Chapter 11
Benchmarking the Energy Efficiency of Servers

Jóakim von Kistowski, Klaus-Dieter Lange, and Jeremy A. Arnold

The measurement and benchmarking of computing server energy efficiency has become an ever more important issue over the last decades. In addition to mobile and other end-user devices, the energy efficiency of data centers and servers has gained attention as power consumption increases and is expected to continue increasing in the future. In 2010, the U.S. Environmental Protection Agency (U.S. EPA) estimated that 3% of the entire energy consumption in the USA is caused by data center power draw (Lange and Tricker, 2011). According to a New York Times study from 2012, data centers worldwide consume about 30 billion watts, which is equivalent to the approximate output of 30 nuclear power plants (Glanz, 2012).

Improving the energy efficiency of data centers and servers requires the ability to measure and rate that efficiency. A comprehensive rating method can enable data center owners to purchase more efficient devices. It can also help service providers to select the most efficient servers for their specific applications. Finally, a reliable rating method makes it possible for regulators to define standards and regulations specifying which devices are considered energy efficient and which are not. To achieve these goals, a rating method must meet a number of criteria based on the generic benchmark quality criteria we discussed in Chapter 1, that is, it must be relevant, reproducible, fair, and verifiable.

Relevance in the context of energy efficiency is challenging, as most servers in modern day data centers are not being utilized to their full capacity. Instead, servers are used to serve requests that arrive over time and are provisioned with additional capacity in order to be able to cope with variations in load such as unexpected bursts. This leads to an average load somewhere between 10% and 50% (Barroso and Hölzle, 2007). However, servers consume a different amount of power depending on the load level. An energy-efficiency benchmark must account for this and measure these states to obtain a complete picture of the server's energy efficiency. Older server efficiency benchmarks did not consider this issue of low load power consumption. While some benchmarks used for power and efficiency testing, such as JouleSort by Rivoire et al. (2007), run multiple workloads in a suite, these benchmarks are executed only at full load.

This chapter describes a rating methodology developed by the SPEC OSG Power Subcommittee for commodity servers. It is designed to characterize and rate the energy efficiency of a SUT for multiple load levels, showcasing load level differ-

S. Kounev et al., *Systems Benchmarking*, https://doi.org/10.1007/978-3-030-41705-5_11

ences in system behavior regarding energy efficiency. The methodology was first implemented in the SPECpower_ssj2008 benchmark and later extended with more workloads, metrics, and other application areas for the SPEC Server Efficiency Rating Tool (SERT). The SERT suite was developed to fill the need for a rating tool that can be utilized by government agencies in their regulatory programs, for example, the U.S. Environmental Protection Agency (EPA) for the use in the Energy Star program for servers.

11.1 SPEC Power and Performance Benchmark Methodology

All SPEC Power benchmarks and rating tools share the underlying *SPEC Power and Performance Benchmark Methodology* (SPECpower Committee, 2014) as their basis. The methodology has been developed by the SPEC OSG Power Committee as a tool for the analysis and evaluation of the energy efficiency of server systems. It was first implemented in SPECpower_ssj2008 (Lange, 2009) and later in the SPEC SERT (Lange and Tricker, 2011) and SPEC Chauffeur Worklet Development Kit (SPECpower Committee, 2017a). In the following, we discuss the measurement methodology and its general building blocks.

11.1.1 Device Setup

The SUT is at the center of the methodology's power measurement setup. It is a physical system that runs the workloads used for evaluation. The SUT's power consumption and its performance during testing are used to derive the energy-efficiency score. Performance metrics are gathered from the SUT using a testing software harness. The actual test execution software on the SUT is referred to as the *host* software. The host spawns separate on-SUT processes, referred to as *clients*, for each logical CPU core (hardware thread). These client processes execute the executable part of the workload. Spawning multiple clients allows for easy parallelization, as workloads can simply be executed in parallel within different isolated client environments. Alternatively, a client may also be configured to span multiple logical CPUs, in which case the executable workload is expected to run in a multi-threaded environment, utilizing all available CPU resources. The overarching goal of this parallelization scheme is to ensure scalability, which, in this case, is considered to be a subset of the *relevance* criterion from Chapter 1.

In most cases, a transactional workload (see Chapter 8) is executed on the clients. The clients collect the performance metrics for their workload and forward this information to the host. The workload is controlled by the *controller* system. It coordinates which workload to run at which load level. It also collects all measurements both from the SUT, as well as from external measurement devices, and it calculates

the metrics and scores. The *director* manages all software instances as well as all measurement devices. The setup is illustrated in Figure 11.1.

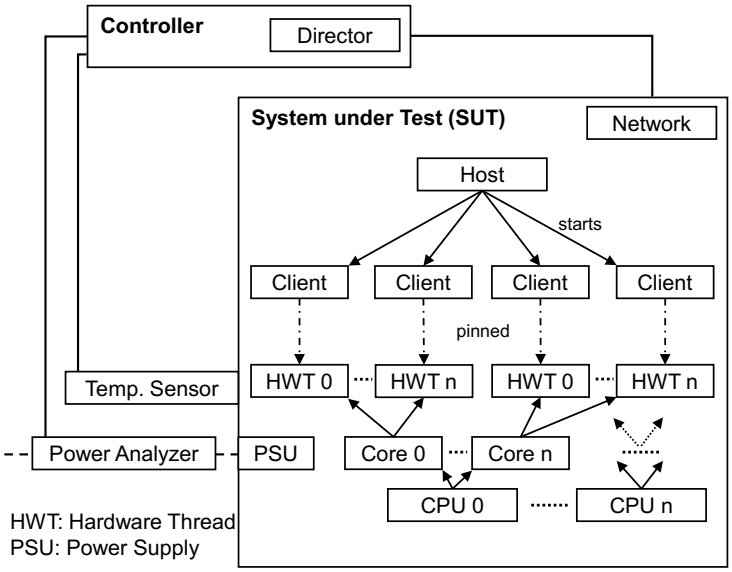

Fig. 11.1: Typical server power measurement device setup

The power methodology requires at least one external power analyzer and one temperature sensor. The power analyzer measures the power consumption of the entire SUT, whereas the temperature sensor verifies the validity of measurements, ensuring that all experiments are conducted under similar environmental conditions. External power and temperature instrumentation are used, as opposed to potential internal instrumentation, as the methodology makes no assumptions about the internal structure of the SUT, allowing for maximum portability. Reliance on external power measurement devices also enables the definition of tight constraints on the accuracy of the power measurement devices. Specifically, power measurement devices must feature a maximum measurement error of 1% or better.

The use of internal instrumentation may be adequate and appropriate for research purposes when working with a specific hardware model and when the researcher understands exactly what the sensors are measuring. Nonetheless, making comparisons across different models or architectures based on internal sensors is likely to result in inaccurate comparisons.

11.1.2 Load Levels

According to Barroso and Hölzle (2007), servers spend most of their time in a CPU utilization range between 10% and 50%. To account for this, workloads within the *SPEC Power and Performance Benchmark Methodology* are designed to measure system energy efficiency at multiple load levels. This sets benchmarks implementing the methodology apart from conventional performance benchmarks, such as SPEC CPU (cf. Chapter 10), or other energy-efficiency benchmarks, such as JouleSort (Rivoire et al., 2007) or the TPC-Energy benchmarks (Poess et al., 2010). To achieve workload execution at different load levels, a methodology-compliant benchmark calibrates the load by first determining the maximum transaction rate for the given workload on the SUT. The maximum transaction rate is measured by running concurrently on each client as many workload transactions as possible. For the calibration, the executable workload is executed according to the *closed workload* scheme (cf. Chapter 8, Section 8.3.2.1); that is, a new transaction is scheduled after the previous transaction in the respective thread (client) terminates.

This calibrated rate is then set as a 100% load level for all consecutive runs. For each target load level (e.g., 100%, 75%, 50%, 25%), the benchmark calculates the target transaction rate and derives the corresponding mean time from the start of one transaction to the start of the next transaction. During the measurement interval, these delays are randomized using an exponential distribution that statistically converges to the desired transaction rate. As a result, lower target loads consist of short bursts of activity separated by periods of inactivity. It follows that these load levels are executed according to the *open workload* scheme (cf. Chapter 8, Section 8.3.2.2), as the point in time when a transaction terminates has no bearing on the time at which the next transaction is dispatched.

Figure 11.2 shows how calibration and the following measurement intervals would run using intervals at 100%, 67%, and 33% as an example. Note that the load levels are defined as percentage of target throughput and do not indicate CPU utilization, which is a common misconception.

11.1.3 Phases and Intervals

The transactional executable workload is executed in three phases to achieve reproducible calibration and measurement results: a warm-up phase, a calibration phase, and a measurement phase. The warm-up phase runs the executable workload at full load for a short period of time discarding any measurements to account for transient side-effects. After warm up, the workload enters the calibration phase. During calibration, transactions are executed as fast as possible to determine the maximum transaction rate on the specific SUT. Finally, the measurement phase takes place. In the measurement phase, transactions are scheduled according to the targeted load level.

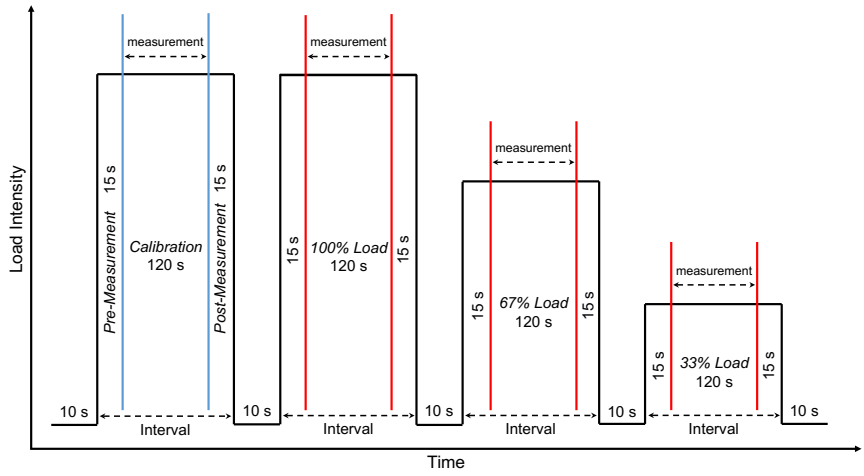

Fig. 11.2: Example intervals for the calibration and measurement phase (Kistowski, Beckett, et al., 2015)

Each phase is split into a configurable number of intervals, which serve different purposes depending on the phase in question. Each interval is the period in time, during which the actual work of the phase is being executed.

The executable workload is put to sleep for 10 s between each interval, allowing external power analyzers to adjust their range settings for the next interval. Each interval is also split into a pre-measurement, measurement, and post-measurement period. The pre-measurement period allows the interval to reach a steady state, whereas the post-measurement period ensures that the workload and hardware do not begin shutdown during measurement. The measurement period performs the phase-specific work. It measures the maximum throughput during calibration and the current throughput and power consumption during the measurement phase. In this time, all transactions are logged and power measurements are reported at 1 s intervals.

Each phase runs its intervals in sequence. The type of sequence depends on the phase in question. The warm-up phase runs multiple intervals of varying length, and the calibration phase runs multiple identical calibration intervals in sequence. The calibration result is the average throughput of those intervals. The measurement phase runs its intervals in a *graduated measurement sequence* executing workloads at gradually diminishing target transaction rates. Running multiple warm-up intervals provides higher visibility as to whether the warm-up time was sufficient to reach steady state.

11.1.4 Basic Energy-Efficiency Metric

The power methodology computes efficiency based on performance and power measurements. In addition to defining how power is measured, it must also define a performance measure and measurement method. This performance measure is intended to be used in conjunction with the power measurements in order to derive an efficiency metric. For this context, throughput (see Chapter 3, Section 3.3.2) has established itself as the commonly used metric.

As the size and execution duration may vary between different workloads, throughput is often normalized by comparing it to the throughput of a reference system—see Equation (11.1). This results in a speedup metric (see Chapter 3, Section 3.3.1).

$$\text{normalized\_throughput} = \frac{\text{measured\_throughput}}{\text{reference\_throughput}} \tag{11.1}$$

The basic energy-efficiency metric for a single point in time or single measurement interval is computed as a ratio of performance to power consumption—see Equation (11.2)—where performance is either throughput or normalized throughput.

$$\text{efficiency} = \frac{\text{performance}}{\text{power\_consumption}} \tag{11.2}$$

Alternatively, efficiency can be calculated as the ratio of work performed—see Equation (11.3)—to the energy expended, which is mathematically equivalent to performance per power consumption in the case of throughput being the primary performance metric.

$$\text{efficiency} = \frac{\text{work\_performed}}{\text{energy\_expended}} \tag{11.3}$$

Using throughput as performance metric, energy efficiency is the ratio of throughput to power consumption in Watts. This is mathematically equivalent to the alternative efficiency ratio, as work units per time divided by power equals work units divided by energy—see Equation (11.4).

$$\text{efficiency} = \frac{\text{throughput}}{\text{power\_consumption}} \left[\frac{s^{-1}}{W}\right] = \frac{\text{work\_units}}{\text{energy\_expended}} \left[\frac{1}{J}\right] \tag{11.4}$$

Both SPECpower_ssj2008 and the SPEC SERT use this base metric for each measurement interval. For each specific executable workload and each concrete load level, average throughput is normalized and then divided by average power consumption—see Equation (11.5).

$$\text{load\_level\_efficiency} = \frac{\text{normalized\_throughput}}{\text{power\_consumption}} \left[\frac{1}{J}\right] \tag{11.5}$$

11.2 SPECpower_ssj2008

The SPECpower_ssj2008 benchmark was developed by the SPEC OSG Power Committee and was the first industry-standard benchmark to measure the energy efficiency of servers. It was developed in conjunction with the initial version of the SPEC Power and Performance Benchmark Methodology and served both as the basis for developing the first draft of the methodology and as its first implementation. The lessons learned during the development of the benchmark were incorporated into the methodology. The benchmark was based on the earlier SPECjbb2005 benchmark, which was a Java implementation of a simple OLTP workload. In SPECpower_ssj2008 the workload was modified to run at ten different load levels (100%, 90%, ..., 10%) as well as an active idle measurement, rather than only measuring performance at full utilization like its predecessor.

The order of the load levels, from 100% down to 10% and then Idle, was chosen intentionally in order to eliminate a sudden change in load between calibration and the first measurement interval. Jumping directly from 100% utilization during calibration to a 10% load is both unrealistic (for most server environments) and difficult to measure accurately, since it may take time for the server to adjust to the new load.

The benchmark is implemented in Java for portability to different operating systems and processor architectures, and it makes use of multiple threads and Java Virtual Machines to scale to different size systems. While the initial version of the benchmark only included support for measuring the energy efficiency of a single server, later updates allowed the benchmark to test multiple servers together. This capability is important for measuring the energy efficiency of blades and similar servers that utilize a shared power infrastructure.

11.2.1 Metric Calculation

The SPECpower_ssj2008 report[1] shows the throughput (ssj_ops) and power consumption as well as a load level efficiency score ("Performance to Power Ratio") for each load level. The load level efficiency score is calculated as in Equation (11.5), but in this case the throughput is not normalized. It was not necessary to normalize the performance in SPECpower_ssj2008 since there was only a single workload and the scores did not need to be combined.

The overall metric is calculated as the sum of the throughput for each load level divided by the sum of the power consumption in each load level (including active idle). This metric weights each of the load levels equally, and making improvements to either the performance score or the power consumption at any load level will result in an improvement to the score.

[1] https://www.spec.org/power_ssj2008/results/res2007q4/power_ssj2008-20071129-00015.html

One drawback of using an energy-efficiency ratio as the only primary metric of SPECpower_ssj2008 results is that it does not account for the capacity of the servers when comparing results. So while System A may be more efficient than System B, this may be irrelevant if System A does not have high enough performance to meet the needs of a particular application when System B does. So the reader of the results should be considering the maximum performance as well as the efficiency.

11.2.2 System Configuration

Another innovation in SPECpower_ssj2008 was the detailed reporting of the system configuration required in the Full Disclosure Report (FDR) for published results. While SPEC benchmarks have historically required documentation of any hardware, software, and other configuration details required to reproduce the result (in support of the reproducible and verifiable benchmark quality criteria described in Chapter 1), the SPEC OSG Power Committee recognized that many components of the system could influence the power consumption even if they did not affect performance.

As a result, the SPECpower_ssj2008 full disclosure report has more detail than most benchmarks regarding specific vendors, power supply details, and the presence of extra hardware such as additional network cards, keyboard, and optical drives which will not affect the reported performance but may influence the power consumption of the server. The run rules also require detailed disclosure of non-default firmware settings, which can often be used to influence power consumption, even if such settings are not often adjusted by most users.

11.3 SPEC Server Efficiency Rating Tool (SERT)

The SPEC Server Efficiency Rating Tool (SERT) has been developed by the SPEC OSG Power Committee as a tool for the analysis and evaluation of the energy efficiency of server systems. In contrast to energy-efficiency benchmarks such as JouleSort (Rivoire et al., 2007), the TPC-Energy benchmarks (Poess et al., 2010), and SPECpower_ssj2008, SERT does not execute an application from a specific domain. It does not aim to emulate real-world end-user workloads, but instead provides a set of focused synthetic micro-workloads called *worklets* that exercise selected aspects of the SUT. The worklets have been developed to exercise the processor, memory, and storage I/O subsystems.

For each of the server components to be stressed, SERT offers a range of worklets designed to exercise the targeted component in a different manner. This allows for thorough analysis of system energy behavior under different workload types designed to target the same component. As an example, the CryptoAES worklet profits from both specialized instruction sets, as well as better CPU to memory connectivity, whereas the SOR worklet primarily scales with processor frequency.

11.3.1 Workload and Worklets

SERT's goal is the execution of different mini-workloads at multiple load levels. Those mini-workloads are referred to as *worklets* and are grouped into worklet collections, referred to as *workloads*. Specifically, a workload is a collection of worklets with a common testing goal. All worklets within a workload are designed to test a common resource by utilizing it in a specific fashion. They execute work units, referred to as *transactions*. The SERT v2.0 suite features three separate workloads: CPU, Memory, and Storage. Each of these workloads consists of multiple worklets, which are executed at several load levels.

Each worklet's performance and power consumption are measured separately for each load level, and the energy efficiency, as well as the normalized energy efficiency, is calculated from the measurement results. The workload score is an aggregate of all the separate worklet scores. It provides a workload efficiency score, which signifies how well the tested system performed for all the worklets in the specified category (for details, see Section 11.3.2).

In the following, we describe each of the workloads in detail. Each workload was designed so that it primarily stresses the server subsystem after which it was named (the CPU workload stresses CPU, the Memory workload stresses memory, the Storage workload stresses internal storage devices). However, it is important to keep in mind that workloads do not exclusively stress that subsystem. Workloads also measure and characterize the energy efficiency of interactions between multiple subsystems. To this end, the CPU workload also utilizes some memory, the memory workload utilizes some CPU, and the storage workload utilizes some CPU and memory. All following descriptions are consistent with the SERT design document (SPECpower Committee, 2017b).

11.3.1.1 CPU Workload

The CPU workload is defined as a collection of seven CPU worklets:

1. **Compress**: De-/compresses data using a modified Lempel–Ziv–Welch (LZW) method (Welch, 1984).
2. **CryptoAES**: Encrypts/decrypts data using the AES or DES block cipher algorithms.
3. **LU**: Computes the LU factorization of a dense matrix using partial pivoting.
4. **SHA256**: Performs SHA-256 hashing transformations on a byte array.
5. **SOR** (Jacobi Successive Over-Relaxation): Exercises typical access patterns in finite difference applications.
6. **SORT**: Sorts a randomized 64-bit integer array during each transaction.
7. **Hybrid / SSJ**: The hybrid SSJ worklet stresses both CPU and memory, with either serving as the primary bottleneck, depending on the system configuration. SSJ performs multiple different simultaneous transactions, simulating an enterprise application.

11.3.1.2 Memory Workload

The memory workload consists of worklets designed to scale with installed memory. Specifically, this means that the worklets are designed to measure a higher (better) performance score with improved memory characteristics (e.g., higher bandwidth, lower latency, total memory size). The primary memory characteristics being tested are bandwidth and capacity.

The memory worklets serve as the major exception to the load level and interval specification in Section 11.1.2. In contrast to other worklets, they do not scale via transaction rate, but instead scale with memory capacity. In addition, they do not use throughput as their performance metric, but they modify it to include bandwidth and/or capacity.

1. **Flood**: A sequential memory bandwidth test that exercises memory using arithmetic operations and copy instructions. Flood is multi-threaded to reward servers that can utilize more memory concurrently with multiple CPUs and DIMMs. It automatically adjusts to use all available system RAM. It runs at two load levels called "Full" and "Half," utilizing all and half of the system memory, respectively. Flood's performance score is a function of both the memory capacity and the bandwidth measured during testing.
2. **Capacity**: A memory capacity test that performs XML operations on a minimum and maximum dataset. Capacity scales with capacity over its load levels. If the worklet's memory set exceeds the amount of physically available memory, it incurs a performance penalty for each transaction that attempts to read data not stored within physical memory. The final metric is a function of transaction rate and physical memory size including performance penalties.

11.3.1.3 Storage Workload

The developers of the SERT suite have included a workload for testing storage in order to enable a well-rounded system test. Storage worklets test the server's internal storage devices.

1. **Random**: Reads and writes data to/from random file locations.
2. **Sequential**: Reads and writes data to/from file locations picked sequentially.

11.3.1.4 Idle Workload

Idle keeps the system in an idle state in order to measure the idle power consumption. It does not measure any efficiency metric (only consumption).

11.3.2 Energy-Efficiency Metrics

SERT calculates separate intermediate energy-efficiency metrics, where each step aggregates the efficiency of the previous step. The calculation is illustrated in Figure 11.3 and consists of the following intermediate metrics:

1. Interval efficiency,
2. Worklet efficiency (over all load levels),
3. Workload efficiency (for all worklets),
4. Total efficiency.

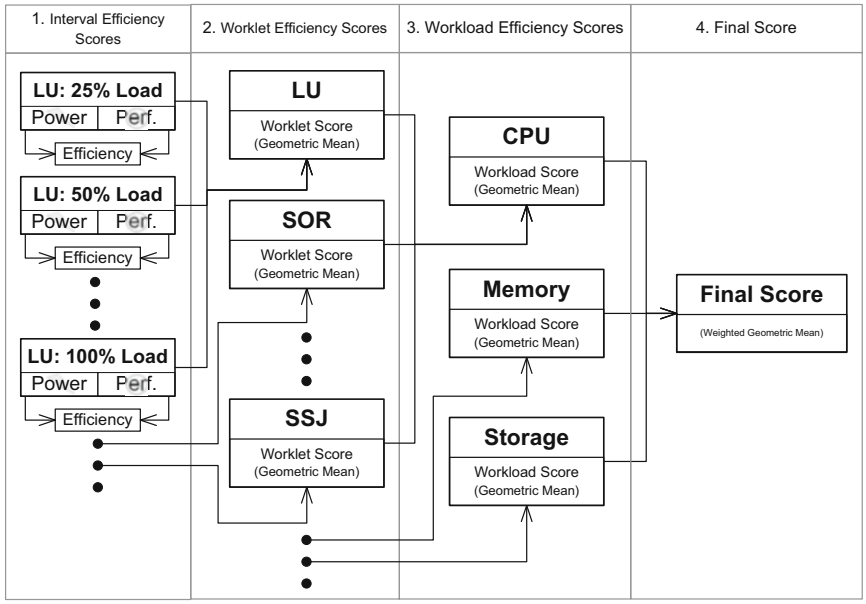

Fig. 11.3: Calculation of energy-efficiency metrics (Kistowski, Lange, et al., 2019)

11.3.2.1 Interval Energy Efficiency

Energy efficiency is calculated separately for each interval using the metric described in Section 11.1.4. As a small change, interval efficiency is multiplied by a constant factor of 1000—Equation (11.6). This is a cosmetic factor used to move the resulting score into a number range easier to read for a human reader.

$$\text{interval\_efficiency} = \frac{\text{normalized\_throughput}}{\text{power\_consumption}} \left[\frac{1}{J}\right] \times 1000 \tag{11.6}$$

11.3.2.2 Worklet Energy Efficiency

The worklet energy-efficiency score is calculated using the geometric mean of each worklet's separate interval scores as follows:

$$\text{worklet\_efficiency} = \left(\prod_{i=1}^{n} (\text{interval\_efficiency}_i) \right)^{\frac{1}{n}}, \tag{11.7}$$

where n represents the number of load levels per worklet, and $\text{interval\_efficiency}_i$ represents the energy efficiency for load level i.

SERT uses the geometric mean over the arithmetic mean as it is known to preserve ratios (such as energy efficiency and the normalized throughput, see Chapter 3, Section 3.5.3.2).

11.3.2.3 Workload Energy Efficiency

The workload energy-efficiency score is calculated by aggregating the efficiency scores of all worklets within the workload using the geometric mean as follows:

$$\text{workload\_efficiency} = \left(\prod_{i=1}^{n} (\text{worklet\_efficiency}_i) \right)^{\frac{1}{n}}, \tag{11.8}$$

where n represents the number of worklets per workload, and $\text{worklet\_efficiency}_i$ is the energy efficiency for each specific worklet, calculated using Equation (11.7).

11.3.2.4 Final Aggregate Energy Efficiency

The server energy-efficiency score is the final aggregate of the workload scores. It is also derived using the geometric mean. In contrast to the other geometric mean aggregates, the final score does not consider all workloads equally. Instead, it uses a weighted mean, putting a different focus on each of the workload scores. For specific use cases, weights may be chosen according to the use case. The U.S. EPA has adopted SERT v2.0 for regulatory purposes. They use the following workload weights:

- **High** CPU weight: 65%,
- **Medium** Memory weight: 30%,
- **Low** Storage weight: 5%.

With these weights, the final score would be calculated as follows:

$$\begin{aligned}\text{server\_efficiency} = exp(&0.65 \times ln(\text{CPU\_workload\_efficiency}) \\ &+ 0.3 \times ln(\text{memory\_workload\_efficiency}) \\ &+ 0.05 \times ln(\text{storage\_workload\_efficiency})).\end{aligned} \tag{11.9}$$

This specific weighting is targeted at regular data center compute nodes, resulting in a high CPU and medium memory weight that is intended to mirror a typical real-world compute workload's resource profile. Storage is weighted with a low 5% weight, as storage servers are not the target devices for this weighting.

11.4 Concluding Remarks

This chapter introduced the SPEC Power and Performance Benchmark Methodology, which is designed to facilitate the measurement and rating of server energy efficiency. Taking into account that servers are usually not fully utilized, the methodology is built to ensure workload execution at multiple load levels to obtain a thorough and relevant view of the server in question. The chapter described two implementations of the methodology in detail: SPECpower_ssj2008 and the SPEC SERT suite.

SPECpower_ssj2008, the first benchmark to implement this methodology, has been successfully applied to measure the energy efficiency of servers since its release in 2007, and it has driven the development of new, energy-efficient servers since then. In contrast, the SERT suite is not a benchmark, but a rating tool intended for use by regulatory programs such as the U.S. EPA Energy Star Version 3.0 Enterprise Servers Program,[2] and the EU Commission Regulation 2019/424.[3] Also, the International Organization for Standardization (ISO), in collaboration with the International Electrotechnical Commission (EIC), adopted the SERT suite in their server energy standard (ISO/IEC 21836),[4] which will foster the usage of the SERT suite globally. SERT implements the methodology, running multiple executable mini-workloads, each at multiple load levels. It aggregates its partial results in a single metric using multiple intermediate geometric means.

The SPEC Power and Performance Benchmark Methodology has had a significant impact on the development of new benchmarks and on the energy efficiency of servers. It has been applied in many benchmarks, including some TPC and VMware benchmarks, and it has helped drive and measure a significant improvement in the energy efficiency of servers since 2007.

Energy-efficient servers are one part of the combination necessary to provide services. The second part, the software itself, can still be wasting energy, either through deficient configuration, suboptimal deployment, unnecessary computations, or a combination of those. Several initiatives are underway to extend the scope of server efficiency, for example, the SPEC Power Research Working Group started

[2] https://www.energystar.gov/products/data_center_equipment/enterprise_servers

[3] https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32019R0424&from=EN

[4] https://www.iso.org/standard/71926.html

to identify what programming languages are more sensitive to compiler optimizations (Schmitt et al., 2020), and ISO/IEC is drafting a new standard energy efficiency of middleware (ISO/IEC JTC 1/SC 39, 2020). These will help to consider how to benchmark and rate the energy efficiency of software in standardized benchmarks.

References

Barroso, L. A. and Hölzle, U. (2007). "The Case for Energy-Proportional Computing". *Computer*, 40(12). IEEE Computer Society: Washington, DC, USA, pp. 33–37 (cited on pp. 251, 254).

Glanz, J. (2012). *Power, Pollution and the Internet*. New York Times, Sept. 22, 2012. New York, USA (cited on p. 251).

ISO/IEC JTC 1/SC 39 (2020). *ISO/IEC DIS 23544: Information Technology—Data Centres—Application Platform Energy Effectiveness (APEE)*. https://www.iso.org/standard/76000.html, last accessed July 2020. Geneva, Switzerland (cited on p. 264).

Kistowski, J. von, Beckett, J., Lange, K.-D., Block, H., Arnold, J. A., and Kounev, S. (2015). "Energy Efficiency of Hierarchical Server Load Distribution Strategies". In: *Proceedings of the 23rd IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2015)*. (Atlanta, GA, USA). IEEE Computer Society: Washington, DC, USA (cited on p. 255).

Kistowski, J. von, Lange, K.-D., Arnold, J. A., Beckett, J., Block, H., Tricker, M., Sharma, S., Pais, J., and Kounev, S. (2019). "Measuring and Rating the Energy-Efficiency of Servers". *Future Generation Computer Systems*, 100. Elsevier Science: Amsterdam, The Netherlands, pp. 579–589 (cited on p. 261).

Lange, K.-D. (2009). "Identifying Shades of Green: The SPECpower Benchmarks". *Computer*, 42(3). IEEE: Piscataway, NJ, USA, pp. 95–97 (cited on p. 252).

Lange, K.-D. and Tricker, M. G. (2011). "The Design and Development of the Server Efficiency Rating Tool (SERT)". In: *Proceedings of the 2nd ACM/SPEC International Conference on Performance Engineering (ICPE 2011)*. (Karlsruhe, Germany). ACM: New York, NY, USA, pp. 145–150 (cited on pp. 251, 252).

Poess, M., Nambiar, R. O., Vaid, K., Stephens, J. M., Huppler, K., and Haines, E. (2010). "Energy Benchmarks: A Detailed Analysis". In: *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking (e-Energy 2010)*. (Passau, Germany). ACM: New York, NY, USA, pp. 131–140 (cited on pp. 254, 258).

Rivoire, S., Shah, M. A., Ranganathan, P., and Kozyrakis, C. (2007). "JouleSort: A Balanced Energy-Efficiency Benchmark". In: *Proceedings of the 2007 ACM*

SIGMOD International Conference on Management of Data. (Beijing, China). ACM: New York, NY, USA, pp. 365–376 (cited on pp. 251, 254, 258).

Schmitt, N., Bucek, J., Lange, K.-D., and Kounev, S. (2020). "Energy Efficiency Analysis of Compiler Optimizations on the SPEC CPU 2017 Benchmark Suite". In: *Companion of the 11th ACM/SPEC International Conference on Performance Engineering (ICPE 2020)*. (Edmonton, Canada). ACM: New York, NY, USA, pp. 38–41 (cited on p. 264).

SPECpower Committee (2014). *Power and Performance Benchmark Methodology V2.2*. Gainesville, VA, USA: Standard Performance Evaluation Corporation (SPEC) (cited on p. 252).

– (2017a). *Chauffeur Worklet Development Kit (WDK) User Guide 2.0.0*. Gainesville, VA, USA (cited on p. 252).

– (2017b). *Server Efficiency Rating Tool (SERT) Design Document 2.0.0*. Gainesville, VA, USA (cited on p. 259).

Welch, T. A. (1984). "A Technique for High-Performance Data Compression". *Computer*, 17(6). IEEE Computer Society: Los Alamitos, CA, USA, pp. 8–19 (cited on p. 259).