BLUE WATERS

SUSTAINED PETASCALE COMPUTING

September 23, 2011

Challenges and Opportunities for Exscale Resource Management and How Today's Petascale Systems are Guiding the Way

Dr. William Kramer
Blue Waters Deputy Director
NCSA

NCSA

GREAT LAKES CONSORTIUM
FOR PETASCALE COMPUTATION

Scheduling is a form of decision making …[it] is concerned with the allocation of scarce resource activities with the objective of optimizing one or more performance measures.

The Handbook of Scheduling – Joseph Y-T. Leung

Characteristics of Peta+scale Systems

- Increasing Heterogeneity of resources
  - Compute units
  - Interconnect pathways
  - Data preservation hierarchies
- Diminishing ratio of bandwidth to capacity
  - Diminishing ratio of bandwidth to operation count
- Increasing gulf between peak and sustained performance
- Increasing concern on cost of ownership
- Decreasing application developer productivity

What is Different About Peta+Exa-scale?

- Bandwidth is the critical resource past Petascale
  - Implications
    - Topology is the key resource related to efficiency and performance
    - All data transfers are expensive
- Errors are everywhere in a Peta+scale system
  - Implications
    - All layers have to deal with errors well and accurately
    - Resiliency is everyone's responsibility
    - Roles and Information is critical to all layers making decisions
- Imbalance is everywhere in a peta+scale system
  - Implications
    - Multi-dimensional optimizations
      - That change occurs and has to be dealt with during application execution
- Peta-scale ➔ Exascale
  - Scaling applications to large core counts.
  - Effectively using multi-core and many cores
  - Using heterogeeous nodes in a single simulation
  - Systems and applications share many responsibilities

Common Wisdom

Pre-Petascale

- CPU cycles are the critical resource
- The primary role of a resource manager is starting jobs
- Once a job starts, the resource manager's work is done
- All the information a job scheduler needs is known at start-up
- All resources a job wants are homogeneous
- Checkpoint/Restart is sufficient resiliency
- Errors information is approximate
- You cannot compare resource manager effectiveness in an objective way

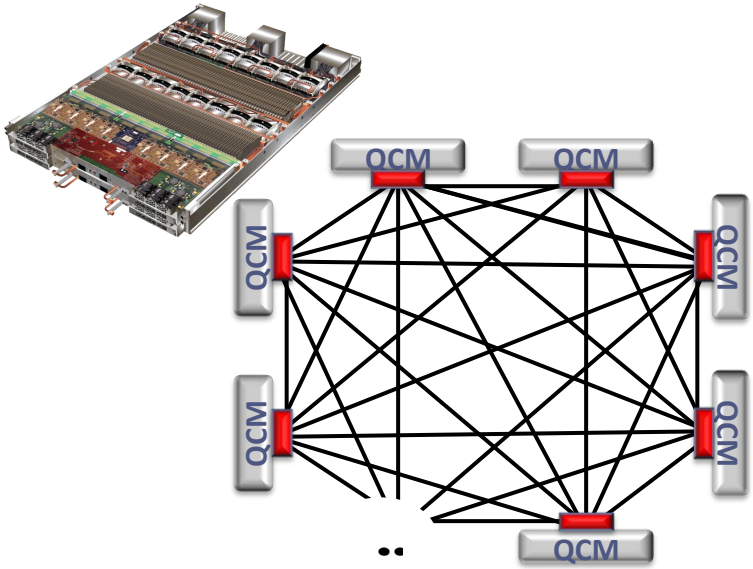
Petascale and Beyond

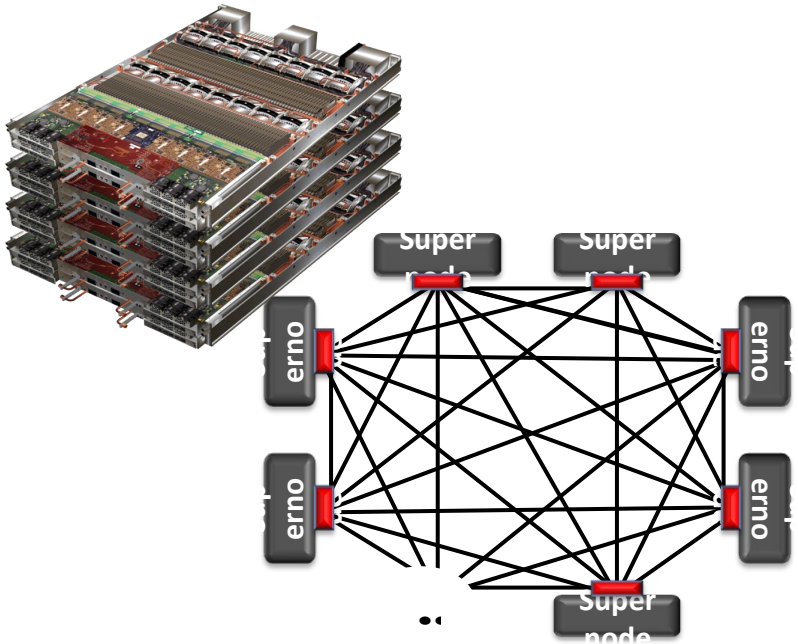
- Bandwidth is the critical resource
- The role of a resource manager is managing job's resource requirements
- The resource manager's work only starts when a job begins processing
- The information a resource manager needs is constantly changing
- Resources a job needs are constantly changing
- Resiliency is an application's responsibility with system's assistance
- Errors information must be accurate
- You can compare resource manager effectiveness and performance

TOPOLOGY AND BALANCE ISSUES

Managing the scarcest resource

Two-level (L, D) Direct-connect Network



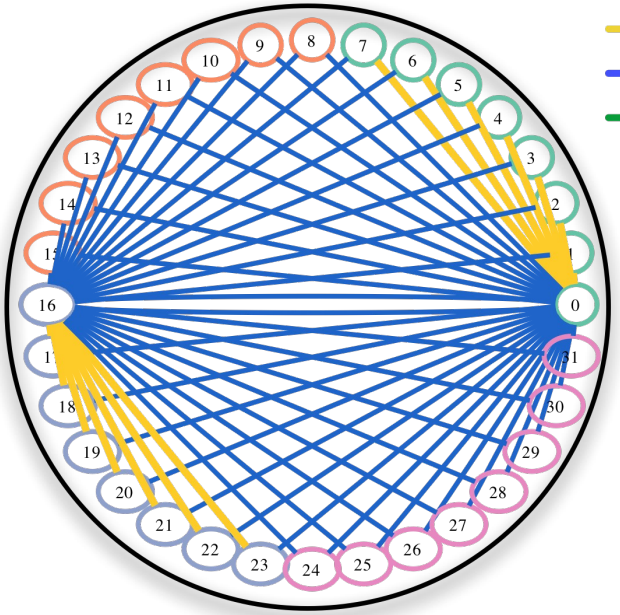


Each Supernode = 32 QCMs
(4 Drawers x 8 SMPs/Drawer)

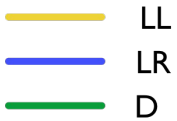
Fully Interconnected with L_{local} and L_{remote} Links

Fully Interconnected with D Links

P775 Two Level – Direct Connect Topology



One supernode in the PERCS topology



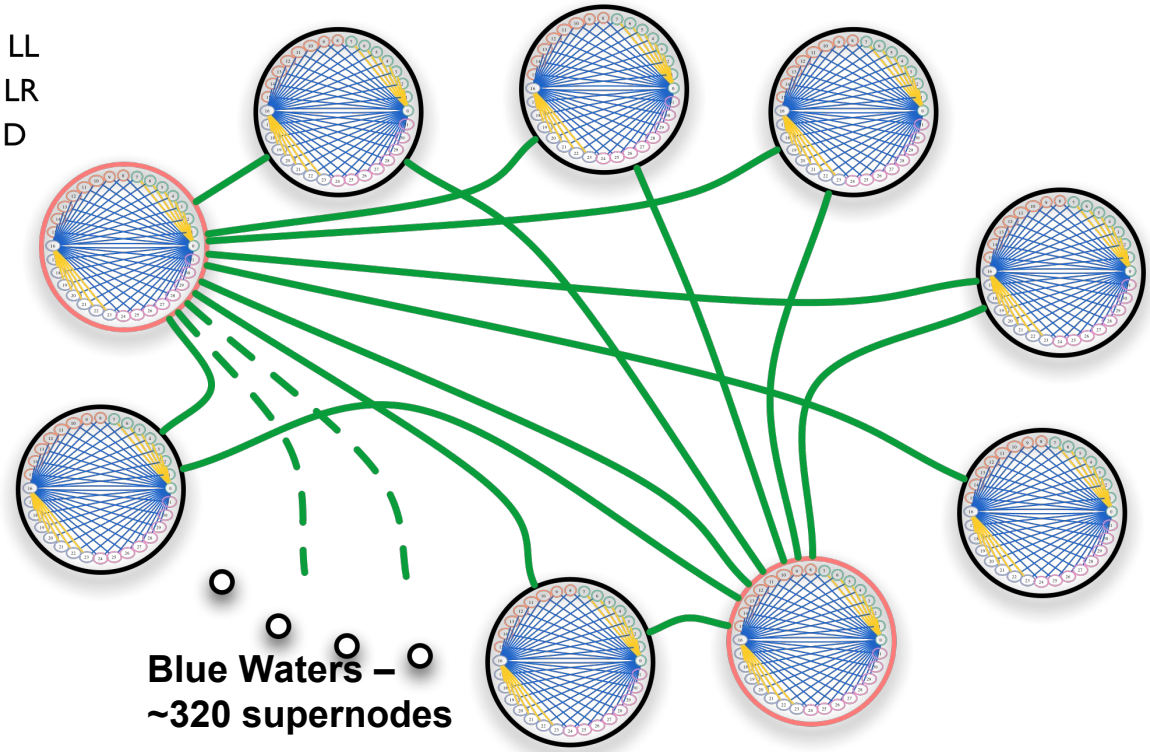


Image courtesy of Abhinav Bhatele-UIUC

Bandwidth Issues

- It may seem that the correct scheduling decisions are to pack the jobs into the drawer, and then use all the drawers in a supernode.
- The is half correct.
  - Most constrained bandwidth is between drawers within a supernode
  - So the best topology could be pack the drawers and then randomly place the job across drawers
  - Avoids bottlenecks within the super node
- Other topologies are not uniform in links or in hierarchy as well

Shared Decision Making

- Neither the Application nor the Resource Manager have complete knowledge
- Resource Managers should adjust for topology benefits and for differential node capability
  - Make the best initial choices
  - Be able to adjust running work so as to effeciently create better topologies
- Applications can now adjust their work layout to a given mapping onto a topology and node capability
  - Charm++, newer “adaptive” MPIs,
  - Zoltan, UNITAH, Paramesh and Chombo

Peta-scale Limitations for Shared Decision Making

- State of the Art today is to give a list of nodes
  - Applications have to figure out how the nodes map to a topology
  - Resource managers may not have those exact nodes available
- Few to no ways for applications to express its topological needs and desires to the resource manager
- Few to no ways for a resource manager to offer the application choices
- For the most part, we are losing our ability to adjust the topologies of running work
- Crude mapping methods for heterogeneous nodes of different capabilities
  - multi-physics applications provide a natural decomposition
  - applications such as NAMD handle heterogeneity since the work is already decomposed into smaller chunks that the runtime can allocate to different processors as needed
  - Some MPI applications may be able to leverage adaptive MPI (AMPI) to handle application-induced load imbalance,

RESILENCY AND EFFICIENCY

Helping Applications be Resilient and Efficient

- Systems will have errors
  - a lot of them but widely separated
- Complete state duplication is not sustainable at Peta+scale to Exascale
- Error information – especially coming through resource managers to applications is prone to wide variations of accuracy
- There needs to be better ways to express urgency and to negotiate tradeoffs
  - Energy efficiency decisions may or may not be local optimization
  - Slowing down one application to save energy may have a cascading effect on many other applications
    - Just like traffic bottlenecks from slow cars

Helping Applications be Resilient

- Applications runs times are able to reallocate work
  - Such applications need to be given precise information
- Then resource management has to provide fine grained assistance to substitute resources with just in time deliver
- There is a need for protocols for applications and resource managers to negotiate the best solutions
  - System: “Your node x just broke”
  - App: “Thanks – I say it was being very slow to respond”
  - App: “Can I have another node to replace it”
  - System: “Yes, but not for 50 minutes”
  - App: “Thanks – but it will take me less time to rebalance my work so we can skip it”

Helping Applications be Resilient

- A different conversation
  - App: “Hey – it looks like my node Y is not responding”
  - System: “ Ok – I will go check it. I can give you another one in 5 minutes”
  - App: “That sounds cool, but can you make 2 other ones so I can make up time”
  - System: “Yes, but not for 7 minutes”
  - App: “Thanks. Could you also adjust my time limit by 10 minutes?”
  - System: “I have something waiting but can give you 10 this time.”
- Now – extend to N-way negotiations

OBJECTIVE MEASURES OF RESOURCE MANAGEMENT

Measuring Scheduling Effectiveness

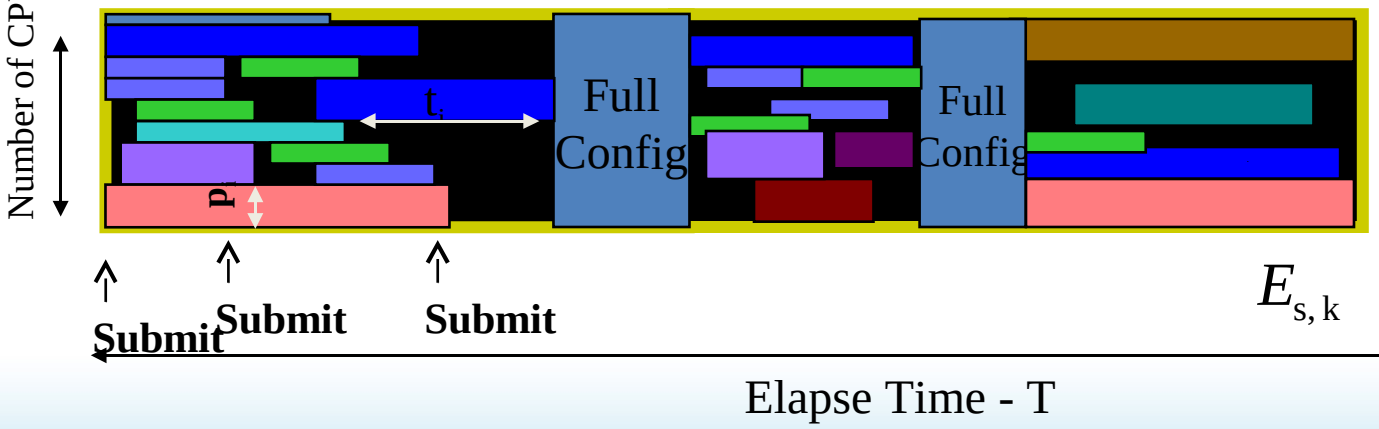
- Giga and Terascale - the norm was throughput tests
  - How fast will a predefined workload run
  - Stress test + scheduling
  - Most had a predefined order
- Throughput tests do not address required features of large scale systems
  - Operational priority changes
  - Scheduler working on limited data
  - Non-portable and inflexible

Tera to Peta-Scale Effectiveness Tests

- The ESP measures
  - Both how much and how often the system can do scientific work
  - How well do systems get the right job to run at the right time
    - Needed for a Service Oriented Infrastructure
  - How easy can the system be managed
- Independent of hardware and compiler optimization improvements
- Limits
  - Created with a homogeneous, reliable system in mind

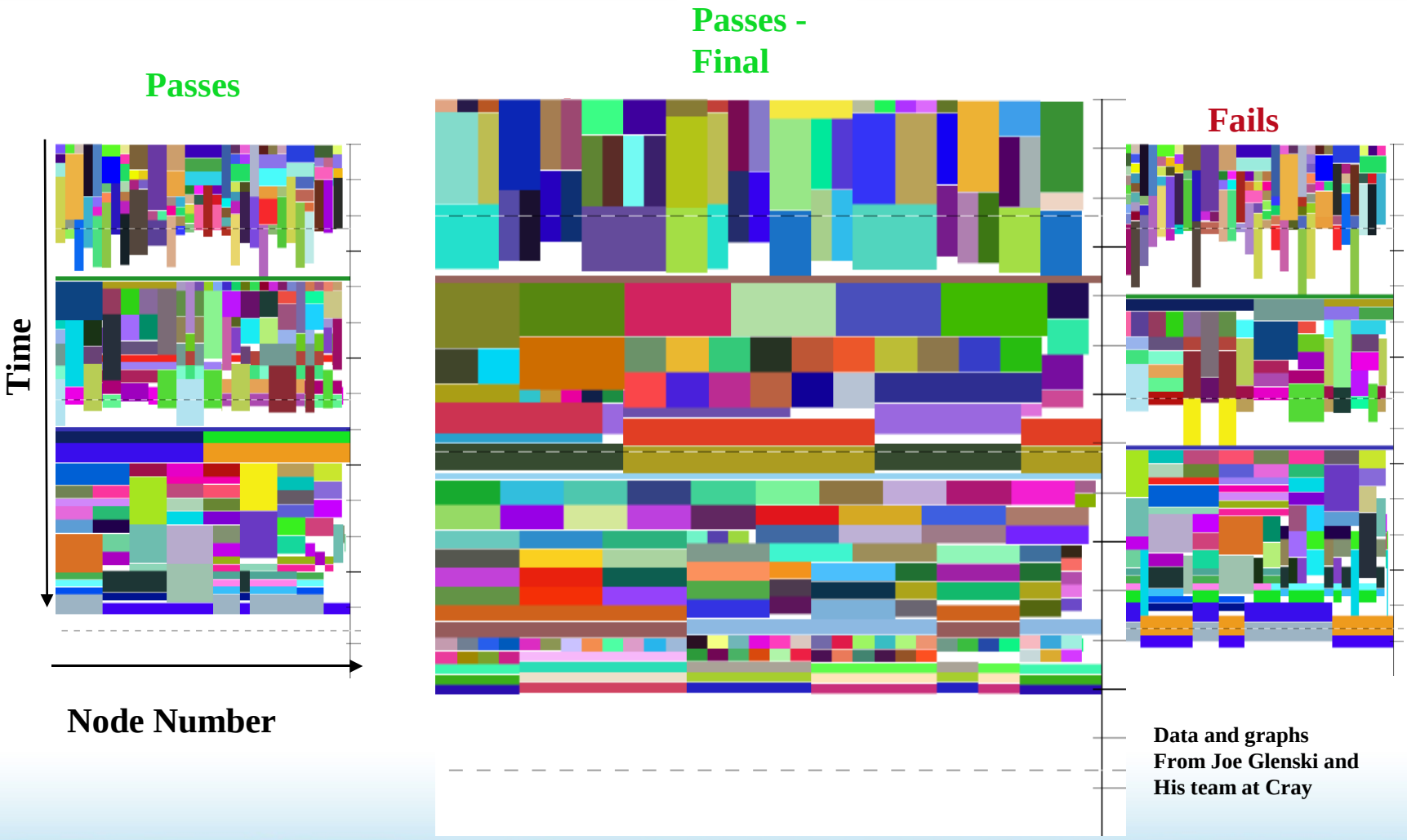
Tera to Peta-scale - Effective System Performance Test

- The ESP measure
  - Both how much and how often the system can do scientific work
  - How well do systems get the right job to run at the right time
    - Needed for a Service Oriented Infrastructure
  - How easy can the system be managed
- Independent of hardware and compiler optimization improvements



$$E_{s,k} = \frac{\sum_{i=1}^{I} (c_i * T_i)}{[N_{s,k} * (T - BEST_s)]}$$

ESP-2 Tests on Cray's Linux Environment



Data and graphs
From Joe Glenski and
His team at Cray

ESP Alone is Not Sufficient for Peta+Scale

- What is still relevant
  - Realistic Resource Requirements
  - On-going submissions – changing context
  - Efficiency of Launch/reallocation
  - Differing operational modes
- New Metrics are needed
  - Random, realistic failures
  - Job Reconfiguration
  - Heterogeneity
- Reduce ESP Limitations
  - No Dynamic Provisioning
  - No Failures
  - Uniform resources
  - Single workflow steps

LESSONS FROM PETA-SCALE

Lessons for Peta+scale Plus

- Peta+scale systems require resource management not only assign initial resources, but also to constantly replenish resources.
  - Applications will need to dynamically load balance with more or less resource units
  - Resource Management will need to dynamically reallocate resources to running applications
    - Needs to be done on demand
    - Needs to be done in a timely manner (wasting other resources while waiting)

Lessons for Peta+scale Plus

- Resource managers have to support application based resiliency by providing accurate information for decisions and providing mechanisms for dynamic resource adjustments.
  - Accurate error and status propagation
  - Give the applications the best possible information
    - Take a lesson from severe weather forecasting – getting more accurate and specific over time
  - Applications will have a latency in their response – both the application and the resource manager needs to deal with that
  - Resiliency should not waste resources

Lessons for Peta+scale

- The science of job scheduling and resource management has to incorporate objective performance and effectiveness criteria in order to evaluate choices and tradeoffs.
- Remember – the key question of all resource management – “Have I made things better?”
  - Without objective measures – how can you tell?
- Snir and Bader productivity rules make sense but can not map to shared use systems
  - The goal of a HPC system is to “minimize the cost of solving P on system S in time T.”
  - The resource manager‘s goal is to take it one step further – “minimize the cost of solving the set of problems {P} on system S in time T”
- Tests and measures must be
  - Repeatable, generalizable, independent (of hardware and workload), comparable

Lessons from Peta+scale

- Multiple Metrics will exist
  - User view point
    - the most productive work they need in the short wall clock time
    - short wait times
    - infinite flexibility
  - System stakeholder
    - a highly utilized system
    - lowest cost of ownership
    - showing a unique resource
  - System Manager
    - Meeting metrics – quality of service, cost, effectiveness

Summary

- The critical Peta+scale resource the needs the most effective management is bandwidth
- Peta+scale and beyond systems require resource management not only assigns initial resources, but it also able to constantly replenish resources.
- Peta+scale resource managers have to support application based resiliency by providing accurate information for decisions and providing mechanisms for dynamic resource adjustments.
- The science of job scheduling and resource management has to incorporate objective performance and effectiveness criteria in order to evaluate choices and tradeoffs.