LA-UR- 10-01119

Approved for public release;
distribution is unlimited.

Title:	Experiences from the Roadrunner Petascale Hybrid System
Author(s):	Darren J. Kerbyson, Z# 176262, CCS-1 Scott Pakin, Z# 179752, CCS-1 Mike Lang, Z# 171038, CCS-1 Jose Carlos Sancho Pitarch, Z# 195715, CCS-1 Kei Davis, Z# 117267, CCS-1 Kevin J. Barker, Z# 201836, CCS-1 Josh Peraza, Z# 235813, CCS-1
Intended for:	SIAM Parallel Processing Conference Seattle, WA February 25, 2010

Los Alamos
NATIONAL LABORATORY
EST. 1943

Form 836 (7/06)

Abstract:

The combination of flexible microprocessors (AMD Opterons) with high-performing accelerators (IBM PowerXCell 8i) resulted in the extremely powerful Roadrunner system. Many challenges in both hardware and software were overcome to achieve its goals. In this talk we detail some of the experiences in achieving performance on the Roadrunner system. In particular we examine several implementations of the kernel application, Sweep3D, using a work-queue approach, a more portable Thread-building-blocks approach, and an MPI on the accelerator approach.

COMPUTER, COMPUTATIONAL & STATISTICAL SCIENCES

Office of Science
U.S. DEPARTMENT OF ENERGY

ASC

Experiences from the Roadrunner Petascale Hybrid System

Darren J. Kerbyson

Scott Pakin, Mike Lang, Jose Sancho, Kei Davis, Kevin Barker, and also Josh Peraza

Performance and Architecture Laboratory (PAL)

http://www.c3.lanl.gov/pal

Computer, Computational & Statistical Sciences Division

Los Alamos National Laboratory

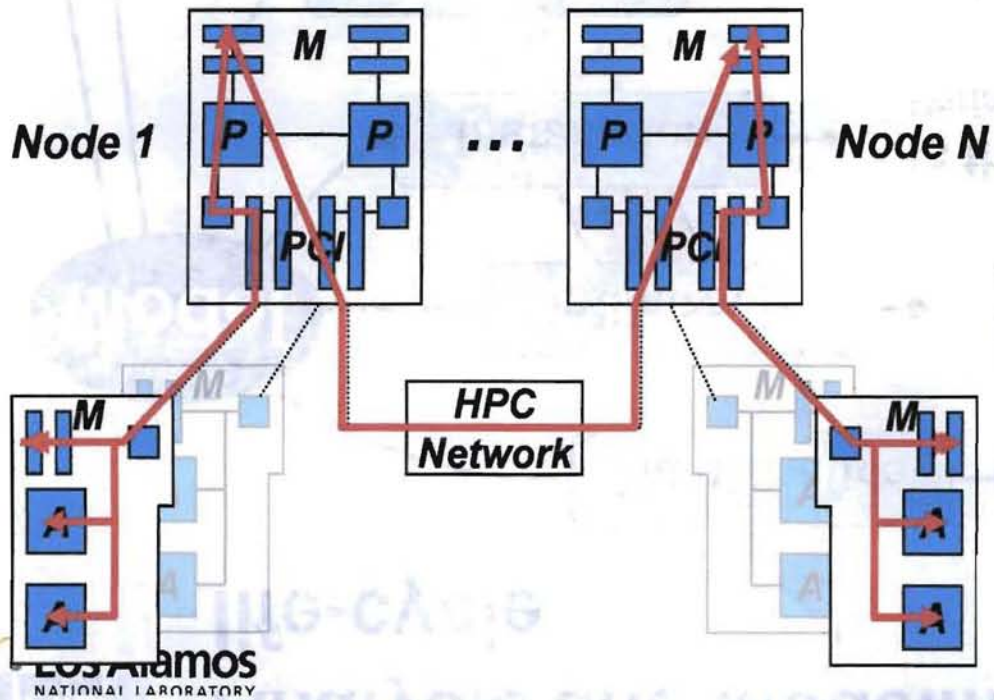
NNSA

Los Alamos
NATIONAL LABORATORY


Los Alamos
NATIONAL LABORATORY
EST. 1943

Design Space Exploration:
Circa 2005 Two-level System Heterogeneous System

- Compute nodes (e.g., with 2-sockets)
- HPC interconnection network (e.g., Infiniband)
- Accelerators placed in each node (e.g., PCI based)



1) Start-up
Node → Accelerator

2) Process on accelerator

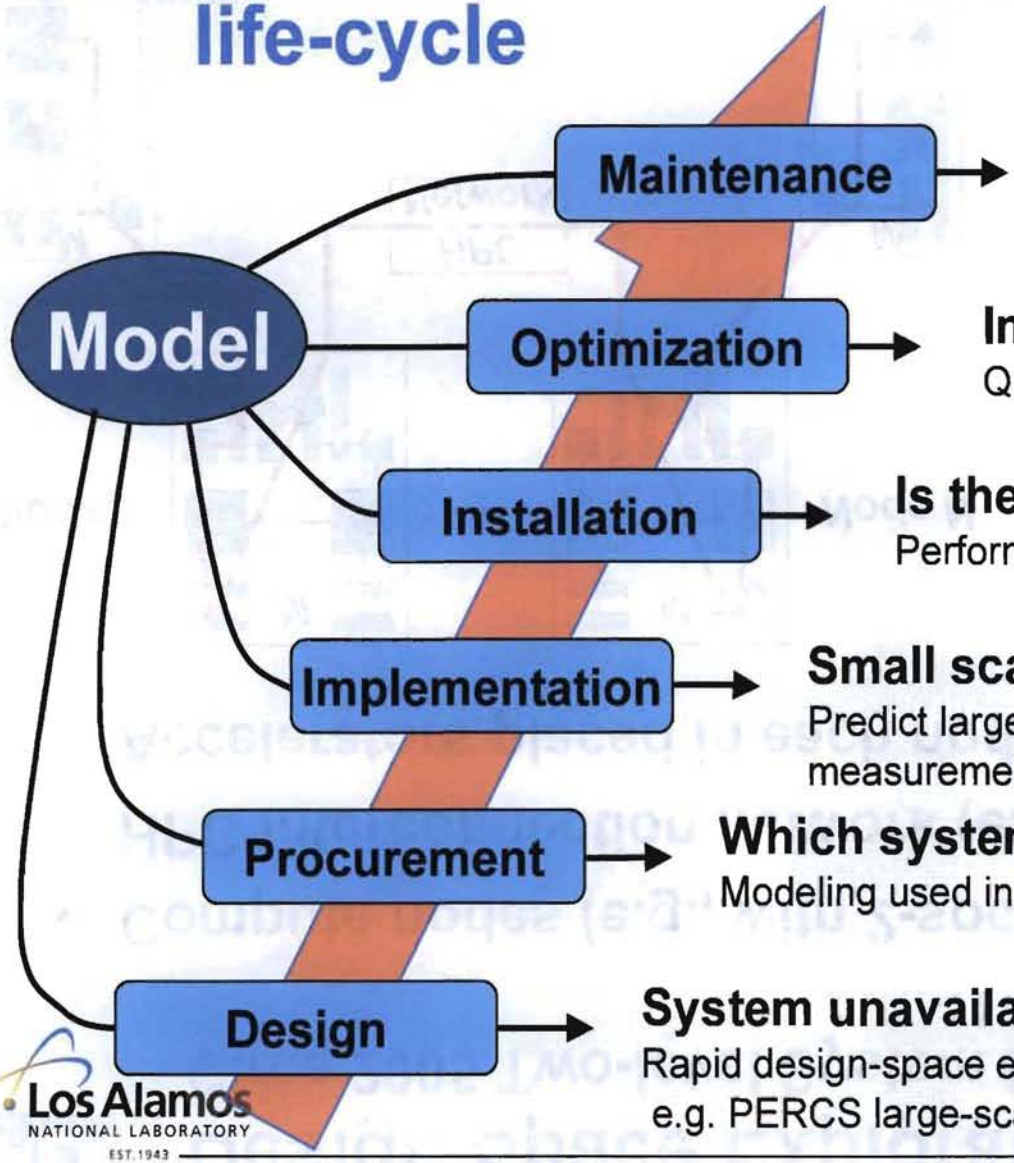
3) Inter-node communication
Accelerator → Node → HPC Network → Node → Accelerator

4) Repeat 2 (& 3)

5) Finalize
Accelerator → Node

"A Performance Analysis of Two-Level Heterogeneous Processing Systems on Wavefront Algorithms", D.J. Kerbyson, A. Hoisie, Unique Chips and Systems, John and Rubio (Eds), pp. 259-290, 2008.

Analysis and Modeling used throughout life-cycle



Runtime operation
The Performance Health Monitor:
Is the system healthy today?

Improvements
Quantify impacts prior to implementation

Is the machine working?
Performance should be as expected

Small scale (nodes) available
Predict large-scale system performance using measurements @ small-scale

Which system should LANL buy ?
Modeling used in procurements for almost a decade

System unavailable for measurement
Rapid design-space exploration.
e.g. PERCS large-scale system performance

PAL Talk Outline

- **Overview of Roadrunner**
  - Time-line
  - Architecture
  - Some low-level performance
- **Two styles of Programming models**
  - Acceleration
  - Reverse Acceleration
- **Example wavefront application**
  - Optimizations and performance
- **Summary**

Roadrunner is a first …

- 1st general purpose HPC system to break a petaflop:

  Initially: 1.38 PF peak

  1.026PF sustained on Linpack benchmark

  (***Kistler, Gunnels, Benton, Brokenshire***)

  First #1 Infiniband machine

  First #1 Linux machine

  First #1 heterogeneous machine

  First #1 commodity Cluster

- **Productive for many applications**
  - Accelerating science
- **Low Power (currently 6th on green500)**

Performance analysis of accelerated systems initiated in 2005

pre-RR

- **mid '05** Modeling of systems with Accelerators
- **Apr '06** Roadrunner RFP available
- **mid '06** IBM chosen as vendor

Roadrunner Base-system

- **Jan '07** Base-system Operational
  71TF Opteron cluster (replaced ASCI Q, Cells to be added later)

Final-system Configuration

- **During '07 Final configuration developed**
  Integrated Tri-blade (Opterons & Cell eDP), & Cell-BE evaluation
- **Jun '07** First testing of Cell eDP
- **Oct '07** Assessment of final configuration

Final-system

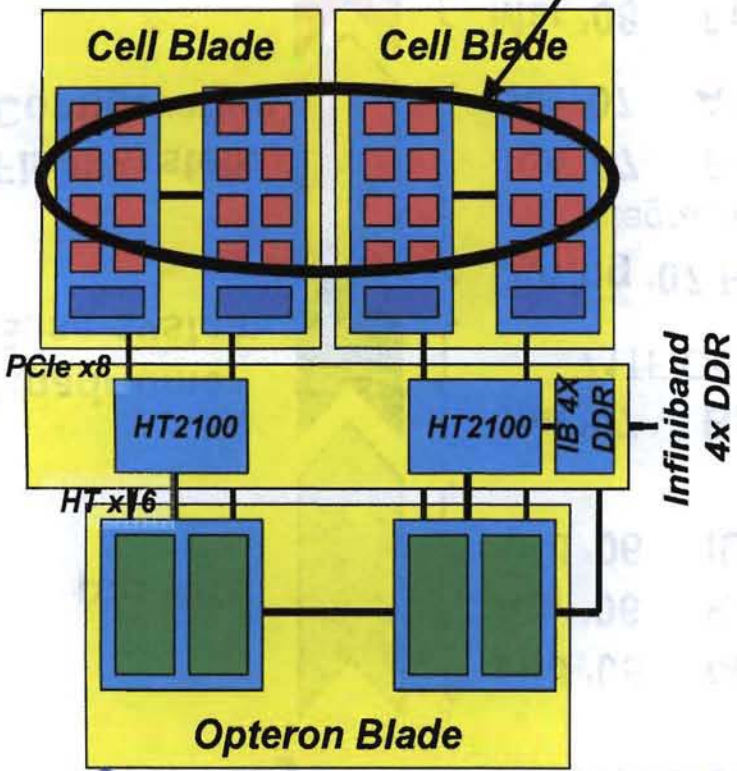
- **Mar '08** First triblades available
- **Apr '08** First Roadrunner CU tested
- **May '08** All 17 CUs (> 1 PF on Linpack benchmark)
- **Nov '08** Acceptance Testing

Production

- **2009**

Roadrunner: Compute Node

91% flops
83% memory bandwidth
50% memory capacity

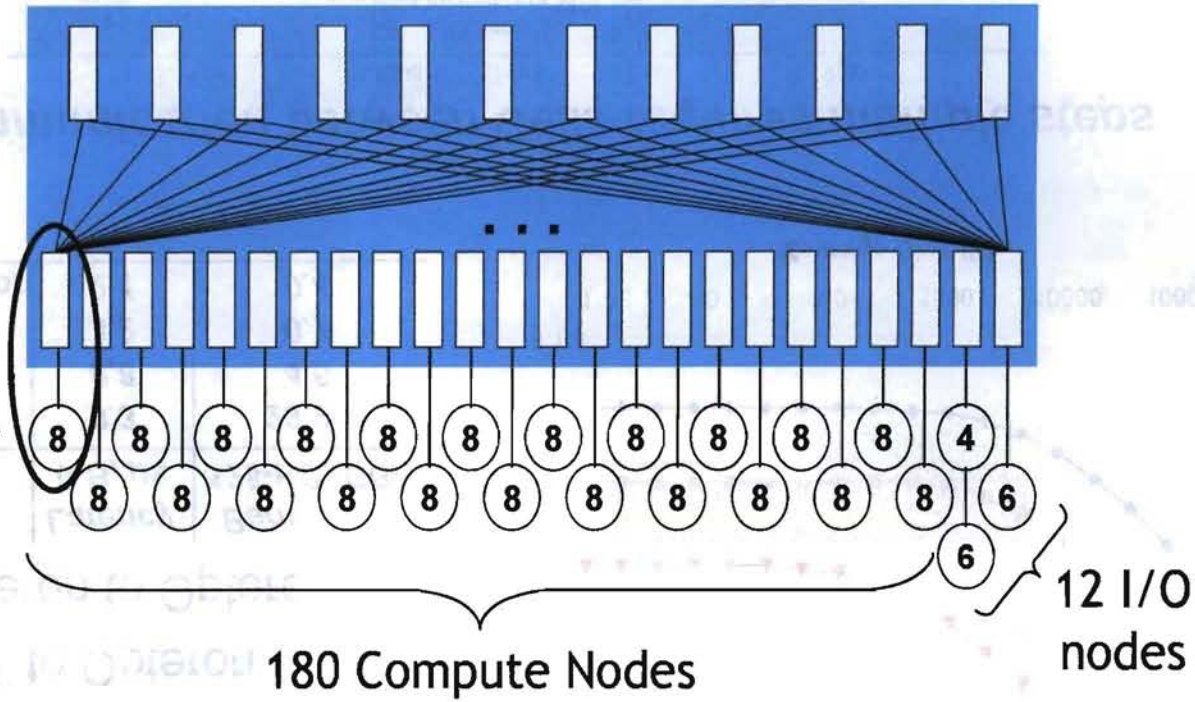


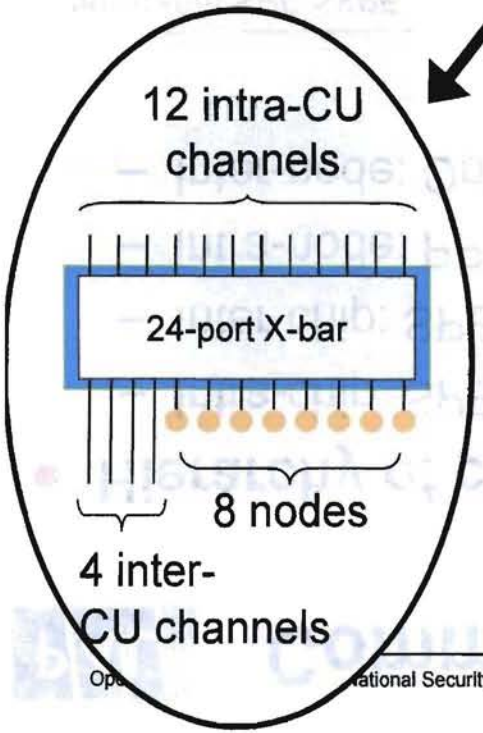
- **Hybrid**
  - AMD Opterons blades
  - IBM PowerXCell 8i (Cell)
  - One-to-one: Opteron-to-cell
- **Compute node**
  - 1x Opteron blade (2x dual-cores)
  - 2x Cell blades (each: 2x PowerXCell8i)
- **All *standard* services provided by the Opterons**
- **Application challenge:**
  - Efficient utilization of the Hybrid system

Roadrunner: Compute Unit (CU)

288 Port 4x DDR Infiniband Switch

36 individual 24-port xbar chips



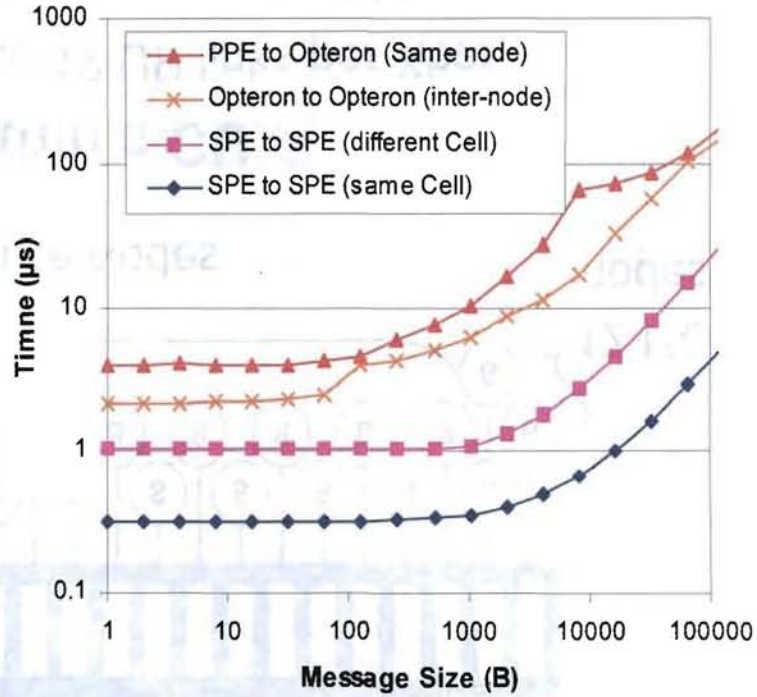


- **Full fat-tree within a CU**
  - 8 DOWN links to 12 UP links per Xbar
- **Reduced fat-tree between CUs**
  - Only 4 inter-CU links per 8 nodes (2:1 reduction)

Communication Performance

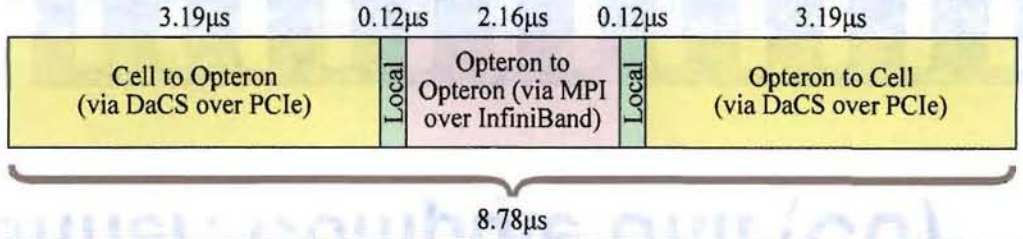
- **Hierarchy of channels, e.g.**
  - Intra-chip: SPE to SPE
  - Inter-chip: SPE to SPE
  - Intra-node: PPE to Opteron
  - Inter-node: Opteron to Opteron

	Latency 0-B, µs	*Bandwidth 128-KB, GB/s*
Intra-chip SPE->SPE	***0.3***	***23.9***
Inter-chip SPE->SPE	***0.8***	***4.5***
Intra-node PPE->Opteron	***3.2***	***0.7***
Inter-node Opteron->Opteron	***2.1***	***0.8***



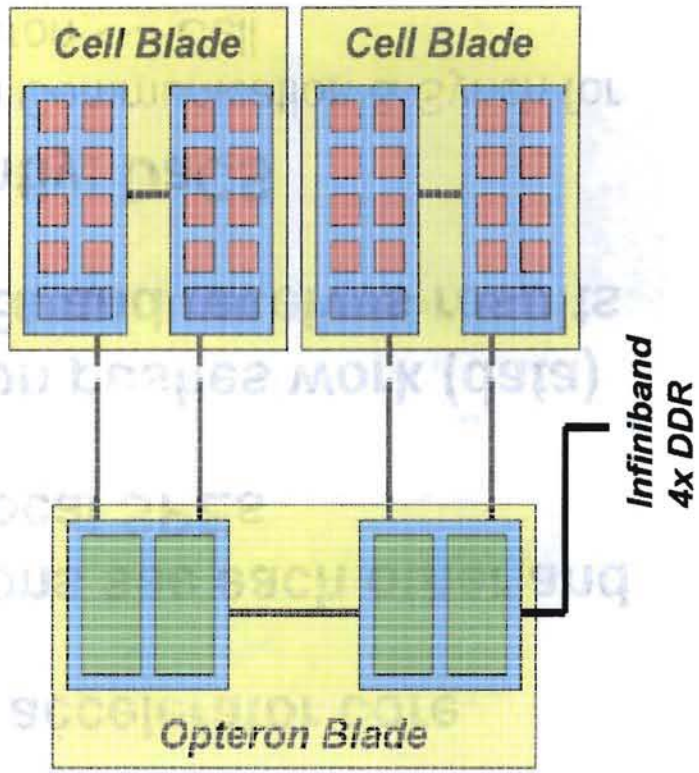
- ***Inter node communication between Cells requires multiple steps***

e.g. 0-byte latency

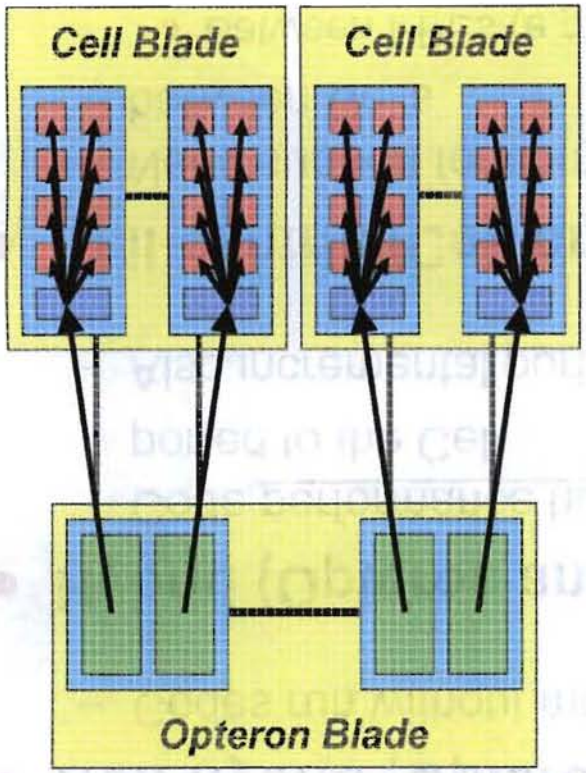


PAL Roadrunner: Usage

- **Non-hybrid (Opteron only)**
  - Codes run without modification
- **Hybrid (Opteron and Cell)**
  - Code performance hotspots ported to the Cell
  - Also incremental porting
- **Cell-centric (Cell only)**
  - Need support for communications between Cells
    - » Between PPEs (e.g. MP Relay)
    - » Between SPEs (e.g. CML)



Hybrid (general accelerator approach)



- One MPI rank per Opteron
- SPE = accelerator core
- Opterons see each other and their local SPEs
- Opteron pushes work (data) to SPEs and receives results
- Currently: DaCS
  - Data Communication & Synch for Opteron <-> Cell
- libSPE (or ALF) for SPE work management

Approach – push down to SPEs

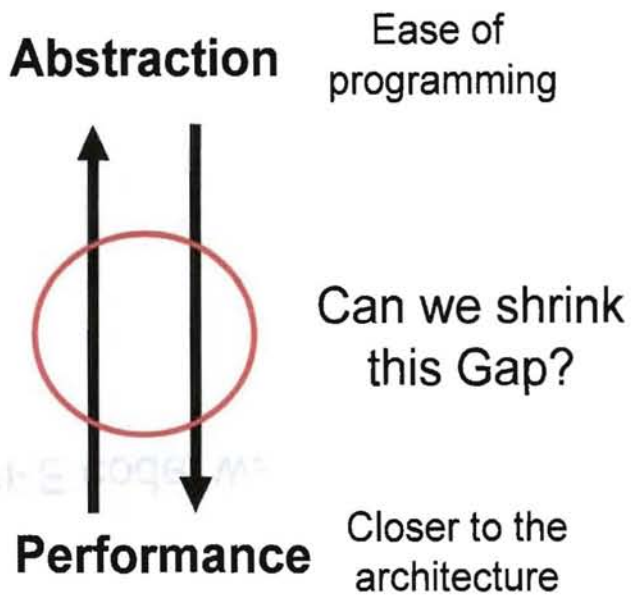
- **Focus on code hotspots**
  - Limited amount of code which use the greatest # cycles
- **Opteron**
  - Initialize DaCS, reserve PPE, launch PPE code, wait/poll PPE to finish
- **PPE**
  - Determine number of SPEs available
  - Create context for each SPE
  - Spawn and execute code on SPEs
- **SPE**
  - Mailboxs to get info from ppe
  - RDMAs to transfer data

Performance vs. Abstraction

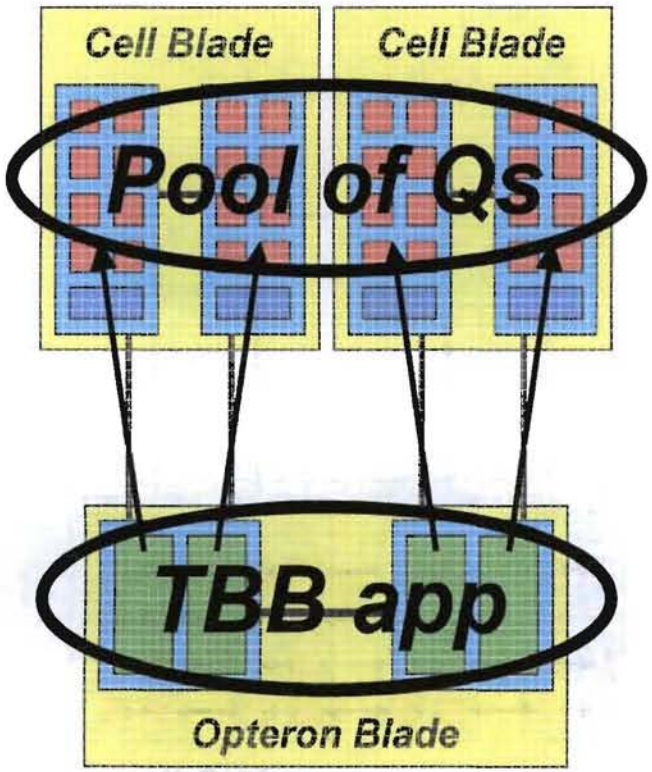
The tension between *performance* and *abstraction* is as old as computing itself.

- **Intimate knowledge of, and direct access to, the underlying hardware allows the extraction of best possible performance**
- **Abstraction makes programming easier, and programs more portable, typically via**
  - Programming languages
  - Libraries
- **Work on the Cell has shown that apps can be optimized to effectively utilize multi-core**
  - With a high learning curve
  - And still at great programmer effort.
- **How can we get higher level abstraction?**

  ...and at what cost in performance?



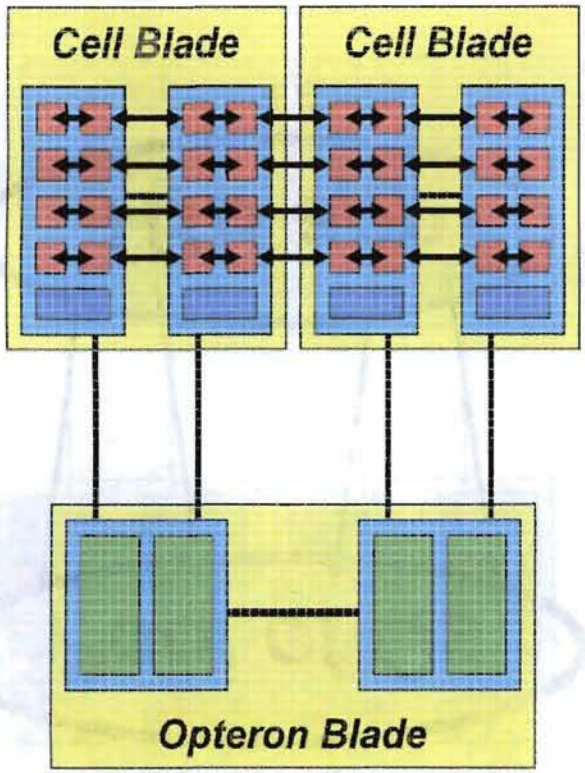
Example: Thread Building Blocks



- **Exploring TBB as a possible programming model for Roadrunner**
  - Opterons run a TBB applications
  - Each SPE implements a work queue
  - PPE not visible
    - » **used only for communication between opteron and SPEs**
- **PPE can be bottleneck and limit performance**
- **Portability at a cost in performance**

Los Alamos NATIONAL LABORATORY EST. 1943

Reverse Acceleration Model (SPE-centric)



- **MPI for the accelerator cores**
- **One MPI rank per SPE**
- **Opteron = NIC (support)**
- **SPEs see each other and their local Opteron**
  - SPEs communicate directly with other SPEs
  - PPE provides support
- **MPI subset, currently:**
  - blocking MPI pt2pt & collectives
  - Small memory footprint
- **"Cluster of 100,000 SPEs"**

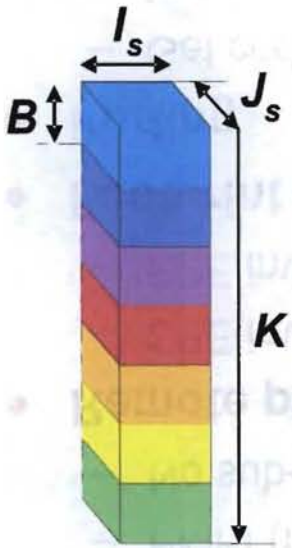
Los Alamos NATIONAL LABORATORY EST. 1943

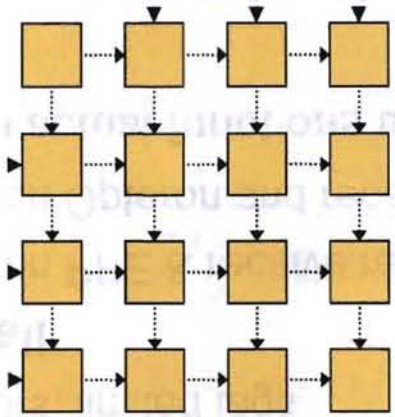
"The Reverse Acceleration Model", S. Pakin, M. Lang, D.J. Kerbyson, IBM J. Res. & Dev., 53(5), 2009

PAL Overview of the Cell-Messaging Layer

- **Subset of MPI**
  - `MPI_Abort(), MPI_Allreduce(), MPI_Barrier(), MPI_Bcast(), MPI_Comm_get_attr(), MPI_Comm_rank(), MPI_Comm_size(), MPI_Finalize, MPI_Init(), MPI_Recv(), MPI_Reduce(), MPI_Send(), MPI_Wtime(), MPI_Wtick()`
  - more Added as needed
  - PMPI (for profiling)
  - No sub-communicators, limited tags
- **Remote procedure call**
  - SPE invoke function on PPE & receive result (e.g. malloc)
  - PPE invoke function on Opteron and receive result (e.g. I/O)
- **Footprint - Only link in actual functions used, worst case ~12KB**
- **Porting**
  - Get code working on a small problem (fit into local-store)
  - Incrementally add data movement to & from main memory
  - Incrementally optimize code (vectorize, align, hand unroll etc)

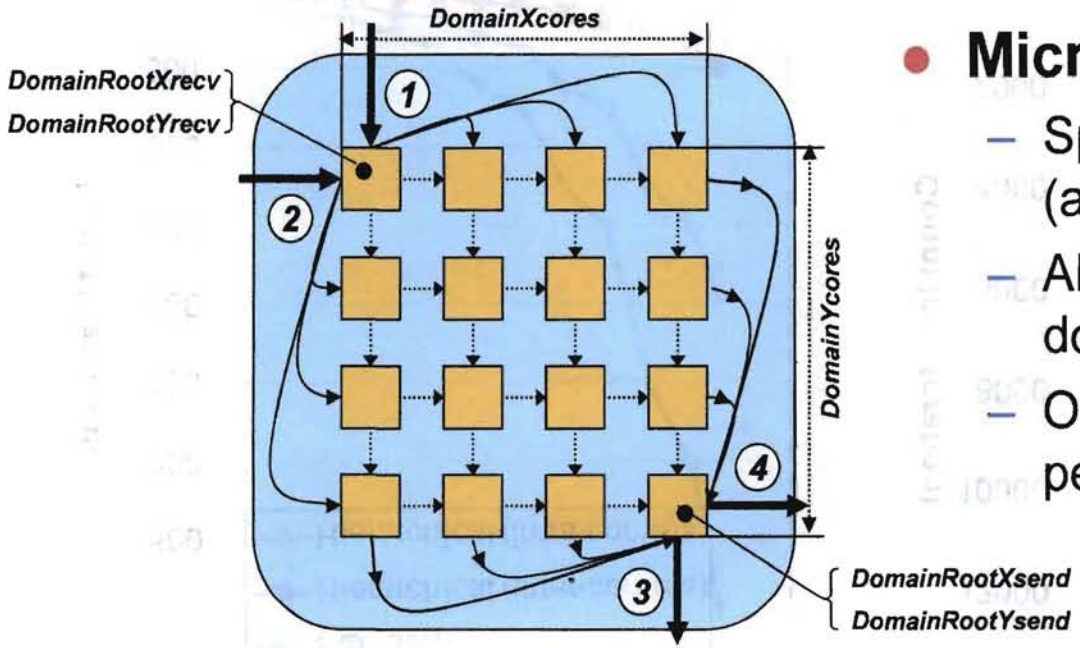
Example: Wavefront processing





- **Processing dependency between grid-points**
  - determines ordering within and between cores
- **Decomposition = data dimensionality minus one**
  - Use blocking to increase efficiency

Explored optimizations using Modeling



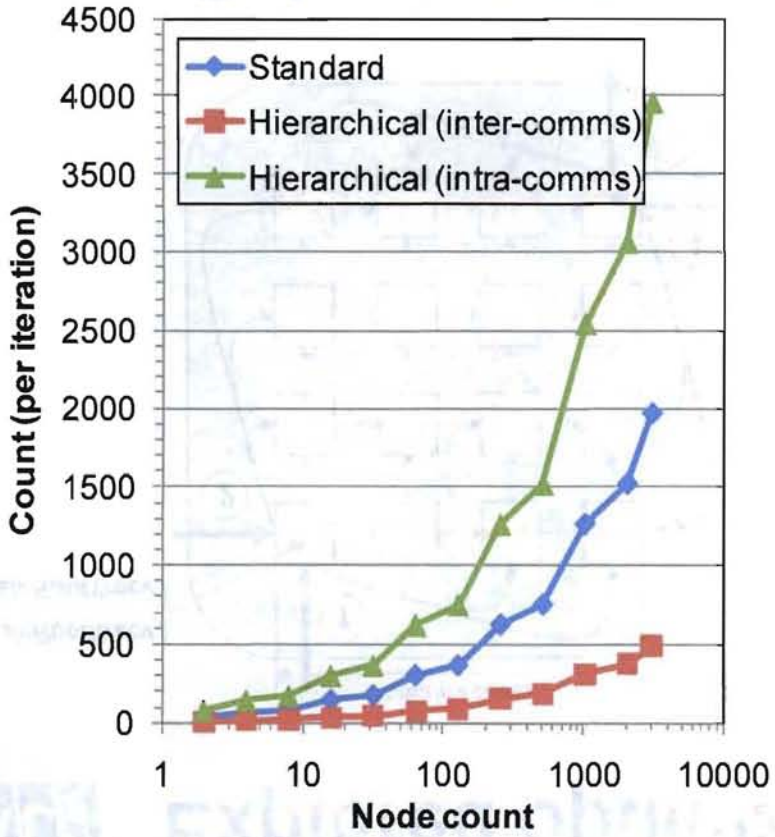
- **Micro-blocking**
  - Split block into smallest unit (a single K-plane)
  - All rapid propagation across domain using on-chip comms
  - One point of entry (and exit) per dimension in a core-domain

- **Reduce inter-domain (slow) communications, but**
  - Increase computation steps, and
  - Increase on-chip (fast) communications
- **Trade-off: computation vs. communication**

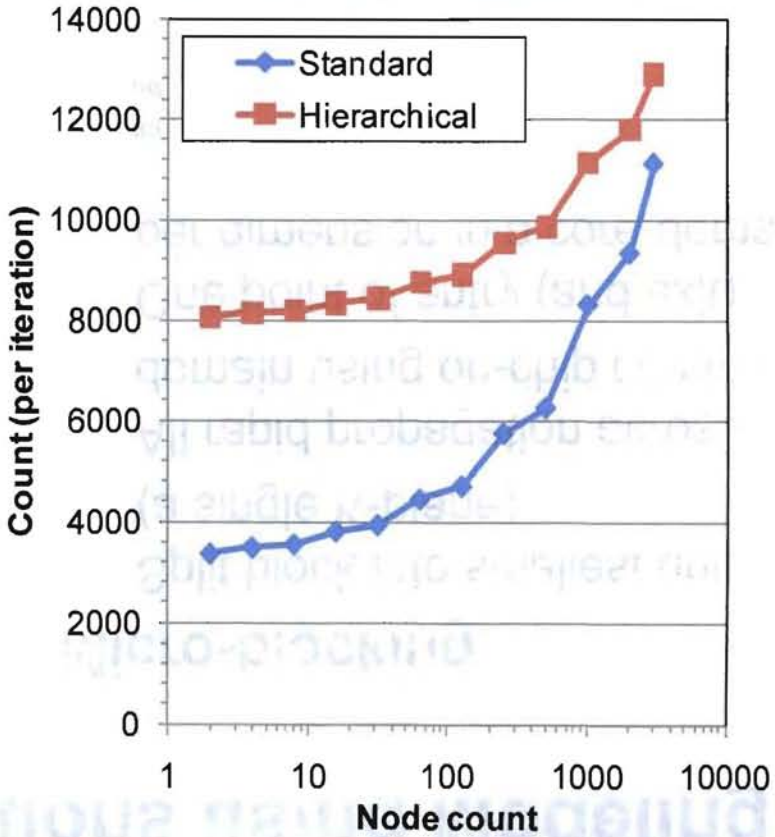
Trading Communication for Computation

- **Reduced pipeline communications BUT increased intra-blade comms (fast) & computation steps**

Communication steps

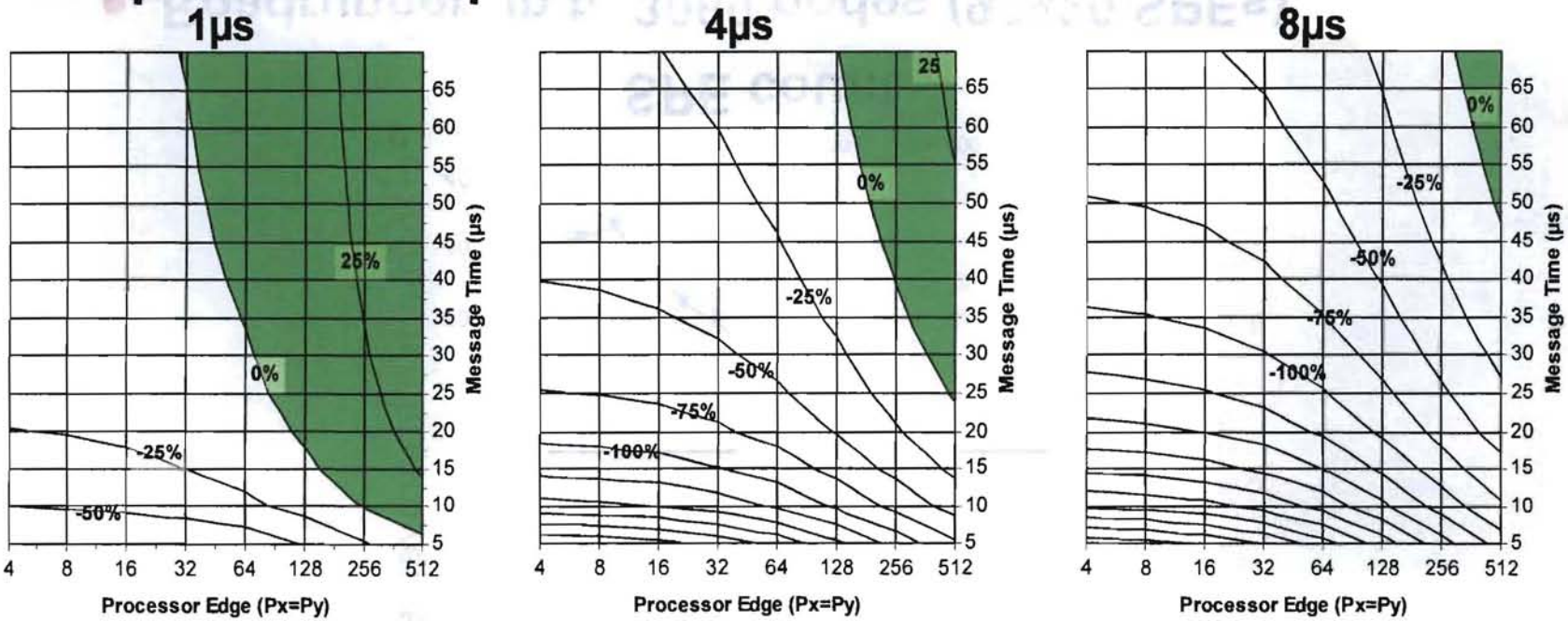


Computation steps



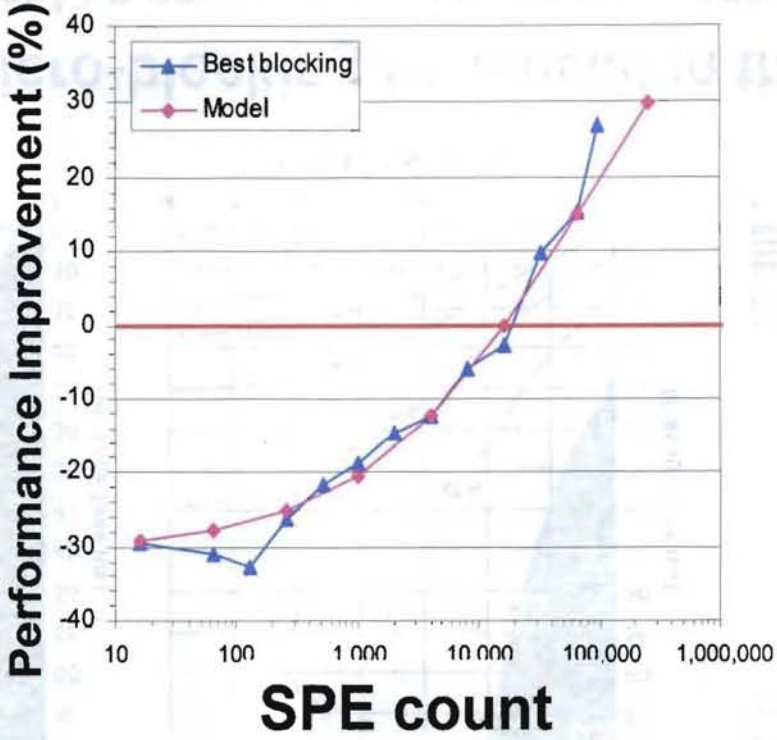
Micro-blocking is advantageous at large-scale & with large message latencies

Compute time / k-plane →



- **Use of micro-blocking beneficial in the green areas**
  - Dependent on scale, inter-domain message time, k-plane compute time (as well as problem size and blocking)

Implementation illustrates high initial accuracy of modeling



- **Roadrunner up to 3060 nodes (97920 SPEs)**
  - Use of Cell-Messaging Layer (CML)

Los Alamos NATIONAL LABORATORY

"Adapting Wave-front Algorithms to Efficiently Utilize Systems with Deep Communication Hierarchies"
D.J. Kerbyson, M. Lang, S. Pakin, to appear in Parallel Computing, 2010

PAL Summary

- **There are many options in programming accelerated systems**
  - Acceleration model
  - Reverse acceleration model
- **Depends on resources available in each core**
- **Code porting / optimizations**
  - Need to rethink many application implementations
  - Algorithmically
  - Architecture specifics hopefully kept to a minimum
- **Portability is important**
  - But may come at a performance cost
- **Do not forget about scaling issues**
  - Node level accelerated performance may be misleading
- **Roadrunner achieving high performance for many applications**

See also MS64 - Friday @ 4:30-6:30pm

Porting Today's Codes to Tomorrow's Architectures

4:30-4:55 The Roadrunner Computing Architecture: System Overview and Programming Models

Ben Bergen, Los Alamos National Laboratory

5:00-5:25 Understanding the Onset and Saturation of Laser Backward Stimulated Raman Scattering Through Large-Scale Plasma Kinetic Simulations on a Hybrid Supercomputer

Lin Yin, Brian Albright, Kevin Bowers, Ben Bergen, Los Alamos National Laboratory

5:30-5:55 Direct Numerical Simulations of Compressible Reacting Turbulence with Type Ia Supernovae Microphysics

Daniel Livescu, Los Alamos National Laboratory

6:00-6:25 SPaSM: Large-Scale Molecular Dynamics Studies of Material Dynamics on Roadrunner

Timothy C. Germann, Los Alamos National Laboratory