Exploring Power-Performance Tradeoffs in Database Systems

Dr. Ralf Schenkel

Presenter:

Seyed Mehdi Khodadad Hosseini

Opponent:

Felix Martin Schuhknecht

Abstract

Introduction

According to the steep growth in consumption of calculation systems, their power usage also grew drastically and these days, the second highest bill of a data center company after its maintenance costs is the power bill. In the year 2006, only in US 1/3 of all power consumptions belonged to the data centers. Therefore, much attention has been paid these days to the design of energy-efficient computing systems and applications.

How to Reach the Energy Efficiency?

To achieve the energy efficiency, we have two ways available:

(1) *Improve throughput while maintaining same power line (Watt)* or

(2) *Reducing the power cost while not sacrificing too much throughput performance.*

During the last decades there were many attempts on hardware improvement and the attention on power-aware applications has been neglected. Due to the importance of database systems in nowadays companies, in this paper they concentrate on reducing the power consumption of such an application. By changing the query optimizer of current DBMSs we can have a significant power saving. The hypothesis of the paper is that the current DBMSs query optimizers only focus on performance and have no consideration on power reduction. As it is obvious that CPU time is negligible compared with I/O time, most of attempts were made for algorithms with less I/O and more CPU bound. To continue this discussion we can separate two kinds of:

Idle power: the power which is consumed whenever the hardware is not engaged.
Active power: the difference between the peak of power and idle one. So we can say that:
Active power = Peak power – Idle power

As it was presented in the paper, the active power of the CPU is so much greater than the HDD active power. Before running a query, the query optimizer of a DBMS first produces different plans for executing the query and then decides which one to choose. Current optimizers choose the one which has the best estimation in terms of performance. Therefore, it is usually optimized towards lowest I/O consumption. The authors believe that by choosing a plan, which might be less optimal in terms of I/O but has less CPU usage, it is possible to save considerable power by sacrificing a negligible performance. Therefore, if a query optimizer considers both time and power usage, it can have valuable power saving during processing of the queries by choosing suitable plans. I conclude this hypothesis with a simple example:

Example1: Imagine two plans A and B, generated by the query optimizer for the same query in *Figure I.* The I/O cost of plan A is slightly higher than that of plan B but it uses much less CPU time than B. Nevertheless, the current query optimizers will choose B, since I/O is always the dominant factor in total processing time. On the other hand, a power-aware DBMS query optimizer is able to choose plan A and therefore, we can have considerable power saving without sacrificing too much performance.

Figure I: The figure shows two plans A and B for the same query. Plan A is less CPU bound but a little more I/O bound so its performance is a little bit better than that of plan B even though it has so much less CPU usage. As we know that the power usage of HDD is negligible compared with CPU power usage, plan B has more power and energy consumption than plan A.

To prove this hypothesis, they needed to calculate the current power usage of the database to be able to see if their attempts lead to power usage reduction or not. For this they connect a power meter via USB to their experimental system. They used PostgreSQL since it is an open source DBMS and they could have access to its query optimizer for changing it to their wished design. They also used two standard benchmarks: TPC-C and TPC-H for the workloads.

The queries in the experiment were divided to two: **Cat I**, those which are most I/O bound which were not supposed to have much chance for power-saving and **Cat II**, the CPU intensive jobs which could lead to considerable power saving. Finally they showed the interesting results of their experiment that we are going to talk about in the following discussions.

Discussion: Positive and Negative Aspects

Valuable results:

The paper proved that the power-saving opportunities exist by the experiments they did. They are valuable and seem to be a good start for more future work in this area. As shown in *Figure II,* there is power and energy saving potential in most of the queries of TPC-H benchmark.

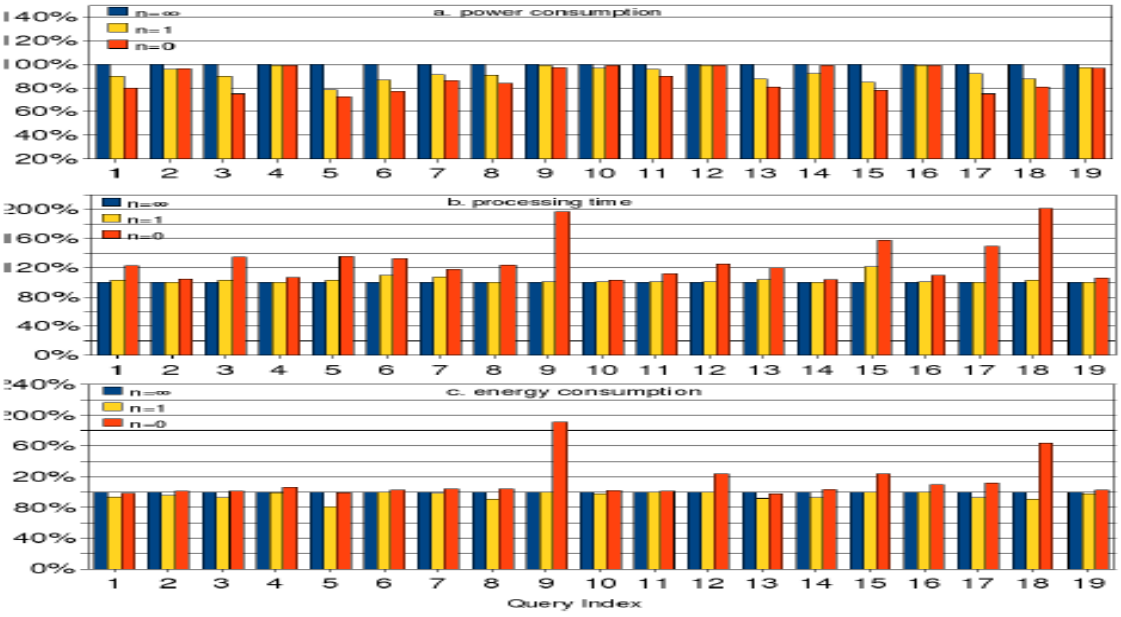

Figure II: The Power consumption, performance and energy consumption of 19 different queries of TPC-H benchmark as measured by power meter with different values of n.(n=0→power consideration only, n= ∞ →performance consideration only, n=1→ power-aware optimizer with equal consideration to performance and power.)

Pointing out an important factor that the current query optimizers do not care about:

As nowadays DBMSs do not consider the power usage in their query optimizers, it seems that the potential of power saving is totally ignored by them, so with the experiments of this research and adding power consideration to the current DBMS optimizer, we could reach power saving as well as energy saving. (**Energy = Power * Time)**

Using a widely used DBMS:

The research was done on a real DBMS which is really valuable and can help us to have the better understanding of the experiments and results. Another good point was that it used PostgreSQL to prove its claim, and it is an open source DBMS which is accessible for everybody to do their own researches with the new methods and compare their results with this topic results. Also using the two standard workloads like TPC-C and TPC-H adds more value to the derived results of this paper.

Less power consumption does not imply less performance:

Of course one of the most interesting results of this paper occurred in the TPC-C workload experiment. There, the time of the power-aware run was three minutes less than its run by the current optimizer. It showed so clear that PostgreSQL optimizer does not guarantee the best plan for query selection in terms of performance. This means, that there might be potential in the power-aware query optimizer to improve both power usage performance.

Significant potential, Excitement Results:

As it was shown in the paper there are opportunities to save power, up to 22% of total energy savings. These are exciting numbers in power saving. Another thing we can draw is that designing power aware query optimizer is a promising direction to enable power conservation. On the other hand, it also demonstrated stable power saving potentials in a wide range of workloads.

As already presented, their work shows a lot of positive aspects and points into the right direction in terms of power saving in DBMS environments. Nevertheless, there are a few weak spots which we will have to analyze in the following:

Plan Generation Phase:

As presented in the papers, the key component of their energy aware system is the modified query optimizer of PostgreSQL. To summarize it again, the optimizer is changed in a way that it scores the generated plans not only with respect to performance but also in terms of energy efficiency. This means, that in the first place, the generation of the set of plans is not changed in any way. This leads to a huge amount of computed query plans where most of them are inferior. Therefore, they could improve their system by pushing parts of the energy awareness already into the generation phase.

Power Model Calibration:

A key component in estimating the power consumption of a generated plan is the power model calibration phase, in which the CPU costs for tuple access and page I/O have to be determined. This is necessary, since one can measure the cost externally only for a whole operation like a sequential scan. Their approach is therefore, to calibrate the values by performing multiple runs in advance and to let the costs converge against the correct values. However, this calibration method is statically and therefore, not suited for immediate changes of workload and DBMS congestion. The follow-up paper seems to address this problem by proposing a first draft of an online modeling approach.

CPU bound vs. I/O bound Queries:

During their experiments, they identified two classes of queries. The queries of category I are mainly I/O bound, whereas most of the queries of category II are CPU bound. If we have a look at current hardware development, we can observe the faster increasing performance of CPU in contrast to available bandwidth, which means that I/O is more and more the bottleneck of a system, in which the data has to be loaded from a persistent disk. This should also lead to an increasing amount of

category I queries and to an decreasing number of category II queries and therefore, the set of queries which are good to optimize in terms of energy efficiency becomes smaller and smaller over time. Unfortunately, the paper does not take these aspects into account. It presents results for category I and II queries, but it does not explain the circumstances under which the system encounters these types of queries. For example, in a main-memory DBMS, the amount of category II queries should dominate by far and energy efficiency can be gained, whereas in standard disk based DBMS, the energy aware optimization might be only marginal.

Adjustment of n:

The key setup tool for adjusting the tradeoff between power and performance is the value n in the cost model $C = PT^n$. If n is 0, the plans with lowest power consumption are chosen. For n = 1, energy consumption is in the focus. The larger n becomes, the more the performance of a plan is important. Therefore, n is the value the administrator has to set to tweak the systems towards low power consumption or processing performance. In the experiments, n is set manually to the values 0, 1 and towards unlimited. Unfortunately, the paper does not give any specific recommendations how to set n in certain situations. And the correct choice of n is crucial to the system behavior, since a d^n-time degradation in performance is necessary to achieve a d-time reduction of power consumption. Therefore, a slightly bad choice of n can mess up the system performance easily. In the follow-up paper, they introduce a method to determine the value of n automatically (control-based query evaluation model) by measuring the current system throughput and adjusting n, such that system processes the queries within a given time bound. However, it would have been nice to have a set of recommendations how to set n manually (or it this doesn't make sense at all) or at least to see the automatic calibration of n in the experiments, since finally the correct choice of this tradeoff value is the major challenge in the usability of this system.

Closed DBMS Software:

Since the key idea of the approach is to integrate the energy aware scoring system into the query optimizer, the system has to be open-source. Of course, this is sufficient for proving the correctness of their approach, it also means that in the near future, it will not be possible to integrate their techniques into existing proprietary systems like Oracle and IBM. While this can't be mentioned as a real negative aspect of these papers, it supports the community of people who see more potential of saving power by improving the underlying hardware like CPU and storage systems, since in existing and running systems, it is much cheaper and easier to modify the hardware than the running DBMS software.

Degradation Is Not Negligible Every Time:

Although the performance degradations in the current experiences were negligible, it is possible for huge databases with grand streams of queries such as Google database or other search engines, that even a little degradation of each query may lead to considerable delays after a while. In this case we may cope with this problem by using a threshold for our power-aware optimizer to switch on the performance-only optimizer.