HIJING++, a Heavy-ion Monte Carlo Generator for the Future (Parallel) Generations

Speaker: Gergely Gábor Barnaföldi, Wigner RCP of the H.A.S.

Group: GGB, G. Bíró, Sz.M. Harangozó, W.T. Deng, M. Gyulassy, G.Y. Ma,
P. Lévai, G. Papp, X.N. Wang, B.W. Zhang

GPU Day 2017, Wigner Datacenter Budapest, Hungary, 22-23 June 2017

Outline

- Motivation for HIJING++
- Technical details of the HIJING++
  - The structure of the program
  - Simulation framework
- New physics & tests
  - Code validation in proton-proton collisions
  - New improvement: Scale-dependent HIJING shadowing
- Outlook...

MOTIVATION

Material properties, phases

- Let's see a simple material…

Material properties, phases

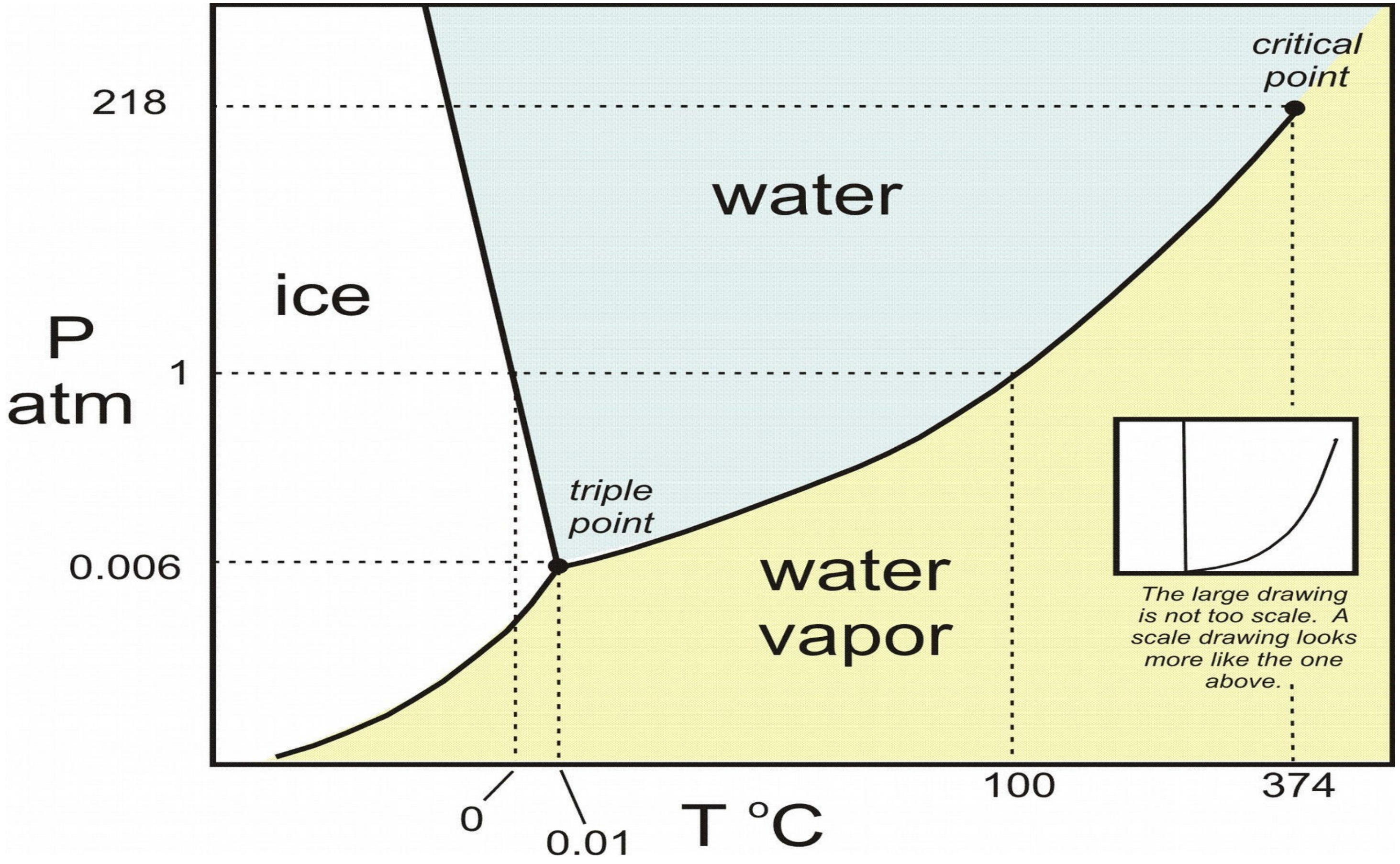
- Let’s see a simple material…

Material properties, phases

- Let’s see a simple material…

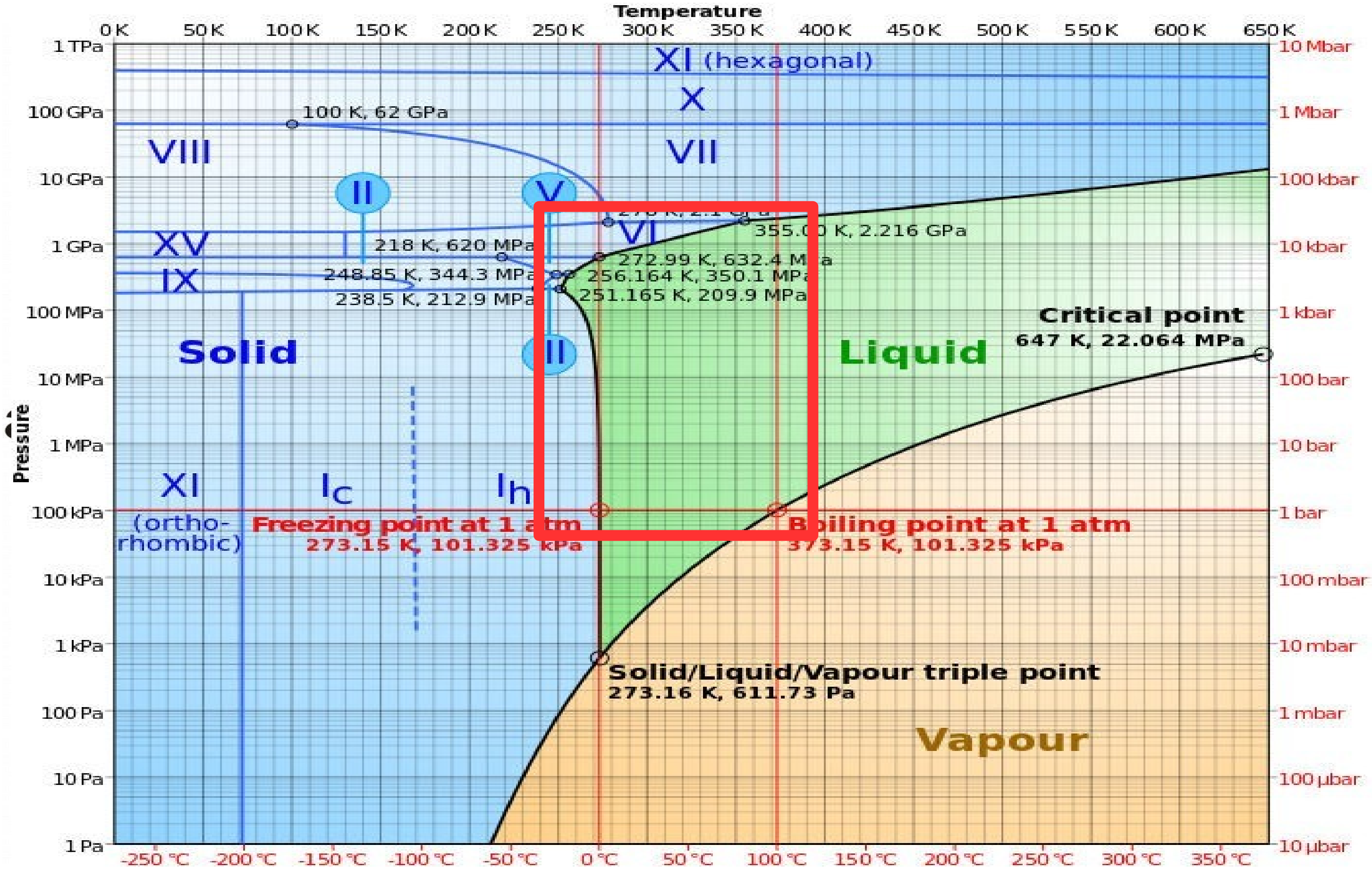
Material properties, phases

- Let's see a simple material…



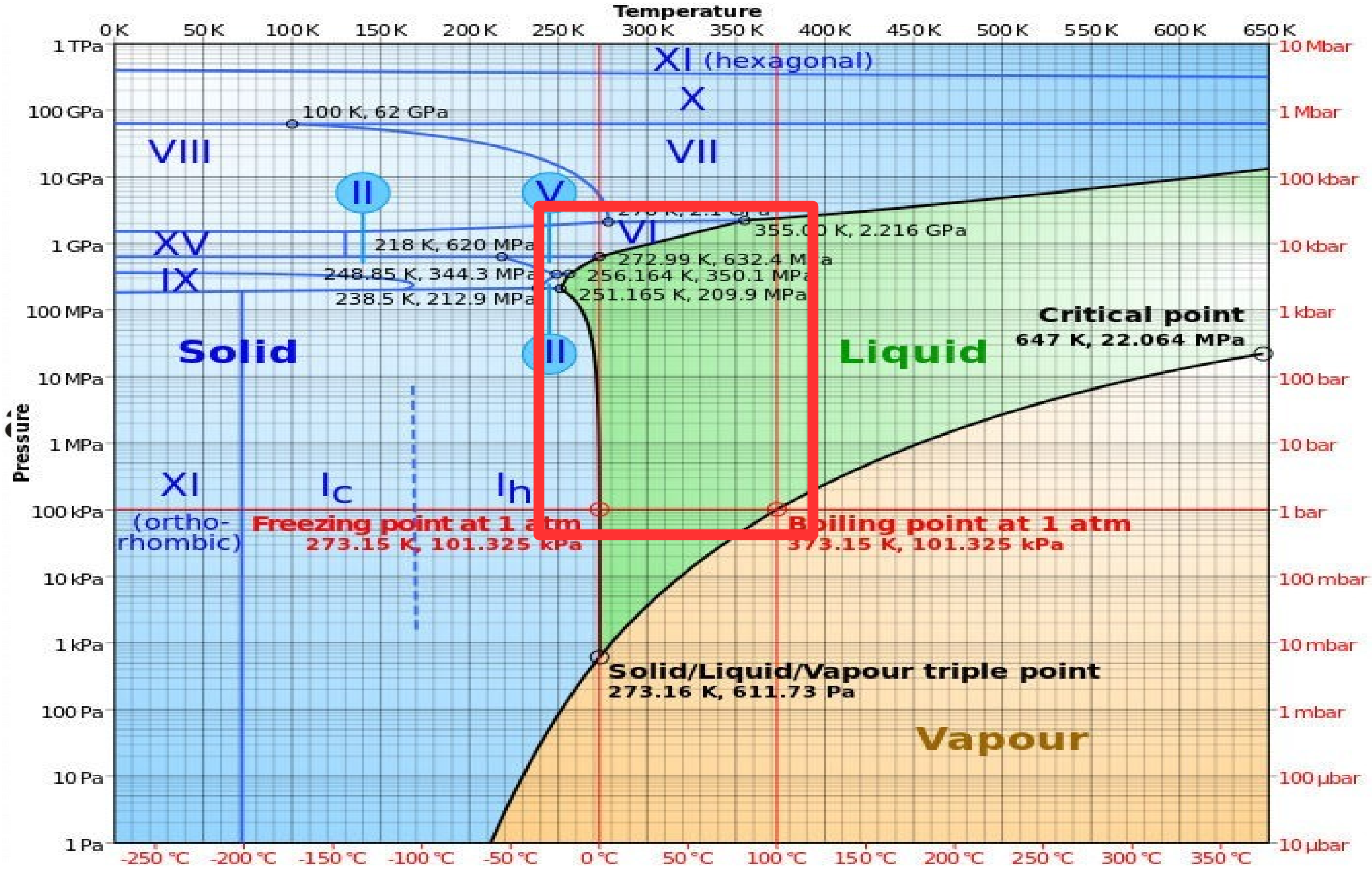
Material properties, phases

- Let’s see a simple material…

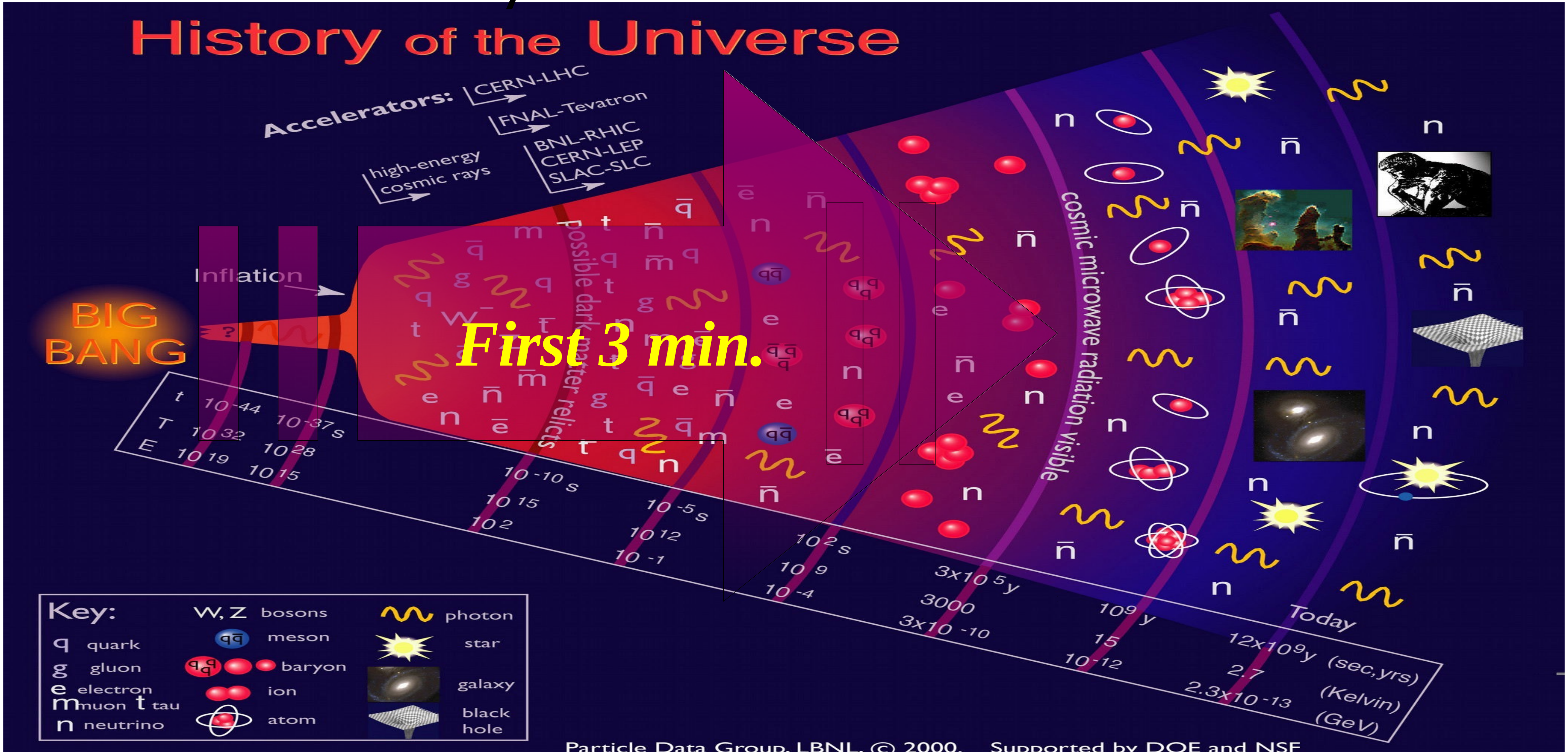


Material properties, phases

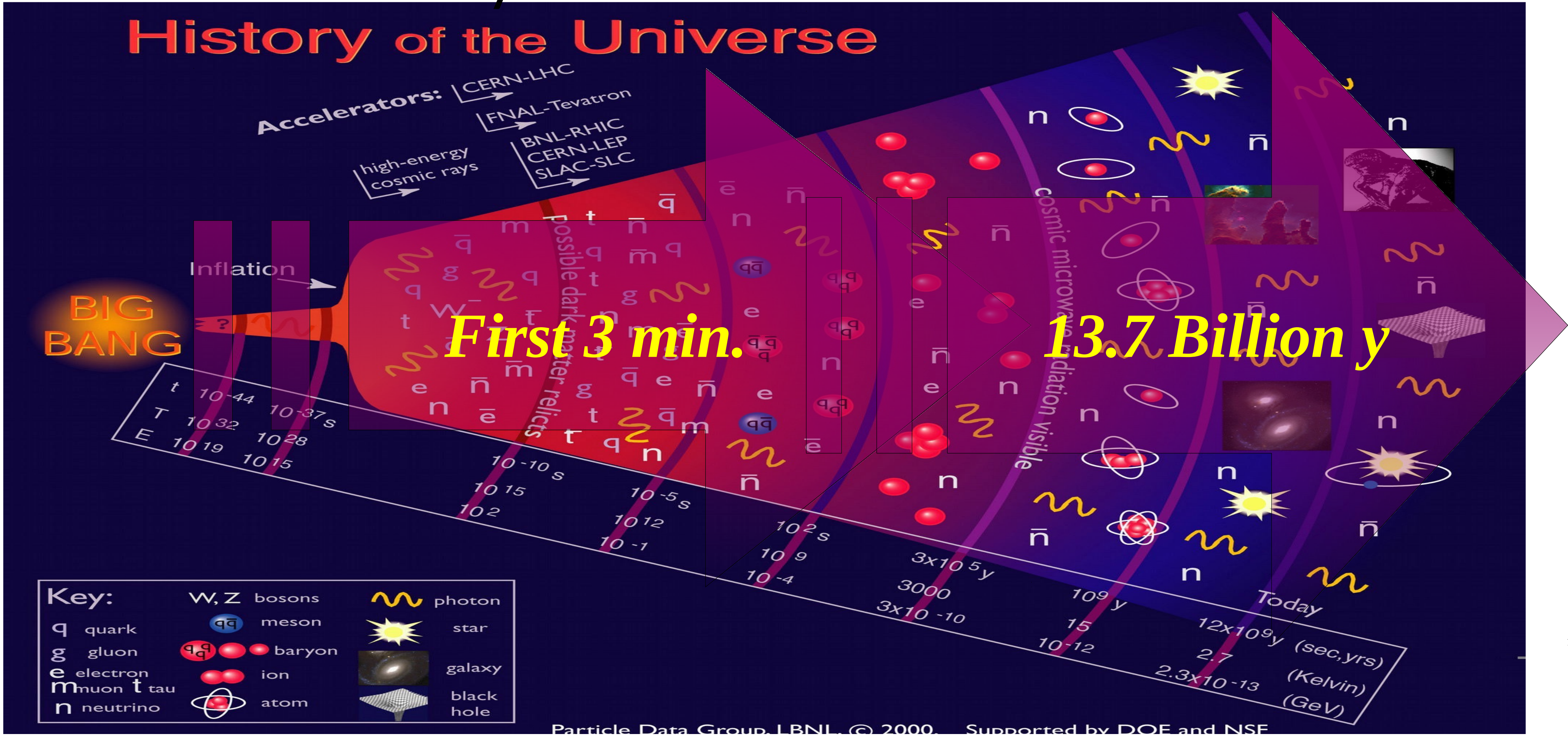
- Let’s see a “simple” material at extreme conditions…



The matter of the early Universe

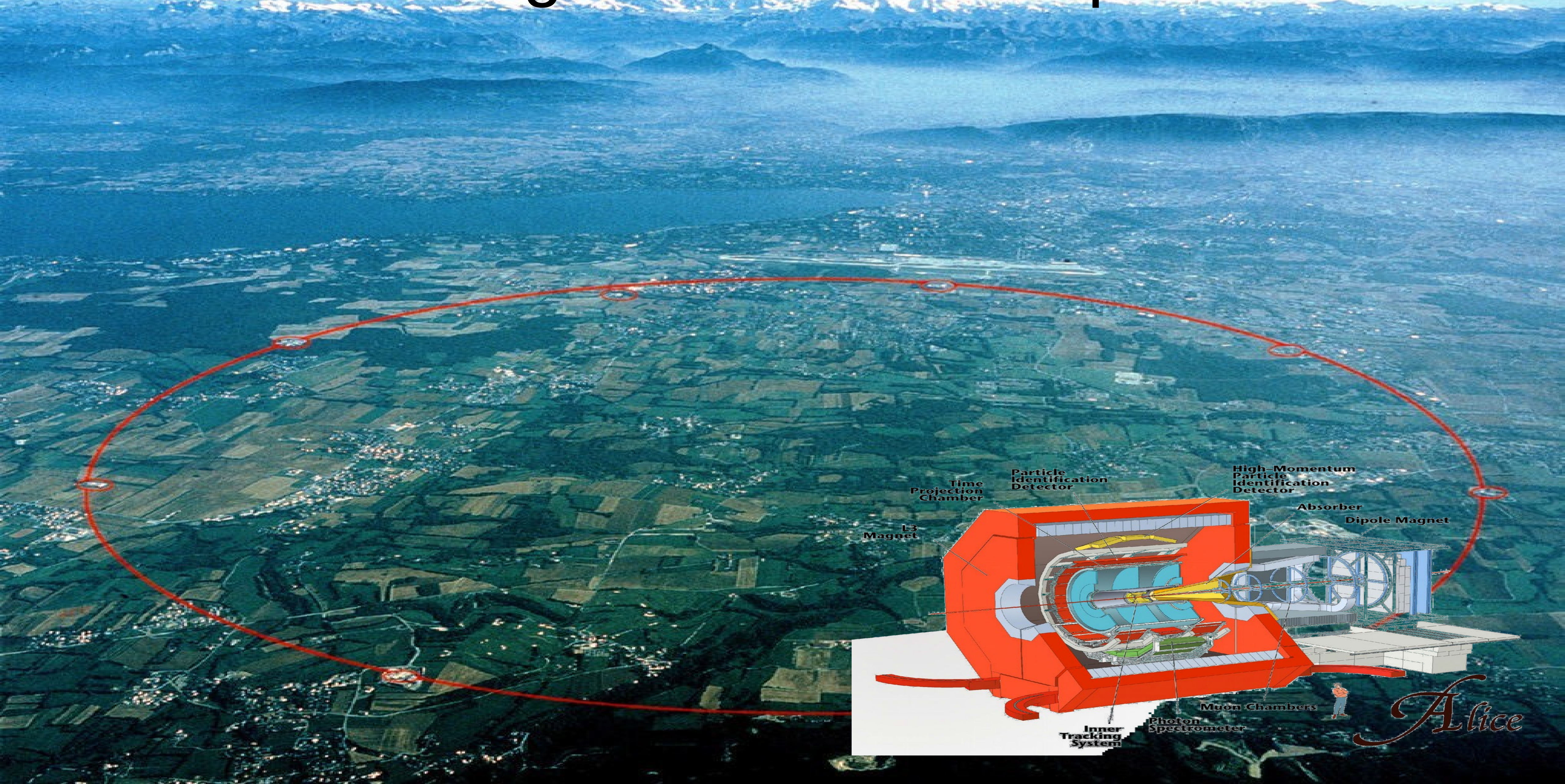


The matter of the early Universe

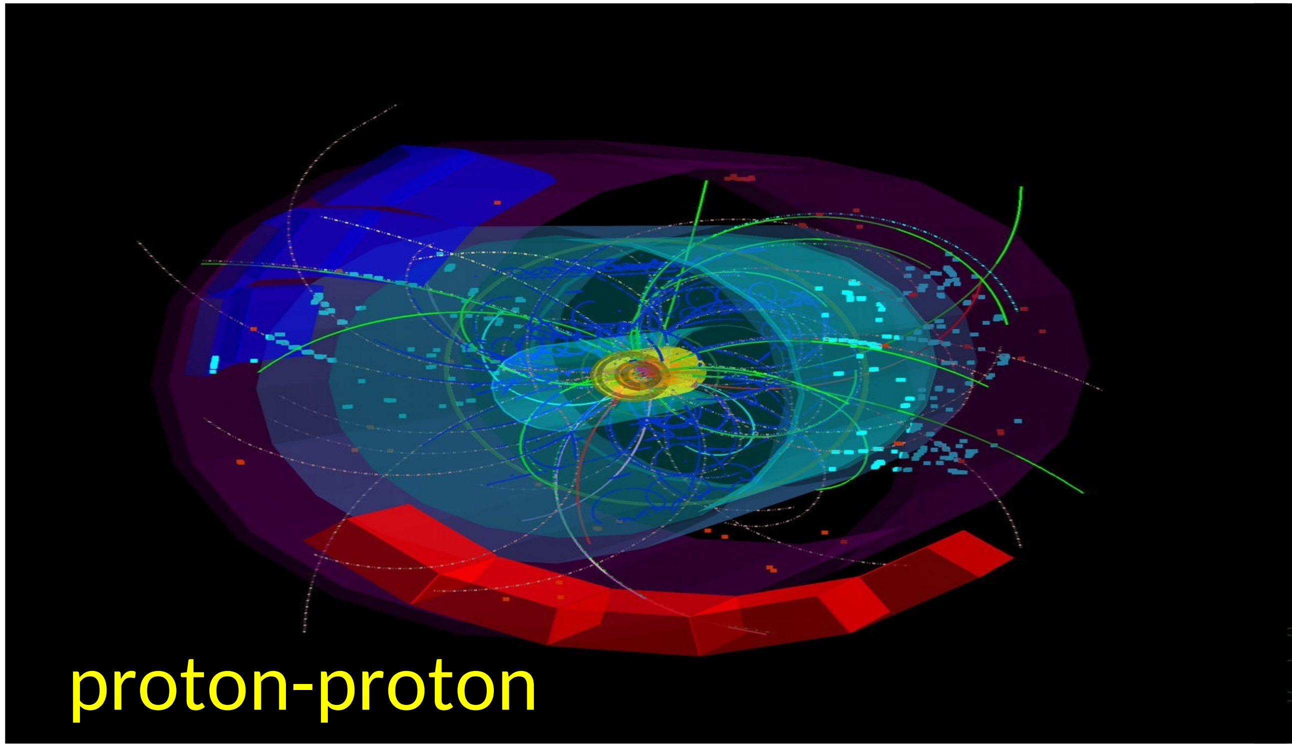


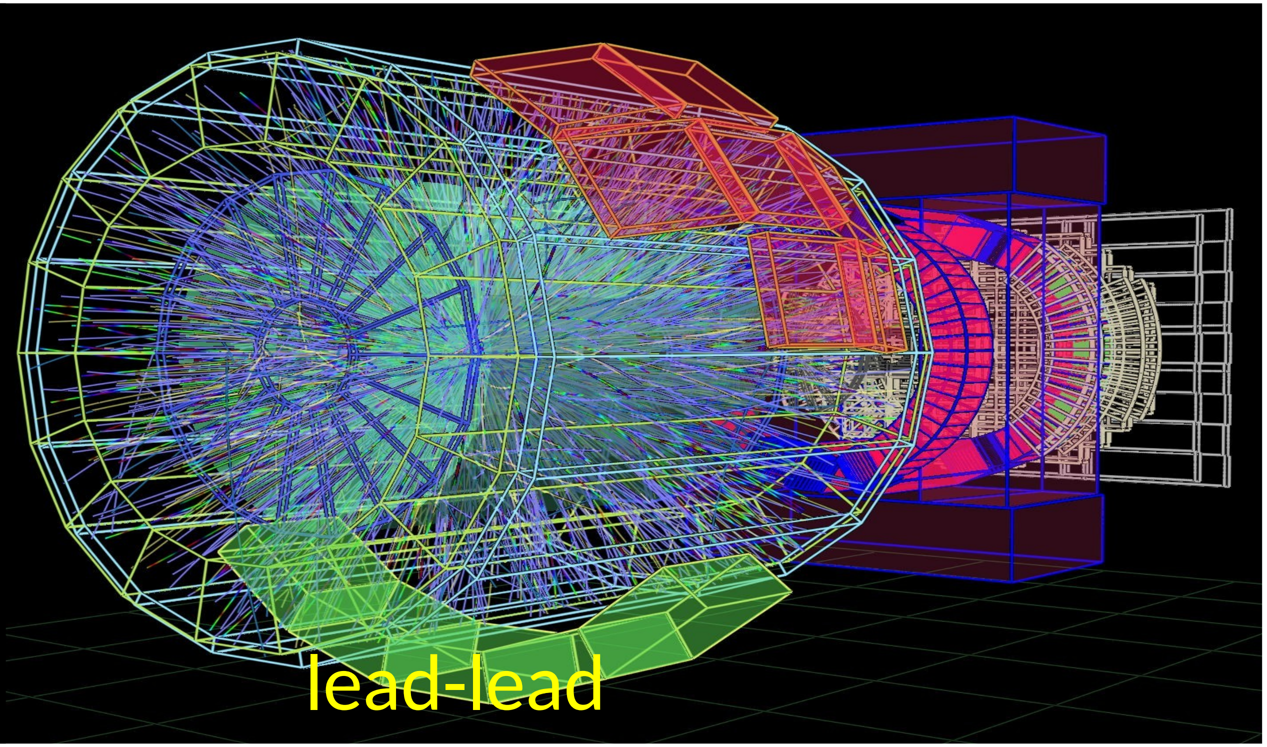
Looking for the origin of the matter with the CERN LHC ALICE Experiment

ALICE – A Large Ion Collider Experiment



The matter of the early Universe in the LHC

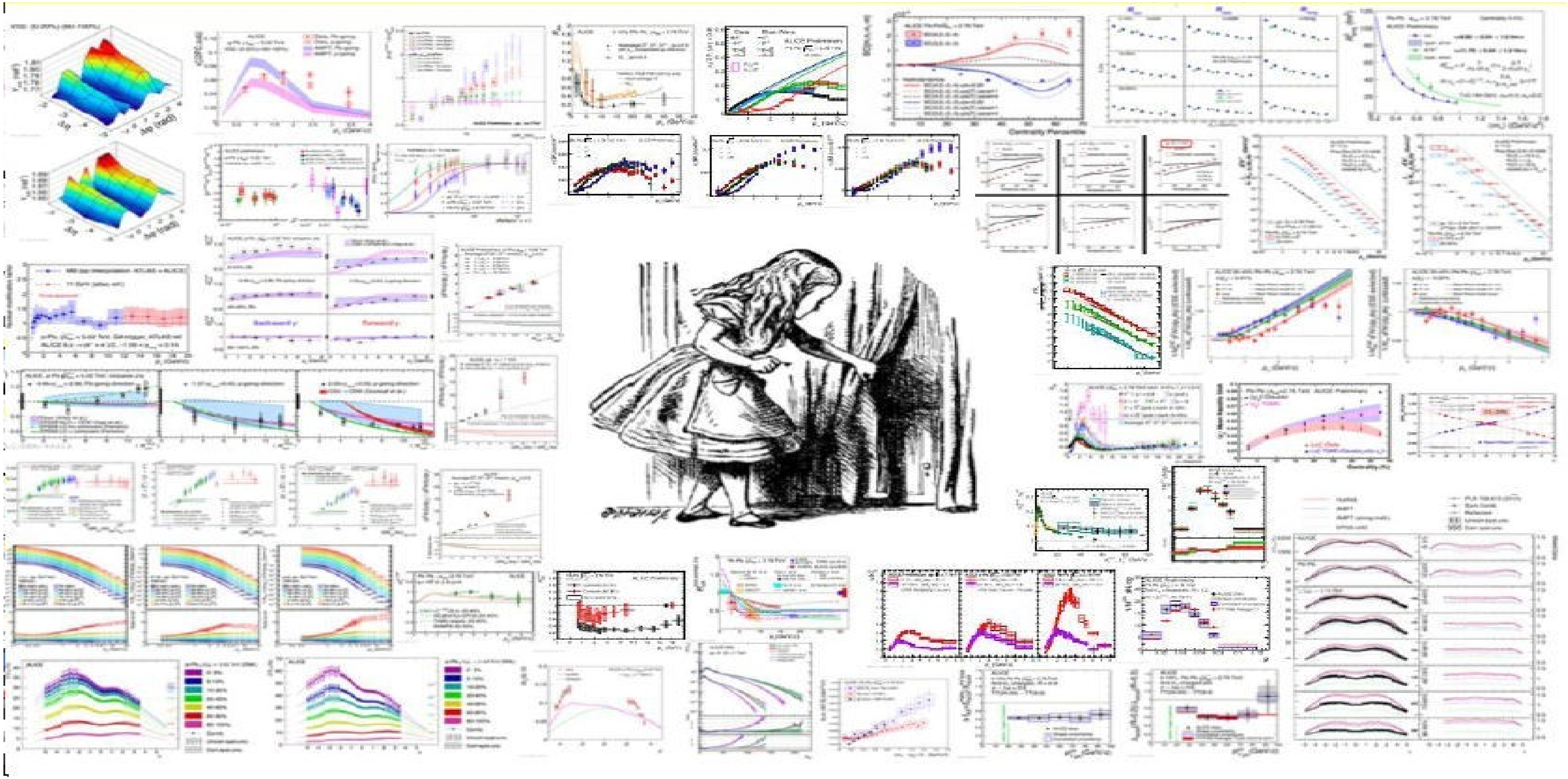




Quark Gluon Plasma (QGP):

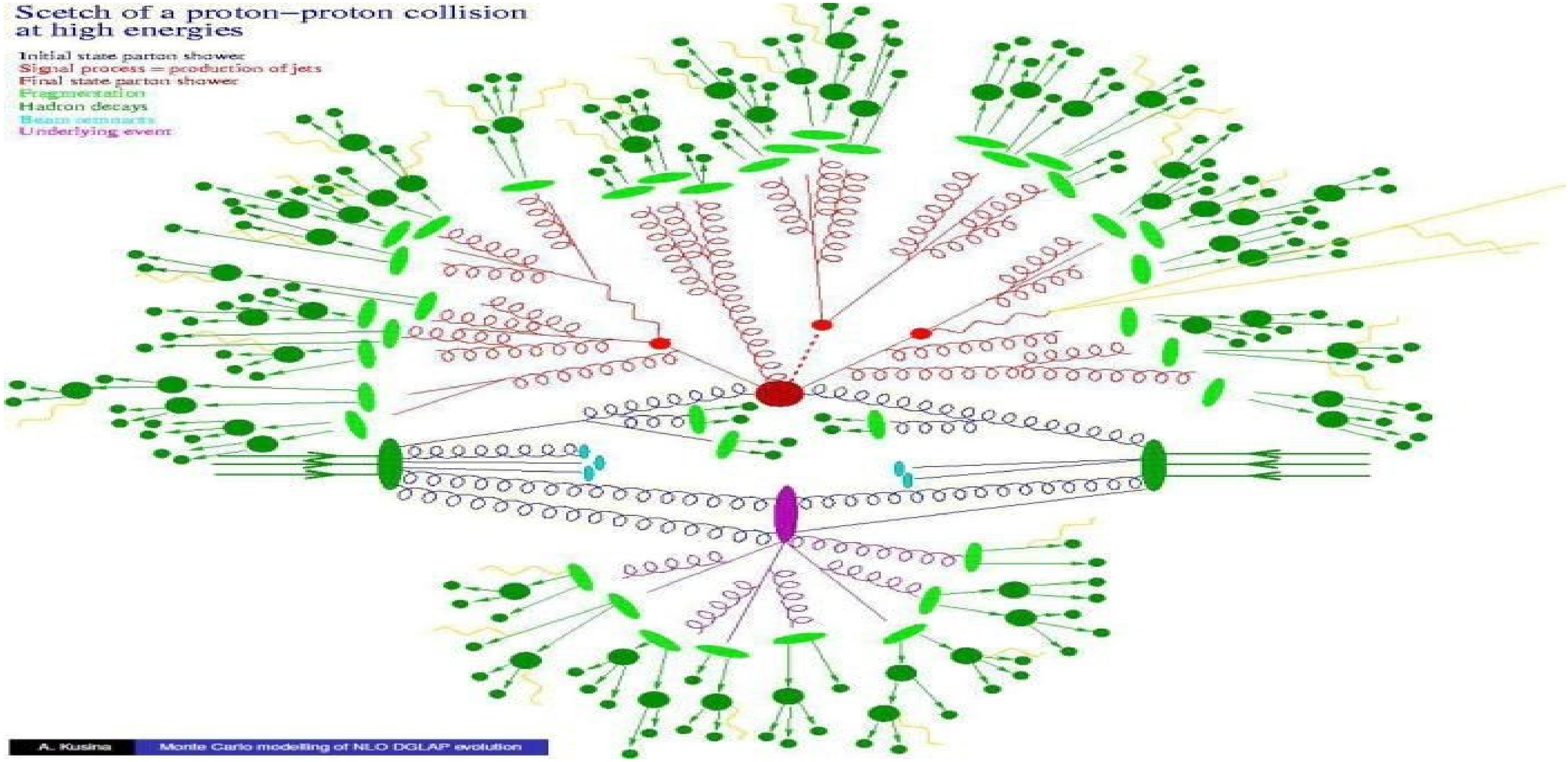
- proton-proton vs. lead-lead
- hot, color (quark+gluon)
- a kind of „ideal fluid“…

Results from the ALICE Collaboration..



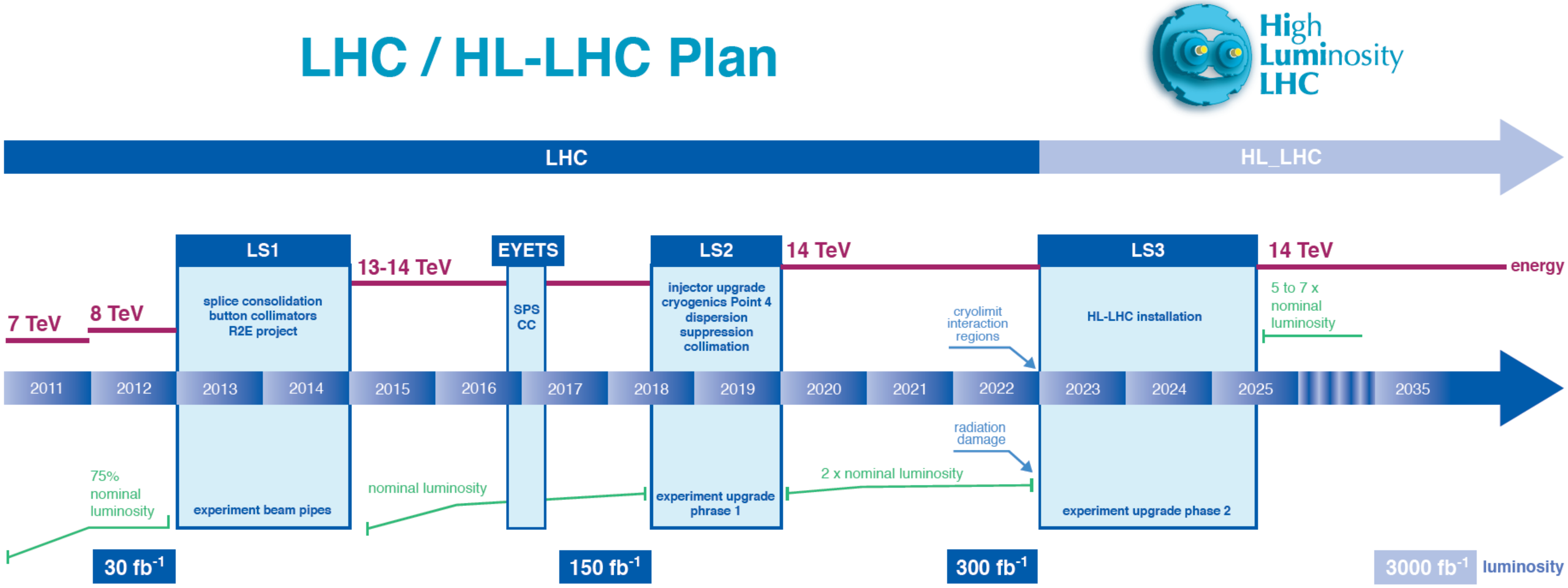
Theoretical description of the simplest collisions: proton-proton

- More accurate measurements → more detailed calculations



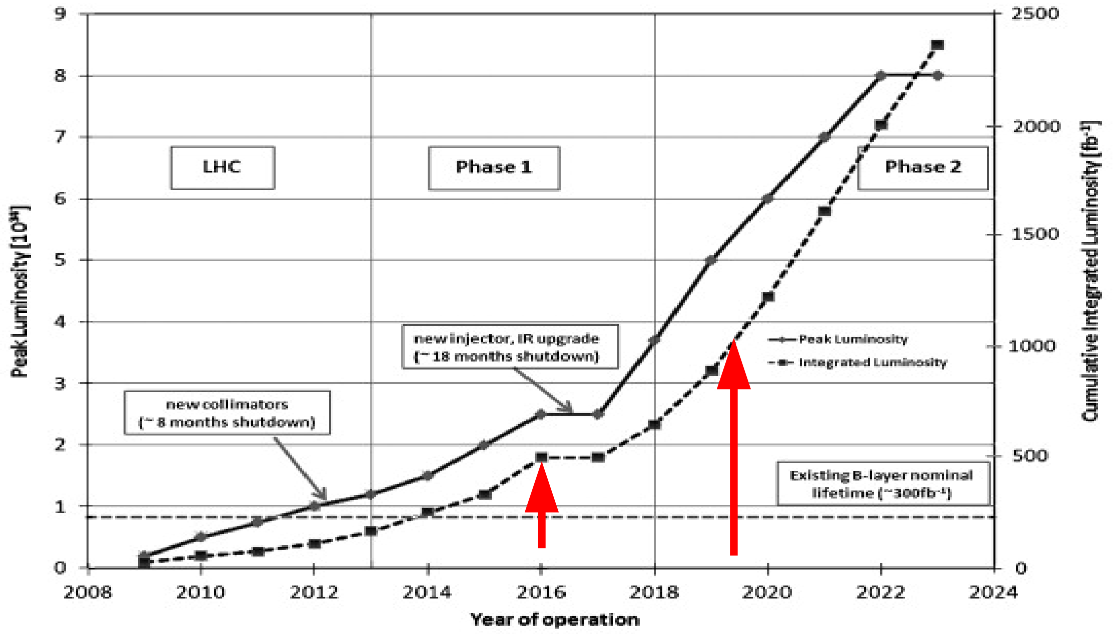
HI data from the Large Hadron Collider

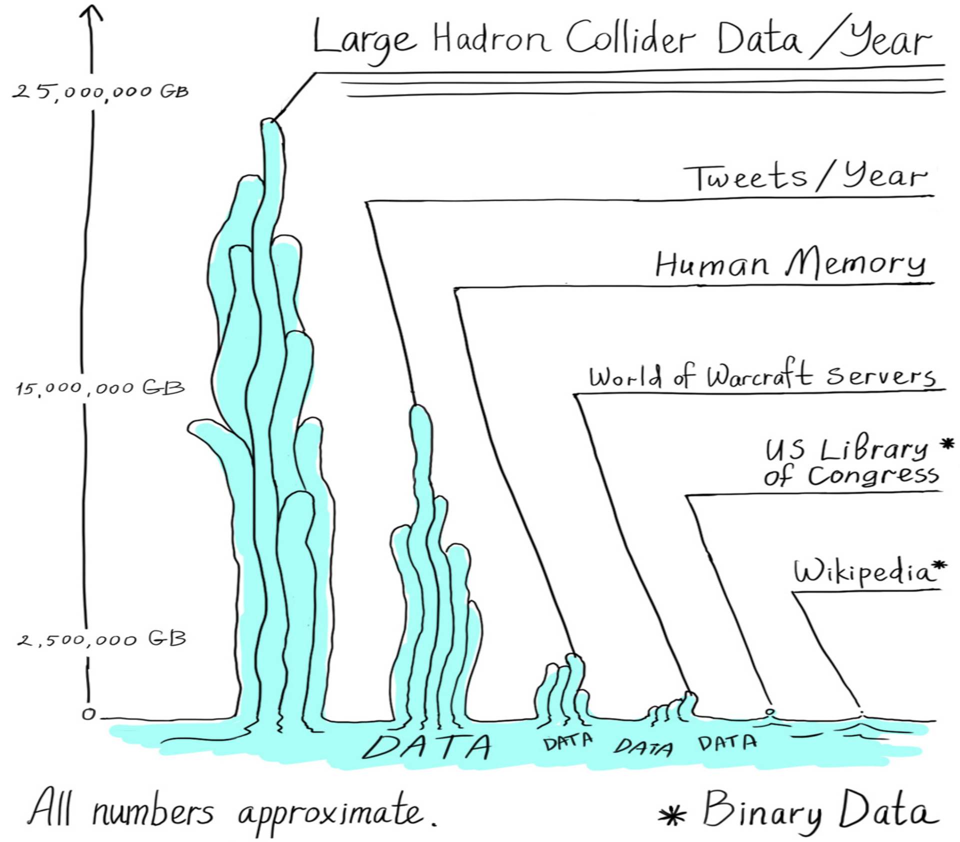
- LHC upgrades & theories required more and faster HI simulations



HI data from the Large Hadron Collider

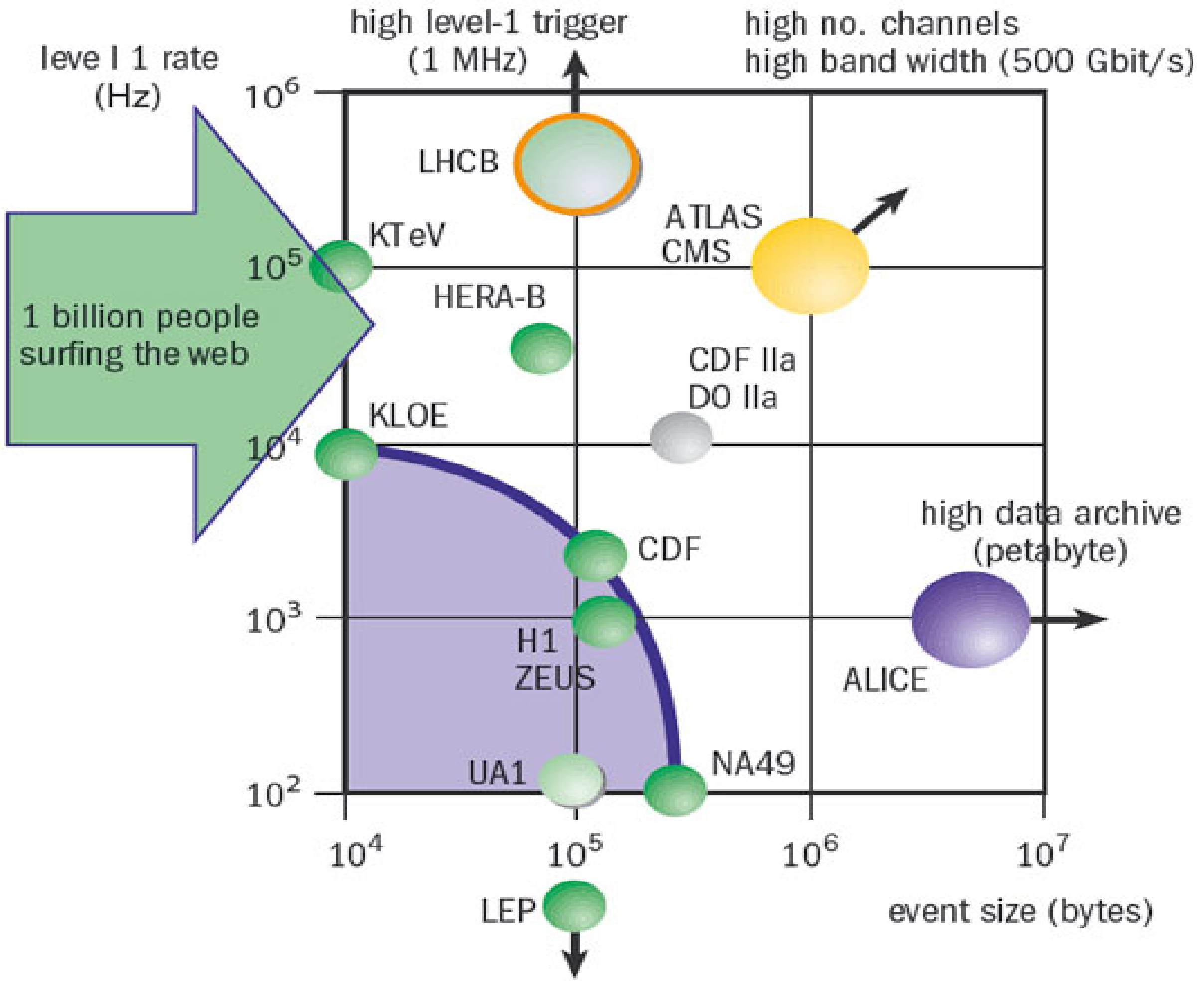
- WLCG – Worldwide LHC Computing GRID:
  - LHC made 15-20 PB data per year
  - ...and now before HL-LHC 2PB/day





More data: motivation for fast computing at CERN

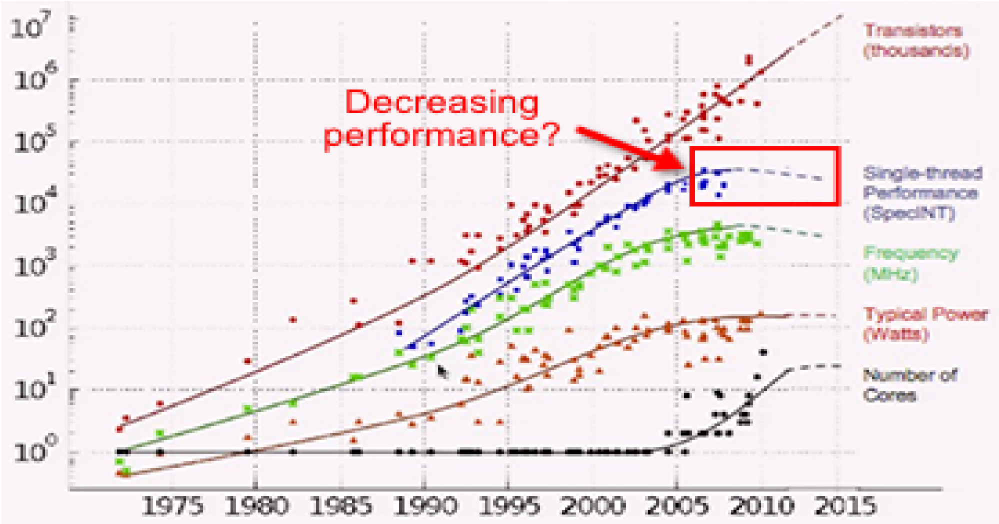
- Ideal: amount of simulated data $\approx$ real data
  - Number of events at LHC: $\mathcal{O}(10^8)$ / s
  - Necessary time for Monte Carlo with ALICE geometry: $3.8\ ms/track$
- Necessary time to simulate 1 s of ALICE data: $\mathcal{O}(days)$



Fast computing = parallel computing

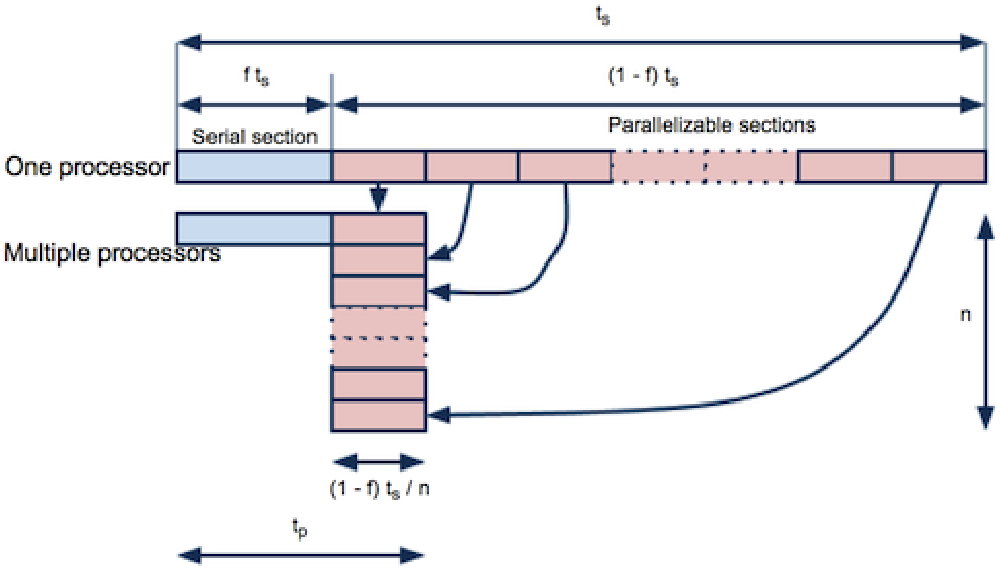
- Moore's law:

Every 2nd year the number of transistors (integrated circuits) are doubled in computing hardwares.



- Amdalh's law:

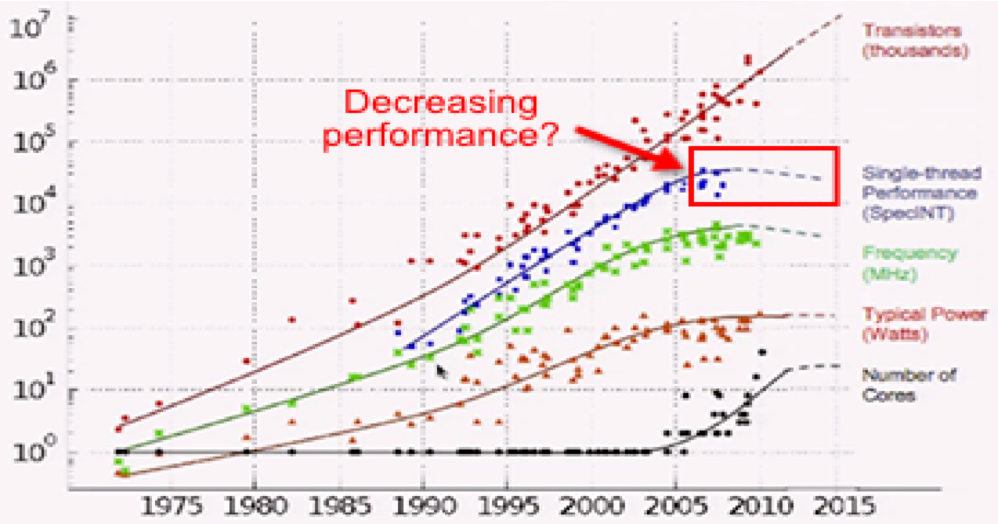
The theoretical speedup is given by the portion of parallelizable program, p, & number of processors, N, is:



Fast computing = parallel computing

- Moore's law:

Every 2nd year the number of transistors (integrated circuits) are doubled in computing hardwares.



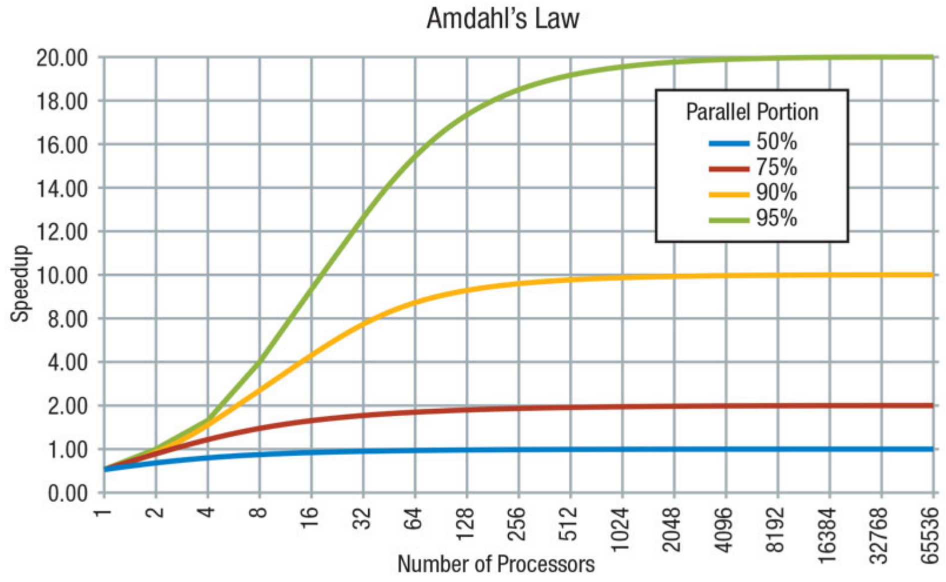
- Amdalh's law:

The theoretical speedup is given by the portion of parallelizable program, p, & number of processors, N, is:

$$\text{Speedup}(N) = \frac{1}{(1-P)+\frac{P}{N}}$$

Serial part of job = 1 (100%) - Parallel part

Parallel part is divided up by N workers



HIJING++

(C++ based HIJING version 3.1 with parallel opportunities)

The HIJING++

HIJING(Heavy-Ion Jet INteraction Generator)

易經

Bagua (eight simbols)

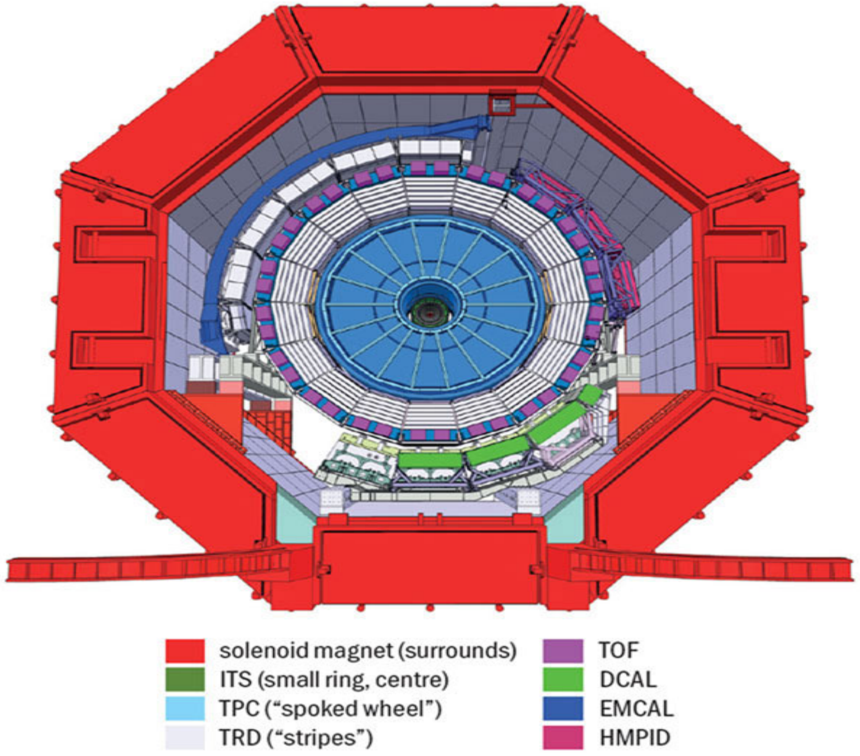
fundamental principles of reality

adjoint representation 8 of *SU(3)*

The HIJING++

HIJING(Heavy-Ion Jet INteraction Generator)

易經



Bagua (eight simbols)

fundamental principles of reality

adjoint representation 8 of *SU(3)*

What is the ‘real’ HIJING???

It is a BIG mess….

We need BIG Wizards to manage it...

and the need for a magic stick...

HIJING++

(C++ based HIJING version 3.1 with parallel opportunities)

The HIJING++

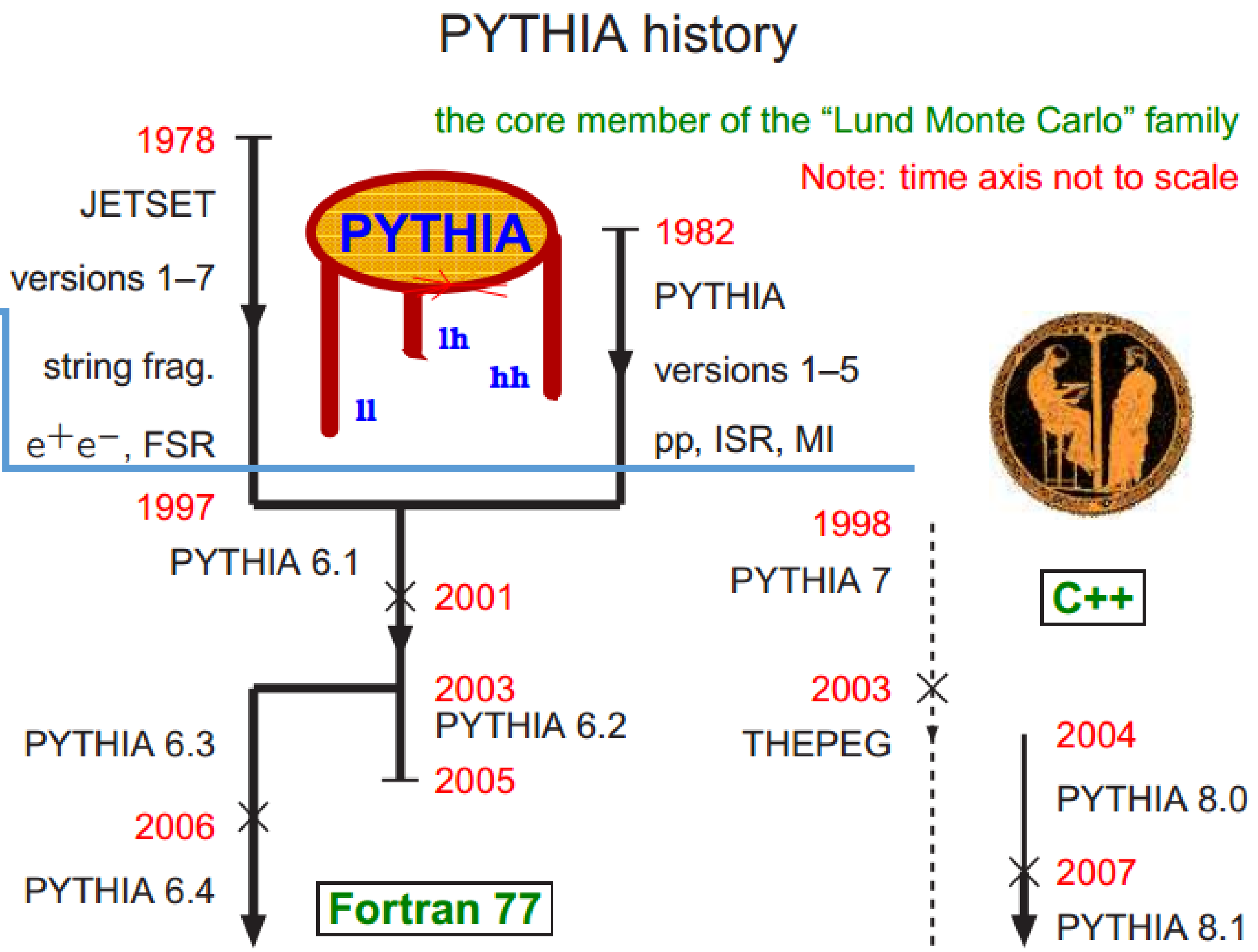
HIJING(Heavy-Ion Jet INteraction Generator)

- HIJING versions
  - FORTRAN v1.36, v2.553
  - C++ v3.0

Reasons to use C++

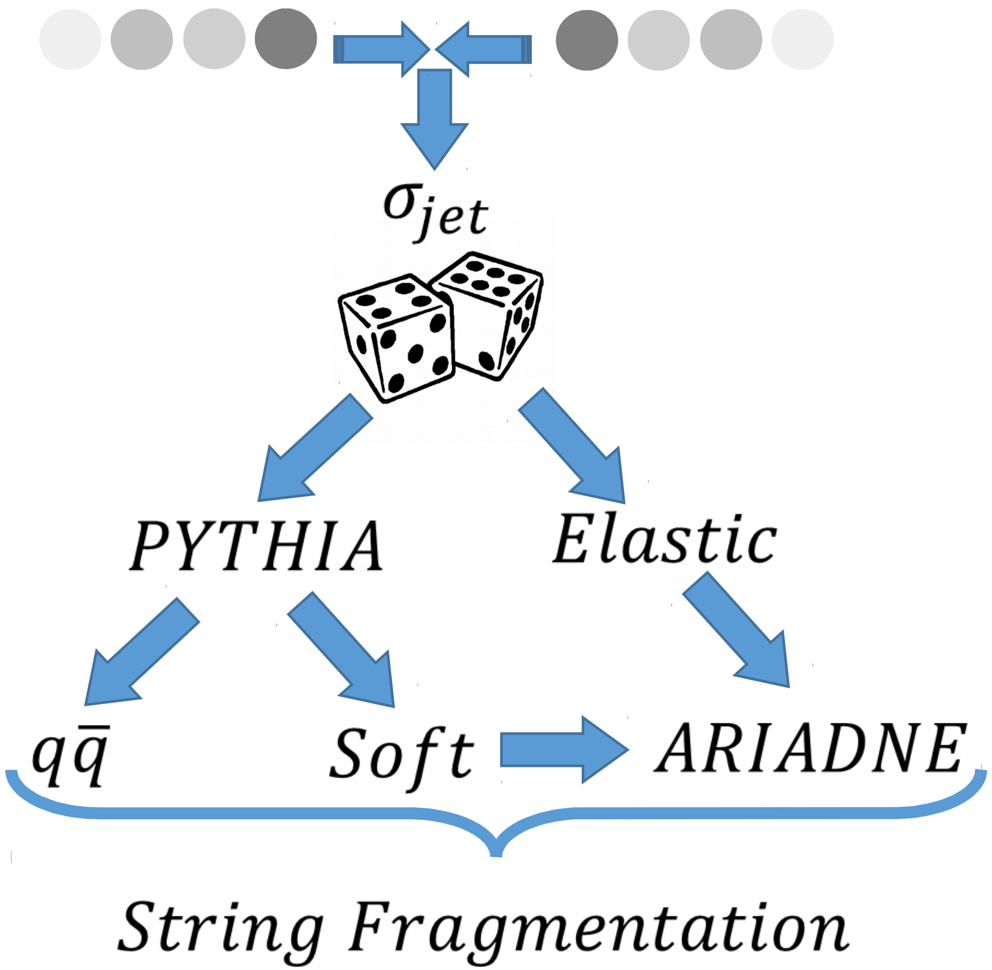
Object oriented language:
Hierarchy, Modularity

C++11/14 has thread support
and compatibility with OpenCL



Program Flow – in general

- Pair-by-pair nucleon-nucleon events
- Multiple soft gluon exchanges between valence- and di-quarks
- String hadronization according to Lund fragmentation scheme

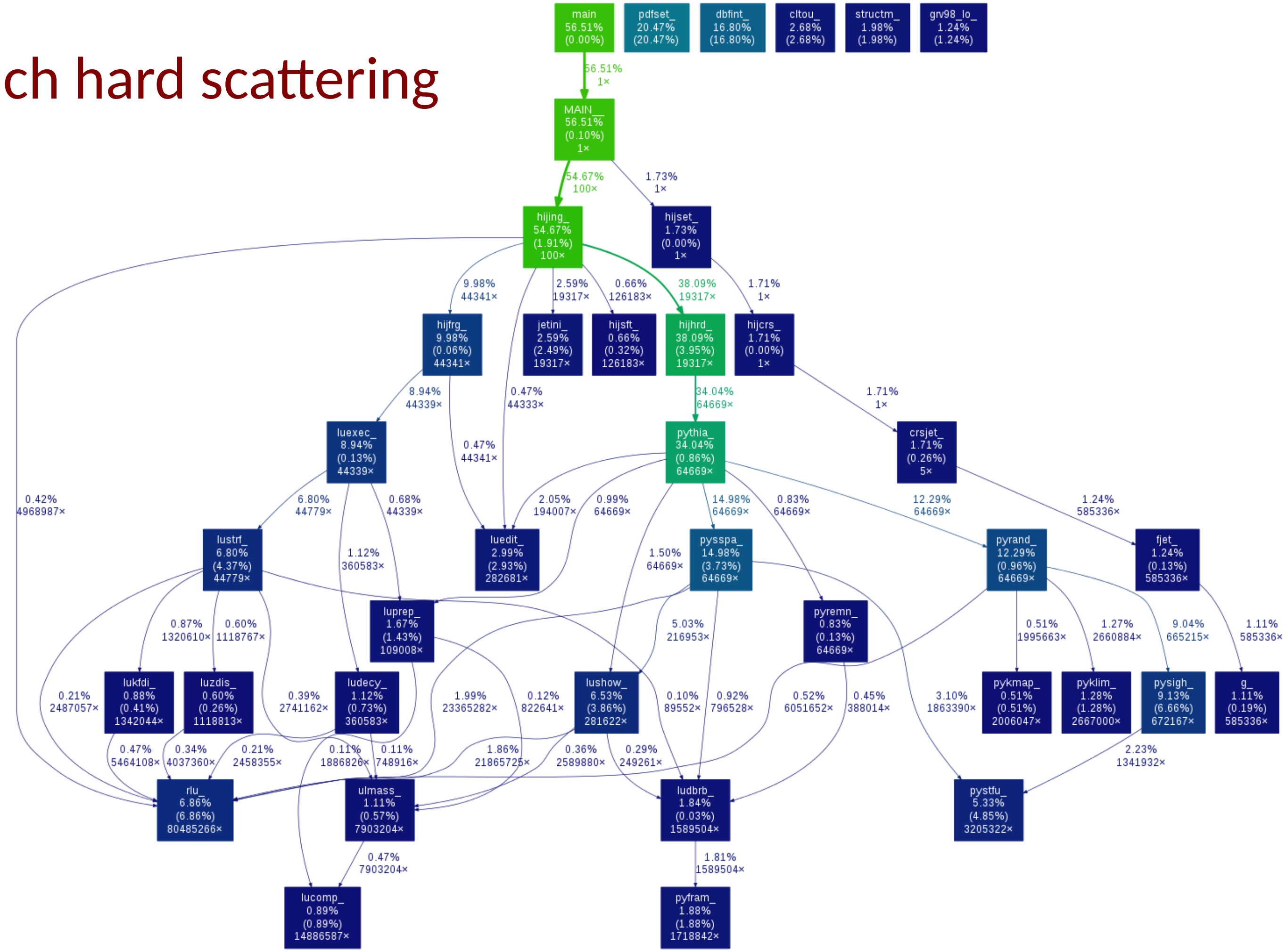


Program Flow – old one

- Generation of kinetic variables for each hard scattering with Pythia 5.3

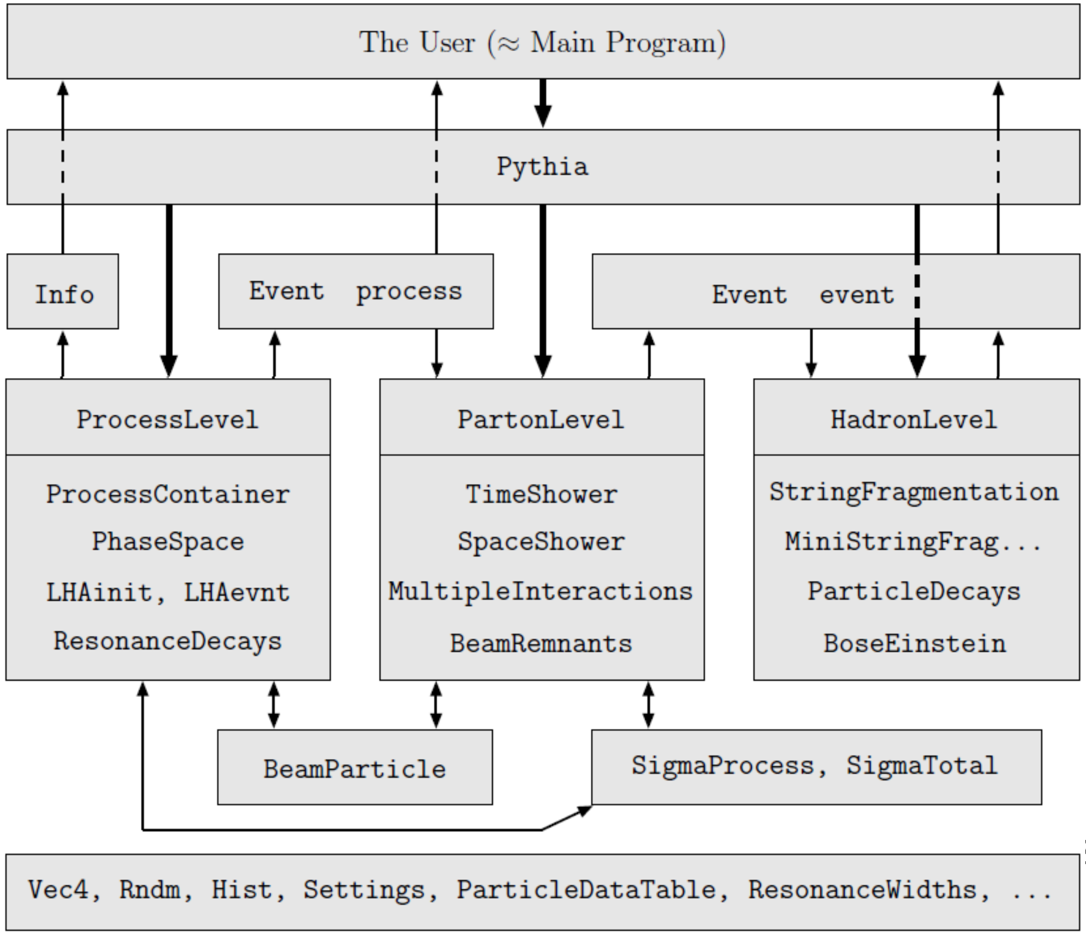
- Multiple soft gluon exchanges between valence- and di-quarks

- String hadronization according to Lund fragmentation scheme



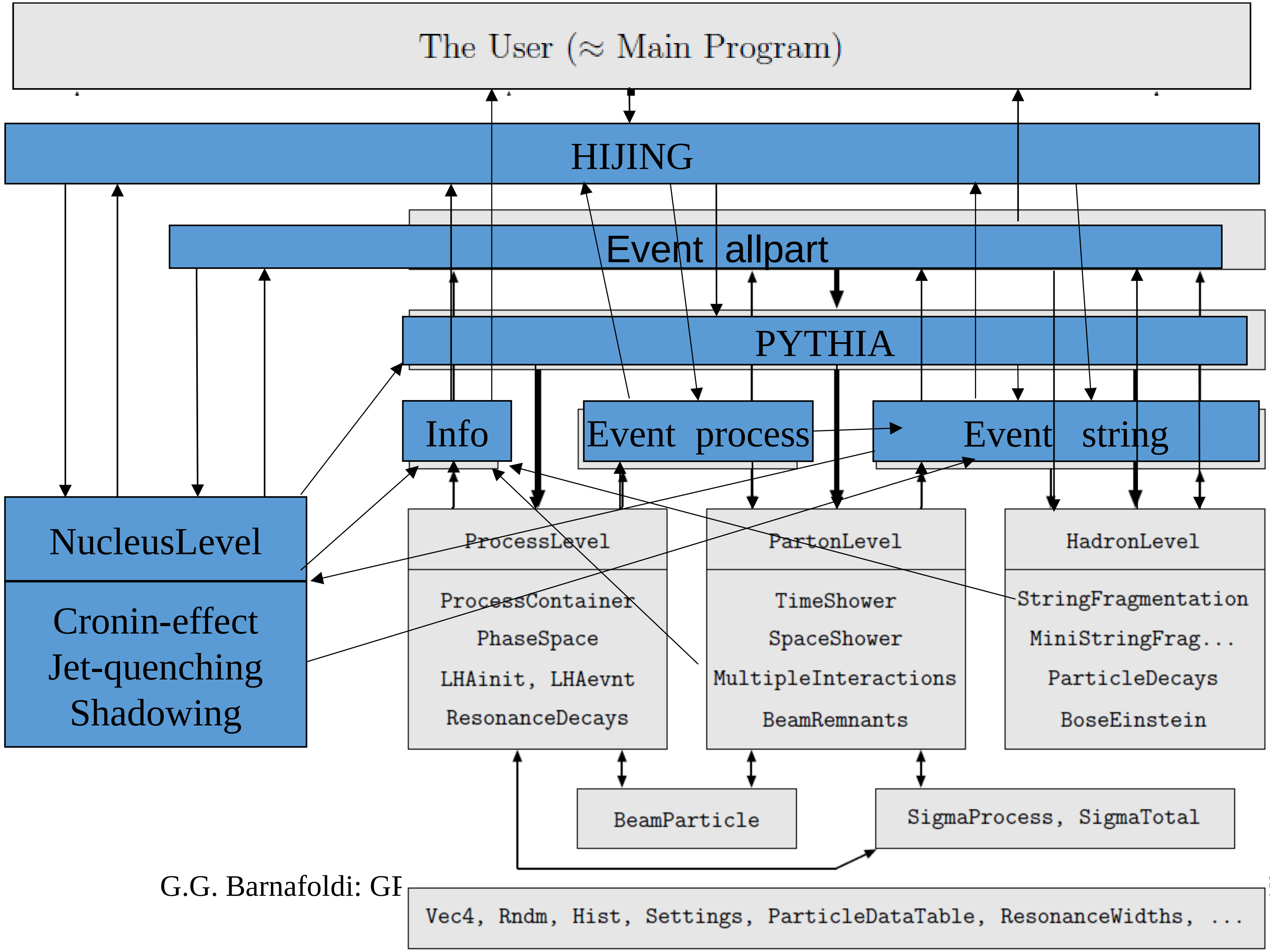
Program Structure

- Pythia8 namespace containers
- Structure similarities
- Actual program flow is more complicated



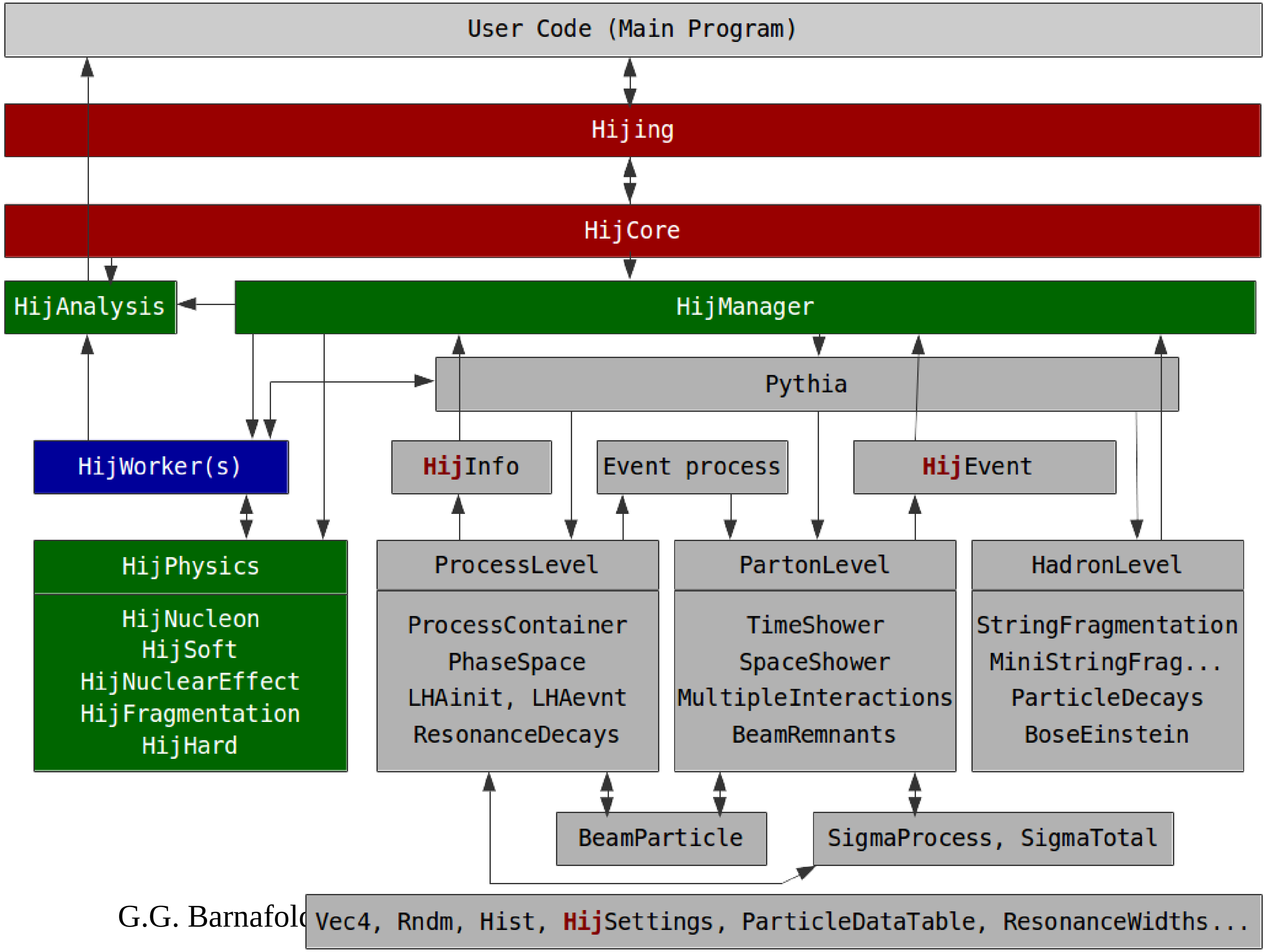
Program Structure

- Pythia8 namespace containers
- Structure similarities
- Actual program flow is more complicated



Program Structure

- Pythia8 namespace containers
- Structure similarities
- Actual program flow is more complicated
- New: HijManager



Program Structure

Hijing class

```
namespace Pythia8 {

    class Hijing {

    public:

        Info            info;
        Rndm            rndm;
        Settings        settings;
        ...

    private:

        HardCollision   hijhard;        // Class for handling the hard collisions

        SoftScatter     hijsoft;        // Class for handling the soft interactions

        Fragmentation   fragmentation;  // Class for handling the Lund string fragmentation

        NucleonLevel    nucleonlevel;   // Class for  the nuclear effects

        ...
    }
}
```

- Processes ordered in class hierarchy
- Former common blocks ▭ class variables
- Processes called through object functions

The 'main' example

Usual form kept for regular users

FORTRAN

```
PROGRAM TEST
...
PARM(1) = 'DEFAULT'
VALUE(1) = 80060
CALL PDFSET(PARM, VALUE)
CALL GetDesc()
...

CALL HIJSET(EFRM, FRAME, PROJ, TARG, IAP, IZP, IAT, IZT)

N_EVENT=1E6
DO 200 IE = 1, N_EVENT
    CALL HIJING(FRAME, BMIN, BMAX)
200 CONTINUE

STOP
END
```

Form also similar to Pythia 8.x

C++

```
#include "Hijing.h"

using namespace Pythia8;

int main() {
    Hijing hijing("../xmldoc", true);
    hijing.readString("PDF:pSet = LHAPDF6:GRV98lo");

    bool okay = hijing.init(200.0, frame,
                  "A", "A", 197, 79, 197, 79);
    if (!okay) return 1;

    int MaxEvent = 1e6;
    for (int iEvent = 0; iEvent < MaxEvent; ++iEvent)
        hijing.next(frame, 0.0, 0.0);
}
```

Program Features

- Calculation by improved models
- Pythia like prompt Histogram creation
- CPU level Parallel computing

```
const std::size_t num_threads = std::thread::hardware_concurrency();
for (std::size_t i = 0u; i < num_threads; ++i){
    async_hijing.at(i) = std::unique_ptr<Hijing>(new Hijing);
}
for (std::size_t I = 0; I < num_threads; ++I){
    ...async run...
    okay[I] = async_hijing[I]->init(...);
    for (int iEvent = 0; iEvent < numEvent; ++iEvent)
        async_hijing[I]->next(...);
    for (int i = 0; i < async_hijing[I]->event.size(); ++i)
        if(...) hist[I]->fill(...);
}
```

THE C++ STANDARD TEMPLATE LIBRARY

- AliRoot compatibility (planned)

Dependencies & External packages

- Boost

sudo apt-get install libboost-all-dev

- LHAPDF 6

./configure –prefix=$HOME/…/share/LHAPDF

make all

insert downloaded PDF library to $HOME/…/share/LHAPDF

optionally modify **pdfsets.index**, add set if needed

export LD LIBRARY PATH=<library path>

- Pythia 8

*./configure --with-lhapdf6-lib=$HOME/…/lib *
--with-boost-lib=/usr/lib/x86_64-linux-gnu
make –j4

- GSL (optional)

HIJING **make** option

Data Analysis

```
#include "Hijing.h"
using namespace Pythia8;

int main() {
    Hist dndpT("dn/dpT for charged particles", 100, 0., 10.);
    ofstream ch_file("ch_hist.dat");
    ...
    bool okay = hijing.init(efrm, frame, proj, targ,
                            aproj, zproj, atarg, ztarg);
    if (!okay) return 1;
    int MaxEvent = 1e6;
    for (int iEvent = 0; iEvent < MaxEvent; ++iEvent) {
        hijing.next(frame, bmin, bmax);
    for (int i = 0; i < hijing.event.size(); ++i)
    if (hijing.event[i].isFinal() && hijing.event[i].isCharged())
        dndpT.fill(hijing.event[i].pT());
    }
    dndpT *= 1.0 / MaxEvent;
    cout << dndpT;
    dndpT.table(ch_file);
    ...
    return 0;
}
```

Pythia 8 Histogram class available

Selection has to be made for every particle

Hist::fill(double Input);

Normalization

standard output and file output both provided

FIRST TESTS, RESULTS, & PREDICTIONS with HIJING++

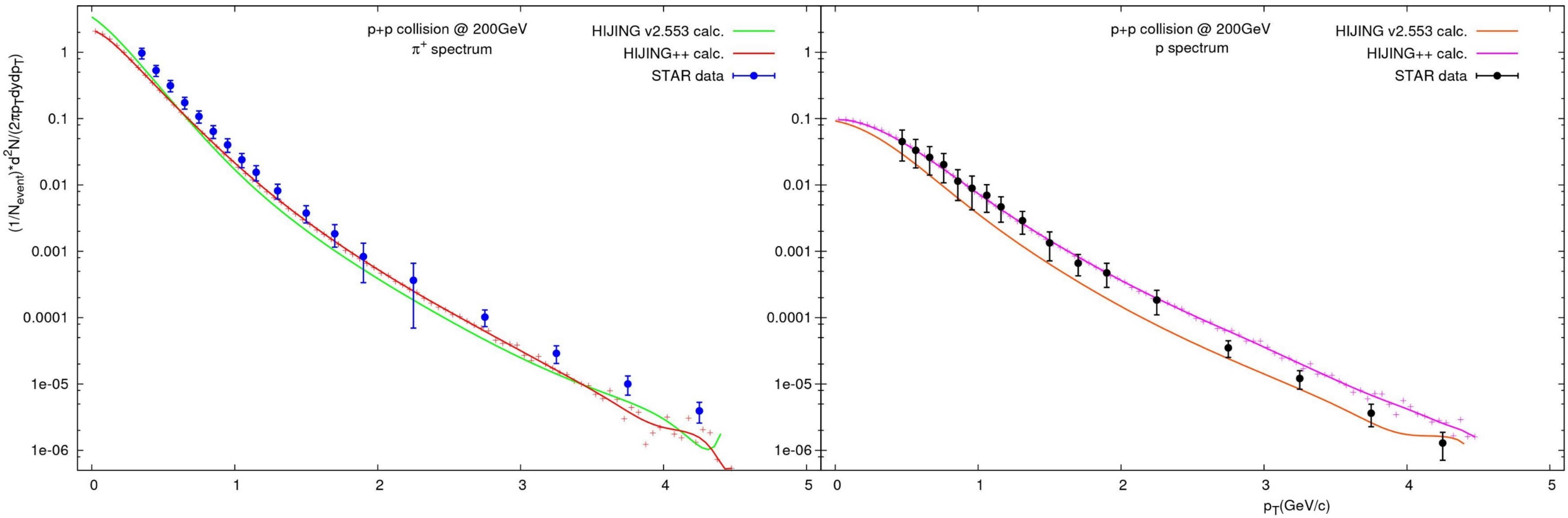
Performance: FORTRAN vs. HIJING(++)

- Runtime for 1 event with HIJING++:
  - pp or pA 35s initialization + event time
  - pA 47s initalization + event time
  - Initialization time with pre-calculated values is 4.5s
- Runtime (event time) measurements (preliminary, for Corei3 2.1 GHz):

(gain)	PYTHIA8	FORTRAN	C++ (single)	C++ (multicore)
pp	0.015 s/evt	0.264 s/evt	0.008 ms/evt	0.008 s/evt/thread
pA	–	3.509 s/evt	0.030 ms/evt	0.030 s/evt/thread
AA	–	379.96 s/evt	1.82 s/evt	1.82 s/evt/thread

Physics tests: pp collisions

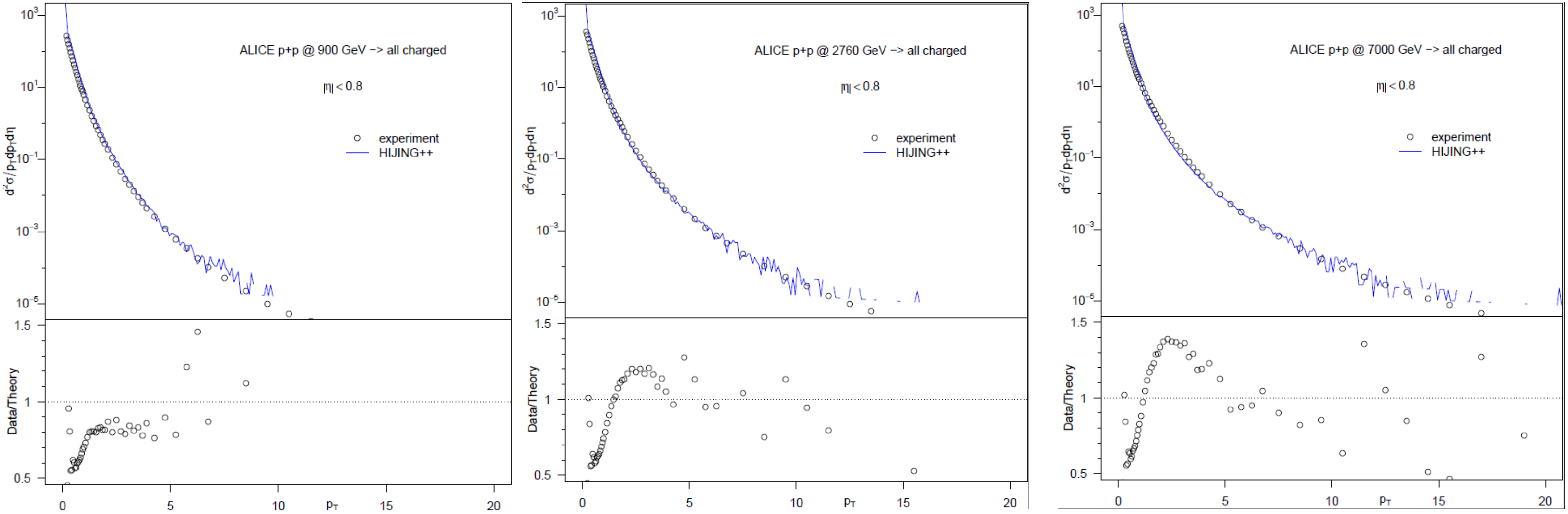
Code validation with „old” version and data in pp at RHIC energies



Data: STAR Collaboration, Phys.Lett. B637 page 161-169 (2006)

Physics tests: pp collisions

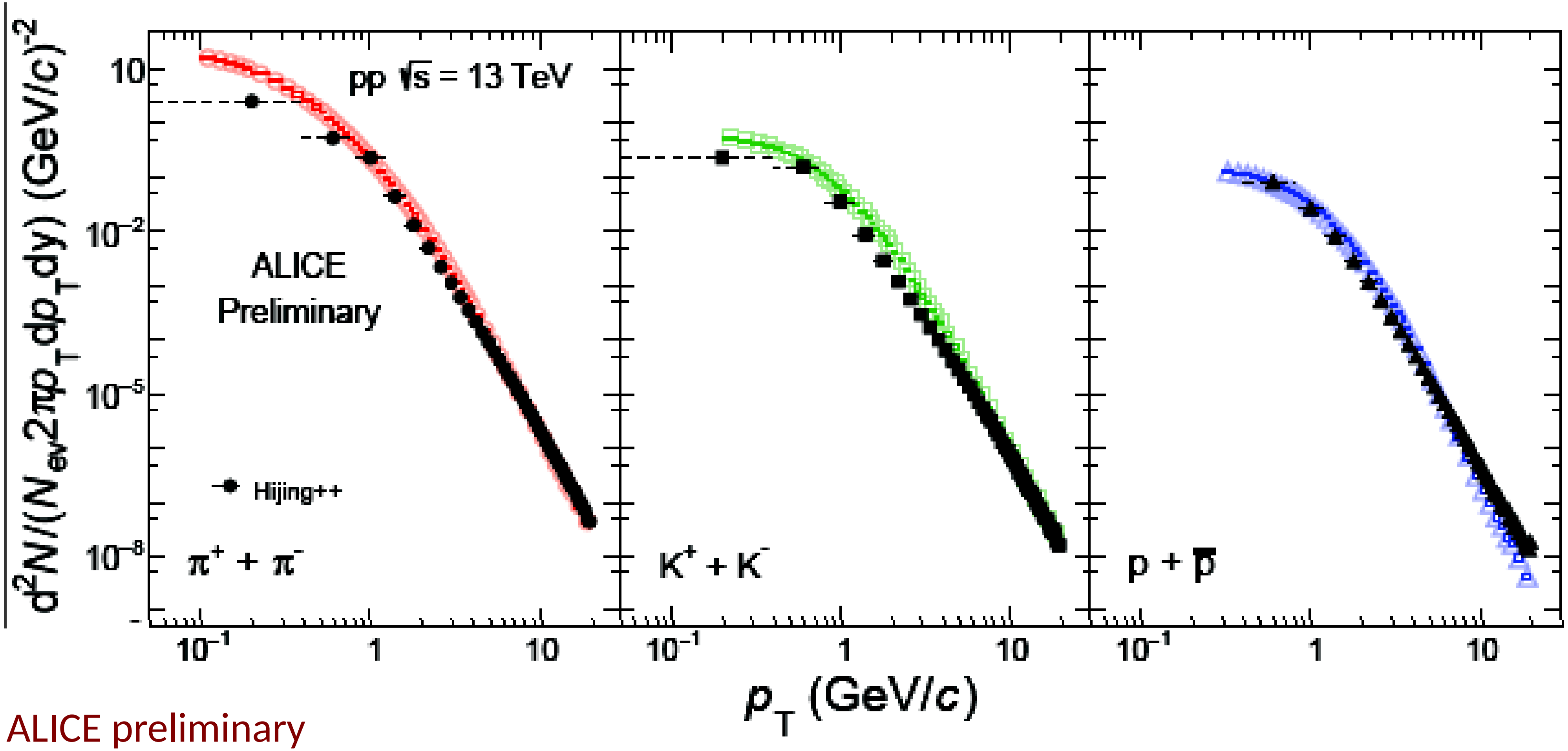
Code validation with „old" version and data in pp at LHC energies



Data: ALICE Collaboration, Eur. Phys. J. C73 2662 (2013)

Physics tests: pp collisions

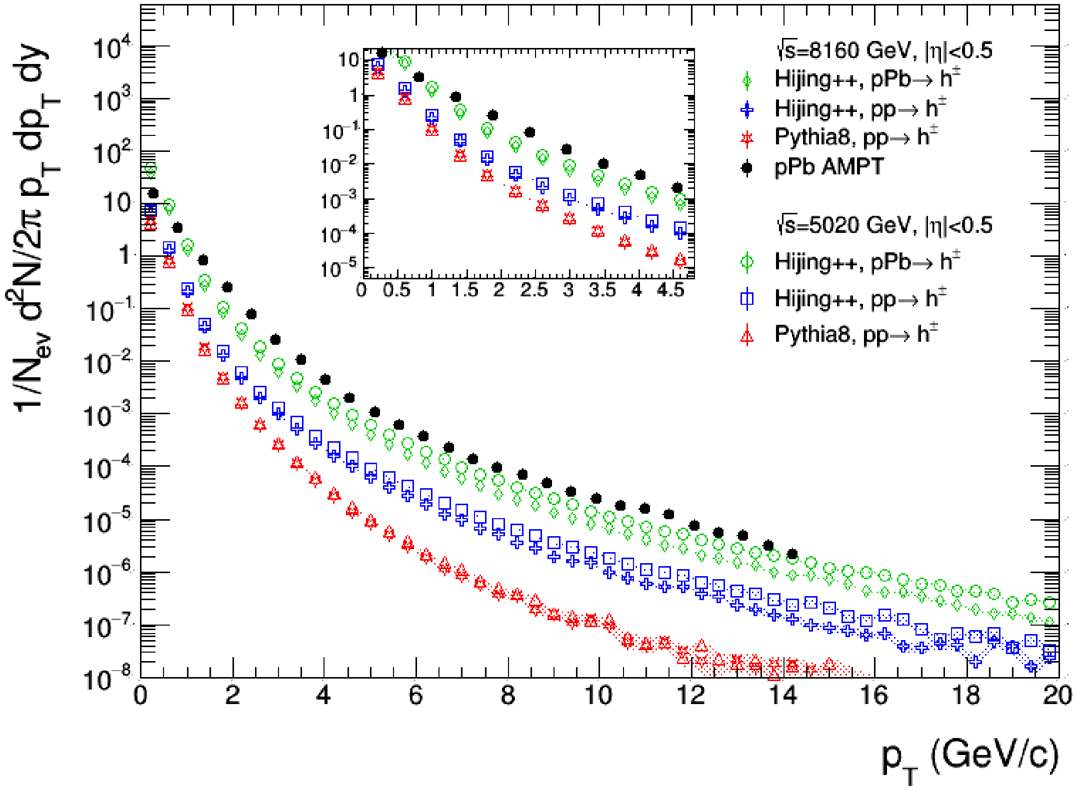
Predictions for the highest energy pp collisions at LHC energies



Data: ALICE preliminary

First caclulations: pp & pPb

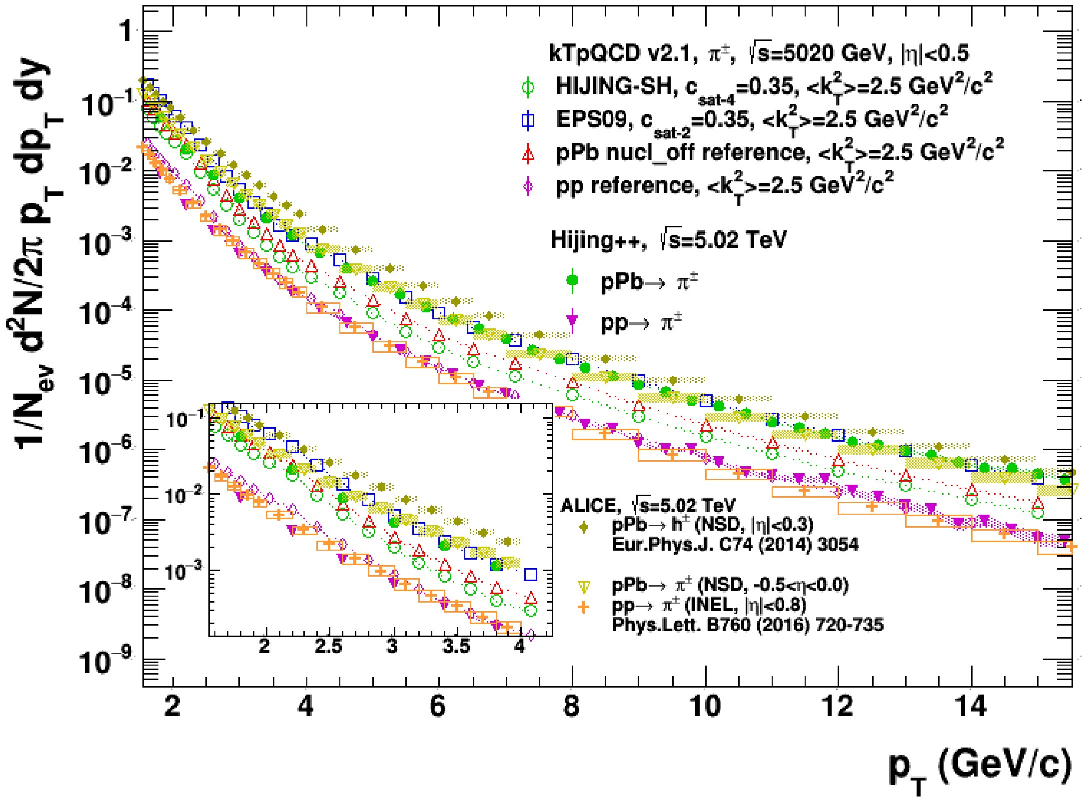
- HIJING++ pPb comparison (y=0)
  - Test: hadron spectra az 5.02 & 8 TeV
  - HIJING++ to Theory (kTpQCD, AMPT)
    - PYTHIA8 on pp
    - AMPT pPb



First caclulations: pp & pPb

HIJING++ pPb comparison (y=0)

- Test: hadron spectra az 5.02 & 8 TeV
- HIJING++ to Theory (kTpQCD, AMPT)
  - PYTHIA8 on pp
  - AMPT pPb
  - kTpQCD_v21 with HIJING & EPS09
- HIJING++ to LHC data:
  - ALICE data @ 5.02 TeV pp & pPb

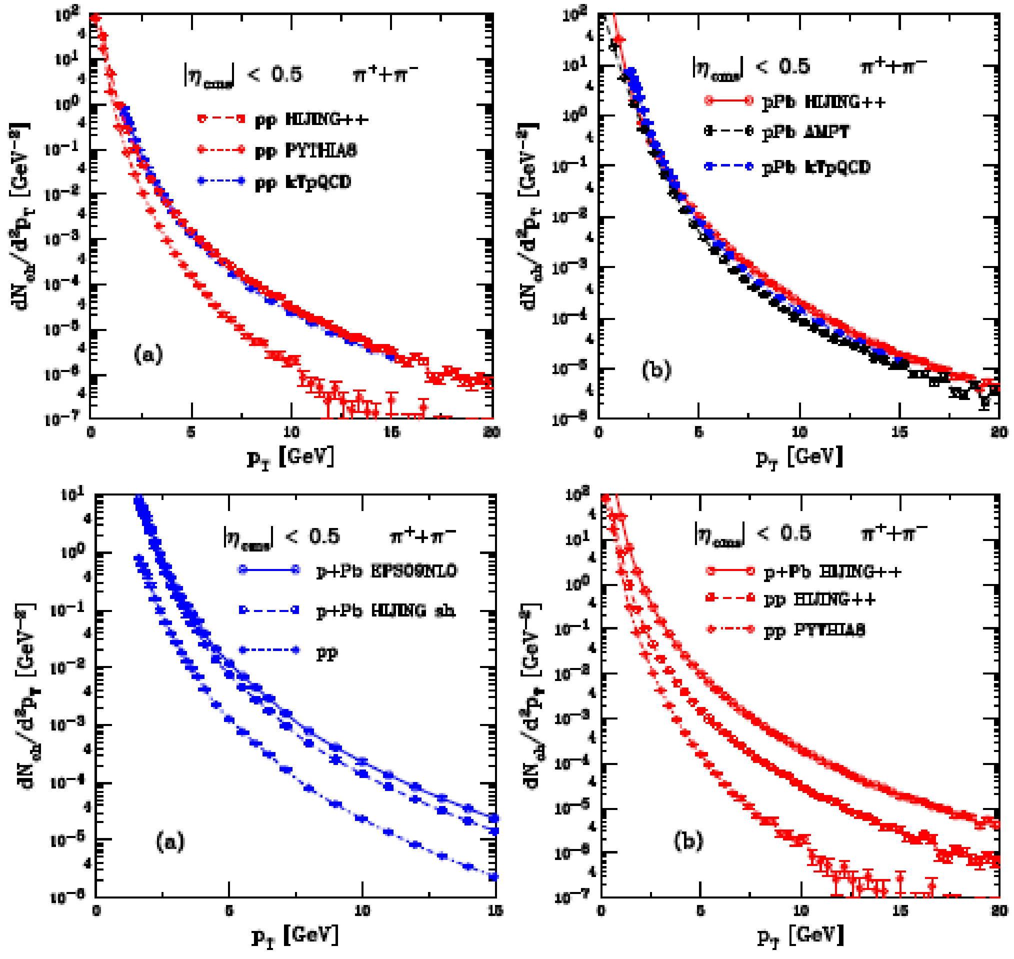


First predictions: pp & pPb

HIJING++ pp & pPb comparison

by R. Vogt (soon on the arXiv)

- Prediction: hadron spectra 8 TeV
- HIJING++ to Theory at 8 TeV
  - PYTHIA8 on pp
  - EPS09NLO
  - AMPT on pPb
  - kTpQCD_v21 on pp & pPb
- Results:
  - Differences at pp level
  - Similar spectra in pPb

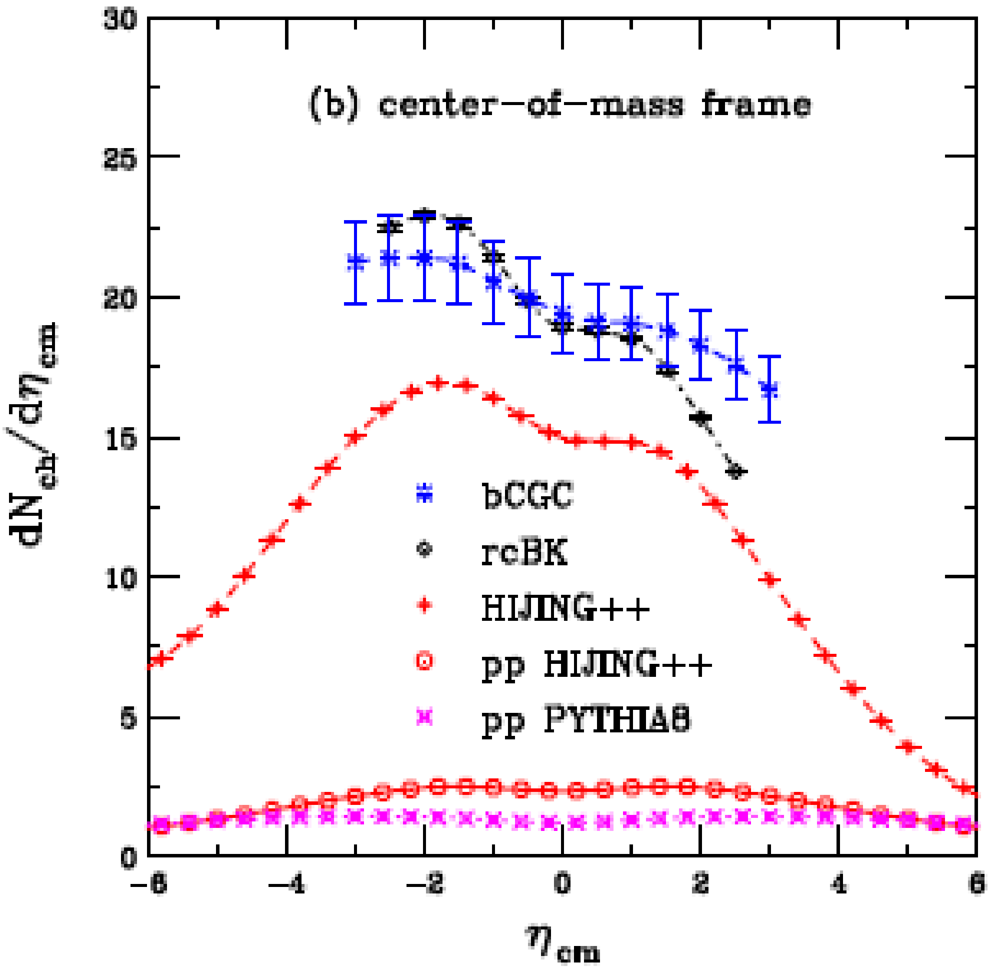


First predictions: pp & pPb

HIJING++ pp & pPb comparison

by R. Vogt (soon on the arXiv)

- Prediction: rapidity distribution 8 TeV
- HIJING++ to Theory at 8 TeV
  - PYTHIA8 on pp
  - rcBK
  - bCGC
- Results:
  - Major deviance for PYTHIA8 at midrapidity is coming from minijets



First predictions: pp & pPb

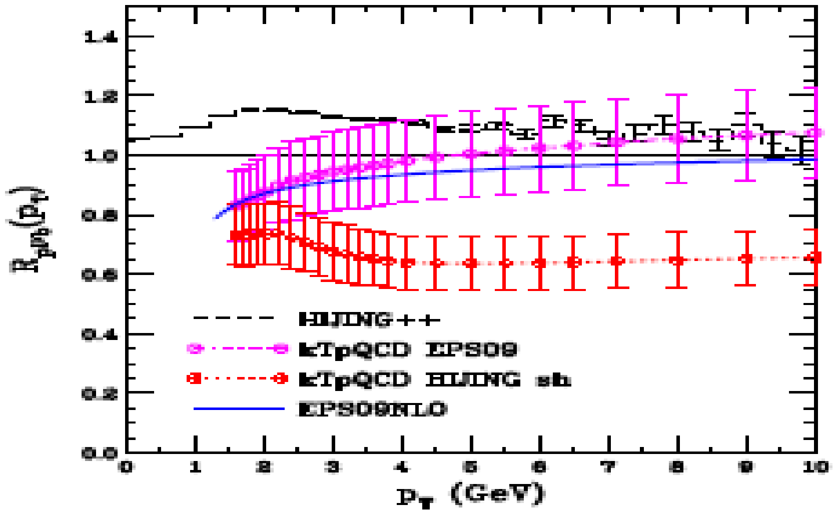
HIJING++ pp & pPb comparison

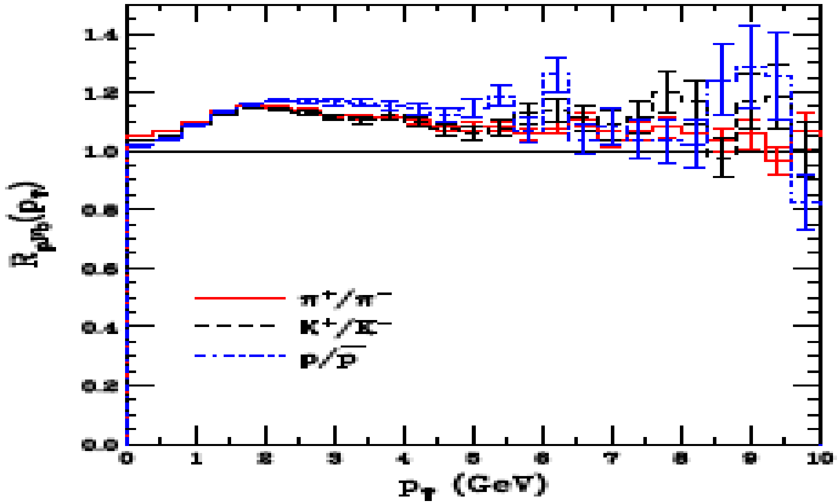
by R. Vogt (soon on the arXiv)

- Prediction:

$$R_{pA}(p_T) = \frac{dN_{pA}/dyd^2p_T}{\langle N_{\rm bin}\rangle dN_{pp}/dyd^2p_T}$$

- HIJING++ to Theory at 8 TeV
  - kTpQCD_v21 with EPS09 & HIJING
  - EPS09NLO
- Results:
  - Better agreement with EPS09
  - No relevant difference between π, K, p





Summary

- HIJING++
  - Coding from FORTRAN → C++ has been done
  - One more step HijCore & HijManager were introduced
  - Performance (parallel) tests are ongoing and promising
- First PHYSICS
  - Physics tests has been started
  - Soon on the arXiv (preliminary results and comparisons by R. Vogt et al)
- Next PYSICS
  - Step-by-step reconsidering of nuclear effect (shadowing with Q^2, jet quenching)

stay tuned….

Proposed Events

- 12^{th} High-pT Physics for the RHIC and LHC Era – “HpT4LHC”
  - Date: 2-5 October, 2017
  - Organizers: Bergen, at the University of Bergen (UiB), Western Norway University of Applied Sciences (HVL)
  - Web: under construction – Maybe WG3 meeting?

- New perspectives on Neutron Star Interiors
  - Date: 9-13 October, 2017
  - ECT*, Trento, Italy
  - Web: http://www.ectstar.eu/node/2230

BACKUP