SPECCast

A Methodology for Fast Performance Evaluation with SPEC CPU 2017 Multiprogrammed Workloads

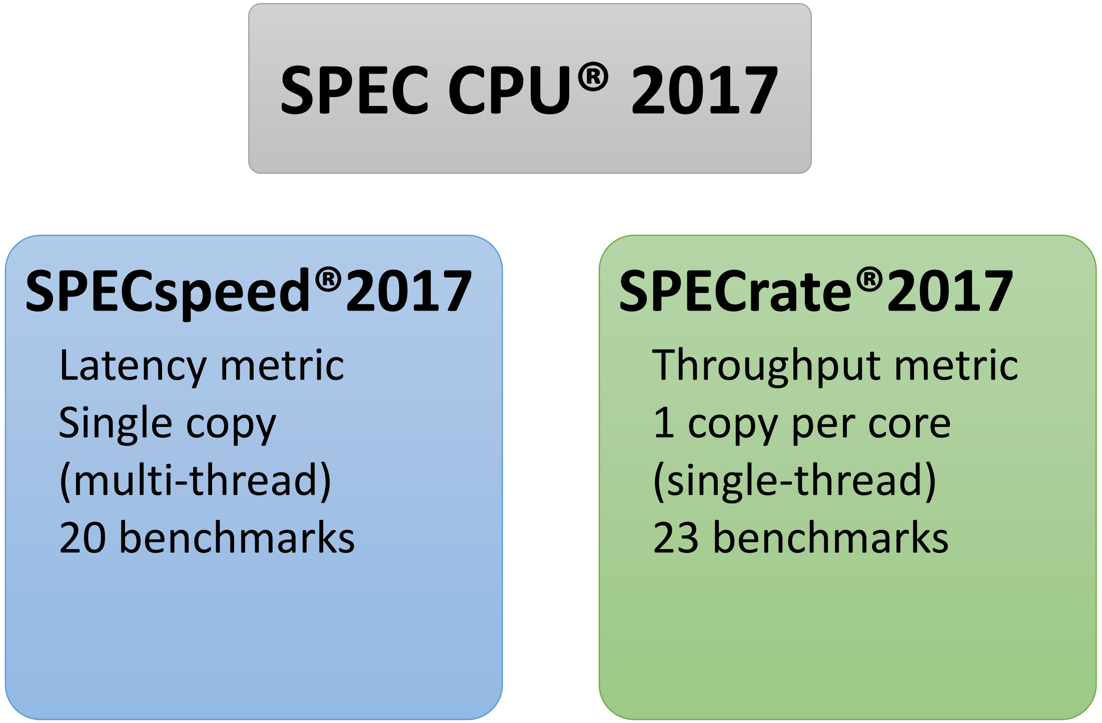
Pablo Prieto, Pablo Abad, Jose Angel Herrero, Jose Angel Gregorio, Valentin Puente

{prietop,abadp,herreroja,monaster,vpuente}@unican.es

UNIVERSIDAD DE CANTABRIA

1. Introduction & Motivation

- Benchmarking is the common practice for Computer Performance Evaluation.
- SPEC suites are the most popular.



SPEC CPU® 2017

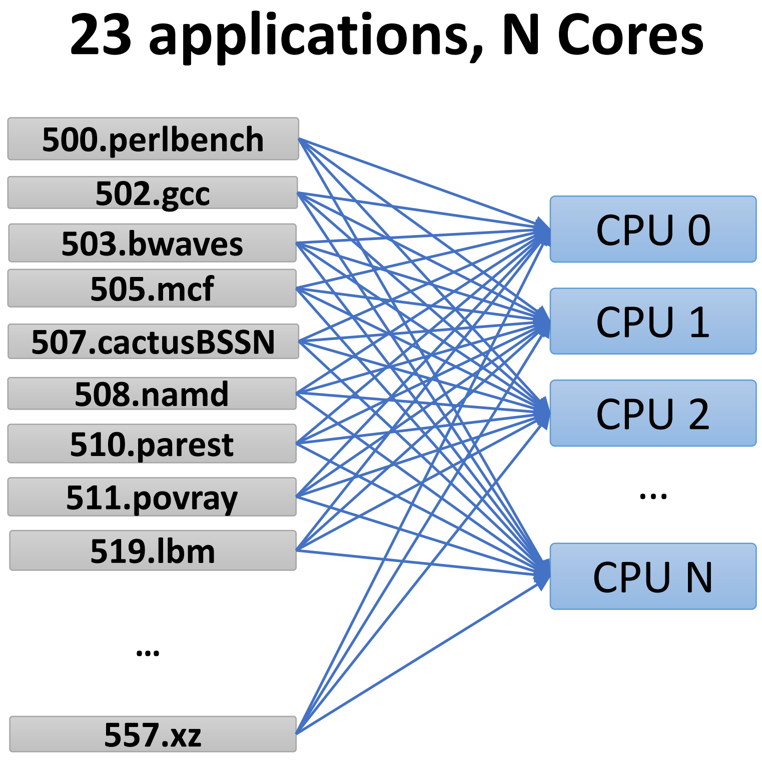
SPECspeed®2017
Latency metric
Single copy
(multi-thread)
20 benchmarks

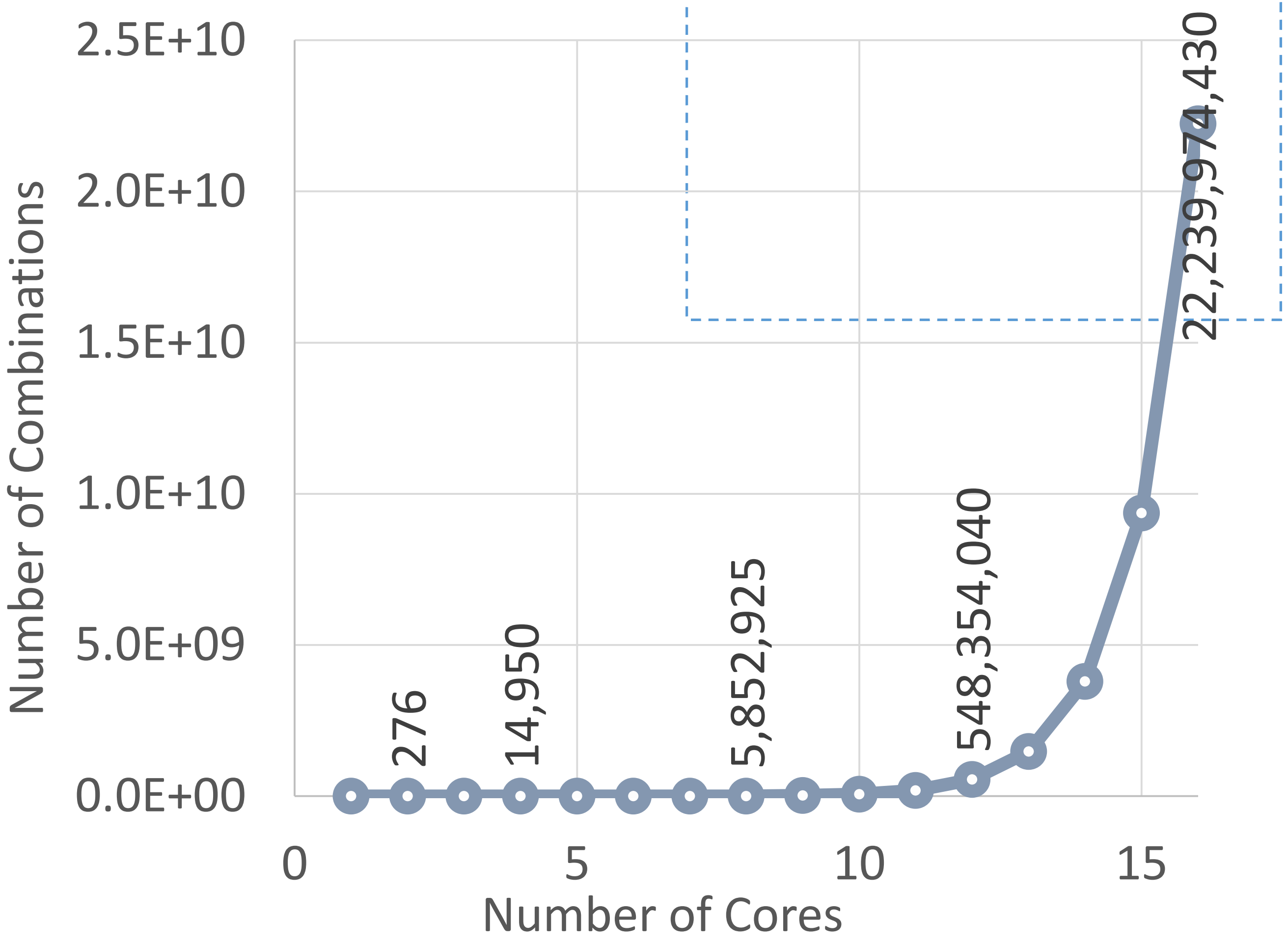
SPECrate®2017
Throughput metric
1 copy per core
(single-thread)
23 benchmarks

- Only 23 benchmarks
- Homogeneous
  - Not representative

GO HETEROGENEOUS!

1. Introduction & Motivation

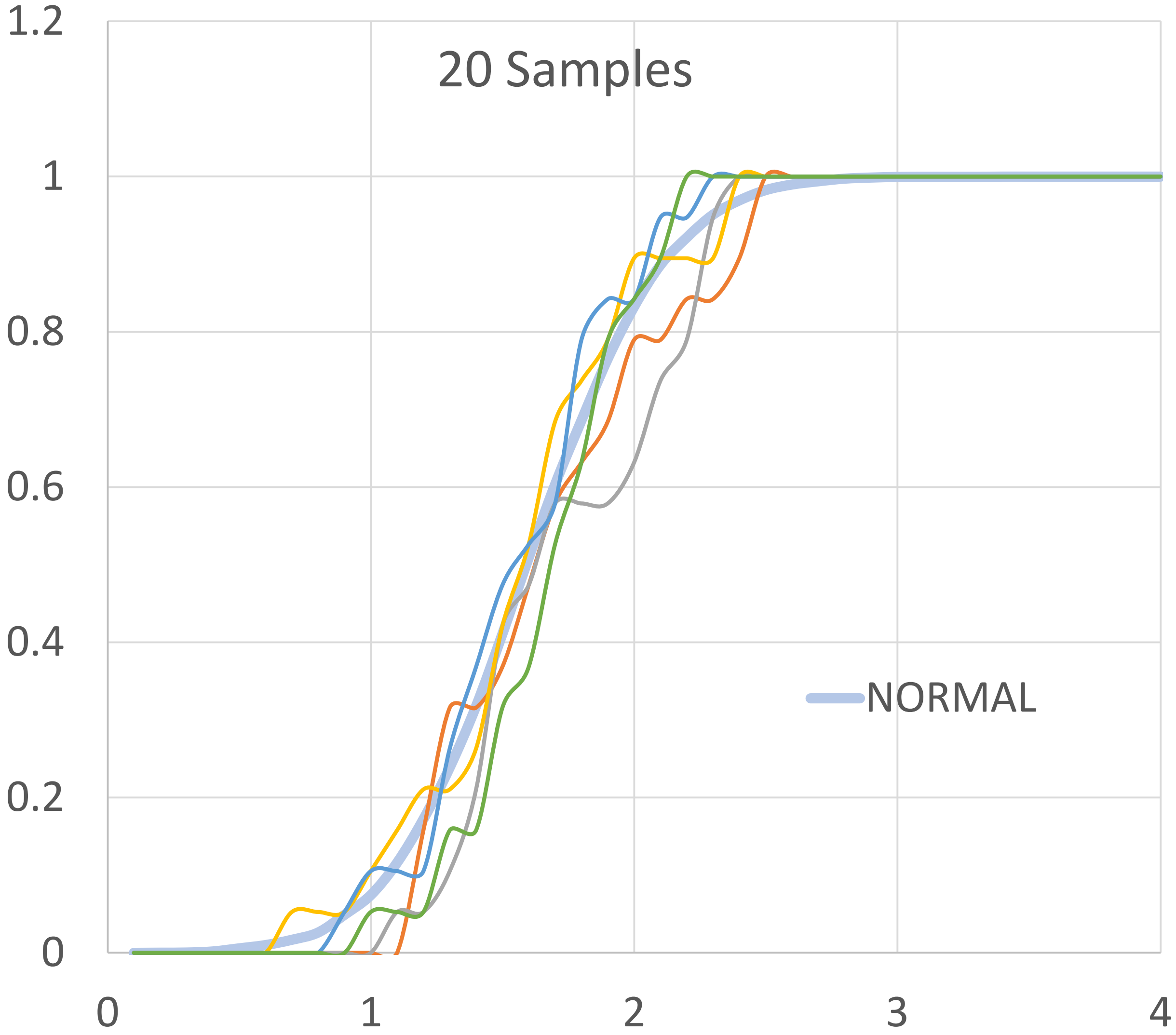


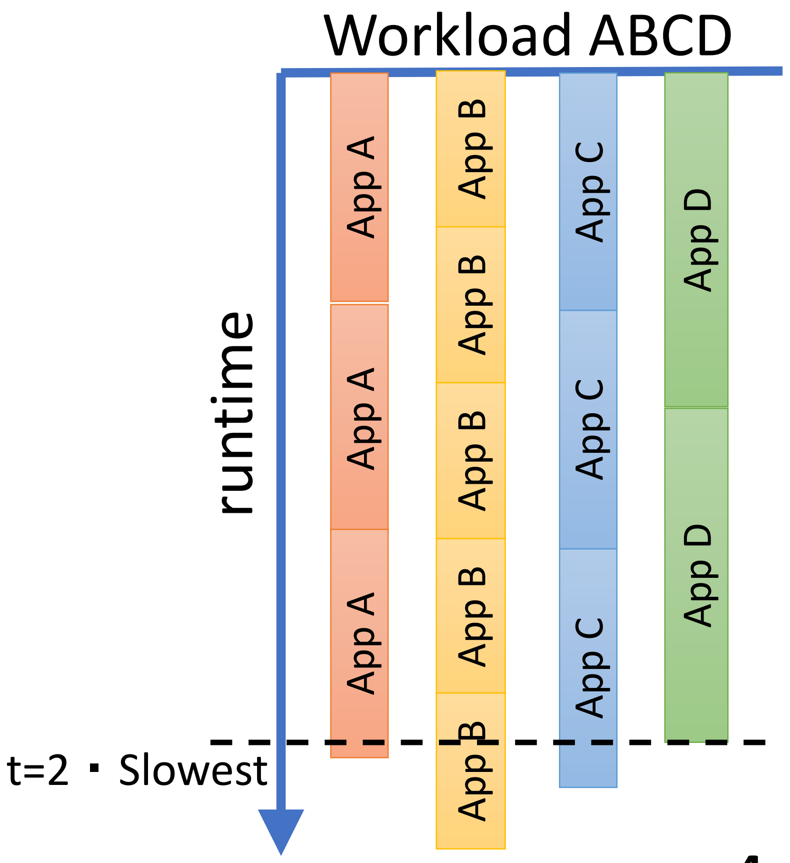


Millions of Workloads → Statistical Analysis

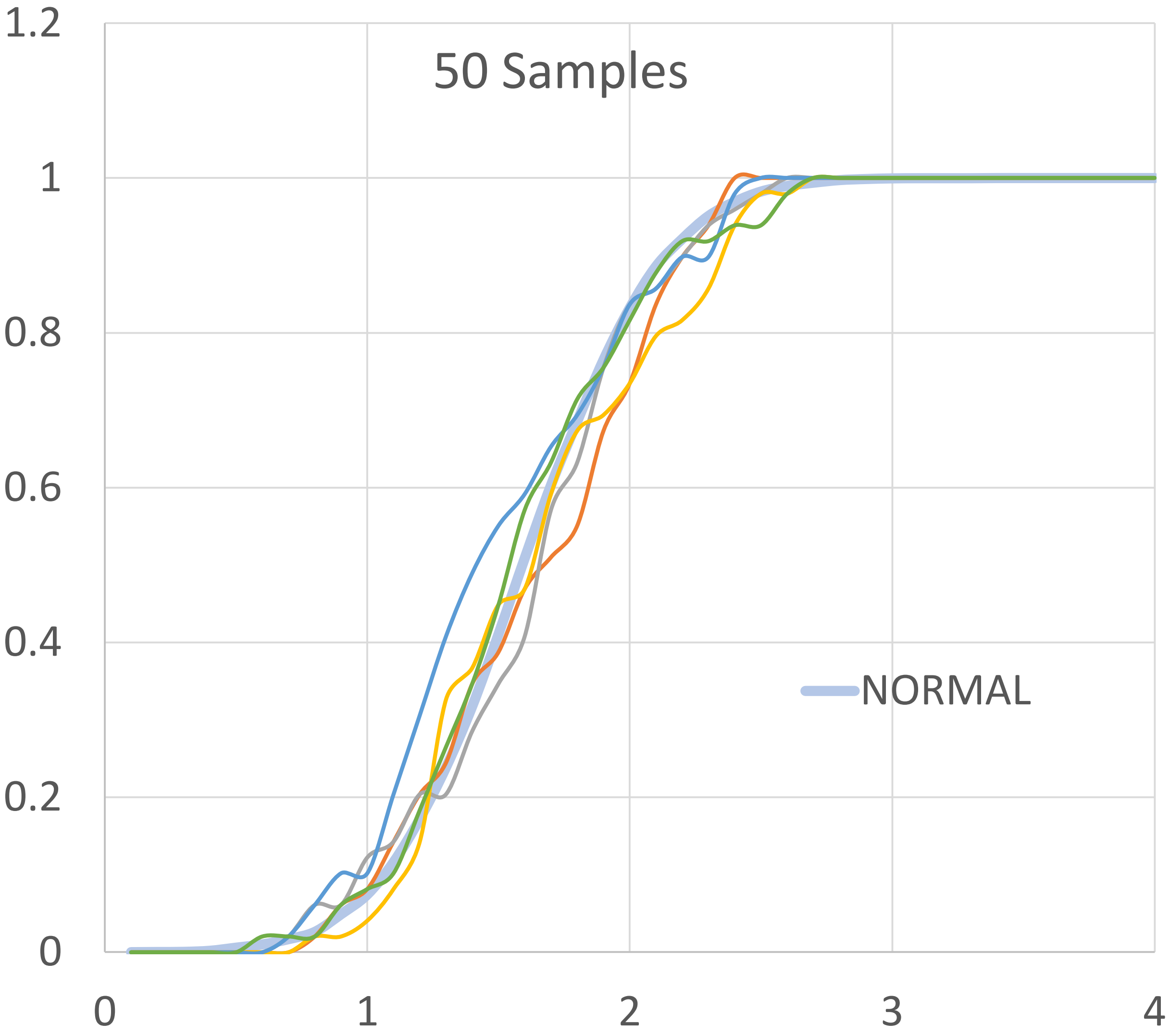
(How many to be statistically meaningful?)

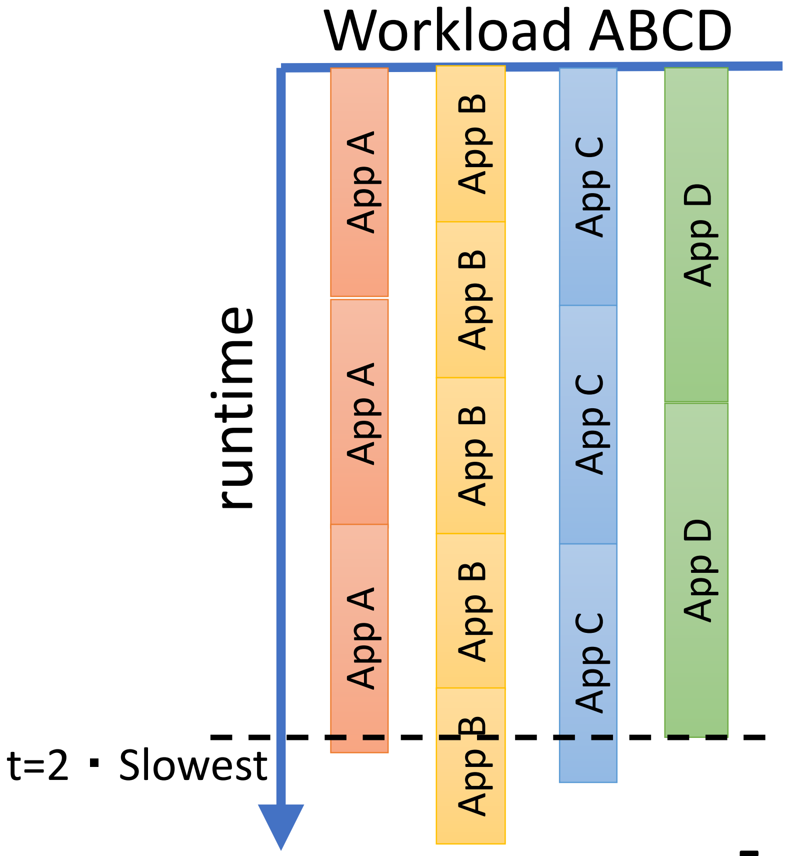
1. Introduction & Motivation



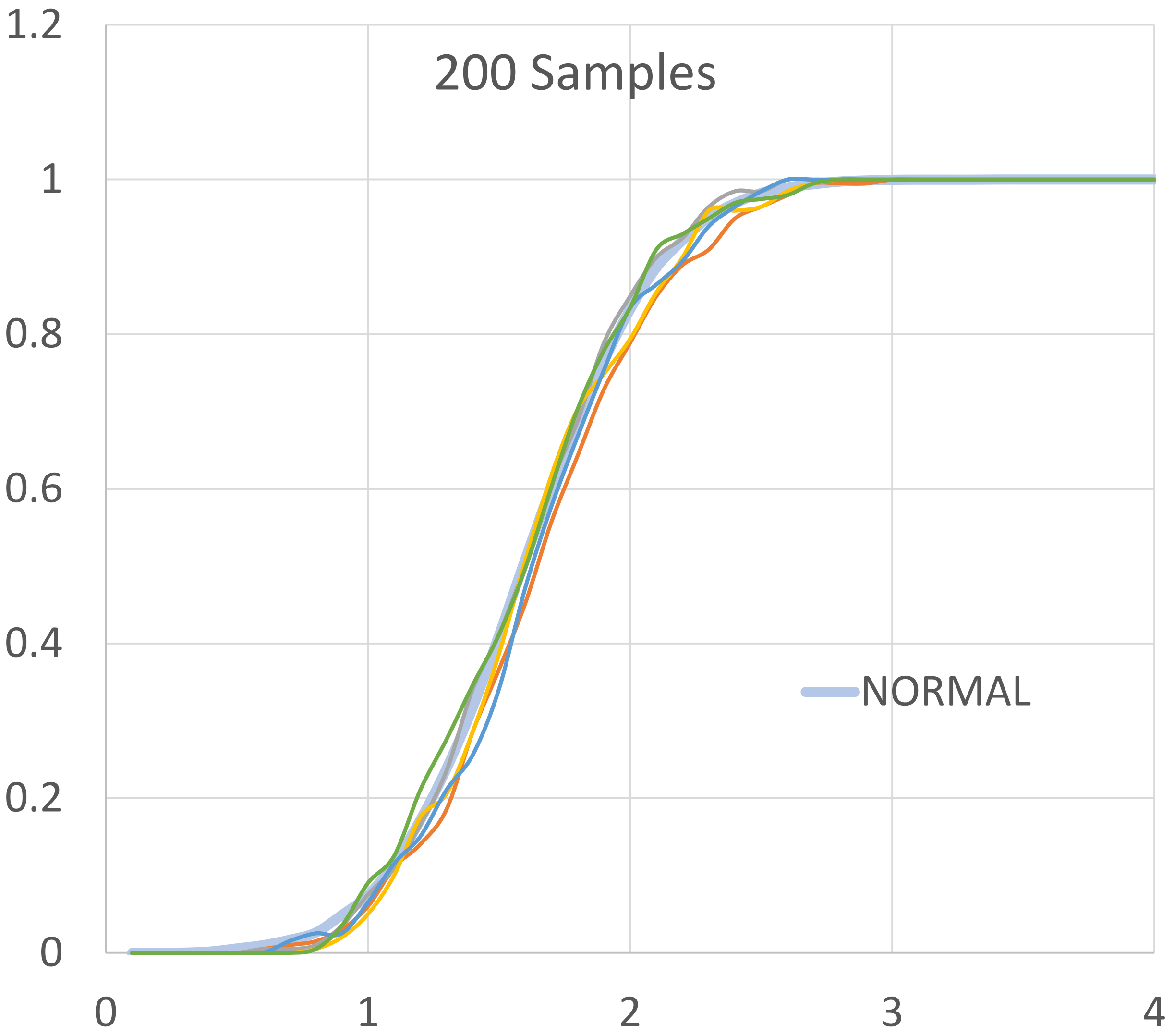


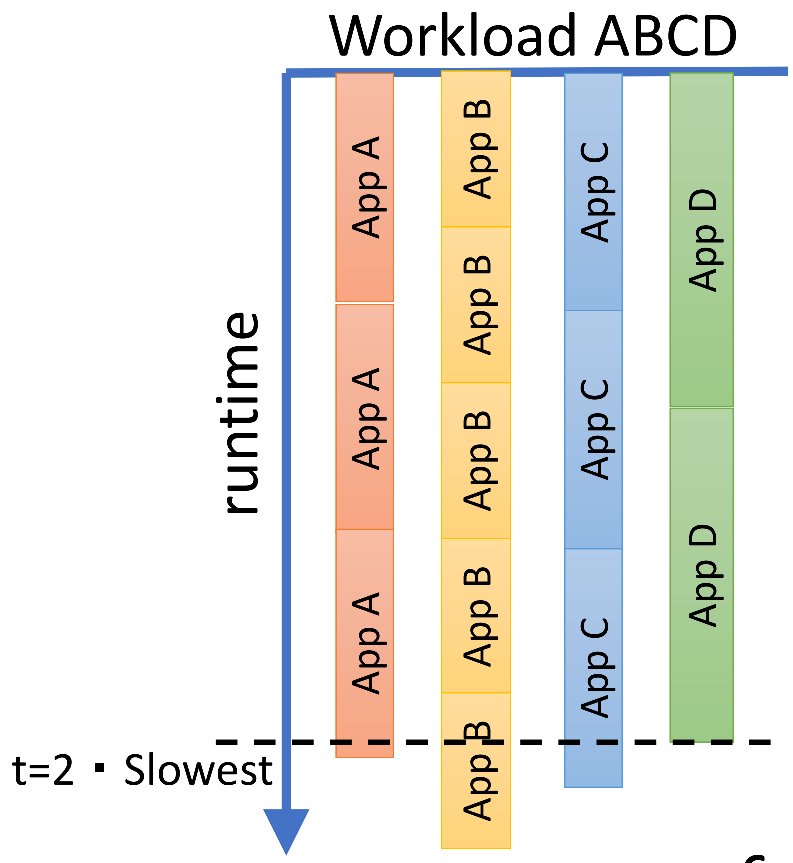
1. Introduction & Motivation



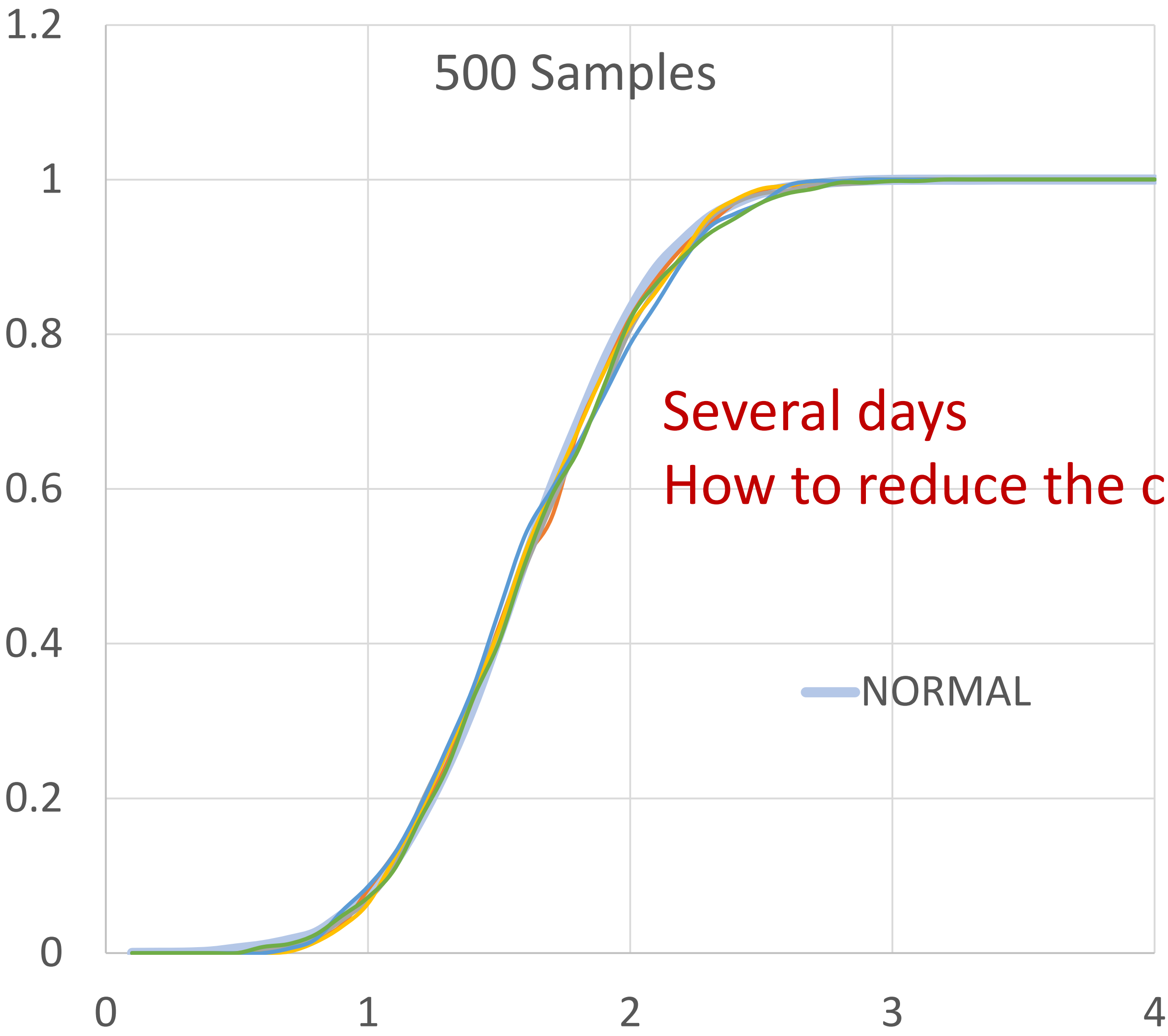


1. Introduction & Motivation



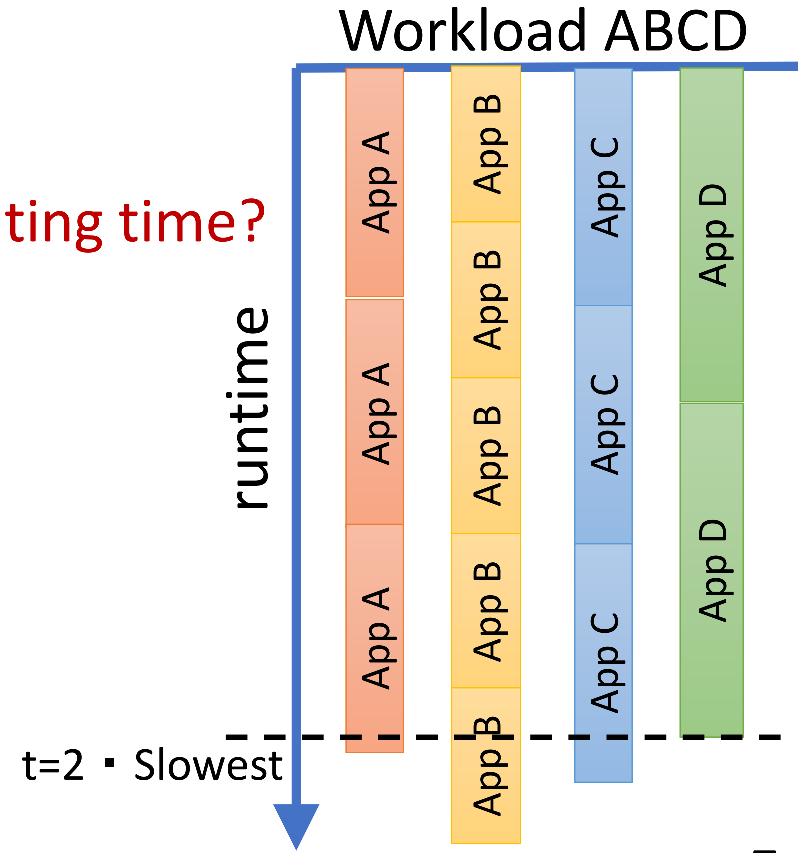


1. Introduction & Motivation



Several days

How to reduce the computing time?

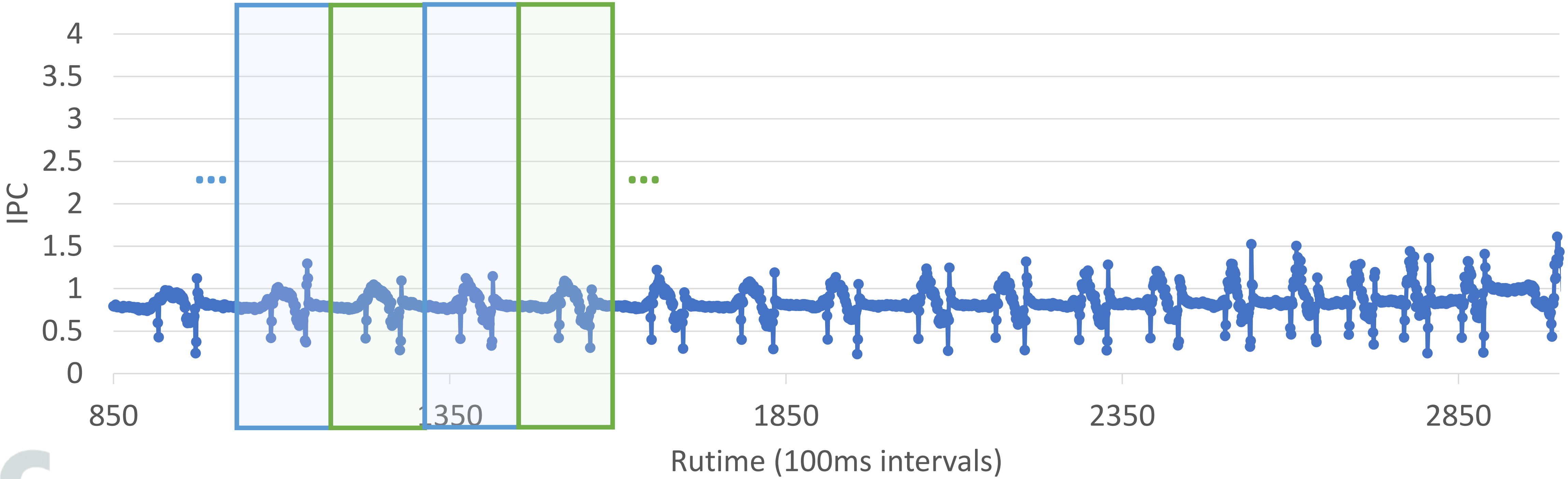


1. Introduction & Motivation

SAMPLING

- Most applications have a loop-based ROI
- One or few iterations are representative of the whole app
  - Example: CactuBSSN (runtime detail)

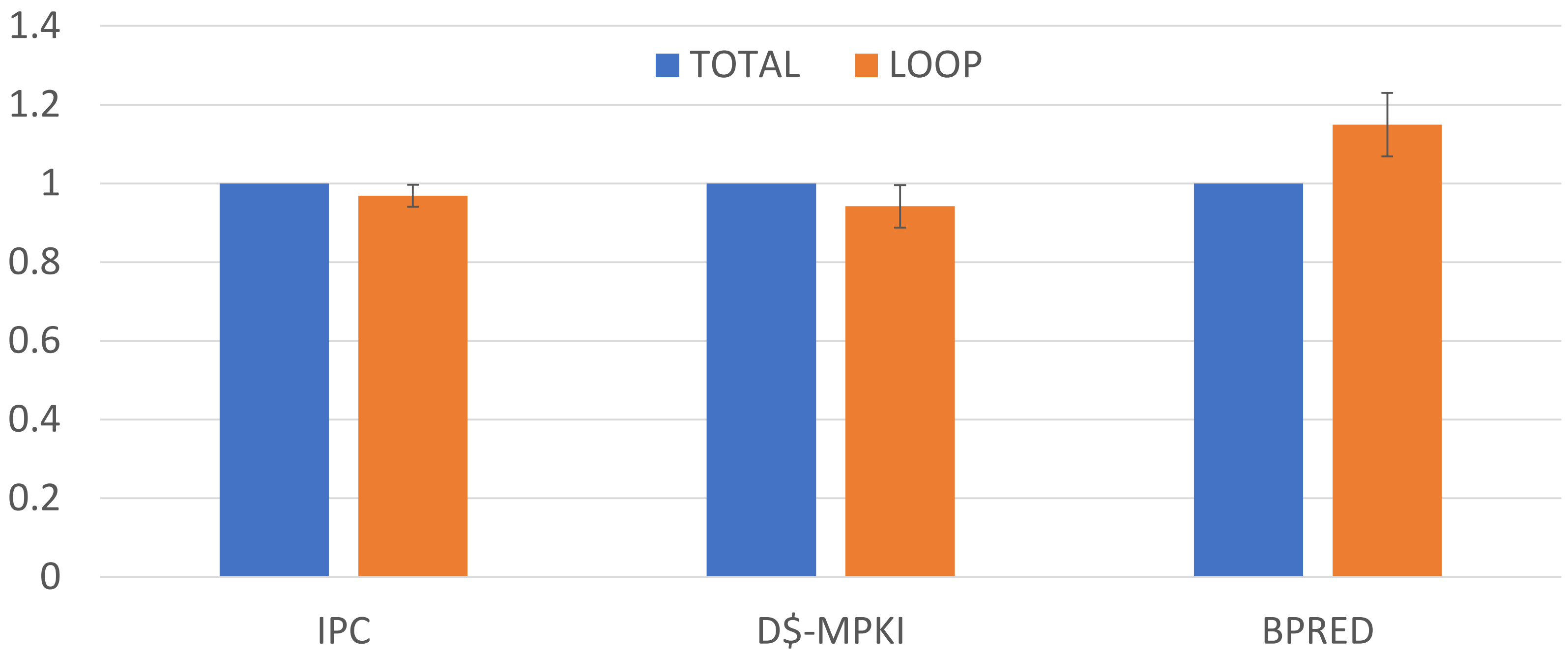
Loop iterations behave “architecturally” similar…



1. Introduction & Motivation

SAMPLING

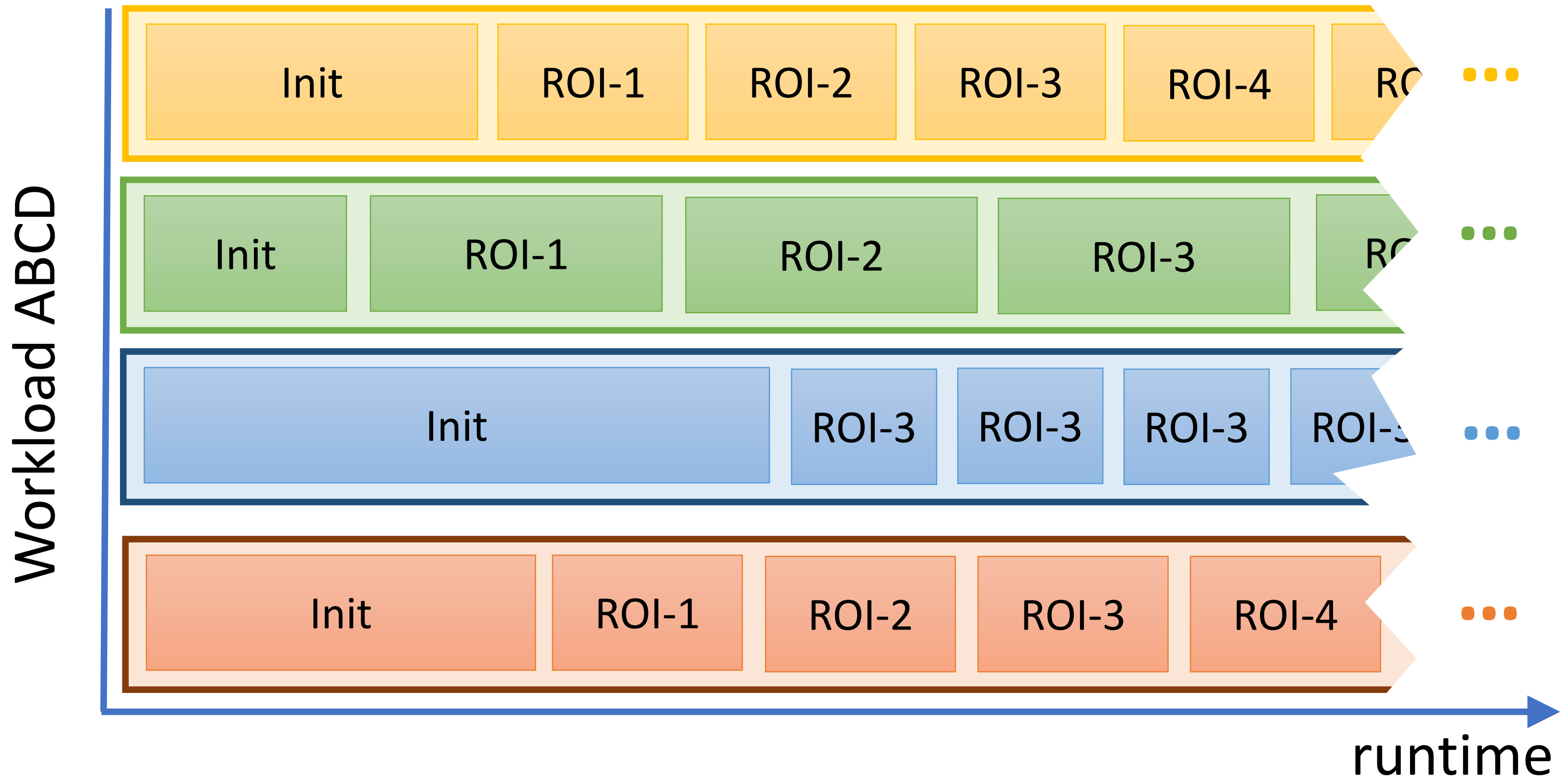
- Most applications have a loop-based ROI
- One or few iterations are representative of the whole app
- A significant degree of similarity is persistent across all SPEC applications.



1. Introduction & Motivation

SYNCHRONOUS EXECUTION

- Applications should execute their ROIs to be meaningful

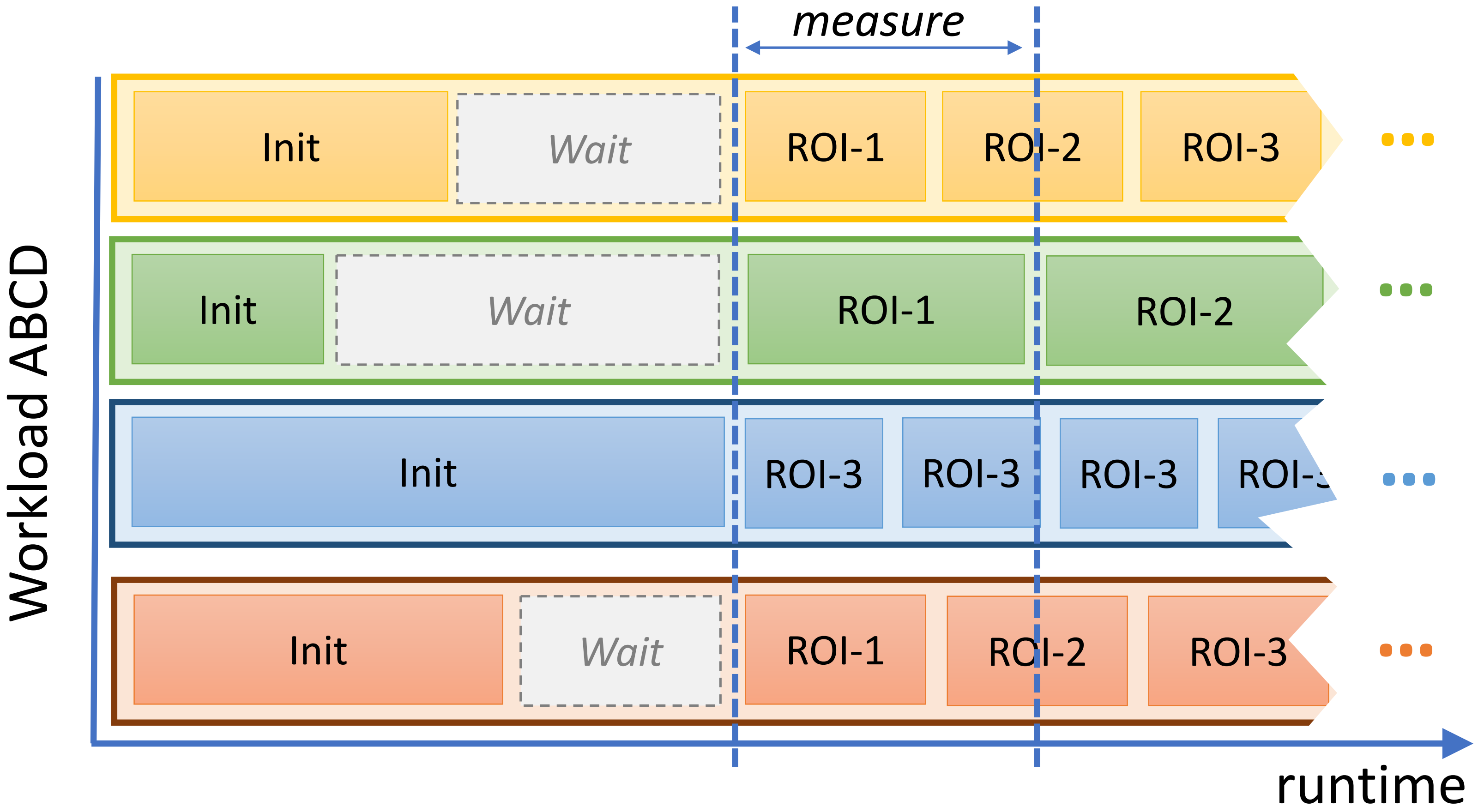


From this…

1. Introduction & Motivation

SYNCHRONOUS EXECUTION

- Applications should execute their ROIs to be meaningful

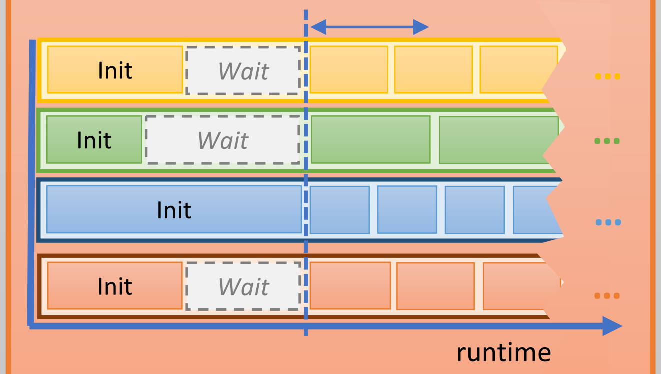


…to this

2. SPECCast

SPECcast

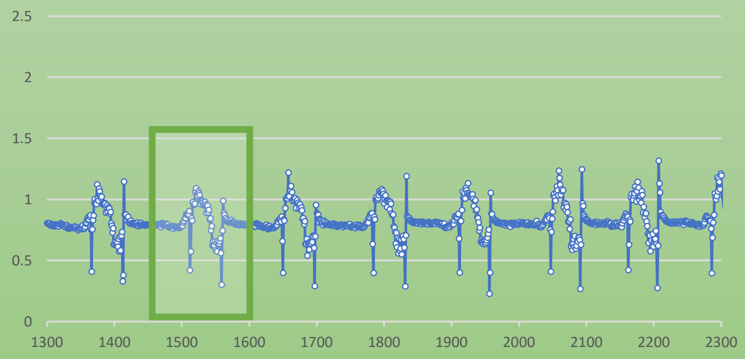
Synchronous Execution



PMU-based Profiling[1]

Instructions/Cycles
Branches/Branch misses
L1-LLC loads/stores/misses
…
Top-Down
…

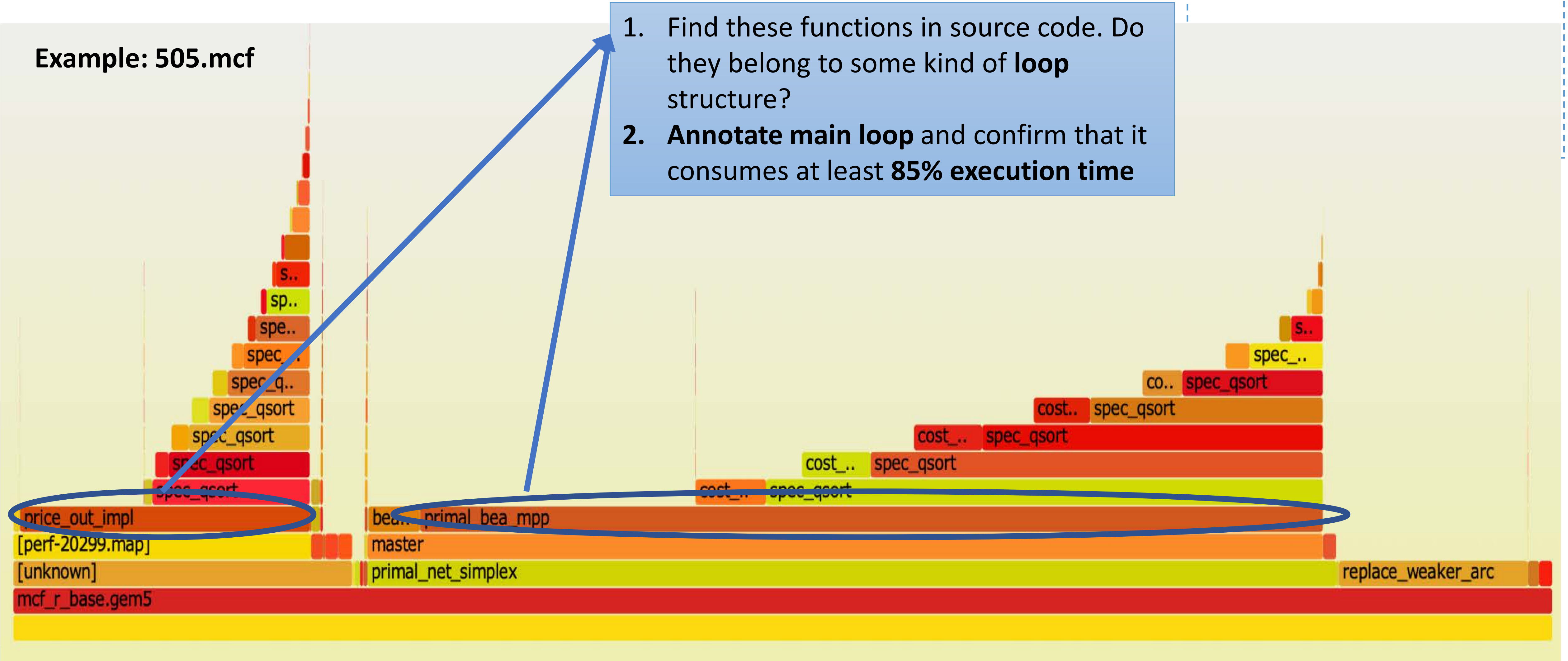
Sampling



[1] Terpstra, D., Jagode, H., You, H., Dongarra, J. "Collecting Performance Data with PAPI-C", 3rd Parallel Tools Workshop, Dresden, Germany, pp. 157-173, 2010.

2. SPECCast – Sampling (Flamegraph)

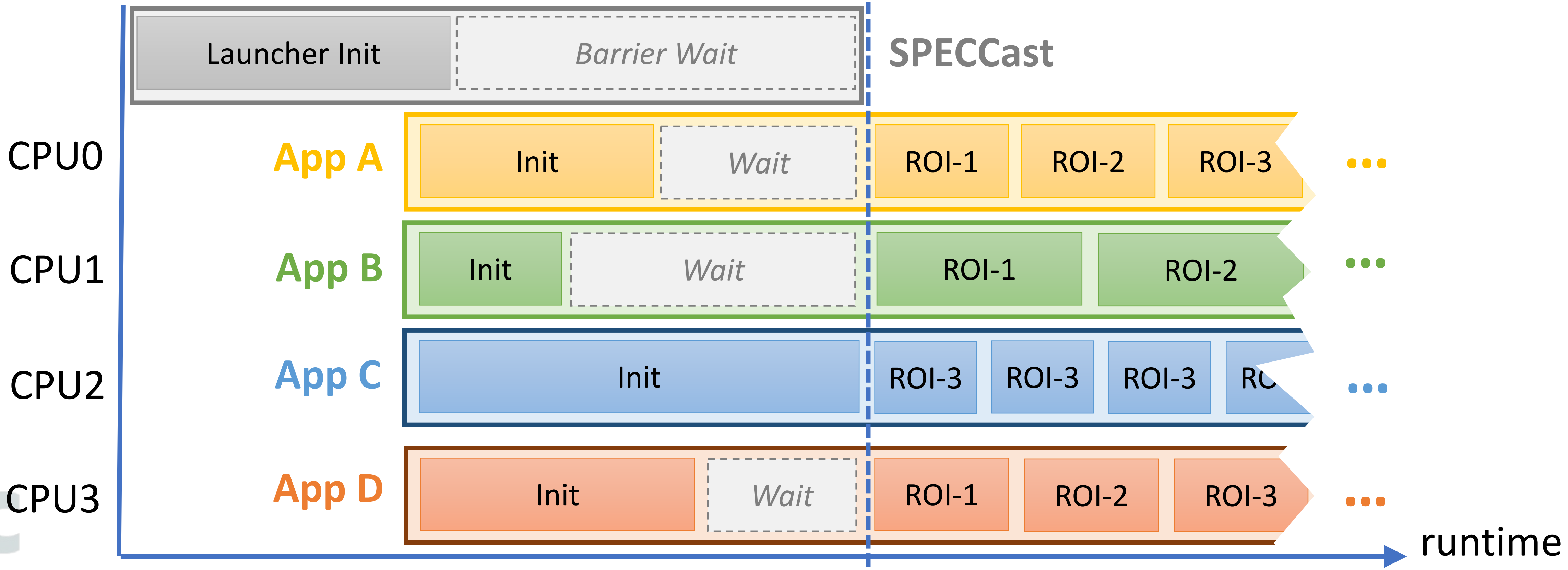
Example: 505.mcf



1. Find these functions in source code. Do they belong to some kind of **loop** structure?
2. **Annotate main loop** and confirm that it consumes at least **85% execution time**

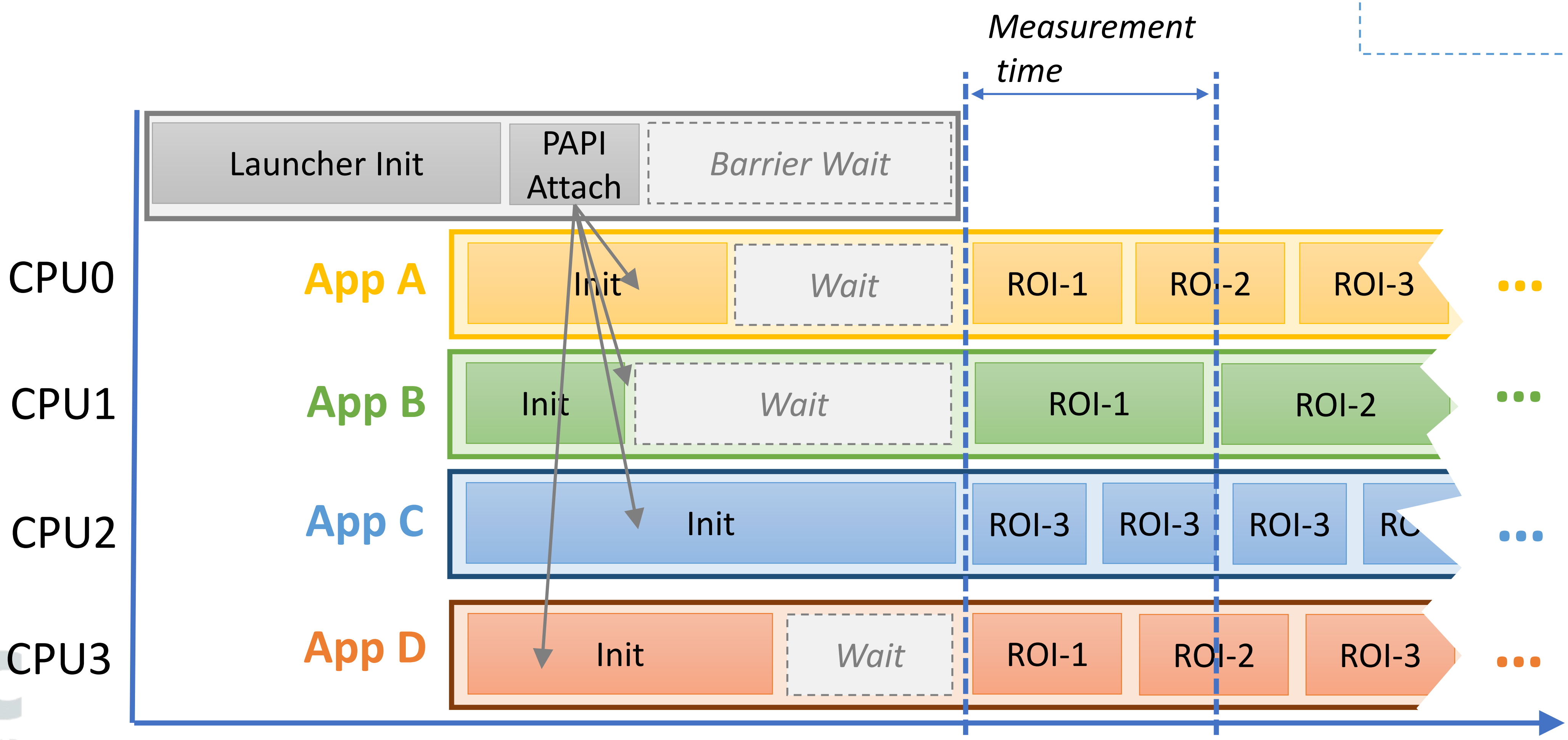
2. SPECCast - Synchronous Execution

- Code annotation to call shared barrier
- SPECCast launcher:
  - Each applications runs in a different core (setaffinity)
  - Apps wait in a barrier to simultaneously execute their ROIs



2. SPECCast – PMU Profiling

- Add PAPI Library to SPECCast launcher
- Define a measurement time and **PMU events**

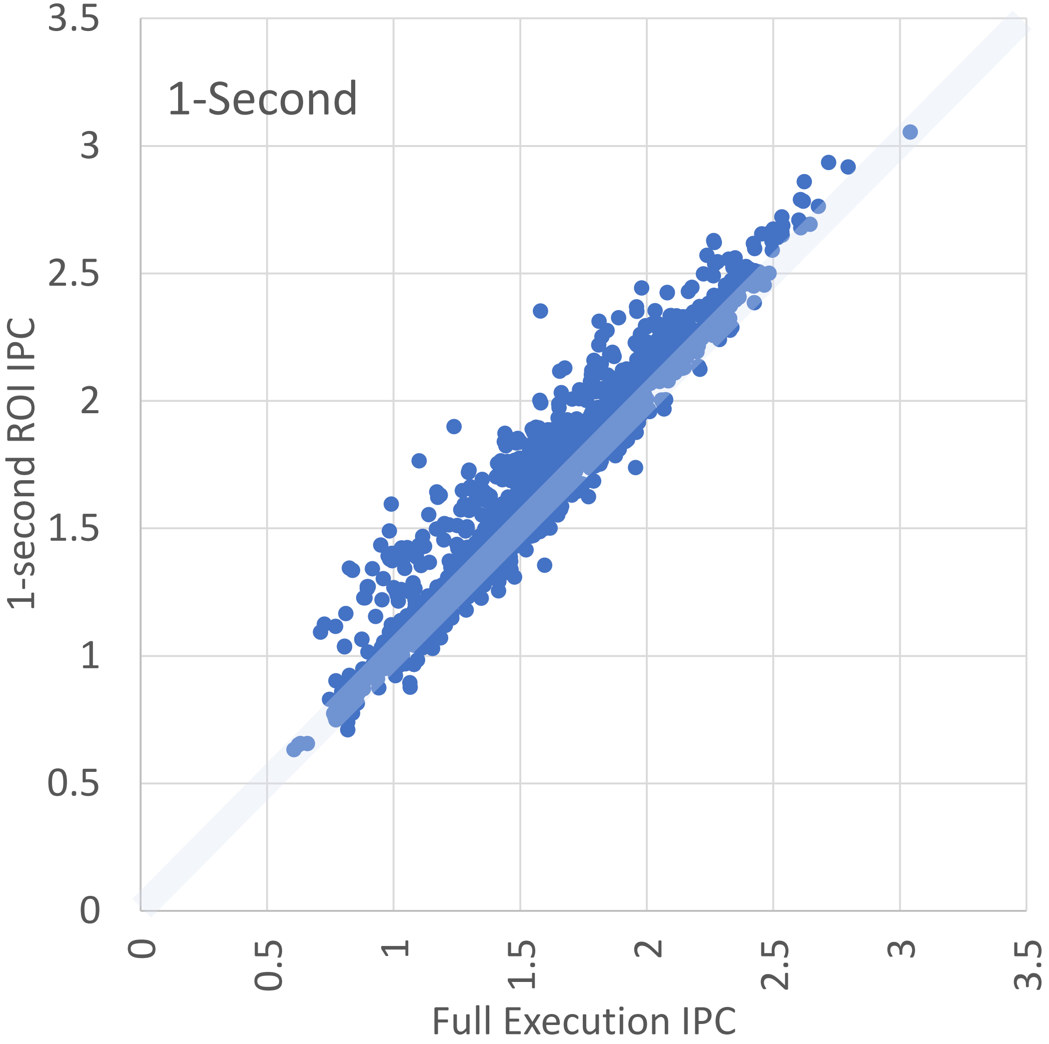


INTSTRUCTIONS
CYCLES
LOADS
STORES
L1 DATA$ MISSES/HITS
L1 INS$ MISSES/HITS
L3$ MISSES/HITS
PREFETCH MISSES
IDLE UNIT CYCLES
…

runtime

2. SPECCast - Computational Effort

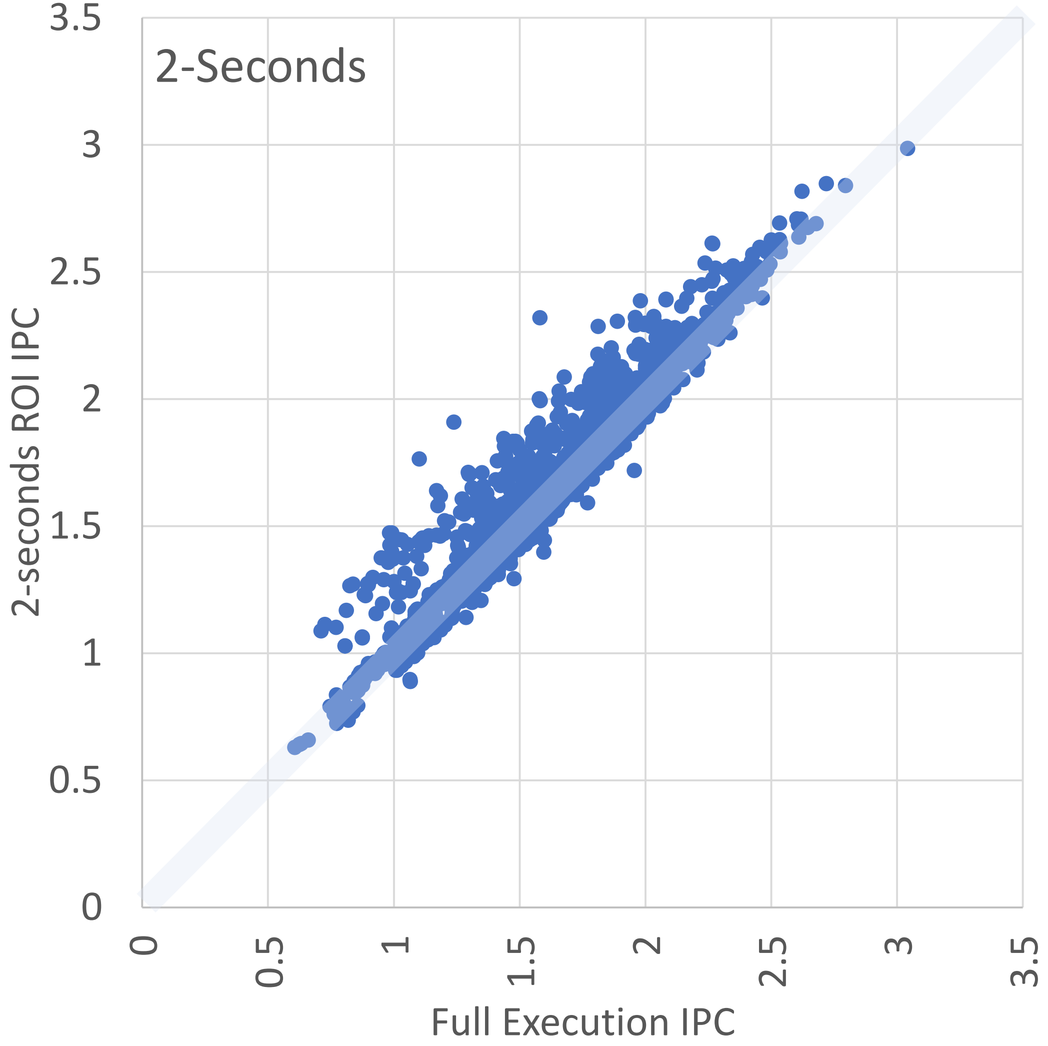
- How much "ROI Measurement Time" is necessary?



UNIVERSIDAD DE CANTABRIA

2. SPECCast - Computational Effort

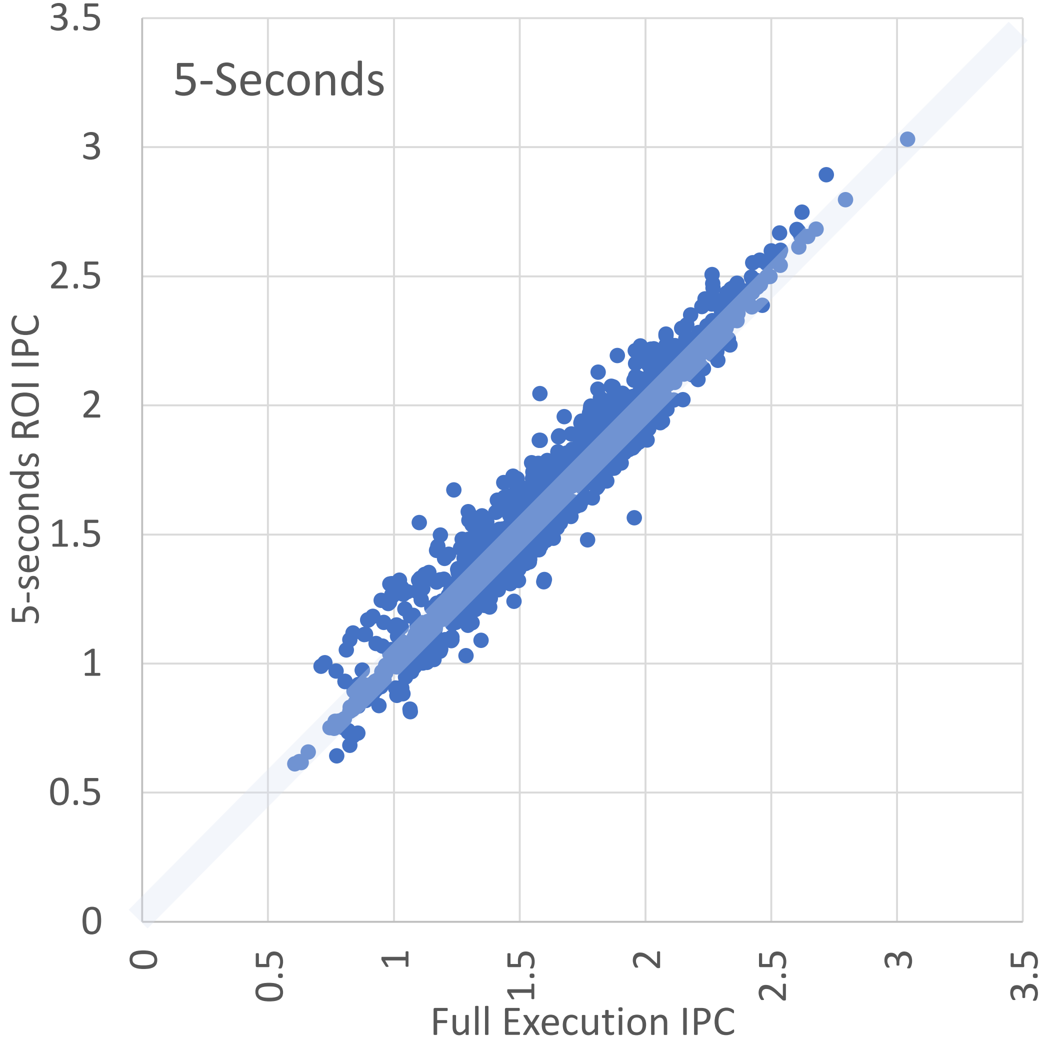
- How much “ROI Measurement Time” is necessary?



UNIVERSIDAD DE CANTABRIA

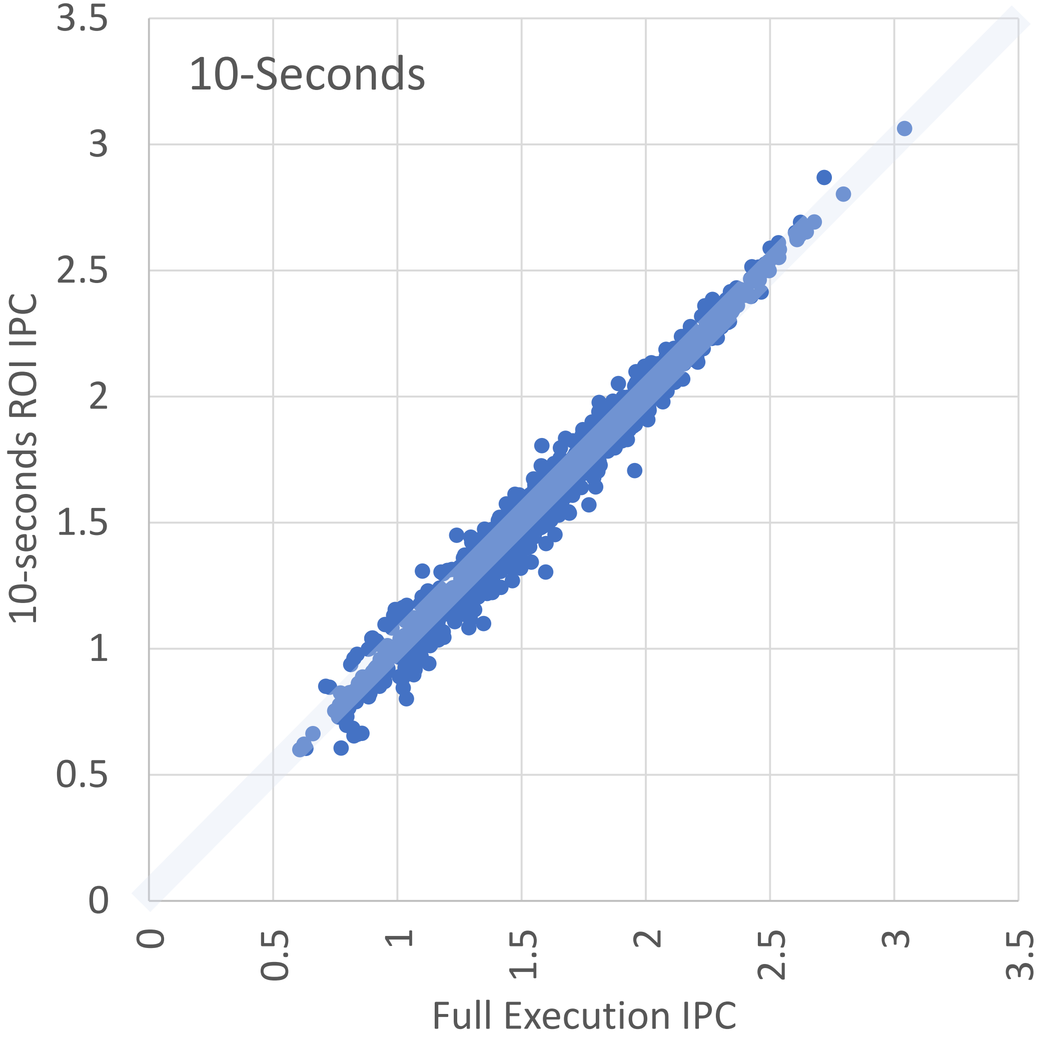
2. SPECCast - Computational Effort

- How much “ROI Measurement Time” is necessary?



2. SPECCast - Computational Effort

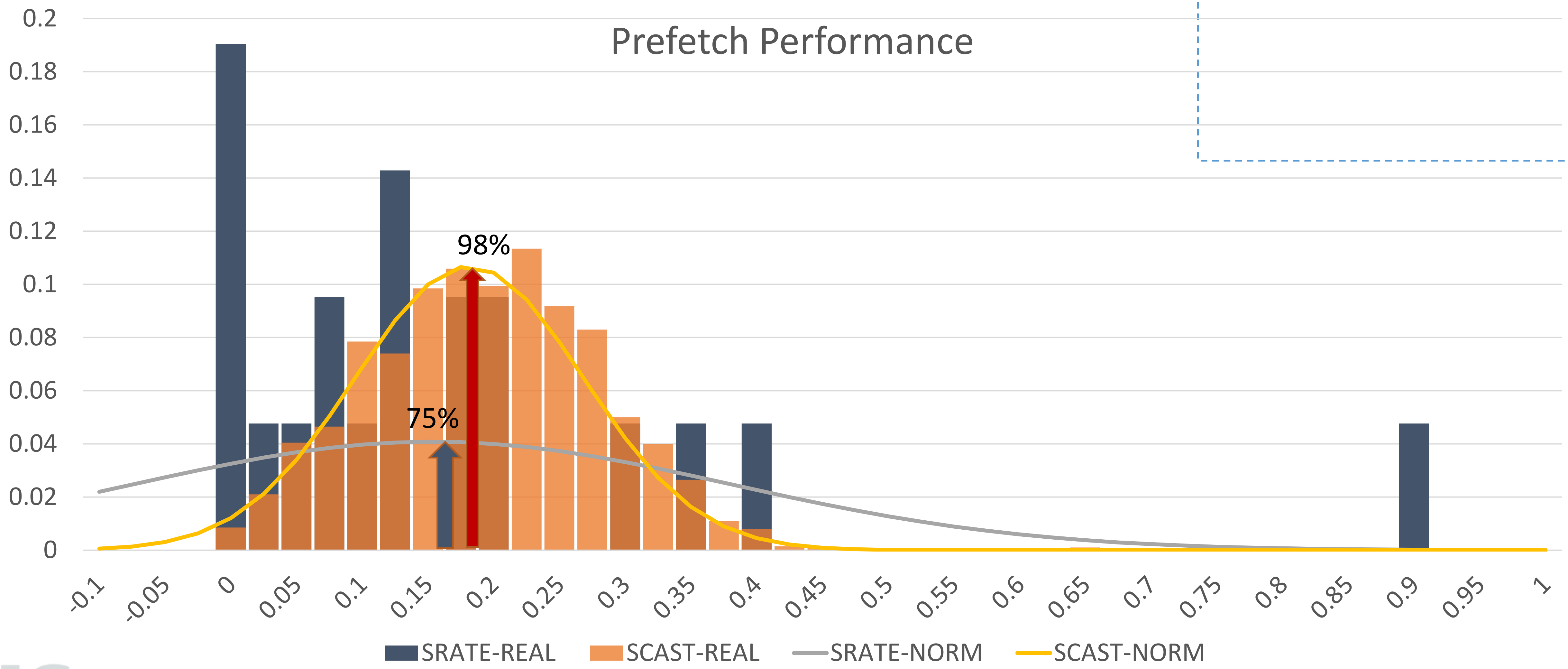
- How much “ROI Measurement Time” is necessary?



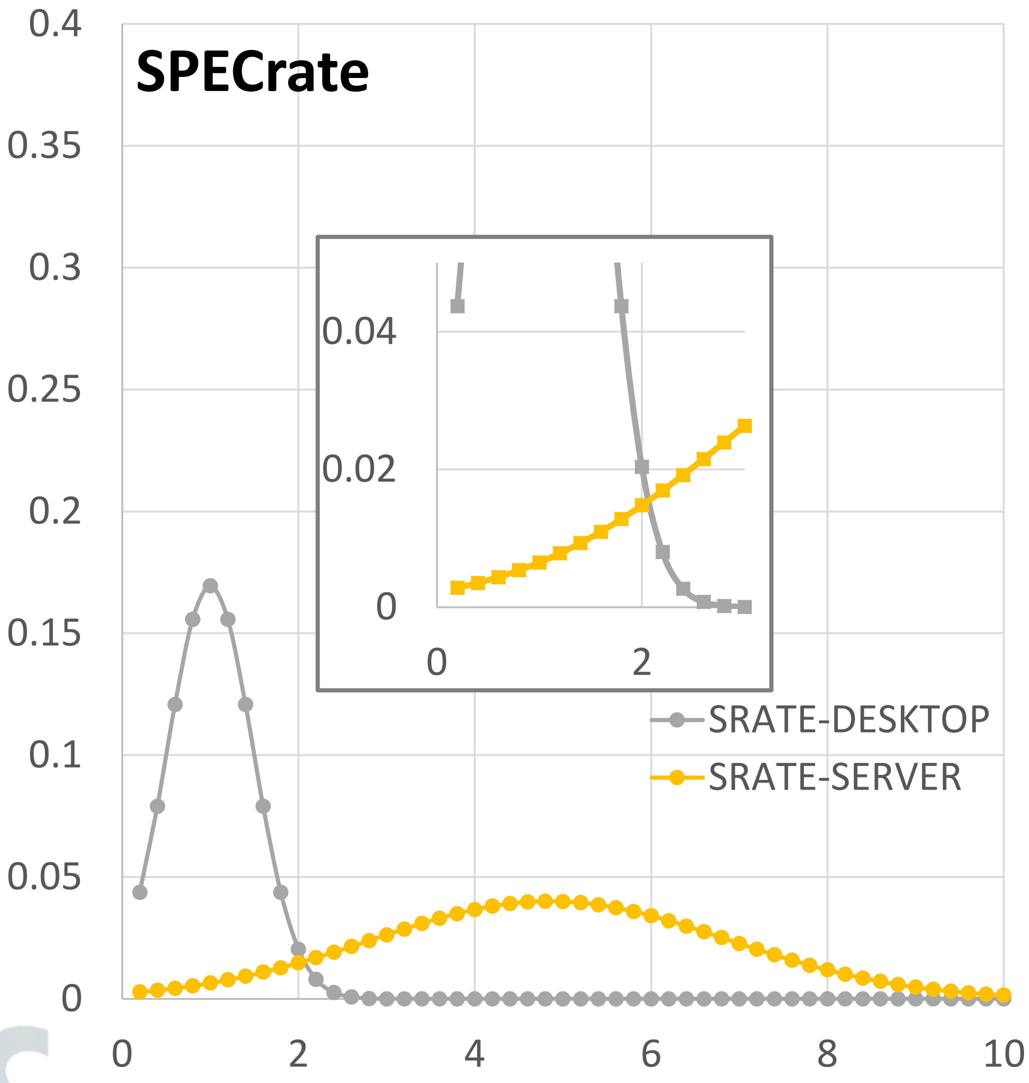
~20 times faster

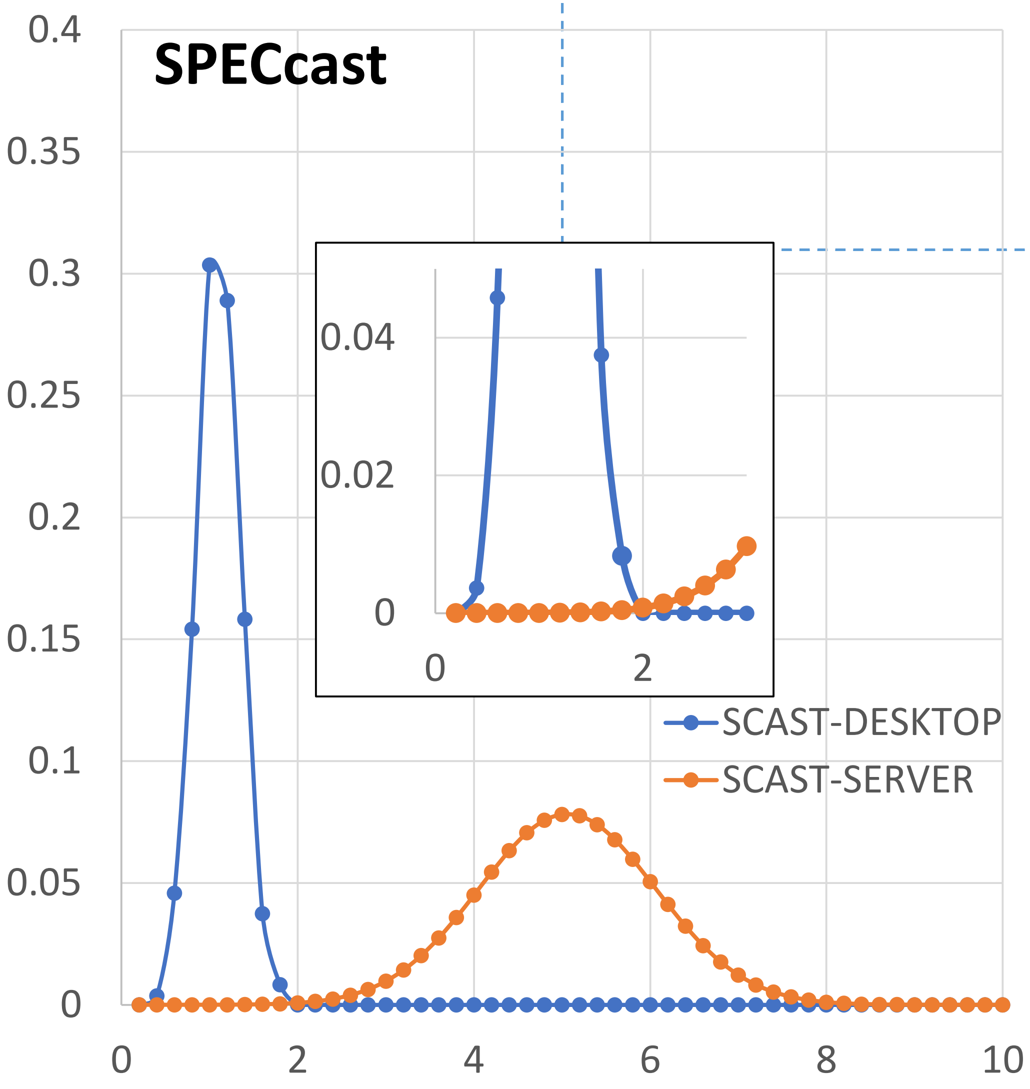
Kolmogorov-Smirnov
P >> 0.05 (close to 1)

3. Use Cases: Prefetch Performance

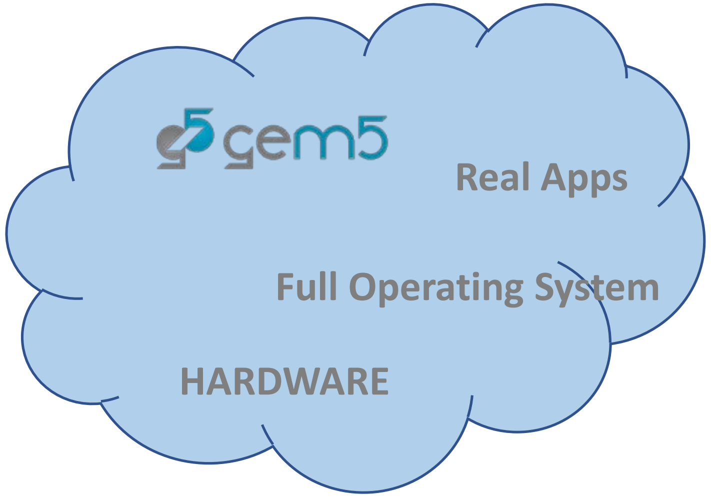


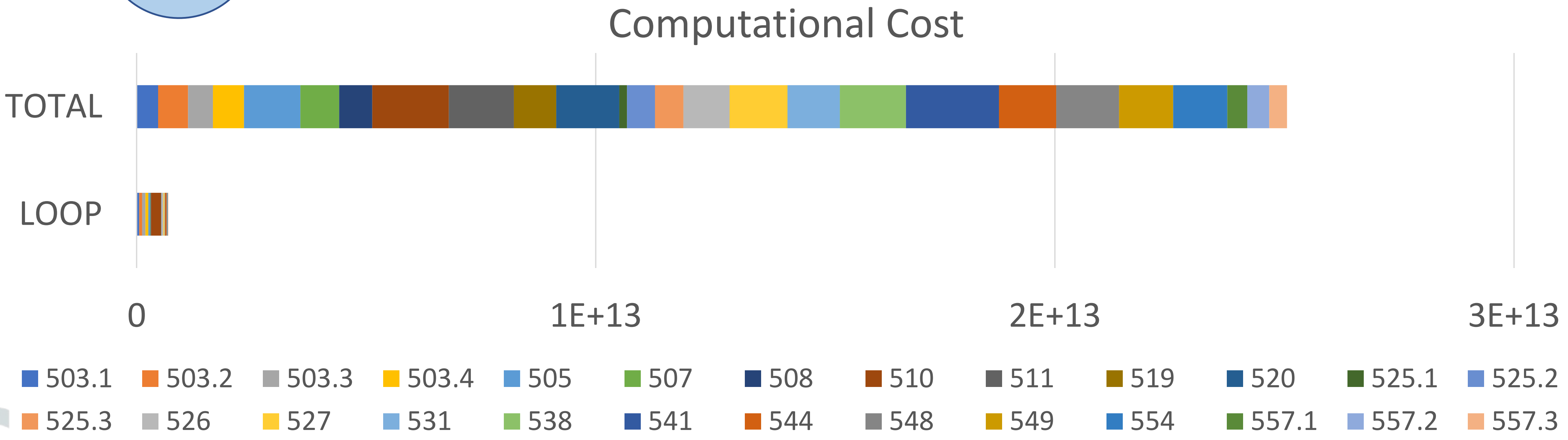
3. Use Cases: System Comparison





3. Use Cases: Simulation





4. Conclusions

- Free Collaborative Tool
  - Available at: https://github.com/prietop/SPECcast

- Work in Progress
  - More applications
  - New metrics
  - Checkpoints

- Any doubts or suggestions please contact:

  prietop@unican.es or abadp@unican.es