INFORMATICA, 2024, Vol. 35, No. 3, 649–669

DOI: https://doi.org/10.15388/24-INFOR567

Offloaded Data Processing Energy Efficiency Evaluation

Victor PROKHORENKO*, Muhammad Ali BABAR

The University of Adelaide, Australia
e-mail: victor.prokhorenko@adelaide.edu.au, ali.babar@adelaide.edu.au

Received: September 2023; accepted: July 2024

Abstract. The growing popularity of mobile and cloud computing raises new challenges related to energy efficiency. This work evaluates four various SQL and NoSQL database solutions in terms of energy efficiency. Namely, Cassandra, MongoDB, Redis, and MySQL are taken into consideration. This study measures energy efficiency of the chosen data storage solutions on a selected set of physical and virtual computing nodes by leveraging Intel RAPL (Running Average Power Limit) technology. Various database usage scenarios are considered in this evaluation including both local usage and remote offloading. Different workloads are benchmarked through the use of YCSB (Yahoo! Cloud Serving Benchmark) tool. 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 is significantly more powerful, and (iii) computationally weaker CPUs may sometimes demonstrate higher energy efficiency in terms of J/ops. An energy efficiency measurement framework is proposed in order to evaluate and compare different database solutions based on the obtained experimental results.

Key words: energy measurement, NoSQL databases, SQL database, cloud computing, edge computing, performance.

1. Introduction

Energy efficiency is becoming increasingly important with the adoption of resource-constrained mobile and wearable devices (Pinto and Castor, 2017; Saxe, 2010). Furthermore, popularity of large-scale and distributed data centres hosting thousands of servers raises additional energy-related challenges (Chong *et al.*, 2014; Owusu and Pattinson, 2012). Energy usage optimization techniques range from individual device sleeping state management to large-scale energy usage consolidation. Two main energy optimization goals can be highlighted.

Firstly, minimizing energy usage in order to prolong system operation is the main goal for wearable and mobile systems. Such systems typically aim to operate for prolonged time periods. Typical examples include mobile phones and smart watches. Given the miniature size, such systems suffer from low energy and computing power capacity. Despite the significant progress in CPU development and manufacturing, user needs commonly exceed

*Corresponding author.

the capacity of current hardware. Thus, apart from improving the underlying hardware, two techniques commonly used in conjunction are employed in order to improve overall user experience. These include idle state management and computation offloading. Both measures can significantly increase system operational time (Mansouri and Babar, 2021; Jiang *et al.*, 2020).

Secondly, consolidating energy usage for multiple devices is applicable to data-centres. Perhaps the most popular approach is to virtualize multiple nodes and co-host them on the same hardware. Also, migrating virtual machines between hardware servers based on current workloads can be beneficial under certain conditions. The main goal of such dynamic VM migration is to pack the running tasks into a lower number of hardware servers. Consequently, this allows to power down (or transition to a low energy consumption state) the currently unused hardware. Two main obstacles in this area are the unpredictability of workload and large overheads associated with workload migration.

In this study, we focus on database-level workload offloading energy efficiency. We selected modern and popular databases including Cassandra,[1] MongoDB,[2] and Redis[3] as NoSQL databases and MySQL[4] as a relational database. We considered two different situations in regard to database server and client locations, namely same node or different nodes. We explore the impact of both CPU power and distance between client and database server nodes on the energy consumption. We made a setup of two physical nodes including a laptop and a powerful server and one virtual computing node managed through OpenStack virtualization platform. We further refer to the laptop and the powerful sever as *mid-range* and *high-end* CPU respectively, indicating that the server hardware we use is significantly more powerful than the laptop.

With the advent of NoSQL databases in last decade, a myriad of studies explored the performance of these databases, mainly including Cassandra, MongoDB, CouchDB,[5] Riak[6], Redis and compared their performance with the well-known relational databases like MySQL, PostgreSQL,[7] and SQL server. The evaluation of these databases was conducted on a variety of hardware infrastructures. Rabl *et al.* (2012) measured the performance of NoSQL and relational databases on a private cloud. Kuhlenkamp *et al.* (2014) performed experiments on the public cloud to evaluate how scalable and elastic NoSQL databases are. We have also recently measured the performance of NoSQL and relation databases on the hybrid cloud (Mansouri *et al.*, 2020). The closest work to ours is energy measurement of three NoSQL databases (Cassandra, MongoDB, and Redis) running on a hardware server (Phung *et al.*, 2019).

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

[1] Cassandra:https://cassandra.apache.org/
[2] MongoDB:https://www.mongodb.com/
[3] Redis: https://redis.io/
[4] MySQL: https://www.mysql.com/
[5] CouchDB: https://couchdb.apache.org/
[6] Riak: https://riak.com/
[7] PostgreSQL: https://www.postgresql.org/

- 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.

The rest of this paper is organised as follows. A review of related literature is presented in Section 2. Section 3 provides a description of our infrastructure and experimental setup. Corresponding experiment results are outlined in Sections 4–6. Section 7 discusses the proposed goal-oriented energy efficiency evaluation framework. Result summary and research limitations are discussed in Section 8. Lastly, Section 9 concludes this study and outlines future research directions.

2. Related Work

Energy measurement approaches can be categorized into *hardware-based monitoring* and *software-based estimation*.[8] Hardware-based approaches require energy monitors which can be expensive depending on the precision provided. Such monitoring devices range from simple analogue Watt-meters to complex internet-connected systems. While costing more, smart monitoring systems enable data offloading to remote hosts for further analysis (Hackenberg *et al.*, 2013; Hudlet and Schall, 2011). A detailed taxonomy of energy measurement approaches for battery-powered devices at various levels of granularity is presented in Ghaleb (2019).

Software-based energy estimation approaches, though less precise, enable estimating the energy usage based on a pre-built power model for a given hardware. Such models are typically based on various performance counters and I/O load observable within the system (Rodríguez-Martínez *et al.*, 2011; Kavanagh and Djemame, 2019; Rauber *et al.*, 2014). Common examples include CPU and RAM frequency. Energy estimation models have been shown to be less precise compared to hardware-based measurement. The variance can achieve 73% to 300% (Fahad *et al.*, 2019). Various metrics useful for practical software energy efficiency evaluation are discussed in Kalaitzoglou *et al.* (2014), Johann *et al.* (2012), Stier *et al.* (2015).

Intel developed a precise power usage model for modern CPUs (Khan, 2018, 2018). This model, known as RAPL (Running Average Power Limit), was developed to design and optimise cooling subsystem based on the observed power consumed and dissipated

[8]Hybrid energy usage frameworks are an alternative solution to energy usage monitoring (Weaver *et al.*, 2012). Specific database-oriented energy usage measurement using hybrid framework has been explored in Jin *et al.* (2013).

by a CPU (Zhang and Hoffman, 2015). Comparing the observed energy usage to RAPL measurements indicates that the overall results seem to match closely (Desrochers *et al.*, 2016). Note that RAPL readouts are less accurate when a system is idle or when a discrete GPU is actively used. The overhead of RAPL measurement itself was measured in Calandrini *et al.* (2013). In addition, direct RAPL benchmarking tools integration have been proposed (Beyer and Wendler, 2020).

Due to the high popularity of virtualization, significant efforts have been put in VM energy usage estimation (Xu *et al.*, 2015). Virtualization complicates the power usage monitoring as the hypervisor may not expose the performance counters to the guest OS. Thus, various estimation approaches have been proposed, including for nested virtualization scenarios (Colmant *et al.*, 2015). It has been shown that Docker containerization is not more energy efficient compared to traditional VM hypervisors (Jiang *et al.*, 2019).

More fine-grained techniques focus at the process-level energy usage estimation (Bourdon *et al.*, 2013). Some prior research on energy efficiency of databases indicates that CPU power consumption may differ up to 60% depending on type of the operations (Tsirogiannis *et al.*, 2010). This research was, however, limited to PostgreSQL and System-X databases only. More recent study, focused on NoSQL databases (HBase,[9] Cassandra, HadoopDB (Abouzeid *et al.*, 2009) and Hive[10]), indicates that a significant portion of energy is wasted during waiting periods (Li *et al.*, 2014). Zhou *et al.* (2020) measured the impact of number of CPU cores in the context of PostgreSQL energy efficiency. Database query-level energy efficiency and optimization is considered in Xu *et al.* (2012). Benchmarks in Schall *et al.* (2010), Cheong *et al.* (2012) indicate that using SSDs instead of mechanical platter HDDs improves database energy efficiency significantly. Furthermore, it has been demonstrated that the choice of the operating system as well as programming language may also affect database energy efficiency (Capra *et al.*, 2012; Pereira *et al.*, 2017).

In addition to measuring efforts, various database energy efficiency optimization techniques are proposed (Graefe, 2008). These include memory hierarchies, adaptive query rescheduling, I/O optimizations and data format improvements. Relying on hardware capabilities shows an improvement of database energy efficiency by up to 40% (Pisharath *et al.*, 2004). Hardware-assisted energy-efficient database accelerator device has been presented in Haas *et al.* (2016). It was also shown that adjusting power usage limits on-the-fly within distributed systems may improve overall energy efficiency for a given workload (Phung *et al.*, 2019). A similar approach proposes to disconnect and reconnect database nodes based on current processing needs to improve energy efficiency (Härder *et al.*, 2011). Energy usage optimisation measures in large scale environments typically focus on optimisations such as power-aware resource management, task scheduling and traffic tuning (Jin *et al.*, 2017).

In contrast to NoSQL solutions energy efficiency evaluation (Gomes *et al.*, 2020), our study also considers a traditional SQL database. We also evaluate more usage scenarios, including close-proximity and long-distance database workload offloading. This

[9] HBase: https://hbase.apache.org
[10] Hive: https://hive.apache.org/

study is an expansion of our prior work on data offloading energy efficiency evaluation (Prokhorenko and Babar, 2023).

3. Experiment Setup

This section outlines the details of the conducted energy efficiency measurement experiments. Firstly, our physical and virtual infrastructure is discussed in Section 3.1. Typical YCSB database workloads used throughout the experiments are described in Section 3.2. Energy usage measurement approach taken is presented in Section 3.3. Lastly, Section 3.4 outlines the conducted experiment details.

3.1. *Hardware Infrastructure*

Two physical nodes and one virtual node hosted on OpenStack cloud were used throughout the experiments. Ubuntu 18.04.4 server edition was used on all hosts. Table 1 summarises hosts details. Due to technical challenges associated with precise VM energy usage measurement, the virtual node was only used in workload offloading scenarios. In other words, the energy consumption was only considered from end-user perspective and measured on physical client nodes only. As clouds provide "infinite" resources (including energy supply) this limitation seems acceptable.

3.2. *Database Benchmark and Workloads*

We conducted a number of experiments to determine energy efficiency of four data management solutions. We selected MongoDB, Cassandra, MySQL and Redis for this study due to their wide adoption in industry and academic sectors. This selection allows to compare a variety of different types of data management systems. The choice features two NoSQL, one traditional SQL and one RAM-based key-value database. Table 2 summarises database solutions benchmarked.

Table 1
Hardware infrastructure.

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

Table 2
Database solutions.

Database	Version	Type
Cassandra	3.11.6	Wide column
MongoDB	3.6.3	Document-oriented
MySQL	5.7.29	Relational
Redis	4.0.9	Key-value

Table 3
YCSB workload descriptions.

Workload	Workload type	Description
A	Update heavy	50% reads, 50% writes
B	Read mostly	95% reads, 5% writes
C	Read only	100% reads
D	Read latest	Reading newly inserted records
E	Short ranges	Querying short ranges of records
F	Read-modify-write	Reading, modifying and writing back records

We used Yahoo! Cloud Servicing Benchmark (YCSB) to run typical database workload as detailed in Table 3. As per YCSB recommendations, the following workload order was used: A, B, C, F, D are executed followed by purging and reloading data prior to running workload E. This order is recommended as workload D inserts new records, increasing the database size. The number of records and operations used throughout the local (scenarios 1–2) and close-proximity (1.5 ms latency) offloading scenarios was set to 100000 (scenarios 3–4). Long-distance (50 ms latency) offloading ran with the number of operations reduced to 10000 (scenarios 5–6).

3.3. *Energy Usage Measurement*

Intel RAPL technology was used to measure CPU energy usage throughout the experiments. In accordance with best practices, overall CPU micro-Joules measurements provided by Linux kernel was used to determine the amount of energy consumed by each workload. RAPL technology does not use a hardware meter but is rather a software power model. This power model uses various hardware performance counters and I/O models to approximate CPU energy usage. However, real-world energy usage measurements conducted by Intel indicate that the model closely matches actual power usage (Rotem *et al.*, 2012). Note that power and energy have different meaning. Specifically, power is measured in Watts and indicates how much energy is used at any given moment. Energy is measured in Watt-hours and refers to the ability to deliver power over time period.

The initial intended purpose of RAPL was to measure how much power does a CPU need on average during a specified time period. This is an essential knowledge required to develop cooling systems for size-constrained devices. Knowing energy usage can also aid choosing batteries suitable for a given device. RAPL technology is available starting from Sandy Bridge generation on desktops and servers.

3.4. *Database Usage Scenarios*

Based on our hardware and virtual infrastructure, we ran six scenarios as described in Table 4. We run YCSB workload and databases locally on Mid-range CPU (scenario 1) and high-end CPU (scenario 2). We then offload workload from mid-range CPU to High-end CPU and OpenStack VMs, which are located in close proximity (scenarios 3 and 4). Finally, we run long-distance offloading from Mid-range and high-end CPU to OpenStack VMs (scenarios 5 and 6).

Table 4
Experimental scenarios.

Scenarios	Source	Destination	Distance	Concept
Scenario 1	High-end CPU	High-end CPU	0	Local
Scenario 2	Mid-range CPU	Mid-range CPU	0	Local
Scenario 3	Mid-range CPU	High-end CPU	Close-proximity	offloading
Scenario 4	High-end CPU	OpenStack VM	Close-proximity	offloading
Scenario 5	Mid-range CPU	High-end CPU	Long	offloading

The results are classified in three groups: *local*, *close-proximity offloading* and *long-distance offloading*. Local refers to the scenario when both database server and client (YCSB) are located on the same host. Some databases (such as MySQL) may use local Unix Domain Sockets instead of network communication in such configurations by default. Close-proximity offloading was performed within Adelaide University LAN with the network latency of 1.5 ms between the nodes. Long-distance offloading was conducted over wider area network with the latency of 50 ms. Experiments were executed using different number of threads (1–20). Direct comparison of energy efficiency between different database solutions is not particularly meaningful due to the difference in architectures and data structures. Nevertheless, some high-level observations can be made based on the obtained results.

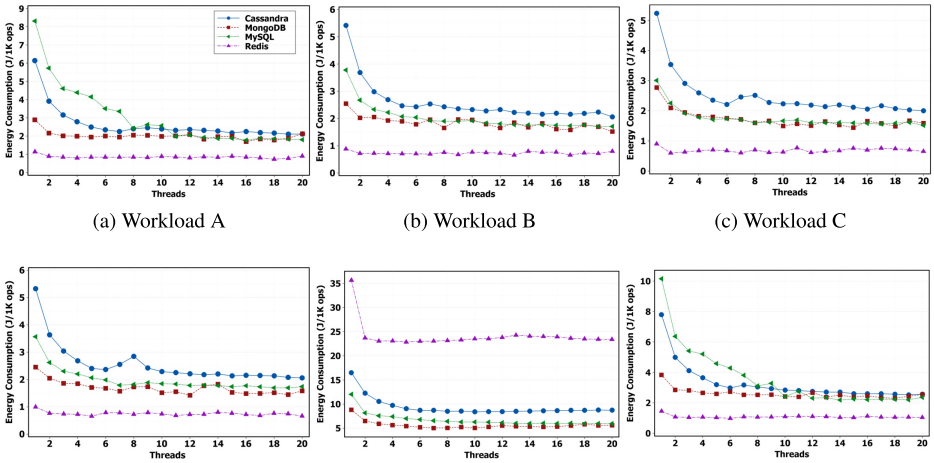
4. Workload Energy Efficiency Evaluation

This section focuses on database solution energy efficiency under typical local usage (RQ1), and on how is the database energy efficiency 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).

4.1. *High-End CPU*

Energy usage per 1000 operations for a high-end CPU is depicted on Fig. 1. As can be seen from the presented graphs, Workloads B, C and D are quite similar in terms of energy efficiency. Namely, the relative database energy efficiency remains the same across workloads. The largest difference in energy efficiency exists for one-threaded experiment runs. This difference steadily drops as number of threads increases. At around 10 threads, the energy consumption stabilizes for all databases and does not improve any further.

In contrast to workloads B, C and D, for workloads A and F, the relative performance of MySQL and Cassandra is swapped for low number of threads. However, at 8 threads this difference diminishes. Consistent to previously conducted performance benchmarking (Mansouri *et al.*, 2020), 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. While the improvement between 1 and 2 threads is significant, Redis is worse more than



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

(d) Workload D (e) Workload E (f) Workload F

Fig. 1. Energy consumption of local processing on high-end CPU for Cassandra, MongoDB, MySQL and Redis (scenario 2).

twice than other databases. This is in contrast to other workloads, where Redis is twice as more energy-efficient.

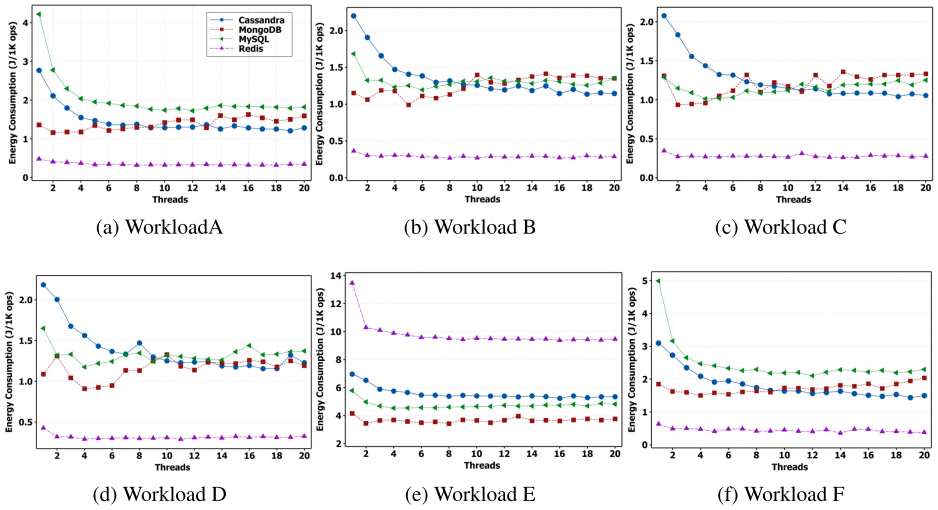
Workload E is most demanding in terms of absolute values of energy usage as querying ranges of records requires more data to be transmitted per each operation. Larger transmissions naturally take longer to complete, thus requiring more energy. Lastly, the overall average thread-related energy usage improvement is a factor of 2 for all databases, except Redis.

4.2. *Mid-Range CPU*

Figure 2 illustrates a surprising result for the mid-range CPU. Namely, while 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 with a roughly same relative database energy efficiency swap for Cassandra and MySQL can be observed for Workloads A/F vs. B-D. 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.

4.3. *Workload Type Effects*

Table 5 summarizes cross-workload energy efficiency difference for workloads A, B, C, D, and F compared to the workload E. For example, it can be seen that Cassandra workload A uses 24% of energy used for workload E. As shown in Table 5 energy consumption difference can reach a factor of 35 for Redis across all databases and workloads. Omitting



(a) WorkloadA (b) Workload B (c) Workload C

(d) Workload D (e) Workload E (f) Workload F

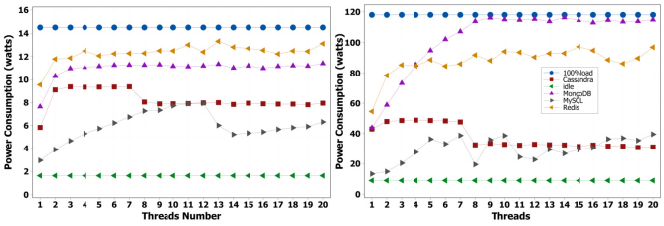
Fig. 2. Energy consumption of local processing on a mid-range CPU for Cassandra, MongoDB, MySQL and Redis.

Table 5
Energy usage efficiency of different workloads (A, B, C, D, and F) to Workload E.

Host name	Database	A	B	C	D	F	Max variance
Mid-range CPU	Cassandra	24%	21.4%	19.7%	23%	28%	5.07
	MongoDB	42.3%	35.8%	35.4%	31.6%	54.1%	3.16
	MySQL	37.8%	28.1%	26%	28.5%	47.7%	3.84
	Redis	3.6%	3%	2.9%	3.4%	4%	34.26
High-end CPU	Cassandra	24.1%	23.6%	23%	23.6%	29.5%	4.35
	MongoDB	37.7%	27.1%	28.3%	28.2%	45.4%	3.69
	MySQL	30.2%	28.8%	25.6%	29.3%	39.6%	3.91
	Redis	3.8%	3.4%	2.8%	2.8%	4.4%	35.78
Mid-range CPU → High-end CPU	Cassandra	33.3%	30.1%	27.7%	32.6%	38.3%	3.61
	MongoDB	27.1%	29.1%	26.7%	25%	35.9%	4
	MySQL	20.7%	25%	37.4%	26.1%	32.9%	4.82
	Redis	4.4%	3.7%	3.4%	4.2%	6.4%	29.49

workload E, which is known to pose challenges for Redis, the difference ranges from a factor of 3.16 for MongoDB to 5.07 for Cassandra (last column in Table 5).

The variance between workloads is not the same across different CPUs and does not generally reduce when a more powerful CPU is used. For instance, workload-induced 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 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.



(a) Mid-range CPU (b) High-end CPU

Fig. 3. CPU power consumption of workload A for all databases, as well as idle and maximum utilization of (a) mid-range CPU, and (b) high-end CPU.

In summary, the answer to RQ1 is: Redis is more energy-efficient for most workloads, except workload 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.

5. CPU-Related Energy Efficiency Evaluation

This section focuses on the role of 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 can certainly complete the tasks faster by achieving a significantly higher ops/sec throughput, the slower CPUs consumes less energy per 1000 operations. Figures 3a and 3b demonstrate power consumption for mid-range and high-end CPUs, respectively.

We measure the idle CPU energy usage, with all databases and non-system processes stopped for one hour to establish idle usage baseline. We also capture the energy consumption of both CPUs under 100% utilization using the "stress" tool[11] to determine maximum possible energy usage. For brevity, we plotted power usage for workload A for all databases. The following observations can be made based on the results.

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. It is also somewhat unexpected to see the minor drops in energy usage for Cassandra and MySQL at 8 and 13 threads, respectively. A likely explanation is that this drop occurs due to the operation latency increasing as number of threads increases. In other words, after a certain load, databases do less useful work and spend more time waiting for the results.

Second, the key insight emerges from the comparison of the absolute values of idle to fully loaded energy usage of the CPUs tested. Specifically, mid-range CPU consumes

[11] Stress tool: https://github.com/cooljeanius/stress-1.0.4

14.5 Watts, while high-end CPU consumes 119 Watts at full load. Therefore, under ideal conditions, the high-end CPU needs to perform 8.2 times more operations per second in order to achieve the same energy efficiency (J/ops) as the mid-range CPU. Under real-world conditions, due to the imperfection of database implementations this factor is reduced to around a factor of 4 for MySQL/Cassandra and a factor of 7 for Redis. While MongoDB could fully utilize the energy available for high-end CPU, it performed worse on mid-range CPU causing the equivalent performance factor to be closer to 10.

In summary, the answer to RQ3 is complex and depends on three key factors. Namely, the maximum power that a given CPU can consume, how much of that energy can a given database utilize and the raw computational CPU power (ops/sec). Precise answer is complicated by the fact that even the same database may utilize a different proportion of CPU power as algorithms may scale differently on different CPUs.

The mid-range CPU proved to be more energy-efficient than the high-end one under all workloads for all databases. However, the high-end CPU achieved a significantly higher throughput for all tasks. Therefore, actual CPU and database selection decision depends on the expected usage scenarios and system goals. Choosing a more energy-efficient CPU could be beneficial if a lower throughput is acceptable.

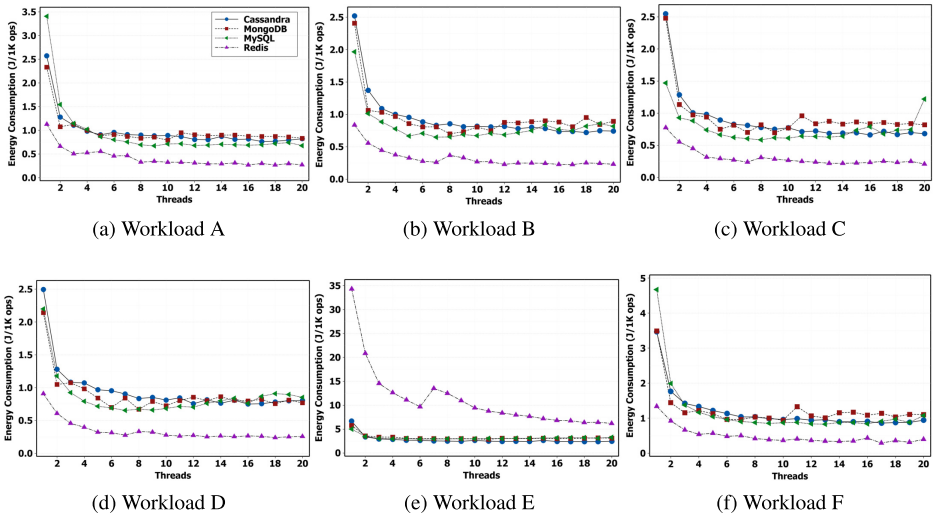
6. Network Offloading Energy Efficiency

This section discusses answers on how beneficial is database workload offloading in terms of energy efficiency (RQ4), and how does the network latency affect offloading energy efficiency (RQ5). Two sets of experiments were conducted to measure network effects. Firstly, a close-proximity offloading was performed within University of Adelaide LAN (Section 6.1). Then, a longer-distance offloading was conducted over WAN (Section 6.2). The energy measurements were only conducted on client side as servers are considered to have "infinite" power available unlike the resource-constrained client nodes. Such offloading is a common strategy in optimizing energy usage and overall system performance (Lu *et al.*, 2020). Measuring energy usage on both client and server sides would most likely demonstrate a higher total energy consumption (compared to local processing) due to the data transmission overheads. Some research suggests that while offloading computations to a remote cloud could be beneficial for some workloads, extra factors such as data privacy and security must be considered (Kumar and Lu, 2010).

6.1. *Close-Proximity Offloading*

In this set of experiments, database servers resided on the high-end CPU, while YCSB was running on the mid-range CPU (Scenario 3). The same set of workloads was executed by YCSB and the total client-side energy consumption was measured. As shown in Fig. 4, the relative database energy efficiency is similar to local workloads. Redis struggles in workload E, while using less energy for other workloads.

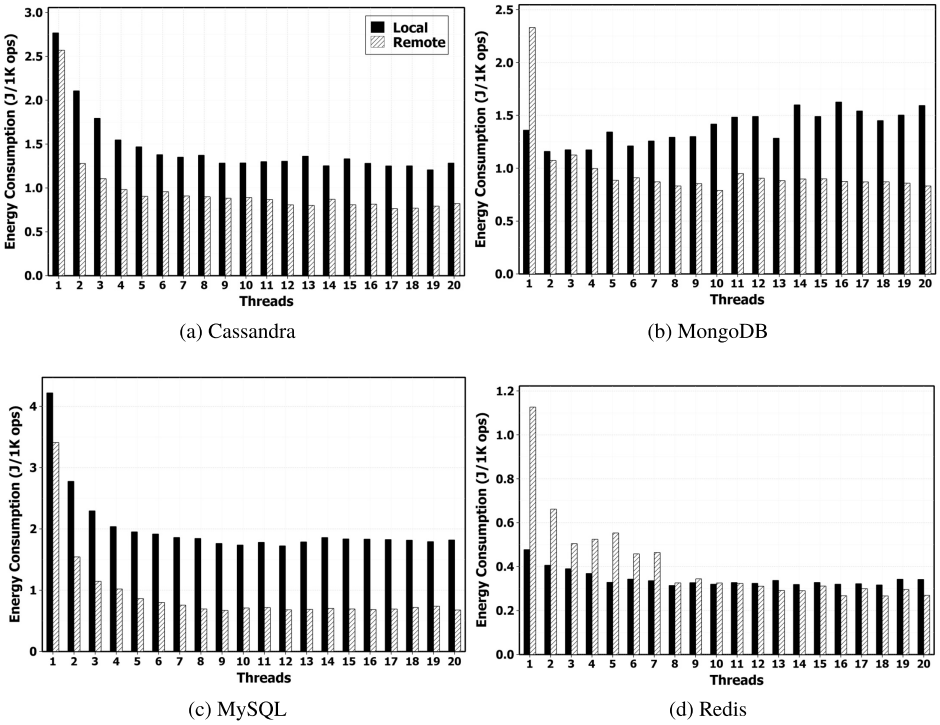
Comparing the local and close-proximity offloading reveals that in some cases offloading can be beneficial in terms of energy efficiency. Figure 5 illustrates the difference



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

(d) Workload D (e) Workload E (f) Workload F

Fig. 4. 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).



(a) Cassandra (b) MongoDB

(c) MySQL (d) Redis

Fig. 5. Energy consumption of workload A for databases running on a mid-range CPU (**local**) vs. energy consumption of databases offloading from mid-range CPU to high-end CPU with distance of several meters (**remote**).

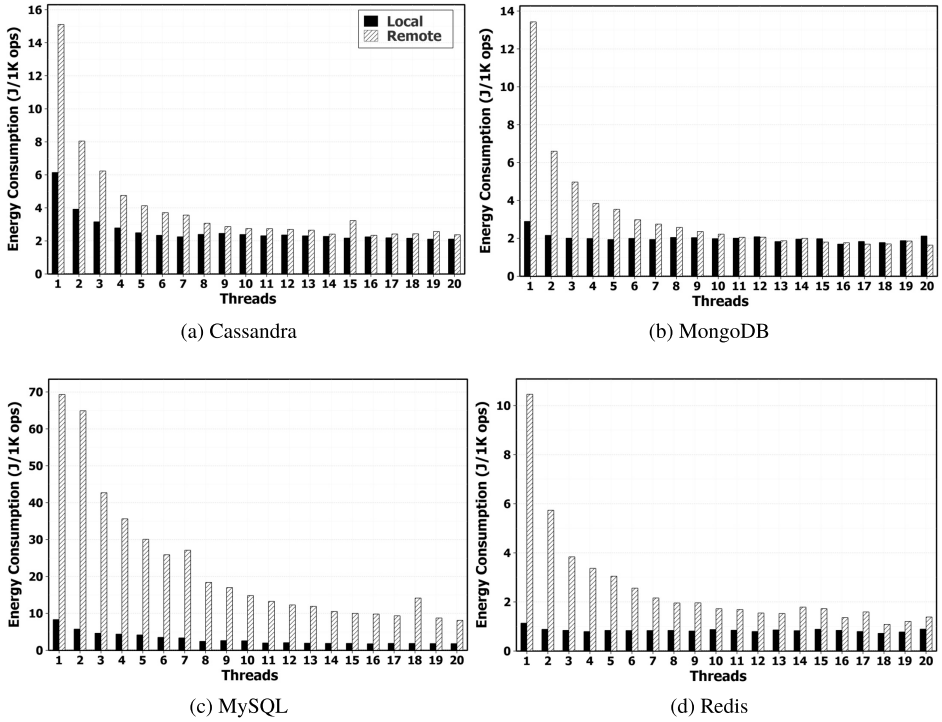
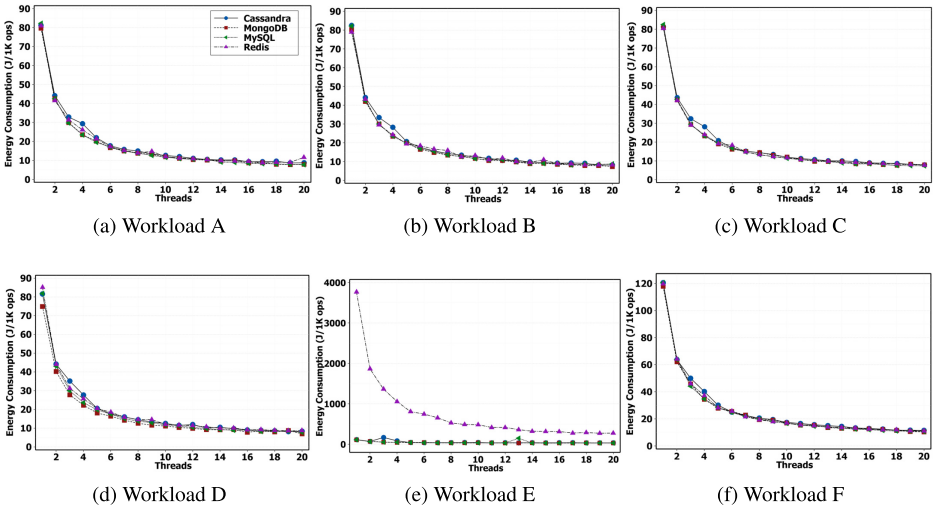


Fig. 6. Energy consumption of workload A for databases running on a high-end CPU (**local**) vs. energy consumption of databases offloading from high-end CPU to OpenStack VM (**remote**).

between energy usage in local and close-proximity offloading scenarios. We compare local mid-range CPU workload execution and offloading from mid-range CPU to high-end CPU.

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 from offloading as much.

Figure 6 shows the results of offloading from high-end CPU to a somewhat slower Openstack VM (Scenario 4). Packet round-trip time was 1.5ms for this set of experiments. This scenario 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 only 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



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

(d) Workload D (e) Workload E (f) Workload F

Fig. 7. Energy consumption of databases, where YCSB runs on mid-range CPU and database servers run on high-end CPU (long distance).

and Cassandra when a low number of threads is used. For up to 6–7 threads, even close-proximity offloading to a slower CPU worsens energy efficiency.

These results are explained by the extra time that 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.

6.2. *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 (50 ms vs. 1.5 ms).

As shown in Fig. 7, 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 average operation latency in local scenarios is typically around tens of microseconds, clearly 50 ms network latency dominates database latency.

Figure 8 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.

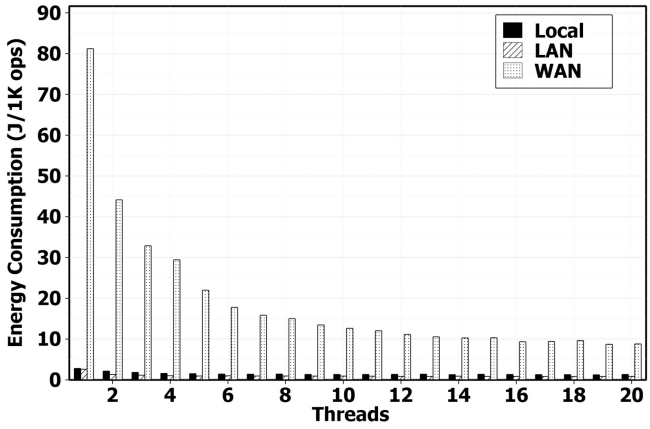


Fig. 8. A comparison of energy consumption of Cassandra (Workload A) running on a mid-range CPU between local and offloaded processing (close-proximity and long distance).

Based on the obtained energy usage pattern measurements conducted, the answers to RQ4 and RQ5 can be formulated as follows. RQ4: workload offloading can be beneficial for some databases (all except Redis) only if the remote CPU is significantly more powerful in terms of computational capacity. RQ5: only low-latency network environments can provide benefits of workload offloading as high network packet latency starts dominating database-induced latencies. Based on these answers, a generic goal-based energy efficiency measurement and decisions approach is presented in the next section.

7. Goal-Based Database Energy Efficiency Evaluation Framework

A lightweight energy efficiency estimating framework has been developed as a part of this study. This framework enables automated database workload offloading energy efficiency measurement for given local and remote CPUs. YCSB-based workloads are used to obtain the measurements.

Network data transmission- and encryption-related energy usage overheads may be measured separately, if necessary. While specific conditions may lead to significant overheads, under the testing environment these overheads were negligible (around 1%). The proposed framework aids comparing a local host against a remote node in terms of energy efficiency.

Depending on the intended database usage and goals, energy-aware decisions can be made based on the obtained results. Two potential goals are considered: highest throughput and lowest energy usage. Different hardware and network environments may lead to these two goals becoming mutually exclusive (e.g. remote node may achieve higher throughput only at the expense of higher energy usage).

Table 6 shows an example output for the "high-end CPU → Openstack VM" evaluation. It is shown that running workload A for Cassandra is always more energy efficient

Table 6
The recommended energy-efficient (micro-Joules/1KOps) scenarios – coloured in green – for all databases running on high-end CPU (local) and offloading from high-end CPU to OpenStack VM (remote) for workloads A and B.

	10 Threads		15 Threads		20 Threads	
Workload A	Local	Remote	Local	Remote	Local	Remote
Cassandra	56.68	62.45	49.96	50.23	47.74	48.12
MongoDB	24.90	33.56	24.23	26.18	24.23	25.51
MySQL	83.91	87.95	47.91	54.16	37.78	39.18
Redis	0.04	0.03	0.03	0.04	0.06	0.07
Workload B	Local	Remote	Local	Remote	Local	Remote
Cassandra	55.41	62.62	50.96	52.56	48.43	48.95
MongoDB	23.98	35.54	25.45	27.43	24.16	23.75
MySQL	42.03	43.14	29.68	29.98	33.11	32.93
Redis	0.06	0.06	0.06	0.06	0.07	0.06

and faster on the local node. In contrast, running workload B on MongoDB and MySQL the remote (Openstack) node becomes beneficial in terms of energy usage for 20 threads. Green cells highlight most energy efficient scenario for each database.

8. Applicability and Lessons Learnt

The most significant limitation of this study is that only CPU energy usage was considered. Taking into account RAM and disk I/O may affect our findings. Another constraint is the limited variety of hardware available, as all tested systems were Intel-based. Some prior measurements indicate that ARM may demonstrate higher energy efficiency (Abdurachmanov *et al.*, 2014). Traditional x86-targeted energy efficiency optimizations were shown not to be suitable for mobile ARM-based database usage scenarios (Yang *et al.*, 2014). A significant difference in energy efficiency has been observed for Intel CPUs of different generations (Lopez-Novoa, 2019). Lastly, only a subset of databases was considered.

Note that due to the limits of RAPL counters, longer workloads may cause energy usage readings to overflow returning negative results. Extra care must be taken in such cases to counter RAPL overflows. Powerful CPUs may overflow the RAPL counter in less than 30 minutes under load.

Redis exhibited stability issues with long-distance offloading. Multiple crashes occurred for Redis-related experiments causing interruptions. Extra care was taken to detect such crashes even during the data loading phase, as otherwise the experiments continued with an incomplete dataset. The high number of crashes forced to run Redis experiments with a reduced number of records. While Redis crashes could be attributed to low link quality, other databases worked correctly.

Additional measures were taken to minimize measurement noise. Clean operating systems were installed for all of the experiments and all non-essential services have been disabled. While this later proved to have an insignificant effect, all of the databases, except the currently tested one, were stopped to exclude possible background maintenance activities.

Compared to local runs, remote offloading faced several technical challenges. Due to firewall rules, SSH tunnelling was used in order to achieve inter-network offloading. SSH protocol involves encryption, which allows to secure the communication between database client and server nodes. Encryption, however, introduces overhead which may impact the energy efficiency. Thus, measurements were conducted to determine the network transmission and encryption overheads separately. Network transmission and encryption related energy consumption overheads were observed to be negligible compared to the database energy consumption and thus not taken into consideration. Actual energy consumption overhead was measured to be on average around 1% of the overall energy consumed during workload execution.

Interestingly, not all databases were able to achieve 100% consumption of the available CPU energy. In other words, stress-testing the CPU consumed significantly more energy than most of the databases did under load of 20 threads. This indicates that CPU architecture-specific optimizations may potentially improve database performance.

Idle energy usage measurements were also taken to determine the associated energy waste. The results indicate high-end and mid-range CPUs consume 9 and 1.6 Watts, respectively. Thus, to be more energy-efficient, the high-end CPU needs to achieve significantly higher throughput.

9. Conclusion and Future Work

It is challenging to provide generalized results in the context of energy efficiency due to the large amount of different factors involved. However, a number of interesting quantitative results have been obtained throughout this research study. 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 optimizations 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. Deploying workload offloading in practice requires preliminary estimations of *hardware capacity*, *workload type* and *network latency*. Unsuitable resources and conditions mean that performing the computations locally would be more energy efficient. This is explained by the high overhead of two-way data transmission and the associated idle waits. Note that the performed measurements revealed that data transmission energy usage is negligible (1.2–1.5%) compared to the idling energy waste. Running an energy usage estimation benchmark in a given environment prior to making a choice is thus highly recommended. Extending the applicability of the offloading approach to clustered database environments and dynamically moving client nodes would require dynamic offloading feasibility assessment.

Future research lines can focus on a number of potentially useful extensions of this work. Firstly, testing more CPUs (AMD and ARM). Secondly, tweaking memory man-

agement related settings such as swap file size, disk I/O caching and amount of memory used by a database might reveal further differences between databases tested.

Thirdly, individual hardware components energy usage can be measured separately. More fine-grained results could be obtained by considering different types of disk storage, such as SSD and spinning hard disks of different speeds.

Fourthly, database-tailored workloads can be executed to estimate the effect of different data storage architectures. For example, document-based storage might demonstrate improved energy efficiency for shorter documents. Similarly, a relational database may perform differently depending on the record fields number.

Lastly, OS-specific effects could also impact energy efficiency of databases. Such difference might not directly relate to the database quality but rather indicate different OS-specific disk I/O caching approaches. This knowledge would be useful in making informed decisions on optimal database selection in terms of energy efficiency.

References

Abdurachmanov, D., Elmer, P., Eulisse, G., Knight, R., Niemi, T., Nurminen, J.K., Nyback, F., Pestana, G., Ou, Z., Khan, K.N. (2014). Techniques and tools for measuring energy efficiency of scientific software applications. *CoRR*, *abs/1410.3440*.

Abouzeid, A., Bajda-Pawlikowski, K., Abadi, D., Silberschatz, A., Rasin, A. (2009). HadoopDB: an architectural hybrid of MapReduce and DBMS technologies for analytical workloads. *Proceedings of the VLDB Endowment*, 2(1), 922–933.

Beyer, D., Wendler, P. (2020). CPU energy meter: a tool for energy-aware algorithms engineering. In: Biere, A., Parker, D. (Eds.), *Tools and Algorithms for the Construction and Analysis of Systems - 26th International Conference, TACAS 2020, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2020, Dublin, Ireland, April 25–30, 2020, Proceedings, Part II*. *Lecture Notes in Computer Science*, Vol. 12079. Springer, pp. 126–133. https://doi.org/10.1007/978-3-030-45237-7_8.

Bourdon, A., Noureddine, A., Rouvoy, R., Seinturier, L. (2013). Powerapi: a software library to monitor the energy consumed at the process-level. *ERCIM News*, *2013*(92).

Calandrini, G., Vicente, A.G., Muñoz, I.B., Revenga, P.A., Lázaro, J.L., Toledo-Moreo, F.J. (2013). Power measurement methods for energy efficient applications. *Sensors*, 13(6), 7786–7796. https://doi.org/10.3390/s130607786.

Capra, E., Francalanci, C., Slaughter, S. (2012). Measuring application software energy efficiency. *IT Professional*, 14(2), 54–61. https://doi.org/10.1109/MITP.2012.39.

Cheong, S.-K., Lim, C.S., Cho, B.-C. (2012). Database processing performance and energy efficiency evaluation of DDR-SSD and HDD storage system based on the TPC-C. In: *2012 International Conference on Cloud Computing and Social Networking (ICCCSN)*. IEEE, pp. 1–3.

Chong, F.T., Heck, M.J.R., Ranganathan, P., Saleh, A.A.M., Wassel, H.M.G. (2014). Data center energy efficiency: improving energy efficiency in data centers beyond technology scaling. *IEEE Design and Test*, 31(1), 93–104.

Colmant, M., Kurpicz, M., Felber, P., Huertas, L., Rouvoy, R., Sobe, A. (2015). Process-level power estimation in vm-based systems. In: *Proceedings of the Tenth European Conference on Computer Systems*, pp. 1–14.

Desrochers, S., Paradis, C., Weaver, V.M. (2016). A validation of DRAM RAPL power measurements. In: Jacob, B. (Ed.), *Proceedings of the Second International Symposium on Memory Systems, MEMSYS 2016, October 3–6, 2016*. ACM, pp. 455–470.

Fahad, M., Shahid, A., Manumachu, R.R., Lastovetsky, A. (2019). A comparative study of methods for measurement of energy of computing. *Energies*, 12(11), 2204.

Ghaleb, T.A. (2019). Software energy measurement at different levels of granularity. In: *2019 International Conference on Computer and Information Sciences (ICCIS)*. IEEE, pp. 1–6. 978-1-5386-8126-8. https://doi.org/10.1109/ICCISci.2019.8716456.

Gomes, C., Tavares, E., Junior, M.N.d.O. (2020). Energy consumption evaluation of nosql dbmss. In: *Anais do XV Workshop em Desempenho de Sistemas Computacionais e de Comunicação*, pp. 71–81. SBC.

Graefe, G. (2008). Database servers tailored to improve energy efficiency. In: Apel, S., Rosenmüller, M., Saake, G., Spinczyk, O. (Eds.), *EDBT'08 Workshop on Software Engineering for Tailor-made Data Management, Proceedings, Nantes, France, March 29, 2008*. University of Magdeburg, pp. 24–28.

Haas, S., Arnold, O., Nöthen, B., Scholze, S., Ellguth, G., Dixius, A., Höppner, S., Schiefer, S., Hartmann, S., Henker, S., Hocker, T., Schreiter, J., Eisenreich, H., Schlüßler, J., Walter, D., Seifert, T., Pauls, F., Hasler, M., Chen, Y., Hensel, H., Moriam, S., Matús, E., Mayr, C., Schüffny, R., Fettweis, G.P. (2016). An MPSoC for energy-efficient database query processing. In: *Proceedings of the 53rd Annual Design Automation Conference, DAC 2016, Austin, TX, USA, June 5–9, 2016*. ACM, pp. 112–11126.

Hackenberg, D., Ilsche, T., Schöne, R., Molka, D., Schmidt, M., Nagel, W.E. (2013). Power measurement techniques on standard compute nodes: a quantitative comparison. In: *2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. IEEE, pp. 194–204.

Härder, T., Hudlet, V., Ou, Y., Schall, D. (2011). Energy efficiency is not enough, energy proportionality is needed! In: Xu, J., Yu, G., Zhou, S., Unland, R. (Eds.), *Database Systems for Adanced Applications - 16th International Conference, DASFAA 2011, International Workshops: GDB, SIM3, FlashDB, SNSMW, DaMEN, DQIS, Hong Kong, China, April 22–25, 2011. Proceedings*. *Lecture Notes in Computer Science*, Vol. 6637. Springer, pp. 226–239.

Hudlet, V., Schall, D. (2011). Measuring energy consumption of a database cluster. In: Härder, T., Lehner, W., Mitschang, B., Schöning, H., Schwarz, H. (Eds.), *Datenbanksysteme für Business, Technologie und Web (BTW), 14. Fachtagung des GI-Fachbereichs "Datenbanken und Informationssysteme" (DBIS), 2.-4.3.2011*. LNI, Vol. P-180. GI, pp. 734–737.

Jiang, C., Wang, Y., Ou, D., Li, Y., Zhang, J., Wan, J., Luo, B., Shi, W. (2019). Energy efficiency comparison of hypervisors. *Sustainable Computing: Informatics and Systems*, 22, 311–321.

Jiang, C., Fan, T., Gao, H., Shi, W., Liu, L., Cérin, C., Wan, J. (2020). Energy aware edge computing: a survey. *Computer Communications*, 151, 556–580.

Jin, C., de Supinski, B.R., Abramson, D., Poxon, H., DeRose, L., Dinh, M.N., Endrei, M., Jessup, E.R. (2017). A survey on software methods to improve the energy efficiency of parallel computing. *International Journal of High Performance Computing Applications*, 31(6), 517–549.

Jin, Y., Xing, B., Jin, P. (2013). Towards a benchmark platform for measuring the energy consumption of database systems. *Advanced Science and Technology Letters*, 29, 385–389.

Johann, T., Dick, M., Naumann, S., Kern, E. (2012). How to measure energy-efficiency of software: metrics and measurement results. In: Kazman, R., Lago, P., Meyer, N., Morisio, M., Müller, H.A., Paulisch, F., Scanniello, G., Zimmermann, O. (Eds.), *First International Workshop on Green and Sustainable Software, GREENS 2012, June 3, 2012*. IEEE Computer Society, pp. 51–54.

Kalaitzoglou, G., Bruntink, M., Visser, J. (2014). A practical model for evaluating the energy efficiency of software applications. In: *ICT for Sustainability 2014 (ICT4S-14), August 25, 2014*. Atlantis Press.

Kavanagh, R.E., Djemame, K. (2019). Rapid and accurate energy models through calibration with IPMI and RAPL. *Concurrency and Computation: Practice and Experience*, 31(13). https://doi.org/10.1002/cpe.5124.

Khan, N.K. (2018). Energy Measurement and Modeling in High Performance Computing with Intel's RAPL.

Khan, K.N., Hirki, M., Niemi, T., Nurminen, J.K., Ou, Z. (2018). RAPL in action: experiences in using RAPL for power measurements. *TOMPECS*, 3(2), 9–1926. https://doi.org/10.1145/3177754.

Kuhlenkamp, J., Klems, M., Röss, O. (2014). Benchmarking scalability and elasticity of distributed database systems. *Proceedings of the VLDB Endowment*, 7(12), 1219–1230.

Kumar, K., Lu, Y. (2010). Cloud computing for mobile users: can offloading computation save energy? *IEEE Computer*, 43(4), 51–56. https://doi.org/10.1109/MC.2010.98.

Li, T., Yu, G., Liu, X., Song, J. (2014). Analyzing the waiting energy consumption of NoSQL databases. In: *IEEE 12th International Conference on Dependable, Autonomic and Secure Computing, DASC 2014, August 24–27, 2014*. IEEE Computer Society, pp. 277–282. https://doi.org/10.1109/DASC.2014.56.

Lopez-Novoa, U. (2019). Exploring performance and energy consumption differences between recent intel Processors. In: *2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation, SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI 2019, Leicester, United Kingdom, August 19–23, 2019*. IEEE, pp. 263–267.

Lu, F., Gu, L., Yang, L.T., Shao, L., Jin, H. (2020). Mildip: an energy efficient code offloading framework in mobile cloudlets. *Information Sciences*, 513, 84–97. https://doi.org/10.1016/j.ins.2019.10.008.

Mansouri, Y., Babar, M.A. (2021). A review of edge computing: features and resource virtualization. *Journal of Parallel and Distributed Computing*, 150, 155–183.

Mansouri, Y., Prokhorenko, V., Babar, M.A. (2020). An automated implementation of hybrid cloud for performance evaluation of distributed databases. *Journal of Network and Computer Applications*, 167, 102740.

Owusu, F., Pattinson, C. (2012). The current state of understanding of the energy efficiency of cloud computing. In: Min, G., Wu, Y., Liu, L.C., Jin, X., Jarvis, S.A., Al-Dubai, A.Y. (Eds.), *11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2012, June 25–27, 2012*. IEEE Computer Society, pp. 1948–1953.

Pereira, R., Couto, M., Ribeiro, F., Rua, R., Cunha, J., Fernandes, J.P., Saraiva, J. (2017). Energy efficiency across programming languages: how do energy, time, and memory relate? In: Combemale, B., Mernik, M., Rumpe, B. (Eds.), *Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering, SLE 2017, Vancouver, BC, Canada, October 23–24, 2017*. ACM, pp. 256–267.

Phung, J., Lee, Y.C., Zomaya, A.Y. (2019). Energy efficiency evaluation of distributed systems. In: Rodrigues, J.M.F., Cardoso, P.J.S., Monteiro, J., Lam, R., Krzhizhanovskaya, V.V., Lees, M.H., Dongarra, J.J., Sloot, P.M.A. (Eds.), *Computational Science – ICCS 2019*. Springer International Publishing, Cham, pp. 756–763. 978-3-030-22750-0.

Pinto, G., Castor, F. (2017). Energy efficiency: a new concern for application software developers. *Communications of the ACM*, 60(12), 68–75. https://doi.org/10.1145/3154384.

Pisharath, J., Choudhary, A.N., Kandemir, M.T. (2004). Reducing energy consumption of queries in memory-resident database systems. In: Irwin, M.J., Zhao, W., Lavagno, L., Mahlke, S.A. (Eds.), *Proceedings of the 2004 International Conference on Compilers, Architecture, and Synthesis for Embedded Systems, CASES 2004, September 22–25, 2004*. ACM, pp. 35–45.

Prokhorenko, V., Babar, M.A. (2023). Energy Efficiency Evaluation of Local and Offloaded Data Processing. *CSIT*. https://doi.org/10.51408/csit2023_12.

Rabl, T., Gómez-Villamor, S., Sadoghi, M., Muntés-Mulero, V., Jacobsen, H.-A., Mankovskii, S. (2012). Solving big data challenges for enterprise application performance management. *Proceedings of the VLDB Endowment*, 5(12), 1724–1735.

Rauber, T., Rünger, G., Schwind, M. (2014). Energy measurement and prediction for multi-threaded programs. In: *2014 Spring Simulation Multiconference, SpringSim '14, Proceedings of the High Performance Computing Symposium*. ACM, p. 20.

Rodríguez-Martínez, M., Valdivia, H., Seguel, J., Greer, M. (2011). Estimating power/energy consumption in database servers. In: Dagli, C.H. (Ed.), *Proceedings of the Complex Adaptive Systems 2011 Conference, Chicago, Illinois, USA, October 31–November 2, 2011*. *Procedia Computer Science*, Vol. 6. Elsevier, pp. 112–117.

Rotem, E., Naveh, A., Ananthakrishnan, A., Weissmann, E., Rajwan, D. (2012). Power-management architecture of the intel microarchitecture code-named sandy bridge. *IEEE Micro*, 32(2), 20–27.

Saxe, E. (2010). Power-efficient software. *ACM Queue*, 8(1), 10. https://doi.org/10.1145/1698223.1698225.

Schall, D., Hudlet, V., Härder, T. (2010). Enhancing energy efficiency of database applications using SSDs. In: Desai, B.C., Leung, C.K., Mudur, S.P. (Eds.), *Canadian Conference on Computer Science & Software Engineering, C3S2E 2010, Montreal, Quebec, Canada, May 19–20, 2010, Proceedings*. *ACM International Conference Proceeding Series*. ACM, pp. 1–9.

Stier, C., Koziolek, A., Groenda, H., Reussner, R.H. (2015). Model-based energy efficiency analysis of software architectures. In: Weyns, D., Mirandola, R., Crnkovic, I. (Eds.), *Software Architecture – 9th European Conference, ECSA 2015, Dubrovnik/Cavtat, Croatia, September 7–11, 2015, Proceedings*. *Lecture Notes in Computer Science*, Vol. 9278. Springer, pp. 221–238.

Tsirogiannis, D., Harizopoulos, S., Shah, M.A. (2010). Analyzing the energy efficiency of a database server. In: Elmagarmid, A.K., Agrawal, D. (Eds.), *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010, June 6–10, 2010*. ACM, pp. 231–242.

Weaver, V.M., Johnson, M., Kasichayanula, K., Ralph, J., Luszczek, P., Terpstra, D., Moore, S. (2012). Measuring energy and power with PAPI. In: *2012 41st International Conference on Parallel Processing Workshops*. IEEE, pp. 262–268.

Xu, C., Zhao, Z., Wang, H., Shea, R., Liu, J. (2015). Energy efficiency of cloud virtual machines: from traffic pattern and CPU affinity perspectives. *IEEE Systems Journal*, 11(2), 835–845.

Xu, Z., Tu, Y., Wang, X. (2012). PET: reducing database energy cost via query optimization. *PVLDB*, 5(12), 1954–1957. https://doi.org/10.14778/2367502.2367546.

Yang, P., Jin, P., Yue, L. (2014). Exploiting the performance-energy tradeoffs for mobile database applications. *Journal of Universal Computer Science,*, 20(10), 1488–1498. https://doi.org/10.3217/jucs-020-10-1488.

Zhang, H., Hoffman, H. (2015). A quantitative evaluation of the RAPL power control system. *Feedback Computing*.

Zhou, Y., Taneja, S., Qin, X., Ku, W., Zhang, J. (2020). EDOM: Improving energy efficiency of database operations on multicore servers. *Future Generation Computer Systems*, 105, 1002–1015.

V. Prokhorenko received the PhD degree in computer science from The University of South Australia, in 2017. He is a researcher with the Centre for Research on Engineering Software Technologies, The University of Adelaide, Adelaide. He has more than 14 years of experience in software engineering with main areas of expertise, including the investigation of technologies related to software resilience, trust management, and big data solutions hosted within OpenStack and Microsoft Azure cloud platforms.

M.A. Babar is a professor in the School of Computer Science, University of Adelaide. Prof. Babar has established an interdisciplinary research centre called CREST (Centre for Research on Engineering Software Technologies), where he directs the research, education, and engineering activities of more than 25 researchers and engineers. He also leads a theme on architecture and platform for security as a service in the Cyber Security Cooperative Research Centre. Prof. Babar has initiated and led several interdisciplinary R&DI programs in collaboration with industry/govt partners in Australia and Europe. Professor Babar has authored/co-authored more than 280 peer-reviewed papers at premier software journals and conferences. Professor Babar obtained a PhD in computer science and engineering from the school of computer science and engineering of the University of New South Wales, Australia.