Data Analysis and Green Computing: Profiling HPC Power and Tracking CO2 Emissions

Green Team Report

Supervisor: Mohsen Seyedkazemi Ardebili

April 2024

Contents

Abstract

The following report includes the observations related to our work, the goal of which is profiling the HPC (High Performance Computing) power consumption and CO2 emissions in the scope of Laboratory of Big Data course.

Contributors:
Davletiyarov Baibek
Liberatoscioli Martina
Squarzoni Lorenzo
Zavaroni Sofia

1 Introduction

Sustainable computing is increasingly vital in our digital era, especially within HPC systems. To achieve this goal, it is crucial to utilize big data analysis of High-Performance Computing (HPC) monitoring data. This work is focused on the two important aspects of it: power profiling, and tracking, predicting CO_2 emissions in the scope of the Laboratory of Big Data Architectures course. As a result of the data analysis the power and energy consumption optimization techniques were suggested, as well as the predictive model of CO_2 emissions was tested with achieving only 10.31% error rate.

1.1 CINECA MARCONI100

MARCONI is the Tier-0 system, co-designed by CINECA and based on the Lenovo NeXtScale platform, that replaced the former IBM BG/Q system (FERMI) in June 2016. MARCONI100 is an upgrade of the "not conventional" partition of the Marconi Tier-0 system. It is an accelerated cluster based on Power9 chips and Volta NVIDIA GPUs, acquired by Cineca within the PPI4HPC European initiative. In total it presents 55 racks, 49 of which used for computational purposes [**Marconi100Site**], for a total of 980 nodes, plus 8 login nodes, able to reach a peak performance of about 32 PFlops. Each rack contains a total of 20 vertically stacked nodes.

1.2 M100 ExaData

All data related to the data center has been collected taking advantage of the ExaMon framework, as described in the related paper [**Marconi100**], then processed to remove redundant and sensitive information and finally transformed into a partitioned Parquet dataset. Among the metrics obtained from HW sensors, there are: the CPU load of all the cores in the supercomputing nodes, CPU clock, instructions per second, memory accesses (bytes written and read), fan speed, the temperature of the room hosting the system racks, power consumption (at different levels), etc. This work mainly focuses on the data related to power utilization and temperature readings.

1.3 Carbon Intensity Data

The data on Carbon Intensity is provided by ElectricityMaps, which provides carbon intensity data coming from electricity consumption for more than 200 zones in the world. The Carbon Intensity metric measures the amount of how "clean" the electricity in a given zone at a given time, representing the grams of carbon dioxide (CO_2) in the atmosphere released per kilowatt hour (kWh). Particularly, for this work the Carbon Intensity data for the North Italy region was used. The dataset includes the carbon intensity of LCA and *direct* [**CarbonIntensity**] impacts, particularly, the LCA (Life-Cycle Assessment) data includes the "compiled" carbon intensity as related to all stages throughout the product's life-cycle: from raw material extraction to production, usage, and disposal, while direct is related to the direct emissions during operation. For our work, we use LCA+Direct metrics.

2 Tools

To begin with, introducing the tools and methods employed for data analysis:

2.1 Dataset Preparation

During the development of the project we dealt with more than one dataset, but all of them required similar manipulations: extracting specific periods, dropping unnecessary columns, or formatting values (eg. timestamps) for proper processing. The power data frame has been further modified: the initial months of data have been dropped and the dataset resampled on an hourly basis, resulting in mean power consumption samples for each hour; the presence of missing values (NaN) has been handled by applying linear interpolation both along rows and columns: in this way, it was possible to ensure that all missing values were filled in a way that maintains continuity in both temporal and spatial dimensions.

2.2 Time Series Prediction with Prophet by Meta

Prophet [4] is a powerful forecasting tool developed by Meta to handle time series data: it can define the seasonality patterns and predict the future behavior of time series. It uses an additive statistical model to perform the prediction in the form of

$$y(t) = g(t) + s(t) + h(t) + \epsilon$$

Precisely, it can be considered as a generalized additive model (GAM), which is a regression model with an application of non-linear smoothers [1], where the components are: g(t) - the trend, s(t) - seasonality, h(t) - holiday effect, and ϵ is an error. Particularly, the trend component g(t) is the piecewise linear trend of aperiodic changes in the time series. In contrast, the seasonal component s(t) is modeled with a Fourier Series to include the periodic patterns. Moreover, the holiday component h(t) is the effect of the holidays that can be also taken into consideration in describing the time series irregularities. The error term is also included to represent the noise and outliers in the data. The prediction result of Prophet also provides information about the confidence interval (uncertainty) of the data, which is related to the changes in the data trend. By default [5], it considers the future trend changes with the same average frequency and magnitude as observed in the previous data, and projects it further using the Laplace approximation to stimulate possible further trends, which will contribute to the range of the uncertainty interval. Generally, Prophet can detect the seasonality and periodicity trends in the data automatically, as well as provide flexibility in terms of the ability to set up and adjust the components manually.

2.3 Trend Detection with Seasonal-Trend Decomposition (STL)

The STL (Seasonal-Trend Decomposition) technique [2] employs several mathematical and statistical tools to describe seasonality, trend, and remainder components in time series data. It uses LOESS, or Local Regression, which is a non-parametric smoothing method used to estimate trend and seasonal components. The LOESS function locally fits data using polynomial regressions over moving windows, allowing for capturing both short-term and long-term fluctuations. The basic formula for a single smoothed observation y_i is:

$$\widehat{y}_i = \sum_{j=1,...n} w_{ij} * y_j$$

where w_{ij} are weights calculated based on the distance between x_i and x_j, often using a kernel function like the tri-cube function, which includes the width of the smoothing window.

3 Carbon Intensity Prediction

Understanding and predicting the carbon intensity is crucial in assessing the environmental impact and in the development of regulatory policies. Therefore, the usage of prediction of future carbon intensity is very crucial to mitigate its effects in peak periods. The distribution of the available data on carbon intensity (CI) is in the range of 3 years from January 2021 to January 2024 [3]. Analysis of the data demonstrates the seasonality, which makes the data generally predictable.

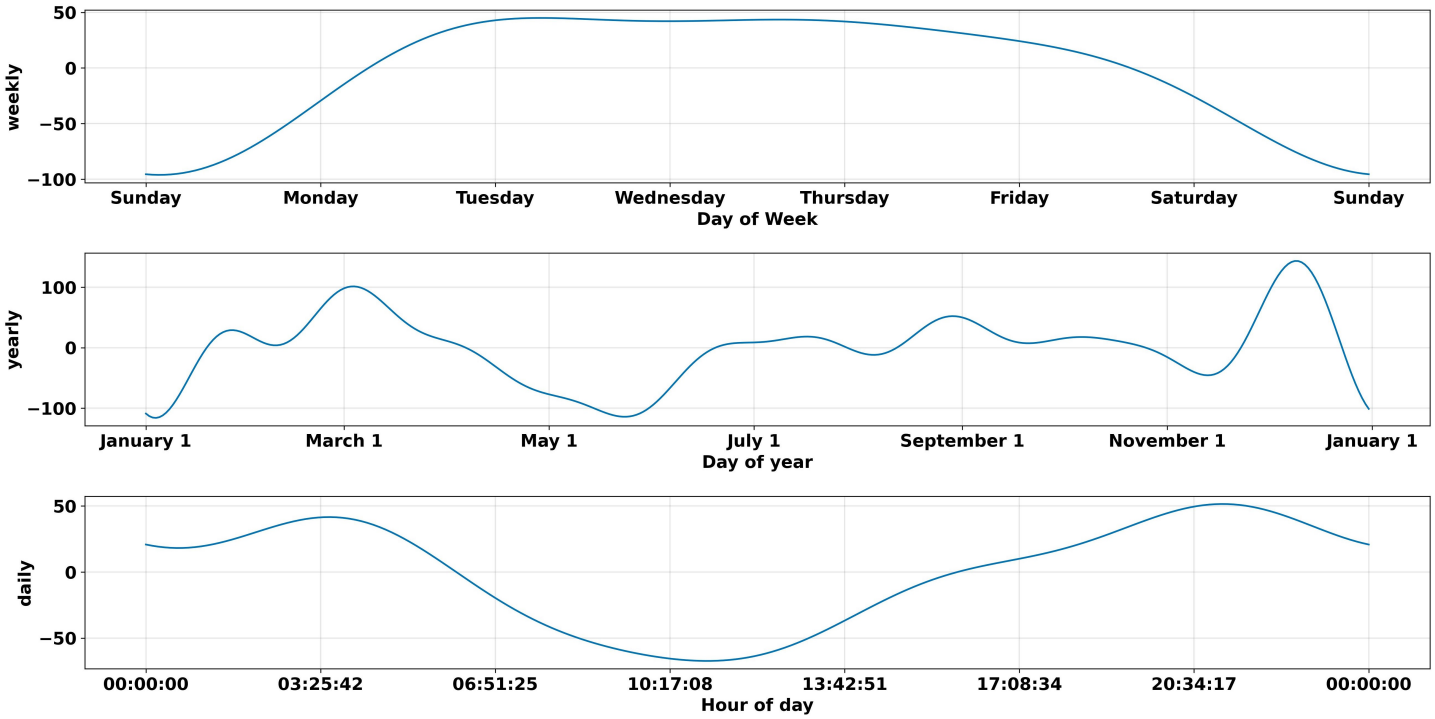

Figure 1: Carbon Intensity seasonality

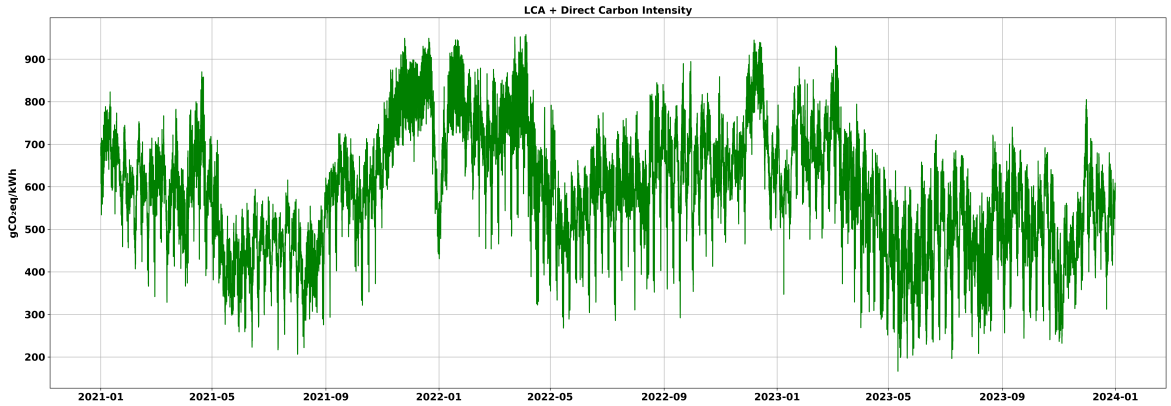

Figure 2: Carbon Intensity LCA + direct

As it can be seen from the visualization, the data has a periodicity which is useful in the prediction of future behavior. The periodicity depends on many factors, for example, the most impactful of the effects is the difference in CI as of summer (hotter periods) and winter (colder periods), as well as the variation during the week - working days demonstrate higher CI, compared to weekend. Moreover, throughout the day the carbon intensity is higher during business hours. Hence, by analyzing these patterns we can predict the future CI, and evaluate its accuracy.

3.1 Prediction Techniques of CI

The methodology for our prediction is based on a comparison of two models: the baseline last value predictor, and the machine learning (ML) model. The first, simple baseline model is the prediction based on the previous observed value: the next value is dependent on the last week's value at the same time and week of the day. The advantage of this prediction is that it is generally a lightweight solution requiring minimal computational resources, however, it has no extended parameter tuning options. The second, ML model is based on the Prophet time series predictor, which is a more complex solution compared to the first one. ML models can capture the patterns and relationships in data points, as well as adapt to the changes in data distribution, being more robust with dynamic data. Hence, using such a lightweight solution in comparison with a complex ML model allows us to compare the efficiency of the predictive ML model.
Our ML model is the Meta's Prophet time series predictor, which is a powerful tool for predicting the time series. To achieve the best performance, we trained the predictor on different periods of the dataset, particularly, from 30-month data to 10 months with a step of 2 months, which demonstrated the impact of historical data. Furthermore, we compared the performance on varying lengths of the forecasting horizon. Moreover, based on the prophet documentation, the available hyperparameters were tuned to find the best-performing configuration.

3.2 Prediction Result

Based on our prediction using the last value predictor, it has demonstrated an absolute error of 16.22% **??**. On the same dataset, our ML model has been trained on 130 different possible configurations of parameters, which resulted in the error rate in a wide range of 10% to 159%. Particularly, as a result of the tuning ML prediction, our key finding demonstrates that we could achieve the 10.31% absolute error relative to the real data with training on 16 month period and prediction horizon for 2 months with the parameters of $'changepoint_prior_scale' as 0.1, and'$

Generally, the usage of the prediction technique helps us to estimate future CI behavior to develop policies toward reducing it. In the context of heavy computational tasks, such models can utilize real-time monitoring of CI levels, allowing the possible implementation of task scheduling techniques in off-peak CI hours, reducing overall the carbon footprint and, hence, the environmental impact.

4 Power and Energy Consumption

4.1 Power analysis

The available data spans in the range of April 2020 to October 2022. The fluctuations of data do not demonstrate the pattern of seasonality visually, which is further confirmed with the STL analysis tool **??**, which also did not demonstrate the seasonality pattern of energy consumption.

Figure 3: Power STL

However, the separate regions of the graph exhibit constant trends during small periods (eg.: July-October). This potentially can be related to external conditions such as outside temperature depending on the season of the year: the cooling may vary depending on these external conditions, which can lead to either increased or decreased energy consumption. Moreover, the general increases in energy consumption can be related to the performing the computations, which leads to higher activity of the node and increased power usage, depending on the heaviness of the task.

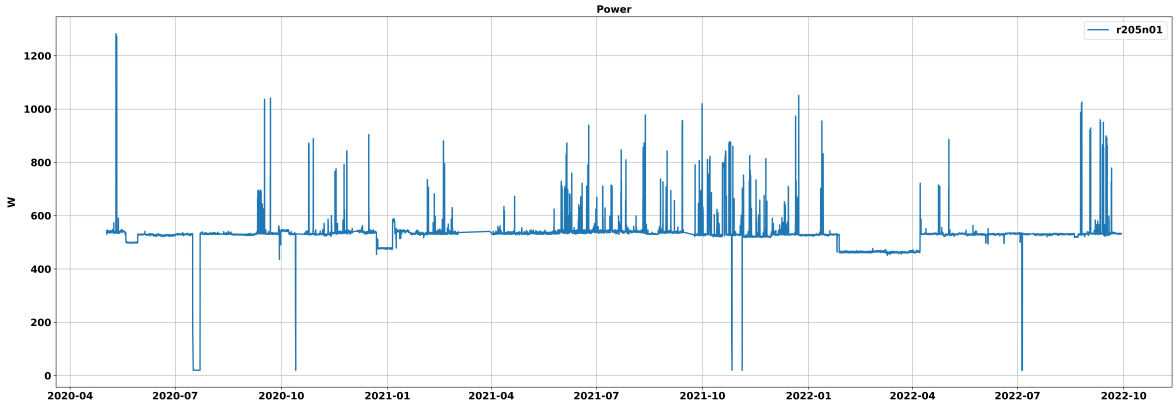

Figure 4: Power r205n01

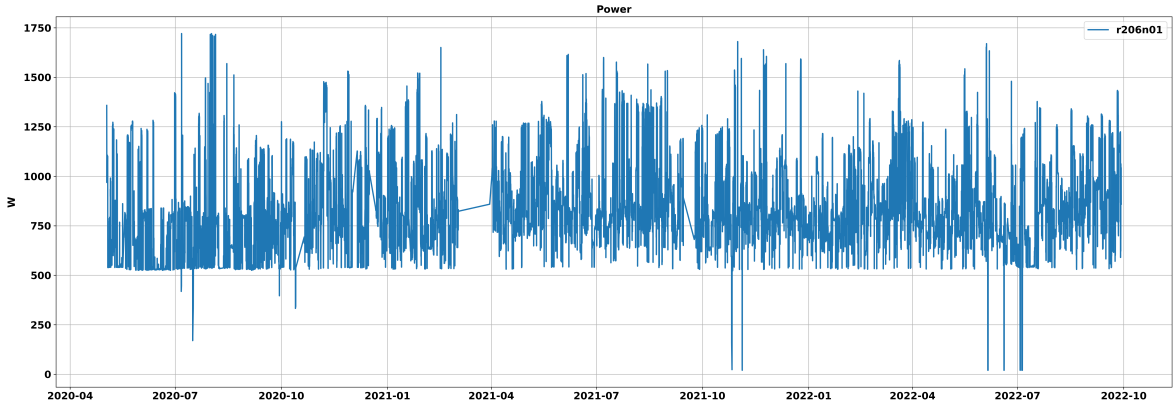

Figure 5: Power r206n01

In terms of power consumption, the comparison of the power data for the r205 and the r206 demonstrates a noticeable difference over the same period, from April 2020 to October 2022. In both cases the power data shows significant fluctuations: r206 exhibits a higher power usage pattern while r205 shows a more stable and consistent pattern. It is possible that the first rack may be used for different purposes, such as job submission and task scheduling, resulting in different power consumption patterns compared to other nodes.

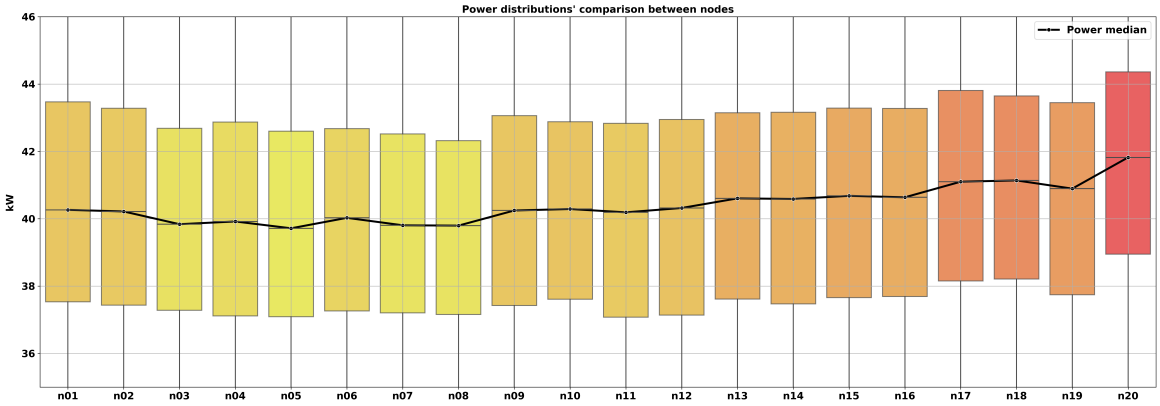

Figure 6: Power distributions comparison between nodes

The figure **??** provides a comparison of power consumption across 20 different nodes (n01 to n20). The boxplot represents the data distributions and a line plot shows the median for each node, highlighting the central tendency of power consumption. Each node shows variability in power consumption, with some nodes having wider variation, indicating more fluctuation in power usage. Moreover, Nodes located upside (n17 to n20) tend to have higher median power consumption compared to those located on the downside of the rack (n01 to n05). Node n20 shows the highest variability and median power consumption.

4.2 Temperature of Operation

Dealing with HPC forces us to take into consideration temperature distribution and the effect it might have on the general performance of the system, given the correlation that exists between power consumption and temperature itself; this correlation can be relatively seen in the following plot **??**, which is really important for our future considerations:

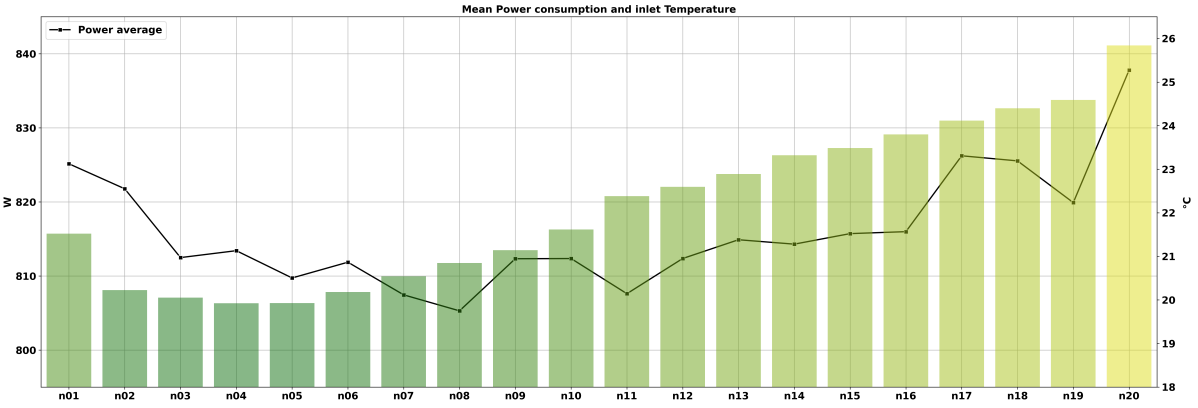

Figure 7: Power and Temperature observation

The line trend represents a piece of data we already investigated, the average power that the different nodes consume, while the histogram shows the average inlet temperature of the air flowing into the nodes; it is clear that the higher nodes tend to receive warmer air with respect to the lower ones, because of heat transfer from bottom to top of the room containing the racks. Particularly, the plot highlights the tendency of warmer nodes to consume more power. To further investigate this possible cause of inefficiency we can comment the second plot **??**:

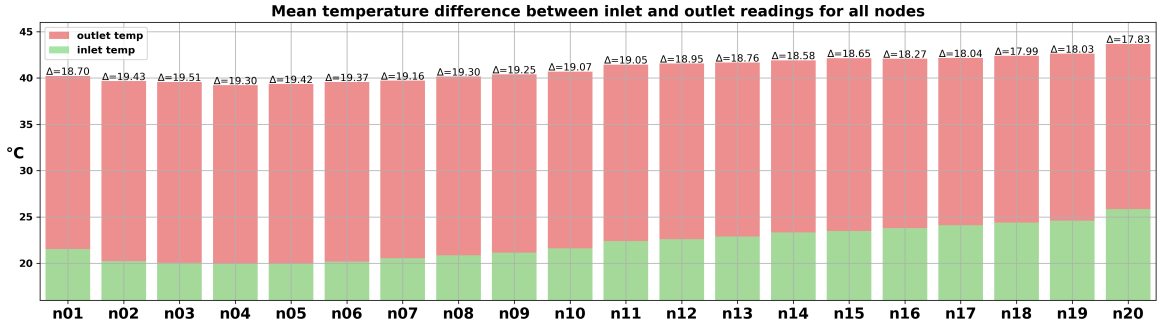

Figure 8: Inlet and Outlet temperature comparison

The average difference in terms of input and output temperature of similarly positioned nodes doesn't demonstrate significant difference moving vertically inside the racks. For this reason we can assume that the distribution of work and tasks throughout the nodes are balanced, given the similar thermal contribution that the nodes produce; this means that a different approach in the assignment of the computational work (task scheduling) might positively impact the overall power consumption, for example, favoring the lower nodes when possible, due to the temperature distribution impacting it's performance, which is directly correlated with the power consumption.

4.3 COP Calculation

Operational Carbon Footprint (COP) is one of the most important metrics for the goal of our observations since it represents the equivalent CO_2 emission generated from a certain amount of energy utilization, expressed in grams of CO_2, which represents a fundamental value for green computing analysis. This quantity is obtained by combining the energy consumption [kWh] with the Carbon Intensity [gCO_2/kWh] referred to the geographical area where the energy is used.

$$COP = E * CI$$

As the COP was calculated **??** and based on both Energy and CI, it has been demonstrated before that energy doesn't show any clear predictable pattern, while CI has better predictability, the further work can be focused on developing an enhanced model to predict the COP.

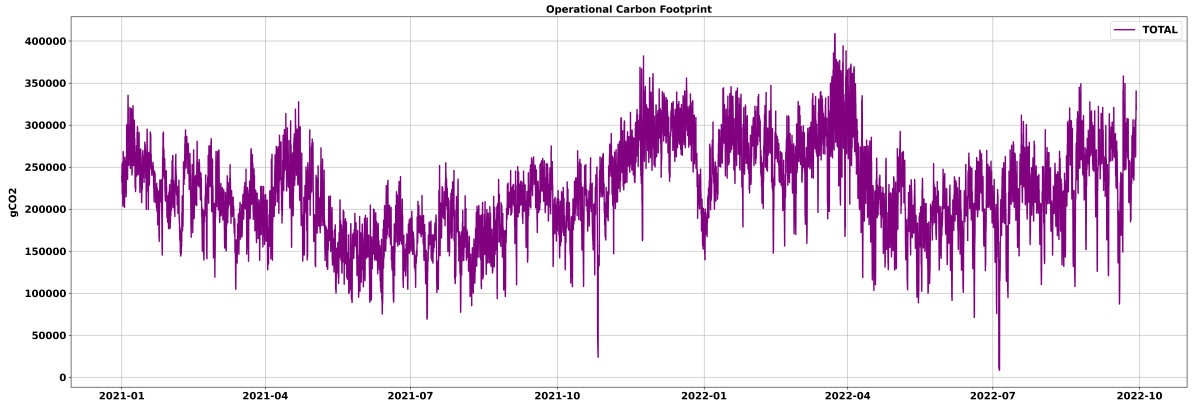

Figure 9: Total COP

4.4 Analysis of Data and Suggestions for Green Computing

Given these assumptions and hypotheses we wanted to evaluate how much this approach might actually impact on the overall energy consumption; to do so we considered two different cases, always having the same goal: to estimate how much we could have saved in terms of electricity cost and gas emission. I. With an approximation, we computed our final values using only the average energy per node; then we substituted the upper four nodes' energy value with the mean value and finally multiplied the obtained energy by the number of racks, taking into consideration that each node is present in all 49 racks. II. Using the whole 2.5-year distribution of hourly samples for the energy dataset, without the data manipulation, we repeated a similar computation. In both cases, the EUR/MWh cost metric we used has been taken from real data related to Italy [**ElectricityCost**], and referred to the cost of wholesale energy directly from the producer; so we need to take into consideration that the actual price per energy tends to be higher for the final customer, like CINECA itself.
Case I returned an approximate yearly saving of around 19.33 MWh, equivalent to 3151.39 €, obtained using the average cost for the energy in that period of time. Case II confirmed the initial result, returning a more accurate 3932.90 € per year, equivalent to 18.62 MWh.
Applying this kind of task management, a total of 9832.24€ could be potentially saved, which is equal to 46.56 MWh in the period of 2.5 years. But most importantly, this approach might have avoided the production of 25 669.45 kg of equivalent CO_2 for the past 21 months. Comparing these results with the total cost and CO_2 emission obtained from the unaltered data within the same period of time, we observe a 16.46% reduction in gas emission and a significant 18.13% reduction in the expenses for energy purchase.

5 Conclusion?

As a result of the data analysis of power consumption and carbon emissions, we can yield a valuable insights and future recommendations. The usage of predictive model (Meta's Prophet) to perform time series predicition of Carbon Intensity has demonstrated an efficiency in forecasting by leveraging historical data and hyperparameters, compared to simple lightweight model. The usage of such predictive model potentially can allow to implement proactive measures to reduce the effect of heavy computations, such as building a task scheduler with consideration of realtime Carbon Intensity data, or modeling of the carbon footprint of the given task. As per power profiling, the analysis of power consumption along the nodes demonstrated a correlation between temperature and energy usage, and it has been estimated that using the task scheduler with tracking this realtime data can potentially allow to reduce the electricity usage, hence, the expenses, and most importantly, reduce the CO_2 emissions.

References

[1] Neil Shephard Andrew C. Harvey. *10 Structural time series models*. 1993. URL: https://www.sciencedirect.com/science/article/abs/pii/S0169716105800458?via%3Dihub.

[2] *Decomposizione Trend-Stagionale attraverso l'utilizzo del LOESS*. URL: https://doc.arcgis.com/it/insights/latest/analyze/stl.htm.

[3] *ELECTRICITY MAPS*. URL: https://www.electricitymaps.com/data-portal/italy.

[4] *Prophet, forecasting at scale*. URL: https://facebook.github.io/prophet/.

[5] Benjamin Letham Sean J Taylor. "Forecasting at scale". In: (2017). DOI: 10.7287/peerj.preprints.3190v2. URL: https://peerj.com/preprints/3190v2/.