International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 06 | Jun 2022 www.irjet.net p-ISSN: 2395-0072

Sustainable Development using Green Programming

Adarsh Sawant[1], Prashant Trivedi[2], Dr. Shivkumar Goel[3]

[1]*Student, Vivekanand Education Society's Institute Of Technology, Collector Colony, Chembur ,Mumbai-400074,India*
[2]*Student, Vivekanand Education Society's Institute Of Technology, Collector Colony, Chembur ,Mumbai-400074,India*
[3]*Associate Professor/HOD, Vivekanand Education Society's Institute Of Technology, Collector Colony, Chembur , Mumbai- 400074,India*

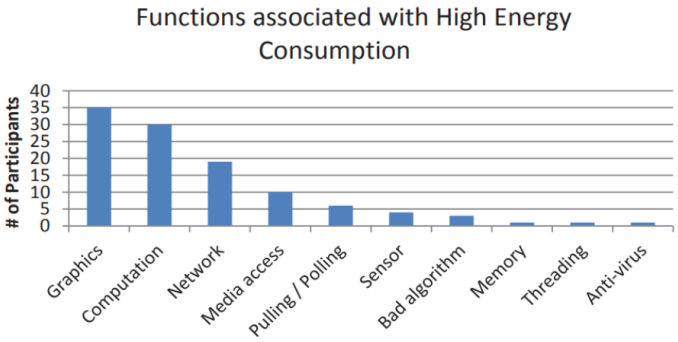
Abstract - *Programmers and developers receive professional training on programming languages and Methodologies but rarely about software energy consumption. Modern technologies such as mobile applications and cloud computing require increased awareness about software energy consumption. Generally, mobile device computations are limited by battery life. Is the programmer knowledgeable enough about software energy consumption and methods to minimize energy consumption? Programmers have limited knowledge about energy efficiency, lack knowledge about the best practices to reduce the energy consumption of software, and are often unsure about how software consumes energy. Education about the importance of energy-effective software will benefit the programmers and minimize energy consumption.*

Key Words: Developers, Energy Consumption, Energy Efficiency, Energy effective software.

1. INTRODUCTION

The non-functional requirement of minimizing software energy consumption becomes a concern for software systems and software developers. In mobile device-based applications, energy consumption affects battery life and limits device usage. For data centers, energy consumption limits the number of machines that can be used and cooled after usage. Energy consumption is also called power consumption by consumers, as energy and power are related. Energy is usually measured in Joules (J) and Power is measured in watts (W), which is the rate of energy conversion, transfer, or consumption. Electricity providers charge users by their energy consumption measured kilowatt-hours (kWh). The basic training of programmers often focuses on methodologies such as object-oriented programming, and non-functional requirements such as performance. Performance optimization is often considered a substitution of energy optimization since a faster system likely consumes less energy. Although this step is in a positive direction, it is not at all sufficient. For instance, parallel processing might improve performance by reducing calculation time.

1.1 Functions Associated with high Energy Consumption



1.Graphics: GPU was not placed in a low-power state when not used; rather it was placed in its peak power state, thereby dissipating a higher amount of energy without any actual benefit.

2.Computation: Any computational complexity measure is related to the running time. The running time of an algorithm on a particular input is the number of primitive operations or "steps" executed.

3.Bad Algorithm: A bad algorithm can increase the complexity of the program resulting in high consumption of energy.

4.Memory: The system architecture consists of two memories, namely the instruction memory and data memory (Harvard architecture), and there is no cache memory. The energy consumption of the instruction memory depends on the code size and on the number of executed instructions that correspond to instruction fetches, whereas that of the data memory depends on the volume of data being processed by the application and on how often the latter accesses data.

5.Antivirus: Using an antivirus program means tons of resources from the memory and therefore the disk drive is getting used. As a result, it can drastically slow down the overall speed of the pc. Moreover, the method of scanning also can cause lags within the network.

6.Network: A pair of researchers in California compiled calculations and estimate that it takes between 170 and 307 Gigawatts of energy each year to support the socket power and infrastructure needed to run the Internet. To put that estimate in some perspective, even the lowest number is the equivalent of powering several billion 100watt incandescent bulbs. While that number is considerable, even taking the maximum number of that estimate, 307 Gigawatts constitutes only 1.9% of the energy expended across commercial, residential, industrial, and transportation sectors.

7.Threading: With more threads, the code becomes difficult to debug and maintain. Thread creation puts a load on the system in terms of memory and CPU resources. We need to do exception handling inside the worker method as any unhandled exceptions can result in the program crashing.

8. Sensor: An additional parameter is the WSN operational life, which strongly depends on the balance between power consumption and energy storage. In particular, WSNs are characterized by limited power storage, with possible mitigation coming from power harvesting techniques. Nevertheless, energy efficiency is a critical issue, to be pursued both at node and network level. Typical assumptions include considering the radio interface as the main contributor to power consumption. As a consequence, great attention has been given in the literature to protocol optimization, aimed for instance at minimizing the amount of data transmissions throughout the network, and the maximization of node low-power residence time.

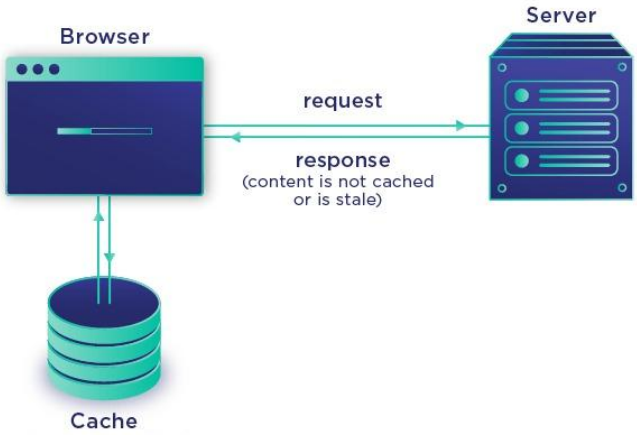
Sensor	Producer	Sensing	Power Consumption
STCN75	STM	Temperature	0.4 mW
QST108KT6	STM	Touch	7 mW
SG-LINK (1000Ω)	MicroStrain	Strain gauge	9 mW
SG-LINK (350Ω)	MicroStrain	Strain gauge	24 mW
iMEMS	ADI	Accelerometer (3 axis)	30 mW
2200 Series, 2600 Series	GEMS	Pressure	50 mW
T150	GEFRAN	Humidity	90 mW
LUC-M10	PEPPERL+FUCHS	Level Sensor	300 mW
CP18, VL18, GM60, GLV30	VISOLUX	Proximity	350 mW
TDA0161	STM	Proximity	420 mW
FCS-GL1/2A4-AP8X-H1141	TURCK	Flow Control	1250 mW

1.2 Methods to Improve Energy Efficiency

1. Better Algorithm: A better algorithm will result in less complexity of the program which reduces the energy consumption.

2. Less Computation: The number of computations decides the amount of energy consumed; basically, fewer computations lead to less energy consumption.

3. Improve caching: A cache is a high-speed data storage layer that stores a subset of data, typically transient in nature, so that future requests for that data are served up faster than is possible by accessing the data's primary storage location. Caching allows you to efficiently reuse previously retrieved or computed data.



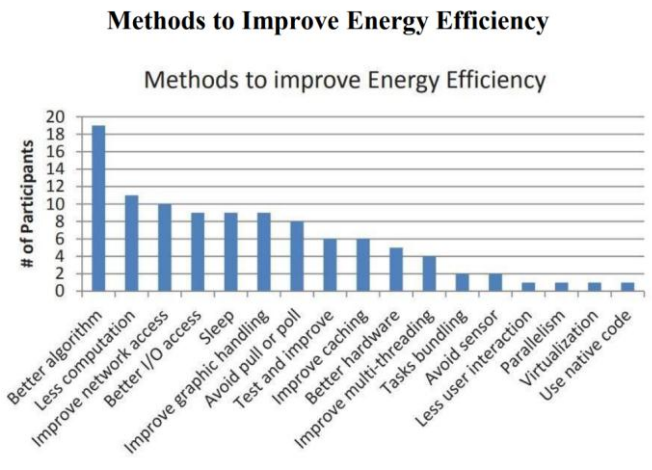
4. Improve Multi-threading: It increases performance in the sense that it (usually, and depending on the task's characteristics) .reduces the wall clock time that is required to perform the task.

5. Virtualization: It is a process that allows for more efficient utilization of physical computer hardware and is the foundation of cloud computing.

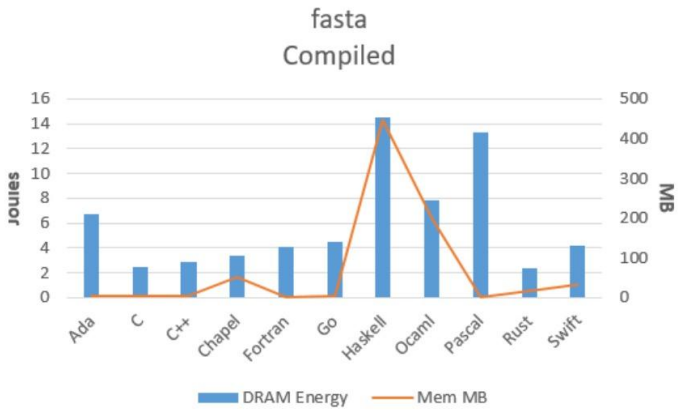
6.Improve Graphic Handling: With the ubiquity and increased usage of mobile devices with higher processing capability, power consumption in such devices is an important issue. In this paper we propose a software based approach to reduce the power consumption of an Android application which is having higher graphics or rendering requirements. Graphics intensive applications such as games, internet browser and video player contribute most in draining the battery. Therefore, we concentrate most on such applications for reducing the power consumption. Our approach involves removing the surface view and directly utilizing the application's main window surface.

7.Better Hardware: ABI Research says hardware optimisation will significantly improve power consumption, with the new generation of chipsets offering typical energy savings of 30% to 70%.

8. Use Native Code: Native apps are designed keeping in mind the framework of the operating system on which alone they will work. This allows fine-tuning of the app according to the requirements and functionalities of the OS. Programmed with the core programming languages of the operating system, they offer faster speed and increased efficiency over Hybrid apps.



2. Power Consumption of Programming Languages



1. A faster program will always use less energy, pointing out that it's not as simple as the law of physics that says E(nergy) = T(ime) x P(ower). This is partly because power isn't expended at a consistent rate, the researchers note, suggesting that may be impacting the work of other researchers investigating whether a program's running time affects its energy consumption. ("Conclusions regarding this issue diverge sometimes…")

2. In one of their benchmark tests, a Chapel program took 55 percent less time to execute than an equivalent program written in Pascal — and yet that Pascal program used 10 percent less energy.

3.So while there's still a common belief that energy consumption goes down when programs run faster, the researchers state unequivocally that "a faster language is not always the most energy efficient."

4.It can be a hard question to answer, since power consumption is affected by many factors (including the quality of the compiler and what libraries are used).

5.Ultimately the researchers were even able to break down energy consumption based on whether it was being consumed by the CPU or DRAM — concluding that the majority of power (around 88 percent) was consumed by the CPU, on average, whether the benchmark program was compiled, interpreted, or run on a virtual machine.

6.Interestingly, interpreted languages showed a slightly higher variation, with the CPU sometimes consuming as much as 92.90 percent of the power or as little as 81.57 percent.

7..After studying their results, the researchers also concluded that the relationship between peak usage of DRAM and energy consumption "is almost non-existent."

8.The research provides some more insights into the perennial question: is faster greener? Yes, it's true that "the top five most energy-efficient languages keep their rank when they are sorted by execution time and with very small differences in both energy and time values."

9..In fact, for nine out of 10 benchmark problems, the top score (for both speed and energy efficiency) came from one of the top three overall fastest and most energy-efficient languages — which didn't surprise the researchers.

10. "It is common knowledge that these top three languages (C, C++, and Rust) are known to be heavily optimized and efficient for execution performance, as our data also shows."

11.But you don't see the same order when you rank the other 24 languages by their run-time as you do when you rank them for energy efficiency. "Only four languages maintain the same energy and time rank (OCaml, Haskel, Racket, and Python), while the remainder are completely shuffled."

12And even on individual benchmark tests, there are cases where fast-performing languages are not the most energy efficient.

13. One more way to develop power-efficient software is to "teach" your application how to do things faster. You can do this by using hardware optimizations like SSE, AVX and others. For instance, by using the Quick Sync feature of the Sandy Bridge platform you can boost your code, and encode or decode video 10 x times faster.

Table 4. Normalized global results for Energy, Time, and Memory

Total					
	Energy		Time		Mb
(c) C	1.00	(c) C	1.00	(c) Pascal	1.00
(c) Rust	1.03	(c) Rust	1.04	(c) Go	1.05
(c) C++	1.34	(c) C++	1.56	(c) C	1.17
(c) Ada	1.70	(c) Ada	1.85	(c) Fortran	1.24
(v) Java	1.98	(v) Java	1.89	(c) C++	1.34
(c) Pascal	2.14	(c) Chapel	2.14	(c) Ada	1.47
(c) Chapel	2.18	(c) Go	2.83	(c) Rust	1.54
(v) Lisp	2.27	(c) Pascal	3.02	(v) Lisp	1.92
(c) Ocaml	2.40	(c) Ocaml	3.09	(c) Haskell	2.45
(c) Fortran	2.52	(v) C#	3.14	(i) PHP	2.57
(c) Swift	2.79	(v) Lisp	3.40	(c) Swift	2.71
(c) Haskell	3.10	(c) Haskell	3.55	(i) Python	2.80
(v) C#	3.14	(c) Swift	4.20	(c) Ocaml	2.82
(c) Go	3.23	(c) Fortran	4.20	(v) C#	2.85
(i) Dart	3.83	(v) F#	6.30	(i) Hack	3.34
(v) F#	4.13	(i) JavaScript	6.52	(v) Racket	3.52
(i) JavaScript	4.45	(i) Dart	6.67	(i) Ruby	3.97
(v) Racket	7.91	(v) Racket	11.27	(c) Chapel	4.00
(i) TypeScript	21.50	(i) Hack	26.99	(v) F#	4.25
(i) Hack	24.02	(i) PHP	27.64	(i) JavaScript	4.59
(i) PHP	29.30	(v) Erlang	36.71	(i) TypeScript	4.69
(v) Erlang	42.23	(i) Jruby	43.44	(v) Java	6.01
(i) Lua	45.98	(i) TypeScript	46.20	(i) Perl	6.62
(i) Jruby	46.54	(i) Ruby	59.34	(i) Lua	6.72
(i) Ruby	69.91	(i) Perl	65.79	(v) Erlang	7.20
(i) Python	75.88	(i) Python	71.90	(i) Dart	8.64
(i) Perl	79.58	(i) Lua	82.91	(i) Jruby	19.84

3. Sustainable Development

Life spans, time to failure, and disposal of old hardware should be considered. Hardware should be updated regularly. The need for regular updating of hardware, both client devices (such as desktop computers, laptops, and other mobile devices such as mobile phones) and infrastructures such as servers and network equipment. The electricity used by computers as well as the energy and resource used to provide a suitable environment. Students can be unaware of the large amount of energy needed to provide suitable physical environments (temperature and humidity) for server farms etc.

4.Survey Results

5. Advantages

1.Efficient usage of Hardware.

2. Cost reduction Sustainable IT development.

3. Efficient usage of Resources.

4. Increased Software efficiency.

5. Reduction in Time Consumption.

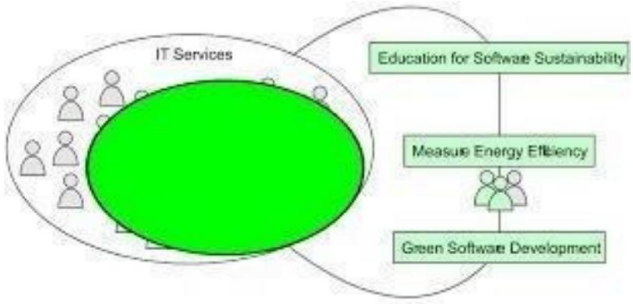
6. Positive impact on Nature.

7. Less greenhouse gas emissions.

8. Reduction of the resource depletion problem.

6.Conclusion

Basically, Green Programmer is a term recently popularized for its environmental intentions and refers to programming code that is written to produce algorithms that have minimal energy consumption. Also, the survey results showcase that, programmers are unaware of green software and the methods to minimize the energy consumption by the software. Education about green software and green coding should be inculcated in the basic curriculum of programmers.



7. References

1.Control Objectives for Information and Related Technology (COBIT 5). http://www.isaca.org/cobit/,2012

2. H. Khalid, E. Shihab, M. Nagappan, and A. E. Hassan. What do mobile app users complain about? A study on free iOS apps. Accepted to be published in IEEE Software, 2014.

3.P. Kurp. Green computing. Communications of the ACM, 51(10):11–13, 2008

4.C. Pang. Technical Summary: What do developers know about Software Energy consumption and power use? http://webdocs.cs.ualberta.ca/ hindle1/2014/green programmers/, 2013.