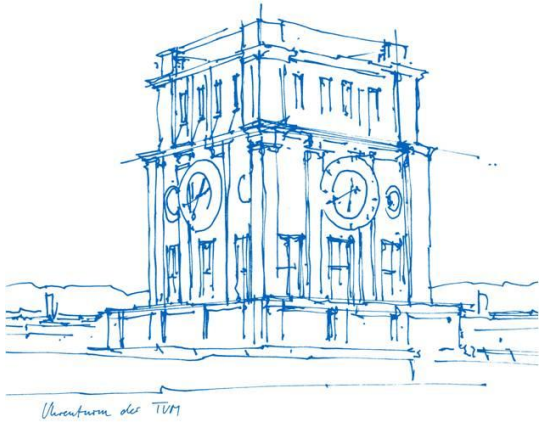
Energy-Efficient Runtime in HPC Systems with Machine Learning

TUM Data Innovation Lab

Gence Özer, Neda Davoudi,

Sarthak Garg, Gabrielle Poerwawinata

Garching, February 2019



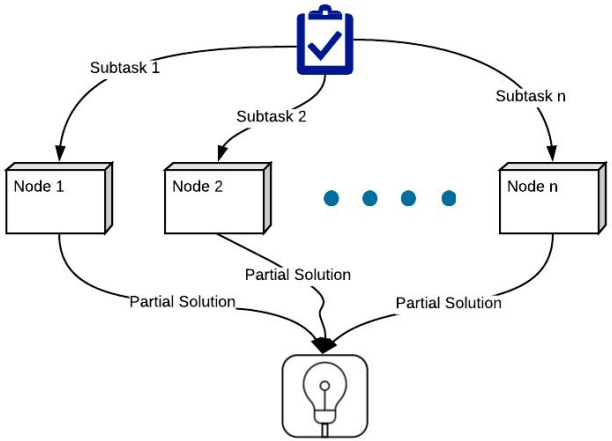
Leibniz Supercomputing Centre
of the Bavarian Academy of Sciences and Humanities

Agenda

1. Introduction
2. Data Collection & Tools
3. Data Exploration & preparation
4. Models
5. Conclusion & Future Work

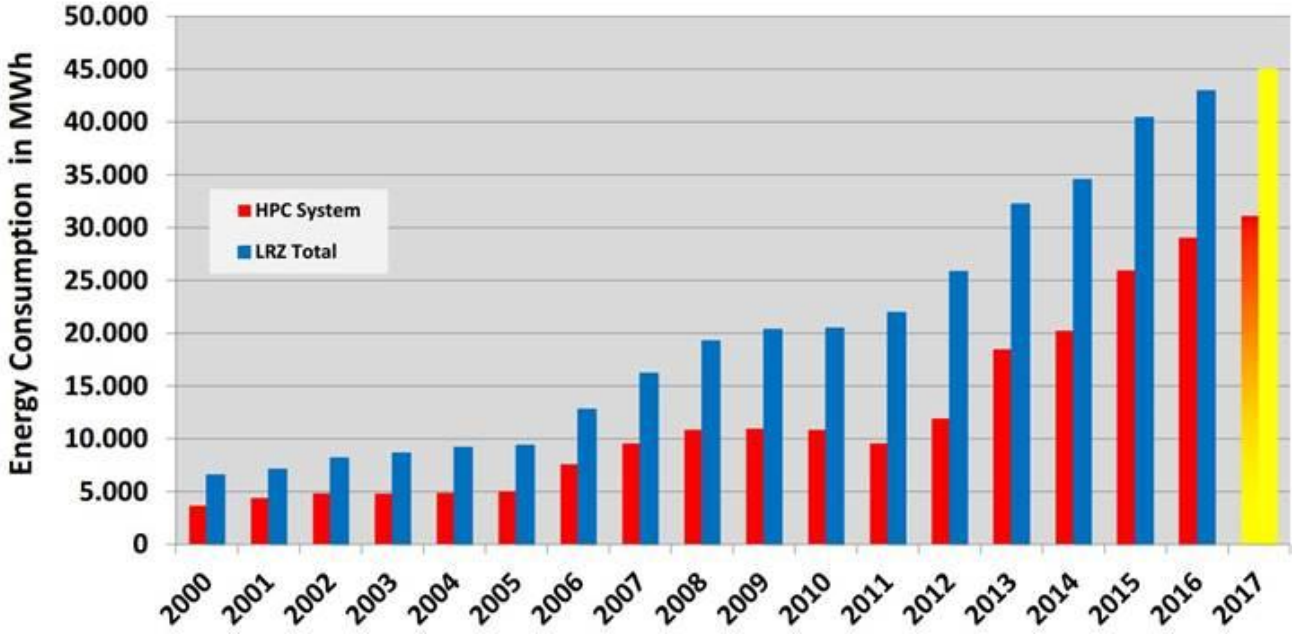
Introduction

- HPC systems enables difficult problems
  - Astrophysics simulations
  - Modeling spread of diseases
  - Calculating the zeros of the Riemann Zeta
- Can accommodate up to millions of cores and terabytes of memory distributed over nodes
- Resources utilized by dividing the tasks into small computational components to run in parallel



Execution of parallel applications

How much energy do HPC systems spend?



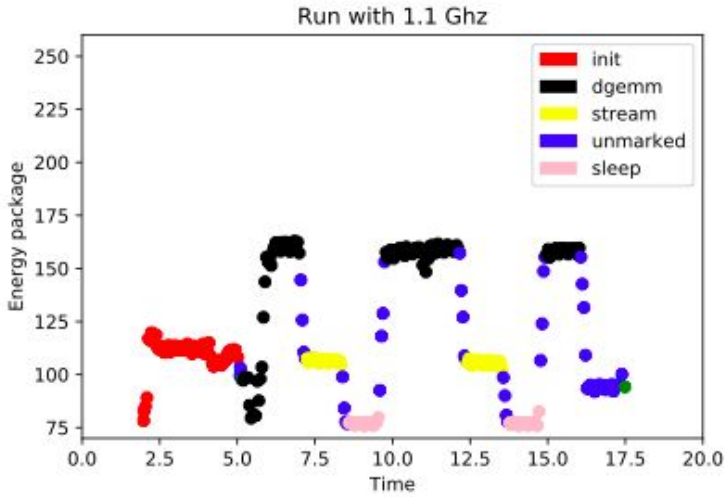
LRZ energy consumption between 2000 - 2017

Severity of Energy Consumption

- Tianhe-2A requires over 18 Megawatt power to operate with full capacity
- 1 hour operating energy equals to travel around the world 3 times with a modern electric car
- Difficult for the operation budgets
- Wasteful energy use is concern for environment

Tianhe-2A,China

What can we do about it?

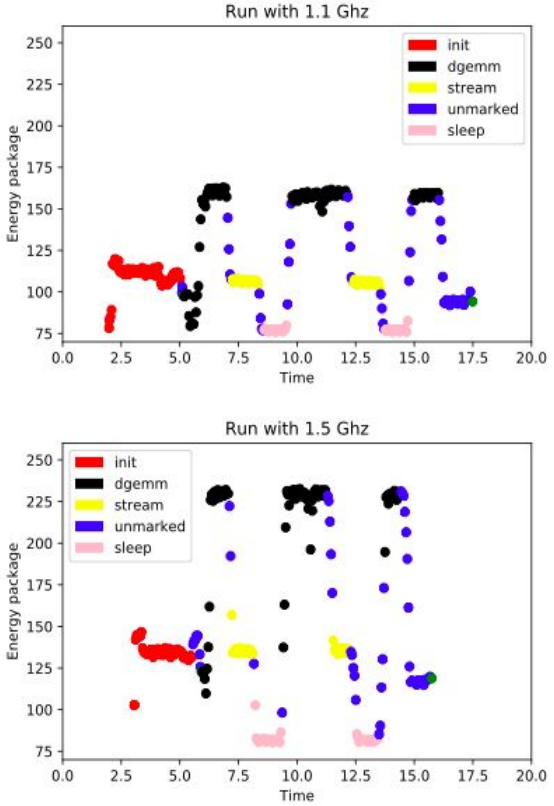


Normalized energy trace 1.1 GHz

- Better hardware, infrastructure and code tuning can increase efficiency
- Runtime control algorithms can have significant impact on efficiency as well
- Regions are application fractions that exhibit similar behaviour
- Modifying the CPU frequency can result in better energy efficiency depending on region

Comparing frequencies

- Traditionally frequency selection is based on heuristics
- Machine learning can be used to predict the optimal frequency in the next timestamp using the information from the previous timestamps
- Optimal frequency minimizes energy with performance constraint



Normalized energy trace 1.1 & 1.5 GHz

Global Extensible Open Power Manager

- Open source community driven runtime solution to address energy problem
- Simple use case: Reads hardware counters and acts with control algorithms
- Generates trace files containing hardware counter readings
- Advanced use case: Dynamically coordinate multiple compute resources by setting dynamic power limits per node basis
- Up to 30% improvement energy consumption on real applications with default control algorithms

Data Generation

- GEOPMbench is an artificial HPC benchmark that can simulate different behaviour using configurations
- Implemented a GEOPM control algorithm to mimic dynamic frequency change behavior on runtime

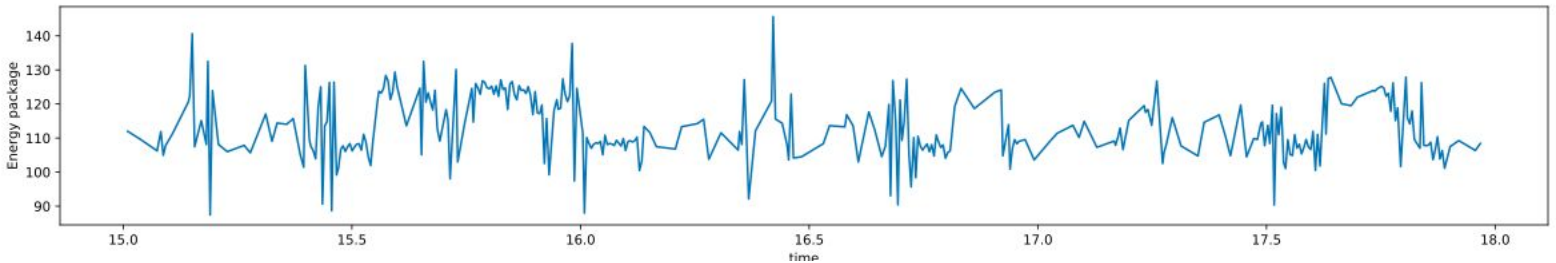
- 300 random configuration for 1 node
- 20 random configuration for different node
- Each configuration is ran 3 times

```
{
  "loop-count": 2,
  "region": ["compute", "memory", "sleep", "memory"],
  "big-o": [1.2, 1.2, 1.2, 1.3]
}
```

Example configuration file

Data Generation

- Performance on artificial data should be validated on real HPC application
- GADGET is a cosmological N-body simulation
- Ran with the control algorithm with GEOPMbench application



Energy trace fragment from GADGET run

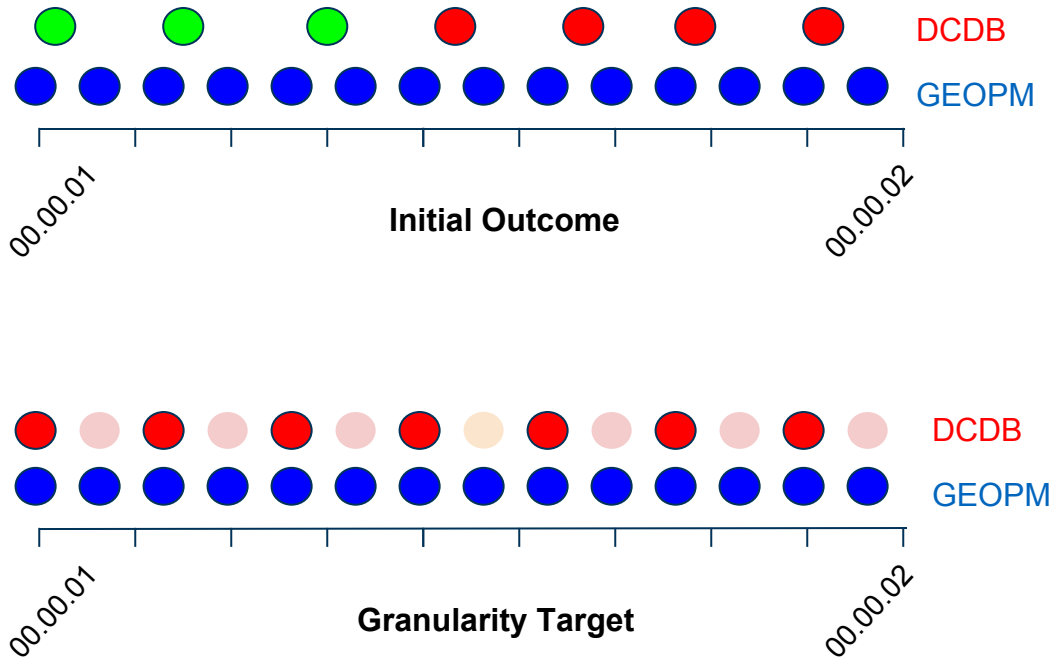
More Features for Traces

- GEOPM has only support for non-programmable hardware counters which limits the amount of data that we can collect
- Programmable counters such as cache misses, branch misses also carries important insights for optimizing the energy efficiency

Data Center DataBase

- DCDB is a tool for collecting and monitoring events counter, driver information and processes information.
- We need other counters,
  - events source: branch-misses, cache-misses
  - device drivers: temperature
  - process information, especially memory usage
- DCDB is configured to save the hardware counters on the nodes that GEOPMbench is run

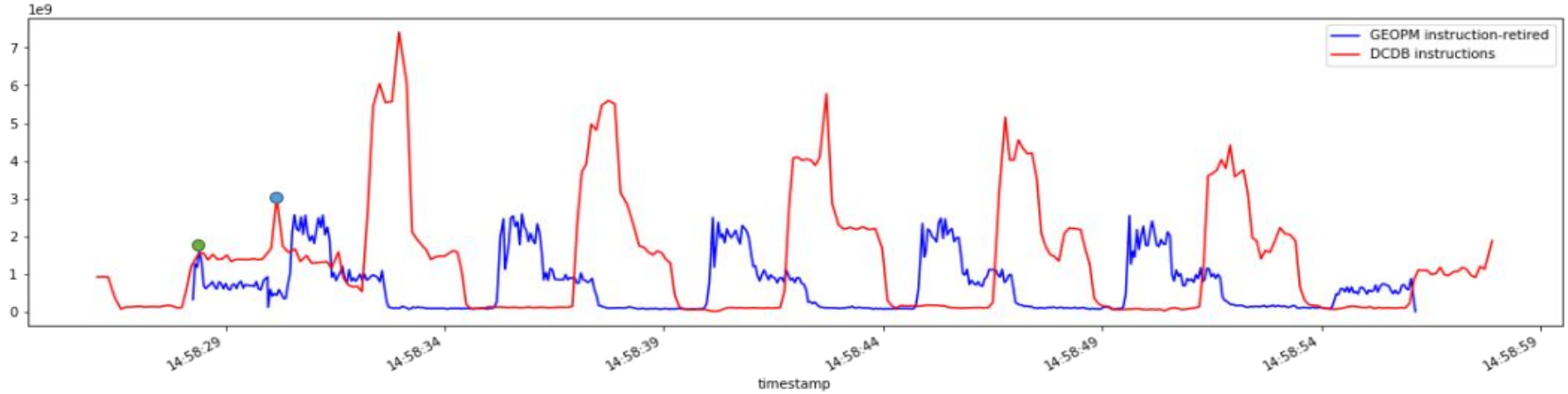
Granularity in Sampling Rate



- Alignment in time series is needed
- Same granularity between 2 data sources
- Converting from monotonic to absolute time, might lead to inaccuracies due to delay in GEOPMbench call

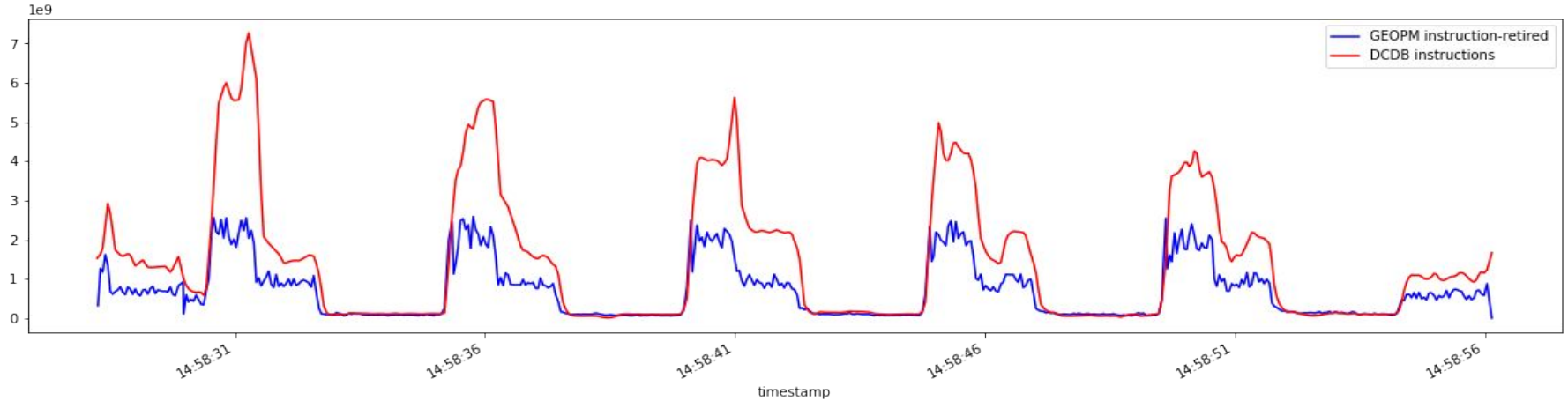
Aligning Time Series

- Removing excess preceding points, approximately 1.7 seconds
- Shift the time series to match the starting point of each series



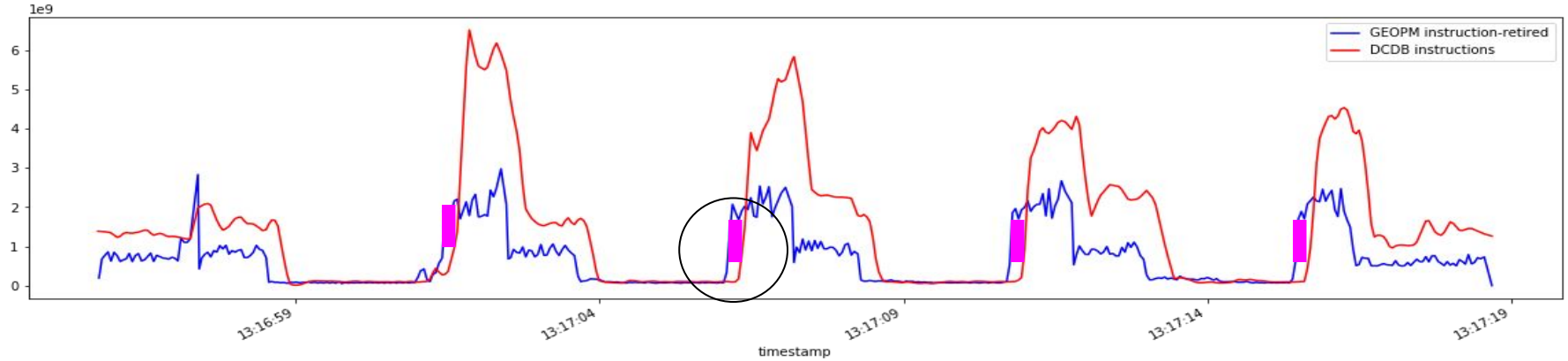
Good Alignment Result

- The excessive point count can vary from trace to trace
- Elimination parameter needs to be adjusted dynamically per sample



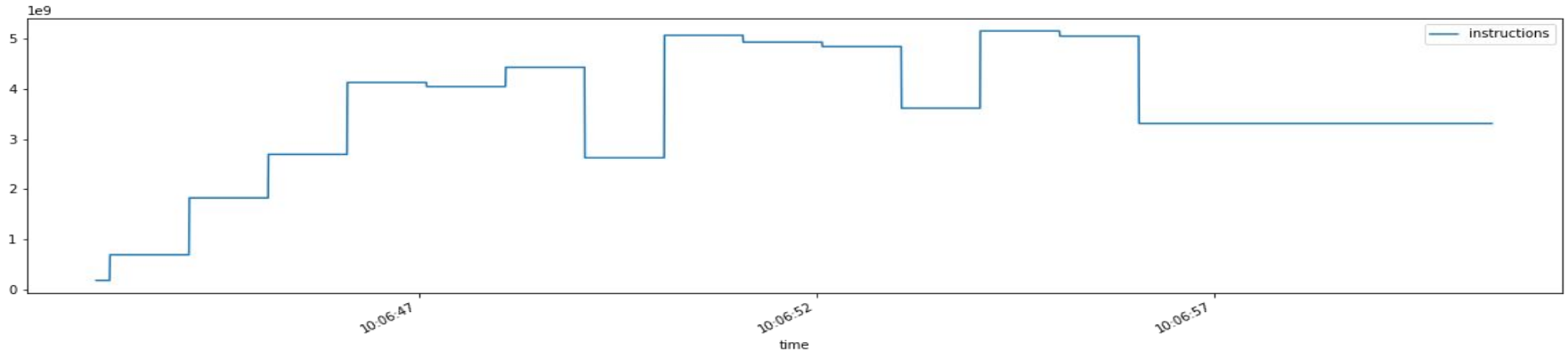
Bad Alignment Result

- Avoid any gap in timestamp



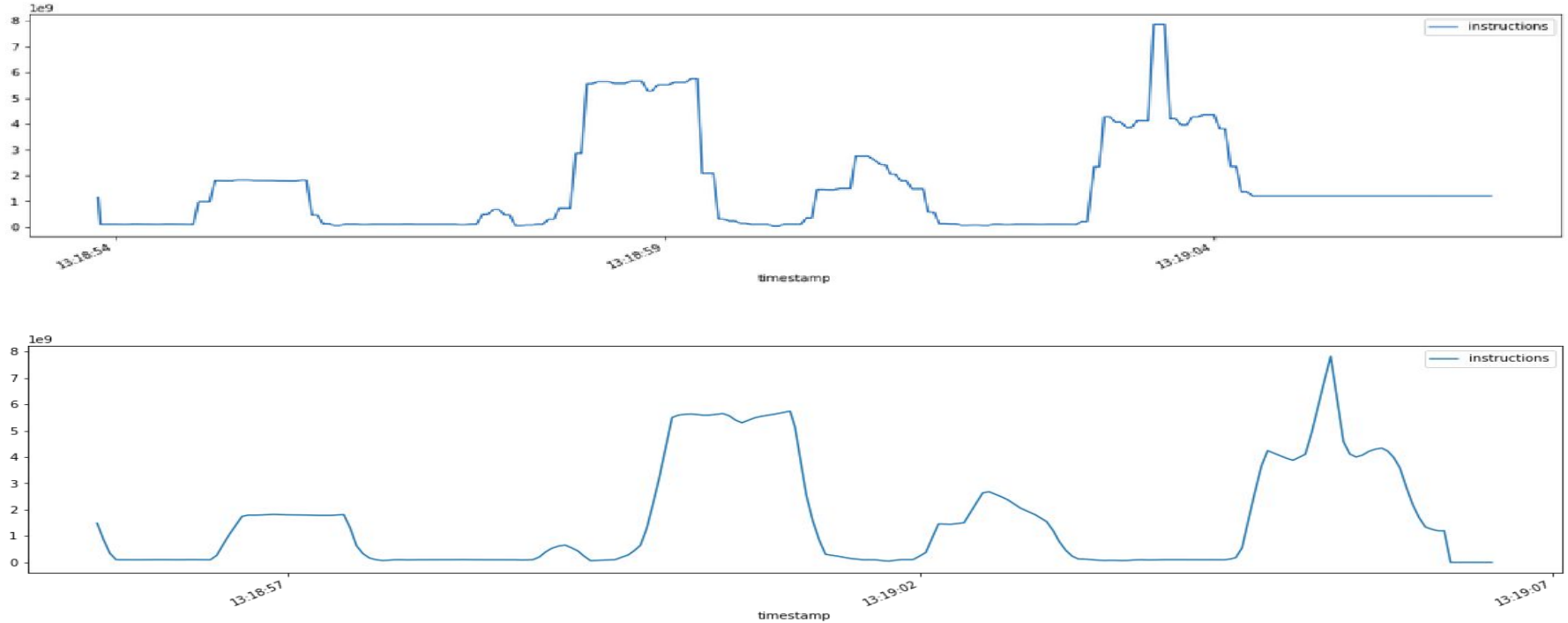
Supersampling, low sampling rate

- One DCDB point were mapped to many GEOPM points



Supersampling with 1s sampling rate

Supersampling vs Interpolation



Supersampling and Interpolation with 100ms sampling rate

Data Collection Summary

- Challenges:
  - 8 minutes required per sample in DCDB data collection
- 70 features in total from GEOPM and DCDB
- Our data key points are:
  - Frequency
  - Region id
  - Package energy
  - Package power
  - Instruction retired

Classification model

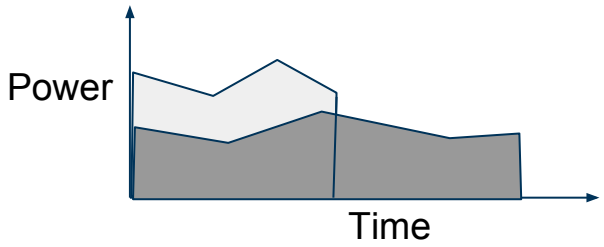
- Detect and classify the current state of the application into characteristic regions
- Choose an optimal frequency policy tailor-made for that region
- In an actual application, the region is 'continuous' and difficult to classify, leading to poor generalization or limited applicability

Regression model

- ‘Forecast' the value of hardware metrics as a function of history values in time series data
- Choose the optimal frequency based on the dependence of forecasted values on frequency
- More promising

Model Objectives

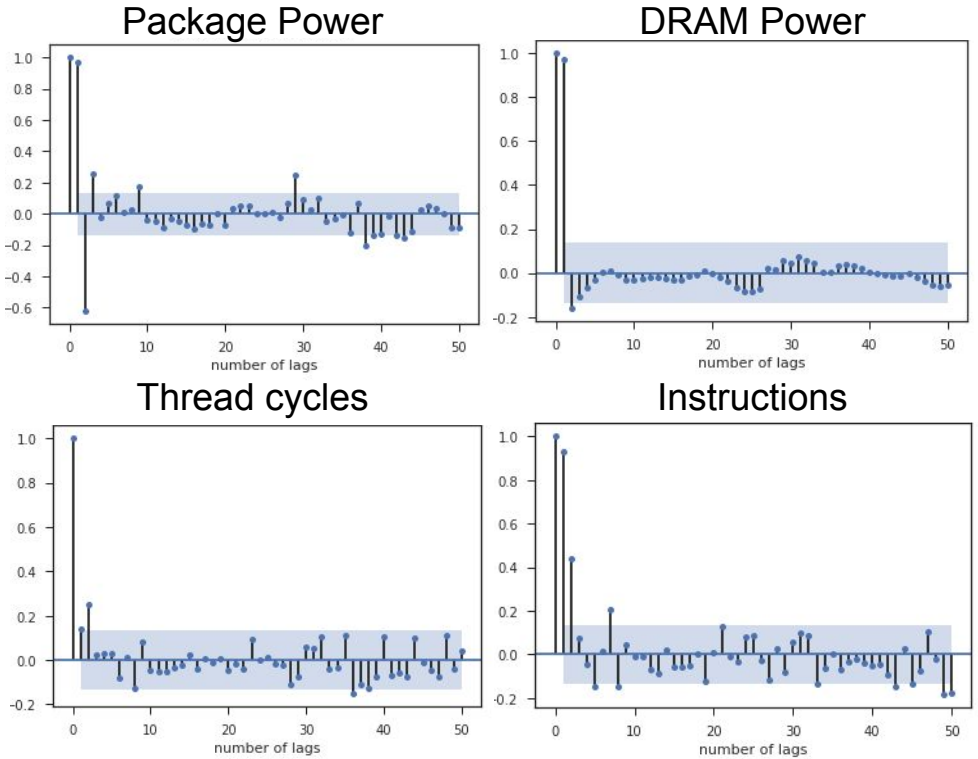
- Good generalization to actual application. This can be tested using validation and testing accuracy, residual plots
- The predicted hardware metrics should be correlated with the frequency to make optimal frequency decisions
- What is the optimal frequency decision policy?
- Instructions retired give a measure of progress
- $E(f) = \frac{\text{Instructions}(f)}{\text{Power}(f)}$; $f_{optimum} = \text{argmax}_f E(f)$



Energy is the area under power-time curve

Partial autocorrelation

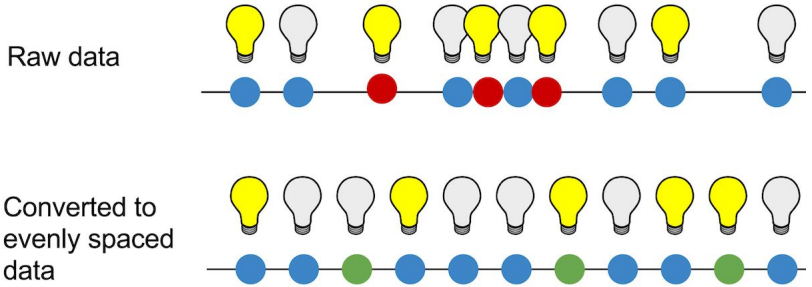
- Application trace (data) is multivariate time series
- Partial correlation of time series with it's own lagged values show which lags are significant
- 2-4 lags enough to decide the present value



Partial autocorrelation plots for key metrics

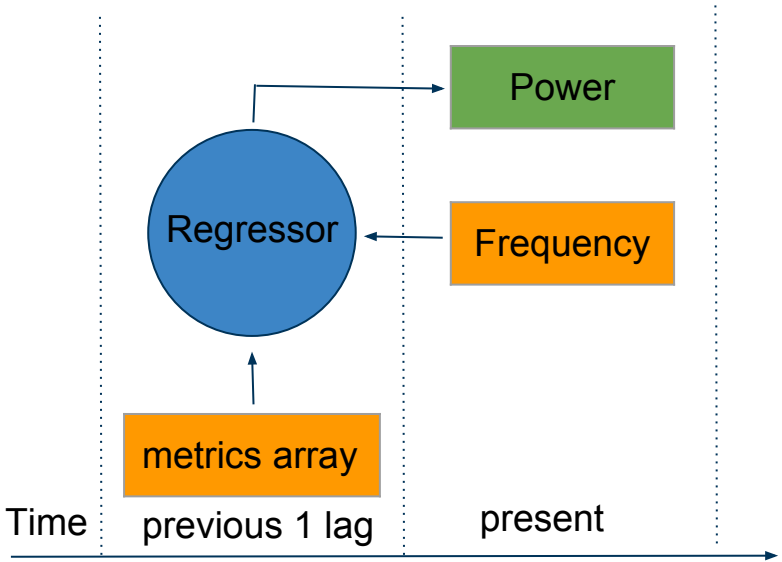
Data preparation

- **Outlier removal:** removed points which don't lie within 3 standard deviations from mean
- **Time Normalization:** to address sampling time variability for accumulating counters
- **Time interpolation:** uniformly spaced time series needed for feature generation. Linear interpolation is used

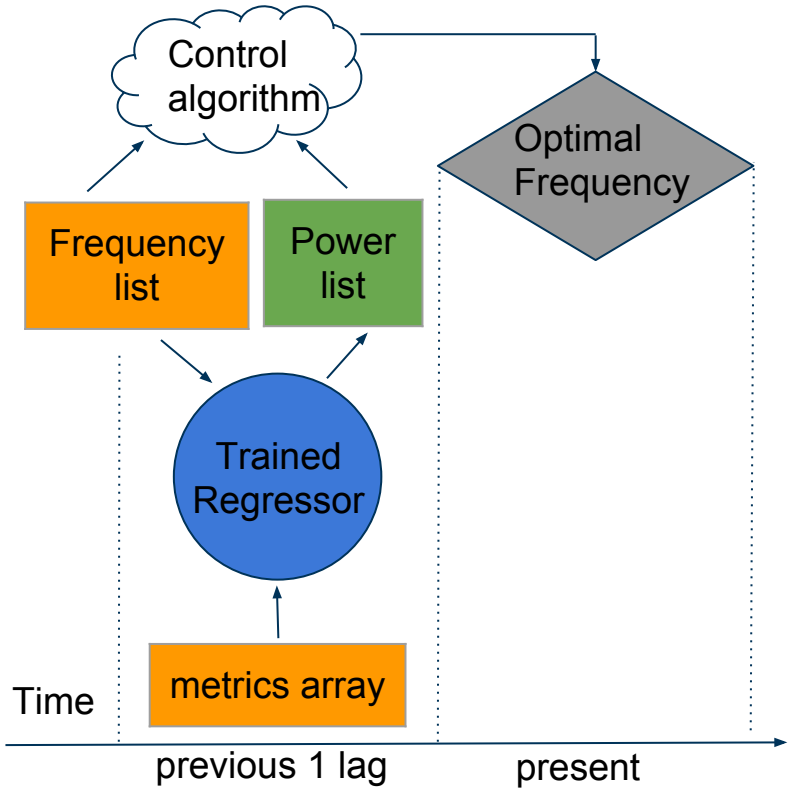


Source: https://datascopeanalytics.com/blog/unevenly-spaced-time-series

One-lag Model



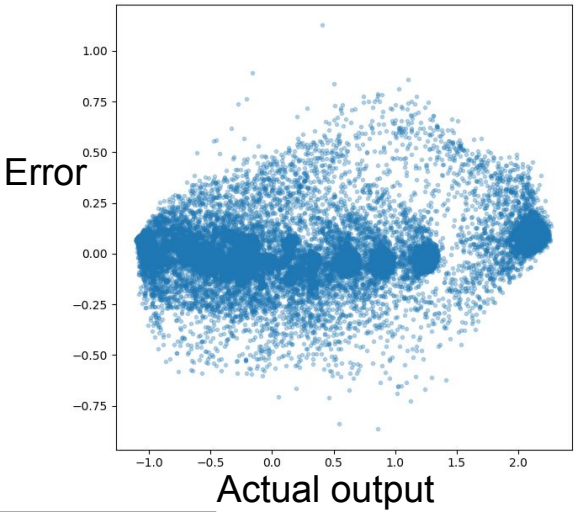
Training (offline)



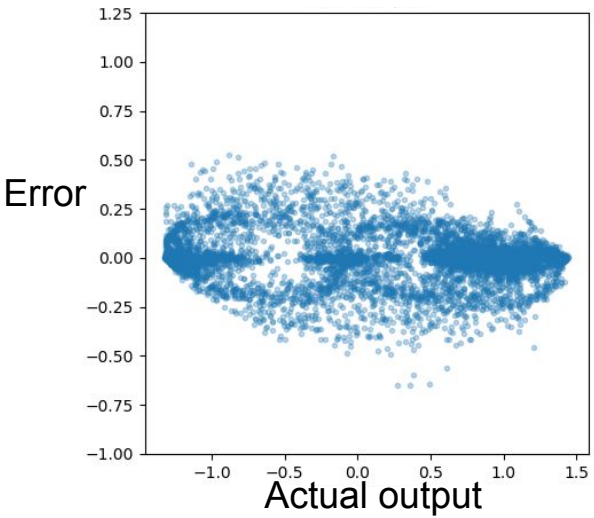
Inference (runtime)

Comparison: One-lag Models

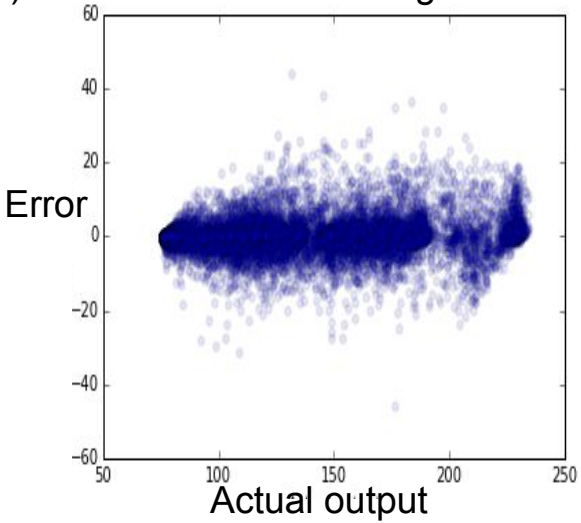
Ridge Linear Regression



Support vector Regression (RBF)



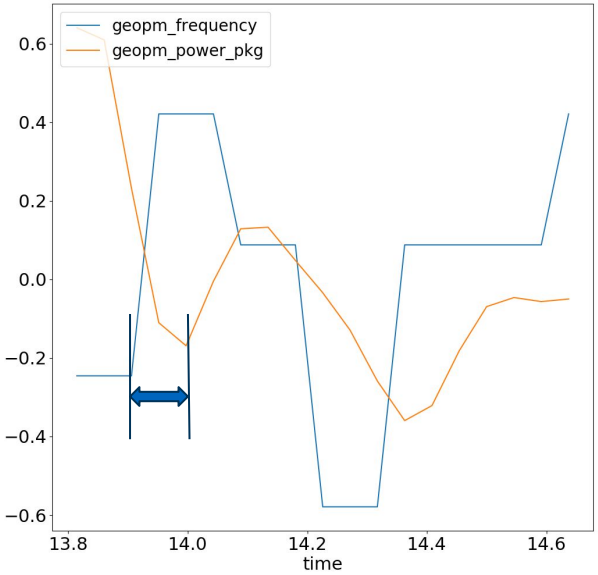
Random forest Regression



Mean relative error	0.181	0.074	0.024

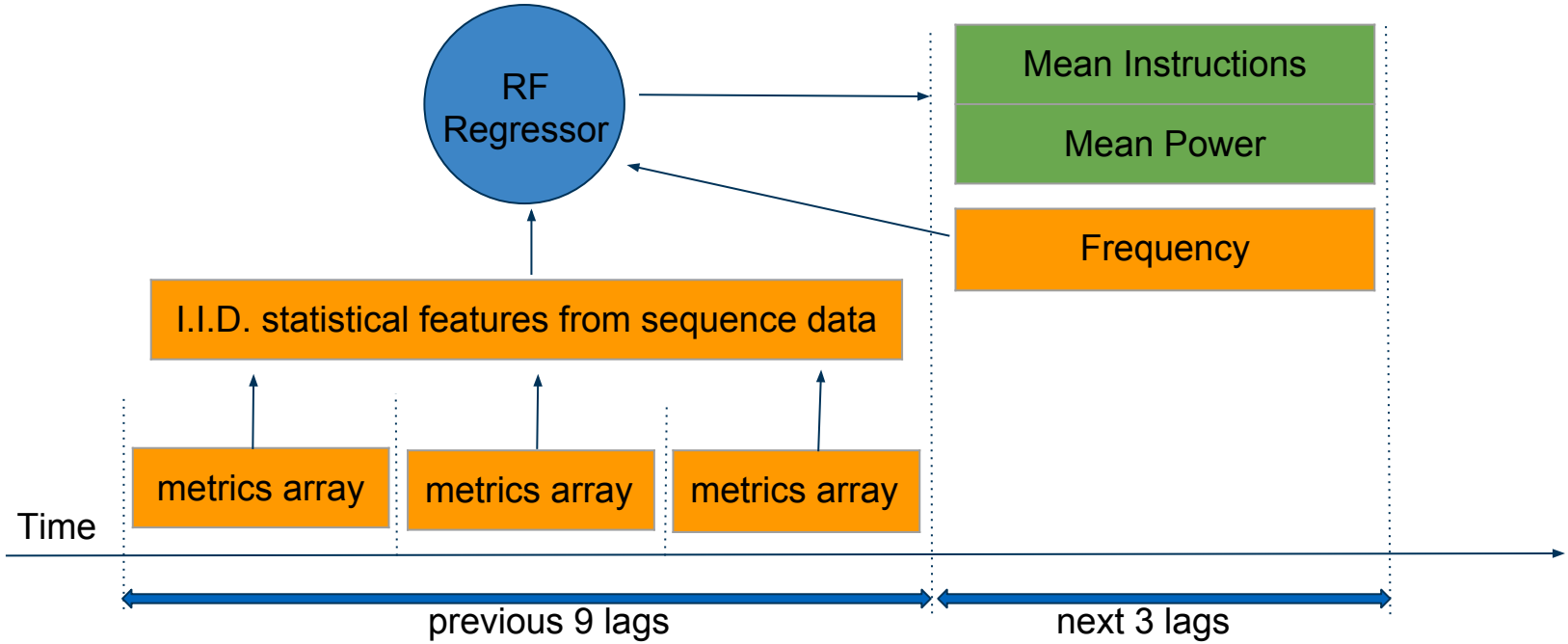
Issues with one-lag models

- Correlation of frequency and power is low (2.4 per 0.1 GHz)
- Frequency lags ahead by about 50-100 ms (1-2 samples). One lag model can’t compensate for this lag
- More reasonable frequency correlation within a ‘region’. Multiple lag model can capture local region behavior
- Predicting power alone not enough for optimal frequency decision. Predict instructions retired also



Frequency - power lag

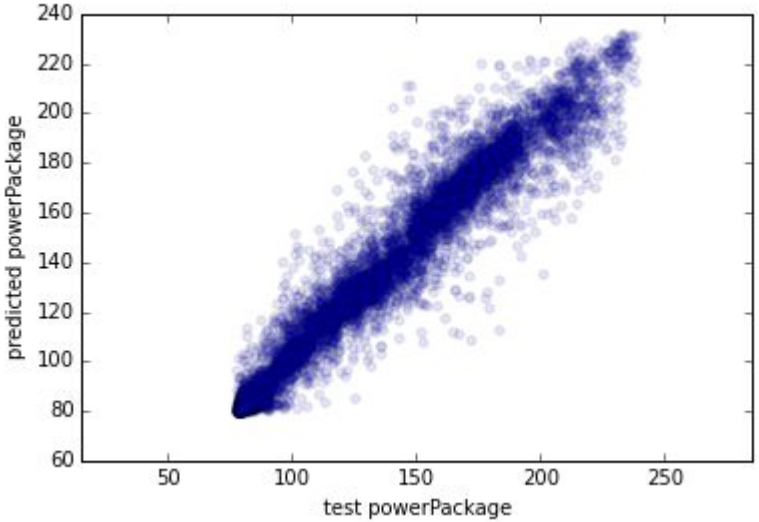
Multiple-lag Model



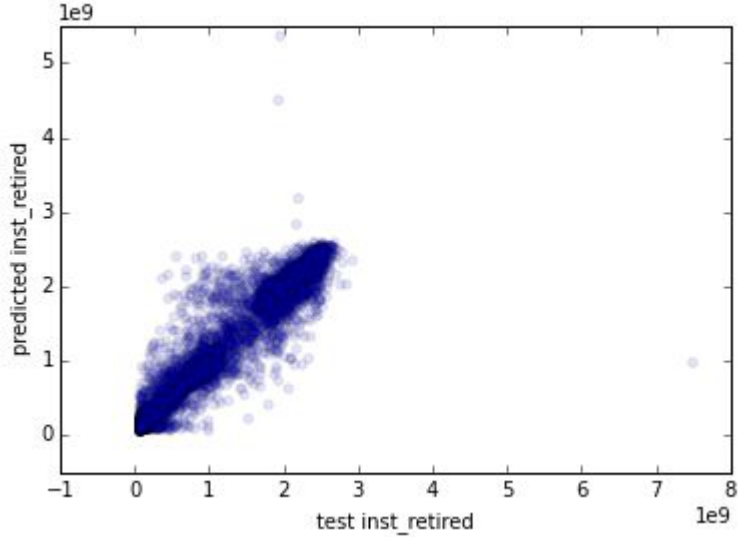
Statistical features

Sl. no.	Feature	Remarks
1	Exponentially weighted mean	‘recent’ values more important.
2	Exponentially weighted gradient	depict trends
3	Standard Deviation	variability
4	Skewness	3rd moment
5	Kurtosis	4th moment
6,7,8	Quantile - 0.25, 0.5, and 0.75	Spread of intervals
9	Sum of differences	sudden changes
10	Sample Entropy	measure of complexity

Results - GEOPM



Predicted Vs Actual (package power)

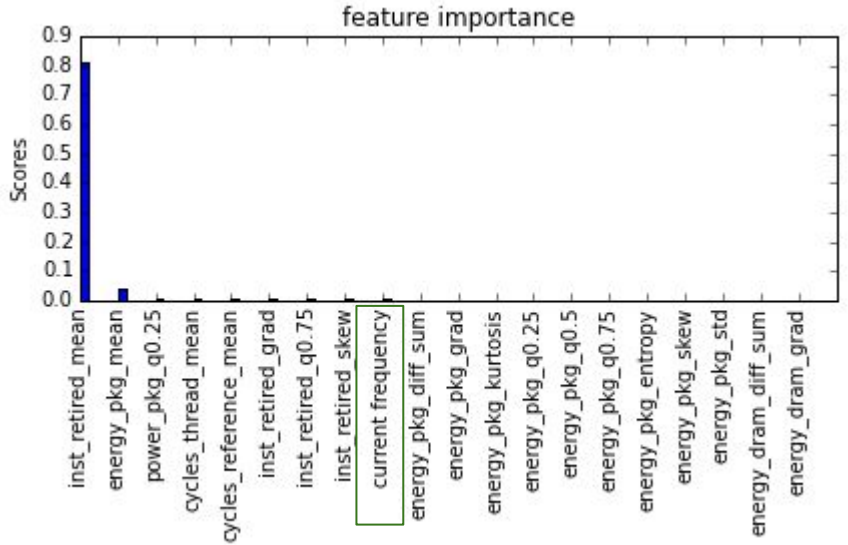


Predicted Vs Actual (inst_retired)

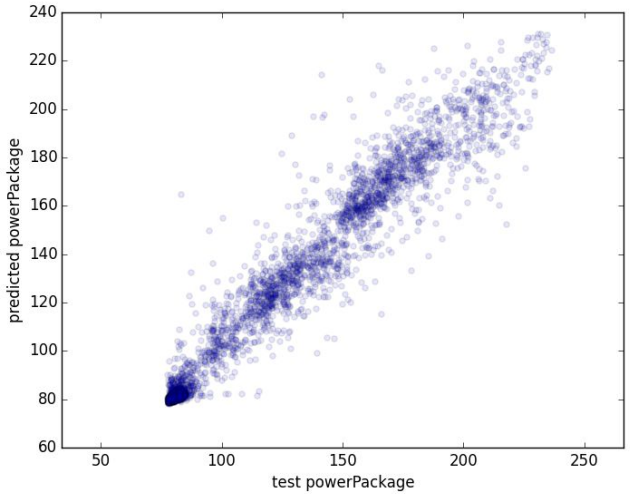
Mean Relative Error: 0.135

Correlation with Frequency

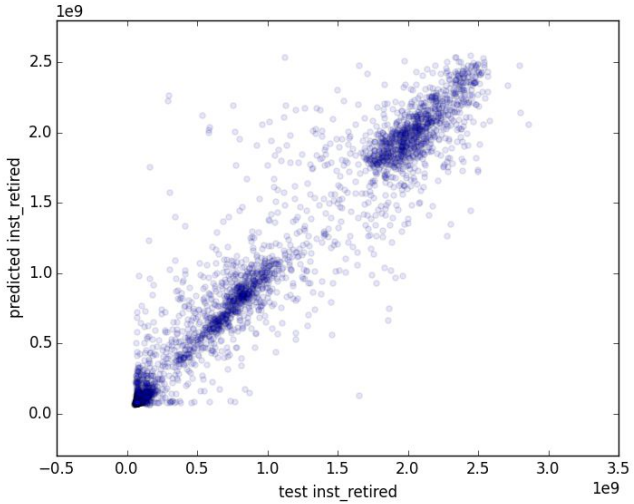
- Instruction retired: Highest impact
- Frequency: Slight correlation
- Energy dram: No contribution



Results - GEOPM + DCDB



Predicted Vs Actual (package power)

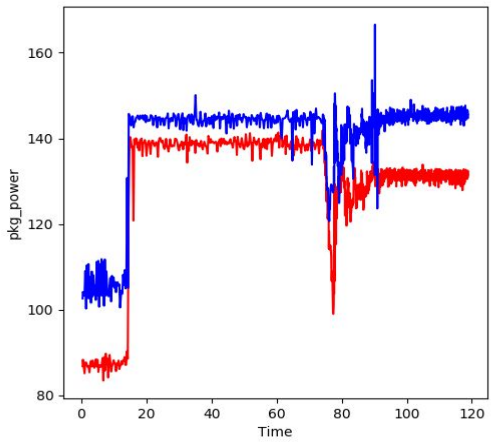


Predicted Vs Actual (inst_retired)

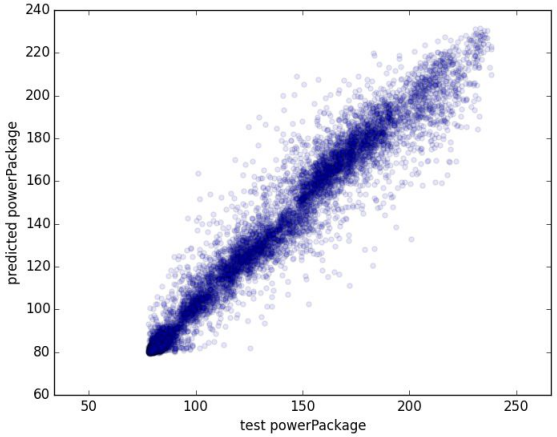
Mean Relative Error: 0.135

Results - Test Node

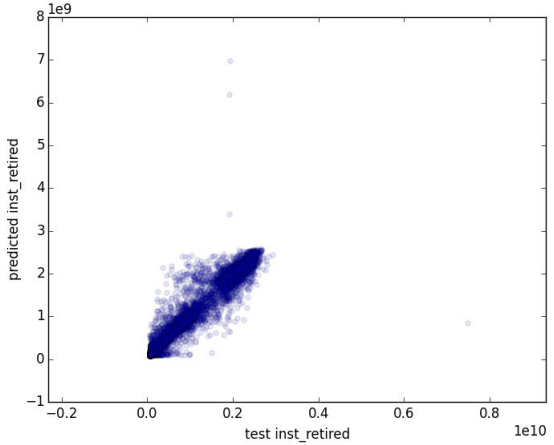
- Model Validation with time series data of same run on another node



Power traces on different nodes



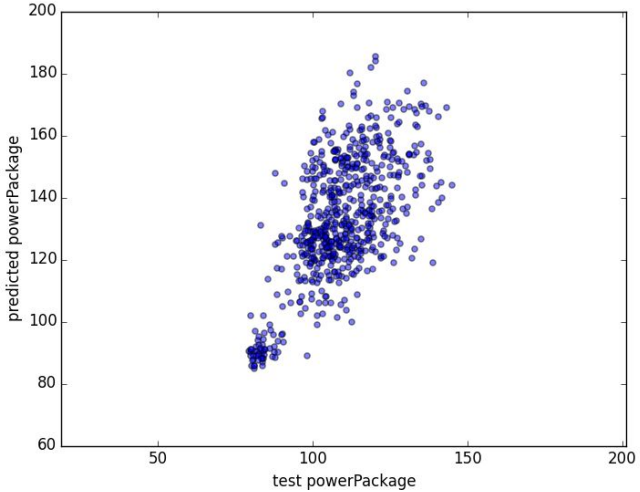
Predicted Vs Actual (package power)



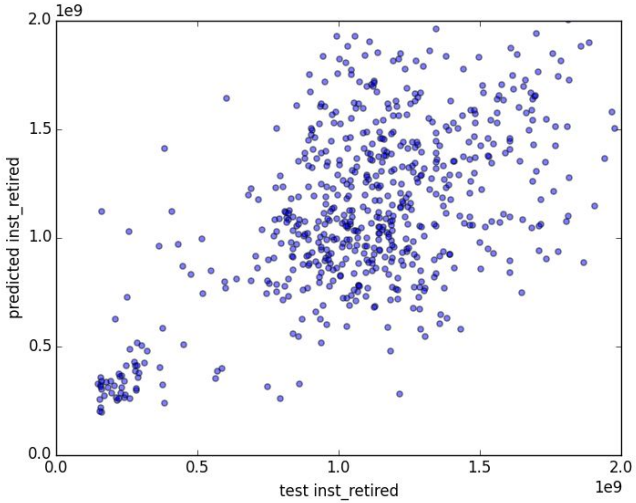
Predicted Vs Actual (inst_retired)

Mean Relative Error: 0.132

Results - Gadget



Predicted Vs Actual (package power)



Predicted Vs Actual (inst_retired)

Mean Relative Error: 0.261

Discussion and Future Works

- Our Random Forest model generalizes well to real applications
- Multiple-lag model performs better than one lag model
- Predicted metrics don’t depend ‘enough’ on frequency to make good decisions
- Implementing a GEOPM agent to get energy saving by current model
- Recurrent Neural Networks may perform better

Summary

- Data collection
  - GEOPM
  - DCDB
- Data Analysis and Preparation
  - Region-specific behaviour
  - Statistical features
  - Aligning software and hardware data
- Machine Learning Implementation
  - Linear and Nonlinear Model
  - Performance Metrics as Targets
- Model Validation
  - Gadget Data
  - Different Node

Thank you for your attention

Special thanks to GEOPM team

Questions?

Features from GEOPMbench

Hardware metrics	Description
Package energy	Total energy of all sockets(CPU) of the node
DRAM energy	Total energy of all DRAM of the node
Package power	Total power of all sockets(CPU) of the node
DRAM energy	Total power of all DRAM of the node
Frequency	Average frequency of all cores.
Thread cycles	Average clock cycles executed by the cores since the beginning of the execution
reference cycles	Average clock reference cycles since the beginning of the execution
Instructions retired	Total number of instructions executed by all sockets(CPU) on the node

Features from GEOPMbench

Application related metrics	Description
Region ID	64-bit integer ID of the region that all ranks are running.
region progress	minimum per-rank reported progress through the current region (0 or 1)
region runtime	maximum per-rank last recorded runtime for the current region
power limit, policy power cap, policy step count, policy max epoch runtime, policy power slack, policy power limit	GEOPM policy related metrics

Region based regression

- More reasonable frequency correlation ‘within’ region.
- A multiple lag model can capture ‘region’ behaviors.

	pkg power/0.5GHz freq	dram power/0.5GHz freq
Sleep Region	2.7	-0.04
Dgemm Region	16.2	1.25
Stream Region	9.6	1.8
Validation acc.	80 %	95 %

One lag with Power output

Dataset / Model	Mean Relative Error
Fixed Frequency / Linear Regression	0.174
Fixed Frequency / Random Forest	0.023
Varying Frequency / Linear Regression	0.181
Varying Frequency / SVM	0.074
Varying Frequency / Random Forest	0.024