Proactive Fault Tolerance Using Preemptive Migration

Christian Engelmann, Geoffroy R. Vallée, Thomas Naughton, and Stephen L. Scott

Computer Science and Mathematics Division
Oak Ridge National Laboratory
Motivation

• The 1PFlop/s \((10^{15} \text{ Floating Point Operations Per Second})\) barrier has been broken
  – #1: LANL Roadrunner with 129,600 processor cores
  – #2: ORNL Jaguar with 150,152 processor cores

• Other large-scale systems exist
  – LLNL @ 212,992, ANL @ 163,840, TACC @ 62,976

• The trend is toward larger-scale systems
  – ORNL ~300,000, LLNL~ 2,000,000

• Significant increase in component count and complexity

• Expected matching increase in failure frequency

• Checkpoint/restart is becoming less and less efficient
Summary of Application Reliability as Measured from System Data Across 21 Los Alamos Platforms (2006)

Checkpoint Efficiency $= \frac{\text{Solve Time}}{\text{Run Time}} = \frac{T_s}{T_r}$

Checkpoint Interval $= \frac{T_c}{MTT}$

$90\%$ Design Point: $\delta \leq \frac{M}{200}$

$D = \sqrt{2\Delta M}$

$R / M$
- 0.0
- 0.1
- 0.2
- 0.3
- 0.4
- 0.5
- 0.6
- 0.7
- 0.8
- 0.9
- 1.0
- 1.1
- 1.2
- 1.3
- 1.4
- 1.5

Reactive vs. Proactive Fault Tolerance

- Reactive fault tolerance
  - Keeps parallel applications alive through recovery from experienced failures
  - Employed mechanisms react to failures
  - Examples: Checkpoint/restart, message logging/replay

- Proactive fault tolerance
  - Keeps parallel applications alive by avoiding failures through preventative measures
  - Employed mechanisms anticipate failures
  - Example: Preemptive migration
Proactive Fault Tolerance using Preemptive Migration

- Relies on a feedback-loop control mechanism
  - Application health is constantly monitored and analyzed
  - Application is reallocated to improve its health and avoid failures
  - Closed-loop control similar to dynamic load balancing

- Real-time control problem
  - Need to act in time to avoid imminent failures

- No 100% coverage
  - Not all failures can be anticipated, such as random double-bit ECC errors
Feedback-Loop Control Types

- Feedback-loop control quality of service depends on
  - Monitoring data capturing capabilities
    - Sensor types
    - Sample frequency
    - Intrusiveness
  - Monitoring data filtering mechanisms
    - Threshold triggering
    - Thrend filtering
  - Monitoring data analysis techniques
    - Reliability analysis
    - Correlation in space (across nodes) and time (across app. runs)

- Four distinct types can be derived
  - Type 1-4
Type 1 Feedback-Loop Control Architecture
Type 1 Feedback-Loop Control Properties

• Alert-driven coverage for basic failures
  – Fan fault, overheating and other precursors to hard errors

• No evaluation of application health history or context
  – Prone to false positives
    • Executing unnecessary migration(s)
  – Prone to false negatives
    • Not executing necessary migration(s)
  – Prone to miss real-time window
    • Application fails during/before migration
  – Prone to decrease application health through migration
    • Migration to already unhealthy nodes
  – No correlation of health context (space) or history (time)
    • Resulting in false positives and false negatives
Type 2 Feedback-Loop Control Architecture
Type 2 Feedback-Loop Control Properties

- Trend-driven coverage for basic failures
  - Fan fault, overheating and other precursors to hard errors
  - Less prone to false positives
  - Less prone to false negatives

- No evaluation of application reliability
  - Prone to miss real-time window
  - Prone to decrease application health through migration
  - No correlation of health context (space) or history (time)
Type 3 Feedback-Loop Control Architecture
Type 3 Feedback-Loop Control Properties

- Reliability-driven coverage of failures
  - Basic and correlated failures
  - Even less prone to false positives
  - Even less prone to false negatives
  - Able to maintain real-time window
  - Not prone to decrease application health through migration
  - Correlation of short-term health context and history

- No correlation of long-term health context or history
  - Unable to match system and application reliability patterns
Type 4 Feedback-Loop Control Architecture
Type 4 Feedback-Loop Control Properties

- Reliability-driven coverage of failures and anomalies
  - Basic and correlated failures, anomaly detection
  - Even less prone to false positives
  - Even less prone to false negatives
  - Able to maintain real-time window
  - Not prone to decrease application health through migration
  - Correlation of short and long-term health context and history
Feedback-Loop Control Types Summary

• Type 1
  – Alert-driven coverage for basic failures
  – Monitor, resource manager, runtime environment

• Type 2
  – Trend-driven coverage for basic failures
  – Additional filter

• Type 3
  – Reliability-driven coverage of failures
  – Additional reliability analysis

• Type 4
  – Reliability-driven coverage of failures and anomalies
  – Additional history database
Related Work

- **System monitoring**
  - OpenIPMI, Ganglia, OVIS 2 and MRNet

- **System log and reliability analysis**
  - USENIX Computer Failure Data Repository (CFDR)
  - hPREFECTs and Sisyphus

- **Transparent migration mechanisms**
  - Xen, BLCR, AMPI, MPI-Mitten

- **Type 1 proactive fault tolerance frameworks**
  - Xen + Ganglia

- **Combining proactive and reactive fault tolerance**
  - Policy analysis using simulation and failure logs
Challenges Ahead

- Health monitoring
  - Identifying deteriorating applications and OS conditions
  - Coverage of application failures: Bugs, resource exhaustion

- Reliability analysis
  - Performability analysis to provide extended coverage

- Scalable data aggregation and processing
  - Key to timeliness in the feedback control loop

- Need for standardized metrics and interfaces
  - System MTTF/MTTR != Application MTTF/MTTR
  - System availability != Application efficiency
  - Monitoring and logging is system/vendor dependent
Questions?