Energy-efficient computing exploiting application dynamicity and Intel telemetry data

Introducing the READEX Project

READEX
Runtime Exploitation of Application Dynamism
for Energy-efficient eXascale computing

European Commission | Horizon 2020
European Union funding
for Research & Innovation

Energy-efficient computing exploiting application dynamicity and Intel telemetry data

Introducing READEX Project

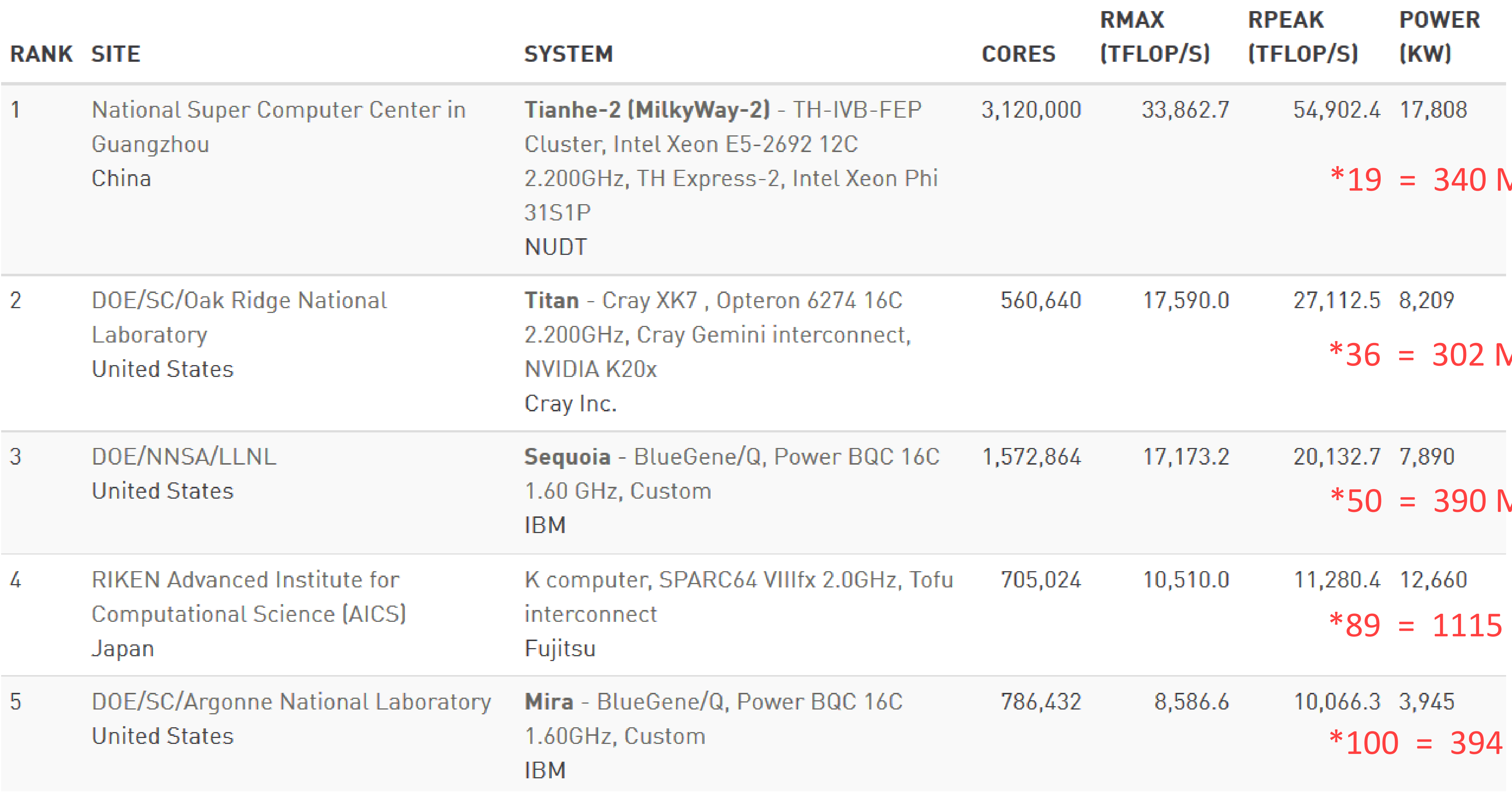
Joseph Schuchart, TU Dresden

READEX
Runtime Exploitation of Application Dynamism
for Energy-efficient eXascale computing

European
Commission

Horizon 2020
European Union funding
for Research & Innovation

TOP 5 Systems: Linear Interpolation for Exascale



RANK	SITE	SYSTEM	CORES	RMAX (TFLOP/S)	RPEAK (TFLOP/S)	POWER (KW)
1	National Super Computer Center in Guangzhou China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3,120,000	33,862.7	54,902.4	17,808 *19 = 340 MW
2	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560,640	17,590.0	27,112.5	8,209 *36 = 302 MW
3	DOE/NNSA/LLNL United States	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM	1,572,864	17,173.2	20,132.7	7,890 *50 = 390 MW
4	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu	705,024	10,510.0	11,280.4	12,660 *89 = 1115 MW
5	DOE/SC/Argonne National Laboratory United States	**Mira** - BlueGene/Q, Power BQC 16C 1.60GHz, Custom IBM	786,432	8,586.6	10,066.3	3,945 *100 = 394 MW

European Commission
Horizon 2020
European Union funding for Research & Innovation

Project overview

READEX

Runtime **E**xploitation of **A**pplication **D**ynamism for **E**nergy-efficient e**X**ascale Computing

Starting date:

1. September 2015

Duration:

3 years

Funding:

European Commission Horizon 2020 grant agreement 671657

Motivation

Challenges

- Energy consumption
- Extreme scale
- Dynamism

Problems

- Awareness
- Ability
- Effort

Solution

- Dynamism
- Automatic tuning
- Design-/Run-time

Periscope Tuning Framework

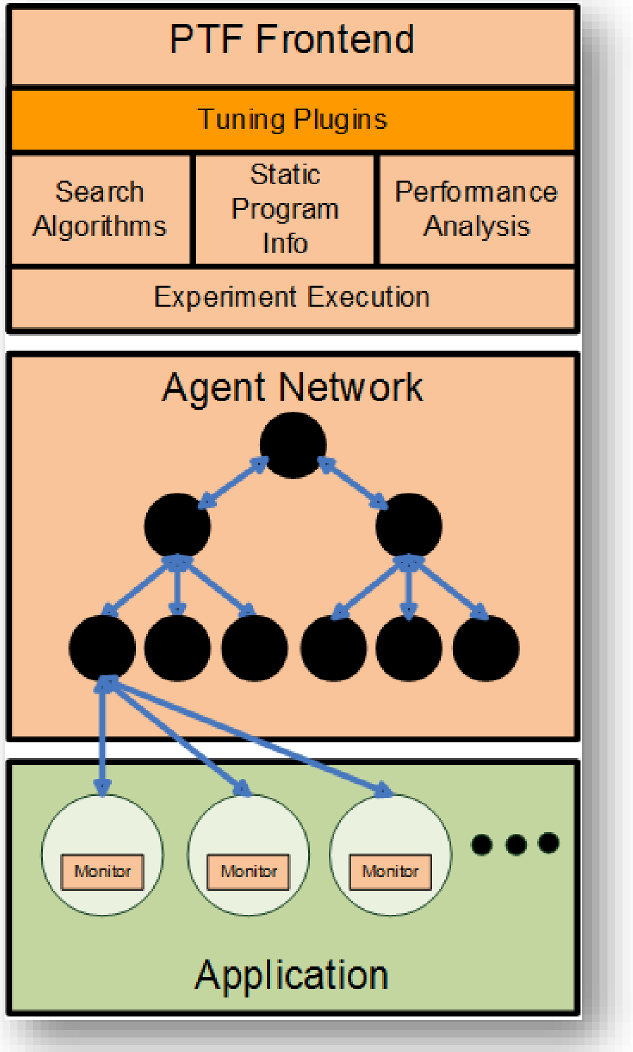
Automatic application analysis & tuning

- Tune performance and energy (statically)
- Plug-in-based architecture
- Evaluate alternatives online
- Scalable and distributed framework

Support variety of parallel paradigms

- MPI, OpenMP, OpenCL, Parallel pattern

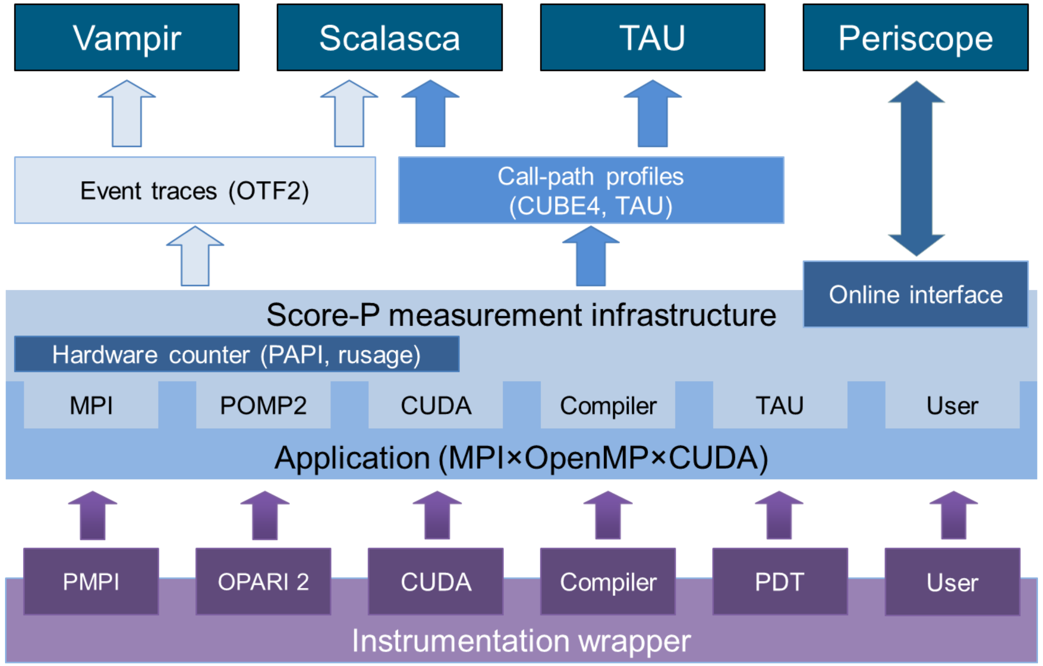
Developed in the AutoTune EU-FP7 project



Score-P

Scalable Performance Measurement Infrastructure for Parallel Codes

Common instrumentation and measurement infrastructure

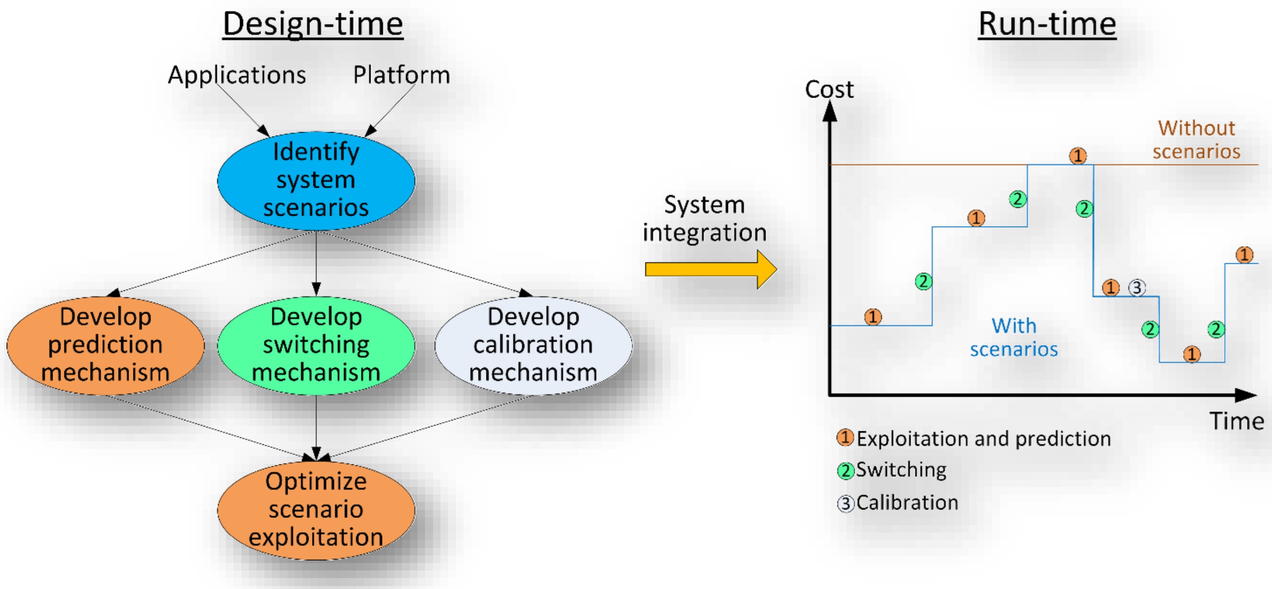


Score-P

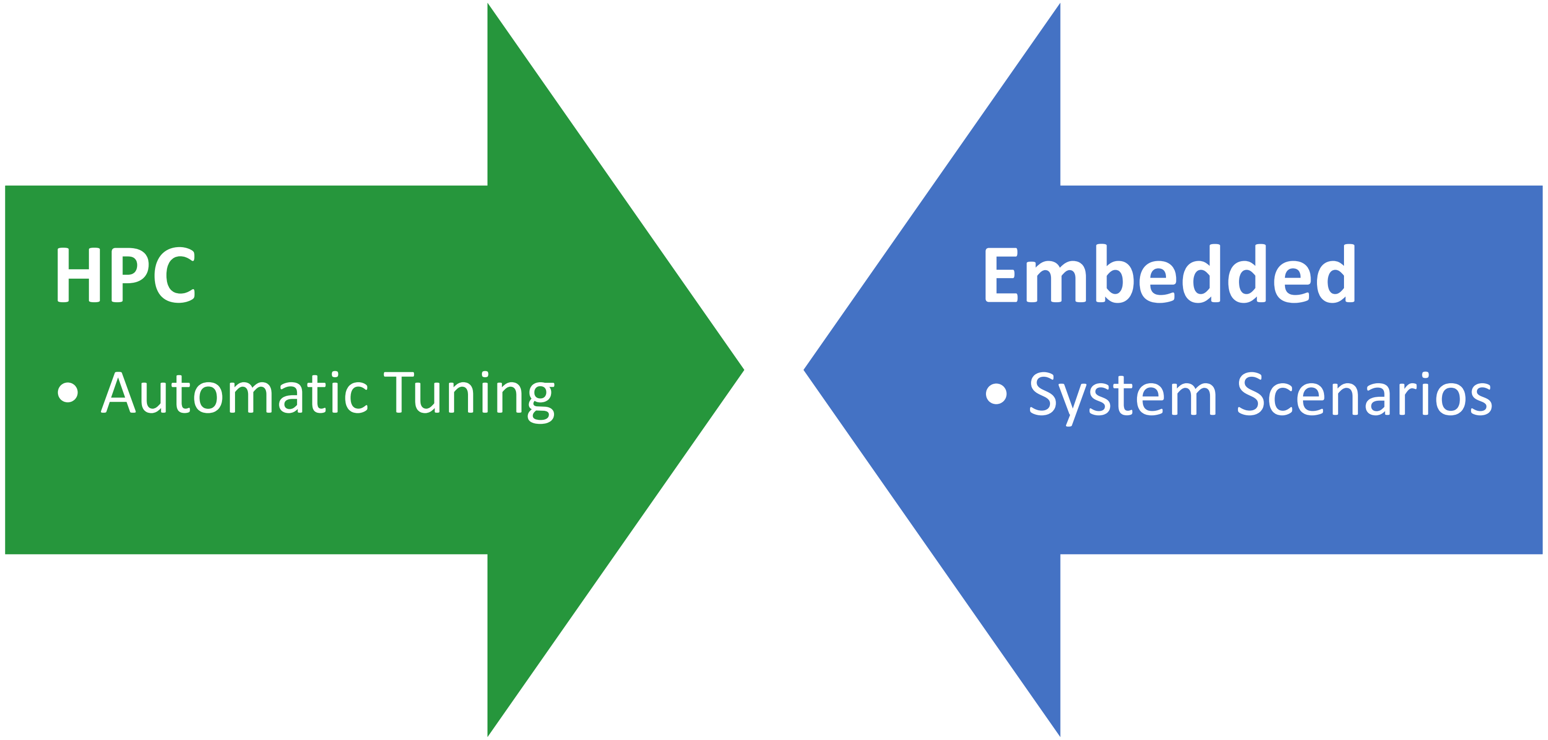
Background

System Scenario based Methodology

- Formalism for dynamic tuning in the embedded systems world
- Detect dynamism in applications at design-time
- Switch parameters at run-time based on detected run-time situations


Design-time
Applications
Platform
Identify system scenarios
Develop prediction mechanism
Develop switching mechanism
Develop calibration mechanism
Optimize scenario exploitation
System integration
Run-time
Cost
Time
Without scenarios
With scenarios
① Exploitation and prediction
② Switching
③ Calibration

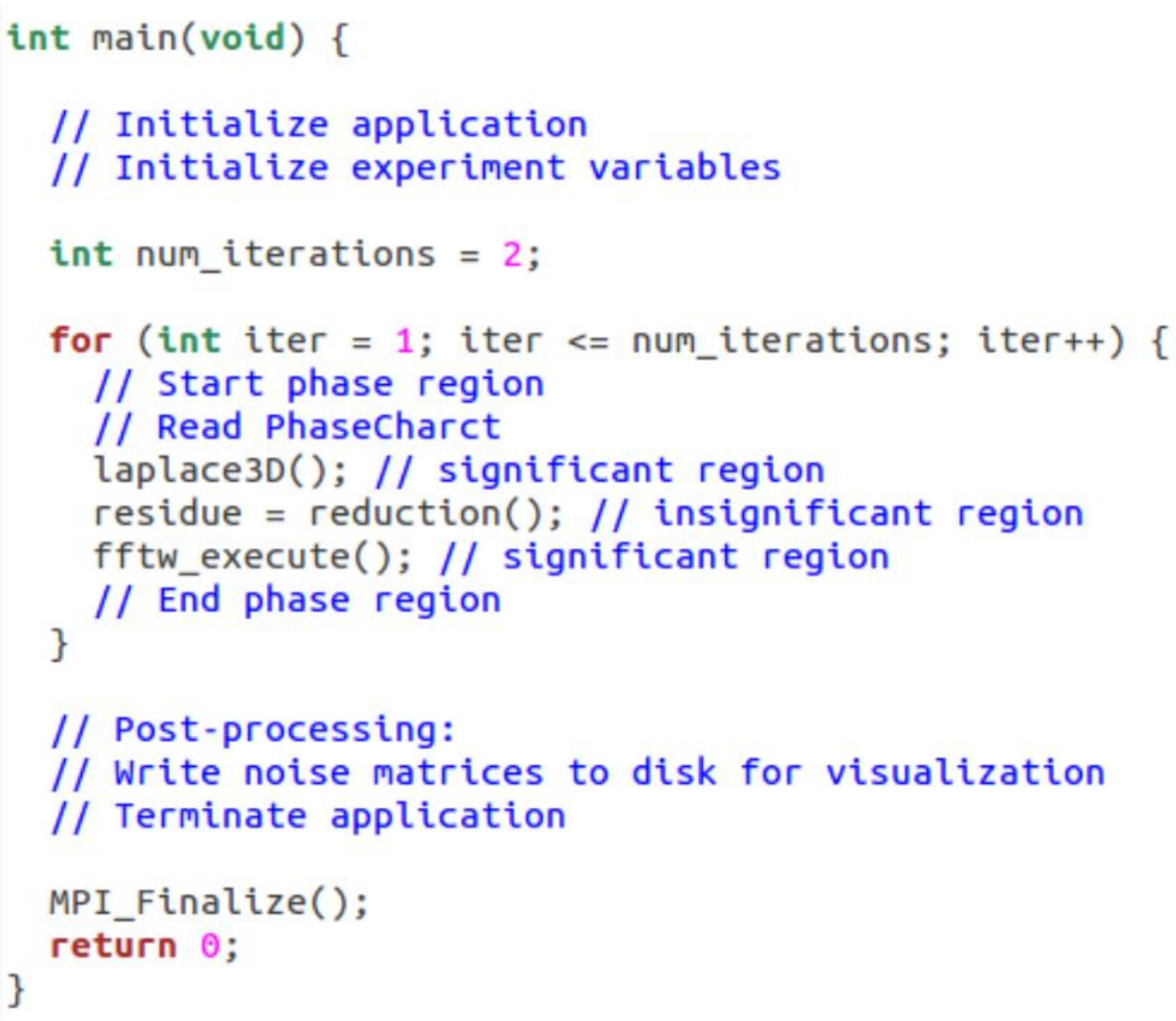
General idea



READEX tools-based methodology for dynamic tuning for energy efficiency

Horizon 2020
European Union funding for Research & Innovation

Application Dynamism: Beyond Static Tuning

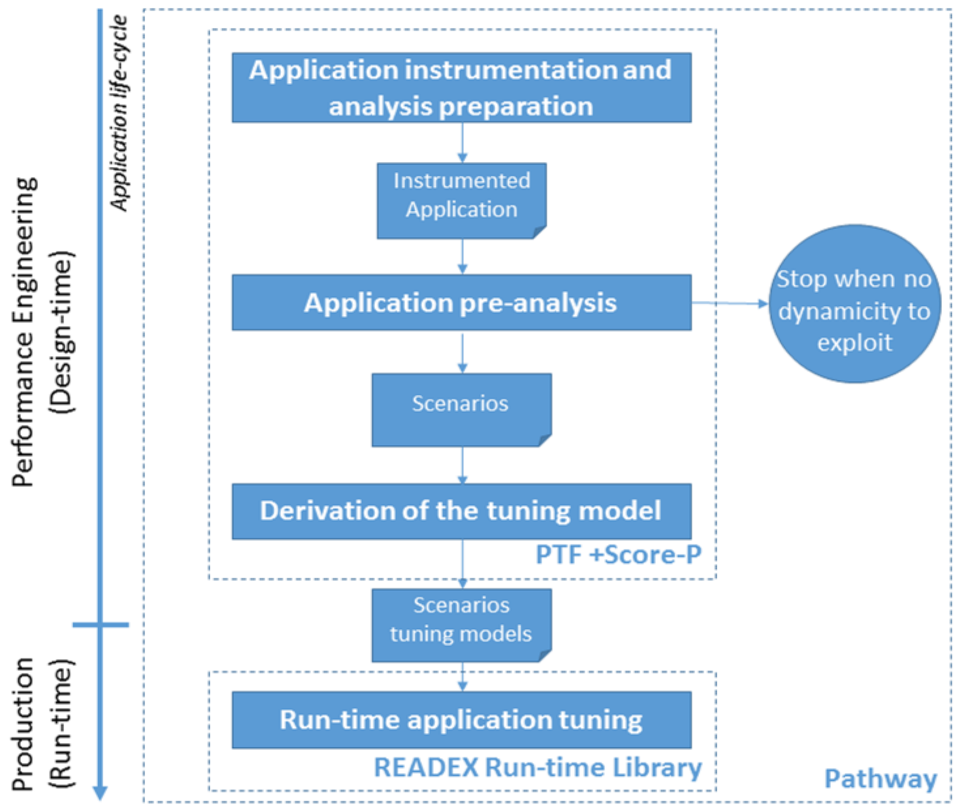


```
int main(void) {

  // Initialize application
  // Initialize experiment variables

  int num_iterations = 2;

  for (int iter = 1; iter <= num_iterations; iter++) {
    // Start phase region
    // Read PhaseCharct
    laplace3D(); // significant region
    residue = reduction(); // insignificant region
    fftw_execute(); // significant region
    // End phase region
  }

  // Post-processing:
  // Write noise matrices to disk for visualization
  // Terminate application

  MPI_Finalize();
  return 0;
}
```

One iteration of the progress loop

Design-time and Run-time

Design-time vs Run-time

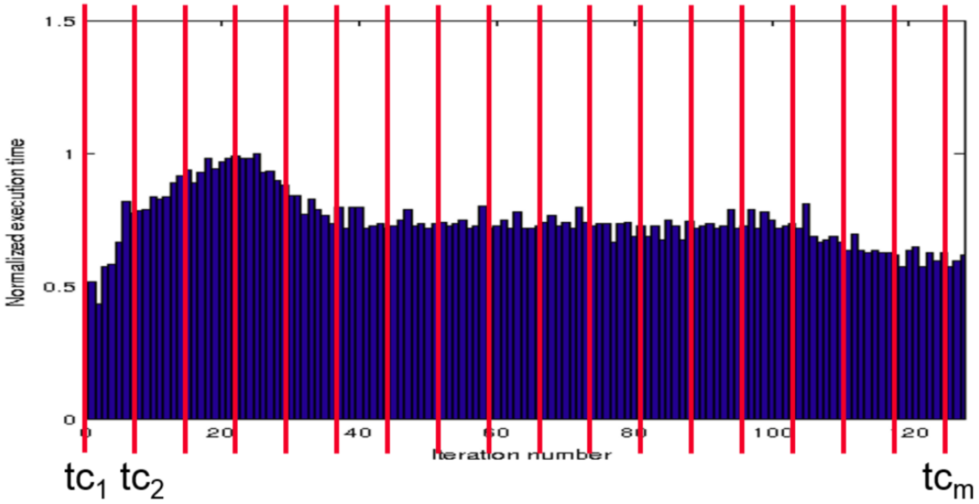


Application life-cycle
Performance Engineering (Design-time)
Production (Run-time)
Application instrumentation and analysis preparation
Instrumented Application
Application pre-analysis
Stop when no dynamicity to exploit
Scenarios
Derivation of the tuning model
PTF +Score-P
Scenarios tuning models
Run-time application tuning
READEX Run-time Library
Pathway

READEX Tools-aided methodology

Design Time Analysis

- Tuning cycles
  - Capture intra-phase dynamism
  - Create phase TM
- Sequence of tuning cycles
  - Capture inter-phase dynamism
  - Create inter-phase TM
- DTA for multiple inputs
  - Capture input dynamism
  - Create application TM

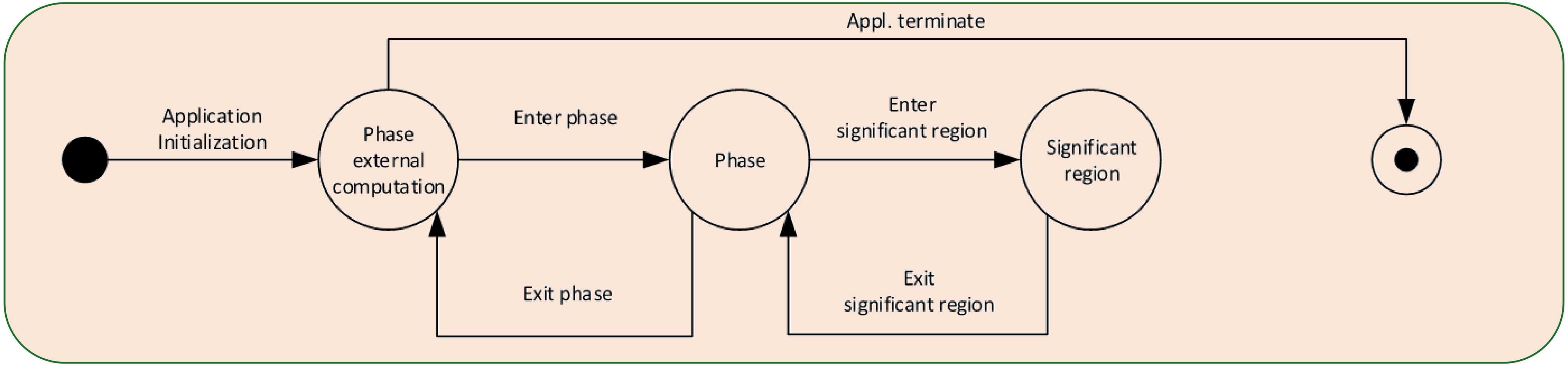


Tuning Model

- Scenarios: set of runtime situations (rts)
- Classifiers: RTS → S
- Selector: Context → CFG

European Commission
Horizon 2020
European Union funding for Research & Innovation

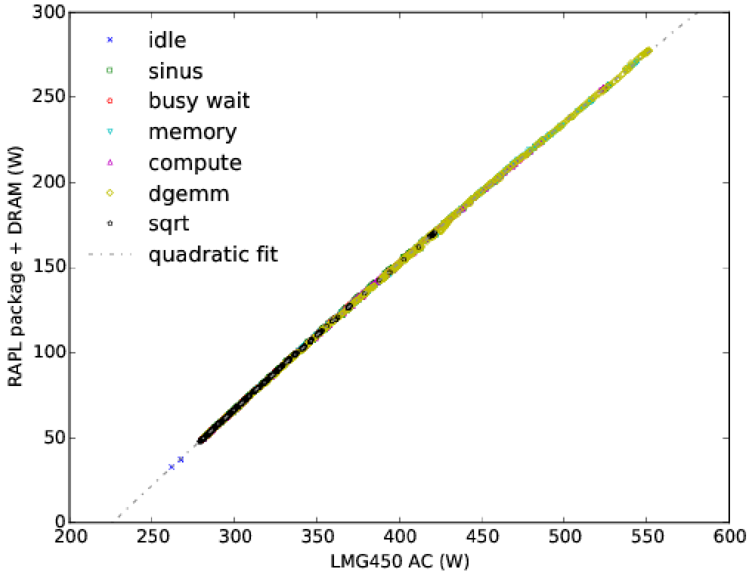
Runtime Tuning with the READEX Runtime Library



Enter phase:	Capture phase identifiers
Enter significant region:	Classify rts; apply selector; perform switching
Exit significant region:	Save objective value
Exit phase:	Perform calibration

Objective Functions

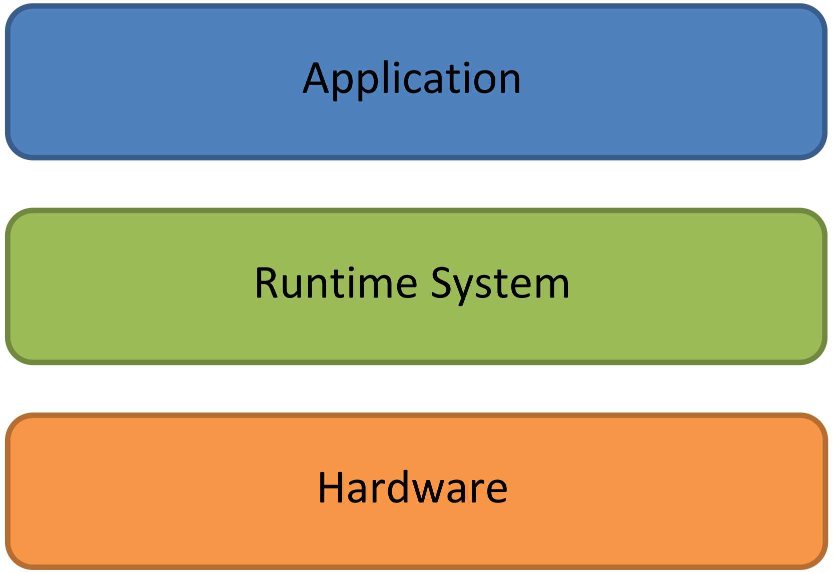
- Energy measurements using
  - The High Definition Energy Efficiency Measurement infrastructure (HDEEM)
  - RAPL on Intel Xeon processors E5 V3 family (access via x86_adapt)
- Use performance counter at design time to determine computational intensity
  - Ratio of compute instructions vs memory accesses
- Adjust tuning parameters to minimize time-to-solution/energy/energy-delay-product



From: Hackenberg, Schöne, Ilsche, Molka, Schuchart, Geyer: *An Energy Efficiency Feature Survey of the Intel Haswell Processor;* The Eleventh Workshop on High-Performance, Power-Aware Computing (HPPAC 2015)

Tuning Parameters

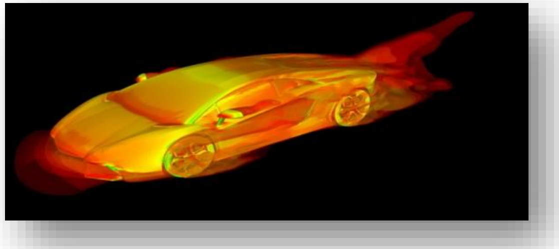
- Hardare tuning parameters:
  - DVFS
  - Uncore frequency (work in progress)
  - T-states (work in progress)
  - Prefetcher settings
- Runtime system parameters
  - Number of OpenMP threads
  - Collaboration with MPC
- Application-level parameters
  - Similar libary calls
  - Dynamic offloading
  - Loop-unrolling factors

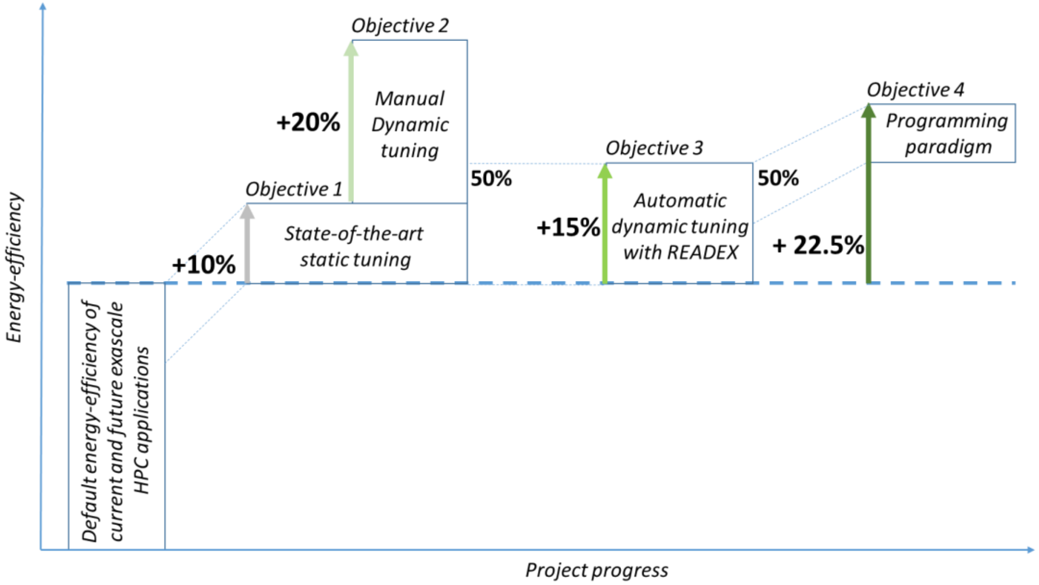


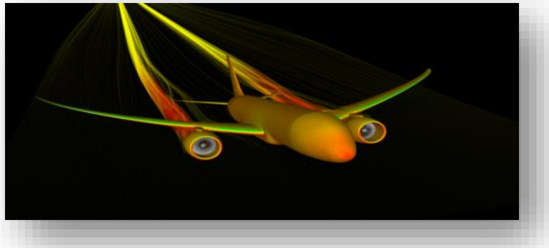
Validation and project goals

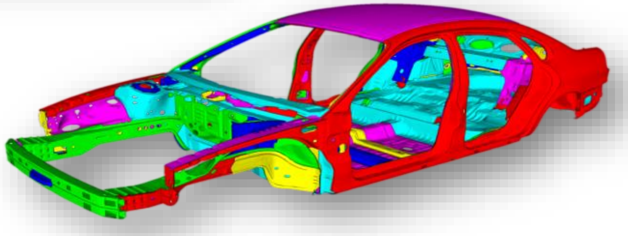
Goal: Validate the effect of READEX using real-world applications

- Co-design process:
  - Hand-tune selected applications
  - Compare results with automatic static and dynamic tuning
- Energy measurements using HDEEM infrastructure









READEX

Runtime Exploitation of Application Dynamism for Energy-efficient eXascale computing

European Commission | Horizon 2020 European Union funding for Research & Innovation

Conclusion

Energy-efficiency at exascale

- Application developers and users will have to care

Lack of capabilities

- Awareness
- Expertise
- Resources

Proposed solution – READEX:

- Exploit dynamism
- Detect at design time, exploit at run-time
- Tools-aided autotuning methodology

READEX – Partners

- Technische Universität Dresden/ZIH (Coordinator)
- Norges Teknisk-Naturvitenskapelige Universitet
- Technische Universität München
- Vysoka Skola Banska - Technicka Univerzita Ostrava
- National University of Ireland/Galway
- Intel Corporation SAS
- Gesellschaft für numerische Simulation mbH

Contact and Funding

- Joseph Schuchart
  joseph.schuchart@tu-dresden.de

- Funded by the European Union‘s Horizon 2020 research and innovation programme under grant agreement No 671657
- Project Runtime: 09/2015 – 08/2018

European Commission | Horizon 2020 European Union funding for Research & Innovation

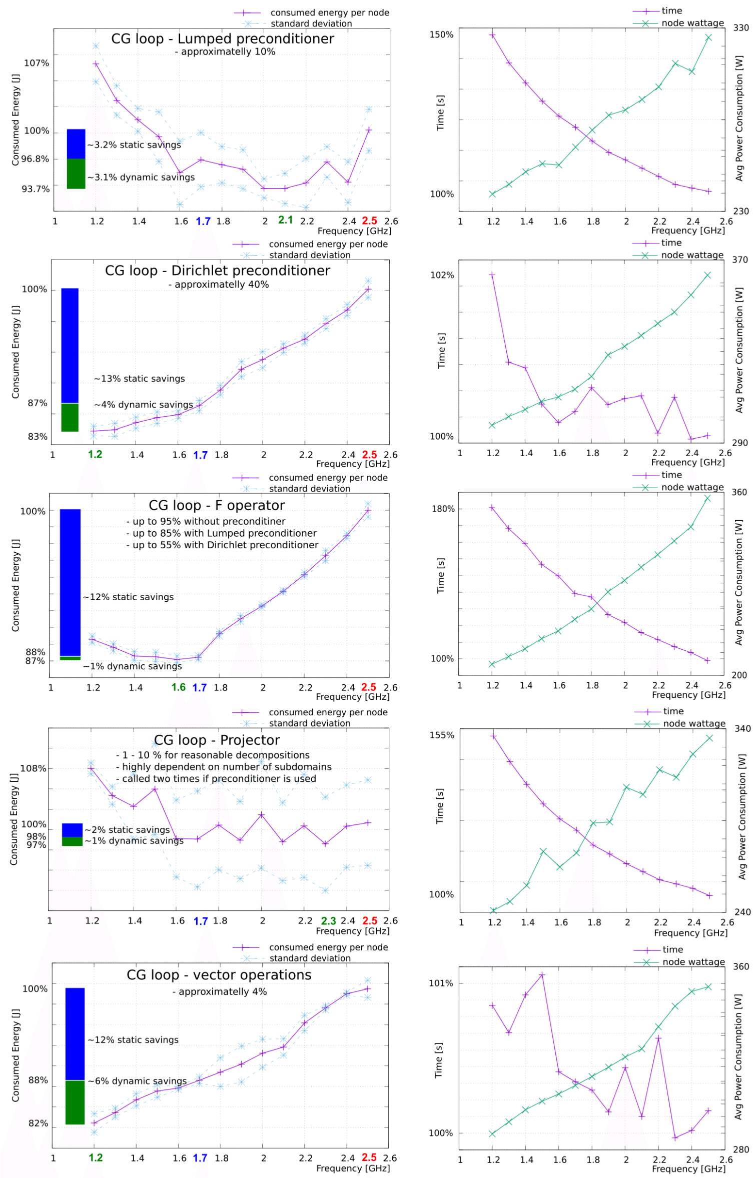
Thank you! Questions?

Runtime Exploitation of Application Dynamism for Energy-efficient eXascale computing

European Commission
Horizon 2020 European Union funding for Research & Innovation

Backup

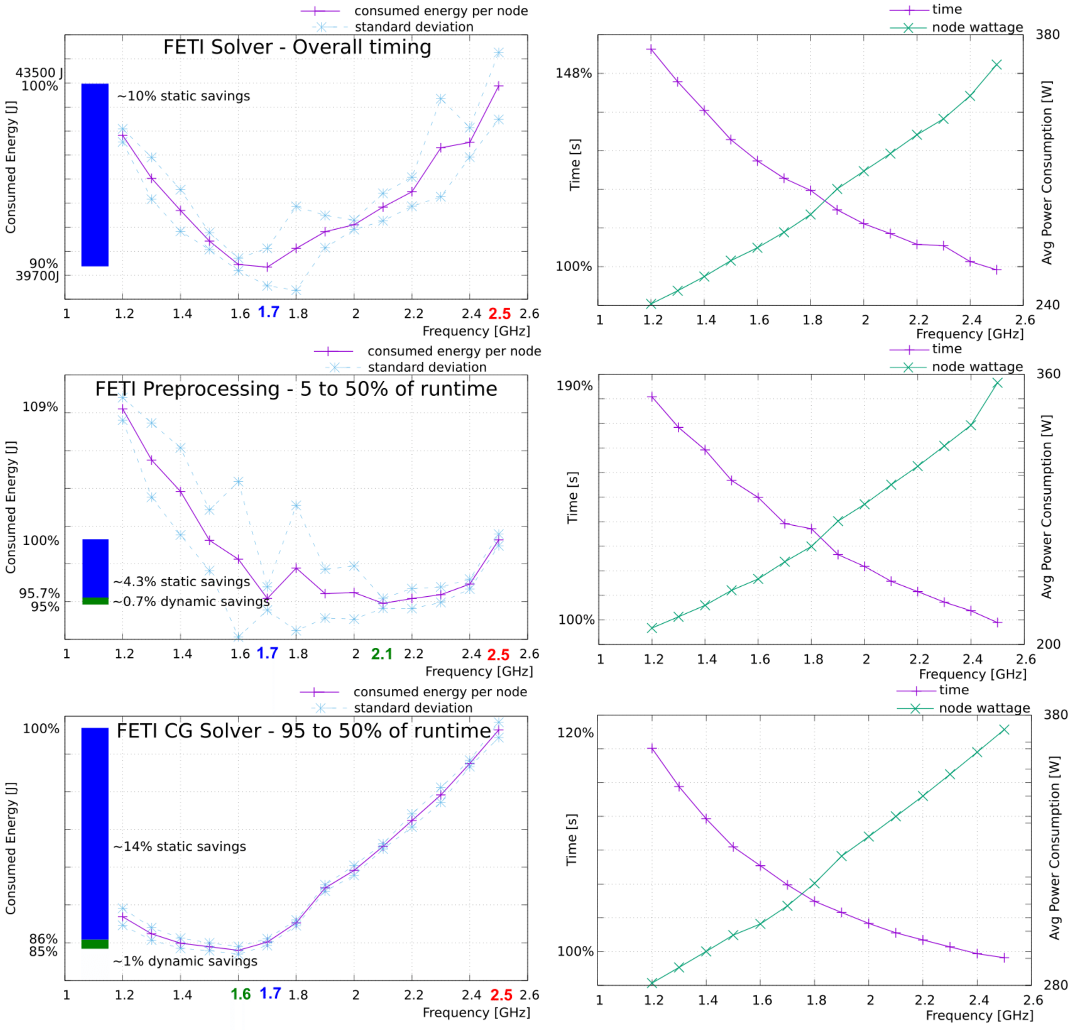
Case Study (I)



READEX
Runtime Exploitation of Application Dynamism for Energy-efficient eXascale computing

European Commission | Horizon 2020 European Union funding for Research & Innovation

Case Study (II)



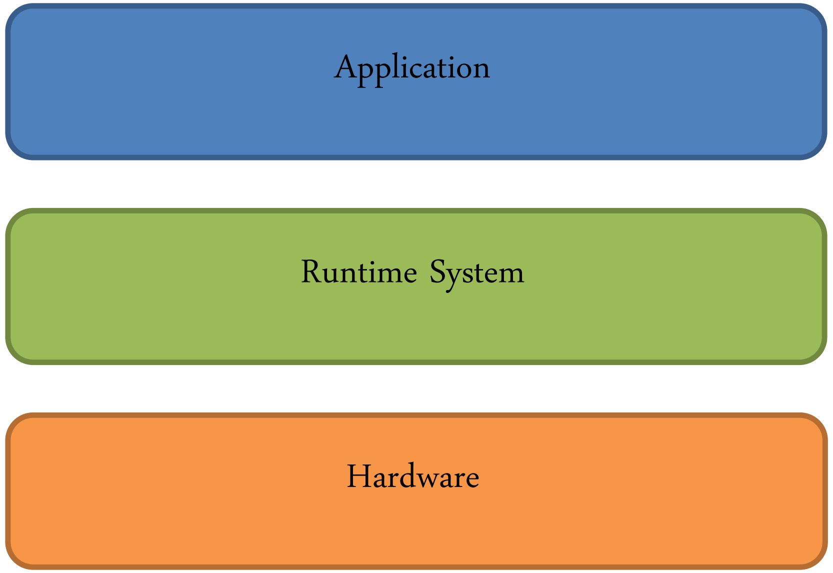
READEX
Runtime Exploitation of Application Dynamism for Energy-efficient eXascale computing

European Commission
Horizon 2020
European Union funding for Research & Innovation

Tuning Parameters

Find set of suitable paramters to modify at run-time

- Optimize for performance
- Optimize for energy-efficiency


Application
Runtime System
Hardware