CS3300 Introduction to Software Engineering

Lecture 14: Sustainability in Software Engineering

Dr. Nimisha Roy

- nroy9@gatech.edu

Sakshi Deshpande

What is Sustainability?

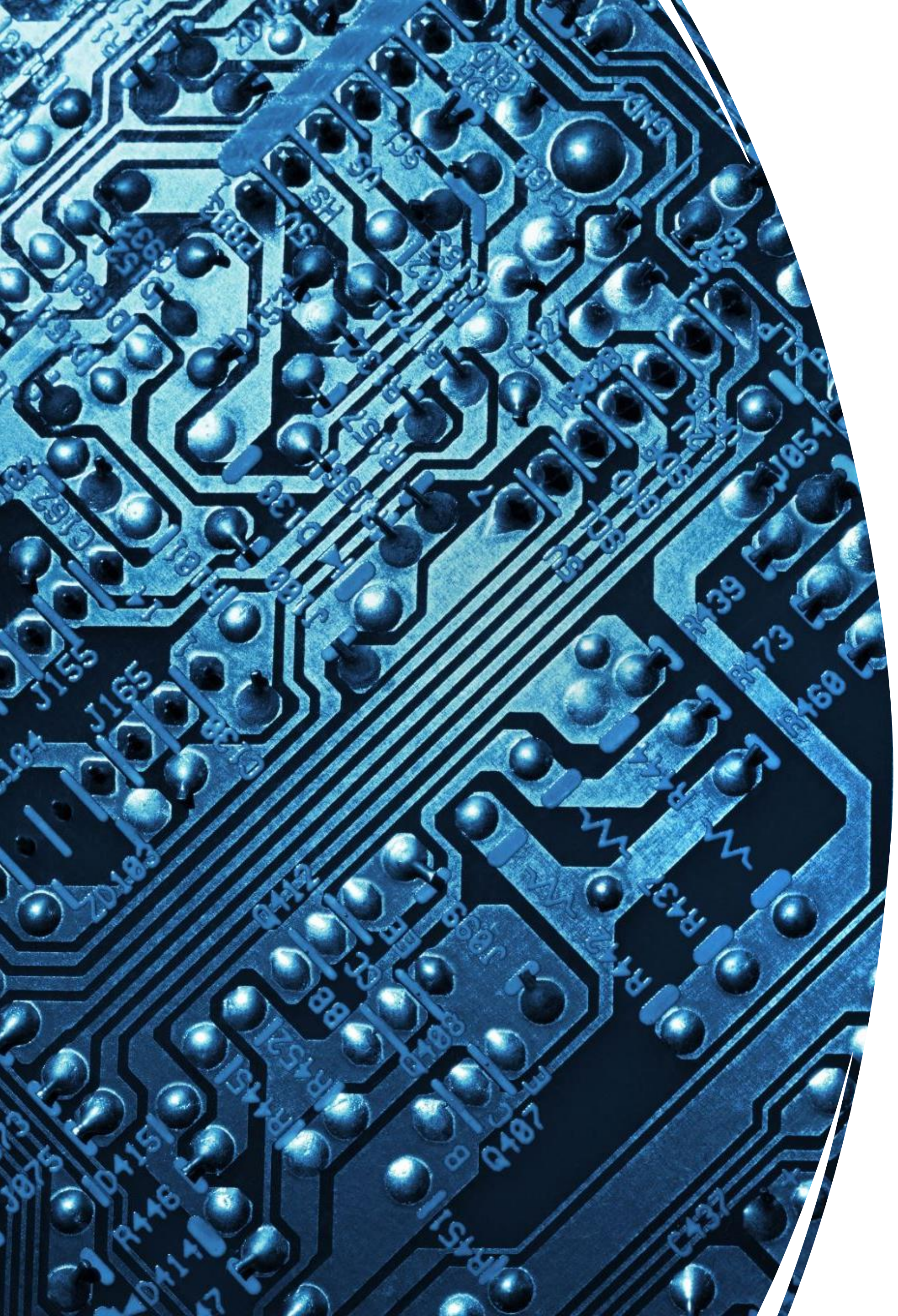
Sustainability as defined by the United Nations is: "**meeting the needs of the present without compromising the ability of future generations to meet their own needs.**"

- It is important for us as humans to protect many aspects of our environment and sustainability development is key to solving many of world's current issues.

United Nations SDG

The Sustainable Development Goals developed by the UN are the blueprint to achieve a better and more sustainable future for all.

- They address the global challenges we face, including those related to poverty, inequality, climate change, environmental degradation, peace and justice.
- **The 17 Goals are all interconnected** and they aim to be achieved by 2030.



Can Software Engineering be Sustainable?

Sustainable Software Engineering is an emerging discipline at the intersection of climate science, software, hardware, electricity markets, and data center design.

- Software engineering can be harmful to the environment if it does not meet certain principles
- There are **6 core principles of sustainable software engineering** which include: carbon efficiency, energy efficiency, carbon awareness, hardware efficiency, measurement, and climate commitments
- You should have covered these 6 principles in the course prior to class

SSE Principle 1: Carbon Efficiency

Basics:

- Carbon efficiency is about reducing the amount of carbon emitted per operation.
- Carbon dioxide (CO_2) is the most common GHG emitted through human activities.
- Achieving net-zero carbon means balancing emissions with extraction.

Example:

- Redesigning a data-intensive algorithm to minimize data transfers across data centers because transferring data consumes energy, leading to carbon emissions.
- By selecting a cloud region closer to the majority of users, data transfer distances and times are reduced, leading to less energy consumption and carbon emissions.

SSE 2: Energy Efficiency

Basics:

- Focus on minimizing the energy consumed per useful operation.
- Energy-efficient operations help in reducing the carbon footprint.
- It's crucial for both environmental and cost-saving reasons.

Example:

- Developing a mobile app that conserves battery by optimizing background processes and turning off sensors when not in use.
- Using cloud-based solutions that ***autoscale***, like Google Cloud's Compute Engine, ensures that only necessary resources are used, preventing energy wastage from over-provisioning.

SSE3: Carbon Awareness

Basics:

- Understanding the carbon cost associated with computation and storage.
- Being aware allows for better decision-making in software design and deployment.

Example:

- Migrating services to cloud regions that use a higher mix of renewable energy to reduce carbon emissions.
- Educating students on the benefits of serverless computing, like Google Cloud Functions, which only run when called upon, reducing idle resource consumption and emissions.

SSE4: Hardware Efficiency

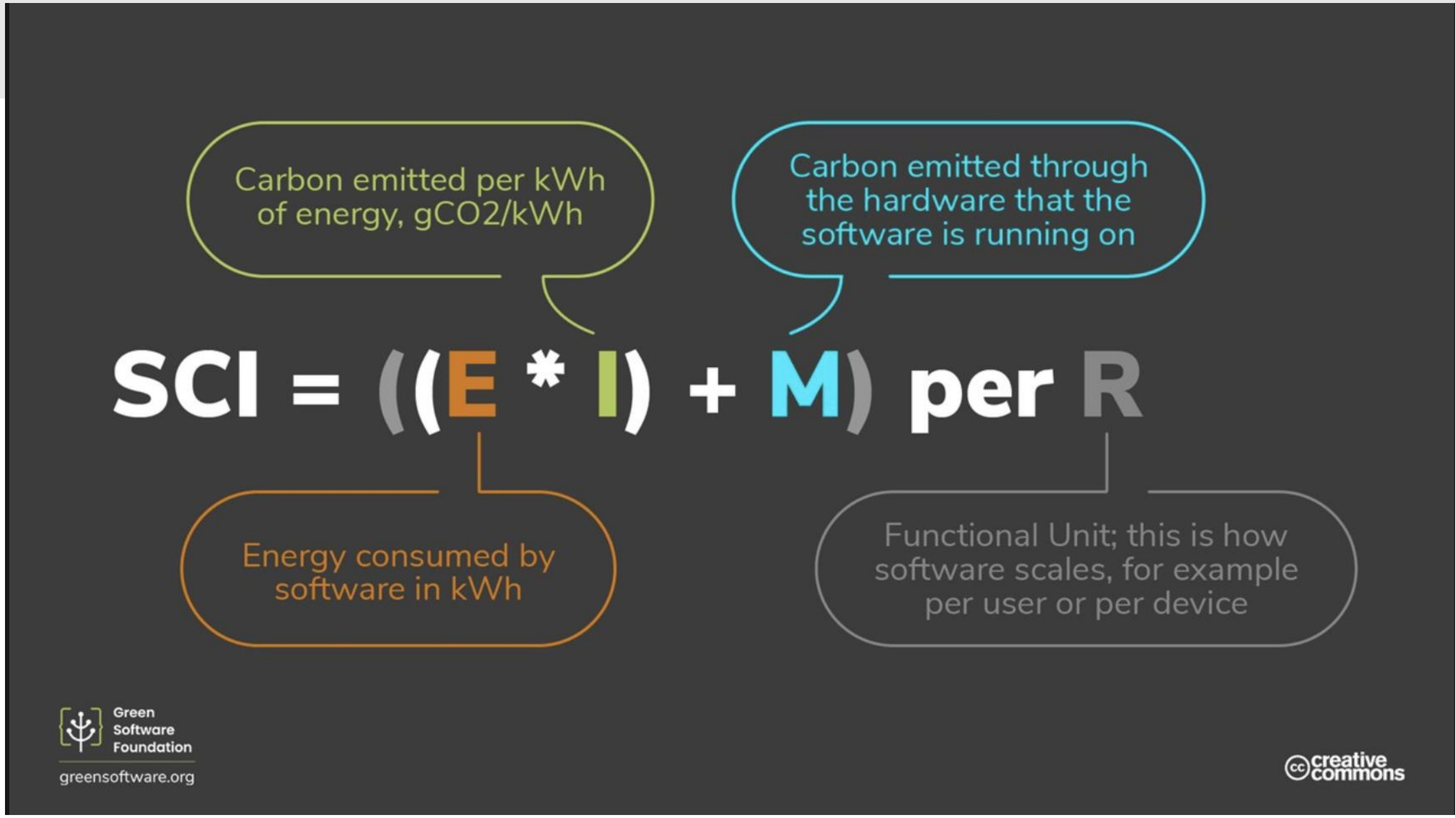
Basics:

- Efficient use of hardware can prolong its lifespan, reducing the need for frequent replacements.
- Manufacturing new hardware has a significant carbon footprint.

Example:

- Designing software that runs efficiently on older devices, extending their utility and reducing electronic waste.
- Trying not to dispose of your devices by extending the lifespan of your device.

SSE5:Measurement



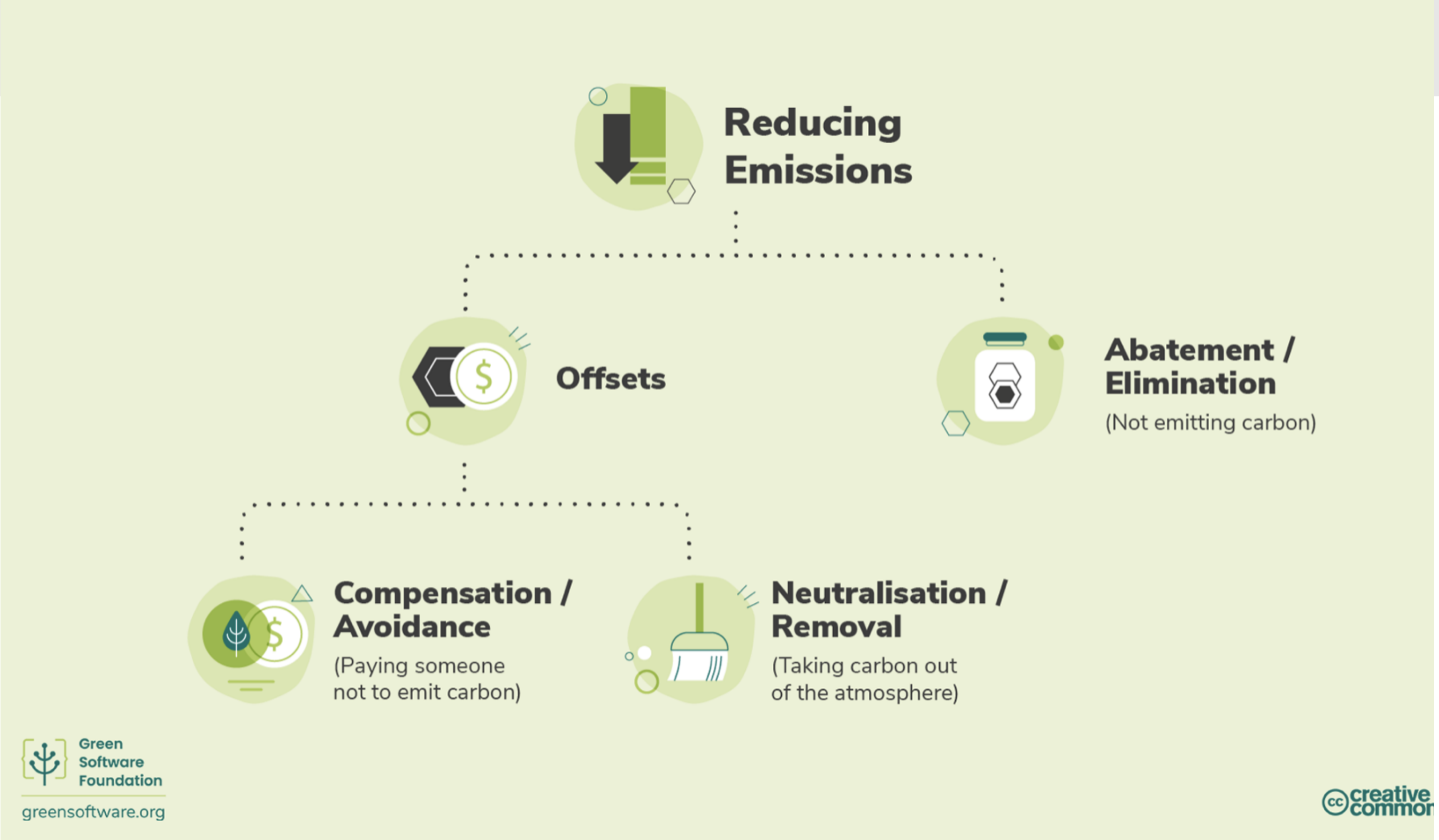
Basics:

- Monitoring the energy and carbon costs is crucial for continuous improvement.
- Accurate measurements can guide sustainable decisions.

Example:

- Utilizing monitoring tools like Google Cloud's Carbon Intelligence Dashboard to keep track of the carbon footprint of workloads.

SSE6: Climate Commitments



Basics:

- Companies can make pledges to achieve specific sustainability milestones.
- Commitments can be towards carbon neutrality, renewable energy use, etc.

Example:

- A software firm pledging to become net-zero carbon by 2025 through renewable energy investments and carbon offsets

Case Study 1: Cloud Computing

The CLEER Model allows companies to analyze the technical potential for energy savings by moving to the cloud. For example, if all US businesses moved their email, productivity software, and CRM to the cloud, then the primary energy footprint can be reduced by 87%.

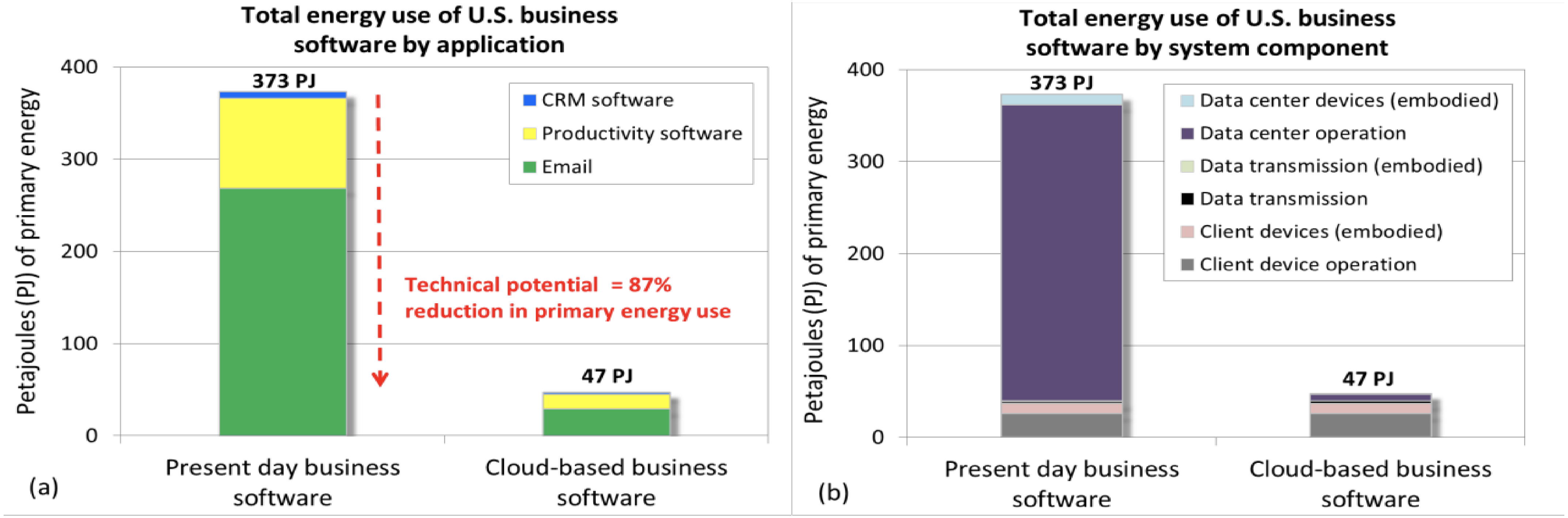


Figure ES-1: Primary energy use of present-day and cloud-based business software systems by: (a) application and (b) system component.

https://escholarship.org/content/qt68b51379/qt68b51379.pdf

Case Study 2: Requirements Engineering

- *Requirements Engineering for Sustainability* denotes the concept of using requirements engineering and sustainable development techniques to **improve the environmental, social, and economic sustainability of software systems** and their direct and indirect effects on the surrounding business and operational context.

Guiding Questions for Green RE:
1. Does the system have an explicit sustainability purpose?
2. Which impact does the system have on the environment?
3. Is there a stakeholder for environmental sustainability?
4. What are the sustainability goals and constraints for the system?

https://ceur-ws.org/Vol-1216/paper8.pdf

Case Study 2: Requirements Engineering Con.

Example 1: A study was conducted that we elaborated a case study on the car sharing system DriveNow that was incorporated green requirements engineering.

The business model is commercial car sharing for registered users with flexible drop-off points for the vehicles on public parking lots. The car sharing system is composed of a web application for registration, reservation and billing, a car fleet maintained by a service partner where each car is equipped with a meter and a transponder, and a central database.

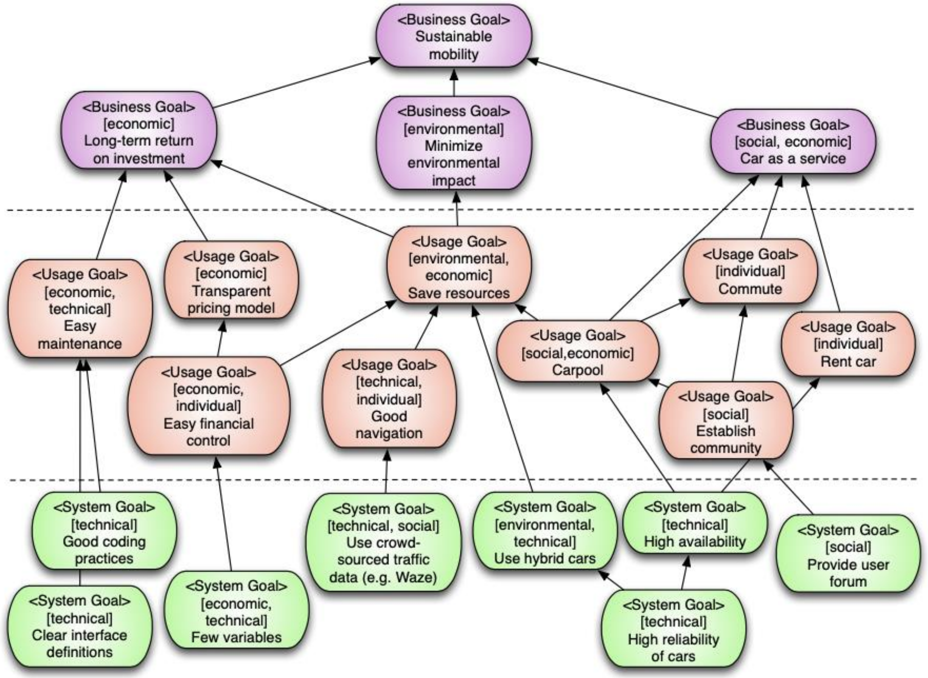


Fig. 4. Sustainability Goals of the Car Sharing Case Study

In the car sharing example, the objectives in the three categories were shown by considering the dimensions of sustainability and how they are presented within the system.

How can we make a change?
Target E - Metrics

SCI = ((E * I) + M) per R

Speedup = T_{φ} / T_{o}
where T_{φ} is the runtime of the baseline code and T_o is the runtime of the optimized/parallelized code.

Powerup= P_{o} / P_{φ}
where P_{φ} is the average power of the baseline code and P_o is the average power of the optimized code. Powerup is greater than 1 the new code has a higher average power. That is often the case with parallel programs as they utilize more cores.

Greenup = $E_{\varphi} / E_{o} = (T_{\varphi} \times P_{\varphi}) / (T_{o} \times P_{o})$ = **Speedup/ Powerup**

https://greensoftware.foundation/articles/gps-up-a-better-metric-for-comparing-software-energy-efficiency

How can we make a change?
Target E - Calculator

SCI = ((E * I) + M) per R

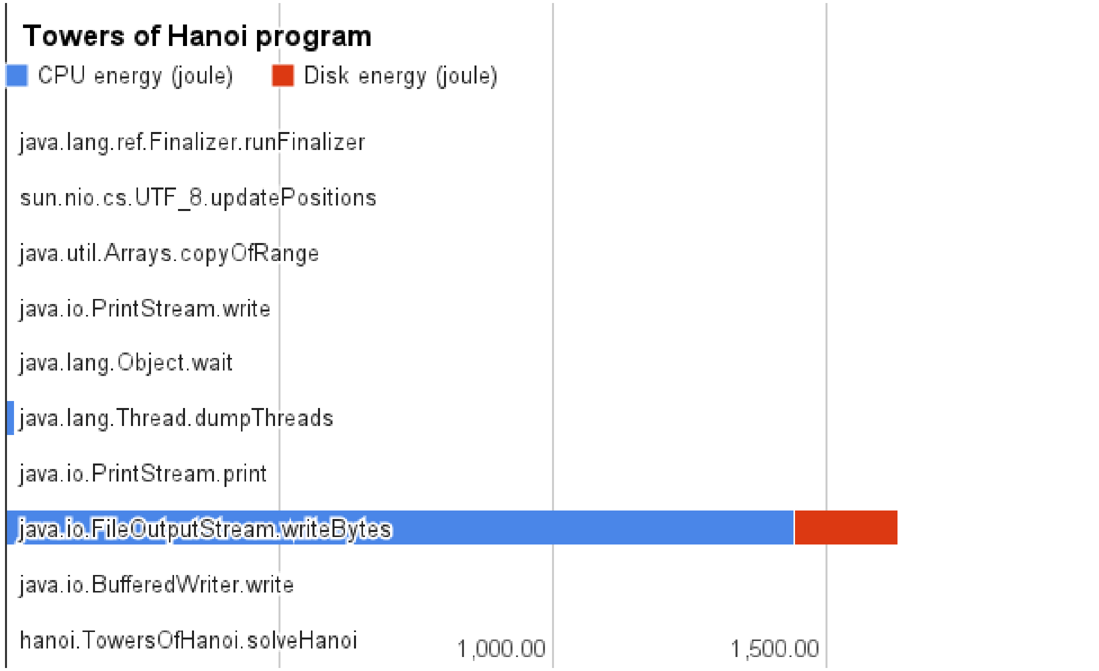
Joular JX

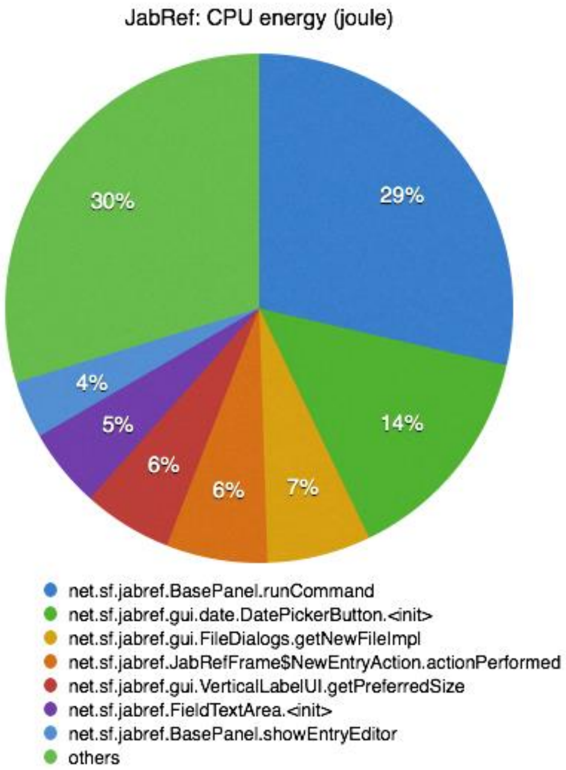
Java agent that hooks to the Java Virtual Machine (JVM) during its start and measures the energy consumption of each executing method of the source code(e.g., classes and methods).

https://www.noureddine.org/articles/measure-the-energy-consumption-of-java-applications-with-jalen

Example: Towers of Hanoi – 1 class and 2 methods. The program takes 169 seconds to solve the game, and writes a text file of 612 MB.

		CPU energy (joule)	Disk energy (joule)
1	hanoi.TowersOfHanoi.moveDisk	1443.03350953069	190.364060261896
2	hanoi.TowersOfHanoi.solveHanoi	0.093799833370705	0





How can we make a change? Target E and M

$$SCI = ((E * I) + M) \text{ per } R$$

- Energy-Efficient Hardware: hardware designed for energy efficiency can further reduce E. Modern CPUs often have architecture-level optimizations for power management.
- Server Selection: In a cloud environment, choosing server types that align with your needs (e.g., compute-optimized, memory-optimized) can help in reducing excess energy consumption.
- Longevity and Durability: Choosing hardware that has a longer lifespan or is more durable reduces the frequency of replacement and thus reduces the carbon footprint associated with producing new hardware (M).

How can we make a change? Target E and M

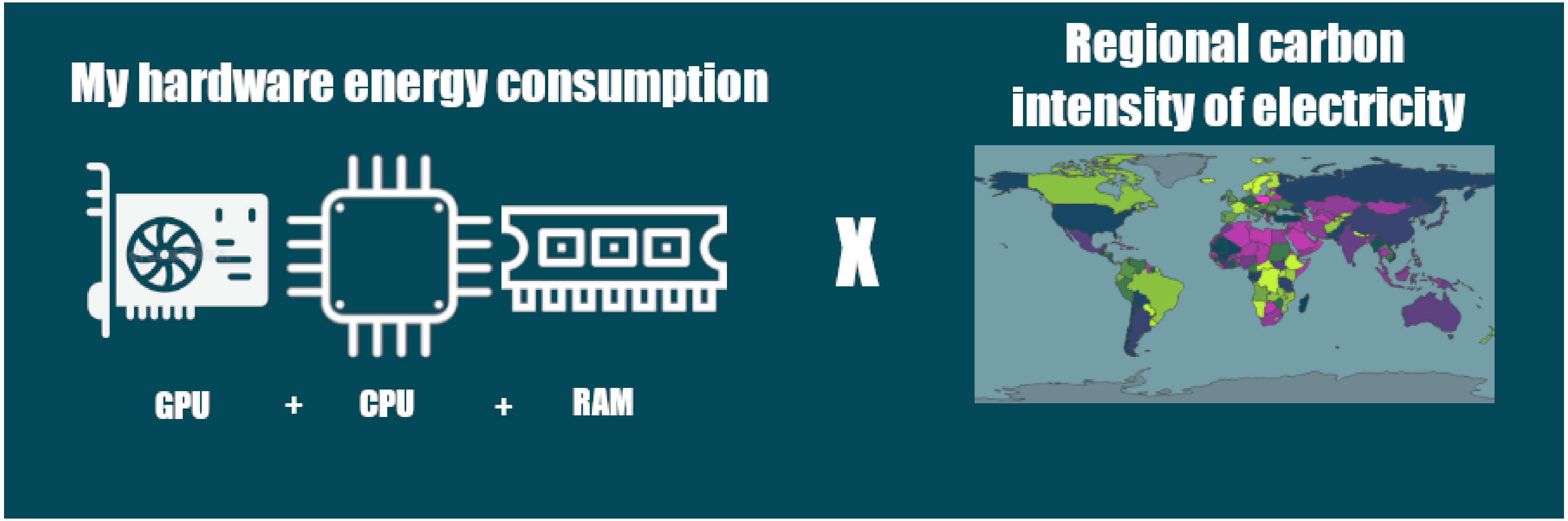
$$SCI = ((E * I) + M) \text{ per } R$$

- Recyclability: Hardware components made from recyclable materials or designed to be easily recyclable can reduce the carbon footprint (M) associated with disposal.
- Manufacturing Processes: Opting for hardware from manufacturers that prioritize sustainable production methods can reduce the M value. This includes reduced emissions in production, sustainable sourcing of materials, and energy-efficient factories.
- Optimized Data Storage: Using SSDs instead of HDDs or optimizing database hardware can lead to faster data access with less energy.

How can we make a change?
Target M : Calculator

SCI = ((E * I) + M) per R

Code Carbon



https://github.com/mlco2/codecarbon#start-to-estimate-your-impact-

How can we make a change?
Target M - Calculator

SCI = ((E * I) + M) per R

Carbon Aware SDK

- Helps you build the carbon aware software solutions with the intelligence to use the ***greenest energy sources***. Run them at the ***greenest time***, or in the ***greenest locations***, or both! Capture consistent telemetry and report on your emissions reduction and make informed decisions.
- Many companies use this tool globally
- Machine Learning (ML) workloads are a great example of long running compute intensive workloads, that often are also not time critical. By moving these workloads to a different time, the carbon emissions from the ML training can be reduced by up to 15%, and by moving the location of the training this can be reduced even further, at times by up to 50% or more.

https://github.com/Green-Software-Foundation/carbon-aware-sdk

Resources for your project

- https://greensoftware.foundation/projects/
- https://www.carbonfootprint.com/calculator.aspx
- https://github.com/joular/joularjx
- https://github.com/mlco2/codecarbon#start-to-estimate-your-impact-

What can you do differently moving forward?

Embrace Cloud Computing

- Cloud platforms like Google Cloud, AWS, and Azure are optimizing for sustainability. Utilizing them can reduce the carbon footprint of your applications. Learn about the sustainability initiatives of your chosen cloud provider.

Think Efficiency:

- Write efficient code. Less resource-intensive applications lead to lower energy consumption. Focus on algorithms and data structures – the foundation for efficient software.

Lifelong Learning:

- The tech industry and its impact on the environment are continually evolving. Stay updated with the latest sustainability practices and trends.

Consider the Full Lifecycle:

- Think beyond just the coding phase. Deployment, maintenance, and even decommissioning software have environmental impacts. Opt for platforms that manage resources well, like serverless architectures or containerized deployments.

Collaborate and Discuss:

- Join or initiate conversations about sustainability in tech within your academic community. Collaborate on projects that emphasize green tech solutions.

Practical Experience:

- Whenever possible, apply sustainable practices in your projects and internships. Seek out companies or projects that prioritize sustainability for internship opportunities.

Ethics Matter:

- Ethical considerations are crucial. The tech we build today shapes the world of tomorrow. Always consider the long-term implications of your software on both society and the environment.

In class assignment

Optimize the provided code to be energy efficient. Observe and note the differences in energy consumption and carbon footprint between the efficient and inefficient versions of 2 methods.

<u>Carbon emission Calculations using a laptop in Georgia</u>

- Let's assume a general-purpose laptop in Atlanta, Georgia. A common laptop CPU can consume between 15-45W, but for our example, let's take an average of 30W.

- In Georgia, according to the U.S. Energy Information Administration, the carbon intensity is roughly 1,000 lbs of CO_2 per megawatt-hour (MWh) of electricity. This is due to a heavy reliance on fossil fuels, particularly coal and natural gas.

- So, if a laptop uses 30W, it consumes 0.03 kWh in an hour. The carbon emission for 0.03 kWh in US would be:
  - 0.03 kWh * 1,000 lbs/MWh = 0.03 lbs of CO_2 per hour.