Fault Tolerance and Recovery of Scientific Workflows on Computational Grids

Gopi Kandaswamy
Anirban Mandal
Daniel A. Reed

Resilience’08, May 22, 2008
Presentation Outline

• Motivation and rationale
• Fault tolerance and recovery service
• Algorithms
  – migration, over-provisioning
• Evaluation with LEAD
• Conclusion
Motivation

• Reliability and performance are related
  – failure is the limiting case of poor performance
  – both involve measures of behavior over time

• Large, complex workflows are sensitive to failures
  – faults are the norm, rather than exceptions
  • distributed systems, services and resources
  – completion “guarantees” are problematic
    • workflow completion is probabilistic in the presence of faults

• Many time-critical workflows are deadline driven
  – severe weather events, disaster response, …
Fault Tolerance/Recovery Service (FTR)

- Application Performance Models
- Resource Reliability Models
- Network Latency and Bandwidth
- Batch Queue Wait Times
- Deadline & Success Probability
- Workflow Step (Application)
- Service Availability Statistics

OVP / MIG
FTR Service

• Resource models
  – MDS for static resource characteristics
  – NWS for network latency and bandwidth between resources
  – QBETS for batch queue wait time prediction on resources
  – simple reliability models

• Application models
  – based on simple parametric historical performance models

• Deadline of workflow steps

• Success probability
  – expected probabilistic completion guarantee

• Grid services’ availability
  – core middleware services like WS-GRAM, GridFTP
FTR Algorithms

• Notation
  \( p_i \): failure probability of resource \( i \) (eg. 1 hr. failure probability)
  \( h_i \): expected execution cost of application on resource \( i \)
  • expected queue wait time
  • expected computation time
  • expected communication time
  \( x \): required success probability
  \( d \): required deadline
  \( m_i \): failure probability of application (based on reliability model)

• Resource \( i \) represents (queue, #nodes) combinations
• Use a simple reliability model
  – assumption: resource failures are independent over time
  – resource failures follow a binomial distribution
Over Provisioning

• Find
  – degree and resources for over-provisioning

• Number of application copies
  – meet a deadline \( d \) with a success probability \( x \)

• Solve the following optimization problem

For given \([1..M]\) resources, find a partition \( P = \{s_1, s_2 \ldots s_n\} \) of \([1..M]\) such that

\[
1 - m_{s_1} * m_{s_2} * \ldots m_{s_n} \geq x \land |P| \text{ is minimum} \land \max \{h_{s_1} \ldots h_{s_n}\} \leq d
\]

Probability of failure

Minimum number of resources meeting deadline
Migration

• Find the best migration path
  – Best resource chain

• Solve the following optimization problem

For given \([1..M]\) resources, find a partition \(P = \{s_1, s_2 \ldots s_n\}\) such that

\[
1 - m_{s_1} \times m_{s_2} \times \ldots \times m_{s_n} \geq x \land |P| \text{ is minimum} \land \sum (h_{s_1} + \ldots + h_{s_n}) \leq d,
\]
Presentation Outline

• Motivation and rationale
• Fault tolerance and recovery service
• Algorithms
  – migration, over-provisioning
• Evaluation with LEAD
• Conclusion
LEAD – Linked Environment and Atmospheric Discovery

• Integrated scalable framework for dynamic and adaptive meso-scale weather prediction
  – computations continually steered by new weather data
  – responds to decision-driven inputs from users
  – steers remote observing technologies to optimize data collection for problem at hand
  – consists of analysis, visualization and data-mining tools

• Framework consists of
  – Teragrid resources at NCSA, UC and IU
  – weather data repositories (static and dynamic)
  – web portal for user interaction
  – http://portal.leadproject.org
Simplified Architecture: LEAD

- Application Performance Models
- Resource Reliability Models
- NWS, MDS
- BQP
- Service Availability
- Deadline & Success Probability

Fault Tolerance/Recovery Service

Portal
http://portal.leadproject.org

BPEL Workflow Engine

Application Service

Notification Service

BQP

Deadline & Success Probability

OVP/MIG
Click pencil icon
Enable FTR check box
WRF-Static running on Tungsten
WRF-Static failed on Tungsten, restarted on UC
WRF-Static finished on UC
Workflow proceeds to next application
Results: Workflow Step Failure Rate

30.84% without recovery

4.95% with recovery
Results: Workflow Failure Rate

- Without recovery: 79.39%
- With recovery: 23.03%
Weekly Statistics (past 5 months)
Conclusions

• Developed a fault tolerance and recovery (FTR) service
  – delivers reliable execution of workflows on grids
    • under deadline and success probability constraints
  – uses migration and over-provisioning techniques

• Deployed FTR with LEAD production infrastructure
  – transparent to users

• Results from LEAD workflows show
  – Reduction of application failure rate from 31% to 5%
  – Reduction of workflow failure rate from 80% to 23%

• Future work
  – accurate reliability estimates of resources
  – other fault-tolerance techniques for different workflow types
Questions?

Thank you..

http://portal.leadproject.org