International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-3 Number-1 Issue-8 March-2013

Application of Green computing in Framing Energy Efficient Software Engineering

[1]Aritra Mitra, [2]Riya Basu, [3]Avik Guha, [4]Shalabh Agarwal, [5]Asoke Nath
[1,2,3,4,5] Computer Science, St. Xavier's College (Autonomous), Kolkata, West Bengal, India

Abstract

Green computing and energy saving is now a very important issue in Computer science and information technology. Due to tremendous growth in information technology now the big challenge is how to minimize the power usage and how to reduce the carbon foot print. Green computing is now a prime research area where the people are trying to minimize the carbon footprint and minimum usage of energy. To minimize the usage of energy there are two independent approaches one is designing suitable hardware and the second one is to redesign the software methodology. In the present paper the authors have tried to explore the software methodologies and designs that can be used today to save energy. The authors have also tried to extend mobile platform battery time as well as the various tools that support the development of energy-efficient software.

Keywords

Green computing, software methodology, platform, energy efficient

1. Introduction

To save energy in any hardware device is now a challenging issue. The scientists are trying to improve the hardware design so that it will take minimum power from the battery. This is done to reduce the power consumption in any battery driven devices such as mobiles, transistors etc. The last few years there has been a tremendous improvement in digital logic circuits which consumes minimum power from the battery or from the main. The Battery technologies have also gradually improved for last one decade. Microprocessors have low power states similarly the display units have dramatically improved their power consumption. However, there is still scope for improvement in power management in battery system. Software can play a very important role in reducing the power used by electronic devices and thus can extend battery time for various devices. This kind of technology is often referred to as green software engineering. In the present paper the authors have tried to explore the concept of green software technology.

2. Some Terms in designing Energy Efficient Software

In this section we will be discussing some important terms which are important in software engineering. These terms are often used while calculating the energy efficiency in performing a specific job.

(a) Joule – It is the international standard unit of energy measurement.
(b) Energy –Energy is defined as the capacity to do work. A device may be called energy efficient provided it requires less energy than a device which takes more energy and less energy efficient. In the present paper, we use the term energy to mean the amount of joules required to carry out a specific task. For example the energy required to lift a 100 gram object 1 meter against the pull of earth's gravity is about 1 Joule.
(c) Power – It is defined as the amount of energy consumed per unit of time, typically measured in Watts, where on watt equals 1 Joule per second. For example, a light bulb rated at 60 Watts consumes 60 Joules in one second. Notebook computers running at their highest energy state are rated between 40 to 60 Watts, but on average consume far less.
(d) Heat – Heat may be looked upon as form of energy whose absorption makes a body hot and abstraction makes a body cold. The engineers try to minimize this in computer design as too much heat means more cooling is required (typically by a fan) which requires more energy. So the H/W and S/W should be designed in such a way that the heat generation is minimum.

3. To design Energy Efficient Software Application

A key tactic for achieving energy-efficient software performance from an application developer's point of view is the effective handling of sleep state transitions. A few general rules can go a long way toward accomplishing this goal-for example:

- Designing of applications that allow screens of the computers to darken and disks to remain idle by avoiding behaviours that unnecessarily prevent systems from remaining in a sleep state. Moving from sleep states to full activity states requires some energy, thus, develop algorithms so that idle processors do not start working unnecessarily.
- Eliminating code wherever possible that keeps processors from transitioning to sleep states.
- Employ development frameworks that allow an application to be respectful of sleep status and resilient in handling nonessential workloads.
- To prevent users from disabling sleep, become more context aware, and take steps to ensure that systems don't enter sleep states when users are passively interacting with them (e.g., watching or listening).
- Develop power-aware strategies for handling timers and looping. Investigate the use of compiler switches that unroll deterministic loops, and make other adjustments that reduce the overall number of instructions executed (e.g. Removing, polling).
- Use energy-aware tools for identification of patterns of processors used in your applications.

4. Some Energy Saving Software Techniques

This section will discuss briefly about the terms related to energy and practices that should be adopted for less consumption of power. We will try to explore why a software developer needs to know about energy efficient software techniques. While most of the energy saving features in a platform are transparent to the software developer and applications have very little direct control, the behavior of the software has significant influence on whether the energy saving features built into the platform are effective. Well behaved software allows the energy saving features to work. Poorly behaved software inhibits the energy saving features and leads to lower battery life and higher energy costs.

Before we describe the software energy-saving techniques, it is important to understand the distinction between active and idle software. Active software is software that is fulfilling its intended purpose such as computing a spreadsheet, playing music or a movie, uploading photos to a web site, browsing the internet, etc. In all of these cases, there is a workload that the CPU or GPU is busy working on. Idle software is software that is essentially running but waiting for an event at which point it will become active. Examples of idle software are: a browser that is started but has not been pointed to a web site, an open word processor program in the background or whose window is minimized, or an instant messaging program that is running but not sending or receiving a message.

A) Computational Efficiency

The goal of computational efficiency is to complete a task as fast as possible. Intuition tells us that if the CPU can accomplish the task in fewer instructions or by doing work in parallel in multiple cores, and then drop the CPU to a low-power state, then the overall energy required to complete the task will be lower. One approach to achieve this will be to write the best algorithm and data structure for the particular problem.

(i) *To introduce Efficient Algorithms*

Efficient algorithms and data structures are the most important research field in computer science. Considerable effort has gone into research to find more efficient means to solve problems and to investigate and document the corresponding time and space tradeoffs. While optimizing specific algorithms is not an area of interest for us, we can conclude from computer science theory that the choice of algorithms and data structures can make a vast difference in the performance of an application. All other things being equal, using an algorithm that computes a solution in $O(n \log n)$ time is going to perform better than one that does the job in $O(n^2)$ time. For a particular problem, a stack may be better than a queue and a B-tree may be better than a binary tree or a hash function. The best algorithm or data structure to use depends on many factors, which indicates that a study of the problem and a careful consideration of the architecture, design, algorithms, and data structures can lead to an application that performs better and consumes less energy. For a detailed study of the analysis of algorithms see [1, 2, 3].

(ii) To introduce Multi-threading

For mobile platforms, power consumption has always been one of the major areas of importance. With multithreaded applications, the job at hand may be able to finish faster than single-threaded applications. As a result, the boost in performance may result in power savings as system resources will be used for less time, as compared to a single-threaded version. There are other considerations introduced with multithreading an application, such as the effects on power/performance when the threads in the application are imbalanced (when one thread does significantly more work than the other threads), differences in CPU utilization of threads(for example, one thread might consume 100 percent of the CPU while the other threads might consume 10-20 percent of the CPU) , and when the threads are running on a single core rather than running on multiple cores. [4]

(iii) To re-design Compilers, Performance libraries and Instruction sets

Another way to achieve better computational efficiency and performance is to use an optimizing compiler, performance libraries, and make use of advanced instruction sets. Many compilers now offer OpenMP directives that look for opportunities for parallelism. OpenMP is a portable, scalable model for developing parallel applications. For example, the Intel Professional Edition Compilers offer support for creating multi-threaded applications. Features include advanced optimization, multi-threading, and processor support that for processor dispatch, vectorization, auto-parallelization, OpenMP, data prefetching, and loop unrolling, along with highly optimized libraries. Microsoft Visual Studio 2005 also includes the OpenMP portable threading library. Performance libraries such as Intel Performance Primitives (IPP) contain highly optimized algorithms and code for common functions such as video/audio, encode/decode cryptography, speech encoding and recognition, computer vision, signal processing, etc. The Intel Math Kernel libraries provide highly efficient algorithms for many math and scientific functions including fast Fourier transforms, linear algebra, and vector math. Advanced instruction sets help the developer take advantage of new processor features that are specifically tailored to certain applications. For example, the Intel Streaming SIMD Extensions (SSE) 4.1 is a set of new instructions tailored for applications that involve graphics, video encoding and processing, 3D imaging, gaming, web servers, and application servers. Optimizing with these instructions will deliver increased performance and energy efficiency to a broad range of 32-bit and 64-bit applications.

B) Data Efficiency

Data efficiency reduces energy costs by minimizing data movement [5]. Data efficiency can be achieved by designing:

1) Software algorithms that minimize data movement
2) Memory hierarchies that keep data close to processing elements
3) Application software that efficiently uses cache memories

As we demonstrated with computational efficiency, data efficiency delivers performance benefits and saves energy.

C) Context Awareness

Context awareness was first introduced by Schilit in 1994[6]. The objective is to create applications that can respond or adapt to changes in the environment. For the physical environment this requires sensors and the ability to generate events or state changes to which the applications can react. For example, many notebook computers respond to the change from AC to battery power by automatically dimming the display. Some notebook PCs park the hard drive heads when sensors detect that the device is falling – to avoid a head crash. Some applications may write cached data to flash when the battery is getting critically low. Context may also apply to a user's situation. For example, a software application acting as an intelligent travel assistant may find and offer alternative flights when it's apparent based on your location (stuck in traffic) that you have missed your scheduled departure. With respect to power, applications should respond to system changes and take actions that will conserve energy – or at least offer options to the user .These events include AC vs. battery, charge remaining on the battery, and the state of various energy consuming devices such as Wifi and Bluetooth radios.

Some Recommendations

It's clear from the experiments that various game and platform settings can influence the amount of power consumed and when set appropriately, can extend battery time. When running on battery power, the following actions can extend play time:

- Reduce display brightness
- Reduce game resolution to lower value

- Reduce the frame rate by capping it to lower value
- Reduce game quality options to lower values

In addition, if WiFi and/or Bluetooth radios are not needed, power can be saved by turning them off. While these recommendations apply to end users, the game options must first be implemented by the game developers. This could be as simple as providing a single choice – "Optimized for Performance" or "Optimized for Power." Games can be even more user-friendly when they know when to offer choices to the gamer – such as when the user switches from AC to Battery.

D) Idle Efficiency

Idle Power for mobile platforms is defined as the power consumed when the system is running in ACPI S0 state (S3-Sleep or S4-Hibernate) with software applications & services running but not actively executing workloads. In this state, there should be minimal background activity. The challenge is to lower the idle floor by improving application idle efficiency which will lead to a significant increase in battery life. This also benefits average power scenarios and helps all but the most demanding (TDP-like) workloads.[7]

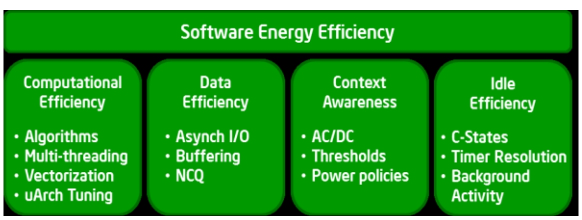

Figure 1: Figure showing the various energy saving software techniques

5. Operating Systems

In addition to the previously mentioned techniques of saving power, the operating system of a particular system also plays an important role. Such as for creating power aware applications in vista, Microsoft has laid emphasis on the following things-

- Sleep mode and resume operations mode
- Preventing System Idle Timeouts
- Extended battery life designs.
- Responding to Common Power Events
- Entertainment and Media PC Scenario designing.
- Using Power Management APIs from Managed Code.
- Testing Applications for Power Management

6. Conclusion and Future Scope

Green technologies and consumer demand for longer battery life has become the need of the hour. The demand for higher performance and new usage models will also continue to grow. Energy-efficiency will be crucial for the computing industry in the future both to increase battery life for mobile platforms and to reduce energy expenses for desktop and server platforms. Software behavior can play an important role on platform power consumption and battery life.

Modern processors and platforms have many energy saving features, particularly for performance and the ability to enter low-power states when idle. In this paper we have discussed some methods that can be used to increase the energy efficiency and extend the battery time. To get the most benefit, developers should do the following:

1) Take advantage of performance features by emphasizing Computational Efficiency.
2) Be frugal with data movement to improve Data Efficiency.
3) Implement intelligent application behaviors by exploiting Context Awareness.
4) Seriously consider the impact of software at idle to improve Idle Efficiency.

Acknowledgment

We are grateful to the faculty members of Computer Science Department of St. Xavier's College (Kolkata) for helping us to venture into various topics of computing like Green Software Engineering. It has immensely helped us to know about upcoming fields of research in computing.

References

[1] Donald E. Knuth, Selected Papers on the Analysis of Algorithms, 2000.
[2] Donald E. Knuth, Art of Computer Programming, 4 Volumes, 2005.
[3] Robert Sedgewick and Philippe Flajolet, An Introduction to the Analysis of Algorithms, 1996.
[4] Rajshree Chabukswar, Maximizing Power Savings on Mobile Platforms.
[5] Arnout, Guido, Data-Efficient Software and Memory Architectures are Essential for Higher Performance and Lower Power. *Information Quarterly*, V.4, No.3, 2005.

[6] B. Schilit, N. Adams, and R. Want. "Context-aware computing applications". IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'94), Santa Cruz, CA, US: 89-101., (1994).
[7] MobileMark 2005 and 2007.
[8] Maximizing Power Savings on Mobile Platforms, Rajshree Chabukswar.
[9] R.M. Ramanathan, *Extending the World's Most Popular Processor Architecture* .
[10] Dr. Bob Steigerwald and Abhishek Agarwal, Developing Green Software.
[11] Dr. Bob Steigerwald, Rajshree Chabukswar, Karthik Krishnan and Jun De Vega, Creating Energy Efficient Software.

Aritra Mitra is a student of computer science department at St. Xavier's College (Kolkata). His interest is in various web based technologies and green computing and is presently working on an online college account system for students.

Riya Basu is student of computer science department at St. Xavier's college (Kolkata). Her interest is in web based technologies. She is currently working on an electronic attendance system in college.

Avik Guha is a student of computer science department at St. Xavier's college (Kolkata). His interest is in cryptography and green computing. He is currently working on an electronic attendance system in college.

Shalabh Agarwal is an assistant professor in computer science department. He is busy with different research projects. His primary research area is green computing, e- learning, software engineering. He has published several papers in national and international journals and conference proceedings.

Asoke Nath, is an associate professor in department of computer science. Apart from his teaching assignment he is involved with various research works in cryptography, steganography, green computing, e-learning. He has presented papers and invited tutorials in different international and national conferences in India and abroad.