Energy to Solution vs Time to Solution, towards energy-aware HPC applications

E. Calore[1] S. F. Schifano[1] R. Tripiccione[1]

[1] INFN Ferrara and Università degli Studi di Ferrara, Italy

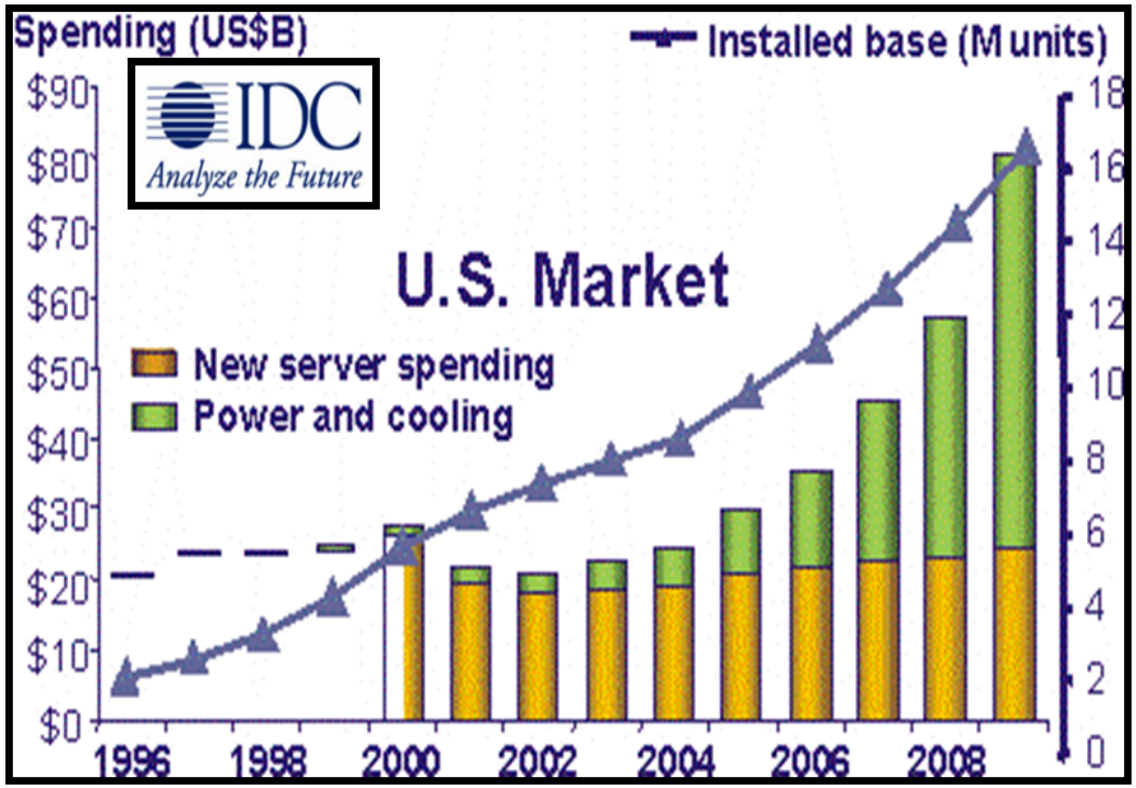
Workshop CCR

La Biodola - 19th May, 2016

Outline

Outline

Energy is becoming more and more important in HPC



HPC facilities may start to account for consumed energy instead of running time

Two research approaches...

Use low-power/embedded hardware for HPC

- may consume less since hardware is designed to be low-power
- may also cost less thanks to economy of scale

Minimize energy consumption on actual high-end systems

- may be possible using new energy monitoring / control hardware
- may be possible by software optimization / tuning

Two research approaches...

Use low-power/embedded hardware for HPC

- may consume less since hardware is designed to be low-power
- may also cost less thanks to economy of scale

Minimize energy consumption on actual high-end systems

- may be possible using new energy monitoring / control hardware
- may be possible by software optimization / tuning

Outline

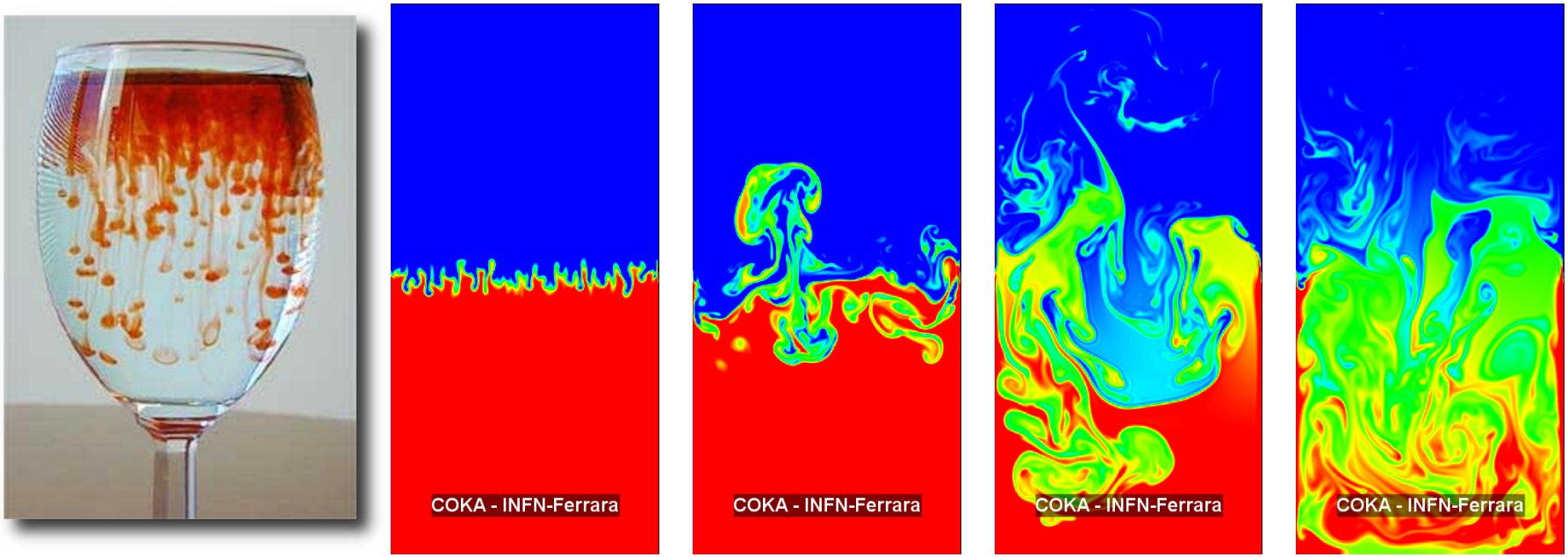
The D2Q37 Lattice Boltzmann Model

- Lattice Boltzmann method (LBM) is a class of computational fluid dynamics (CFD) methods
- LBM methods simulate a discrete **Boltzmann** equation, which under certain conditions, reduce to the **Navier-Stokes** equation
- **virtual particles** called **populations** arranged at edges of a discrete and regular grid are used to simulate a synthetic and simplified dynamics
- the interaction is implemented by two main functions applied to the virtual particles: **propagation** and **collision**
- D2Q37 is a D2 model with 37 components of velocity (populations)
- suitable to study behaviour of **compressible** gas and fluids optionally in presence of **combustion** [1] effects
- correct treatment of Navier-Stokes, heat transport and perfect-gas ($P = \rho T$) equations

[1] chemical reactions turning cold-mixture of reactants into hot-mixture of burnt product.

Simulation of the Rayleigh-Taylor (RT) Instability

Instability at the interface of two fluids of different densities triggered by gravity.



A cold-dense fluid over a less dense and warmer fluid triggers an instability that mixes the two fluid-regions (till equilibrium is reached).

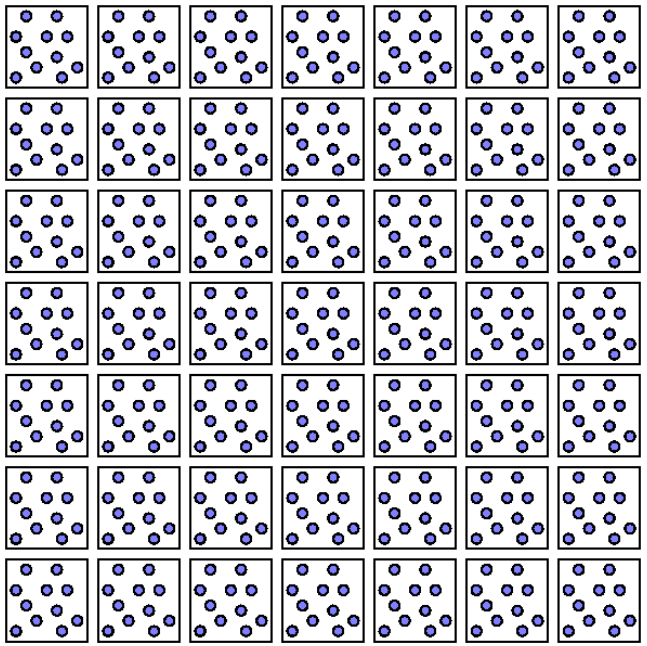
Computational Scheme of LBM

```
foreach time-step

  foreach lattice-point
    propagate();
  endfor

  foreach lattice-point
    collide();
  endfor

endfor
```



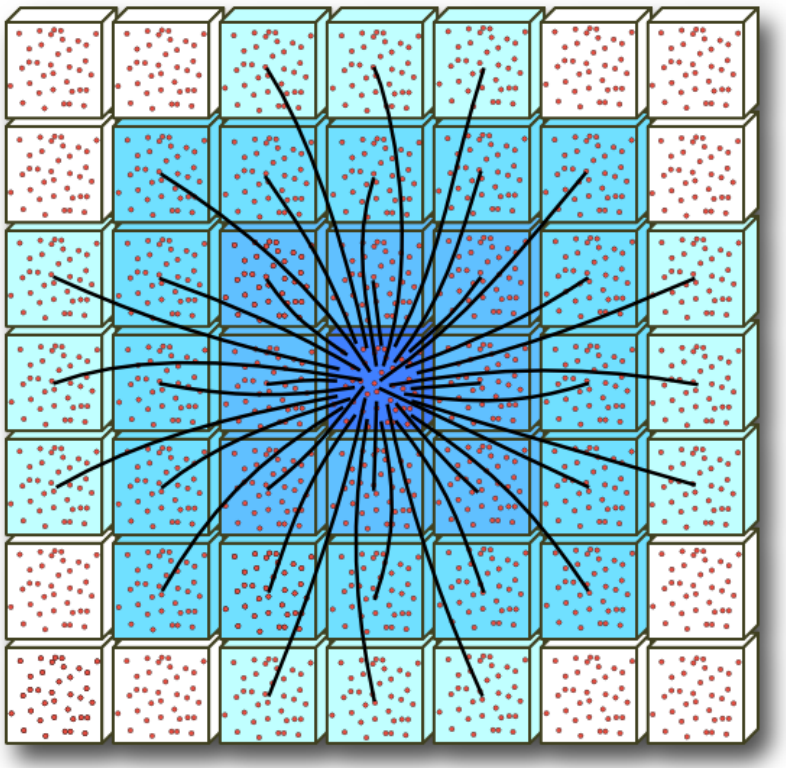
Embarassing parallelism

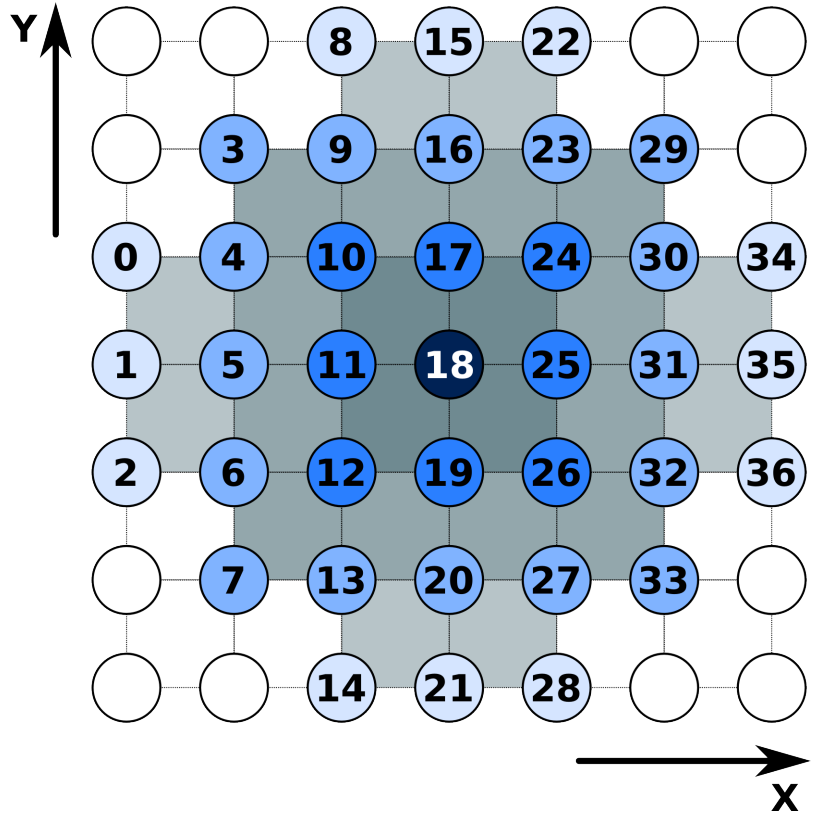
All sites can be processed in parallel applying in sequence propagate and collide.

Challenge

Design an efficient implementation able exploit a large fraction of available peak performance.

D2Q37: propagation scheme



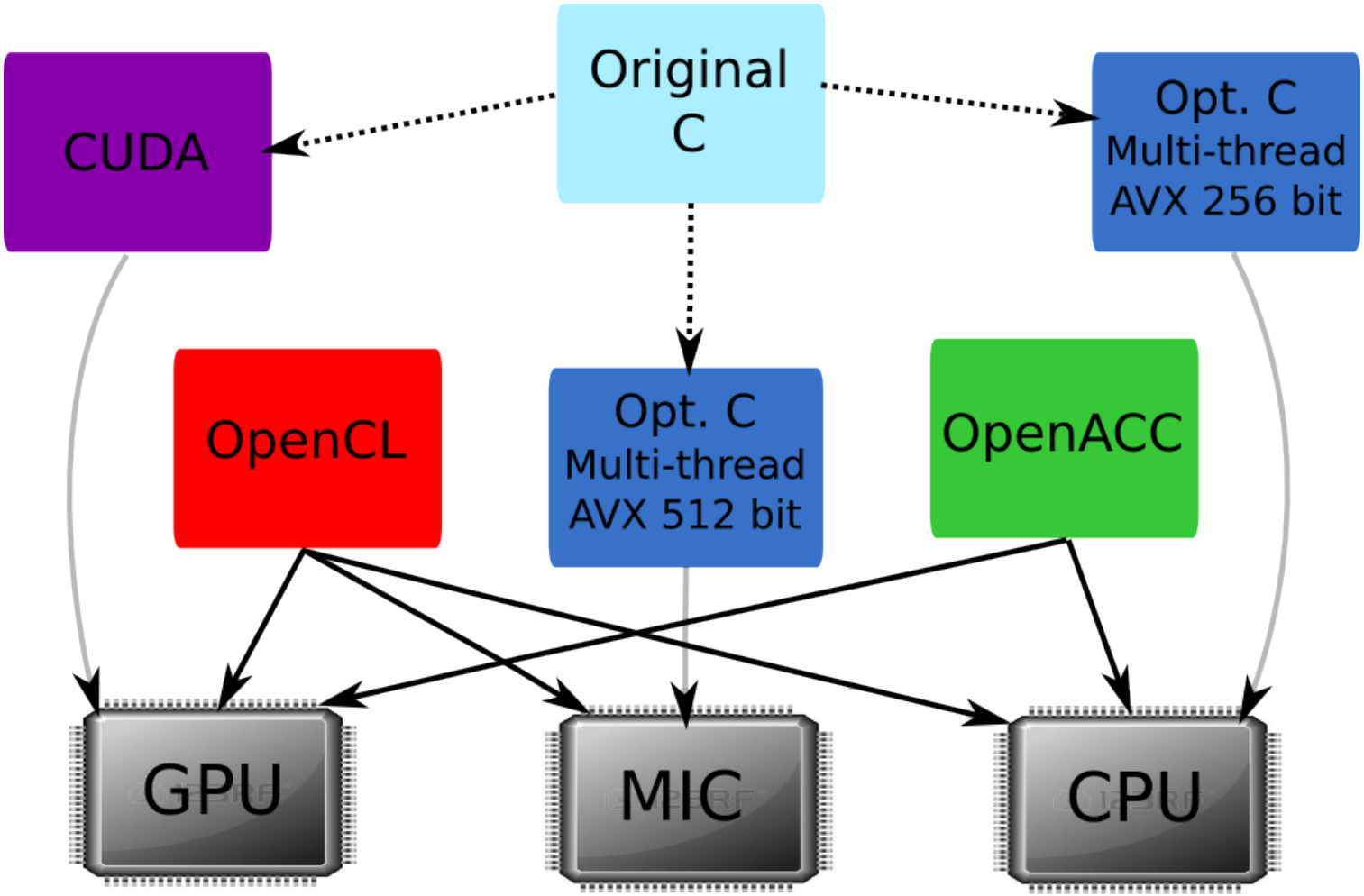


- perform accesses to neighbour-cells at distance 1,2, and 3
- generate memory-accesses with **sparse** addressing patterns

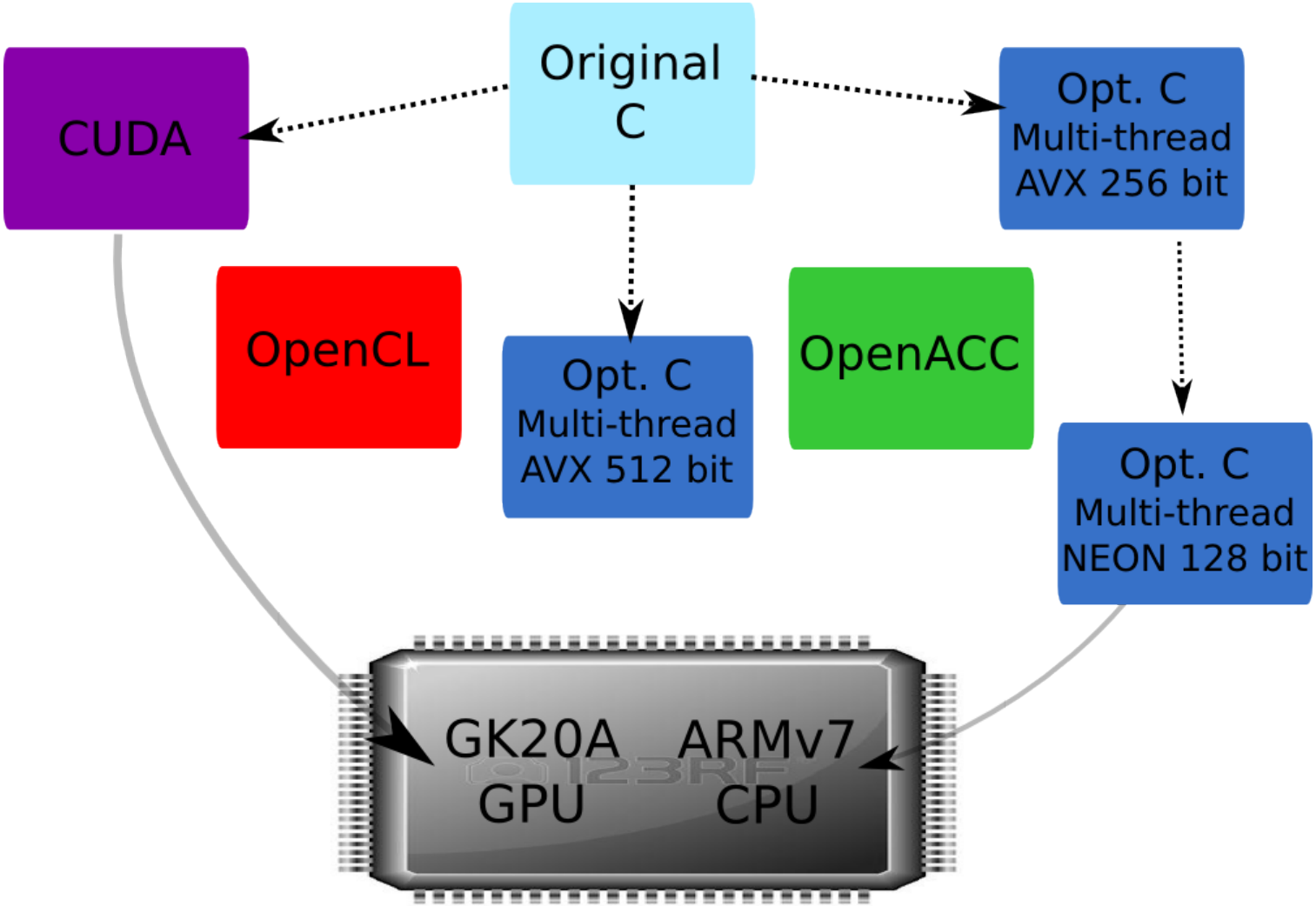
D2Q37 collision

- collision is computed at each lattice-cell after computation of boundary conditions
- computational intensive: for the D2Q37 model requires $\approx$ 7500 DP floating-point operations
- completely local: arithmetic operations require only the populations associate to the site
- computation of propagate and collide kernels are kept separate
- after propagate but before collide we may need to perform collective operations (e.g. divergence of of the velocity field) if we include computations conbustion effects.

Initial Code implementations

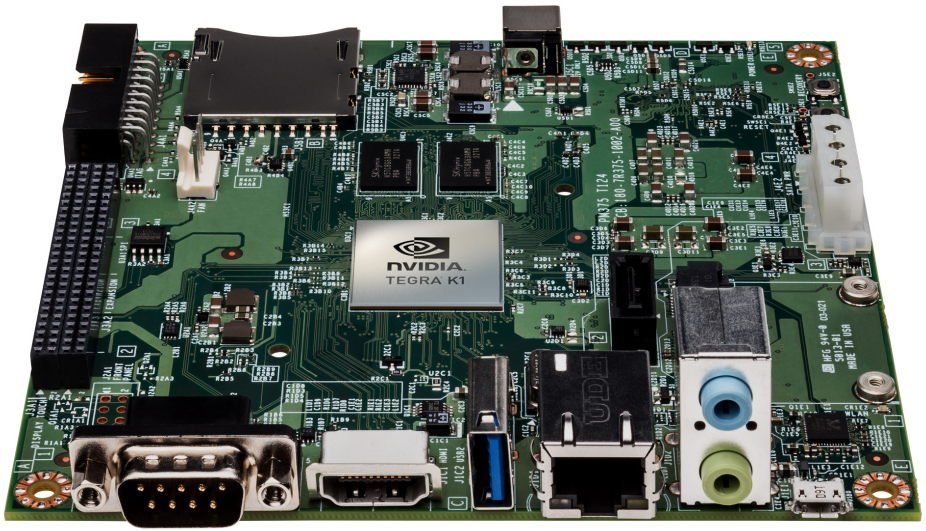


Code implementations



Outline

NVIDIA Jetson TK1



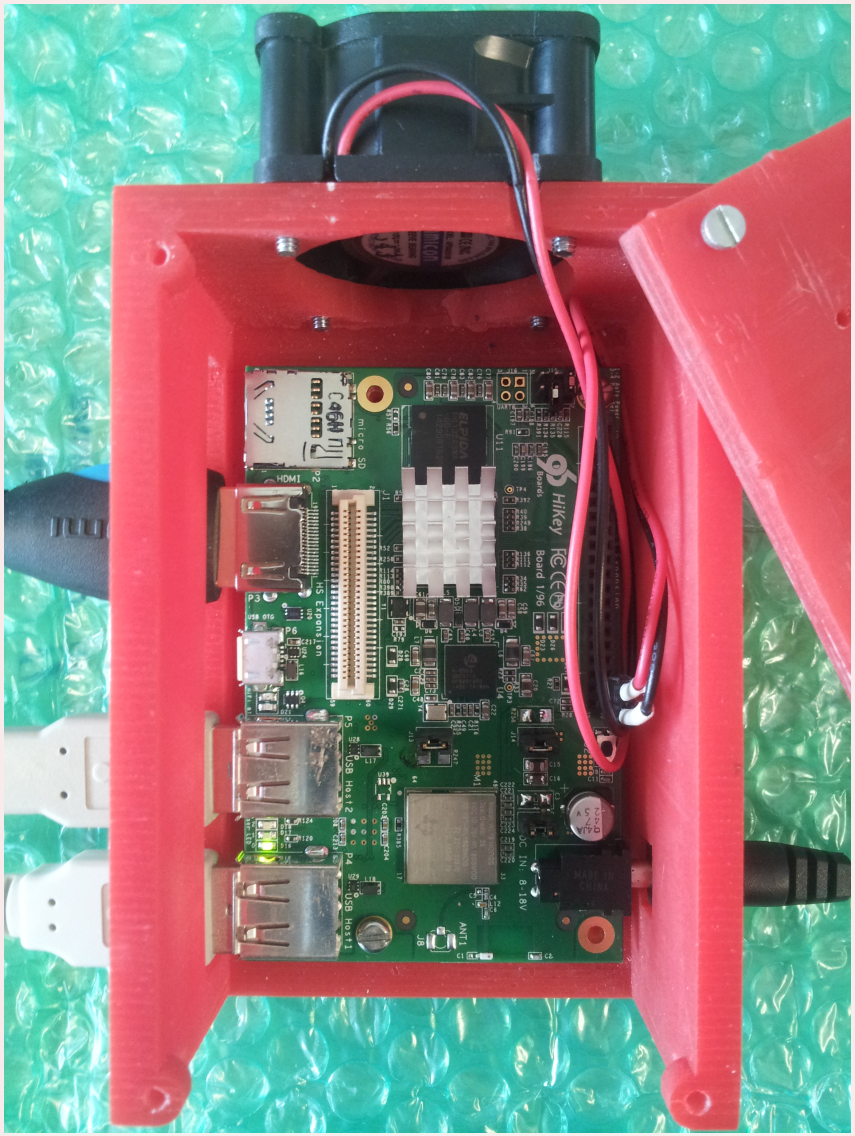
SoC: Tegra K1

- CPU: NVIDIA "4-Plus-1" 2.32GHz ARM quad-core Cortex-A15, with battery-saving shadow-core
- GPU: NVIDIA Kepler "GK20a" GPU with 192 SM3.2 CUDA cores

Awarded for the Best Paper

7th Workshop on UnConventional High Performance Computing (UCHPC), Porto 2014

96Boards - HiKey



SoC: HiSilicon Kirin 6220

- CPU: 8 core ARM Cortex-A53 running at 1.2GHz (64-bit aarch64)
- GPU: ARM Mali 450-MP4 GPU
- MEM: 1GB of 800MHz LPDDR3

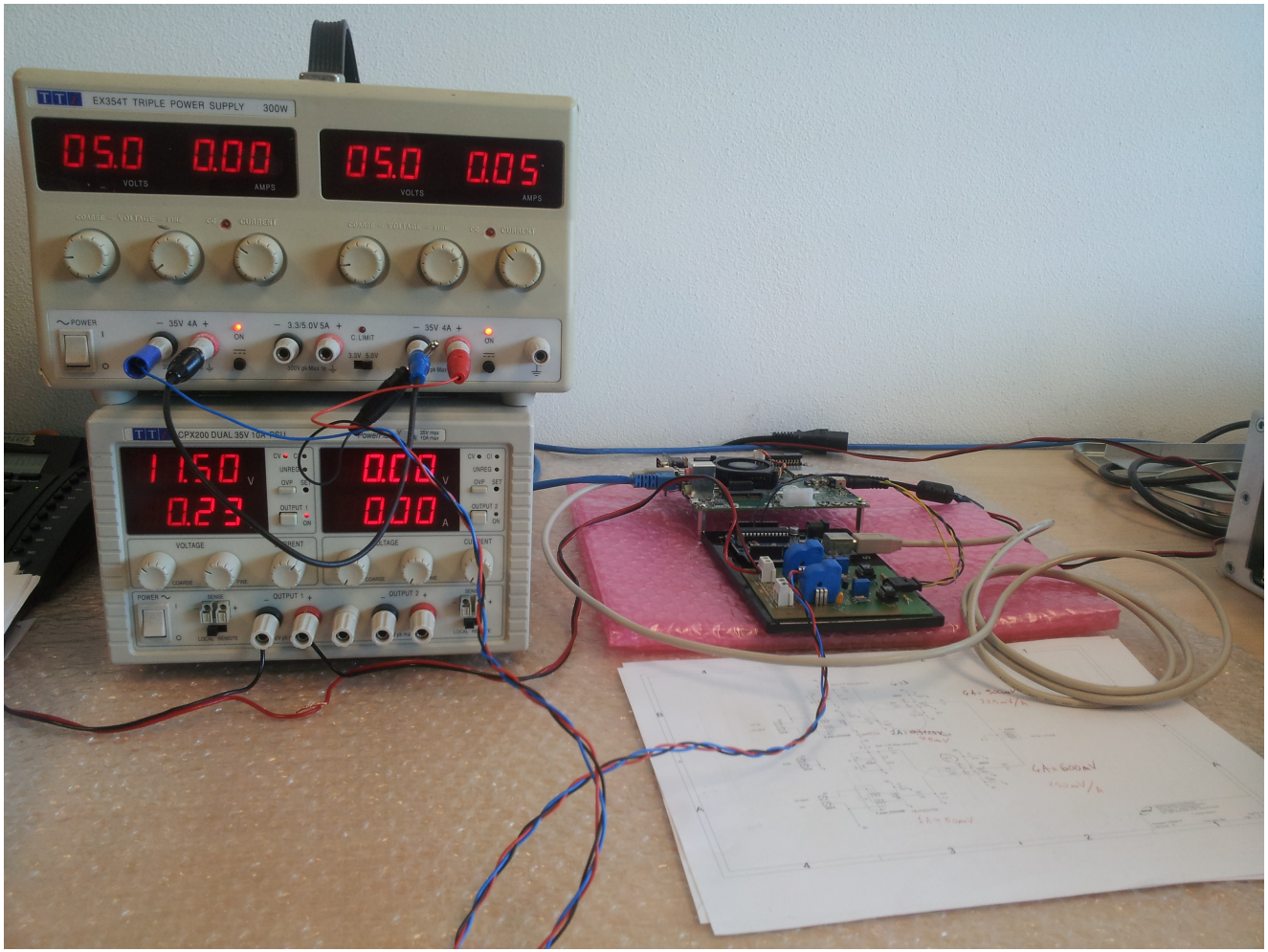
Awarded for the Best Paper

8th Workshop on UnConventional High Performance Computing (UCHPC), Vienna 2015

3D printed case to fit a fan
(Thanks to V. Carassiti and A. Cotta Rusmino, INFN Ferrara)

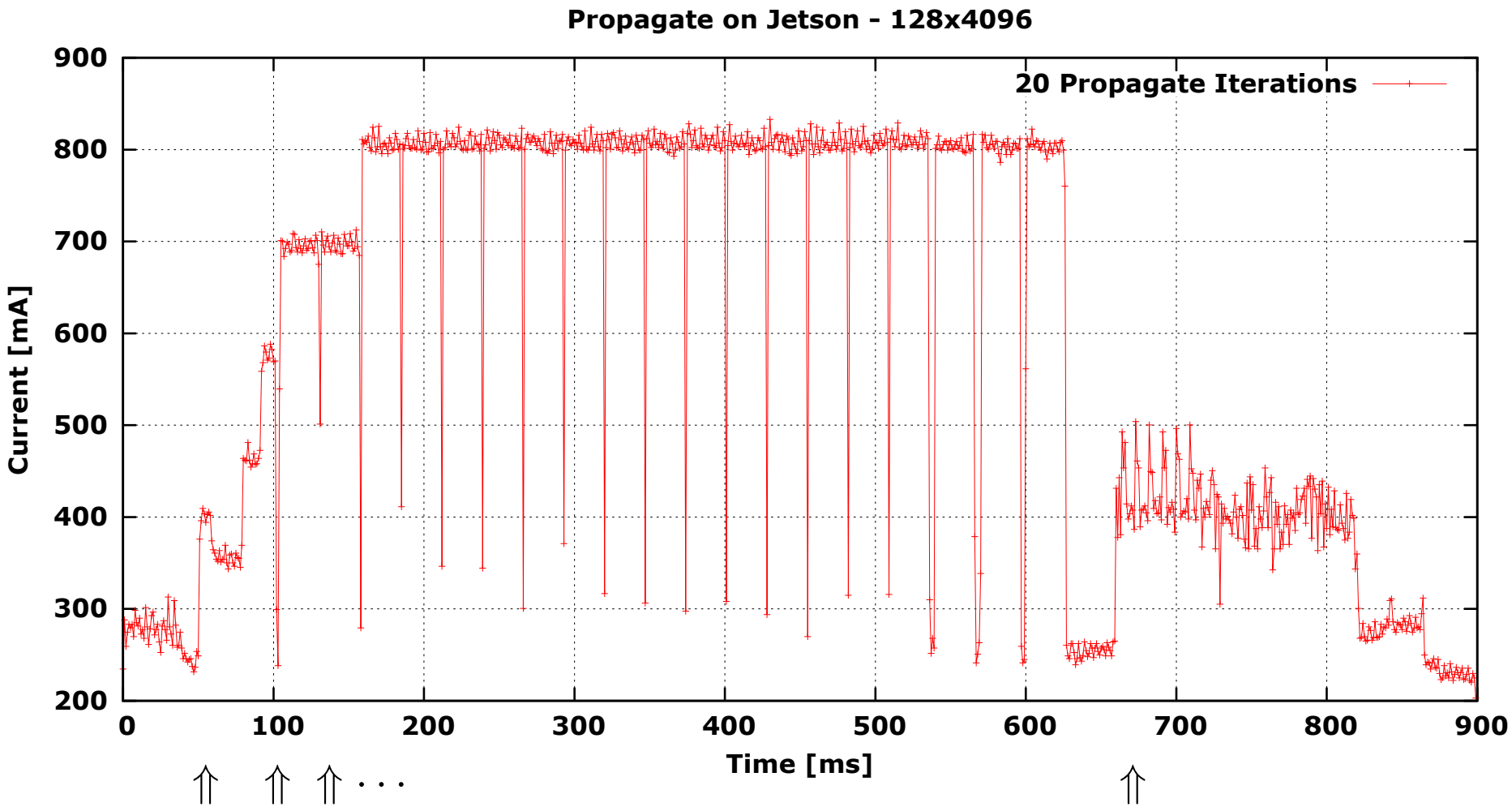
Setup to sample instantaneous current absorption

One current to voltage converter...
...plus an Arduino UNO (microcontroller + 10-bit ADC + Serial over USB)



Current to Voltage + Digitization with Arduino + USB Serial

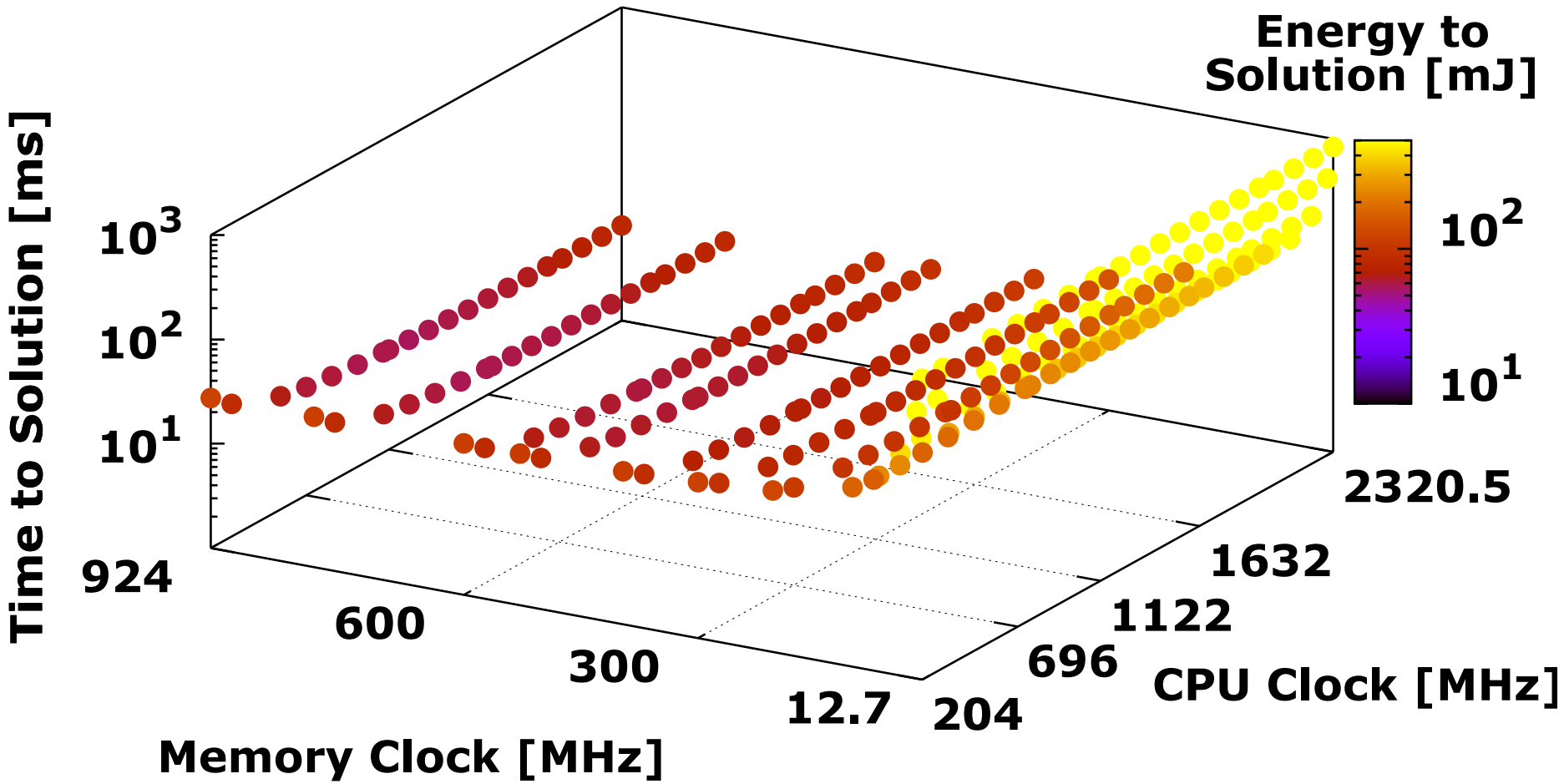
Acquired data example with default frequency scaling



Iterations can be counted

This is a D2H transfer

TK1 CPU - Time/Energy to solution (Propagate)



Outline

Performance Application Programming Interface

- PAPI is a portable interface to hardware performance counters on modern microprocessors
- Additional PAPI Components provide access to a collection of components that expose performance measurement opportunities across the hardware and software stack.

RAPL Component

With Sandy Bridge, Intel introduced the "Running Average Power Limit", or RAPL feature. Although primarily intended to control or limit power usage on chip, this feature also has power and energy measurement capabilities.

NVML Component

NVML component, it demos the component interface and implements two counters nvmlDeviceGetPowerUsage, nvmlDeviceGetTemperature from Nvidia Management Library. Power reported in mW and temperature in Celsius.

Performance Application Programming Interface

- PAPI is a portable interface to hardware performance counters on modern microprocessors
- Additional PAPI Components provide access to a collection of components that expose performance measurement opportunities across the hardware and software stack.

RAPL Component

With Sandy Bridge, Intel introduced the "Running Average Power Limit", or RAPL feature. Although primarily intended to control or limit power usage on chip, this feature also has power and energy measurement capabilities.

NVML Component

NVML component, it demos the component interface and implements two counters nvmlDeviceGetPowerUsage, nvmlDeviceGetTemperature from Nvidia Management Library. Power reported in mW and temperature in Celsius.

Performance Application Programming Interface

- PAPI is a portable interface to hardware performance counters on modern microprocessors
- Additional PAPI Components provide access to a collection of components that expose performance measurement opportunities across the hardware and software stack.

RAPL Component

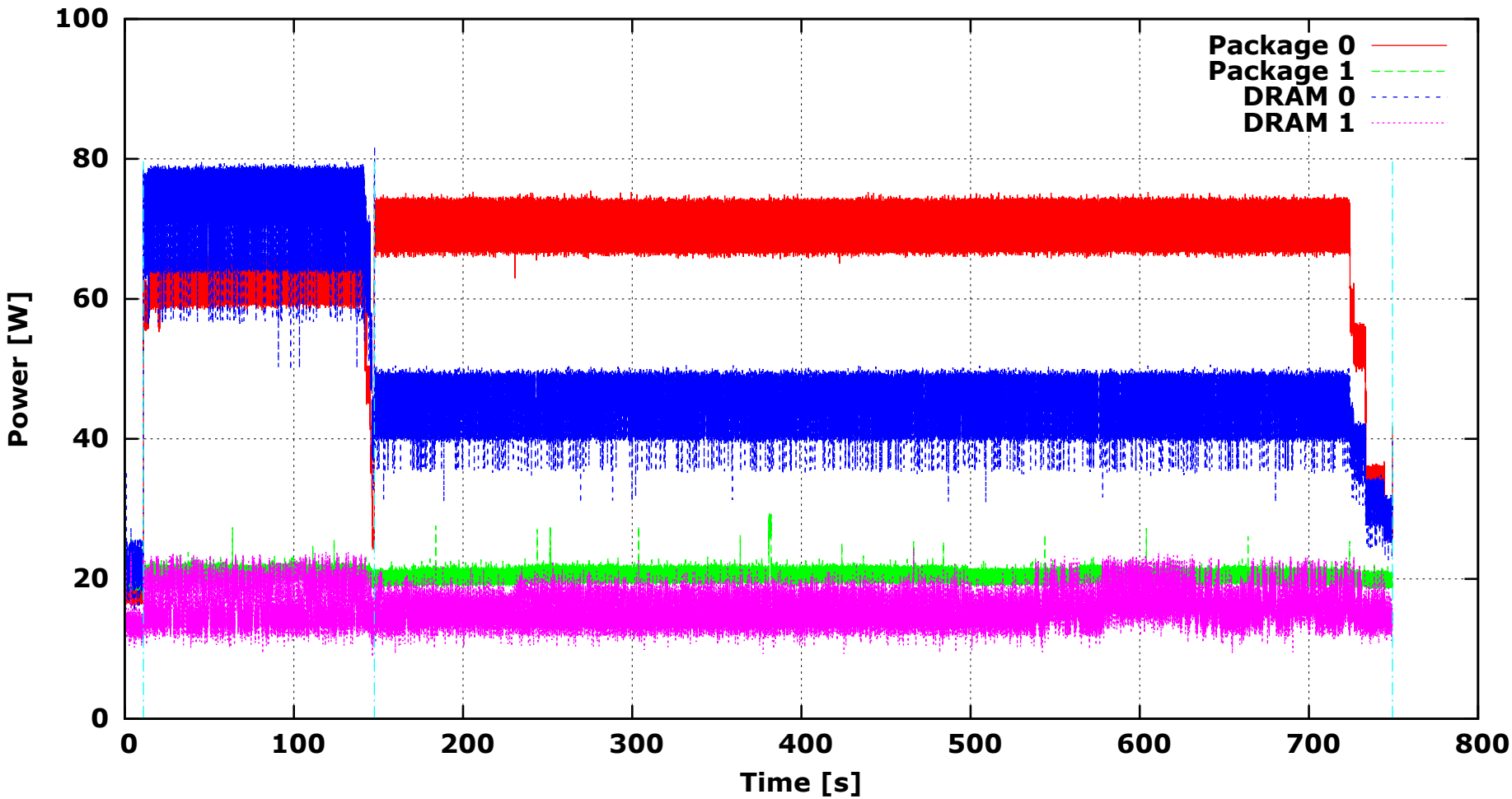
With Sandy Bridge, Intel introduced the "Running Average Power Limit", or RAPL feature. Although primarily intended to control or limit power usage on chip, this feature also has power and energy measurement capabilities.

NVML Component

NVML component, it demos the component interface and implements two counters nvmlDeviceGetPowerUsage, nvmlDeviceGetTemperature from Nvidia Management Library. Power reported in mW and temperature in Celsius.

Outline

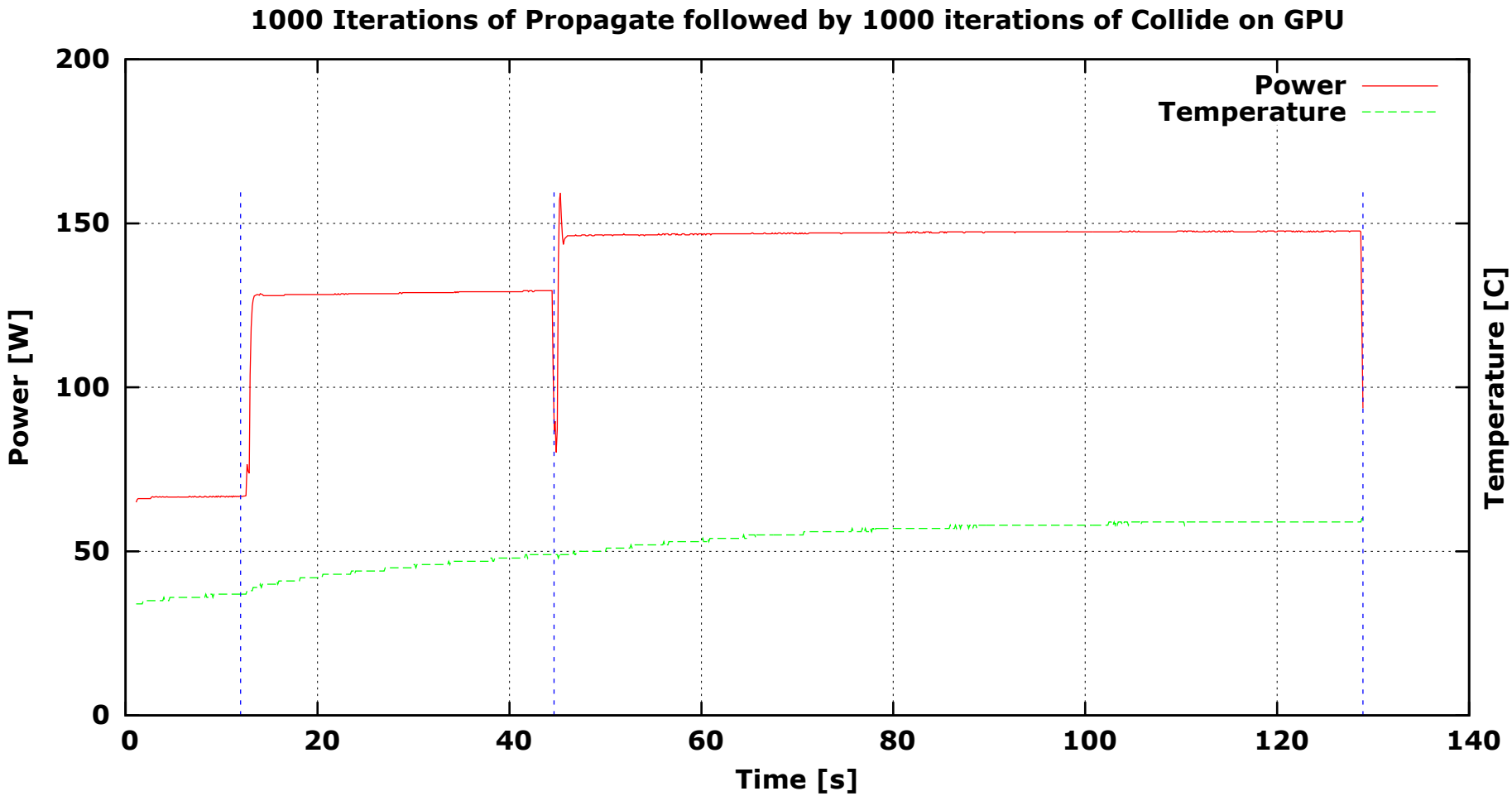
Intel Xeon CPU E5-2630 v3 @ 2.40GHz



1000 iterations of Propagate followed by 1000 iterations of Collide running on Socket 0 as 8 OpenMP threads.

Outline

Half of an NVIDIA K80 GPU

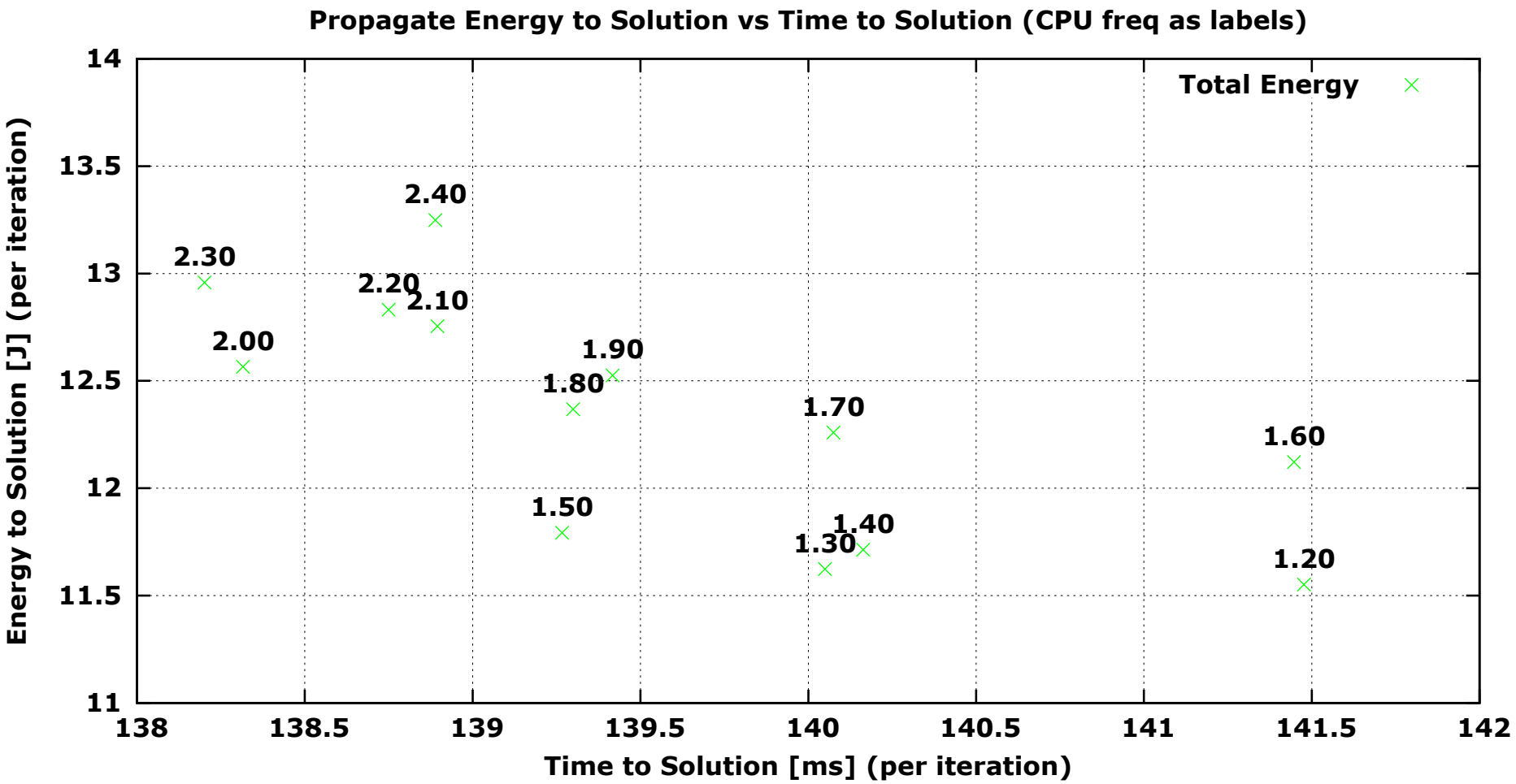


1000 iterations of Propagate followed by 1000 iterations of Collide running on one of the two physical GPUs.

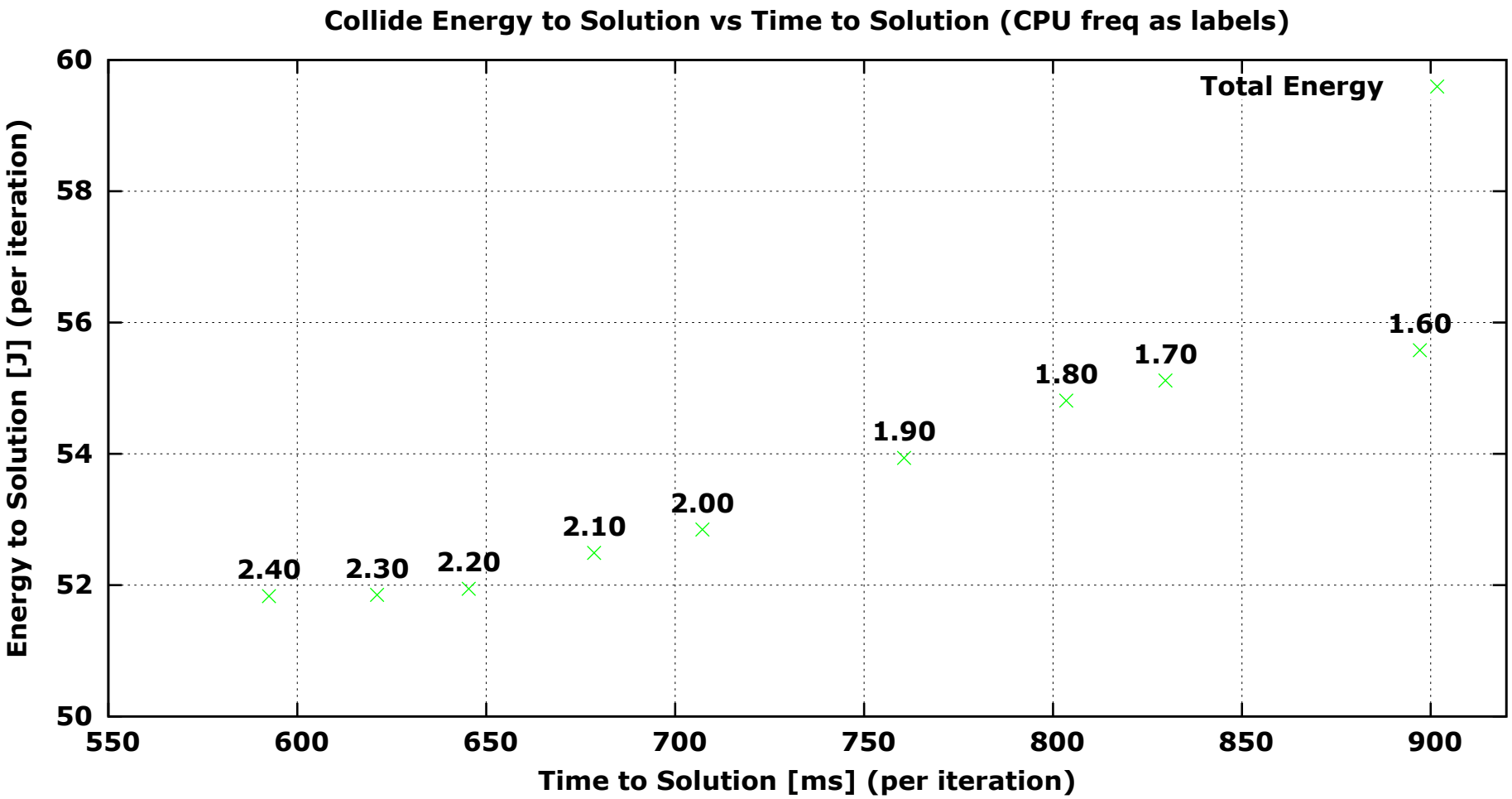
Outline

Outline

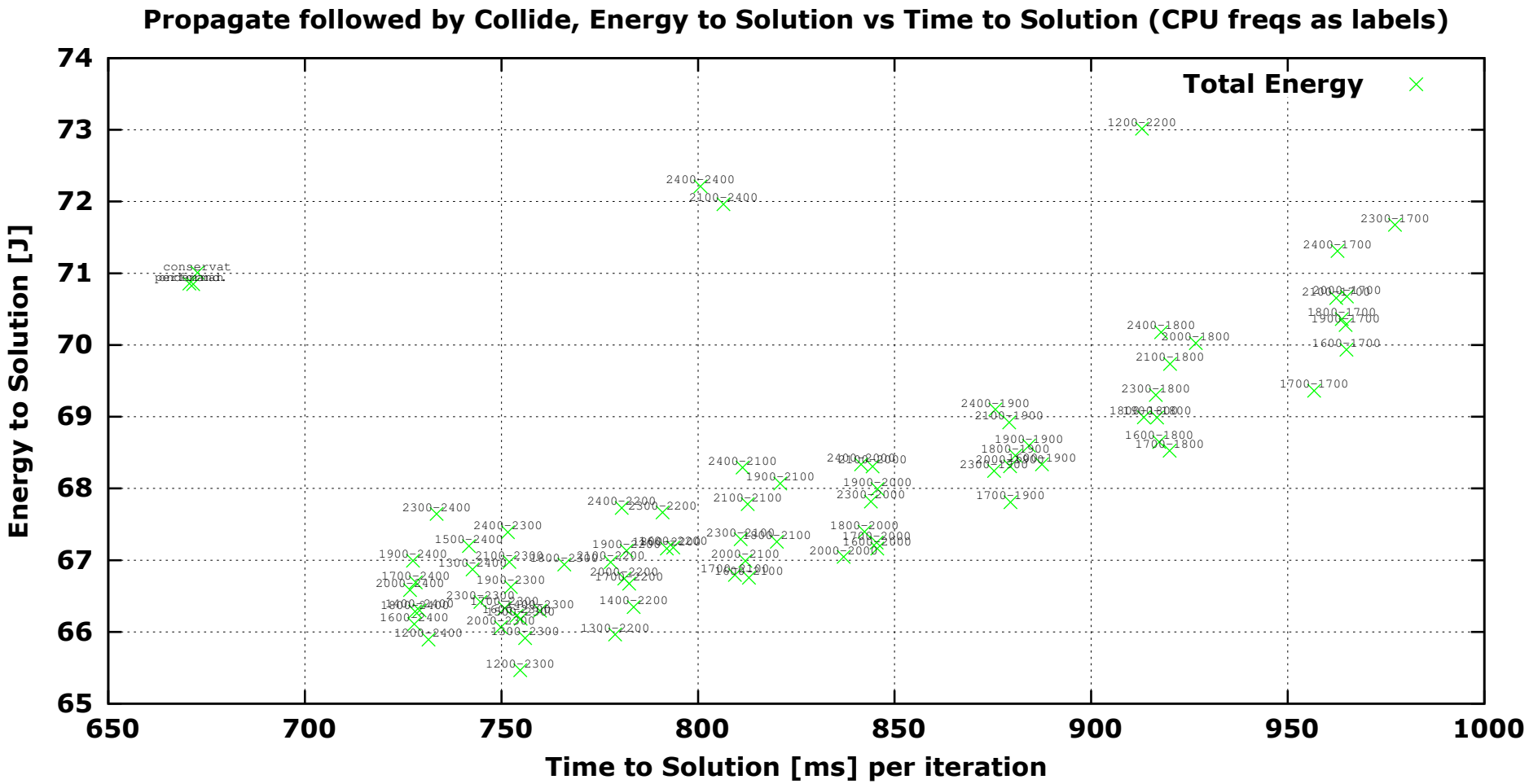
Propagate CPU



Collide CPU

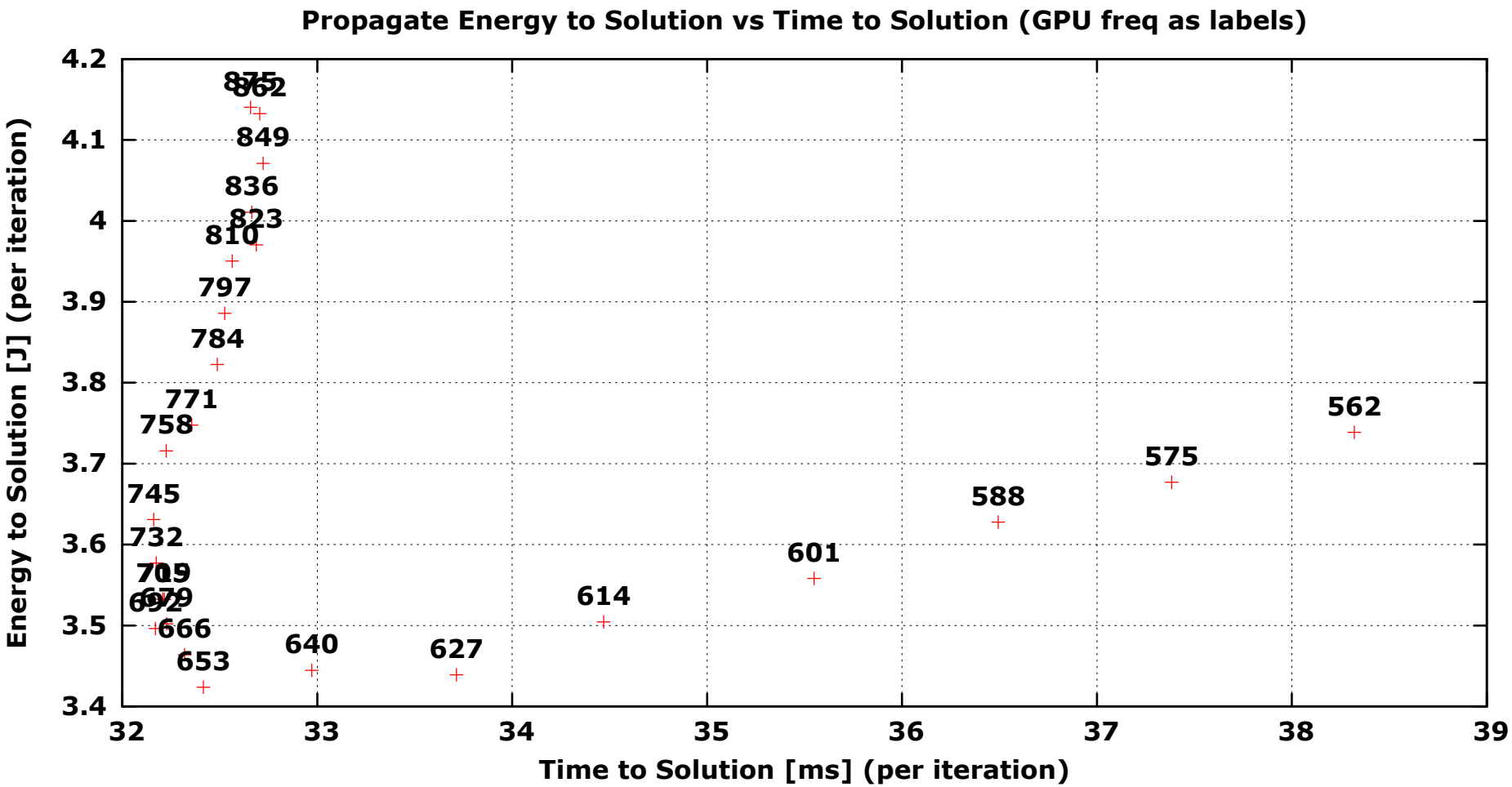

Collide Energy to Solution vs Time to Solution (CPU freq as labels)
Total Energy
Energy to Solution [J] (per iteration)
Time to Solution [ms] (per iteration)
60
58
56
54
52
50
550
600
650
700
750
800
850
900
2.40
2.30
2.20
2.10
2.00
1.90
1.80
1.70
1.60

Changing the CPU Clock for each function

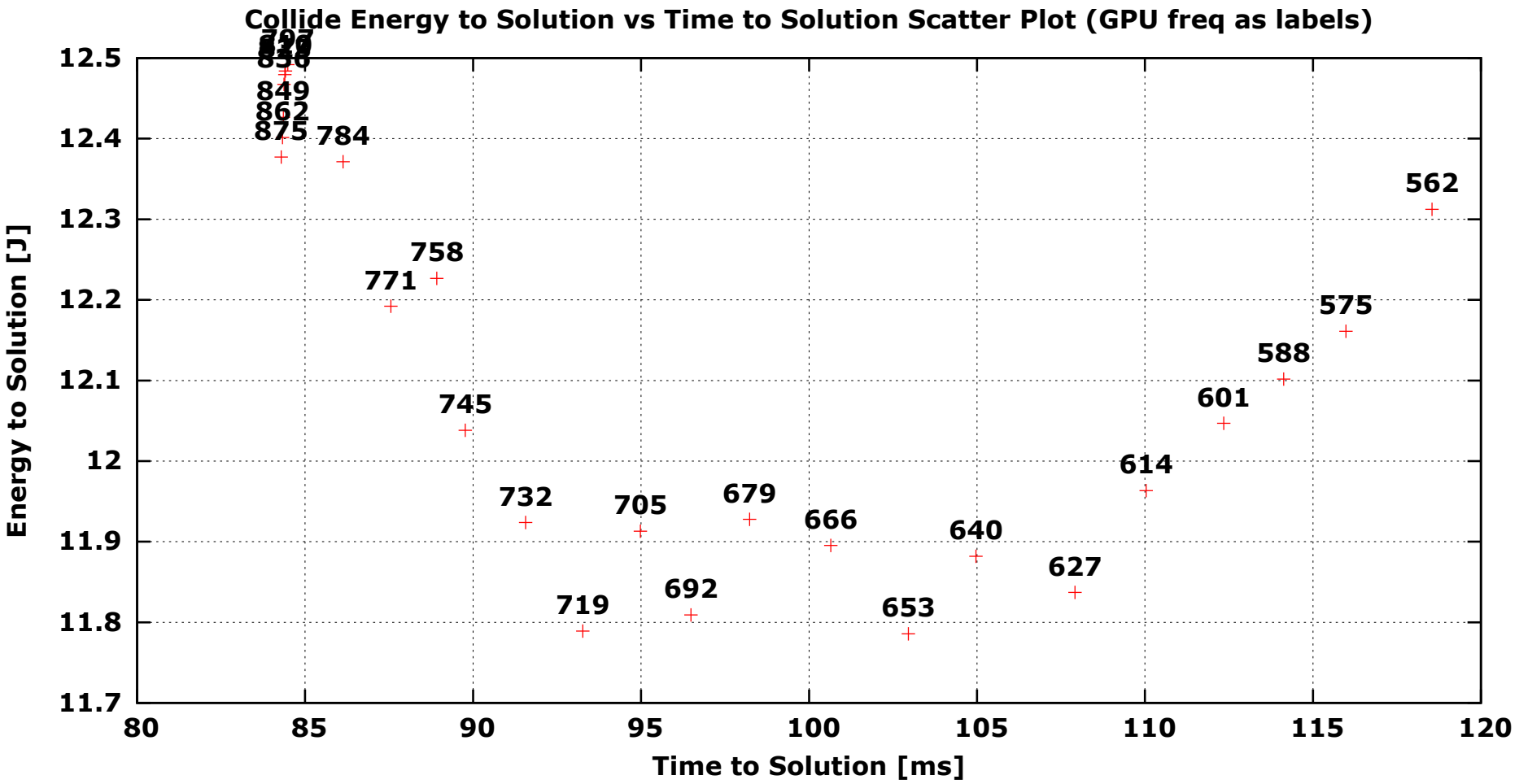


Outline

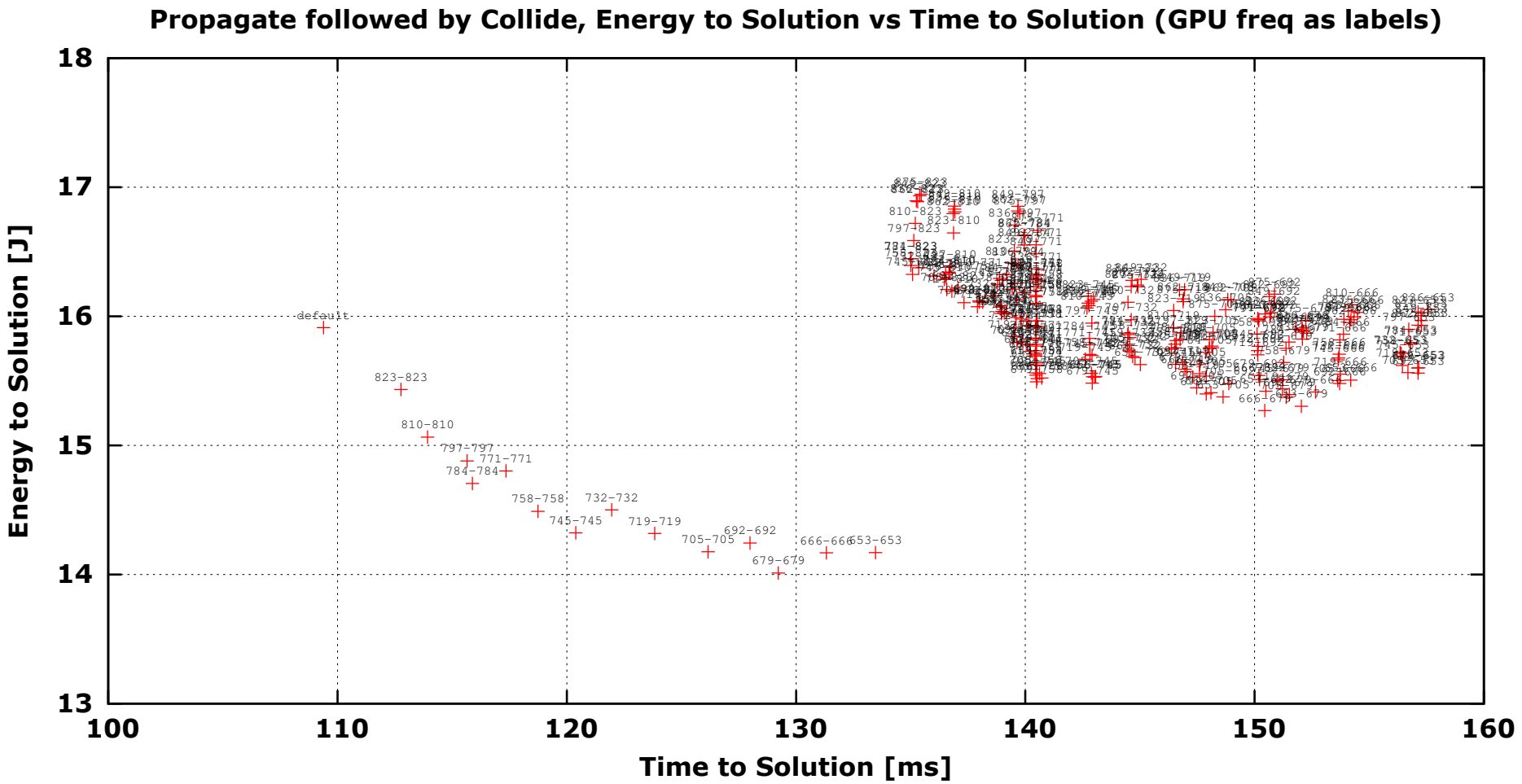
Propagate GPU



Collide GPU


Collide Energy to Solution vs Time to Solution Scatter Plot (GPU freq as labels)
Energy to Solution [J]
Time to Solution [ms]
12.5
12.4
12.3
12.2
12.1
12
11.9
11.8
11.7
80
85
90
95
100
105
110
115
120
849
862
875
784
758
771
745
732
719
705
692
679
666
653
640
627
614
601
588
575
562

Changing the GPU Clock for each kernel



Outline

Conclusions

- limited but not negligible power optimization is possible by adjusting clocks on a kernel-by-kernel basis ($\approx 10 \cdots 20\%$).
- best region is close to the system highest frequencies.
- options to run the processor at very low frequencies seem almost useless.
- smaller transition times between frequency changes would be useful, in particular for GPUs.

Future works

- deeply investigate software tuning and multi-objective optimization techniques.
- evaluate communication costs between different boards.
- evaluate the energy saving impact on a full HPC node embedding several accelerators.

Conclusions

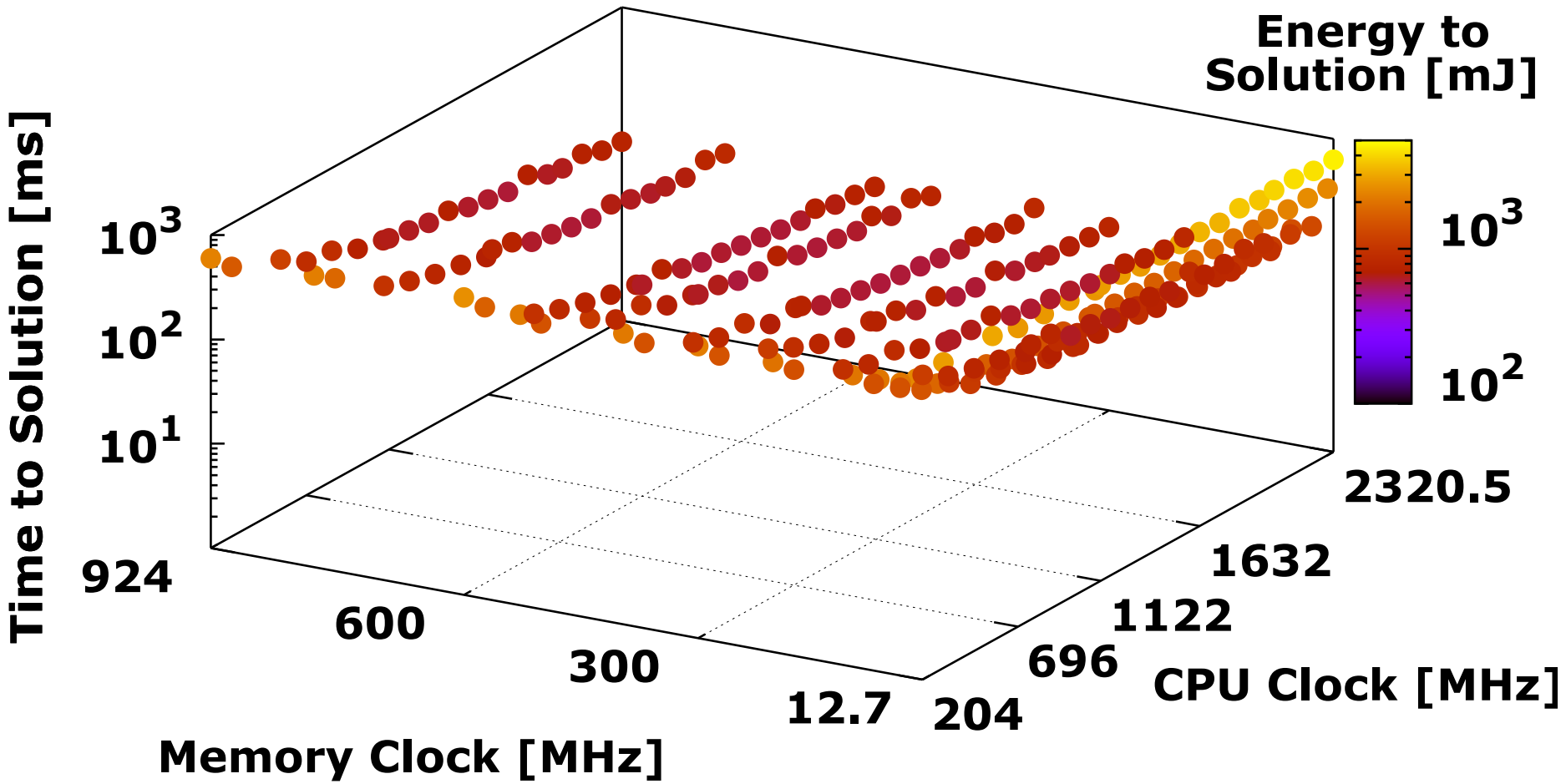
- limited but not negligible power optimization is possible by adjusting clocks on a kernel-by-kernel basis ($\approx 10 \cdots 20\%$).
- best region is close to the system highest frequencies.
- options to run the processor at very low frequencies seem almost useless.
- smaller transition times between frequency changes would be useful, in particular for GPUs.

Future works

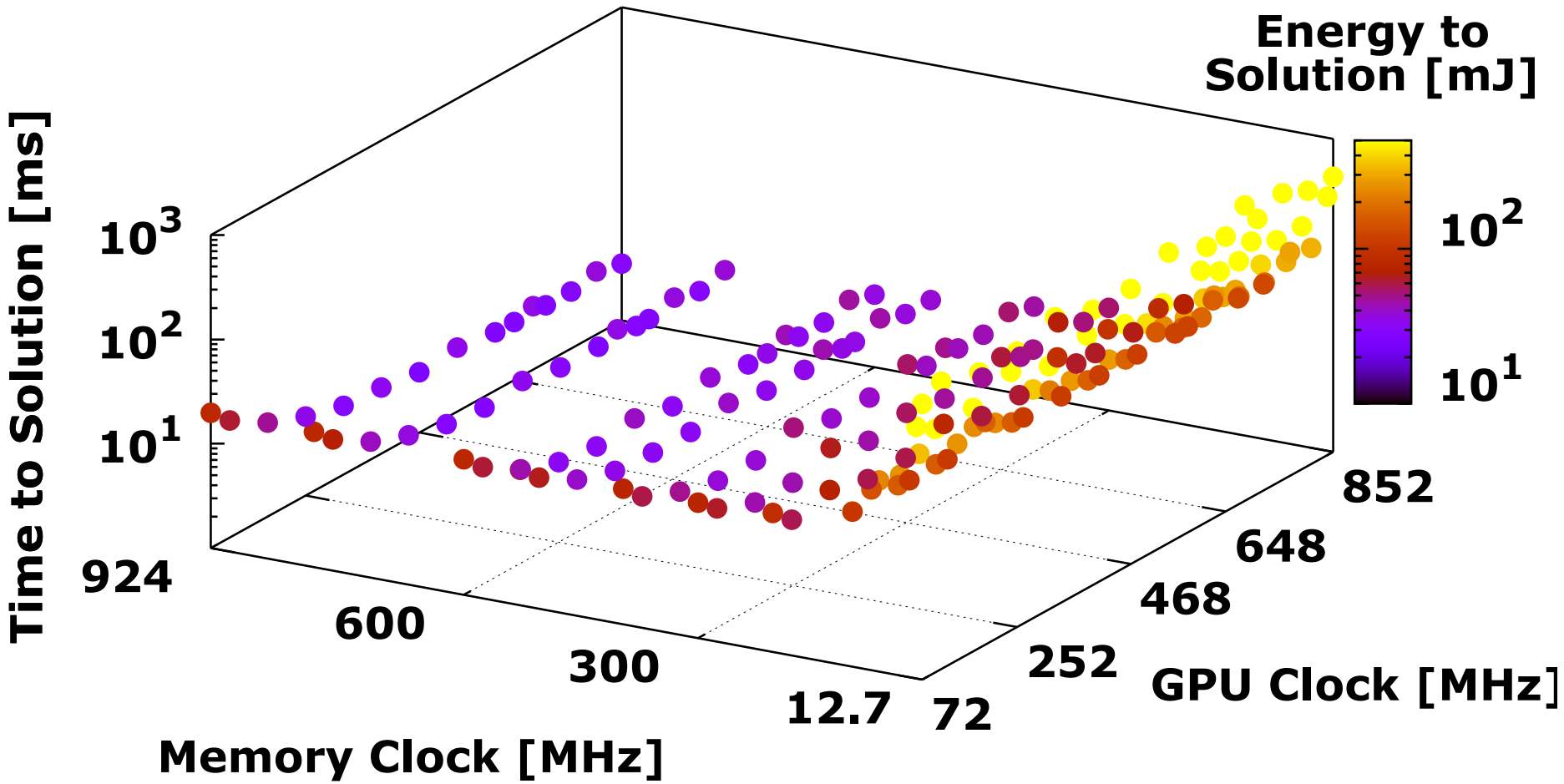
- deeply investigate software tuning and multi-objective optimization techniques.
- evaluate communication costs between different boards.
- evaluate the energy saving impact on a full HPC node embedding several accelerators.

Thanks for Your attention

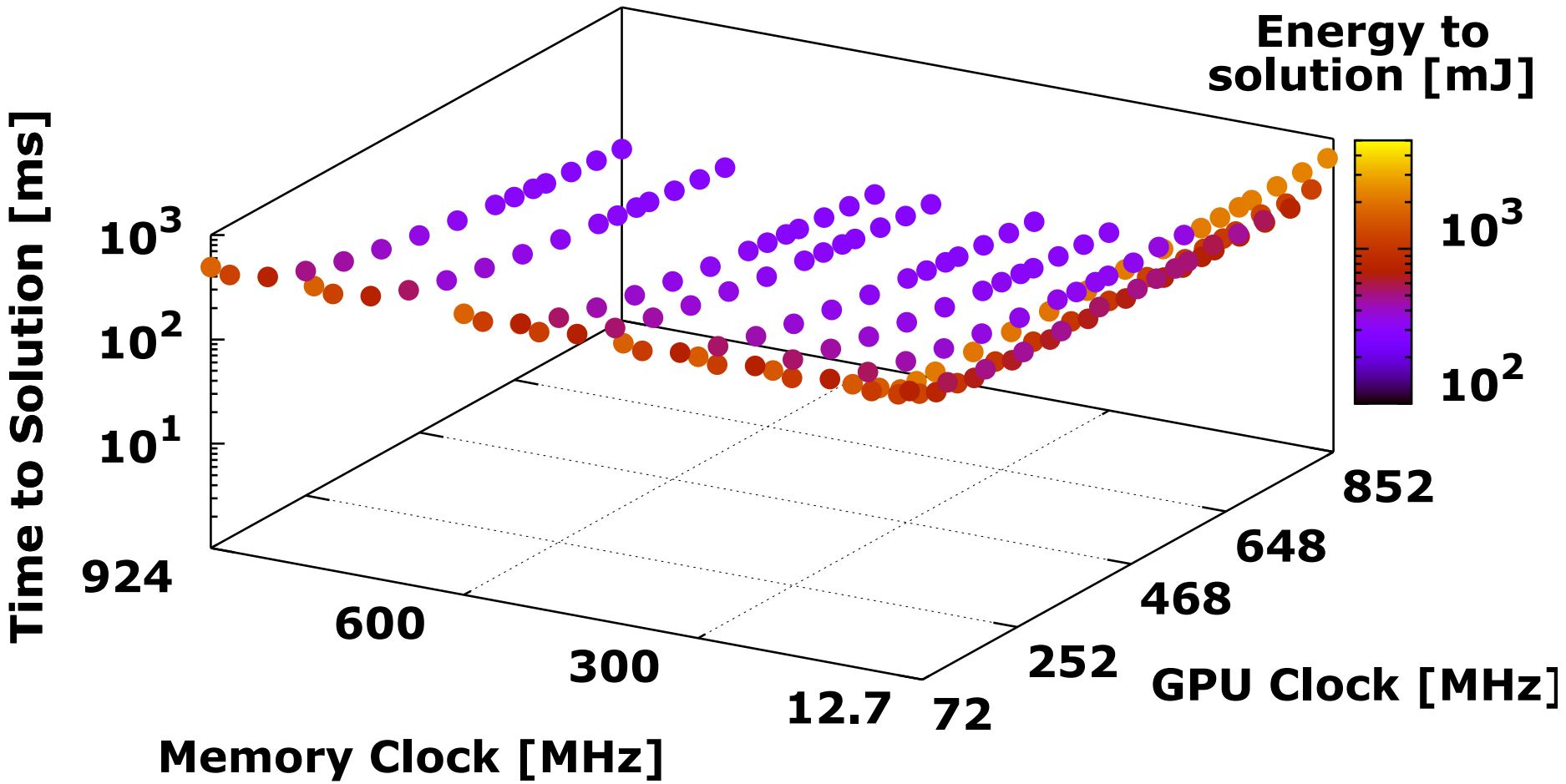
TK1 CPU - Time/Energy to solution (Collide)



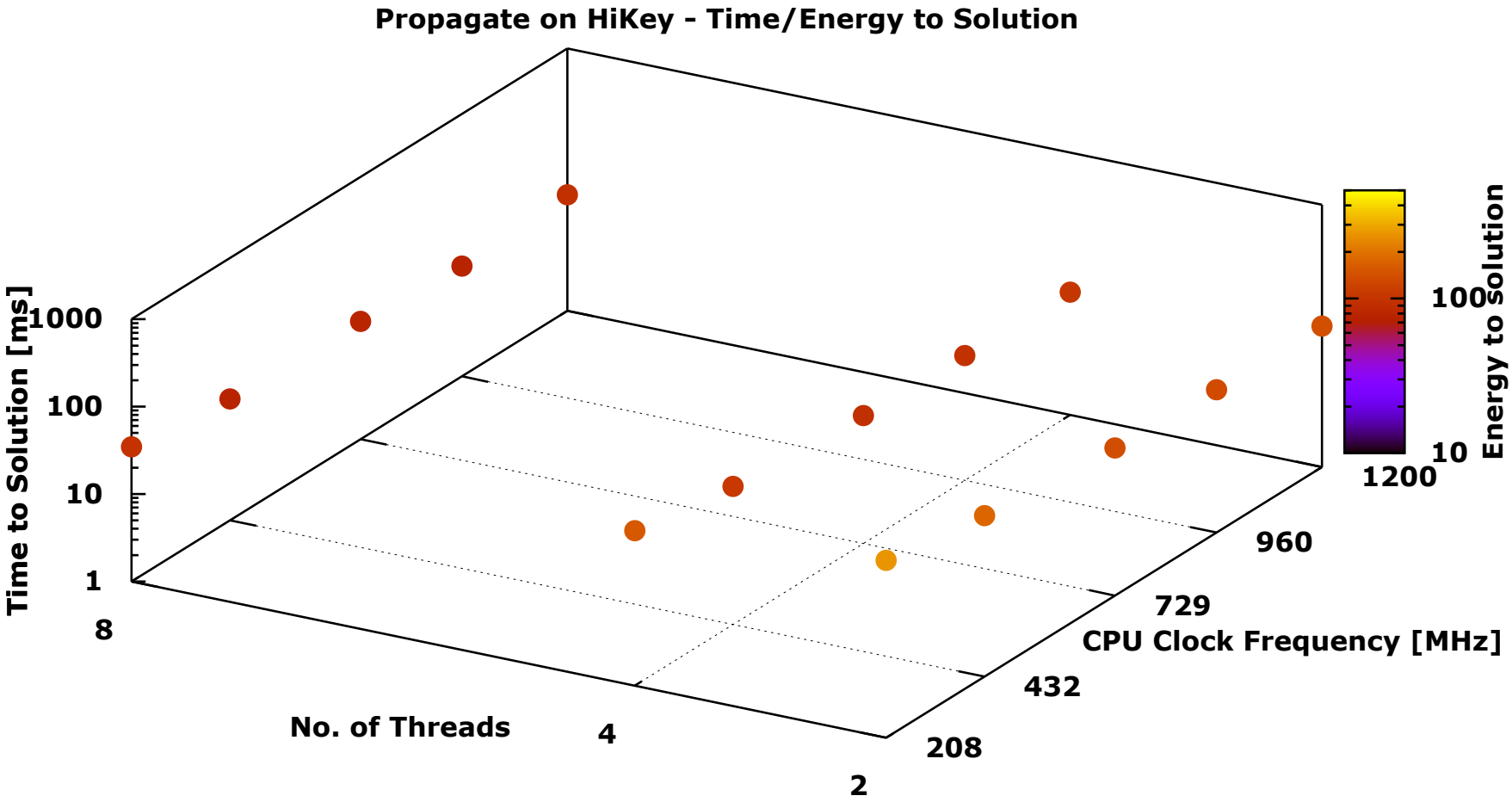
TK1 GPU - Time/Energy to solution (Propagate)



TK1 GPU - Time/Energy to solution (Collide)



HiKey - Time/Energy to solution (Propagate)



HiKey - Time/Energy to solution (Collide)