International Journal of Computational Research and Development (IJCRD)
Impact Factor: 4.775, ISSN (Online): 2456 - 3137
(www.dvpublication.com) Volume 2, Issue 2, 2017

COMPARISON OF SORTING ALGORITHMS BASED ON ENERGY CONSUMPTION

S. G. Balakrishnan* & Anirban Basu**
Department of Computer Science and Engineering, APS College of Engineering, Bangalore, Karnataka

Cite This Article: S. G. Balakrishnan & Anirban Basu, "Comparison of Sorting Algorithms Based on Energy Consumption", International Journal of Computational Research and Development, Volume 2, Issue 2, Page Number 74-76, 2017.

Abstract:

Use of computers, mobile phones and various hand held electronic devices are increasing rapidly in recent years. Several applications run simultaneously in these devices and some of the devices like mobile phones and laptops are battery powered. Desktop computers and servers running several applications starting from simple word processing to complex big data analytics, also consume a large amount of energy. Energy consumed mostly by hardware of the devices such as by CPU, memory units, networking interfaces and so on have become area of serious concern and the subject of Green Software Engineering have emerged to study ways for reducing energy consumption. Computer scientists and engineers are seriously looking at reducing energy consumption by various means including redesigning the algorithms. In the area of Green Software Engineering, accurate measurement of energy consumption is very important and various tools have been suggested for the purpose. In this paper, we have measured the energy consumption of various sorting algorithms using open source library jRAPL [6]. The energy consumed in joules was measured on six sorting algorithms and the values compared for different data sets. Comparison of the results is given in the paper.

Key Words: Component; Sorting Algorithms, Energy Consumption, Energy Efficient Algorithms & Green Software Engineering

1. Introduction:

Designing CPUs that perform tasks efficiently without overheating is a major consideration of nearly all CPU manufacturers to date depending upon the intended application. For example, the CPUs in mobile phones often use just a few watts of electricity, while some microcontrollers used in embedded systems may consume only a few mill watts or even as little as a few microwatts. In comparison, CPUs in general-purpose personal computers, such as desktops and laptops, dissipate significantly more power because of their higher complexity and speed.

CPUs for desktop computers typically use a significant portion of the power consumed by the computer. While energy-saving features have been instituted in personal computers, the overall consumption of today's high-performance CPUs is considerable.

Global warming, and environmental hazards are becoming subjects of great concern and studies are being undertaken on using computing resources efficiently to reduce negative impacts on the environment. Although most studies and regulations focus on hardware related measurement, analysis and control for energy consumption, it is true that most of the hardware systems have significant software components. Although the software systems do not consume energy directly, they affect the hardware utilization, leading to indirect energy consumption. Therefore it is essential to engineer the software to optimize its energy consumption.

With the use of computing with large pool of servers, it is feared that energy dissipation from these data centers is becoming dominant factors in the environmental degradation. With increase in consciousness about the environment, a new subject called Green Software Engineering [1] is emerging to handle the challenges in this area. The main reasons for reducing energy consumption in computing are for minimizing *greenhouse carbon gas emission*, and *heat dissipation.*

The Software Engineering community has recently started paying attention to sustainability and the subject of Green Software Engineering is emerging. The aim is to reduce the environmental impact caused by the software and greenness in software is an emerging quality attribute that need to be taken into account.

Therefore, design of energy efficient techniques to reduce heat generation is becoming an important issue in Computing for sustainability. Energy consumption is expected to reduce by 15 percent from total energy consumption by 2020 possibly greening through ICT [2]. Dynamic voltage and frequency scaling (DVFS) is used in recent day's computers to save energy consumption by controlling the CPU's frequency and voltage. Enhanced heterogeneous multicore processors have also been introduced to reduce energy consumption [3]. For scheduling and managing a heterogeneous processor's optimal energy-efficiency a simplified runtime platform model H-EARtH algorithm is proposed. A small number of static and runtime parameters to select the core and frequency of heterogeneous processor's [4]. There are many energy consumption and control devices exist, that include Intel's Running Average Power Limit (RAPL) and AMD's TDP Power Cap. It can be accessed through machine-specific registers (MSR) that run on Linux. Subsequently, techniques for writing and reading to them are needed for user-level measurement and control [5]. The energy consumption of Java programs running in Intel's

processor can be measured using an API called jRAPL. The highly protected MSR are accessed by operating systems using MSR kernel module in Linux [6].

With widespread use of internet for e-commerce there are billions of users doing online shopping, electronic mail, trading, weather reporting, ticket reservation and so on. Most of these applications use sorting algorithms to provide relevance details to the users in chronological order. However, energy consumption of these algorithms can vary because of its time complexity. The selection of appropriate sorting algorithms can save the energy consumption of the processing unit. It is therefore important to use the sorting algorithms which uses less energy and the paper deals with this aspect and compares the energy consumption of six sorting algorithms.

2. Sorting Algorithms used in the Study:

In this section, sorting algorithms used in various applications such as e-commerce, e-mail etc. are analysed. These algorithms have been analysed in terms of their time and space complexity but their energy consumption has not been analysed.

Selection sort [7] is noted for its simplicity, and it has performance advantages over more complicated algorithms in certain situations, particularly where auxiliary memory is limited. It has $O(n^2)$ time complexity, where n is the number of items sorted. Bubble sort [7] has worst-case and average complexity both $O(n^2)$. Bubble sort should be avoided in the case of large data size. It will not be efficient in the case of a reverse-ordered collection. Insertion sort [7] is a simple sorting algorithm that builds the final sorted array (or list) one item at a time. It is less efficient on large lists compared to advanced algorithms such as quicksort, heapsort, or merge sort. Heap sort [7] is a comparison-based sorting algorithm. It can be thought of as an improved selection sort. It has the advantage of a more favorable worst-case O(n log n) runtime. Merge sort [7] is a divide and conquer algorithm and is an efficient, general-purpose, comparison-based sorting algorithm.

In sorting n objects, merge sort has an average and worst-case performance of O(n log n). Quick sort (sometimes called partition- exchange sort) [7] is an efficient sorting algorithm, serving as a systematic method for placing the elements of an array in order. Quicksort can operate in-place on an array, requiring small additional amounts of memory to perform the sorting. Mathematical analysis of quicksort shows that, on average, the algorithm takes O(n log n) comparisons to sort n items [7].

These six algorithms were analysed based on their energy consumption for different data sets.

3. Experimental Results and Analysis:

An open source library jRAPL is used because it allows energy or power consumption to be reported at a fine-grained manner, e.g., monitoring DRAM, CPU core and CPU uncore separately. Energy consumption of CPU core was considered rather than DRAM and CPU uncore. Experiments were conducted on machine with intel core i5, 3.40GHz, with 4GB of DDR3 memory. Running ubuntu 13.10, oracle 9i database and Java Development Kit (JDK) version 1.6.0 26. For the java virtual machine (JVM), the parallel garbage collector is used, and just-in-time compilation is enabled to be realistic with real-world Java based sorting algorithms applications.

Four datasets are used and each is a multiple of 500 data items. The data items are generated randomly Figure 1 and Figure 2 illustrate respectively Screenshot of Quicksort & Selection sort algorithms with a dataset of 500 data items.

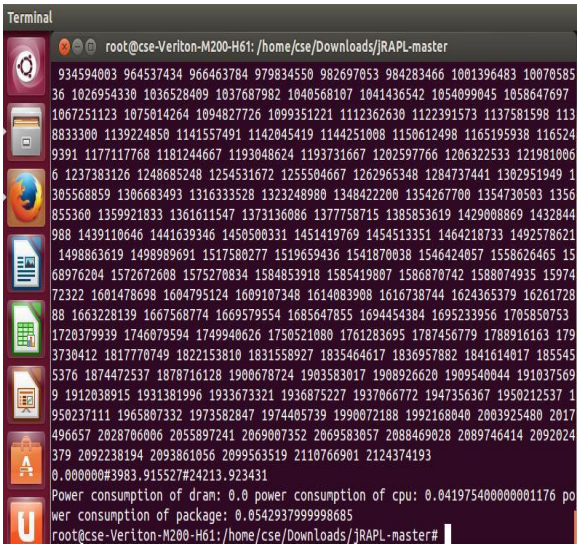

Figure 1: Screenshot of Quicksort algorithm with a dataset of 500 data items

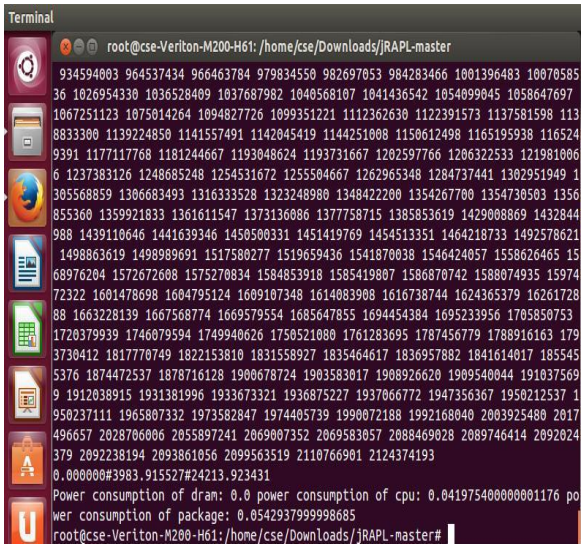

Figure 2: Screenshot of Selection sort algorithm with a dataset of 500 data items

The energy consumed in different sorting algorithms is compared and given in Figure 3.

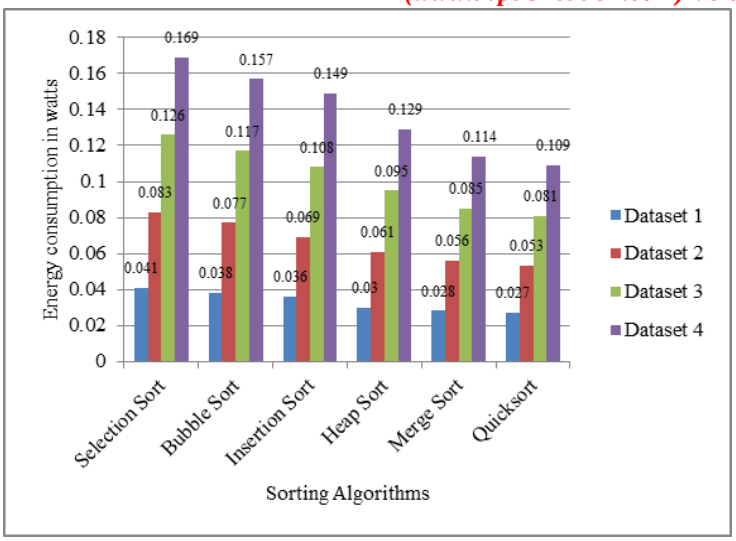

Figure 3: Energy Consumption in Joules of different Sorting Algorithms

4. Conclusion:

Accurate measurement and analysis of energy consumption is very important to software engineers while developing the applications. In this paper, we have measured the energy consumption of various sorting algorithms written in Java, using open source library jRAPL. The energy consumed by six sorting algorithms has measured and the experimental results shows that Quicksort consumes least energy among the six algorithms for all different datasets while the most energy consumed is Selection sort. Detailed analysis of energy consumption of other algorithms is being carried out.

5. Acknowledgment:

The work has been carried out under VGST Project on Green Software Engineering of the Government of Karnataka,

6. References:

1. C. Calero and M. Piattini, “Introduction to Green in Software Engineering”Springer International Publishing Switzerland, 2015.
2. Smart 2020: Enabling the Low Carbon Economy in the Information Age, tech. report, Climate Group, 2008; www.smart2020.org/_assets/fi les/02_Smart2020Report.pdf.
3. N. Rajovic et al., “Supercomputing with Commodity CPUs:Are Mobile SoCs Ready for HPC?,” Proc. Int’l Conf. High PerformanceComputing, Networking, Storage, and Analysis (SC13), 2013, article no. 40.
4. EfraimRotem et al., “H-EARtH: Heterogeneous Multicore Platform Energy Management”, IEEE Computer, Volume: 49, Issue: 10, Oct. 2016, pp. 47-55.
5. Ryan E. Grant et al., “Standardizing Power Monitoring and Control at Exascale” IEEE Computer, Volume: 49, Issue: 10, Oct. 2016, pp. 38-46.
6. Kenan Liu, “Data-Oriented Characterization of Application-Level Energy Optimization” , Springer Link LNCS, 2015, volume 9033, pp 316-331.
7. https://en.wikipedia.org/wiki/