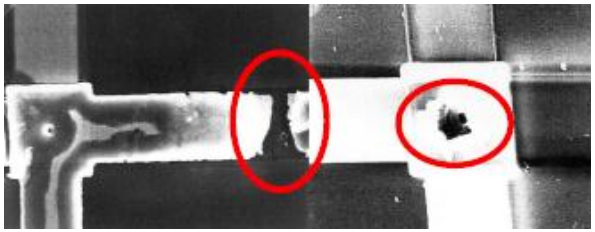
mSWAT: Low-Cost Hardware Fault Detection and Diagnosis for Multicore Systems

Siva Kumar Sastry Hari, Man-Lap (Alex) Li,
Pradeep Ramachandran, Byn Choi, Sarita Adve

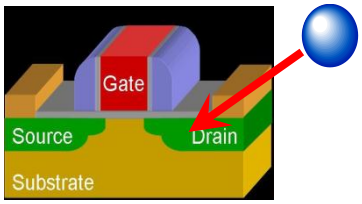
Department of Computer Science
University of Illinois at Urbana-Champaign
swat@cs.uiuc.edu

Motivation

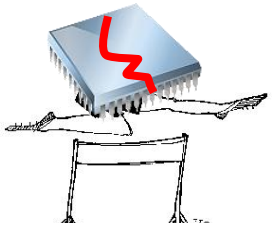
- **Hardware will fail in-the-field due to several reasons**



Wear-out
(Devices are weaker)



Transient errors
(High-energy particles)



Design Bugs

… and so on

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

- **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**

SWAT: Low-Cost Hardware Reliability

SWAT Observations

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

SWAT Approach

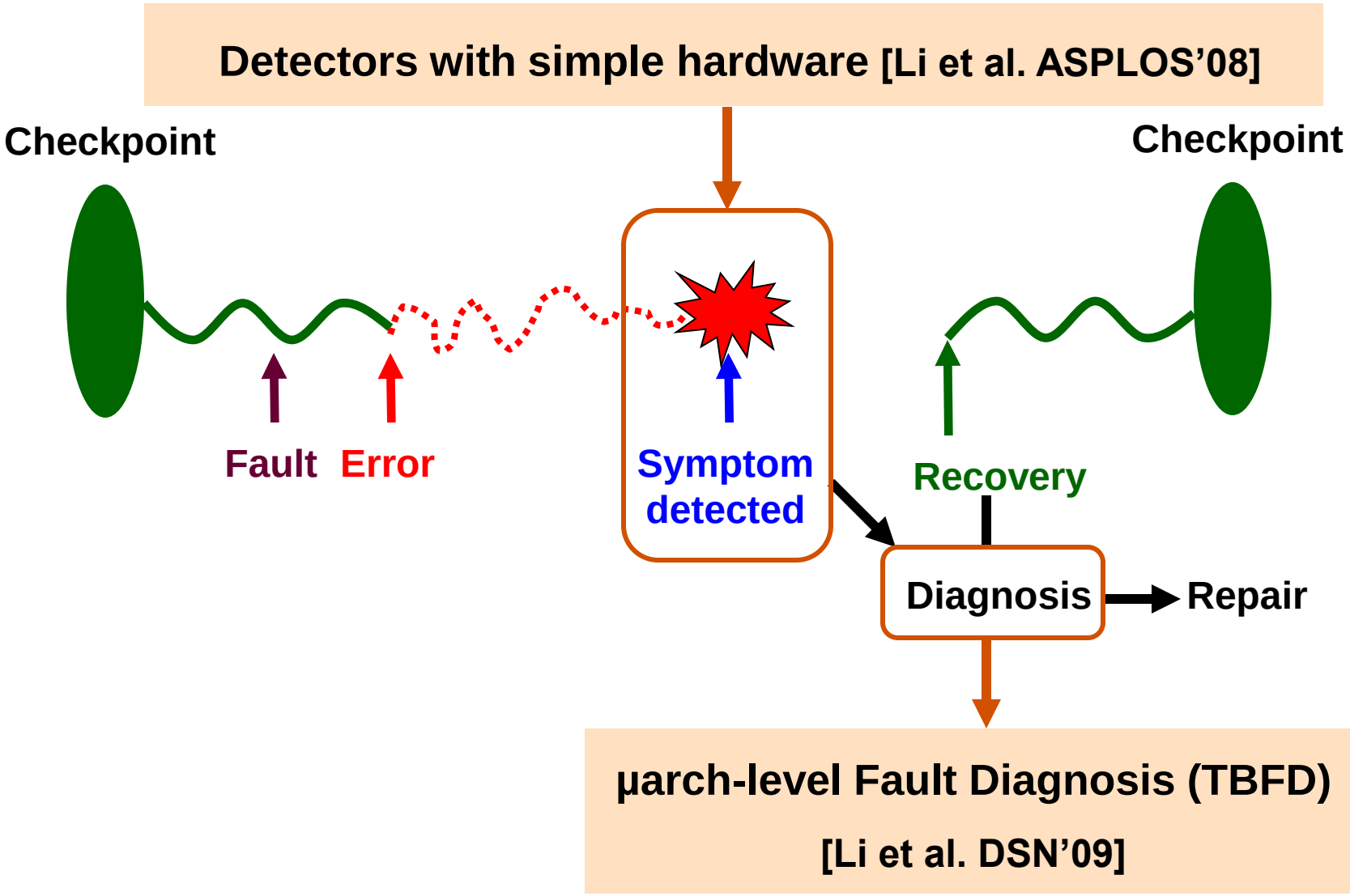
⇒ Watch for software anomalies (symptoms)

Zero to low overhead "always-on" monitors

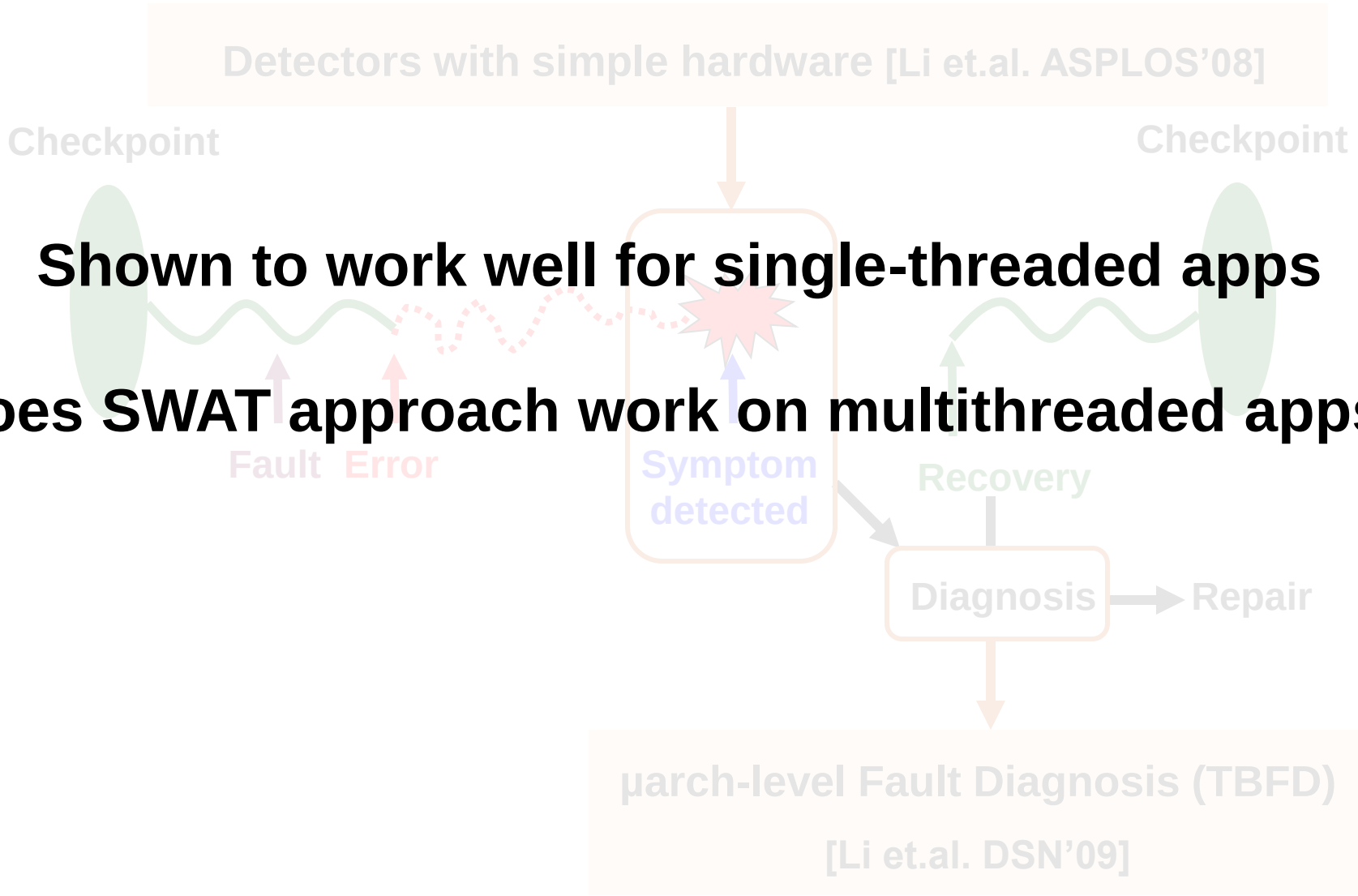
Diagnose cause after symptom detected

May incur high overhead, but rarely invoked

SWAT Framework Components



Challenge

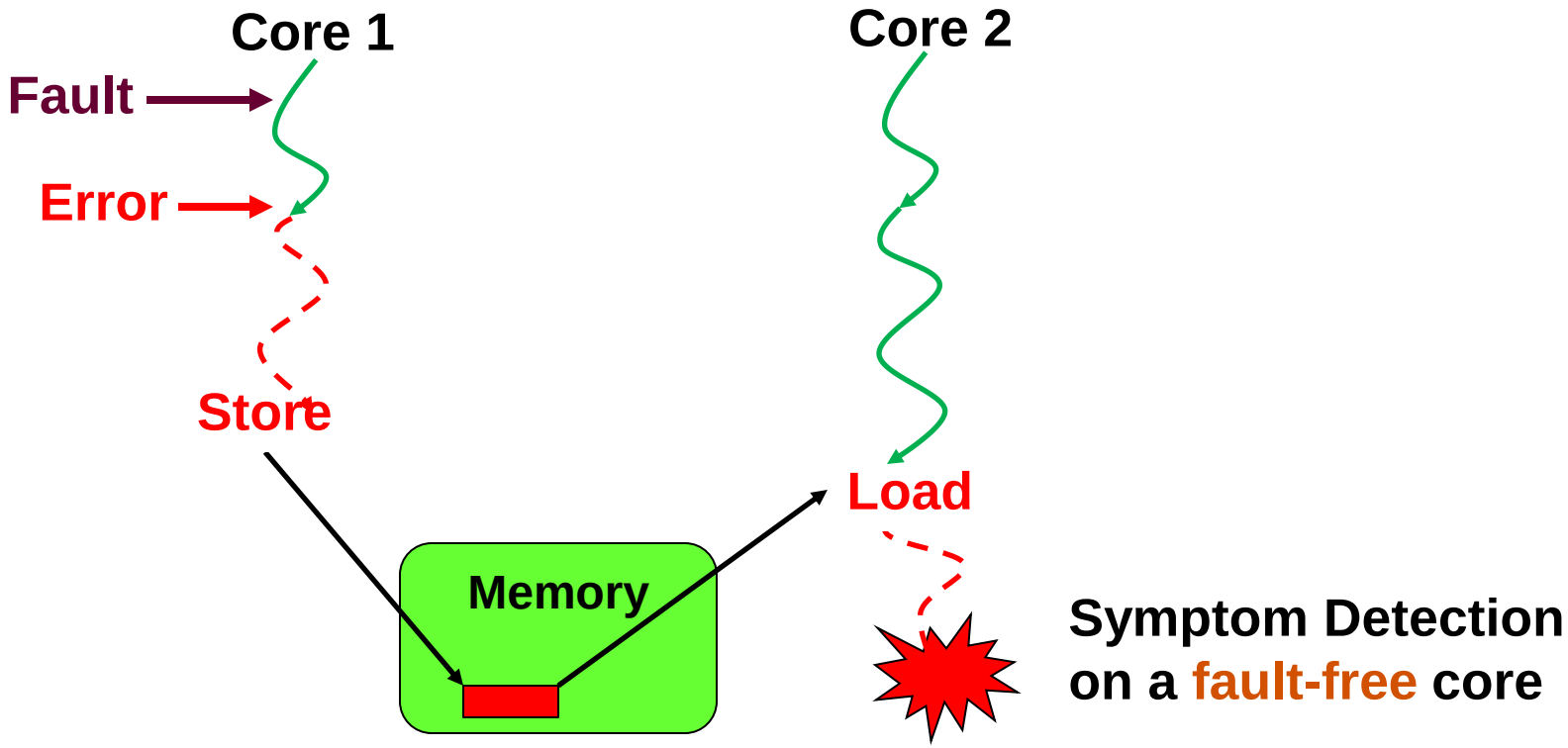


Shown to work well for single-threaded apps

Does SWAT approach work on multithreaded apps?

Challenge: Data sharing in multithreaded apps

- **Multithreaded apps share data among threads**



- **Does symptom detection work?**
- **Symptom causing core may not be faulty**
  - **How to diagnose faulty core?**

Contributions

- **Evaluate SWAT detectors on multithreaded apps**
  - **Low Silent Data Corruption rate for multithreaded apps**
  - **Observed symptom from fault-free cores**

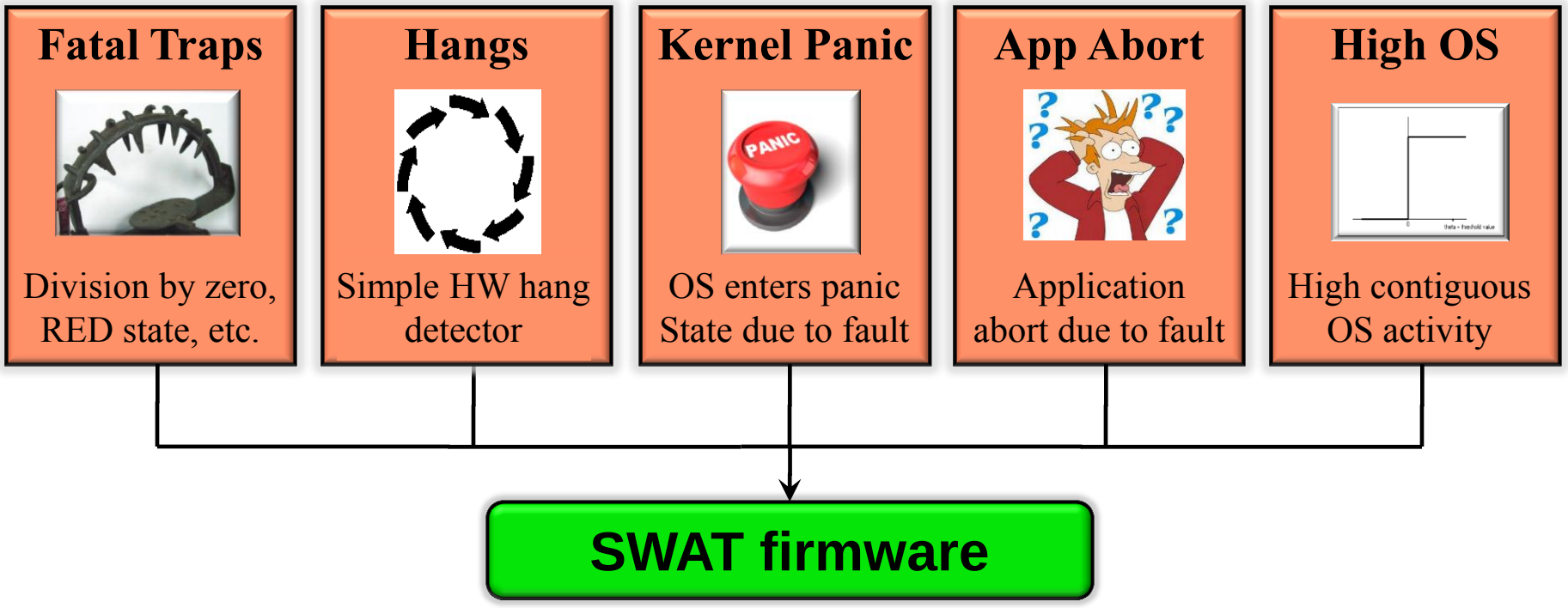
- **Novel fault diagnosis for multithreaded apps**
  - **Identifies the faulty core despite error propagation**
  - **Provides high diagnosability**

Outline

- Motivation
- **mSWAT Detection**
- **mSWAT Diagnosis**
- **Results**
- **Summary and Future Work**

mSWAT Fault Detection

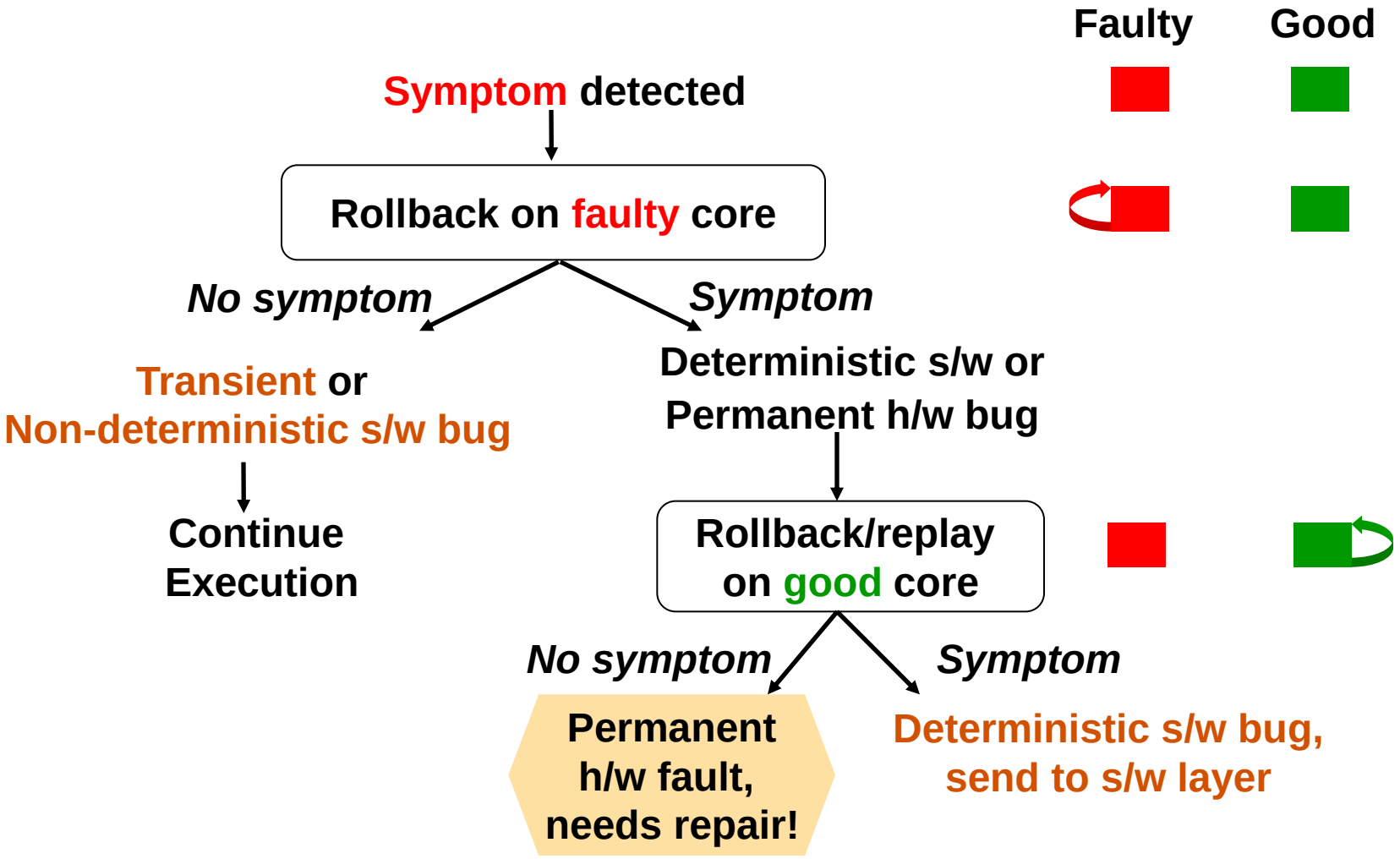
- **SWAT Detectors:**
  - **Low-cost monitors to detect anomalous sw behavior**
  - **Incur near-zero perf overhead in fault-free operation**



- **Symptom detectors provide low Silent Data Corruption rate**

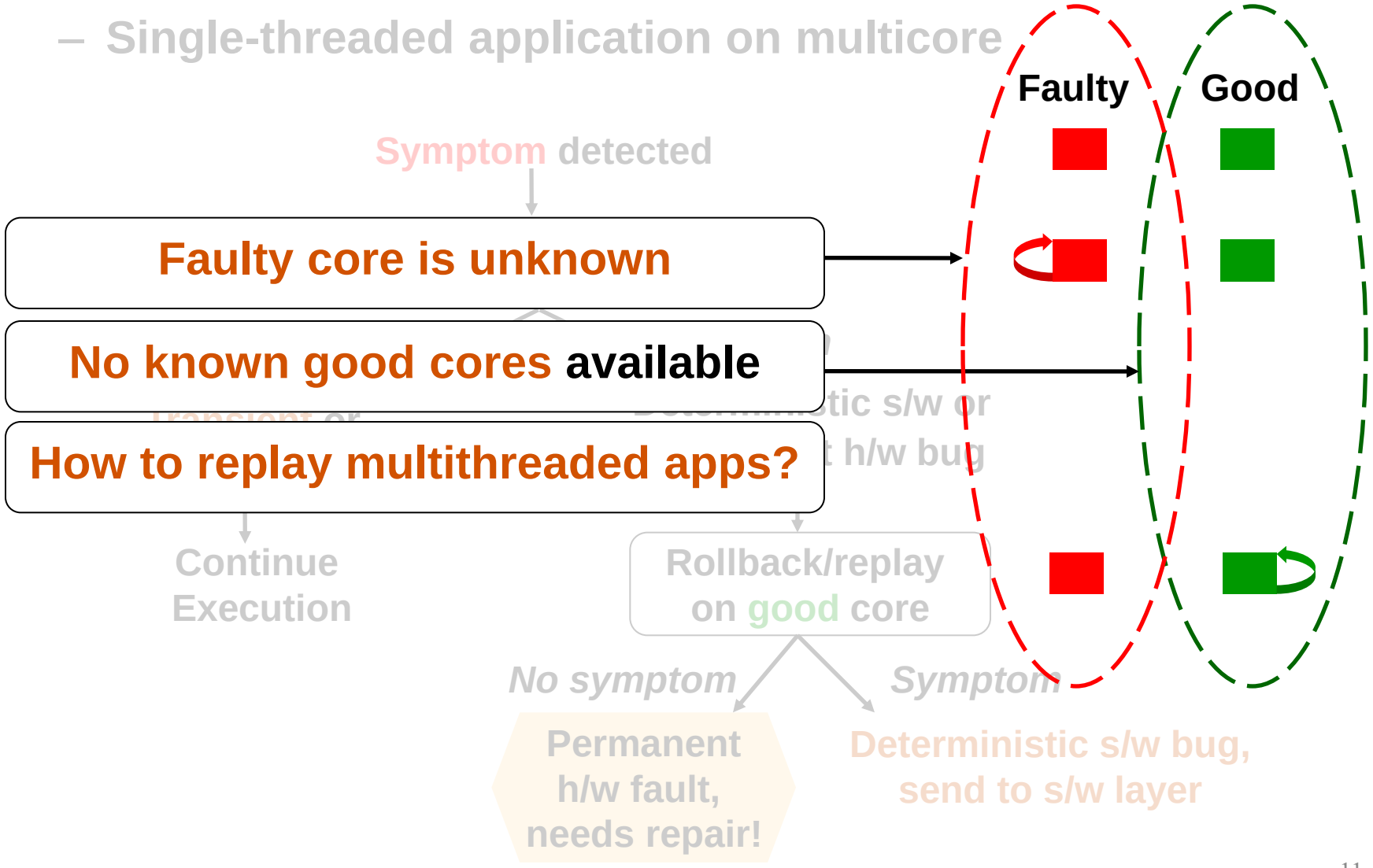
SWAT Fault Diagnosis

- **Rollback/replay on same/different core**
  - **Single-threaded application on multicore**



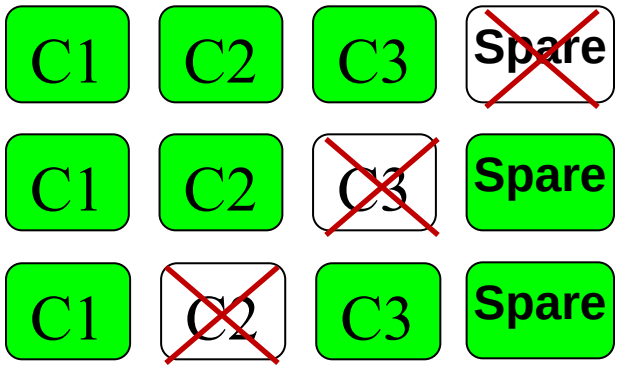
Challenges

- Rollback/replay on same/different core
  - Single-threaded application on multicore



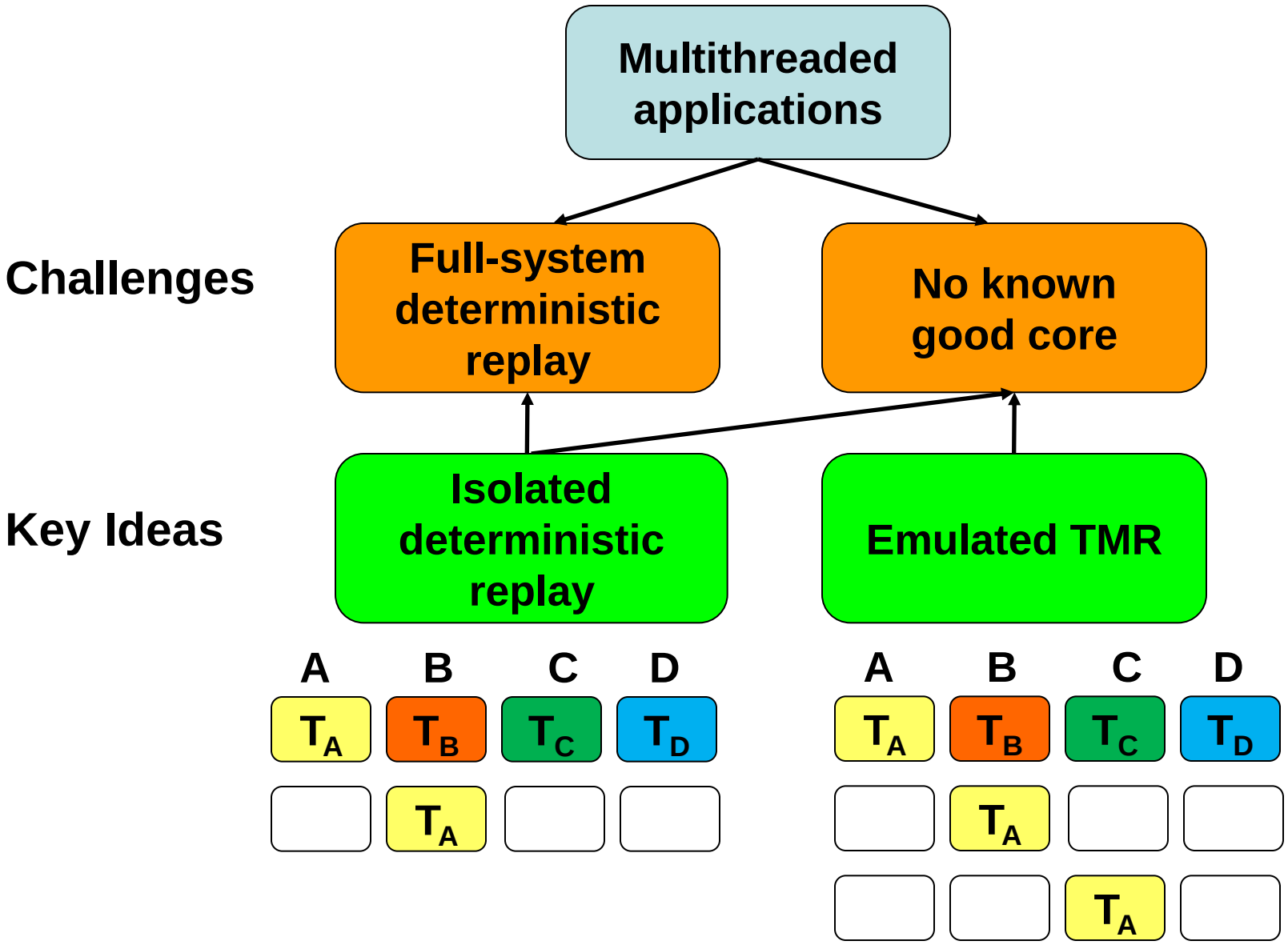
Extending SWAT Diagnosis to Multithreaded Apps

- **Assumptions: In-core faults, single core fault model**
- **Naïve extension – N known good cores to replay the trace**
  - **Too expensive – area**
  - **Requires full-system deterministic replay**
- **Simple optimization – One spare core**

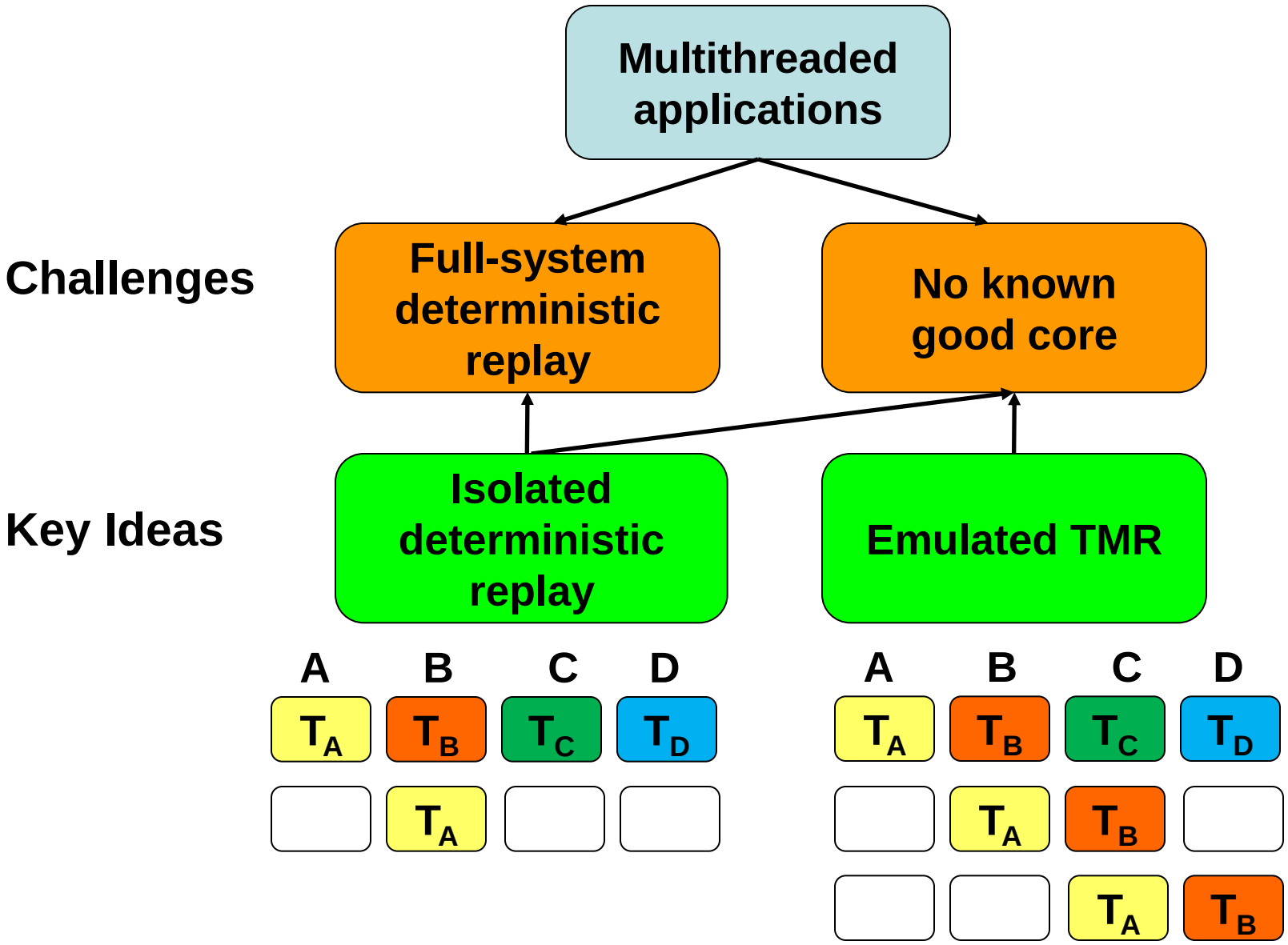


  - **Not scalable, requires N full-system deterministic replays**
  - **High hardware overhead – requires a spare core**
  - **Single point of failure – spare core**

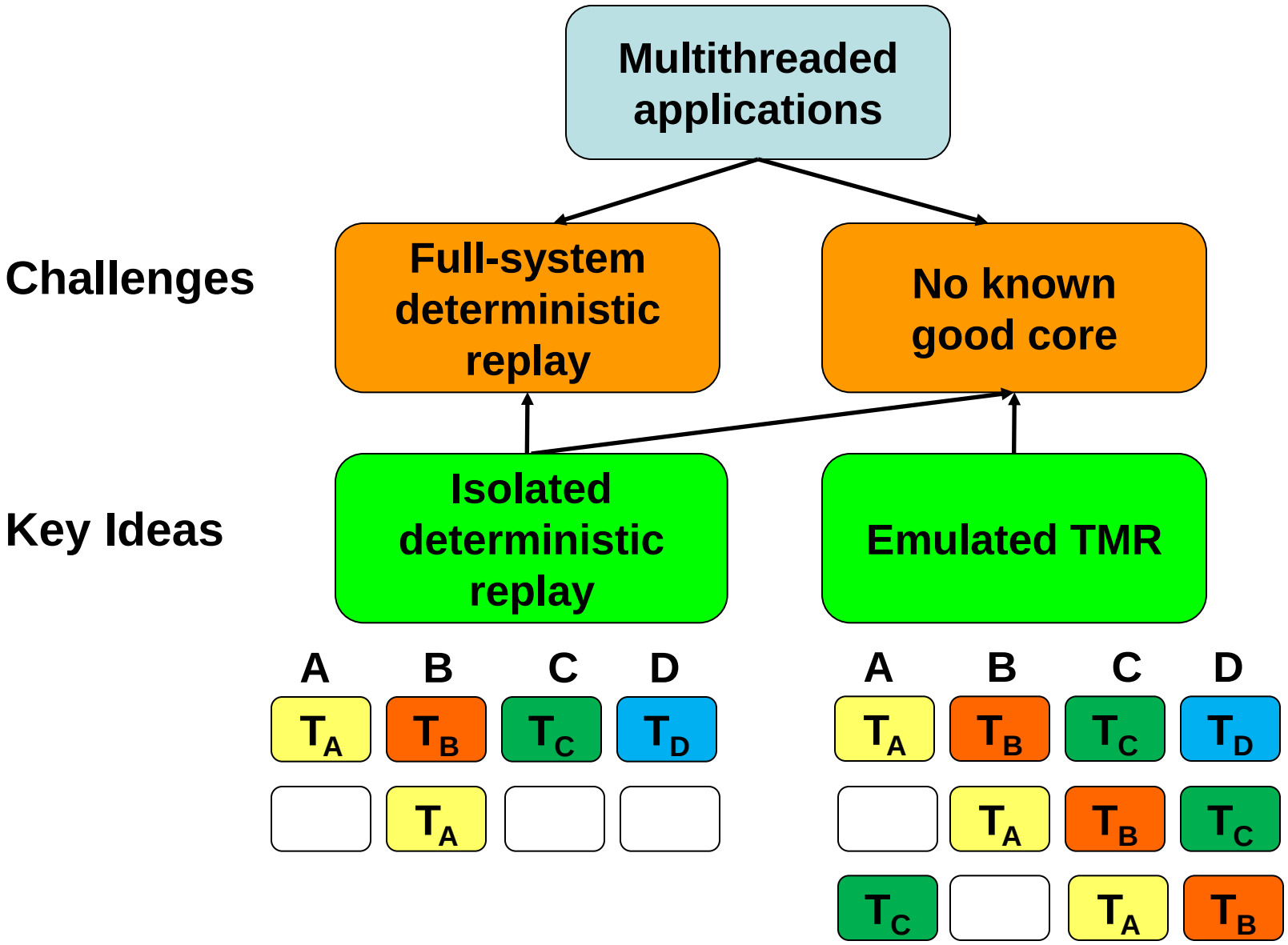
mSWAT Diagnosis - Key Ideas



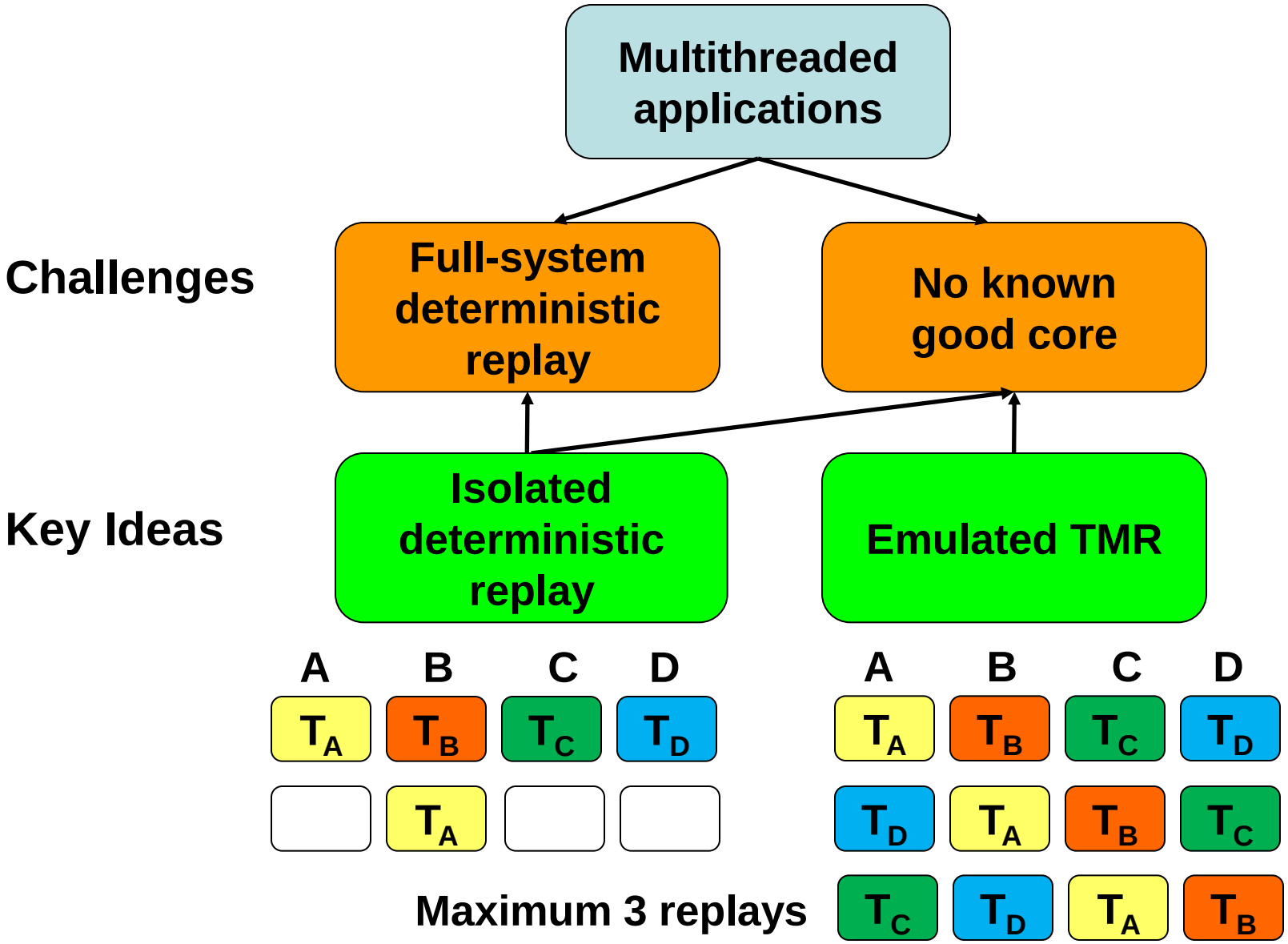
mSWAT Diagnosis - Key Ideas



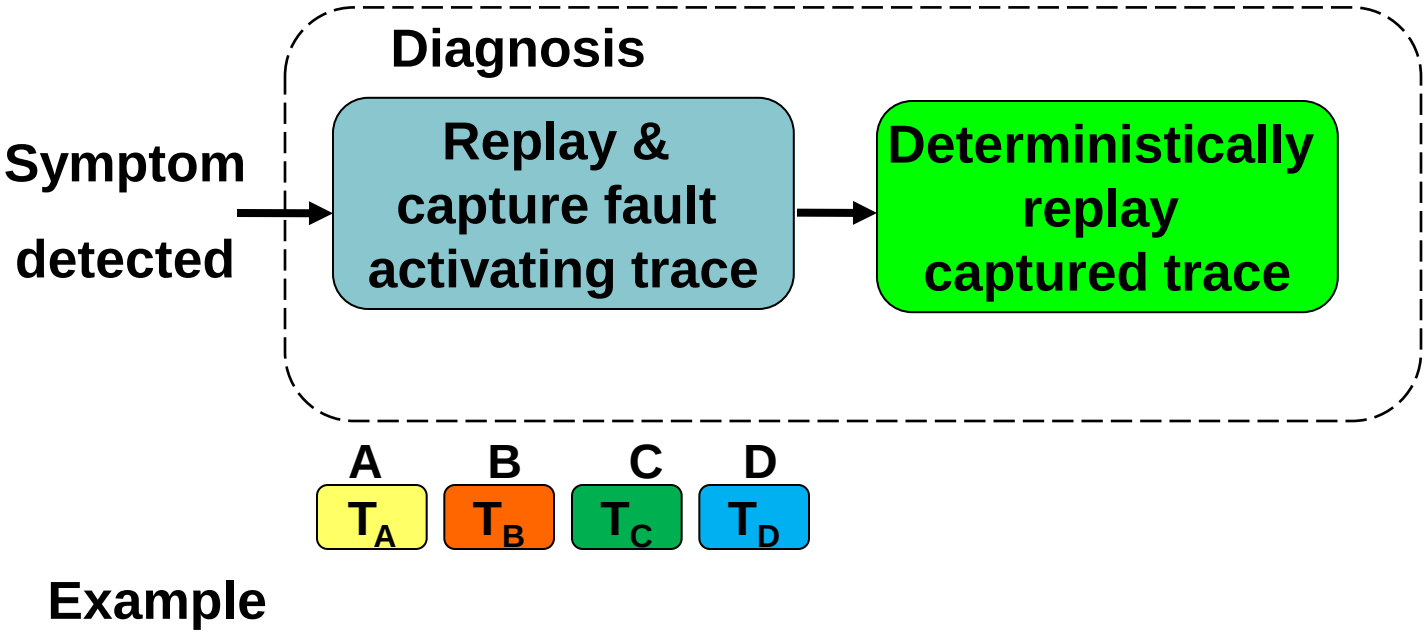
mSWAT Diagnosis - Key Ideas



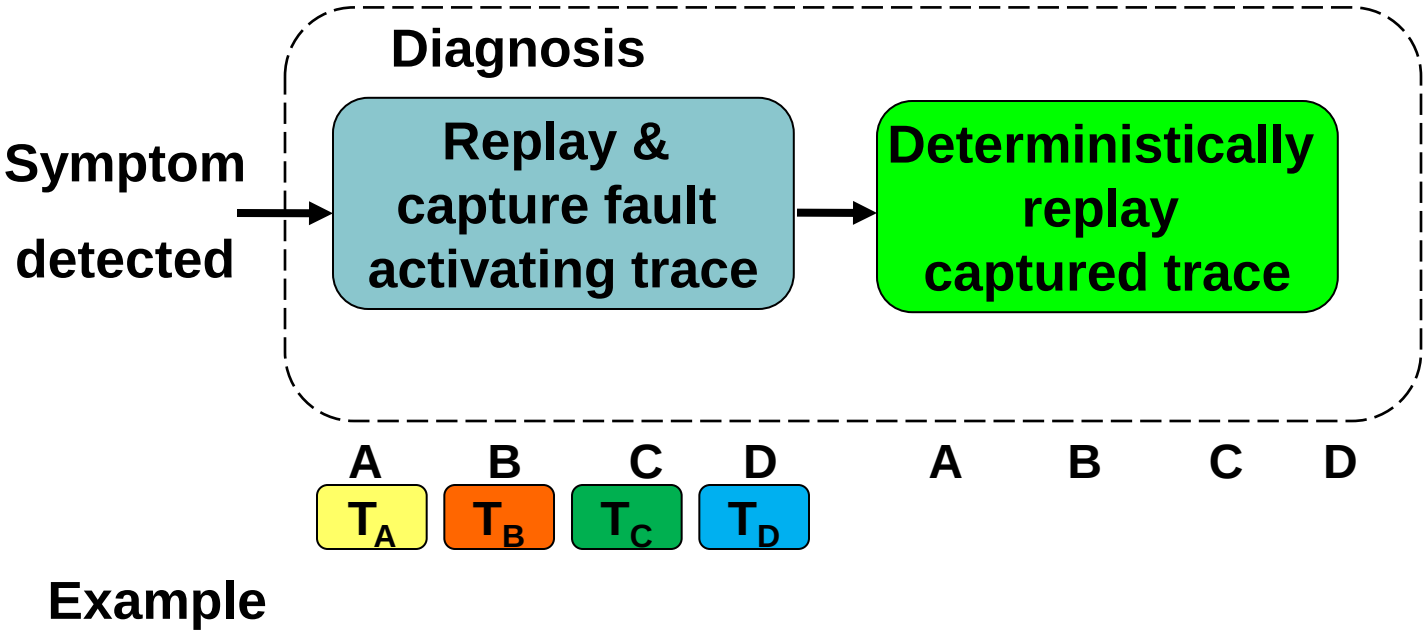
mSWAT Diagnosis - Key Ideas



Multicore Fault Diagnosis Algorithm Overview

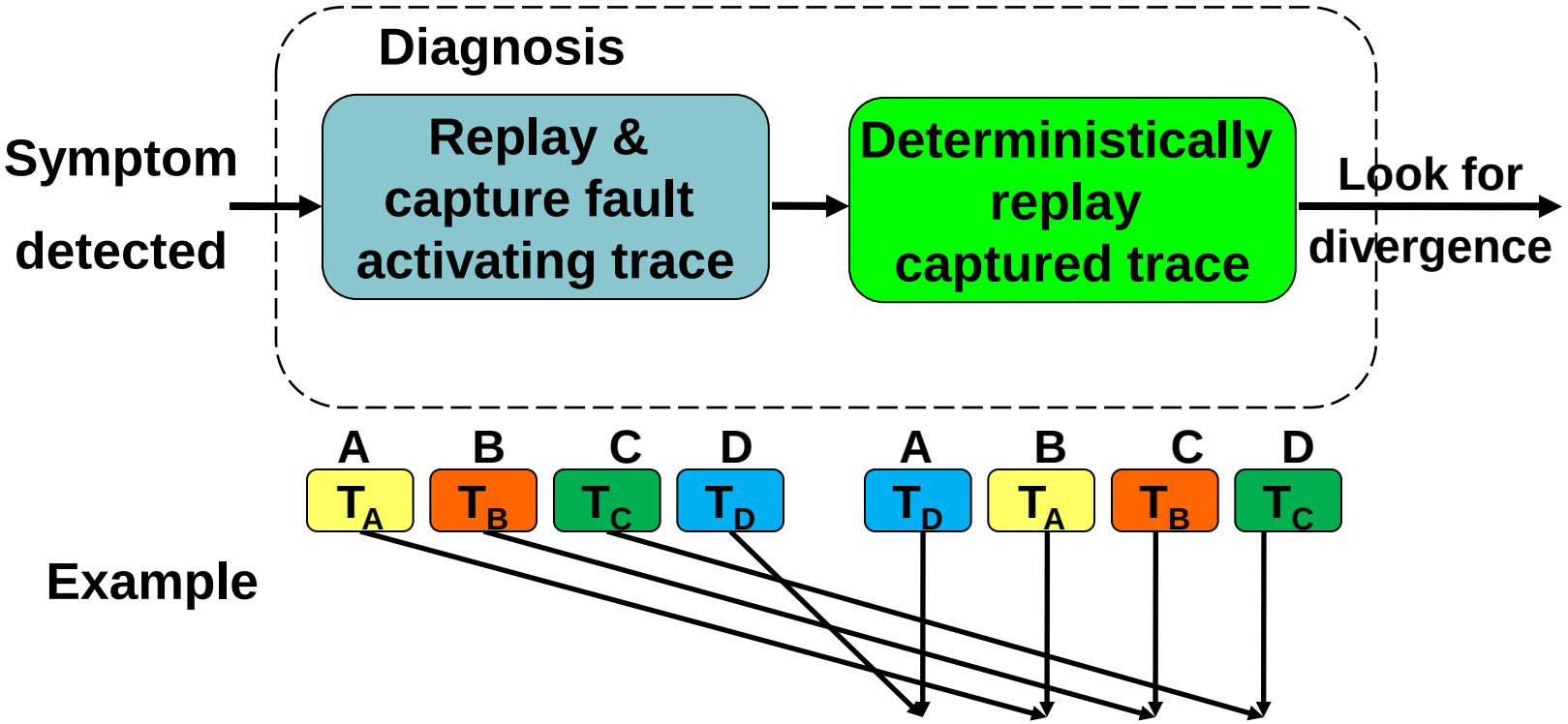


Multicore Fault Diagnosis Algorithm Overview

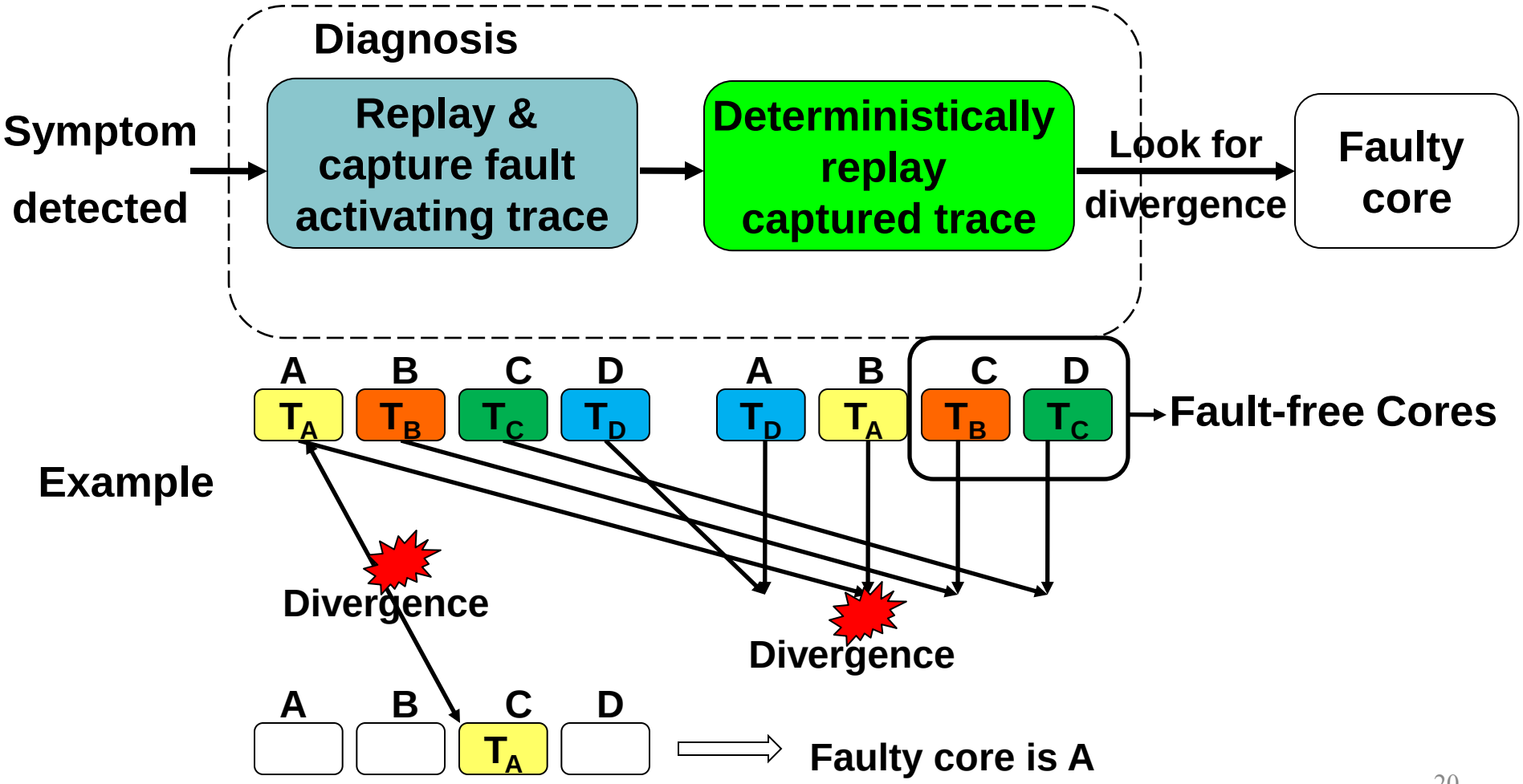


Example

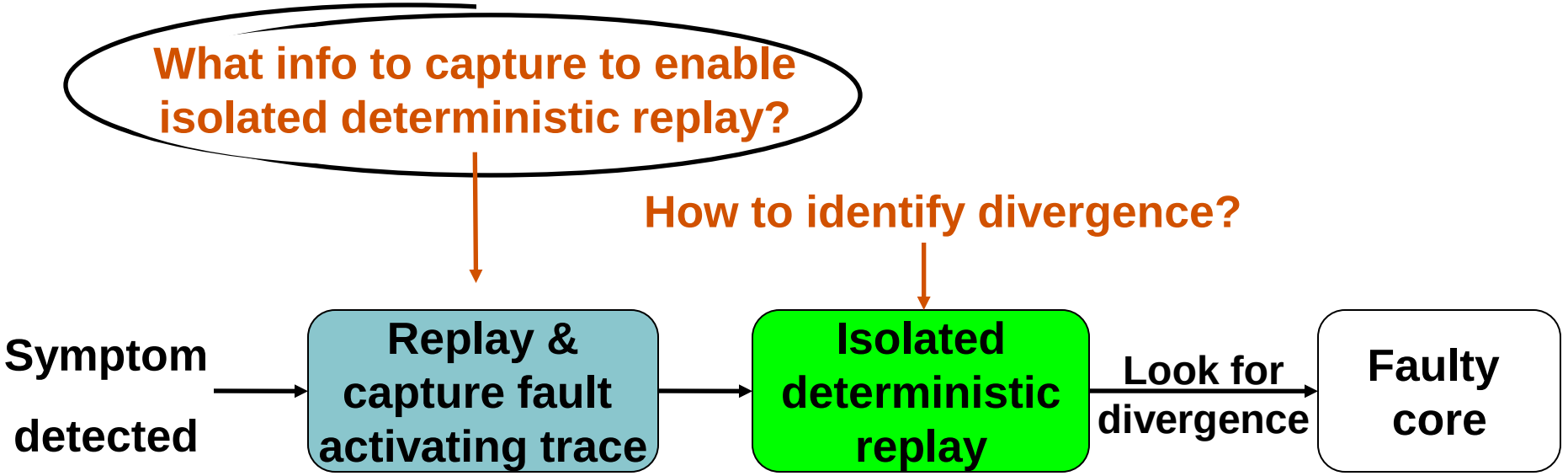
Multicore Fault Diagnosis Algorithm Overview



Multicore Fault Diagnosis Algorithm Overview

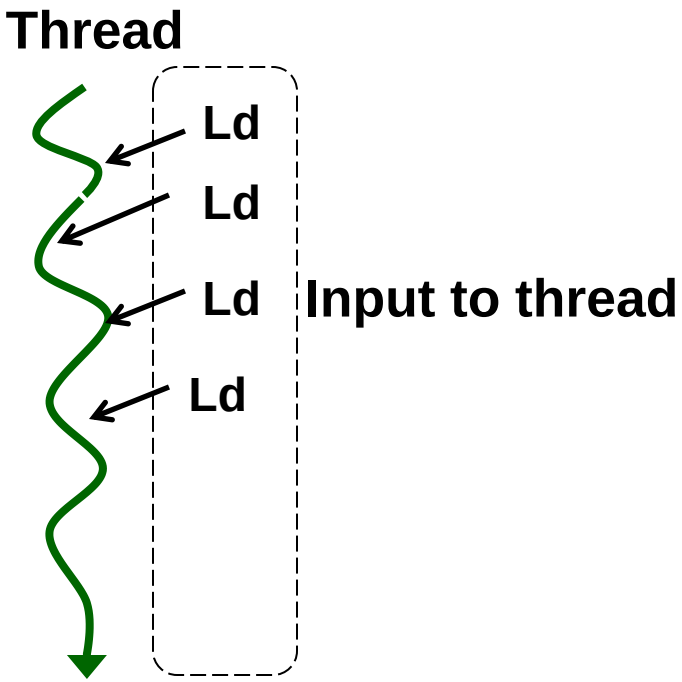


Digging Deeper



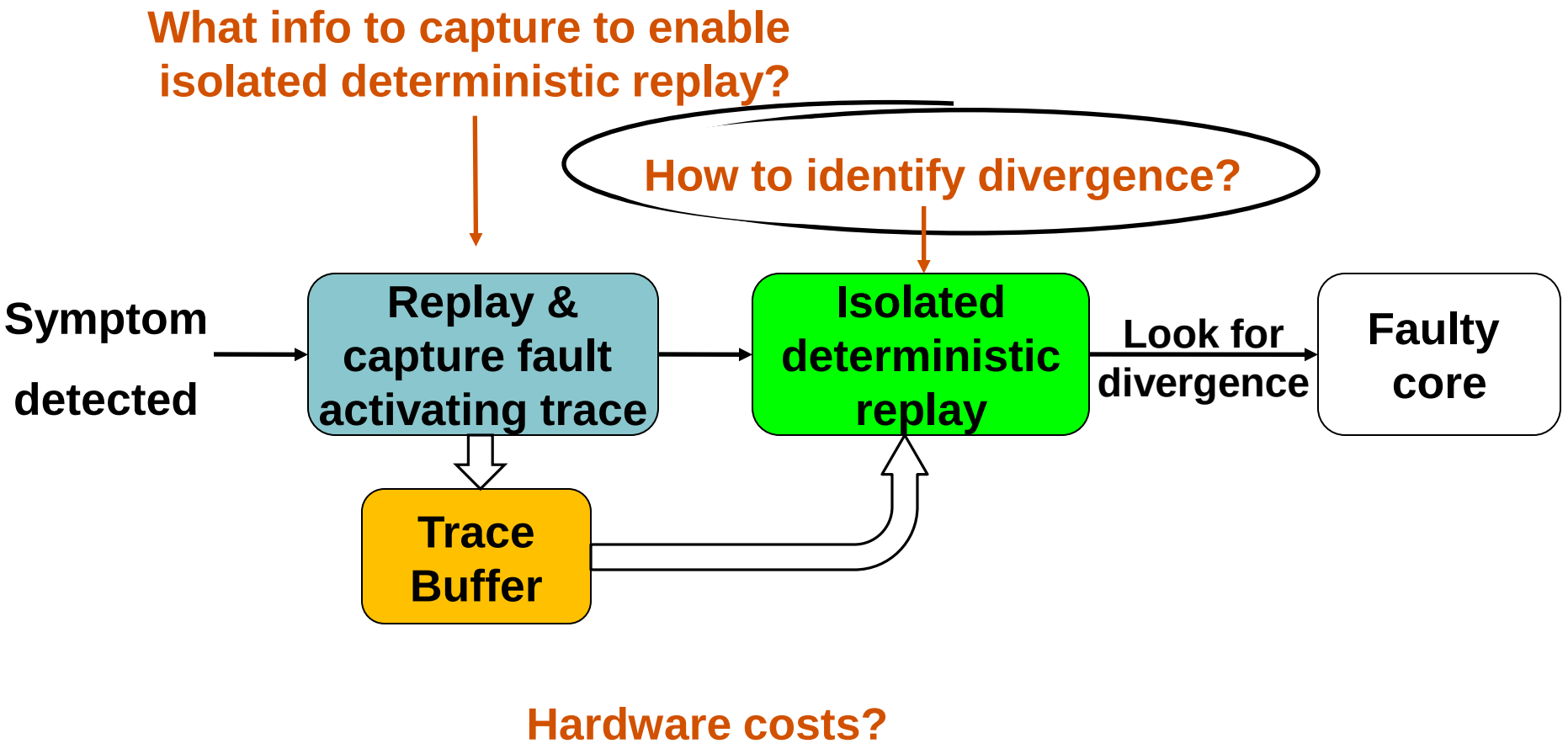
Hardware costs?

Enabling Isolated Deterministic Replay



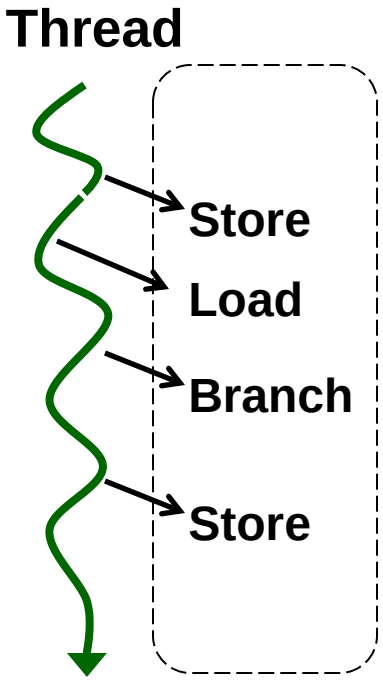
- **Recording thread inputs sufficient – similar to BugNet**
  - **Record all retiring loads values**

Digging Deeper (Contd.)

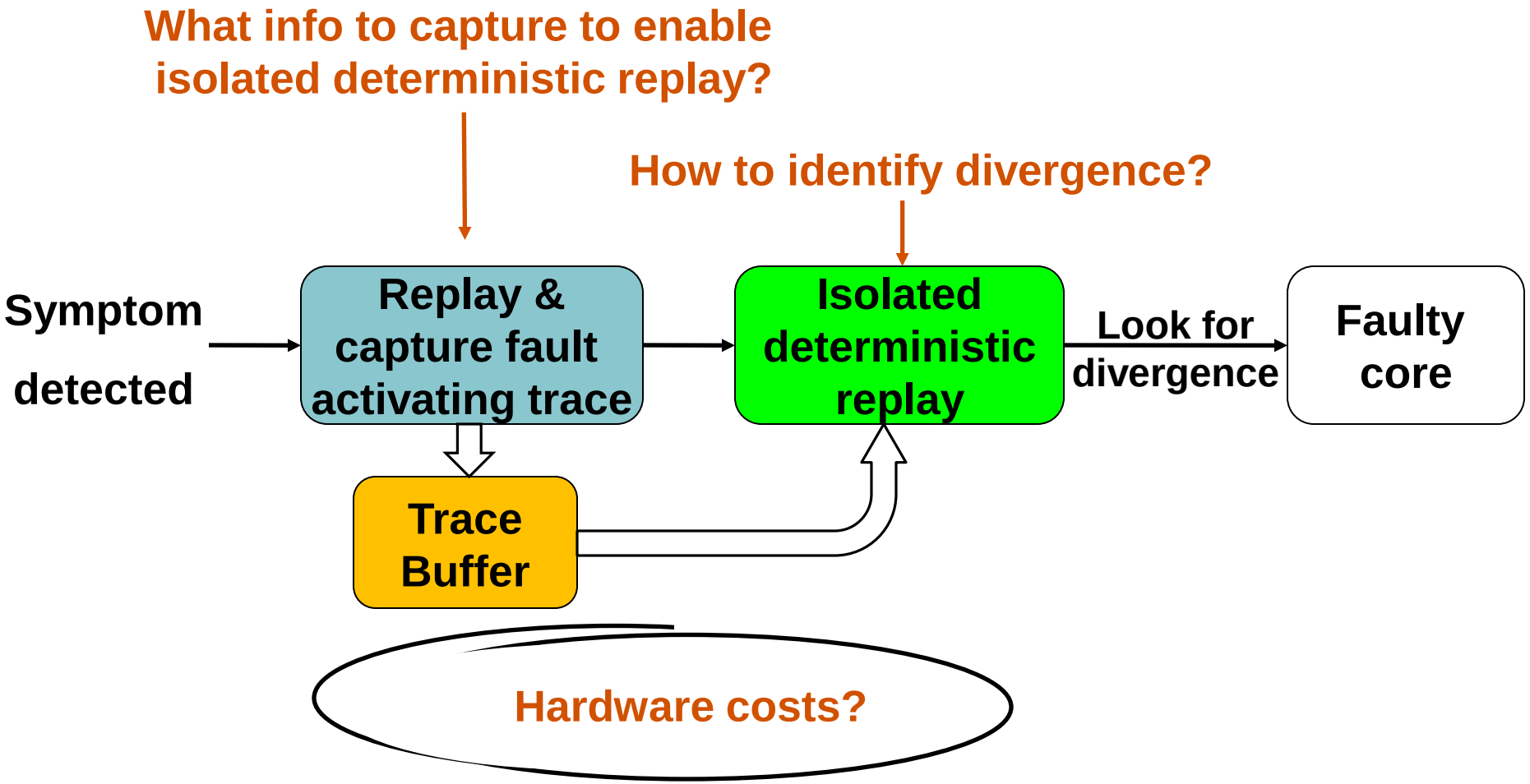


Identifying Divergence

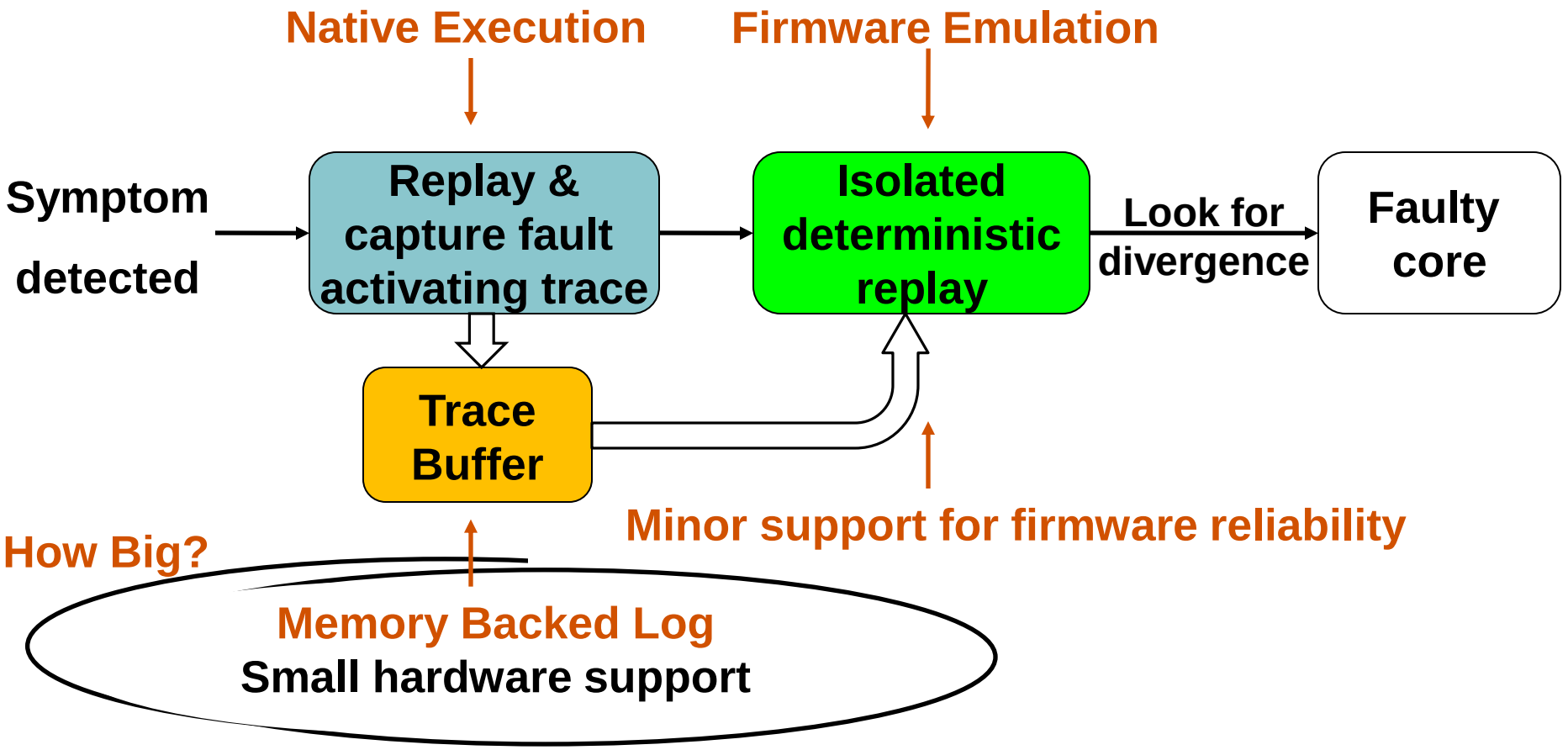
- Comparing all instructions ⇒ Large buffer requirement
- Faults corrupt software through memory and control instrns
  - Capture memory and control instructions



Digging Deeper (Contd.)



Hardware Costs



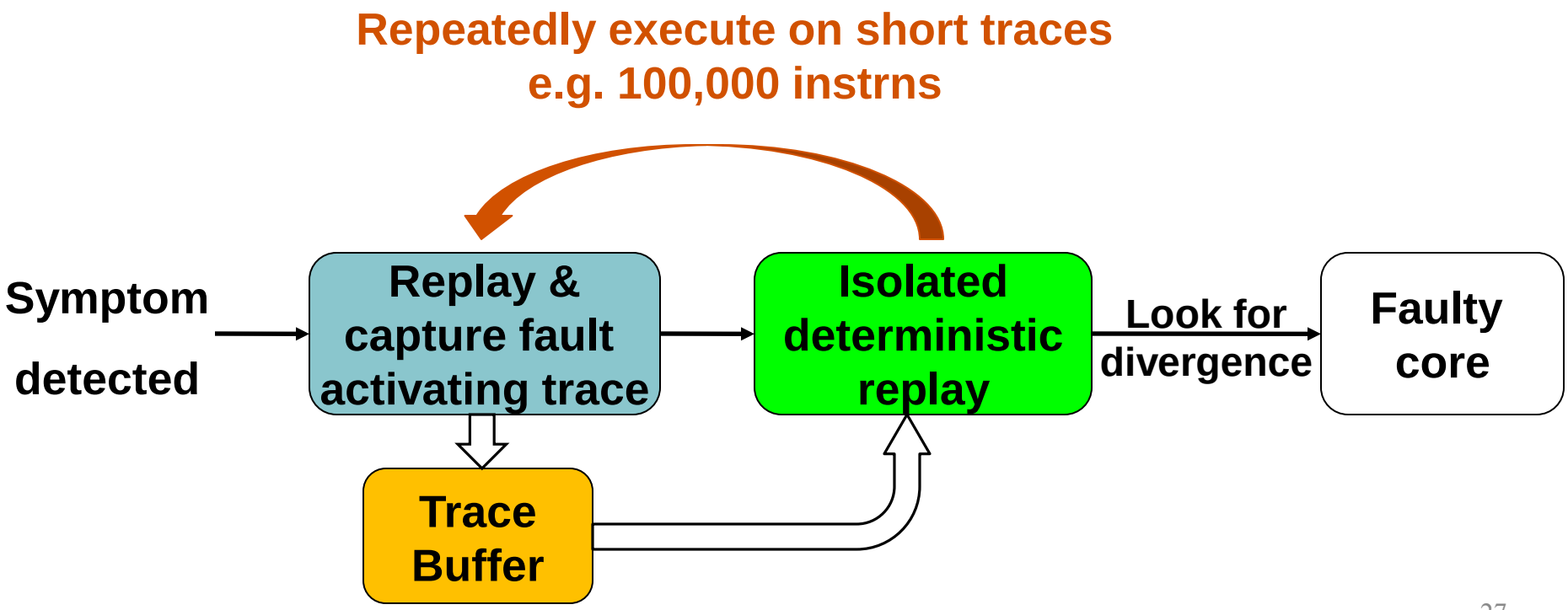
What if the faulty core subverts the process?

Key Idea: On a divergence two cores take over

Trace Buffer Size

- **Long detection latency ⇒ large trace buffers (8MB/core)**
  - **Need to reduce the size requirement**

  ⇒ Iterative Diagnosis Algorithm



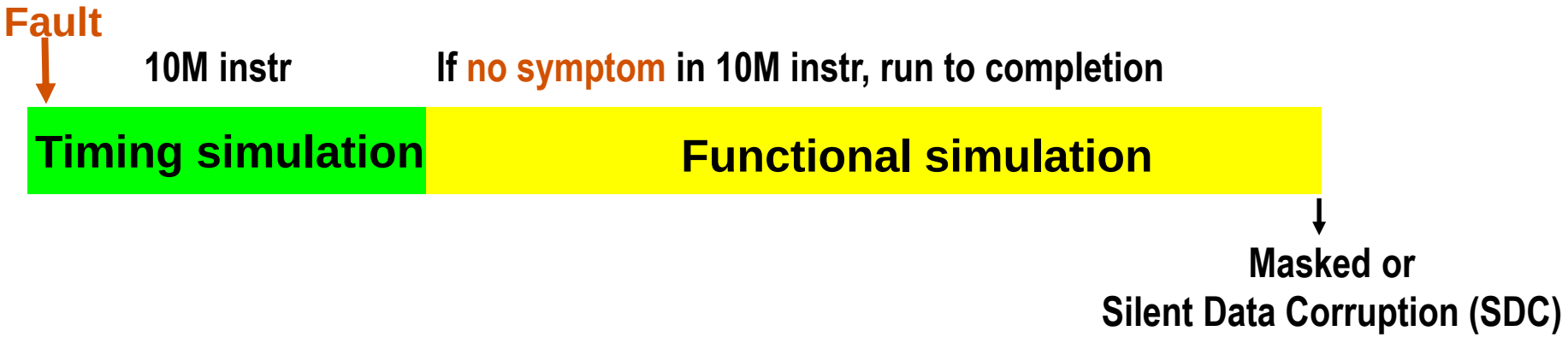
Experimental Methodology

- **Microarchitecture-level fault injection**
  - **GEMS timing models + Simics full-system simulation**
  - **Six multithreaded applications on OpenSolaris**
    - **4 Multimedia apps and 1 each from SPLASH and PARSEC**
  - **4 core system running 4-threades apps**

- **Faults in latches of 7 μarch units**
  - **Permanent (stuck-at) and transients faults**

Experimental Methodology

Detection:

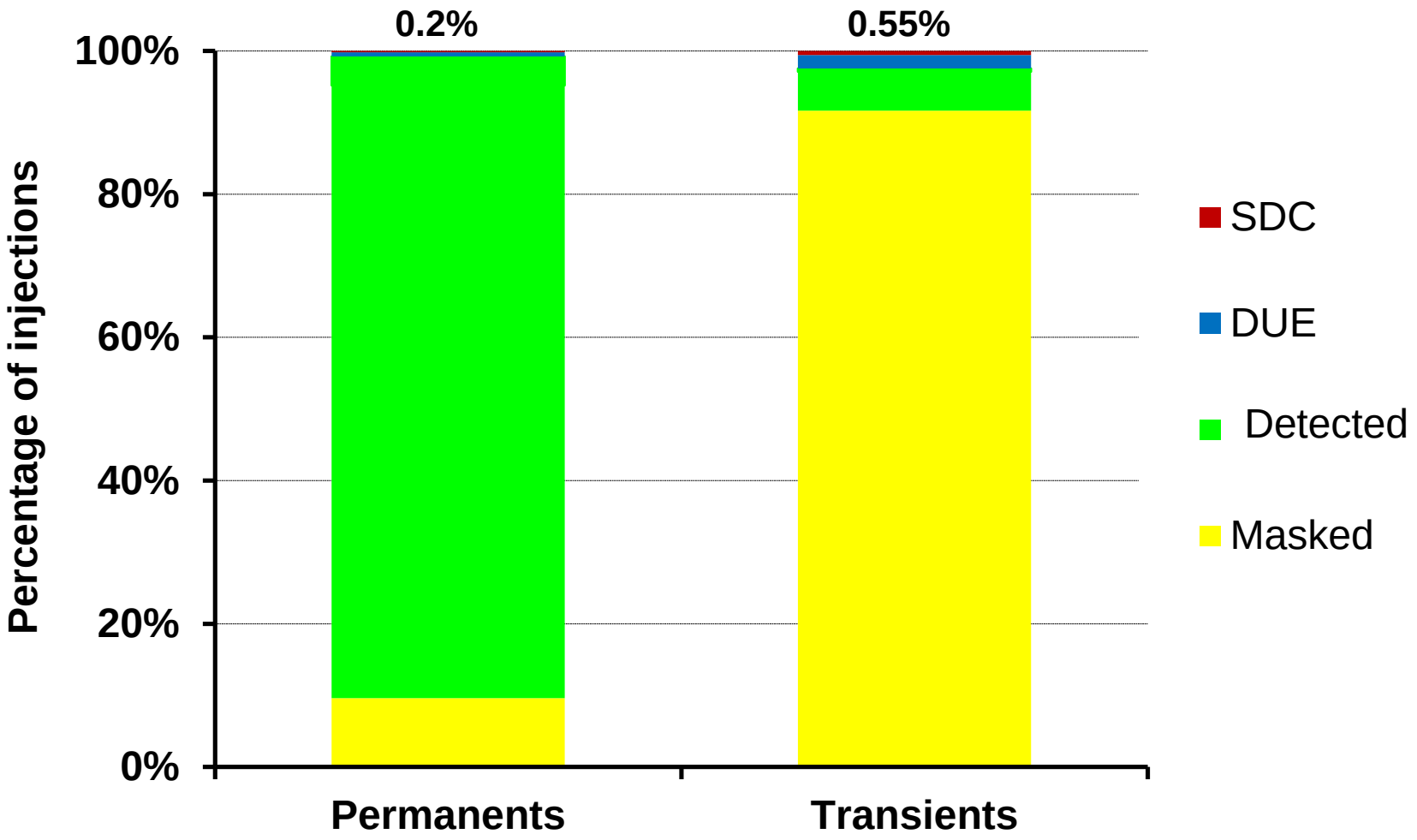


- **Metrics: SDC Rate, detection latency**

Diagnosis:

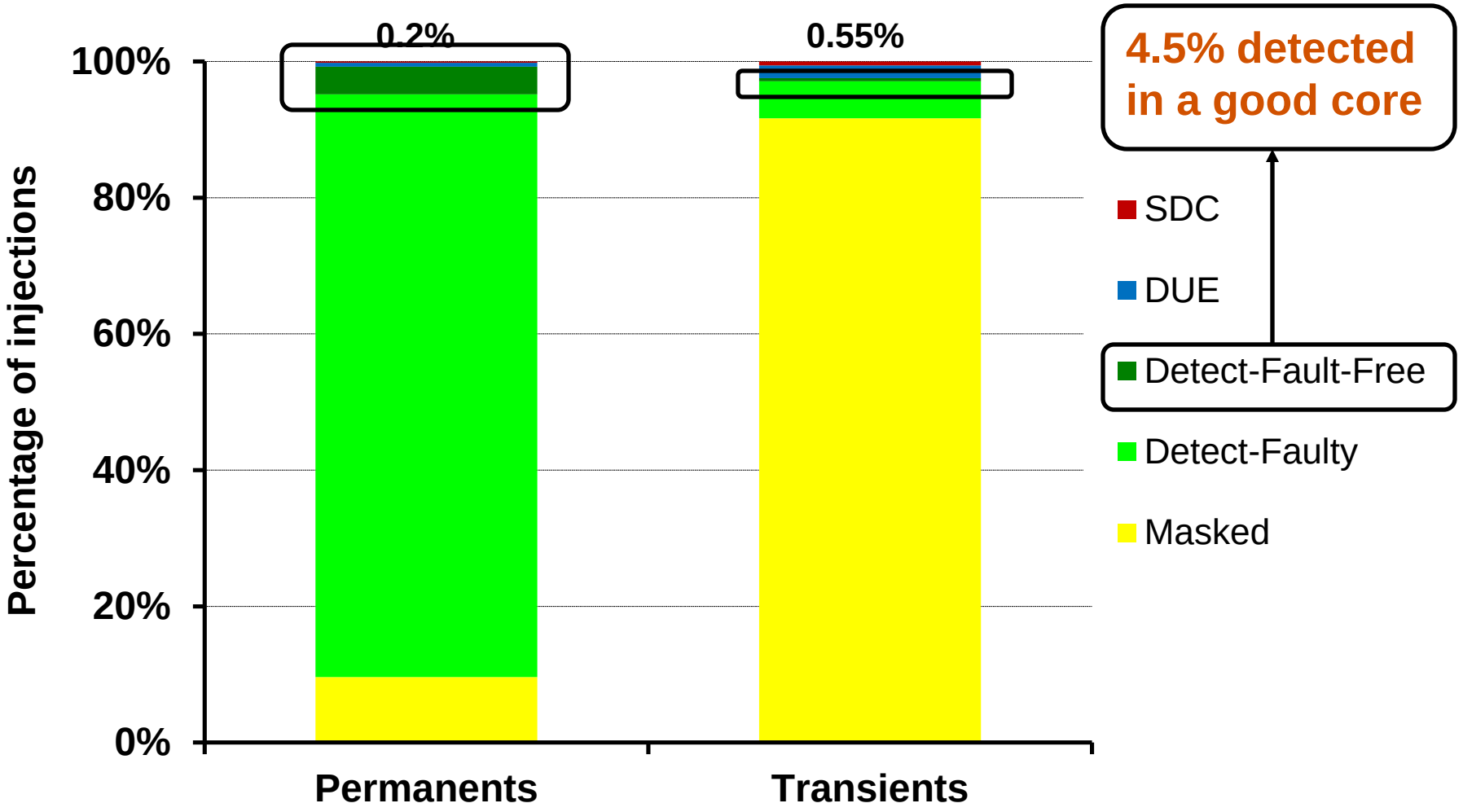
- **Iterative algorithm with 100,000 instrns in each iteration**
  - **Until divergence or 20M instrns**
- **Deterministic replay is native execution**
  - **Not firmware emulated**
- **Metrics: Diagnosability, overheads**

Results: mSWAT Detection Summary



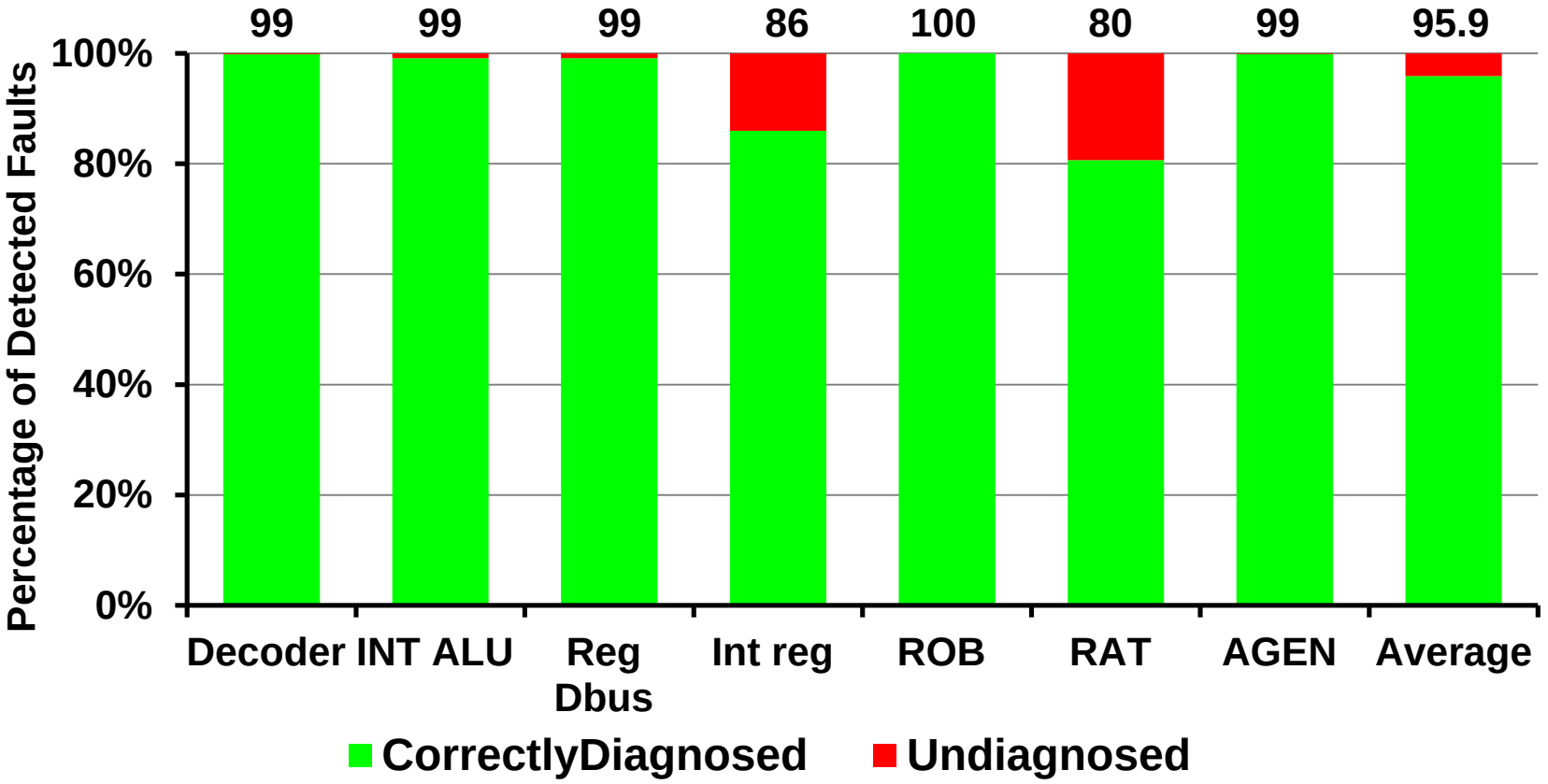
- **SDC Rate: Only 0.2% for permanents & 0.55% for transients**
- **Detection Latency: Over 99% detected within 10M instrns**

Results: mSWAT Detection Summary



- **SDC Rate: Only 0.2% for permanents & 0.55% for transients**
- **Detection Latency: Over 99% detected within 10M instrns**

Results: mSWAT Diagnosability



- **Over 95% of detected faults are successfully diagnosed**
- **All faults detected in fault-free core are diagnosed**
- **Undiagnosed faults: 88% did not activate faults**

Results: mSWAT Diagnosis Overheads

- **Diagnosis Latency**
  - **98% diagnosed <10 million cycles (10ms in 1GHz system)**
  - **93% were diagnosed in 1 iteration**
    - **Iterative approach is effective**

- **Trace Buffer size**
  - **96% require <400KB/core**
    - **Trace buffer can easily fit in L2 or L3 cache**

mSWAT Summary

- **Detection: Low SDC rate, detection latency**
- **Diagnosis – identifying the faulty core**
  - **Challenges: no known good core, deterministic replay**
  - **High diagnosability with low diagnosis latency**
  - **Low Hardware overhead - Firmware based implementation**
  - **Scalable – maximum 3 replays for any system**
- **Future Work:**
  - **Reducing SDCs, detection latency, recovery overheads**
  - **Extending to server apps; off-core faults**
  - **Validation on FPGAs (w/ Michigan)**