Journal of Information Systems Engineering and Management
2025, 10(24s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

From Performance's Designing Feedback in the Light of Energy Efficiency Improving: How & Proofs?

Simon Pierre Dembele[1], Francesco Colace*[2], Angelo Lorusso[3] and Domenico Santaniello[4]
[1]*Institute of Computer Science, University of Tartu, Tartu, Estonia*
[2,3]*Department of Industrial Engineering, University of Salerno, Fisciano, Italy*
[4]*Department of Cultural Heritage, University of Salerno, Fisciano, Italy*

ARTICLE INFO

Received: 26 Dec 2024

Revised: 15 Feb 2025

Accepted: 25 Feb 2025

ABSTRACT

At the core of any Data Storage Systems (DSS), ongoing research efforts are dedicated to refining algorithms for effective data storage and retrieval. The primary objective of any data management systems (DBMS) is improving the performance of data management such optimizing queries response times. Decades of research have borne fruit in the form of performance-oriented optimization techniques, revolutionizing DSS usability and enhancing the user experience significantly. However, the explosion of digital data and the rise of various data-driven services, combined with the ongoing decline in processing equipment costs, are contributing significantly to the heightened energy consumption of data storage systems. With the pervasive incorporation of digital services into our daily lives, energy management has become a major challenge for digital companies to mitigate their environmental footprint and reduce their energy cost. Considering this perspective, several initiatives have been undertaken, such as the development and exploitation of renewable energies, or the redefinition of the Database (DB)'s primary quality objective: from the performance-oriented approaches to an energy-integrated approaches. In this paper, we suggest leveraging the advancements in performance characterizing to integrate energy efficiency into DSS. Initially, we assess the energy efficiency of a particular optimization technique, specifically, data partitioning applied during physical design. Next, we evaluate the energy efficiency of the query processor system by introducing a multi-objective formulation that considers two non-functional constraints: performance and energy consumption. Lastly, we present a simulator that incorporates an energy model, enabling the generation of query plans optimized for energy efficiency.

Keywords: Energy Efficiency, Data Storage Systems (DSS)

INTRODUCTION

In recent years, we have observed a rapid transformation of our daily activities due to significant technological advancements. The pivotal role of data in driving this digital transformation is widely acknowledged by all. Presently, the increasing significance of data is widely recognized, serving as a guiding force for strategic decision-making in businesses and a tool for governments to monitor, control, and prevent crises. At the core of any business, data plays a central role while simultaneously acting as a new environmental pollutant. This pollution primarily stems from the greenhouse gas emissions generated by data storage systems, which consume substantial energy when processing these data.

These systems are designed to store, process, recover, and share data. They are the backbone on which data centers are built. In addition to offering a secure and enduring storage framework, they facilitate efficient access to stored data for users or programs through query languages like SQL(Structured Query Language). The objective of performance optimization is to reduce query response times by maximizing the utilization of system resources (CPU, memory, disk, network) while ensuring data consistency. Attaining this goal necessitates a deep comprehension of the logical and physical design of the data and applications implemented on the system (Kamatkar et al., 2018).

The urgency of environmental sustainability has never been greater, as reflected in current and coming statistics and associated repercussions on life. Factors such as the proliferation of the Internet, the emergence of new technologies, heightened user demands, and the exponential growth of data production are significantly amplifying the energy and water consumption of data centers, and consequently, their carbon footprint. This increase is primarily due to digital infrastructure installation, including storage and communication devices. The carbon footprint of data centers is the highest it has ever been(Kilgore, 2023). Actually, Datacenters alone are responsible for 2.7% of the EU's electricity demand and their consumption is expected to increase by 3.21% by 2030. China electricity consumption in Datacenters sector will exceed 400 billion kWh By 2030, accounting for 3.7% of the country's total electricity consumption (Mar, 2022).

In response to this situation, technologies have progressed towards energy-saving strategies referred to as Energy-Efficiency (EE) solutions. Thus, the conventional approach of improving the performance of IT systems has converged towards satisfying the constraints of environmental sustainability by reducing the Carbon impact. Among the initiatives proposed are:

- Some regulations of so-called **Good Practice Standards** in favor of climate change, such as Estonia's "*National Energy and Climate Plan*", which aims "*to provide Estonian people, companies and other member states with as much information as possible about the measures that Estonia plans to use to achieve the energy and climate policy targets*".
- More **Concrete Actions** within Datacenters by *relocating them to regions with more favorable climates, or opting for energy-efficient chemical components, optimizing the design of logic circuits, designing algorithms efficiently, and establishing energy management techniques, etc.*

This research tackles the issue of Databases over-consumption in Datacenters by combining empirical approaches with those of physical design optimization techniques. The goal is to improve energy efficiency. Our initiative focuses on the physical design phase, specifically data partitioning technique, and the incorporation of an energy model into PostgreSQL Query Processor *(QP)*.

A Query Optimizer plays a pivotal role by: (i) presenting a search space of query execution plans, (ii) offering cost estimates for all operations constituting the query, and (iii) outlining a strategy for exploring this space to select the optimal performance plan [39]. Following a similar approach as performance-oriented cost models, we introduce a framework named "*GreenPipeline*" built on the PostgreSQL system by integrating our energy model into its Optimizer, with the aim of selecting energy-optimizing plan, or a plan that optimizes a compromise threshold between the two Non-Functional Objectives (Performance and Energy). Figure 1 illustrates how our future optimizer will work considering performance and energy cost model.

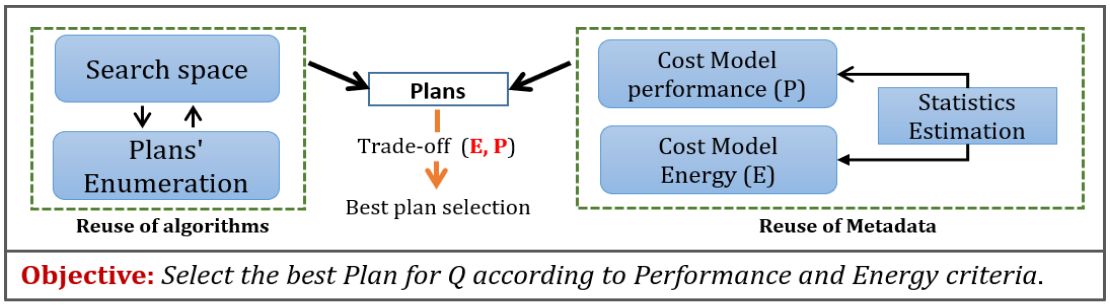


Figure 1: Energy integration illustration.

Our Contribution

The major technical contributions of this paper can be summarized as follows:

- We present an in-depth review of existing works focusing on the integration of energy into databases. We classify these researches into four categories based on their deployment levels (Figure 2): (1) hardware solutions, (2) software solutions, (3) environmental management, and (4) behavioral standards to guide or facilitate EE techniques.

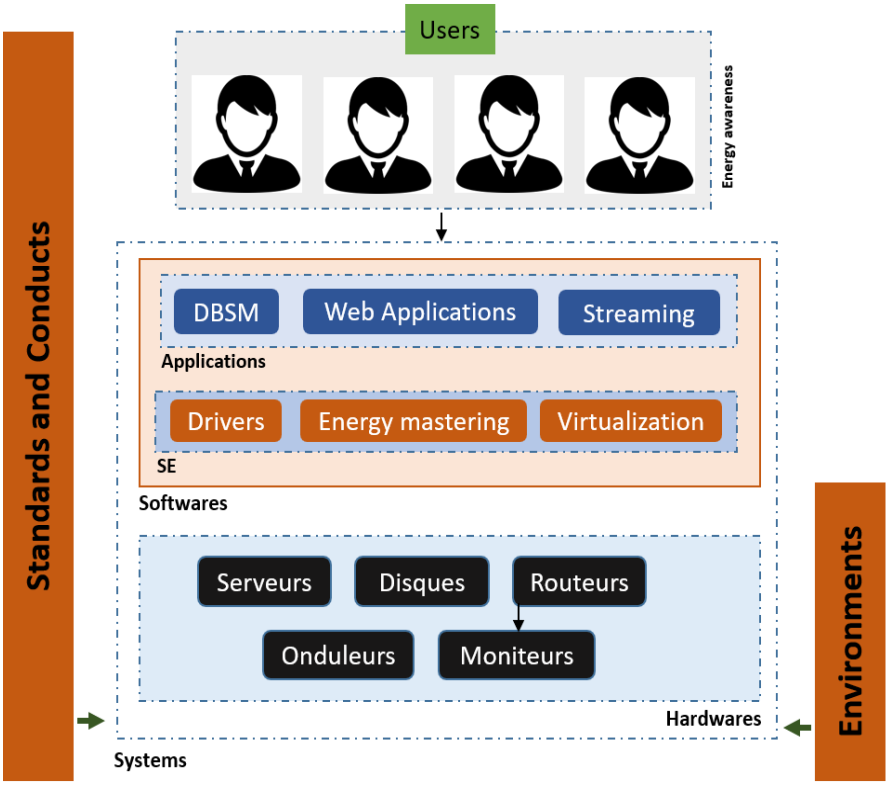


Figure 2: EE deployment levels

- We conduct an empirical study to assess the influence of data partitioning techniques on energy consumption during the processing of Select-Join-Projection (SJP) queries.
- After proposing our strategy for defining energy models, we apply it on a row-stored data engine (PostgreSQL). In the model conception process, machine learning algorithms are used to estimate the weights of certain parameters. This study rely on three algorithms: Non Linear Regression (NLR), Artificial Neural Networks (ANN), and Random Forests (RF) in order to compare their predictions and evaluate their influence on the models accuracy.
- We propose an evaluation method for plan selection based on a trade-off between energy and performance. This threshold signifies a preference for either Performance (T) or Energy cost (E) superiority.
- Finally, we introduce a Framework that incorporates the validated model and plan evaluation method into the PostgreSQL optimizer for energy estimation and saving studies.

Paper Organization

After covering the basics to facilitate the understanding of the paper, we present the previous works on energy efficiency in databases, focusing more on the Software's approaches in Section 2. In Section 3, we discuss the results obtained when querying un-partitioned and partitioned data. In Section 4, we propose our modeling approach and then we evaluate our approaches by conducting prediction studies. Section 5 describes the main features of our Framework, and then presents the energy savings achieved by deploying our tool. We conclude our work in section 6.

BACKGROUND

Energy Concept

Formally, energy and power can be defined as follows (Beloglazov et al., 2011):

$$P(t) = d/dtE(t)\ ;\ E(t) = \int_0^t p\,(\tau)d\tau \quad (1)$$

where *P, t,* and *E* represent respectively the power, time, and energy.

- **Query's energy:** In the context of databases, usually two aspects of power are considered: average or peak energy. They represent respectively, the Average(Avg) energy and Maximum(Max) power consumed during the query execution. the energy of a DSS when executing query (Q) denoted by E_Q is defined as follows:

$$E_Q = (Avg|Max)\left\{\left(\sum_{1}^{n} Power_{op_i} * Time_{op_i}\right) + \varepsilon\right\} \qquad (2)$$

where n and ε are the number of operators in the plan and the energy measurement errors.

- **Energy efficiency:** refers to the optimal use of energy by a system to provide the same service. Energy efficiency (EE) is defined as the ratio of performance, measured in rate of work done, to the amount of energy used(Tsirogiannis et al., 2010).

$$EE = \frac{Useful\ Energy\ output}{Total\ Energy\ input} = \frac{Performance}{E} \qquad (3)$$

LITERATURE REVIEW

Energy consumption has been widely studied in the computer architecture field for decades. Energy efficiency techniques in systems are categorized into two primary domains: (1) Static Energy Management and (2) Dynamic Energy Management. Static management aims to enhance energy efficiency during component design, relying on electronic circuit optimization and components miniaturization. On the other hand, dynamic management focuses on regulating energy consumption during system operation (Lu & De Micheli, 2001). Our research emphasizes techniques based on the dynamic management of energy consumption, which we have classified into four main categories (see Figure 3) based on their deployment level: (1) Hardware solutions, (2) Software solutions, (3) Environmental management, and (4) Standards of conduct.

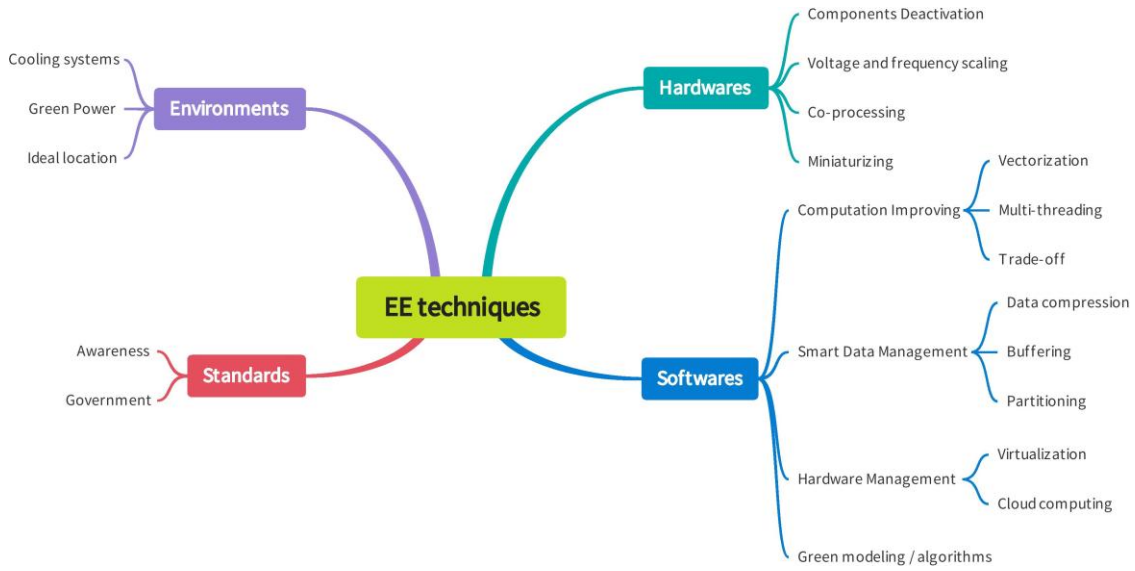


Figure 3: EE approaches in dynamic energy management.

Hardware Solutions

At the hardware and micro-architectural levels, three alternative solutions are implemented: (a) Dynamic Component Deactivation (DDC), (b)Dynamic voltage and frequency scaling (DVFS) and (c) combining processing units (co-processing) (Beloglazov et al., 2011). The DVFS technique is widely applied as energy-efficiency approaches in micro-architectural devices (CPU, memory, etc.), in packet forwarding on interconnection networks and in the problem of task scheduling (Nishikawa et al., 2015),(Haj-Yahya et al., 2020),(L. Zhou et al., 2020),(Pagani et al., 2017),(Jaiantilal et al., 2010),(Salami et al., 2020) ,(Y. Zhou et al., 2018).

Software Solutions

In the field of databases, the software perspective consists of investigating query optimization methods, scheduling algorithms, physical design and data update techniques with energy concerns in view. These approaches are generally characterized by (1) the development of an energy cost model and (2) the use of a cost-driven technique in the optimization phase.

The majority of efforts in this field have suggested methods to improve the efficiency in query processor. This choice is justified by the observation that this particular component constitutes the most energy-intensive in DSS architecture. By extracting the common characteristics of a set of queries, which define the key factors influencing its energy consumption when processing them in a specific architecture (centralized, distributed, parallel). The authors of these works (Xu et al., 2015)(Liu et al., 2013)(Rodriguez-Martinez et al., 2011)(Guimarães et al., 2016)(Roukh et al., 2017)(Dembele, Bellatreche, Ordonez, & Roukh, 2020)(Dembele, Bellatreche, & Ordonez, 2020)(Dembele, Bellatreche, Ordonez, Gmati, et al., 2020)(Guo et al., 2017)(Kunjir et al., 2012) develop energy models based on a modeling level (operator, pipeline, request) and a model execution (sequential, parallel, distributed). Subsequently, they validate these models by scrutinizing their precision through variations in Datasets or initial architectures. Following this validation phase, they undertake studies on estimation and energy conservation by incorporating the model into the query optimizer. *Our proposal in this work differs from the works suggested in the literature by the fact that we highlight the step of machine learning to derive the weights of the final model and conduct our experiments on an engine that offers an intra-parallelism mode. The step of machine learning is crucial because from our experiences, we understood that it can have a huge impact on the accuracy of the model. The current approach relies solely on one machine learning technique, whether it's simple linear regression or polynomial regression. We propose to confront three techniques: polynomial regression, random forest and neural networks*.

Virtualization, cloud computing, and co-processing methods are advancing and demonstrating considerable promise in the field of energy efficiency (Kaur & Chana, 2015),(Huh, 2018)(Quang-Hung et al., 2013). Additionally, data compression is also widely applied as an EE approach because by comparing the energy absorbed by the processor during the compression and decompression steps and the energy required in the input/output subsystems to load the data.

Temperature Management

The ideal temperature is adequate for the proper functioning of these components. Heat tends to increase hardware failures. The higher the amount of heat released, the more energy the cooling systems consume. The ambient temperature recommended by ASHRAE8 for Class A1 to A2 DSS is between 18^◦ and 27^◦ Celsius (64^◦ to 81^◦ Farenheit). Major corporations, including Google and Facebook, adopt a strategy of relocating data centers to regions with a temperate climate, often situated at the edges of oceans. While this approach undeniably decreases energy consumption, it raises concerns about potential pollution released into the water. The environmental repercussions of such discharges on the aquatic ecosystem and its inhabitants warrant careful consideration.

Standards and Conduct

In response to global warming, governments and major actors have imposed regulations with the objective of reducing greenhouse gas emissions and mitigating economic, safety and environmental impacts. For example, we have the code of conduct established in 2008 by the European Commission[1], the ENERGY STAR[2] program of the US Department of Energy and the holding of the Conferences of the Parties (COP3) organized by the United Nations (UN) annually.

Energy Efficiency during Physical Design

Physical design is one of the steps of the database lifecycle. The objective is to suggest optimization structures that will impact the performance of database applications. These optimization structures encompass the choice of storage format, selection of relevant indexes, query execution mode, definition of materialized views, denormalization of data, and use of data partitioning techniques, etc(March & Carlis, 1985). In this paper, we focus on data partitioning. Split into two classes: horizontal and vertical, partitioning significantly affects query performance. Our goal in conducting this experiment is to investigate whether the Partitioning technique, in addition to being efficient, is also energy-efficient or not.

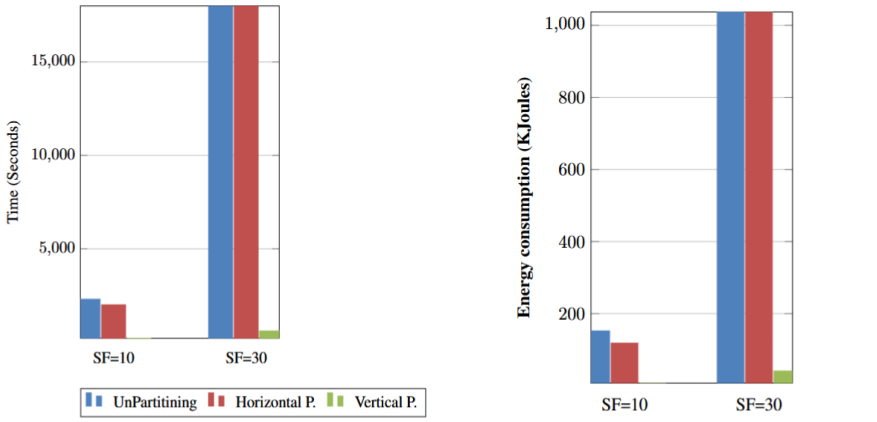
[1] https://energy.ec.europa.eu/topics/energy-efficiency/energy-efficiency-targets-directive-and-rules/energy-efficiency-directive_en

[2] https://www.energy.gov/eere/buildings/energy-starr

[3] https://unfccc.int/fr

To do this, using the PostgreSQL System, we collect the time and energy consumed by each of the 22 Select-Join-Projection queries of the TPCH benchmark on the same data in the different scenarios. In the first scenario, the data is not partitioned, in the second scenario, the data is partitioned horizontally using PostgreSQL's Auto-Partitioning based on the number of available nodes, and in the third scenario, the data is partitioned vertically. MonetDB's column-oriented storage model is used to partition the data vertically. We load the same data on this engine and evaluate its energy consumption.

We use a *Power meter* to measure the average energy consumed in each scenario and retrieve the execution time of the queries on both 10 GB and 30 GB Datasets. The figure 4 indicates, for each Datasets, the total query execution time and the total average energy consumed in each configuration.



(a) Processing time (b) Energy consumption

Figure 4: Time and Average energy consumption.

The energy savings achieved through Vertical Partitioning (VP) for all queries on Datasets SF10 are 20 times smaller compared to the scenario with No-Partitioning, and 24 times smaller on Datasets 30. In the case of Horizontal Partitioning, the energy savings are 29 times lower compared to the scenario with No-Partitioning on Datasets SF10 and 0.02 times higher on Datasets SF30. From the experimentation results, we find that the performance of Vertical Partitioning greatly exceeds the case of No-Partitioning and Horizontal Partitioning in terms of query execution time and energy consumption. The choice made by database administrators in the physical design step can have a considerable impact on performance and energy consumption.

Thus, we conclude that Vertical Partitioning is a way to optimize not only the performance, but also the energy consumption for analytical queries

POWER MODEL PROFILING

Designing

Creating an energy model can be a laborious and time-consuming task as it necessitates a comprehensive understanding of various aspects, including the deployment platform and the specifics of the storage system (such as the data model, storage structure, execution mode, type of requests, etc.). In this manuscript, we refrain from delving into these procedures as they have been thoroughly addressed in our earlier publications (Dembele, Bellatreche, Ordonez, & Roukh, 2020) (Dembele et al., 2018) (Dembele, Bellatreche, & Ordonez, 2020) (Dembele, Bellatreche, Ordonez, Gmati, et al., 2020) (Dembele, Ladjel, et al., 2020). One of the primary aims of this paper is to introduce and analyze the results of our energy conservation efforts, utilizing the model we have established, and assess the influence of learning techniques on the model accuracy.

$$Model_{Query}^{Energy} = \left\{ \begin{matrix} Cost_{ElemOP}^{CPU} \oplus \\ Cost_{ElemOP}^{Disk} \oplus \\ Cost_{ElemOP}^{Memory} \oplus \\ Cost_{ElemOP}^{Network} \oplus \\ \dots \end{matrix} \right\} (E, P^n, ModeEx, ModelEx) + \varepsilon \quad (4)$$

where $ElemOP$ represents the number of operators constituting the query plan, E is the metric that we wish to evaluate, P^n is a list of characteristics which describes the component, $ModeEx$ is the execution mode and $ModelEx$ is the execution model.

Following our methodology (Figure 5) and considering a set of parameters that can affect the model's lifespan, we present a generic formulation (Equation 4) for energy models in the context of DSS when handling queries. The formalism results from the combination of physical resources (CPU, Memory, Disk, bus, network, etc.) invoked when processing operations constituting the query plan in a specific environment impacted by the execution model (pipeline, etc.) and the execution mode (sequential, etc.). The main challenge lies in the inability to determine or estimate each parameters value directly in isolation mode.

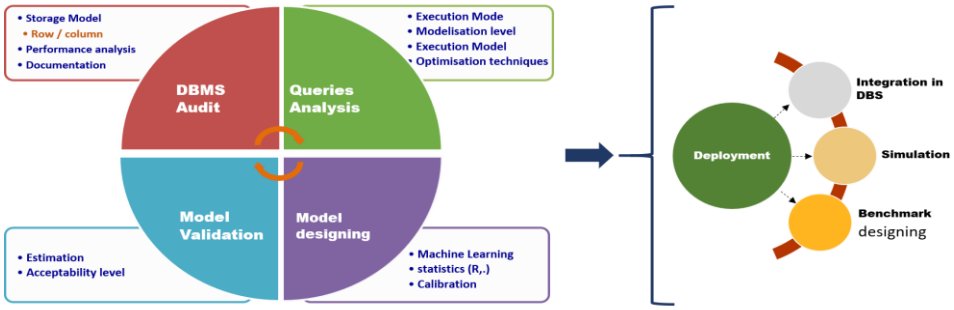


Figure 5: Energy cost model designing methodology.

We deploy our BD in a centralized environment on a single machine therefore we ignore the cost of communication. In our model, we consider a parallel execution mode (Intra-Query) and a pipeline execution model.

For a query Q_i, we denoted by $Plan_i$ its execution plan consisting of k pipelines noted $\{PL_1^i, PL_2^i, PL_3^i, \dots PL_k^i\}$, its average power is estimated as follows:

$Power(Q_i) = \frac{\sum_{j=1}^{k} Power\left(PL_j^i\right) * Time\left(PL_j^i\right)}{Time(Q_i)}$, where $Time(Q_i)$, $Time\left(PL_j^i\right)$ represent respectively, the execution time of the query Q_i and the execution time of PL_j^i. The power dissipated when processing a pipeline is the combination of the energy consumption of the main resources identified. The formula is given by the equation 5.

$$Power(PPL_i, DoP) = \left(f_{c_{DoP}} \oplus 1\right) * \beta_{CPU} * \sum_{k=1}^{n} C\,T_{CPU_k} \oplus \lambda_{IO} * \sum_{k=1}^{n} C\,T_{ES_k} \quad (5)$$

where β_{CPU} and λ_{IO} are the model parameters (i.e., unit power costs) for the operators. The CT_{ES_k} is the predicted number of I/O required for executing a specific operator. The CT_{CPU_k} is the predicted number of CPU Cycle and buffer cache get that DBMS needs to run a specific operator. The n is the number of operators in the pipeline. $f_{c_{DoP}}$ denote an energetic factor. This factor is defined according to the degree of parallelism (DoP) and expresses the difference between the power consumed during the parallel mode processing compared to the sequential processing at the CPU level(Dembele, Bellatreche, Ordonez, & Roukh, 2020). $\oplus$ expresses the correlation between the parameters.

Hardware Configuration

The experiments are conducted using a DELL PRECISION T1700, Intel Core i7 4770 CPU@ 3.40GHz (1 CPU-4Core-8 Threads), 16Go DDR3 main memory, ATA Disk Western Digital (WD 500 GB). The operating system Ubuntu 18.04 bionic (kernel 5.0.0-27-generic) is used and the PostgreSQL system (version 10.10), an open-source row-store. We collect statistics about average time and power consumption for each query in the particular DBMS. For power measurment, we use power meters called Watt-UP-PRO[4] at a 1Hz frequency, placed between the power source and the DBs server. In addition to the training Datasets and the TPC-H benchmark queries, we use the Datasets and queries from the SSB and TPC-DS benchmarks.

[4] http://www.energyalternatives.ca/pdf/wattsup_TTW.pdf

Machine Learning

Following a sequence of observations, during which the selected queries for model training were executed sequentially, we apply regression, Random Forest, and neural network algorithms in the R language to ascertain the parameter values for the models.

Based on our experiments, the application of the polynomial regression technique of degree 2 yielded acceptable results. The average power of a pipeline is then written as :

$$\begin{aligned} Power(PPL_i, DoP) &= \beta_1 * CT_{IO} + \beta_2 * CT_{CPU} * f_{c_{DoP}} \\ &+ \beta_3 * CT_{IO}^2 + \beta_4 * CT_{CPU}^2 * f_{c_{DoP}} \\ &+ \beta_5 * CT_{IO} * CT_{CPU} * f_{c_{DoP}} \\ &+ \beta_0 + \epsilon \end{aligned} \quad (6)$$

ϵ represents measurement errors and $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ are regression coefficients that will be estimated by machine learning technique.

In order to construct the other models (Neural Networks, Random Forest), we carried out multiple experiments until we determined the most optimal hyper-parameters. The hyper-parameter values used in both cases are summarized in Table 1(b) and 1(c). In order to assess the quality of the models, we used R software to calculate statistical quantities that measure the adequacy of the learning data.

In Table 1(a), we provide either the coefficient of determination denoted by R^2 or the mean square error for each model.

Table 1: Accuracy analysis

(a) Models accuracy

	NLR		ANN		RF	
	(1) R-squared	(2) RMSE	(3) RMSE	(4) MAPE	(5) R-squared	(6) RMSE
Values	0.66	4.81	1.94	3.38	0.72	4.55

(b) Random Forest arguments

	Random Forest (RF)		
	Mtry	Ntree	Nodesize
Values	2	8	14

(c) Neural Network Arguments

	Artificial Neural Network (ANN)							
	(1) Neural Structure	(2) Hidden layer	(3) Steps	(4) Algorithms	(5) Error function	(6) Activation function	(7) Lifesign	(8) Stop threshold
Values	2-4-1	1	100000	RProp-	SSE	Logistic	Minimal	0,02

Validation Study

We used Estimation Error (ERR) methodology to quantify the accuracy of ours models. It consists to determine difference between the Real Energy (RE) measured by Powermeter and the Energy Estimated(EE) in percentage. It is expressed by the following formula for a single query:

$$Error_{Qi} = \frac{|RE - EE|}{RE} \times 100\% \quad (7)$$

To measure the accuracy of the model on a query set, we determine the mean of the errors (E_{Mean}) of predictions formulated as follows:

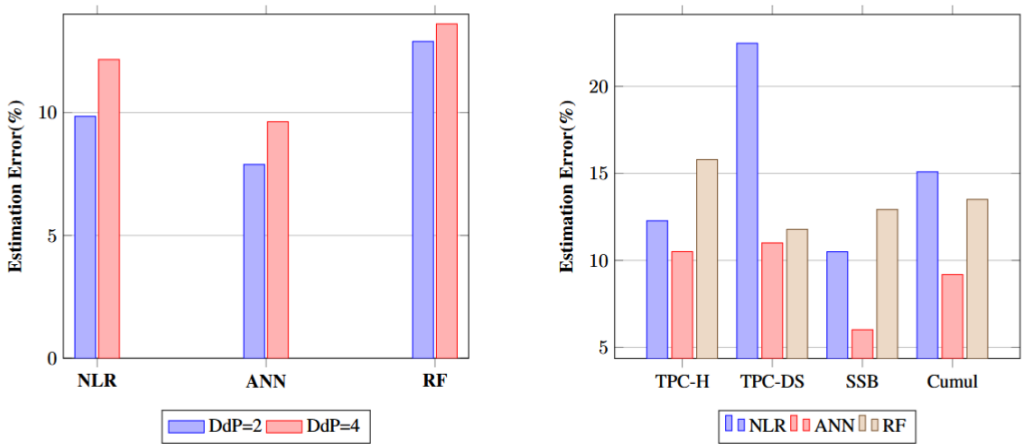
$$E_{Mean} = \frac{1}{n} \times \sum_{i=1}^{n} E\,rror_{Qi} \quad (8)$$

In order to compare the effectiveness of learning methods, we will use the average of prediction errors on a set of queries.

When working with a 10 GB TPCH Datasets, the mean errors differ depending on the degree of parallelism. Specifically, when utilizing the nonlinear regression technique, the mean errors are 9.85% and 12.16% for degrees 2 and 4, respectively. Meanwhile, the Neural Networks technique yields mean errors of 7.89% and 9.63% for degrees 2 and 4. Lastly, the random forest regression technique (FAR) results in mean errors of 12.90% and 13.61% for degrees 2 and 4, respectively.

With the 20 GB TPCDS schema, characterized by greater complexity compared to TPCH, we achieved an average error below 15% across all learning techniques, with the nonlinear regression technique exhibiting a maximum error of 30%.

Utilizing a 100 GB Datasets with the less complex SSB schema as compared to TPC-H, we observed an average error of 12.98% and a maximum interpretable error of 18.56% when employing the nonlinear regression technique. Meanwhile, when using neural networks, we achieved an average error of 7.76% with a maximum error of 13.11%.



(a) TPCH SF=10 GB Datasets.

(b) TPCH, TPC-DS, SSB Datasets

Figure 6: The average estimation error across the three techniques.

Discussion

As depicted in Figure 6, our experiments reveal that the Neural Network (ANN) technique exhibits superior accuracy compared to NonLinear Regression (NLR) and Random Forests (RF). While the accuracy of Neural Networks tends to enhance with larger Datasets, it is noteworthy that Random Forests and NonLinear Regression are more straightforward to train and optimize due to their hyperparameters. In the context of an optimizer's utilization, the ability to seamlessly integrate a model is crucial, as it should remain lightweight to avoid disrupting the energy of the system it seeks to estimate. While the Neural Network technique (ANN) proves suitable for energy estimation studies, it presents certain drawbacks in query optimization due to its computational time and energy consumption considerations.

However, this study unmistakably illustrates that the selection of the learning technique significantly influences the precision of the resultant model. It must be determined beforehand, taking into account other factors such as integration flexibility, prediction time performance, and the operational objectives of the model.

TOWARDS GREEN DATABASES: GREENPIPELINE

This section outlines the design and showcases our experimental results using the tool we introduced, named "GreePipeline." Our tool was created atop the PostgreSQL system, incorporating modifications to accommodate our cost model and query plan evaluation model. To enhance user-friendliness, we introduced a Graphical User Interface (GUI) for defining specific parameter values and real-time visualization of energy consumption for queries. The workflow of our Framework, developed in Java, is detailed in Figure 7. The tool(demo available on GreenPipeline) is available on https://github.com/dembeles/GreenPipelineDB.git.

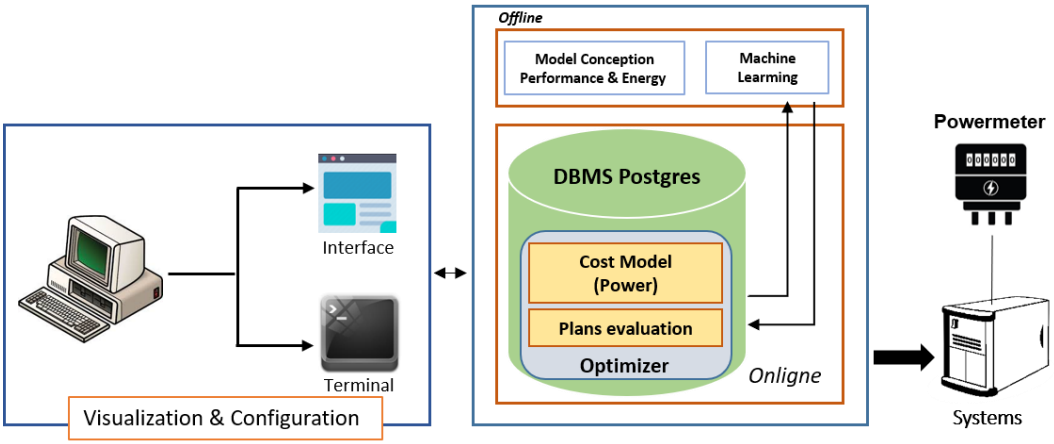


Figure 7: Framework Workflow.

The evaluation model is defined using a criterion that assesses either performance superiority (T) or energy cost (E). It is defined as follows:

$$C_{plan} = \alpha \times log(T) + (1 - \alpha) \times log(E).$$

C_{plan} represents cost of the plan. α serves as a compromise coefficient between T (performance) and E (energy cost), ranging from 0 to 1. Extreme values of α can lead to the degradation of one of the metrics. An optimal value for α enables the generation of execution plans with both acceptable performance and energy consumption.

Results

In our experimental setup, we chose to study three scenarios:

- *α is set to 1, with time_wt* $= 1$ *at 1 and power_wt* $= 0$ *at 0 :* the configuration underscores that optimizing performance is the primary objective during query processing.
- *α is set to 0, with time_wt* $= 1$ *at 0 and power_wt* $= 0$ *at 1 :* the configuration signifies that the primary optimization goal during query processing is energy efficiency.
- *α is set to 0.5, with time_wt* $= 1$ *at 0.5 and power_wt* $= 0$ *at 0.5 :* this configuration aims to strike a balance between the two factors. An execution plan is chosen that fulfills the energy constraint without compromising performance.

Additionally, besides varying the scale factor (SF) across different benchmarks, we perform TPC-H and SSB benchmark queries on each specified configuration ($\alpha = 1$, $\alpha = 0$, $\alpha = 0.5$). Power values and execution statistics for each query are collected for every configuration. Figure 8 illustrates the real-time variations in energy consumption during the first 100 seconds, based on the results obtained from the optimizer. Analyzing energy consumption variations, it becomes evident that the performance-oriented configuration $\alpha = 1$ unsurprisingly consumes the highest amount of energy.

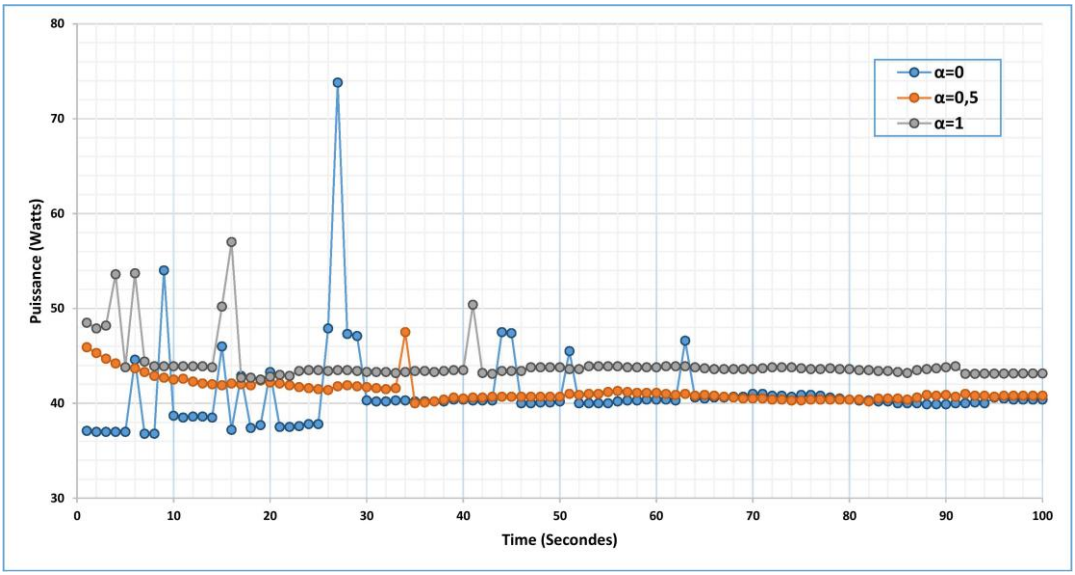


Figure 8: Variations in energy consumption captured in real time.

Table 2 presents a summary of the average energy dissipation and the processing time for query loads across each configuration. Additionally, the table provides a concise overview of energy savings, power variations, and performance degradation for the energy-oriented configurations ($\alpha = 0$, $\alpha = 0.5$) in comparison to the performance-oriented configuration ($\alpha = 1$).

Comparing the optimized workloads in terms of power and performance, we note power savings of 1 to 14 watts, resulting in overall power savings ranging from 2.7% to 21.08%.

The $\alpha = 0$ configuration is evidently more energy-efficient than other configurations, exhibiting approximately 1.5 times greater energy efficiency than the $\alpha = 1$ configuration. However, the energy-centric $\alpha = 0$ configuration is marked by a performance degradation. Despite observing a similar performance dip in the $\alpha = 0.5$ configuration, we achieved a cumulative energy savings of 26.4%.

Our proposed approach, validated through a series of experiments, demonstrates an average energy saving of 12.5% for the $\alpha = 0$ configuration and 13.9% for the $\alpha = 0.5$ configuration

Table 2: Assessment of query execution in various configurations.

Workload		Tradeoff	Power	Time	Energy	Economy		Performance
Benchmarks	Size	(α)	(In Watts)	(Seconds)	(KJoules)	Power	Energy	Degradation
SSB	15 Go	1	73.18	1606.66	117.58	-	-	-
		0	63.95	1682.61	107.60	12.6%	8.5%	-4.7%
		0.5	68.57	1644.63	112.77	6.3%	4.1%	-2.4%
TPC-H	1 Go	1	53.28	30.77	1.64	-	-	-
		0	44.50	35.38	1.57	16.5%	4.0%	-14.9%
		0.5	51.89	35.84	1.86	2.7%	-	-16.4%
TPC-H	10 Go	1	59.55	2337.03	139.17	-	-	-
		0	58.34	3786.19	220.87	2.0%	-	-62.0%
		0.5	46.52	2696.77	125.45	21.8%	9.8%	-15.4%

CONCLUSION

Undoubtedly, this paper serves as compelling evidence that achieving Energy Efficiency (EE) is feasible by incorporating best practices gleaned from prior research into performance optimization. Our approach commenced by substantiating, via a sequence of experiments, that the utilization of the partitioning technique significantly enhances both performance and energy efficiency. Subsequently, we introduced our framework, named "GreenPipeline," representing a pioneering step towards eco-friendly databases. The distinctive feature of this framework lies in an cost model integration, capable of accurately predicting the energy cost associated with a given plan, and an evaluation model designed to select the most optimal plan based on predefined optimization criteria.

As our approach relies on cost-based principles, it becomes imperative to develop a precise model for selecting the optimal plan. Given the complexity and needs for reproducibility, we outlined two key aspects. Firstly, we presented a step-by-step guide for formulating an energy model tailored to the execution of queries within the realm of data storage systems. Secondly, we introduced a generic formulation including various factors that have the potential to influence energy consumption. Furthermore, this study discuss the influence of learning techniques on the model's precision and addresses the challenges associated with integration based on the targeted objectives.

After the design and implementation of our framework, using diverse benchmarks such as TPCH and SSB, and measuring actual energy consumption with a wattmeter, we noted energy savings of up to 26.4%. This paper reveals that the application of some good practices in the context of databases encourages the exploration of energy saving opportunities.

We plan to extend our work by exploring the energy consumption of recursive (computation-intensive) queries and maintenance queries in a more sophisticated environment than a centralized system that we used in the context of this work. The feedback from the application of this extension will allow us to export our skills by evaluating the energy consumption of the SparkSQL processing system and its Catalyst optimizer engine.

REFERENCES

[1] Beloglazov, A., Buyya, R., Lee, Y. C., Zomaya, A., et al. (2011). A taxonomy and survey of energy-efficient data centers and cloud computing systems. *Advances in Computers*, *82*(2), 47–111. https://doi.org/10.48550/arXiv.1007.0066

[2] Dembele, S. P., Bellatreche, L., & Ordonez, C. (2020). *Towards green query processing - auditing power before deploying*. DOI: 10.1109/BigData50022.2020.9377819

[3] Dembele, S. P., Bellatreche, L., Ordonez, C., Gmati, N., Roche, M., Nguyen-Huu, T., & Debreu, L. (2020). Thinking smart [big steps towards query eco-processing - thinking smart]. *ARIMA J.*, *34*, 7. DOI : 10.46298/ARIMA.6767

[4] Dembele, S. P., Bellatreche, L., Ordonez, C., & Roukh, A. (2020). Think big, start small: A good initiative to design green query optimizers. *Cluster Computing*, *23*(3), 2323–2345. DOI : https://doi.org/10.1007/s10586-019-03005-0.

[5] Dembele, S. P., Ladjel, B., & Carlos, O. (2020). *Vers le traitement et optimisation écologique des requêtes*. CARI 2020 - Colloque Africain sur la Recherche en Informatique et en Mathematiques Apliquees

[6] Dembele, S. P., Roukh, A., & Bellatreche, L. (2018). Vers des optimiseurs verts de requêtes en mode parallèle. *Business Intelligence & Big Data, 14ème Edition de La Conference EDA, Tanger, Maroc*, *B-14*, 179–194.

[7] Guimarães, M., Saraiva, J., & Belo, O. (2016). Some heuristic approaches for reducing energy consumption on database systems. *DBKDA 2016*, 59.

[8] Guo, B., Yu, J., Liao, B., Yang, D., & Lu, L. (2017). A green framework for DBMS based on energy-aware query optimization and energy-efficient query processing. *Journal of Network and Computer Applications*, *84*, 118–130.DOI: https://doi.org/10.1016/j.jnca.2017.02.015

[9] Haj-Yahya, J., Alser, M., Kim, J., Yaglikçi, A. G., Vijaykumar, N., Rotem, E., & Mutlu, O. (2020). SysScale: Exploiting multi-domain dynamic voltage and frequency scaling for energy efficient mobile processors. *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, 227–240. DOI:10.1109/ISCA45697.2020.00029

[10] Huh, J.-H. (2018). Server operation and virtualization to save energy and cost in future sustainable computing. *Sustainability*, *10*, 1919.

[11] Jaiantilal, A., Jiang, Y., & Mishra, S. (2010). Modeling CPU energy consumption for energy efficient scheduling. *Proceedings of the 1st Workshop on Green Computing*, 10–15.DOI: https://doi.org/10.1145/1925013.192501

[12] Kamatkar, S. J., Kamble, A., Viloria, A., Hernández-Fernandez, L., & Cali, E. G. (2018). Database performance tuning and query optimization. *Data Mining and Big Data: Third International Conference, DMBD 2018, Shanghai, China, June 17–22, 2018, Proceedings 3*, 3–11.DOI: https://doi.org/10.1007/978-3-319-93803-5_1

[13] Kaur, T., & Chana, I. (2015). Energy efficiency techniques in cloud computing: A survey and taxonomy. *ACM Comput. Surv.*, *48*. DOI: https://doi.org/10.1145/2742488

[14] Kilgore, G. (2023). *Carbon footprint of data centers*. 8BillionTrees. https://8billiontrees.com/carbon-offsets-credits/carbon-ecological-footprint-calculators/carbon-footprint-of-data-centers

[15] Kunjir, M., Birwa, P. K., & Haritsa, J. R. (2012). Peak power plays in database engines. *EDBT*, 444–455.DOI: https://doi.org/10.1145/2247596.2247648

[16] Liu, X., Wang, J., Wang, H., & Gao, H. (2013). Generating power-efficient query execution plan. *2nd International Conference on Advances in Computer Science and Engineering (CSE 2013)*.

[17] Lu, Y.-H., & De Micheli, G. (2001). Comparing system level power management policies. *IEEE Design & Test of Computers*, *18*(2), 10–19. DOI: 10.1109/54.914592

[18] Mar, H. (2022). *Energy and water consumption in data centers: Sustainability risks*. ieee.es. https://www.ieee.es/Galerias/fichero/docs_analisis/2022/DIEEEA69_2022_MARHID_Datos_ENG.pdf

[19] March, S. T., & Carlis, J. V. (1985). Physical database design: Techniques for improved database performance. *Query Processing in Database Systems*. doi.org/10.1007/978-3-642-82375-6_17 https://api.semanticscholar.org/CorpusID:38669348

[20] Nishikawa, N., Nakano, M., & Kitsuregawa, M. (2015). Application sensitive energy management framework for storage systems. *IEEE Transactions on Knowledge and Data Engineering*, *27*(9), 2335–2348. https://doi.org/10.1109/TKDE.2015.2416737

[21] Pagani, S., Pathania, A., Shafique, M., Chen, J., & Henkel, J. (2017). Energy efficiency for clustered heterogeneous multicores. *IEEE Transactions on Parallel and Distributed Systems*, *28*(5), 1315–1330. DOI: 10.1109/TPDS.2016.2623616

[22] Quang-Hung, N., Thoai, N., & Son, N. T. (2013). Enery efficient allocation of virtual machines in high performance computing cloud. *CoRR*, *abs/1310.7801*. https://doi.org/10.1007/978-3-662-45947-8_6

[23] Rodriguez-Martinez, M., Valdivia, H., Seguel, J., & Greer, M. (2011). Estimating power/energy consumption in database servers. *Procedia Computer Science*, *6*, 112–117. DOI: https://doi.org/10.1016/j.procs.2011.08.022

[24] Roukh, A., Bellatreche, L., Bouarar, S., & Boukorca, A. (2017). Eco-physic: Eco-physical design initiative for very large databases. *Information Systems*, *68*, 44–63. DOI: https://doi.org/10.1016/j.is.2017.01.003

[25] Salami, B., Noori, H., & Naghibzadeh, M. (2020). Fairness-aware energy efficient scheduling on heterogeneous multi-core processors. *IEEE Transactions on Computers*, 1–1. DOI: 10.1109/TC.2020.2984607

[26] Tsirogiannis, D., Harizopoulos, S., & Shah, M. A. (2010). Analyzing the energy efficiency of a database server. *Sigmod*, 231–242. https://doi.org/10.1145/1807167.1807194

[27] Xu, Z., Tu, Y.-C., & Wang, X. (2015). Online energy estimation of relational operations in database systems. *IEEE Transactions on Computers*, *64*(11), 3223–3236. DOI: 10.1109/TC.2015.2394309

[28] Zhou, L., Bhuyan, L. N., & Ramakrishnan, K. K. (2020). Swan: A two-step power management for distributed search engines. In *Proceedings of the ACM/IEEE international symposium on low power electronics and design* (pp. 67–72). https://doi.org/10.1145/3370748.3406573

[29] Zhou, Y., Taneja, S., Alghamdi, M., & Qin, X. (2018). Improving energy efficiency of database clusters through prefetching and caching. *2018 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, 388–391. https://doi.org/10.1109/CCGRID.2018.00065