ParaVerser: Harnessing Heterogeneous Parallelism for Affordable Fault Detection in Data Centers

Minli Julie Liao[*], *Sam Ainsworth*[†], *Lev Mukhanov*[‡], *Adrian Barredo*[#], *Markos Kynigos*[§], *Timothy Jones**[*]

[*]University of Cambridge
[#]Barcelona Supercomputing Center
[†]University of Edinburgh
[§]The University of Manchester
[‡]Queen Mary University London

Errors at Server-Scale

Frequent, undetected hard faults causing silent data corruption (SDC)

- **Insufficient** existing software scanners:
  - Infrequent out-of-production tests
    → SDC goes on undetected for **months**
  - In-production light-weight tests
    → **Low** error detection coverage
- **Unaffordable** dual/triple-core lockstep:
  - Guarantees full-coverage error detection
  - Real-time with low performance overhead
  - **Double/triple** energy + hardware area

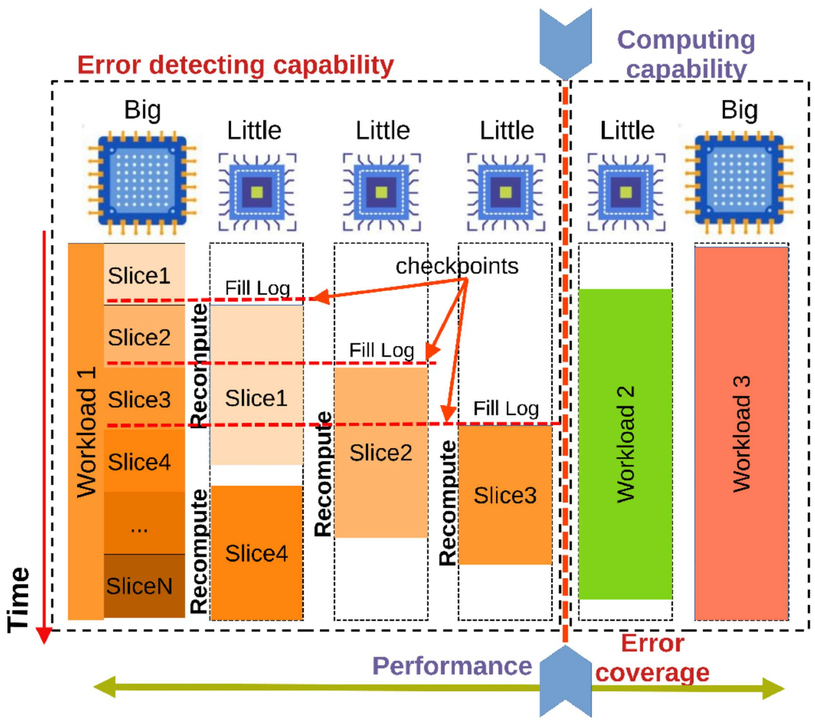
Heterogeneous Parallel Error Detection

Full-coverage error detection through redundancy with low energy overhead

- **Extra parallelism** in redundant execution enabled through checkpointing + load-store logging
- Multiple energy **efficient** checker cores **in parallel** for redundant execution:
  - Keep up with high performance main core
  - At much lower energy cost

ParaVerser

Affordable, adjustable error detection with heterogeneous cores in data centers



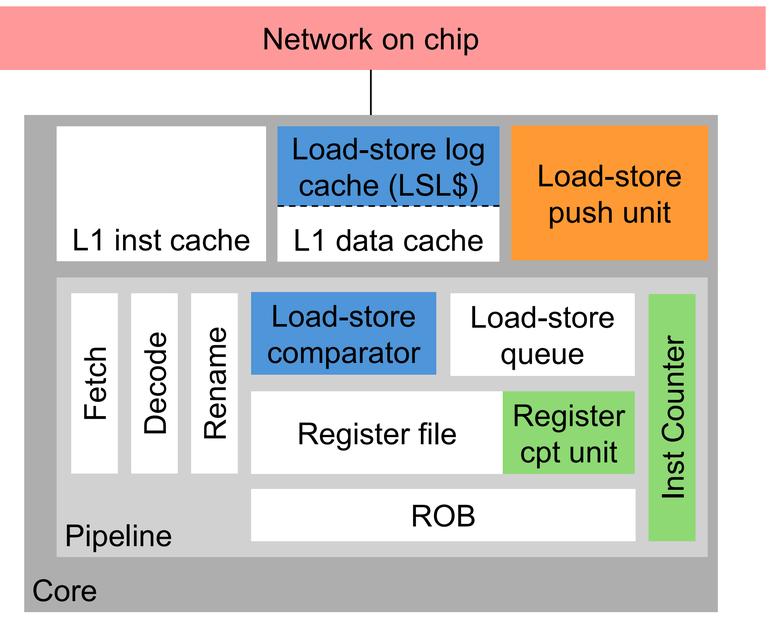
Adjustable performance and error coverage

- **Full-coverage**: error detection first
  - Guarantees **full** error detection **coverage**
  - Insufficient resource in error detection
    → original execution **stalls**
- **Opportunistic**: performance first
  - Guarantees **minimal** performance **overhead**
  - Insufficient resource in error detection
    → original execution **continues** with segments left **unchecked**

Affordable hardware overhead

Minor alterations to **existing** cores in the system

- Repurpose existing L1 data cache to store load-store log (LSL) when used as checker
  - 1 extra bit per cache line
- **1064B per-core** overhead
  - Mainly for register checkpoint (cpt)
  - Any core can be main or checker, same overhead for all cores



Basic Operations

Main core

1. Take register cpt + start inst counter → cpt
2. Run: log and push loads/stores to checker's LSL$ → LSL
3. Take cpt + stop counter → cpt + count

Checker core

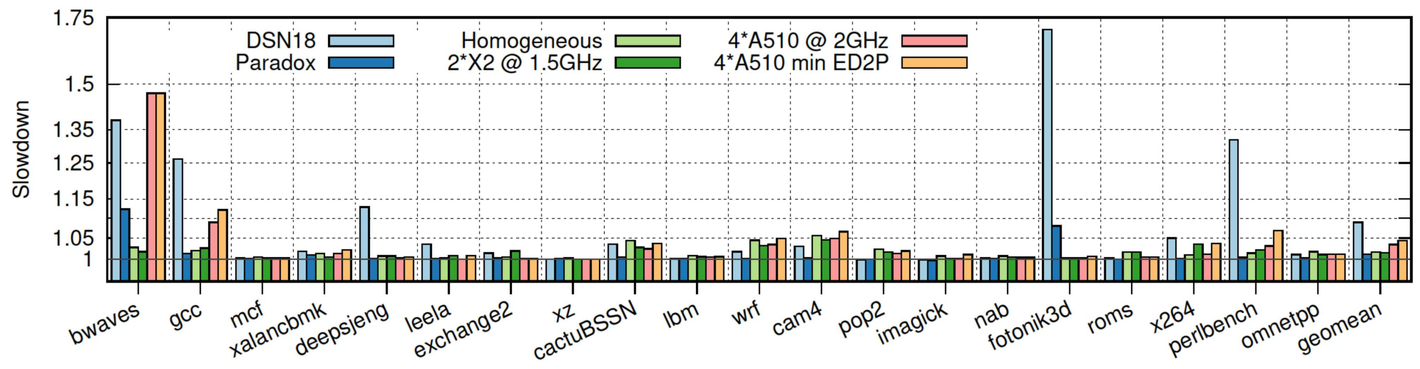
1. Set register from cpt + start inst counter
2. Run: get loads/stores from LSL$ + compare (e.g. address, store value)
3. Stop at count + take cpt + compare

Evaluation

High performance X2 + energy efficient A510 system
Full-speed X2 main core + Various checker core type/count at various DVFS points

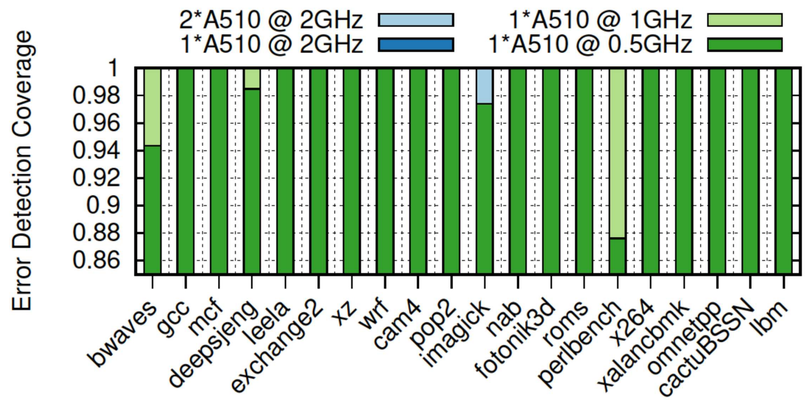
Full-coverage error detection

- DSN18 & Paradox: prior works, **25% area** overhead **dedicated** checker cores
- Homogeneous: similar to dual-core **lockstep**, **95%** energy overhead
- **ParaVerser** with 4*A510 min ED2P (energy * delay^2) DVFS config checkers:
  - **4.3%** performance **degradation** (vs Baseline without error detection)
  - **70% reduction** in energy overhead (vs Homogeneous)



Opportunistic error detection

- **<1%** performance overhead
- High error detection coverage with very little resource
- Potential for sampling



NoC overhead and hash-mode

- LSL traffic cause heavy NoC contention with slower NoC
- Hash-mode hashes all LSL traffic except for load value
  - Greatly reduce NoC contention
  - Similar slowdown to fast NoC (2x width, +1/3 clock rate)