READEX: A Tool Suite for Dynamic Energy Tuning

Michael Gerndt

Technische Universität München

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

European Commission

Horizon 2020
European Union funding
for Research & Innovation

Campus Garching

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

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

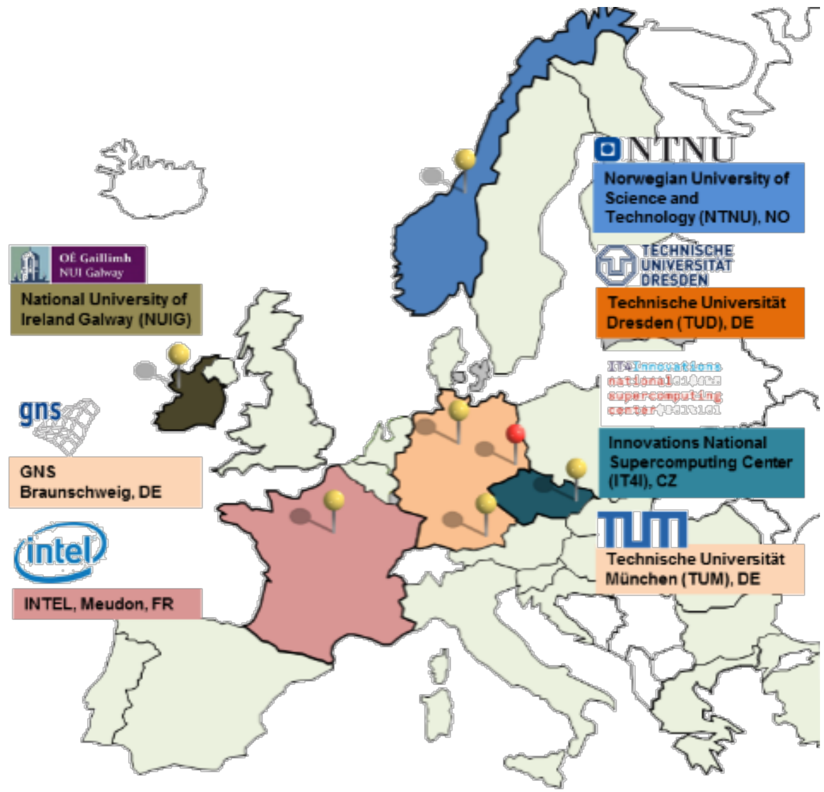
SuperMUC: 3 Petaflops, 3 MW

READEX

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

09/2015 to 08/2018

www.readex.eu



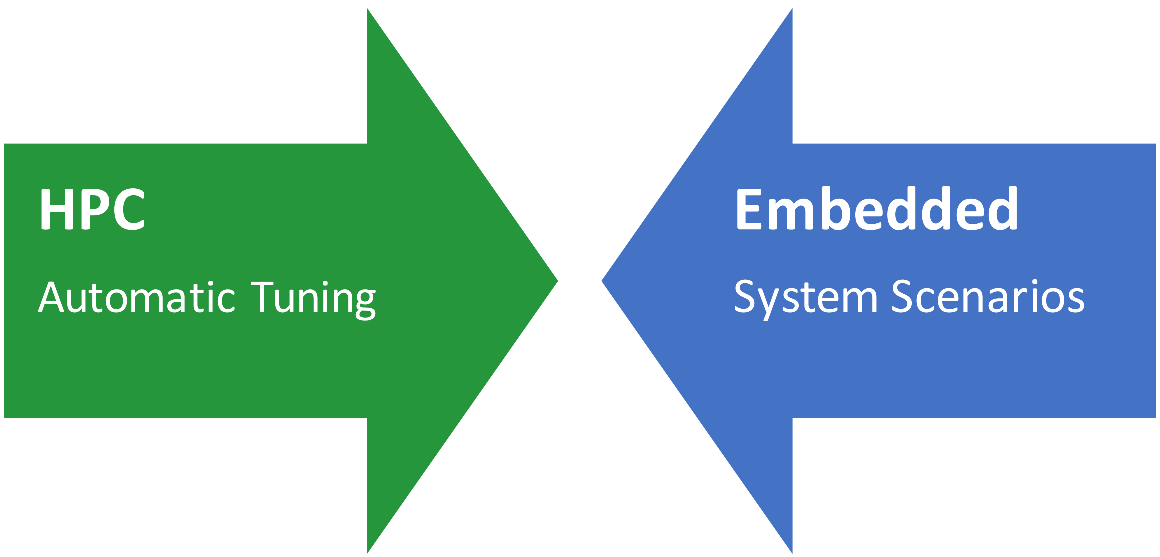
READEX
Runtime Exploitation of Application Dynamism for Energy-efficient eXascale computing

European Commission

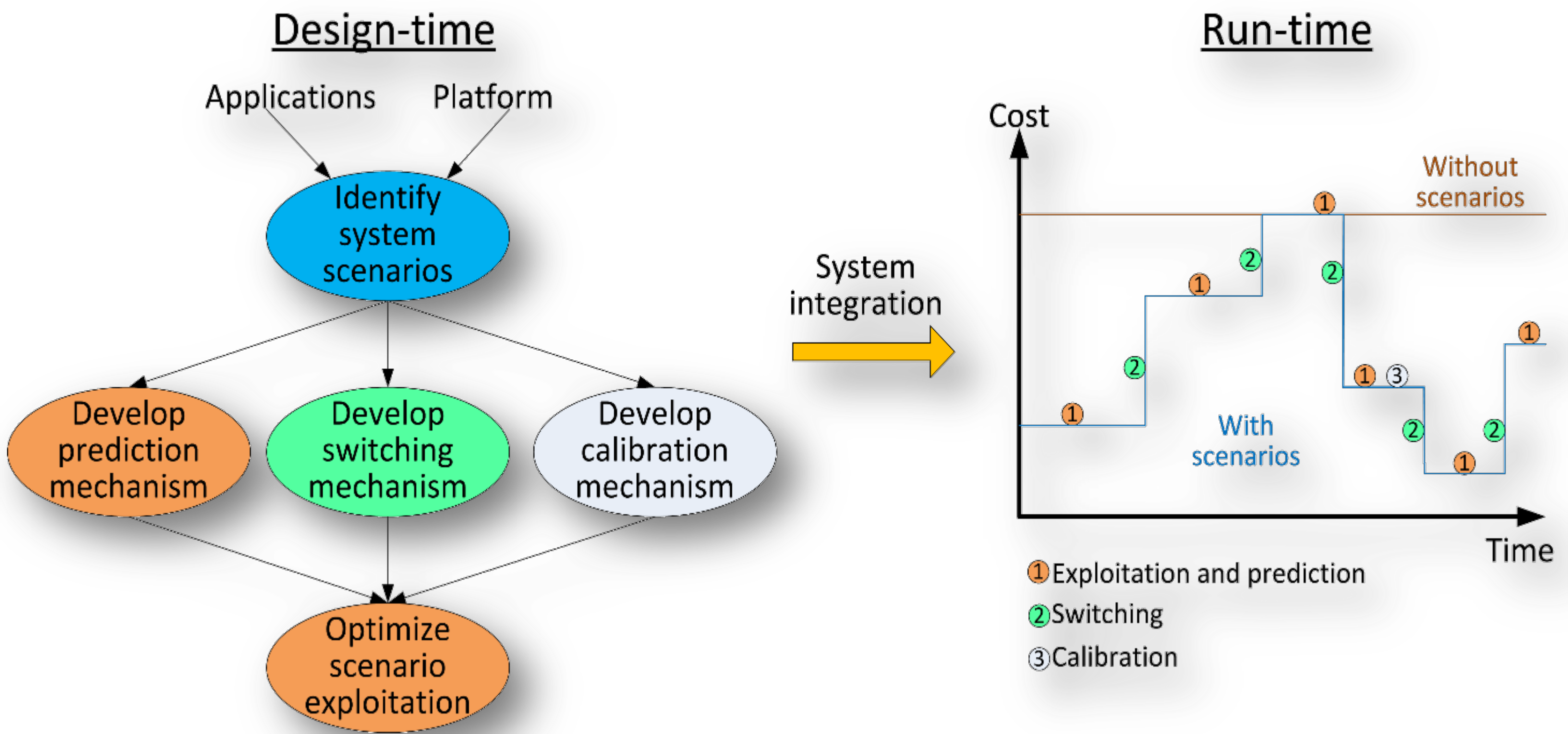
Horizon 2020 European Union funding for Research & Innovation

Objectives

- Tuning for energy efficiency
- Beyond static tuning: exploit dynamism in application characteristics
- Leverage system scenario based tuning

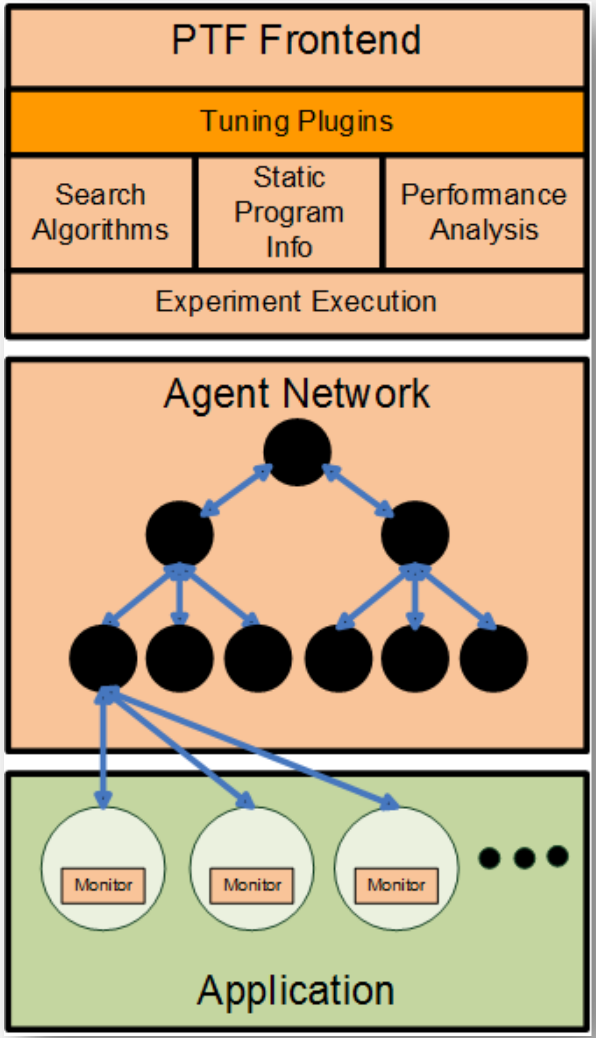


Systems Scenario based Methodology



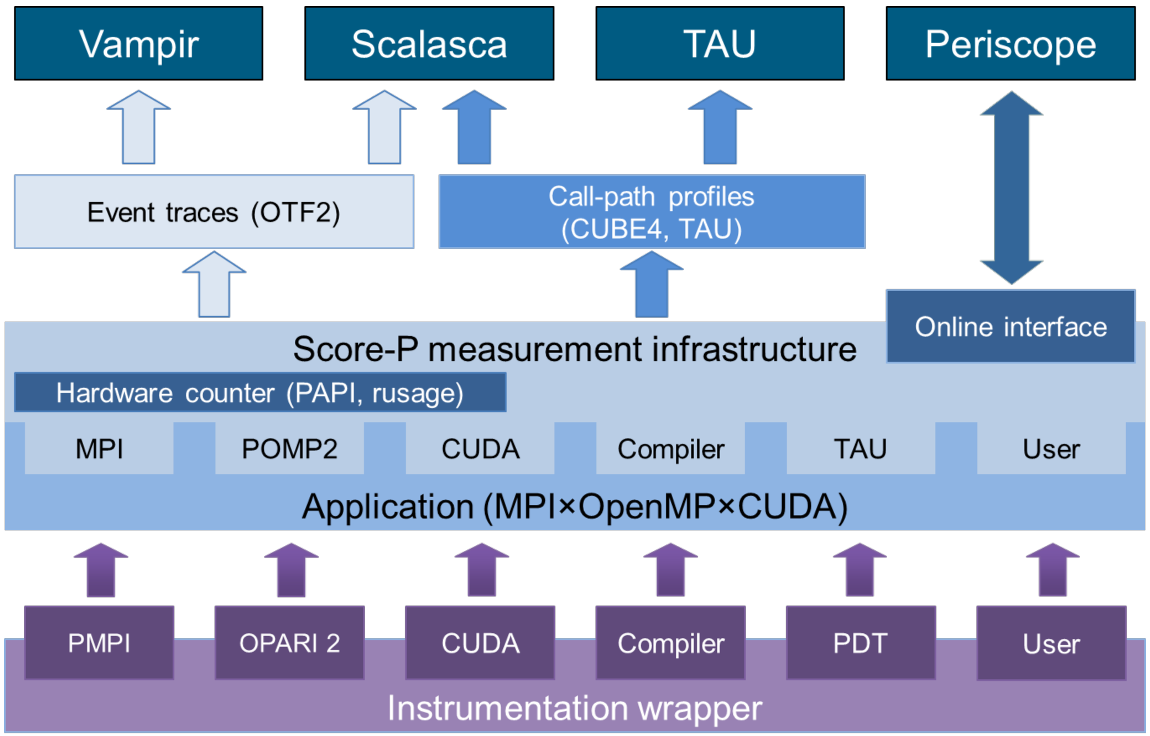
Periscope Tuning Framework (PTF)

- 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
- AutoTune EU-FP7 project

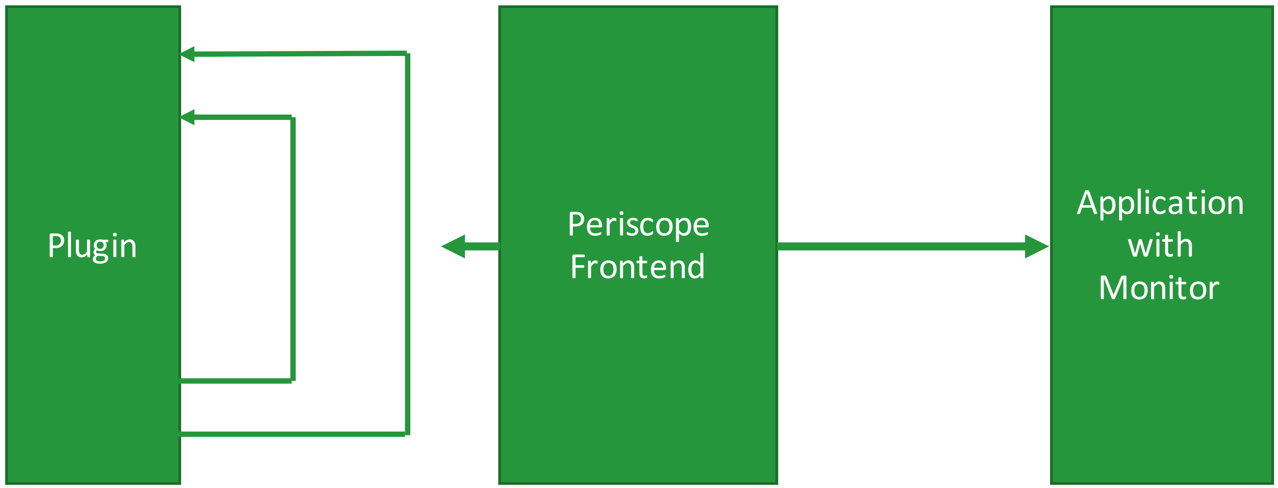


Scalable Performance Measurement Infrastructure for Parallel Codes

Common instrumentation and measurement infrastructure



Tuning Plugin Interface



Search Space Exploration inside of Tuning Steps

Scenario execution

- Tuning actions
- Measurement requests

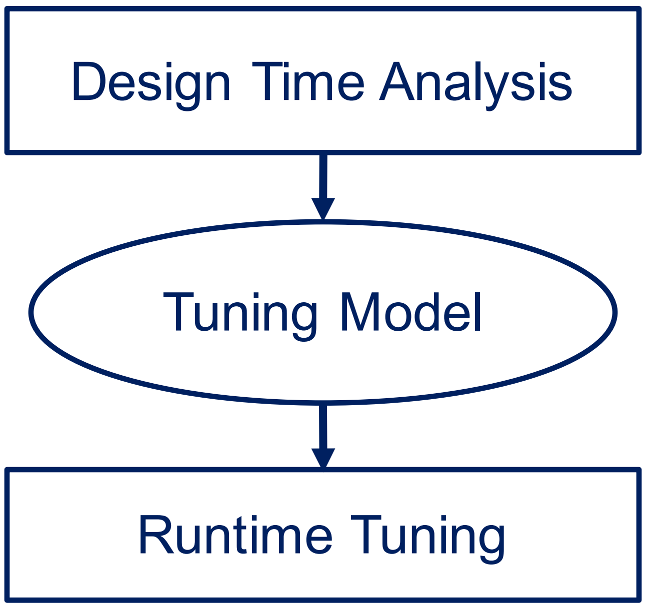
Tuning Plugins

- MPI parameters
  - Eager Limit, Buffer space, collective algorithms
  - Application restart or MPIT Tools Interface
- **DVFS**
  - **Frequency tuning for energy delay product**
  - **Model-based prediction of frequency**
  - **Region level tuning**
- Parallelism capping
  - Thread number tuning for energy delay product
  - Exhaustive and curve fitting based prediction

Dynamic Tuning with the READEX Tool Suite

- READEX extends the concept of tuning in Periscope
- Dynamic tuning
  - Instead of one optimal configuration, SWITCH between different best configurations.
  - Dynamic adaptation to changing program characteristics.

Scenario-Based Tuning



Periscope Tuning Framework (PTF)

READEX Runtime Library (RRL)

Intra-phase Dynamism

```
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 regi
        residue = reduction(); // insignificant reg
        fftw_execute();           // significant regi
        // End phase region
    }

    //  -processing:
          ise matrices to disk for visualizatio
          e application

    MPI_Finalize();
    return 0;
}
```

Phase region

Phase

Intra-phase dynamism

FREQ=2 GHz

FREQ=1.5 GHz

Significant region

Runtime situation

READEX Intra-phase Tuning Plugin

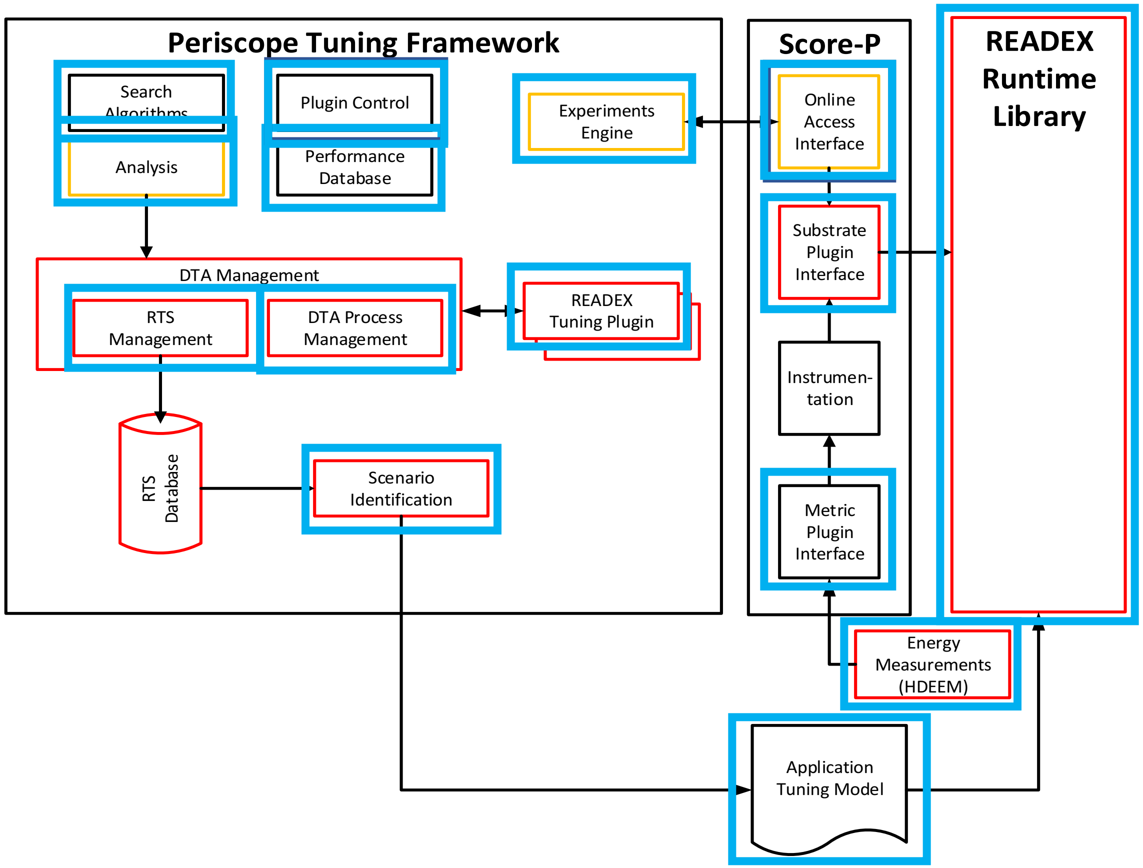
Tuning plugin supporting

- Core and uncore frequencies, numthreads parameters, application tuning parameters
- Configurable search space via READEX Configuration File
- Several objective functions: energy, CPUenergy, EDP, EDP2, time
- Several search strategies: exhaustive, individual, random, genetic

Approach

1. Experiment with default configuration
2. Experiments for selected configurations
    - Configuration set for phase region
    - Energy and time measured for all runtime situations
3. Identification of static best for phase and rts specific best configurations

Pre-Computation of Tuning Model



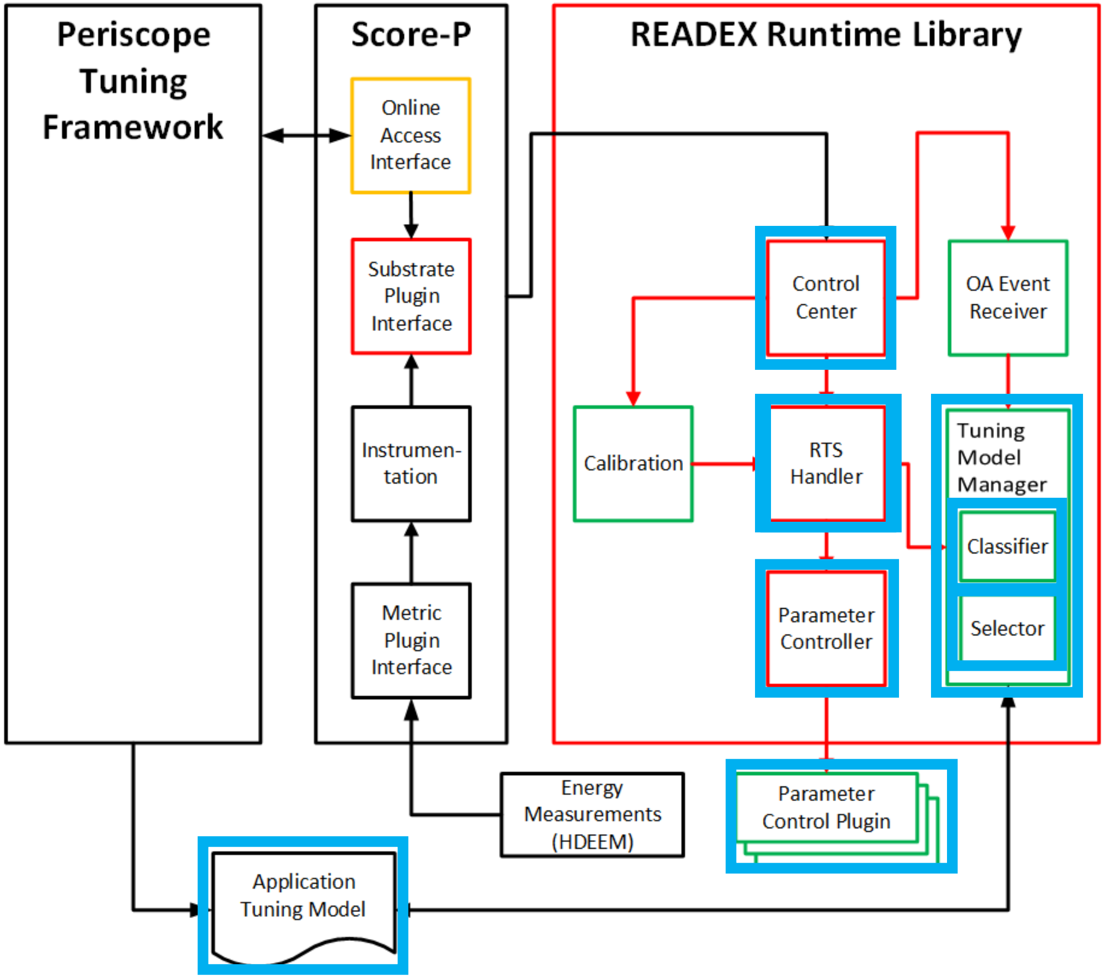
READEX **R**untime **L**ibrary (RRL)

- Runtime Application Tuning performed by the READEX Runtime Library.
- Tuning requests during Design Time Analysis are sent to RRL.
- A lightweight library
  - Dynamic switching between different configurations at runtime.
  - Implemented as a substrate plugin of Score-P.
- Developed by TUD and NTNU

Runtime Scenario Detection and Switching Decision during Production Run

- During Runtime Application Tuning
- Scenario classification

- Switching decision component
- Manipulation of tuning parameters



BEM4I – Dynamic switching – Energy

http://bem4i.it4i.cz/

blade summary, energy	assemble_k [J]	assemble_v [J]	gmres_solve [J]	print_vtu [J]	main [J]
default settings	1467	1484	2733	1142	**6872**
static tuning only	1876	1926	1306	402	**5537**
dynamic tuning only	1348	1335	1150	268	**4138** ✓
static + dynamic tuning	1343	1322	1161	265	**4125**
static savings [%]	-27.9%	-29.8%	52.2%	64.8%	**+19.4%**
dynamic savings [%]	8.4%	10.9%	57.5%	76.8%	**+40.0%**
static + dynamic savings [%]	8.1%	10.0%	57.9%	76.5%	**+39.8%**

```
"static": {
    "FREQUENCY": "25",              <--------- 2.5 GHz
    "NUM_THREADS": "12",            <--------- 12 OpenMP threads
    "UNCORE_FREQUENCY": "22"        <--------- 2.2 GHz
},
```

```
"assemble_k": {
  "FREQUENCY": "23",
  "NUM_THREADS": "24",
  "UNCORE_FREQUENCY": "16"
},

"assemble_v": {
  "FREQUENCY": "25",
  "NUM_THREADS": "24",
  "UNCORE_FREQUENCY": "14"
},

"gmres_solve": {
  "FREQUENCY": "17",
  "NUM_THREADS": "8",
  "UNCORE_FREQUENCY": "22"
},

"print_vtu": {
  "FREQUENCY": "25",
  "NUM_THREADS": "6",
  "UNCORE_FREQUENCY": "24"
}
```

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

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

Scalability Tests – OpenFOAM – Analysis

simpleFoam

- strong scaling test
- Motorbike example

- optimum detected for every run
- Static: 11.7%
- Dynamic: 4.4%
- Total: 15.5%

- Dynamic savings increases with higher number of nodes

	Default energy	Default time	Best static configuration	Static savings	Dynamic savings	Overall savings
1 node	37864.34 J	113.1 s	2.2 GHz UCF, 1.6 GHz CF	7344.46 J (19.40%)	105.85 J of 30519.88 J (0.35%)	7450.31 J (19.68%)
2 nodes	37229.74 J	57.99 s	2.2 GHz UCF, 1.6 GHz CF	6175.44 J (16.59%)	118.24 J of 31054.3 J (0.38%)	6293.68 J (16.9%)
4 nodes	38158.96 J	30.04 s	2.0 GHz UCF, 1.8 GHz CF	6223.96 J (16.31%)	107.28 J of 31935.0 J (0.34%)	6331.24 J (16.59%)
8 nodes	41179.44 J	19.48 s	2.0 GHz UCF, 1.8 GHz CF	4493.2 J (10.91%)	945.12 J of 36686.24 J (2.58%)	5438.32 J (13.21%)
16 nodes	57980.96 J	14.86 s	2.2 GHz UCF, 1.8 GHz CF	6780.96 J (11.70%)	2224.96 J of 51200.0 J (4.35%)	9005.92 J (15.53%)
32 nodes	90374.4 J	16.498 s	2.2 GHz UCF, 1.4 GHz CF	22944.0 J (25.39%)	7113.6 J of 67430.4 J (7.87%)	30057.6 J (33.26%)

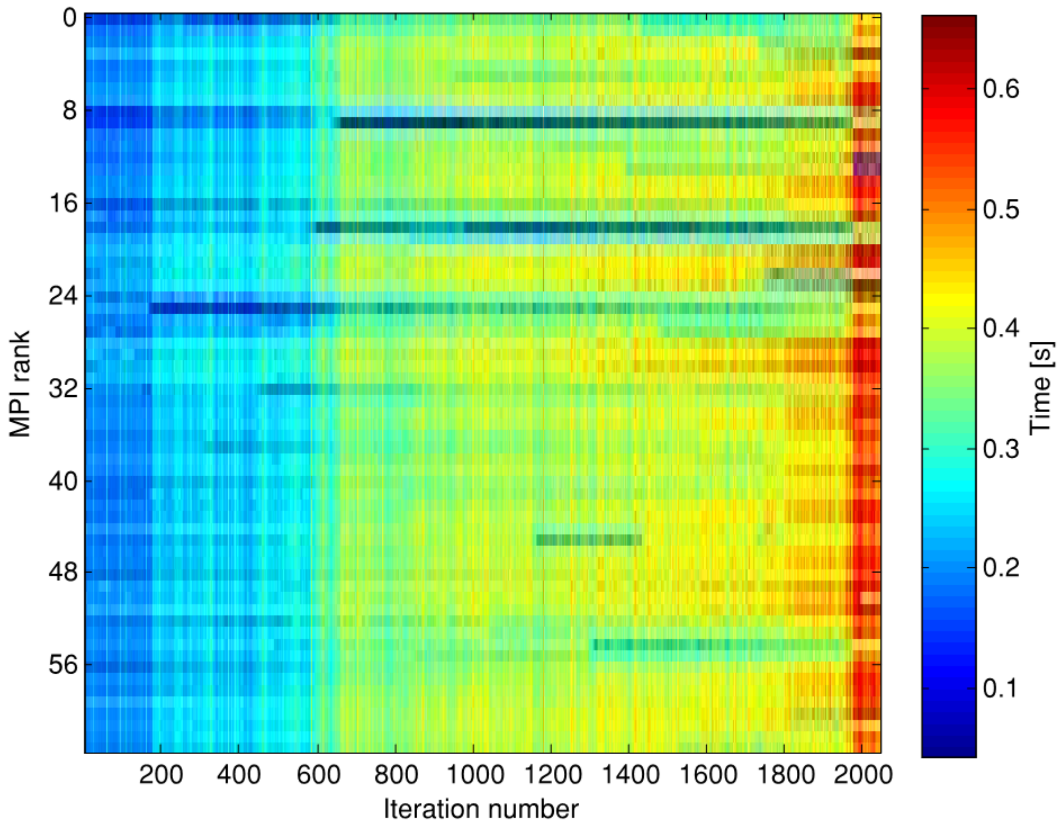
Does not scale anymore

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

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

Inter-phase Dynamism

All-to-all
Performance
2048 phases



PEPC Benchmark of the DEISA Benchmark Suite

Inter-Phase Analysis

- Variation of behavior among phases
- Group/cluster phases
- Select a best configuration for each cluster of phases

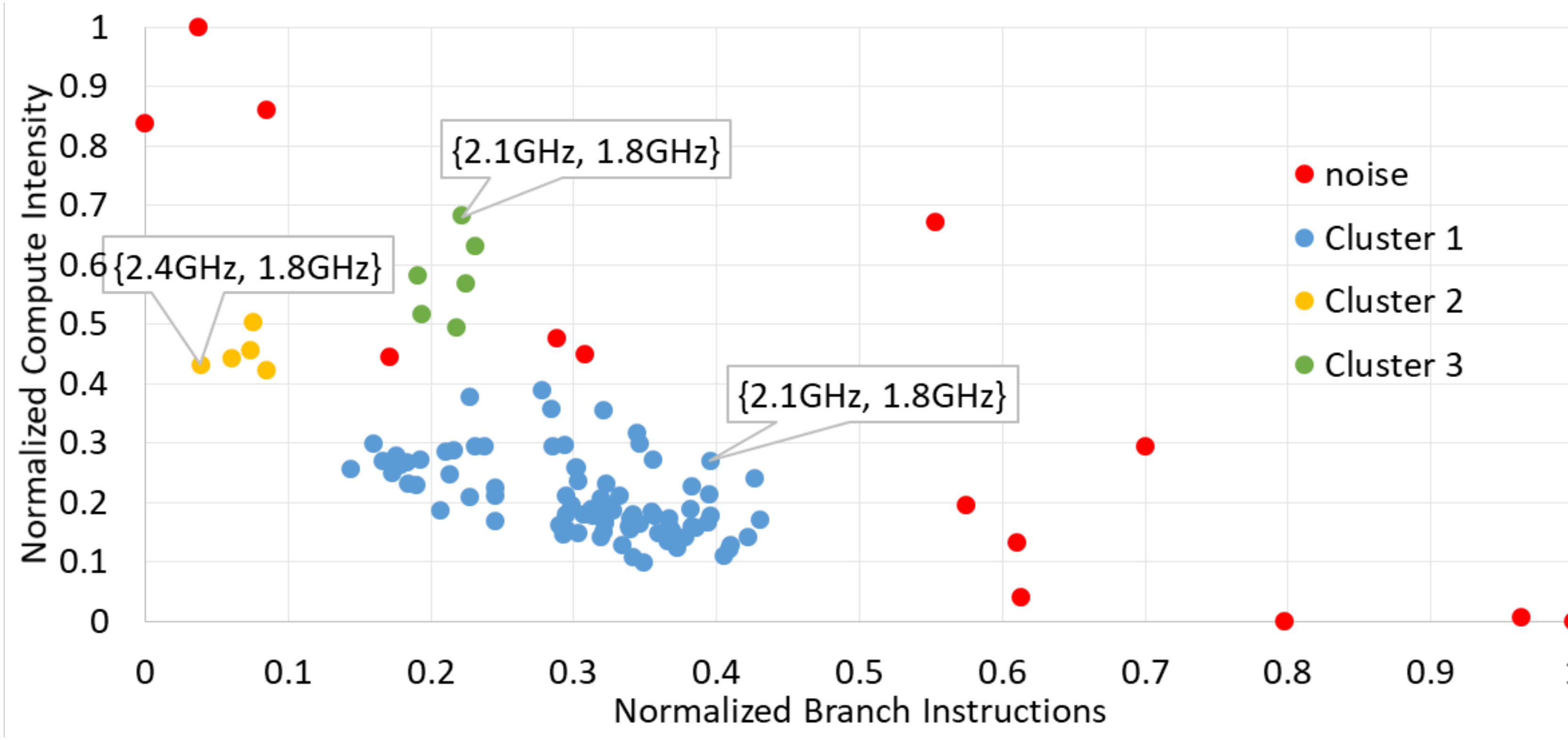
What do we need?

- Identifiers of phase characteristics (Phase Identifiers)
- Provided by application expert (??)

Inter-Phase Analysis – Approach

- Developed the *interphase_tuning* plugin
- 3 tuning steps:
  - **Analysis step:**
    - Random search strategy is used to create the search space
    - Don’t want to explore the whole tuning space
    - Cluster phases and find best configuration for each cluster
  - **Default step:**
    - Run the application for the default setting
  - **Verification step:**
    - Select the best configuration for each phase, as determined for its cluster.
    - Aggregate the savings over the phases

INDEED



- 3 clusters identified
- Noise points marked in red

Cluster Prediction in RRL

- How to handle phase identifiers to predict clusters?
  - Call path of an rts now includes the cluster number
- Solution:
  - Add the cluster number as a user parameter
  - Add PAPI events to measure L3_TCM, Total_Instr and conditional branch instructions

```
…
SCOREP_OA_PHASE_BEGIN()
    SCOREP_USER_PARAMETER_INT64(cluster, predict_cluster())
    …
SCOREP_OA_PHASE_END()
```

  - Predict the cluster of the upcoming phase
  - If the cluster was mispredicted for the phase, correct it at the end of the phase

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

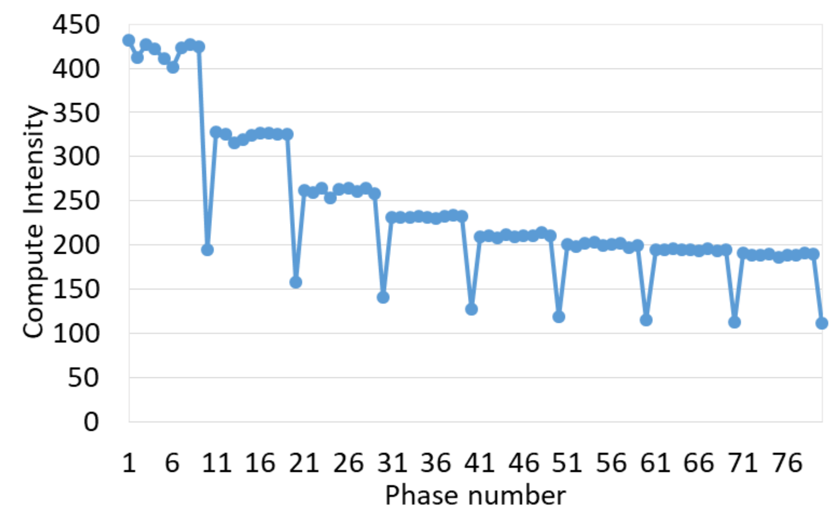
European Commission | Horizon 2020 European Union funding for Research & Innovation

Evaluation of the readex_interphase plugin

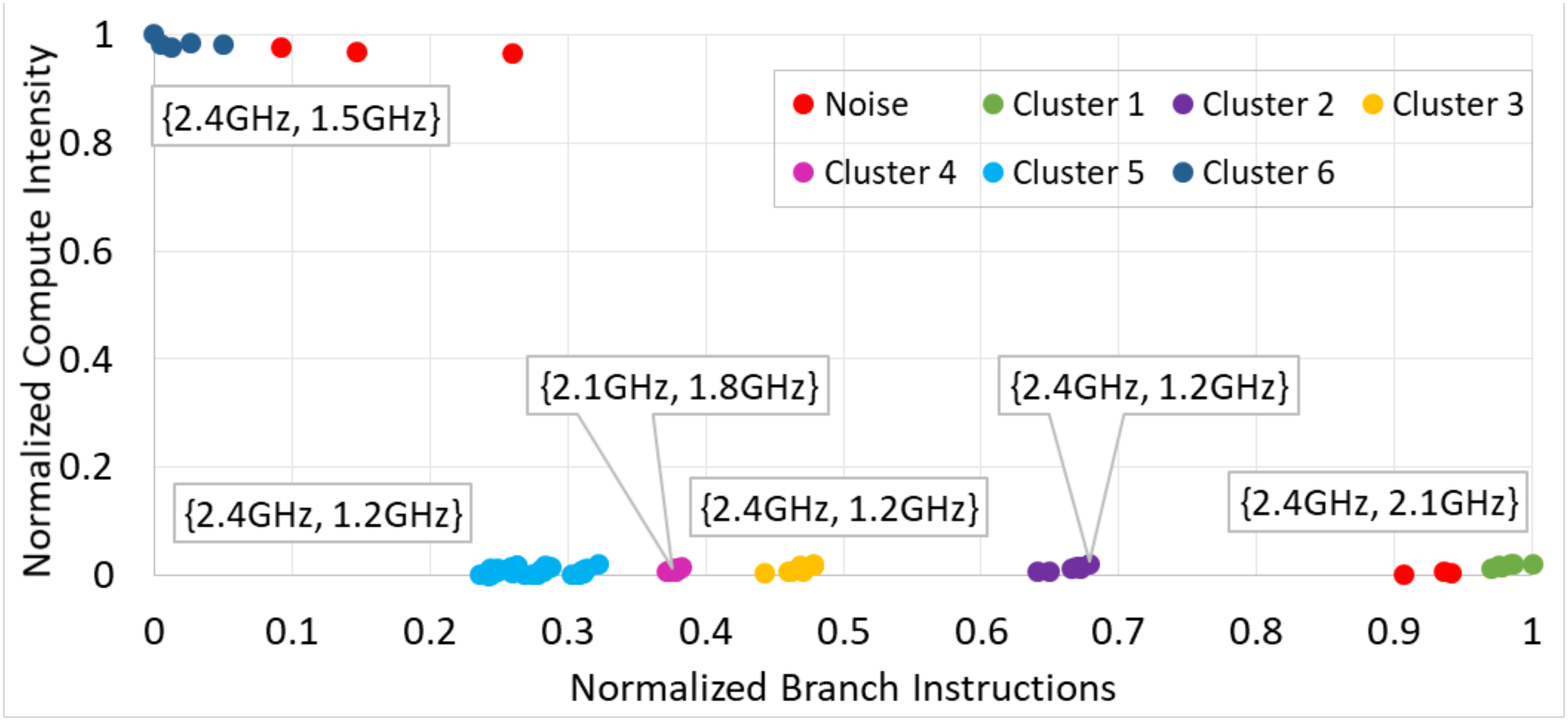
- Performed on two applications: miniMD, INDEED
- Experiments conducted on the Taurus HPC system at the ZIH in Dresden
- Each node contains two 12-core Intel Xeon CPUs E5-2680 v3 (Intel Haswell family)
- Runs with a default CPU frequency of 2.5 GHz, uncore frequency of 3 GHz
- Energy measurements provided on Taurus via HDEEM measurement hardware
- Provides processor and blade energy measurements

miniMD

- Lightweight, parallel molecular dynamics simulation code
- Performs molecular dynamics simulation of a Lennard-Jones Embedded Atom Model (EAM) system
- Written in C++
- Provides input file to specify problem size, temperature, timesteps
- Evaluation of DTA:
  - Hybrid (MPI+OpenMP) AVX vectorized version
  - Problem size of 50 for the Lennard-Jones system.



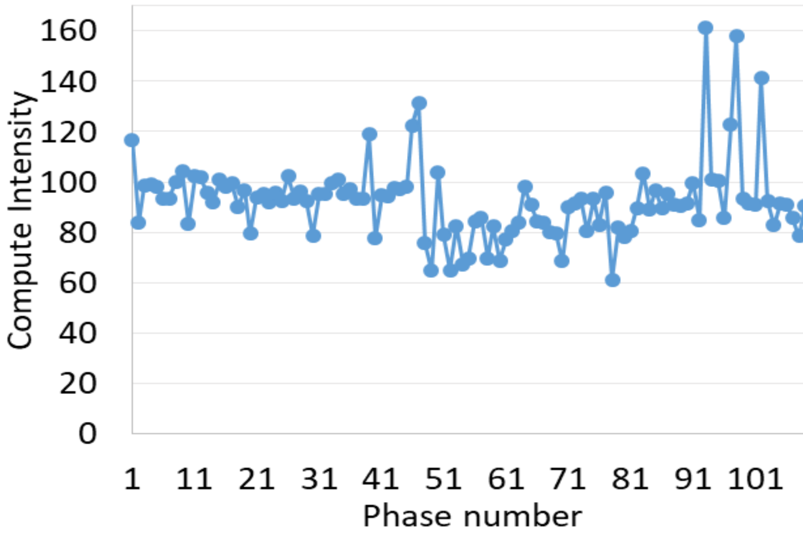
miniMD (2)



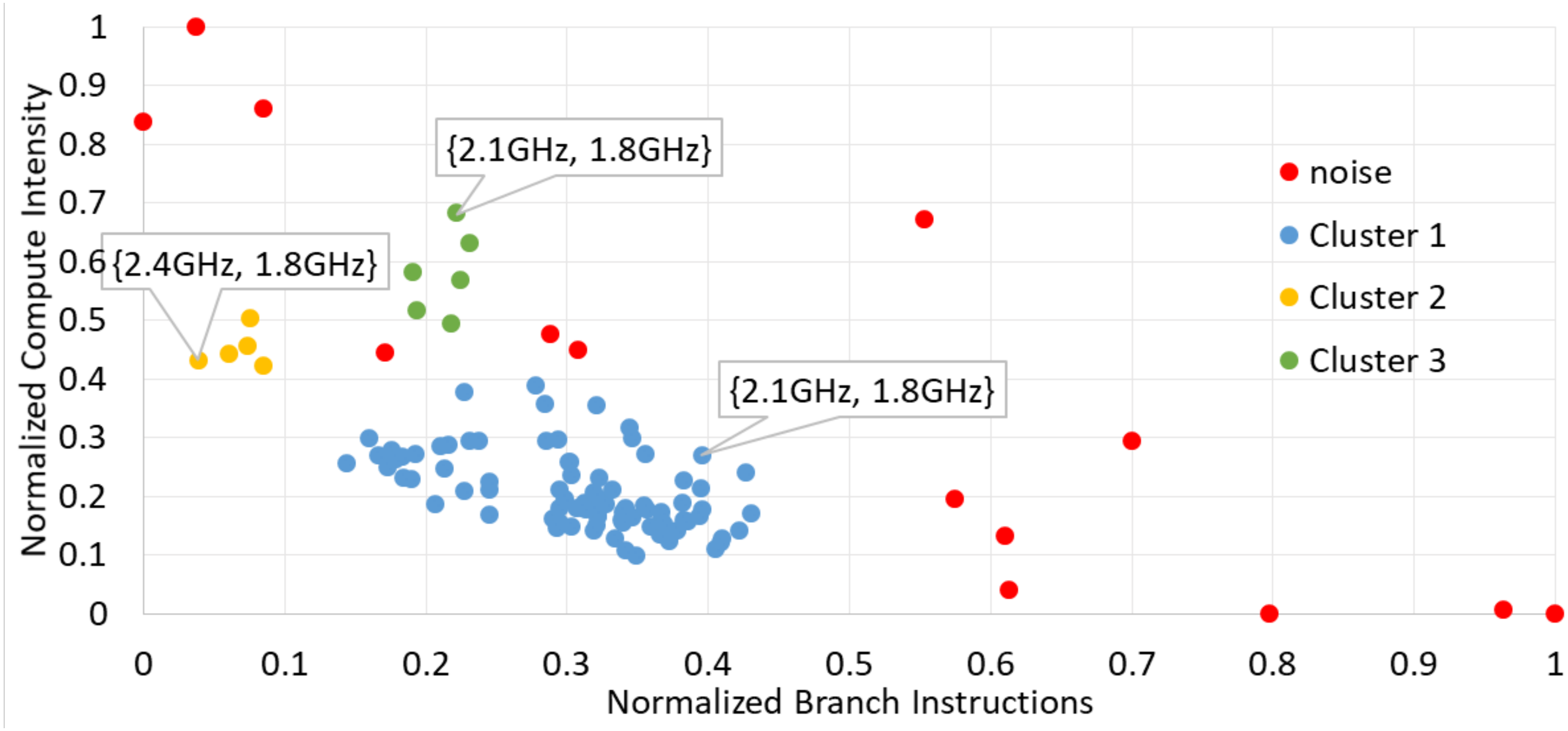
- 6 clusters identified
- Noise points marked in red

INDEED

- INDEED performs sheet metal forming simulations of tools with different geometries moving towards a stationary workpiece
- Contact between tool and workpiece causes:
  - Adaptive mesh refinement
  - Increase in number of finite element nodes
  - Increasing computational cost
- Time loop computes the solution to a system of equations until equilibrium is reached.
- OpenMP version evaluated



INDEED (2)



- 3 clusters identified
- Noise points marked in red

Energy Savings

Application	Phase best for the rts's (%)	rts best for the rts's (%)
miniMD	14.51	0.03
INDEED	9.24	10.45

- miniMD records lower dynamic savings
  - miniMD has only two significant regions
  - One region is called only once during the entire application run
- Better static and dynamic savings for the rts's of INDEED
  - INDEED has nine significant regions
  - Provides more potential for dynamism

Application Tuning Parameters (ATP)

- Exploit the dynamism in characteristics through the use of different code paths (e.g. preconditioners)
- Identify the control variables responsible for control flow switching.
  - provides APIs to annotate the source code

Evaluation of ATP: Espreso

- Finite Element (FEM) tools and domain decomposition based Finite Element Tearing and Interconnect (FETI) solver
  - Contains a projected conjugate gradient (PCG) solver.
  - Convergence can be improved by several preconditioners.
- Evaluated preconditioners on a structural mechanics problem with 23 million unknowns
  - On a single compute node with 24 MPI processes.

15.9 s

4091.5 j

Preconditioner	# iterations	1 iteration		Solution	
None	172	125 ms	31.6 J	21.36 s	5 501.31 J
Weight function	100	130+2 ms	32.3+0.53 J	12.89 s	3 284.07 J
Lumped	45	130+10 ms	32.3+3.86 J	6.32 s	1 636.11 J
Light dirichlet	39	130+10 ms	32.3+3.74 J	5.46 s	1 409.82 J
Dirichlet	30	130+80 ms	32.3+20.62 J	6.34 s	1 594.50 J

Configuration Variable Tuning

- Recently added *readex_configuration* tuning plugin
- Application Configuration Parameters with search space
- Replace selected value in input files and rerun the application

Input Identifiers

- What about different inputs?
- Annotation with *Input Identifiers* like problem size
- Apply Design Time Analysis and merge generated tuning models
- Selection at runtime based on the input identifiers

Summary

- Energy-efficiency tuning
  - Design Time Analysis – Tuning Model – Runtime Tuning
- Support for
  - Intra-phase dynamism
  - Inter-phase dynamism
  - Application Tuning Parameters
  - Application Configuration Parameters
  - Different input configurations
- Based on
  - Periscope Tuning Framework
  - Score-P Monitoring

Thank you! Questions?

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

European Commission
Horizon 2020
European Union funding
for Research & Innovation