

#### Vol 11, Issue 2, February 2024

# GreenPipeline: Data-Driven in the Light of Energy Constraints

# <sup>[1]</sup> Simon Pierre DEMBELE

<sup>[1]</sup>Institute of Computer Science, University of Tartu Corresponding Author Email: <sup>[1]</sup>simon.pierre.dembele@ut.ee

Abstract— The high energy consumption has now become a critical and urgent issue for database community. Query optimizer are identified as one of the most energy hungriest component in Data Storage Systems (DSSs). This is mainly due to the users' needs to ingest, store and process data very frequently. With the torch of environmental sustainability being waved and the exorbitant cost of energy, the development and application of energy reduction techniques within these systems is more urgent than ever. In this paper, we propose a tool baptized "GreenPipeline" that enhances the query execution engine of the PostgreSQL system by integrating energy constraints during Analytical Queries processing. This initiative aligns with the development of environmentally friendly databases and has two primary objectives: to evaluate the energy efficiency benefits achieved by incorporating an energy model into query processing and to provide users a means to estimate the system's energy consumption without requiring physical equipment. More specifically, our primary goal is to integrate our energy cost model and an evaluation plan methodology into the core of the PostgreSQL optimizer. We offer an evaluation plan approach that enables the optimization of either energy efficiency, performance, or a compromise between the two. Preliminary results highlight the effectiveness of our tool in reducing energy consumption for various query sets from different benchmarks.

Keywords: Data storage, optimization, energy model, energy-aware query processing, benchmarking.

#### I. INTRODUCTION

The pursuit of eco-efficient solutions to mitigate the economic cost and environmental degradation resulting from the substantial energy consumption of computer systems has captured the attention of researchers, both in academic and professional environments. Some strategies explored to address these issues are categorized into two approaches: hardware-based and software-based approaches. In addition to Dynamic Energy Management (DEM) and the utilization of virtualization techniques in the software approach, it has been observed that it is feasible to regulate energy efficiency at a more specific level within the internal architecture of Database Management Systems (DBMS) [6][7].

Indeed, Data storage systems architectures encompass a set of interconnected components, each playing a specific role in routine operations such as data retrieval, modification, construction, organization, transformation, backup, or data restoration. Among these components, the query processing engine stands out as one of the most complex. On one hand, it is tasked with efficient access and manipulation of massive data, and on the other hand, it interacts with other phases related to the selection and maintenance of optimization structures, such as indexing methods [8].



Fig. 1: Energy distribution among different components.

Data storage systems have been recognized as one of the primary contributors to energy consumption in Data center. This energy consumption is attributed to various elements, including servers, storage devices, networks, and supporting infrastructure components [9][11][1][2]. **Fig. 1** provides a representation of the power distribution within a Data center.

The integration of energy constraints begins with the identification of points within the DSS where integration and exploitation are feasible. One of the key integration points is the query processing system. In comparison to traditional database systems, where the choice of query execution plan is primarily based on a performance-focused cost model, *Green databases* must process data in an *energy-efficient manner*. One approach is to select the final plan based on an energy model with the aim of minimizing the overall data storage system's energy consumption while preserving performance and user-friendliness.

This approach necessitates the development of precise energy cost models that consider various parameters related to the database components, such as the database schema (table sizes, tuple lengths, etc.), query workload (query type,



## Vol 11, Issue 2, February 2024

join selectivity factors, selection predicates, intermediate result sizes, etc.), and hardware characteristics (number of CPU cores, buffer size, disk page size, etc.).

In this paper, we introduce a benchmark for energy-aware query optimization, referred to as "GreenPipeline". This benchmark harnesses the PostgreSQL query optimizer and incorporates the energy dimension into the optimization process. Our approach is inspired by the research presented in [10], where the authors present a model for sequential query processing. We enhance their work by capitalizing on the intra-parallelism capabilities introduced in PostgreSQL since version 9.6 on multi-core architectures, enabling intra-query parallelism processing. To achieve this, we have developed and integrated mathematical cost models that estimate energy consumption, considering the parallel processing mode within the query optimizer. Following an extensive series of experiments to evaluate the energy savings achieved through our approach. We also present a user-friendly graphical interface, the interface serves as a diagnostic tool for end-users, developers, and database administrators (DBAs), empowering them to enhance their awareness of energy consumption.



Fig. 2: Parallel Plan Illustration for PostgreSQL.

#### **II. MOTIVATION**

The main motivations of this work are, on one hand, financial and well-being benefits: energy efficiency contributes to the reduction of  $CO_2$  emissions, which has a positive impact on socio-economic and health aspects, especially in terms of the sustainability and the health of individuals. On the other hand, the goal is to examine the

feasibility of this approach and assess potential energy savings.

Traditional optimizers rely on a cost model to evaluate the performance of each plan and select the most optimal one. The primary objective of this approach is not energy optimization but rather runtime. PostgreSQL's optimizer executes a query by segmenting it into a set of pipelines, delimited by blocking operators. The introduction of the intra-parallel execution mode enables the optimizer to generate intra-parallel execution plans based on the chosen number of workers. To force the optimizer to generate a plan using three workers, it is necessary to initialize the "*max\_parallel\_workers\_per\_gather*" variable to 3 in the query planner. **Fig. 2** provides an illustration of the execution plan for the following query:

# **SELECT** sum(sales) **FROM** customers **WHERE** customers.city='London';

For each query, the optimizer generates a certain physical execution plan. Each plan consists of a set of operators that define a unique way of extracting data stored in secondary memory. Each operator is characterized by algorithmic implementation, input and output parameters, its degree of parallelism, and so on. Therefore, different plans may utilize system resources differently. To verify this observation in terms of system energy consumption, we estimate the energy cost of a query by varying the degree of parallelism (the number of workers) from 0 to 4. As illustrated in **Fig. 3**, the degree of parallelism (**DoP**) impacts both the time and energy of the chosen plan.

In light of these findings, the proposal for a green framework that integrates an energy model into the query optimizer could lead to energy optimizations by selecting a plan that minimizes energy consumption.

#### **III. QUERY MODELS DEFINITON**

The approach towards an energy-efficient optimizer during query processing relies on the definition and incorporation of two distinct models. The first model should have the ability to accurately and flexibly estimate the energy cost of a query by incorporating all relevant parameters related to query operations. The second model is dedicated to the selection of a plan that meets the specific needs of the user.

#### A. Energy Cost Model

We describe briefly our model that estimate the energy cost. The main steps of building green query processors are:

- 1) Identification of relevant energy-sensitive parameters belonging to hardware and software components,
- 2) Elaboration of mathematical cost models estimating consumed energy when executing a query on a target DBMS,
- 3) Setting of values of the energy-sensitive parameters using machine learning techniques.



### Vol 11, Issue 2, February 2024

We employ non-linear regression techniques for estimating our energy cost units and utilize linear regression to determine the energy factor added when utilizing multiple cores. Comprehensive information on energy modeling steps can be found in our papers [5][4][3].

During the Query (Q) plan execution, which consists of a set of pipelines, two primary actions come into play:

- The CPU carries out a series of instructions for each operator, and the workload is influenced by the number of tuples. This parameter is referred to as the CPU cost and is denoted by *C*<sub>CPU</sub>. In a parallel plan, the CPU cost is distributed among the cores according to the degree of parallelism (*DoP*).
- A specific number of data blocks are transferred from secondary memory to main memory, which is referred to as Disk access number Input/Output (I/O), denoted by *C*<sub>1/O</sub>.

Therefore, for the query (Q) represented by the plan (*Plan*) consisting of k pipelines denoted as  $PL_1, PL_2, \ldots, PL_k$ , the energy cost is computed as follows:

$$Power(Q, DoP) = \frac{\sum_{i=1}^{k} Power(PL_i, DoP_i) * Time(PL_i, DoP_i)}{Time(Q)}$$

where parameters Time(Q),  $Time(PL_i, DoP_i)$  represent respectively, the execution time of Q and the time of PLi with degree  $DoP_i$ .

To estimate the energy consumption in parallel mode, we introduce an energetic factor denoted fc. This factor will be defined according to the degree of parallelism and expresses the energic difference between the power consumed during the parallel mode processing compared to the sequential processing at the CPU level.

The following formula defines this factor:  $f_{cp} = (P_{mp} - P_0)/P_0$ . We denote by  $P_{mp}$  the average power dissipated on n cores to treat a query and  $P_0$  the power consumed on the sequential plan (degree of parallelism set to 0). The following equation:  $f_{cDoP} = \alpha * DoP + \beta + \epsilon$  define the energetic factor between one and several cores obtained by applying the relationship defined by the simple linear regression technique. DoP is the degree of parallelism which correspond to the number of cores involve in the query processing et  $\epsilon$  represents the estimation error. The parameters  $\alpha$  and  $\beta$  are regression coefficients that will be estimated.



Fig 3: Query Q3 plan cost associated to Degree of Parallelism 0, 2, and 4 with  $W_{time} = 1 W_{Power} = 0$ 

The power dissipated when processing a sequential and parallel pipeline is the combination of the energy consumption from the parameters identified. The formula is:

$$Power (PPL_i, DoP) = (f_{C_{DOP}} + 1) * W_{CPU} * \\ \sum_{k=1}^{n} C_{CPU_k} \bigoplus W_{I/O} * \sum_{k=1}^{n} C_{I/O}$$
(1)

where  $W_{CPU}$  and  $W_{I/O}$  are the model parameters (i.e., unit power costs) for the operators. The  $C_{I/Ok}$  is the predicted number of I/O required for executing a specific operator. The  $C_{CPUk}$  is the predicted number of CPU Cycle and buffer cache that DBMS needs to run a specific operator. The *n* is the number of operators in the pipeline.

Before integrating our model into the optimizer, we conducted a quality assessment. This evaluation involved determining the prediction error, which was calculated by comparing the measured energy consumption (using a wattmeter) with the estimated energy consumption in percentage. To do this, we utilized datasets and queries from the TPC-H, SSB, and TPC-DS benchmarks, details can be found in [5].

#### **B.** Plan Evaluation Mode

Rather than choosing a plan with optimal performance, the energy cost model can be used to choose a plan that saves energy or to define a certain threshold of the trade-off between energy and performance. The adjustable trade-off between performance and energy is made by using a criterion that reflects the choice of users/ administrators. The criterion model adopted has the following formulation:

$$C_{plan} = \alpha \times log(T) + (1 - \alpha) \times log(P)$$
<sup>(2)</sup>

with *T* denote the performance, *P* denote the energy cost, *Cplan* denote the plan cost and  $\alpha$  is constant value in interval [0, 1]. This coefficient reflects the superiority of performance (T) or energy cost (P). In the relationship described by equation 2, extreme values of the criterion  $\alpha$  lead to the deterioration of one of the factors. We can select plans with



# Vol 11, Issue 2, February 2024

desired characteristics for different optimization goals with the choice of  $\alpha$ :

```
Cost count_power_cost_NLR(double io_cost, double cpu_cost, int wh)
2
 3
                // io_cost : Cout du DISK
 4
                // cpu_cost: Cout du CPU
5
                // wh: Degre de parallelisme
 6
               Cost power_cost = 0.0;
               double fcd=wh*0.0552+0.00783;
 8
0
10
                if(wh!=0){
11
                 power_cost = B0
12
                   pow(io_cost, 1) * pow(cpu_cost, 0) * B1 + pow(io_cost, 2) * pow(cpu_cost, 0) * B3
                   (pow(io_cost, 0) * pow(cpu_cost, 1) * B2 + pow(io_cost, 0) * pow(cpu_cost, 1) * fcd * B2)
(pow(io_cost, 0) * pow(cpu_cost, 2) * B4 + pow(io_cost, 0) * pow(cpu_cost, 2) * fcd * B4)
13
                 +
14
15
                   (pow(io_cost, 1) * pow(cpu_cost, 1) * B5 + pow(io_cost, 1) * pow(cpu_cost, 1) * fcd * B5);
16
17
               else {
                 power_cost = B0
18
19
                 + pow(io_cost, 1) * pow(cpu_cost, 0) * B1 + pow(io_cost, 0)* pow(cpu_cost, 1) * B2
20
                   pow(io_cost, 2) * pow(cpu_cost, 0) * B3 + pow(io_cost, 0) * pow(cpu_cost, 2) * B4
21
                 + pow(io_cost, 1) * pow(cpu_cost, 1) * B5;
22
23
24
                return power_cost;
```



- *α* is set to 1, so *time\_wt* = 1 and *power\_wt* = 0: the configuration underscores that optimizing performance is the primary objective during query processing.
- *α* is set to 0, so *time\_wt* = 0 and *power\_wt* = 1: the configuration signifies that the primary optimization goal during query processing is energy efficiency.
- $\alpha$  is set to 0.5, so *time\_wt* = 0.5 and *power\_wt* = 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.

#### C. Models integration

To incorporate the energy cost model and the evaluation model, adjustments need to be made in the source code of the PostgreSQL system. These adjustments involved several files, with particular attention paid to the following two files:

- **Path:**"src/backend/optimizer/path/costsize.c": This file contains all the functions to estimate the costs of each algebraic operator. We retrieve the I/O and CPU costs for each operator, and then estimate its power using the function that implements our cost model within the file. **In Fig. 4**, you will find the code for the *count\_power\_cost\_NLR()* function, where Bi represents the regression coefficients, and DdP denotes the degree of parallelism.
- **Path:**"src/backend/optimizer/util/pathnode.c": We modify the following functions *compare\_fractional\_path\_costs()* and *compare\_path\_costs()* so that the plan selection is being based on our evaluation model which combines the two constraints.

## **IV. SYSTEM ARCHITECTURE**

The GreenPipeline architecture consists of two main components: a back-end and a Graphical User Interface (GUI). The back-end comprises a Database Management System (DBMS) that integrates our energy cost model and a criterion model reflecting user and administrator preferences. The GUI assists users in configuring settings and displaying query results. In our research, we utilized PostgreSQL, which supports parallel query execution. This tool was developed in a Java environment. The workflow of our Framework is described in the **Fig. 5**.



**Fig 5:** Workflow of GreenPipeline Framework

#### A. Experimental design

The adjustable parameters and the free license character make PostgreSQL the perfect tool for our experiments. As mentioned before on PostgreSQL, its open source character allows us to modify this kernel to validate the efficiency of our model. We use the PostgreSQL release 10.3 for our experiments. A graphical interface (GUI) part allowing to



### Vol 11, Issue 2, February 2024

define the values of certain parameters, a part to visualize in real time the energy consumption of the demands and a part to monitor and measure the values of the performance meters. The user can compare and decide what algorithm enable or disable. The tool (demo available on https://www.youtube. com/watch?v=aAaoBnQsz-0) is available on https://github. com/dembeles/GreenPipelineDB.git. **Fig. 6** provides an overview of settings and outputs interface. Using the underlying system does not necessitate the GUI; it is solely

designed to showcase various potential interactions with the system. The module enables you to accomplish the following tasks:

- Settings of system parameter values
- Plan visualization
- Visualization of consumption timeline, energy prediction values and Query results.

		PQEcoP		- 0	• ®	PQEcoP	8		PQEcoP	
iery EcoPro Que	y Explain Statist	ics Settings				Query EcoPro Query Explain Statistics Settings		Query EcoPro Query Explain Statist	ics Settings	
$\sim$	PowerMel	ler			_	₹Single Query				
(ha)	Driver P Port	ath /usr/iocal/wat	tsup	_		colort a name c name count/it as pombre disat		Power Real Values		
Le B	Che	ek				from customer, region, nation		Average Power:	- Watts	
U	Tradeoff					and n_regionkey = r_regionkey or n_hun hun harme r_regionkey n_name			Index	
stgreSQL	Power 0	Performance 1	Set			group by nonene, remains order by remaine,		Average chergs	- Judies	
onnection		Machine Lean	ning Teechr	niques	_			Status	Apparell non connecte	
ostname lio	alhost	Regression f	olynomiale M	lultiple						
tabasa te		Artificial Nei     Rendom Fai	ural Network					Power Values Estimated		
t [54	32		e			_Multiple Queries		Machine Learning Method		
ername po	itgres	Query				File Path (Justitest/	and and a			
sword pr	itgres	_ Parallel mode				Results Open	8	Polynomial Regression		
Connexion		Degre of para	il le lism	Set		Query1		Value 79.81 Watts		
Data			-			ALGERIA		EER -		
fo	DB Info					ARGENTINA, Desktop Pictures P BRAZIL Documents Public	console.log	Neural Network		
192e [74]	MB	Size Attribute	as Cardina	Page		CANADA Downloads R CHINA	hsqlprefs.dat model.nls	Value -		
es Ø	nation	24 kB	4 0			EGYPT ETHIOPIA Entities		FFR		
oding UTF	part	18 MB	9 100000	2048		GERMANY NetBeansProjects in Videos				
k size 819	orders	16 MB 118 MB	9 750000	13050		INDIA INDONESIA File <u>N</u> ame:		Random Forast		
	Loartsupp	27 Mil	5 400000	R7.2% [2]	21	IRAN IRAQ Files of Type: All Files		Value -		
nizer tuning Parallel Scan	Ø Er	nable Merge Join	En	able NL/		JAPAN JORDAN		EER -		
		nable Hash Join	🗹 Pa	rallel Brmap H	Head	•	Open Cancel			
Parallel Index 9	un esti									
Parallel Index 5 Parallel Hosh J ry EcoPro Que	v Explain Statist	PQEcoP ics [Settings]	⊌ Pa	– D	* *	Pun Deplan Power Estimation PowerTimeline PQEcoP Quarte EcoPre 2 Quarte EcoPre	- c &	c	reenPipeline	- 0
Parallel Index 5 Parallel Hash J ry EcoPro Que eries statisti	r Explain (Statist	PQEcoP ics Settings Display Gra	⊻ Par	– D	es	Pun         Deplan         Power Estimation         PowerTimeline           PQEcoP         Query EcoPro:         Query EcoPro:         Statistics:         Satistics:         Satistics:           Visingle Query         Visingle Query         Visingle Query         Visingle Query         Visingle Query		SQL Query Execution PL	reenPipeline an Power Timeline	– C Settings
Barallel Index 5 Barallel Hash J EcoPro   Que ies statisti ary1	v Explain ( Statist	PQEcoP Ics Settings Display Gra	⊯ Pau	- C	- 8 es	Run         Diplian         Power Estimation         PowerTimeline           PQEcoP         Query Ecolina         Statistics         Statistics           VSingle Query         Statistics         Statistics         Statistics           VSingle Query         Statistics         Statistics         Statistics           Participation         Statistics         Statistics         Statistics	- • »	Q SQL Query Execution Pl V Parallel Settings	reenPipeline an Power Timeline	– C Settings
arallel Index 5 arallel Hash J EcoPro Que ies statistic ary1	v Explain Statist	PQEcoP ics [Settings] Display Gra	i Pau ph i spli	t in Pipeline		Bun         Deption         Power Estimation         Power Estimation           PQEcoP         Query Explain         Statistics         Settings           VStragle Query         Statistics         Settings           VStragle Query         Settings         Vision of the power of the	_ • *	C SQL Query Execution PL ♥ Parallel Settings Degre Of Parallelism (Dof	reenPipeline an Power Timeline	– 🗆 Settings
coPro Que es statisti y1 Node	explain Statist	PQEcoP ics Settings Display Gra CPU IO Cost IO Cost	Power Cost	t in Pipeline	es	Bun         Depinn         Power Estimation         Power Estimation           PQEcoP         Quary Ecolification         Quary Ecolification         Power Ecolification           VSimple Quary         Statistics:         Satistics:         Satistics:         Satistics:           Values         Courty Ecolification         Satistics:         Satistics:         Satistics:         Satistics:           Values         Courty Ecolification         Satistics:         Satistics:         Satistics:         Satistics:           Values         Courty Ecolification         Satistics:         Satist		SQL Query Execution Pl Parallel Settings Degre Of Parallelism (Dof Fragmentation Tecnical	reenPipeline an Power Timeline b) 2 Range	– C
allel Index S allel Index S oPro Que s statisti Q Node Aggrega	cost e 4487.18	PQEcoP rcs Settings Display Cra Cost O Cost O Cost 0.0	Power Cost 0.91	t in Pipeline Rows	es	Run         Digition         Power/Estimation         Power/Estimation           PQEcoP         Query Ecolific         Query Explain         Statistics         Sattings           Visingle Query         Query Ecolific         Query Ecolific         Sattings         Visingle Query           Fellect n_name         name countril jai nombre_client         Sattings         Sattings           existent n_name         name countril jai nombre_client         Sattings         Sattings           existent n_name/station         name countril jai nombre_client         Sattings         Sattings		SQL Query Execution Pl Parallel Settings Degre Of Parallelism (Dof Fragmentation Tecnical Workload Queries	reenPipeline an Power Timeline ) 2 Range	– C Settings
oPro Que s statisti Node Aggrega Gather	e 4487.18	PQEcoP Ics Settings Display Crast Cost 433.5 0.0 73.35 0.0	Power Cost 0.91	t in Pipeline		Run         Deplan         Power Estimation         Power Estimation           PQEcoP         Query Ecolific         Query Explain         Statistics         Settings           VSingle Query         Query Ecolific         Query Ecolific         Settings         Visingle Query           Psied to the region radion and region		SQL Query Execution Pl ✓ Parallel Settings Degre Of Parallelism (Dof Fragmentation Tecnnical ☐ Workload Queries Workload Path:	reenPipsline an Power Timeline by 2 Range build-postgres-demo/	– C Settings
illel Index S illel Index S illel Hash J oPro [ Que s statistic s statistic ] Node Aggrega Gather Merge	e 4487.18 3981.43	PQEcoP ics Settings Display Cra Cost 4335 0.0 72.25 0.0	Power Cost 0.91 0.91	t in Pipeline Rows 28900 28900	es	Run         Depinn         Pear Estimation         PoserEstimation           PQEcoP         PQEcoP         PQEcoP           Quary EcoPro:         Quary Explain         Statistics         Sattings           VStragle Quary         Statistics         Sattings         Sattings           Statistics         Sattistics         Sattings         Sattistics           VStragle Quary         Sattistics         Sattistics         Sattistics           Sattistics         Sattistics         Sattistics         Sattistics           Sattistics         Sattistics         Sattistics         Sattistics           Sattistics         Sattistics         Sattistics         Sattistics           Sattistics         Sattistics         Sattistics         Sattistics	- • *	SQL Query Execution Pl Parallel Settings Degre Of Parallelism (Dof Fragmentation Tecnical Workload Queries Workload Path: Concurrency Level (MPL).	reenPipeline an Power Timeline b) 2 Range build-postgres-demo/	_ C Settings
oPro Que s statisti Node Aggrega Gather Merge Aggrega	r Explain Statist e 4487.18 3981.43 e 730.18	PQEcoP           cs_Settings           Display Gra           CPU         Lo           433.5         0.0           72.25         0.0           509.59         0.0	Power Cost 0.91 0.91	- Rows 28900 28900 28900	es	Run         Bigsinn         Power Estimation         Power Entimation           PQEcoP         Quary Explain         Statistics         Settings           Winingte Quary         Statistics         Settings         Settings           Winingte Colore         Quary Explain         Statistics         Settings           Winingte Colore         Quary Explain         Statistics         Settings           Informations         Figure 1, paints         Settings         Settings           Informations         Figure 1, paints         Settings         Settings           Informations         Figure 1, paints         Figure 1, paints         Settings           Informations         Figure 1, paints         Figure 1, paints         Settings		C SQL Query Execution PL Parallel Settings Degre of Parallelism (Dor Fragmentation Tecnnical Workload Queries Workload Path: Concurrency Level (MPL). Query Status	reenPipeline an Power Timeline ) 2 Range  build-postgres-demo/ 2 Rows	_ C Settings
illel Index 5 lilel Hash J oPro Que s statisti Que s statisti Que a statisti Que Aggrega Gather Merge Aggrega Sort	Cost 4487.18 3981.43 c 730.19 110.29	PQEcoP ics Settings CPU IO Cost IO Cost 4335 0.0 72.25 0.0 500.59 0.0 110.3 2500.0	<ul> <li>♥ Fai</li> <li>Power Cost</li> <li>0.91</li> <li>0.91</li> <li>0.91</li> <li>0.91</li> <li>0.91</li> </ul>	Rows         28900           28900         28900           44118         44118	55 (1)	Run         Explain         Power Estimation         Power Estimation           PQEcoP         Query Explain         Statistics         Statistics           Visingle Query         Participant Statistics         Statistics         Statistics           Visingle Query         Participant Statistics         Statistics         Statistics           Visingle Query         Participant Statistics         Statistics         Statistics           Visingle Query         Protection         Statistics         Statistics           vising contractive or protection         Protection         Statistics         Statistics           vising contractive or protection         Protection         Protection         Statistics           - Mathiple Query         Protection         Protection         Protection         Protection		SQL Query Execution PL Parallel Settings Degre Of Parallelism (Dor Fragmentation Tecnical Workload Queries Workload Path: Concurrency Level (MPL). Query Status	reenPipeline an Power Timeline 2 Range puild-postgres-demo/ 2 Rows	_ C Settings
oPro Que s statisti Q Aggrega Gather Merge Sort	Cost a 4487.18 3981.43 a 730.18 110.29 2000	PQEcoP CPU PQ Cost Cost 433.5 0.0 72.25 0.0 10.3 2500.0 0.0	Paulon Power Cost 0.91 0.91 0.91 0.91 0.91	Rows         1           28900         28900           28900         28900           44118         44118		Run         Explain         Power Estimation         Power Estimation           PQEcoP         Quary EcoPire         Quary Explain         Statistics         Settings           Visingle Quary         Quary EcoPire         Quary EcoPire         Settings         Visingle Quary           Visingle Quary         Power Enderskip         Settings         Visingle Quary         Settings           Visingle Quary         Power Settings         Power Settings         Settings         Settings           visingle Quary         Power Settings         Power Settings         Power Settings         Power Settings          Mathiple Quar         Power Line         Reatings         Power Line         Power Line           File Path         Reatings         Reatings         System Power Line         Power Line	ver Consumption	SQL Query Execution Plu Parallel settings Degre of Parallelism (Dof Fragmentation Tecnical Workload Queries Workload Path: Concurrency Level(MPL); Query Status	reenPipeline an Power Timeline bi Power Timeline puild-postgres-demo/ 2 Rows	_ C Settings tpch/tmp ;
allel Index Sallel	Cost           e         4487.18           3981.43         9           e         730.18           110.29         2840.8	PQEC0P rcs Settings Display Cra Cost 00 Cost 00 Cost 00 Cost 10 Cost	Parent Power Cost           0.91           0.91           0.91           0.91           0.91	Rows         28900           28900         28900           28900         28900           44118         44118		Run         Explain         Power Estimation         Power Firmine           PQEcoP         Quary Explain         Statustics         Satustics         Satustics           V Single Quary         Statustics         Satustics         Satustics         Satustics           Satustics         Satustics         Satustics         Satustics         Satustics         Satustics           V Single Quary         Satustics         Satustics         Satustics         Satustics         Satustics           Satustics         Satustics         Satustics         Satustics         Satustics         Satustics           Satustics         Satustics         Satustics         Satustics         PowerLines         Satustics           - Multiple Quart         PowerLines         Readime System PowerLines         Satustics         Satustics         Satustics	ver Cosumption	SQL Query Execution Pl Parallel Settings Degre Of Parallelism (Dof Fragmentation Tecnical Workload Queries Workload Path: Concurrency Level (MPL). Query Status V Single SQL Query	reenPipeline an Power Timeline b) 2 Range putd-postgres-demo/ 2 Rows	- C Settings
coPro Que es statisti Aggrega Gather Merge Sort Hash Joli Seq Sca	Cost         Statist           e         4487.18           3981.43         9981.43           e         730.19           110.29         2840.8           2234.18         2234.18	PQEcoP           ics         Settings           Display         Cost           Cost         Cost           4335         0.0           72.25         0.0           509.59         0.0           101.03         2500.0           606.62         0.0           441.16         1793.0	Power Cost 0.91 0.91 0.91 0.91 0.92 0.92	Rows 28900 28900 28900 44118 44118		Run         Explain         Power Estimation         Power Estimation           PQEcoP         Query Explain         Statistics         Settings           Visingle Ouery         Particular Statistics         Settings         Visingle Ouery           Visingle Ouery         Particular Statistics         Settings         Visingle Ouery           Visingle Ouery         Protein Statistics         Settings         Visingle Ouery           vising Charles         ranse r. ranse, ranse, count (1) as nombre glant         setting - Settings         Visingle Ouery           vising Charles         ranse r. ranse, count (1) as nombre glant         setting         Reatifies           Multiple Ouer         PowerLine         Reatifies         System PowerLine           Marciaritical         1.3         Accounting         Accounting	· · · · · · · · · · · · · · · · · ·	C SQL Query Execution PL Parallel Settings Degre of Parallelism (Dof Fragmentation Tecnnical Workload Queries Workload Queries Workload Path: Concurrency Level (MPL). Query Status Single SQL Query select	reenPipeline an Power Timeline b) 2 Range build-postgres-demo/ 2 Rows	_ C Settings
coPro Que es statisti ya Aggrega Gather Merge Aggrega Sort Hash joli	Cost         Cost           a         4487.18           3981.43         9           110.29         2840.8           2234.18         0.0	CPUEco           CSU         Cost           CSU         Cost           4335         0.0           7225         0.0           50055         0.0           1003         2500.0           606.62         0.0           411.10         1793.0           2.34         0.0	Power Cost 0.91 0.91 0.91 0.91 0.92 0.91 0.91	Rows 28900 28900 28900 44118 44118 170		Run         Explain         Power Estimation         Power Timeline           POEcoP         Query Ecoline         Query Epolen         Statistics         Statistics           V Single Query         Participant Anthe Country as nonbre client         Timeline         Timeline           Validation         Participant Anthe Country as nonbre client         Timeline         Timeline           Validation         Prove Ontoney         Prove Ontoney         Prove Ontoney           Image: Participant Anthe Country         Prove Ontoney         Prove Onto	wer Consumption	SQL Query Execution PL Parallel Settings Degre Of Parallelism (Dor Fragmentation Tecnical Workload Queries Workload Queries Workload Path: Concurrency Level (MPL) Query Status Single SQL Query Select Lorderkey,	reenPipeline an Power Timeline ) 2 Range puild-postgres-demo/ 2 Rows	_ C Settings
coPro Qua es statisti yil Node Aggrega Aggrega Sort Hash Joli Seq Sca Hash	Cost a 4487.18 3981.43 c 730.19 110.29 2840.8 2234.18 0.00 14.04	PQEcoP CPU PO Cost Cost 433.5 0.0 72.25 0.0 509.56 0.0 110.3 2500.0 606.62 0.0 441.18 1793.0 2.34 0.0	Power Cost 0.91 0.91 0.91 0.91 0.92 0.91 0.91 0.91	Rows         Image: Control of the	es	Run         Explain         Power Estimation         Power Timeline           PUT         Query Explain         Statistics         Satistics           Visingle Query         Query Explain         Statistics         Satistics           Visingle Query         Power Extended         Satistics         Satistics           Visingle Query         Power Extended         Satistics         Satistics           visingle Query         Power Extended         Power Extended         Power Extended           visingle Query         - regordway         - regordway         Power Extended         Power Extended           - Multiple Query         - regordway         Readtime System Power Extended         No         No           - Multiple Query         - Satistics         System Power Extended         No         No           - Multiple Query         - Satistics         System Power Extended         No         No         No           - Multiple Query         - Satistics         Satistics         System Power Extended         No	wer Consumption	SQL Query     Execution Pli       ✓ Parallel settings       Degre Of Parallelisn (Dof       Fragmentation Tecnical       Workload Queries       Workload Path:       Concurrency Level (MPL).       Query       Status       ✓ Single SQL Query       select       Lorderkey, o_orderdate       Lorderkey, o_orderdate	reenPipeline an Power Timeline 2 Range puild-postgres-demo/1 2 Rows	_ C Settings
EcoPro Qua ies statisti- ery) Aggrege Gather Merge Sort Hash Join Seq Sca Hash	Cost           a         4487.18           3991.43         a           a         2497.18           3991.43         a           a         2490.8           a         2234.18           a         0.0           a         14.04	PQECP rcs Settings Display Cra Cost 00 Cost	Power Cost 0.91 0.91 0.91 0.92 0.91 0.91 0.91 0.91 0.91 0.91 0.91 0.91 0.91	Rows         Image: Control of the		Run Explan Power Estimation PowerTimeline  PQEcoP  Query Ecotro Query Explan Statutics Sattings  VSingle Query  Select, name r, name, count(*) as nombre, client Fride ratio  Pride ratio  PowerLine  Fride ratio  PowerLine  Pride ratio  PowerLine  PowerLine PowerLine PowerLine PowerLine PowerLine PowerLine PowerLine PowerLine PowerLine PowerLine PowerLine PowerLine PowerLine PowerLine PowerLine PowerLi	ver Consumption	SQL Query Execution Pl SQL Query Execution Pl Parallel Settings Degre of Parallelism (Dof Fragmentation Tecnical Workload Queries Workload Path: Concurrency Level (MPL). Query Status Select Lorderkey, o_credrade o_shiponont	reenPipeline an Power Timeline b) 2 Range putd-postgres-demo/ 2 Rows N	_ C Settings
EcoPro Que  EcoPro Que  EcoPro Que  EcoPro Que  EcoPro Que  Aggrege  Gather Merge  Aggrege  Sort Hash Jol  Seq Sca Hash Hash Jol	Cost Cost 4487.18 3991.43 4487.18 3991.43 2840.8 110.29 2840.8 2234.19 0.0 14.04	PQEcoP ics Settings □ Display Gra Cost 0 Cost 0 Cost 10 Cost 10 Cos	Power Cost 0.91 0.91 0.91 0.91 0.91 0.91 0.91 0.91	Rows         Image: Control of the second secon		Run Explan Power Estimation Power Timeline POEcoP Query EcoPro Query Explane Statistics Settings Vising Query Setting Course Setting Course Setting Course Setting Course Pro Parts Pro Pa	ver Consumption	C SQL Query Execution PL Parallel Settings Degre of Parallelism (Dof Fragmentation Tecnnical Workload Queries Workload Queries Single SQL Query Select Lorderkay, o_orderdate o_shispriorit Execution Output	reenPipeline an Power Timeline b) 2 Range build-postgres-demo/ 2 Rows v	_ C Settings
EcoPro Qua EcoPro Qua Esservi Node Aggrege Aggrege Sort Hash Jol Seq Sca Hash Hash Jol Esservi Esservi Esservi Esservi Aggrege Sort Esservi Esservi Esservi Aggrege Sort Esservi E	Cost (22418) 3981.43 4487.18 3981.43 4487.18 3981.43 110.29 2840.8 1224.18 0.0 14.04 14.04	CPUECOF           Display Craw           Cost         OCost           Cost         OCost           4335         0.0           7225         0.0           50959         0.0           103         2500.0           606.62         0.0           411.10         1793.0           2.34         0.0           2.34         0.0	Power Cost 0.91 0.91 0.91 0.91 0.91 0.91 0.91 0.91	Rows           28900           28900           28900           28900           24118           44118           170           170           170           18126 48		Run         Explain         Power Estimation         PowerTimeNet           PUT         Outry Explain         Statistics         Statistics         Statistics           Visingle Query         Parlact name name name country sectors         Statistics         Statistics         Statistics           Visingle Query         Parlact name name name name name name name name		SQL Query     Execution PL       ♥ Parallel Settings       Degre Of Parallelism (Dor       Fragmentation Tecnical       ₩orkload Queries       ₩orkload Path:       Concurrency Level (MPL):       Query       Status       ♥ Single SQL Query       Select       L orderkey, o_orderdate       o shipointit       Execution Output	reenPipeline an Power Timeline ) 2 Range  putte-postgres-demo/ 2 Rows	_ C Settings
EcoPro Qua EcoPro Qua Instantisti mya Aggrega Gather Merge Sort Hash Joli Hash Joli I stim	Cost 22401 448718 398143 6730.19 110.29 2840.8 223418 0.0 110.29 2840.8 22448 110.29 2840.8 22448 0.0 23418 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.	PQEcoP CPU Cost COS Cost 433.5 0.0 72.25	Power           Cost           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.92           0.91           0.91           0.91           0.92           0.91           0.91           0.92           0.91           0.91           0.92           0.91           0.92	Rows         Rows           28900         28900           28900         28900           28900         28900           28900         28900           120         170           170         170	es •	Run Egein Poer Estimation Poer Timeline  PQEcoP  Query Ecolina Statistics Settings  V Strajdo Cuery  Select n_name_rame, count(*) as nombre_client from out-mere, region autom and n_region (*) a		SQL Query     Execution Pli       ✓ Parallel Settings       Degre Of Parallelism (Dof       Fragmentation Tecnical       Workload Queries       Workload Path:       Concurrency Level (MPL).       Query       Status       ✓ Single SQL Query       Select       LorderKey, o orderdate o shipprioriti       Execution Output	reenPipeline an Power Timeline an Power Timeline Po	_ C Settings
EcoPro Qua EcoPro Qua Es statisti m/a Aggrega Cather Merga Sort: Hash joli Seq Sca Hash Hash Joli	Cost     A48718     A98718     Cost     A48718     A98718     A99143     Cost     A48718     A99143     Cost     A48718     A99143     Cost     A48718     A99143     Cost     A48718     A99144     Cost     A48718     Cost     Cost     A48718     A48718     Cost     A48718     Cost     A48718     A48718     Cost     A48718     A48718     Cost     A48718     A48718     Cost     A48718     A48718     Cost     Cost     A48718     Cost     A48718     Cost     Cost     A48718     Cost     Cost     A48718     Cost     Cost     Cost     Cost     A48718     Cost     Cost     Cost     A48718     Cost     Cost	Operation         Operation           Inc.         Settings           Inc.         Display Gra           Cost         Cost           Cost         Cost           439.5         0.0           72.25         0.0           509.59         0.0           606.62         0.0           441.16         1793.0           2.34         0.0           2.34         0.0           2.34         0.0	Power           Cost           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91	Rows         Rows           28900         28900           28900         28900           28900         44118           44118         170           170         170		Run Explan Poser Estimation PoserTimeline  POEcoP  Query Ecotro Query Explan Statistics Settings  VSinge Query Explan Statistics Settings  VSinge Query Explan Statistics Settings  Poser Sett	ver Consumption	SQL Query       Execution PL         ✓ Parallel Settings         Degre Of Parallelism (Dof         Fragmentation Tecnical         Workload Queries         Workload Path:         Concurrency Level (MPL);         Query         Status         ✓ Single SQL Query         Select         Lorderkey, o_crederdate         Lorderkey         Execution Output	reenPipeline an Power Timeline b) 2 Range pultd-postgres-demo/ 2 Rows v	_ C Settings
EcoPro Que EcoPro Que ies statisti- nyi A Aggrege Gather Merge Sort Hash Joli Seq Sca Hash Hash Joli	Cost         Cost           e         4487.18           3981.43         3981.43           e         70.19           110.29         2840.8           2224.18         0.0           14.04         14.04	CPUE cost         Settings           Display Cra         00           Cost         00           433.5         0.0           72.25         0.0           509.59         0.0           110.3         2500.0           606.62         0.0           2.34         0.0           2.34         0.0           441.16         1793.0           2.34         0.0	Power Cost           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91           0.91	Rows         Rows           28900         28900           28900         28900           28900         24118           44118         44118           170         170           16136.48         16136.48		Run Explan Power Estimation PowerTimeIme POEcoP Query Explan Statistics Sattings VSingle Query Select T, parse, parse, count(1) as nonbre-glent results aviers c-factories, region, ration aviers c-factories, region, ration, r	wer Consumption	C SQL Query Execution PL Parallel Settings Degre of Parallelism (Dof Fragmentation Tecnical Workload Queries Workload Queries Workload Queries Workload Queries Workload Queries Workload Queries Workload Queries Workload Queries Uorderksy, o_orderdate o_shispriorit Execution Output	reenPipeline an Power Timeline ) 2 Range  puild-postgres-demo/ 2 Rows  v v	Lain EX

#### **B.** Main results

We used the TPC-H and SSB benchmarks to assess our proposal across different scaling factors. For each scaling factor, we evaluated the average energy consumption and the time required to process a query sets on each configuration. We compared the power variation and performance degradation of energy-oriented configurations ( $\alpha = 0$ ,  $\alpha = 0.5$ ) with that of performance-oriented configuration ( $\alpha = 1$ ). We noted that the  $\alpha = 0$  configuration exhibits superior power efficiency compared to the other configurations, approximately 1.5 times more efficient than the  $\alpha = 1$  configuration. However, the energy-focused  $\alpha = 0$ 

configuration takes more time to process queries, resulting in a performance degradation at that level. Although a deterioration in performance was observed, an energy saving of 26.4% is materialized. Through a series of experiments on the TPC-H and SSB benchmarks, our framework achieved an average energy saving of 12.5% in the configuration  $\alpha = 0$ and 13.9% in the configuration  $\alpha = 0.5$ .

#### V. CONCLUSION

The Green Pipeline Framework is an initiative aimed at promoting environmentally-friendly database development. It has two primary goals: to evaluate the energy efficiency advantages of integrating an energy model into query



s...developing tesear

## International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

# Vol 11, Issue 2, February 2024

processing and to offer users a means of estimating system energy usage without the necessity of physical equipment.

#### REFERENCES

- Consumer Research Associates. 2014 (accessed 2019.01.22). Energy Efficiency Policy Options for Australian and New Zealand Data Centres: E3 Equipment Energy Efficiency. Report. ENERGY RATING. http://energyrating.gov.au/ document/report-energy-efficiency-policy-options-australianand-new-zealand-data-centres.
- [2] Simon Pierre Dembele, Ladjel Bellatreche, Angelo Lorusso, Francesco Marongiu, and Domenico Santaniello. 2023. In-Memory Database Query Energy Estimation: Modeling & Green Strategy Support. In 2023 IEEE World Conference on Applied Intelligence and Computing (AIC). 278–285.
- [3] Simon Pierre Dembele, Ladjel Bellatreche, and Carlos Ordonez. 2020. Towards Green Query Processing - Auditing Power Before Deploying. IEEE.
- [4] Simon Pierre Dembele, Ladjel Bellatreche, Carlos Ordonez, Nabil Gmati, Mathieu Roche, Tri Nguyen-Huu, and Laurent Debreu. 2020. Thinking Smart [Big Steps Towards Query Eco-Processing - Thinking Smart]. ARIMA J. 34 (2020), 7.
- [5] Simon Pierre Dembele, Ladjel Bellatreche, Carlos Ordonez, and Amine Roukh. 2019. Think big, start small: a good initiative to design green query optimizers. Cluster Computing (30 Oct 2019).
- [6] Goetz Graefe. 2008. Database Servers Tailored to Improve Energy Efficiency. In Proceedings of the 2008 EDBT Workshop on Software Engineering for Tailor-Made Data Management (Nantes, France) (SETMDM '08). Association for Computing Machinery, New York, NY, USA, 24–28.
- [7] Stavros Harizopoulos, Mehul A. Shah, Justin Meza, and Parthasarathy Ranganathan. 2009. Energy Efficiency: The New Holy Grail of Data Management Systems Research. ArXiv abs/0909.1784 (2009).
- [8] Majid Khan and MNA Khan. 2013. Exploring query optimization techniques in relational databases. International Journal of Database Theory and Application 6, 3 (2013), 11– 20.
- [9] Emerson Liebert. 2007. Five Strategies for Cutting Data Center Energy Costs Through Enhanced Cooling Efficiency. White Paper. http://www.emersonnetworkpower.com/ documentation/en-us/brands/liebert/documents/white%20pap ers/data-center-energy-efficiency\_151-47.pdf
- [10] Amine Roukh, Ladjel Bellatreche, and Carlos Ordonez. 2016. EnerQuery: Energy-Aware Query Processing. In Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (Indianapolis, Indiana, USA) (CIKM '16). ACM, New York, NY, USA, 2465–2468.
- [11] Dimitris Tsirogiannis, Stavros Harizopoulos, and Mehul A Shah. 2010. Analyzing the energy efficiency of a database server. In sigmod. 231–242.