International Journal of Multidisciplinary Research and Publications
ISSN (Online): 2581-6187

VM Power Consumption Estimation as Part of SCI Computation

Weerachai Sarakun[1], Yachai Limpiyakorn[2]
[1, 2]Department of Computer Engineering, Chulalongkorn University, Bangkok 10330, Thailand
Email address: 6672098121@student.chula.ac.th[1], yachai.l@chula.ac.th[2]

***Abstract*—** *The growing demand for computational resources and IT technologies, including cloud services, data centers, and virtualization systems like VMware, has significantly increased energy consumption and environmental concerns. This research demonstrates a methodology for computing operational emissions, or carbon emissions due to energy consumption, as part of the Software Carbon Intensity (SCI) score. The method primarily focuses on the utilization of hardware resources, including CPU, memory, in virtual machine (VM) environments. With today technology of hypervisors, multiple virtual machines can be created on a single host machine. Each virtual machine has its own operating system and hardware resources. The study thus presents an energy consumption estimation model for VMs and resource-based power distribution method. Power consumption is estimated using boundary conditions, idle states, and trapezoidal integration. The resulting total energy consumption multiplied by the emission factor of electricity will contribute the value of operational emissions. The proposed framework would be considered as a preliminary step useful for data center operators and organizations aiming to achieve sustainable IT practices while maintaining operational efficiency.*

***Keywords*—** *Operational emissions: software carbon intensity: time-weighted average: VMware: sustainability.*

I. INTRODUCTION

The rapid expansion of computational and IT resource demands has led to widespread adoption of cloud services, data centers, and virtualization technologies such as VMware. While these technologies enhance resource allocation efficiency and reduce operational costs, they have also raised critical concerns about energy consumption and environmental impact. Large-scale data centers contribute significantly to energy usage and carbon emissions, necessitating more sustainable and efficient practices. A key metric for evaluating the environmental impact of software and IT operations is Software Carbon Intensity (SCI), which measures the carbon emissions resulting from hardware resource usage, such as CPU and memory utilization. These emissions directly influence the carbon footprint of organizations and data centers, making SCI a critical consideration for promoting environmentally friendly operations. This research demonstrates a methodology for calculating operational emissions as part of the SCI score. Operational emissions involve computing carbon emissions due to energy consumption. The VMware environment is selected for demonstrating the calculation using the proposed methodology. At this stage, we primarily focus on the utilization of hardware resources, namely CPU and memory.

In literature, several strategies and best practices have been presented to enhance energy and resource efficiency, reduce carbon emissions, and foster sustainable resource management. Building upon prior research, such as the development of energy estimation models for VMs and resource-based power distribution methods, this study addresses the challenges of achieving practical energy measurements and efficient power consumption allocation in virtualization environments. In 2009, Kansal et al. [1] presented a solution for VM power metering to attack the difficulty that virtual machine power cannot be measured purely in hardware. The authors used resource utilization data to estimate VM energy consumption. The approach was promising and can be implemented on current virtualized platforms without adding any additional hardware or software instrumentation. In 2020, Gul et al. [2] presented energy-aware VM consolidation schemes that considered a server capacity in terms of CPU and RAM to reduce overall energy consumption. The study addressed the impact of RAM, which consumes about 25% of the joint energy of a server's CPU and RAM, on energy efficiency in VM consolidation. In 2021, Davy [3] studied on AWS EC2 power consumption estimation that employs techniques like PUE analysis and resource-based power distribution. By leveraging insights from these studies, the research aims to provide practical solutions for optimizing SCI and advancing sustainability in IT operations. In 2023, Choudhury et al. [4] proposed an energy-efficient VM consolidation approach that considered weighted summation of CPU and memory utilization. The method tried to reduce energy consumption by optimizing VM placement based on resource utilization metrics. The study introduced a symbiotic association between hosts and VMs, analogous to biological interactions, to enhance energy efficiency. Pamadi et al. [5] presented the recent advancements in dynamic resource allocation, energy efficiency, virtual machine migration, adaptive resource management, and load balancing strategies. The authors highlighted the role of reinforcement learning in optimizing resource allocation, reporting a 25% improvement in resource utilization through real-time decision-making. Additionally, the authors discussed energy-efficient resource management techniques by emphasizing the balance between performance and energy efficiency that would reduce energy consumption up to 30%.

II. BACKGROUND

A. Software Carbon Intensity

Software Carbon Intensity (SCI) [6] is a standardized methodology for calculating carbon emissions associated with a software system by tracking its hardware and electricity

Weerachai Sarakun, and Yachai Limpiyakorn, "VM Power Consumption Estimation as Part of SCI Computation," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 7, Issue 8, pp. 129-133, 2025.

demands. It aims to guide users and developers in making informed choices about tools, architectures, and services to optimize carbon efficiency. SCI can be computed for a variety of applications, ranging from large-scale cloud systems to small open-source libraries. Developers play a key role in reducing SCI scores during all stages of software development, including writing energy-efficient code, optimizing database designs, and adopting carbon-aware build pipelines. A lower SCI score indicates better carbon efficiency, while achieving a zero SCI score is generally not feasible. The SCI calculation process involves four key steps:

- Defining the Software Boundary− identifying the software components, including applications, servers, hardware, and supporting infrastructure, that contribute to SCI.
- Determining Functional Units− selecting an appropriate functional unit, such as the number of users, API calls, or machine learning model training instances, for measuring carbon emissions.
- Selecting Carbon Emission Calculation Methods− employing suitable methods to measure carbon emissions for each software component, either through real-world data or laboratory simulations.
- Calculating and Aggregating SCI Scores− computing the carbon emissions for each software component and aggregating these values to determine the overall SCI score. The SCI formula for each component is expressed as in (1).

$$SCI = \frac{O+M}{R} \quad (1)$$

where

O = operational emissions compute carbon emissions due to energy consumption, as in (2).

M = embodied emissions measure carbon emissions embedded in hardware during production and disposal, as in (3).

R = functional unit, such as per user or per API call

$$O = E \times I \quad (2)$$

where

E = energy consumption of the hardware in kilowatt-hours (kWh), and

I = carbon intensity of the energy in the specific region, measured in gCO_2eq/ kWh

$$M = TE \times TS \times RS \quad (3)$$

where

TE = total carbon emissions over the hardware's lifecycle,

TS = fraction of time the hardware is used by the software, and

RS = proportion of hardware resources utilized by the software e.g. allocated CPU cores

SCI scores are then aggregated across all components using a common functional unit to provide a comprehensive SCI metric.

B. Time-Weighted Average

The Time-Weighted Average (TWA) [7] is a statistical method used for analyzing time-series data where measurements are taken at irregular intervals. The approach derives a more accurate average by weighing each data point according to the duration of its corresponding time interval, ensuring that longer intervals contribute proportionally to the overall average. This method is often used in scenarios such as IoT systems, remote sensing, and battery testing.

The key idea behind TWA is to calculate the area under the curve of the time-series data, which represents the cumulative values over time. This area is divided by the total duration of the time period under consideration, yielding a weighted average that reflects both the values of the data points and the length of time they persist. By using this approach, TWA avoids the inaccuracies that arise when treating all data points equally, regardless of their temporal duration.

Another important aspect that complements TWA is the Last Observation Carried Forward (LOCF/B) [8], which is a technique for handling missing data in time-series analysis, where the most recent available value is carried forward to replace missing data points. This method helps ensure that gaps in the data do not skew the results and is particularly useful in cases where measurements are sparse or irregular. Accompanied by TWA, LOCF/B can improve the continuity and accuracy of the data analysis by filling in missing intervals with relevant values, ensuring that the area under the curve is correctly computed.

While TWA provides a more accurate representation of data through its focus on the area under the curve and its ability to handle irregular time intervals, LOCF/B ensures continuity when data gaps exist, enhancing the reliability of the TWA calculation in the contexts like energy consumption estimation and resource utilization in systems such as VMware environments.

III. Methodology

Despite the lack of direct energy consumption data for individual VMs, trends in energy usage can be inferred from resource utilization metrics to compute operational emissions as part of the SCI score. The preliminary study is carried out to demonstrate how to estimate energy consumption in VMware, simply focusing on resource utilization of CPU and memory.

A. Data Gathering

The data of CPU and memory usage were collected from the VMware vSphere system, including allocated resource reservations and host energy consumption. The data were collected daily between 00:00:00 and 23:59:59, starting from October 22, 2024 to October 29, 2024. The data captured logs were packed in Parquet format, and separated into an individual file per day.

B. Power Consumption Estimation

Using the daily log data, the computation starts from dividing power consumption into three components:

- *Idle state*− The energy consumption during idle state is approximated by the minimum power consumption observed on a daily basis. It is assumed that there are off-peak periods during the day when the VM operates at minimal capacity or remains idle. During these times, energy consumption is expected to be at its lowest. Therefore, the minimum recorded power consumption, Min(Power Consumption), is used during the idle state.

Weerachai Sarakun, and Yachai Limpiyakorn, "VM Power Consumption Estimation as Part of SCI Computation," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 7, Issue 8, pp. 129-133, 2025.

- *Beyond boundary of gathered data*− Since the time frame of 24-hour is adopted, the data will be collected within a 24-hour period. However, when gathering data, it is common to encounter issues such as missing values at the exact start (00:00:00) or end (23:59:59) of the day. To address this issue, the LOCF/B (Last Observation Carried Forward/ Backward) method is applied to fill in the missing data points. For example, let (T_0, P_0) and (T_{24}, P_{24}) denote the power consumption P_0 and P_{24} at time 00:00:00 and 23:59:59, respectively. Suppose the starting time data available for gathering at (T_5, P_5), and ending at (T_{17}, P_{17}), we then substitute P_0 and P_{24} with P_5 and P_{17}, respectively. Inferring missing data by LOCF/B is reasonable. The missing values normally occurring at the start and the end of the 24-hr time frame could be considered a short period around 0-5 minutes that the levels of power usage would not significantly differ. Therefore, rather than filling the missing value with zero, the alternative of LOCF/B is preferable.
- *Within boundary of gathered data*− For the intervals of available data, the computation with linear weighting is used to account for variations in energy usage over time. The method of trapezoidal integration is applied to estimate the energy consumed during each interval. The area under the power curve (area of Isosceles Trapezoid) is computed as the sum of a rectangular area and a triangular area, as in (4).

$$E = A_{rect} + A_{tri} \quad (4)$$

where

E = total energy consumption

A_{rect} = rectangle area, determined by the minimum power consumption (P_{min}) and the time interval width (Δt)

A_{tri} = triangle area, determined by the difference in power consumption (P_{diff}) and the time interval width (Δt)

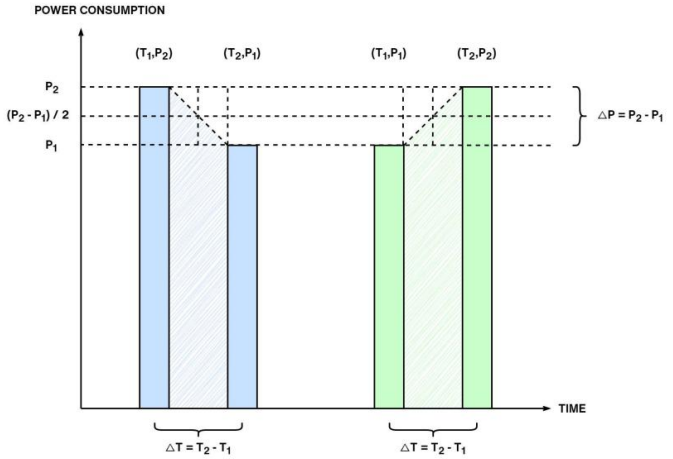

Fig. 1. Trapezoidal integration for energy usage estimation.

Fig. 1 illustrates the diagram of trapezoidal integration applied for estimating energy consumption of gathered data within a certain time interval. The notion can be aggregated for the entire day to compute the total energy consumption.

For example, given the data of power usage at 06:00:00 and 09:00:00 represented by (T_6, P_6) and (T_9, P_9), respectively, the energy consumption for this interval is calculated as in (5) and (6).

$$A_{rect} = Min\,(P_6, P_9) \times (T_9 - T_6) \quad (5)$$

$$A_{tri} = |P_9 - P_6| \times \frac{(T_9 - T_6)}{2} \quad (6)$$

The fact that either $P_1 < P_2$ or $P_1 > P_2$, the summation of power usages denoted by the rectangle and triangle areas can be simplified as following.

$$E = P_1 \times (T_2 - T_1) + \frac{(P_2 - P_1) \times (T_2 - T_1)}{2}$$

$$= P_1T_2 - P_1T_1 + \frac{(P_2T_2 - P_2T_1 - P_1T_2 + P_1T_1)}{2}$$

$$= P_1T_2 - P_1T_1 + \frac{P_2T_2}{2} - \frac{P_2T_1}{2} - \frac{P_1T_2}{2} + \frac{P_1T_1}{2}$$

$$= \frac{P_1T_2}{2} - \frac{P_1T_1}{2} + \frac{P_2T_2}{2} - \frac{P_2T_1}{2}$$

$$= \frac{P_1}{2} \times (T_2 - T_1) + \frac{P_2}{2} \times (T_2 - T_1)$$

$$= \frac{(P_1 + P_2)}{2} \times (T2 - T1)$$

The power usage of each interval is then simplified as in (7).

$$E = \frac{P_2 + P_1}{2} \times (T_2 - T_1) \quad (7)$$

The daily energy consumption can then be aggregated from each interval as in (8).

$$E = \sum_{k=0}^{n-1} \frac{P_k + P_{k+1}}{2} \times (T_{k+1} - T_k) \quad (8)$$

C. Demonstration

The daily energy consumption for each VM is calculated using trapezoidal integration for each time interval. An example is shown in Fig. 2, demonstrating the process of calculating power consumption within a 24-hour timeframe using sample data points. The day is divided into 24 time points, with data collected at 7 specific points. After filling in the missing data at time 0 and 24, as shown in Table I, the trapezoidal integration formula (8) is applied to calculate the energy consumption for each interval. The results of these calculations are presented in Table II.

TABLE I. Sample and filled data for 24-Hour period.

Time (hr)	Energy (W)
0	*275*
5	275
6	300
9	245
10	260
11	200
15	220
17	245
24	*245*

TABLE II. Power consumption calculation using Trapezoidal integration.

Time (hr)	Power (Watt)	Interval	Formula	Energy (Watt)
0	*275*	*0-5*	*275 x (5 - 0)*	*1375*
5	275	5-6	(275+300)/2 x (6 - 5)	287.5
6	300	6-9	(300+245)/2 x (9 - 6)	817.5
9	245	9-10	(245+260)/2 x (10 - 9)	252.5
10	260	10-11	(260+200)/2 x (11 - 10)	230
11	200	11-15	(200+220)/2 x (15 - 11)	840
15	220	15-17	(220+245)/2 x (17 - 15)	465
17	245	17-24	245 x (24 - 17)	1715
24	245			
Total				**5982.5**

Weerachai Sarakun, and Yachai Limpiyakorn, "VM Power Consumption Estimation as Part of SCI Computation," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 7, Issue 8, pp. 129-133, 2025.

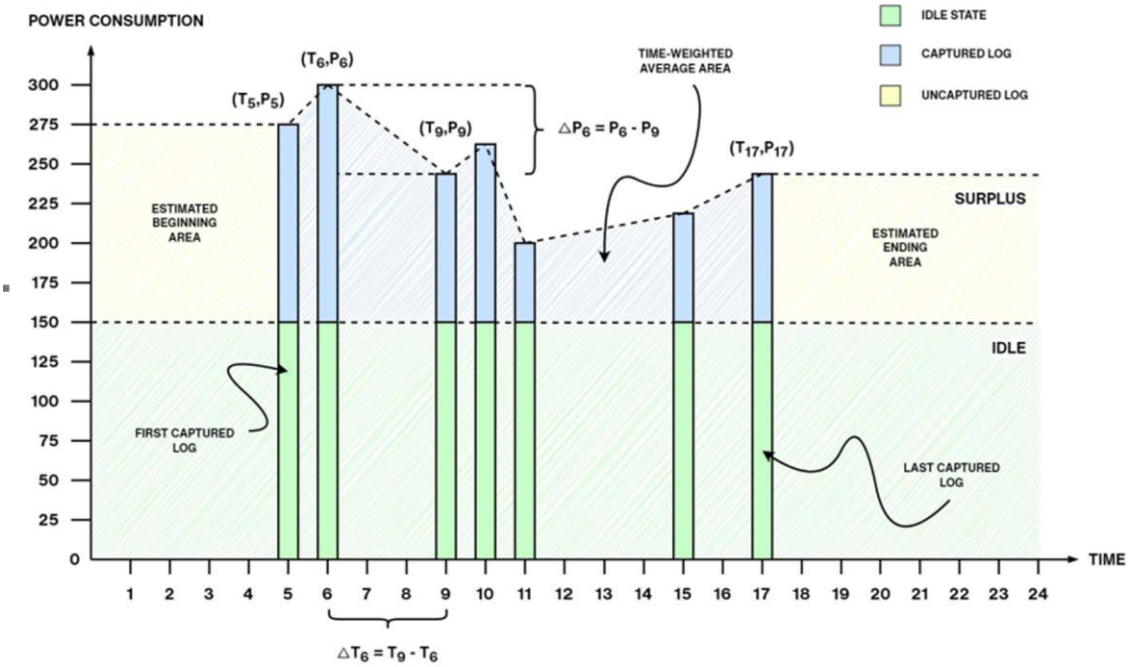

Fig. 2. Trapezoidal integration for computing power consumption in 24hr-time frame.

D. VM Power Consumption Allocation

With respect to Hypervisor technology behind virtualization or decoupling of hardware from software, IT administrators can create multiple virtual machines on a single host machine. When calculating the power consumption of individual Virtual Machines within a host, it is essential to allocate the total power consumed by the host proportionally among the VMs based on their resource utilization. The notion ensures that each VM is responsible for its share of the power used, accounting for both idle and active states of the host:

- *Idle power consumption* represents the power consumed when the host is in an idle state, with none or minimal activity from the VMs. The power is evenly distributed among all VMs, regardless of their activity levels, to reflect the shared responsibility of maintaining the host's baseline operation.
- *Surplus power consumption* represents the power consumed beyond the idle state, caused by the active utilization of resources by the VMs. This surplus power is allocated based on the proportional utilization of CPU and memory by each VM.

E. Resource Utilization Ratio

The hypervisor allocates the underlying physical computing resources such as CPU and memory to individual virtual machines as required. To allocate surplus power fairly, a resource utilization ratio is calculated for each VM. This ratio is derived from the VM's share of total CPU and memory usage as in (9).

$$Ratio = 0.5 \times \left(\frac{U_{cpu}}{\Sigma H_{cpu}} \right) + 0.5 \times \left(\frac{U_{mem}}{\Sigma H_{mem}} \right) \quad (9)$$

where

U_{cpu} = CPU resources allocated to the VM,

U_{mem} = memory resources allocated to the VM,

ΣH_{cpu} = Sum of CPU allocations for all VMs on the host,

ΣH_{mem} = Sum of memory allocations for all VMs on the host.

The formula weights CPU and memory utilization equally (50% each). Adjustments can be made to the weights depending on the importance of these resources in specific scenarios.

Suppose the host machine consists of three VMs, each of which is allocated the resources as below:

- VM1 with CPU=100, Memory=50
- VM2 with CPU=200, Memory=250
- VM3 with CPU=300, Memory=150

The total CPU and memory allocations are 600 and 450, respectively. The ratio of resource utilization of each VM can be computed as shown in Table III.

TABLE III. Summary of the ratios of resource utilization of each VM.

VM	CPU	Memory	Total CPU	Total Memory	CPU Utilization (%)	Memory Utilization (%)	Ratio (%)
VM1	100	50	600	450	16.67	11.11	13.89
VM2	200	250	600	450	33.33	55.56	44.45
VM3	300	150	600	450	50	33.33	41.66

The allocation of power is divided into two categories: idle power and surplus power. Idle power is evenly distributed among all VMs, as every VM shares the responsibility for maintaining the host's baselined operations, no matter active or idle status. It is assumed that even inactive, the VMs contribute to maintaining the host's operational requirements. Surplus power, on the other hand, is allocated based on actual resource utilization, such as CPU and memory usage due to active operations. Each VM's resource utilization reflects individual activity level, ensuring a fair, usage-dependent distribution.

Weerachai Sarakun, and Yachai Limpiyakorn, "VM Power Consumption Estimation as Part of SCI Computation," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 7, Issue 8, pp. 129-133, 2025.

Suppose the host consumes 400 watts, dividing into 300 watts consumed in the idle state and 100 watts as surplus energy beyond the idle state. The energy reported in the idle state will be evenly distributed across each VM, while the surplus energy will be allocated based on the utilization ratios from Table III. The power consumption allocated for each VM are summarized in Table IV.

TABLE IV. Summary of power consumption allocation for each VM.

VM	Ratio (%)	Idle Power (Watt)	Surplus Power (Watt)	Total Power (Watt)
VM1	13.89	100	13.89	113.89
VM2	44.45	100	44.45	144.45
VM3	41.66	100	41.66	141.66
Total	*100*	*300*	*100*	*400*

Fig. 3 illustrates the power consumption of a host ranging from October 22 to October 29, 2024. The chart daily reported the statistical values including min, max, and average.

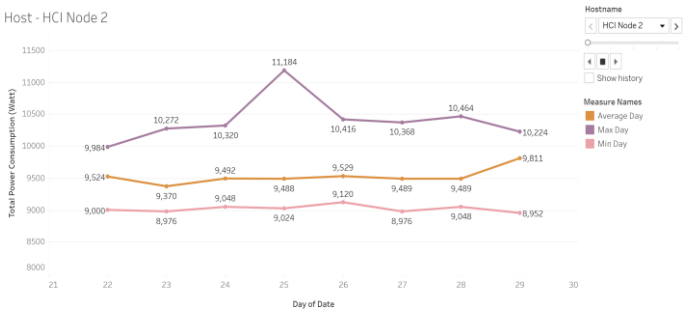

Fig. 3. Visualization of power consumption of a host during 8 days.

F. Operational Emissions

Once the VM energy consumption has been estimated, the Operational emissions (O), part of SCI score, can be calculated. Suppose the energy consumption of a VM in a day is 10 kWh, and the Emission Factor (EF) of Electricity, grid mix [9] is 0.499 $kgCO_2/kWh$. The derived value of Operational emissions equals $10 \times 0.499 = 4.99$ $kgCO_2$.

IV. CONCLUSION

This research presents a method for estimating Operational emissions and distributing host-level power consumption across individual VMs based on weighted utilization factors. In the early stage, the work has simply focused on the resource utilization of CPU and memory. The proposed method seems promising for approximating energy consumption in VMware environments. The full-blown stage of study would support sustainability goals by optimizing energy usage and reducing the carbon footprint in virtualized infrastructures.

REFERENCES

[1] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A Bhattacharya,"Joulemeter: Virtual Machine Power Measurement and Management," Microsoft Research , University of Southern California and IIT Kharagpur, 2009.

[2] B. Gul, I. A. Khan, S. Mustafa, O. Khalid, S. S. Hussain, D. Dancey, and R. Nawaz, "CPU and RAM Energy-Based SLA-Aware Workload Consolidation Techniques for Clouds," in IEEE Access, vol. 8, pp. 62990-63003, 2020, doi: 10.1109/ACCESS.2020.2985234

[3] B. Davy, "Estimating AWS EC2 Instances Power Consumption," [Online] Available: https://medium.com/teads-engineering/estimating-aws-ec2-instances-power-consumption-c9745e347959/. [Accessed: 14 Dec 2024].

[4] A. Choudhury, K. K. Nath, M. Ghose and Y. Thakran, "Memory and CPU utilization-aware Energy-Efficient VM Placement and Consolidation in Cloud Data Centers," 2023 IEEE Guwahati Subsection Conference (GCON), Guwahati, India, 2023, pp. 1-6, doi: 10.1109/GCON58516.2023.10183444.

[5] E. R. V. Pamadi, S. Goel, and P. K. G. Pandian, "Effective Resource Management in Virtualized Environments," Journal of Emerging Trends and Novel Research, ISSN:2984-9276, vol.1, issue 7, pages 1- 10, 2023.

[6] Green Software Foundation, "Software Carbon Intensity (SCI) Specification," [Online] Available: https://sci.greensoftware.foundation/. [Accessed: 14 Dec 2024].

[7] D Kohn, "What time-weighted averages are and why you should care," [Online] Available: https://dev.to/timescale/what-time-weighted-averages-are-and-why-you-should-care-1ge6/. [Accessed: 14 Dec 2024]

[8] O'Connor and Alec B., "LOCF Approach to Handling Missing Data Overestimates the Pain Score Improvement of Drop-Outs". The Journal of Pain, vol. 11, issue 5, pp 500 – 501, 2010.

[9] Thailand Greenhouse Gas Management Organization (TGO), " Emission Factor (CFO)," [Online] Available: https://thaicarbonlabel.tgo.or.th/index.php?lang=TH&mod=YjNKbllXNXBlbUYwYVc5dVgyVnRhWE56YVc5dQ/ [Accessed: 14 Dec 2024].