

Software Managed Resiliency

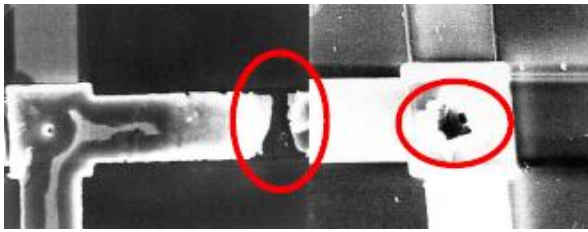
Siva Hari

**Lei Chen, Xin Fu, Pradeep Ramachandran, Swarup Sahoo,
Rob Smolenski, Sarita Adve**

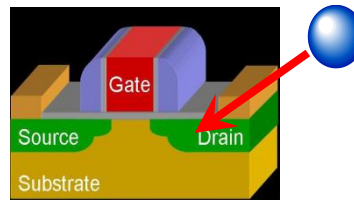
**Department of Computer Science
University of Illinois at Urbana-Champaign
swat@cs.illinois.edu**

Motivation

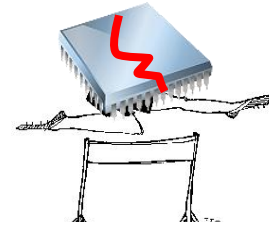
- Hardware will fail in-the-field due to several reasons



Wear-out
(Devices are weaker)



Transient errors
(High-energy particles)



Design Bugs



... and so on

⇒ **Need in-field detection, diagnosis, recovery, repair**

- Reliability problem pervasive across many markets
 - Traditional redundancy solutions (e.g., nMR) too expensive
 - ⇒ **Need low-cost solutions for multiple failure sources**
 - * **Must incur low area, performance, power overhead**

Observations

- Need handle only hardware faults that propagate to software
- Fault-free case remains common, must be optimized

⇒ Watch for software anomalies (symptoms)

- Zero to low overhead “always-on” monitors

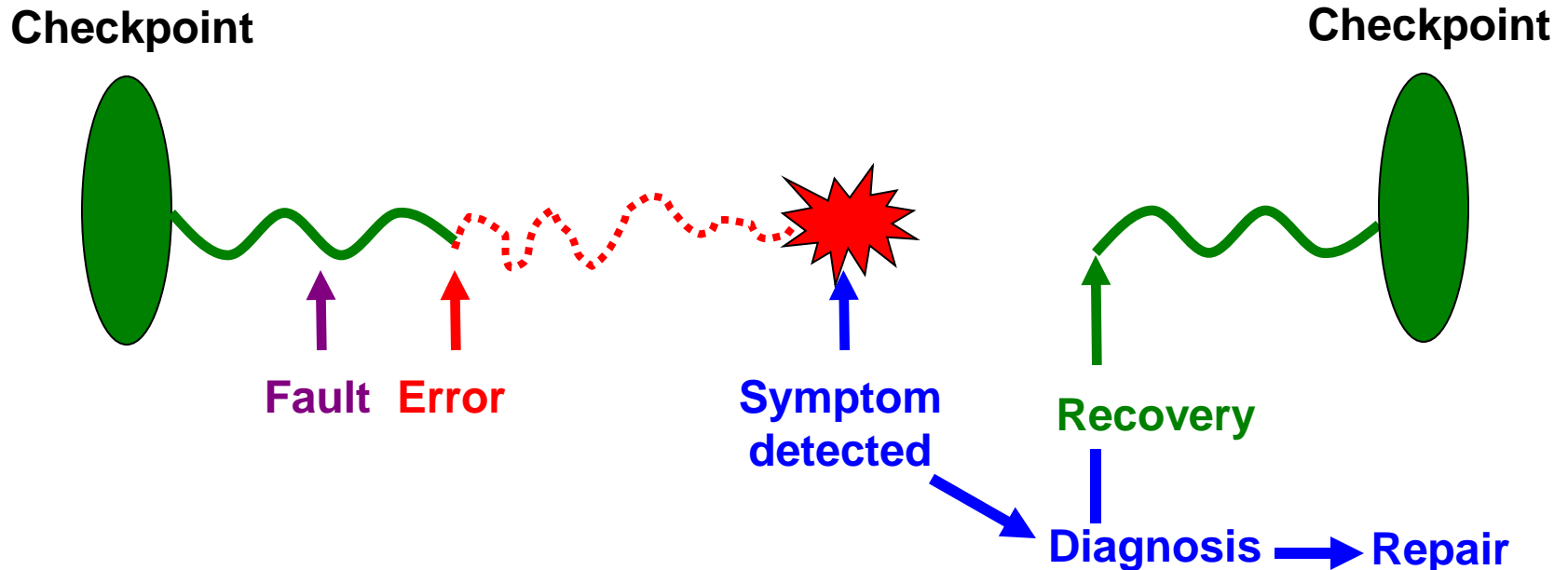
Diagnose cause after symptom detected

- May incur high overhead, but rarely invoked

⇒ **SWAT: SoftWare Anomaly Treatment**

SWAT Framework Components

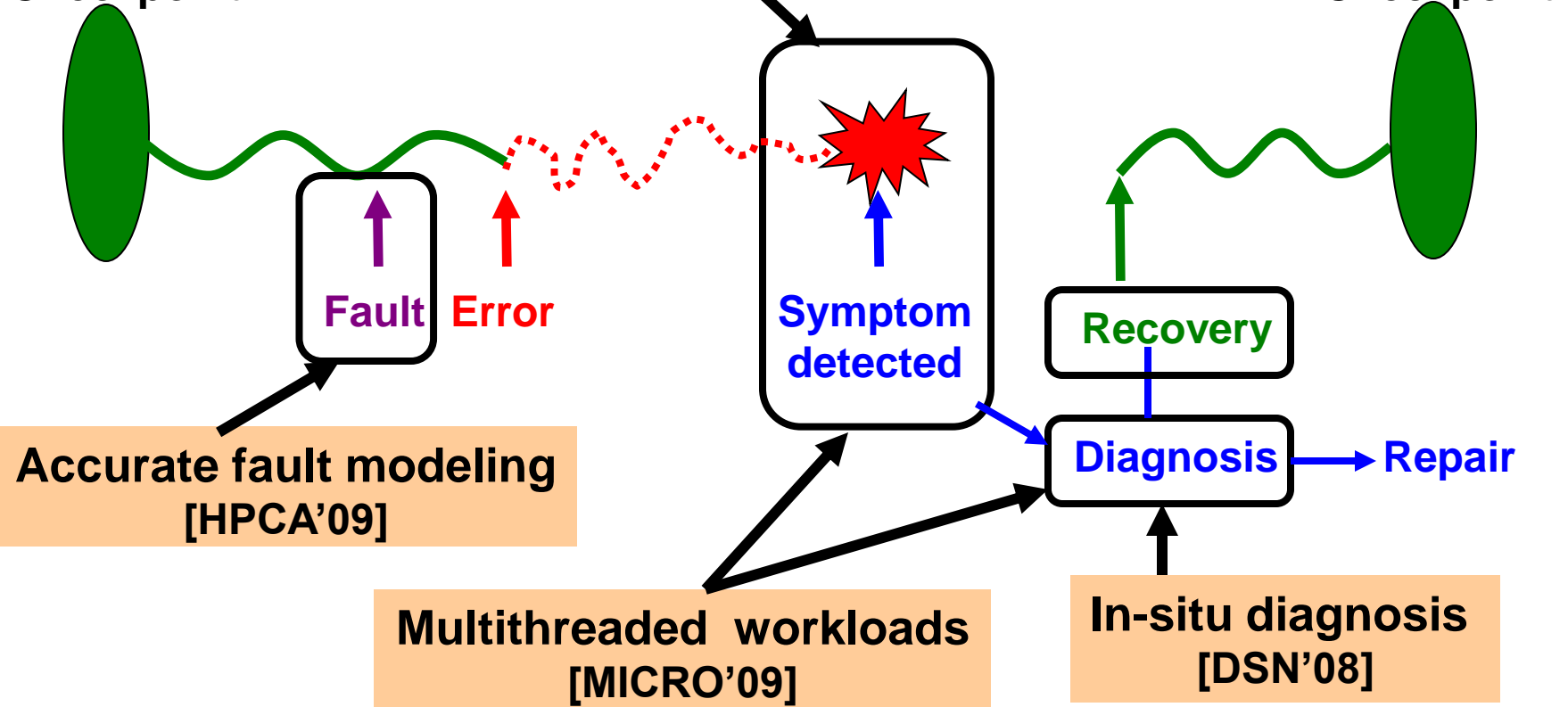
- **Detection:** Symptoms of software misbehavior
- **Diagnosis:** Rollback/replay on multicore
- **Recovery:** Checkpoint/rollback, output buffering
- **Repair/reconfiguration:** Redundant, reconfigurable hardware
- **Flexible control through firmware**



SWAT Contributions So Far

Very low-cost detectors, 99+% coverage
[ASPLOS'08, DSN'08]

Checkpoint



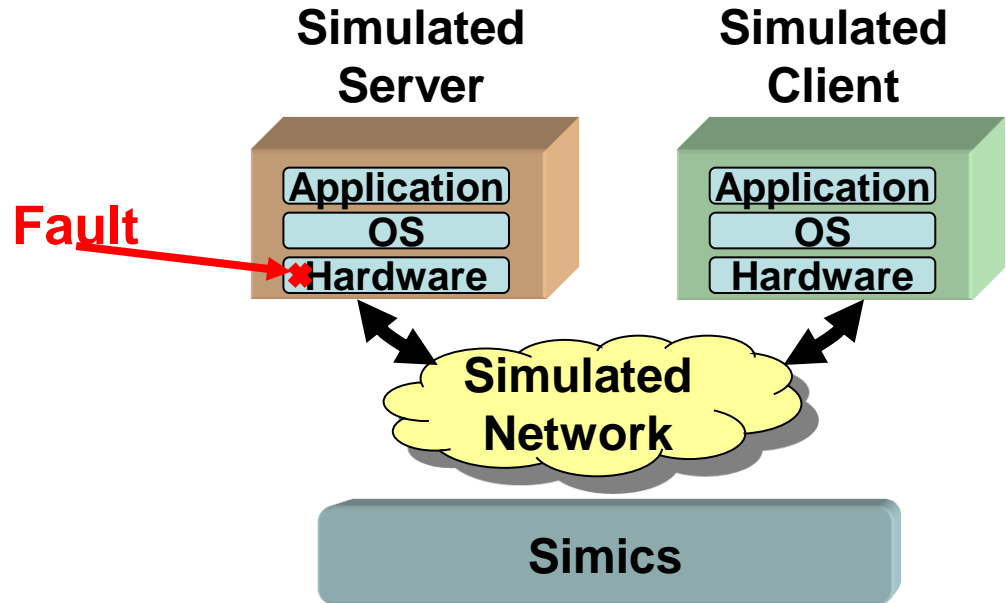
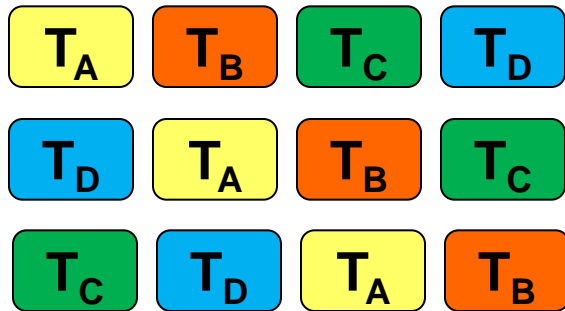
Accurate fault modeling
[HPCA'09]

Multithreaded workloads
[MICRO'09]

In-situ diagnosis
[DSN'08]

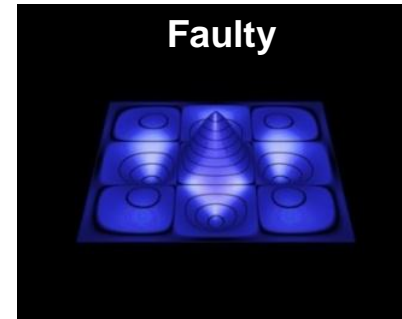
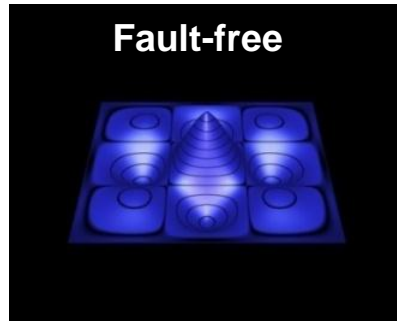
Checkpoint

SWAT on Multicore and Distributed Systems



- Off-core faults
- Client/server interactions
- Exploit safe parallel languages
 - DeNovo + Deterministic Parallel Java for safe parallelism
 - Will explore for resiliency

Application-Driven Resiliency



- **Leverage app knowledge and structure for s/w reliability**
 - **Can improve both SDCs and recovery**
E.g., stateless recovery for server threads,
transactional semantics for clients,
assertions in production code

Validation and Prototype



- **FPGA prototype with Michigan CrashTest**
 - Realistic fault models
 - Full multicore implementation with firmware

Thank You