Energy Measurement System for Data Lake

Philippe Roose, Hernán H Álvarez Valera, Alexandre Maurice, Franck Ravat, Jiefu Song, Nathalie Vallès-Parlangeau

Energy Measurement System for Data Lake: An Initial Approach

Hernan Humberto ALVAREZ-VALERA[1[1]], Alexandre MAURICE[1[2]], Franck RAVAT[2[3]], Jiefu SONG[2[4]], Philippe ROOSE[3[5]], and Nathalie VALLES-PARLANGEAU[3[6]]

[1] Domolandes, Saint Geours de Maremne, France
{humberto.valera,alexandre.maurice}@domolandes.fr
[2] IRIT-UT Capitole, Toulouse, France {franck.favat,jiefu.song}@irit.fr
[3] LIUPPA, Anglet, France
{philippe.roose,nathalie.valles-parlangeau}@univ-pau.fr

Abstract. Data Lakes are increasingly deployed as a solution for Big Data analytics. Recent improvements in Data Lake technology have focused on improving data access, governance, and discoverability. However, the energy consumption of data operations, a non-trivial issue for eco-conscious organizations, is currently overlooked. Furthermore, existing monitoring tools do not adequately address the complexities of Data Lake architectures.
This paper presents the initial phase of developing a system for measuring energy in Data Lake pipeline operations. The novelty of our solution lies in the fact that we define four measures to assess the power usage of crucial hardware components in a Data Lake context: CPU, RAM, NIC, and storage devices. To validate our approach, we developed a monitoring tool grounded in real-world datasets from a Data Lake benchmark.

Keywords: Data Lakes · energy consumption · data processing.

1 Introduction

Big Data Analytics (BDA) involves advanced analytic techniques for processing large and diverse data sets from various sources. To enable accurate predictions and decision-making, BDA requires efficient data management solutions. Consequently, Data Lakes have become a prevalent solution for BDA nowadays.

Despite the importance of Data Lakes, current research, mainly focusing on metadata management [19, 12], data quality [1], and data discovery [2, 6], overlooks their energy consumption. This gap represents a real issue, as the Paris Agreement emphasizes the urgency of reducing greenhouse gas (GHG) emissions to ensure a sustainable future, to limit global warming to 1.5 degrees above pre-industrial levels [4].

To optimize Data Lake energy consumption, we must first measure all its activities granularly, considering all the concerning operating system processes

[4] https://unfccc.int/sites/default/files/english_paris_agreement.pdf

and sub-processes. This involves tracking activities like ingestion, transformation, and analysis. Operationally, a Data Lake begins by ingesting heterogeneous data from various sources, and storing it in its native format. The data is then processed (cleaned, transformed, etc.) to meet quality standards and business needs. Lastly, the Data Lake enables various access methods for analysis, including statistical, business intelligence, and machine learning applications.[13].

All these activities load differently four hardware components: CPU, RAM, Network Interface Controller (NIC), and Storage Devices, each with its different energy consumption profile. For instance, during the ingestion phase, the NIC manages data transfer, while storage devices (SSDs or HDDs) ensure data durability and accessibility. In the storage phase, these devices take on data organization, indexing, and retrieval. Finally, in the data processing phase, the CPU executes tasks such as data transformations and cleaning, with the RAM providing temporary storage for accessed data and temporary results.

The challenge is the lack of a comprehensive tool to precisely and granularly analyze the energy consumption of all Data Lake processes across the four mentioned hardware components.

In this article, we present a novel and comprehensive solution for measuring the energy consumption of a Data Lake. Our solution comprises a model describing the measurement of the four mentioned hardware components (CPU, RAM, NIC, SD). This model is integrated into a monitoring system for each operating-system process of all Data Lake activities. We evaluated our solution by analyzing the entire ingestion phase of the Audal data lake [14], which incorporates the *DLBench+* benchmark for these activities. The results display metrics on the energy consumption evolution of each hardware component according to different evaluation configurations.

The paper is organized as follows. Section 2 describes the state of the art, section 3 presents the model describing the energy consumption of Data Lake processes, section 4 illustrates our energy measuring system for a Data Lake, and finally, section 5 describes the experiments we performed and their corresponding results description.

2 Related work

Accurately measuring a Data Lake's energy consumption requires analyzing the energy usage of the four specified hardware components across all data lake activities. Currently, there is no solution for comprehensively measuring the energy consumption of Data Lakes. Yet, in the context of **database systems**, multiple works measure energy consumption for its optimization. This consumption is determined per query or workload through models, considering only CPU and disk usage [5]. Similarly, in **cloud environments**, research targets energy-saving strategies for data management, primarily focusing on CPU and disk usage [18]. Finally, with **frameworks** such as Hadoop, Yarn, and Spark, energy is analyzed across CPU, network devices, and disk, addressing cluster node management, data control, and resource allocation scheduling [17, 4, 9, 3, 11].

Since these solutions are not specifically designed for Data Lakes, the primary tools available for measuring their energy consumption are physical sensors and OS-level software solutions. Current physical sensors offer accurate energy measurements for overall systems. However, they lack the granularity needed to isolate and evaluate individual OS processes or specific hardware components components[7]. This detailed information is required for comprehensive and effective energy measurement for **exclusively** Data Lake processes.

For their part, recent OS-level software solutions such as Powerjoular[5] and Scaphandre[6] measure the power consumption of running OS processes. While they have proven effective in different scenarios, they mainly focus on CPU consumption. As mentioned, this approach offers a limited perspective on energy consumption, particularly when assessing Data Lakes where processes also load other important hardware components. Moreover, from our experiments in certain settings, we found that the power consumption did not align closely with the processor time taken by the process and its associated heat generation. For a more detailed explanation, please refer to our project repository[7].

All these studies and solutions offer good insights into energy consumption. However, Data Lakes, which are implemented by different frameworks, remain unexplored and need a more technology-agnostic approach. Moreover, current solutions don't consider simultaneously these four hardware components.

3 Power measurement model

A Data Lake activity (ingestion, transformation, analysis) corresponds to multiple processes within the operating system, each uniquely identified by a Process ID (PID). Assessing a Data Lake's energy consumption requires identifying and calculating the energy used by these processes.

We developed a model based on the mathematical formulas presented in [16]. This model correlates hardware workload with energy metrics across CPU, RAM, NIC, and disk. It details the power consumption (in Watts, W) of each hardware component, based on the load an operating-system process produces at a specific moment. For calculating energy (in Joules, J), the model incorporates the duration of the analysis (T), allowing the understanding of long-term efficiency and operational costs.

The CPU is the hardware component responsible for executing an OS process's instructions through logical and arithmetic operations. The time and energy required for these executions can vary based on the CPU's characteristics. To estimate a process's power and energy consumption across all the CPU cores ($\sum_{i=0}^{nCores} E_{CPU_i}^{PID}(t)$), the equation incorporates: 1) the CPU's capacitance based on the processor's **TDP** (thermal design power)[8], voltage, and frequency, with

[5]https://www.noureddine.org/research/joular/powerjoular

[6]https://github.com/hubblo-org/scaphandre

[7]https://github.com/hhumbertoAv/DL_Energy_System

[8]https://www.intel.com/content/www/us/en/support/articles/000055611/processors.html

a surplus of 0.7 [10], 2) The CPU's current frequency (f) and voltage (V), 3) the current ratio of process load to total CPU load: $(U^{PID}_{CPU_i})/(U_{CPU_i})$, 4) the CPU's fan energy consumption, proportional to frequency ($fans_s$), and 5) the time in seconds the CPU consumes power, used to calculate CPU energy consumption in Watts per second (T).

$$\sum_{i=0}^{no.Cores} E^{PID}_{CPU_i}(t) = \frac{0.7 \times TDP}{f_{TDP} \times V^2_{TDP}} \times f_i \times V_i^2 \times \frac{U^{PID}_{CPU_i}}{U_{CPU_i}}(t) + fans_s(T) \quad (1)$$

This equation is based on the principle that the primary power consumption of the CPU, like any integrated circuit, stems from the charging and discharging of its capacitors. This activity occurs during computations at a specific voltage and rate (frequency) [8].

The RAM is the hardware element that the CPU actively uses to read and write temporal data in a program execution context. To calculate the consumption of a process in the RAM, the formula correlates the number of read N_R and write N_W operations, and their corresponding energy usage, E_R, E_W.

$$E_{RAM_{app}} = N_R \times E_R + N_W \times E_W \quad (2)$$

The NIC and the storage devices are the hardware components that transmit and store data in a computer, respectively. In the context of a Data Lake, the transfer rate of the NIC directly affects the speed of data ingestion from external sources. Conversely, the transfer rates of storage devices impact data storage and retrieval performance, affecting both the organization of storage and subsequent data processing.

For both devices, the energy formula correlates energy (E_{D_p}) with the power consumption of the device when active (W_{u_D}), the process transfer rate as a fraction of the device's maximum transfer capacity, and the analysis time (T_p):

$$E_{D_p} = (W_{u_D} \times \frac{L_p}{L_{MAX_D}}) \times T_p \quad (3)$$

For all the formulas, energy consumption over a given time T is computed by averaging power measurements taken at regular intervals throughout T.

4 Energy measuring system for a Data Lake

Based on the model presented above, our Data Lake energy measurement system is implemented by a daemon-type application optimized for GNU/Linux systems. To obtain the hardware load data (CPU frequency, RAM operations, and I/O rates), it sources from the operating system's virtual file systems (sysfs, proc, and perf). Then, for power information, the daemon accesses data from datasheet

databases and operating system interfaces, such as the Linux kernel registers [16].

As Algorithm 1 depicts, the system deploys the daemon across every node of the Data Lake. During bootstrapping, users specify Data Lake process names or PIDs to be monitored. The system then tracks the energy use of these processes and, if configured, their subprocesses' power consumption to cover multi-processing and multi-threading contexts. Additionally, the system allows for configurable measurement frequencies during the model's window-time T to prevent CPU overloads caused by the monitoring daemon.

Algorithm 1: Energy Consumption Monitoring Procedure

```
   Input: List of service names in the Data Lake.
   Output: Energy consumption metrics (Joules)
 1 Initialization: Start the monitoring daemon.
 2 while daemon is running do
 3 |   foreach service name do
   |   |   /* Launch monitoring in a new thread                    */
 4 |   |   Start a new thread for service monitoring.
 5 |   |   Obtain main process PID.
 6 |   |   Get the list of all sub-process PIDs.
 7 |   |   foreach PID do
 8 |   |   |   if process with PID is active then
 9 |   |   |   |   Measure&record power consumption of CPU, RAM, NIC, SD.
10 |   |   |   |   Calculate power consumption (W), add it to a window period
   |   |   |   |     P, and wait the freq. time
11 |   |   |   |   Compute energy consumption (J) in P (J = W × P).
12 |   |   |   |   Store energy consumption metrics for the service.
13 |   |   |   else
14 |   |   |   |   Remove inactive PID from the monitoring list.
15 |   |   |   end
16 |   |   end
17 |   end
   |   /* Pause to minimize CPU contention                         */
18 |   Sleep for a configured time interval before the next scan.
19 end
```

5 Experimental evaluation

To test our system, we require a real benchmark implementation for Data Lake instances. These benchmarks are essential to evaluate operational efficiency across the (meta)data processing pipeline, considering different hardware setups and workloads. To our knowledge, *DLBench+*[14] is the only existing benchmark in this context. The goal of our experiments is to measure the energy use of the data ingestion in a Data Lake.

5.1 AUDAL Datalake and DLBench+

AUDAL [15] is a Data Lake instance built for textual and tabular document analysis. It uses MongoDB, Neo4J, SQLite, and ElasticSearch as (meta)data storage and management systems. Using *AUDAL*, *DLBench+* introduces an ingestion activity composed of several data and metadata functional operations described in Figure 1.

Regarding **data** operations, *DLBench+* first retrieves two large-volume raw data sets through three operations: In **OP1**, initially scans and sources scientific PDF files from the HAL[9] repository until reaching a specified document count. **OP2** then acquires a set of SQL tables with diverse data about educational institutions and bus stop boardings, among others. Finally, **OP3** converts all documents, including image-only ones, into readable text and stores them in an Elasticsearch index.

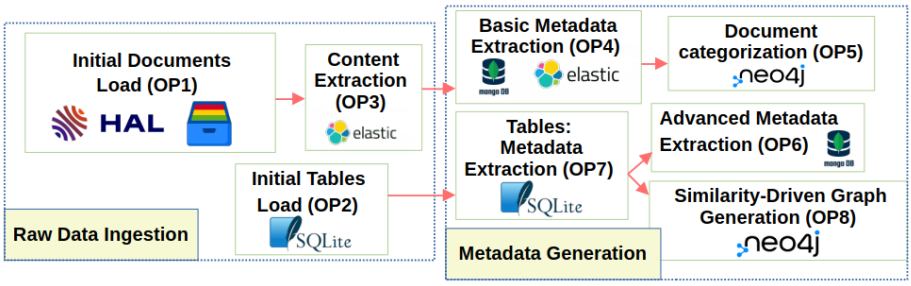

Fig. 1. Operations in DLBench+

Once DLBench+ ingests the raw data, it executes five operations to generate **metadata**. **OP4** first extracts the basic metadata from PDFs (like word dictionaries, titles, and authors, among others) for storage in Elasticsearch and MongoDB. Then, **OP5** creates nodes, category groups, and "in" relations in a Neo4j instance for each dictionary word. **OP6** processes metadata by removing stopwords, lemmatizing, tokenizing words, and establishing coincidence vectors in MongoDB. Following this, **OP7** extracts table metadata using initially downloaded data, creates MongoDB keywords from table values, and populates SQLite entries. **OP8**, lastly, forms Neo4j nodes for tables, columns, rows, refined tables, and names, defining relations based on Jaccard and Jaro-Winkler similarity metrics and potential primary/foreign key matches.

5.2 System Setup and Experimental Methodology

We deployed AUDAL and carried out energy evaluation of DLBench+ execution on a server with software and hardware specifications described in 5.2.

[9] https://hal.science/search/index

HARDWARE SETUP		
Hardware	**Spec.**	**Power Params.**
CPU	Intel Core I7-850H 2.20GHZ (12 vcores)	TDP: 45W
RAM	samsung M471A2K43CB1-CRC	Values in section XX
NIC	Cannon Lake PCH CNVi WiFi	Dowload Power: 0.55W Upload Power: 1.029 W
SD	Samsung MZVLW512HMJP	Write Power: 6.1 W Read Power: 5.1 W
SOFTWARE SETUP		
Software	**Spec.**	
O.S.	Ubuntu Linux - Kernel V. 6.2.0-26	
Frameworks	Neo4j 4.1.12, Elasticsearch 7.17.10, MongoDB 6.0.5, SQLite 3.37.2	

Table 1. Server's software and hardware configuration

We analyze two main variables. The first is the level of parallelism, determined by the number of dedicated cores concurrently running AUDAL's functional operations (from 1 to 5). To ensure this core allocation and to prevent measurement interference from our system's interruptions, we configured the process affinity[10] provided by GNU/Linux environments. This confines AUDAL processes to cores $1-5$ and restricts our tool exclusively to the rest of the cores.

The second variable is a scale factor associated with the volume ingested by the Data Lake. We evaluated scale factors of 1, 3, and 5, which correspond to processing an average of $5,000$, $15,000$, and $25,000$ documents, respectively. This translates to average data volumes of $4.3GB$, $12.5GB$, and $18.75GB$.

We executed all of DLBench+'s functional operations for every combination of parallelism level and scale factor. For each scale factor, we assessed the energy consumption starting with 1 core and scaling up to 5 cores. To ensure the reliability of our findings, we repeated the entire experiment sequence twice, with minor differences in the outcomes.

5.3 Energy measurement Results

After conducting the experiments, we observed distinct trends in processing time and energy consumption, influenced by parallel processing and scale factor evolution for data volume. This led to interesting differences in consumption across the four hardware components. The following paragraphs will detail the results and analysis, each accompanied by key insights gained.

Data vs. Metadata In DLBench+, at its maximum scale (18.75 GB of raw documents and 1.4 GB of raw tables) in 5 cores, metadata processing consumes 73.4% (1363.26 Joules) of the total energy, which amounts to 1859.55 Joules for both metadata and data processing combined. Data processing, in contrast, accounts for only 26.6% (496.29 Joules) of the total energy use. Data processing

[10] https://man7.org/linux/man-pages/man1/taskset.1.html

requires 9.56 hours, whereas metadata processing takes longer, at 12.62 hours. Despite the narrower time difference, metadata processing's energy consumption is significantly higher, demonstrating its greater resource intensity compared to data processing, as can be seen in the operations OP.

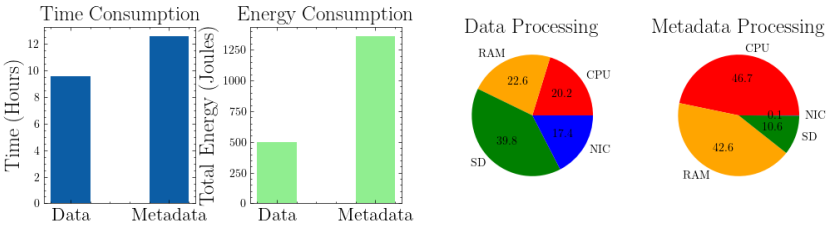

Fig. 2. Data vs. Metadata

On the other hand, as figure 2 shows, the energy consumption within metadata processing is primarily allocated to the CPU and RAM, with about 47% and 43% of its total energy used, respectively. In comparison, data processing distributes energy more evenly, with the largest share for SD storage at 40% and 17% for the NIC, indicative of I/O-oriented tasks. This distinction in energy usage highlights the different computational demands of metadata and data processing.

Key Insight: In a Data Lake, data and metadata processing consume energy and time unevenly. In DLBench+, data operations (OP1 - OP3), focusing on intensive I/O tasks like SQLite and Elasticsearch queries, use more energy in storage and NIC. In contrast, metadata operations (OP4 - OP8), such as document categorization and similarity graph generation using RAM-intensive technologies like Neo4j, place more stress on the CPU and RAM.

Parallelism: is it always a good idea? Using more cores may be expected to improve processing efficiency. However, as Figure 3 depicts at the right, DLBench+ results for the max scale data processing show that moving from 1 to 2 cores reduced processing time from 32.08 to 16.86 hours, a 47.5% improvement. By the fifth core, the time decreased further to 9.56 hours, a 41.6% reduction from 2 cores, which reveals the limits of parallel processing. For energy, the CPU's consumption reduced modestly from 105.10 Joules with 1 core to 100.49 Joules with 5 cores, a 4.4% decline, due to the independent parallel tasks in data processing (See: OP1 to OP3). The SD also presented a reduction from 212.54 to 197.46 Joules, a 7.1% decrease, as faster CPU task completions reduced read/write wait times. RAM and NIC displayed varying energy use, suggesting their tasks are less affected by CPU and SD efficiencies.

For metadata processing, as Figure 3 describes at the left, the increase from 1 to 2 cores leads to an 8.1% rise in processing time, likely due to increased contention for CPU resources. With 3 cores, performance drops further, indicating inefficiencies in handling cache-dependent tasks and more idle states in

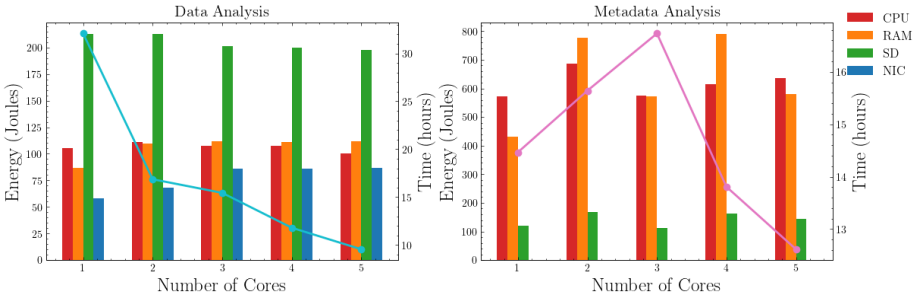

Fig. 3. Data and Metadata Processing Efficiency Metrics

the CPU. However, at 5 cores, there's a noticeable improvement, with processing time dropping by 12.8% from one core. This improvement could be due to better load balancing and reduced contention. With more cores, the system can distribute tasks more effectively, lessening the competition for CPU resources. This change implies that at 5 cores, the system overcomes initial bottlenecks and manages CPU resources more efficiently.

Regarding energy, the pattern is not linear. While the overall CPU energy use from 1 to 5 cores increases by 11.2%, this increase is not uniform across all cores. Particularly at 3 cores, where we see the highest processing time, the CPU energy consumption does not peak correspondingly, due to inefficient utilization or idle states.

Key Insight: The processing time of data or metadata in a Data Lake may not be faster or more energy efficient if the number of CPU cores increases. It depends on the operations involved and the operating system optimization. The reasons are related to resource contention during intensive processing on a single resource, execution of dependent tasks, limitations of parallel execution, management of CPU internal caches, and handling of idle resource states.

Scale factor and resource energy usage Figure 4 describes at the left processing data sets of 4.2, 12, and 18.7 GB for SF 1, SF 3, and SF 5 in DLBench+. CPU usage increases by about 200% from SF 1 to SF 5, due to the increased parallel processing in text conversion and indexing (as in operations OP1 - OP3). For the same reason, the SD energy consumption also rises significantly, about 470% at SF 5, reflecting more I/O operations during data retrieval and storage in larger data sets, along with frequent switching between active and idle states. NIC's consumption increases by 150% but shows a linear increase, corresponding to the demand for transferring large amounts of data.

Meanwhile, RAM exhibits more uniform energy increases, about 60% from SF 1 to SF 5. This increase is linked to the need for more memory for caching, leading to increased read and write operations, particularly in processes involving document scanning and conversion.

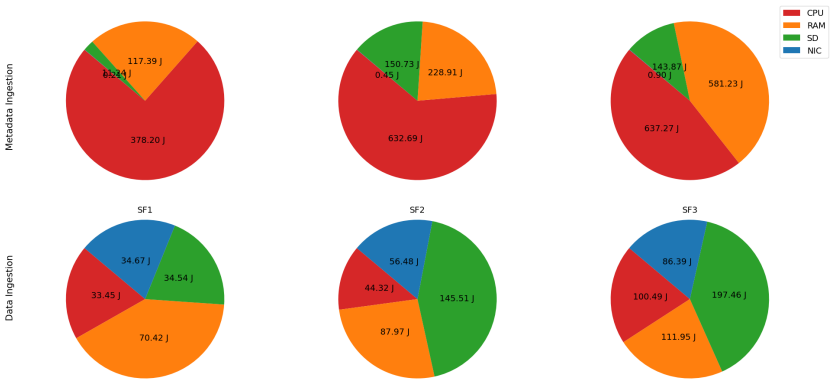

Fig. 4. Data Processing energy consumption by scale factor

These disproportionate increases in CPU and SD energy consumption alter their energy proportions among the components, with the CPU's share rising to about 20% and the SD to nearly 40% at SF 5, highlighting the shifting balance towards processing and storage demands as data scale and complexity escalate.

Concerning metadata processing energy consumption depicted in Figure 4 at the right, at SF 1, CPU accounts for 74.59% of energy usage because of complex operations such as text lemmatization or node categorization (OP5 - OP8). However, as the scale factor rises to SF 3 and SF 5, corresponding to larger data volumes, CPU's energy proportion decreases to 62.47% and then to 46.75%. That is because the RAM energy consumption becomes more prominent, jumping from 23.15% at SF 1 to 42.64% at SF 5. This increase is likely due to the heightened demand for memory access and caching, particularly for operations involving the creation and management of Neo4j nodes, which become more memory-intensive with larger datasets.

Regarding the SD, its energy proportion also rises to 10.55% at SF 5, indicating increased storage activity, potentially for document indexing and managing, or due to constant idle states. However, the SD and NIC do not represent a significant portion of the total energy consumption.

Moreover, it's interesting to note that the CPU and RAM evolve differently in terms of energy consumption, likely due to the non-linear increment tendency of cache misses when dealing with larger data sets.

Key Insights: Energy consumption in a Data Lake varies by operation type and hardware, not just data volume. Moreover, larger datasets in data processing notably increase CPU and SD energy, affected by parallel processing and I/O operations, with rises in RAM and NIC use. Finally, for metadata processing, larger datasets significantly boost CPU and RAM energy due to more RAM accesses and CPU-intensive tasks.

Energy usage by operation Figure 5 delineates the energy dynamics in both data and metadata operations. The three most energy-intensive operations are OP8, OP6, and OP1. OP8, in metadata processing, stands out as the highest energy consumer, utilizing a total of 1027.9729 Joules. This energy demand is primarily due to CPU-intensive Neo4J operations, which involve complex calculations for nodes and tables. OP6 follows closely, with a notable 498.5907 Joules consumption in RAM, attributed to memory-intensive tasks like lemmatization and tokenization in metadata analysis.

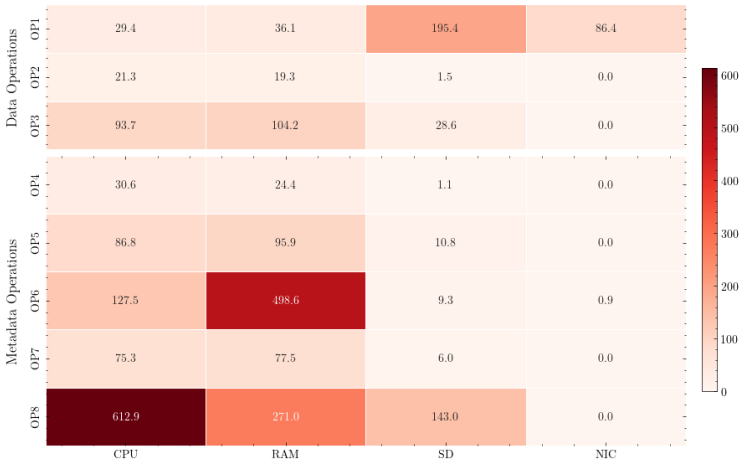

Fig. 5. Heatmap: Data and Metadata operations energy costs

OP1 is the third most energy-expensive, consuming 195.3762 Joules predominantly in the SD component, largely due to I/O operations for data acquisition and data initial processing. These operations collectively account for 75% of the total energy expenditure, highlighting the procedures and hardware components that are key targets for potential energy-aware strategies in Data Lakes.

6 Conclusions and future work

This paper presents an energy measurement system for Data Lake instances, focusing on the analysis of four key hardware components: CPU, RAM, NIC, and storage devices. In our study, which extends beyond the usual focus on CPU consumption, we highlight the importance of considering all hardware elements to fully understand energy consumption in Big Data Analysis (BDA). Addressing the paradox of seeking frugality in BDA, we recognize the need for a clear view of energy costs in such architectures. Our evaluation of DLBench+ provides this perspective, paving the way for future enhancements in Data Lake deployments. This includes potential load-balancing strategies to improve energy efficiency, an area of ongoing research [16].

References

1. Azeroual, O., Schöpfel, J., Ivanovic, D., Nikiforova, A.: Combining Data Lake and Data Wrangling for Ensuring Data Quality in CRIS (May 2022), https://hal.archives-ouvertes.fr/hal-03694519
2. Castro, R., Abedjan, Z., Koko, F., Yuan, G., Madden, S., Stonebraker, M.: Aurum: A data discovery system. In: 2018 IEEE 34th International Conference on Data Engineering (ICDE) (2018). https://doi.org/10.1109/ICDE.2018.00094
3. Conejero, J., Rana, O., Burnap, P., Morgan, J., Caminero, B., Carrión, C.: Analyzing hadoop power consumption and impact on application qos. Future Generation Computer Systems (2016)
4. Feller, E., Ramakrishnan, L., Morin, C.: Performance and energy efficiency of big data applications in cloud environments: A hadoop case study. Journal of Parallel and Distributed Computing **79** (2015)
5. Guo, B., Yu, J., Yang, D., Leng, H., Liao, B.: Energy-efficient database systems: A systematic survey. ACM Comput. Surv. **55**(6) (dec 2022). https://doi.org/10.1145/3538225, https://doi.org/10.1145/3538225
6. Helal, A.: Data lakes empowered by knowledge graph technologies. SIGMOD '21 (2021). https://doi.org/10.1145/3448016.3450584
7. Humberto, A.V.H.: An energy saving perspective for distributed environments: Deployment, scheduling and simulation with multidimensional entities for Software and Hardware. Ph.D. thesis, UPPA, https://www.theses.fr/s342134 (2022)
8. Intel: Enhanced intel speedstep technology for the intel pentium m processor (2004)
9. Li, H., Wang, H., Fang, S., et al.: An energy-aware scheduling algorithm for big data applications in spark. Cluster Computing . https://doi.org/10.1007/s10586-019-02947-9
10. Noureddine, A., Rouvoy, R., Seinturier, L.: Monitoring energy hotspots in software. Automated Software Engg. **22**(3), 291–332 (Sept 2015). https://doi.org/10.1007/s10515-014-0171-1
11. Polato, I., Barbosa, D., Hindle, A., Kon, F.: Hadoop energy consumption reduction with hybrid hdfs. SAC '16 (2016). https://doi.org/10.1145/2851613.2851623
12. Ravat, F., Zhao, Y.: In: New Trends in Databases and Information Systems. pp. 37–44. Springer International Publishing, Cham (2019)
13. Ravat, F., Zhao, Y.: Metadata management for data lakes. In: New Trends in Databases and Information Systems. Cham (2019)
14. Sawadogo, P.N., Darmont, J.: Dlbench+: A benchmark for quantitative and qualitative data lake assessment. Data & Knowledge Engineering (2023)
15. Sawadogo, P., Darmont, J., Noûs, C.: Joint management and analysis of textual documents and tabular data within the audal data lake (2021)
16. Valera, H.H.A., Dalmau, M., Roose, P., Larracoechea, J., Herzog, C.: An energy saving approach: Understanding microservices as multidimensional entities in p2p networks. SAC '21 (2021). https://doi.org/10.1145/3412841.3441888
17. Wu, W., Lin, W., Hsu, C.H., He, L.: Energy-efficient hadoop for big data analytics and computing: A systematic review and research insights. Future Generation Computer Systems (2018)
18. You, X., Lv, X., Zhao, Z., Han, J., Ren, X.: A survey and taxonomy on energy-aware data management strategies in cloud environment. vol. 8, pp. 94279–94293 (2020). https://doi.org/10.1109/ACCESS.2020.2992748
19. Zhao, Y., Megdiche, I., Ravat, F.: Data lake ingestion management (2021). https://doi.org/10.48550/ARXIV.2107.02885, https://arxiv.org/abs/2107.02885