GPUDay 2026

Toolbox for HEP and the sustainability of HEP event generators

HUN-REN
Hungarian Research Network

HUN REN | wigner

Szabolcs Molnár
Gábor Bíró
Gergely Gábor Barnaföldi
Gábor Papp

Contact: molnar.szabocs@wigner.hun-ren.hu

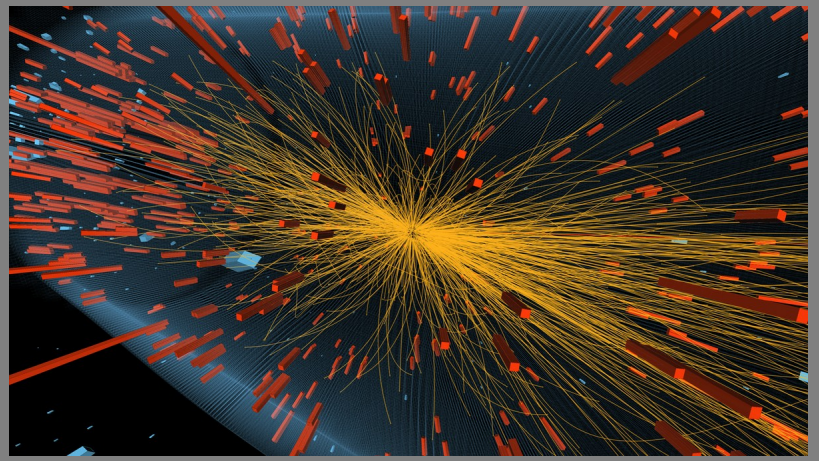
Table of contents

Docker

- Open source platform for containerization.
- Docker containers bundle an app with everything it needs (code, runtime, libraries).
- Containers share the host OS kernel, starting in seconds and using far fewer resources than virtual machines. ⇨ Lightweight and fast
- Each container runs independently, avoiding dependency conflicts. ⇨ Isolated environments
- Build images from simple Dockerfiles, ensuring consistent setups across dev, test, and production. ⇨ Declarative & reproducible
- Docker Hub for sharing images, Docker Compose for multi-service apps, and native integration with Kubernetes.

Monte Carlo event generators & HIJING ++

- MC event generators simulate the complete evolution of particle collisions, proton-proton, proton-heavy ion and heavy ion-heavy ion.
- HIJING++ (Heavy Ion Jet INteraction Generator), written in C++ and has built in parallelization (CPU)
- MC event generators generate a large amount of data, require lot of resources to run
- The need for computational power is growing ⇨ Optimization is important
- Tuning of MC event generators is required (e.g. Monash tune of Pythia8)



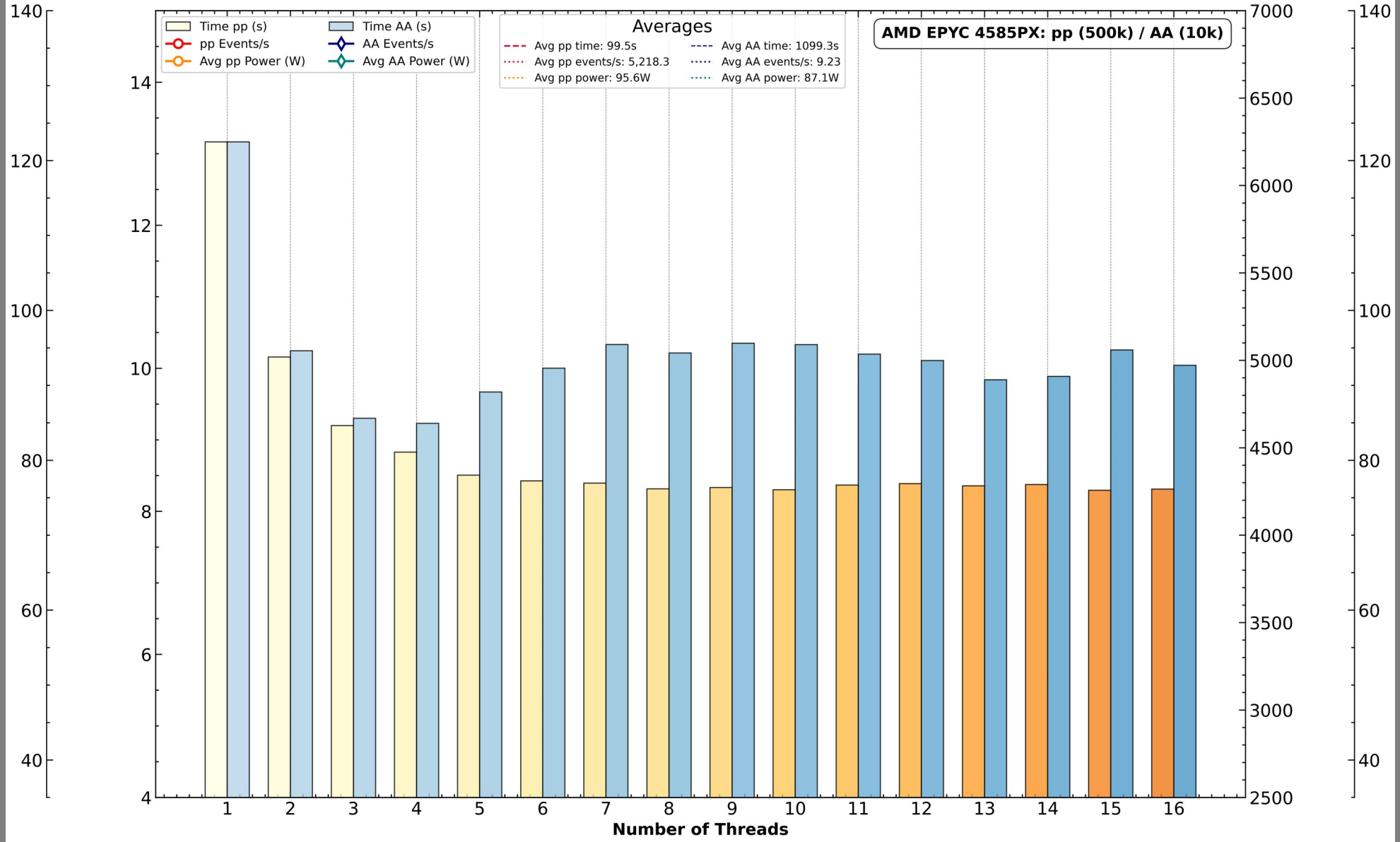
HEP Toolbox

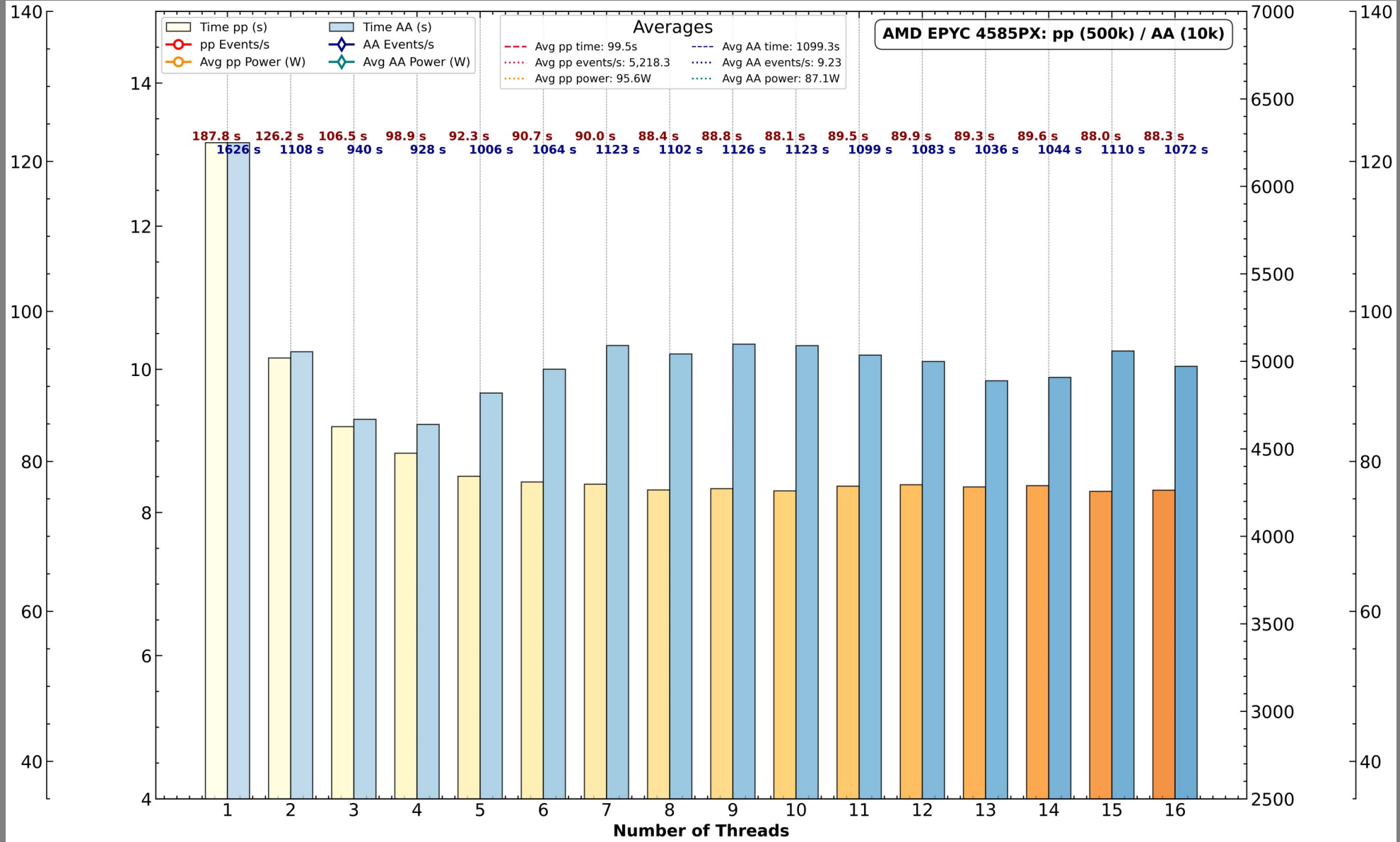
- Professor, Rivet, Pythia ⇨proripy
- Based on Ubuntu 22 LTS (for now), 100% open source
- Future plans to create a version with 

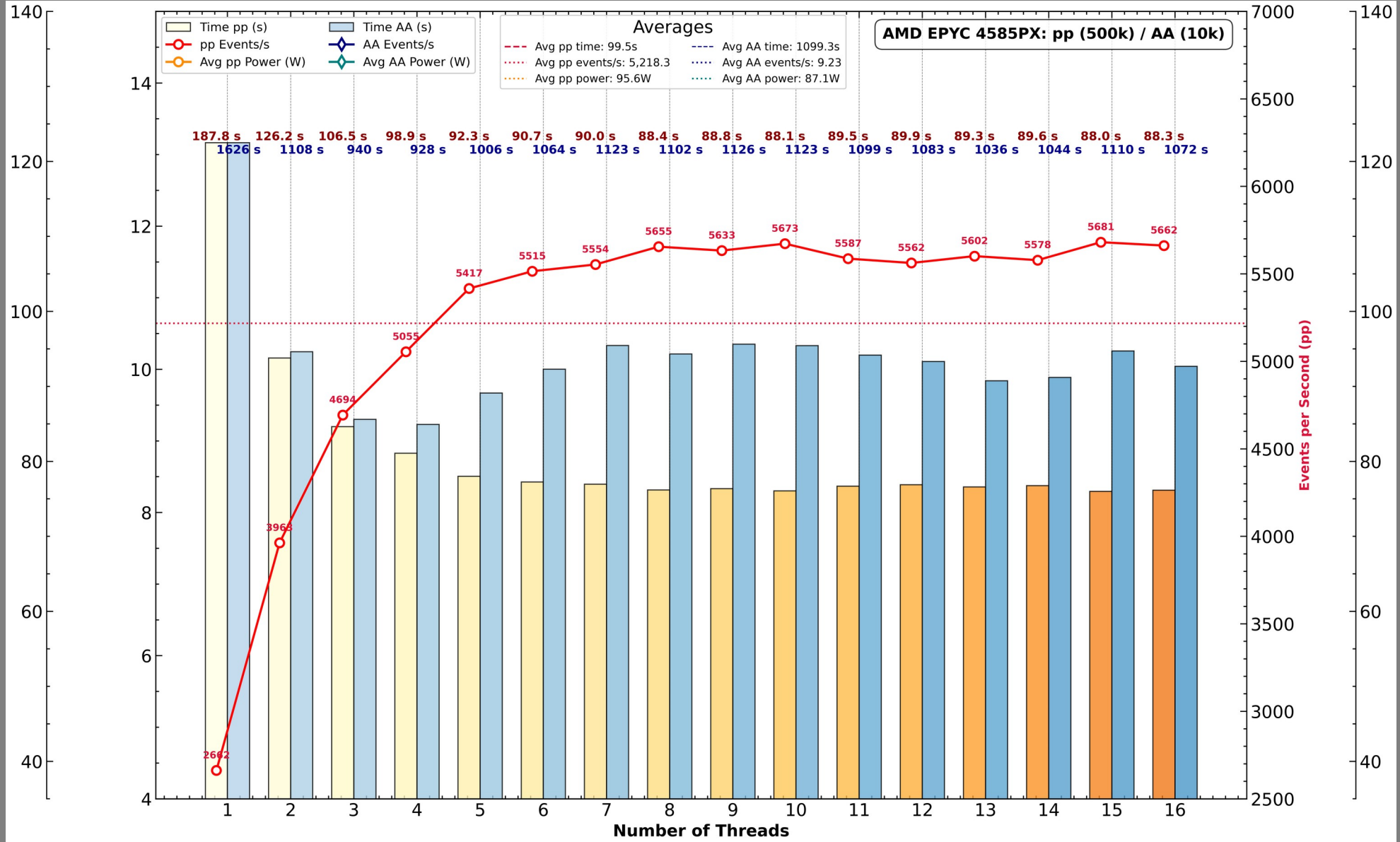
base, rolling release image version
- Image includes: Pythia8, Rivet, Professor, ROOT, YODA, HepMC3, FastJet3, HighFive, LHAPDF6
- Code debuggers included: C/C++, Python3, Go, Ruby, Rust (+ GNU Debugger as extra)
- Image has an SSH version for VSCode integration
- Latest version: 77rev/proripy:4.4
- It was initially created to make the tuning and debugging of HIJING++ more convenient but now has over 1300 downloads.

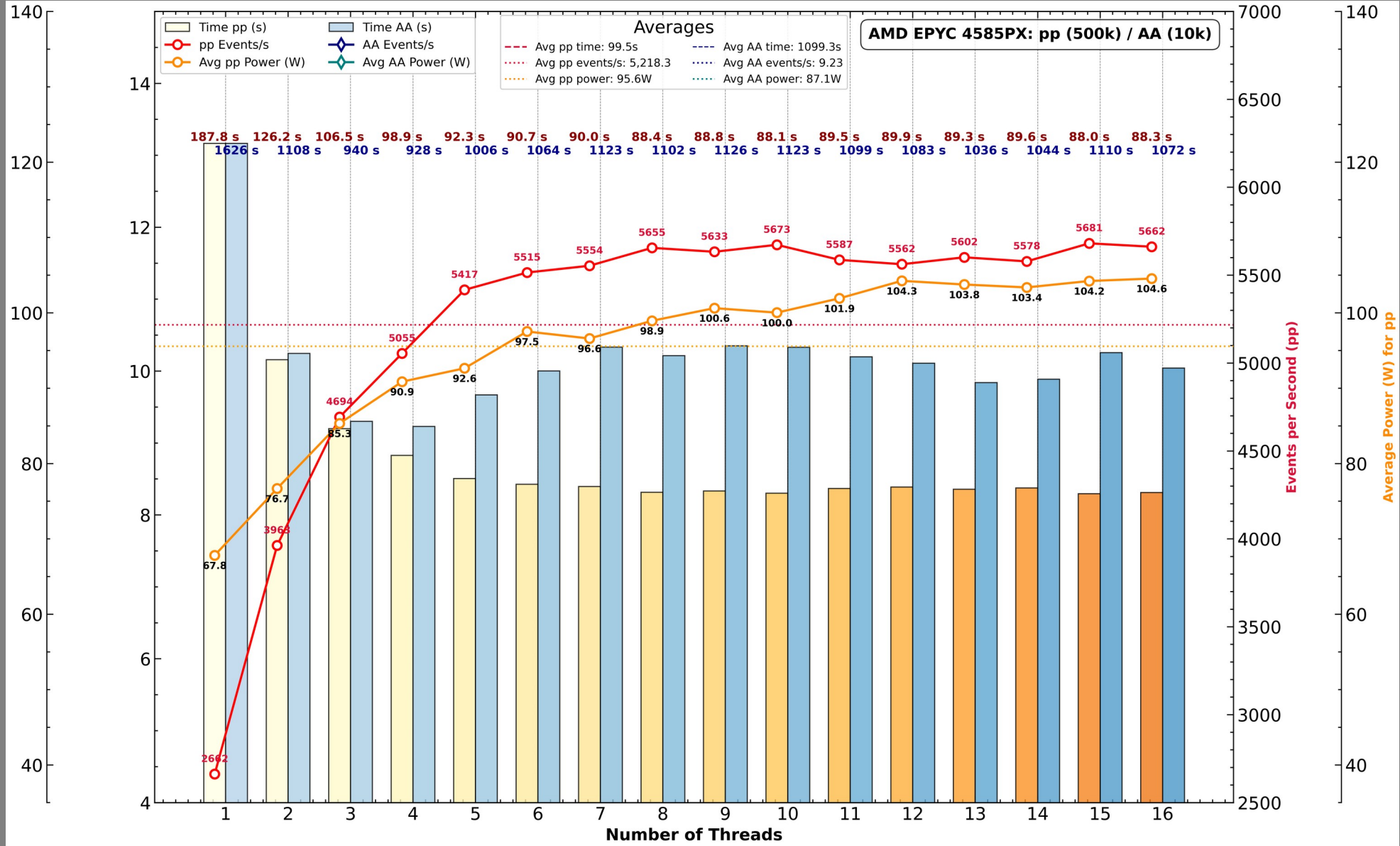
HEP Toolbox

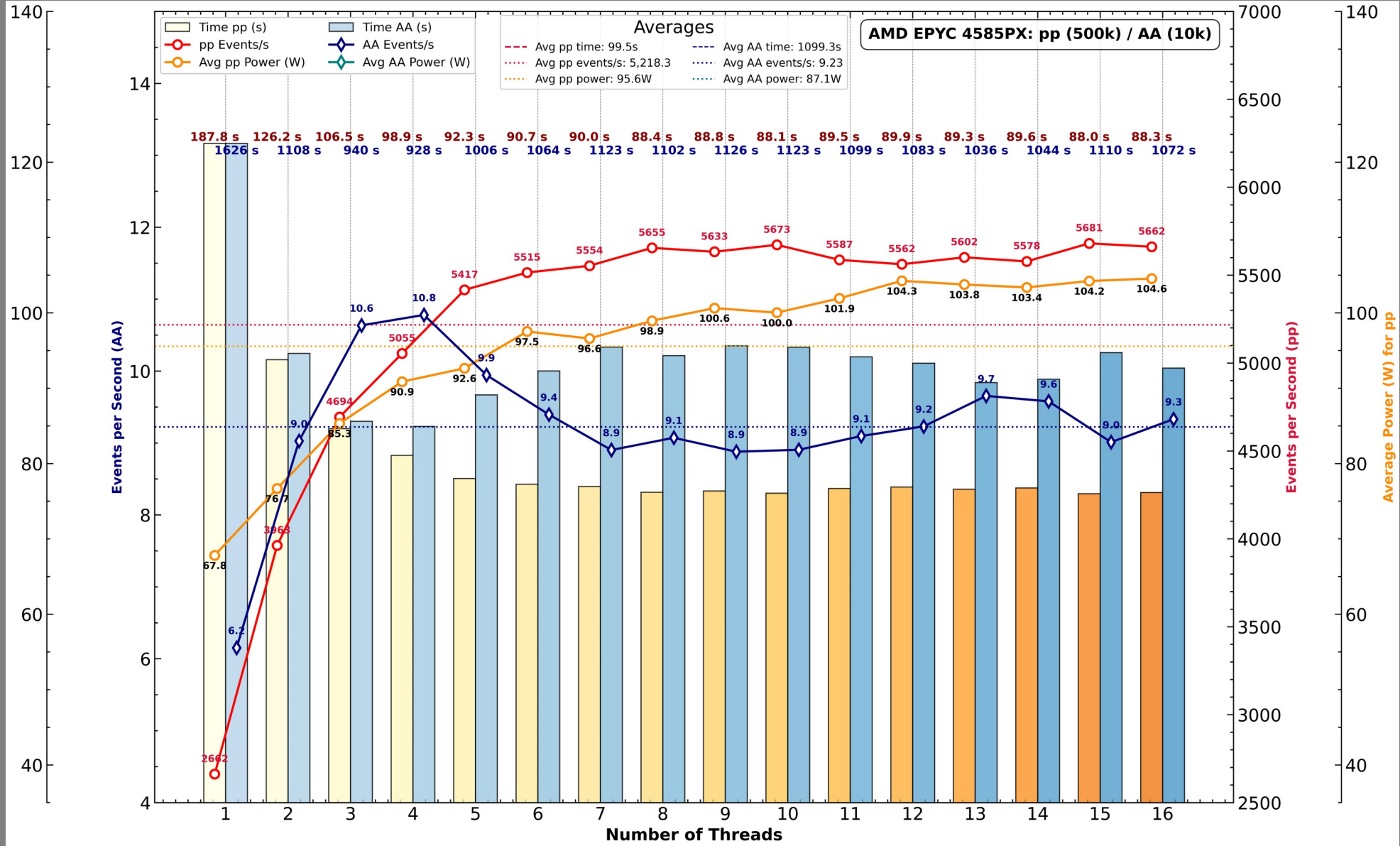
- Available on hub.docker.com but also on its own gitlab project
- Custom benchmarking script included from version 4.4
- Benchmarking script can be used to measure MC event generator run time, CPU power consumption and test the impact of parallelization on runtime and power consumption
- HIJING++ ran with 500,000 pp events and 10,000 AA events
- We investigated 5 different CPUs
  - Intel Xeon E5-2650 | 8C/16T @ 2.6GHz | 2012 | 95 W
  - AMD EPYC 7302 | 16C/32T @ 3.35GHz | 2019 | 155 W
  - AMD EPYC 7502P | 32C/64T @ 3.35GHz | 2019 | 180 W
  - AMD Ryzen 7 8845HS | 8C/16T @ 5.14GHz | 2023 | 54 W
  - AMD EPYC 4585PX | 16C/32T @ 5.76GHz | 2025 | 170 W

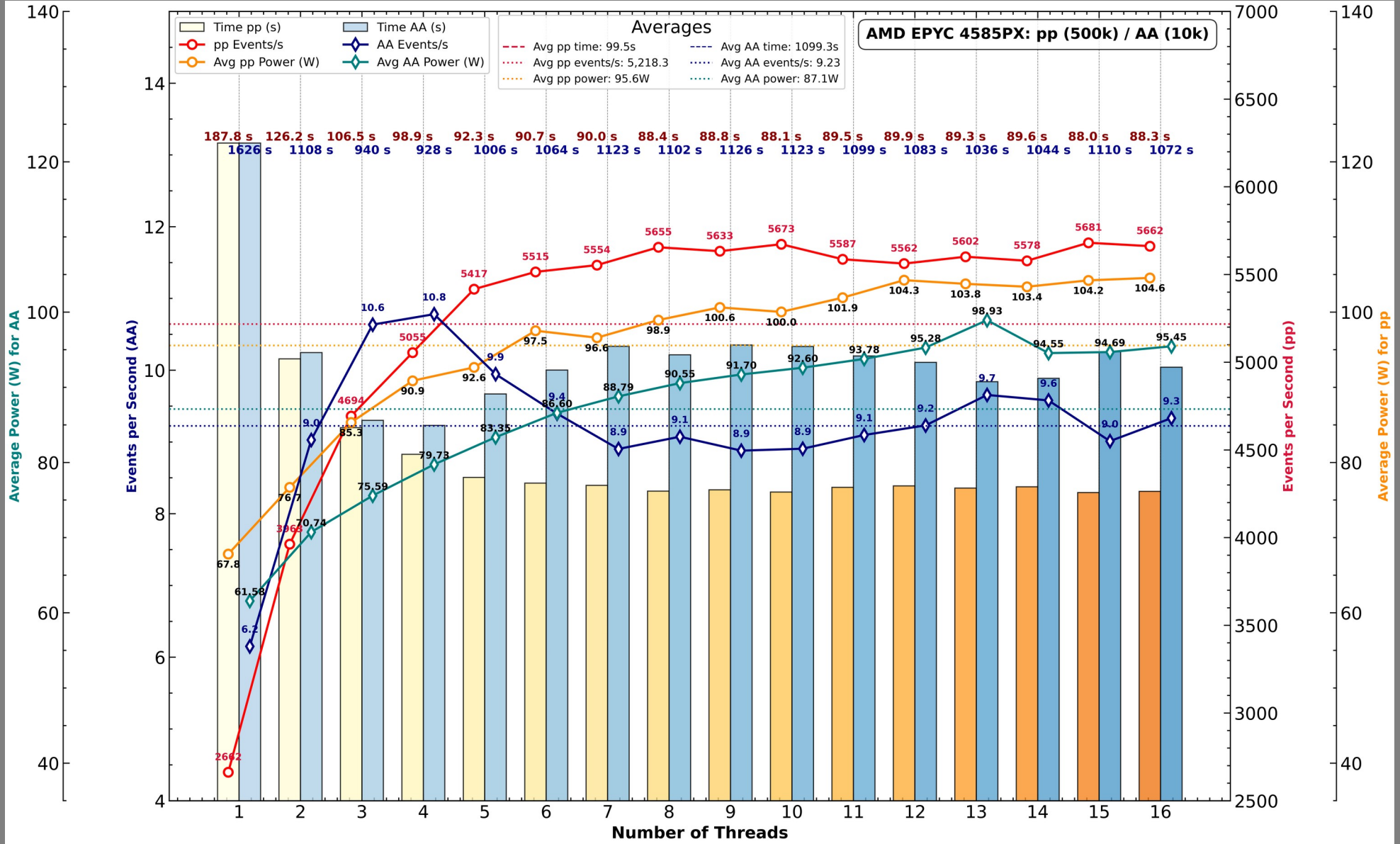

Time pp (s)
pp Events/s
Avg pp Power (W)
Time AA (s)
AA Events/s
Avg AA Power (W)
Averages
Avg pp time: 99.5s
Avg pp events/s: 5,218.3
Avg pp power: 95.6W
Avg AA time: 1099.3s
Avg AA events/s: 9.23
Avg AA power: 87.1W
AMD EPYC 4585PX: pp (500k) / AA (10k)
Number of Threads


AMD EPYC 4585PX: pp (500k) / AA (10k)
Time pp (s)
pp Events/s
Avg pp Power (W)
Time AA (s)
AA Events/s
Avg AA Power (W)
Averages
Avg pp time: 99.5s
Avg pp events/s: 5,218.3
Avg pp power: 95.6W
Avg AA time: 1099.3s
Avg AA events/s: 9.23
Avg AA power: 87.1W
187.8 s 126.2 s 106.5 s 98.9 s 92.3 s 90.7 s 90.0 s 88.4 s 88.8 s 88.1 s 89.5 s 89.9 s 89.3 s 89.6 s 88.0 s 88.3 s
1626 s 1108 s 940 s 928 s 1006 s 1064 s 1123 s 1102 s 1126 s 1123 s 1099 s 1083 s 1036 s 1044 s 1110 s 1072 s
Number of Threads


AMD EPYC 4585PX: pp (500k) / AA (10k)
Time pp (s)
Time AA (s)
pp Events/s
AA Events/s
Avg pp Power (W)
Avg AA Power (W)
Averages
Avg pp time: 99.5s
Avg AA time: 1099.3s
Avg pp events/s: 5,218.3
Avg AA events/s: 9.23
Avg pp power: 95.6W
Avg AA power: 87.1W
187.8 s 126.2 s 106.5 s 98.9 s 92.3 s 90.7 s 90.0 s 88.4 s 88.8 s 88.1 s 89.5 s 89.9 s 89.3 s 89.6 s 88.0 s 88.3 s
1626 s 1108 s 940 s 928 s 1006 s 1064 s 1123 s 1102 s 1126 s 1123 s 1099 s 1083 s 1036 s 1044 s 1110 s 1072 s
2662
3968
4694
5055
5417
5515
5554
5655
5633
5673
5587
5562
5602
5578
5681
5662
Number of Threads
Events per Second (pp)


AMD EPYC 4585PX: pp (500k) / AA (10k)
Time pp (s)
Time AA (s)
pp Events/s
AA Events/s
Avg pp Power (W)
Avg AA Power (W)
Averages
Avg pp time: 99.5s
Avg AA time: 1099.3s
Avg pp events/s: 5,218.3
Avg AA events/s: 9.23
Avg pp power: 95.6W
Avg AA power: 87.1W
187.8 s 126.2 s 106.5 s 98.9 s 92.3 s 90.7 s 90.0 s 88.4 s 88.8 s 88.1 s 89.5 s 89.9 s 89.3 s 89.6 s 88.0 s 88.3 s
1626 s 1108 s 940 s 928 s 1006 s 1064 s 1123 s 1102 s 1126 s 1123 s 1099 s 1083 s 1036 s 1044 s 1110 s 1072 s
2662
3963
4694
5055
5417
5515
5554
5655
5633
5673
5587
5562
5602
5578
5681
5662
67.8
76.7
85.3
90.9
92.6
97.5
96.6
98.9
100.6
100.0
101.9
104.3
103.8
103.4
104.2
104.6
Number of Threads
Events per Second (pp)
Average Power (W) for pp

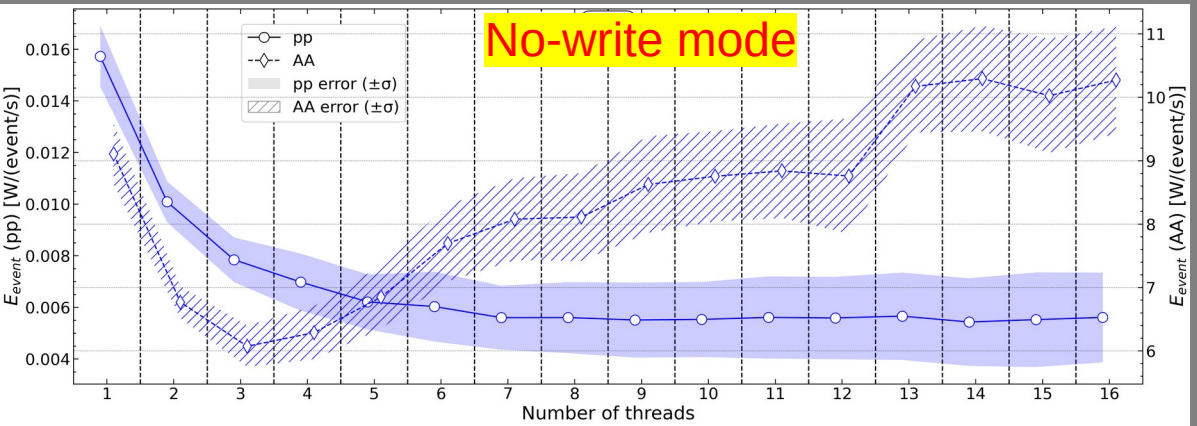

AMD EPYC 4585PX: pp (500k) / AA (10k)
Time pp (s)
Time AA (s)
pp Events/s
AA Events/s
Avg pp Power (W)
Avg AA Power (W)
Averages
Avg pp time: 99.5s
Avg AA time: 1099.3s
Avg pp events/s: 5,218.3
Avg AA events/s: 9.23
Avg pp power: 95.6W
Avg AA power: 87.1W
187.8 s 126.2 s 106.5 s 98.9 s 92.3 s 90.7 s 90.0 s 88.4 s 88.8 s 88.1 s 89.5 s 89.9 s 89.3 s 89.6 s 88.0 s 88.3 s
1626 s 1108 s 940 s 928 s 1006 s 1064 s 1123 s 1102 s 1126 s 1123 s 1099 s 1083 s 1036 s 1044 s 1110 s 1072 s
2662 3968 4694 5055 5417 5515 5554 5655 5633 5673 5587 5562 5602 5578 5681 5662
67.8 76.7 85.3 90.9 92.6 97.5 96.6 98.9 100.6 100.0 101.9 104.3 103.8 103.4 104.2 104.6
6.2 9.0 10.6 10.8 9.9 9.4 8.9 9.1 8.9 8.9 9.1 9.2 9.7 9.6 9.0 9.3
Events per Second (AA)
Events per Second (pp)
Average Power (W) for pp
Number of Threads


AMD EPYC 4585PX: pp (500k) / AA (10k)
Time pp (s)
Time AA (s)
pp Events/s
AA Events/s
Avg pp Power (W)
Avg AA Power (W)
Averages
Avg pp time: 99.5s
Avg AA time: 1099.3s
Avg pp events/s: 5,218.3
Avg AA events/s: 9.23
Avg pp power: 95.6W
Avg AA power: 87.1W
Average Power (W) for AA
Events per Second (AA)
Events per Second (pp)
Average Power (W) for pp
Number of Threads
187.8 s 126.2 s 106.5 s 98.9 s 92.3 s 90.7 s 90.0 s 88.4 s 88.8 s 88.1 s 89.5 s 89.9 s 89.3 s 89.6 s 88.0 s 88.3 s
1626 s 1108 s 940 s 928 s 1006 s 1064 s 1123 s 1102 s 1126 s 1123 s 1099 s 1083 s 1036 s 1044 s 1110 s 1072 s

Sustainability

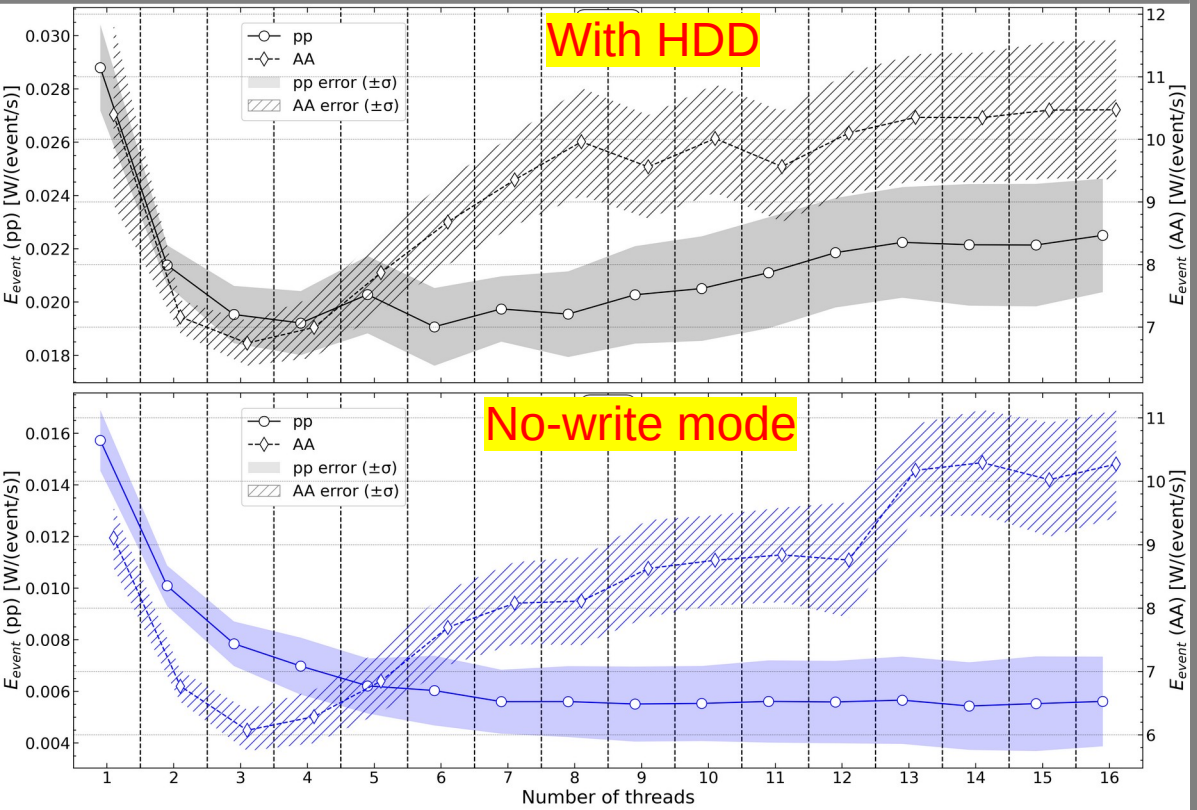
- By introducing a way to measure the cost of each event, E = P/T, we can plot this as a function of the number of threads and find the most optimal level of paralleliaztion.
- Tuning a MC event generator is a computationally intensive task, requiring many runs and thus a lot of power
- Nowadays, especially with the High-Luminosity LHC era, the required scale of simulated data is expected to increase significantly
- By optimizing the simulations, one can find a ‘carbon-aware’ way to run MC event generators, leading to more sustainable efficiency

AMD EPYC 4585PX

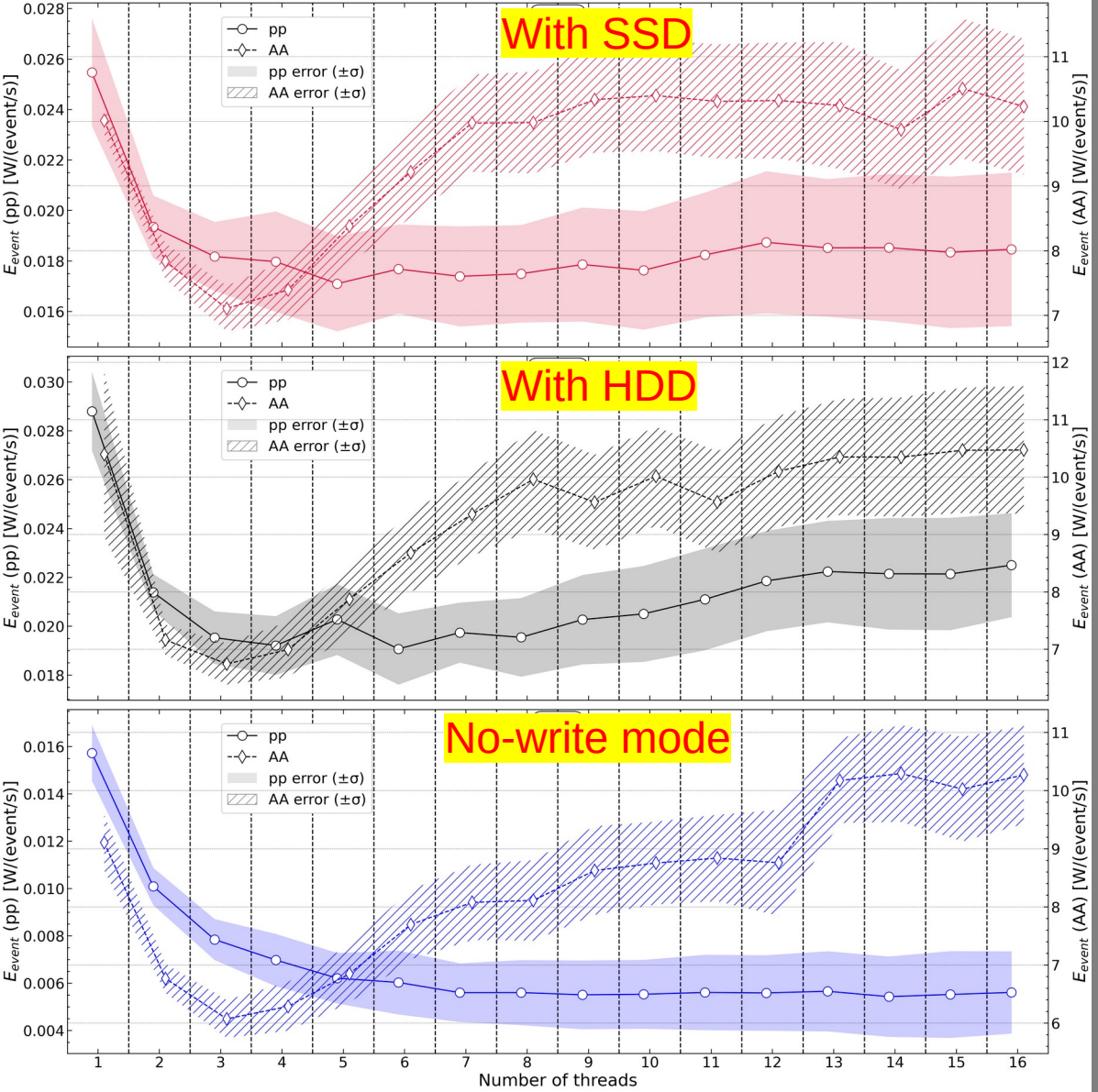


- Power consumption was measured every second and the average was considered.
- The error comes from how the power consumption was measured.
- Most optimal doesn’t mean the fastest
- Little difference between the runtimes of SSD and HDD.
- No-write included as reference, no files were created, pure speed test of the CPU

AMD EPYC 4585PX



- Power consumption was measured every second and the average was considered.
- The error comes from how the power consumption was measured.
- Most optimal doesn’t mean the fastest
- Little difference between the runtimes of SSD and HDD.
- No-write included as reference, no files were created, pure speed test of the CPU



AMD EPYC 4585PX

- Power consumption was measured every second and the average was considered.
- The error comes from how the power consumption was measured.
- Most optimal doesn't mean the fastest
- Little difference between the runtimes of SSD and HDD.
- No-write included as reference, no files were created, pure speed test of the CPU

Results

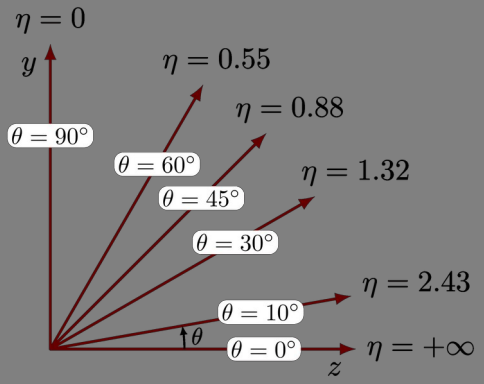
- Significant improvement over single thread mode in the case of HIJING++ at both the fastest and most optimal thread counts (25-51 %)

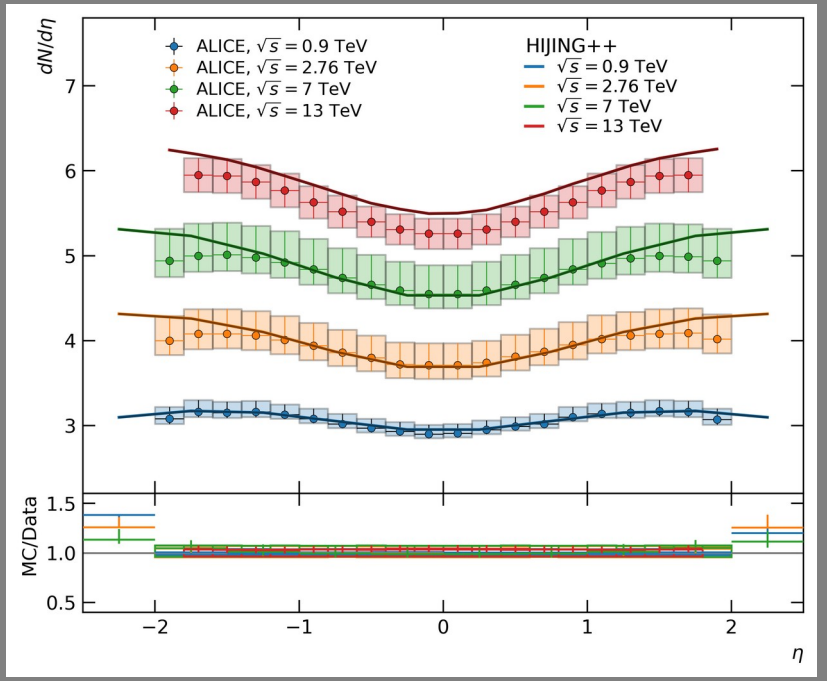
- Overall power consumption was calculated from:
  - 1024 configs
  - 5 million events each
  - 10 iterations
  - 6 energies
- In the case of AA collisions it would be 140-180 times more costly
- Simplified estimates, no Rivet analysis or Professor usage included

- AMD EPYC 4585PX (pp case)
  - Fastest: 1566 kWh – 15 threads
  - Optimal: 1458 kWh – 5 threads
  - Reduction: 6.8 %

Recent tuning results of HIJING++

- The HEP Toolbox was used which made the process convenient
- Preliminary results for pseudorapidity distribution at 4 different energies
- Pseudorapidity is used to measure the angle of the particles' trajectory relative to the beam, η=-ln(tan(θ/2))
- 6 parameters were used in the tuning: 0.9 TeV, 2.76 TeV, 7 TeV, 13 TeV





Usage of the Toolbox

- Tutorial available on the gitlab*
- Downloading the image: **docker pull 77rev/proripy:4.4**
  or ssh at the end if you need that verison.
- Run:
  docker run -it -v $HOME/Work:/home/devuser/Work 77rev/proripy:4.4
- It gives you a terminal inside the docker container, to run the same container again: **docker start *[container name]***
- Then to get the interactive terminal:
  docker exec -it --user root *[container name]* bash
- To start the ssh version:
  docker run -it -p 1779 77rev/proripy:ssh
  it will be accessible on the 1779 port and you can get the IP via the docker inspect command.

Conclusion

- Toolbox (Docker image) with Pythia8, Rivet, ROOT, etc. that work out-of-the-box.
- MC Event generators require optimization for sustainability ⇨ CPU Benchmarking required.
- Newer CPUs are a lot more efficient for these tasks, optimizable via parallelism.
- Tuning of MC event generators is convenient via the Toolbox.
- Gitlab page will be available shortly as well as publication on arXiv and EPJC.

Thank you for your attention!

Acknowledgments
- Team: Gábor Bíró, Gergely Gábor Barnaföldi, Gábor Papp
- Grants:
  - NKKP ADVANCED_25-153456
  - 2025-1.1.5-NEMZ_KI-2025-00005
  - 2025-1.1.5-NEMZ_KI-2025-00013
  - 2024-1.2.5-TÉT-2024-00022

WSCLAB
Wigner Scientific Computing Laboratory

NTT DATA

Check out the Docker Hub

This presentation is uploaded to Indico.