Measuring the power draw of computers

What you cannot measure, you cannot improve

Mercredi 19 Mai

Power draw of computers

Applications

- Monitor energy usage on data center
  or/and
- accurately measure each layer

A not so trivial topic

- Difficulty to isolate the energy hungry elements
- Dependent on the built in sensor and constructor support.
- Low level (close to hardware) programming

What we learn in highschool

- **Joule**: energy transferred to an object when a force of one newton acts on that object in the direction of the force's motion through a distance of one metre (1 newton-metre or Nm)
  - The energy required to lift a medium-sized tomato up 1 metre
- **Watt**: 1 joule per seconds
- **kWh**: ????? Joules

What we learn in highschool

- **Joule**: energy transferred to an object when a force of one newton acts on that object in the direction of the force's motion through a distance of one metre (1 newton-metre or Nm)
    - The energy required to lift a medium-sized tomato up 1 metre
- **Watt**: 1 joule per seconds
- **kWh**: **3600000** Joules
    - 3 hours of GPU computation

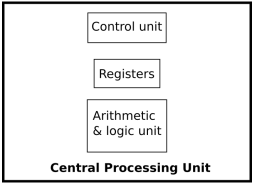
What we learn in highschool

- **Joule**: energy transferred to an object when a force of one newton acts on that object in the direction of the force's motion through a distance of one metre (1 newton-metre or Nm)
  - The energy required to lift a medium-sized tomato up 1 metre
- **Watt**: 1 joule per seconds
- **kWh**: **3600000** Joules
  - 3 hours of GPU computation

How a computer uses energy?

What we learn at the university

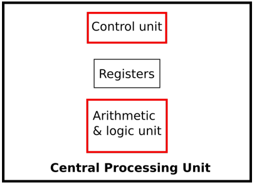
Let's start with the cpu



- From 100Khz in 1971 to some Ghz today
- Composed of millions of transistors (Moore law)
- Cristal of qwartz giving the frequency of the cpu
- Optimization of the frequency to save power (turboboost)

What we learn at the university

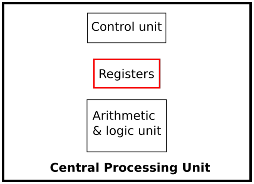
Let's start with the cpu



One cpu Core

- Instructions set : boolean, floating operations
  - RISC (AMD), CISC (Intel), dedicated FPGA instructions
    `/proc/cpuinfo`
- Conditions the power draw
- Low level programmation with binary networks

Let's start with the cpu



- Registers : fast memory used by the ALU
- 10-100 registers with 8-64 bits

and continue with the memory

Central Processing Unit

Memory caches (L1, L2, ...)

Read Access Memory (RAM)

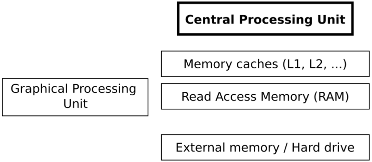
External memory / Hard drive

- Memory hierarchy
  - Closer to the cpu → smaller and faster

```
pgay@ansabere$ lscpu
L1d cache:                        384 KiB
L1i cache:                        256 KiB
L2 cache:                         4 MiB
L3 cache:                         16 MiB
```

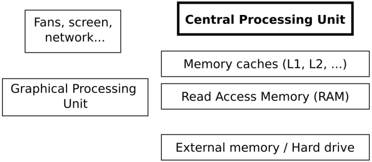
- Moving data up and down the memory hierarchy costs time and power
- Taken into account in optimization code to limit these moves.
  - Eg: Row major or column major storage in matrix multiplication

GPU : major actor in the consumption



- Consumes more than the whole computer (Bridges, Imam, and Mintz 2016)

Other components



- Consumes more than the whole computer (Bridges, Imam, and Mintz 2016)
- Overall a full a diagnostic might be complex
  - lack of available sensors

GPU versus CPU

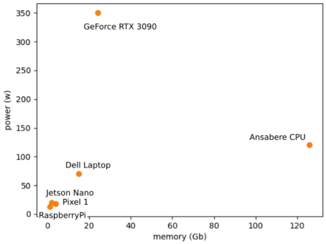
- Invented by nvidia in 1999
- Thousands of cores to enable parallelism
- Lower amount of RAM memory available
- Higher latency : GPU clock speed $<$ CPU clock speed
- Higher memory throughput : GPU operates on larger chunks of data
  - GPU can fetch data from its RAM more quickly
  - CPU bandwidth $<$ GPU bandwidth
- Smaller set of instructions dedicated to graphics and matrix calculus
- More power hungry and requires a CPU

Energy efficient since the computations is faster.

Other hardwares

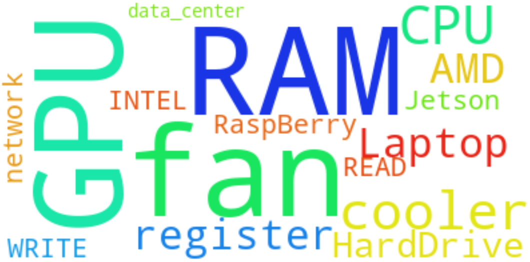
- AMD CPU: RISC instruction set lower energy than Intel processors
- Programmable circuits with custom instruction set
  - Field-programmable gate array
  - Application-specific integrated circuit (ASIC):
    Implements the Tensor Processing Unit.
- Small devices
  - Rasberrypi
  - Jetson Cards

Some perspective numbers



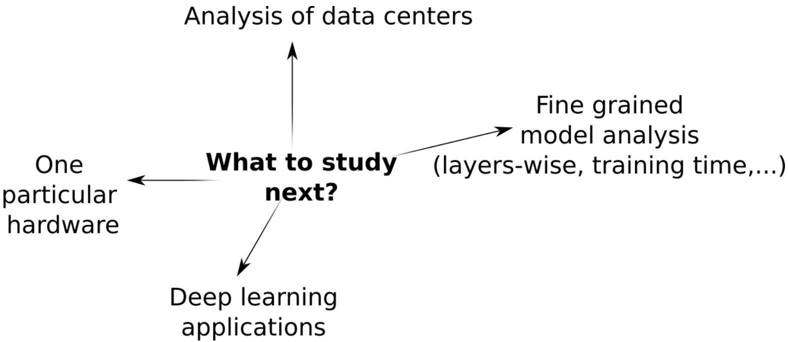
Power usage versus memory capacity

- How to rank machines by efficiency ?
- Compromise between, power, memory, computing capacity



How to measure all of it?

Different angles to tackle



Related work on consumption measurements

- Opensource libraries for machine learning carbon footprint (Henderson et al. 2020; Anthony, Kanding, and Selvan 2020)
  - based on RAPL and nvidia-smi
- Fine grained studies on a specific Jetson hardware (Rodrigues, Riley, and Luján 2018; Holly, Wendt, and Lechner 2020)
- Generic libraries from the data center community : Papi, Likwid
- Machine learning based prediction models (Cai et al. 2017, Jia et al. 2015)
- French Startup : https://github.com/hubblo-org

Hard to get recover exactly what you measure on your power meter.
Developping from scratch requires complex low level programming skills

Related work on consumption measurements

- Opensource libraries for machine learning carbon footprint (Henderson et al. 2020; Anthony, Kanding, and Selvan 2020)
  - based on RAPL and nvidia-smi
- Fine grained studies on a specific Jetson hardware (Rodrigues, Riley, and Luján 2018; Holly, Wendt, and Lechner 2020)
- Generic libraries from the data center community : Papi, Likwid
- Machine learning based prediction models (Cai et al. 2017, Jia et al. 2015)
- French Startup : https://github.com/hubblo-org

Hard to get recover exactly what you measure on your power meter.
Developping from scratch requires complex low level programming skills

RAPL to measure Intel CPUs

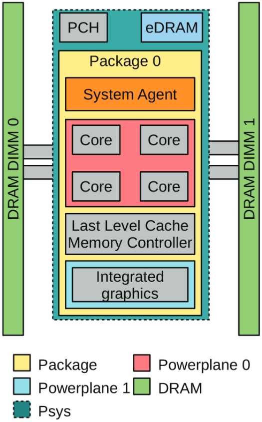
Running Average Power Limit

- Model based power estimation.
- Reports the accumulated energy consumption
- Recording at 1000Hz
- Requires administrator privilege

RAPL Organisation

Different counters for physically meaningfull domains:

- Power Plane 0 : CPU
- Power Plane 1 : Processor graphics on the socket.
- DRAM : energy consumption of the RAM
- Psys : System on Chip energy consumption



Access to RAPL measurements

- Model specific registers

  `/dev/cpu/core_id/msr`

  - Read MSR register bit by bit (not trivial)
  - See intel documentation (not trivial)
  - And activate the kernel module

    ```
    sudo modprobe msr
    ```

- **Linux**: Exposition of a sysfs tree with powercap

  Accumulation of energy consumption in Joules

  ```
  sudo chmod -R 755 /sys/class/powercap/intel-rapl/
  ```

nvidia-smi

NVIDIA System Management Interface, based on top of the NVIDIA Management Library (NVML)

- Gpu global statisics and memory usage per process

```
ansabere$ nvidia-smi -q -x
```

  - The power consumption is given for the entire board
  - +/- 5% accuracy of current power draw.

- Per process Average utilization values for streaming multiprocessors (SM)

```
ansabere$ nvidia-smi pmon # up  to  4  devices
# gpu        pid  type    sm   mem   enc   dec    command
# Idx          #   C/G     %     %     %     %    name
    0       1114     G     -     -     -     -    Xorg
    0       1289     G     -     -     -     -    gnome-shell
    0    1135553     C    76     0     -     -    python
```

nvidia-smi

NVIDIA System Management Interface, based on top of the NVIDIA Management Library (NVML)

- Gpu global statisics and memory usage per process

  ```
  ansabere$ nvidia-smi -q -x
  ```

  - The power consumption is given for the entire board
  - +/- 5% accuracy of current power draw.

- Per process Average utilization values for streaming multiprocessors (SM)

```
ansabere$ nvidia-smi pmon # up  to  4  devices
# gpu        pid  type    sm   mem    enc    dec    command
# Idx          #   C/G     %     %      %      %    name
    0       1114     G     -     -      -      -    Xorg
    0       1289     G     -     -      -      -    gnome-shell
    0    1135553     C    76     0      -      -    python
```

Deep Learning Power Measure @UPPA

We are developing a python module for :

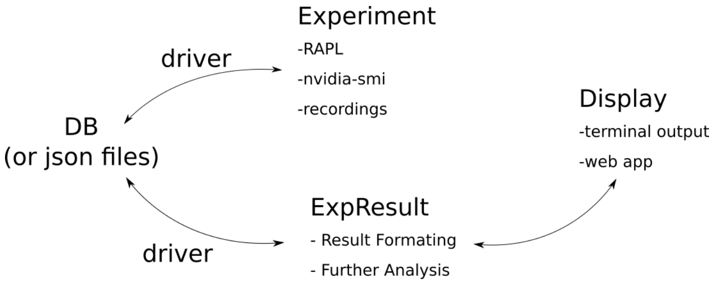
- Recording the power of a specific process
- Focus on accessibility and analysis for data scientist
- Model card, number of parameters and macs

```
process, queue = exp.measure_yourself(period=2)

 ###################
#  place here the code that you want to profile
################

q.put(experiment.STOP_MESSAGE)
```

Overview of the different modules

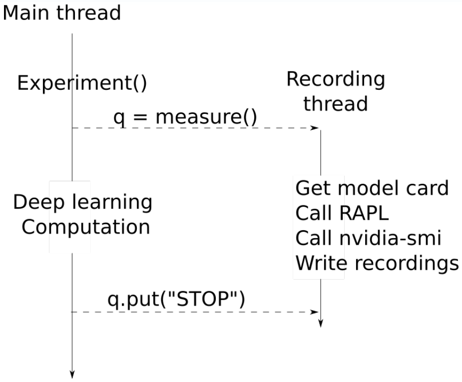


Getting the model card

```
net = ... the model you are using for your experiment
input_size = ... (batch_size, *data_point_shape)
exp = experiment.Experiment(driver, model=net,
            input_size=input_size)
```

- Pytorch module to obtain parameters and MAC number
- More generic principle of model card (Mitchell et al. 2019)

Multi threading under the hood



- Energy recording only for the main thread
- Queue to communicate between the threads

Mutli threading

```
def processify(func):
    def process_func(self, queue, *args, **kwargs):
        ... Exception handling there
        ret = func(self, queue, *args, **kwargs)


    @wraps(func)
    def wrapper(self, *args, **kwargs):
        queue = Queue()
        p = Process(target=process_func,
                    args=[self, queue] + list(args), kwargs=kwargs)
        p.start()
        return p, queue

@processify
def measure_yourself(self, queue, period=1)
    call rapl and nvidia-msi ...
```

Get power draw by process

- RAPL and nvidia-smi provides the global power consumption
- Using memory and processor usage from psutil to obtain the consumption by program
- However some of the components are shared from all programs.

Divide in equal parts? ignore these parts?

Experiment

Let's test a small network on a random synthetic image

- Energy consumed by 200K forward passes
- 1 convolutional layer with a (3×3) kernel
- input image is $(3 \times 128 \times 128)$

Energy consumed by one convolutional layer

batch size	1	10	100	1000	10000
MAC count	444K	4440K	44400K	444000K	4440000K
CPU	763J	7KJ	134KJ	1257KJ	5080KJ
cuda enabled : GPU	800J	3KJ	7KJ	81KJ	805KJ
cuda enabled : CPU	192J	331J	596J	7KJ	59KJ

- Nvidia still uses CPU power (and memory)
- GPU energy efficient because faster.

Overall, program duration is a good indicator for this experiment

Comparison between a convolutional and a linear layer

	MAC	energy (CPU + GPU)	time
Linear layer	49153K	1600J	8 sec.
Conv layer	44400K	7000J	21 sec.

- Linear layer with 10 outputs
- Batch size set to 200
- MAC and energy are not correlated in this example

Perspectives

Fine grained data center studies of deep learning practices

- Make the code usable
- Use it to discover how to measure computer power
- Support different types of hardware

A lot to discover for deep learning!

References I

Anthony, Lasse, Benjamin Kanding, and Raghavendra Selvan (July 2020). "Carbontracker: Tracking and Predicting the Carbon Footprint of Training Deep Learning Models". In: arXiv preprint https://arxiv.org/abs/2007.03051.

Bridges, Robert A, Neena Imam, and Tiffany M Mintz (2016). "Understanding GPU power: A survey of profiling, modeling, and simulation methods". In: **ACM Computing Surveys (CSUR)** 49.3, pp. 1–27.

Cai, Ermao et al. (2017). "Neuralpower: Predict and deploy energy-efficient convolutional neural networks". In: **Asian Conference on Machine Learning**. PMLR, pp. 622–637.

Henderson, Peter et al. (2020). "Towards the Systematic Reporting of the Energy and Carbon Footprints of Machine Learning". In: **ArXiv** abs/2002.05651.

References II

Holly, Stephan, Alexander Wendt, and Martin Lechner (2020). "Profiling Energy Consumption of Deep Neural Networks on NVIDIA Jetson Nano". In: **2020 11th International Green and Sustainable Computing Workshops (IGSC)**. IEEE, pp. 1–6.

Jia, Wenhao et al. (2015). "GPU performance and power tuning using regression trees". In: **ACM Transactions on Architecture and Code Optimization (TACO)** 12.2, pp. 1–26.

Mitchell, Margaret et al. (2019). "Model cards for model reporting". In: **Proceedings of the conference on fairness, accountability, and transparency**, pp. 220–229.

References III

Rodrigues, Crefeda Faviola, Graham Riley, and Mikel Luján (2018). "SyNERGY: An energy measurement and prediction framework for Convolutional Neural Networks on Jetson TX1". In: **Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)**. The Steering Committee of The World Congress in Computer Science, Computer . . ., pp. 375–382.