A Message-Logging Protocol for Multicore Systems

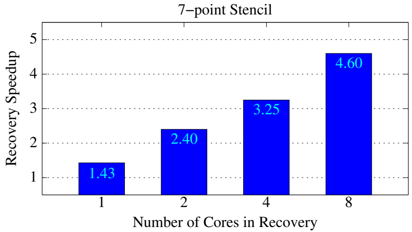
Esteban Meneses, Xiang Ni and Laxmikant V. Kalé

Parallel Programming Laboratory
University of Illinois at Urbana-Champaign

FTXS 2012

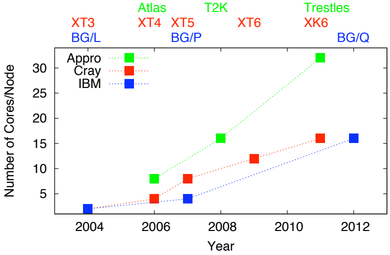
Message Logging

- Local rollback
- Less energy consumed
- Parallel recovery with migratable tasks



Multicore Systems

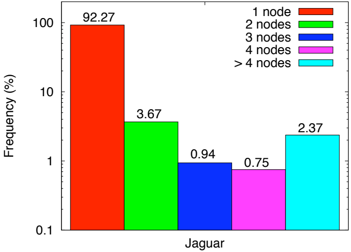
- Keep scaling FLOPS/s
- Almost Top 500 list entirely
- More cores per shared-memory node

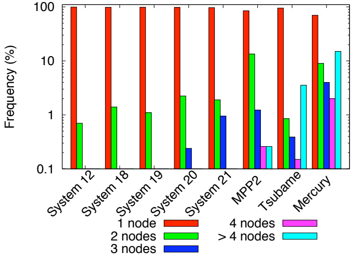


Agenda

Failures in HPC Systems

- The right unit of failure
  - Core, subset of cores, node, subset of nodes
- System logs
  - The Computer Failure Data Repository (CFDR)
  - Collaborations
  - Failure databases
- Jaguar
  - Top 6 in the world
  - 537-day study (8/08-2/10): 1253 separable events
  - Errors: machine check exceptions (MCE), interconnect (CRC), software





One failure, one node

x: number of nodes in a failure
Modeled through a random variable

Exponential decay
Geometric distribution

$$f(x) = (1-p)^{(x-1)}p$$

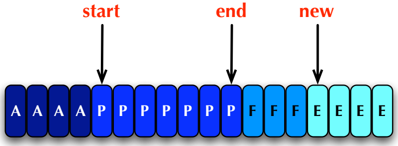
Heavy-tailed curve
Zipf's distribution

$$f(x) = \frac{\frac{1}{x^s}}{\sum_{i=1}^{n} \frac{1}{i^s}}$$

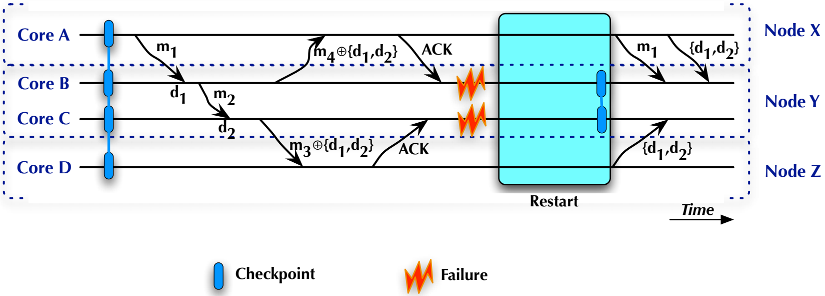
Message Logging

- Messages stored at sender
- Non-deterministic decisions recorded (determinants)
- Message reception order
- Causal message logging
    - Determinants stored in their causal path
    - Piggybacking determinants

- Failure unit: Core $\rightarrow$ Node
- Intra-node messages *not* stored
- Only inter-node messages piggyback determinants
- Shared data structure for determinants
- Lockless determinant queue



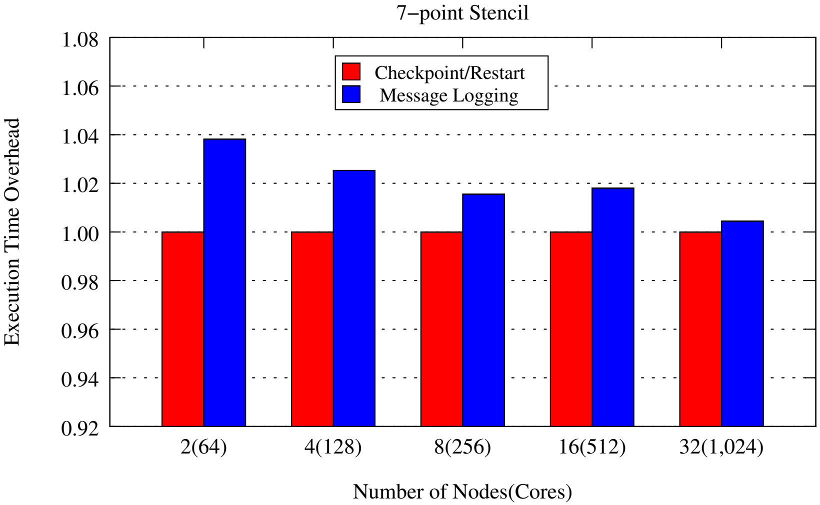
Protocol


Core A
Core B
Core C
Core D
Node X
Node Y
Node Z
m1
d1
m2
d2
m4⊕{d1,d2}
ACK
m3⊕{d1,d2}
ACK
Restart
m1
{d1,d2}
{d1,d2}
Time
Checkpoint
Failure

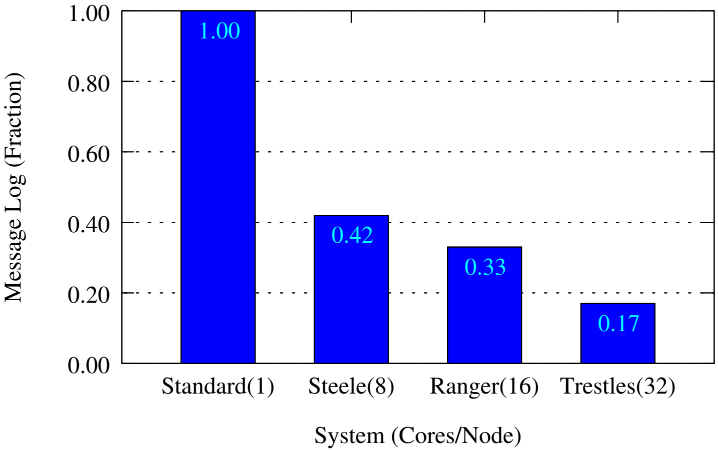
Implementation

- Charm++ runtime system
- A heavyweight process per node
- One process = one communication thread + worker threads
- Two fault-tolerance strategies
  - Double in-memory checkpoint/restart
  - Causal message-logging for multicore systems
- Testbed: Steele (RCAC), Ranger (TACC) and Trestles (SDSC)

Low Execution Time Overhead



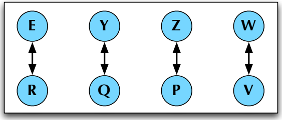
Reduced Memory Overhead



Efficient single-node failure reliability

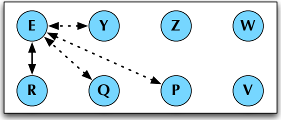
x: number of nodes in a failure
Modeled through a random variable
n: total number of nodes in the system
g: average number of acquaintances per node

Checkpoint/Restart



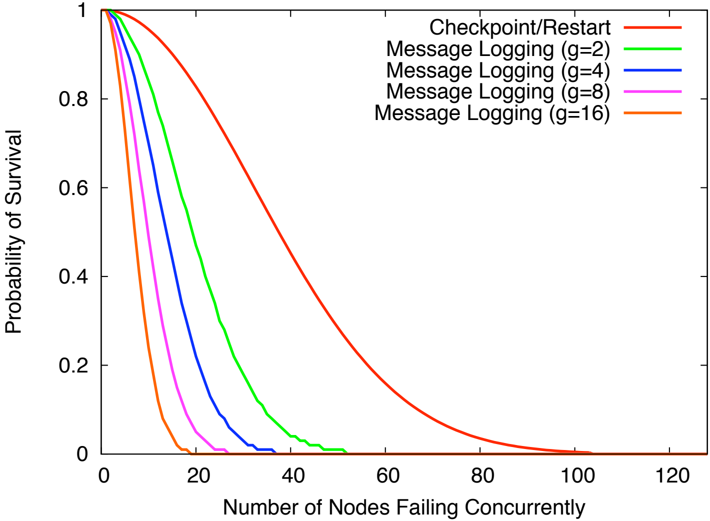
$$\frac{\prod_{i=0}^{x-1}(n-2i)}{\prod_{i=0}^{x-1}(n-i)}$$

Message-Logging



$$\left[\frac{\binom{n-x}{g}}{\binom{n-1}{g}}\right]^x$$

Probability of Survival



Survivability

- Survivability $\mathcal{S}$ is the weighted average over all possible failures
- $\mathcal{S} = \sum_{i=1}^{n} s(i)p(i)$

	Geometric	Zipf's
Checkpoint/Restart	0.9997	0.9992
Message Logging (g=2)	0.9988	0.9966
Message Logging (g=4)	0.9980	0.9945
Message Logging (g=8)	0.9964	0.9911
Message Logging (g=16)	0.9933	0.9854

Conclusions and Future Work

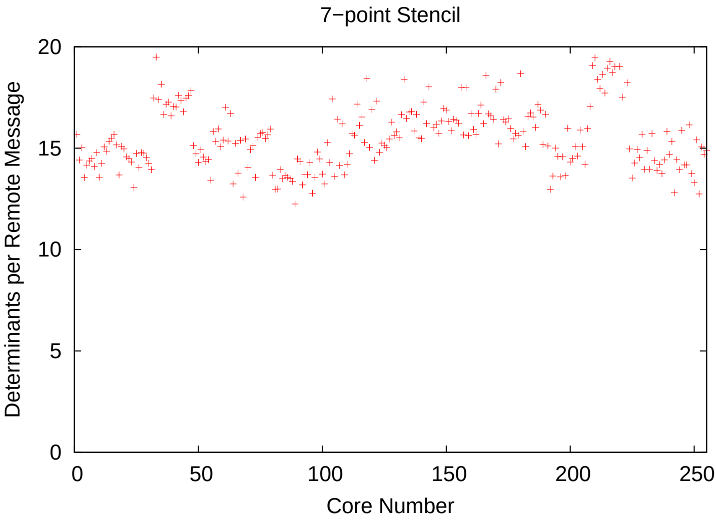
- Conclusions:
  - Most of failures in HPC systems involve one node
  - A message-logging protocol for multicore systems can be efficiently implemented
  - This protocol is almost as resilient as checkpoint/restart
- Future Work:
  - Explore more applications
  - Understand scalability of message logging protocol

Acknowledgements

- This work is partially supported by the US Department of Energy under grant DOE DE-SC0001845 and by a machine allocation on XSEDE under award ASC050039N.
- Failure datasets:
  - Jaguar: Terry Jones (ORNL).
  - Mercury: Ana Gainaru (UIUC).
  - Tsubame: Leonardo Bautista-Gomez (TokyoTech).

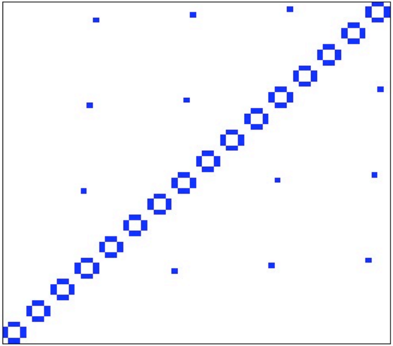
Thank You!
Q&A

Number of Determinants per Message



Communication Graph

NPB-CG



NPB-MG