# Energy and Power Efficiency for Applications on the Latest NVIDIA Technology [S62419]

Alan Gray, NVIDIA

GTC 24, 20th March 2024

# Contributors

- Dmitry Alexeev
- Thomas Bradley
- Alan Gray
- Markus Hrywniak
- Ian Karlin
- Vishal Mehta
- Gabriele Paciucci
- Swapna Raj
- Louis Stuber
- Mathias Wagner

With thanks to Szilárd Páll (KTH/PDC) for discussions and feedback.

# HPC and AI Energy and Power

## Introduction

- Traditionally, the most important goal has been to minimize time to solution (or equivalently maximize performance).
- With increasing energy costs and environmental impact, it is becoming increasingly important to also consider energy minimization
  - **Energy = Power x Time**
  - Power must be considered in conjunction with time to solution.
  - *Minimizing energy to solution is exactly the same as maximizing Performance/Watt*
- NVIDIA GPUs can be configured to run at reduced clock frequencies, which effects power, time and hence energy.
  - It is important to consider not only GPU behaviour, but in the context of the server and datacenter.
- This presentation analyses the impact of tuning energy usage on a range of HPC and AI applications on modern NVIDIA GPU-accelerated servers.
  - We hope this is useful to help users to decide and apply the configuration that best suits their workload and goal.
- Beyond clock frequency tuning: application level choices can be assessed on how they impact performance and energy.
  - Explored through the GROMACS application
- Note: GTC 23 Presentation "Optimizing Energy Efficiency for Applications on NVIDIA GPUs", includes how-to commands https://www.nvidia.com/en-us/on-demand/session/gtcspring23-s52087/

# Key Findings

- Reducing clock frequency will decrease the power (and vice versa) while increasing the time to solution.
- Maximum frequency gives best performance, but not best energy.
- There exists a frequency sweet spot for best energy, for each application.
- Tuning for energy must be done in context of server and datacentre, since non-GPU power overheads are significant.
- Further energy tuning can be done by exploring application-level choices.
- Most often, optimizing apps to maximize performance will also minimize energy (at any chosen clock frequency).

# Energy Optimization

## Outline

- Overview of HPC and AI Application Benchmarks

- H100 GPU measurements
  - Time, GPU power and GPU energy variance with clock frequency on H100 systems for the range of applications

- DGX-A100 measurements
  - Comprehensive full-server measurements and analysis on a DGX server with 8xA100 GPUs for subset of apps

- H100 full-server estimates
  - Learnings from DGX-A100 applied to single-H100 measurements to estimate energy-saving potential for apps on typical multi-H100 server configurations

- Application-level choices in GROMACS

- Summary

# Overview of HPC and AI Application Benchmarks

# HPC and AI Application Benchmarks

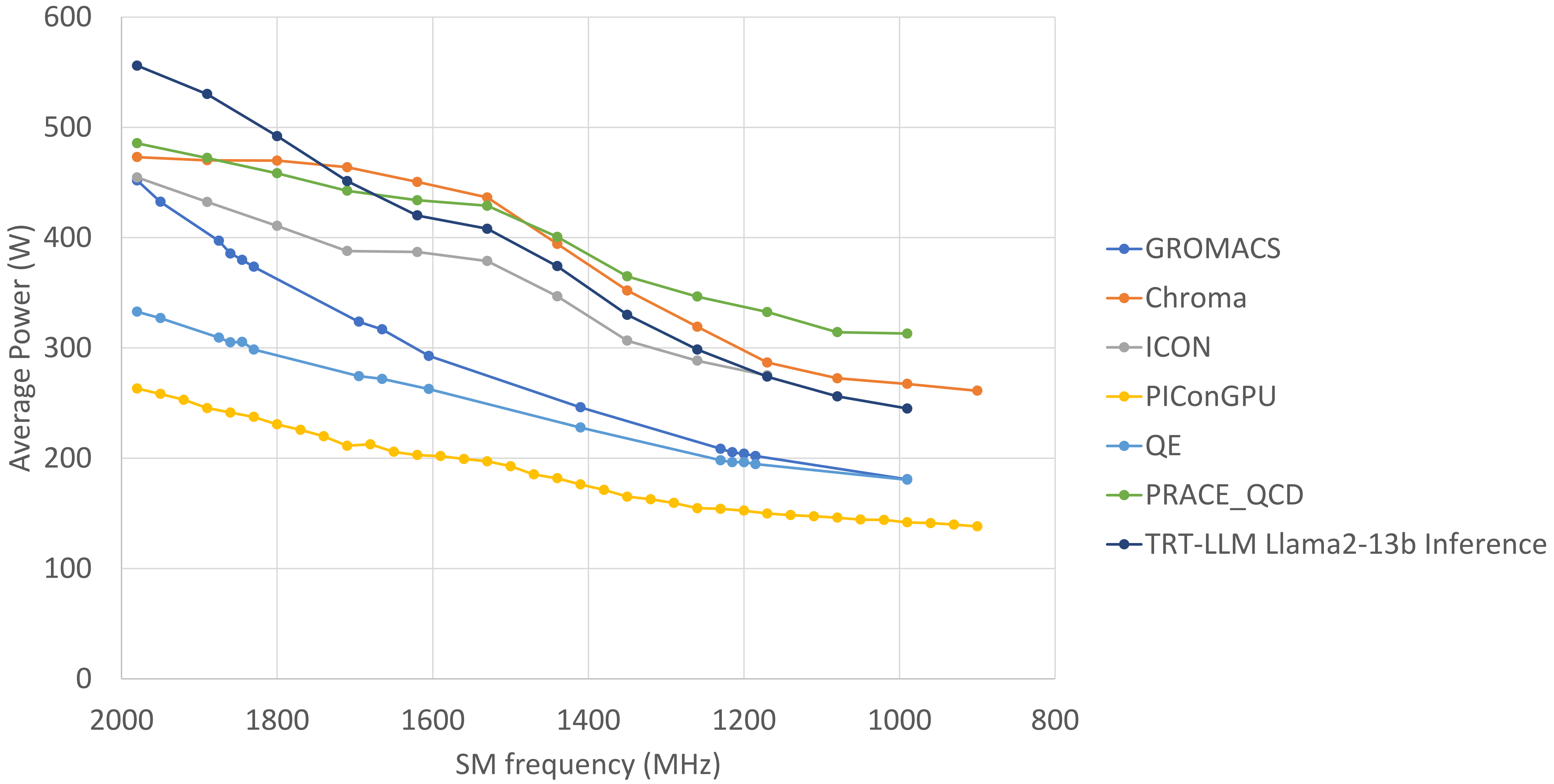
## Chosen to be representative of typical workloads

- Molecular Dynamics
  - **GROMACS** (https://www.gromacs.org/)
    - STMV workload. Mainly limited by on-GPU computations and associated instruction scheduling.
- Particle Physics (Lattice QCD)
  - **CHROMA** (https://jeffersonlab.github.io/chroma/)
    - HMC Medium workload. Mainly limited by HBM memory bandwidth.
  - **PRACE QCD** (https://repository.prace-ri.eu/git/UEABS/ueabs )
    - PRACE Unified European Applications Benchmark Suite QCD Part 1 workload, based on MILC kernels. Mainly limited by HBM memory bandwidth.
- Weather
  - **ICON** (https://www.dwd.de/EN/research/weatherforecasting/num_modelling/01_num_weather_prediction_modells/icon_description.html)
    - QUBICC R02B05 workload. Mainly limited by HBM memory bandwidth.
- Plasma Physics
  - **PIConGPU** (Particle in Cell) (https://github.com/ComputationalRadiationPhysics/picongpu)
    - SPEC 256^3 workload. Mainly limited by on-GPU computation, memory accesses and associated instruction scheduling.
- Quantum Chemistry (Density Functional Theory)
  - **Quantum Espresso (QE)** (https://www.quantum-espresso.org/)
    - TA205 workload. Alternating phases of compute-intensive linear algebra and HBM memory bandwidth intensive work.
- AI Inference
  - **TensorRT-LLM** (https://github.com/NVIDIA/TensorRT-LLM)
    - LLaMA2-13B model with input 2048, output 128, batch size 48, and 100 iterations (also include sweep through other variants). Limited by tensor-core compute and HBM memory bandwidth

# H100 GPU Measurements

# Application Power on H100

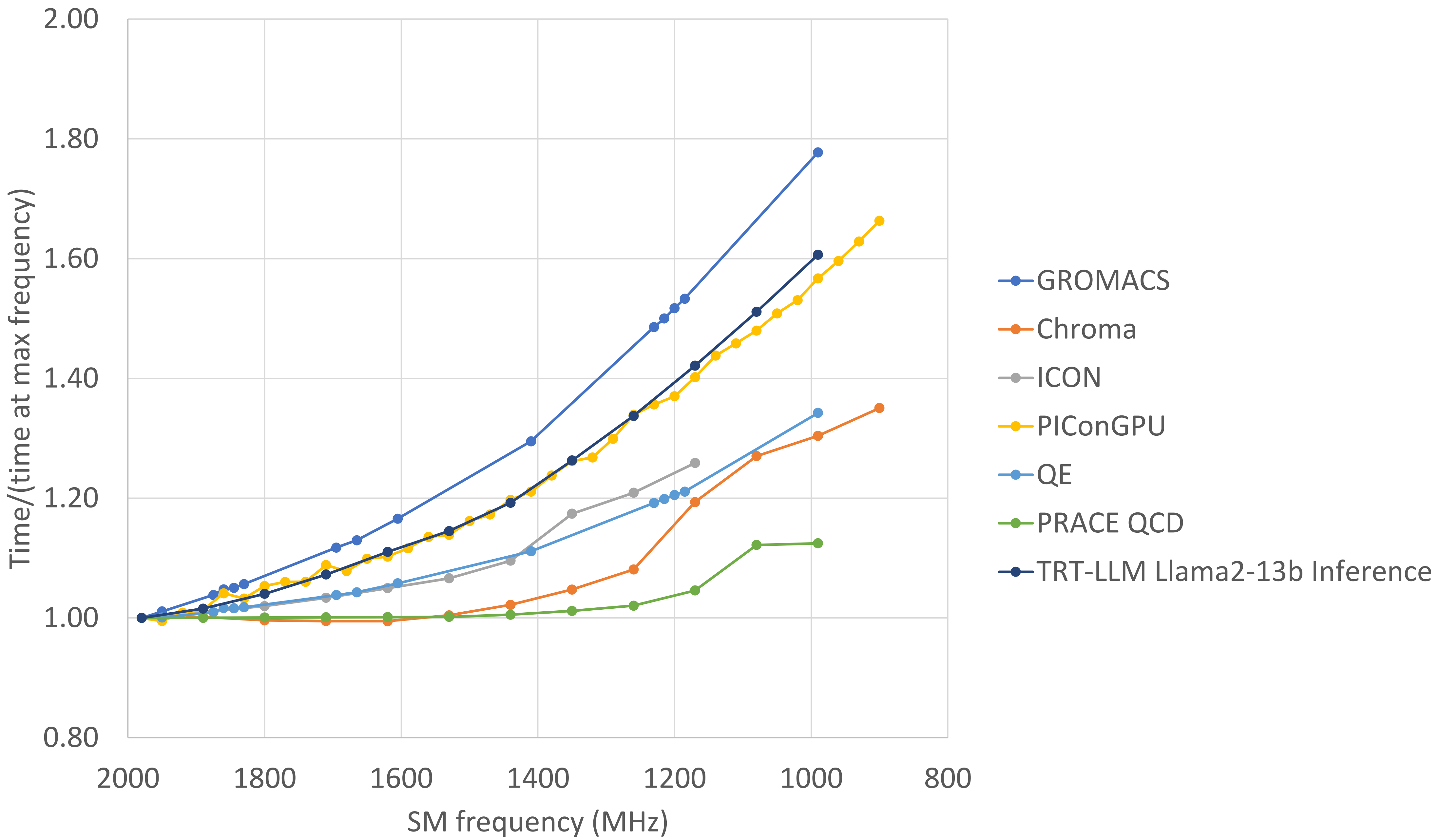
GPU power measured with decreasing GPU clock frequency




- GPU Power draw decreases with decreasing GPU clocks
- This behaviour must be considered together with walltime (next slide) to assess scope for reducing energy.
- Gradients and curves are app-dependent

# Application Walltime on H100

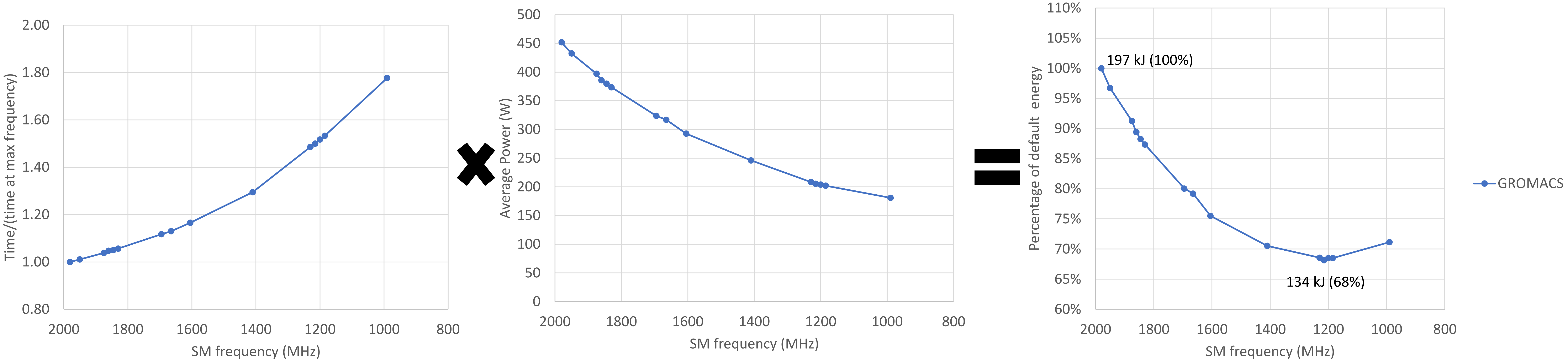
Normalized walltime with decreasing GPU clock frequency




- Walltime increases with decreasing GPU clock frequency
- Gradients/curves are app dependent
- Combined with previous power measurements, we can assess overall energy usage (Energy = Power x Time)

# GPU-only energy on H100 with reduced clock frequency

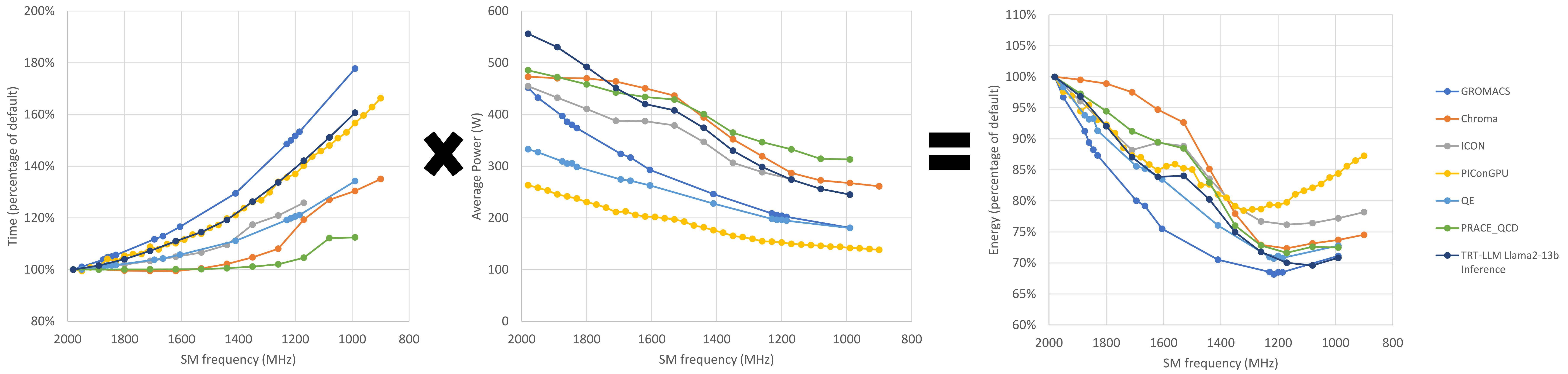
Time x Power = Energy




- We first show only a single app (GROMACS) for clarity
- Only 68% of default GPU energy used (i.e. 32% energy saving) by reducing SM frequency from 1980 MHz to 1200 MHz.

# GPU energy savings on 1xH100 with reduced clock

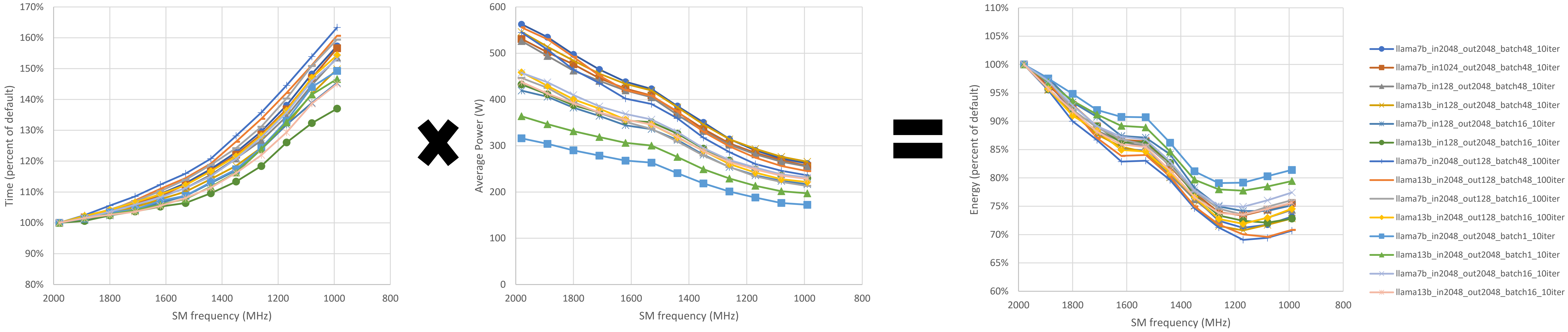
Time x Power = Energy




- GPU energy saving for all benchmarks at reduced frequency, in the range of 20-30%.
- Geomean GPU-only saving is 27.3%.
- Best-energy clock setting is similar across apps (around 1200MHz).
- HOWEVER: this is only GPU. Other non-GPU power/energy usage must also be factored in for holistic picture.

# TRT-LLM Inference GPU energy savings on 1xH100 with reduced clock

Time x Power = Energy

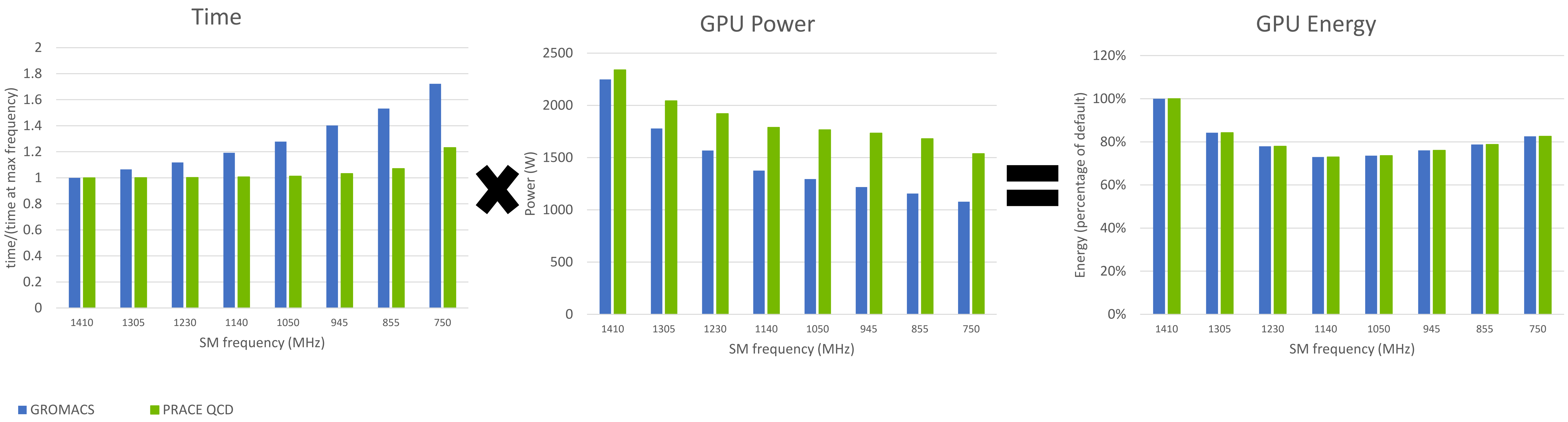



- Sweep of different options
- Energy savings available for all, with similar sweet spot.
- Larger energy savings with batching.

# DGX-A100 Measurements

# GPU-only energy with reduced clock frequency for 8xA100 on DGX

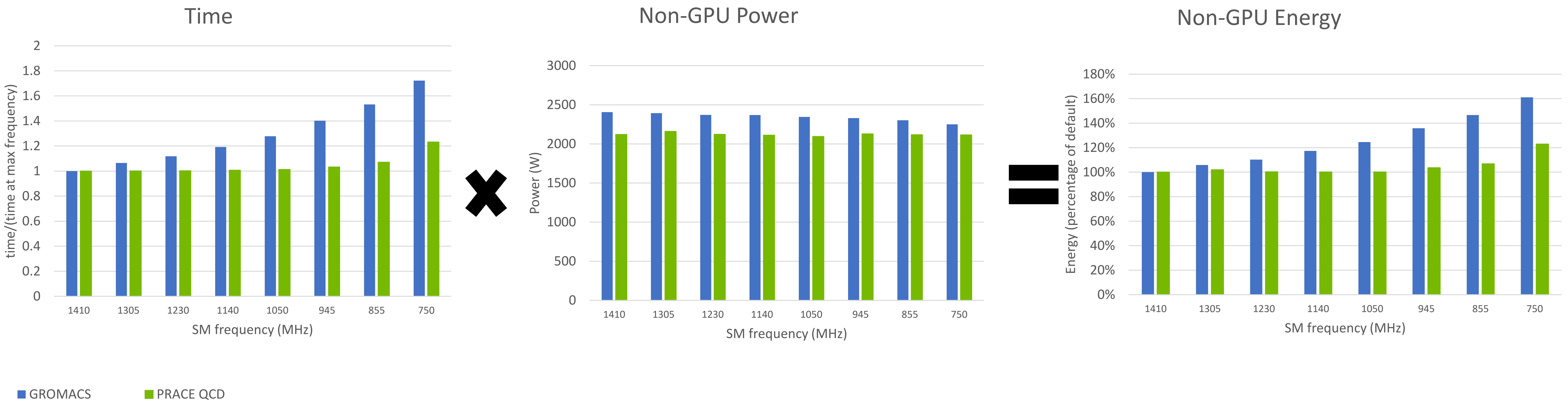
Time x Power = Energy




- Consider subset of 2 apps: GROMACS and PRACE QCD
- For each app, ensemble of 8 jobs across 8 A100 GPUs (and 2xAMD Rome CPUs) to fully saturate server
- When only considering GPU power (and hence energy), we observe ~25-30% energy savings at 1050 MHz

# Non-GPU energy with reduced GPU clock frequency

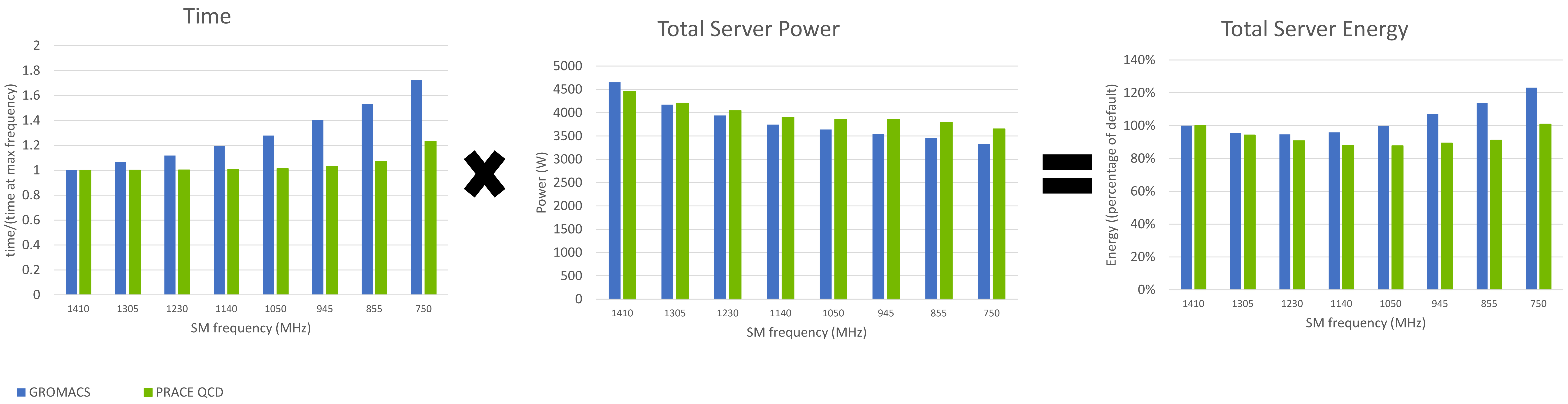
Time x Power = Energy




- Measured total server PSU power, with GPU power subtracted
- Non-GPU power draw is higher than GPU power draw, and is largely constant with decreasing GPU clock
- When combined with increasing walltimes (due to decreased GPU clock), results in app-dependent energy increases.

# Total server energy with reduced GPU clock frequency

Time x Power = Energy

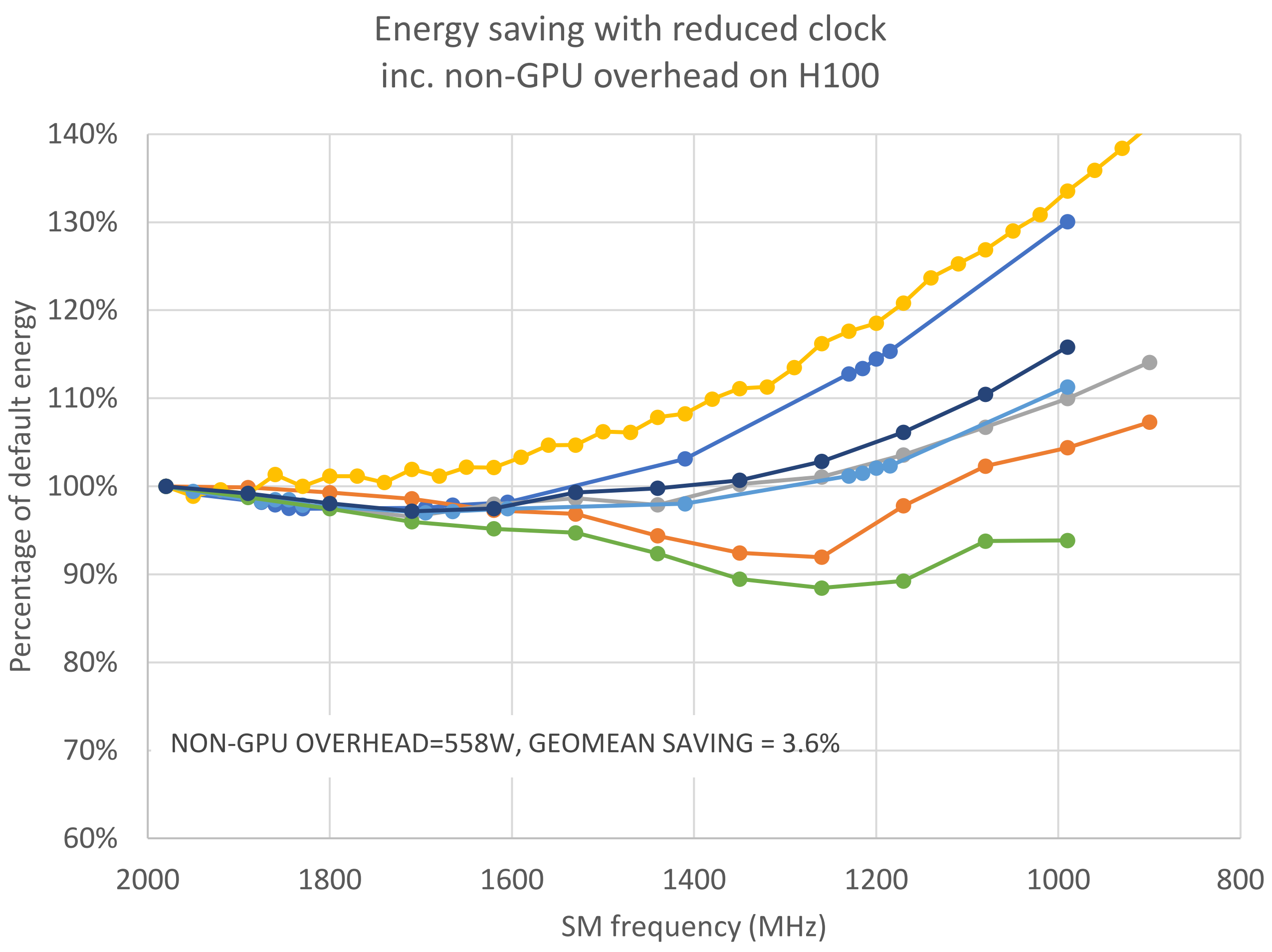



- Combination of GPU-only and non-GPU for total server energy.
- We still have energy savings, but non-GPU power draw is reducing overall impact.
- Non-GPU impact worse for GROMACS, due to walltime sensitivity to reduced clock.
- Best-energy frequency is now shifted and not consistent across apps.
- As we will now discuss, typical modern HPC server will have less non-GPU impact and better overall savings.

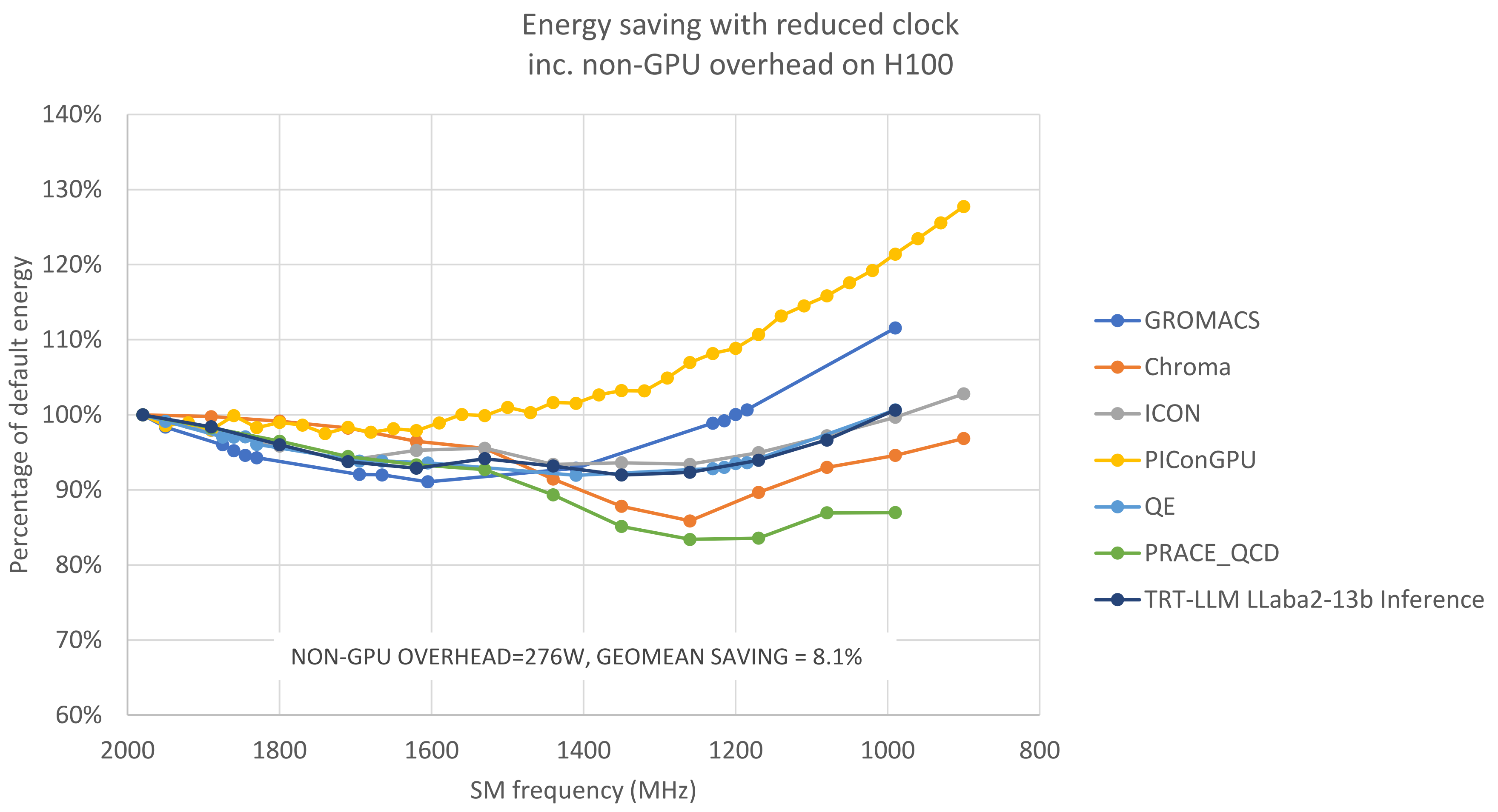
# H100 Full Server Estimates

# H100 HPC Server Energy Saving Estimates

Time x Power = Energy



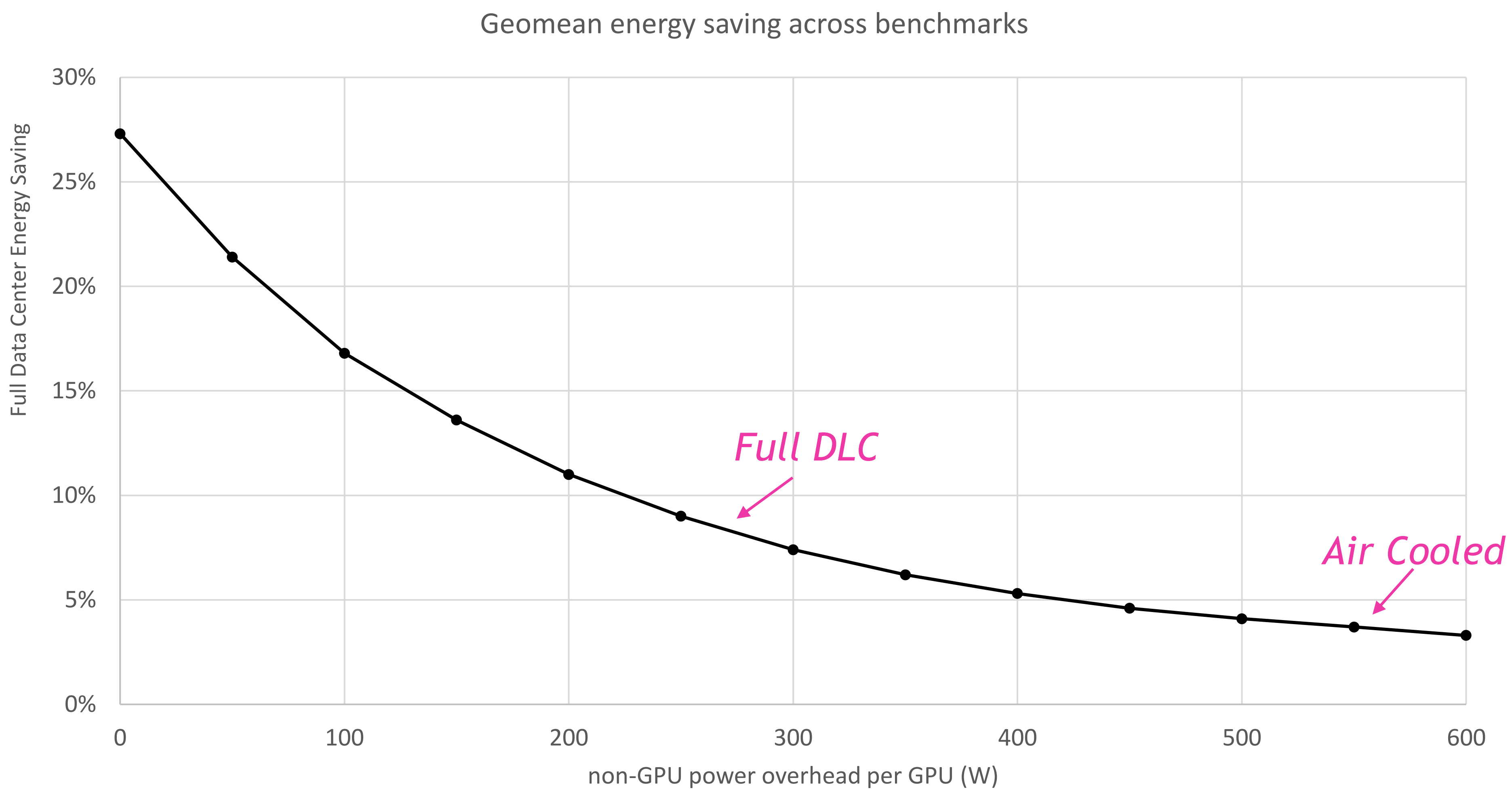

*Air Cooled*




*Full DLC*

- We estimate non-GPU power overheads for Air Cooling and Direct Liquid Cooling (DLC), including all components in server and datacentre.
  - See https://www.nvidia.com/en-us/on-demand/session/gtcspring23-s52087/
- We calculate adjusted energy saving characteristics, including these overheads
- We can also calculate the geomean energy saving across apps for the full range of power overheads

# Energy savings dependent on non-GPU Power




- Based on DGX-A100 measurements, we have modelled power profile for several HGX-H100 server configurations. Includes typical non-server overheads in datacenter
- Overall saving strongly depends on (constant power) non-GPU overheads. Energy savings maximized when
  - Non-GPU power minimized
  - Non-GPU power can ramp down in a similar way to GPU power
- Liquid cooling has a strong benefit in reducing energy utilization
- Best clock is dependent on workload (must be tuned)

# Application-level choices - GROMACS

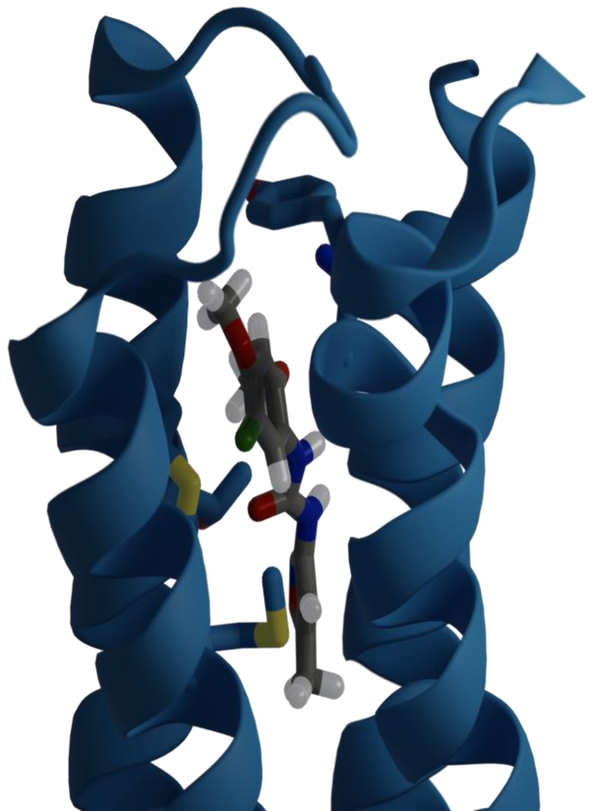
# GROMACS GPU APP-LEVEL CHOICES

- Simulation package for biomolecular systems - one of the most highly used scientific software applications worldwide, and a key tool in understanding important biological processes.
  - https://www.gromacs.org/
  - https://developer.nvidia.com/blog/tag/gromacs/
- Evolves systems of particles through repeated updates based on forces.
- Users can choose which components are offloaded to GPU at runtime
  - Non-bonded short-range forces (NB)
    - Most demanding force calculations - minimal required for GPU-accelerated GROMACS
  - Particle Mesh Ewald long-range forces (PME)
  - Bonded Forces (Bonded)
  - Update and Constraints (Update)
- *PME, Bonded* and *Update* can be independently offloaded, each depending on *NB* offload. Performance and energy of such choices will be assessed.

Also:

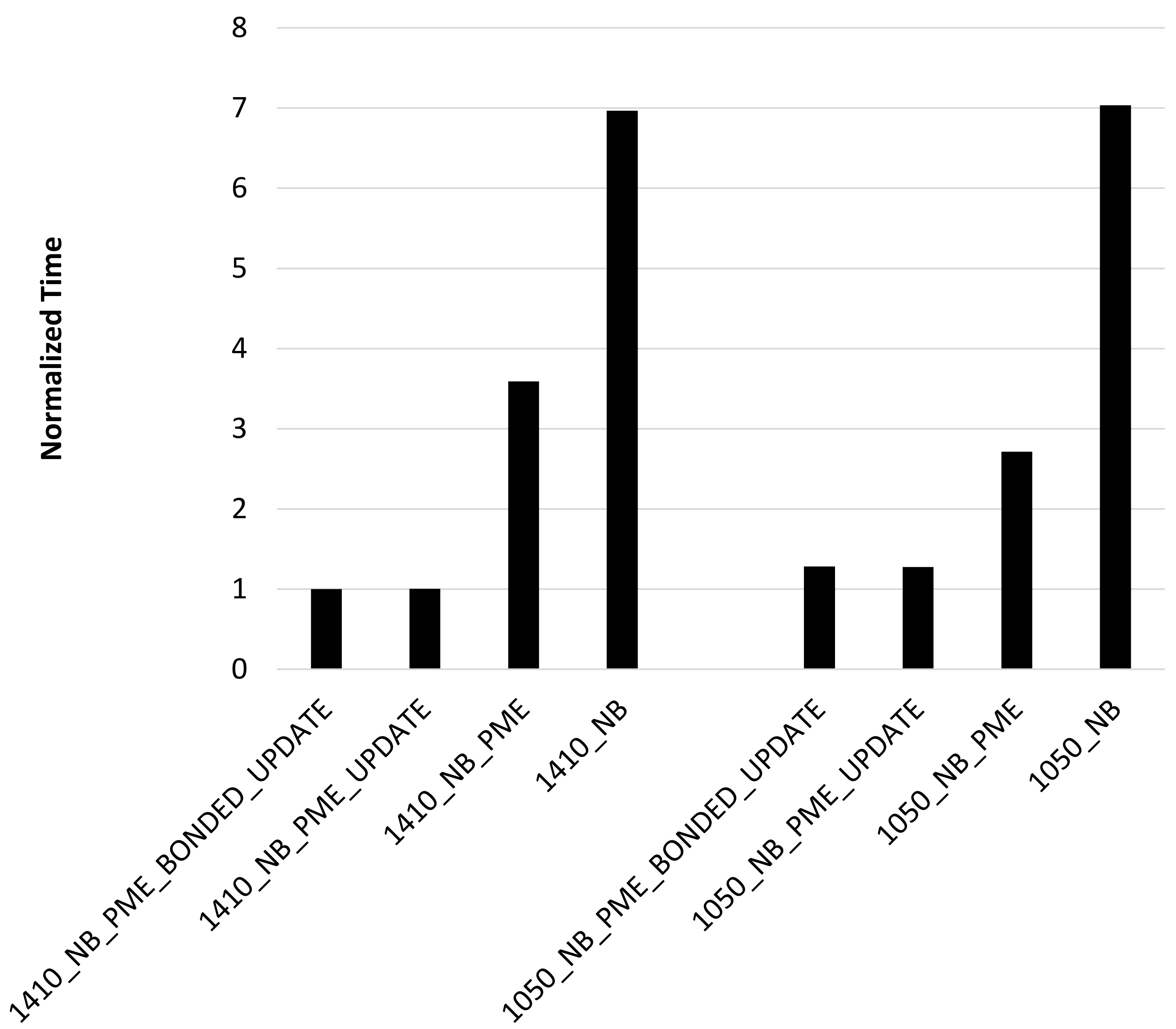
- Choice of neighbour search frequency
- Choice of tabulated or analytical Ewald non-bonded kernels

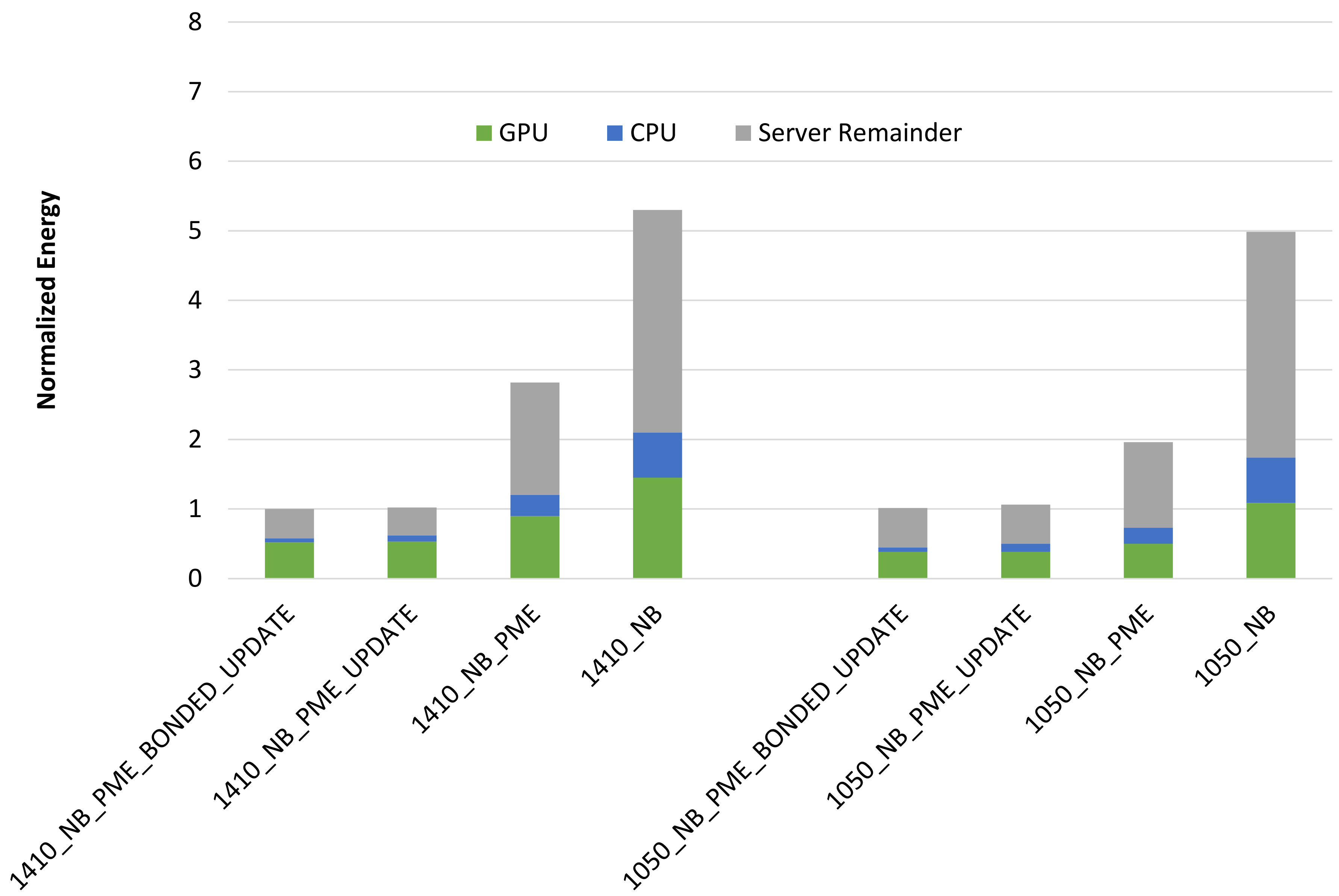
All results are for STMV benchmark.



# GROMACS Time and Energy on DGX-A100

Label: clock-frequency_offloaded-parts



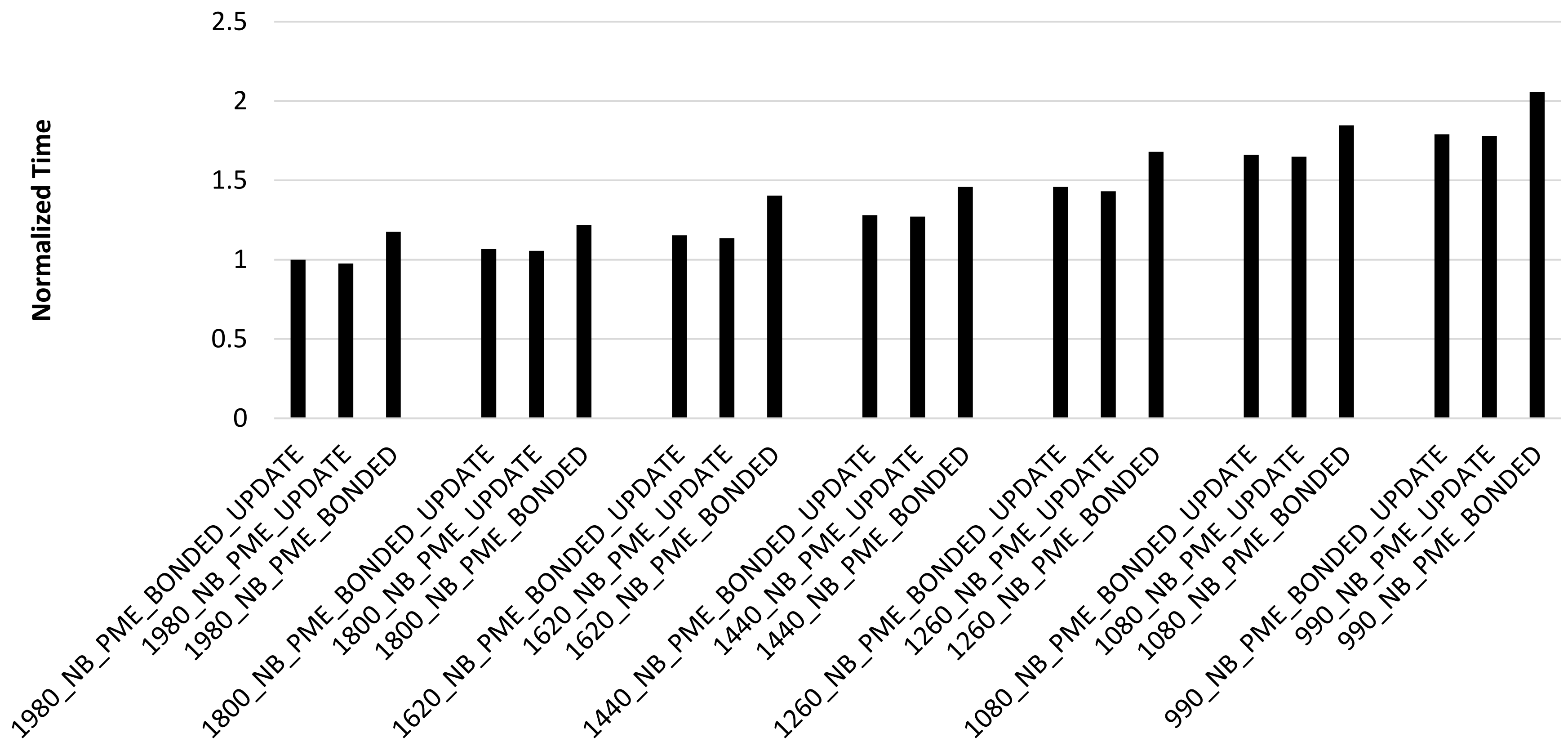



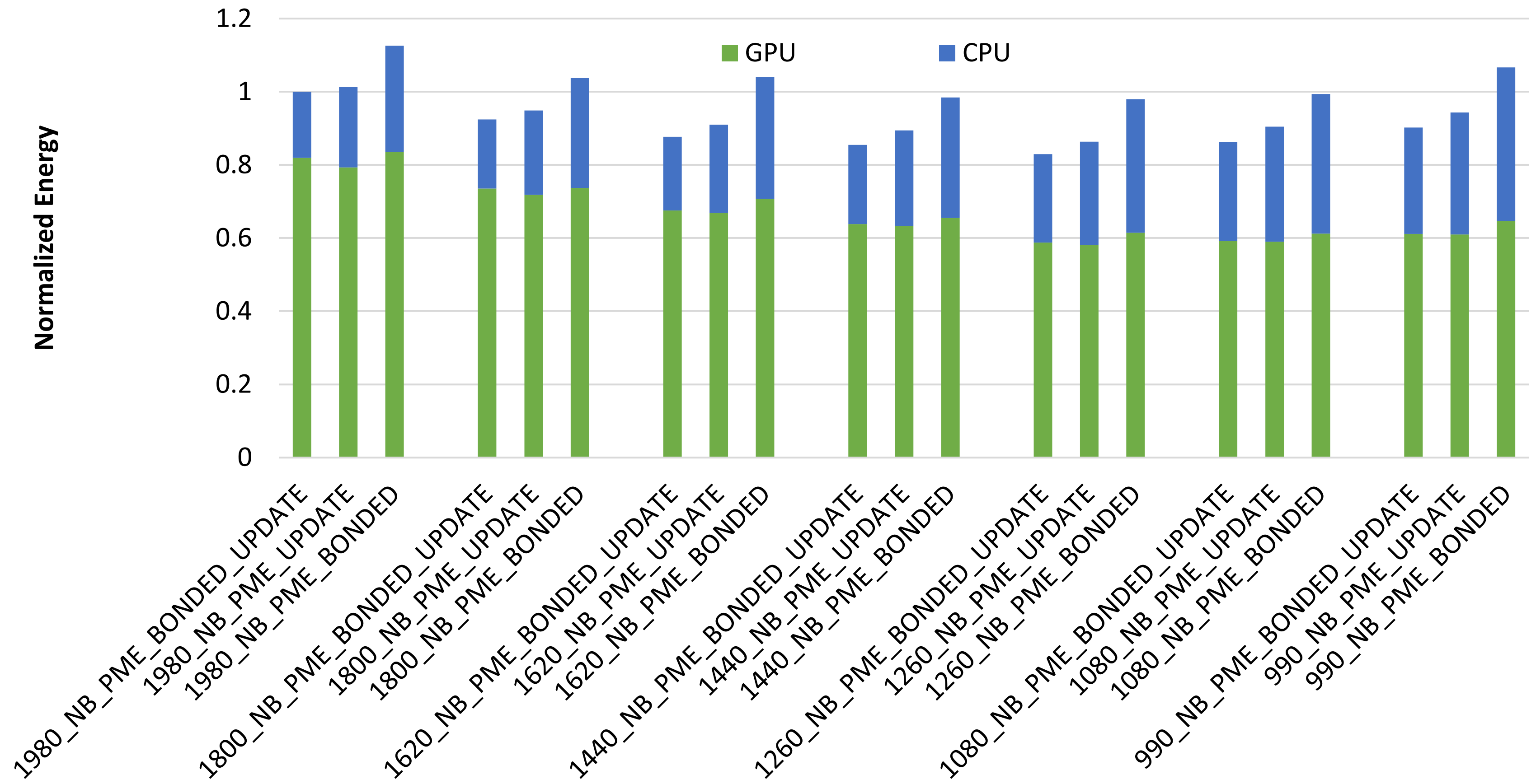

- Running PME or Update on CPU is a lot slower and a huge waste of energy
- Running Bonded on CPU or GPU is a close-call in time and energy.
- Choice which minimizes runtime also minimizes energy.

# GROMACS Time and Energy on Grace+Hopper

## Energy is GPU + CPU (and respective memories) only

- Grace+Hopper (GH200) is NVIDIA's newest product with NV ARM CPU and H100 GPU.
  - Very high bandwidth NVLINK C2C CPU-GPU interconnect (vs PCIe)
  - 72 ARM cores per H100 (vs 16 X86 cores per A100 for Selene results).
  - This test case is around 2X faster than X86+A100.

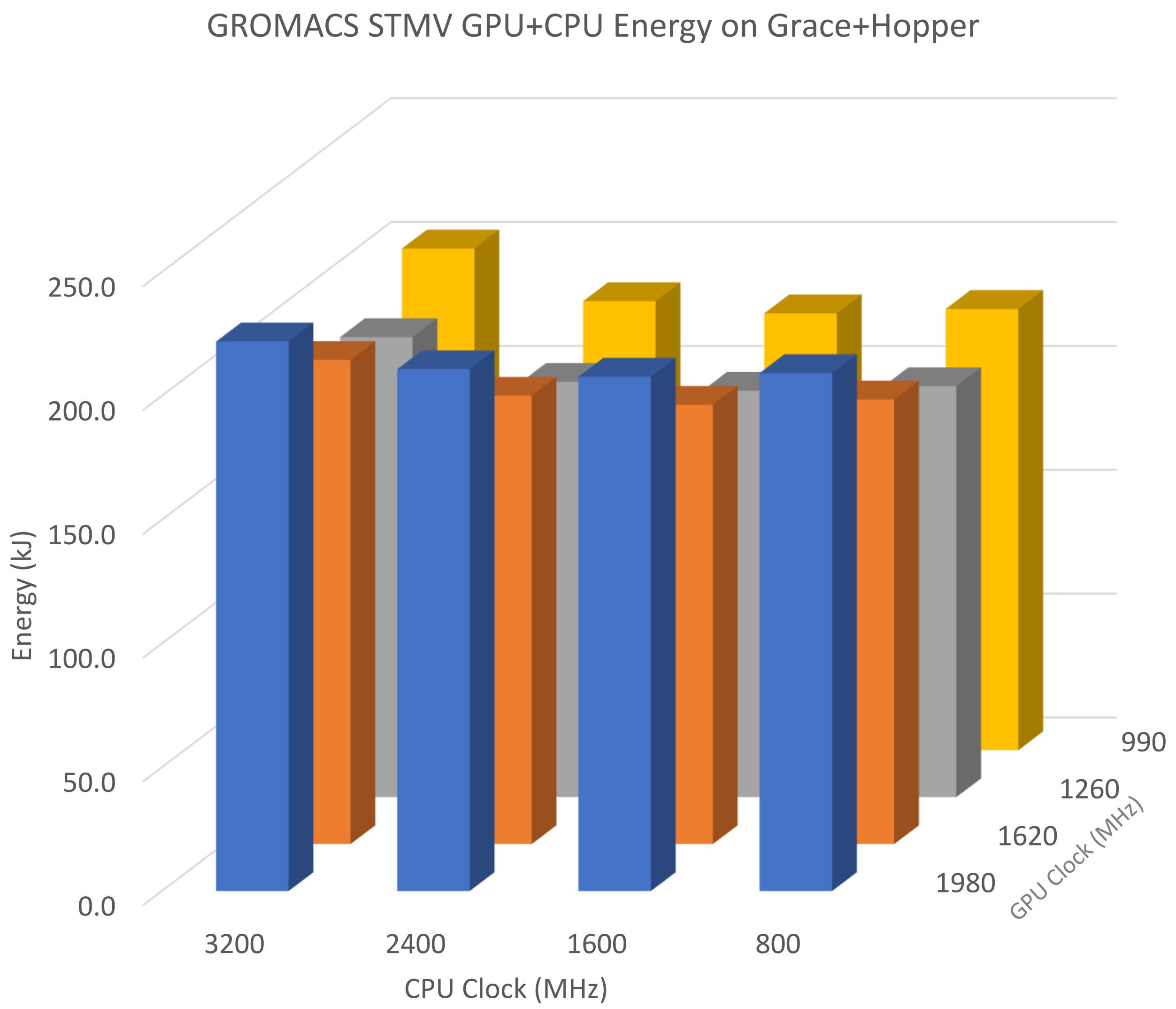



- Update on CPU has less of a disadvantage, due to C2C and more CPU capability per GPU (but still slower than GPU with higher energy).
- At energy-efficient 1260 MHz, bonded on CPU is slightly faster but higher energy (due to CPU load)
  - User choice between runtime and energy minimization.

# Tuning both GPU and CPU clocks on Grace+Hopper

## GROMACS STMV




- CPU clock frequency provides another tunable parameter
- Overall best energy for this case is at CPU:1600 MHz GPU: 1260 MHz.

# GROMACS Tabulated vs Analytical Ewald NB kernels

## Performance and Energy of Algorithmic Choice

- For non-bonded (NB) force calculations on GPU, GROMACS has the option of using tabulated (TAB) or analytical (ANA) Ewald kernels.
- TAB uses tabulated data which is read from cache (more memory loads), while ANA recalculates the data (more FLOPS).

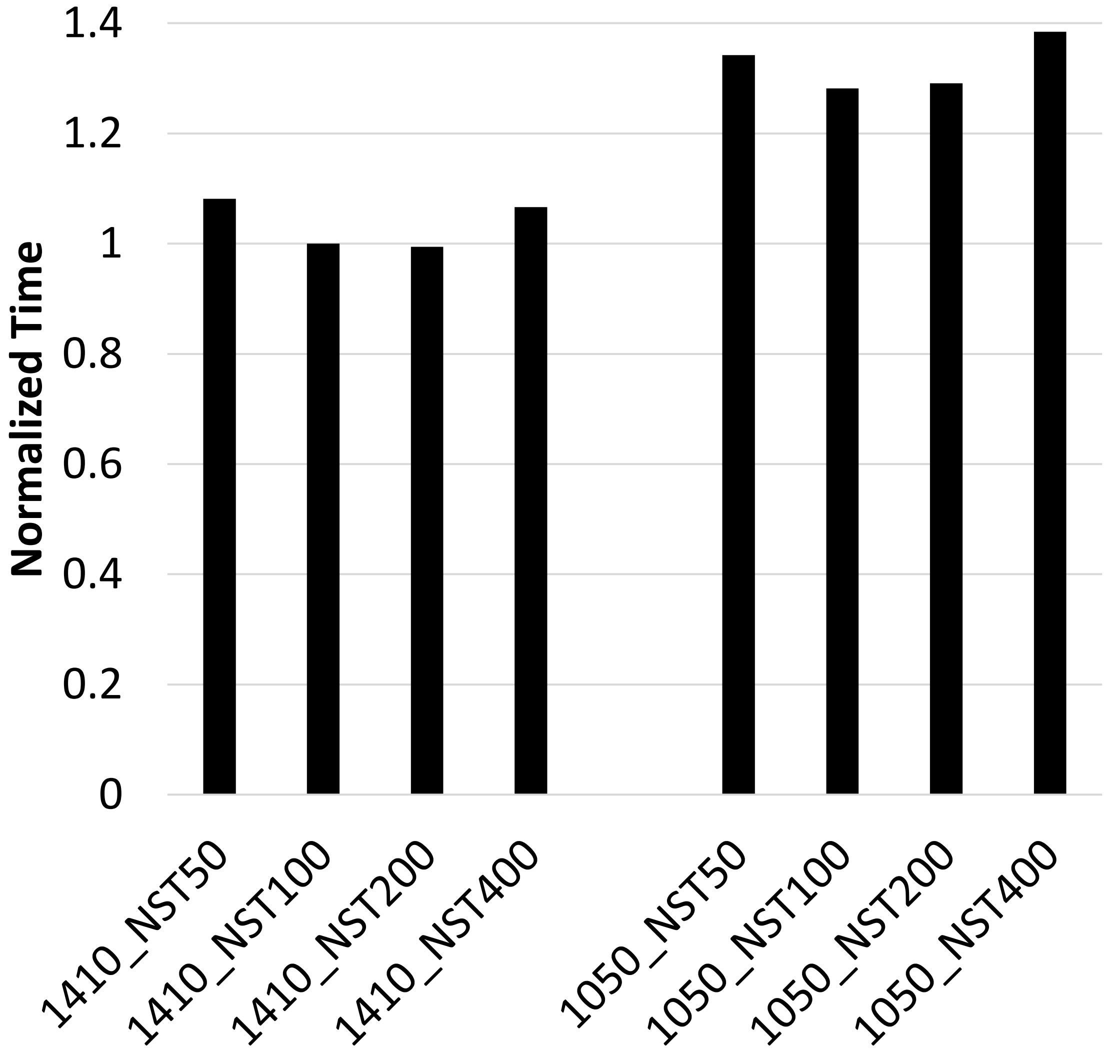
| | H100 time | H100 energy | A100 time | A100 energy | L40S time | L40S energy | A40 time | A40 energy |
|---|---|---|---|---|---|---|---|---|
| ADHD | 0.994 | 0.961 | 0.999 | 0.985 | 0.993 | 0.951 | 1.000 | 0.970 |
| EAG1 | 1.007 | 0.962 | 0.998 | 1.010 | 1.000 | 0.978 | 1.000 | 0.989 |
| STMV | 1.006 | 0.954 | 1.019 | 1.007 | 0.996 | 0.992 | 0.978 | 0.978 |
| grompp-fsw | 1.008 | 0.970 | 1.034 | 1.050 | 0.966 | 0.970 | 0.976 | 0.972 |
| grompp-fsw_rc1.2 | 1.009 | 0.957 | 1.046 | 1.026 | 0.970 | 0.972 | 0.972 | 0.976 |
| grompp-psh | 0.994 | 0.965 | 1.086 | 1.085 | 0.948 | 0.945 | 0.954 | 0.949 |
| grompp-psw | 0.997 | 0.979 | 1.039 | 1.025 | 0.975 | 0.979 | 0.972 | 0.972 |

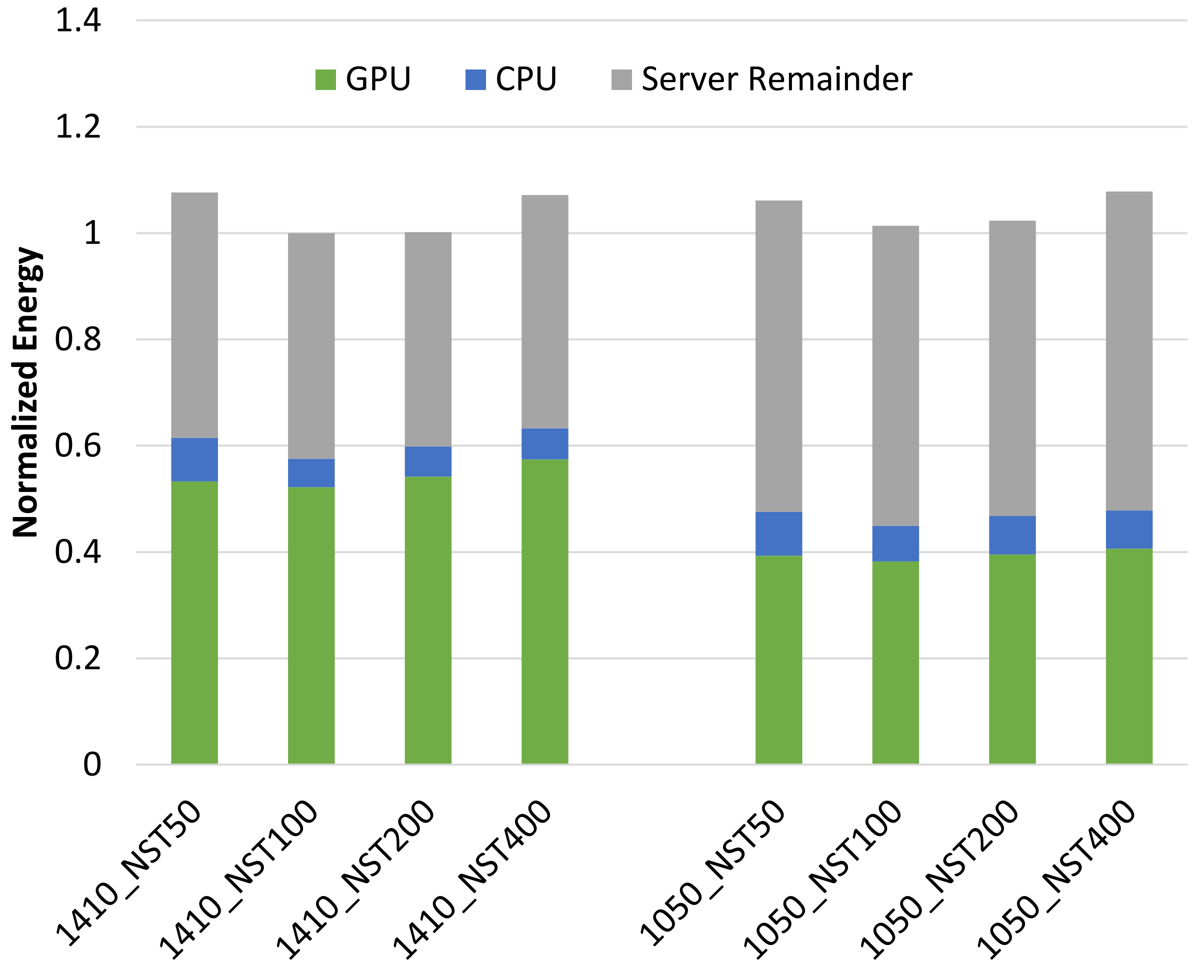
- Benefit of TAB over ANA. >1 (green) means TAB better, <1 (red) means ANA better. Grey means less than 2% difference.

- TAB is better for A100, and ANA is better for other architectures (which have extra floating point throughput per SM to handle extra FLOPS).
- H100 interesting, since no significant effect on time, but significantly lower energy with ANA
- https://gitlab.com/gromacs/gromacs/-/issues/4778

# GROMACS Neighbour Search Frequency

## Performance and Energy of Algorithmic Choice




- Nstlist: tunable runtime option to specify number of steps between neighbour list generation.
- **Tuning nstlist for time/performance also tunes for energy**

# Summary

# Summary

- Reducing GPU clock frequency
  - Increases runtime
  - Decreases power
  - Impacts Energy = Power x Time (equivalently Performance/Watt)
- Large GPU-only energy savings are available by finding the frequency sweet spot
- Inclusion of non-GPU power draw reduces the energy-saving impact, but it remains significant.
- Overall (full data center) energy saving can be maximised through minimizing non-GPU power usage
  - In particular, Direct Liquid Cooling offers a large benefit to the energy-saving potential.

**Technology providers:** strive to minimize the power consumed by all the components in the server and data center. Allow power draw for all components to reduce in line with GPU.

**Users/admins:** for any specific workload, vary GPU clock frequency, measure power and walltime, and calculate energy to find the sweet-spot. Power must include that from non-GPU components.

Application level choices:

- In vast majority of cases, choices which maximize performance will also minimize energy (due to minimizing time and energy wasted due to power overheads).
- Where choices have similar performance, fine tuning of energy optimization is possible through e.g. minimizing CPU computation or favouring computation over memory loads on GPU. Experimentation necessary.

NVIDIA®