Fault Tolerance Research in SCS lab (Dr Lan's group)

Wei Tang

Cs550 lecture 3/10/2011

Extreme scale computing

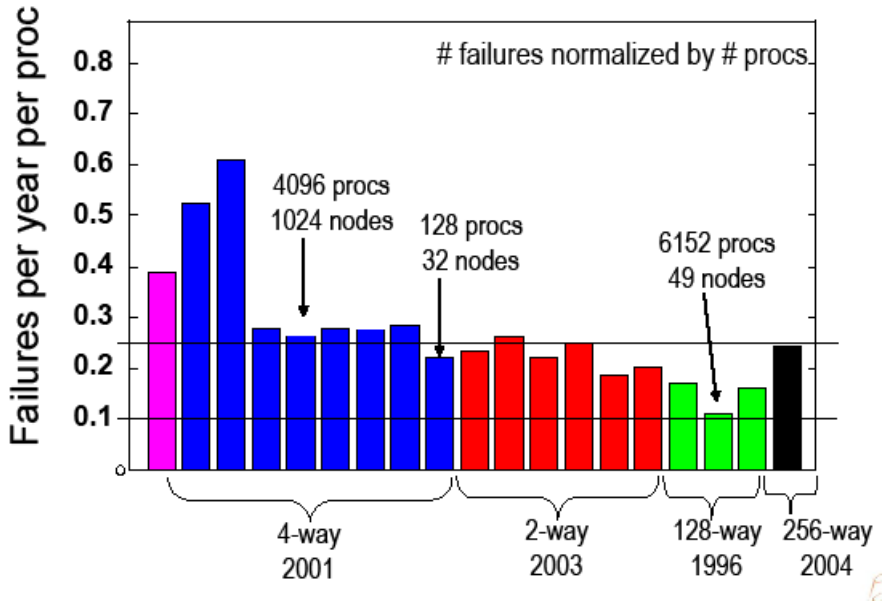
- The leading edge of high performance computing
  - FLOP: Float computing per second
  - Petascale (10^{15}), exascale(10^{18}), ….
  - One exaflop equals:
    - The combined performance of 50 million laptops - enough to reach 1000 miles from the ground when stacked, weighing over 100,000 tons.
    - 1000 times the power of today's most powerful supercomputer.
- Key challenges:
  - The energy and power challenge
  - The memory and storage challenge
  - The concurrency and locality challenge
  - The resilience challenge

Reliability concerns

- Systems are getting bigger
  - 2k-16k processors is today's "medium" size (92% of TOP500)
  - O(100,000) processor systems are being designed/deployed
- Even highly reliable HW can become an issue at scale
  - 1 node fails every 10,000 hours
  - 6,000 nodes fail every 1.6 hours
  - 64,000 nodes fail every 5 minutes

fault tolerance/resilience

Losing the entire job due to one node's failure is costly in time and CPU cycles!



FENCE & RAPS

FENCE
Pre-failure prediction & tolerance

Take precaution action based on failure forecasting

RAPS
Post-failure diagnosis & recovery

Quickly resume computing after failure occurrence

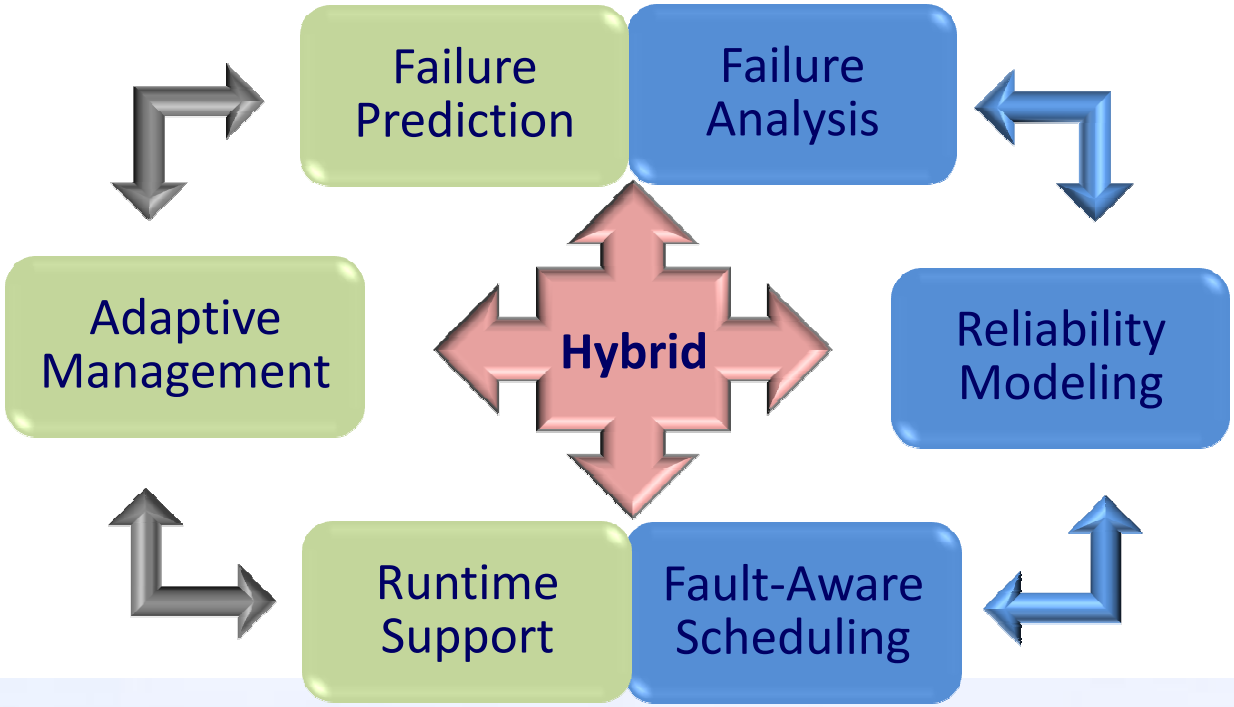
In collaboration with

- Xian-He Sun (IIT)
- N. Desai, D. Buettner, R. Thakur, S. Coghlan, R. Gupta and P. Beckman (ANL)
- B.H. Park and A. Geist (ORNL)
- J. White and E. Hocks (SDSC)

Pre-Failure Prediction & Tolerance

- Checkpoint is widely used for fault tolerance
  - Simple
  - I/O intensive, may trigger a cycle of deterioration
  - Reactively handle failures through rollbacks
- Newly emerging proactive methods
  - proactive failures and avoiding rollbacks
  - But, relies on accurate prediction of failures

FENCE overview



- FENCE: Fault awareness ENabled Computing Environment
  - *Proactive actions* prevent applications from anticipated failures
  - *Reactive actions* minimize the impact of unforeseeable failures

Research Issues

- Failure prediction
  - Using data mining or machine learning mechanism
- Adaptive management
  - Based on quantitative performability modeling
- Fault-aware rescheduling
  - Design runtime strategies to facilitate spare node allocation and job selection
- Runtime system support
  - Develop prototype system

Failure Prediction

- Challenges:
  - Potentially overwhelming amount of system data collected across the system
    - How to automate system log anlaysis
  - Faults are many and complex
    - There is no one-size-fit-all predictive method!
  - System changes are common in production system
    - How to adapt to system changes
- Our approach: dynamic meta-learning for failure prediction
  - Filtering methods, base predicative methods, ensemble learning techniques, …

  - **•J. Gu, Z. Zheng, Z. Lan, J. White, E. Hocks, B.H. Park, "Dynamic Meta-Learning for Failure Prediction in Large-Scale Systems: A Case Study", *Proc. of ICPP08, 2008.***
  - **•P. Gujrati, Y. Li, Z. Lan, R. Thakur, and J. White, "A Meta-Learning Failure Predictor for Blue Gene/L Systems", *Proc. of ICPP07, 2007.***

Evaluation: Dynamic Meta-Learning

- RAS logs from production systems at national labs and supercomputing centers
- Cray XT4 at ORNL (7,832 XT4 compute nodes)
- Blue Gene/L at SDSC (3,072 nodes)
- Blue Gene/L at ANL (1,024 nodes)
- Blue Gene/P at ANL (40,960 nodes)
  - Note: each node contains two to four cores
- Huge volume of data!

  Over 100+GB data

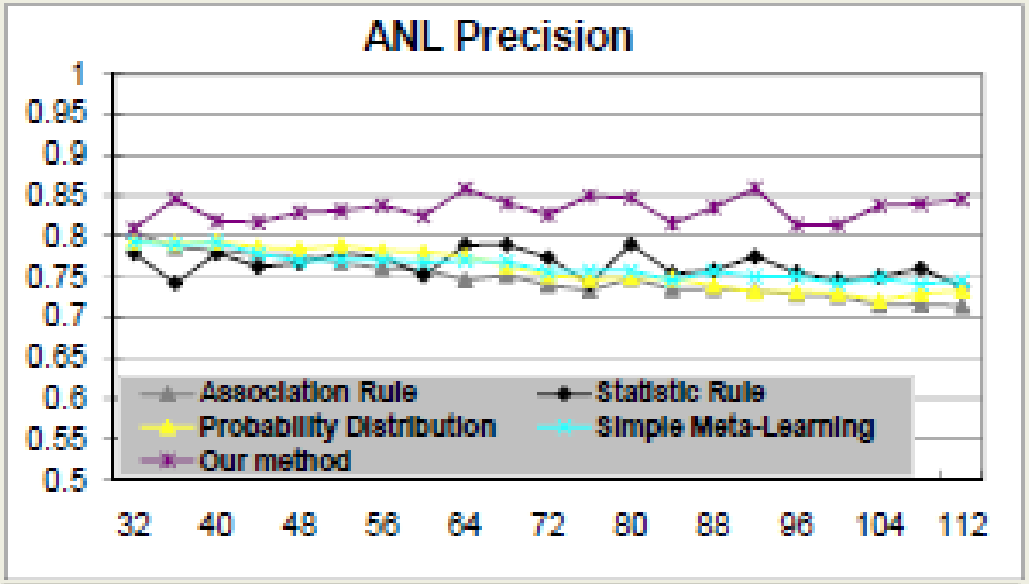
Evaluation metrics

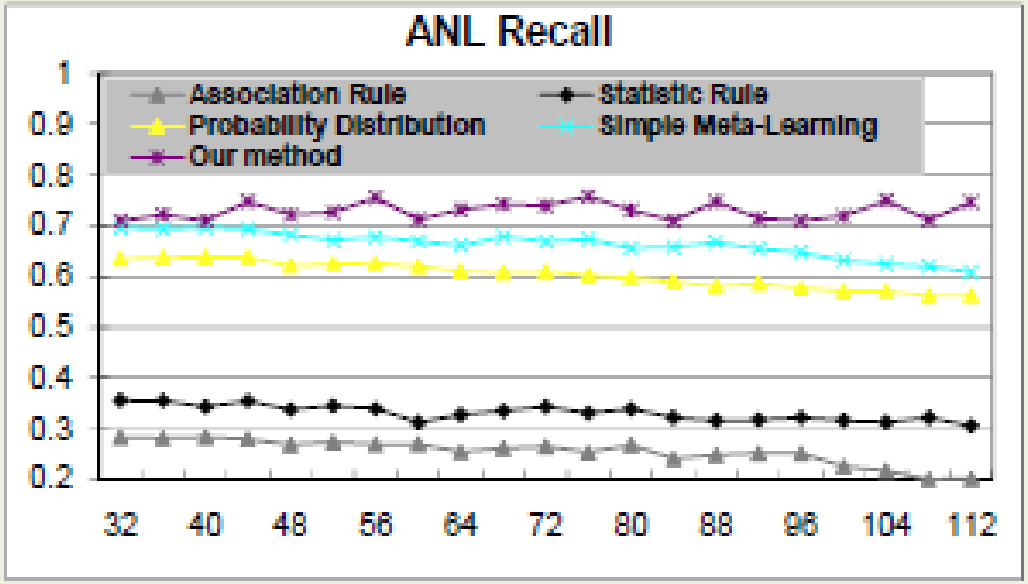
- **Precision**: A measure of the ability of a system to present only relevant items.
- **Recall**: a measure of the ability of a system to present all relevant items.

	Predicted Negative	Predicted Positive
Negative Cases	Tn	Fp
Positive Cases	Fn	Tp

TN / True Negative: case was negative and predicted negative
TP / True Positive: case was positive and predicted positive
FN / False Negative: case was positive but predicted negative
FP / False Positive: case was negative but predicted positive

$$\text{Precision} = \frac{tp}{tp + fp} \qquad \text{Recall} = \frac{tp}{tp + fn}$$





- Precision is between 0.8-0.9, meaning less than 20% false alarms
- Recall is between 0.7-0.8, meaning being capable of capturing over 70% failures

$$precision = \frac{T_p}{T_p + F_p}$$

$$recall = \frac{T_p}{T_p + F_n}$$

Adaptive Management

- Runtime adaptation:
  - SKIP, to remove unnecessary overhead
  - CHECKPOINT, to mitigate the recovery cost in case of unpredictable failures
  - MIGRATION, to avoid anticipated failures
- Challenges:
  - Imperfect prediction
  - Overhead/Benefit of different actions
  - The availability of spare resources

Z. Lan and Y. Li, "Adaptive Fault Management of Parallel Applications for High Performance Computing", *IEEE Trans. on Computers*, regular paper (14 pages), 2008.

Fault-Aware Rescheduling

- Focus on re-allocating *active jobs (i.e., running jobs)* to avoid imminent failures
  - A dynamic, non-intrusive strategy to *allocate spare nodes* for failure prevention
  - A general 0-1 Knapsack model to *select active*

To determine a binary vector $X = \{x_i \mid 1 \le i \le J_s\}$ such that

maximize $\sum_{1 \le i \le J_s} x_i \cdot v_i$, $x_i = 0$ *or* 1

s.t. $\sum_{1 \le i \le J_s} x_i \cdot p_i^s \le S$

Y. Li, Z. Lan, P. Gujrati, and X. Sun, "Fault-Aware Runtime Strategies for High Performance Computing", *IEEE Trans. on Parallel and Distributed Systems*, regular paper (14 pages), 2008.

Research Highlights

Pre-failure prediction & tolerance

Take precaution action based on failure forecasting

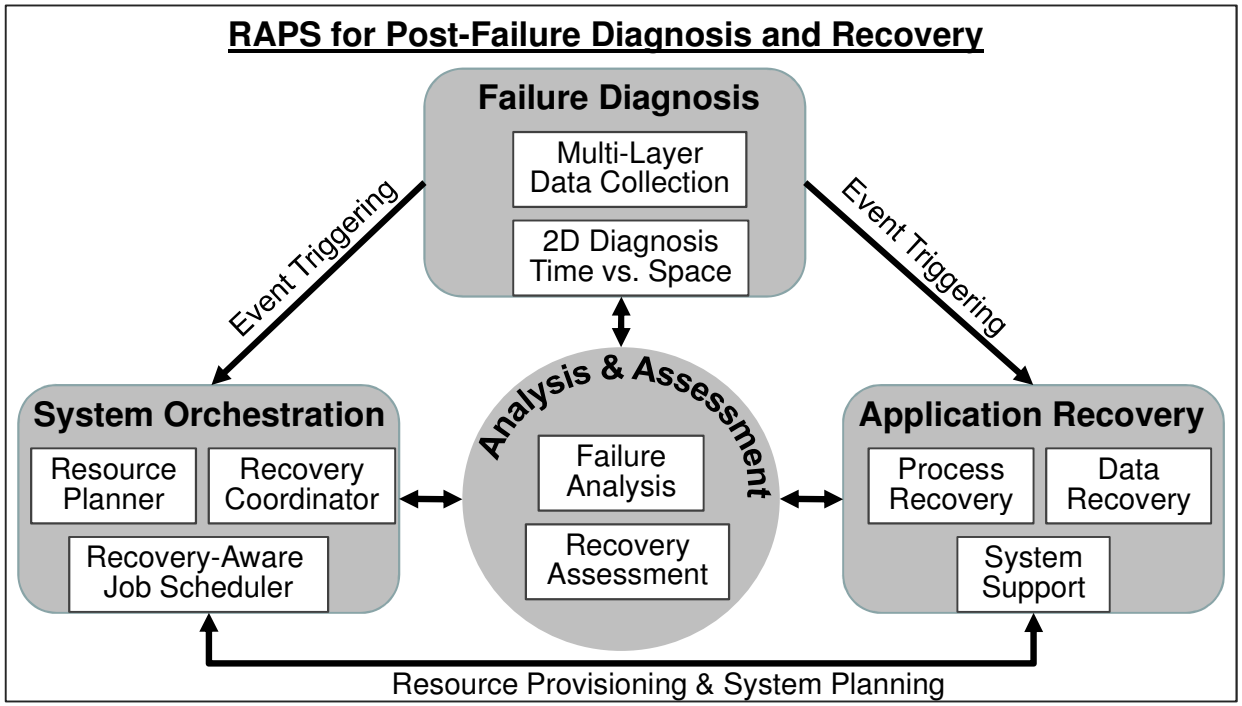
Post-failure diagnosis & recovery

Quickly resume computing after failure occurrence

Post-Failure Diagnosis & Recovery

- **Relying on pre-failure prediction and tolerance alone is insufficient!**
  - Despite progress on failure prediction, unexpected failures occur in Practice
- Little attention has been paid to post-failure diagnosis and Recovery
  - Manual troubleshooting
  - Manual job resubmission
  - Inefficient application recovery
  - No systematic study

RAPS Overview



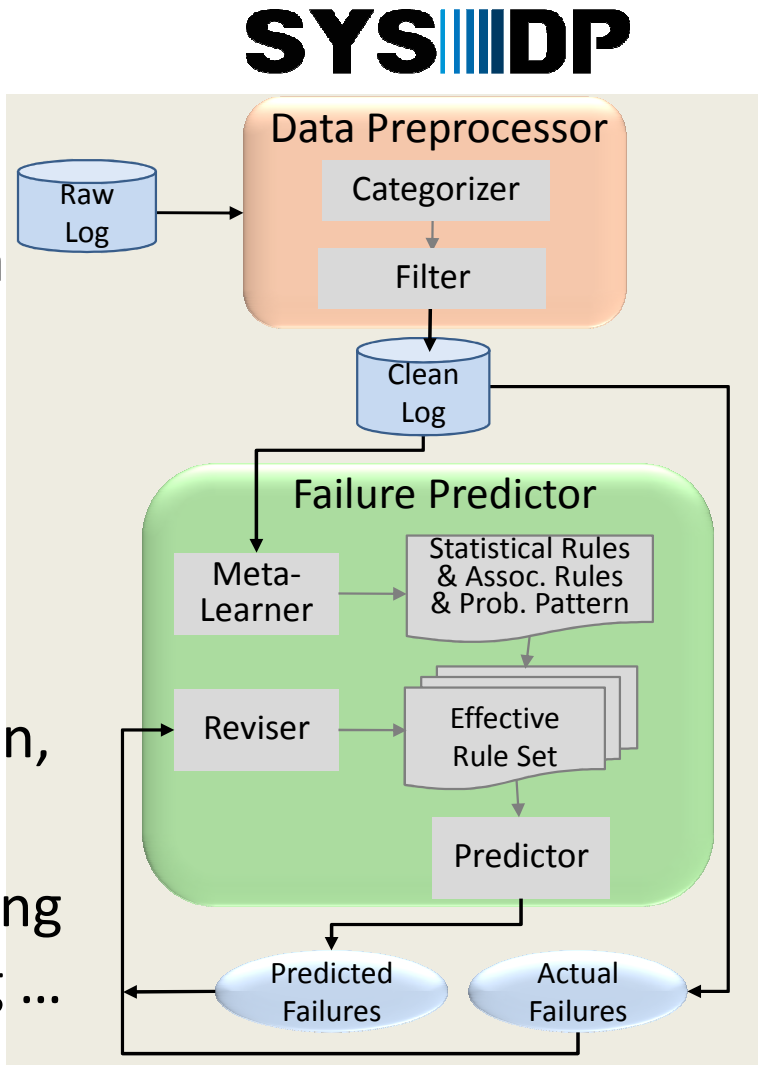
- RAPS: Recovery Aware Parallel computing Systems
  - Algorithm design
  - System development
  - Empirical study

Research Issues

- Fast failure diagnosis
  - Explore data p mining/pattern recognition techniques
- System orchestration
  - Integrate resource manager, job scheduler, and various recovery techniques
- Fast application recovery
  - Develop user-transparent support to improve process and data recovery of parallel applications
- Analysis and assessment

Failure Diagnosis

- Challenges:
  - Overwhelming amount of data
  - Redundant & unformatted data
  - Faults are many and complex
  - System changes are common
- Our prototype tool
  - Data mining, pattern recognition, statistical learning, …
  - Preprocessing, ensemble learning techniques, dynamic relearning …



Fast Process Recovery

- Currently, a restart requires the entire checkpoint image before it can proceed
  - Substantial restart latency in networked environments
  - Network transmission and I/O operation time
- Insatiable data demand from applications leads to larger checkpoint size
  - Thus, longer restart latency

- Our solution: FREM (Fast REstart Mechanism)
  - Overlapping process execution with checkpoint image retrieval

FREM Overiview

- Key idea:
  - To enable quick restart on a partial checkpoint image by recording the process data accesses after each checkpoint
- A prototype implementation with the BLCR tool in Linux 2.6.22
- Address various OS issues
  - Hardware bypassing, page swapping, dynamic memory usage, ...

Y. L and Z. Lan, "A Fast Recovery Mechanism for Checkpointing in Networked Environments", *Proc. of DSN08*, 2008.

Summary

- Our research addresses real problems in real systems, with the goal to improve performance and reliability
- Pre-failure prediction and proactive approaches
- Post-failure diagnosis and recovery