FRACTAL: Fast Reverse Address translation over Coherence for Tracing And Logging

Francesco Ciraolo[1], Patrick Carpanedo[1], Daniele Ottaviano[2], Matias Ou[1], Marco Caccamo[2], Renato Mancuso[1]

[1]Boston University, [2]Technische Universität München

RAGE 2026
Saint Malo, May 11th, 2026

Alexander von
HUMBOLDT
STIFTUNG

BOSTON
UNIVERSITY

Outline

BOSTON UNIVERSITY

Motivations

- Observability in computing systems is complex yet crucial
- Coherent managers are granted with complete view of LLC misses for free
  - No HW/SW manipulation required
  - Low overhead, as the snoop is already issued by the coherence protocol
  - Each miss generates a snoop
- However, snoops only carry a PA. No information of the virtual space, or source process, is preserved
  - Different runs of the same process, produce uncorrelated traces
  - Detection of hot objects is not straightforward
  - Attributing memory accesses to processes requires non trivial memory allocation manipulation

BOSTON UNIVERSITY

Reverse MMU

- It's unthinkable to perform a PA to VA translation for each LLC miss
  - Requires a complete exploration of the page tables of the target process
  - Number of memory accesses explodes
- **FRACTAL** follows the MMU translation for a given process and VAs' range
- At each leaf entry, it reconstructs the VA associated
- VA, ASID, and PA are stored in a lookaside buffer, reverse TLB (rTLB)

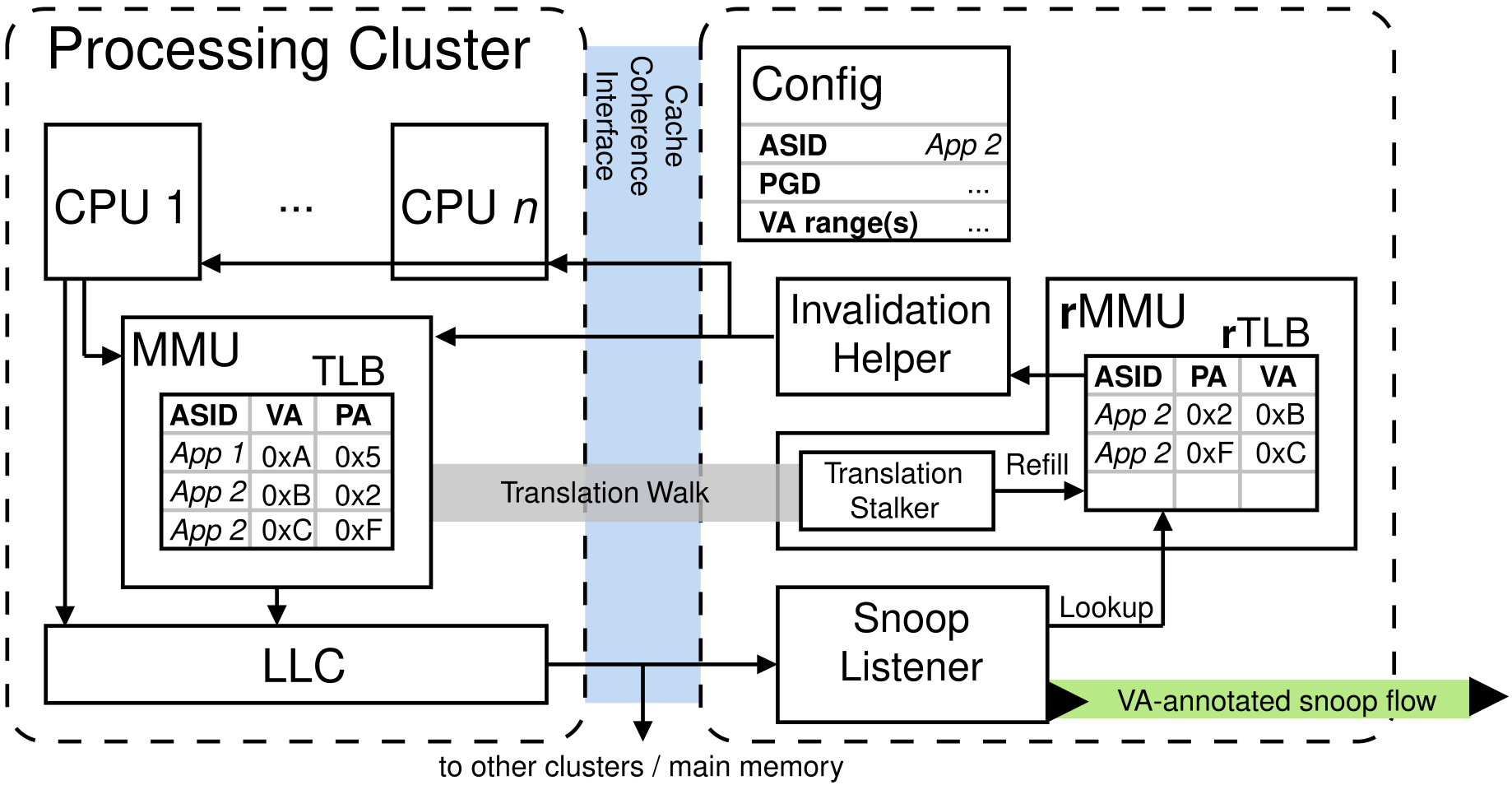
BOSTON UNIVERSITY

Enhanced snoops

- At each received snoop, the PA is looked up in the rTLB
- If it hits, an **enhanced snoop** is generated, annothing the original with the corresponding VA and ASID
- The result is then forwarded to the module in charge of the use case of interest, e.g., a logger

BOSTON UNIVERSITY

FRACTAL Design



Observability granularity

- **FRACTAL** monitoring based on LLC misses
- Such trace could be insufficient for use cases requiring all memory accesses, i.e. miss **and** hits, as in a instruction-level tracing
- Fully exploiting the coherence interface **FRACTAL** can issue, with a tunable overhead, invalidation messages for cache lines of interest or finer grain monitoring
- This results in a snoops for every access to the targeted lines

BOSTON UNIVERSITY

Outline

BOSTON UNIVERSITY

Why another

- Platform independent
  - No specialized HW required
- Negligible overhead
- High configurability
- Support for multiple ranges and processes

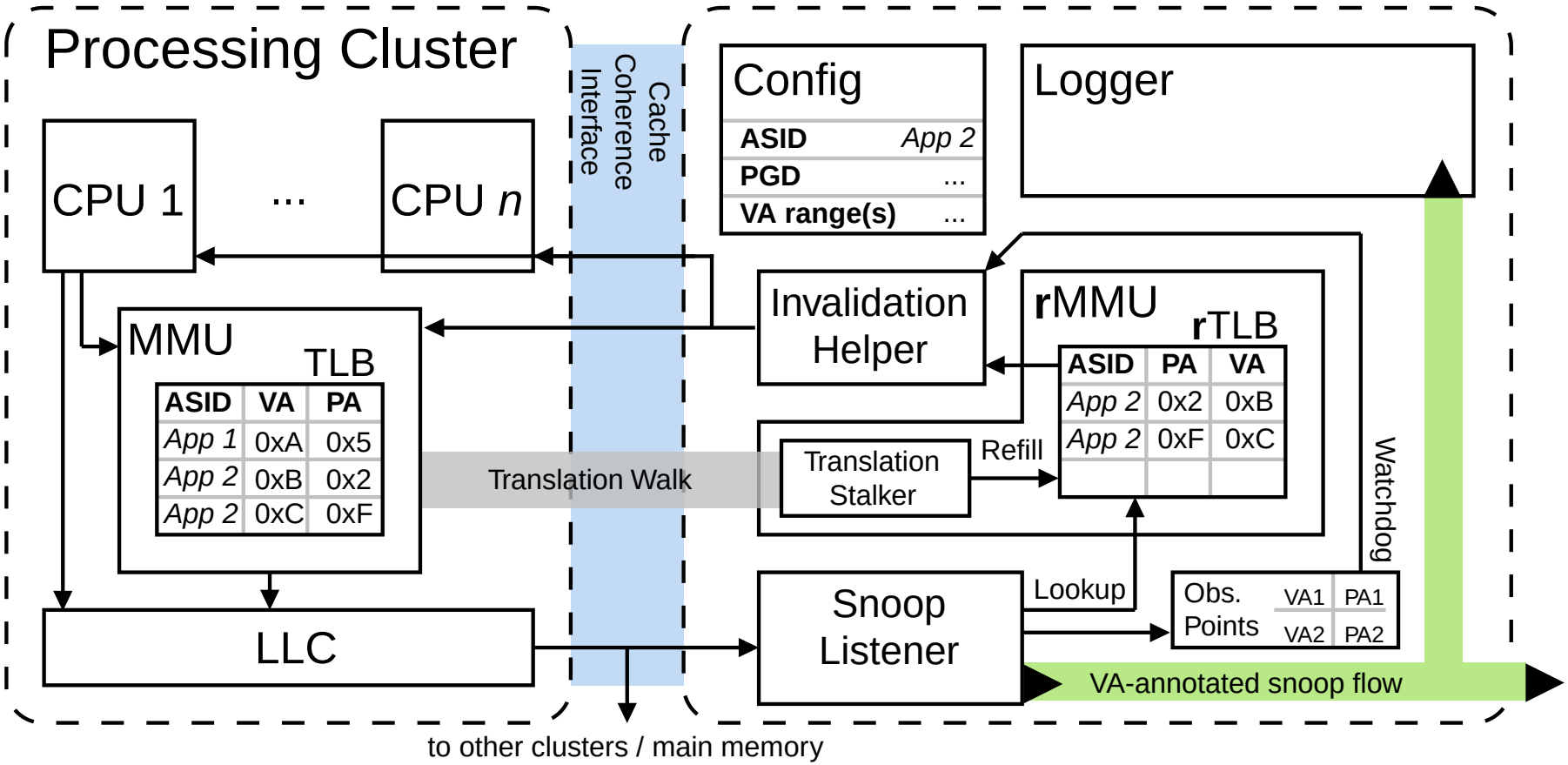
BOSTON UNIVERSITY

How to log with FRACTAL

- The enhanced snoops are forwarded to a logger module
- The module adds additional metadata, e.g., timestamp
- Finally, the resulting entry is written to any destination
  - In our prototype, a BRAM block

BOSTON UNIVERSITY

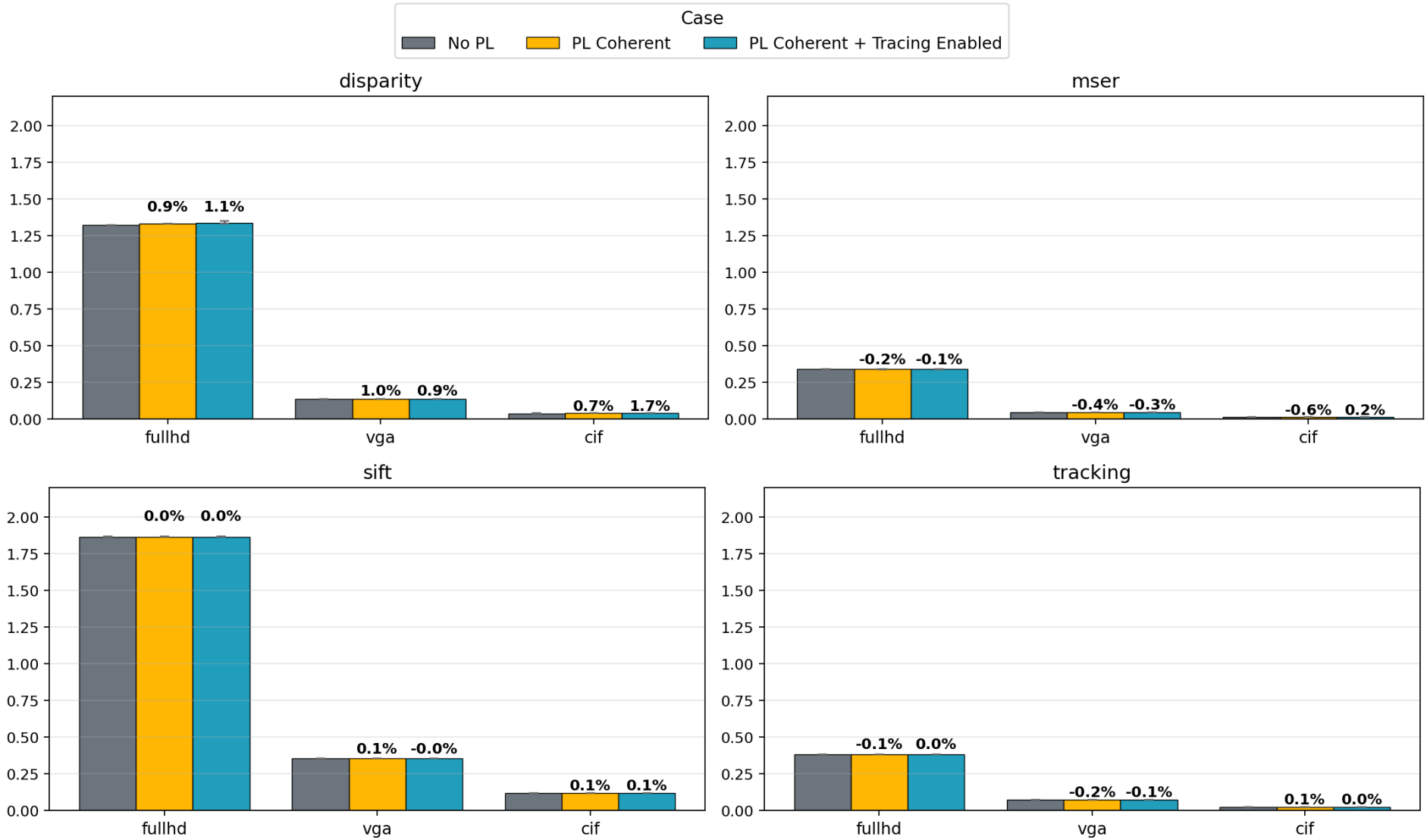
Logging Design



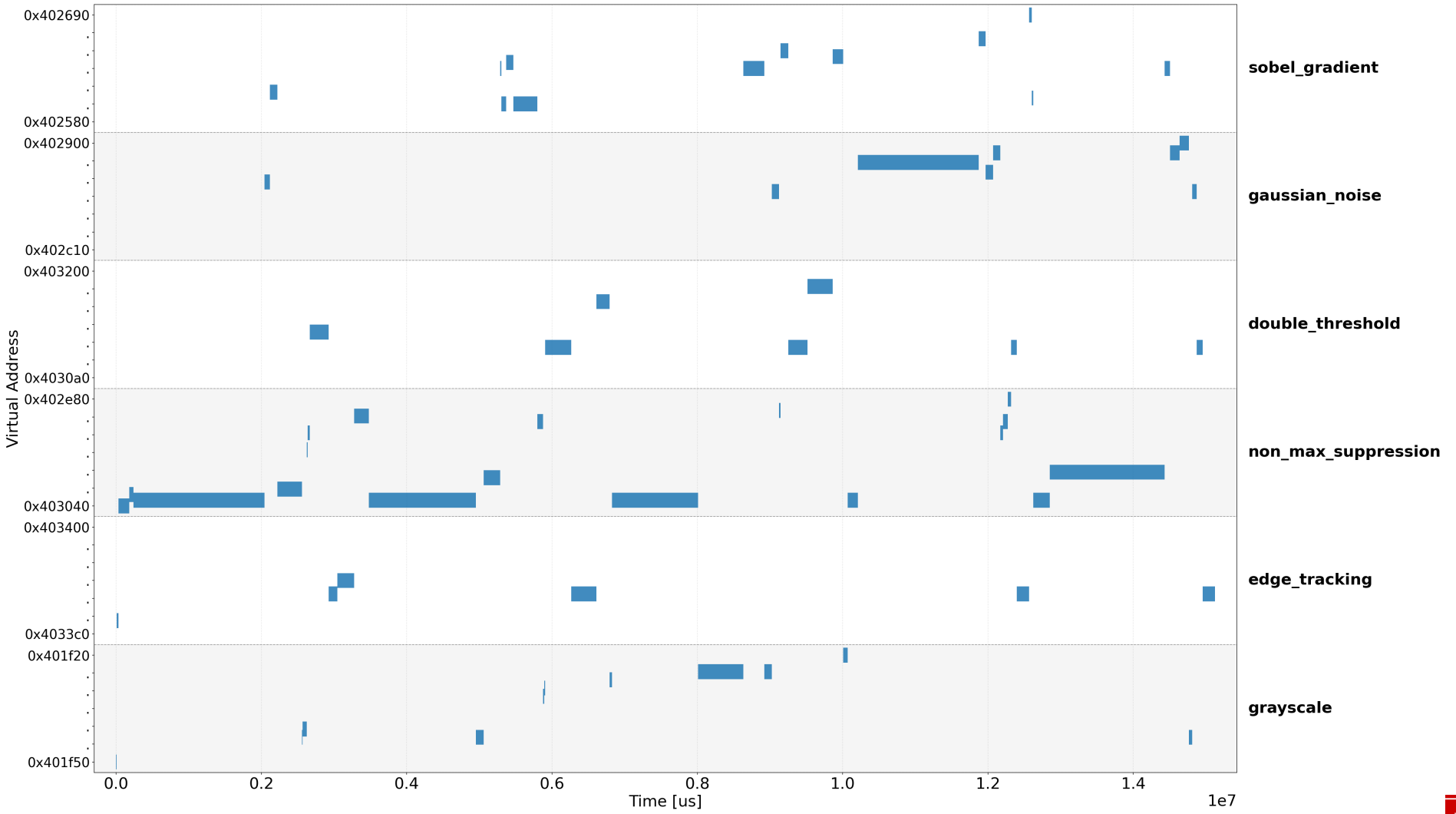
Outline

BOSTON UNIVERSITY

Overhead


Case
No PL
PL Coherent
PL Coherent + Tracing Enabled
disparity
0.9% 1.1%
1.0% 0.9%
0.7% 1.7%
mser
-0.2% -0.1%
-0.4% -0.3%
-0.6% 0.2%
sift
0.0% 0.0%
0.1% -0.0%
0.1% 0.1%
tracking
-0.1% 0.0%
-0.2% -0.1%
0.1% 0.0%
fullhd
vga
cif
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00

Traceability



Future steps

- Exploring FRACTAL for processes' milestones progression assestment
- Low impact watchdog for security monitoring
- Consolidationg code base for open source release

BOSTON UNIVERSITY