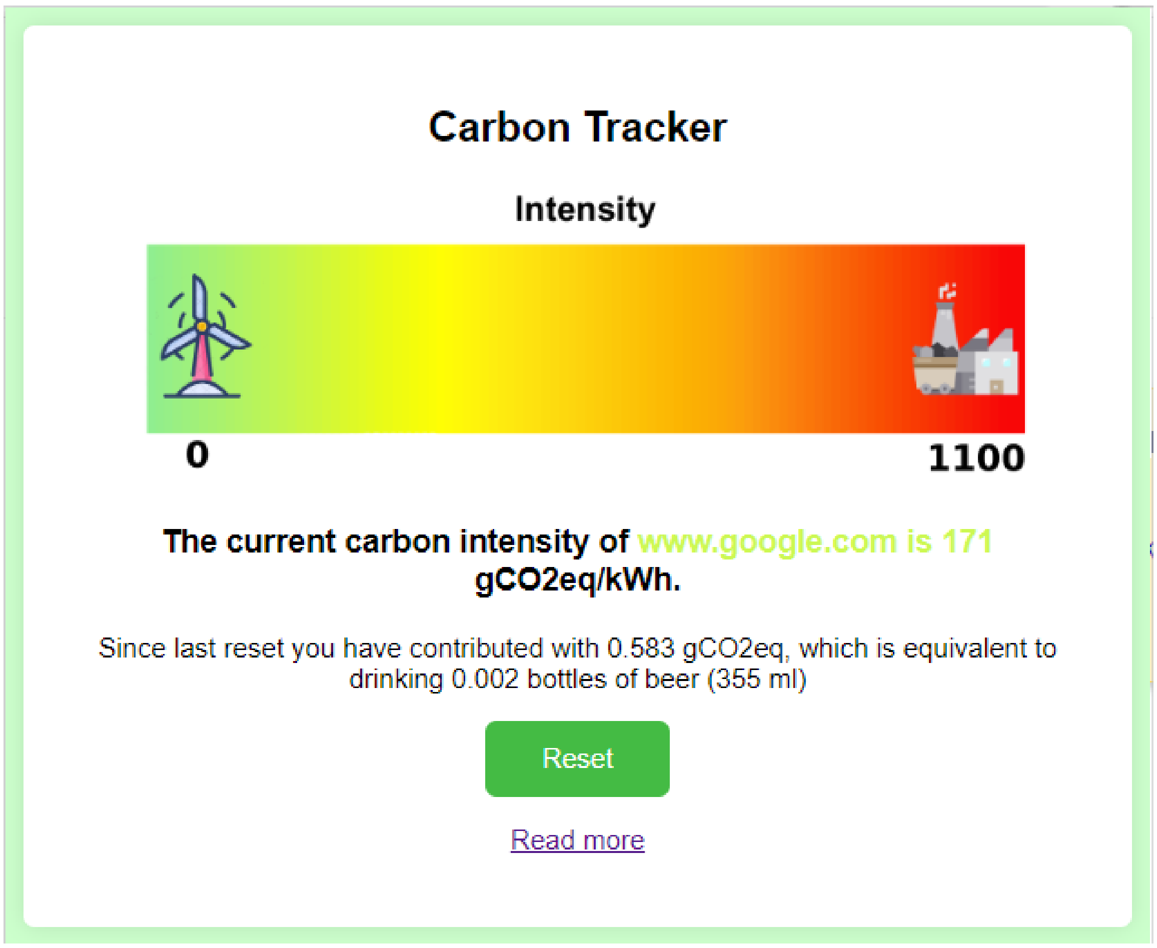
Report for Carbon Tracker Chrome Extension

Elias Stenhede Johansson, Gustav Leth-Espensen,
Florian Ecker-Eckhofen, Gustav Emil Nobert

April 2023


Carbon Tracker

Intensity

The current carbon intensity of www.google.com is 171 gCO2eq/kWh.

Since last reset you have contributed with 0.583 gCO2eq, which is equivalent to drinking 0.002 bottles of beer (355 ml)

Abstract

Carbon Tracker is a chrome extension that is designed to measure and display the end user's carbon footprint in real-time while browsing the web, informing the user about its environmental impact. Carbon emissions are calculated based on the carbon intensity of the power grid where the servers the user visits are located. Overall, the extension is a small but helpful tool for users who want to become aware of their carbon emissions and make more sustainable choices online.

1 Introduction

In today's digital age, data centers are the backbone of the internet and cloud computing services. They are responsible for storing and processing huge amounts of data. To do that, data centers and server farms consume an enormous amount of power, and the environmental impact of this energy consumption is increasing daily. This browser extension's goal is to educate users by informing them of the greenhouse gas emissions caused by servers and data centers while browsing the web.

The currently existing options for measuring carbon emissions when using the internet are limited to measuring the specific website that you are visiting. This can give you an idea of how the individual website affects the environment. However, this information is only limited to the base version of the website you visit and only to individual websites. This can make it hard for end users to grasp the amount of greenhouse gasses that they actually emit during a browsing session.

Therefore, the goal of this plugin is to highlight the greenhouse gas emissions that are produced by end-users while browsing the internet. We will in this report use the term "carbon" to refer to the amount of greenhouse gases equivalent to CO2, and "carbon intensity" to refer to how many grams of CO2 equivalent gasses are released for each kWh of electric energy on the power grid.

This paper explores the following research questions:

RQ1: How can a tool be built to make individuals aware of the carbon they emit while browsing the internet?

RQ2: Can such a tool teach the user about carbon intensity?

To answer these research questions, a browser extension was made that measures the carbon emission every half second and displays the entire session in real time.

2 Motivation

Accessing information on how much carbon is produced while browsing the internet can raise awareness about the environmental impact of our online activities. The average end user may not be aware that streaming a video or sending an email contributes to carbon emissions. By displaying the carbon emissions of the websites visited, it is easier to understand the carbon footprint of online activities. This knowledge can motivate end users to change their behavior and choose greener options, such as reducing online time.

The browser extension can help end users make informed decisions on which websites and online services they use. If end users find that a website is sending excessive amounts of data, they can choose to avoid it or look for alternative platforms with more sustainable practices. When holding websites accountable for their carbon emissions, it is easier to encourage them to adopt greener technologies and practices, such as minimizing their internet traffic.

One of the key things in this project was the accessibility of the product, and the ability of the average internet user to use it. The accessibility is mainly introduced in two ways, installing the product and using it. Chrome extensions can be made to fit both of these criteria. Extensions are known to be super easy to install and the extension itself is by design easy to use with very limited user interaction.

3 Methodology

Calculating the emissions

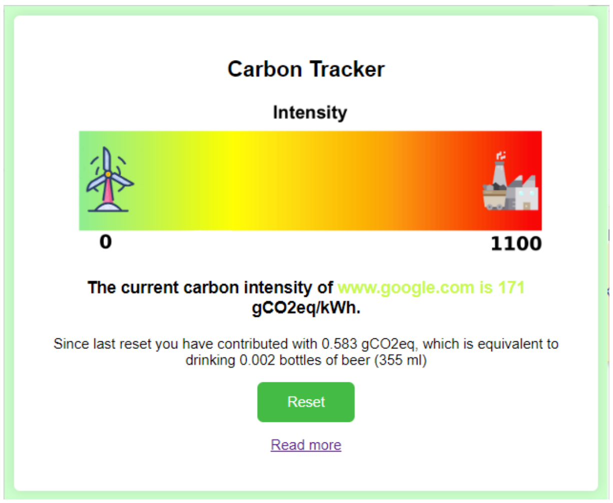
To calculate the user's footprint the plugin monitors the browser's incoming HTTP traffic. The size of the traffic is extracted by reading the HTTP header content-length. The carbon intensity for the area in which the server is located is then retrieved using an API, and the total carbon emissions are calculated as

$$C_{\text{transfer}} \cdot \sum_{\text{servers}} \text{carbonIntensity}(\text{server}) \cdot \text{bytesTransferred}(\text{server}) \tag{1}$$

Where $C_{\text{transfer}} = 0.81\,\text{kWh/GB}$ is an estimated constant, taken from the sustainable web design project. This number is valid for the "average" internet user, however specific numbers of kWh/GB are not available for most services and thus this is the best guess.

Informing the user

The information is then shown to the user when the plugin is open. By using color codes and a graphic it visualizes to the user if the intensity of the page is high or low. The color of the text also varies depending on the carbon intensity of the server's home country.



The current carbon intensity of www.google.com is 171 gCO2eq/kWh.

Since last reset you have contributed with 0.583 gCO2eq, which is equivalent to drinking 0.002 bottles of beer (355 ml)

Figure 1: Snapshot of what Carbon Tracker might display when the user is browsing the internet.

Additionally, the plugin informs the user how much carbon they have emitted since they last clicked the ”reset” button. This can be used by the user to monitor their sessions and learn something about their emissions throughout their session.

4 Implementation

Plugin

The plugin is built using Chromes plugin APIs and consists of a service worker and a web page. The service worker is consistent across all tabs and monitors incoming network traffic, and writes to storage, which is then consumed by the Javascript running on the users' browser tabs. The plugin utilizes Chromes web request API to add listeners to the events `onHeadersRecieved` and `onResponseStarted`.

In `onHeadersRecieved` the origin of the request is extracted, and the content-header http length if it is present. The amount of bytes from the origin is then saved.

`oResponseStarted` is needed because the other event does not expose the IP address of the origin. In this listener, the IP is extracted, and the plugin calls the backend to get the carbon intensity if it does not already have the intensity for the origin.

Lastly, the plugin writes to local storage every 500 ms, and it's then read by the page scripts. By utilizing this publisher-subscriber pattern, it's easy to keep the data consistent across all tabs.

The script running in the browser tabs extracts the total amount of bytes received from an origin, and their intensity. It also retrieves the local carbon intensity, which is also written to storage by the service worker (the carbon intensity values are valid for an hour since that is the time granularity of the Electricitymaps API). It then calculates the total emission using eq. (1) and displays it to the user. See figures 2 and 3 for illustrations of the extension's workflow.

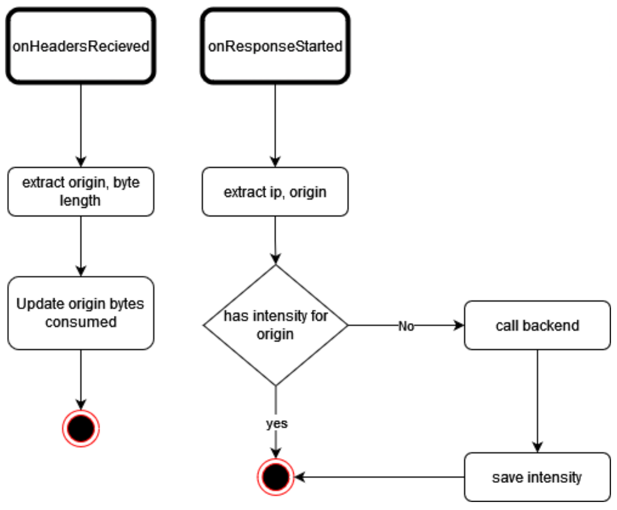


Figure 2: Schematic illustration of the flow in the front end.

The backend

The other part of the system is comprised of a REST Node.js server. The server supports the plugin by matching the IP addresses of the servers the user receives responses from, with countries. The plugin calls the backend with the IP address of the server it wants the carbon intensity for, and the backend then queries the ipinfo API to get the country code the IP is present in. It then calls the electricity maps API and retrieves a two-day forecast for the carbon intensity in that country, and returns the current intensity.

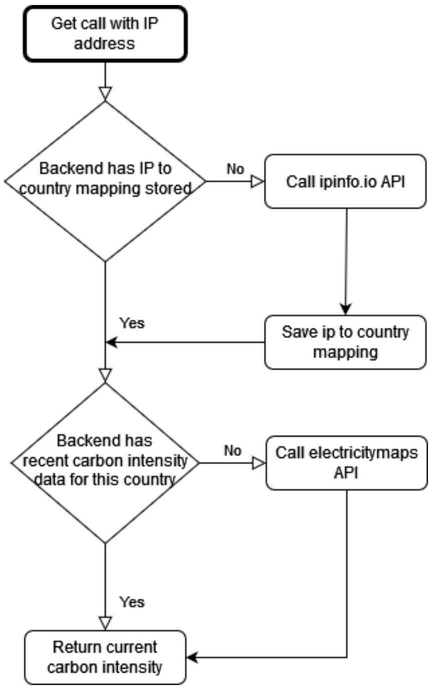

Get call with IP address
Backend has IP to country mapping stored
No
Call ipinfo.io API
Yes
Save ip to country mapping
Backend has recent carbon intensity data for this country
No
Call electricitymaps API
Yes
Return current carbon intensity

Figure 3: Schematic illustration of the flow on the server side.

The front end and back end are separated to keep the API keys secret. Getting the carbon intensity could be done by the front end, but the API keys the backend utilizes would need to be set up by each user when they install the plugin, and would possibly be too much work for a user. The target end users are not tech-literate enough to execute that process. It might also be against Electricitymaps TOS. The backend stores IP and country carbon intensity data and responds with that instead of querying third-party API's on every request. Querying the third-party APIs would be expensive, and this approach can utilize the API's cheaper payment plans.

This system architecture is used because it supports the extraction of the data needed to support our research questions. The data needed are where the servers the user communicates with are located, and how much data they sent. By utilizing the Chrome API, the plugin can easily do this, without being invasive, and it's easy to set up. In the future, the plugin is also easy to expand with more visual tools, to improve user education while they are browsing even further.

5 Discussion

CO2 emissions from the information technology sector are projected to reach 14% in 2030, and will therefore be a significant amount of the emissions. The methodologies' aim is to educate the user as well as possible, about the CO2 emissions related to browsing the web. By using a methodology that focuses on the provider of the services, users get informed about the carbon emissions of their providers, instead of only focusing on the users' costs and emissions. The learnings of the user can then be used in the future, to better understand discussions about service providers and web traffic. And hopefully, help them make better choices.

By informing the user about the carbon intensity of the websites they visit, the user can choose when it wants to perform certain activities. Lately, there has been an increased focus on electricity prices because of the Russian invasion of Ukraine. The war has also plunged the carbon intensity of the grid into a state of constant fluctuation, and while the changing prices are on people's minds, this is an opportune moment to educate them on carbon intensity. By seeing how the intensity varies for the web pages they visit, the user can make informed choices about when to use the web, and when to do something else, if they care about their carbon footprint.

It also shows the user the carbon intensity of their providers' servers. If a provider consistently provides their services from a country with higher carbon intensity than other providers utilized by the user, the user might begin to notice, and prioritize other providers that have a smaller carbon footprint.

The ease of use is paramount to the end user. The convenience that a Chrome extension provides makes the user more likely to get the knowledge concerning their carbon emission that this project hopes to achieve.

5.1 Limitations

There are multiple limitations to our methodology. The calculated number for the carbon emissions is based on an article that estimates a constant, that factors in all the emissions related to network traffic. That number does not capture the amount of work the server needs to do to process the data, or what is needed to keep these servers running.

Furthermore, it's not possible to calculate the energy usage of the server the user communicates with. Our methodology would not capture some things that are computation heavy for the server such as gaming on the web, since it exclusively measures the network sent. Other services such as chatGPT will also spend a lot of energy on producing it's messages, which will not be measured either.

The current methodology might also unfairly brand some websites as carbon-intensive. Some server farms might have their own energy grid, where the power is supplied by a green infrastructure setup around them. The intensity number used in the calculations is based on the entire country, so if the servers actually use green energy, the plugin won't know.

Additionally, the plugin only supports traffic over HTTP. Other protocols might be used such as web Sockets, or UDP which should be supported in the next version. Most of the data consumed by the typical end-user is streaming. Streaming is very intensive, and educating the user on the emissions on their streaming is gonna be included in the next version. The Chrome web request API does not provide support for monitoring web socket traffic, and it's therefore not possible to implement at the current time, without being invasive to the user's privacy. Changing from a Chrome plugin to another setup might make this feature implementable.

Even with the above limitations our tool still has a use case. It limits the precision of the data, but the data does not have to be exact to still be informative to the user. The most impactful of the limitations is the potential unfair branding of websites, which could be mitigated by manually checking websites and keeping track.

6 Future work

The plugin is in its current state and displays some information to its user, but it could provide even more detailed information. In the next iterations, more tools and functionality could be implemented. It could for example support setting a ”threshold” for the user, where the user inputs how much carbon he wants to emit during a day and it could notify it when the threshold is reached. The threshold would then be a physical thing, such as a kilometer driven in a car, a length of a shower, or other things the user could relate to.

Another useful user feature would be a comparison between the different pages the user has visited, and how much time was spent on these sites. By visualizing the time spent and the carbon emitted on every page using charts, the user is informed about their trade-offs. This data would allow the user to reflect on how their time is spent, and which sites they want to emit their carbon for. This feature would also provide the tool with some individual sustainability.

Lastly, nothing is measured if the user uploads data. The plugin is reliant on the incoming HTTP content-length header and doesn't capture outgoing traffic. Uploading is a fraction of the internet traffic for most users, and is therefore considered out of scope for this plugin.

This version of the extension is only a prototype, to make it into a commercialized product one would need unlimited access to the APIs used. This is very expensive costing for instance 500€ per country per month for the electricity maps. Furthermore. user surveys would need to be conducted to improve the product as a whole.

7 Related work

Browser extensions similar to this one have been created before. For example the Firefox extension eco2rd and the chrome extension carbonalyser. There is also the co2.js framework which can be used to approximate carbon emissions from web traffic. These extensions are visually more informative than our extension but lack the functionality of taking the *current* carbon intensity of the electricity grid. The same goes for the co2.js library which uses the average carbon intensity from the past year for each country.

This implies that carbon emissions could be both over- or underestimated. This is especially prone to being very inaccurate in countries with a large amount of wind and/or solar power, as the carbon intensity then regularly fluctuates up to several orders of magnitude.

8 Conclusion

An early stage of a tool to inform users about their CO2 emissions and the carbon intensity of the electrical grid is developed. The tool supports basic functionality for users to learn but has the possibility to be developed even further to provide the users with more details.

9 Appendix

Github Repository: https://github.com/GustavLeth/Sustainable_Software_Engineering