POP - providing insight on application and system behavior

Jesus Labarta

EU H2020 Center of Excellence (CoE)

Motivation

Why?

- Complexity and variability in machines and codes →
  - Important to study microscopic behavior to understand macroscopic effects
  - Important to "isolate" programmer from details of machine. Let programmer focus on science

What can POP contribute?

- Detailed Insight and predictive capabilities
- Programming models and practices

- For "co-design" and use
- Hub
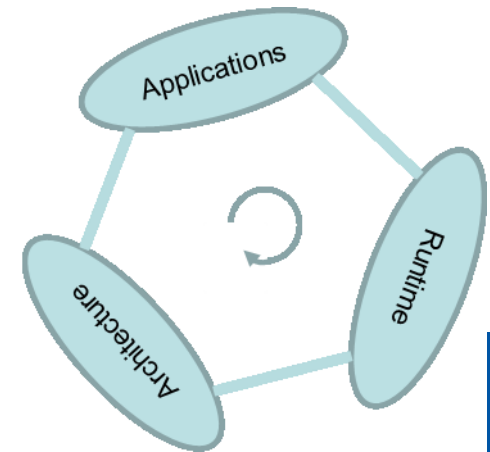
Targeted customers: all actors in EsDs

- **Code developers**
  - Assessment of detailed actual behavior
  - Suggestion of more productive directions to refactor code

- **Users**
  - Assessment of achieved performance on specific production conditions
  - Possible improvements modifying environment setup
  - Evidences to interact with code provider

- **Infrastructure operators**
  - Assessment of achieved performance in production conditions
  - Possible improvements modifying environment setup
  - Information for allocation processes
  - Training of support staff
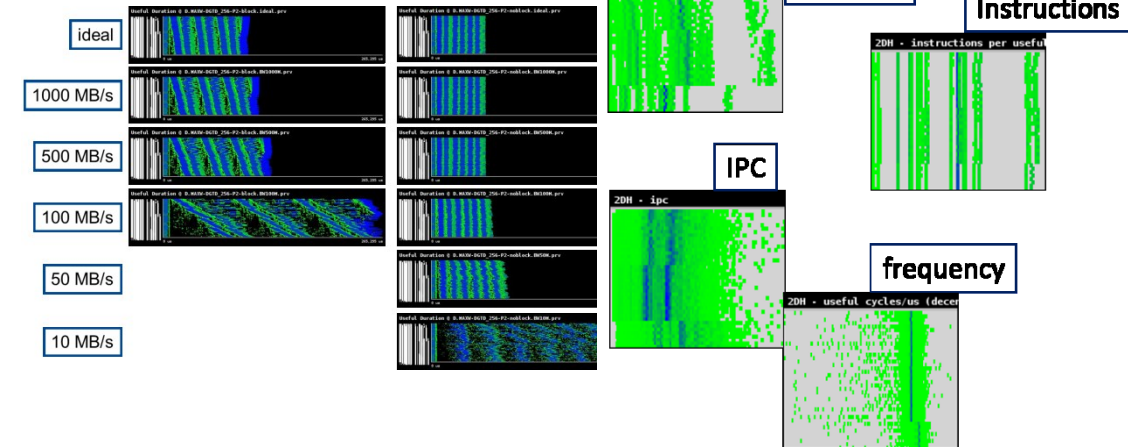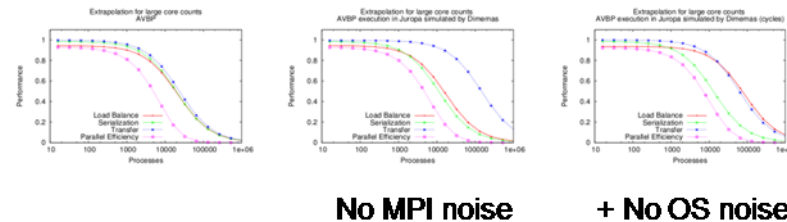
- **Vendors**
  - Benchmarking
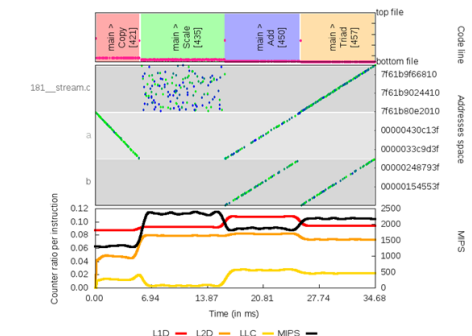 - Customer support
  - System dimensioning/design

Detail and insight !!!

- Understand actual behavior in detail towards co-design and use
- What if
- Projection
- …

- Unifying methodologies
  - Across tools and platforms
- Further tool development and analytics

No MPI noise   + No OS noise

Programming models and runtimes

- Developing programming model
  - Productivity
  - Portability: Homogenizing heterogeneity

- Providing advanced implementations
  - Compiler, scheduling policies, different target platforms (CPU, GPU, FPGA, big.LITTLE, ...), dynamic load balance, ...

- Promoting best practices and a throughput oriented methodology

- Channeling experiences to standardization bodies
  - Active members of MPI Forum and OpenMP ARB

OmpSs Programming model

Tasks	3.0
Dependencies	4.0
Task reductions, priorities	4.1
Concurrent ... resources ... ... label ... multi-dependencies

OpenMP Forerunner

Runtime   Heterogeneity

Core heterogeneity
GPU, Phi and FPGA
Multiple task implementations
Self-tuning, criticality-aware

Locality

Distributed / Cluster