A System for Energy-Efficient Data Management

Yi-Cheng Tu [1], Xiaorui Wang [2], Bo Zeng [3], and Zichen Xu [2]
[1] *Dept. of Computer Science and Engineering, University of South Florida, U.S.A.*
[2] *Dept. of Electrical and Computer Engineering, The Ohio State University, U.S.A.*
[3] *Dept. of Industrial and Management Systems Engineering, University of South Florida, U.S.A.*
Emails: ytu@cse.usf.edu, xwang@ece.osu.edu, bzeng@usf.edu, xu.925@osu.edu

ABSTRACT

Energy consumption of computer systems has increased at a steep rate in recent years. Following extensive energy-related research and practice in the hardware and OS communities, much attention has been paid to developing energy-efficient applications. With database systems being a heavy energy consumer in modern data centers, we face the challenge of designing DBMSs with energy as a first-class performance goal. This paper presents our on-going work in designing and implementing a DBMS that enables significant energy conservations while maintaining other performance targets. We follow two new strategies in DBMS implementation to achieve our system design goal. The first one is to change the resource consumption patterns via energy-aware query optimization and reorganizing data records to enable load consolidation in disks. The second strategy is active control of power modes of hardware (i.e., CPU and hard disks) toward energy reduction. Specifically, we use control-theoretic techniques to allow dynamic adjustment of CPU frequency and online data migration to achieve disk load consolidation. Preliminary results have shown the effectiveness of our design.

1. INTRODUCTION

The past decade has witnessed prosperity in green computing research, with a focus on making low-level components (i.e., hardware and OS) of a computing system energy-efficient. Much interest has been shifted to the application level in the recent few years, with the belief that system-level energy management mechanisms can be ineffective without the assistance of energy-aware applications. We believe databases serve as a salient example that supports this view: a DBMS is much like an OS itself by managing resources (e.g., disk, memory buffer) acquired from the real OS for different tasks (i.e., query processing, indexing). Therefore, there is the need to develop DBMSs with energy efficiency as a first-class performance goal. Such work is also well motivated by the substantial economical/social benefits it brings, as databases are found to be a major energy sink in a typical data center [11].

This paper presents our on-going work in designing and implementing an energy-efficient DBMS (E^2-DBMS) that enables significant energy conservation as compared to traditional DBMSs. Specifically, the design goal of E^2DBMS can be formulated as an optimization problem: *minimize energy consumption while maintaining a certain level of query processing performance*. Note that the constraint of the problem is essential as efficient query processing is always the main concern in a database service.

Related Work. Our community has shown much interest in green databases over the past few years. Earlier work [5, 7] provided high-level ideas on improving energy efficiency of data- bases, and both emphasized energy-aware query optimization. Another work [9] proposed query rescheduling and CPU frequency control as two means for green data management and supported their claims with experimental results. Various other topics such as energy quantification of database servers [11, 12, 15], benchmarking [13], cost-based query plan evaluation [8] are also reported. As compared to the above projects, we focus on a systematic framework for energy conservation inside a DBMS rather than individual means. The key challenge is to identify the main components in existing DBMSs that lead to low energy efficiency and relevant mechanisms to alleviate the problem. For that, work by Lang *et al.* [9] is close in spirit to ours. The proposed E^2DBMS system, however, is unique in that: (1) we consider energy-aware storage management as a means for energy reduction, and (2) we use formal control-theoretic methods (instead of heuristics) to ensure performance goals are met.

2. OVERVIEW OF OUR APPROACH

In E^2DBMS, we mainly explore the following two strategies to improve database energy efficiency.

Modifying resource consumption patterns. The

main idea is to tune the DBMS towards energy-efficient computational paths and system configurations. The intuition behind this is: traditional DBMSs are optimized towards query performance only, therefore we can search the space of all system states to locate those that carry low power cost and a small performance penalty in database operations. Clearly, the effectiveness of such an idea depends on the existence of plans/configurations in the search space that provide aforementioned tradeoff between energy and performance. While one report [15] showed concern on the existence of such tradeoffs, we believe there are abundant opportunities, as supported by our previous work [16] and evidence shown in this paper.

Harnessing low-power modes of hardware. While modifying resource consumption pattern achieves platform-free energy saving, controling the low-power modes of hardware may derive more dramatic gains. An effective way to reduce energy consumption of computer systems is to transition the hardware components from high-power states to low-power states [3] when performance allows. Most components in a modern database server such as processors [14], main memory [4], and hard disks [6] have adjustable power states. Components are fully operational, but consume much less power while having degraded functionality in low-power states.

In the following sections, we elaborate on concrete mechanisms to implement the above strategies in a DBMS as well as the technical challenges. There are two important observations about our solutions. First, the mechanisms we propse do not aim at finding the fastest execution path for queries. Instead, they often involve identifying the best tradeoff between energy and performance. This also means that existing DBMSs optimized towards query processing time are not energy-efficient platforms. Second, such mechanisms will not be effective if deployed at the OS level, since the modeling, monitoring, and configuration of database system can only be realized inside the DBMS. This necessitates DBMS-level efforts in making databases green.

The following discussions are organized based on the target (hardware) resource of the relevant mechanisms. Previous work [11, 15, 16] has shown that CPU and storage system are the main consumers of *active energy* [1] in a typical database system. Memory also draws significant amount of power but we do not believe it provides meaningful energy saving

[1] The portion of energy consumed by database workload thus can be controlled by E^{2}DBMS. In opposite to that is the *base energy* – energy consumed by the system when no workload is issued and all hardware run on their lowest power modes.

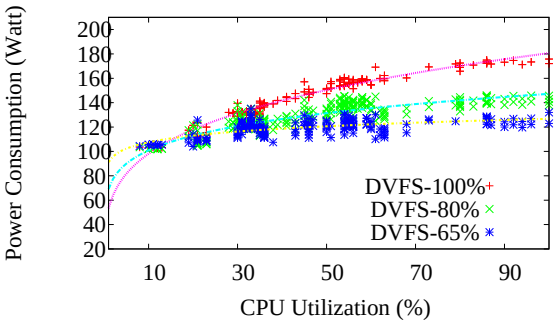

Figure 1: CPU power consumption in a database server under different workload intensities and DVFS levels.

opportunities. Since the performance of database depends heavily on the amount of memory, turning memory chips into sleep mode will lead to excessively long query processing time, thus eating the gain of running in low-power modes.

CPU Resource Control. The mechanism we harness to change CPU usage pattern is *energy-aware query optimization*. CPU is a *power proportional* hardware, meaning its power consumption is (roughly) proportional to the intensity of the computational load it handles (Fig. 1). Significant power can be saved if we can find query execution plans that do not require as much CPU time as those found by existing query optimizers. Such plans will also be energy-efficient if they require the same or slightly longer I/O time, and they are often ignored by the query optimizer in existing DBMSs since the latter only considers performance in the query cost model. In our previous work [16], we systematically studied the power profiles of various database benchmarks, and found that queries with such low-CPU plans that are missed by traditional query optimizers are very common (e.g., in 10 out of the 19 queries we studied in TPC-H). Therefore, we can design a novel query optimizer to identify such plans upon evaluating the energy cost of alternative plans. Note that hard disks are not energy-proportional therefore this mechanism has little effects on disk energy consumption.

For modern CPUs, it is well-known that the Dynamic Voltage and Frequency Scaling (DVFS) technique can allow *cubic reductions* in power density relative to performance loss [14]. This has been verified using various database workloads in previous work [9] and our experiments (Fig. 1). Therefore, we design a DVFS controller of CPUs as an integrated part of E^{2}DBMS to harvest the power savings generated by low DVFS levels. The power saving potential from adjusting DVFS is significant: a

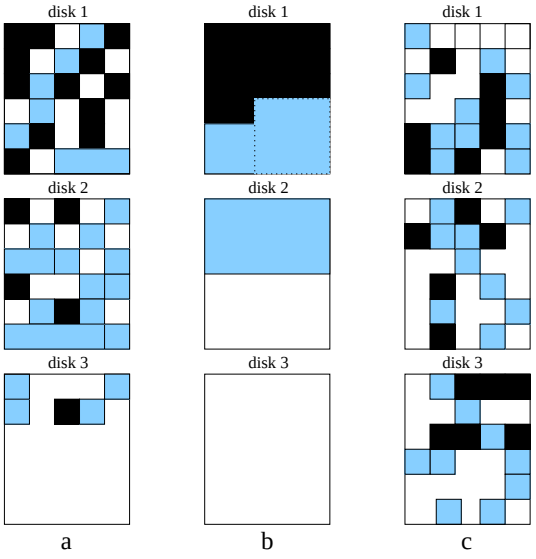

Figure 2: Three distributions of data blocks on three hard disks. Darker color means higher data popularity in block

60W power difference can be observed in the server we tested (Fig. 1). This confirms the results from [9], in which a 45% reduction of active energy is reported. On the other hand, since query processing time is often dominated by I/O time (especially in I/O-bound workloads), a performance penalty associated with low-power CPU modes will not affect system performance dramatically. Modern CPUs typically provide OS-level handles for efficient adjustment of DVFS levels, making DVFS control extremely light-weighted. Note that the proposed CPU control will not be effective if deployed on the OS level, since the control decisions depend heavily on the collection and analysis of database states.

Storage Management. Traditional multi-disk storage systems seek *load balancing* to achieve best I/O performance. In E^{2}DBMS, however, the storage manager follows a *load consolidation* approach by putting most I/O load into a subset of the physical disks. By this, energy saving opportunities are created since disks with low I/O load can spin down or enter low power/performance mode. Specifically, we explore an *energy-aware data placement* mechanism that changes the data access patterns in individual disks via dynamically reconfiguring data placement at runtime. Note that the key problem in load consolidation is to balance energy savings and disk contention (i.e., data access performance). For example, the placement scheme shown in Fig. 2b will maximize power savings but yield unacceptable performance for disk 1. On the contrary, Fig. 2a shows a better solution by distributing hot items into two disks. In the load balancing scheme (Fig. 2c), blocks of all color depth will be randomly distributed into all disks thus energy reduction will be minimized. The key issue for the data placement algorithm is to provide the highest level of load consolidation with a small overhead, and maintain the service quality. Obviously, this mechanism only provides higher level of load consolidation - it requires the collaboration of low-level mechanisms that can change the power modes of disks to actually save energy.

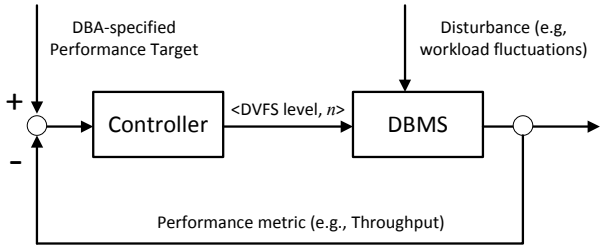

Figure 3: The feedback control loop for energy-efficient CPU usage

Traditional disk farms are notoriously energy inefficient: all disks used to hold the database keep spinning at full speed at all times while user accesses are often skewed towards a very small fraction of the database [10]. This clearly implies a big waste of energy and also opportunities for optimization. Fortunately, most hard disks shipped today can be transitioned to low-power modes with a fraction of the power consumption in active mode. Furthermore, the emergence of multi-speed hard disks [6] offer more flexible power/performance tradeoffs. With the aforementioned dynamic data placement mechanism to generate opportunities for load consolidation, the main problem is to develop a dynamic power management (DPM) module to determine the exact power mode for each disk.

3. E^{2}DBMS DESIGN ISSUES

This section briefly introduces the technical challenges and our solutions in implementing E^{2}DBMS towards its design goal at runtime.

Energy-aware CPU control. For CPUs, we map the design problem into a feedback control loop (Fig. 3) that periodically changes the database and hardware configurations (i.e., the input signal in the loop) such that the system performance (i.e., the output signal) traces a target specified by the DBA. In this feedback loop, the system input (i.e., control signal) is a multi-dimensional value: the DVFS level of CPUs and a parameter n that determines the preference of power over performance in query

optimization . The output signal is a system-level performance metric such as query response time or throughput. The intuition of our design comes from the fact that *there is a monotonic relationship between power and performance* when the system runs under different configurations. Therefore, if the query processing performance of the DBMS is unnecessarily better than the threshold, we can decrease the DVFS level of the CPUs for power savings, or be more aggressive in choosing power-efficient query plans. If the system performance drops below the threshold, we can do the opposite to achieve faster query processing. *As long as we keep the actual system performance on the desirable level, system power will be minimized.* The first challenge is an actuator of the control signal, i.e., mechanisms to ensure the DBMS run in the state specified by the signal. The second challenge is the choice of the input signal whose value enables different tradeoffs between power and performance in query optimization. More importantly, we need to make the system trace the desired performance threshold in a dynamic environment.

The DVFS part of the control signal can be easily enforced by a light-weighted system call. In the Intel CPU we tested, this can be done by writing the DVFS level into a cached system file with a 20 microsecond overhead. To enforce the tradeoff set by the parameter n, our previous work [16] presents an *energy-aware query optimizer* that selects query plans based on this signal. For each query plan, the optimizer evaluates its running time T and power cost P, and gives a score as $C = PT^n$ – the plan with the lowest score is selected. Here a big problem is the accurate estimation of P, we propose a two-level approach for this: we first use standard simple workloads to generate the static power cost of individual database operators, and use those as parameters to derive the total cost of an entire plan. Then such parameters are updated periodically following a *Recursive Least Square* manner to gain better accuracy in a dynamic environment.

The second challenge is often seen in database tuning problems. We use well-established control-theoretic system modeling and controller design techniques to achieve our design goal. As compared to traditional heuristics-based database tuning methods, control theory provides: (1) guaranteed control performance such as accuracy, stability, and short settling time; and (2) tolerance of modeling errors and fluctuations of system/workload features. The control loop design will begin by modeling the relationship between system input (i.e., control signal) and output (i.e., performance). We will use *system identification* techniques to derive such models via experiments with standard inputs. The next step is to design the controller towards guaranteed performance. Our plan is to apply the Proportional-Integral (PI) control theory to achieve stability and zero steady state control – the latter ensures maximal energy saving.

Energy-aware storage system. For the storage system, the main issue is to find the proper level of load consolidation and power modes in disk arrays to meet the performance and power requirements. Here our system design goal can be viewed as a classical optimization with performance target as a constraint and energy as the optimization target. As an input to this problem, the access rate of individual data items typically changes over time in a database system. This requires the optimization solution to be recomputed therefore efficient algorithm is needed. Another fact that complicates the problem is that the realization of a new optimization solution, unlike the adjustment of DVFS and query optimizer parameter, bears very high costs. For example, changing the data placement scheme (to reach a new level of load consolidation) involves significant I/O (and power) cost by moving data around. Transitioning disks to another state also involves a performance penalty on the level of seconds [6]. In this project, we tackle load consolidation in storage systems by treating it as a *dynamic optimization* problem and propose efficient algorithms with provable accuracy bounds.

The main task is to develop an integrated optimization model for data migration and disk state adjustment and efficient algorithms to achieve the optimal (or near-optimal) balance between power consumption and query performance. Starting from an M/G/1 queueing model for single-disk behaviour, we have developed a mixed integer programming model over multiple periods for inter-disk fragment migration and have tested it with industrial solvers. The next step would be to analyze the model to identify structural properties of optimal solutions for the purpose of creating guidelines to reduce search space for computing algorithms.

Another interesting problem is a DBMS mechanism to reorganize data records for greater chances of load consolidation. Our studies show that the data popularity can be traced down to individual tuple level in a typical database. Without extra efforts, a data page will hold records with different levels of popularity thus all pages have similar aggregated page-level popularity. Referring this to Fig. 2, all blocks will have similar color depth and load consolidation becomes impossible. We will

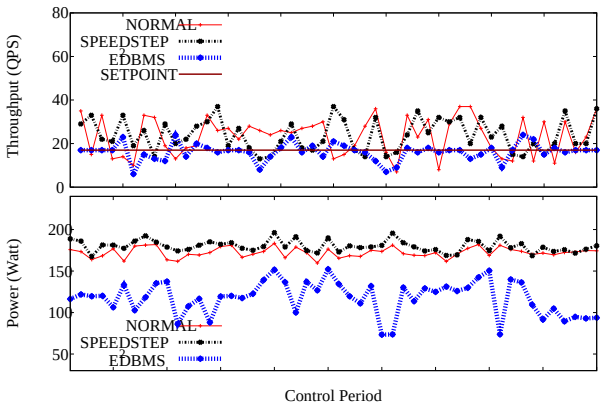

Figure 4: Database throughput and active power consumption in 50 control periods

use a *popularity-based intra-disk data fragmentation* method to dynamically organize database records in a disk such that data items with similar temperature are stored in consecutive pages. Specifically, we divide data on a disk into many fragments, each of which occupy a fixed number of consecutive pages. At runtime, we monitor the access frequency of data records and dynamically place records into a proper fragment according to their popularity. Such fragments will be used as the basic unit of data migration mentioned above. We propose an *incremental restoration* algorithm to solve such a tuple packing problem – the main idea is to constrain data reorganization to resident (in-memory) pages as much as we can, thus minimizing extra I/O cost.

4. PROGRESS AND INITIAL RESULTS

We are in the process of actively studying the above issues and implementing the system. An early version of E^{2}DBMS that augments the PostgreSQL kernel with the energy-aware query optimizer was made public [17]. In this section, we report part of our experimental results with a focus on revealing potential of the E^{2}DBMS system.

We have built an initial version of the control framework proposed in Figure 3. In this framework, we designed and implemented a proportional-integral (PI) controller that tracks system throughput to a reference value via adjusting DVFS levels of the CPU. Such a design is based on a dynamic DBMS system model derived from comprehensive system identification study. The implemented framework was tested for its robustness under various database workloads and environmental setups. For example, Fig. 4 shows the performance and power consumption in 50 control periods during one experiment, in which the throughput setpoint is 18

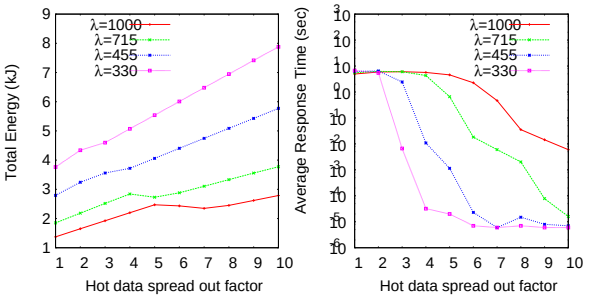

Figure 5: Energy cost and response time of a read-only workload on a 10-disk system

queries per second. We fed the database with a composite workload that consists of TPC-C and TPC-H queries in a "racing" environment, in which multiple CPU-intensive programs compete with the DBMS for CPU resources. The two baselines are "Normal" that always sets the DVFS level to 100% and "SpeedStep" – an Intel built-in technique to adjust DVFS on the OS level. Compared with those two baselines, our controller (i.e., "E^{2}DBMS") maintains the throughput (blue dotted line) around the setpoint without introducing too many overshoots thus delivers significant energy savings – 51.3% more savings than the OS-level mechanism "SpeedStep" was observed. Other empirical results and controller stability analysis of the CPU control framework can be found in [18].

We also performed simulation experiments to verify our ideas in power-aware storage management. In this set of experiments, we simulate a database table whose tuples span 10 hard disks. We follow the well-established b/c model [10] in generating random read workloads. It is well known that database access patterns are highly skewed and can be described as an 80/20 or even a 90/10 model. Given the popularity of tuples, we implement a static data fragmentation scheme such that the c% hot tuples are placed into a variable number F (called *data spread-out factor*) of the 10 disks. Disk specifications are based on those of a commercial product – the Hitachi UltraStar 7K4000 [1]. Such disks have two power modes: a high power mode that consumes 11.4 watts at 7200rpm spinning speed and a low power mode with 4.6 watts at 6300rpm. We assume the hot disks always run at 11.4w/7200rpm and the cold disks at 4.6w/6300rpm.

Fig. 5 shows the energy consumption and performance of such schemes under four 90/10 workloads with task arrival rate ranging from 330/s to 1000/s. By comparing the total consolidation ($F = 1$) case with the load balancing ($F = 10$) case, energy savings of 50.8% (for $\lambda = 1000$) to 52.2% (for $\lambda = 330$)

can be observed.[2] However, the $F = 1$ cases show bad performance - average read response time of all disks is on the level of seconds. However, the response time drops quickly when we spread the load to more disks. For example, it reaches the level of milliseconds for $\lambda = 330$ when the hot items are spread into only three disks. We believe the above results clearly show the potential of our idea of performing DBMS-level load consolidation. Specifically, those F values in the middle range provide desirable tradeoffs between energy and performance - about 50% energy can be saved with little performance degradation. OS-level consolidation, which is simulated by those with high F vales, can only reach a state of balanced load without saving any energy. Note that all results presented in this section only considered direct energy savings. If we consider energy savings from cooling systems,[3] the total energy reduction will be even higher.

5. SUMMARY

In this paper, we argue for the great potential of energy-efficient database systems. Such potential can only be realized via designing DBMSs with energy consumption as a first-class performance goal. We present our work in building such a DBMS named E^{2}DBMS, which achieves high energy efficiency via two strategies: modifying resource consumption patterns and controlling power modes of hardware. Such strategies are implemented through a series of mechanisms that target at the main active energy consumers in a typical database server: CPU and disks. Experiments run on an initial E^{2}DBMS prototype and simulation platform demonstrate significant energy savings.

Acknowledgements

This work is supported by US National Science Foundation (NSF) grants IIS-1117699 and IIS-1156435.

6. REFERENCES

[1] Ultrastar 7k4000 OEM specification. http://www.hgst.com/tech/techlib.nsf/products/ultrastar_7k4000.

[2] F. Ahmad and T. N. Vijaykumar. Joint optimization of idle and cooling power in data centers while maintaining response time. In *ASPLOS'10*, pages 243–256, 2010.

[2]The low intensity workload consumed more energy because our workload contains a fixed number (20,000) of operations. Disk idle time is longer in low intensity cases. This also shows that a higher percentage of energy can be saved when system is not heavily loaded.

[3]According to [2], for one watt of direct power saving, one to two watts can be saved from cooling.

[3] R. Bianchini and R. Rajamony. Power and energy management for server systems. *IEEE Computer*, 37(11):68–74, 2004.

[4] W. Felter *et al.* A performance-conserving approach for reducing peak power consumption in server systems. In *Procs. Intl. Conf. Supercomputing (ICS)*, 2005.

[5] G. Graefe. Database servers tailored to improve energy efficiency. In *Procs. EDBT Workshop on Software Engineering for Tailor-made Data Management*, 2008.

[6] S. Gurumurthi *et al.* DRPM: Dynamic Speed Control for Power Management in Server Class Disks. In *ISCA*, pages 169–179, 2003.

[7] S. Harizopoulos *et al.* Energy efficiency: The new holy grail of data management systems research. In *CIDR*, 2009.

[8] M. Kunjir *et al.* Peak Power Plays in Database Engines. In *EDBT*, 2012.

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

[10] M. Nicola and M. Jarke. Performance modeling of distributed and replicated databases. *IEEE Trans. Knowledge and Data Engineering*, 12(4):645–672, Jul/Aug 2000.

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

[12] M. Poess and R. Nambiar. Tuning servers, storage and database for energy efficient data warehouses. In *ICDE*, 2010.

[13] M. Poess *et al.* Energy benchmarks: a detailed analysis. In *TPCTC*, pages 131–140, 2010.

[14] K. Skadron *et al.* Temperature-aware microarchitecture: Modeling and implementation. *ACM Trans. Architecture and Code Optimization*, 1(1), 2004.

[15] D. Tsirogiannis *et al.* Analyzing the energy efficiency of a database server. In *SIGMOD*, pages 231–242, 2010.

[16] Z. Xu, Y.-C. Tu, and X. Wang. Exploring power- performance tradeoffs in database systems. In *ICDE*, pages 485–496, 2010.

[17] Z. Xu, Y.-C. Tu, and X. Wang. PET: Reducing database energy cost via query optimization. In *Procs. VLDB Endowment*, 2012.

[18] Z. Xu, X. Wang, and Y.-C. Tu. Power-Aware Throughput Control for Database Management Systems. In *the 10th International Conference on Autonomic Computing*, pages 315–324, 2013.