Multi-Dimensional Analysis of Software Power Consumptions in Multi-Core Architectures

Maxime Colmant — PhD Defense — 24th November, 2016

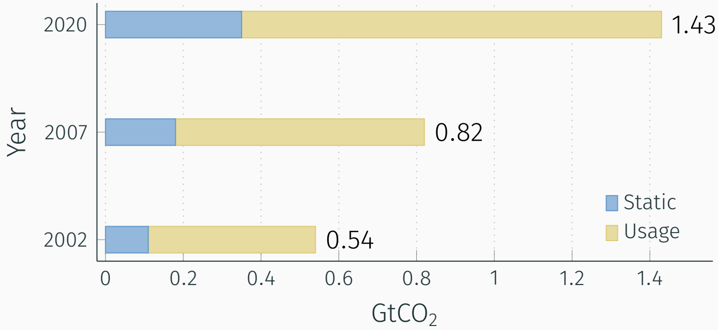
Jury:

Mr.	Olivier	Barais	University Rennes 1	Reporter
Mr.	Rüdiger	Kapitza	University Braunschweig	Reporter
Mr.	Giuseppe	Lipari	University Lille 1	Examiner
Mrs.	Anne-Cécile	Orgerie	CNRS	Examiner
Mr.	Romain	Rouvoy	University Lille 1	Supervisor
Mr.	Lionel	Seinturier	University Lille 1	Supervisor
Mr.	Alain	Anglade	ADEME	Guest

Table of contents

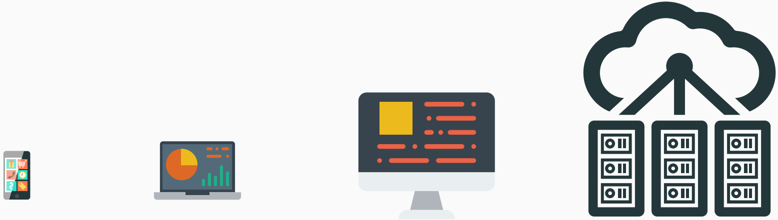
Introduction

The Global ICT[1] Footprint[2]

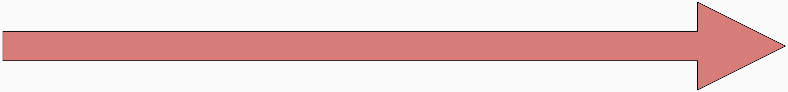


[1]Information and Communications Technology

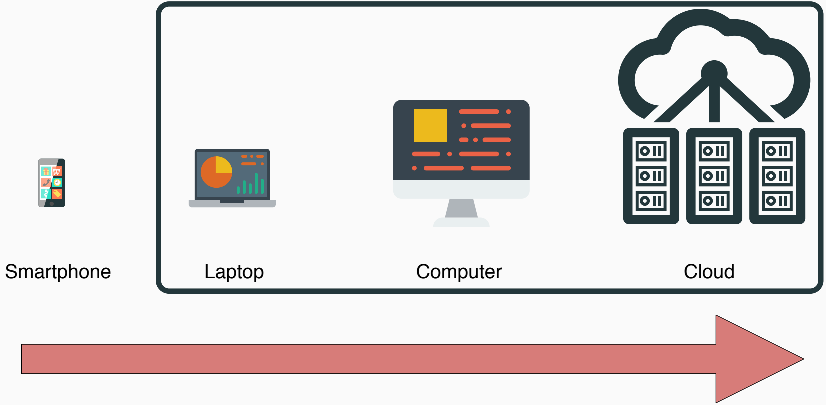
[2]The Climate Group. *SMART 2020: Enabling the low carbon economy in the information age.* 2008.



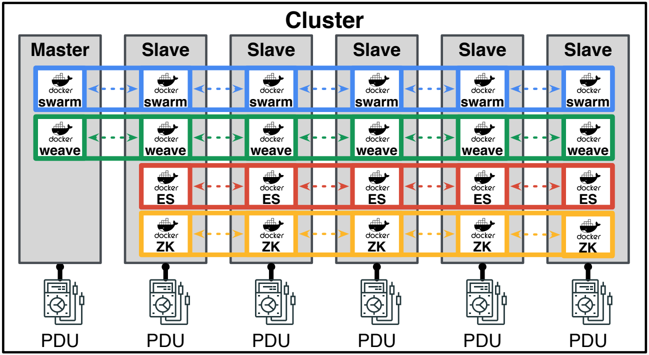
Smartphone Laptop Computer Cloud

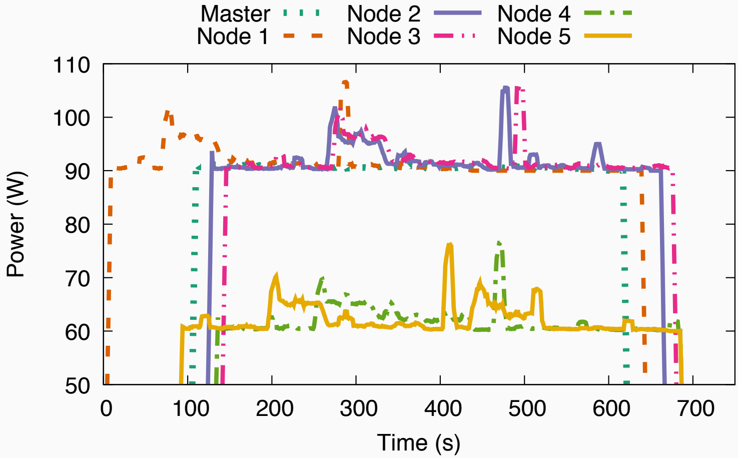


Multi-core CPU Architectures Are Everywhere!

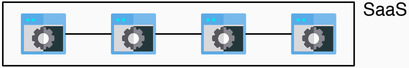


Case Study

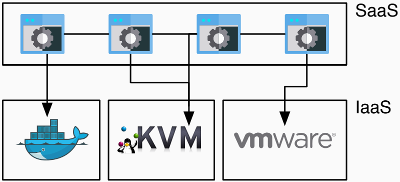

Cluster
Master
Slave
Slave
Slave
Slave
Slave
docker swarm
docker weave
docker ES
docker ZK
PDU


Master
Node 1
Node 2
Node 3
Node 4
Node 5
Power (W)
110
100
90
80
70
60
50
0
100
200
300
400
500
600
700
Time (s)

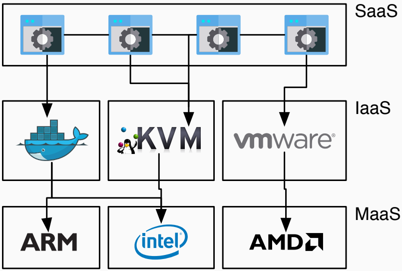
What is a Multi-Core Software System?



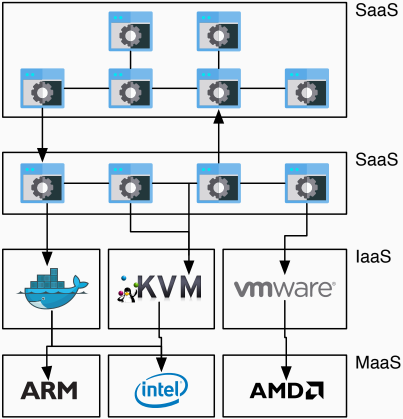
What is a Multi-Core Software System?



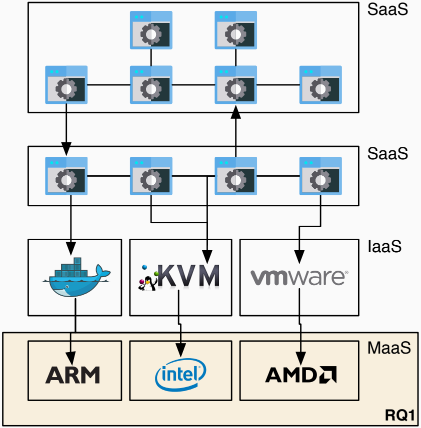
What is a Multi-Core Software System?



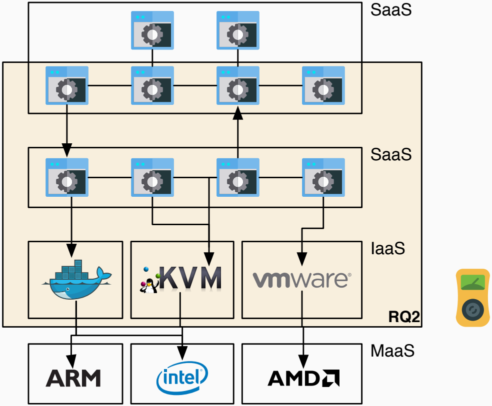
What is a Multi-Core Software System?



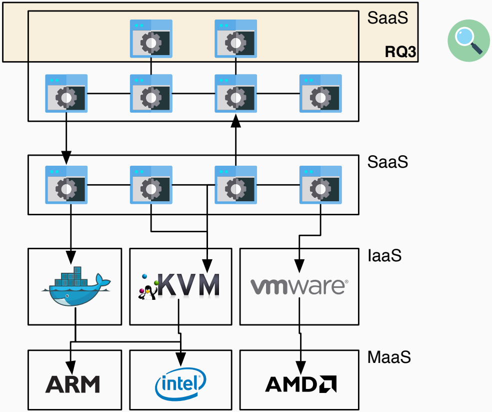
RQ1: Can we model the software power consumption regardless of the underlying architecture?



RQ2: Can we propose a uniform view of the service power consumption?

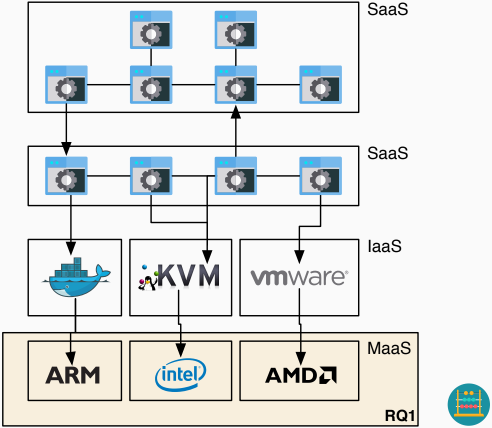


RQ3: Can we analyze the power consumption of the artifacts which compose a software?

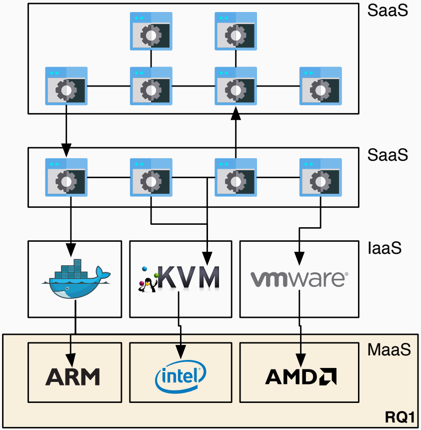


Contributions

RQ1: Can we model the software power consumption regardless of the underlying architecture?



RQ1: Can we model the software power consumption regardless of the underlying architecture?



Learning CPU Power Models

What is a Power Model?

- Math. function (metrics) $\Rightarrow$ Power

What is a Power Model?

- Math. function (metrics) $\Rightarrow$ Power
- Mostly linear

    Univariate: $P = a_x + b$
    Multivariate: $P = a_x + b_y + c$

What is a Power Model?

- Math. function (metrics) $\Rightarrow$ Power
- Mostly linear

  Univariate: $P = a_x + b$

  Multivariate: $P = a_x + b_y + c$

- Or polynomial

  $P = a_{x^2} + b_x + c$

What is a Power Model?

- Math. function (metrics) $\Rightarrow$ Power
- Mostly linear

  Univariate: $P = a_x + b$

  Multivariate: $P = a_x + b_y + c$
- Or polynomial

  $P = a_{x^2} + b_x + c$
- CPU metrics

  From HW sensors (motherboard, power meters)

  From Hardware Performance Counters (HPCs)

What is a Power Model?

- Math. function (metrics) $\Rightarrow$ Power
- Mostly linear

  Univariate: $P = a_x + b$

  Multivariate: $P = a_x + b_y + c$
- Or polynomial

  $P = a_{x^2} + b_x + c$
- CPU metrics

  From HW sensors (motherboard, power meters)

  From Hardware Performance Counters (HPCs)

- [Nou14][3]: $P_{cpu}^{app} = 0.7 * TDP * CPU_{stats}$

[3] A. Noureddine. “Towards a Better Understanding of the Energy Consumption of Software Systems”. PhD thesis. Université des Sciences et Technologie de Lille - Lille I, 2014.

Existing CPU Power Models

Ref.	Processor(s)	Feature(s)	Regression(s)	Benchmarks
[Ber+10]	Core 2 Duo	14 PCs regrouped by component	multiple linear by component	*sampl.:* μ-benchs *eval.:* SPEC CPU 06
[Col+15]	Xeon W3520 & i3 2120	non-halted cycles reference cycles	polynomial	*sampl.:* stress *eval.:* PARSEC, SPECjbb
[CM05]	XScale PXA255	5 PCs	multiple linear	*eval.:* SPEC CPU 00, Java CDC/CLDC
[Dol+15]	Xeon E3-1275	3 PCs HW sensors	linear	*sampl.:* linpack, stream, iperf, IOR *eval.:* Quantum Espresso
[ERK06]	Turion, Itanium 2	HW sensors	multiple linear	*sampl.:* Gamut *eval.:* SPECs, Matrix, Stream
[IM03]	Pentium 4	15 PCs	multiple linear	*eval.:* μ-benchs, AbiWord, Mozilla, Gnumeric
[RRK08]	Core 2 Duo & Xeon, Itanium 2, Turion	HW sensors PCs	multiple linear	*sampl.:* calibration suite *eval.:* SPECs, stream, Nsort
[Yan+14]	Xeon E5620 & E7530	7 components 91 preselected	support vector	*sampl.:* NPB, IOzone, CacheBench *eval.:* SPEC CPU 06, IOzone
[Zha+14]	Sandy Bridge	non-halted cycles	linear	*eval.:* Google, SPEC CPU 06
???	ARM	???	???	???

Only for Intel or AMD architectures

Existing CPU Power Models

Ref.	Processor(s)	Feature(s)	Regression(s)	Benchmarks
[Ber+10]	Core 2 Duo	14 HPCs regrouped by component	multiple linear by component	*sampl.:* μ-benchs *eval.:* SPEC CPU 06
[Col+15]	Xeon W3520 & i3 2120	non-halted cycles reference cycles	polynomial	*sampl.:* stress *eval.:* PARSEC, SPECjbb
[CM05]	XScale PXA255	5 HPCs	multiple linear	*eval.:* SPEC CPU 00, Java CDC/CLDC
[Dol+15]	Xeon E3-1275	3 HPCs HW sensors	linear	*sampl.:* linpack, stream, iperf, IOR *eval.:* Quantum Espresso
[ERK06]	Turion, Itanium 2	HW sensors	multiple linear	*sampl.:* Gamut *eval.:* SPECs, Matrix, Stream
[IM03]	Pentium 4	15 HPCs	multiple linear	*eval.:* μ-benchs, AbiWord, Mozilla, Gnumeric
[RRK08]	Core 2 Duo & Xeon, Itanium 2, Turion	HW sensors HPCs	multiple linear	*sampl.:* calibration suite *eval.:* SPECs, stream, Nsort
[Yan+14]	Xeon E5620 & E7530	7 components 91 preselected	support vector	*sampl.:* NPB, IOzone, CacheBench *eval.:* SPEC CPU 06, IOzone
[Zha+14]	Sandy Bridge	non-halted cycles	linear	*eval.:* Google, SPEC CPU 06

HW sensors: coarse-grained CPU metrics

Existing CPU Power Models

Ref.	Processor(s)	Feature(s)	Regression(s)	Benchmarks
[Ber+10]	Core 2 Duo	14 HPCs regrouped by component	multiple linear by component	*sampl.:* μ-benchs *eval.:* SPEC CPU 06
[Col+15]	Xeon W3520 & i3 2120	non-halted cycles reference cycles	polynomial	*sampl.:* stress *eval.:* PARSEC, SPECjbb
[CM05]	XScale PXA255	5 HPCs	multiple linear	*eval.:* SPEC CPU 00, Java CDC/CLDC
[Dol+15]	Xeon E3-1275	3 HPCs HW sensors	linear	*sampl.:* linpack, stream, iperf, IOR *eval.:* Quantum Espresso
[ERK06]	Turion, Itanium 2	HW sensors	multiple linear	*sampl.:* Gamut *eval.:* SPECs, Matrix, Stream
[IM03]	Pentium 4	15 HPCs	multiple linear	*eval.:* μ-benchs, AbiWord, Mozilla, Gnumeric
[RRK08]	Core 2 Duo & Xeon, Itanium 2, Turion	HW sensors HPCs	multiple linear	*sampl.:* calibration suite *eval.:* SPECs, stream, Nsort
[Yan+14]	Xeon E5620 & E7530	7 components 91 preselected	support vector	*sampl.:* NPB, IOzone, CacheBench *eval.:* SPEC CPU 06, IOzone
[Zha+14]	Sandy Bridge	non-halted cycles	linear	*eval.:* Google, SPEC CPU 06

HPCs: fine-grained CPU metrics

Ref.	Processor(s)	Feature(s)	Regression(s)	Benchmarks
[Ber+10]	Core 2 Duo	14 HPCs regrouped by component	multiple linear by component	*sampl.:* μ-benchs *eval.:* SPEC CPU 06
[Col+15]	Xeon W3520 & i3 2120	non-halted cycles reference cycles	polynomial	*sampl.:* stress *eval.:* PARSEC, SPECjbb
[CM05]	XScale PXA255	5 HPCs	multiple linear	*eval.:* SPEC CPU 00, Java CDC/CLDC
[Dol+15]	Xeon E3-1275	3 HPCs HW sensors	linear	*sampl.:* linpack, stream, iperf, IOR *eval.:* Quantum Espresso
[ERK06]	Turion, Itanium 2	HW sensors	multiple linear	*sampl.:* Gamut *eval.:* SPECs, Matrix, Stream
[IM03]	Pentium 4	15 HPCs	multiple linear	*eval.:* μ-benchs, AbiWord, Mozilla, Gnumeric
[RRK08]	Core 2 Duo & Xeon, Itanium 2, Turion	HW sensors HPCs	multiple linear	*sampl.:* calibration suite *eval.:* SPECs, stream, Nsort
[Yan+14]	Xeon E5620 & E7530	7 components 91 preselected	support vector	*sampl.:* NPB, IOzone, CacheBench *eval.:* SPEC CPU 06, IOzone
[Zha+14]	Sandy Bridge	non-halted cycles	linear	*eval.:* Google, SPEC CPU 06

Power models are mostly linear

Existing CPU Power Models

Ref.	Processor(s)	Feature(s)	Regression(s)	Benchmarks
[Ber+10]	Core 2 Duo	14 HPCs regrouped by component	multiple linear by component	*sampl.:* μ-benchs *eval.:* SPEC CPU 06
[Col+15]	Xeon W3520 & i3 2120	non-halted cycles reference cycles	polynomial	*sampl.:* stress *eval.:* PARSEC, SPECjbb
[CM05]	XScale PXA255	5 HPCs	multiple linear	*eval.:* SPEC CPU 00, Java CDC/CLDC
[Dol+15]	Xeon E3-1275	3 HPCs HW sensors	linear	*sampl.:* linpack, stream, iperf, IOR *eval.:* Quantum Espresso
[ERK06]	Turion, Itanium 2	HW sensors	multiple linear	*sampl.:* Gamut *eval.:* SPECs, Matrix, Stream
[IM03]	Pentium 4	15 HPCs	multiple linear	*eval.:* μ-benchs, AbiWord, Mozilla, Gnumeric
[RRK08]	Core 2 Duo & Xeon, Itanium 2, Turion	HW sensors HPCs	multiple linear	*sampl.:* calibration suite *eval.:* SPECs, stream, Nsort
[Yan+14]	Xeon E5620 & E7530	7 components 91 preselected	support vector	*sampl.:* NPB, IOzone, CacheBench *eval.:* SPEC CPU 06, IOzone
[Zha+14]	Sandy Bridge	non-halted cycles	linear	*eval.:* Google, SPEC CPU 06

Non free or private workloads

Challenges

1. Portability

Challenges

1. Portability
2. Accuracy

Challenges

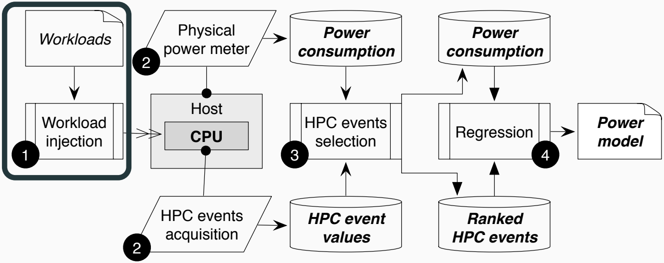
1. Portability
2. Accuracy
3. Reproducibility

Challenges

1. Portability
2. Accuracy
3. Reproducibility

Towards an automatic approach for learning CPU power models

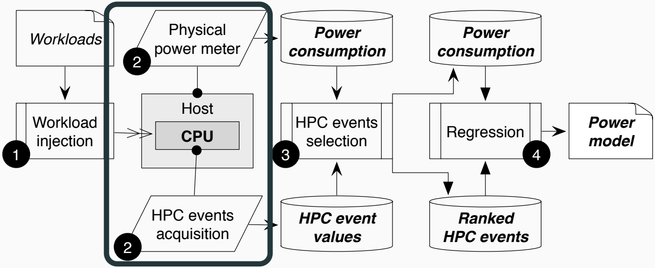
Our approach: Open-Testbed To Automatically Learn Power Models



1. Input workload injection
   - Configurable
   - PARSEC (open-source, multi-threaded)[4]
   - Run several applications (`x264`, `vips`, etc.)

[4] C. Bienia et al. "PARSEC 2.0: A New Benchmark Suite for Chip-Multiprocessors". In: *Proceedings of the 5th Annual Workshop on Modeling, Benchmarking and Simulation*. 2009.

Our approach:
Open-Testbed To Automatically Learn Power Models

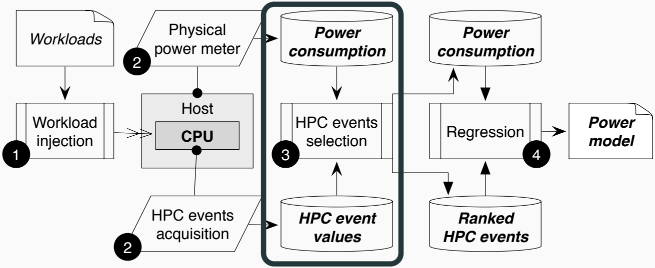


❷ Acquisition of raw input metrics

- Automatically explore the high number of the available HPCs (Xeon W3520: 514 HPCs)
- Take care of HPC multiplexing[5]

[5]Intel. *Intel 64 and IA-32 Architectures Software Developer's Manual.* 2015.

Our approach: Open-Testbed To Automatically Learn Power Models

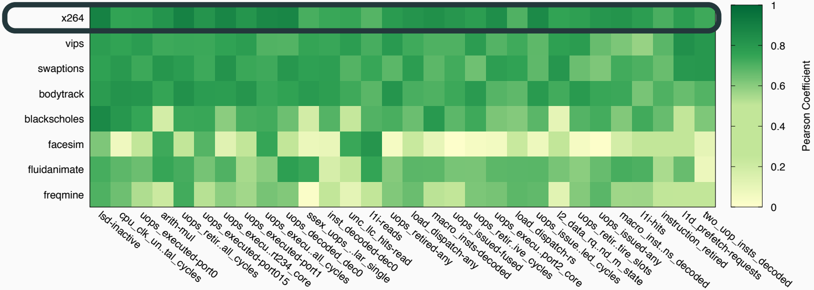


❸ Selection of relevant HPCs

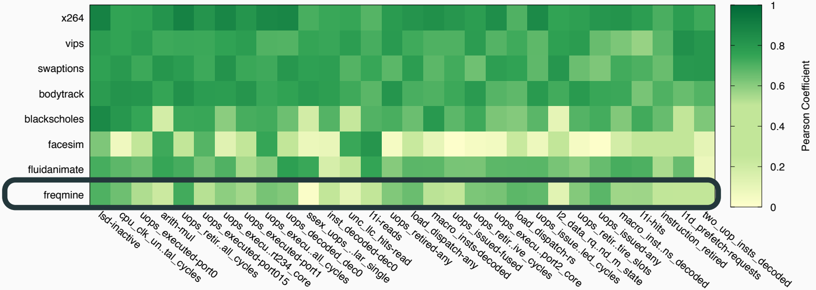
- Pearson coefficient (HPC ⇔ Power)
- 1^{st} phase: quickly filtering out uncorrelated HPCs (< 0.5) (Xeon W3250: 253 left out)
- 2^{nd} phase: full sampling for the remaining HPCs

Our approach: Open-Testbed To Automatically Learn Power Models

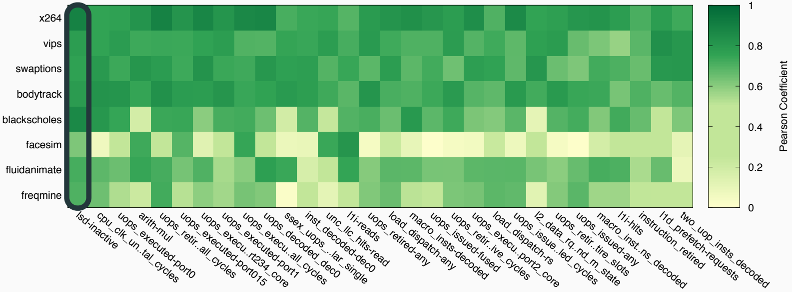
Pearson coefficients of the Top-30 correlated events for the PARSEC benchmarks on a Xeon W3520.



Pearson coefficients of the Top-30 correlated events for the PARSEC benchmarks on a Xeon W3520.

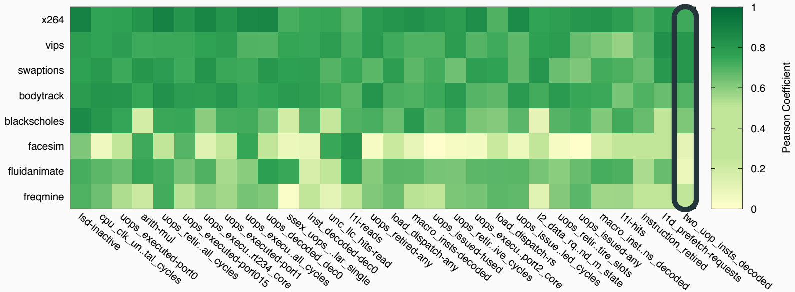


Pearson coefficients of the Top-30 correlated events for the PARSEC benchmarks on a Xeon W3520.

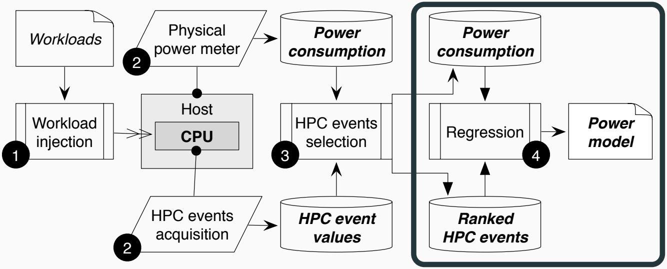


Our approach:
Open-Testbed To Automatically Learn Power Models

Pearson coefficients of the Top-30 correlated events for the PARSEC benchmarks on a Xeon W3520.



Our approach: Open-Testbed To Automatically Learn Power Models



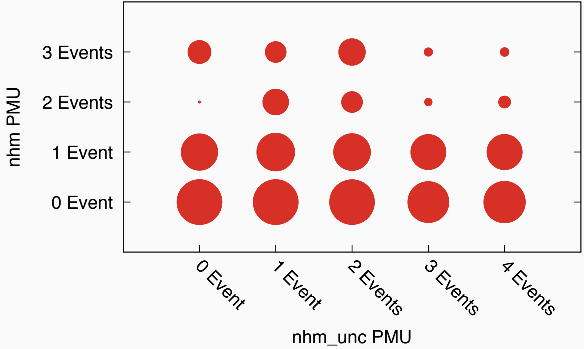
❹ Power model inference

- Minimize the number of HPCs
- Robust ridge regression (SotA?)

Our approach: Open-Testbed To Automatically Learn Power Models

Average error per combination of HPCs for `freqmine`, `fluidanimate`, `facesim` on a Xeon W3520.

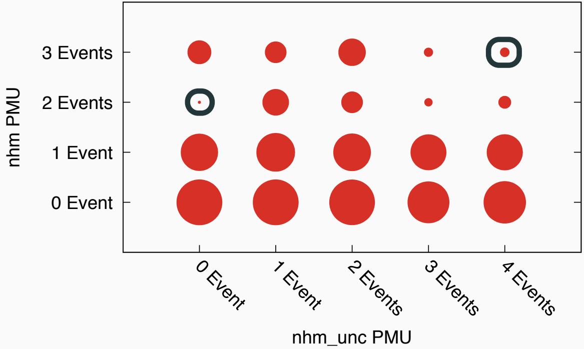
$$P_{idle} = 92\ W\ ;\ P_{CPU} = \frac{1.40 \cdot \text{HPC (l1i:reads)}}{10^8} + \frac{7.29 \cdot \text{HPC (lsd:inactive)}}{10^9}$$



Our approach:
Open-Testbed To Automatically Learn Power Models

Average error per combination of HPCs for `freqmine`, `fluidanimate`, `facesim` on a Xeon W3520.

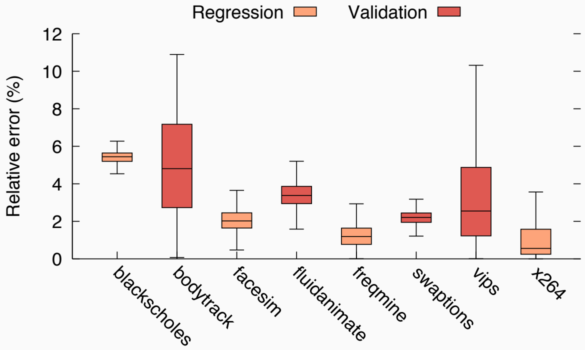
$$P_{idle} = 92\ W\ ;\ P_{CPU} = \frac{1.40 \cdot \mathrm{HPC}\ (\mathrm{l1i{:}reads})}{10^8} + \frac{7.29 \cdot \mathrm{HPC}\ (\mathrm{lsd{:}inactive})}{10^9}$$



Our approach: Open-Testbed To Automatically Learn Power Models

Relative errors for the PARSEC suite on the Cortex A15.

$$P_{idle} = 3.5\ W\ ;\ P_{CPU} = \frac{1.18 \cdot \text{cpu\_cycles}}{10^9} + \frac{1.26 \cdot \text{inst\_spec\_exec\_integer\_inst}}{10^{10}} + \frac{1.84 \cdot \mathit{bus\_cycles}}{10^{11}}$$



Summary

- Portability

  Beyond SotA: 4 CPUs ($2\times$Intel, 1 AMD, **1 ARM**)

Summary

- **Portability**

  Beyond SotA: 4 CPUs ($2\times$Intel, 1 AMD, **1 ARM**)

- **Accuracy**

  Avg. error on the 4 CPUs: 1.5%

Summary

- Portability

  Beyond SotA: 4 CPUs (2×Intel, 1 AMD, **1 ARM**)

- Accuracy

  Avg. error on the 4 CPUs: 1.5%

- Reproducibility

  Built on open-source workloads

Summary

- Portability

  Beyond SotA: 4 CPUs (2×Intel, 1 AMD, **1 ARM**)
- Accuracy

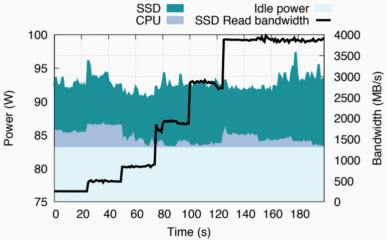
  Avg. error on the 4 CPUs: 1.5%
- Reproducibility

  Built on open-source workloads
- Extensibility

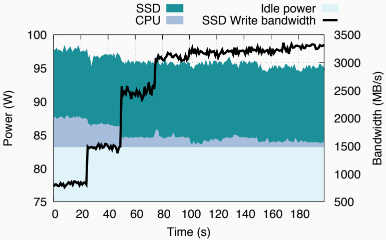
  Can we extend our learning approach to SSD power models?

Comparison of power consumptions between CPU and SSD by varying the throughput with the `fio` tool.

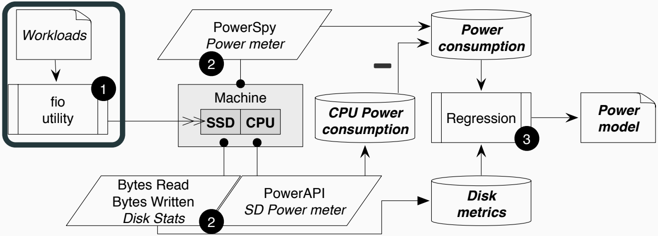
(a) SSD read operations.



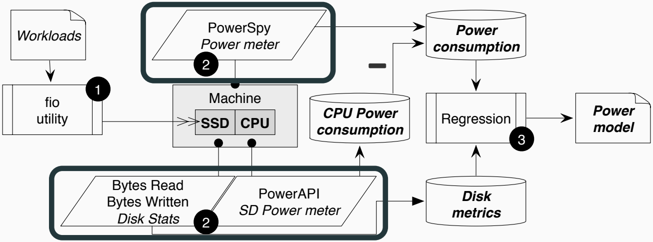
(b) SSD write operations.



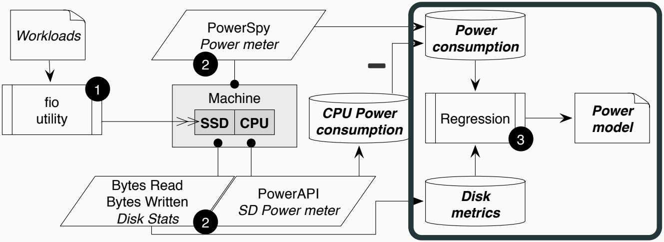
Our approach: Open-Testbed To Automatically Learn Power Models (2)



Our approach: Open-Testbed To Automatically Learn Power Models (2)

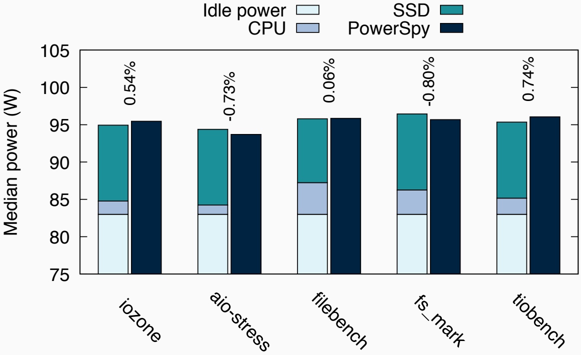


Our approach: Open-Testbed To Automatically Learn Power Models (2)

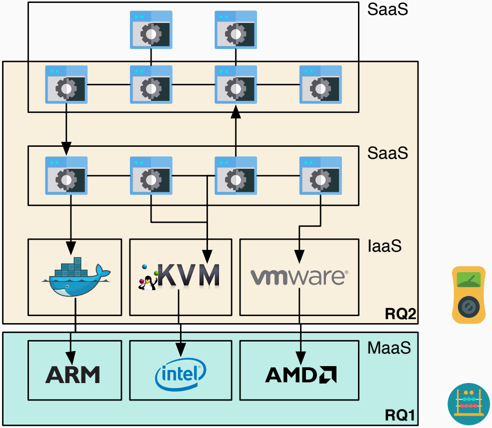


Our approach: Open-Testbed To Automatically Learn Power Models (2)

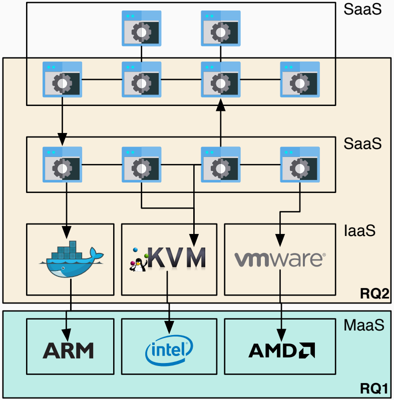
Power consumption of the host for 5 workloads on a Xeon E5-2630.

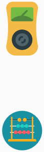


RQ2: Can we propose a uniform view of the service power consumption?



RQ2: Can we propose a uniform view of the service power consumption?

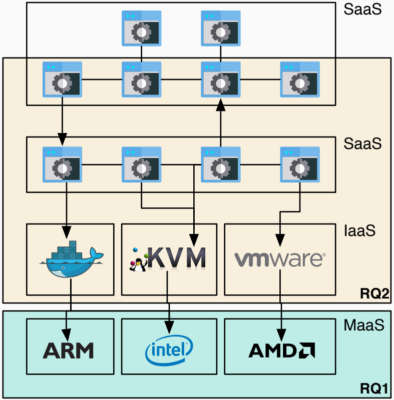




Challenges

1. Native
2. Virtualized
3. Distributed

RQ2: Can we propose a uniform view of the service power consumption?



Challenges

1. **Native**
2. Virtualized
3. Distributed

PowerAPI: a Toolkit for Building SD Power Meters

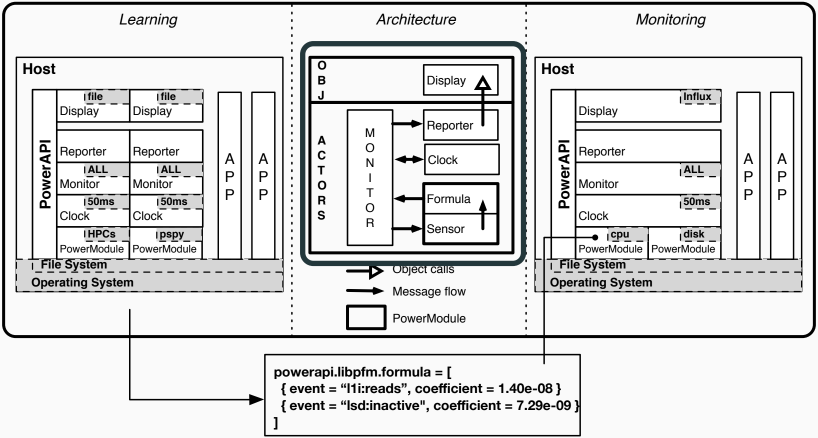
- Code freely available on GitHub: `http://powerapi.org`
  - Scala / Akka
  - LoC: 8.7k
  - Docker
  - AGPLv3

- Code freely available on GitHub: `http://powerapi.org`
    - Scala / Akka
    - LoC: 8.7k
    - Docker
    - AGPLv3

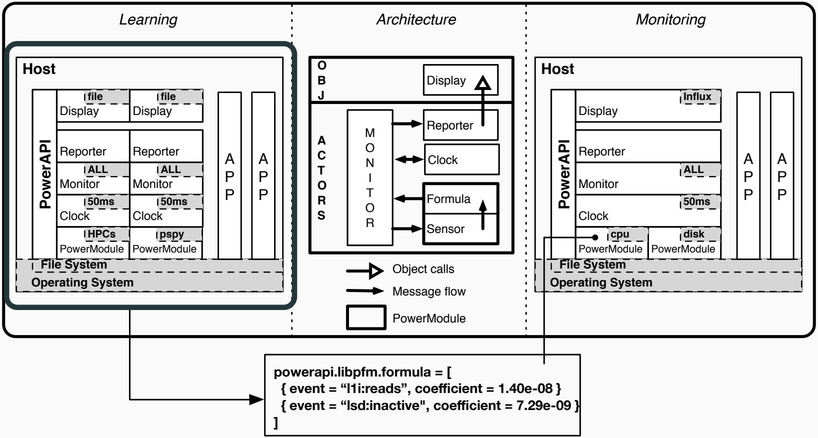
- 2^{nd} major iteration[6]
    - Full support of multi-core CPU architectures (HT, DVFS, TB)
    - Learning techniques
    - Better support of Akka

[6] A. Noureddine. "Towards a Better Understanding of the Energy Consumption of Software Systems". PhD thesis. Université des Sciences et Technologie de Lille - Lille I, 2014.

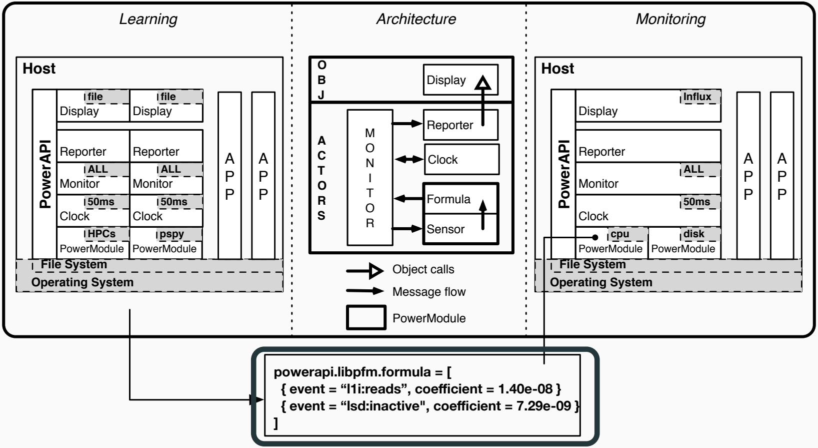
PowerAPI's architecture & deployment.



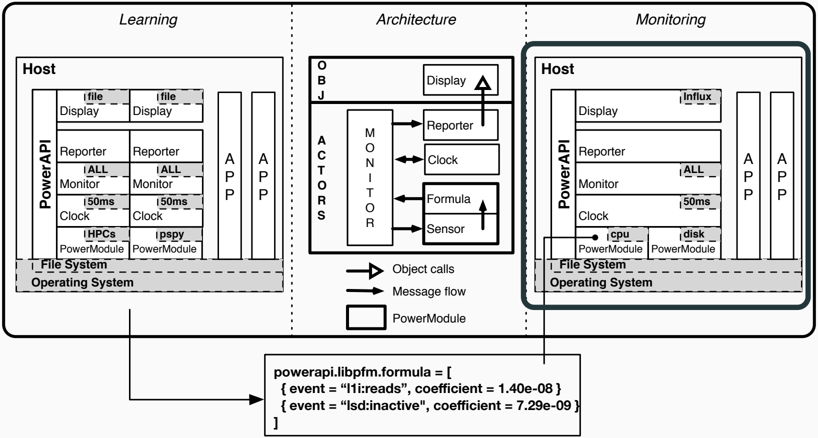
POWERAPI's architecture & deployment.



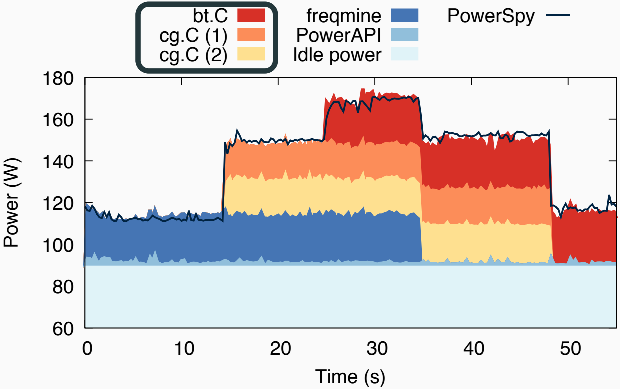
POWERAPI's architecture & deployment.



POWERAPI's architecture & deployment.

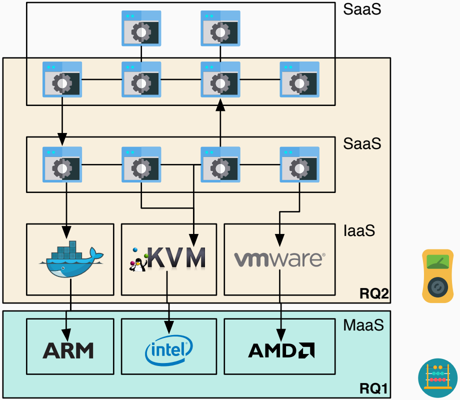


SD Power Meter For Monitoring Concurrent Apps



- On the Intel Xeon W3520
  - Monitoring freq.: 4Hz
  - Avg. error: 2%
  - Low overhead: 2 W

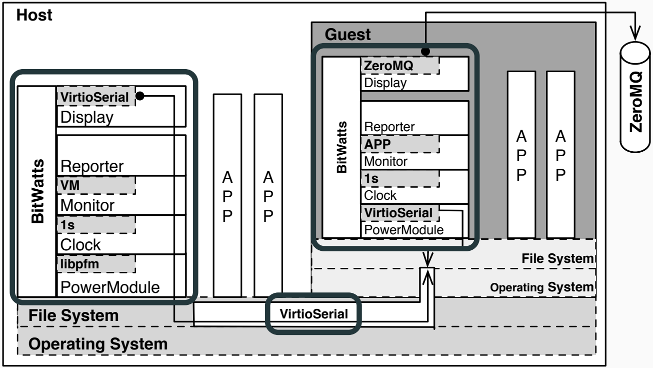
RQ2: Can we propose a uniform view of the service power consumption?



Challenges

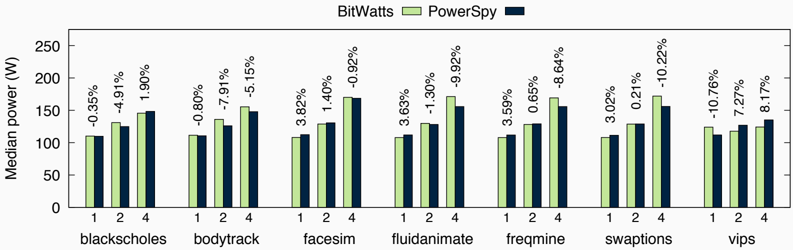
1. Native
2. **Virtualized**
3. Distributed

BitWatts Architecture



Evaluation

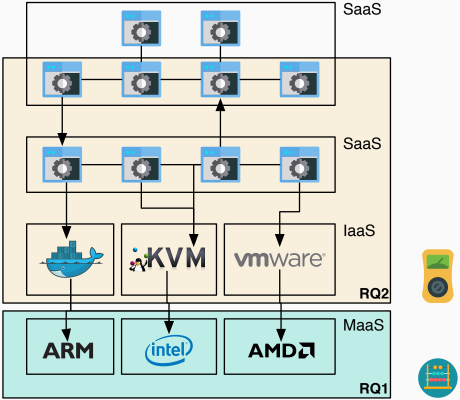
Scaling PARSEC on multiple VMs on a Xeon W3520.



- Errors: from 1% (`fluidanimate`) up to 10% (`swaptions`)
- Beyond SotA [Ber+12]:[7] VM as a White-Box (+ multi-tenant)

[7] R. Bertran et al. “Energy Accounting for Shared Virtualized Environments Under DVFS Using PMC-based Power Models”. In: *Future Generation Computer Systems* (2012).

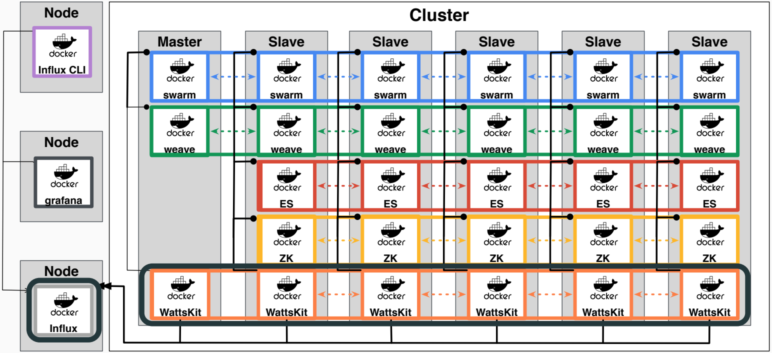
RQ2: Can we propose a uniform view of the service power consumption?



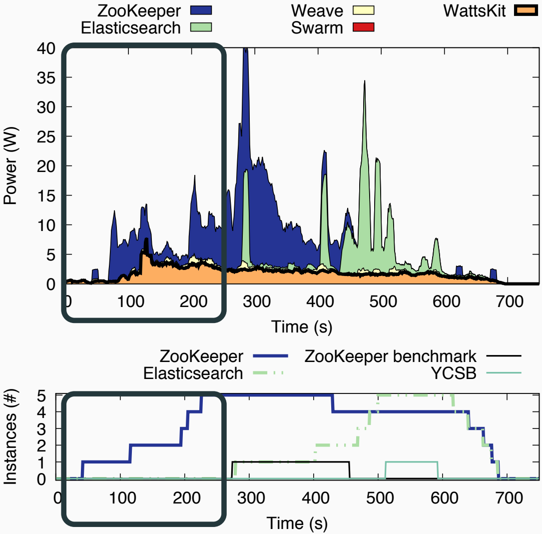
Challenges

1. Native
2. Virtualized
3. **Distributed**

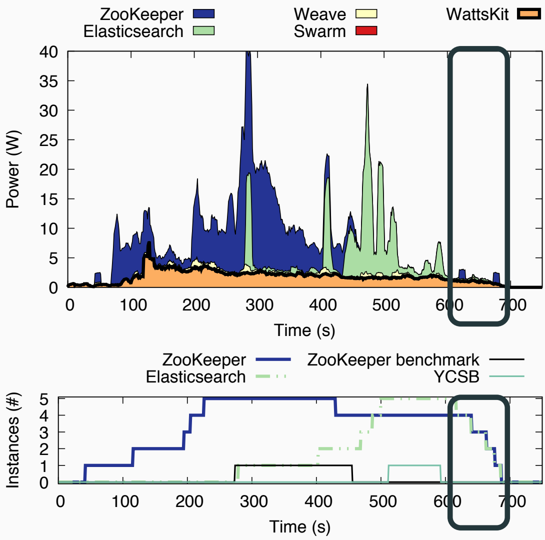
A Service-Level Power Monitoring

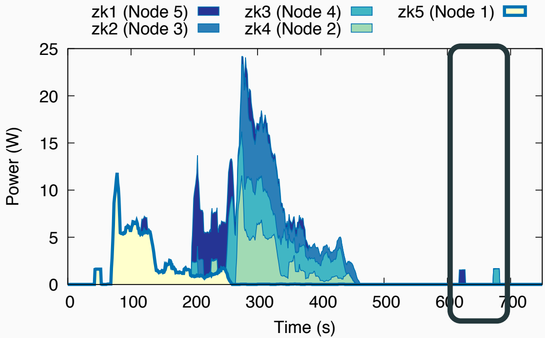


A Service-Level Power Monitoring



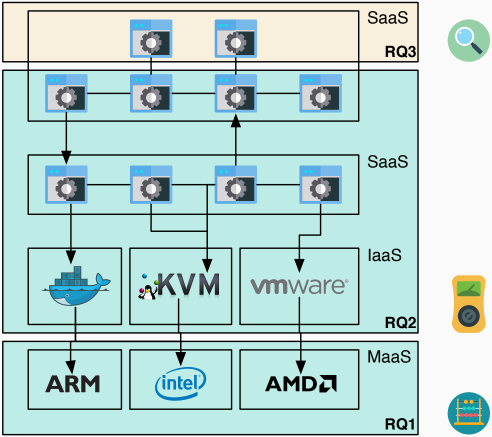
A Service-Level Power Monitoring

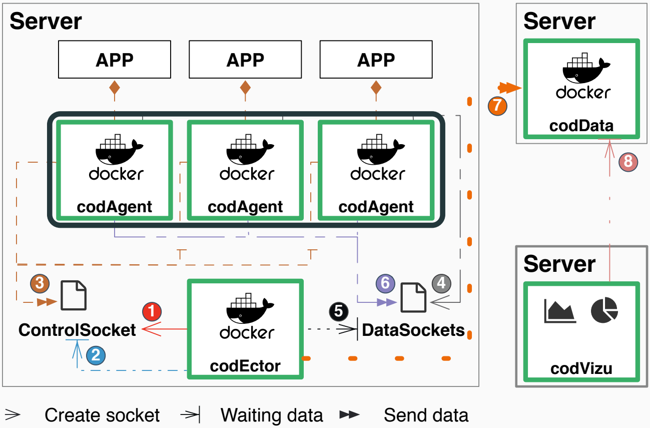



zk1 (Node 5)
zk2 (Node 3)
zk3 (Node 4)
zk4 (Node 2)
zk5 (Node 1)
Power (W)
Time (s)
0
5
10
15
20
25
0
100
200
300
400
500
600
700

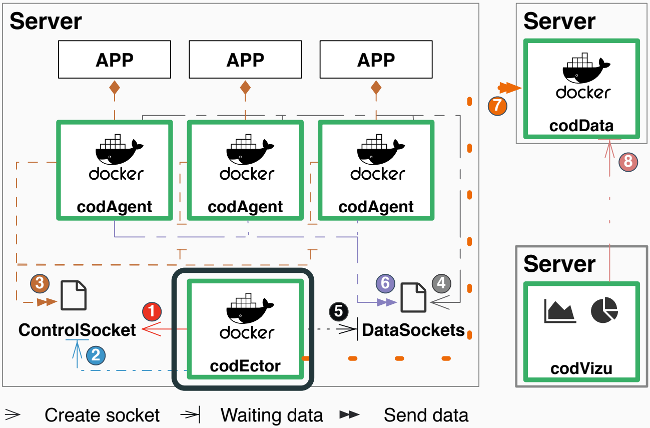

ZooKeeper
Elasticsearch
ZooKeeper benchmark
YCSB
Instances (#)
Time (s)
0
1
2
3
4
5
0
100
200
300
400
500
600
700

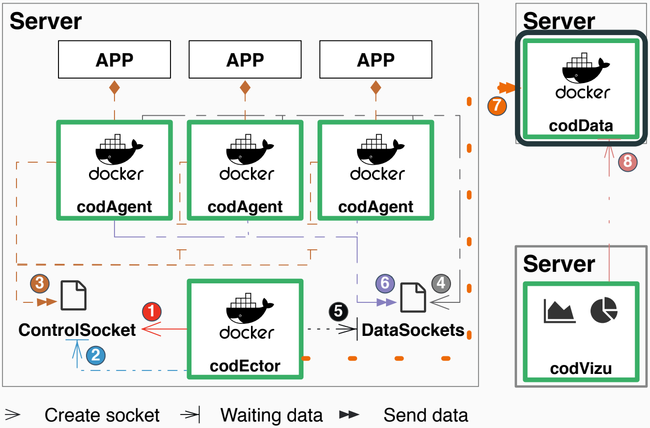
RQ3: Can we analyze the power consumption of the artifacts which compose a software?

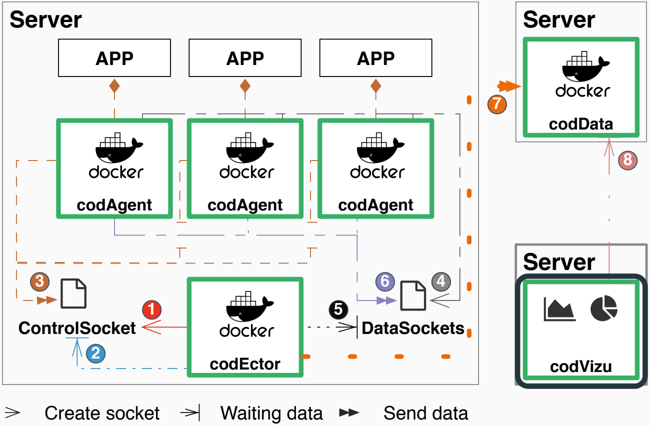



Server
APP
APP
APP
docker
codAgent
docker
codAgent
docker
codAgent
3
ControlSocket
1
2
docker
codEctor
5
DataSockets
6
4
7
Server
docker
codData
8
Server
codVizu
Create socket
Waiting data
Send data

Overview of the codEnergy Approach

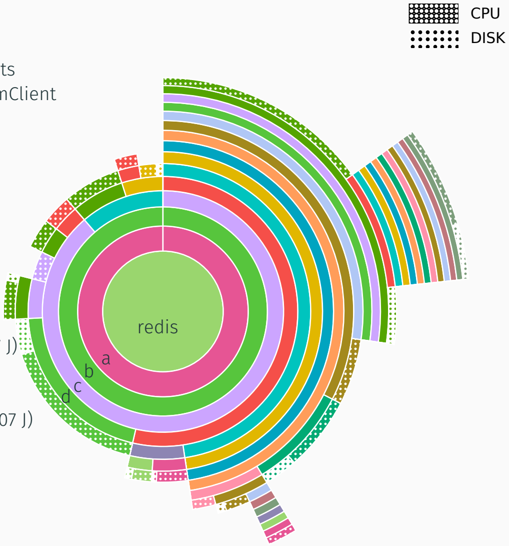



Server
APP
APP
APP
docker
codAgent
docker
codAgent
docker
codAgent
3
6
4
ControlSocket
1
docker
codEctor
5
DataSockets
2
Server
7
docker
codData
8
Server
codVizu
Create socket
Waiting data
Send data


Server
APP
APP
APP
docker
codAgent
docker
codAgent
docker
codAgent
3
ControlSocket
1
2
docker
codEctor
5
6
4
DataSockets
7
Server
docker
codData
8
Server
codVizu
Create socket
Waiting data
Send data

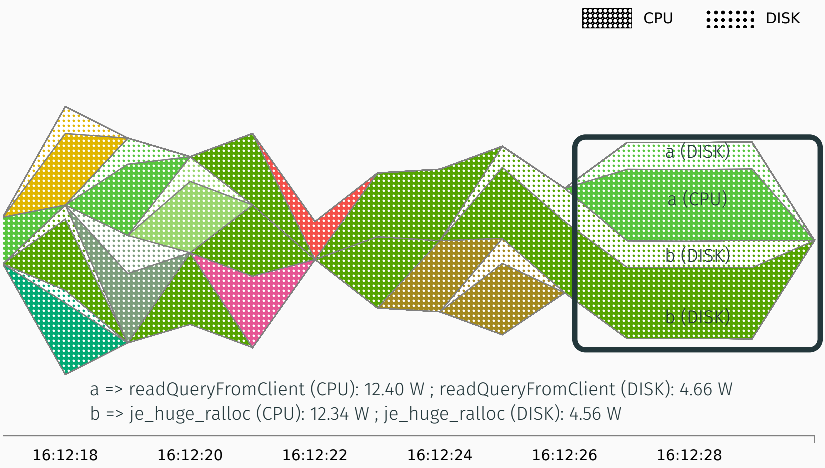
codVizu: Sunburst (1)

a: main
b: aeMain
c: aeProcessEvents
d: readQueryFromClient

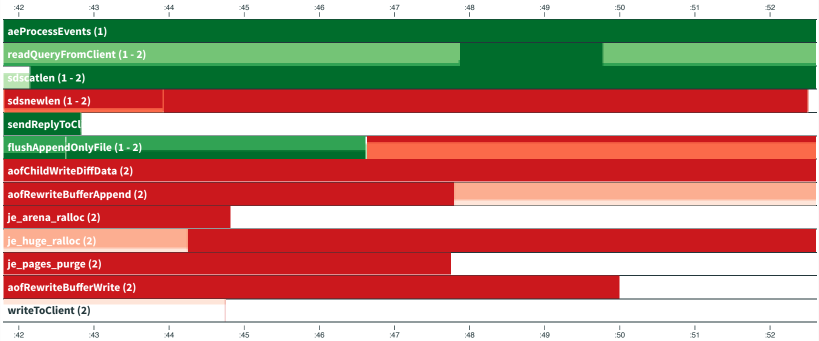


d (DISK)
25.44 J (3.88% of 656.07 J)

d (CPU)
109.41 J (16.68% of 656.07 J)

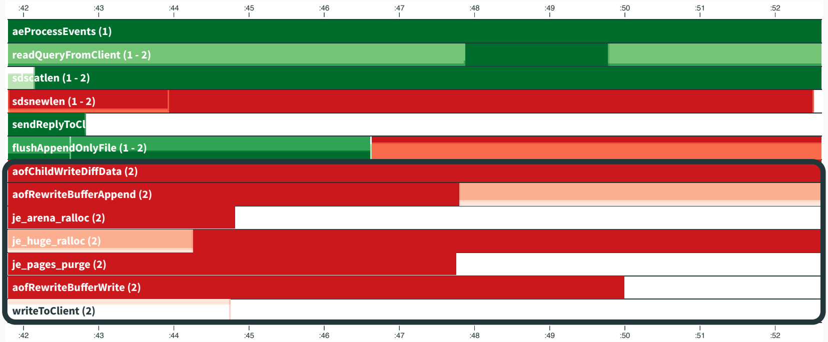

CPU
DISK
a (DISK)
a (CPU)
b (DISK)
b (DISK)
a => readQueryFromClient (CPU): 12.40 W ; readQueryFromClient (DISK): 4.66 W
b => je_huge_ralloc (CPU): 12.34 W ; je_huge_ralloc (DISK): 4.56 W
16:12:18
16:12:20
16:12:22
16:12:24
16:12:26
16:12:28

`redis 2` (1) vs `redis 3` (2).



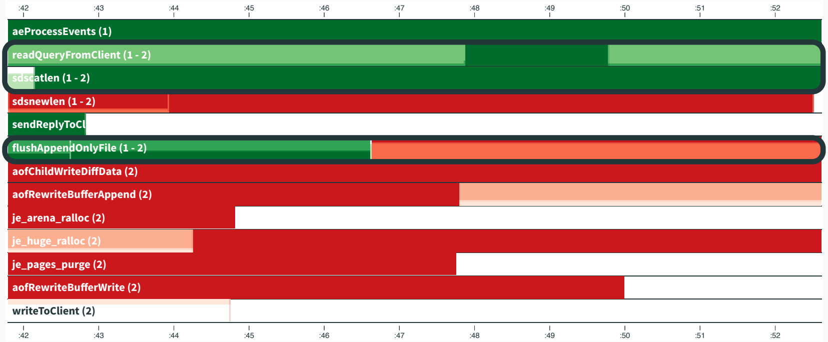
Energy Difference Between Software Systems

`redis 2` (1) vs `redis 3` (2).

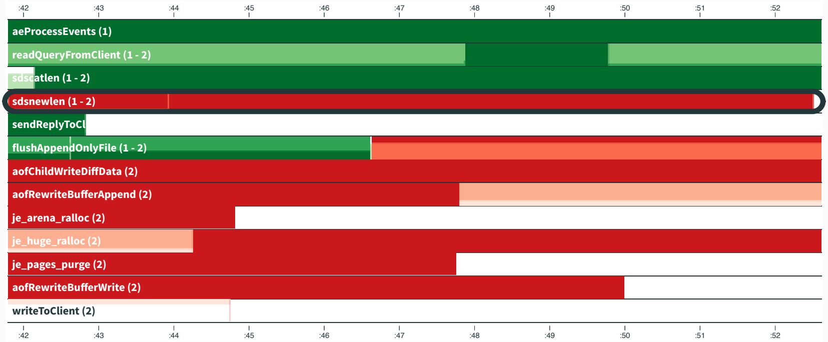


Energy Difference Between Software Systems

redis 2 (1) vs redis 3 (2).



`redis 2` (1) vs `redis 3` (2).



Conclusion & Perspectives

Contributions

Multi-dimensional analysis of software power consumptions on multi-core architectures

Multi-dimensional analysis of software power consumptions on multi-core architectures

- **RQ1**: Can we model the software power consumption regardless of the underlying architecture?

  Open-testbed approach for learning multi-core power models

Multi-dimensional analysis of software power consumptions on multi-core architectures

- RQ1: Can we model the software power consumption regardless of the underlying architecture?

  Open-testbed approach for learning multi-core power models

- RQ2: Can we propose a uniform view of the service power consumption?

  In width energy monitoring with PowerAPI, BitWatts & WattsKit

Contributions

Multi-dimensional analysis of software power consumptions on multi-core architectures

- **RQ1**: Can we model the software power consumption regardless of the underlying architecture?

  Open-testbed approach for learning multi-core power models

- **RQ2**: Can we propose a uniform view of the service power consumption?

  In width energy monitoring with PowerAPI, BitWatts & WattsKit

- **RQ3**: Can we analyze the power consumption of the artifacts which compose a software?

  In depth energy monitoring with codEnergy

Short-Term Perspectives

- Define a new scheduler for saving energy in cloud data centers
- Continuous optimization of the power models in a cluster
- Turning-off nodes of a cluster during inactivity periods
- Leveraging source-code energy monitoring
- Extend codEnergy to other programming languages

Short-Term Perspectives

- Define a new scheduler for saving energy in cloud data centers
- **Continuous optimization of the power models in a cluster**
- Turning-off nodes of a cluster during inactivity periods
- Leveraging source-code energy monitoring
- Extend codEnergy to other programming languages

Short-Term Perspectives

- Define a new scheduler for saving energy in cloud data centers
- Continuous optimization of the power models in a cluster
- Turning-off nodes of a cluster during inactivity periods
- Leveraging source-code energy monitoring
- Extend codEnergy to other programming languages

Long-Term Perspectives

- The power rising of GPU cards
- Proposing a wider energy cartography of a system
- Using genetic programming to improve the energy-efficiency at source-code level
- Defining solutions to automatically optimize the software energy-efficiency

Long-Term Perspectives

- **The power rising of GPU cards**
- Proposing a wider energy cartography of a system
- Using genetic programming to improve the energy-efficiency at source-code level
- Defining solutions to automatically optimize the software energy-efficiency

Long-Term Perspectives

- The power rising of GPU cards
- Proposing a wider energy cartography of a system
- **Using genetic programming to improve the energy-efficiency at source-code level**
- Defining solutions to automatically optimize the software energy-efficiency

Thanks for your attention.

Conferences

[Col+15] M. Colmant et al. “Process-level Power Estimation in VM-based Systems”. In: *Proceedings of the 10th European Conference on Computer Systems (EuroSys)*. 2015.

[CRS14] M. Colmant, R. Rouvoy, and L. Seinturier. “Improving the Energy Efficiency of Software Systems for Multi-Core Architectures”. In: *Middleware 2014 Doctoral Symposium*. 2014.

[CRS15] M. Colmant, R. Rouvoy, and L. Seinturier. “Estimation de la consommation des systèmes logiciels sur des architectures multi-coeurs”. In: *Conférence d’informatique en Parallélisme, Architecture et Système (Compas)*. 2015.

[Hav+ar] A. Havet et al. “GENPACK: A Generational Scheduler for Cloud Data Centers”. In: *IEEE International Conference on Cloud Engineering (IC2E)*. 2017. (To appear).

Under Evaluation

[Col+16] M. Colmant et al. “The Next 700 CPU Power Models”. In: *ACM Trans. Model. Perform. Eval. Comput. Syst. (ACM TOMPECS)* (2016).

References I

[Ber+10] R. Bertran et al. “Decomposable and Responsive Power Models for Multicore Processors Using Performance Counters”. In: *Proceedings of the 24th ACM International Conference on Supercomputing*. 2010.

[Ber+12] R. Bertran et al. “Energy Accounting for Shared Virtualized Environments Under DVFS Using PMC-based Power Models”. In: *Future Generation Computer Systems* (2012).

[BL09] C. Bienia and K. Li. “PARSEC 2.0: A New Benchmark Suite for Chip-Multiprocessors”. In: *Proceedings of the 5th Annual Workshop on Modeling, Benchmarking and Simulation*. 2009.

[CM05] G. Contreras and M. Martonosi. “Power Prediction for Intel XScale® Processors Using Performance Monitoring Unit Events”. In: *Proceedings of the International Symposium on Low Power Electronics and Design*. 2005.

[Col+15] M. Colmant et al. “Process-level Power Estimation in VM-based Systems”. In: *Proceedings of the 10th European Conference on Computer Systems (EuroSys)*. 2015.

[Col+16] M. Colmant et al. “The Next 700 CPU Power Models”. In: *ACM Trans. Model. Perform. Eval. Comput. Syst. (ACM TOMPECS)* (2016).

[Col+17] M. Colmant et al. “WattsKit: Software-Defined Power Monitoring of Distributed Systems”. In: *To be chosen*. 2017.

[CRS14] M. Colmant, R. Rouvoy, and L. Seinturier. “Improving the Energy Efficiency of Software Systems for Multi-Core Architectures”. In: *Middleware 2014 Doctoral Symposium*. 2014.

[CRS15] M. Colmant, R. Rouvoy, and L. Seinturier. “Estimation de la consommation des systèmes logiciels sur des architectures multi-coeurs”. In: *Conférence d’informatique en Parallélisme, Architecture et Système (Compas)*. 2015.

[CRS17] M. Colmant, R. Rouvoy, and L. Seinturier. “codEnergy: an Approach For Leveraging Source-Code Level Energy Analysis”. In: *To be chosen*. 2017.

[Dol+15] M. F. Dolz et al. “An analytical methodology to derive power models based on hardware and software metrics”. In: *Computer Science - Research and Development* (2015).

References II

[ERK06] D. Economou, S. Rivoire, and C. Kozyrakis. "Full-System Power Analysis and Modeling for Server Environments". In: *In Workshop on Modeling Benchmarking and Simulation*. 2006.

[Hav+ar] A. Havet et al. "GENPACK: A Generational Scheduler for Cloud Data Centers". In: *IEEE International Conference on Cloud Engineering (IC2E)*. 2017. (To appear).

[IM03] C. Isci and M. Martonosi. "Runtime Power Monitoring in High-End Processors: Methodology and Empirical Data". In: *Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture*. 2003.

[Int15] Intel. *Intel 64 and IA-32 Architectures Software Developer's Manual*. 2015. URL: `http://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-vol-1-manual.pdf` (visited on 08/01/2016).

[Kur+14] M. Kurpicz et al. *How energy-efficient is your cloud app?* Conférence d'informatique en Parallélisme, Architecture et Système (Compas) - Poster session. 2014.

[Nou14] A. Noureddine. "Towards a Better Understanding of the Energy Consumption of Software Systems". PhD thesis. Université des Sciences et Technologie de Lille - Lille I, 2014.

[RRK08] S. Rivoire, P. Ranganathan, and C. Kozyrakis. "A Comparison of High-level Full-system Power Models". In: *Proceedings of the Conference on Power Aware Computing and Systems*. 2008.

[The08] The Climate Group. *SMART 2020: Enabling the low carbon economy in the information age*. 2008. URL: `http://gesi.org/article/43` (visited on 09/23/2016).

[Yan+14] H. Yang et al. "iMeter: An integrated VM power model based on performance profiling". In: *Future Generation Computer Systems* (2014).

[Zha+14] Y. Zhai et al. "HaPPy: Hyperthread-aware Power Profiling Dynamically". In: *Proceedings of the USENIX Annual Technical Conference*. 2014.