Correctness Field Testing of Production and Decommissioned HPC Platforms at Los Alamos National Laboratory

Sarah Michalak
William Rust
John Daly
Andrew DuBois David DuBois

For more info: SC14 paper with the same title

Acknowledgements

Terri Bednar

Laura Davey

Mike Ferguson

Gary Grider

Chuck Hales

Greg Hamilton

Tim Harrington

Doug Hefele

Kaki Kelly

Anthony Lopez

Jim Lujan

Andy Martinez

Cameron McNairy

Amy Meilander

Andrew Montoya

Lisa Moore

Terri Morris

John Morrison

Nick Nagy

Randal Rheinheimer

Norbert Seifert

Randy Smith

Bob Tomlinson

Philip Ulibarri

Mark Vernon

Andy White

…

Motivation for Testing

Circa 10 years ago: growing awareness of silent data corruption (SDC)

Investigate impact of incorrect results on LANL HPC platforms and users

Neutron beam testing, laboratory testing, field testing **Field**

testing goals:

Characterize incorrect results on LANL HPC platforms

Identify recurring problems so they can be fixed

Leverage existing technology to start testing quickly

Related Studies

SDC/incorrect results have been documented:

- Pre-production-use testing of Teraflops platform [Constantinescu, 2000]
- Systems at UWI [Kola et al. 2005]
- CERN [Panzer-Steindel, 2007; Kelemen, 2007]
- NASA [Behnke, 2005]
- NetAppliance™ storage systems [Bairavasundaram, 2008]
- Neutron-beam testing [Constantinescu, 2005; Ando et al. 2008; Sanda et al. 2008; Hong et al., 2009; Michalak et al. 2012]
- Proton-beam testing [Hiemstra et al., 1999, 2000, 2001, 2002]

Test Codes and Test Environment

HPL: mainly compute with some communication

Used on all platforms; OSATSDC test code available

Crisscross: data transfers (communication)

- Added original version in 2009; revised version in 2011
  - Original version: 128 MB incrementing pattern
  - Revised version: 6 patterns and user-defined transfer sizes

Automated test environment

- Launching of tests, writing of output lines, notification of incorrect results
- Number of jobs scales to the platform's limit
- Testing during production use and post-decommissioning
  - For production use testing, minimize impact on users by running short 1- or 2-node tests that were pre-emptible on otherwise-idle nodes
  - Amount of testing determined by external operational needs

Summary of Hardware Tested

Architectures of 12 Platforms Tested
(10/07-10/12)

Pf.	CPU	CPUs/Compute Node	Memory	Memory/Node (GB)	# Compute Nodes	Network
1	1.25 GHz Alpha 21264	4	EDO DRAM	4, 8, 16, or 32	1,408	Elan3 Quadrics
2	single-core 1.6 GHz Intel Itanium 2	4/Brick	DDR1 SDRAM	1 TB shared memory	64 Bricks	NUMAlink3
3	single-core 2.4 GHz Intel Xeon	2	DDR1 SDRAM	2	958	Myrinet 2000 Lanai XP
4	single-core 2.0 GHz or 2.4 GHz AMD Opteron	2	DDR1 SDRAM	8 or 16	937	Myrinet 2000 Lanai XP
5	dual-core 1.8 GHz AMD Opteron or	2	DDR1 SDRAM	4, 8 or 16	3,315	Myrinet 2000 Lanai XP single-core 2.0 GHz or 2.4 GHz AMD Opteron
6	single-core 2.6 GHz AMD Opteron 2 DDR1 SDRAM 8 1,290 IB SDR 7 dual-core 2.2 GHz AMD Opteron 4 DDR2 SDRAM 16 139 IB SDR 8 dual-core 2.2 GHz AMD Opteron 4 DDR2 SDRAM 32 1,834 IB SDR 9 8-core 2.0 GHz AMD Opteron 4 DDR3 SDRAM 64 416 IB QDR 10 dual-core 1.8 GHz AMD Opteron 2 DDR2 SDRAM 16 12 IB DDR 11 quad-core 2.2 GHz or 2.3 GHz AMD Opteron 4 DDR2 SDRAM 32 360 IB DDR					
12	quad-core 2.3 GHz AMD Opteron	4	DDR2 SDRAM	32	64	IB DDR

Numbers of Platforms Tested with HPL and Crisscross

	HPL	Crisscross
# of Platforms Tested During Production Use	6	3 #
of Platforms Tested Post-Decommissioning	10	6

Summary of Results

HPL and Crisscross Testing

(without Platform 2)

HPL

# Nodes	# Tests	# Incorrect	HPL Node Years	Total Node Years
1	17.5B	433	505	511
2+	4.3B	2,099	533	540
Total	21.8B	2,532	1,038	1,051

Crisscross

# Nodes	# Transfers	# Incorrect Transfers	Total Transferred (PB)
1	1.4T	9	120
2	1.3T	0	145
Total	2.7T	9	264

Pf.	HPL	Crisscross	Pf.	HPL	Crisscross
1	Intermittent (70)	-	7	None	Intermittent (9)
2	None	-	8	None	None
3	None	-	9	None	-
4	Transient (2)	None	10	None	None
5	Transient (2)	None	11	Intermittent (2,457)	-
6	Transient (1)	None	12	None	None

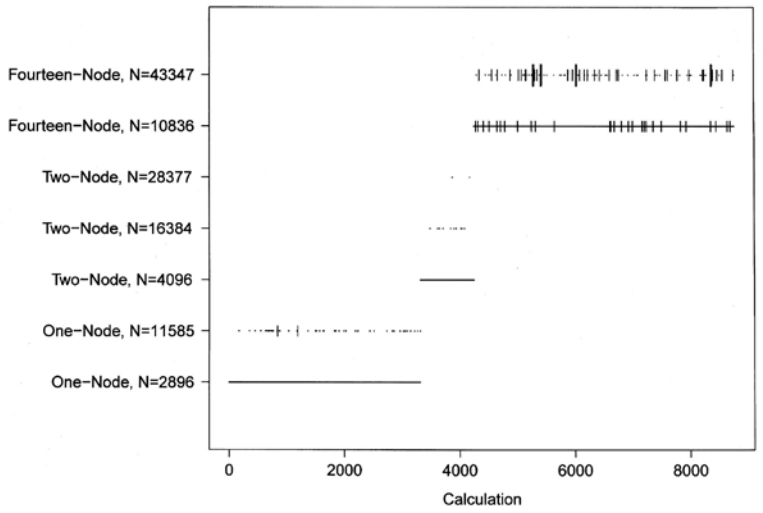
Intermittent Errors: Platform 1

70 incorrect HPL calculations post-decommissioning

1,114 nodes tested

0.19 Node Years One-Node HPL; 3 Node Years Multi-Node HPL

Platform 1 subjected to manipulation of ambient temperature, which was not associated with incorrect



results

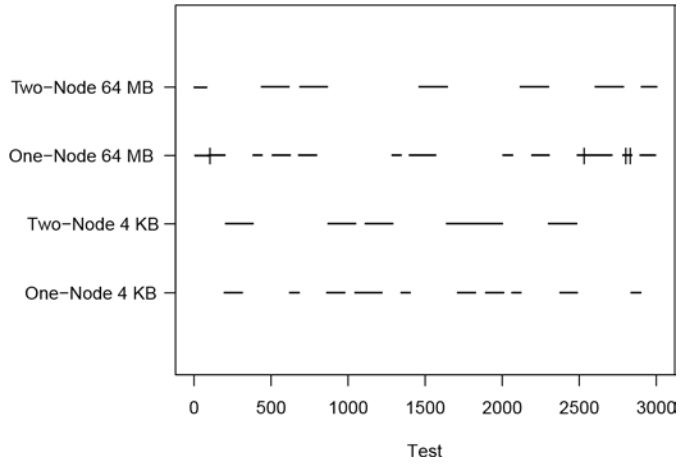
Intermittent Errors: Platform 7

9 incorrect Crisscross transfers on one node during production use

139 nodes tested

13 PBs within-node transfers; 8 PBs between-node transfers

Error messages and maintenance actions before and after the incorrect transfers



Intermittent Errors: Platform 11

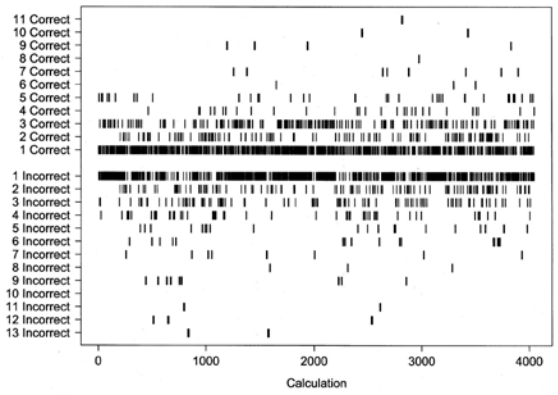
2,457 incorrect HPL results post-decommissioning:

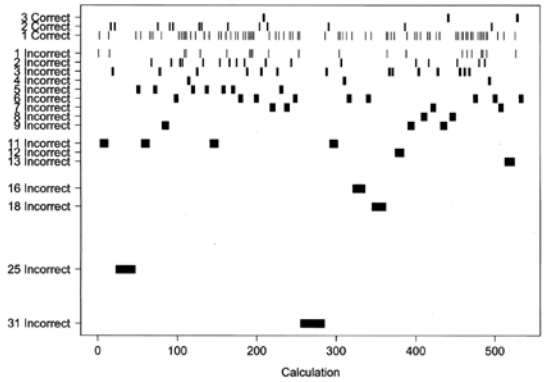
315 nodes tested

2 Node Years One-Node HPL; 9 Node Years Multi-Node HPL

Node 1 and Node 2: 2,029/4,050 incorrect

Node 1: 428/536 incorrect; Node 2: 0/535 incorrect





Nodes moved to another platform and ran HPL for several months with no incorrect results

Conclusions I

Large-scale testing over a 5-year period suggests that:

- Transient errors appear manageable
- Intermittent errors could have a much larger effect

Routine testing can be used to:

- Characterize errors
- Identify nodes experiencing intermittent errors
- Identify related issues: slow nodes, timestamp issues, grep issue, … Data cleaning is needed and supports this

Intermittent errors may resolve with routine maintenance actions

Automated test environment makes routine testing efficient and enables necessary changes to testing protocol

Conclusions II

Test results support resilience efforts:

Errors observed in the field that resilient methods could address (intermittent and transient) and that can be used to test resilient methods

Correctness testing during production use can be instrumented to minimize impact on users

Need to consider both the benefits of testing and the power use and wear costs

Some apparently silent errors may not be silent when all relevant logs are checked – how many users do this?

Systems need to propagate all potentially relevant error data to users/user processes so they can decide how to proceed with a computation