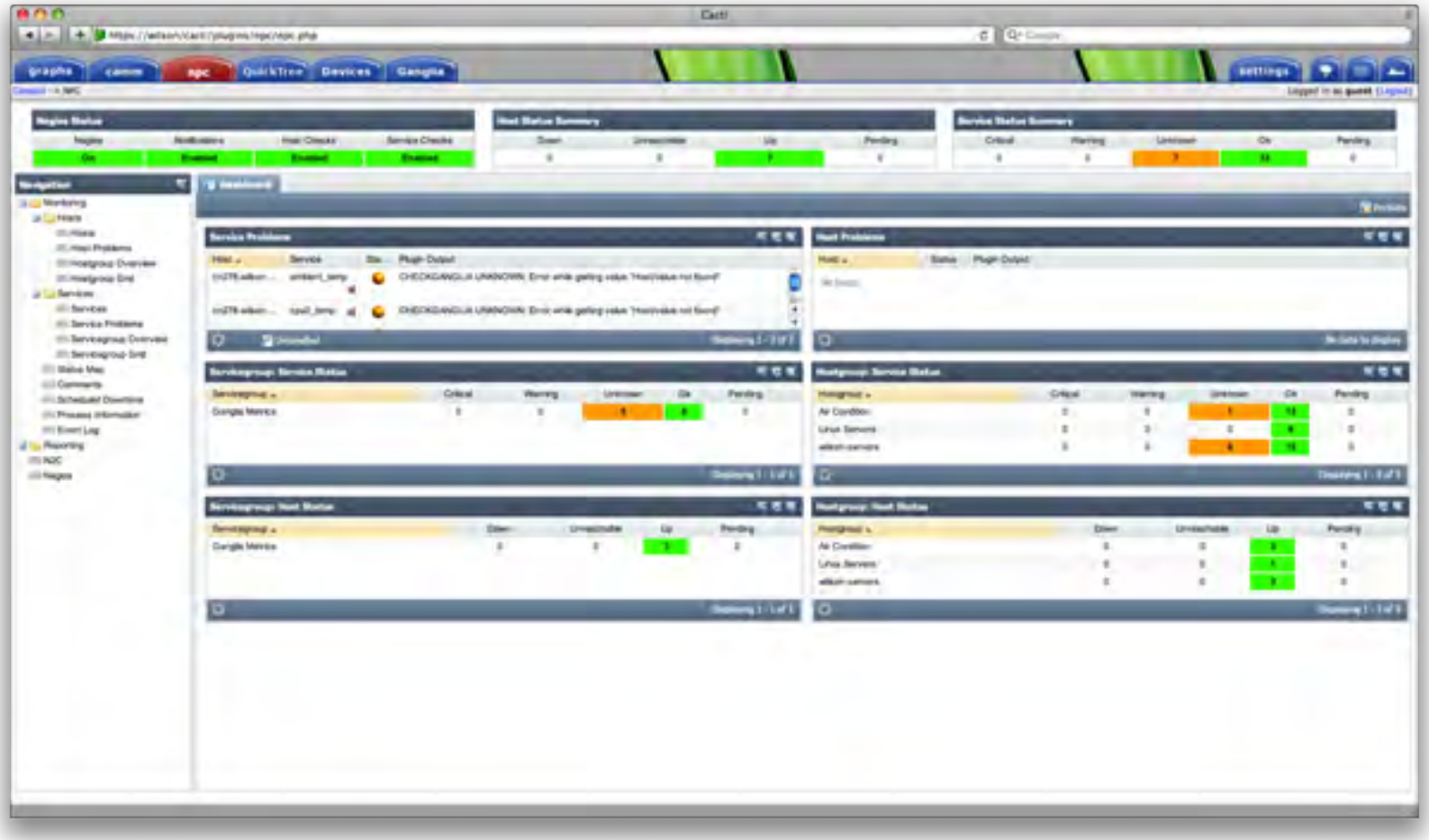
Webinterface for visualization of sensor information using rrd tools by Tobi Oettiker

- Easy to install and configure
- Creation of nice looking rrd graphs with a few mouse clicks
- Templates simplify graph creation for identical hardware
- Nice UI, e.g. zoomable graphs, timespan, ...
- Access control, using realms and per graph permissions
- Plugin architecture for extensions
- Consolidate interface for different packages, e.g. ganglia and nagios

Cacti Sensor Information

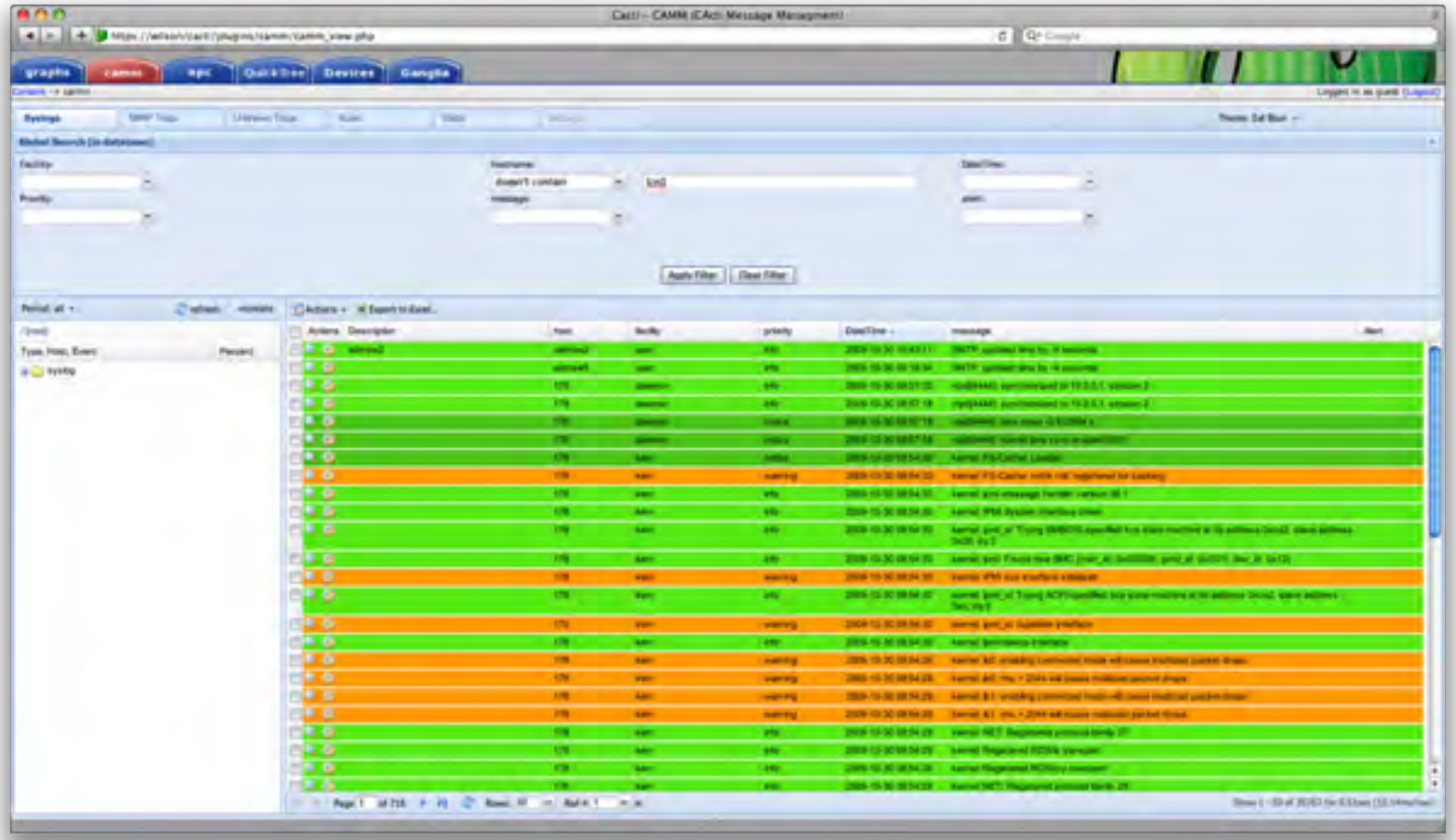


Cacti Nagios Interface



Cacti Ganglia

Cacti Syslog & SNMP



In Case of Emergency

We have a three way shutdown procedure

- Software tries graceful shutdown
- Software initiates hard power off
- Circuit breaker coupled to thermostat:
  $\Rightarrow$ physically disconnect power from rack

Last Line of Defense

Red Button switching off power

Outline

With kind permission by Prof. Hartmut Wittig

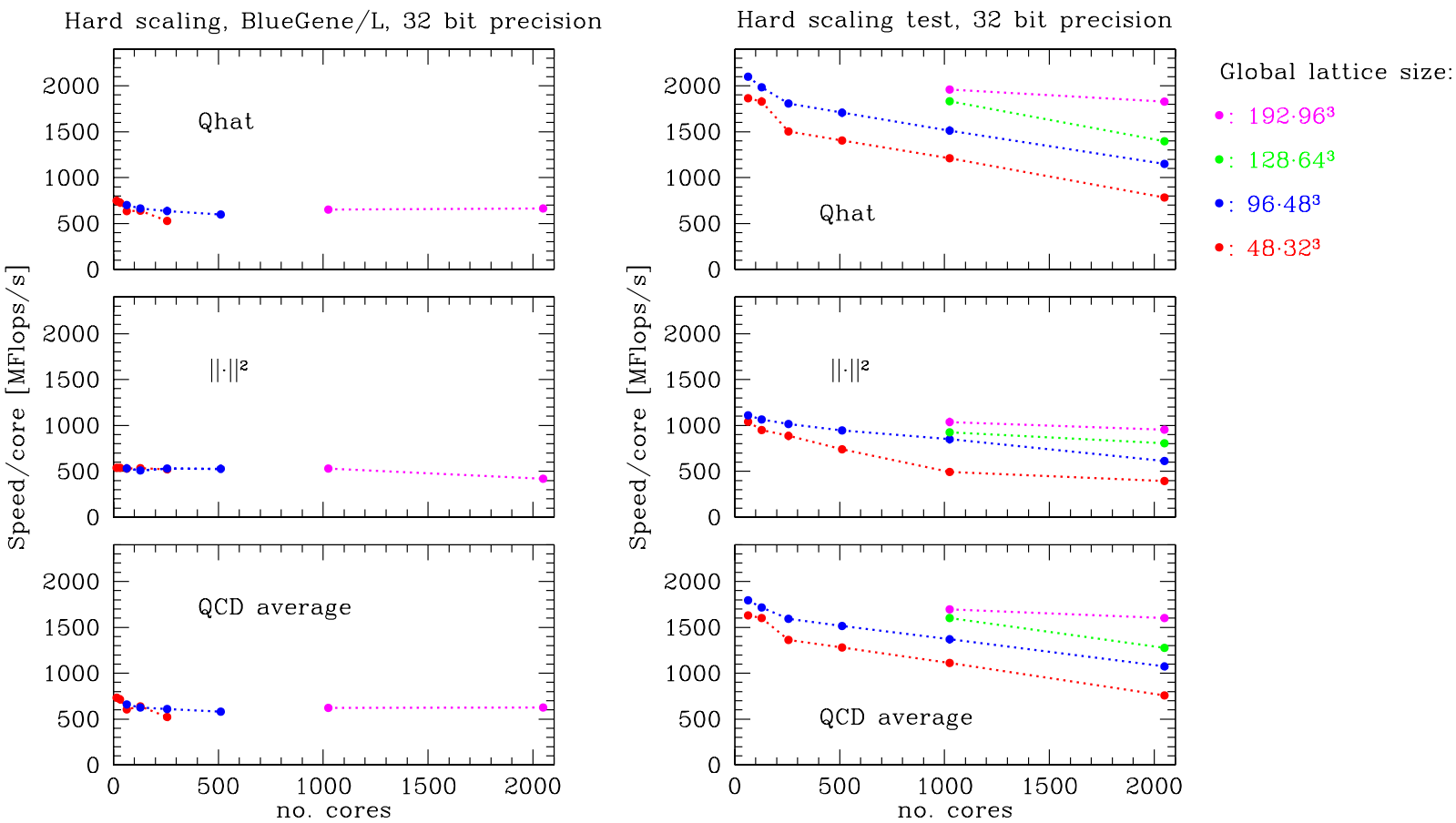
Some preliminary benchmarks

- Benchmark based on typical lattice QCD operations:
  Application of Wilson Dirac Operator: $\widehat{Q}$
  Norm squared of a vector: $||\cdot||$
  Linear combination: $z = ax + y$
- Weighted average gives synthetic QCD average performance
- Standard C Code (MPI)
- Optimization: SS3, prefetch
- Compare Hard scaling AMD 2356 with:
  BlueGene/L and BlueGene/P

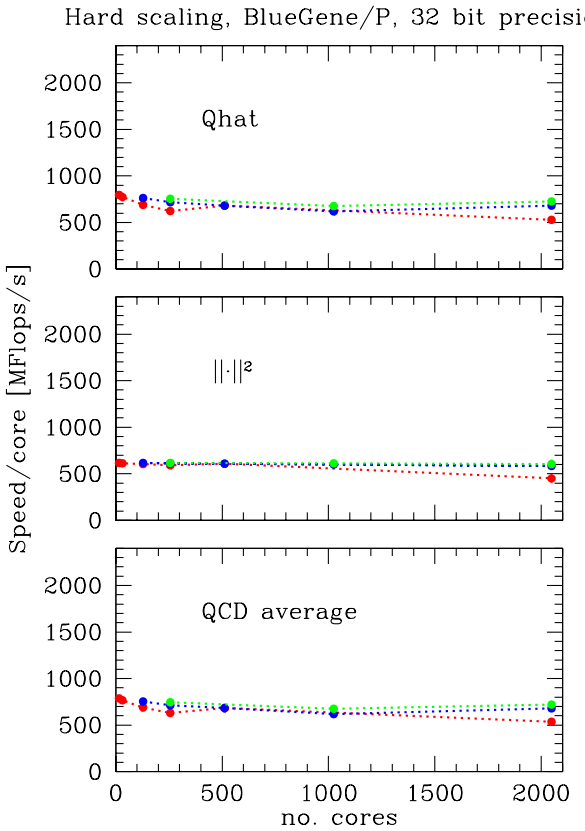
Hard scaling

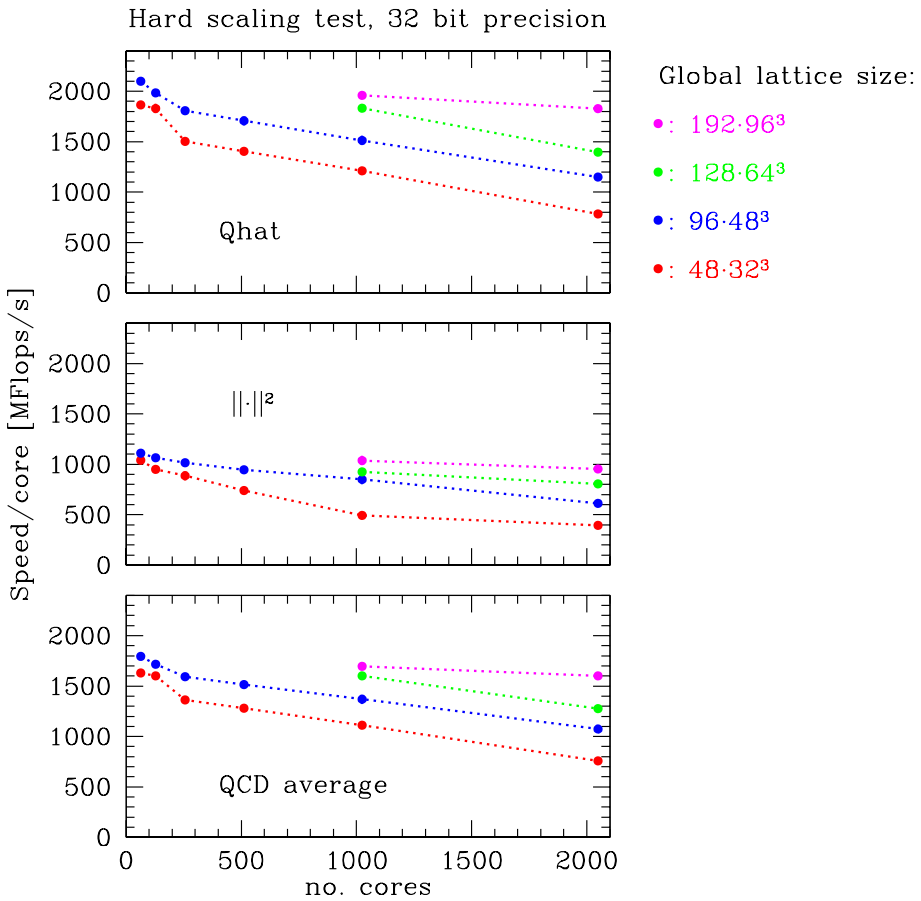
- Increase the number of CPU's for a constant global problem size:
  $\Rightarrow$ local problem size is decreasing with number of cpu's
- Two effects:
  Problem might fit into cache
  Surface/Volume ratio increases
  $\Rightarrow$ Communication becomes important
- We have results for 4 Lattices:
  192×96^3
  128×64^3
  96×48^3
  48×32^3

Preliminary Hard Scaling: AMD vs BlueGene/L

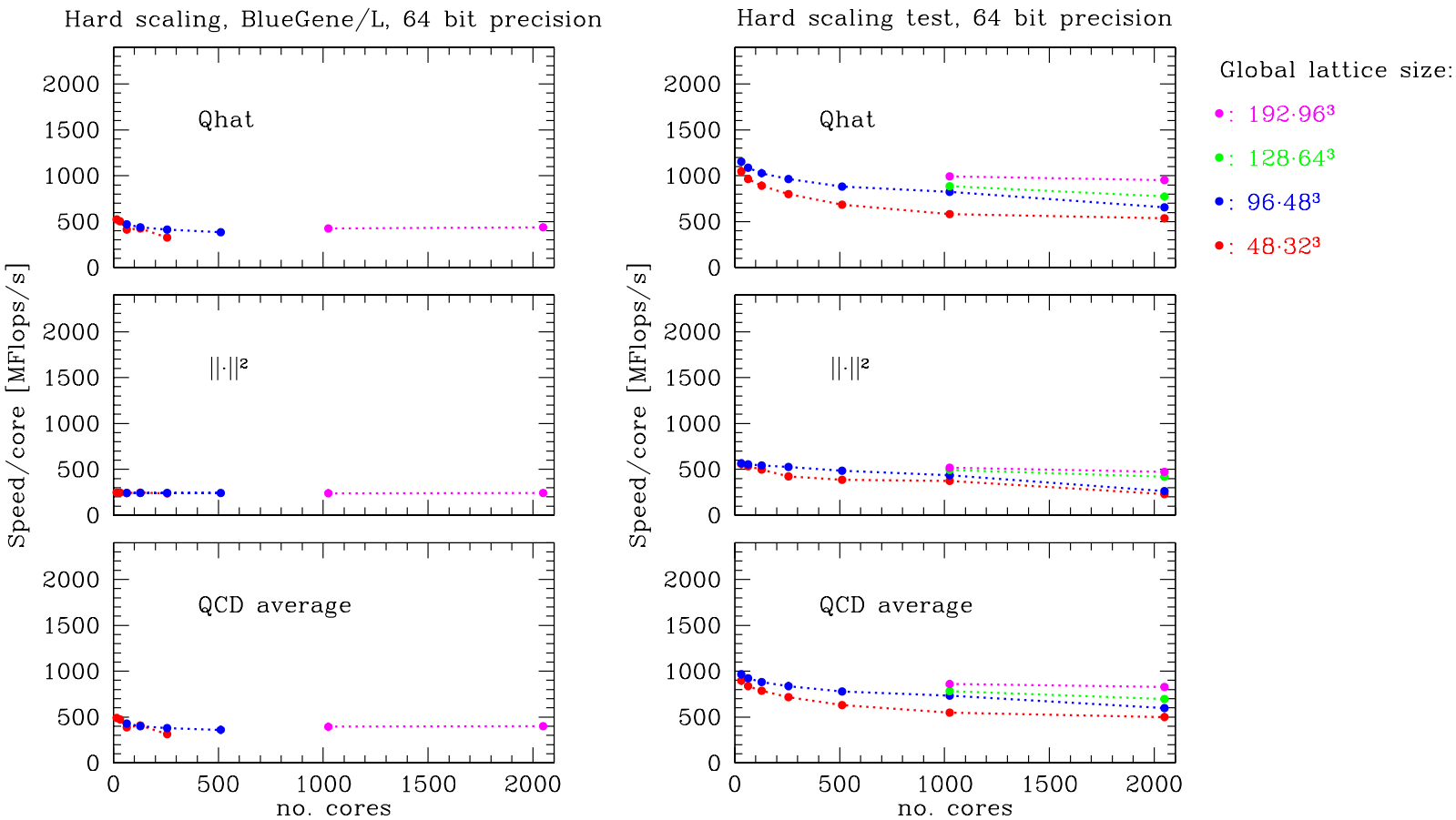


Preliminary Hard Scaling: AMD vs BlueGene/P

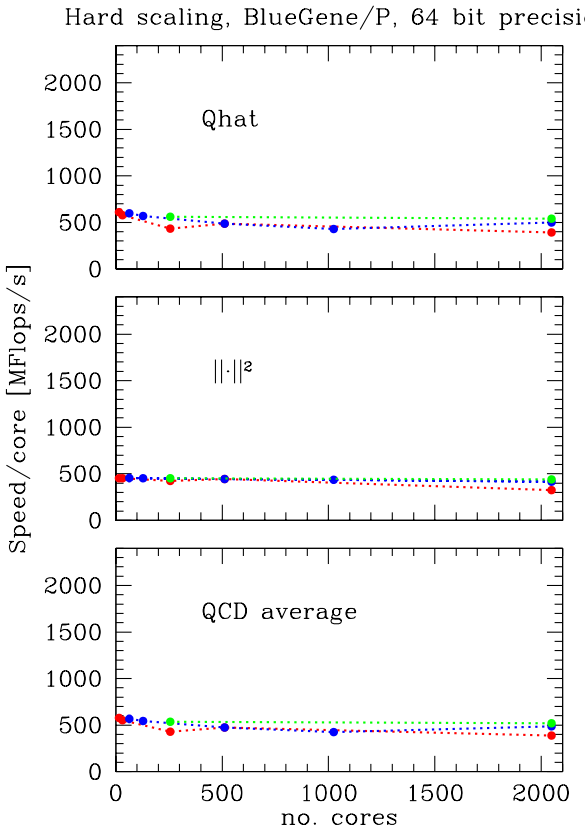


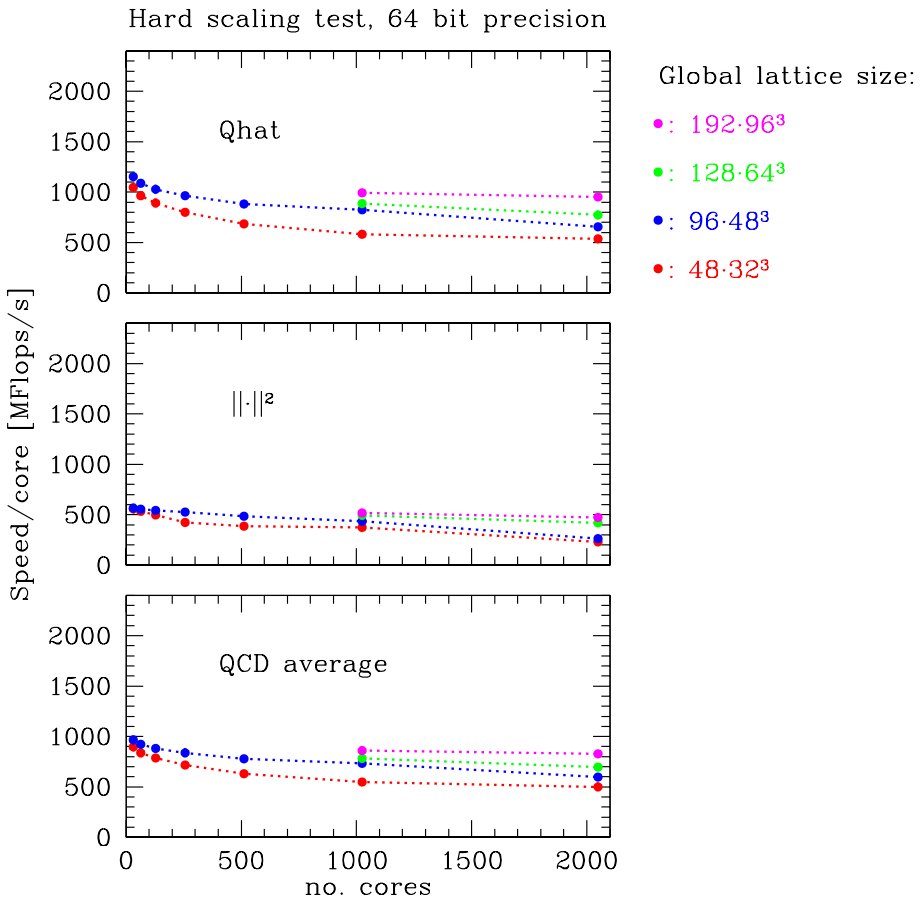


Preliminary Hard Scaling: AMD vs BlueGene/L



Preliminary Hard Scaling: AMD vs BlueGene/P

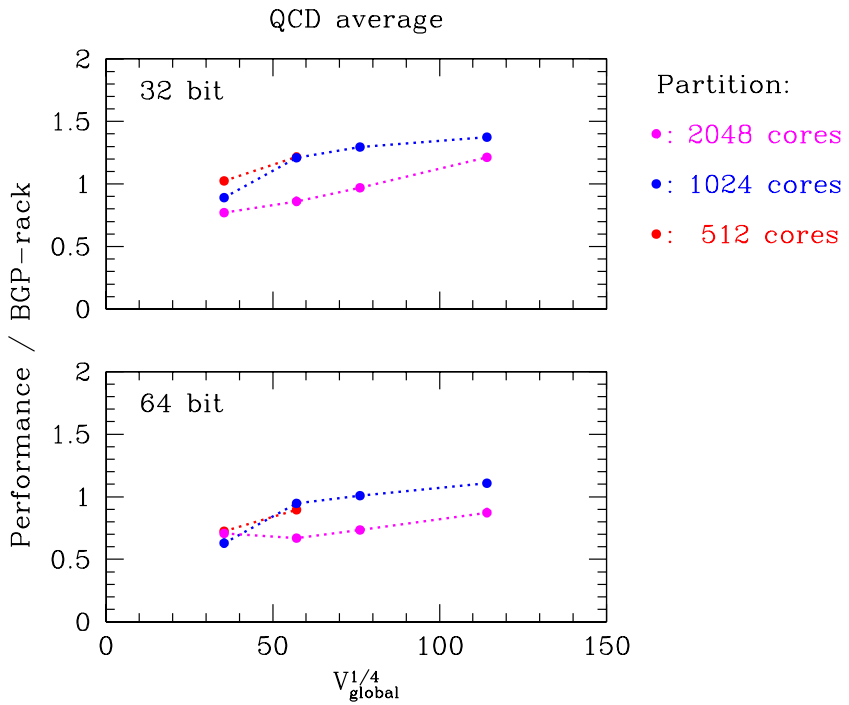




AMD vs BlueGene

Clusterperformance in units of BlueGene/P racks:

$$\text{Performance} = \frac{P_{\text{AMDcore}}\left[\text{MFlops}\right] \times 2240}{P_{\text{BGPcore}}\left[\text{MFlops}\right] \times 4096}$$



Outline

Summary

- Stable system with few downtimes
- 3.5 TFlops/User (Linpack)
- 2240 core Cluster running Lattice QCD code @ 3.7 TFlops sustained
  $\Rightarrow$ 0.3 €/MFlops
  $\Rightarrow$ Waste heat/cooling: 23 kW/TFlops
- Hard scaling shows up to 50 % drop (flatter curve for BlueGene)
  Need to compare with Nehalem system