Validation of GPU-accelerated parton cascade with elliptic flow

Qing-Jun Liu

Zhiyuan School of Liberal Arts,
Beijing Institute of Petro-chemical Technology, 102617

第十五届QCD相变与重离子物理研讨会，珠海，2023年12月14-19日

Outline

1. Develop PCG for acceleration of parton cascade
2. Validate PCG with elliptic flow, together with dn/dη
3. Conclusions

- **There is a need for the acceleration of event generation**

1. **Compare with measured data and carry out systematics study STAR, ALICE, and CEE at Lan-Zhou China, etc..**
   STAR Collab., arXiv:2308.16846v1

2. **Theoretical Lab. for predicting the new HIJING, ZPC, ART, VUU/BUU, AMPT, UrQMD, CMC,etc..**
   M. Bleicher and E. Bratkovskaya,
   https://doi.org/10.1016/j.ppnp.2021.103920.

3. **Large scale simulations cost tremendous amount of CPU time**

So, acceleration of event generators is interesting

- **GPU can be a good choice, more powerful than CPU**

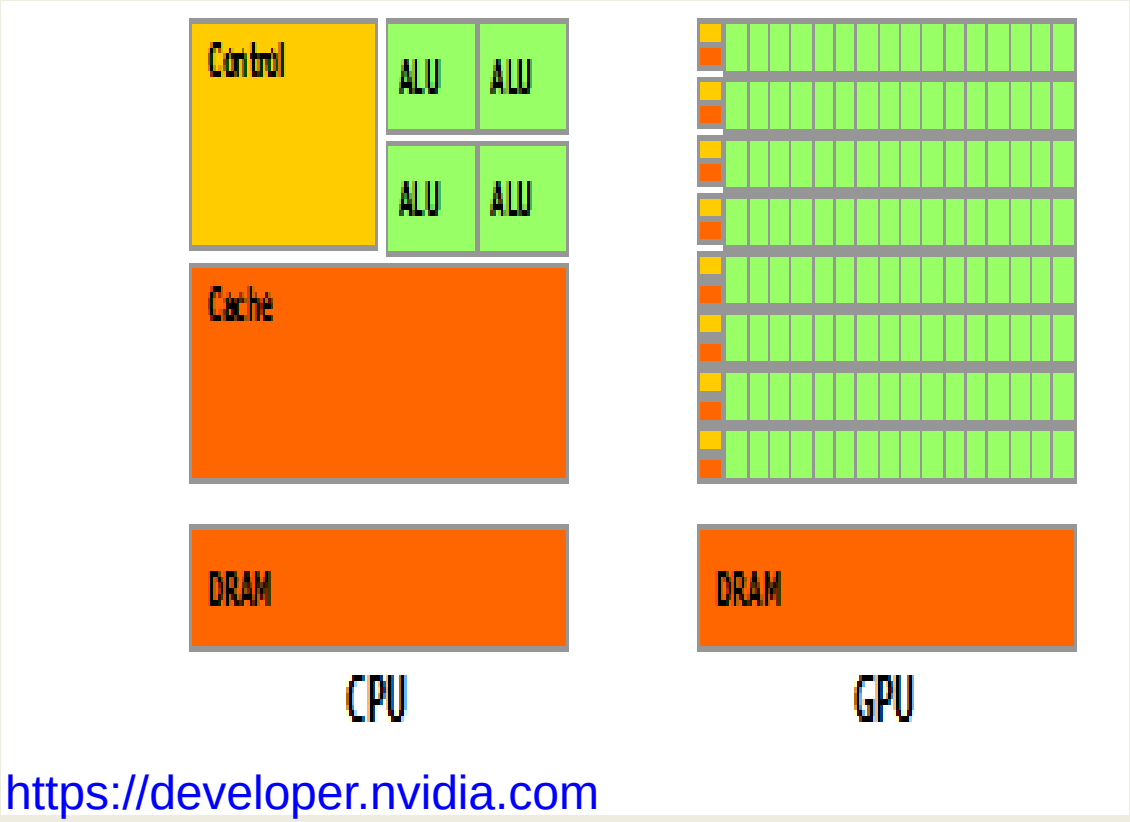
GPUs:

V100, A100, and DCU a GPU-like accelerator

GPU vs CPU:

(1) More ALUs

(2) Parallel SMs

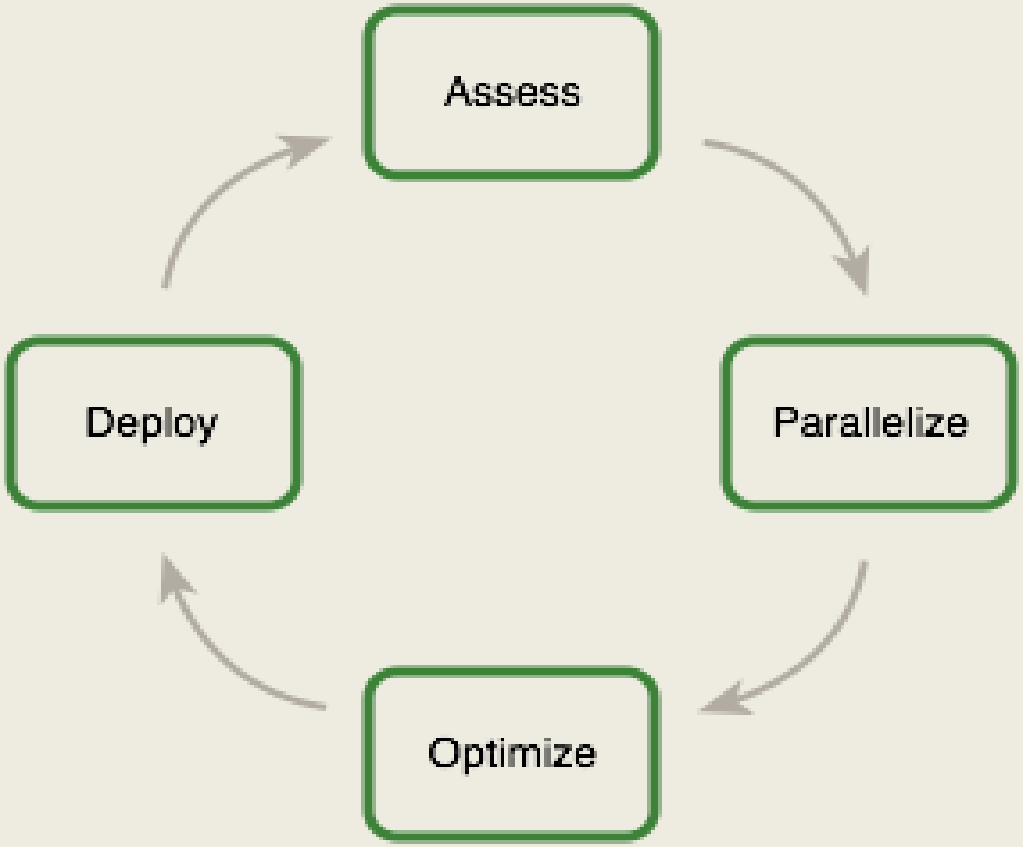


Naturally, one want to try GPUs for the simulation

Develop PCG (Parton Cascade on GPU)

- Methodology APOD

A for Assess
P for Parallelize
O for Optimize
D for deploy



https://developer.nvidia.com/blog/assess-parallelize-optimize-deploy

Nvidia advocates APOD for transporting software onto GPU, and we adopt APOD to develop PCG

- **Assess**

(1) In AMPT there are three modules **HIJING , ZPC , ART**

(2) ZPC is the most compute-intensive at top LHC energy

(3) What ZPC does is to simulate parton cascade process , via successive **two-parton collision detection and two-parton collisions**

For collision detection, **parton pairs can be processed in parallel** on GPUs.

The hot spot is successive collision detection

- **Assess**

Recoding parton cascade based on ZPC

a) Simulate elastic scattering on GPU CUDA C or on CPU

b) Detect two-parton collision on GPU

code in CUDA C, consists of several kernels to facilitate

parallel collision detection, according to criteria stated in ZPC

Note: kernels are CUDA C functions that can run on GPU.

Hence, based on ZPC we coded PCG（ Parton Cascade on GPU）

- **Parallelize**

Algorithm

1. compute and save colliding time and indices
   for all of the 0.5N(N-1) pairs

2. Decide which pair to commit the first collision

3. let the pair of partons collide
   So the pair acquire new space and momentum information

4. update colliding time
   for the pairs, each of which involve a parton that just collided

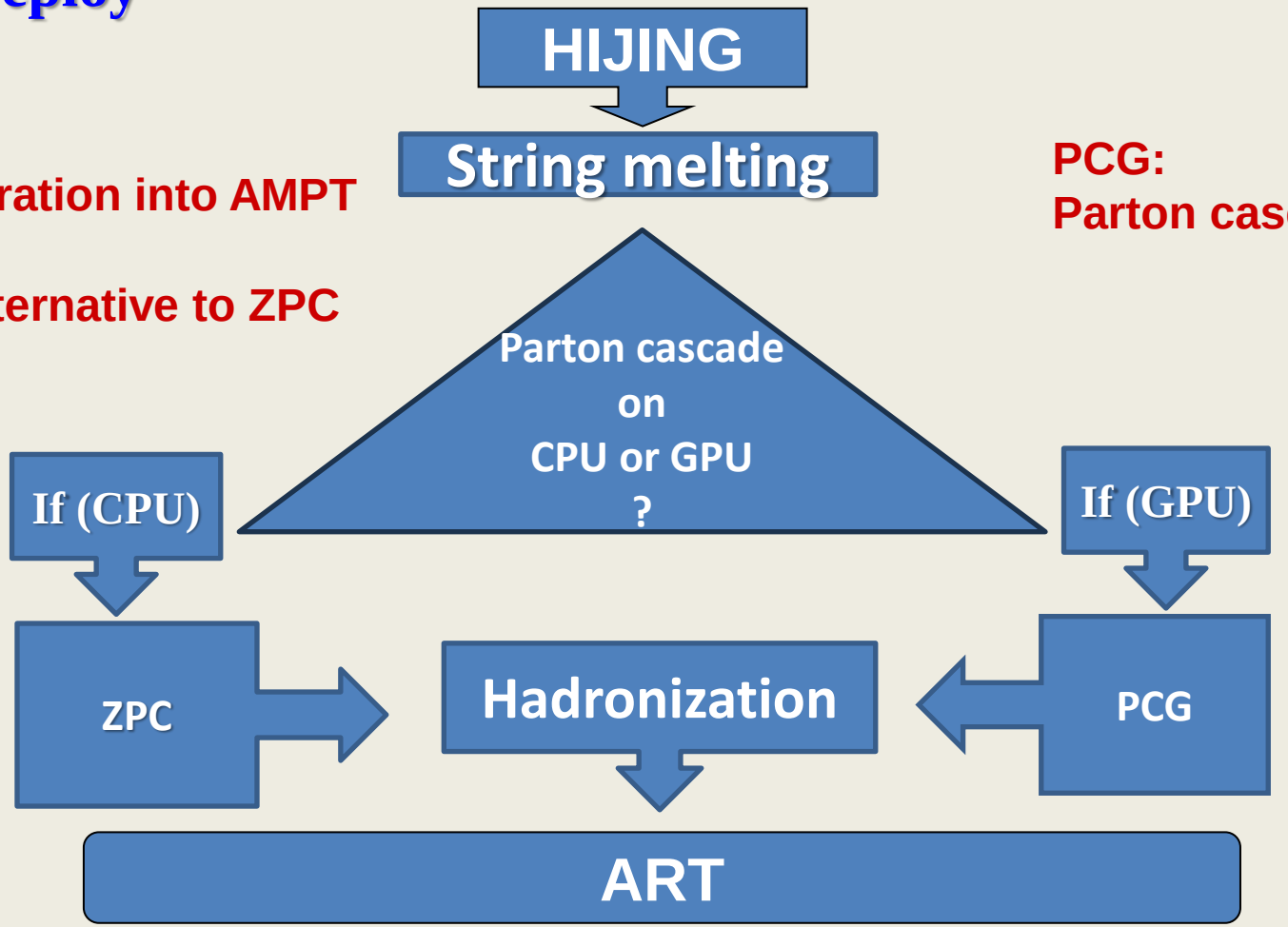
5. decide which pair to commit the next collision

6. If time for next collision > thresholdTime, PCG terminates, otherwise go to step 3 and repeat steps 3 to 6 sequentially.

PCG contains several kernels written in CUDA C, and with/without Fortran subroutines for simulating collisions

- **Deploy**

integration into AMPT
as
an alternative to ZPC



PCG:
Parton cascade on GPU,

PCG does what ZPC does: successive collision detection and collision simulation, but on GPU

1. Check for Correctness

Action item1: compare parton collision history, from PCG and ZPC respectively. Ideally we expect

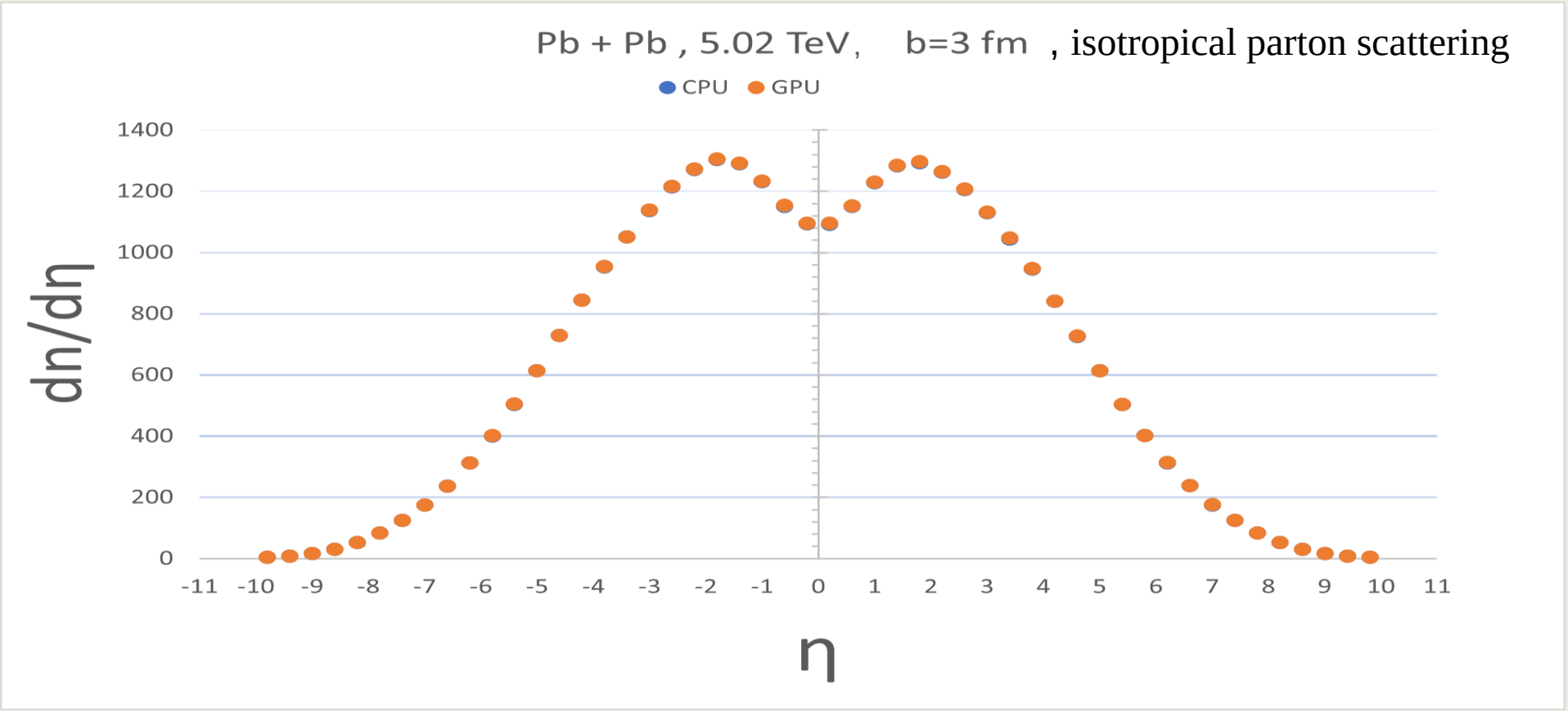
- **before the pair to collide, (r_μ, p_μ) for the two cases are the same**
- **after the pair collide, (r_μ, p_μ) are the same for the two cases**

Findings: For randomly chosen small amount of events, we have seen exactly the same parton collision history for collisions of Pb + Pb at $\sqrt{s_{NN}} = 5.02$ TeV for all centralities, using CPU for two-parton collision simulation and with the two-parton scattering assumed isotropical.

with CPU for simulating 2-parton collision, PCG runs ok

Action item2: Compare pseudo-rapidity distributions of charged hadrons

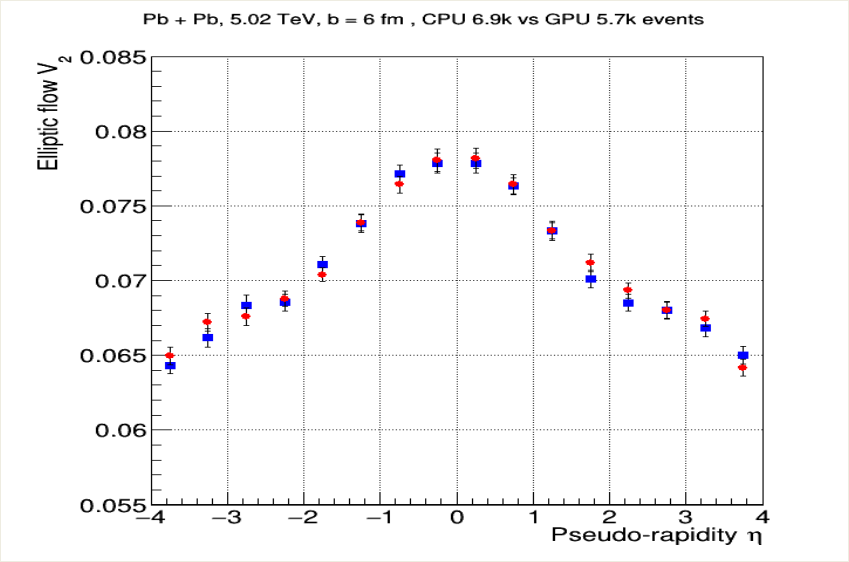
Findings: They are consistent nicely for Pb + Pb at $\sqrt{s_{NN}} = 5.02$ TeV



CPU: **AMD EPYC 7302, 16 core 3.0 GHz, GPU: V100**

Therefore, PCG accelerated with GPU runs ok

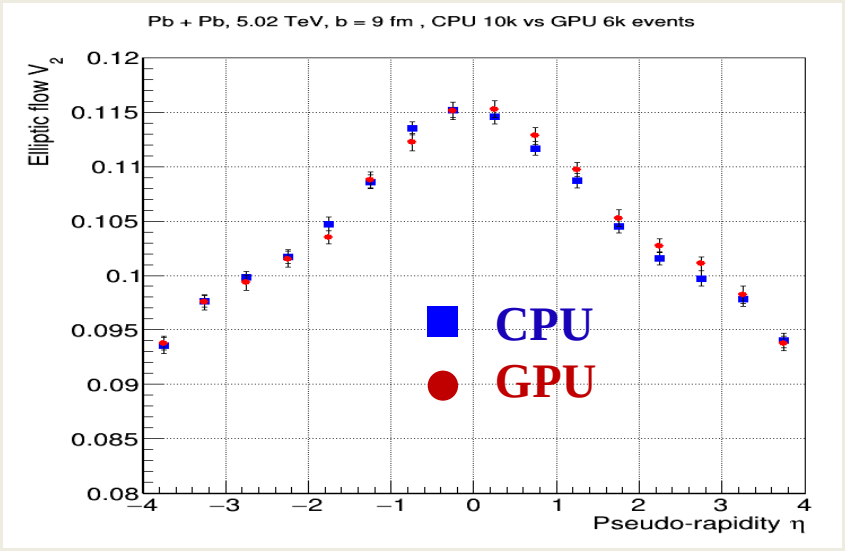
Action item3: compare v_2 from the two samples (isotropical parton scattering)

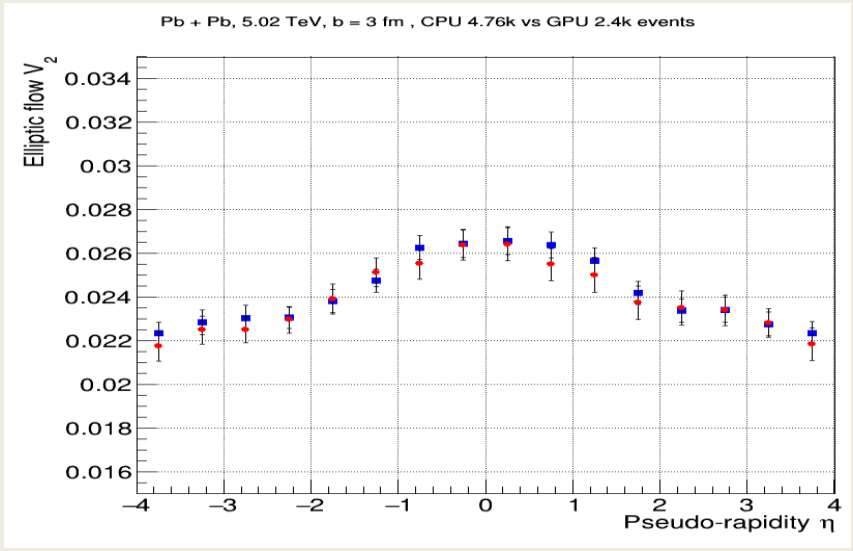


Elliptic flow $v_2 = \langle\cos(2(\varphi-\Psi_{RP})\rangle$

- **Momentum space azimuthal anisotropy the second order**
- **Extracting EOS and properties of QGP**
- **From almond shape coordinate space azimuthal anisotropy**

Indications: Two samples output consistent v_2





2. Check for speed-up for simulating parton cascade

Speed-up (加速比）= <timeZPC>/<timePCG> ≈ 13

Collision centrality: central toward mid-central collisions

Collision type: Pb + Pb at $\sqrt{s_{NN}}$ = 5.02 TeV, b = 3 fm

CPU: AMD EPYC 7302, 16 GB, 16 core 3.0 GHz

GPU: V100, 16GB HBM2

OS : Ubuntu 22.04.3 LTS

Note: applied isotropical parton scattering, set in input file

PCG can be a good choice, alternative to ZPC, speedup can be better

• Conclusions

Elastic parton cascade via 2-parton collisions have been simulated on GPU based on ZPC (Zhang's Parton Cascade) and thus PCG (Parton Cascade on GPU) had been developed.

Elliptic flow , together with pseudo-rapidity distribution of charged hadrons before hadronic cascade, agree in the two scenarios, one with the parton cascade simulated on CPU and the other on GPU, for Pb +Pb collisions at $\sqrt{s_{NN}} = 5.02$ TeV .

PCG, an alternative to ZPC , can speed up elastic parton cascade, more than 13 times faster for Pb +Pb collisions with b=3 fm at $\sqrt{s_{NN}} = 5.02$ TeV, V100 versus AMD EPYC 7302.

---------- 谢谢！ ----------

Acknowledgement

This work was partly supported by Beijing Natural Science Foundation grant no. 1132017, Beijing Municipal Education Commission's major scientific project grant no. KZ201910017019, the GHfund B(20210702) and the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry.