Modeling Techniques Towards Resilience

Christian Engelmann
Research Staff Member
Oak Ridge National Laboratory

for

Chokchai (Box) Leangsuksun
President, ATPAC
SWEPCO Endowed Professor
Louisiana Tech University

Prepared by Narate Taerat and Nichamon Naksinehaboon

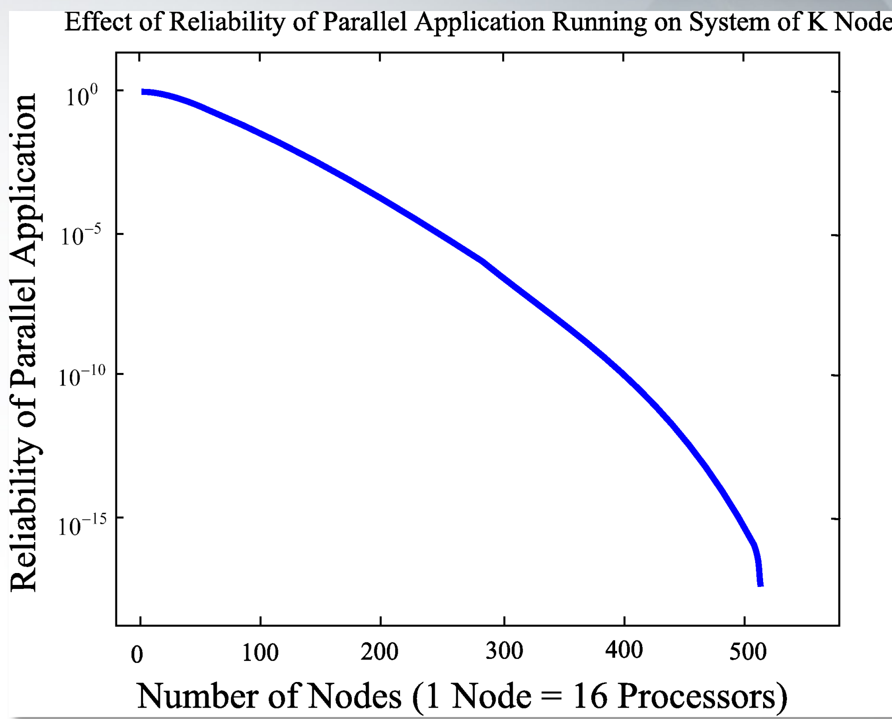
LOUISIANA TECH
UNIVERSITY

Outline

- Motivation
- Resilience and resilience modeling
- Existing models
  - Fundamental models
  - Resilience supporting models
- Conclusion

Motivation: Failures Happen (Often)

- Even the most reliable system will fail at some point
- Scaling up in components reduces a system's reliability
- The 1,000,000-node example:
  - Desired system MTTF: 12h
  - Needed node MTTF: 12,000,000h (that's 1,370 years)
- We need to learn to live with failures as they happen (often)

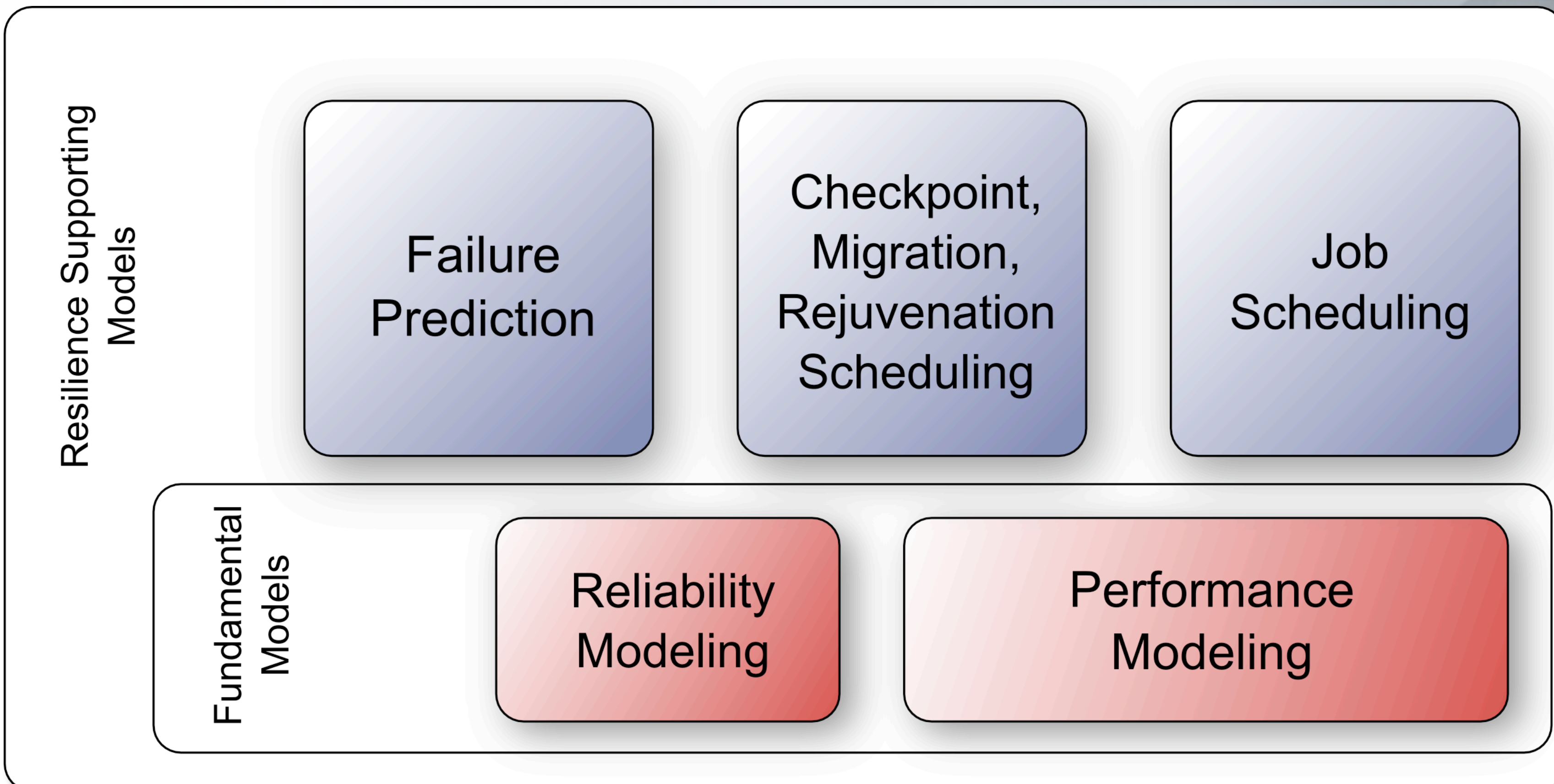


Resilience and Resilience Modeling

- Resilience:
  - Continuous, progressive computing despite failures
  - Extends beyond traditional fault tolerance approaches
- Resilience modeling:
  - Mathematical representation of system behavior
    - Failure mode, root cause, impact and probability
    - Resilience mechanism type, capabilities and impact
  - Optimization towards “resilience computing”

LOUISIANA TECH UNIVERSITY

Resilience Models



LOUISIANA TECH UNIVERSITY

Fundamental Models

- Study of failure distribution in HPC systems, e.g., Exponential vs. Weibull distributions
  - Schroeder and Gibson ’06, Y. Liu ’08
- Scalable stochastic HPC system reliability model
  - Gottumukkala ’09
- RAS models for specific HPC system components
  - Tang ’08

Resilience Supporting Models

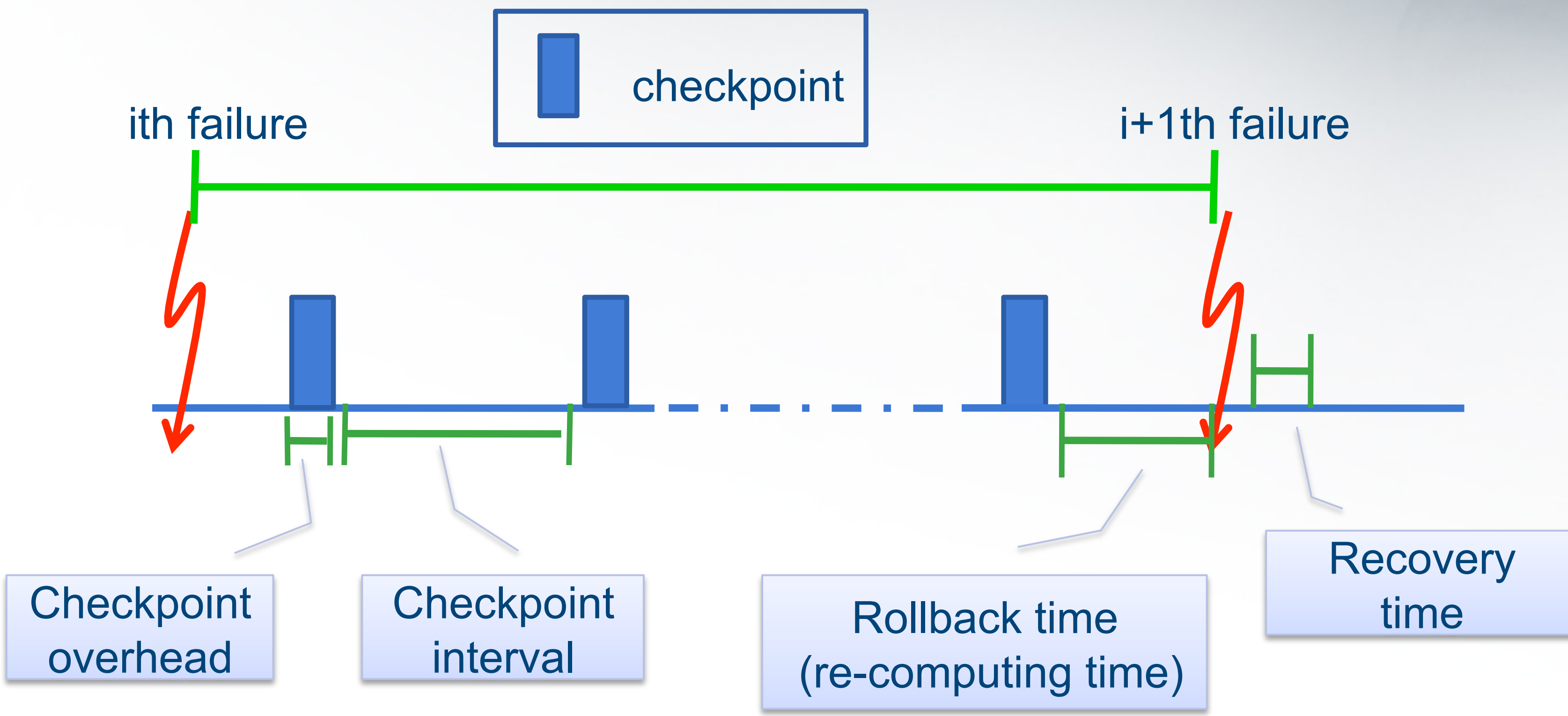
- Checkpoint scheduling
- Migration scheduling
- Rejuvenation scheduling
- Job scheduling
- Failure prediction
- Runtime decision for optimization

Checkpoint, Migration and Rejuvenation Scheduling

- Objective:
  - Reduce waste time due to failure occurrences
- Use model to:
  - Minimize mechanism overhead
  - Minimize waste time
- Important modeling factors:
  - Reliability of the entire system and of individual nodes
  - Availability of the entire system
  - Overhead induced by mechanism

LOUISIANA TECH
UNIVERSITY

Checkpoint Scheduling



Checkpoint Scheduling Models (1/2)

- Globally optimal checkpoint placement for balancing overhead and re-computing time
  - Consider the expected total waste time and expected re-computing time
  - Plank ’74, Daly ’04, Ling ’01, Osaki ’06, Y. Liu ’07
- Locally optimal checkpoint placements
  - Decide whether to perform the current/requested checkpoint or not by probability of failure occurrence
  - Oliner ’06

Checkpoint Scheduling Models (2/2)

- Checkpoint scheduling that maximizes overall system availability
  - Geist ’88, Plank ’99, Wong ’93
- Incremental checkpoint placement
  - When to perform regular/incremental checkpoints
  - Palaniswamy and Wilsey ’93, S. Yi ’06, Naksinehaboon ’08

Missing in Checkpoint Scheduling Models

- Some need better checkpoint overhead accuracy
  - Constant vs. non-constant overheads
- Some are only for single nodes or small systems
  - Effects in large-scale systems
- Some are impractical
  - Too complex mathematics
  - Too many considered factors

Migration Scheduling

- Objective:
  – Determine appropriate time to migrate process or VM
- Use model to:
  – Co-relate system behavior with failure events
  – Support proactive, preventative approaches
- Important modeling factors:
  – Reliability of the entire system and of individual nodes
  – Environmental monitoring data and system log analysis
  – Failure prediction

Migration Scheduling Models

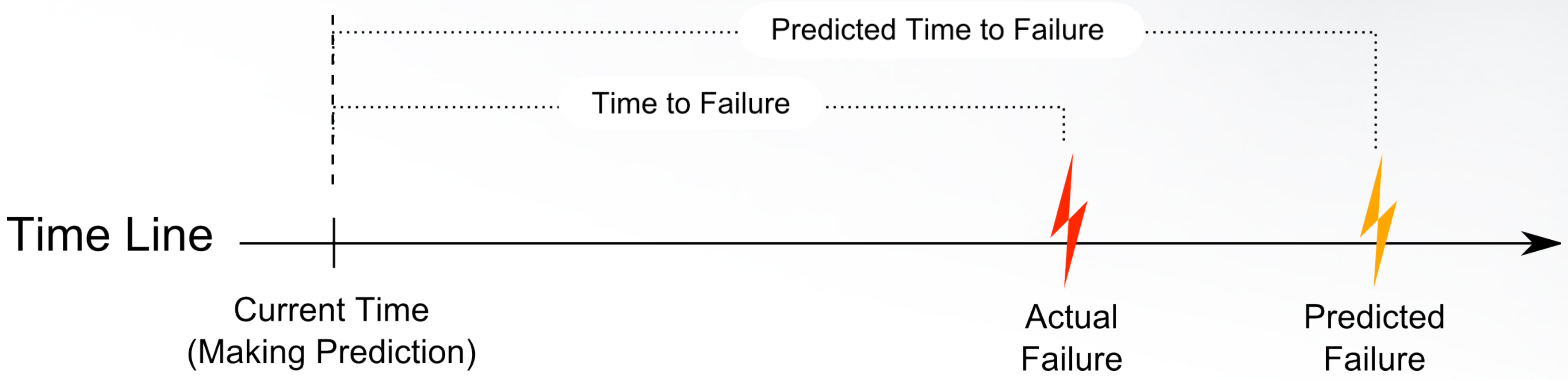
- Load balancing
  - R. Fernandes and L. Senger ‘04, M. Harchol-Balter and A. Downey ’95
- Scheduling for checkpoint-enabled systems
  - Z. Lan and Y. Li ’08
- Network-aware migration scheduling
  - A. Stage and T. Setzer ’09

Failure Prediction

- Failure:
  - Inability of an item to perform a required function
  - Item can be a component, set, or the entire system
- Goal:
  - To know a failure before hand, so that we can handle it
  - Predict ***time*** to failure
  - Predict failure emergence in some ***time interval***

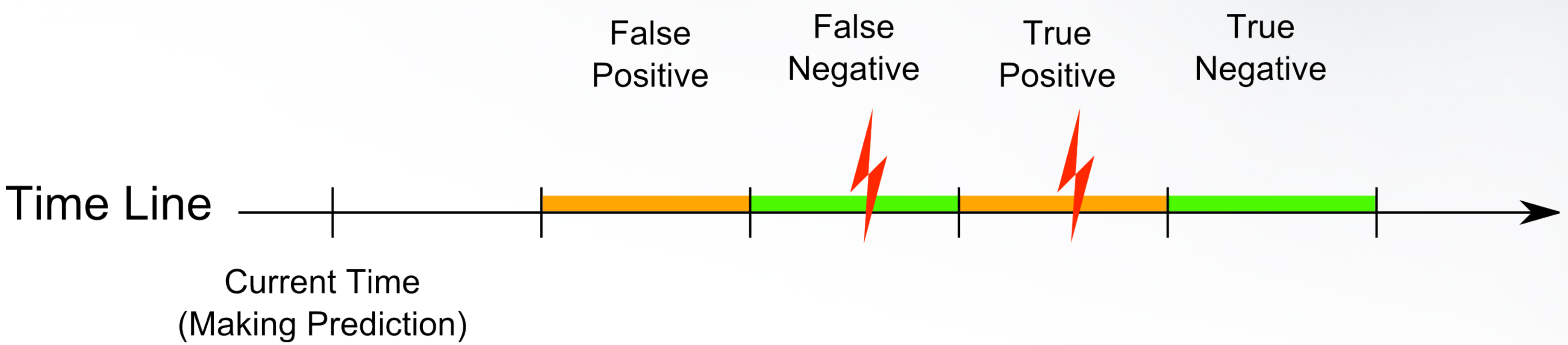
Time to Failure Prediction

(Obviously, predict time to failure)



Failure Emergence Prediction

(Predict if a failure will emerge in the time interval)



Failure Prediction Models

- Using historical failure data and other related data
- Solely based on TTF or UP/DOWN time
  - Univariate Time Series Analysis (Sahoo '03)
  - Markov Model (with two states: up and down) (Kang '07)
- Incorporating additional data, e.g. system event logs
  - Association Rule Mining (Sahoo '03)
  - Markov Model (F. Salfner '06)
  - Neural Network (Charoenpornwattana '08)
  - Naïve Bayes (Hamerly '01)

Missing in Failure Prediction Models

- System failure modeling with more factors
  - More data is needed, more accurate data is needed
    - Resource consumption: CPU, network, file I/O, …
    - Resource exhaustion: Out of memory, file I/O overload, ...
    - Hardware (IPMI) data: Temperatures, voltages, …
- More sophisticated causality analysis
- More job/application-specific modeling
- Trade off between prediction accuracy, incurred overheads and provided resiliency

Conclusion

- Existing models address only a few problems
- There are many opportunities in modeling
- How can we bring “resilience computing” to life:
  - Make runtime and applications smarter and adaptive
  - Collect more data in standardized format
  - Beware of inaccurate data: garbage in = garbage out
  - Explore more modeling techniques (e.g., adv. statistics)
- Resilience computing = modeling + mechanisms

LOUISIANA TECH UNIVERSITY