Computer Tools for Nuclear Physics

Microscopic Transport Models
(outline)

Krzysztof Piasecki

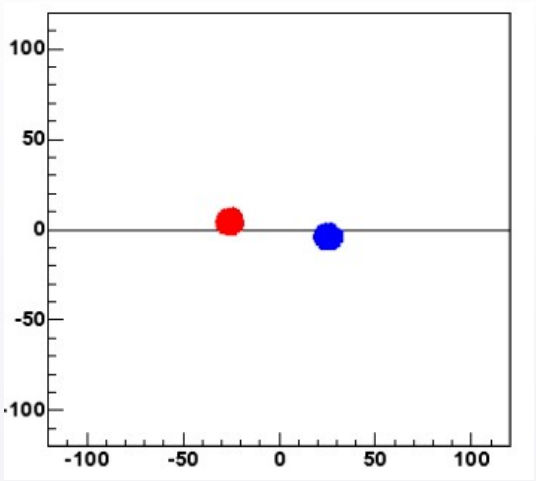
1. Choice of kinematics. Adequate degrees of freedom.
2. Energy of nucleus and nuclear matter
3. BUU transport equation. Cascade approach.
4. QMD family. Interactions between nucleons
5. Soft QCD (strings): LUND model within PYTHIA
6. Summary of model properties and additional info
7. Interlude: parsing text files, managing processes on Linux
8. Characteristics of selected transport models:
   GiBUU, UrQMD, SMASH, PHSD, JAM2/RQMD

krzysztof.piasecki@fuw.edu.pl

Microscopic Transport models

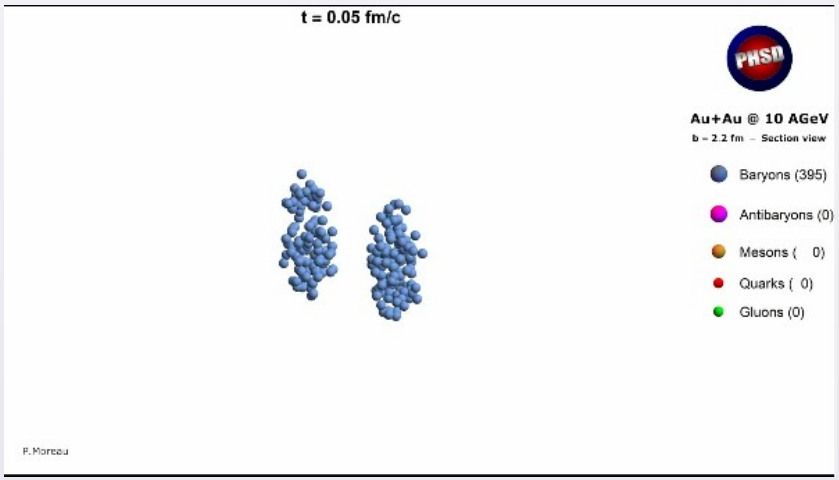
- **QMD** simulation of Au+Au collision at beam kinetic energy T_{Beam} **= 15 MeV/nucleon** (15*A* MeV)

[www.fuw.edu.pl/~kpias/rhic/QMD_Tb.015_b08_010.mpg]



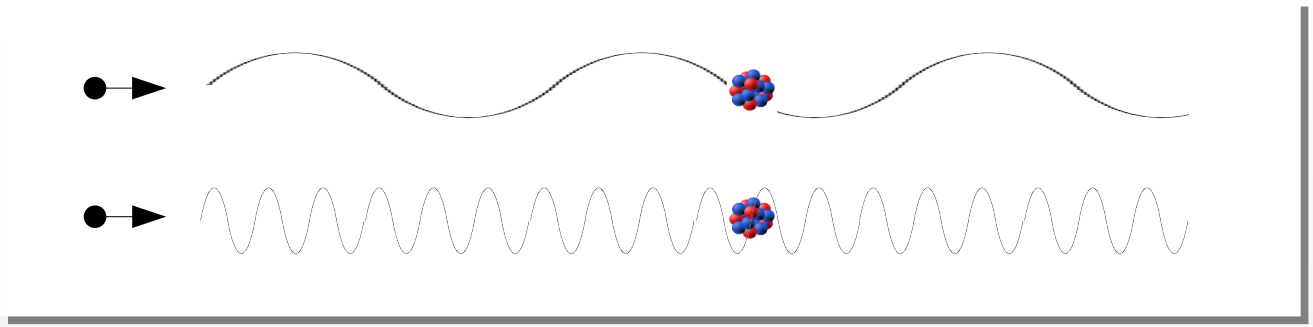
- **PHSD** simulation of Au+Au collision at beam kinetic energy T_{Beam} **= 10 GeV/nucleon** (10*A* GeV)

[theory.gsi.de/~ebratkov/phsd-project/PHSD/documents/movie_AuAu_10AGeV.mp4]



Collision of heavy ions and beam energy

- Which basic parameters change with beam energy?



T_{Beam} / nucleon	γ	β	$\lambda_{de\ Broglie}$ [fm]
5 MeV	1.005	0.10	13
50 MeV	1.05	0.3	4
500 MeV	1.5	0.75	1
5 GeV	6	0.99	0.2
50 GeV	55	0.9998	0.02
500 GeV	530	0.999998	0.002

Mean field sector
Nucleon ”sees” nucleus as a whole

Hadron-hadron sector
Hadron production in hh channels

Quark-gluon sector

- How to calculate it?

$$T_{Beam} = m_N c^2 (\gamma - 1) \rightarrow \gamma = 1 + \frac{T}{m_N c^2}$$

$$\beta = \sqrt{1 - \gamma^{-2}}$$

$$\lambda_{de\ Broglie\ (N)} = \frac{h}{p_N} = \frac{\hbar c \cdot 2\pi}{m_N c^2 \cdot \gamma\beta}$$

Model framework adequate to collision description

- With increasing beam energy the **relativistic kinematics** becomes more and more relevant (e.g. for $T_{Beam} = 500 \cdot A$ MeV, coefficient $\gamma = 1{,}5$).

- Increase of beam energy causes contraction of de Broglie wavelength, and therefore also the objects adequate to the scale (so-called **degrees of freedom**) :

  At **low energies** the interaction of selected nucleon with nucleus is described by the **potential**, generated by the whole nucleus (so-called ***Mean field***) .

  At **higher energies** ($T_{Beam} \gtrsim 100 \cdot A$ MeV) the **nucleon-nucleon collisions** (NN) appear.
  These collisions may lead to production of new hadrons (h) , e.g.:

  mesons π , K , ϕ, J/ψ , D … baryons N, Δ , Λ , Σ , Ξ …

  These hadrons move further within the nuclear matter – and can collide with other hadrons.
  The occuring process can be **two-body** ($h_1h_2 \rightarrow h_3h_4$) , **three-body** ($h_1h_2 \rightarrow h_3h_4h_5$) , …
  Probabilities of these processes are quantified by **energy-dependent cross sections** .

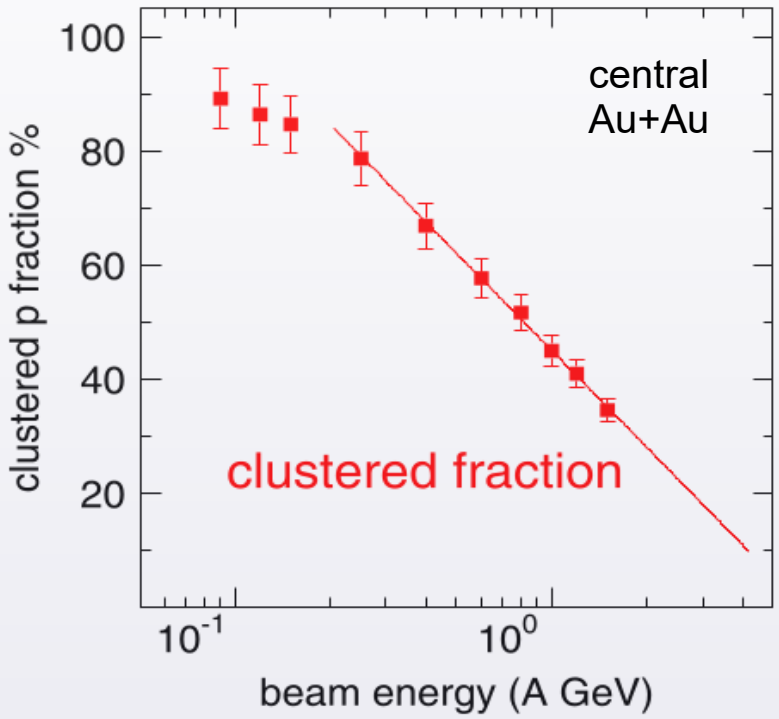
  At **ultrarelativistic energies** ($T_{Beam} \gtrsim 5 \cdot A$ GeV) the **quark-gluon** processes appear.

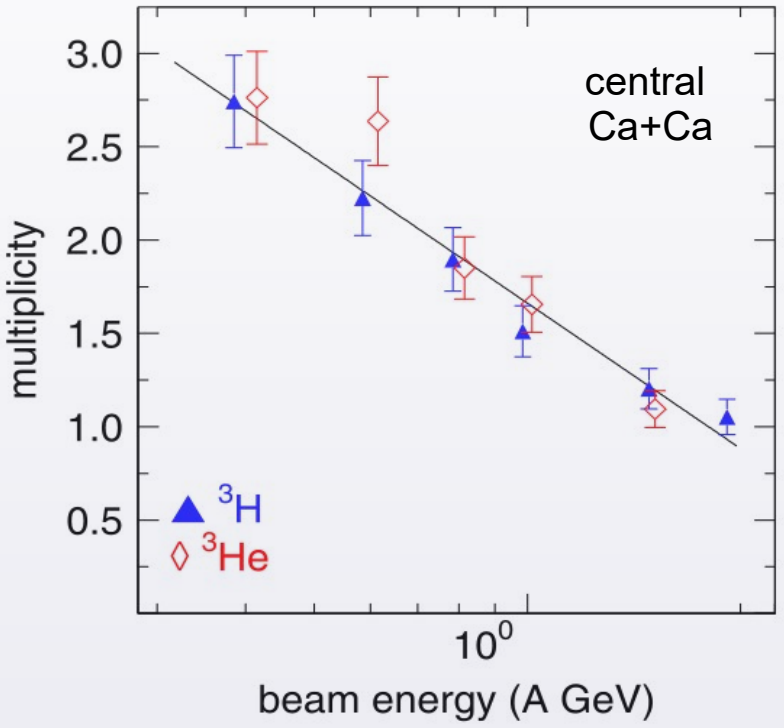
Selected model framework should be adequate to the beam energy.

Model framework adequate to collision description

- Proton-neutron (p-n) interaction may lead to the production of ***deuteron*** (d).
  Deuteron-neutron (d-n) interaction may create ***triton*** (t) and deuteron-proton (d-p) create ***^{3}He*** . Etc...
  These particles are called the ***Light Charged Particles*** (***LCP, clusters, small nuclear fragments***).

- At $T_{Beam} \sim 2 \cdot A$ GeV, as much as ¼ protons are bound in LCP. Trace amounts of deuterons are seen in LHC.





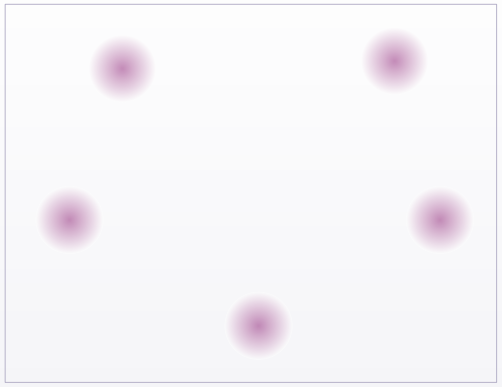
W.Reisdorf et al., Nucl.Phys. A 848, 366 (2010)

- Usually, if LCP appears in the models, they are assembled from p and n, already after the simulation.
  Full description of production and dynamics of LCP is currently absent.
  Attempts: ***PHQMD*** (*under development*).

➡ **If description of LCP is relevant, one should consider the predictive power of model-candidates.**

Energy in nucleus (schematic view)

In ∞

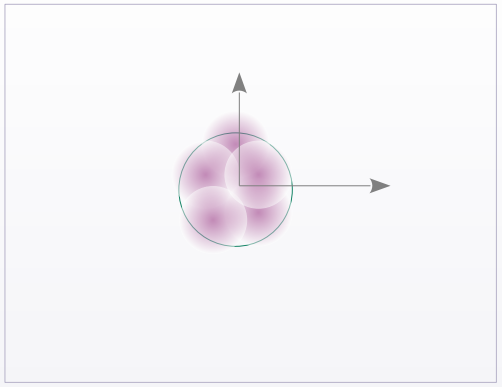


Nucleons are free,
$\bar{p}_i = \bar{0}$.

$$E_{System} = \sum_i m_{N,i} c^2$$

$$\frac{E_{System}}{A} = m_N c^2$$

Nucleus as a whole:



In nucleus' own frame, $\bar{p}_{nucleus} = \bar{0}$.

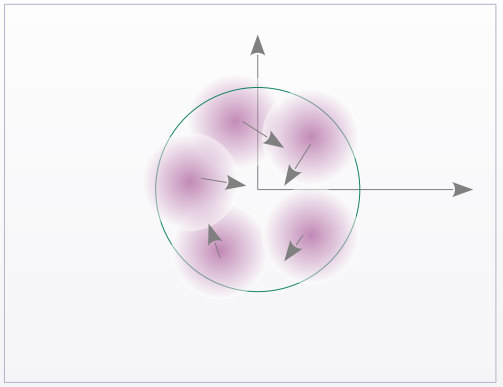
$$E_n = M_n c^2 \underbrace{=} \sum_i m_{N,i} c^2 - B_J$$

quantitatively

$$\frac{E_n}{A} = m_N c^2 - \frac{B_{nucleus}}{A}$$

$\approx$ 8 MeV

Nucleus as a sum of nucleons:



Pauli exclusion → nucleons must move,
$\bar{p}_i \neq \bar{0}$.

$$E_{nucl.} = \sum_i e_i$$

$$e_i = \sqrt{\left(m_{N,i} c^2\right)^2 + \left(\vec{p}_i c\right)^2} + \frac{U_i}{A}$$

In reality the formulae are more complex.

① Coulomb interaction must be added (φ, $\bar{A}$) .

② Within QCD with strong interactions,
so-called scalar and vector potentials appear,
which e.g. change mass of nucleon inside nuclear matter.

$$e_i = \sqrt{\left(m_{N,i} c^2 - V_S\right)^2 + \left(\vec{p}_i c - \vec{V}_W\right)^2} + V_{W,0}$$

(not by heart :))

Nuclear matter equation of state (sketch)

- **Real atomic nucleus**: system of nucleons with finite *N* and *V* that interacts strongly, weekly and by Coulomb. Nucleons from ∞ fall into the potential well, wherein the acquire kinetic energy (Fermi motion). Energy is decreased by binding energy E_B. In the **liquid drop model**:

$$E_B = a_V A - a_S A^{2/3} - a_C \frac{Z^2}{A^{1/3}} - a_A \frac{(N-Z)^2}{A} \pm \delta(A,Z)$$

For average nucleus: $E_B / A \approx$ 8 MeV.

- **Nuclear matter**: abstract matter built of nucleons, of infinite *N* and *V*, but finite ρ .
Depth of binding depends on: temperature, density, n/p ratio.

$$E_B(\rho = \rho_0,\ T = 0) = a_V A \quad \ldots \quad - a_A \frac{(N-Z)^2}{A}$$

(ρ_0 = 0.17 fm^{-3} , "normal density of nuclear matter")

For symmetric (*N* = *Z*) nuclear matter in normal state: $E_B / A \approx$ 16 MeV.

- During the collision the matter condenses and subsequently expands. How E_B / A depends on density ρ?
Two points are fixed:

① for $\rho = 0$ (free nucleons) , $E_B / A =$ 0 MeV
② for $\rho = \rho_0$ (stable nucleus) , $E_B / A =$ –16 MeV

- One can derive models of interactions that aim to reproduce findings [①, ②] .
One can also insert into the model some functional dependence that agrees to [①, ②].
The testing ground are the kinematic distributions of particles: if/when model agrees with experiments?

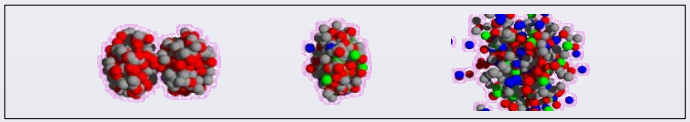
Nuclear matter equation of state (sketch)

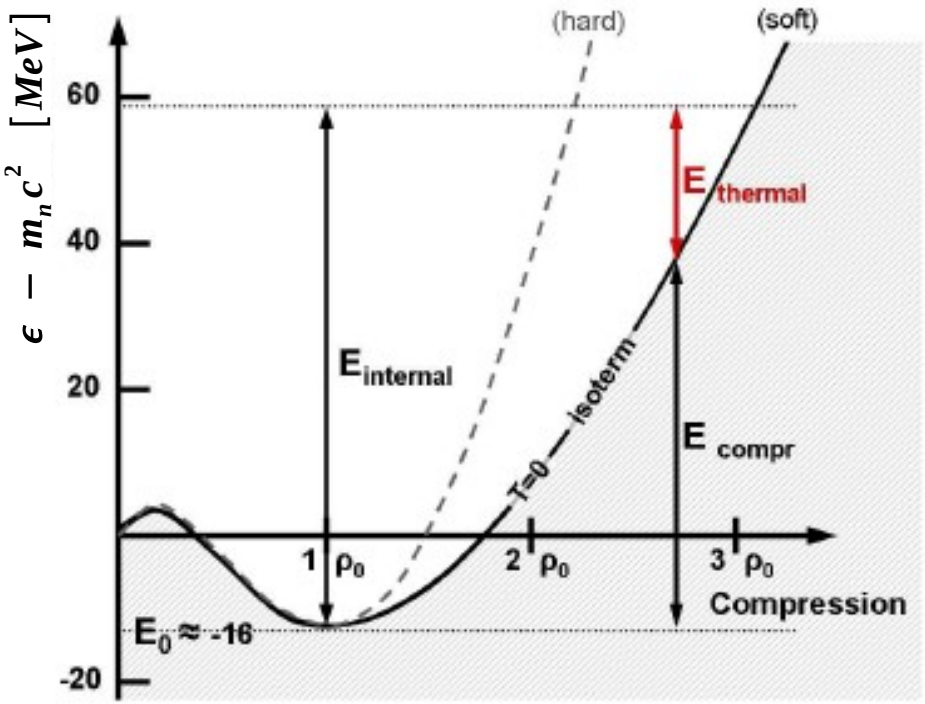
- **Nuclear matter equation of state ("EoS") :**

$$\epsilon \equiv \frac{E}{A}(\rho, T) = m_n c^2 + \epsilon_{Kin}(\rho, T=0) + u(\rho, T=0) + \epsilon(\rho, T>0)$$

- It's worth to compare the plot to the dynamics of the nuclear collision. Let's trace it:

$\rho_0 \rightarrow 3\rho_0 \rightarrow \rho_0$





- **Thermodynamical quantities:**

pressure $p = \ldots = -\rho \left.\frac{\partial \epsilon}{\partial \rho}\right|_{T=0}$

Bulk modulus $\kappa = \ldots = +\rho \left.\frac{\partial p}{\partial \rho}\right|_{T=0}$

- **In nuclear studies:**

Incompressibility modulus $\kappa \equiv 9 \left.\frac{\partial p}{\partial \rho}\right|_{T=0}$

- **In the state of normal nuclear density ($\rho = \rho_0$) :**

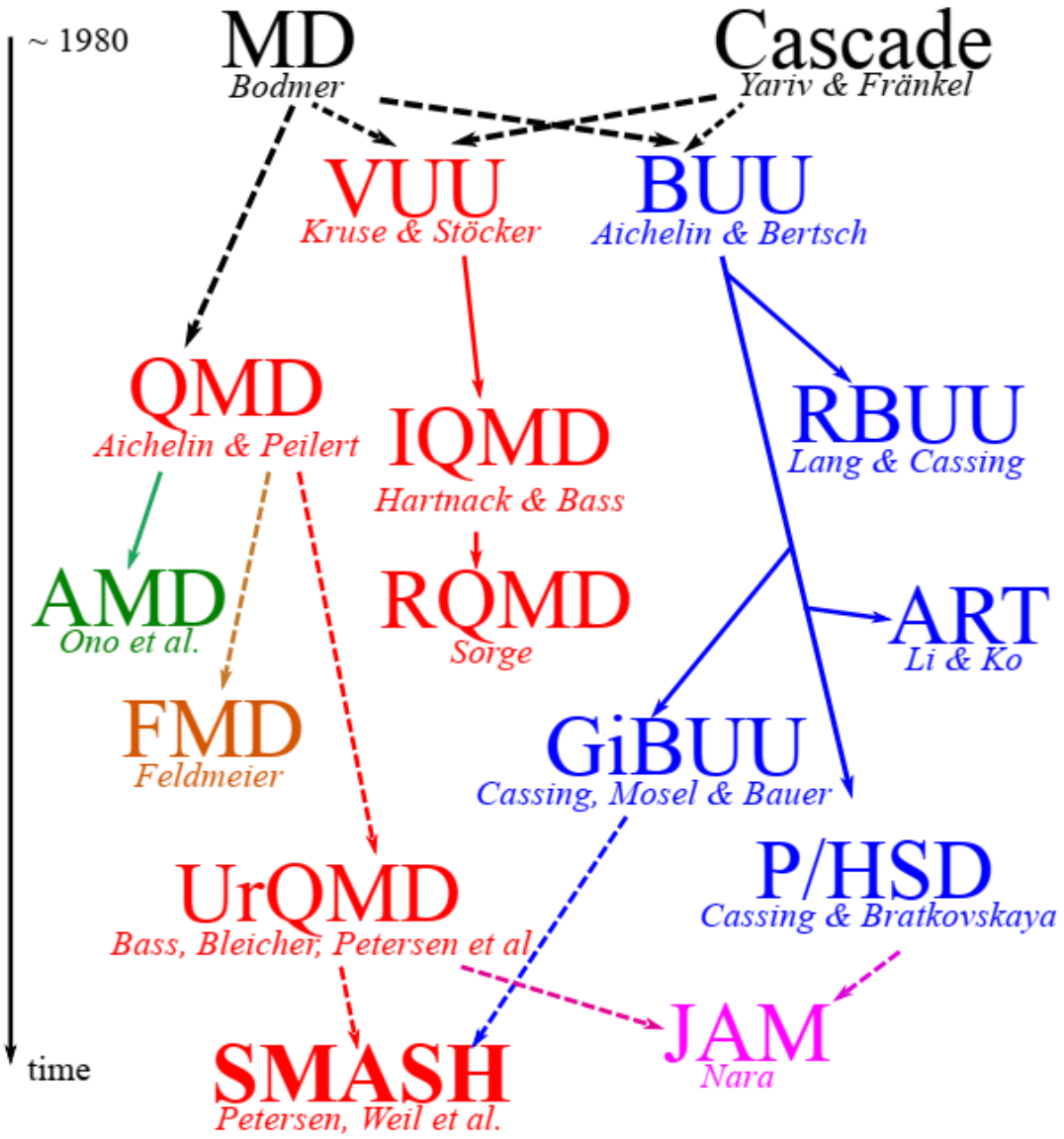
$$\kappa(\rho_0) = \ldots = 9 \left.\frac{d^2 \epsilon}{d \rho^2}\right|_{\rho=\rho_0,\ T=0}$$

(or: ~curvature of ϵ at $\rho = \rho_0$)

- **High (low) κ : "*hard*" ("*soft*") *EoS***

Usually in papers: "soft" : $\kappa \approx 200$ MeV ,
"hard": $\kappa \approx 380$ MeV

Transport models across decades



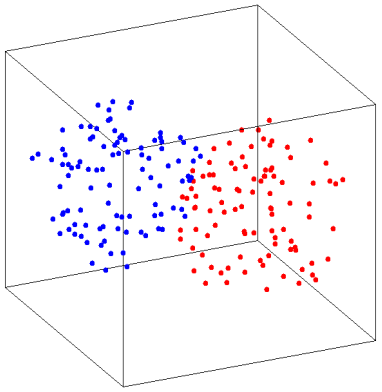
Many models were developed (more than shown above), but many of them inherit the main framework. We shall describe two branches: family of **BUU** and **QMD** models.

Framework of family of BUU models (outline)

- **B**olzmann – **Ü**hling – **U**hlenbeck transport equation. Exemplary introductions:

  B.Serot, J.Walecka, arXiv:nucl/th/9701058 section 7A
  C.Hartnack et al., Eur. Phys. J. A 1, 151 (1998)

- Outline. We consider *N* particles moving in the phase space (x^3, p^3). Their distribution is described by the function $f(\mathbf{r}, \mathbf{p}, t)$. Particles are **in a field described by a potential *U*** (mean field; it's the BUU's feature).



$$N = \int d^3\boldsymbol{r} \int d^3\boldsymbol{p}\ f(\boldsymbol{r},\boldsymbol{p},t)$$

$$df \equiv \frac{\partial f}{\partial t}dt + \frac{\partial f}{\partial r_i}dr_i + \frac{\partial f}{\partial p_i}dp_i$$

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \nabla_r f \cdot \boldsymbol{v} - \nabla_p f \cdot \nabla_r U$$

$$\frac{\partial f}{\partial r_i}dr_i = (\nabla_r f)\cdot d\boldsymbol{r}_i \qquad d\boldsymbol{r} = \boldsymbol{v}\,dt \qquad d\boldsymbol{p} = -\nabla_r U\,dt$$

- Potential *U* of the mean field:

$$U = U(\rho) + U_{Coulomb} + U(\vec{p})$$

$$U(\rho) = \alpha\frac{\rho}{\rho_0} + \beta\left(\frac{\rho}{\rho_0}\right)^{\gamma}$$

(*nuclear matter "equation of state", typical parameterization*)

$$U(\vec{p}) = \delta\frac{8}{\rho_0(2\pi)^3}\int d^3p'\frac{f(\boldsymbol{r},\boldsymbol{p}')}{1+\left(\frac{\boldsymbol{p}-\boldsymbol{p}'}{\Lambda}\right)^2}$$

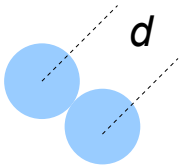
(*momentum-dependent term of potential typical parameterization*)

- If *N = const*, we would have: $\frac{df}{dt} = 0$

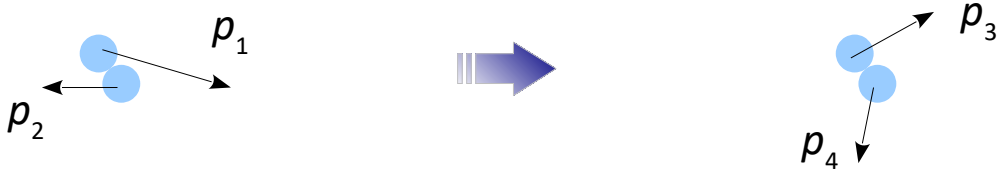
Framework of family of BUU models (outline)

- … but there are also **collisions**. Let's consider a two-body process (1,2) → (3,4).
  Probability of collision is described by the cross section σ (more precisely, dσ/dΩ) .
  A collision occurs if:

$$d < \sqrt{\frac{\sigma_{NN}}{\pi}}$$



- A collision generates a single loss in cells of momenta p_1 and p_2 for the sake of gain in p_3 and p_4.



- But one should account for the Pauli exclusion (momenta cells p_3 i p_4 are not always available).
  It is included probabilistically:

f (**r**,**p**,t)	is a probability of occupation of momentum cell p.
$1 - f$(**r**,**p**,t)	is a probability that momentum cell p is free.

→ Additional term in BUU equation, describing collisions: $\frac{df}{dt} = I_{coll}$

$$I_{coll} = \sum_{\substack{\text{reaction} \\ \text{channels}}} \int \frac{d^3p_2\, d^3p_3\, d^3p_4}{(2\pi)^6} \frac{d\sigma_{12\to 34}}{d\Omega'} \cdot \Big[\underbrace{(1-f_1)(1-f_2)f_3f_4}_{\text{Gains}} - \underbrace{f_1f_2(1-f_3)(1-f_4)}_{\text{Losses}}\Big] \cdot \underbrace{\delta(p_1+p_2-p_3-p_4)}_{\substack{\text{Momentum} \\ \text{conservation}}}$$

- The **BUU** (Boltzmann-Ühling-Uhlenbeck) equation in short:

$$\frac{df}{dt} = I_{coll}$$

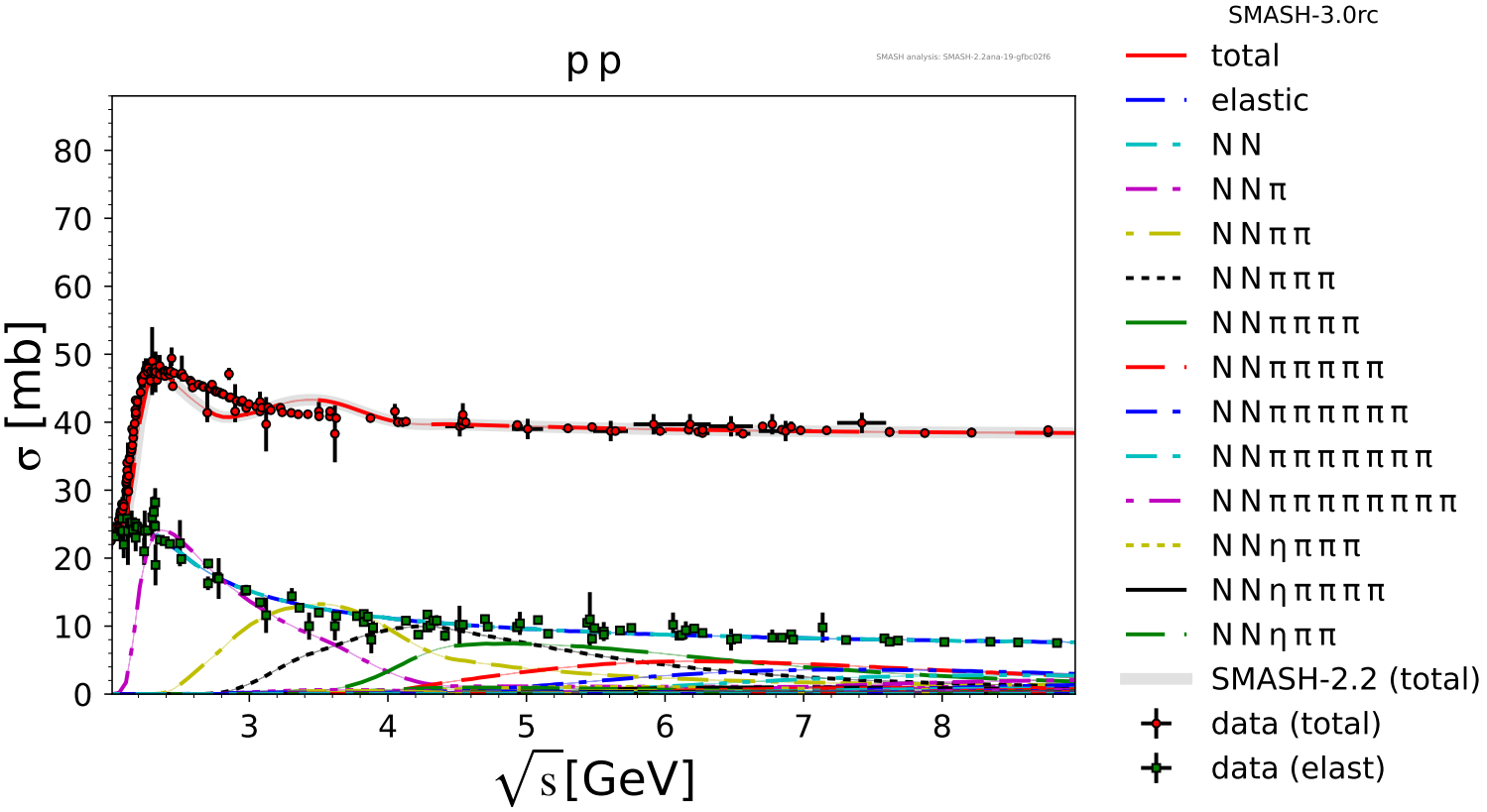
Framework of family of BUU models (outline)

- **Production of new particles** is also realized through collisions.
  Each transport model, for all the implemented channels, uses a list of parametrizations of σ = f(energy) .
  They are usually fits to the experimental databases. Some channels (like string production) are algorythmic.

- For channels experimentally (nearly) impossible to extract (e.g. $N\Delta \rightarrow X$),
  physics arguments are invoked like: detailed balance, isospin symmetry, strangeness balance etc.

- **Flagship papers of specific models** usually report the relevant formulae and figures of σ = f(energy).

  A nice demonstrator is SMASH's web-based collection of its σ plots [**here**]. E.g.



BUU-type transport models: working principle

- For each collision a distribution (of nucleons) $f(\boldsymbol{r}, \boldsymbol{p}, t)$ *in the initial state* is prepared:

$$f(\boldsymbol{r},\boldsymbol{p},t) = \sum_{i=1}^{A_{beam}+A_{target}} \delta(\boldsymbol{r}-\boldsymbol{r}_i)\ \delta(\boldsymbol{p}-\boldsymbol{p}_i)$$

- **Initialization of positions**

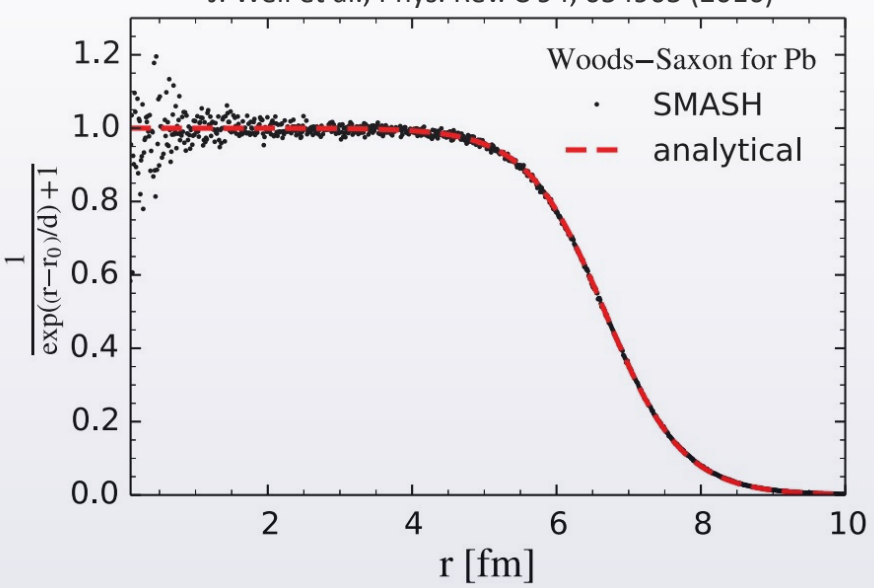
Usually: **Woods-Saxon** distribution with parameters based on experimental charge distribution

$$\frac{d^3N}{dr^3} = \frac{\rho_0}{\exp\left(\frac{r-r_0}{d}\right) + 1}$$

TABLE I. This table summarizes the specific parameters used in the Woods–Saxon initialization for some nuclei.

Nucleus	A	r_0 [fm]	d [fm]
U	238	6.86	0.556
Pb	208	6.67	0.54
Au	197	6.38	0.535
Cu	63	4.20641	0.597

J. Weil et al., Phys. Rev. C 94, 054905 (2016)



- **Initialization of momenta**

Usually: for nucleons away by r from the center, p is pulled randomly from unit sphere, i.e.: $p \in [0, p_{Fermi}(r)]$

$$p_{\text{Fermi}}(\vec{r}) = \hbar \left[\frac{3}{2}\pi^2\, \rho(\vec{r}) \right]^{1/3}$$

Usually, Fermi momentum in the middle of a nucleus $\approx$ 270 MeV/c .

BUU-type transport models: working principle

- Motion of a given particle is governed by the *1-body hamiltonian* :

  Case of non-relativistic potential: $H = \sqrt{p^2 + m^2} + U$

  Case of relativistic potential: $H = \sqrt{\left(\vec{p} - \vec{V}_V\right)^2 + \left(m - U_S\right)^2} + V_0$

- *In consecutive time steps* (n) position and momentum of i-th particle evolves due to:

$$\begin{cases} \dot{\boldsymbol{r}}_i = \nabla_{\boldsymbol{p}_i} H \\ \dot{\boldsymbol{p}}_i = -\nabla_{\boldsymbol{r}_i} H \end{cases} \Rightarrow \begin{cases} \boldsymbol{r}_i(n+1) = \boldsymbol{r}_i(n) + \dfrac{\boldsymbol{p}_i\left(n+\frac{1}{2}\right)}{m} \cdot \Delta t + \nabla_{p_i} U_i \cdot \Delta t \\ \boldsymbol{p}_i\left(n+\frac{1}{2}\right) = \boldsymbol{p}_i\left(n-\frac{1}{2}\right) - \nabla_{\boldsymbol{r}_i} U_i(n) \cdot \Delta t \end{cases}$$

⊕ collisions: if $d < d_{min}$ and no Pauli blocking → then $\delta(p_1)$ and $\delta(p_2)$ changes into $\delta(p_3)$ and $\delta(p_4)$

- <u>Caution</u>: Within BUU the only field-type (continuous) interaction – is through the mean field.
  No single nucleon–nucleon interactions (except for collisions).

  → BUU usually knows nothing about adjacent nucleons coalescing into LCPs (d, t, ^{3}He, …).

Cascade approach

- **Cascade** is a collective name for various simulation codes (mode in them) that share following features:

  - ▻ Sharp sphere **potential** well (**constant** inside target nucleus) .
    As $\vec{F} = -\nabla V \Rightarrow$ no force applied to tracked hadrons when inside nucleus

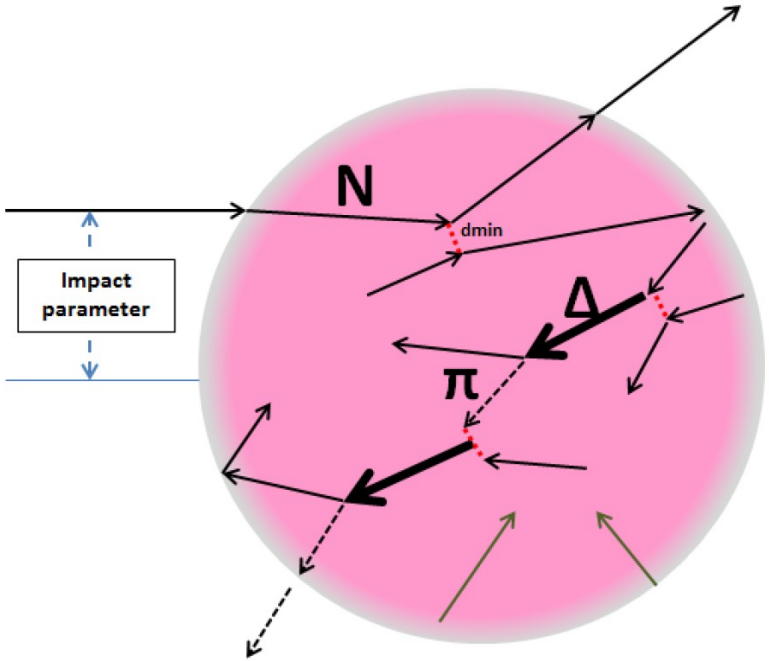
  - ▻ Potential has an **edge** (around nucleus' **skin**)
    ⇒ nucleons are reflected back or tunnelled through

  - ▻ *Caution*: the initial positioning is usually not done according to the uniform sphere, but to Woods-Saxon or modified oscillator distributior

  - ▻ Overall **nucleon density stays constant** (not for high-energy Heavy-Ion collisions)

  - ▻ Particles follow straight lines.
    They can do **binary collisions**, but Pauli exclusion is checked.

  - ▻ In collisions π or Δ can be created.
    Δs decay into [N π].
    Some Cascade codes generate **LCPs**.



[Credit: S. Sharma, Ph.D., Jagiellonian Univ., 2015]

- **Where can these models be useful?**

  - ☞ Collisions **initiated by hadrons** (p, π, …) or **light nuclei**, as long as assumption of constant ρ holds.

- In **QMD** models the objects are the particle wave packets:

$$\Psi = \prod_i \psi_i \sim \prod_i \exp\left[-\frac{(\boldsymbol{x_i}-\boldsymbol{r_i}(t))^2}{L}\right] \cdot \exp\left[i\,\boldsymbol{x_i}\,\boldsymbol{p_i}(t)\right]$$

e.g. C.Hartnack et al. Eur. Phys. J. A 1, 151 (1998) arxiv.org/abs/nucl-th/9811015

Parameter *L* describes the packet size. It is found such that Ψ describes the density drop at the nucleus' skin.

- An *i*-th particle moves in the potential ⊕ collides with neighbours.

However, the potential is built from a sum of potentials from '*j-th*' particles surrounding the '*i*-th' one.

Quantum hamiltonian:

$$\langle H\rangle = \langle T\rangle + \langle V\rangle = \sum_i T_i + \sum_i \sum_{j>i} \int \psi_i^* \psi_j^*\; V^{ij}(x_1,x_2)\;\; \psi_i \psi_j \;\; dx_1 dx_2$$

Equation of motion:

$$\begin{cases} \dot{\boldsymbol{r}}_i = \dfrac{\boldsymbol{p_i}}{m} + \nabla_{p_i} \sum_j \langle V_{ij}\rangle \\ \dot{\boldsymbol{p}}_i = -\nabla_{r_i} \sum_{j\neq i} \langle V_{ij}\rangle \end{cases}$$

(in fact, the simulation traces the centroids of wave packets)

- **N_i–N_j interactions :**

$$V_{ij} = V_{ij}^{Skyrme} + V_{ij}^{Yukawa} + V_{ij}^{p-dependent} + V_{ij}^{Coulomb} + V_{ij}^{p-n\ assymetry}$$

$$= \left[t_1 + t_2\,\rho^{\gamma-1}(\boldsymbol{x_i})\right] \cdot \delta(\boldsymbol{x}_i - \boldsymbol{x}_j) + t_3 \frac{\exp\left\{-|\boldsymbol{x}_i - \boldsymbol{x}_j|/\mu\right\}}{|\boldsymbol{x}_i - \boldsymbol{x}_j|/\mu} +$$

$$+ t_4 \ln^2\left(1 + t_5(\boldsymbol{p}_i - \boldsymbol{p}_j)^2\right) \cdot \delta(\boldsymbol{x}_i - \boldsymbol{x}_j) + \frac{Z_i Z_j e^2}{\{\boldsymbol{x}_i - \boldsymbol{x}_j\}} + t_6 \frac{1}{\rho_0}\, T_i^3 T_j^3 \cdot \delta(\boldsymbol{x}_i - \boldsymbol{x}_j)$$

- From the *talk* of Ch. Hartnack and J. Aichelin (2015), the authors of **IQMD**:

Definition of the potentials

$$
\begin{aligned}
V^{ij} &= G^{ij} + V^{ij}_{\rm Coul} \\
&= V^{ij}_{\rm Skyrme} + V^{ij}_{\rm Yuk} + V^{ij}_{\rm mdi} + V^{ij}_{\rm Coul} + V^{ij}_{sym} \\
&= t_1\delta(\vec{x}_i-\vec{x}_j) + t_2\delta(\vec{x}_i-\vec{x}_j)\rho^{\gamma-1}(\vec{x}_i) + t_3\frac{\exp\{-|\vec{x}_i-\vec{x}_j|/\mu\}}{|\vec{x}_i-\vec{x}_j|/\mu} + \\
& t_4\ln^2(1+t_5(\vec{p}_i-\vec{p}_j)^2)\delta(\vec{x}_i-\vec{x}_j) + \frac{Z_iZ_je^2}{|\vec{x}_i-\vec{x}_j|} + \\
& t_6\frac{1}{\varrho_0}T_3^iT_3^j\delta(\vec{r}_i-\vec{r}_j)
\end{aligned}
$$

Bethe Weizsaecker –mass formula:

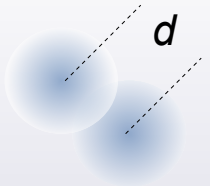
Volume term **+Surface term** **+Coulomb term** **+symmetry term**

(nucl. eos) **(+pairing term not included)** **(asy- eos)**

Outline of transport models. QMD Family

- **Hadron-hadron collision**. Occurs if the centroids between 2 hadrons approach closer than:

$$d < \sqrt{\frac{\sigma_{hh}}{\pi}}$$



For fermions, Pauli exclusion is accounted for.

- By selecting components V_{ij} : $V^{\text{Skyrme}} + V^{\text{Yukawa}} + V^{\text{p-dependent}}$, through the respective convolution one can "reproduce" the equation of state of nuclear matter.

$$U(\rho) = \alpha \frac{\rho_{int}}{\rho_0} + \beta \left(\frac{\rho_{int}}{\rho_0}\right)^{\gamma} + U^{\text{p-dependent}}$$

$$U^{p-dependent} = \delta \cdot \ln^2\left(\epsilon \cdot (\Delta p)^2 + 1\right)$$

Δp : particle momentum w.r.t $\langle p \rangle$ of medium

One can choose the V_{ij} parameters so that the simulation reproduces given Equation of State, (including "recreation" of the hard/soft scenarios)

	α (MeV)	β (MeV)	γ	δ (MeV)
S	-356	303	1.17	—
SM	-390	320	1.14	1.57
H	-124	71	2.00	—
HM	-130	59	2.09	1.57

- The **Isospin-QMD** (**IQMD**) variant :
  - Separate treatment of n, p, Δ, π
  - Possible proton-neutron interactions
  - one can describe the asymmetric (n ≠ p) part of EoS

IQMD provides the framework for description of binding of nucleons into LCPs (clusters).

Available energy in channel

- **Fact:** each scalar product of two four-vectors is invariant w. r. t. Lorentz Transform.

$$p_\mu p^\mu = E^2 - \vec{p}^2 = const \quad \rightarrow \text{ quite obvious, as: } \quad E^2 - \vec{p}^2 = m^2$$

$$\left(\sum_i E_i\right)^2 - \left(\sum_i \vec{p}_i\right)^2 = \text{invariant} \equiv s$$

at the same time:

$$\begin{cases} \sum_i E_i = const \\ \sum_i \vec{p}_i = const \end{cases}$$

→ "*s*" is not only invariant.
It also is the integral of motion
[= const(*t*)].

- Centre of mass frame (CM) : such frame that $\sum_i \vec{p}_{i,CM} = \vec{0}$

$$s \equiv \left(\sum_i E_{i,CM}\right)^2 - \left(\sum_i \vec{p}_{i,CM}\right)^2 = \left(\sum_i E_{i,CM}\right)^2$$

$$\rightarrow \quad \sqrt{s} = \sum_i E_{i,CM}$$

$\sqrt{s}$ is called the "**available energy**". It amounts to the sum of energy of the system in the CM frame.

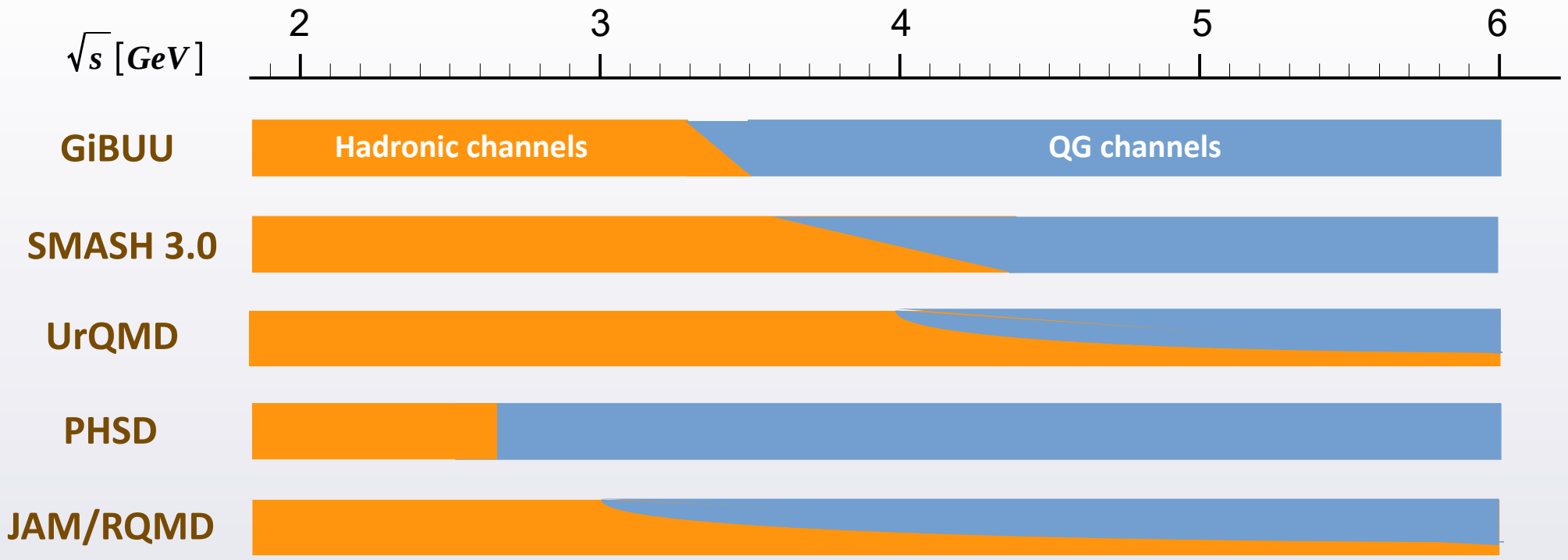
Hadron vs quark sectors

- Due to shortening of $\lambda_{\text{de Broglie}}$ with beam E, with increasing energy we move from the hadron sector to that dominated by quarks and gluons (QG). Transport models aim to describe the QG sector too.

  Some models propose the insertion of the QGP phase as the initial stage of collision, by transitioning to the **hydronamical description**, after which the **hadronization** occurs (QG fluid → hadron gas) .

  In addition, most of models also at the hadronic stage switch on the QG degrees of freedom.
  It is possible if in the individual hadron-hadron collision the $\sqrt{s}$ crosses the threshold value.

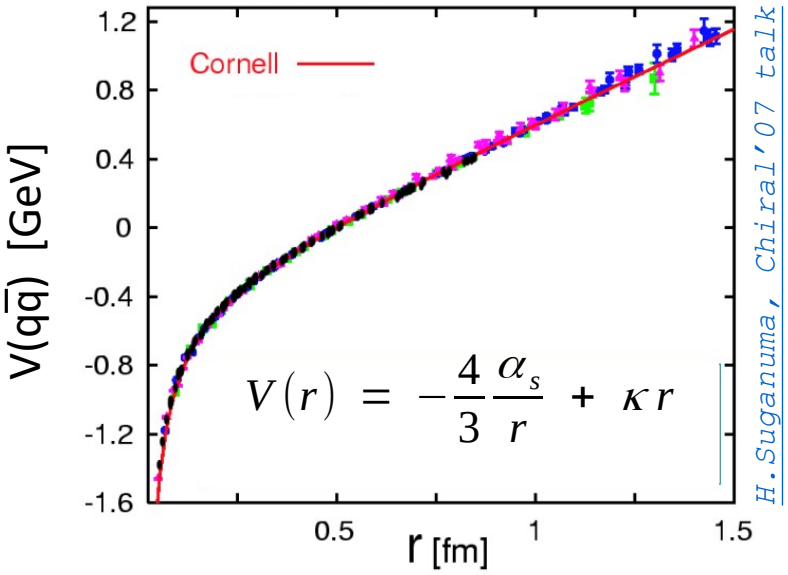
- For the baryon-baryon (BB) channels:

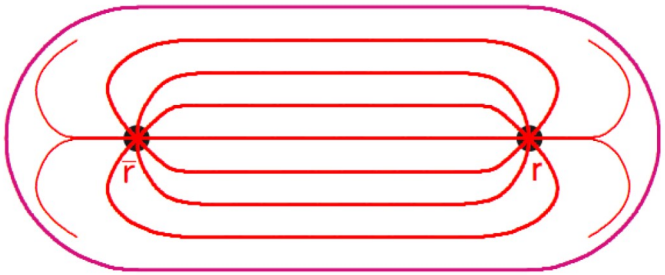


- Description of QG channels: so-called ”strings”. Between 2 quarks a segment is created (”string”) with homogeneous energy distribution. This string is divided into portions, which generate hadrons.

Soft QCD region ("strings")

- The **QCD potential**:
  attractive strong interaction of $q\bar{q}$.





T.Sjöstrand, Pythia talk

Lines of QCD field are compressed to "tubes" (unlike lines of EM field).

Region between two quarks contains energy with (approximately) flat density distribution.

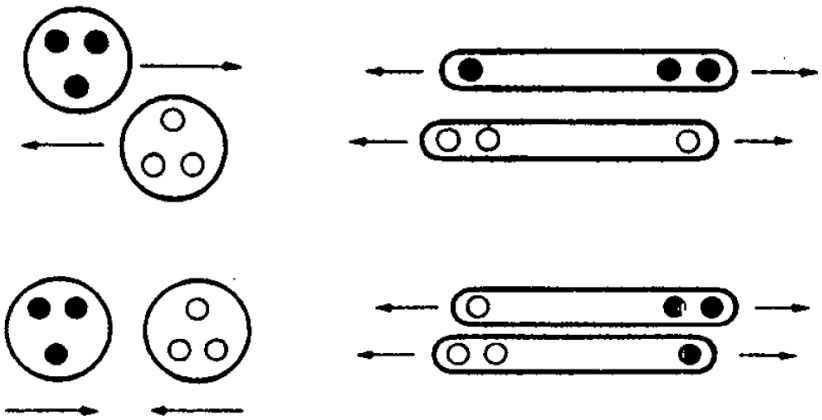
A model object – thin line connecting quarks with linear *E* distribution – is called "**string**".

- In HI Coll. modelling, strings appear in the context of each hadron-hadron collision, if $\sqrt{s}$ exceeds some threshold (a few GeV). But where and how?

▷ Always between selected quarks. Two options:

⊙ **Momentum-exchange**:
two quarks collide and exchange momenta.
It creates elongation in each hadron (thus, a string).

⊙ **Color-exchange**:
1 quark from h_1 feels 2 quarks from h_2 (1st string).
The remainder interacts likewise (2nd string).



H.Schmidt, J.Schukraft,
J.Phys.G 19, 1705 (1993)

- **LUND String Model**

[B. Andersson et al, Phys. Rep. 97, 31 (1983)]

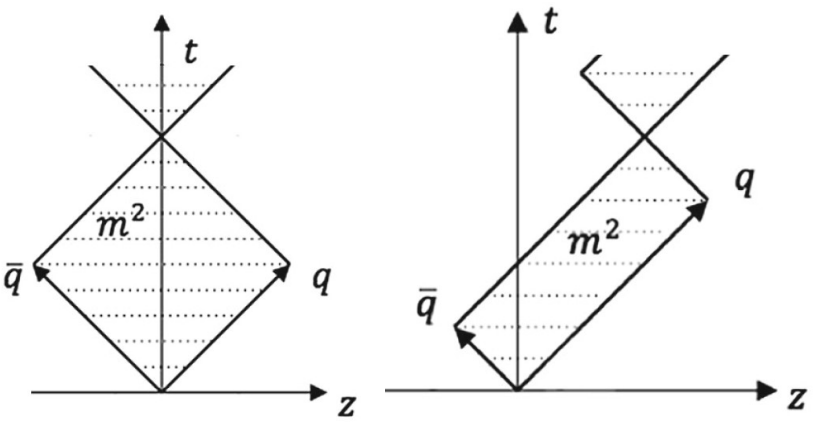
Consider a pair of q, $\bar{q}$ quarks, for simplicity – massless ones and moving only in Z dimension.

How do they move? (= How the string evolves?)

Hamiltonian:

$$H = |p_1| + |p_2| + \kappa|x_1 - x_2|$$

CM Frame **Boosted Frame (Lab)**

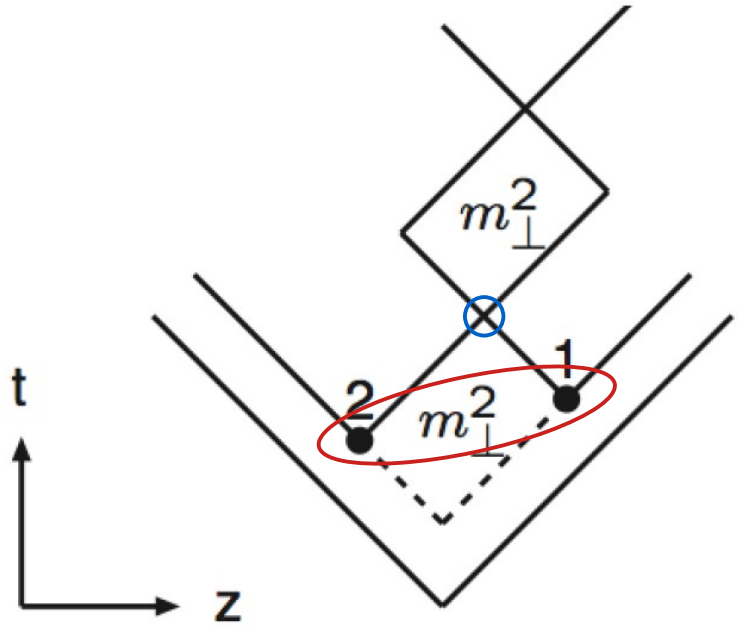


- At some moment it becomes energetically favourable to break the string ("**string fragmentation**") , take energy from a strip around breakpoint, and produce from it the new $q\bar{q}$ pair.

A sketch shows two such **pair production vertices (1, 2)**. But partners move away.

However, right quark from vertex "2" gets bound with left quark from vertex "1" . A new hadron is born (**hadronization**). (Time duration of hadron formation process is debated).

Models assume the **first crossing** to be the creation point of a new hadron.

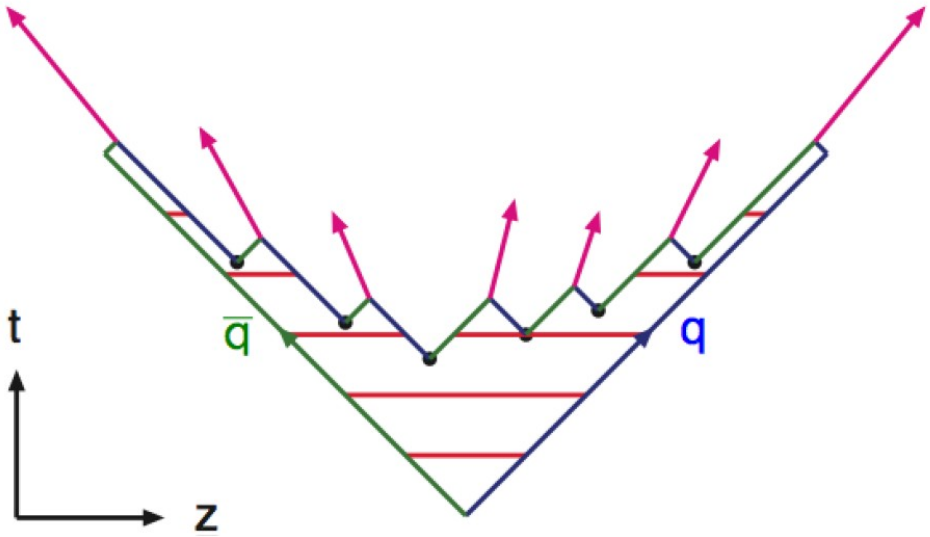


T.Sjöstrand, Pythia talk

Soft QCD region ("strings")

- Overall picture: many hadrons created from fragmentation of string

T.Sjöstrand, Pythia talk



- Quantum-mechanically the pair creation is considered as tunneling through the hole after broken string. The concept is also generalized to massive quarks. A **probability for each quark** is derived:

$$P_q \sim \exp\left(-\frac{\pi m_{\perp q}^2}{\kappa}\right) = \exp\left(-\frac{\pi p_{\perp q}^2}{\kappa}\right) \exp\left(-\frac{\pi m_q^2}{\kappa}\right)$$

  - ▷ Production of quark flavours is in proportion: $u\bar{u} : s\bar{s} : c\bar{c} \approx 1 : 0.3 : 10^{-11}$
  - ▷ P_T spectrum is Gaussian for each quark, and the same for each quark flavour.

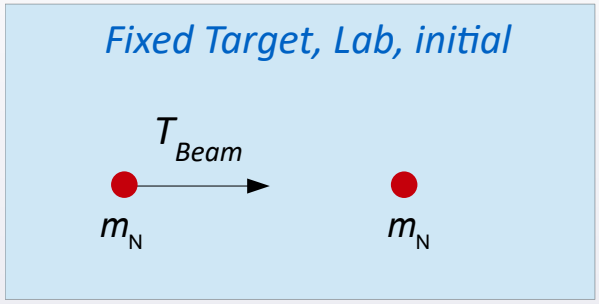
- During formation of a new hadron from a pair of quarks, the hadron's p_T is a sum of constituents.
- A popular implementation of the model is made by **PYTHIA** generator for elementary collisions.
  [Homepage] [Interactive Manual] [Tutorial] [Manual Writeup] .

PYTHIA is called by GiBUU, SMASH and JAM2/RQMD models.
UrQMD and PHSD have their own string fragmentation approaches.

- Characteristics of main properties of some selected (”currently on market”) transport models.

Property	GiBUU	IQMD	UrQMD	JAM2/RQMD	SMASH	PHSD
Relativistic Potential	optionally	n,p : no K : yes	yes	yes	yes	yes
Potential = Mean field	yes	n,p : no K : yes	no	optionally	yes	yes
Potential = sum of nucleon pots.	no	yes	yes	optionally	no	no
Electromagnetic potential	yes	yes	yes for baryons no for π	optionally	optionally	no
Momentum-dependent potential	yes	yes	no	yes	no	yes
Creation of LCP (clusters)	at end of simulation	at end of simulation	no	at end of simulation	at end of simulation	no
modifications of hadron mass in the medium	yes	yes	no	baryons: yes K, π: no	no	yes
Soft QCD region described by ”strings”	Pythia	no	Own approach	Pythia	Pythia	Own approach

- For fixed-target collisions, in order to heat up the system (in CM), we provide T_{Beam} energy (in Lab). Let's consider the nucleon-nucleon (NN) collision at the fixed target. How to translate $T_{\text{Beam}} \rightarrow \sqrt{s}$?



In **Lab**:

$$s \equiv \left(2 m_N + T_{Beam}\right)^2 - \vec{p}^{\,2}_{Beam}$$

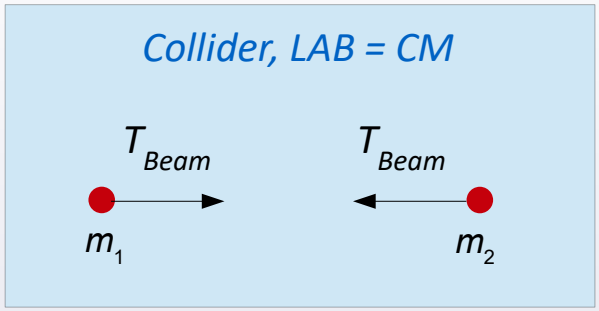
. . .

$$\sqrt{s} = \sqrt{2 \cdot m_N \left(2 m_N + T_{Beam}\right)}$$

$\sqrt{s}$ rises as $\sqrt{T_{Beam}}$

E.g. for T_{Beam} = 1 GeV, $\sqrt{s}$ = 2.3 GeV

- $\sqrt{s}$ is commonly used in the collisions of two opposing beams. Why so?



Momenta sum up to $\vec{0}$. We obtain:

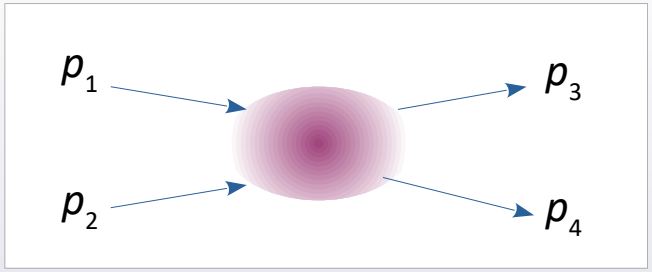
$$\sqrt{s} = \sum_i E_{i,CM} = 2 \cdot \left(m_N + T_{Beam}\right)$$

Nb. in the collider mode it's easier to inject much energy to the system of colliding nuclei:

$$\sqrt{s} \sim T_{Beam}$$

What else should we know? Mandelstam variables

- Consider process with initially 2 particles with 4-momenta p_1, p_2 and finally also 2, with 4-momenta p_3, p_4.



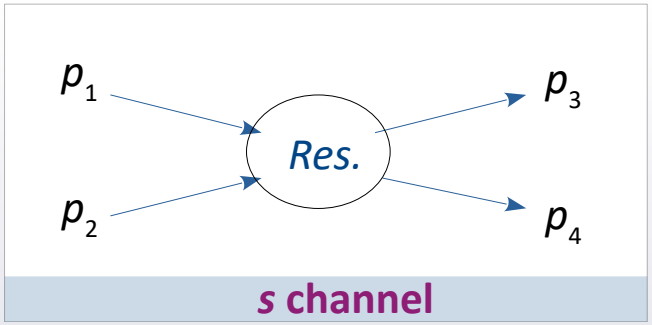
Energy + momentum conservation ≡ 4-momentum conservation.

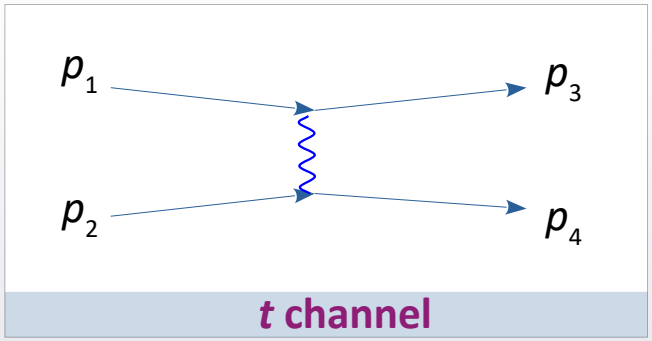
$$p_1 + p_2 = p_3 + p_4$$

- So it's worth introducing the "**available energy**" :

$$s \equiv (p_1 + p_2)^2 = (p_3 + p_4)^2$$

If a process is the fusion of particles into a resonance, then its mass $m = \sqrt{s}$. We say a reaction is in the "**s channel**".





- But the 4-momentum conservation law can be rearranged:

$$p_1 - p_3 = p_4 - p_2$$

and it's worth introducing here "**4-momentum transfer**":

$$t \equiv (p_1 - p_3)^2 = (p_4 - p_2)^2$$

If a process involves an exchange of a virtual particle, then its squared 4-momentum equals to *t*. A reaction is "**in *t* channel**"

- Example: Process $\pi^- \Lambda \to K^- N$ is a "t-channel reaction".
  Process $\pi^- \Lambda \to \Sigma^{*-} \to K^- N$ is a "s-channel reaction".

Interlude: processing text file with formatted data

- **Simple tools in Bash for previewing text data:**

```
cat [file]                              : To display file contents
head  -[No. of lines]  [file]           : To display first lines of file
tail  -[No. of lines]  [file]           : To display last  lines of file
less [file]                             : File reader  (handy hotkeys:  g G / q )
wc -l [file]                            : To count the number of lines

cat mydata.txt | wc -l                  : The | (pipe) symbol redirects an output A → input B
grep [phrase] [file]                    : To search for a (line with) phrase in a file
```

- **Examples:**

```
less /etc/os-release
grep PRETTY_NAME /etc/os-release

ln -s  ~kpiasecki/soft/test/data.txt  .
head -100 data.txt | wc -l
grep "# event"  data.txt  |  grep out  |  wc -l
```

- **Comment:**

Shells like Bash offer many text-based tools that e.g. sort, format, replace phrases etc.
They allow for manipulation using variables, conditional instructions,
loops – including loops over files and directories. One can write "scripts" (codes) and execute them.

Many tutorials on Bash and extensive forums are available on the web.
You can find an interesting cheat-sheet here.
The introduction to Bash on the pages of UW Faculty of Physics is e.g. here.

- **Running long processes on Linux:**

Assume you want to run `MySimuApp` , placed in your directory. You suspect that it would run a long time.

Instead of: `./MySimuApp`

type in this way: `nohup nice ./MySimuApp 1>myLogFile.out 2>&1 &`

- ▷ `nice ...` is the priority manager. ***Important*** that all the users run longer codes through `nice`: the machine resources will be distributed in a just manner.
- ▷ `1>myLogFile.out` will redirect any message from being printed on the screen → to your Log file
  `2>&1` will redirect any error message to that file.
- ▷ `... &` will allow you to continue using your terminal during the code's running
- ▷ `nohup ...` will completely detach your process from the terminal.
  Without this, if your terminal is killed, also your process stops immediately.

- **[Linux] How to check if your process runs? How to stop it?**

- ❍ for dynamic list of running processes: `top` or `htop`
- ❍ for listing of these process: `ps -u yourLogin --forest`
- ❍ to fish the process No. of your application: `ps -u yourLogin --forest | grep MySimuApp`
- ❍ to kill the process identified by given PID (No.) : `kill -9 PID`
- ❍ to kill the graphical window : `xkill` ← then mouse-click on that window

Important: if you think your simulation goes wrong, *kill it* before rerunning.

- **awk :** a simple, yet powerful tool to process text files.
  - ▻ It parses a file **line by line**.
  - ▻ Commands via: **one-liner** or **script-like**.
  - ▻ You can define **variables** and perform **arithmetics + logics** on them. Also, awk has **special variables**.
  - ▻ You can do **if**'s , **loops** (while/for) .

- General structure of awk code: ***condition { action }***

  Actions can be done at the beginning, BEGIN { *action* }
  per every line, { *action* }
  per every line that passes a filter *filter* { *action* }
  or at the end of processed file: END { *action* }

  Filter without action will print the whole line: *filter*

- awk as **one-liner** command : `awk 'filter {command}' filename.txt`

  awk invoking **script** with commands: `awk -f mycode.awk filename.txt`

- Nomenclature: **each line = "record"**. Within line, **each word = "field"**

  Some **special variables** that awk offers per each line: **NR** = current line number **NF** = No. of fields in current line
  $1 , $2 , … = contents of 1st , 2nd , …th field

- Many tutorials on the web, e.g.: ▻ by AGH [**1**, **2**, **3**].
  ▻ by Baeldung [**here**]
  ▻ by Bruce Barnett [**here**].

- **awk (cont).** Let's look at the exemplary text file, `data.txt` :

```
#!OSCAR2013 particle_lists t x y z mass p0 px py pz pdg ID charge
# Units: fm fm fm fm GeV GeV GeV GeV GeV none none e
# SMASH-2.1.1
# event 0 out 123
40 0.687568 1.38769 23.1843 0.938 1.21816871 0.0609451595 -0.164364771 0.757205986 2112 0 0
40 19.5657 -1.19493 5.98978 0.938 1.15461879 0.627001274 -0.0742496884 0.233788244 2112 1 0
...
```

and try these **one-liners**:

`awk 'NF==12 && $10==2112' data.txt` : Take lines with 12 words and neutrons. Print lines.

`awk 'NR>4 && NR<8 {print $5}' data.txt` : Take lines 5-7. Print 5th field only (masses)

`awk 'NF == 12 && $10 == 211' data.txt > pions.dat` : Take π^+ lines. Save them in new file.

- You can use **variables** right away (without declaration). They are automatically initialized to empty string / zero. Let's try it within our first **awk script**. Type e.g.: `nano mystat.awk`

```
NF == 12 {
  NumParticles++
  TotPx += $7
}
END {
  print NumParticles" "TotPx
}
```

… and run it: `awk -f mystat.awk data.txt`

- Now, if you insert into the 1st line of script: `#!/usr/bin/awk -f`
and allow your file to be runnable: `$ chmod 755 mystat.awk`
you can run it directly from prompt: `$ ./mystat.awk data.txt`

- **If conditional statements**. You can use the C++ syntax.

  Imagine we want to select π^+, calculate their $\langle pz \rangle$ and print just this number.
  Let's try it in the file `avgpz.awk` :

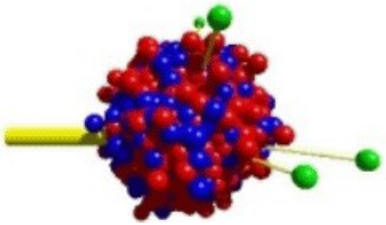
```
NF == 12 {
  if (NF == 12 && $10 == 211) {
    npi++
    pztot += $9
  }
}
END {
  avg = pztot/npi
  print avg
}
```

  Now, let's run it: `awk -f avgpz.awk data.txt`

- If you really have to, you can do the same in the one-liner:

```
awk 'NF == 12 { if ($10 == 211) { npi++ ; pztot += $9 } } END { avg = pztot/npi;
print avg }' data.txt
```

  … btw. awk offers much more, including arrays, for loops, functions, analytic math functions, ...
  It can also be embedded in a bash script.



GiBUU
The Giessen Boltzmann-Uehling-Uhlenbeck Project

Home page:		gibuu.hepforge.org
Download	Download instructions:	gibuu.hepforge.org/trac/wiki/download
	Prerequisites:	gibuu.hepforge.org/trac/wiki/tools
	Download:	gibuu.hepforge.org/downloads
Installation	Compilation instructions:	gibuu.hepforge.org/trac/wiki/compiling
How to run:		gibuu.hepforge.org/trac/wiki/running
Help	Tutorial:	gibuu.hepforge.org/trac/wiki/tutorial
	Presentations:	gibuu.hepforge.org/trac/wiki/Talks
	Animations:	gibuu.hepforge.org/trac/wiki/MovieMain
Table of PID codes:		gibuu.hepforge.org/trac/wiki/ParticleIDs

Main papers:

O. Buss *et al.* ”Transport-theoretical description of Nuclear Reactions”, Physics Reports 512, 1 (2012), arXiv: 1106.1344

- **On the NPD FUW's training computer:**

```
mkdir gibuu ; cd gibuu
cp -r /home/kpiasecki/soft/gibuu/ctnp/*   .
```

Let's inspect the input file (c.f. next page), e.g:

```
less 001_AlAl2.0_test.job
```

Rich collection of input options (*please consider them thoughtfully*) **:**

http://gibuu.hepforge.org/Documentation/

Rich collection of demonstrative input files in this folder:

```
/home/kpiasecki/soft/gibuu/testRun/jobCards/
```

Let's run the simulation (here: in the background . Log goes into file: `caca_1.91_test.log` **)**

```
nohup ./GiBUU.x <001_AlAl2.0_test.job 1>alal_2.0_test.log 2>&1 &
```

Many output files. Among them – file with events containing particles (see next page):

```
less EventOutput.Real.oscar
```

- **Input file** `001_AlAl2.0_test.job` :

```
! file: ./inputOutput/input.f90
&input
        eventtype             =      1
        numEnsembles          =    100
        num_runs_sameEnergy   =     10
        numTimeSteps          =   160
        time_max              =     40.0
        printParticleVectorTime = T
        timeForOutput         = 0.
        timeSequence          = 20.
        path_To_Input    = '{...}/buuinput'
        version = 2023
/
{....}
```

```
! file: ./init/initHeavyIon.f90
&heavyIon
      impact_parameter    = -6.
      impact_profile      = 0
      distance            = 10.
      coulomb             = F
      ekin_lab_Target     = 0.00
      ekin_lab_Projectile = 2.0
      adjustGridFlag      = T
      cmsFlag             = T
/
! file: ./density/nucleus.f90
&projectile
       projectile_Z= 13, projectile_A= 27
/
! file: ./density/nucleus.f90
&target
       target_Z= 13, target_A= 27
/
{....}
```

- **Output file** `EventOutput.Real.oscar` :

Table of PID codes

```
#!OSCAR2013 particle_lists t x y z mass p0 px py pz pdg ID
# Units: fm fm fm fm GeV GeV GeV GeV GeV none none
# File generated by GiBUU (https://gibuu.hepforge.org)
# event     1 out   80
  0.000000E+0   7.183791E+0   3.961299E+0   2.780585E+1   9.227111E-1  1.225751E+0   2.703973E-2 -9.247542E-2   8.011161E-1  2212   100000
  0.000000E+0   5.118310E+0 -7.210583E-1   2.525251E+1   7.657091E-1  1.101021E+0 -5.729713E-3 -1.723748E-2   7.909611E-1  2112   100001
  0.000000E+0   2.584306E+0 -1.719659E-1   2.539883E+1   7.726719E-1  1.088114E+0 -1.631816E-1 -5.991384E-2   7.461942E-1  2112   100002
  0.000000E+0   2.133399E+0   3.397612E-1   2.692126E+1   7.944067E-1  1.045015E+0 -2.850862E-2 -9.578418E-2   6.717517E-1  2112   100003
  0.000000E+0   6.169842E+0 -1.890743E+0   2.453204E+1   8.264268E-1  1.030645E+0 -6.987948E-2   1.303097E-1   5.971482E-1  2112   100004
....
  0.000000E+0 -7.235081E+0   1.162180E+0 -2.613539E+1   7.677036E-1  1.057389E+0   3.919098E-2   1.860452E-1 -7.017274E-1  2212   103239
# event     1 end     0
# event     2 out   80
  0.000000E+0   3.456383E+0   6.567166E-1   2.607133E+1   7.648380E-1  8.782891E-1 -6.281280E-3   4.809546E-2   4.290245E-1  2212   100040
  0.000000E+0   5.786570E+0   1.426497E-1   2.657666E+1   7.663178E-1  1.130114E+0 -1.239606E-1   9.851192E-2   8.153864E-1  2112   100041
  0.000000E+0   6.166525E+0 -3.298759E+0   2.666215E+1   8.635919E-1  1.227330E+0   9.365502E-2 -4.166068E-2   8.660087E-1  2112   100042
....
```

Simulating Many Accelerated Strongly-interacting Hadrons

smash

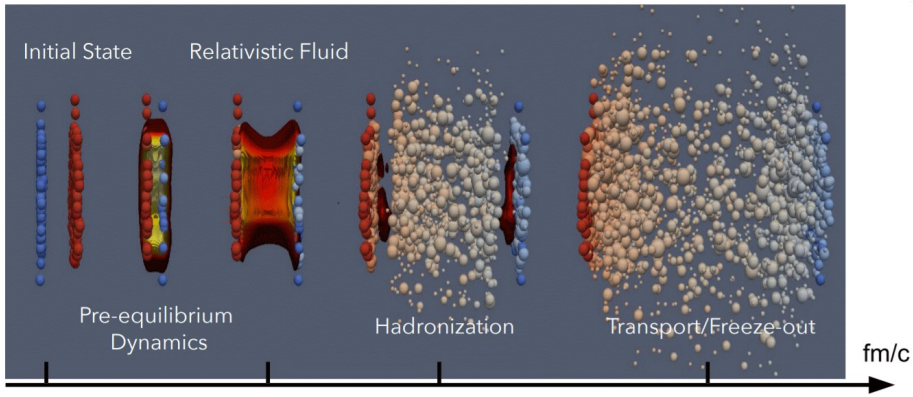
A relativistic hadronic transport approach

- **Home page:** theory.gsi.de/~smash/userguide/current
  smash-transport.github.io
- **Download** github.com/smash-transport/smash
- **Installation** theory.gsi.de/~smash/userguide/current/md_README.html
- **How to run:** theory.gsi.de/~smash/userguide/current/page_smash_invocation.html
- **Help**
  Tutorial: theory.gsi.de/~smash/userguide/current/
  Presentations: D. Oliinychenko (2020) , A. Schäfer (2019)
  Animations: smash-transport.github.io/img/movie.mp4
- **Table of PDG codes**: github.com/smash-transport/smash/blob/master/input/particles.txt
- **Elementary cross sections:** theory.gsi.de/~smash/analysis_suite/SMASH-3.0rc/cross_sections/

- **Main papers**:

J. Weil *et al.* "Particle production and equilibrium properties within a new hadron transport approach for heavy-ion collisions" Physical Review C 94, 054905 (2016), arXiv: 1106.1344

H. Petersen *et al.* "SMASH – A new hadronic transport approach" Nuclear Physics A 982, 399 (2019), arXiv: 1808.06832



- **On the NPD FUW's training computer:**

```
mkdir smash ; cd smash
cp -r /home/kpiasecki/soft/smash/ctnp/*   .
```

Let's inspect the input file, e.g. :

```
less config.yaml
```

Rich collection of input options (*please consider them thoughtfully*) **:**

https://theory.gsi.de/~smash/userguide/current/doxypage_input.html

Let's run the simulation (here: in the background . Log goes into file: `myrun.log` **)**

```
nohup nice ./smash -i config.yaml -o outdir/ 1>myrun.log 2>&1 &
```

File with events containing particles can be found in folder `outdir` **:**

```
less outdir/particle_lists.oscar
```

SMASH Model (BUU family)

- **Input file** `config.yaml` :

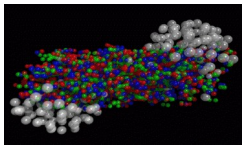
```
{....}
General:
    Modus:           Collider
    Time_Step_Mode:  Fixed
    Delta_Time:      0.1
    End_Time:        40.0
    Randomseed:      -1
    Nevents:         3
    Ensembles:         10

Output:
    Output_Interval: 20.0
    Particles:
        Format:          ["Oscar2013"]
        Extended:        False
        Only_Final:      "Yes"
```

```
Modi:
    Collider:
        Projectile:
            Particles: {2212: 28, 2112: 30} #58Ni
        Target:
            Particles: {2212: 28, 2112: 30} #58Ni
        Impact:
            Sample: "quadratic"
            Range: [0.0, 10.0]

        E_Kin: 1.50
        Calculation_Frame: "center of mass"
        Fermi_Motion: "on"
        Collisions_Within_Nucleus: True

Potentials:
    Skyrme: ...
```

- **Output file** `particle_lists.oscar` :

Table of PID codes

```
#!OSCAR2013 particle_lists t x y z mass p0 px py pz pdg ID charge
# Units: fm fm fm fm GeV GeV GeV GeV GeV none none e
# SMASH-1.7
# event 1 in 394
-2.22563 -3.64001 -1.32594 -3.67708 0.938 2.236845 -0.03016814  0.00476330 2.030443 2112 0 0
-2.22563 -5.40173 -3.32581 -2.98890 0.938 2.101389  0.05289568 -0.05137847 1.878977 2112 1 0
-2.22563 -0.84595 -2.10703 -5.13988 0.938 2.523251  0.22803293  0.04328960 2.330897 2112 2 0
…
# event 1 out 394
    0    -3.67003 -1.32120 -1.65681 0.938 2.236845 -0.03016814  0.00476330 2.030443 2112 0 0
    0    -5.34571 -3.38022 -0.99825 0.938 2.101389  0.05289568 -0.05137847 1.878977 2112 1 0
```



Ultrarelativistic Quantum Molecular Dynamics

- **Home page:** urqmd.org

  Download Home page → contact with the authors of the code.

  Installation + how to run: urqmd.org/en/compiling-and-running-urqmd/

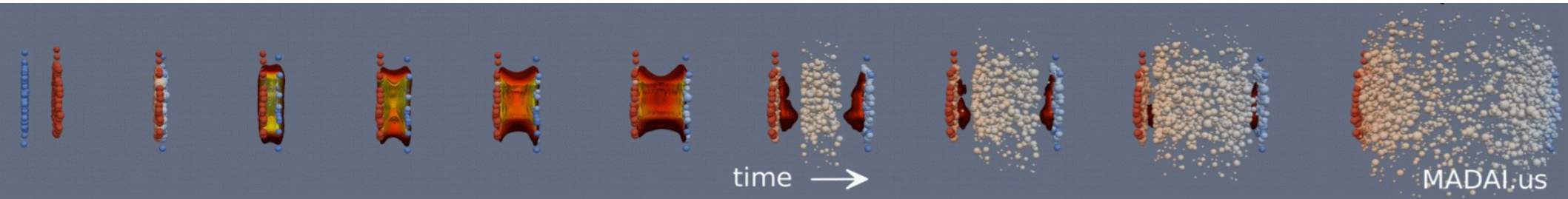
  Help Manual: [Copy]
  Presentation: M. Bleicher (2018)

  Table of PID codes: urqmd.org/en/particle-ids

- **Main papers**:

S.A. Bass *et al.* *"Microscopic models for ultrarelativistic heavy ion collisions"*
Progress in Particle and Nuclear Physics 41, 225 (1998), arXiv: nucl-th/9803035

M. Bleicher *et al.* "Relativistic hadron-hadron collisions in the ultra-relativistic quantum molecular dynamics model"
Journal of Physics G: Nuclear and Particle Physics 25, 1859 (1999), arXiv: hep-ph/9909407



- **On the NPD FUW's training computer:**

```
mkdir urqmd ; cd urqmd
cp -r /home/kpiasecki/soft/urqmd-3.4/ctnp/*   .
```

Let's inspect these input files :

```
less runqmd.bash
less inputfile_NiNi1.5
```

Rich collection of input options is in § 5 of Manual (*please consider them thoughtfully*) **:**

[Manual]

Let's perform the simulation (here: in the background. Log goes into file: `mysim.log` **)**

```
nohup nice ./runqmd.bash 1>mysim.log 2>&1 &
```

We've got a collection of files with events. In between, the file with extension `f14` **with events (and subsequent timesteps), containing particles :**

```
less test.f14
```

- **Input file** `inputfile.NiNi1.5` :

```
pro 58 28
tar 58 28
nev 10
imp -10.
elb 1.5
tim 50 20
cto 18 1

f13
#f14
f15
#f16
f19
f20
{....}
```

- **Input file** `runqmd.bash` :

```
{....}
export ftn09=inputfile_NiNi1.5
export ftn13=test.f13
export ftn14=test.f14
export ftn15=test.f15
export ftn16=test.f16
export ftn19=test.f19
export ftn20=test.f20
{....}
```

- **Output file** `test.f14` :

Table of PID codes

charge

```
UQMD   version:       30400   1000  30400  output_file  14
projectile:  (mass, char)    58  28   target:  (mass, char)    58  28
transformation betas (NN,lab,pro)      0.0000000  0.6665679 -0.6665679
impact_parameter_real/min/max(fm):    6.28  0.00 10.00  total_cross_section(mbarn):     3141.59
equation_of_state:    0  E_lab(GeV/u): 0.1500E+0  sqrt(s)(GeV): 0.2517E+0  p_lab(GeV/u): 0.2250E+0
event#         1 random seed:  1600419089 (auto)   total_time(fm/c):       40 Delta(t)_O(fm/c):       20.000
(....)
pvec: r0         rx         ry          rz          p0          px          py          pz          m      ityp 2i3 ch lcl ncl or
    119      20
     73      47      22       4       7      19       0       0
 0.20000E+2  0.74819E+2  0.36667E+0  0.10799E+2  0.12222E+2 -0.81322E-2  0.91387E-1  0.80353E+0  0.92473E+0   1   1   1   0   0   0
 0.20000E+2  0.34167E+2  0.27792E+0  0.59270E+2  0.11642E+2  0.18126E-1 -0.53284E-1  0.68819E+0  0.93800E+0   1   1   1  79   1   0
 0.20000E+2  0.78149E+0 -0.30699E+2  0.10397E+2  0.13424E+2  0.88550E-1  0.91699E-2  0.97368E+0  0.92038E+0   1   1   1   0   0   0
{....}
        119          40
      83       54      22       7       7      19       0       0
 0.40000E+2  0.74819E+1  0.36667E+0  0.24239E+2  0.12282E+1 -0.81322E-2  0.91318E-1  0.80353E+0  0.92437E+0   1   1   1   0   0   0
```

Parton-Hadron-String Dynamics
Transport approach

- **Home page:** theory.gsi.de/~ebratkov/phsd-project/PHSD/

  Download *[contact with authors]* theory.gsi.de/~ebratkov/phsd-project/PHSD/index4.html

  Installation + how to run: § 1.2 of Manual.

  Help Manual: [Copy]
  Presentations: E. Bratkovskaya, Home page
  Animations: Au+Au @ 10A GeV , Au+Au @ 35A GeV , Pb+Pb @ 158A GeV

  Table of PID codes: Tables 1.1 i 1.2 in Manual

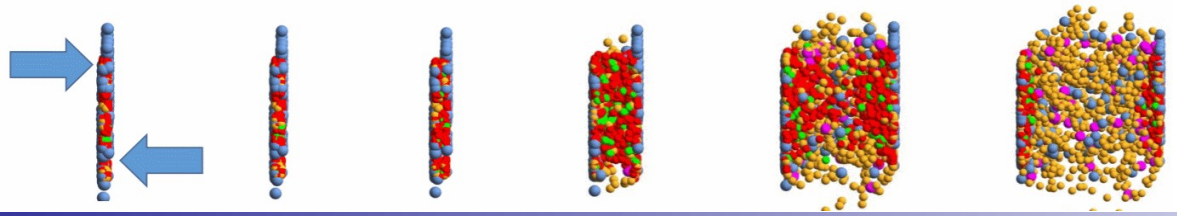
- **Main papers**:

  ⊙ PHSD:

  W. Cassing, E.L. Bratkovskaya "Parton transport and hadronization from the dynamical quasiparticle point of view", Physical Review C 78, 034919 (2018), arXiv: 0808.0022

  ⊙ HSD:

  W. Cassing, E.L. Bratkovskaya "Hadronic and electromagnetic probes of hot and dense nuclear matter" Physics Reports 308, 65 (1999), arXiv: 1808.06832



PHSD Model

- **On the NPD FUW's training computer:**

```
mkdir phsd ; cd phsd
cp -r /home/kpiasecki/soft/phsd/ctnp/*   .
```

Let's inspect the input file, e.g. :

```
less inputPHSD
```

Rich collection of input options is in § 1.4 of Manual (*please consider them thoughtfully*) **:**

[Manual]

Let's run the simulation (here: in the background. Log goes into file: `mysim.log` **)**

```
nohup nice ./phsd 1>mysim.log 2>&1 &
```

We've got a collection of files with events.
In between, the file `phsd.dat` **with subsequent events containing particles :**

```
less phsd.dat
```

- **Input file** `inputPHSD` **:**

```
1,      MASSTA: target mass
1,      MSTAPR: protons in target
58,      MASSPR: projectile mass
28,      MSPRPR: protons in projectile
1.9,    ELAB:   (=4060000. Lab energy per nucleon LHC),=21300 RHIC,=13433049 (5 TeV) = 26120000 (7 TeV)
150,     BMIN:    minimal impact parameter in fm ! no effect for p+A
150,     BMAX:    maximal impact parameter in fm ! no effect for p+A
0.5,     DeltaB: impact parameter step in fm (used only if IBweight_MC=0)
150,     NUM:     optimized number of parallel ensambles ("events")
10,      ISUBS:   number of subsequent runs
{....}
```

- **Output file** `phsd.dat` **:**

Table of PID codes

```
125        1     1  0.6427918E+01     1
  27 -0.204536E+00  0.418460E+00  0.317638E+00  0.675407E+00  0.318581E+00  0.446779E+00  0.137598E+00  0.671028E+00
  2212     1 -0.25568587E+00 -0.16134205E+00 -0.12625135E+01  0.16016214E+01        1         51
  2112     0 -0.29324052E+00  0.86835183E-01 -0.75108749E+00  0.12399625E+01        1          0
  2212     1  0.61904673E-01  0.12047570E+00 -0.11011121E+01  0.14528037E+01       -1          0
  2212     1 -0.25516313E+00  0.11847670E+00 -0.90409690E+00  0.13328092E+01       -1          0
  2212     1  0.43485938E-02  0.11743490E-01 -0.73762423E+00  0.11933526E+01       -1          0
  2212     1 -0.69573373E-01 -0.43003834E-02 -0.66405290E+00  0.11513771E+01       -1          0
(....)
   111     0  0.37781116E-01  0.11493234E+00 -0.78976639E-01  0.19979531E+00        2         20
129        1     2  0.64279118E+01     1
  39  0.840226E+00  0.295869E+00 -0.752309E+00  0.163571E+00 -0.783561E+00  0.221714E+00 -0.245801E+00  0.586710E+00
  2212     1 -0.14489351E+00  0.92953674E-01 -0.11530347E+01  0.14963180E+01        1          0
  2212     1 -0.10142475E+00 -0.41412741E-01 -0.11190853E+01  0.14643080E+01        1          0
  2212     1  0.14764933E+00 -0.58835067E-01 -0.90158129E+00  0.13107077E+01        1          0
```

p_x p_y p_z E

charge

JAM2/RQMD Model (QMD technically built on BUU)

- **Naming.** The model was first published in 1999 as **JAM**. It focused on the region of $T_{\mathrm{Beam}} \sim 10A$ GeV, so details of binding energy (EoS) were not so important.
  Since ~ 2019 the EoS was carefully modelled and the code was named **RQMD.RMF**.
  In 2022 the code was ported to C++. The name is "**JAM2**" and "**RQMD.RMF**" is its variant.

- **Home page:** [GitLab] (Jam2) [link] for old Jam1

  Download & Installation [GitLab] (Jam2)

  Help. Short Manual (txt) [GitLab]
  For Jam1: old but explains a lot: [Copy]

  Table of PID codes: [File 1] , [File 2] (retrieved pages) or Appendix A in old Manual.

- **Main papers**:

  ⊙ RQMD.RMF approach:

  Y. Nara, "Sensitivity of the excitation functions of collective flow to relativistic scalar and vector meson interactions in the relativistic quantum molecular dynamics model RQMD.RMF"
  Physical Review C 100, 054902 (2019), arXiv: 1906.03537

  ⊙ JAM (1st paper):

  Y. Nara et al. "Relativistic nuclear collisions at 10A GeV energies from p+Be to Au+Au with the hadronic cascade model"
  Physical Review C 64, 024901 (1999), arXiv: nucl-th/9904059

JAM2/RQMD Model (QMD technically built on BUU)

- **On the NPD FUW's training computer:**

```
mkdir jam2 ; cd jam2
cp -r /home/kpiasecki/soft/jam2/ctnp/*   .
```

Let's inspect the input file, e.g. :

```
less jam.inp
```

Collection of input options (*please consider them thoughtfully*) **:**

[This short Manual] (for Jam1, see § 4.6 of this Manual)

Let's run the simulation (here: in the background . Log goes into file `mysim.log` **)**

```
nohup nice ./runjam.sh 1>mysim.log 2>&1 &
```

We've got the file `phase.dat.gz` **with subsequent events containing particles. Unzip & view:**

```
gzip -d phase.dat.gz
less phase.dat
```

JAM2/RQMD Model (QMD technically built on BUU)

- **Input file** `jam.inp` :

```
Main:numberOfEvents = 10
Random:seed = 3668752

Cascade:TimeStepSize = 0.10
Cascade:TimeStep = 200

MeanField:mode=14
MeanField:potentialType = 3
MeanField:EoS= 32   # =31:NS1, =32:NS2 ...
MeanField:gaussWidth = 2.0

111:mayDecay = off     ! pi0
221:mayDecay = off     ! eta
...
Beams:beamA = 197Au
Beams:beamB = 197Au
Beams:eCM = 4.5

#Beams:beamA = 208Pb
#Beams:beamB = 208Pb
#Beams:eLab = 158

Beams:bmin = 4.6
Beams:bmax = 9.4
```

- **Output file** `phase.dat` :

Table of PID codes

```
#      2000      0.82165      0.67597      1.35698    2
#         1        223         25         21   9.10      36      58      43
1  2112  1   9.3960E-2 -7.4691E-2  4.6286E-3 -9.9425E-2  1.3700E+0 -7.8427E+0  3.5348E+0 -1.6458E+1  3.0200E+1  3.0200E+1
1  2112  1   9.3960E-2 -4.2849E-2  2.0180E-2 -8.9590E-2  1.3145E+0 -6.7592E+0  9.4156E+0 -1.7980E+1  3.0200E+1  3.0200E+1
1  2212  1   9.3960E-2 -1.1528E-2  2.2750E-2 -8.4282E-2  1.2676E+0 -7.5301E+0  5.3435E+0 -1.6708E+1  3.0200E+1  3.0200E+1
1  2112  5   9.3957E-2  3.0711E-2  2.4778E-2 -2.6403E-2  1.0527E+0  4.7449E+0  5.2862E+0 -4.7811E+0  3.0200E+1  3.0200E+1
{....}
1   111  8   1.3498E-1  1.4090E-1 -1.5388E-1 -1.4472E-1  2.8757E-1  2.5947E+0 -5.0680E+0 -8.0321E+0  3.0200E+1  3.0200E+1
#         2        227         78         62  7.12      85      152     124
1  2212  1   9.3960E-1 -2.1659E-2  1.1828E-1 -5.13744E-1 1.0776E+0 -6.4530E+0  4.1473E+0 -9.0628E+0  3.0200E+1  3.0200E+1
1  2212  1   9.3960E-1  4.0508E-2 -1.0861E-1 -7.09055E-1 1.1827E+0  6.2874E-2 -6.4578E-1 -1.1831E+1  3.0200E+1  3.0200E+1
```

mass | p_x | p_y | p_z | E