Balancing Lifetime and Soft-Error Reliability to Improve System Availability

Junlong Zhou[1*], Xiaobo Sharon Hu[2], Yue Ma[2], Tongquan Wei[1]

[1]Department of Computer Science and Technology
East China Normal University, China
[2]Department of Computer Science and Engineering
University of Notre Dame, USA
[*]Presenter

Outline

- Introduction
  - Availability
- Preliminary Knowledge
  - Permanent Fault
  - Transient Fault
- Our Method
  - Formula to Calculate the MTTF due to Transient Fault
  - Framework to Maximize System Availability
  - Heuristic Algorithm to Improve Reliability
- Simulation Setup and Results
- Summary and Future Work

Introduction to Availability

- What's availability?
  - Availability is the state if an application being accessible to the end user, when running on a specific platform.
  - Unavailability (or called outage/downtime) is the time that a system is not available to an end user.

Cost of Outage/Downtime

Financial Cost of Outage Per Hour among Various Industries

Industry	Cost ($)
Brokerage Retail	6.5
Credit Card Sales Authorization	2.6
Airline Reservation Centers	90,000
Package Shipping Services	28,250
Manufacturing Industry	26,761
Banking Industry	17,093
Transportation Industry	9,435

amazon

$4000/minute

Semiconductor Industry

intel inside

$16,667/minute

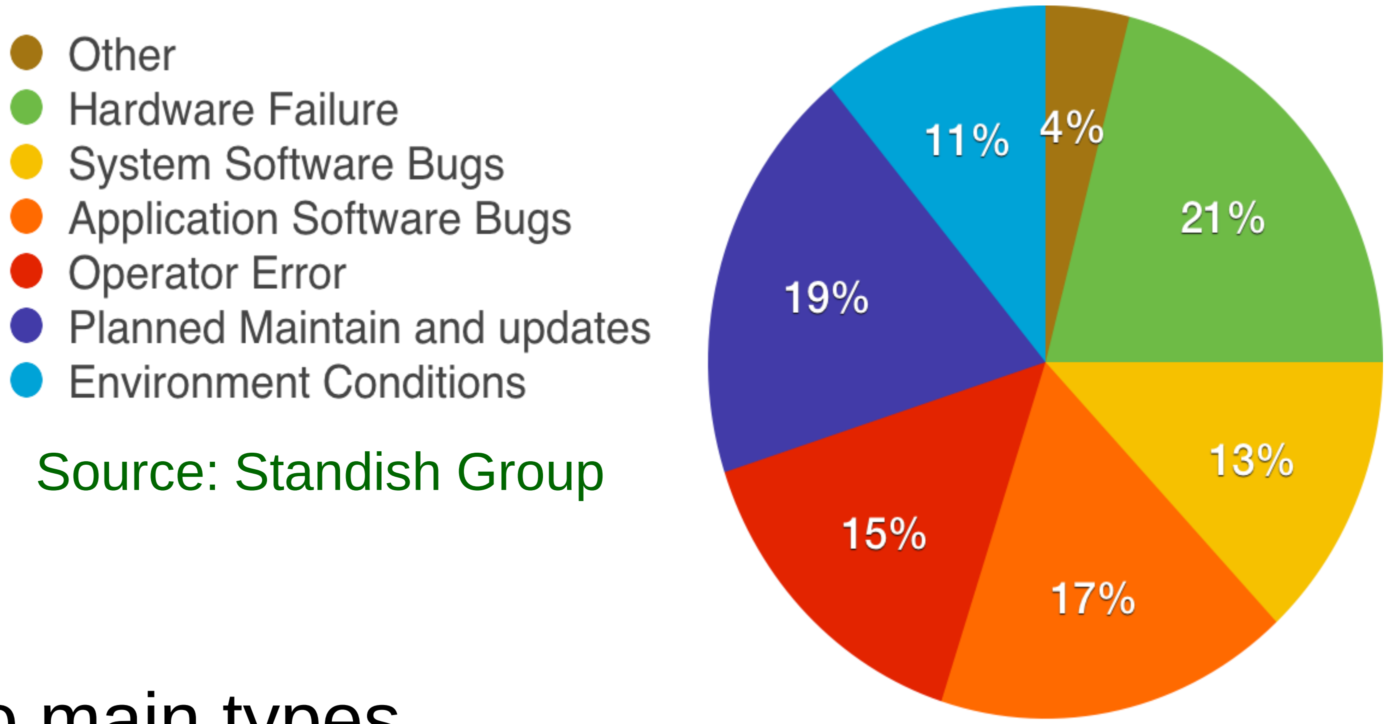
Merrill Lynch

$108,000/minute

Source: Contingency Planning Research & Strategic Research Corp.

Thus, we want to decrease outage (increase availability).

Types of Outages



Source: Standish Group

- Two main types
  - Planned outages: Planned Maintain and Updates
  - Unplanned outages: Hardware Failure + System Software Bugs + Application Software Bugs

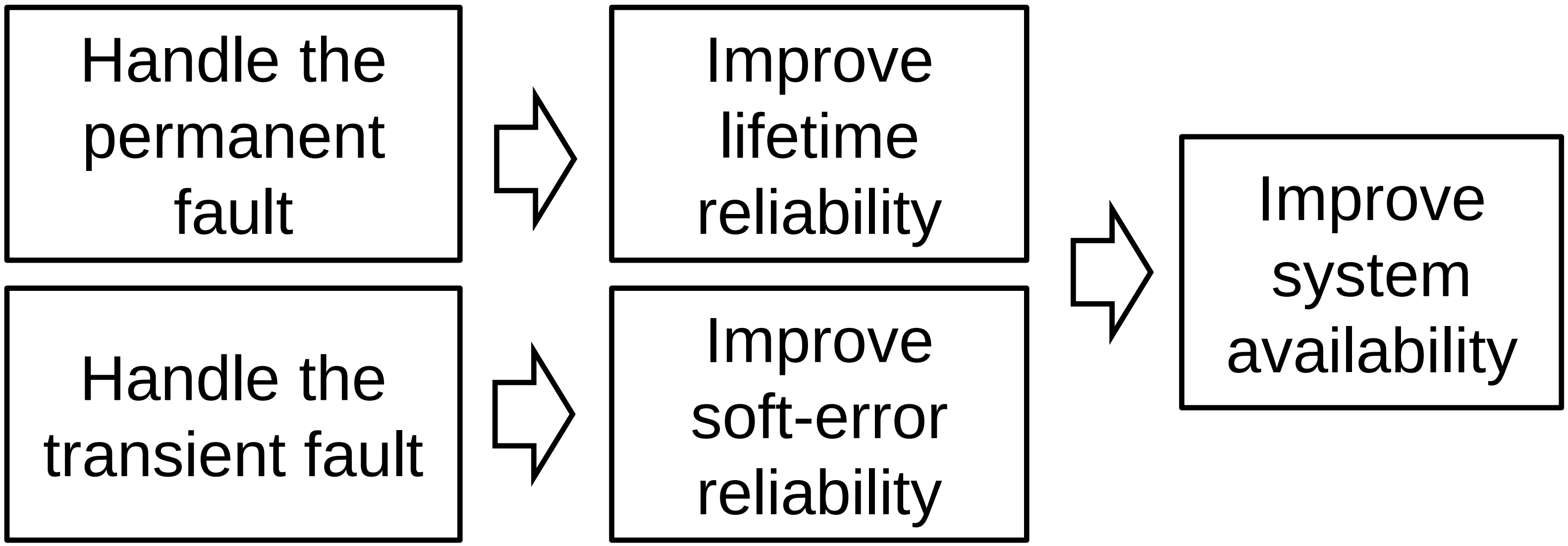
Reasons of Outages

- Planned outages (controllable)
  - Such as data base reorganization, release changes, network reconfiguration

more than 50%, important

- Unplanned outages (unexpected)
  - Such as hardware failure and software failure

Improve System Availability



Questions:

1. What are the permanent fault and transient fault?
2. How to improve the two reliabilities for achieving high availability?

Permanent Fault

- What's permanent fault? (or hard error)
  - A type of failure that continues to exist until the faulty hardware is repaired or replaced

- What causes permanent fault?
  - Four main IC-dominant failure mechanisms: EM, TDDB, SM, and TC

IC-Dominant Failure Mechanisms

- Electromigration (EM)
  - Dislocation of metal atoms caused by momentum imparted by electrical current in wires and vias
- Time-dependent dielectric breakdown (TDDB)
  - Deterioration of the gate oxide layer
- Stress migration (SM)
  - Caused by the directionally biased motion of atoms in metal wires due to mechanical stress
- Thermal cycling (TC)
  - Wear due to thermal stress induced by mismatched coefficients of thermal

IC-Dominant Failure Mechanisms

- Failure rate λ for EM, TDDB, and SM can be computed as

$$\lambda = K_1 \cdot e^{-\frac{k_2}{T}}$$

T: the temperature
K_1, K_2: temperature-independent constants

IC-Dominant Failure Mechanisms

- Number of cycles to failure N_{TC} can be computed as

$$N_{TC} = K_3 \cdot (\Delta T - \Delta T_0)^{K_4} \cdot e^{-\frac{k_5}{T_{max}}}$$

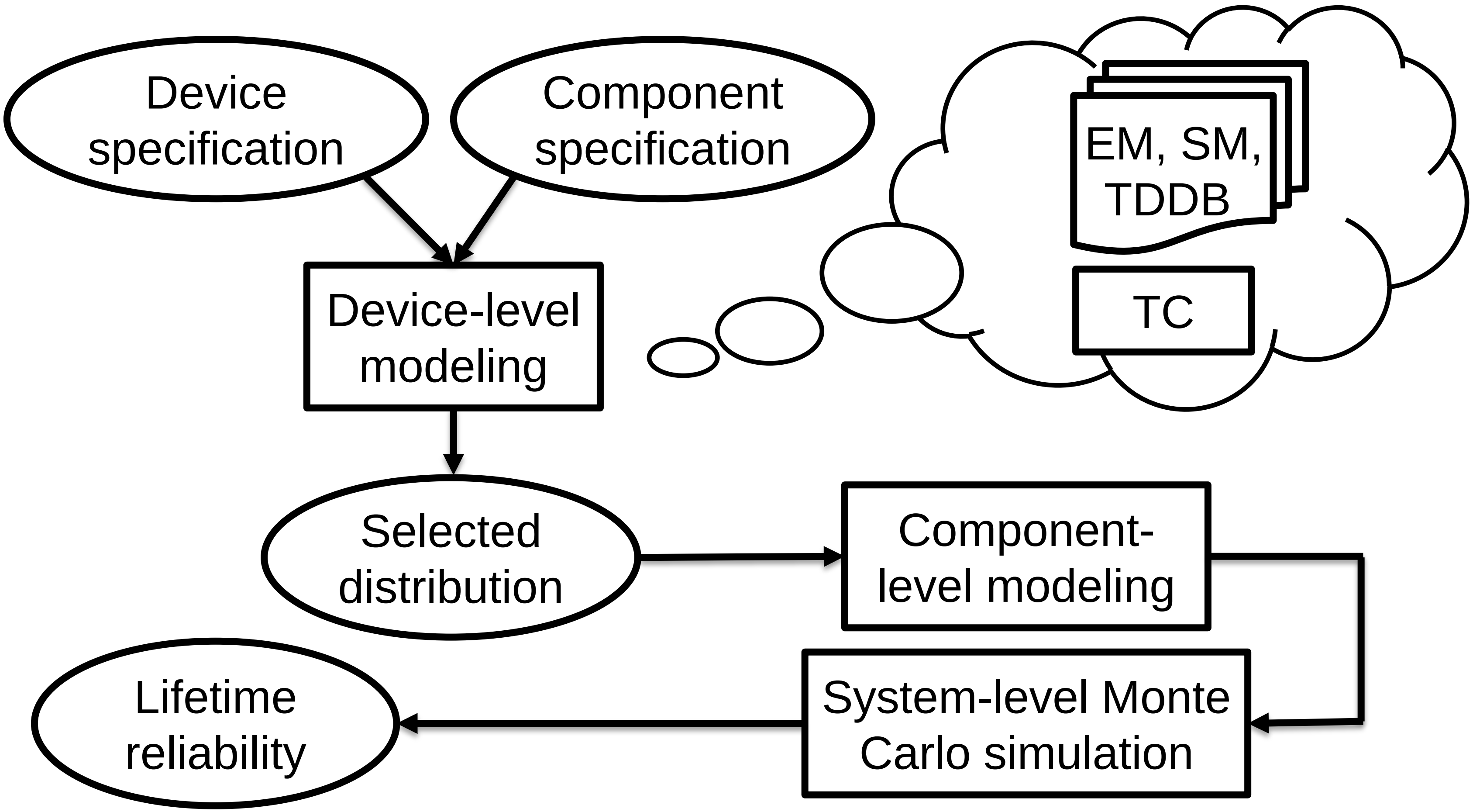
K_3, K_4, K_5: temperature-independent constants
ΔT: the thermal cycle amplitude
ΔT_0: the temperature at which inelastic deformation begins
T_{max}: the maximal temperature during the cycle

Estimate Lifetime Reliability

- Mean Time to Failure (MTTF)
  - A common metric to quantify lifetime reliability
- Xiang's tool to derive the MTTF due to permanent fault
  - Integrates three levels of models: device-, component-, and system-level models
  - Consider four IC-dominant failure mechanisms: EM, TDDB, SM, and TC
  - Output: the system-level MTTF

Xiang's tool: Y. Xiang, T. Chantem, R. P. Dick, X. Sharon Hu, L. Shang, "System-Level Reliability Modeling for MPSoCs," CODES+ISSS, 2010.

Estimate Lifetime Reliability

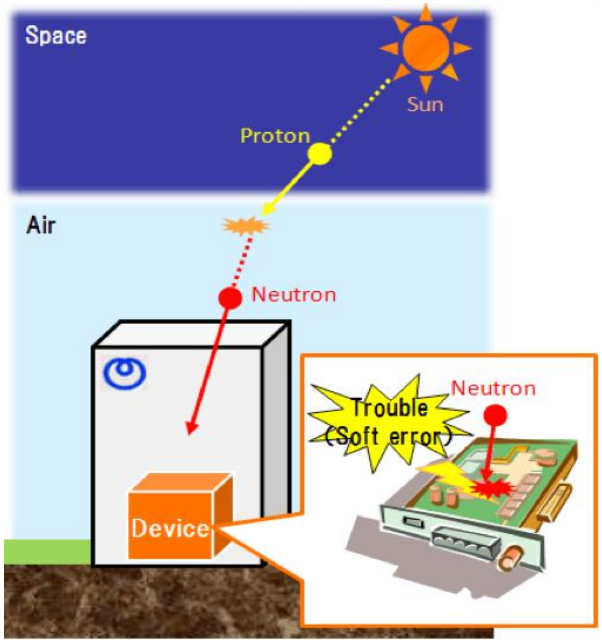


Flowchart of Xiang's tool to estimate lifetime reliability

Transient Fault

- What's transient fault? (soft error)
  - A type of failure that appears for a short time and then disappears without damage to the device

- What causes transient fault?
  - Electromagnetic interference or cosmic radiation



Transient Fault

- Soft-error reliability can be determined by the exponential failure law

$$R(f) = e^{-\lambda(f)\frac{C}{f}}$$

$\lambda(f)$: the fault rate when task τ operating at frequency f
C: the number of task cycles

A Uniform Metric

- Need a uniform metric to quantify the two reliabilities
  - User's concern: mean time to first failure, regardless of the type of failure
  - Difficult to gauge how tradeoffs should be made to achieve overall high system reliability without a uniform metric
    - Certain design decisions (e.g., task mapping and voltage scheduling) may increase lifetime reliability but decrease soft-error reliability or vice versa

Existing Reliability-Aware Methods

- Most only focus on one of the two reliability concerns
  - Lifetime reliability: e.g., Chantem et al., DATE 2013; Amrouch et al., ICCAD 2014; Duque et al., DATE 2015
  - Soft-error reliability: e.g., Li et al., ISCA 2008; Sridharan et al., ISCA 2010

- A few focus on handling permanent and transient faults simultaneously
  - E.g., Chou et al., DATE 2011, Das et al., DATE 2014: use separate metrics to perform reliability evaluation

Our Contributions

1. An analytical approach to calculate the MTTF due to transient fault
   - Enable the quantification of two reliabilities by a uniform metric

2. A single-objective optimization problem to maximize system availability
   - Consider both transient and permanent faults

3. A framework and a heuristic algorithm
   - Framework: solve the optimization problem
   - Heuristic algorithm: improve reliability for a specific scenario

System Model & Assumptions

- A uniprocessor platform
  - Supports a discrete set of frequencies
- A set of tasks repeatedly running on the processor
  - Tasks are non-preemptive and independent
  - Execution of task set in different runs are independent
- Replication to tolerate transient faults
  - Single-fault-tolerance
  - Occurrence of faults in tasks are independent
  - No fault propagation

Calculate the MTTF due to Transient Fault

The time to failure due to a transient fault of task τ_i in the kth run of task set $\mathcal{T}_n$

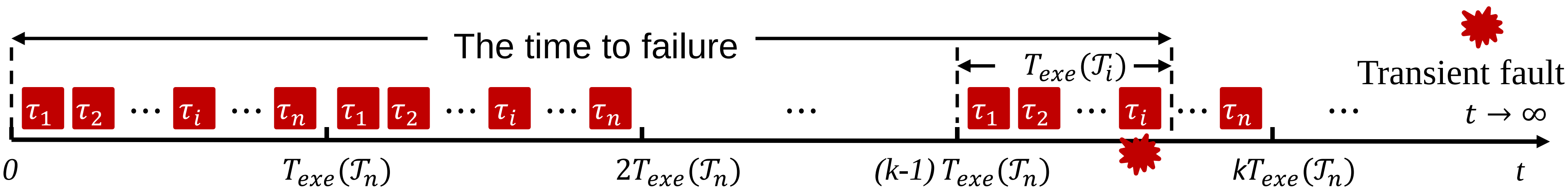


Illustration: the task set $\mathcal{T}_n = \{\tau_1, \tau_2, \cdots, \tau_n\}$ executes successfully during the prior $k - 1$ runs, but fails in the kth run due to the transient fault in task τ_i.

- $T_{exe}(\mathcal{T}_n) = \sum_{i=1}^{n} t_i$: total execution time of task set $\mathcal{T}_n$
- $T_{exe}(\mathcal{T}_i) = \sum_{j=1}^{i} t_j$: execution time of tasks τ_1 to τ_i

Calculate the MTTF due to Transient Fault

Then, the MTTF due to transient fault, denoted by $MTTF_T$, can be calculated as

$$MTTF_T = \sum_{k=1}^{\infty} \sum_{i=1}^{n} \{(k-1)T_{exe}(\mathcal{T}_n) + T_{exe}(\mathcal{T}_i)\} \cdot P_{succ}\left(\mathcal{T}_{n,k-1}\right) \cdot P_{fail}(\tau_i)$$

- $P_{succ}\left(\mathcal{T}_{n,k-1}\right)$: probability that the first $k-1$ runs of $\mathcal{T}_n$ are all successful
- $P_{fail}(\tau_i)$: probability that τ_i is erroneous but $\tau_1 - \tau_{i-1}$ in the same run of $\mathcal{T}_n$ are successful

Calculate the MTTF due to Transient Fault

Through a series of algebraic transformation, $MTTF_T$ can be derived as

$$MTTF_T = \frac{T_{exe}(\mathcal{T}_n) + T_{exp}(\mathcal{T}_n)}{P_{fail}(\mathcal{T}_n)} - T_{exe}(\mathcal{T}_n)$$

- $T_{exp}(\mathcal{T}_n) = \sum_{i=1}^{n} T_{exe}(\mathcal{T}_i) \cdot P_{fail}(\tau_i)$ denotes the expected time to failure when the fault occurs in the first run.

Correctness of Our MTTF Formulation

- The $MTTF_T$ is derived when assuming a workload $\mathcal{T}_n$ is being repeatedly executed forever.
  - What the $MTTF_T$ would be if treat two or more runs of $\mathcal{T}_n$ as the given workload being repeated forever?

Theorem 1: For any given task set $\mathcal{T}_n$ and any integer m (≥ 2), let $\mathcal{T}_n^m$ be the task set containing m runs of $\mathcal{T}_n$, i.e., $\mathcal{T}_n^m = \underbrace{\{\tau_1, \tau_2, \cdots, \tau_n, \tau_1, \tau_2, \cdots, \tau_n, \cdots\cdots, \tau_1, \tau_2, \cdots, \tau_n\}}_{n \times m}$.

Then $MTTF_T(\mathcal{T}_n^m) = MTTF(\mathcal{T}_n)$ holds.

Problem Formulation

- Definition of availability: $A = \frac{MTTF}{MTTF+MTTR}$
- For a system that may suffer from both transient and permanent faults, maximizing system availability (objective) becomes

$$\text{max:} \{\frac{MTTF_T}{MTTF_T + MTTR_T}, \frac{MTTF_P}{MTTF_P + MTTR_P}\}$$

same goal

$$\text{max:} \{\Upsilon MTTF_T, MTTF_P\}$$

equivalent to

$$\text{max min:} \{\Upsilon MTTF_T, MTTF_P\}$$

where $\Upsilon = \frac{MTTR_P}{MTTR_T}$ is assumed to be a given constant.

Four Scenarios in Our Problem

- Determine the $MTTF_T$ using our formula and the $MTTF_P$, using Xiang's tool
  - Identify which reliability dominates

- Group the relationship between $\Upsilon MTTF_T$ and $MTTF_P$ into four scenarios

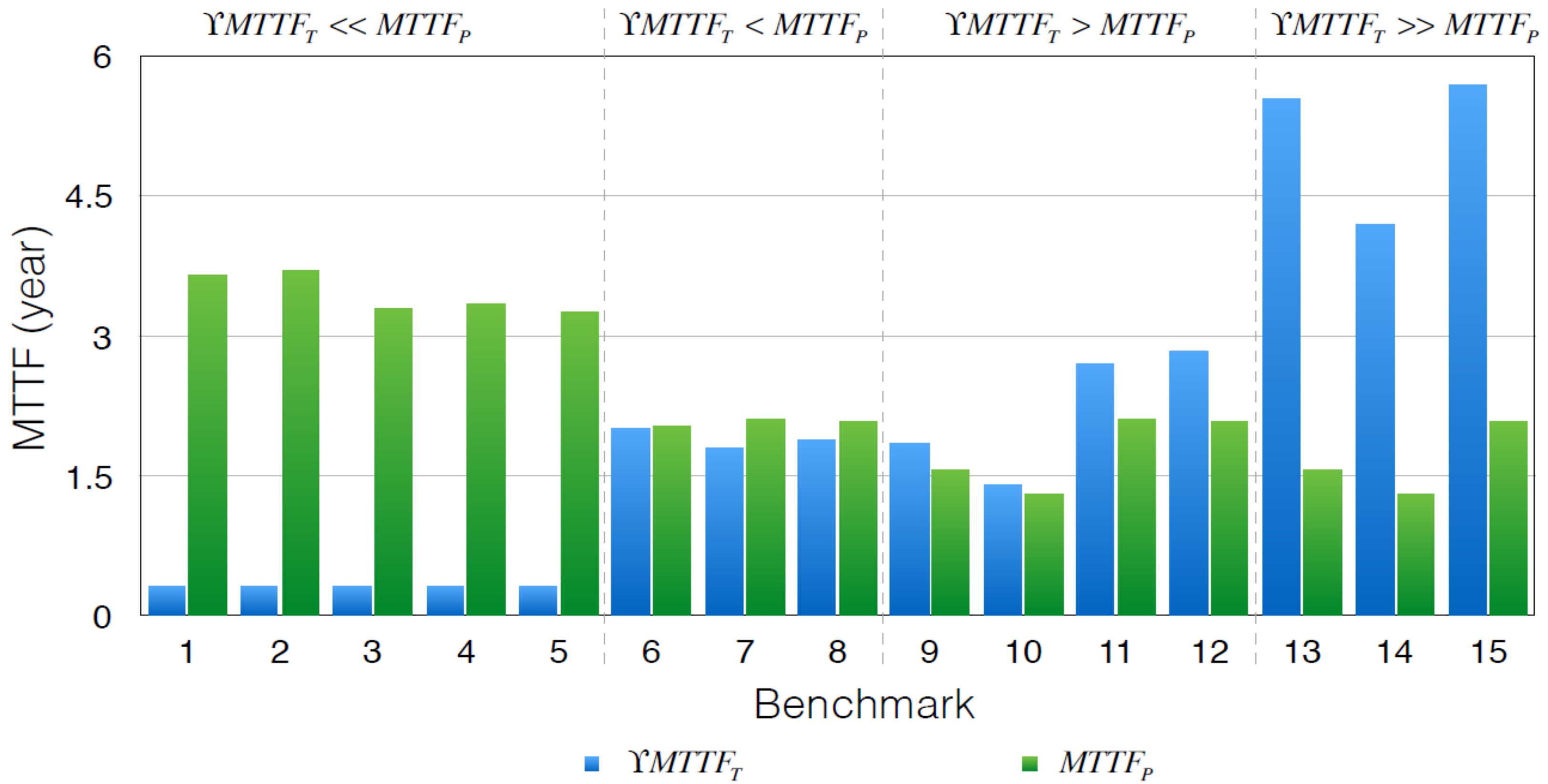
  ① $\Upsilon MTTF_T \ll MTTF_P$

  ② $\Upsilon MTTF_T < MTTF_P$

  ③ $\Upsilon MTTF_T > MTTF_P$

  ④ $\Upsilon MTTF_T \gg MTTF_P$

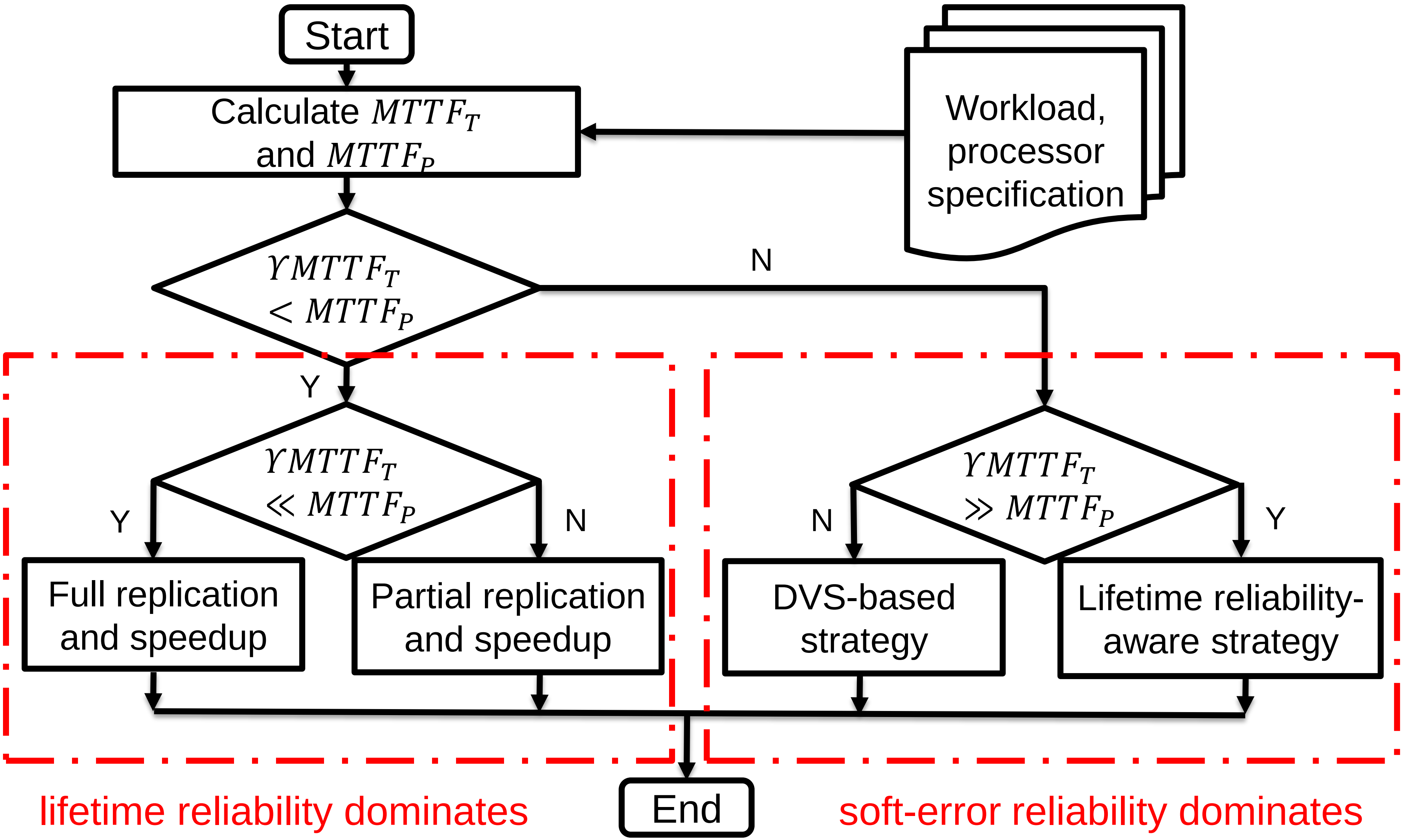
The Existence of Four Scenarios



Setup

- the same core, benchmarks, and parameter settings as in GVLSI's work and $\Upsilon = 1$

Framework to Maximize Availability



Countermeasures for Four Scenarios

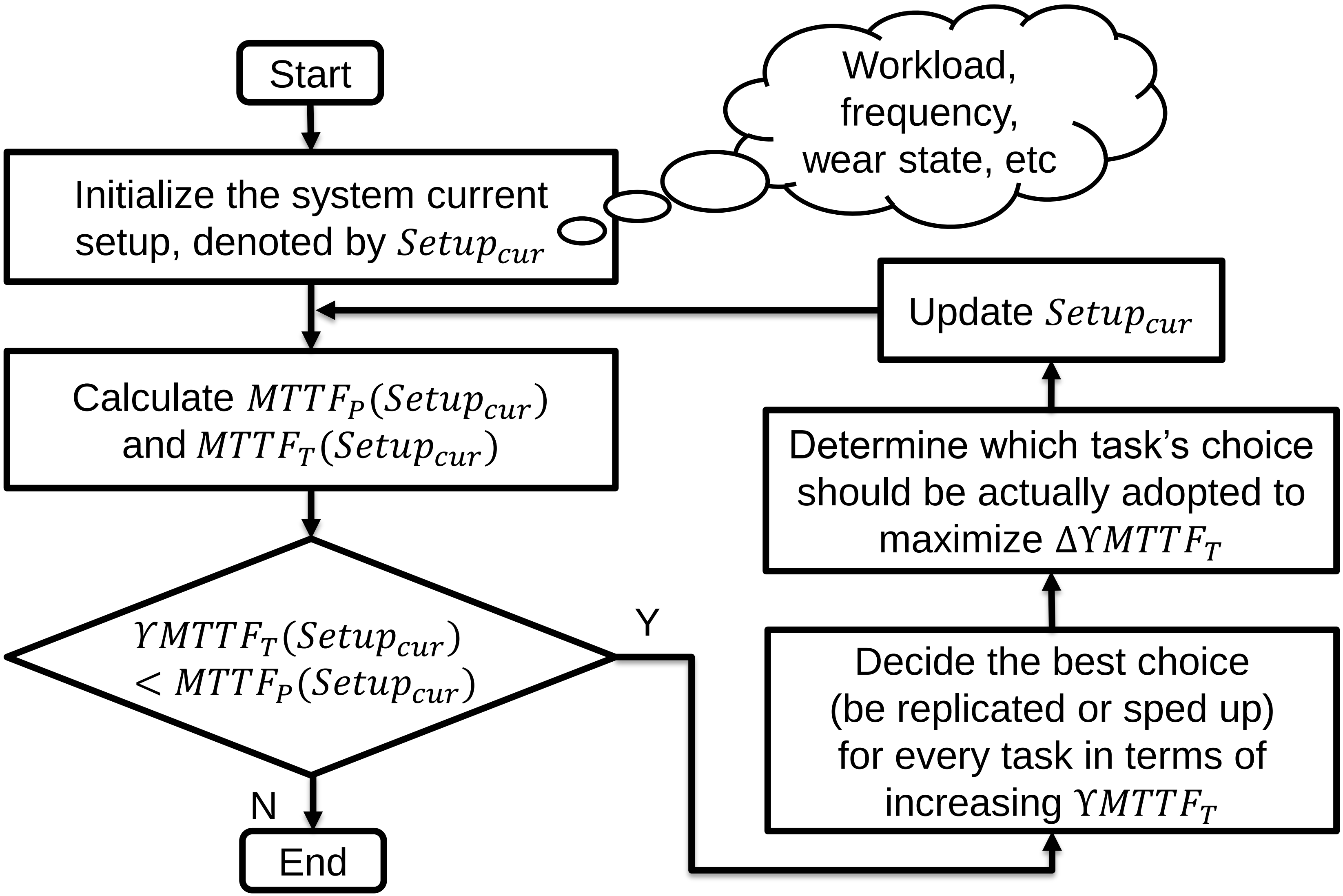
- $\Upsilon MTTF_T \ll MTTF_P$
  - Full replication and *speedup*
    - Every task with a recovery, at maximal frequency
- $\Upsilon MTTF_T < MTTF_P$
  - Partial replication and speedup
    - A part of tasks are replicated or sped up
- $\Upsilon MTTF_T > MTTF_P$
  - DVS-based strategy
    - Reduce the temperature by scaling frequency
- $\Upsilon MTTF_T \geq MTTF_P$
  - Lifetime reliability-aware strategy
    - E.g., mitigate aging speed

Our Focus

- Full replication and speedup
  - Simple and easy to implement
- DVS−based strategy
  - Widely explored
- Lifetime reliability-aware strategy
  - Widely explored
- Partial replication and speedup
  - A few works, lack a specific strategy

So Partial replication and speedup is our concern

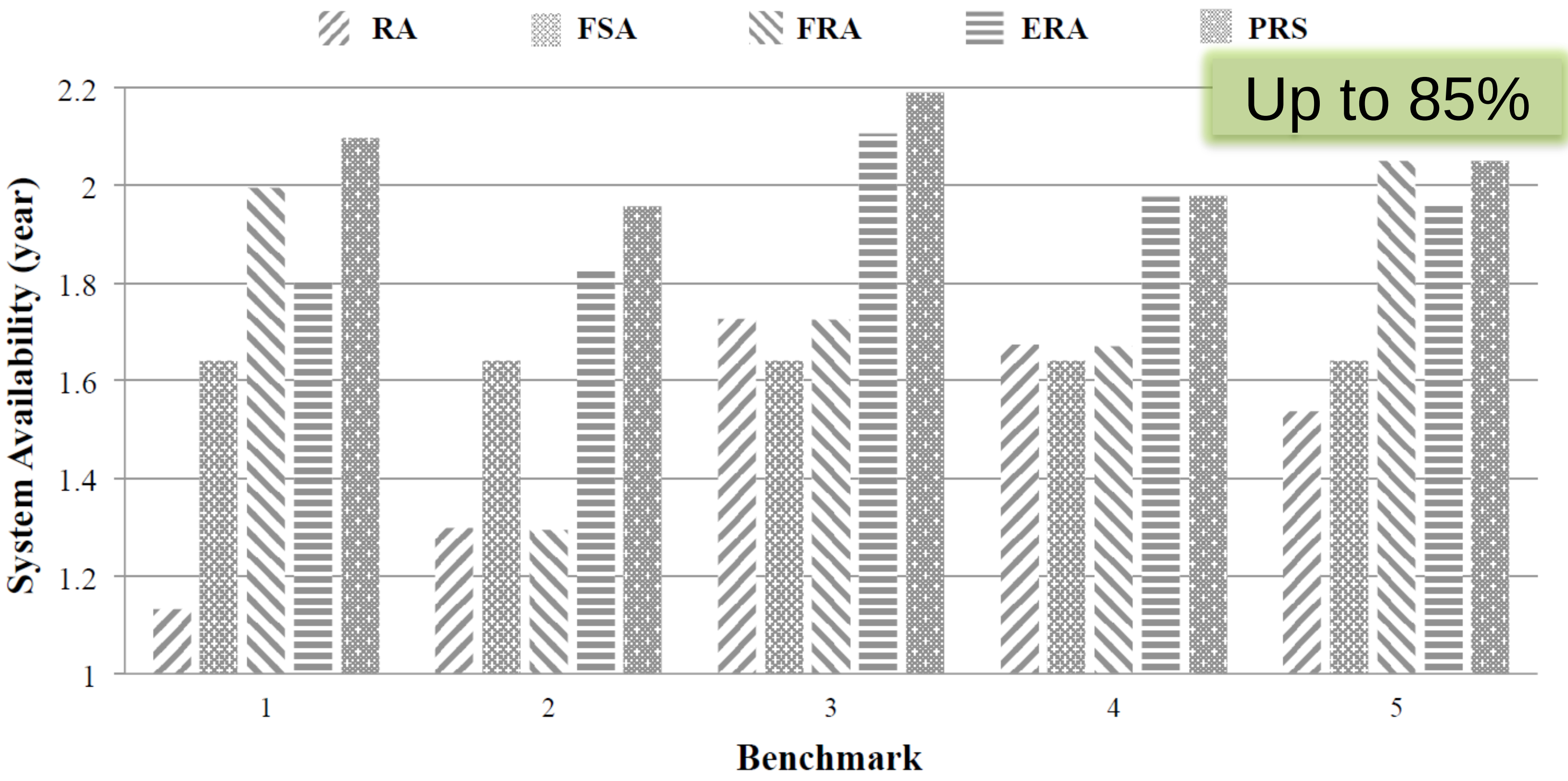
Our Heuristic Algorithm



Simulation Setup

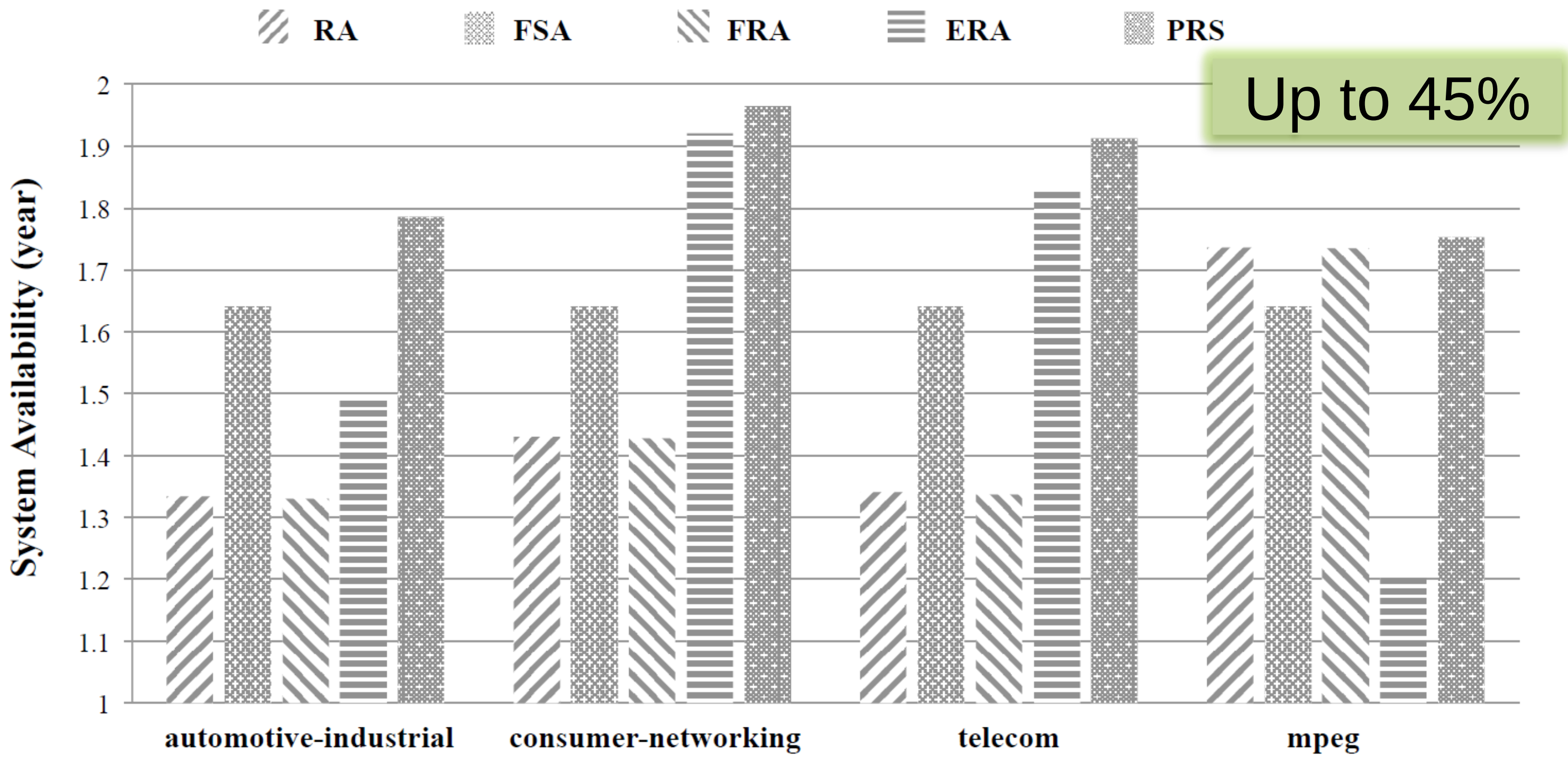
- Hardware platform
  - Alpha 21264 microprocessor, 5 frequency levels
- Synthetic, real-world app. based benchmarks
  - Five sets of 20 randomly generated tasks
  - Embedded System Benchmark Suite [Univ. of Michigan] including
    - Autom.-industrial, consumer-networking, telecom, mpeg
- Simulation tools
  - HotSpot 5.0 [Univ. of Virginia], Xiang's tool
- Algorithms used for comparison
  - Random, full speed, full replication algorithms
  - Energy-efficient and reliability-aware algorithm [ACM TAES 2013]

Simulation Results



RA: random algorithm, FSA: full speed algorithm, FRA: full replication algorithm, ERA: energy-efficient and reliability-aware algorithm, PRS: Partial replication and speedup

Simulation Results



RA: random algorithm, FSA: full speed algorithm, FRA: full replication algorithm, ERA: energy-efficient and reliability-aware algorithm, PRS: Partial replication and speedup

Summary & Future Work

- Existing reliability-aware methods lack a uniform metric to quantify lifetime and soft-error reliability

- We proposed an analytical approach to enable the evaluation of two reliabilities

- We presented a framework and a heuristic algorithm to maximize system availability
  - Our framework: designed for all scenarios
  - Our algorithm: designed for a specific scenario

- In the future, we will consider real-time systems and multicore platforms

Thank you!