A Survey on Energy-Efficient Data Management

Jun Wang, Ling Feng
Dept. of CS&T, Tsinghua Univ., Beijing, China
wjun09@mails.tsinghua.edu.cn
fengling@tsinghua.edu.cn

Wenwei Xue, Zhanjiang Song
Nokia Research Center, Beijing, China
wayne.xue@nokia.com
zhanjiang.song@nokia.com

ABSTRACT

Energy management has now become a critical and urgent issue in green computing. A lot of efforts have been made on energy-efficiency computing at various levels from individual hardware components, system software, to applications. In this paper, we describe the energy-efficiency computing problem, as well as possible strategies to tackle the problem. We survey some recently developed energy-saving data management techniques. Benchmarks and power models are described in the end for the evaluation of energy-efficiency solutions.

1. INTRODUCTION

Now and in the future, green computing will be a key challenge for both information technology and business. Green computing aims at environmentally sustainable computing and responsible use of computers and related resources. Murugesan *et al.* defined the field of green computing as "*the study and practice of designing, manufacturing, using, and disposing of computers, servers, and associated subsystems such as monitors, printers, storage devices, and networking and communications systems efficiently and effectively with minimal or no impact on the environment* [28]."

With the limited primary sources of energy and rapid climbing of energy demanded by computing, the commitment to reduce power consumption and environmental impact becomes increasingly important. *Energy efficiency* thus constitutes a focal point for green computing.

In fact, most people in the world today are aware of the energy problem at a high level: even if our primary sources of energy are running out, the demand for energy in both commercial and domestic environments is increasing, and the side effect of consistent energy use influences negatively our global environment. Based on the report of the US Environmental Protection Agency, "*the servers and data centers in USA alone consumed about 61 billion kilowatt-hours (kWh) at a cost of $4.5 billion, which was about 1.5% of the total U.S. electricity consumption in 2006, and this energy consumption is expected to double by 2011 if continuously powering computer servers and data centers using the same methods* [11]." Xu *et al.* showed that electricity consumed by computer servers and cooling systems in a typical data center contributes to around 20 percent of the total ownership cost, equivalent to one-third of the total maintenance cost [42]. When a data center reaches its maximum provisioned power, it has to be replaced or augmented at a great expense [33]. In the very near future, energy efficiency is expected to be one of the key purchasing arguments in the society.

Nowadays, power and energy have started to severely constrain the design of components, systems, computing clusters, data centers, and applications. Better equipment design and better energy management policies are desirable to address these concerns.

More and more computer designers and users are concerned about energy-efficient computing. In this survey article, we overview these great efforts, with an emphasis on energy-efficient data management, which was ignored and starts to get attention very recently.

The remainder of this paper is organized as follows. In Section 2, the energy-efficiency computing problem and some guidelines to tackle the problem are discussed. In Section 3, we overview work done on energy-efficient data management. Evaluation techniques on energy performance including benchmarks and power models are described in Section 4. We conclude the paper in section 5.

2. FUNDAMENTALS OF ENERGY EFFICIENT COMPUTING

In this section, after a brief description of the energy-efficiency problem, we outline some feasible ways to tackle the problem.

2.1 Energy-Efficiency Problem

Energy consumption can be generally defined as:

$$Energy = AvgPower \times Time$$

where $Energy$ and $AvgPower$ are measured in $Joule$ and $Watt$, respectively, and $1\ Joule = 1\ Watt \times 1\ Second$.

Energy efficiency is equivalent to the ratio of performance, measured as the rate of work done, to the power used [37] and the performance can be represented by response time or throughput of the computing system.

$$\begin{aligned} Energy\ Efficiency &= \frac{Workdone}{Energy} = \frac{Workdone}{Power \times Time} \\ &= \frac{Performance}{Power} \end{aligned} \tag{1}$$

The main approach towards energy-efficiency is efficient power management. According to equation (1), there are two ways to enhance energy-efficient computing: either improving the performance with the same power, or reducing power consumption without sacrificing too much performance. For energy-efficient systems, while maximal performance for some tasks (or the whole workload) is still desirable in some cases, the systems must also ensure the energy usage is minimized. Preferably, a computing system consumes the minimum amount of energy to perform a task at the maximal performance level [10].

Note that the relationship between performance and energy efficiency is not mutually exclusive. A maximal performance could also be achieved by deactivating some resources or lowering certain individual performance without affecting the workload's best possible completion time or throughput in order to optimize energy usage. Brown *et al.* treated energy efficiency as an optimization problem [10]. To minimize the total energy, an energy-efficient system must adjust the system's hardware resources dynamically, so that only what is needed to execute tasks is made available. Rivoire *et al.* pointed out two major complementary ways to solve the energy-efficiency problem: either building energy efficiency into the initial design of computer components and systems, or adaptively managing the power consumption of systems or groups of systems in response to changing conditions related to the workload or environment [36].

2.2 Solution Guidelines

To deliver effective solutions to the energy-efficiency problem, the following six considerations can be taken as the solution design guidelines.

1) *Comprehensive Examination of System Components.* To save power consumption, we shall first investigate where the power is spent and how to optimize the power usage. Within a computer system, there are generally five energy consumers, namely, *processor, disk, memory, I/O devices,* and *chipset.* Achieving energy-efficiency requires improvements in the energy usage profile of every system component.

2) *Adopting Power-Manageable Hardware Components.* Adopting power-manageable hardware components could help improve energy-efficiency. For example, the voltage of hardware components can be increased or decreased through dynamic voltage scaling (DVS), which is a power management technique in computer architecture, depending upon circumstances. Dynamic voltage scaling to decrease voltage is known as undervolting, and this situation can conserve power [12]. In addition, employing small form factor disk drives, solid state disk drives, large memory configurations, low power processors and memories could decrease power consumption [31]. HP and IDC also estimated that about 69 percent energy reduction can be achieved within a three-year period for IT organizations that migrate to blade self-contained architecture, where blades can span from servers and storage devices to workstations and virtual desktops [20, 1].

3) *Building Power Models for Computing Systems.* Also, one needs to know how a computing system is constructed and how an energy-efficient system operates. It is important to construct a power model that allows the system to know how the power is consumed, and how the system can manipulate and tune that power [10].

4) *Understanding and Measuring System Performance.* To counter for performance with the least power consumption, computing systems must have ways to timely understand and measure system performance related to task execution under different dynamic workloads.

5) *Constructing Energy Optimizers.* The system must accommodate an energy optimizer component, which is responsible for an energy-efficient hardware configuration throughout the system operation at all times. The optimization approaches may be based on either heuristic or analytical techniques, as indicated by Brown *et al.* in [10].

6) *Reducing Peak Power.* Barroso *et al.* explained that current desktop and server processors can consume less than one-third of their peak power at very low activity modes, which can thus save around 70 percent of peak power [7]. Tsirogiannis *et al.* indicated that almost 50 percent of peak power is actually consumed at idle [37].

3. ENERGY-EFFICIENT DATA MANAGEMENT

Over the past few decades, performance remains as the main goal of database management systems (DBMSs). In light of an increasing concern about energy, energy management starts to draw attention of the database community. The 2008 Claremont report on database research emphasized explicitly the importance of *power-aware DBMSs that limit energy costs without sacrificing scalability* [4]. To achieve energy-efficient DBMSs, solutions from both hardware and software perspectives are desirable. That is, DBMSs not only need to accurately estimate and online measure the hardware energy consumption characteristics under both static and dynamic loads, but also dynamically adapt and tune data management strategies to meet response time and energy goals when initial prediction for energy consumption deviates from the real case [25, 18, 42, 41, 37].

In this section, we describe efforts on energy-efficient data management at server, sensor, and mobile sides.

3.1 Server Side

Research on energy-efficient data management at server side so far mainly considers the most fundamental database operation - query processing and optimization.

3.1.1 Hardware-Based Approach

Under the assumptions that a CPU consumes a significant amount of power compared to other components in a database system, and the performance of speed and energy of hard disk drive is close but does not exactly follow the change in the read block size, Lang *et al.* proposed a PVC (Processor Voltage/Frequency Control) mechanism to trade energy consumption for performance [25]. It aims to execute instructions at a lower processor voltage and frequency by leveraging the ability of modern processors. The CPU frequency is determined by two settings: the front side bus (FSB) speed and the CPU multiplier. There are basically two methods to modulate CPU frequency, namely, p-state transitioning and underclocking. P-states are characterized by the combination of CPU multiplier and CPU voltage settings. Underclocking has the ability to more finely tune the CPU speed by slowing the FSB speed. [25] adopted underclocking to modulate the CPU frequency, and their experiments showed that PVC can be used to reduce the CPU energy consumption by 20% and 49%, while incurring 6% and 3% response time penalties on MySQL and a commercial DBMS, respectively.

3.1.2 Software-Based Approaches

Hardware-based approaches constitute only part of solutions. Considering hardware heterogeneity and limited power knobs that most hardware offers today, data management software shall play an effective role in energy optimization as well. Physical data independence and query optimization of DBMSs do provide opportunities for software-level control over power-performance tradeoffs [18].

Harizopoulos *et al.* categorized three kinds of software-based approaches for reducing energy in DBMSs [18].

(1) *Energy-aware optimization*, i.e., using existing system-wide knobs and internal query optimization parameters to achieve the most energy-efficient configuration for the underlying hardware. Xu *et al.* gave a strategy to find query plans with low power costs [42, 41]. To do that, a static power profile for each basic database operation in query processing is defined and maintained as system parameters of DBMSs. The power cost can be obtained from the specifications of hardware components and divided by related estimated time through an iterative approach. The power cost of a plan can be calculated from those of the higher-level operations, containing such basic operations like CPU power cost per tuple/indexed tuple, power cost for reading/writing one page without buffering, and so on. Different power cost functions can thus be constructed for accessing single relation via different access methods and join operations.

(2) *Resource use consolidation*, i.e., shifting computation and relocating data to consolidate resource use in time and space. Whenever system resources are not fully utilized, the system may allow other concurrent tasks to utilize the resources or allow the resource to enter a suspended or reduced power mode to save energy.

(3) *Redesign software components* to minimize energy use, reduce code bloat, and sacrifice certain properties (or allow under-perform in certain metrics) to improve energy efficiency. For instance, Lang *et al.* proposed a QED (Improved Query Energy-efficiency by Introducing Explicit Delays) mechanism, which uses query aggregation to leverage common components of queries in a workload. In QED, queries are delayed and placed into a queue on arrival. When the queue reaches a certain threshold, all the queries in the queue are examined to determine if they can be aggregated into a small number of groups, such that queries in each group can be evaluated together [25]. On a workload with sim-

ple selection queries on MySQL, QED saves 54% of the CPU energy consumption while increasing the average query response time by 43%.

3.1.3 Tradeoff Between Energy and Performance

For getting the greatest energy efficiency in DBMSs, we must find the relationship between energy (or power) and performance. It is interesting to note two different opinions in recent research work.

- [42, 25, 18] claimed that energy efficiency and performance are two different optimization goals, and there exists the tradeoff of energy efficiency and performance.
- [37] claimed that energy efficiency and performance are consistent, and it said that "*within a single node system (intended for use in scale-out architectures), the most energy-efficient configuration is typically the highest performing one.*"

The cause of appearing the above two different statements may be their different assumptions and different estimation methods used. In the first line of work, the baseline power (i.e., idle power) is not included in calculating the energy cost of the DBMS after processing the workload. The second work considered CPU power only rather than the system's overall active power. By experiments, they found out that the CPU power does not vary linearly with CPU utilization (which is also the number of cores in the multi-core machine), and utilization is a poor proxy for CPU power. The CPU power used by various operators can vary up to 60%, even when they have the same utilization. By measuring the power of system components from idle to full utilization, almost 50% of peak power is consumed at idle regardless of query complexity and strategy. This fixed power cost adds a large constant term to the denominator in energy-efficiency equation (1), which makes all subsequent relative power increases worth the added performance, especially when the dynamic power range is small [37]. However, in DBMSs, there do exist queries that require many CPU operations, as well as I/O-intensive queries. Considering CPU alone, the margin of improvements might be greatly reduced by factoring in the power costs of all components.

3.2 Sensor Side

Nowadays, wide applications of sensors in health care, agri-food, environmental and security sectors call for effective sensor data processing and management techniques. As the power source of sensors often comes from a battery with a limited energy, reducing energy consumption of sensors is critical to provide a long enough lifetime service and avoid inconvenient replacement of the battery in a hostile or unpractical environment. For a sensor node, the communication cost is often several orders of magnitude higher than the computation cost, and in wireless sensor networks, the majority of the energy is actually consumed for sensor's communication rather than computation. Reducing communication workloads among sensor nodes is apparently the most effective energy-efficient operation.

Anastasi *et al.* have made a very good comprehensive survey on existing energy-efficient sensing techniques, including *duty cycling*, *data-driven*, and *mobility-based schemes*. Detailed description of each scheme can be found in [5].

1) *Duty cycling schemes* are oblivious to data sampled by sensor nodes. Radio transceiver nodes are put in a sleep mode whenever communication is not required. Duty cycling schemes can be achieved through two different and complementary approaches, i.e., *topology control* and *power management*. The topology control mechanism dynamically adapts the network topology based on the application needs so as to allow network operations while minimizing the number of active nodes. The power management mechanism aims at certain sensor nodes, which can sleep or wake-up by an *on-demand, scheduled rendezvous,* or *asynchronous* protocol [5].

2) *Data-driven schemes* are designed to reduce the amount of sampled data without sacrificing sensing accuracy required by applications through *data reduction* and *energy-efficient data acquisition. Data reduction* approaches address the case of unneeded samples. Techniques developed involve in-network processing, data compression, and data prediction. *Energy-efficient data acquisition* approaches mainly focus on reducing the energy spent in sensing by means of adaptive, hierarchical, and model-based active sampling [5]. For instance, to reduce the communication cost, Jain *et al.* proposed a Dual Kalman Filter architecture as a general and adaptive filtering solution to the stream resource management problem [21]. The architecture lays two equal Kalman Filters between clients and servers. The dual filters predict future data values. Only when the filter at the remote source fails to predict future data within the precision constraint, the sensor sends update value to the server.

3) Mobility has been considered as an alternative solution for energy efficient data collection in wireless sensor networks. The sensors are equipped with mobilities for changing their location. *Mobility-based schemes* can be classified as mobile-sink and mobile-

relay schemes, depending on the type of the mobile entity [5].

3.3 Mobile Side

Advancement in computing and communication has led to an increased use of a large number of mobile devices. Mobile devices are equipped with more computing and sensing facilities include GPS, WiFi, Bluetooth, accelerometers, audio, video, light sensors, and so on. These embedded sensors in mobile devices are also major power consumers, while the devices have still limited power sources such as batteries. Recently, the problem of energy-efficient data management on mobile devices has received much attention. Whang *et al.* proposed to support functionalities with low power consumption, which is also a crucial requirement for a ubiquitous database management systems [39].

Energy-efficient techniques developed at the mobile side center around *position tracking, continuous context monitoring,* and *complex event detection.*

1) *Position tracking* is an important feature of a modern mobile device. The most common method is to use GPS. However, GPS is extremely power hungry. To minimize energy consumption and improve accurate, Kjærgaard *et al.* and Paek *et al.* proposed to periodically duty-cycle GPS [24, 30]. The idea is to use historic information to estimate and predict users' movement while concurrently utilizing other sensors such as accelerometer and Bluetooth. In a mobile broadcast environment, index has extensively been adopted to support efficient location-based data access and query [43, 13, 40], since efficient indexing structures can contribute significantly to reduce the tuning time, which is frequently used to estimate the power consumption of a mobile client.

2) For *continuous context monitoring*, Kang *et al.* proposed a middle-tier framework between applications and embedded sensors in a mobile environment [23]. Wang *et al.* also presented an energy efficient mobile sensing framework to automatically recognize user state and detect state transition [38]. These two works also turned on a minimum set of sensors and used appropriate sensor duty cycles to reduce unnecessary, expensive computation and communication in the context monitoring process for energy savings.

3) With the widespread use of wireless connectivity and end-user mobile devices, *complex event detection* can be carried out by mobile devices. Considering the interaction between front-end and server, Neophytou *et al.* proposed three power-aware query operator placement algorithms that determine which part of a continuous query plan is executed at the stream management server and which part is executed at the users' wireless devices [29]. Gredik *et al.* also presented a distributed real-time approach to monitor moving object. They utilized the computational power at mobile objects to alleviate server-side load and communication cost [16].

4. EVALUATION METHODS

How to compare different energy-efficient computing methods against each other and how to estimate the developed methods whether or not they correspond to reality constitutes another important question. Substantial efforts on evaluation metrics and models for energy-efficiency computing have also been made in both academia and industry.

4.1 Models and Metrics

The real power consumption of a specific system depends on many factors, such as workload, system balance, and environmental parameters. Measuring power consumption needs accurate power and thermal models on individual components, systems, data and computing centers, and applications.

Three different types of modeling approaches exist in the literature, including *simulation-based*, *detailed analytical*, and *high-level black-box* approaches [33, 34].

1) Considering the difficulty in obtaining detailed knowledge about many components in a full system, *simulation-based* approaches intend to model individual components rather than the whole system or a collection of systems by simulation [17, 19, 9].

2) Without simulation, *detailed analytical* approaches periodically collect hardware and software metrics [6, 22]. For example, Bircher *et al.* used microprocessor performance counters for online measurement of complete system power consumption [8]. Xu *et al.* presented a power model to accurately measure the energy costs of database query execution plans [42, 41] with the hypothesis that *the peak power consumption of an entire system during the measurement interval is identical to the aggregate of the individual nameplate power consumption.*

3) *High-level black-box* approaches construct a real-time model by fitting a model to the real-time metrics collected without relying on implementation knowledge. For example, Economou *et al.* used a one-time calibration phase to generate a power consumption model by correlating AC power measurements with user-level system utilization metrics at a system level [14]. Fan *et al.* aggregated power usage of large collections of servers for different classes of applications over history data [15]. Meisner *et al.* incorporated suspending and waking transitions

to the power model [27]. Lang *et al.* proposed a mathematical model for the energy consumption of a MapReduce cluster, which adopted the workload characteristics and hardware characteristics as abstract meta-models [26]. Poess *et al.* developed a power consumption model based on data readily available in the TPC-C full disclosure report of published benchmarks [31].

4.2 Benchmarks

Researchers, governmental agencies, and industry standard consortia for performance measurements, including Transaction Processing Performance Council (TPC), Standard Performance Evaluation Corporation (SPEC), and Storage Performance Council (SPC), have also actively developed benchmarks to measure energy consumption of computer systems. Poess *et al.* provided a very comprehensive overview of the currently available energy benchmarks, and analyzed their commonalities and differences along various dimensions, including hardware components, workload and type of application, along with metric attributes and accuracy and calibration requirements [31, 32].

Basically, there are two types of energy benchmarks being developed so far. They are *specialized benchmarks* and *extended benchmarks* with additional energy metrics added to the existing benchmarks.

For example, TPC established a working group which adds energy efficiency metrics to all its benchmarks. The primary metric, reported by TPC-Energy [3], is in the form of "*Watts per Performance*" for the overall System Under Test (SUT), where the performance unit is specific to each TPC Benchmark. SPEC presented the SPECpower_ssj2008 [2] benchmark, which examined the relationship of power and performance and power consumption for servers at different performance levels, spanning from 100 percent utilization to idle in 10 percent segments, over a set period of time [32].

Rivoire *et al.* developed a benchmark called *JouleSort* for evaluating energy efficiency of various sorting algorithms. It used the same workload as the other external sort benchmarks, but its metric covered total energy, which is a combination of power consumption and performance. *JouleSort* is now an I/O-centric benchmark that measures the energy efficiency of systems at peak use [35].

5. CONCLUSION

In this survey, we present the concept and challenges of energy-efficiency problem in green computing. We focus on energy-efficient data management at server, sensor, and mobile sides in this paper. Various power models and energy benchmarks are also presented in the paper.

Acknowledgment

The work is funded by National Natural Science Foundation of China (60773156, 61073004) and Chinese Major State Basic Research Development 973 Program (2011CB302203-2).

6. REFERENCES

[1] HP Blade System c-Class portfolio. *http://hl8004.wwwl.hp.com/products/blades/components/c-class-components.html.*

[2] SPECpower_ssj2008. *http://www.spec.org/power_ssj2008/.*

[3] TPC Energy Specification Version 1.2.0. *http://www.tpc.org/tpc_energy/spec/.*

[4] R. Agrawal, A. Ailamaki, P. Bernstein, E. Brewer, M. Carey, S. Chaudhuri, A. Doan, D. Florescu, M. Franklin, H. Garcia-Molina, et al. The claremont report on database research. *Communications of the ACM*, 52(6):56–65, 2009.

[5] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella. Energy conservation in wireless sensor networks: A survey. *Ad Hoc Networks*, 7(3):537–568, 2009.

[6] R. Azimi, M. Stumm, and R. Wisniewski. Online performance analysis by statistical sampling of microprocessor performance counters. In *ICS*, pages 101–110. ACM, 2005.

[7] L. Barroso and U. Holzle. The case for energy-proportional computing. *IEEE Computer*, 40(12):33, 2007.

[8] W. Bircher and L. John. Complete system power estimation: A trickle-down approach based on performance events. In *ISPASS*, pages 158–168, 2007.

[9] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: a framework for architectural-level power analysis and optimizations. *ACM SIGARCH Computer Architecture News*, 28(2):94, 2000.

[10] D. Brown and C. Reams. Toward energy-efficient computing. *Communications of the ACM*, 53(3):50–58, 2010.

[11] R. Brown, E. Masanet, B. Nordman, B. Tschudi, A. Shehabi, J. Stanley, J. Koomey, D. Sartor, P. Chan, J. Loper, et al. Report to congress on server and data center energy efficiency. *Public law*, pages 109–431, 2007.

[12] J. Chen and C. Kuo. Energy-efficient scheduling for real-time systems on dynamic voltage scaling (DVS) platforms. In *RTCSA*, pages 28–38. IEEE, 2007.

[13] Y. Chung, S. Yoo, and M. Kim. Energy and Latency Efficient Processing of Full-text Searches on a Wireless Broadcast Stream. *TKDE*, 22(2):207, 2010.

[14] D. Economou, S. Rivoire, C. Kozyrakis, and P. Ranganathan. Full-system power analysis and modeling for server environments. In *MoBS*. Citeseer, 2006.

[15] X. Fan, W. Weber, and L. Barroso. Power provisioning for a warehouse-sized computer. In *ISCA*, page 23. ACM, 2007.

[16] B. Gedik and L. Liu. Mobieyes: Distributed processing of continuously moving queries on moving objects in a mobile system. In *EDBT*, pages 523–524. Springer, 2004.

[17] S. Gurumurthi, A. Sivasubramaniam, M. Irwin, N. Vijaykrishnan, M. Kandemir, T. Li, and L. John. Using complete machine simulation for software power estimation: The softwatt approach. In *HPCA*. IEEE, 2002.

[18] S. Harizopoulos, M. Shah, J. Meza, and P. Ranganathan. Energy efficiency: The new holy grail of data management systems research. In *CIDR*, 2009.

[19] T. Heath, A. Centeno, P. George, L. Ramos, Y. Jaluria, and R. Bianchini. Mercury and freon: temperature emulation and management for server systems. *ACM SIGARCH Computer Architecture News*, 34(5):116, 2006.

[20] IDC White Paper. Forecasting Total Cost of Ownership for Initial Deployments of Server Blades. *ftp://hp.pl/pub/ c-products/blades/idc-tco-deployment.pdf*, 2006.

[21] A. Jain, E. Chang, and Y. Wang. Adaptive stream resource management using kalman filters. In *SIGMOD*, pages 11–22. ACM, 2004.

[22] R. Joseph and M. Martonosi. Run-time power estimation in high performance microprocessors. In *ISLPED*, pages 135–140. ACM, 2001.

[23] S. Kang, J. Lee, H. Jang, H. Lee, Y. Lee, S. Park, T. Park, and J. Song. Seemon: scalable and energy-efficient context monitoring framework for sensor-rich mobile environments. In *MobiSys*, pages 267–280. ACM, 2008.

[24] M. Kjærgaard, J. Langdal, T. Godsk, and T. Toftkjær. Entracked: energy-efficient robust position tracking for mobile devices. In *MobiSys*, pages 221–234. ACM, 2009.

[25] W. Lang and J. Patel. Towards eco-friendly database management systems. In *CIDR*, 2009.

[26] W. Lang and J. Patel. Energy Management for MapReduce Clusters. *Proceedings of the VLDB Endowment*, 3(1), 2010.

[27] D. Meisner, B. Gold, and T. Wenisch. PowerNap: eliminating server idle power. *ACM SIGPLAN Notices*, 44(3):205–216, 2009.

[28] S. Murugesan. Harnessing green it: Principles and practices. *IT professional*, 10(1):24–33, 2008.

[29] P. Neophytou, M. Sharaf, P. Chrysanthis, and A. Labrinidis. Power-Aware Operator Placement and Broadcasting of Continuous Query Results. In *MobiDE*. ACM, 2010.

[30] J. Paek, J. Kim, and R. Govindan. Energy-Efficient Rate-Adaptive GPS-based Positioning for Smartphones. In *MobiSys*, 2010.

[31] M. Poess and R. Nambiar. Energy cost, the key challenge of today's data centers: a power consumption analysis of TPC-C results. *Proceedings of the VLDB Endowment*, 1(2):1229–1240, 2008.

[32] M. Poess, R. Nambiar, K. Vaid, J. Stephens Jr, K. Huppler, and E. Haines. Energy benchmarks: a detailed analysis. In *the 1st International Conference on Energy-Efficient Computing and Networking*, pages 131–140. ACM, 2010.

[33] P. Ranganathan, S. Rivoire, and J. Moore. Models and metrics for energy-efficient computing. *Advances in Computers*, 75:159–233, 2009.

[34] S. Rivoire, P. Ranganathan, and C. Kozyrakis. A comparison of high-level full-system power models. *HotPower. USENIX Association*, 2008.

[35] S. Rivoire, M. Shah, P. Ranganathan, and C. Kozyrakis. JouleSort: a balanced energy-efficiency benchmark. In *SIGMOD*, page 376. ACM, 2007.

[36] S. Rivoire, M. Shah, P. Ranganathan, C. Kozyrakis, and J. Meza. Models and metrics to enable energy-efficiency optimizations. *IEEE Computer*, 40(12):39, 2007.

[37] D. Tsirogiannis, S. Harizopoulos, and M. Shah. Analyzing the energy efficiency of a database server. In *SIGMOD*, pages 231–242. ACM, 2010.

[38] Y. Wang, J. Lin, M. Annavaram, Q. Jacobson, J. Hong, B. Krishnamachari, and N. Sadeh. A framework of energy efficient mobile sensing for automatic user state recognition. In *MobiSys*, pages 179–192. ACM, 2009.

[39] K. Whang, I. Song, T. Kim, and K. Lee. The ubiquitous DBMS. *ACM SIGMOD Record*, 38(4):14–22, 2009.

[40] J. Xu, B. Zheng, W. Lee, and D. Lee. Energy efficient index for querying location-dependent data in mobile broadcast environments. In *ICDE*, pages 239–250, 2003.

[41] Z. Xu. Building a power-aware database management system. In *IDAR*, pages 1–6. ACM, 2010.

[42] Z. Xu, Y. Tu, and X. Wang. Exploring Power-Performance Tradeoffs in Database Systems. In *ICDE*, pages 485–496. IEEE, 2010.

[43] B. Zheng, W. Lee, K. Lee, D. Lee, and M. Shao. A distributed spatial index for error-prone wireless data broadcast. *VLDB Journal*, 18(4):986, 2009.