CSIT Conference 2023, Yerevan, Armenia, September 25 - 30

Energy Efficiency Evaluation of Local and Offloaded Data Processing

Victor Prokhorenko, Muhammad Ali Babar
Centre for Research on Engineering Software Technologies (CREST) Lab.
The University of Adelaide
Adelaide, Australia
{victor.prokhorenko},{ali.babar}@adelaide.edu.au

Abstract—**This work evaluates four database solutions in terms of energy efficiency. This study measures energy efficiency of Cassandra, MongoDB, Redis, and MySQL data storage solutions on a selected set of physical and virtual nodes by leveraging Intel RAPL (Running Average Power Limit) technology. Extensive experimental results show that (i) Redis and MongoDB are more efficient in energy consumption under most usage scenarios, (ii) remote offloading saves energy if the network latency is low and destination CPU significantly more powerful, and (iii) computationally weaker CPUs may sometimes demonstrate higher energy efficiency in terms of J/op.**

Keywords— **Energy Measurement, NoSQL Databases, SQL Database, Cloud Computing, Edge Computing, Performance.**

I. INTRODUCTION

Energy efficiency is becoming increasingly important with the adoption of resource-constrained mobile and wearable devices [1, 2]. Furthermore, the popularity of large-scale and distributed data centres hosting thousands of servers raises additional energy-related challenges [3, 4].

In this study, we focus on database-level workload offloading energy efficiency. We selected modern and popular databases including Cassandra[1], MongoDB[2], Redis[3] and MySQL[4]. We considered different situations in regard to database server and client locations. We explore the impact of both CPU power and distance between client and database server nodes on the energy consumption. We further refer to our laptop and server nodes as *mid-range* and *high-end* CPUs respectively.

We focused on evaluating energy efficiency, taking into consideration the CPU power and distance between client and server through the following research questions (RQ).

- RQ1: Which database solution is more energy efficient under typical local usage?
- RQ2: How is the database energy efficiency affected by multiple threads?
- RQ3: In terms of energy efficiency, what is the role of CPU computational power in typical database workloads?
- RQ4: How beneficial is database workload offloading in terms of energy efficiency?
- RQ5: How does the network latency affect offloading energy efficiency?

To answer these questions, a set of experiments has been conducted in various conditions and environments. The main contributions of this study include insights on energy efficiency of the selected database under different scenarios as well as the goal-oriented database workload offloading energy efficiency evaluation and comparison framework.

II. RELATED WORK

Energy measurement approaches can be categorized into *hardware-based monitoring* and *software-based estimation*[5]. While costing more, smart hardware monitoring systems enable data offloading to remote hosts for further analysis [7, 8]. A detailed taxonomy of energy measurement approaches for battery-powered devices at various levels of granularity is presented in [9].

Software-based energy estimation models are typically based on various performance counters and I/O load observable within the system [10–12]. Energy estimation models have been shown to be less precise compared to hardware-based measurement. The variance can achieve 73% to 300% [13]. Various metrics useful for practical software energy efficiency evaluation are discussed in [14, 15].

In contrast to NoSQL solutions energy efficiency evaluation [16], our study also considers a traditional SQL database and more offloading scenarios (close-proximity and long-distance).

III. EXPERIMENT SETUP

Two physical nodes and one virtual node hosted on OpenStack cloud were used throughout the experiments (I). Ubuntu 18.04.4 server edition was used on all hosts. The energy consumption was only considered from end-user perspective and measured on physical client nodes only. We conducted a set of experiments to determine energy efficiency of MongoDB, Cassandra, MySQL and Redis due to their wide adoption. This selection allows to compare a variety of different types of data management systems.

---

[1] Cassandra:https://cassandra.apache.org/
[2] MongoDB:https://www.mongodb.com/
[3] Redis: https://redis.io/
[4] MySQL: https://www.mysql.com/
[5] Hybrid energy usage frameworks are an alternative solution to energy usage monitoring [5]. Specific database-oriented energy usage measurement using hybrid framework has been explored in [6].

https://doi.org/10.51408/csit2023_12

TABLE I
HARDWARE INFRASTRUCTURE

Host Name	Type	CPU	RAM
Mid-range	Phys.	Intel Core i5-5200U	16GB
High-end	Phys.	Intel Core i9-9900KF	32GB
OpenStack	Virt.	Intel Xeon E5-2623v3	8GB

TABLE II
EXPERIMENTAL SCENARIOS

Scenario	Source	Destination	Distance
1	High-end	High-end	0
2	Mid-range	Mid-range	0
3	Mid-range	High-end	Short (1.5ms)
4	High-end	OpenStack	Short (1.5ms)
5	Mid-range	High-end	Long (50ms)

We used Yahoo! Cloud Servicing Benchmark (YCSB) to run core database workloads in the recommended order[6]. Intel RAPL technology was used to measure CPU energy usage throughout the experiments. Overall CPU micro-Joules measurements provided by Linux kernel were used to determine the amount of energy consumed by each workload. We ran six scenarios as described in Table II.

IV. WORKLOAD ENERGY EFFICIENCY EVALUATION

This section focuses on database solution energy efficiency under typical local usage (RQ1), and on how the database energy efficiency is affected by multiple threads (RQ2). The first experiment set was conducted on a single host with no network communication and encryption overhead (scenarios 1-2).

A. High-end CPU

Workloads B, C and D are quite similar in terms of energy efficiency. At around 10 threads, the energy consumption stabilizes for all databases and does not improve any further. Consistent to previously conducted performance benchmarking [17], workload E proves to be most challenging for Redis with the other databases not being affected as much. Moreover, the energy efficiency of Redis does not improve much as number of threads increases, with no apparent changes after 2 threads. This is in contrast to other workloads, where Redis is twice as more energy-efficient. Workload E is the most demanding in terms of absolute values of energy usage as querying ranges of records requires more data to be transmitted per operation. Larger transmissions naturally take longer to complete, thus requiring more energy.

B. Mid-range CPU

Surprisingly, despite achieving lower throughput, the slower CPU is more energy-efficient in terms of absolute values of micro-Joules consumed per 1000 operations. This remains true for all types of workloads. Somewhat lower energy usage improvement factor for increasing number of threads can be explained by the lower number of CPU cores available. Overall, the workload energy usage efficiency is consistent across the different CPUs in local usage scenarios.

C. Workload type effects

Cassandra workload A uses 24% of energy used for workload E. Energy consumption difference can reach a factor of 35 for Redis across all databases and workloads. The difference ranges from a factor of 3.16 for MongoDB to 5.07 for Cassandra. Variance between workloads is not the same across different CPUs and does not generally reduce when a more powerful CPU is used. Workload energy consumption variance is 3.16 for MongoDB on a mid-range CPU, whereas on a high-end CPU this variance increases to 3.69. Disregarding workload E, we see that with the exception of MySQL, workload F is least energy efficient for all databases across all configurations tested. Workload A is the second hardest for local usage scenarios in terms of energy efficiency. For offloading scenarios, however, pattern changes for MongoDB and MySQL. Averaging the results across databases and usage scenarios, workloads can be sorted in terms of energy consumption as follows: E, F, A, D, B, C.

In summary, the answer to RQ1 is: Redis is more energy-efficient for most workloads (except E), for which MongoDB is the most efficient by a slight margin. The answer to RQ2 is: Increasing number of threads improves overall energy efficiency on average by a factor of 2, with no visible differences after number of threads reaches 10.

V. CPU-RELATED ENERGY EFFICIENCY EVALUATION

This section focuses on the CPU computational power from energy efficiency perspective in typical database workloads (RQ3). In our case, slower CPU achieved higher energy efficiency. While the more powerful CPU completes the tasks faster, the slower CPU consumes less energy per operation.

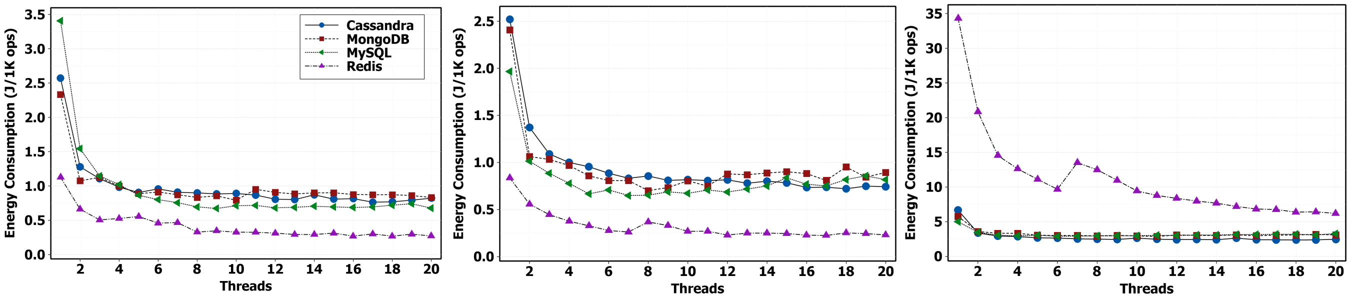
First, we see that most of the databases do not fully utilize CPU maximum power. On a high-end CPU, only MongoDB was able to achieve close to 100% power. Redis was also consuming a significant portion of maximum power. In contrast, MySQL and Cassandra could only consume around 15-25% and 30-60% on high-end and mid-range CPUs, respectively. These sub-optimal results indicate that there is still space in optimizing MySQL and Cassandra energy efficiencies.

Second, the key insight emerges from the comparison of the absolute values of idle to fully loaded energy usage of the CPUs tested. Mid-range CPU consumes 14.5W, while high-end CPU consumes 119W at full load. Therefore, the high-end CPU needs to perform 8.2 times more ops/sec to achieve the same energy efficiency (J/ops) as the mid-range CPU. Under real-world conditions, this factor is reduced to around a factor of 4 for MySQL/Cassandra and a factor of 7 for Redis.

In summary, the answer to RQ3 depends on three key factors: the maximum power that the given CPU can consume, how much of that energy can the given database utilize and the raw computational CPU power. The mid-range CPU proved to be more energy-efficient one under all workloads for all databases. However, choosing a more energy-efficient CPU could be beneficial only if lower throughput is acceptable.

---

[6] https://github.com/brianfrankcooper/YCSB/wiki/Core-Workloads



(a) Workload A (b) Workload B (c) Workload E

Fig. 1. Energy consumption of all databases, where database servers run on a high-end CPU and YCSB client runs on a mid-range CPU (close proximity).

VI. NETWORK OFFLOADING ENERGY EFFICIENCY

This section discusses answers to how beneficial is database workload offloading in terms of energy efficiency (RQ4), and how does the network latency affect offloading energy efficiency (RQ5). Firstly, a close-proximity offloading was performed within University of Adelaide LAN. Then, a longer-distance offloading was conducted over WAN.

A. Close-proximity offloading

In this set of experiments, the database servers resided on the high-end CPU, while YCSB was running on the mid-range CPU (Scenario 3). As shown in Figure 1, the relative database energy efficiency is similar to local workloads. Redis struggles in workload E, while using less energy for other workloads.

Offloading workloads to high-end CPU generally improves energy efficiency. However, for the effect to become visible and substantial for MongoDB, the number of threads must be increased. Offloading did not gain any significant improvement for Redis in terms of energy efficiency. Even at 20 threads, the energy consumption difference remains negligible. For Cassandra, MongoDB and MySQL, the energy efficiency improvement factor reaches 1.5, 1.9 and 2.7 respectively. Note that such significant energy usage improvement can only be achieved with multi-threaded loads as single-threaded workloads do not benefit as much from offloading.

Scenario 4 (high-end CPU to a somewhat slower Openstack VM offloading) reveals a drastic change compared to the mid-range to high-end offloading. Only MongoDB managed to achieve a slight improvement, reaching a factor of 1.3 at 20 threads. MySQL suffered most, consuming around 4 times more energy. Cassandra was impacted slightly by the offloading with the energy consumption growing by around 10%. Redis consumed more than 50% more energy. The situation is worse for MongoDB and Cassandra when a low number of threads is used. For up to 6-7 threads, close-proximity offloading to a slower CPU worsens energy efficiency.

These results are explained by the extra time that the client idles while waiting for server results. The associated network traffic and bandwidth were measured and observed to be significantly lower than the physical link capacity. Thus, it is highly unlikely that network link limitations caused a bottleneck.

B. Long-distance offloading

In this set of experiments, YCSB workload runs on the mid-range CPU and the high-end CPU hosts databases (Scenario 5). Compared to close-proximity offloading, the packet latency is significantly higher in this scenario (50ms vs. 1.5ms).

As shown in Figure 2, measuring energy efficiency for long-distance workload offloading revealed that with an exception of workload E, the difference between different databases virtually disappears. Workload E is a known weak point for Redis. Other databases demonstrate close results with deviations rarely exceeding 10%. As the average operation latency in local scenarios is typically around tens of microseconds, clearly 50ms network latency dominates the database latency.

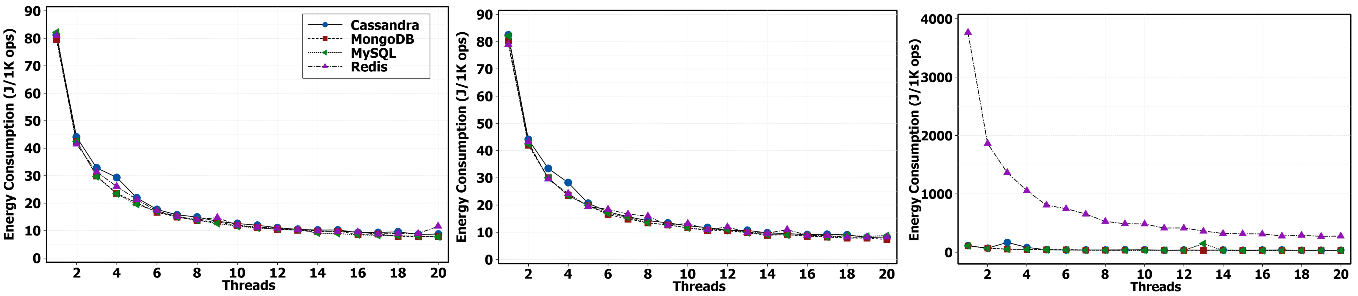
Figure 3 illustrates the absolute values of energy usage for three scenarios: local processing on a mid-range CPU as well as close-proximity (LAN) and long distance (WAN) offloading from mid-range CPU to high-end CPU. We see that long-distance offloading worsens energy efficiency even if the remote CPU is significantly more powerful. Energy usage increases by an order of magnitude, making such offloading pointless. Results for MongoDB, MySQL and Redis are omitted for brevity due to similar result patterns.

Based on the measurements conducted, the answers to RQ4 and RQ5 can be formulated as follows. RQ4: workload offloading can be beneficial (except Redis) only if the remote CPU is significantly more powerful. RQ5: only low-latency network environments can be benefitial as high network latency starts dominating the database latencies.

VII. CONCLUSION AND FUTURE WORK

It was shown that newer CPUs, while being faster, may not necessarily be more energy efficient in all workloads. We also see that most of the tested databases were unable to utilize the amount of energy consumed under stress testing. As energy efficiency can be improved by a factor of 3 to 4, the area of software optimisations is worth investigating.

Energy efficiency improvements were only achieved under two conditions: (i) The remote CPU is significantly more computationally powerful than the local CPU, and (ii) the network latency is low. When either condition was not fulfilled, the energy efficiency decreased dramatically. This is explained by the high overhead of two-way data transmission and the associated idle waits. Note that the performed measurements



(a) Workload A (b) Workload B (c) Workload E

Fig. 2. Energy consumption of databases, where YCSB runs on mid-range CPU and database servers run on high-end CPU (long distance).

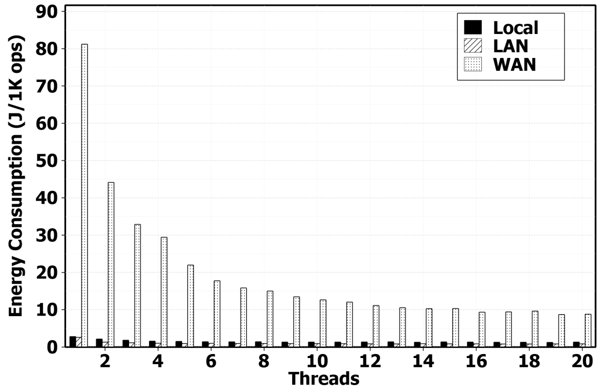


Fig. 3. A comparison of energy consumption of Cassandra (Workload A) running on a mid-range CPU between local and offloaded processing (close-proximity and long distance).

revealed that data transmission energy usage is negligible (1.2-1.5%) compared to the idling energy waste.

The most significant limitation of this study is that only CPU energy usage was considered. Taking into account RAM and disk may affect our findings. Future research lines can focus on a number of potentially useful extensions of this work. Firstly, testing more CPUs architectures and measuring hardware component energy usage separately. Secondly, tweaking memory management related settings such as swap file size, disk I/O caching and amount of RAM. Lastly, OS-specific effects could also impact energy efficiency of databases.

REFERENCES

[1] Gustavo Pinto and Fernando Castor. Energy efficiency: a new concern for application software developers. *Commun. ACM*, 60(12):68–75, 2017.

[2] Eric Saxe. Power-efficient software. *ACM Queue*, 8(1):10, 2010.

[3] Frederic T. Chong, Martijn J. R. Heck, P. Ranganathan, A. A. M. Saleh, and Hassan M. G. Wassel. Data center energy efficiency: Improving energy efficiency in data centers beyond technology scaling. *IEEE Des. Test*, 31(1):93–104, 2014.

[4] Francis Owusu and Colin Pattinson. The current state of understanding of the energy efficiency of cloud computing. In Geyong Min, Yulei Wu, Lei (Chris) Liu, Xiaolong Jin, Stephen A. Jarvis, and Ahmed Yassin Al-Dubai, editors, *11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2012, June 25-27, 2012*, pages 1948–1953. IEEE Computer Society, 2012.

[5] Vincent M Weaver, Matt Johnson, Kiran Kasichayanula, James Ralph, Piotr Luszczek, Dan Terpstra, and Shirley Moore. Measuring energy and power with papi. In *2012 41st International Conference on Parallel Processing Workshops*, pages 262–268. IEEE, 2012.

[6] Yong Jin, Baoping Xing, and Peiquan Jin. Towards a benchmark platform for measuring the energy consumption of database systems. *Advanced Science and Technology Letters*, 29:385–389, 2013.

[7] Daniel Hackenberg, Thomas Ilsche, Robert Schöne, Daniel Molka, Maik Schmidt, and Wolfgang E Nagel. Power measurement techniques on standard compute nodes: A quantitative comparison. In *2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 194–204. IEEE, 2013.

[8] Volker Hudlet and Daniel Schall. Measuring energy consumption of a database cluster. In Theo Härder, Wolfgang Lehner, Bernhard Mitschang, Harald Schöning, and Holger Schwarz, editors, *Datenbanksysteme für Business, Technologie und Web (BTW), 14. Fachtagung des GI-Fachbereichs "Datenbanken und Informationssysteme" (DBIS), 2.-4.3.2011*, volume P-180 of *LNI*, pages 734–737. GI, 2011.

[9] Taher Ahmed Ghaleb. Software energy measurement at different levels of granularity. pages 1–6. IEEE, 2019.

[10] Manuel Rodríguez-Martínez, Harold Valdivia, Jaime Seguel, and Melvin Greer. Estimating power/energy consumption in database servers. In Cihan H. Dagli, editor, *Proceedings of the Complex Adaptive Systems 2011 Conference, Chicago, Illinois, USA, October 31-November 2, 2011*, volume 6 of *Procedia Computer Science*, pages 112–117. Elsevier, 2011.

[11] Richard E. Kavanagh and Karim Djemame. Rapid and accurate energy models through calibration with IPMI and RAPL. *Concurr. Comput. Pract. Exp.*, 31(13), 2019.

[12] Thomas Rauber, Gudula Rünger, and Michael Schwind. Energy measurement and prediction for multi-threaded programs. In *2014 Spring Simulation Multiconference, SpringSim '14, Proceedings of the High Performance Computing Symposium*, page 20. ACM, 2014.

[13] Muhammad Fahad, Arsalan Shahid, Ravi Reddy Manumachu, and Alexey Lastovetsky. A comparative study of methods for measurement of energy of computing. *Energies*, 12(11):2204, 2019.

[14] Timo Johann, Markus Dick, Stefan Naumann, and Eva Kern. How to measure energy-efficiency of software: Metrics and measurement results. In Rick Kazman, Patricia Lago, Niklaus Meyer, Maurizio Morisio, Hausi A. Müller, Frances Paulisch, Giuseppe Scanniello, and Olaf Zimmermann, editors, *First International Workshop on Green and Sustainable Software, GREENS 2012, June 3, 2012*, pages 51–54. IEEE Computer Society, 2012.

[15] Christian Stier, Anne Koziolek, Henning Groenda, and Ralf H. Reussner. Model-based energy efficiency analysis of software architectures. In Danny Weyns, Raffaela Mirandola, and Ivica Crnkovic, editors, *Software Architecture - 9th European Conference, ECSA 2015, Dubrovnik/Cavtat, Croatia, September 7-11, 2015, Proceedings*, volume 9278 of *Lecture Notes in Computer Science*, pages 221–238. Springer, 2015.

[16] Carlos Gomes, Eduardo Tavares, and Meuse Nogueira de O Junior. Energy consumption evaluation of nosql dbmss. In *Anais do XV Workshop em Desempenho de Sistemas Computacionais e de Comunicação*, pages 71–81. SBC, 2020.

[17] Yaser Mansouri, Victor Prokhorenko, and M. Ali Babar. An automated implementation of hybrid cloud for performance evaluation of distributed databases. *Journal of Network and Computer Applications*, 167:102740, 2020.