Studies of the power consumption for LHCb HLT1

Volodymyr Svintozelskyi

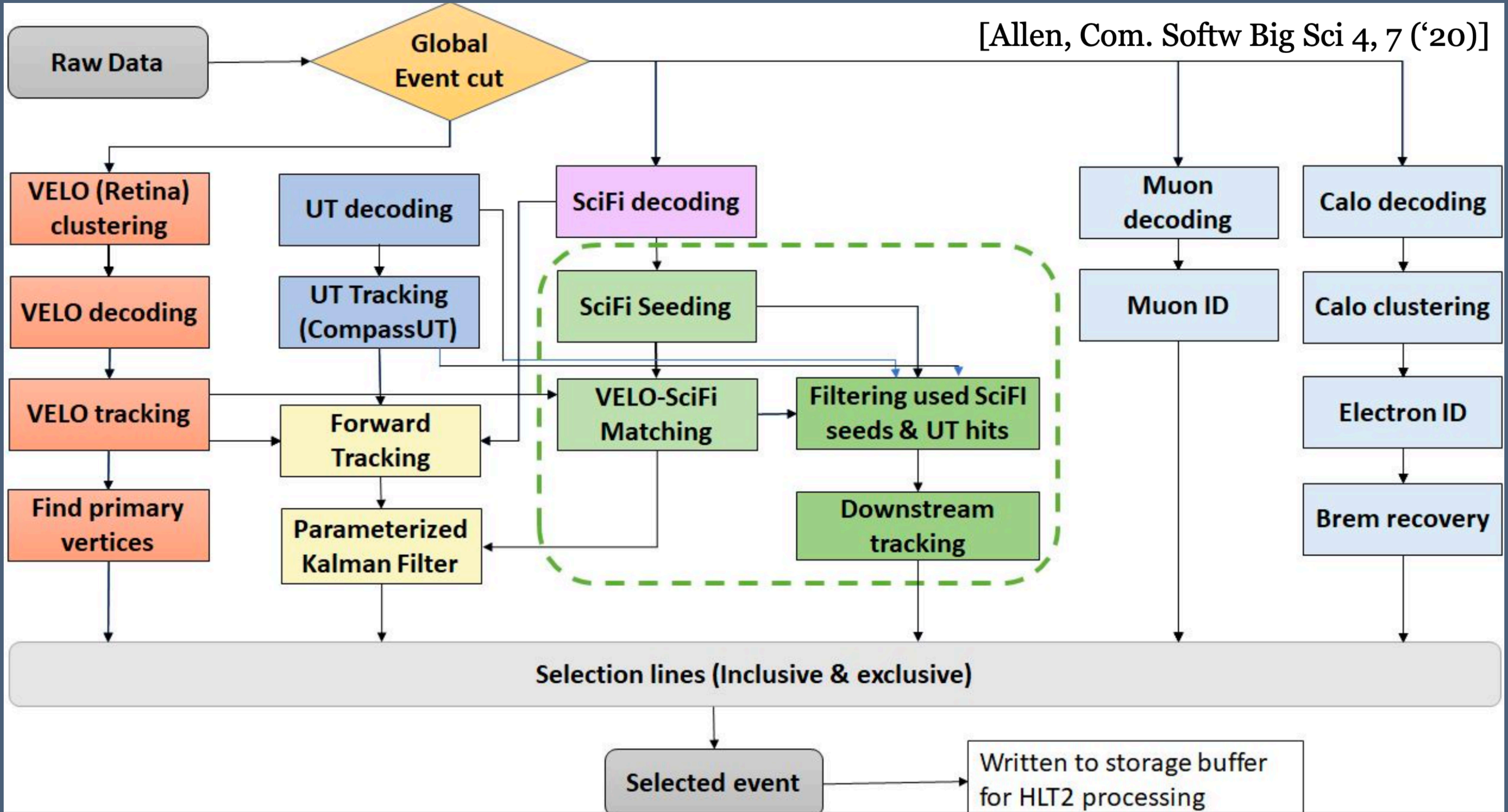
IFIC, Univ. of Valencia and CSIC (ES)
Taras Shevchenko National University of Kyiv

Supervisor: Arantza De Oyanguren Campos

[LHCB-TDR-021]

Real-time analysis at **LHCb**

- Allen: the LHCb high-level trigger 1 (HLT1) application on GPUs
- Fast detector reconstruction in O(300) Nvidia RTX A5000
- More than 250 algorithms with a high level of parallelization form an execution sequence
- Allen is an example software, but the conclusions are general



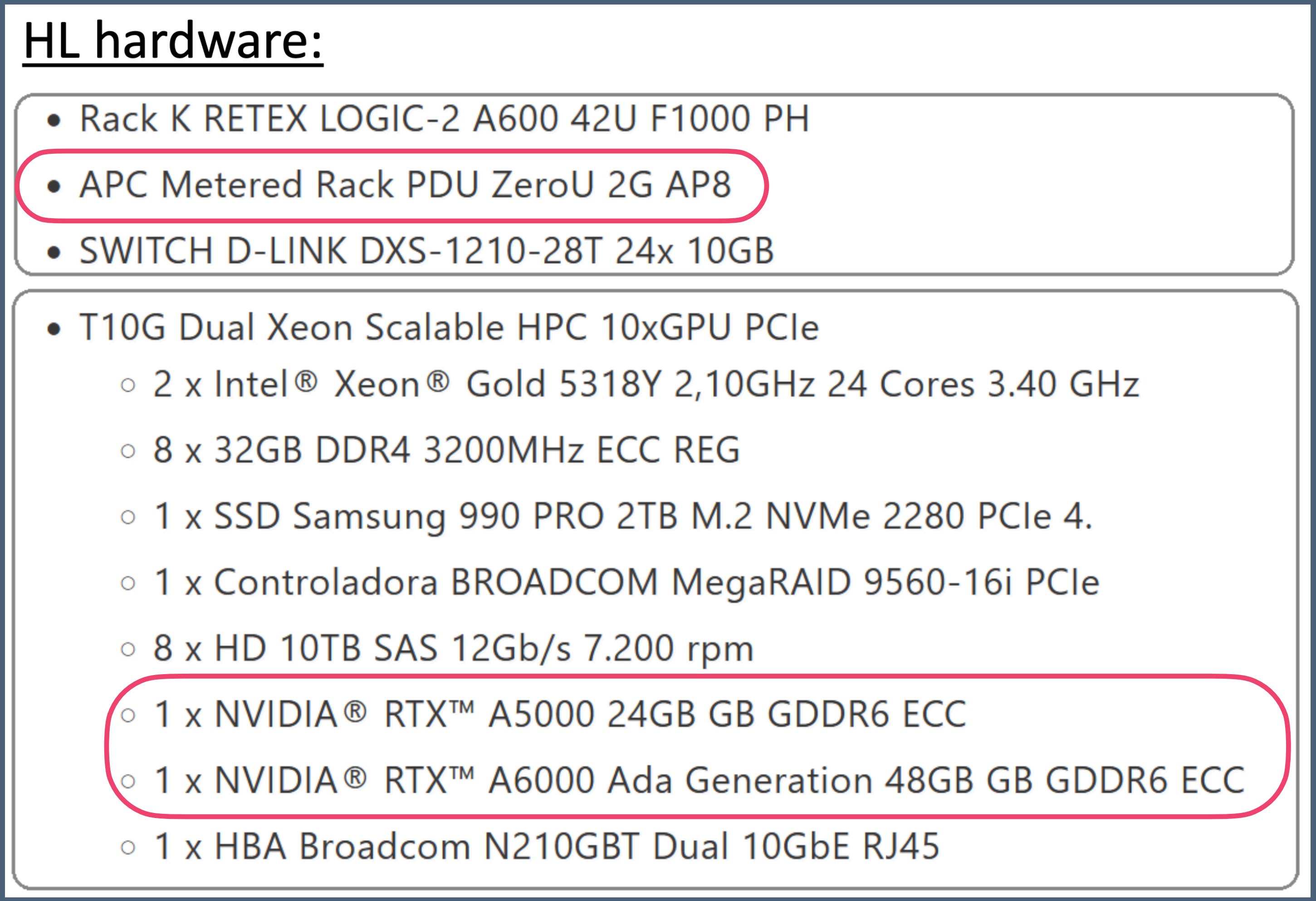
[Allen, Com. Softw Big Sci 4, 7 ('20)]

The **HIGH-LOW** project at Valencia

Design of High-Performance Algorithms for Low-Power Sustainable Hardware for LHC Experiments and Their Upgrades

- Transversal project: ATLAS and LHCb experiments
- PIs: Luca Fiorini (ATLAS), & Arantza Oyanguren (LHCb)
- Funded by the Spanish Ministry of Science and Innovation *(TED2021-130852B-I00)*
- About 10 people (physicists + engineers + students)

Aim: Benchmarking new hardware architectures and developing fast and high efficient algorithms with reduced power consumption



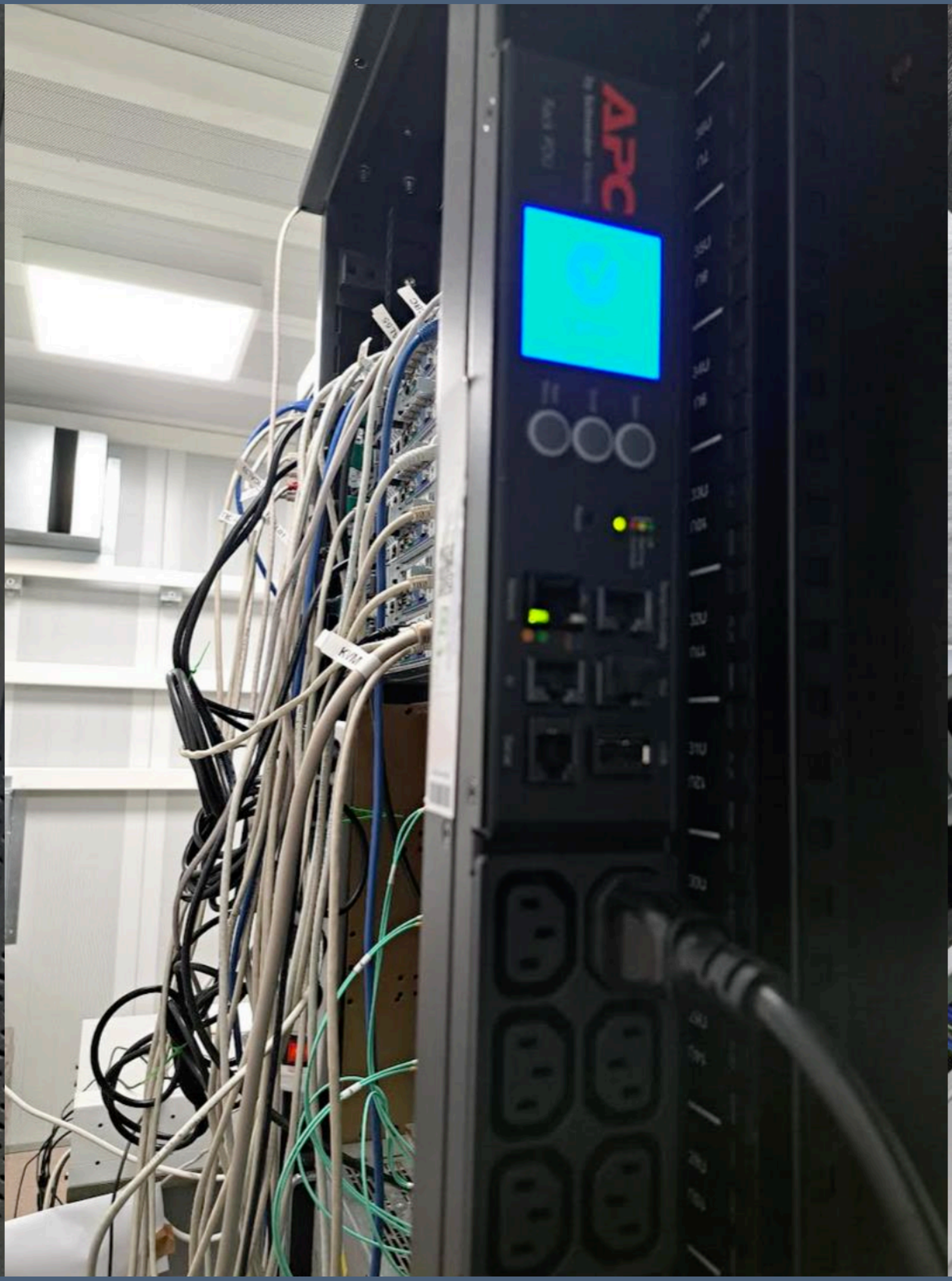
HL hardware:

- Rack K RETEX LOGIC-2 A600 42U F1000 PH
- APC Metered Rack PDU ZeroU 2G AP8
- SWITCH D-LINK DXS-1210-28T 24x 10GB

- T10G Dual Xeon Scalable HPC 10xGPU PCIe
  - ○ 2 x Intel® Xeon® Gold 5318Y 2,10GHz 24 Cores 3.40 GHz
  - ○ 8 x 32GB DDR4 3200MHz ECC REG
  - ○ 1 x SSD Samsung 990 PRO 2TB M.2 NVMe 2280 PCIe 4.
  - ○ 1 x Controladora BROADCOM MegaRAID 9560-16i PCIe
  - ○ 8 x HD 10TB SAS 12Gb/s 7.200 rpm
  - ○ 1 x NVIDIA® RTX™ A5000 24GB GB GDDR6 ECC
  - ○ 1 x NVIDIA® RTX™ A6000 Ada Generation 48GB GB GDDR6 ECC
  - ○ 1 x HBA Broadcom N210GBT Dual 10GbE RJ45

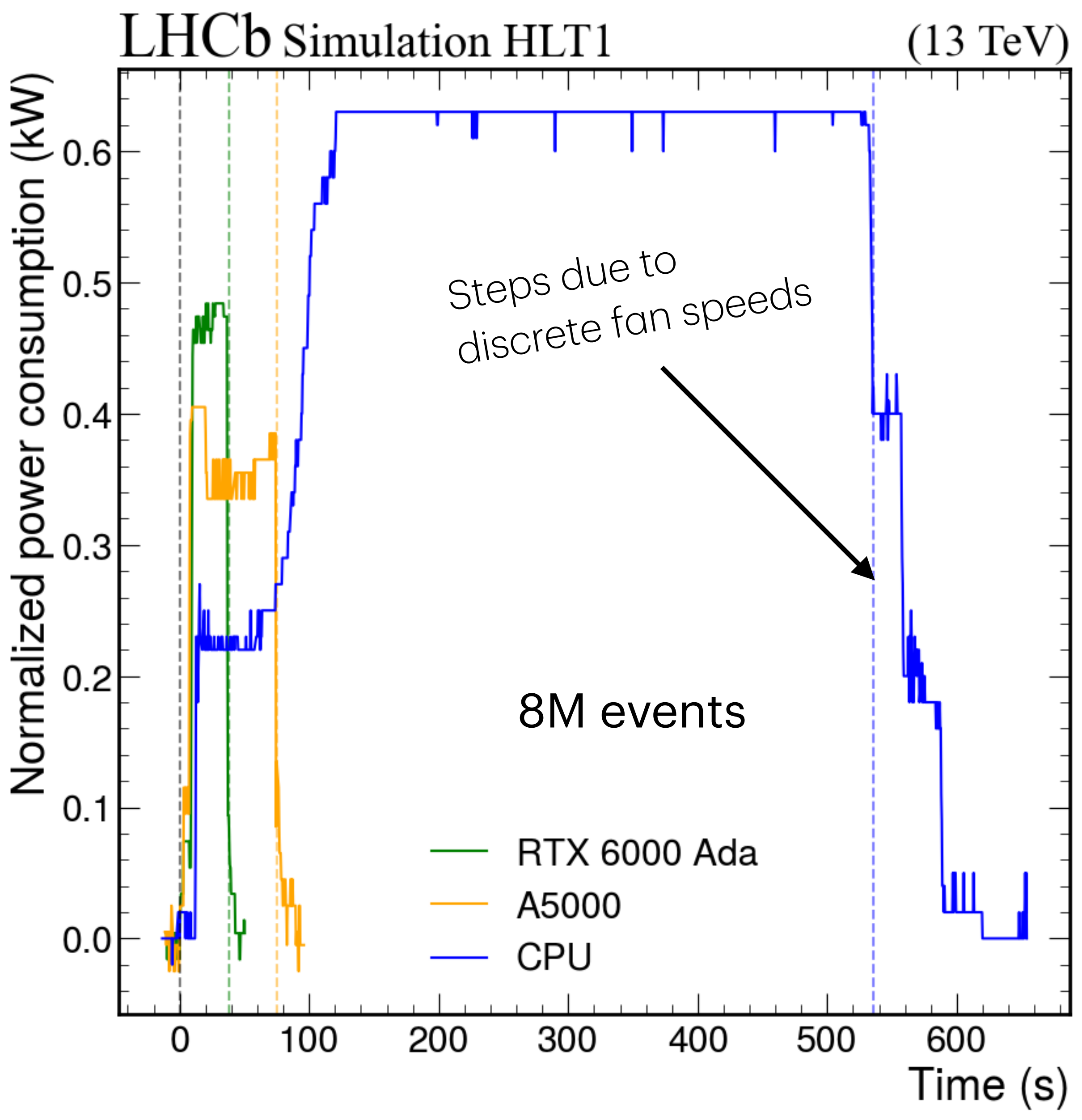
The **HIGH-LOW** project at Valencia

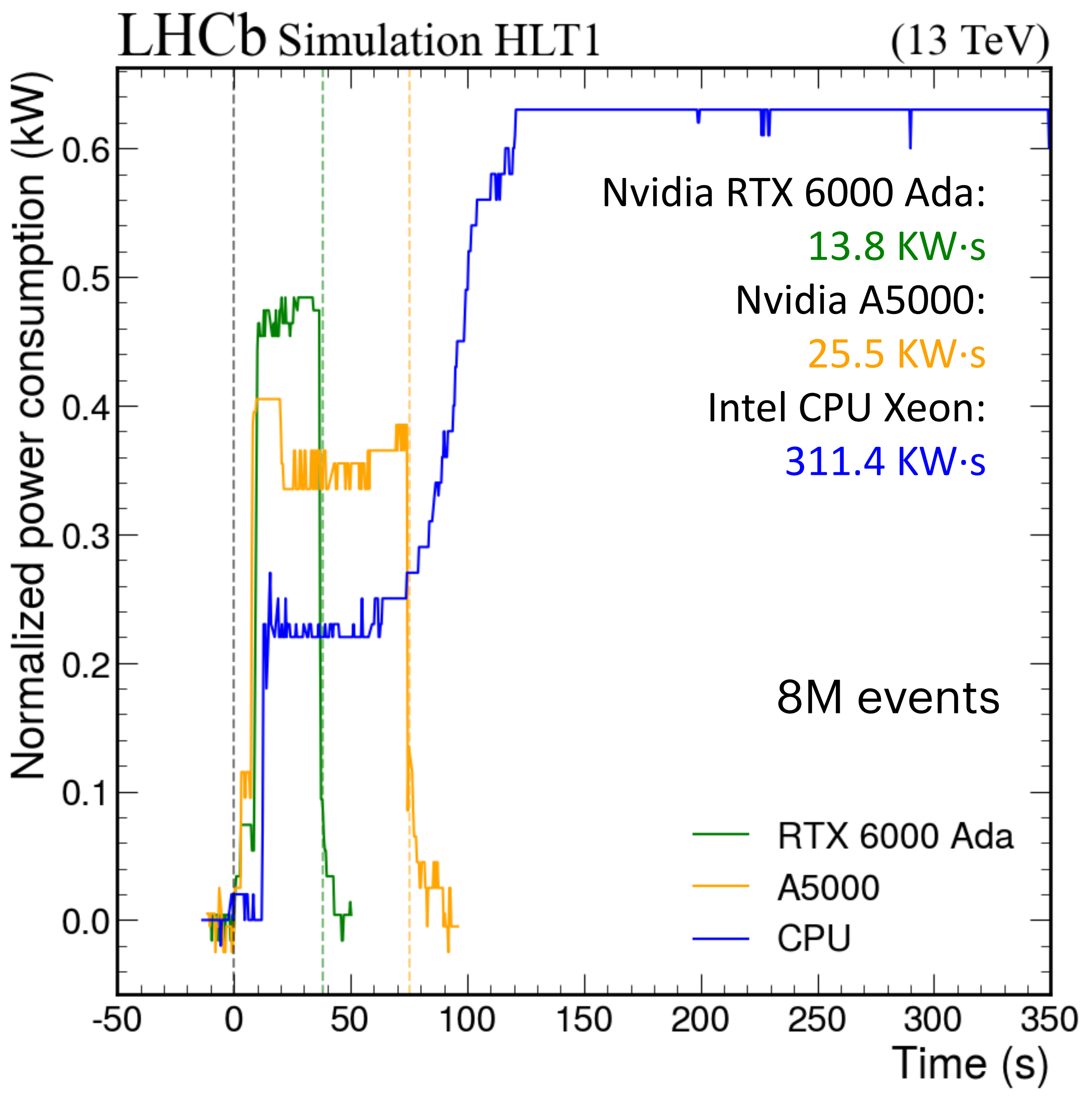
IFIC
INSTITUT DE FÍSICA CORPUSCULAR



Hardware power efficiency

What is the effect of faster hardware on overall power consumption?

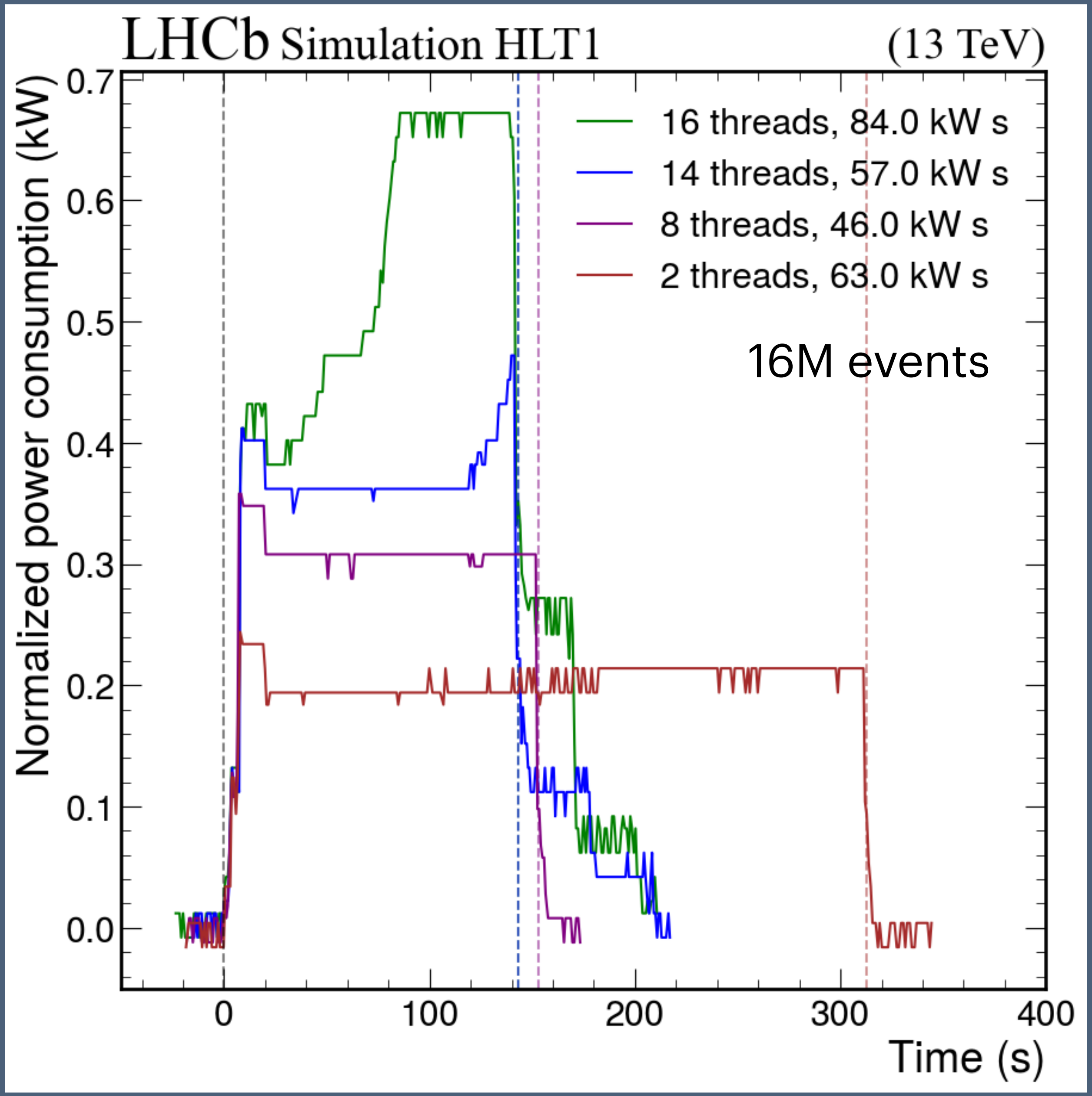


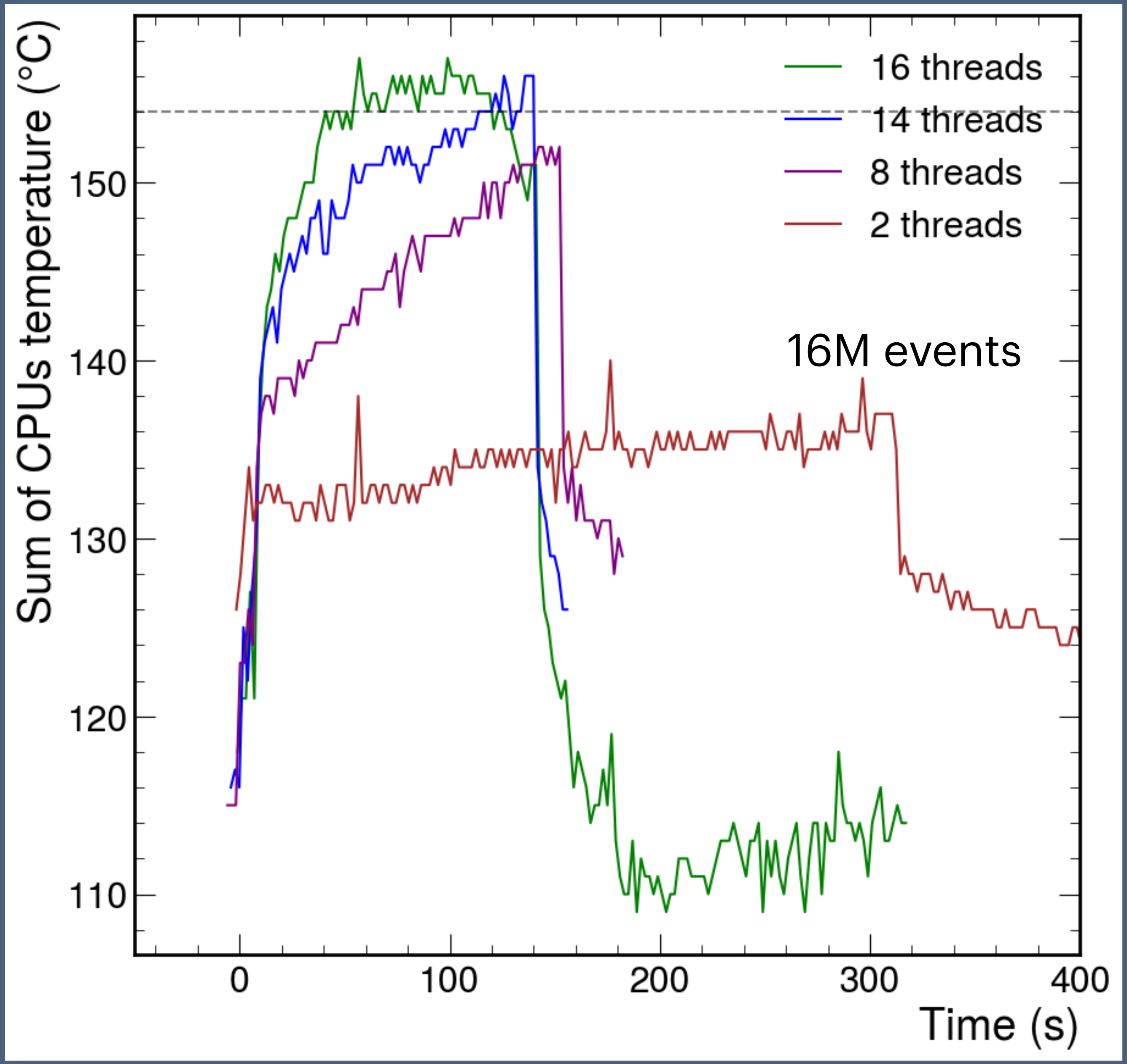


The LHCb decision to run HLT1 on GPUs may have saved up to a factor 10 in energy consumption!

Hardware utilisation

Higher number of GPU streams leads to faster CPU heating → higher consumption

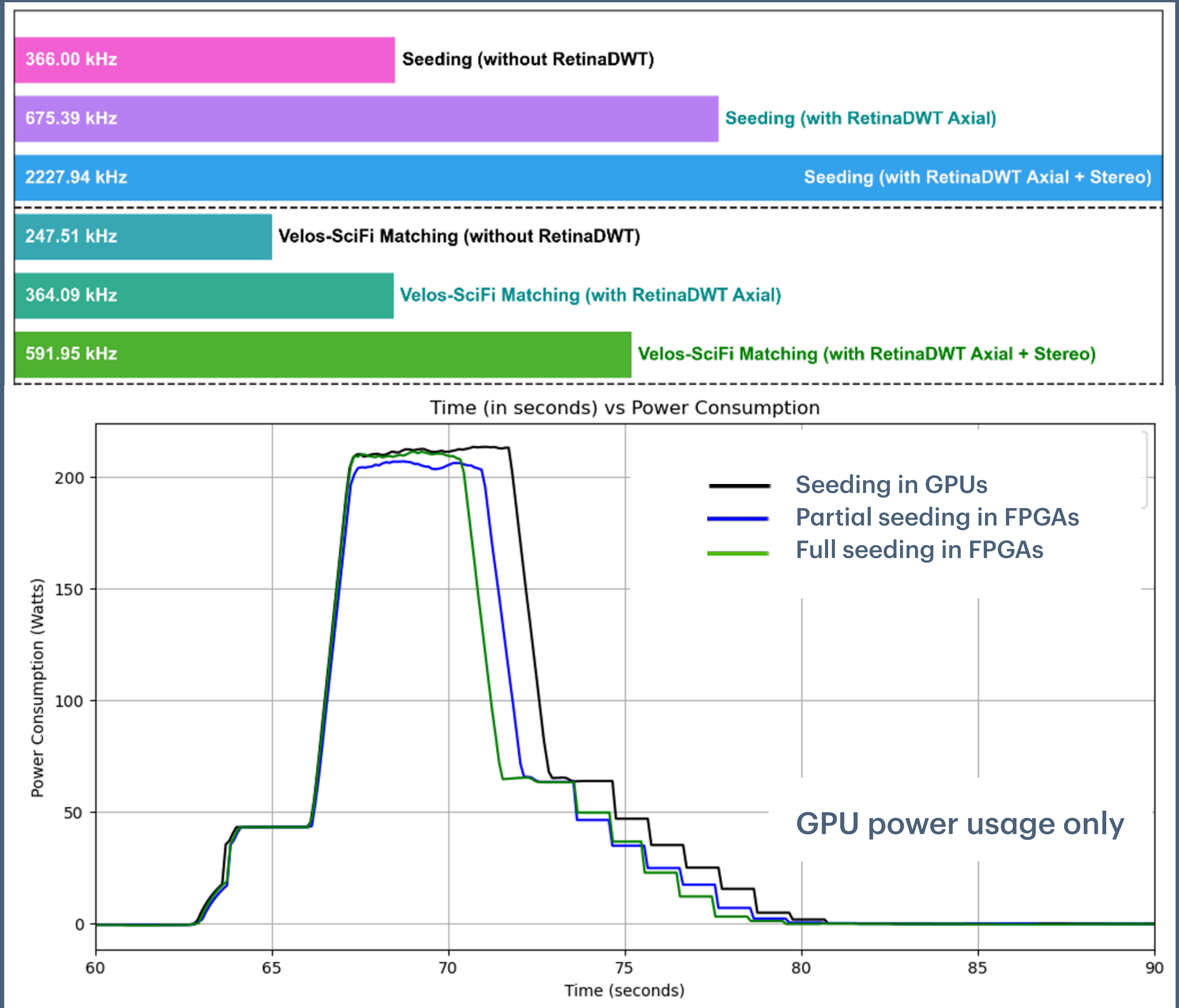




* Part of Allen algorithms needs to be executed on CPU, which leads to CPU heating and cooler fan speed ramp up

FPGA's for less consumption?

The same idea can be extended to FPGA:



Seeding algorithm for making tracklets in the last LHCb tracker (SciFi) in FPGAs:

Throughput increases by 30%

→ Saving 6.2 mW · s/event

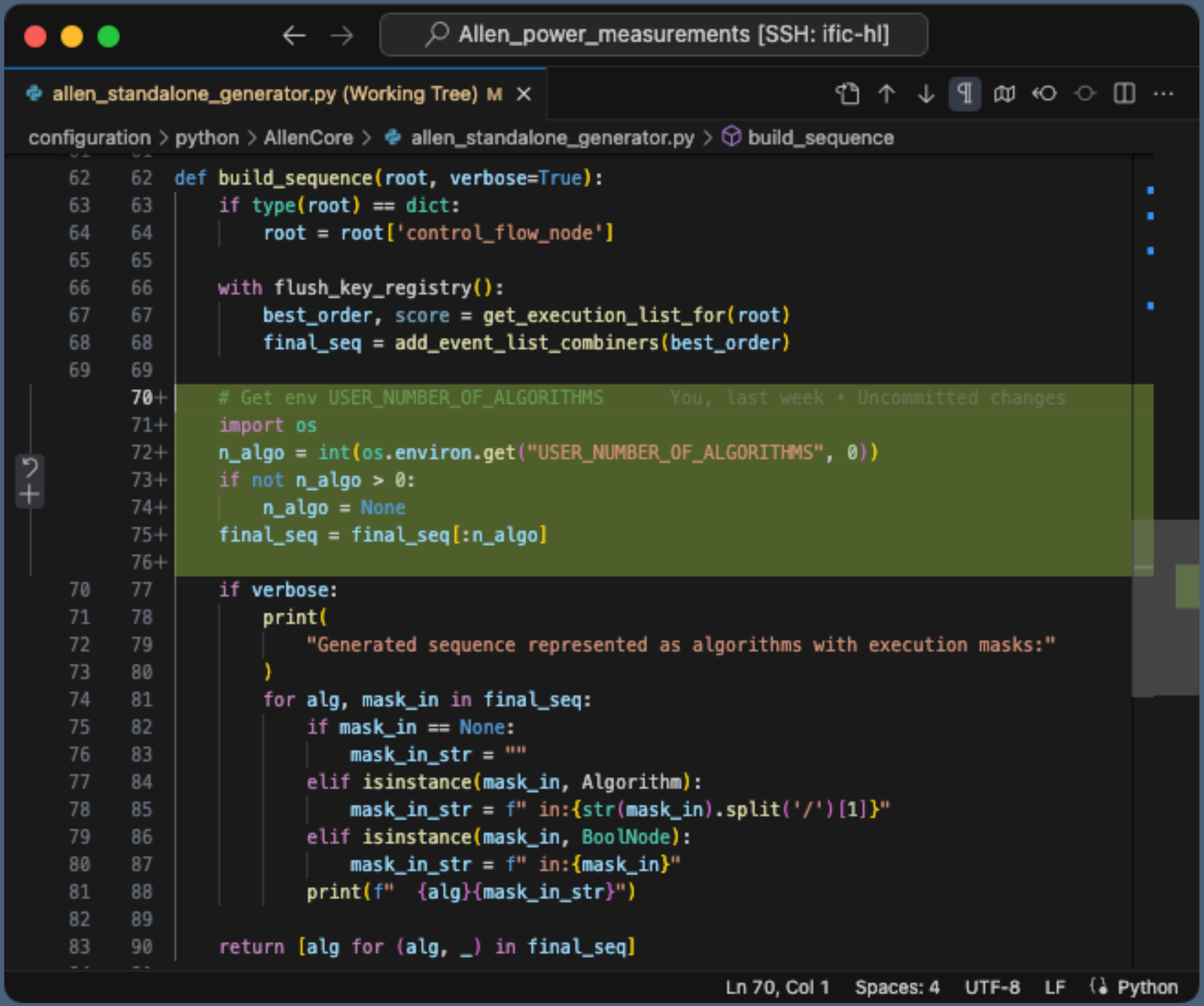
(for 30 MHz rate: 186kW/s)

Use hybrid systems to take benefits of each one

Sustainability of real-time analysis at 5 TB/s data rate - Sustainable HEP 2024

Per-algorithm power consomution

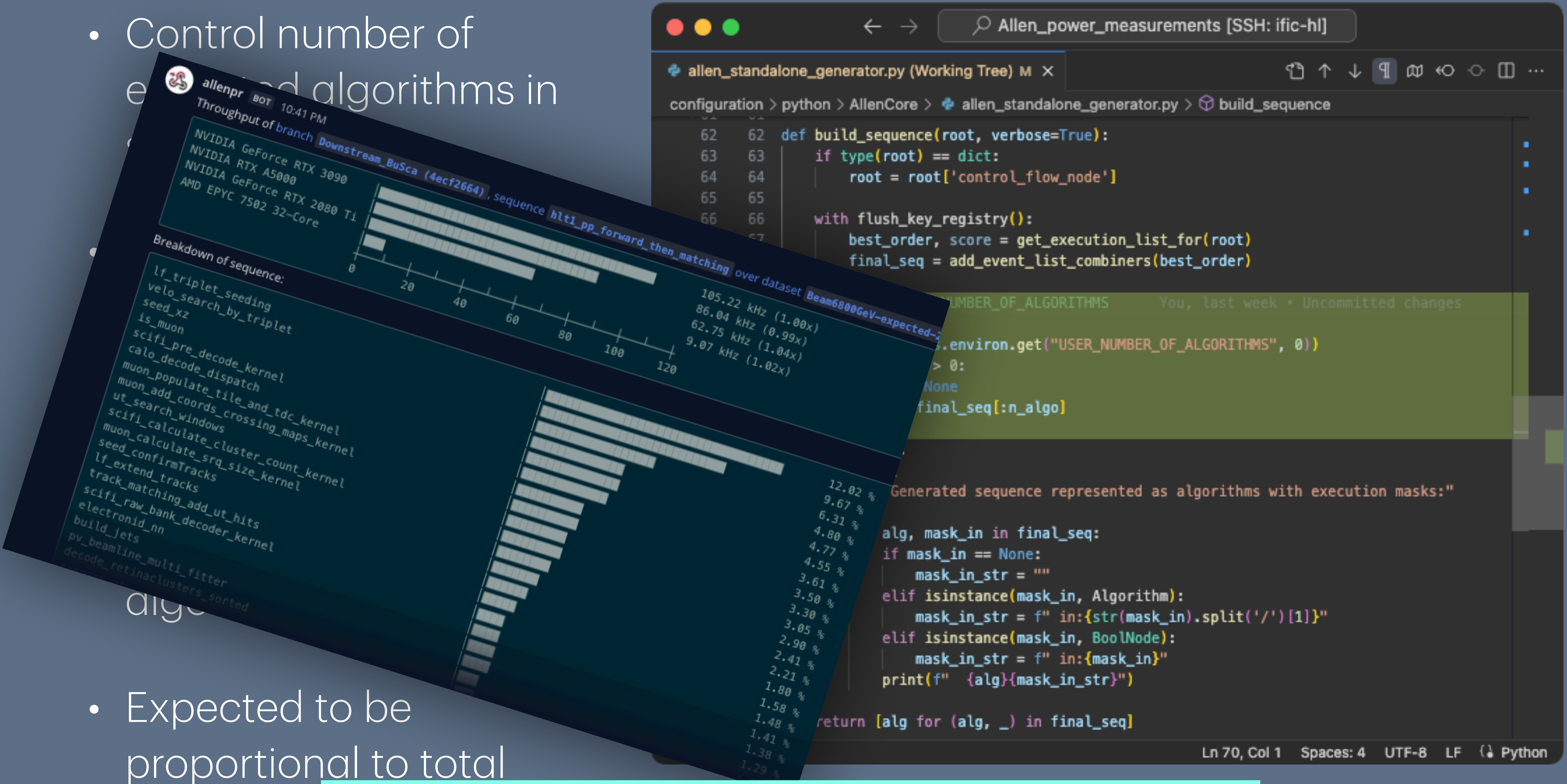
- Control number of executed algorithms in sequence
- Run power consumption test for first N and $N+1$ algorithms
- The difference is the consumption gain from algorithm N
- Expected to be proportional to total execution time


Allen_power_measurements [SSH: ific-hl]
allen_standalone_generator.py (Working Tree) M ×
configuration > python > AllenCore > allen_standalone_generator.py > build_sequence
You, last week • Uncommitted changes
Ln 70, Col 1
Spaces: 4
UTF-8
LF
Python

```python
def build_sequence(root, verbose=True):
    if type(root) == dict:
        root = root['control_flow_node']

    with flush_key_registry():
        best_order, score = get_execution_list_for(root)
        final_seq = add_event_list_combiners(best_order)

    # Get env USER_NUMBER_OF_ALGORITHMS
    import os
    n_algo = int(os.environ.get("USER_NUMBER_OF_ALGORITHMS", 0))
    if not n_algo > 0:
        n_algo = None
    final_seq = final_seq[:n_algo]

    if verbose:
        print(
            "Generated sequence represented as algorithms with execution masks:"
        )
        for alg, mask_in in final_seq:
            if mask_in == None:
                mask_in_str = ""
            elif isinstance(mask_in, Algorithm):
                mask_in_str = f" in:{str(mask_in).split('/')[1]}"
            elif isinstance(mask_in, BoolNode):
                mask_in_str = f" in:{mask_in}"
            print(f"  {alg}{mask_in_str}")

    return [alg for (alg, _) in final_seq]
```

Per-algorithm power consomution

- Control number of e d algorithms in
- algo
- Expected to be proportional to total execution



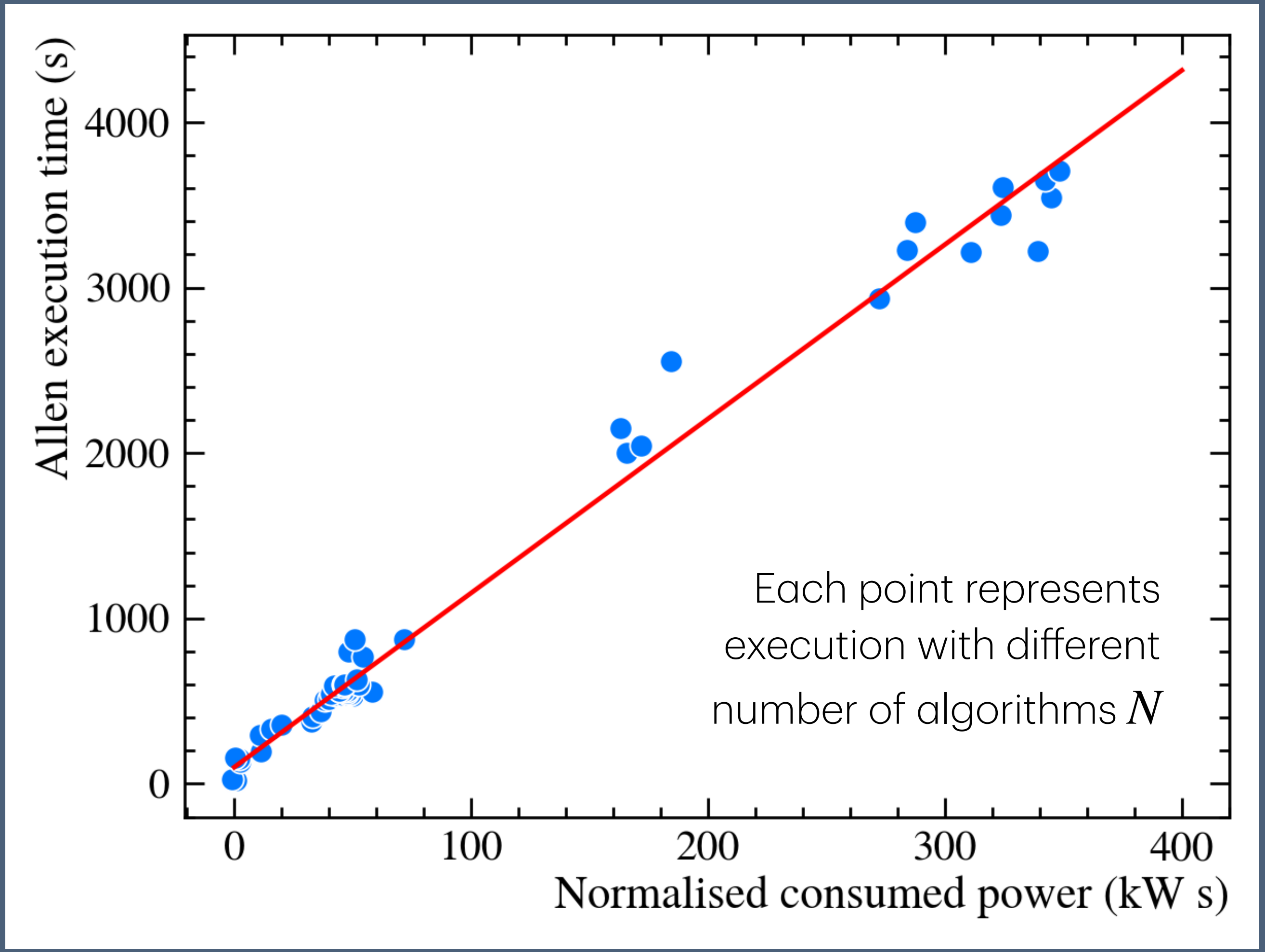
```
allenpr BOT 10:41 PM
Throughput of branch Downstream_BuSca (4ecf2664), sequence hlt1_pp_forward_then_matching over dataset Beam6800GeV-expected-
NVIDIA GeForce RTX 3090       105.22 kHz (1.00x)
NVIDIA RTX A5000               86.04 kHz (0.99x)
NVIDIA GeForce RTX 2080 Ti     62.75 kHz (1.04x)
AMD EPYC 7502 32-Core           9.07 kHz (1.02x)
                  0   20   40   60   80   100   120

Breakdown of sequence:
lf_triplet_seeding                        12.02 %
velo_search_by_triplet                     9.67 %
seed_xz                                    6.31 %
is_muon                                    4.80 %
scifi_pre_decode_kernel                    4.77 %
calo_decode_dispatch                       4.55 %
muon_populate_tile_and_tdc_kernel          3.61 %
muon_add_coords_crossing_maps_kernel       3.50 %
ut_search_windows                          3.30 %
scifi_calculate_cluster_count_kernel       3.05 %
muon_calculate_srq_size_kernel             2.90 %
seed_confirmTracks                         2.41 %
lf_extend_tracks                           2.21 %
track_matching_add_ut_hits                 1.80 %
scifi_raw_bank_decoder_kernel              1.58 %
electronid_nn                              1.48 %
build_jets                                 1.41 %
pv_beamline_multi_fitter                   1.38 %
decode_retinaclusters_sorted               1.29 %
```

```
Allen_power_measurements [SSH: ific-hl]
allen_standalone_generator.py (Working Tree) M ×
configuration > python > AllenCore > allen_standalone_generator.py > build_sequence

def build_sequence(root, verbose=True):
    if type(root) == dict:
        root = root['control_flow_node']

    with flush_key_registry():
        best_order, score = get_execution_list_for(root)
        final_seq = add_event_list_combiners(best_order)

UMBER_OF_ALGORITHMS      You, last week • Uncommitted changes

.environ.get("USER_NUMBER_OF_ALGORITHMS", 0))
> 0:
None
final_seq[:n_algo]


Generated sequence represented as algorithms with execution masks:"

alg, mask_in in final_seq:
if mask_in == None:
    mask_in_str = ""
elif isinstance(mask_in, Algorithm):
    mask_in_str = f" in:{str(mask_in).split('/')[1]}"
elif isinstance(mask_in, BoolNode):
    mask_in_str = f" in:{mask_in}"
print(f"  {alg}{mask_in_str}")

return [alg for (alg, _) in final_seq]

Ln 70, Col 1   Spaces: 4   UTF-8   LF   {} Python
```

Inspired by Allen algorithm breakdown!

Per-algorithm power consomution

- Due to time limitations, only specific N were chosen for studies
- The HighLow machine is a shared resource - required to run overnight to minimize effect from other users
(but still there)
- Multiple runs are necessary for stat. error estimation
- No anomaly is detected → no significant performance drops?



Each point represents execution with different number of algorithms N

Summary

- Move towards heterogeneous computing systems
- Use the best and more efficient available hardware (vs €)
- Optimize the utilization of the hardware
- Keep an eye on your power consumption!

- Planned to make Allen support more hardware architectures (ARM; Intel GPUs)

- Work is still ongoing ☺️

Thanks for your attention!