Green Coding

Software energy optimization by understanding the impact of programming languages

Mohammed Chakib Belgaid

Supervisors: Pr. Romain Rouvoy
Pr. Lionel Seinturier

University of Lille

This dissertation is submitted for the degree of
Doctor of Philosophy

EDSPI June 2022

I would like to dedicate this thesis to my loving parents ...

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Mohammed Chakib Belgaid
June 2022

Acknowledgements

And I would like to acknowledge ...

Abstract

This is where you write your abstract …

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowdays, computers are invasing our daily life from work to leasure, from fancy smartphone to an embded peacemaker that regulatres the heartbeat of some people. As human beigns we are known to use tools to enhance our bodies. And thanks to computers we pushed that step even further where now we are using machines to extend our brains. Some even argue that one day they will replace us, and therefore we created our own ending. Well this is the problem for the future generations. For the moment, the main concern of humanity is to keep this planet livebale until we find another alternative.

Unfortunately those machines doesn't help that much. as according to ....

TODO : add the impact of the ict ..etc

Researchers are trying to solve this issue through different angles. While some scientists are trying to find an alternative green source to generate energy. others focus on reducing it'sconsumption. In computer science we are concerned with the ladder solution Therefore most of the works are done on tunning the software and the hardware in ordrer to have a more efficient way use this energy. In this thesis we are trying to find a way to make the energy consumption of the computer to be as low as possible. Our approach is to reduce the energy consumption of the software services by changing certains parameters, such as palteform, programming langauge, and/or the design pattern.

The best way to do so is to formulate a theory behind the energy consumption of algorthms such as the complexity and the o notation. Unfortunately this is not possible in the current state of the art. due to the lack of knowledge about the energy consumption of the algorithms, and the strong correlation between this consumption and the hardware configuration.

Unlike algrothim optimization in the field of perfornacne which is agnostic toward the plateform. the energy consumption of the algorithms is dependent on execution environement. Therefore, for the moment we will start by formulating some hypthosis and explore them using empirical analysis.

Our work will be presented through the following chapters :

1. 1.3: Where we discuss the work done on the energy consumption and optimization in software engineering
2. **??**: It will present a set of guidelines and tools to help practitioners measure the energy consumption of their algorithms.
3. : we will present the impact of programming languages on the energy consumption of the algorithms.
4. : it will discuss the behaviour of python and the possibile ways to tune it in ordrer to reduce the energy consumpion
5. : will present a study on java programming language and the impact of the JVM choice on the energy consumpion
6. : as a prespective we introduce the impact of parallelisme on the enrgy consumption on time agnostic cases

....

1.1 Motivation

....

1.2 Research Contributions

1. Introduces
2. Shows how ....
3. Proposes ...

1.3 Publications

1. ...
2. ...
3. ...

Chapter 2

Literature Review

Efficiency in energy usage is a well-known topic. In the majority of fields, the purpose is to minimize the energy consumption of electrical devices/components. Modern times even see energy classification (A, B,... F) for Homes, cars and electronic products, to provide the consumer an indication of the energy consumption of his devices, which will reflect on his power bill. This creterions is extended even to the hardware components of computer. figure2.1 [1]

In computer science, the objective is essentially same. Numerous studies have been conducted on energy optimization. Some of these studies concentrate on minimizing energy consumption at the hardware level, while others optimize energy consumption via software.

As an example **?** evaluated the development of power use effectiveness (PUE) in data centers thkat belongs to various organizations participating in the European code of conduct for energy efficiency program [**?**]. The research found a gradual decline in the PUE of data centers, which measures the ratio of the overall energy supplied to the energy used by IT equipment. A low PUE implies that the majority of energy is utilized to power the data center's IT equipment, while just a little amount is needed for cooling and lighting.

we will place a greater emphasis on the software level in order to decrease the amount of energy that is used.more particularly on the execution phase of the program cycle. We will be proceeding through an empirical analysis of the energy consumption of the software while changing some components of the source code without impacting its behavior. In order to do this, we will elaborate a benchmarking process and a set of tools that are intended to assist practitioners in better comprehending and optimizing the energy usage of their applications. Thus, we will begin by examining the state of the art of empirical analysis and retrieving the best empirical experimentation methodologies in the research field.Then, we

[1] https://www.cpubenchmark.net/compare/Intel-i9-12900KS-vs-Intel-i9-12900KF/4813vs4611

	Intel Core i9-12900KS	Intel Core i9-12900KF
Max TDP	150W	241W
Power consumption per day (kWh)	0.3	0.5
Running cost per day	$0.075	$0.120
Power consumption per year (kWh)	109.5	175.9
Running cost per year	$27.38	$43.98

Shown CPU power usage is based on linear interpolation of Max TDP (i.e. max load). Actual CPU power profile may vary.

Figure 2.1: electrical cost comparasion between two cpus

will narrow these practices down to computer science so that we can finally adapt them to energy consumption.

Section 2.1 will discuss the pitfalls and best practices associated with empirical research before applying them to our field of interest. After that Section 2.3 describes software energy measurements. It provides examples of hardware and software measuring instruments and describes their differences, benefits, and drawbacks. It also examines the sources of energy measurement variations, which represent a significant obstacle to achieving precise readings and a higher accuracy. Then, in section 2.4, we will go through some of the previous work on improving the energy consumption of softwares.

2.1 benchmarking

definition :benchmark: [noun] something that serves as a standard by which others may be measured or judged. that serves as a basis for evaluation or comparison (as of computer system performance) this is [5]

difference tests [104] [69] [15]

2.2 introduction

list of benchmarks [106]:

- Not evaluating potential performance degradation
- Benchmark subsetting without proper justification

- Selective data set hiding deficiencies
- Microbenchmarks representing overall performance
- Throughput degraded by x% => overhead is %
- Creative overhead accounting
- No indication of significance of data
- Incorrect averaging across benchmark scores
- Benchmarking of simplified simulated system
- Inappropriate and misleading benchmarks
- Same dataset for calibration and validation
- No proper baseline
- Only evaluate against yourself
- Unfair benchmarking of competitors
- Not all contributions evaluated
- Only measure runtime overhead
- False positives/negatives not tested
- Elements of solution not tested incrementally
- Missing platform specification
- Missing software versions
- Subbenchmarks not listed
- Relative numbers only

[86]

startup performance vs steady performance [Buytaert and Eeckhout]
simgrid simulation of power consumption of parallel aplication using simgrid [52]
impact of manifacturing process on the energy variation [27]
different energy consumption between identical processers [idle and hight load][108]

the temperature is the main raison a high energy variation [110]
external factors are impact on the energy variation [84]
the place of the server in the room doesn't affect the energy variation [36]
comparaison of three measuring tools and they did exhebit 10% variation each time [57]
low-cost scalable variation-aware algorithm [57]
reducing the energy variation by disabling turbo boost [2]
reducing the energy consumption in parallel systems [23]
[77]

2.3 energy measurement

to be able to reduce the energy cost of running programs we should first be able to estame this energy consumption. many studies have been conducted to estimate a such energy consumption. that varries from static analysis of the source code to enfere it's energy consumption like Pereira et al. where they provided a tool to highlight the most energy consuming parts of the code [95].the main advantage of this apporach is to be able to estimate the energy consumption of a program without running it. however. unlike the conplexity of programs, the energy consumption is highly related the execusion environment. Therefor, Static analysis might not represent the real behaviour of the same program when it is run in the production environment. To treat the problem of representativeness, many researchers tend to measure the energy consumption of the programs while they are running them. Hence we will get more accurate results. there is a variety of tools to measure such energy, and they cover a large spectrium of usage depends on how **accurate** and **pricise** those results are needed to be in tone hand, and the price that practinoners are ready to pay for a such accuray/pricision. Below we will present some tools that are well known in the literature and discuss their main advantages / drawbacks for our case.

2.3.1 hardware tools

according to Hackenberg et al. there are four main creterions to evaluate an energy measurement tool [47]

- *spatial granularity: the more specific the target of monitoring we are able to measure, the more efficient we can do optimization since we will know what causes the pitfalls of the energy consumption.*
- temporal granularity: same as spatial granularity, temporal granularity helps us to identif the sequence of code that need to be optimized.

- *scalability: this is mainly related to the cost of the tools and the ease of their integration for our system.*

- accuracy:to elimanate extra hazards and get more representative measurement.

We believe that accuracy can be extracted from those creterions since it is a result of the combaination of the two first ones. therefore, we will focus more on the first 3 creterions from later on. In the follwong section we will discuss some of the well known tools that are used in literature.

Nowdays, most of the high performance computing systems (HPC) implements a tool to report the energy consumption of the nodes for the sake of monitoring and administration. Those tools are mainly integrated withing the power suply units (PSU) or the power distributuin units (PDU). then , they provide an interface and a log to follow the history of the energy consumption. Despite their scalability and ease of integration. such tools lack both in spatial and temporal granularity since they monitor the whole energy of the nodes, and most of the times they have a very low sampling frequency. most of those tools are provided directly by the manufacturers. such as *IBM EnergyScale technology [79] [19] [Caldeira et al.] or* Dell poweredge [72], MEGware Clusttstafe [13] As we said earlier the true purpose of those tools is more monitoring than analysing the energy consumption. WattsUp Pro, is a device that can be installed between the powerer source of the machine and the system under test. it allows a sampling frequency up to 1Hz and has a internal memory to store a wide variety of data, such as the maximum voltage, current.. etc that later can be exported via USB port for personal usage or lined to some graph programs like Logger Pro or LabQuest.The main advantage of this tool is the ability to monitor the energy consumption from a different device which will reduce the risk of interference with the energy consumption of the test [54]

despite his high temporal granularity , wattsup pro lacks in term of spatial granularity since it monitors the energy consumption of the whole system. To have a finer granularity we need to isolate the energy consumption of each component. . For this we will need more intrusive tools such.

PowerMon and its upgradded version powerMon2 [10] are based on an microcontroler chip that can monitor up to 6 channel (8 for powermon2) simultanetly. therefore we are able to monitor the powerconsumption of 4 devices at the same time. the frequency sample of this tool is up to 50Hz with an accuracy of 1.2%. Powermore2 comes with smaller size that can fit within 3.5 inches rack drive.

PowerInsight [66] is another finegrained measurement tool that is based on an ARM bagelbone processor [28] which is able to measure up to 30 channel simultanetly with a frequency of 1KHz per channel.

on the other hand GreenMiner [53],

powerpack [43] in the other hand is an api that synchronizes the data gathered from other monitoring tools such as Watt's Up Pro, NI and RadioShack pro and the lines of code

other monitor tools have been relized by the manufactures such as IBM Power executive [62] which allows their customer to monitor the power consumption and thermal behaviour of the of BladeCenter systems in the datacenter.

as we see in The CPU is the part responsible of the most energy consumption of softwares. hence the finner we go to measure this energy consumption the better it is for our work. Fortunately, Intel and later AMD proposed a tool that estimate the power consumption of different part of the CPI based on counter performances. RAPL (RUnning Average Power Limit) [46] [48] is a set of registers that was introduced by intel in their CPU since Sandy bridge generation, and later it was fllowed by AMD since Family 17h Zen. It is capable of monitoring different components of CPU , such as DRAM, Cores , and the integrated GPU for desktop processors.

The advanage of a such apporach is that the absence of any intrusive measurement tools. further more they have a high temporal granurality with a sampling frequency up to 1KHz [56].

with similar approach we can find NVIDIA reporting tools such as GPU TESLA [16] and the NVML library [38]

2.3.2 software tools

Now that we got our hands on a precise tools that allows us to monitor

powerapi [29]

smartwatts formula [40]

Wattwatcher [67]

joulemeter [63][60]

jrapl [45] [70]

joulinar [59] [89]

jalen [90]

selfwatts [41]

powerjoular [88]

2.3.3 hybrid tools

2.4 programming languages and performances

2.4.1 energy

energy comparaison of programming languages in the game benchmark by debian [97]
toward green ranking [31]
java vs kotlin in web performances [14]
rosetta code [87] [80]
network energy comparaison [7]
Aequitas [101]
vm placement [82]
Mishra et al.
static cost of Vms [64]
carbon footprint of training neural network [105]
SPELL, the energy leaks detector tool [96]
impact of energy profiling on software [60]
impact of maintability on software energy evolution [20]
defintion [109]
comparaison [22]
emprical study [34]
impact of website implemenation on energy consumption [98] [75]
PHP [12] [33]
simulation of web cache [21]
java spring analysis [42]
performance [81]
cross compiler benchmarking [112]

2.4.2 python

webframeworks on python [Pankiv]
hope library [4]
bytecode effect [11]
numba [32]
intel python [68]

python 2 vs 3 [83]
python runtime performances [100] [85]
static vs dynamic languages [Pang et al.]
concurent benchmark framework [94]

2.4.3 jvm

java vs python [35]
hotspot vs J9 [91] - same but for big querries [25]
constant overheat of the jvm [65]
infer the energy cost based on the byte code instructions [73]
java collection energy footprint [99] [39]
java classes energy footprint [50]
framework to reduce java collections [74]
energy consumption of java dev tools [9]
Microbenchmarks on jvm [71] [9]
android automatic refactoring to reduce energy consumption [8] [102]
java vs native c in android [30]

2.4.4 benchmarking

NAS [Bailey et al.] statistics [51] benchmark crimes [106] microservices [44] impact of webservers on web applications energy [76]

2.5 Reproducibility in benchmarking

2.5.1 Virtual machines

To resolve this problem, practitioners tend to use Virtualisation. Using virtual machines aka VM gives reaserchers the freedom to choose their own tools, software and operating system that they are the most confortable with without paying the price to change the actual working environment, which will give them eventually more controle over the dependencies and the execution environement. Moreover, using a vm will solve the *replication crisis* thanks to the virtual images, even the most complex architecture can be reproduced easily by just instantiating a copy of the image. Since the virtual machines are agnostic to the host architecture, reaserchers won't have to worry about where and how their experiments are replicated because they have already setup the execution environement. Another advantage

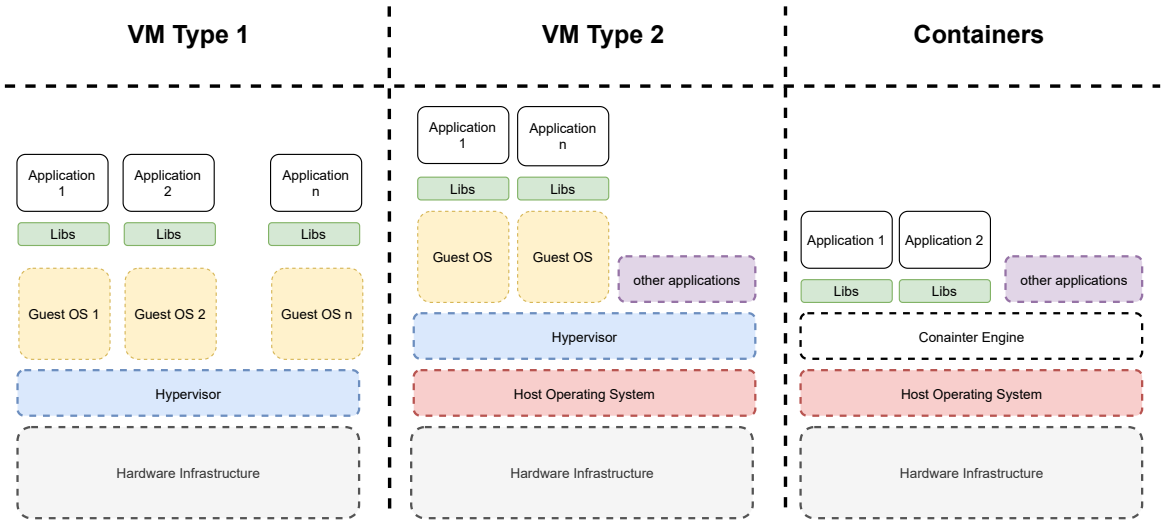

Figure 2.2: Different Methods of Virtualzation

to the virtual machines is the snapshot mechanism, it allows reaserchers to create backups and revert some changes with simple clicks. Last but not least,thanks to the isolation, virtual machines push the reproducibility further by allowing the future usages to see all the variables -controled and uncontroled- and do other analysis without dealing with any dependencies. In his paper [55] bill howe lists the advantages of using virtual machines in reaserchers experements including the economical impact and cultural limitation to a such approach.

which allow them to have control over the ressouces, the dependencies and the execution environment. Moreover, thanks to the snapshots, deploying a software is easily done by instantiating a copy of that image. However, this choice comes with a certain cost. Because the intervention of the hypervisor,the software will use two kernels , the virtual machine one and the host machine one, which will provide a noticiable overhead, and will impact the performances of the tests.Therefore, we can't use virtual machines for exepements that are related to performance. Another limiation with the virtual machine is the the isolation. It is true that this feature will prevent the experience environement with any undesirable interference from the outside world. but sometimes this contact is needed, especially when the experiment is dependent to an external part, such as sensors. In energicial tests we tend to use hardware powermeters which will make it difficult to use the virtual machines in this case .

2.5.2 Containers

another solution would be using something that allows us to have the isolation from the host os and the ease of replication that virtual machines offer, and the direct interaction with the

hardware that the classical method give. contarization offers a such advantages while keeping the isolation and the ease of replication for application.

Figure 2.2 explains the differents in architecture between the clasic types2 of Virtualisation and Containers.

- Type 1: runs directly on the hardware, it is mainly used by the cloud providers where there is no main OS, but just virtaul machines, we can site for this the open-srouce XEN and VMware ESX
- Type 2: runs over the hostmachine Operating System, mostly used for personal comptuers, VMware server and virtualBox are famous examples of this type, most of the reaserchers expermentation are run witht this type, howerver due to the 2 Operatings syestms the applications tend to be more slower
- containers : Instead of its own kernel, containers used the hots kernel to run their Os, which makes them ligher, quickers and use the full pontial use of the hardware. For this we can cite *Docker*, *Linux LXCLXD* [1]

2.5.3 Docker vs Virtual Machine

Depsite that Type 1 is more performant than type 2, the second one is the most used in reaserch, since most researchers conducts their experements in their own machine. In the other hand, docker is the most famous thechonology for for containers. In ower case we are more prone to docker for two reasons.

1. we need a litetweight orchestrator to not affect the energy consumption of ower tests. As prior work mentioned [cite Morabito (2015) and van Kessel et al. (2016)]
2. since we are using the hardware itself to measure the energy consumption, we are required to interract with the host OS itself.

Special notice to virtualwatts. A framework that allows us to retireve the energy consumption of a virtual machine.

2.5.4 Docker and energy

Now that we have chosen to go with the containers technorolgy to encapsulate our tests. What would be the impact of this solution on the energy consumption of our tests.

Based on the studies of [103], who analysed the impact of adding the docker layer on the energy consumption. In their experement. Eddie Antonio et al run multiple benchmarks with

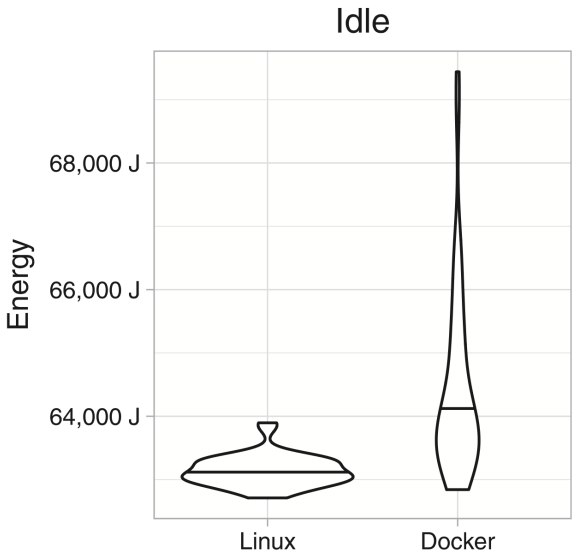

Figure 2.3: energy consumption of Idle system with and without docker [103]

and without docker. and compared their energy consumption and execution time. The first step was to see the impact of docker deamon while there is no work. to see the impact of the orchestrator alone. Later they had the experiment with the following benchmarks

- wordpress
- reddis
- postfresSQL

The following figures represents the energy consumption of the system while it is idle. As we can see in figure 2.3 Docker braught arround 1000joules overhead. In the other hand as we can see in figure 2.4. docker increased the execution time of the benchmark by 50 seconds which caused an increase in energy since they are highly correlated. The authors also highlated the fact that this increase of energy consumption is due to the docker deamon and not the fact that the application is in a container. Moreover they estimted the price of this extra energy and it was less than 0.15$ in the worst case. Which is non significant compared to the advantages that docker bring for isolation and reproducibility.
if we recap this study in one sentence,it would be the following one. The dockerised softwares tend to consume more energy, because mainly they take more time to be executed. The average power consumption is higher with only **2Watts** and it is due to the docker deamon.This overhead can be up to 5% for IO intensive application, but it is mearly noticiable when it comes to CPU or DRMA intesive works

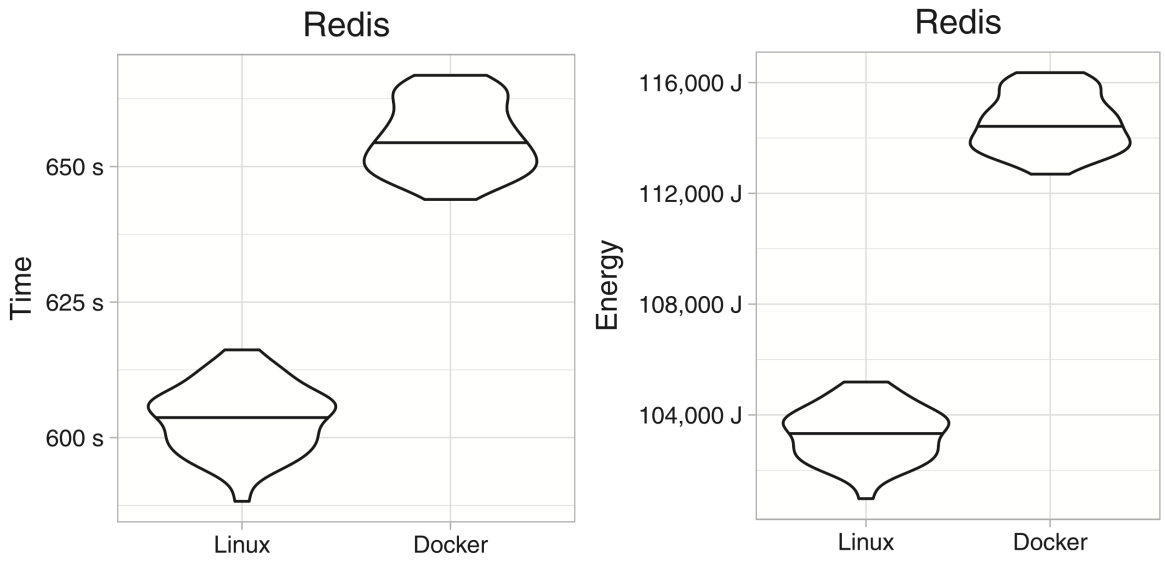

Figure 2.4: execution time and energy consumption of Redis with and without docker [103]

2.6 Energy Variation

2.6.1 Studying Hardware Factors

This variation has often been related to the manufacturing process [26], but has also been a subject of many studies, considering several aspects that could impact and vary the energy consumption across executions and on different chips. On the one hand, the correlation between the processor temperature and the energy consumption was one of the most explored paths. Kistowski *et al.* showed in [61] that identical processors can exhibit significant energy consumption variation with no close correlation with the processor temperature and performance. On the other hand, the authors of [111] claimed that the processor thermal effect is one of the most contributing factors to the energy variation, and the CPU temperature and the energy consumption variation are tightly coupled.

TODO : add the corelation value

This exposes the processor temperature as a delicate factor to consider while comparing energy consumption variations across a set of homogeneous processors.

The ambient temperature was also discussed in many papers as a candidate factor for the energy variation of a processor. In [107], the authors claimed that energy consumption may vary due to fluctuations caused by the external environment. These fluctuations may alter the processor temperature and its energy consumption. However, the temperature inside a data center does not show major variations from one node to another. In [37], El Mehdi Dirouri *et al.* showed that switching the spot of two servers does not affect their

energy consumption. Moreover, changing hardware components, such as the hard drive, the memory or even the power supply, does not affect the energy variation of a node, making it mainly related to the processor. This result was recently assessed by [111], where the rack placement and power supply introduced a maximum of 2.8 % variation in the observed energy consumption.

Beyond hardware components, the accuracy of power meters has also been questioned. Inadomi *et al.* [58] used three different power measurement tools: RAPL, Power Insight[2] and BGQ EMON. All of the three tools recorded the same 10 % of energy variation, that was supposedly related to the manufacturing process.

2.6.2 Mitigating Energy Variations.

Acknowledging the energy variation problem on processors, some papers proposed contributions to reduce and mitigate this variation. In [58], the authors introduced a variation-aware algorithm that improves application performance under a power constraint by determining module-level (individual processor and associated DRAM) power allocation, with up to 5.4× speedup. The authors of [49] proposed parallel algorithms that tolerate the variability and the non-uniformity by decoupling per process communication over the available CPU. Acun *et al.* [3] found out a way to reduce the energy variation on Ivy Bridge and Sandy Bridge processors, by disabling the Turbo Boost feature to stabilize the execution time over a set of processors. They also proposed some guidelines to reduce this variation by replacing the old—slower—chips, by load balancing the workload on the CPU cores and leaving one core idle. They claimed that the variation between the processor cores is insignificant. In [24], the researchers showed how a parallel system can be used to deal with the energy variation by compensating the uneven effects of power capping.

In [78], the authors highlight the increase of energy variation across the latest Intel micro-architectures by a factor of 4 from Sandy Bridge to Broadwell, a 15 % of run-to-run variation within the same processor and the increase of the inter-cores variation from 2.5 % to 5 % due to hardware-enforced constraints, concluding with some recommendations for Broadwell usage, such as running one hyper-thread per core.

2.7 Conclusion

As we have seen in the state of the art, many methodes have been proposed to reduce the energy footprint of the the ICT,which they can be applied in deffirent parts of the lifecycle

[2]https://www.itssolution.com/products/trellis-power-insight-application

of the program, from consumption to the execution. Furthermore, the execution phase took attention of many researchers because it's the part where the most energy is consumed. This thesis will focus on that aspect as well. However, unlike the most work that have been done on the hardware aspect, we will target the software impact on this energy, starting from the choice of the programming language up to how to tune some features of a frameworks in order to make the software consumes less energy.To do so we use the emprical apporach due to it's concinstency for the moment.Unlike the perfromance which essentialy related to the complexity of the algorithm, the energy consumption is more impacted by the hardware. Therefore, to optimize the enrgy consumption we choose a spiral methode. based on 3 phases:

1. execute the code
2. measure the program
3. infere the guidelines

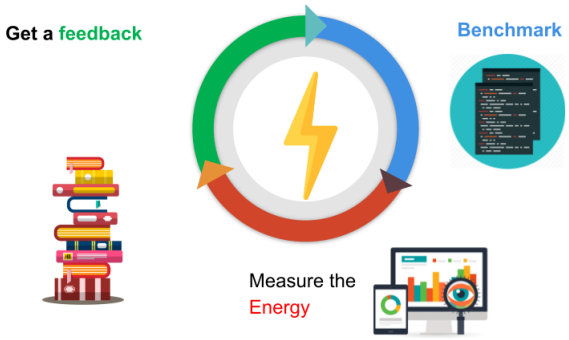

Figure 2.5: the spiral methode of energy optimization

the aim of this work is to present a set of guidelines to create a benchmarking system to to measure the energy consumption of different programs. After that, we will use this system to compare the energy consumption of different programming languages. We will extend the work of Pereira et al. to a closer distance to production environment by comparing a set of usecases. starting by GRPC framework, and a set of Web Framewroks. Finally we will discuss the impact of the execution environment on the energy consumption of two of the most famous programming languages JAVA and Python. and present how can tunning the Virtual Machine can reduce the energy consumption.

Bibliography

[1] Abuabdo, A. and Al-Sharif, Z. A. (2019). Virtualization vs. Containerization: Towards a Multithreaded Performance Evaluation Approach. In *2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA)*, pages 1–6.

[2] Acun, B., Miller, P., and Kale, L. V. (2016a). Variation among processors under turbo boost in hpc systems. In *Proceedings of the 2016 International Conference on Supercomputing*, pages 1–12.

[3] Acun, B., Miller, P., and Kale, L. V. (2016b). Variation Among Processors Under Turbo Boost in HPC Systems.

[4] Akeret, J., Gamper, L., Amara, A., and Refregier, A. (2015). HOPE: A Python just-in-time compiler for astrophysical computations. *Astronomy and Computing*, 10:1–8.

[5] Arcuri, A. and Briand, L. (2011). A practical guide for using statistical tests to assess randomized algorithms in software engineering. In *Proceeding of the 33rd International Conference on Software Engineering - ICSE '11*, page 1, Waikiki, Honolulu, HI, USA. ACM Press.

[Bailey et al.] Bailey, D., Barszcz, E., Barton, J., Browning, D., Carter, R., Dagum, L., Fineberg, S., Frederickson, P., Lasinski, T., Schreiber, R., Simon, H., Venkatakrishnan, V., and Weeratunga, S. The NAS Parallel Benchmarks. page 79.

[7] Balasubramanian, N., Balasubramanian, A., and Venkataramani, A. (2009). Energy consumption in mobile phones: a measurement study and implications for network applications. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, pages 280–293.

[8] Banerjee, A. and Roychoudhury, A. (2016). Automated re-factoring of android apps to enhance energy-efficiency. In *Proceedings of the International Conference on Mobile Software Engineering and Systems*, pages 139–150.

[9] Baskar, P., Joseph, M. A., Narayanan, N., and Loya, R. B. (2013). Experimental investigation of oxygen enrichment on performance of twin cylinder diesel engine with variation of injection pressure. In *2013 International Conference on Energy Efficient Technologies for Sustainability*, pages 682–687. IEEE.

[10] Bedard, D., Lim, M. Y., Fowler, R., and Porterfield, A. (2010). Powermon: Fine-grained and integrated power monitoring for commodity computer systems. In *Proceedings of the IEEE SoutheastCon 2010 (SoutheastCon)*, pages 479–484. IEEE.

[11] Ben Asher, Y. and Rotem, N. (2009). The effect of unrolling and inlining for Python bytecode optimizations. In *Proceedings of SYSTOR 2009: The Israeli Experimental Systems Conference on - SYSTOR '09*, page 1, Haifa, Israel. ACM Press.

[12] Benmoussa, K., Laaziri, M., Khoulji, S., Larbi, K. M., and Yamami, A. E. (2019). A new model for the selection of web development frameworks: Application to PHP frameworks. *International Journal of Electrical and Computer Engineering (IJECE)*, 9(1):695–703.

[13] Breitbart, J., Weidendorfer, J., and Trinitis, C. (2015). Case study on co-scheduling for hpc applications. In *2015 44th International Conference on Parallel Processing Workshops*, pages 277–285. IEEE.

[14] Bujnowski, G. and Smołka, J. (2020). Java and kotlin code performance in selected web frameworks. *Journal of Computer Sciences Institute*, 16:219–226.

[15] Bukh, P. N. D. (1992). The art of computer systems performance analysis, techniques for experimental design, measurement, simulation and modeling.

[16] Burtscher, M., Zecena, I., and Zong, Z. (2014). Measuring gpu power with the k20 built-in sensor. In *Proceedings of Workshop on General Purpose Processing Using GPUs*, pages 28–36.

[Buytaert and Eeckhout] Buytaert, A. G. D. and Eeckhout, L. Statistically Rigorous Java Performance Evaluation. page 20.

[Caldeira et al.] Caldeira, A. B., Grabowski, B., Haug, V., Kahle, M.-E., Laidlaw, A., Maciel, C. D., Sanchez, M., and Sung, S. Y. Ibm power system s822.

[19] Caldeira, A. B., Grabowski, B., Haug, V., Kahle, M.-E., Maciel, C. D., and Sanchez, M. (2014). Ibm power systems s814 and s824 technical overview and introduction. *IBM Redbook REDP-5097-00*.

[20] Calero, C., Mancebo Pavón, J., and García, F. (2021). Does maintainability relate to the energy consumption of software?

[21] Cardenas, L. G., Gil, J. A., Domenech, J., Sahuquillo, J., and Pont, A. (2005). Performance comparison of a Web cache simulation framework. In *19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA Papers)*, volume 2, pages 281–284 vol.2.

[22] Chamas, C. L., Cordeiro, D., and Eler, M. M. (2017). Comparing rest, soap, socket and grpc in computation offloading of mobile applications: An energy cost analysis. In *2017 IEEE 9th Latin-American Conference on Communications (LATINCOM)*, pages 1–6. IEEE.

[23] Chasapis, D., Casas, M., Moretó, M., Schulz, M., Ayguadé, E., Labarta, J., and Valero, M. (2016a). Runtime-guided mitigation of manufacturing variability in power-constrained multi-socket numa nodes. In *Proceedings of the 2016 International Conference on Supercomputing*, pages 1–12.

[24] Chasapis, D., Schulz, M., Casas, M., Ayguadé, E., Valero, M., Moretó, M., and Labarta, J. (2016b). Runtime-Guided Mitigation of Manufacturing Variability in Power-Constrained Multi-Socket NUMA Nodes.

[25] Chiba, T., Yoshimura, T., Horie, M., and Horii, H. (2018). Towards selecting best combination of sql-on-hadoop systems and jvms. In *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, pages 245–252. IEEE.

[26] Coles, H., Qin, Y., and Price, P. (2014a). Comparing Server Energy Use and Efficiency Using Small Sample Sizes. Technical Report LBNL-6831E, 1163229.

[27] Coles, H. C., Qin, Y., and Price, P. N. (2014b). Comparing server energy use and efficiency using small sample sizes. Technical report, Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States).

[28] Coley, G. (2012). Beaglebone rev a6 system reference manual. *Obtenido*, 4(23):2012.

[29] Colmant, M., Rouvoy, R., Kurpicz, M., Sobe, A., Felber, P., and Seinturier, L. (2018). The next 700 cpu power models. *Journal of Systems and Software*, 144:382–396.

[30] Corral, L., Georgiev, A. B., Sillitti, A., and Succi, G. (2014). Method reallocation to reduce energy consumption: an implementation in android os. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, pages 1213–1218.

[31] Couto, M., Pereira, R., Ribeiro, F., Rua, R., and Saraiva, J. (2017). Towards a green ranking for programming languages. In *Proceedings of the 21st Brazilian Symposium on Programming Languages*, pages 1–8.

[32] Crist, J. (2016). Dask Numba: Simple libraries for optimizing scientific python code. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 2342–2343.

[33] Das, R. and Saikia, D. L. P. (2016). Comparison of Procedural PHP with Codeigniter and Laravel Framework. /paper/A-new-model-for-the-selection-of-web-development-to-Benmoussa-Laaziri/6eb6977a37b2d38e232472b43897ab6047bd751c.

[34] de Matos, F. F. S., Rego, P. A., and Trinta, F. A. M. (2021). An empirical study about the adoption of multi-language technique in computation offloading in a mobile cloud computing scenario. In *CLOSER*, pages 207–214.

[35] Destefanis, G., Ortu, M., Porru, S., Swift, S., and Marchesi, M. (2016). A Statistical Comparison of Java and Python Software Metric Properties. In *Proceedings of the 7th International Workshop on Emerging Trends in Software Metrics*, WETSoM '16, pages 22–28, New York, NY, USA. ACM.

[36] Diouri, M. E. M., Glück, O., Lefevre, L., and Mignot, J.-C. (2013). Your cluster is not power homogeneous: Take care when designing green schedulers! In *2013 International Green Computing Conference Proceedings*, pages 1–10. IEEE.

[37] El Mehdi Diouri, M., Gluck, O., Lefevre, L., and Mignot, J.-C. (2013). Your cluster is not power homogeneous: Take care when designing green schedulers!

[38] Fahad, M., Shahid, A., Manumachu, R. R., and Lastovetsky, A. (2019). A comparative study of methods for measurement of energy of computing. *Energies*, 12(11):2204.

[39] Fernandes, B., Pinto, G., and Castor, F. (2017). Assisting non-specialist developers to build energy-efficient software. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pages 158–160. IEEE.

[40] Fieni, G., Rouvoy, R., and Seinturier, L. (2020). Smartwatts: Self-calibrating software-defined power meter for containers. In *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*, pages 479–488. IEEE.

[41] Fieni, G., Rouvoy, R., and Seiturier, L. (2021). Selfwatts: On-the-fly selection of performance events to optimize software-defined power meters. In *2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pages 324–333. IEEE.

[42] Gajewski, M. and Zabierowski, W. (2019). Analysis and Comparison of the Spring Framework and Play Framework Performance, Used to Create Web Applications in Java. In *2019 IEEE XVth International Conference on the Perspective Technologies and Methods in MEMS Design (MEMSTECH)*, pages 170–173.

[43] Ge, R., Feng, X., Song, S., Chang, H.-C., Li, D., and Cameron, K. W. (2009). Powerpack: Energy profiling and analysis of high-performance systems and applications. *IEEE Transactions on Parallel and Distributed Systems*, 21(5):658–671.

[44] Grambow, M., Meusel, L., Wittern, E., and Bermbach, D. (2020). Benchmarking microservice performance: A pattern-based approach. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pages 232–241, Brno Czech Republic. ACM.

[45] Guimarães, M., Saraiva, J., and Belo, O. (2016). Some heuristic approaches for reducing energy consumption on database systems. *DBKDA 2016*, page 59.

[46] Hackenberg, D., Ilsche, T., Schöne, R., Molka, D., Schmidt, M., and Nagel, W. E. (2013). Power measurement techniques on standard compute nodes: A quantitative comparison. In *2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 194–204. IEEE.

[47] Hackenberg, D., Ilsche, T., Schuchart, J., Schöne, R., Nagel, W. E., Simon, M., and Georgiou, Y. (2014). Hdeem: high definition energy efficiency monitoring. In *2014 Energy Efficient Supercomputing Workshop*, pages 1–10. IEEE.

[48] Hackenberg, D., Schöne, R., Ilsche, T., Molka, D., Schuchart, J., and Geyer, R. (2015). An energy efficiency feature survey of the intel haswell processor. In *2015 IEEE international parallel and distributed processing symposium workshop*, pages 896–904. IEEE.

[49] Hammouda, A., Siegel, A. R., and Siegel, S. F. (2015). Noise-Tolerant Explicit Stencil Computations for Nonuniform Process Execution Rates. *ACM Transactions on Parallel Computing*, 2(1):1–33.

[50] Hasan, S., King, Z., Hafiz, M., Sayagh, M., Adams, B., and Hindle, A. (2016). Energy profiles of java collections classes. In *Proceedings of the 38th International Conference on Software Engineering*, pages 225–236.

[51] He, S., Manns, G., Saunders, J., Wang, W., Pollock, L., and Soffa, M. L. (2019). A statistics-based performance testing methodology for cloud applications. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering - ESEC/FSE 2019*, pages 188–199, Tallinn, Estonia. ACM Press.

[52] Heinrich, F., Carpen-Amarie, A., Degomme, A., Hunold, S., Legrand, A., Orgerie, A.-C., and Quinson, M. (2017). Predicting the performance and the power consumption of mpi applications with simgrid.

[53] Hindle, A., Wilson, A., Rasmussen, K., Barlow, E. J., Campbell, J. C., and Romansky, S. (2014). Greenminer: A hardware based mining software repositories software energy consumption framework. In *Proceedings of the 11th working conference on mining software repositories*, pages 12–21.

[54] Hirst, J. M., Miller, J. R., Kaplan, B. A., and Reed, D. D. (2013). Watts up? pro ac power meter for automated energy recording.

[55] Howe, B. (2012). Virtual Appliances, Cloud Computing, and Reproducible Research. *Computing in Science Engineering*, 14(4):36–41.

[56] Ilsche, T., Hackenberg, D., Graul, S., Schone, R., and Schuchart, J. (2015). Power measurements for compute nodes: Improving sampling rates, granularity and accuracy. In *2015 Sixth International Green and Sustainable Computing Conference (IGSC)*, pages 1–8, Las Vegas, NV, USA. IEEE.

[57] Inadomi, Y., Patki, T., Inoue, K., Aoyagi, M., Rountree, B., Schulz, M., Lowenthal, D., Wada, Y., Fukazawa, K., Ueda, M., et al. (2015a). Analyzing and mitigating the impact of manufacturing variability in power-constrained supercomputing. In *SC'15: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–12. IEEE.

[58] Inadomi, Y., Ueda, M., Kondo, M., Miyoshi, I., Patki, T., Inoue, K., Aoyagi, M., Rountree, B., Schulz, M., Lowenthal, D., Wada, Y., and Fukazawa, K. (2015b). Analyzing and mitigating the impact of manufacturing variability in power-constrained supercomputing.

[59] Islam, S., Noureddine, A., and Bashroush, R. (2016). Measuring energy footprint of software features. In *2016 IEEE 24th International Conference on Program Comprehension (ICPC)*, pages 1–4. IEEE.

[60] Jagroep, E., Procaccianti, G., van der Werf, J. M., Brinkkemper, S., Blom, L., and van Vliet, R. (2017). Energy efficiency on the product roadmap: an empirical study across releases of a software product. *Journal of Software: Evolution and process*, 29(2):e1852.

[61] Joakim v Kisroski, Hansfreid Block, John Beckett, Cloyce Spradling, Klaus-Dieter Lange, and Samuel Kounev (2016). Variations in CPU Power Consumption.

[62] Koomey, J. et al. (2011). Growth in data center electricity use 2005 to 2010. *A report by Analytical Press, completed at the request of The New York Times*, 9(2011):161.

[63] Kothari, N. and Bhattacharya, A. (2009). Joulemeter: Virtual machine power measurement and management. *MSR Tech Report*.

[64] Kurpicz, M., Orgerie, A.-C., and Sobe, A. (2016). How much does a vm cost? energy-proportional accounting in vm-based environments. In *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, pages 651–658. IEEE.

[65] Lafond, S. and Lilius, J. (2006). An energy consumption model for an embedded java virtual machine. In *International Conference on Architecture of Computing Systems*, pages 311–325. Springer.

[66] Laros, J. H., Pokorny, P., and DeBonis, D. (2013). Powerinsight-a commodity power measurement capability. In *2013 International Green Computing Conference Proceedings*, pages 1–6. IEEE.

[67] LeBeane, M., Ryoo, J. H., Panda, R., and John, L. K. (2015). Watt watcher: fine-grained power estimation for emerging workloads. In *2015 27th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*, pages 106–113. IEEE.

[68] Li, Y. and Schwiebert, L. (2016). Boosting Python Performance on Intel Processors: A Case Study of Optimizing Music Recognition. In *2016 6th Workshop on Python for High-Performance and Scientific Computing (PyHPC)*, pages 52–58.

[69] Lilja, D. J. (2005). *Measuring computer performance: a practitioner's guide*. Cambridge university press.

[70] Liu, K., Pinto, G., and Liu, Y. D. (2015). Data-oriented characterization of application-level energy optimization. In *International Conference on Fundamental Approaches to Software Engineering*, pages 316–331. Springer.

[71] Longo, M., Rodriguez, A. V., Mateos Diaz, C. M., and Zunino Suarez, A. O. (2019). Reducing energy usage in resource-intensive java-based scientific applications via micro-benchmark based code refactorings.

[72] Lovicott, D. (2009). Thermal design of the dell™ poweredge™ t610™, r610™, and r710™ servers. *Round Rock. Texas*.

[73] Ma, H., Simon, D., Siarry, P., Yang, Z., and Fei, M. (2017). Biogeography-based optimization: a 10-year review. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 1(5):391–407.

[74] Manotas, I., Pollock, L., and Clause, J. (2014). Seeds: A software engineer's energy-optimization decision support framework. In *Proceedings of the 36th International Conference on Software Engineering*, pages 503–514.

[75] Manotas, I., Sahin, C., Clause, J., Pollock, L., and Winbladh, K. (2013a). Investigating the impacts of web servers on web application energy usage. In *2013 2nd International Workshop on Green and Sustainable Software (GREENS)*, pages 16–23. IEEE.

[76] Manotas, I., Sahin, C., Clause, J., Pollock, L., and Winbladh, K. (2013b). Investigating the impacts of web servers on web application energy usage. In *2013 2nd International Workshop on Green and Sustainable Software (GREENS)*, pages 16–23.

[77] Marathe, A., Zhang, Y., Blanks, G., Kumbhare, N., Abdulla, G., and Rountree, B. (2017a). An empirical survey of performance and energy efficiency variation on intel processors. In *Proceedings of the 5th International Workshop on Energy Efficient Supercomputing*, pages 1–8.

[78] Marathe, A., Zhang, Y., Blanks, G., Kumbhare, N., Abdulla, G., and Rountree, B. (2017b). An empirical survey of performance and energy efficiency variation on Intel processors.

[79] McCreary, H.-Y., Broyles, M. A., Floyd, M. S., Geissler, A. J., Hartman, S. P., Rawson, F. L., Rosedahl, T. J., Rubio, J. C., and Ware, M. S. (2007). Energyscale for ibm power6 microprocessor-based systems. *IBM Journal of Research and Development*, 51(6):775–786.

[80] Mirowski, P., Mathewson, K., Branch, B., Winters, T., Verhoeven, B., and Elfving, J. (2020). Rosetta code: Improv in any language. In *Proceedings of the 11th International Conference on Computational Creativity*, pages 115–122. Association for Computational Creativity.

[81] Mishra, S. and Srivastava, S. (2021). Web development frameworks and its performance analysis—a review. *Smart Computing*, pages 337–343.

[82] Mishra, S. K., Puthal, D., Sahoo, B., Jayaraman, P. P., Jun, S., Zomaya, A. Y., and Ranjan, R. (2018). Energy-efficient vm-placement in cloud data center. *Sustainable computing: informatics and systems*, 20:48–55.

[83] Modzelewski, K. (2020). Pyston v2: 20% faster Python.

[84] Mukherjee, T., Banerjee, A., Varsamopoulos, G., Gupta, S. K., and Rungta, S. (2009). Spatio-temporal thermal-aware job scheduling to minimize energy consumption in virtualized heterogeneous data centers. *Computer Networks*, 53(17):2888–2904.

[85] Murri, R. (2013). Performance of Python runtimes on a non-numeric scientific code. page 6.

[86] Mytkowicz, T., Diwan, A., Hauswirth, M., and Sweeney, P. F. (2009). Producing wrong data without doing anything obviously wrong! *ACM Sigplan Notices*, 44(3):265–276.

[87] Nanz, S. and Furia, C. A. (2015). A comparative study of programming languages in rosetta code. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 1, pages 778–788. IEEE.

[88] Noureddine, A. (2022). Powerjoular and joularjx: Multi-platform software power monitoring tools. In *18th International Conference on Intelligent Environments*.

[89] Noureddine, A., Islam, S., and Bashroush, R. (2016). Jolinar: analysing the energy footprint of software applications. In *Proceedings of the 25th International Symposium on Software Testing and Analysis*, pages 445–448.

[90] Noureddine, A., Rouvoy, R., and Seinturier, L. (2015). Monitoring energy hotspots in software. *Automated Software Engineering*, 22(3):291–332.

[91] Oi, H. (2011). Power-performance analysis of jvm implementations. In *ICIMU 2011: Proceedings of the 5th international Conference on Information Technology & Multimedia*, pages 1–7. IEEE.

[Pang et al.] Pang, A., Anslow, C., and Noble, J. What Programming Languages Do Developers Use? a Theory of Static vs Dynamic Language Choice. page 9.

[Pankiv] Pankiv, A. Concurrent benchmark system for web-frameworks on Python. page 26.

[94] Pankiv, A. (2019). *Concurrent benchmark system for web-frameworks on Python*. PhD thesis, Ukrainian Catholic University.

[95] Pereira, R., Carcao, T., Couto, M., Cunha, J., Fernandes, J. P., and Saraiva, J. (2017a). Helping Programmers Improve the Energy Efficiency of Source Code. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pages 238–240, Buenos Aires, Argentina. IEEE. ## main idea
This paper presents a technique to spot the energy leaks inside a program using a spectrum technique called SPELL (Spectrum-based Energy Leak Localization)
How it works
We have a matrix (n*m) where n is the number of the tests and m is the number of components (like classes, methods, packages, etc.) after this we calculate the oracle (a vector that says which component is responsible for what)
Contributions
This paper helps developers to spot the red areas in their code and optimize the energy consumption. As an example it helped to reduce the energy consumption of a java application 50% faster and with 18% more efficiency.

[96] Pereira, R., Carção, T., Couto, M., Cunha, J., Fernandes, J. P., and Saraiva, J. (2017b). Helping programmers improve the energy efficiency of source code. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pages 238–240. IEEE.

[97] Pereira, R., Couto, M., Ribeiro, F., Rua, R., Cunha, J., Fernandes, J. P., and Saraiva, J. (2017c). Energy efficiency across programming languages: how do energy, time, and memory relate? In *Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering*, pages 256–267.

[98] Philippot, O., Anglade, A., and Leboucq, T. (2014). Characterization of the energy consumption of websites: Impact of website implementation on resource consumption. In *ICT for Sustainability 2014 (ICT4S-14)*, pages 171–178. Atlantis Press.

[99] Pinto, G., Liu, K., Castor, F., and Liu, Y. D. (2016). A comprehensive study on the energy efficiency of java's thread-safe collections. In *2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 20–31. IEEE.

[100] Redondo, J. M. and Ortin, F. (2015). A Comprehensive Evaluation of Common Python Implementations. *IEEE Software*, 32(4):76–84. benchmarks link http://www.reflection.uniovi.es/python.

[101] Ribic, H. and Liu, Y. D. (2016). Aequitas: Coordinated energy management across parallel applications. In *Proceedings of the 2016 International Conference on Supercomputing*, pages 1–12.

[102] Rodriguez, A. (2017). Reducing energy consumption of resource-intensive scientific mobile applications via code refactoring. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pages 475–476. IEEE.

[103] Santos, E. A., McLean, C., Solinas, C., and Hindle, A. (2018). How does docker affect energy consumption? evaluating workloads in and out of docker containers. *Journal of Systems and Software*, 146:14–25.

[104] Sonnenwald, D. H., Whitton, M. C., and Maglaughlin, K. L. (2003). Evaluating a scientific collaboratory: Results of a controlled experiment. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 10(2):150–176.

[105] Strubell, E., Ganesh, A., and McCallum, A. (2019). Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*.

[106] van der Kouwe, E., Andriesse, D., Bos, H., Giuffrida, C., and Heiser, G. (2018). Benchmarking Crimes: An Emerging Threat in Systems Security. *arXiv:1801.02381 [cs]*.

[107] Varsamopoulos, G., Banerjee, A., and Gupta, S. K. S. (2009). Energy Efficiency of Thermal-Aware Job Scheduling Algorithms under Various Cooling Models. In *Contemporary Computing*, volume 40. Springer.

[108] von Kistowski, J., Block, H., Beckett, J., Spradling, C., Lange, K.-D., and Kounev, S. (2016). Variations in cpu power consumption. In *Proceedings of the 7th ACM/SPEC on International Conference on Performance Engineering*, pages 147–158.

[109] Wang, X., Zhao, H., and Zhu, J. (1993). Grpc: A communication cooperation mechanism in distributed systems. *ACM SIGOPS Operating Systems Review*, 27(3):75–86.

[110] Wang, Y., Nörtershäuser, D., Le Masson, S., and Menaud, J.-M. (2018a). Potential effects on server power metering and modeling. *Wireless Networks*, pages 1–8.

[111] Wang, Y., Nörtershäuser, D., Le Masson, S., and Menaud, J.-M. (2018b). Potential effects on server power metering and modeling. *Wireless Networks*.

[112] Yet, A. W. F. (2016). Cross-language compiler benchmarking.