Ongoing Developments in Hadronics

Tatsumi Koi and Dennis Wright
Geant4 Collaboration Meeting
22 September 2011

Hot Topics in Hadronics

- New neutron models
- Hadronic cross section re-design
- FTF and cascade developments
- Interfaces between hadronic models

General Interaction Data Interface (GIDI) and Geant4 Low Energy Nuclear Data (G4LEND)

- GIDI is a newly developed data format for nuclear data
  - More modern design than current nuclear formats established in the 1960's
- GIDI also provides access routines for the data
- Conversion of the current nuclear data libraries (ENDF, JEF, JENDL,,,) are on going
  - ENDL99.1 and ENDF.VII.0 is already open to public
- LEND is a Geant4 interface to GIDI and an alternative choice of the low energy neutron transportation in Geant4

Update in Neutron High Precision Package

- Many bugs are fixed by CIMAT group
  - See next slides
- G4NDL3.15
  - Migrate to ENDF-VII with processing by NJOY99
  - Several ENDF-VII files can not be converted from several reasons
    - If the data is available in ENDF-VI library, then compensated by it.
  - Total 380 nuclei (including 3 natural abundance) are available
    - Public version limited up to U.
- Thermal Scattering
  - Also migrate to ENDF-VII Thermal Scattering Data
  - 20 scatters (H in H2O, H in Polyethelene,,,,)
  - This model and cross section will be automatically triggered by G4NistMaterials (see “Hadron Cross Section Re-Desigh”)
  - Not only HP but also co-working with LEND

CIEMAT's contribution to the G4NeutronHP development

•Eight different standard evaluated cross section libraries in the ENDF-6 format have been translated to the G4NDL format used by G4NeutronHP:

ENDF-VII.0 (USA) – 385 isotopes, ENDF-VI.8 (USA) – 317 isotopes, JEFF-3.0 (EU) – 373 isotopes, JEFF-3.1 (EU) – 334 isotopes, **JENDL-4.0 (Japan) – 400 isotopes**, JENDL-3.3 (Japan) – 332 isotopes, BROND-2.2 (Russia) – 120 isotopes, CENDL-3.1 (China) – 239 isotopes.

The latest GEANT4 version tested was: **G4NDL-3.14 – 181 isotopes**.

•The libraries will be distributed from the IAEA nuclear data service web and will allow the users to choose which library they want to use in their applications.

•Several bugs in G4NeutronHP have been corrected and new capabilities have been implemented. All of them have been included in the new release. With additional effort, G4NeutronHP could come close to the reliability of MCNPX in neutron transport problems.

•Validation tools and strategies have been developed, capable of making a systematic comparison between Geant4 and MCNPX and all the libraries. These tools have been allowed to debug and improve the Geant4 code. The results of the validations will be made public, to allow the users to compare results between different codes and libraries, isotope by isotope.

E. Mendoza and D. Cano-Ott

GOBIERNO DE ESPAÑA
MINISTERIO DE CIENCIA E INNOVACIÓN
Ciemat
Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas

Hadron Cross Section Re-Design

- Requests from low energy neutron transportation.
  - Enable to calculate material specialized cross sections for G4NistMaterials
    - Passing "Material" to CrossSectionDataSet
  - Enable to use materials those made from excited isomers in simulation
    - Add new attribute to G4Isotope (Cross Section)
    - Add new attribute to G4Nucleus (Model)
- Known problems
  - GPIL calculation does not take account of Isotope wise cross sections
  - Therefore, Low energy neutron cross sections (HP and LEND) should integrate isotope-wise cross section internally and return an element-wise cross section
- Followings slides show our current solution
  - Additional rounds may be required

New Interface of VCrossSetionDataSet

```
virtual
G4bool IsElementApplicable(const G4DynamicParticle*, G4int Z,
                           const G4Material* mat = 0);
virtual
G4bool IsIsoApplicable(const G4DynamicParticle*, G4int Z, G4int A,
                       const G4Element* elm = 0,
                       const G4Material* mat = 0);

//============== GetCrossSection methods=============================

virtual
 G4double GetElementCrossSection(const G4DynamicParticle*, G4int Z,
                       const G4Material* mat = 0);
virtual
 G4double GetIsoCrossSection(const G4DynamicParticle*, G4int Z, G4int A,
                       const G4Isotope* iso = 0,
                       const G4Element* elm = 0,
                       const G4Material* mat = 0);
```

New logic of CrossSectionDataStore::GetCrossSection (const G4DynamicParticle* par, const G4Element* elm, const G4Material* mat)

```
G4int i = nDataSetList-1;
 G4int Z = G4lrint(elm->GetZ());
 G4double xsec = 0.0;
 if (dataSetList[i]->IsElementApplicable(part, Z, mat)) {
  // element wise cross section
  xsec = dataSetList[i]->GetElementCrossSection(part, Z, mat);
 } else {
  // isotope wise cross section
  G4int nIso = elm->GetNumberOfIsotopes();
  G4Isotope* iso = 0;
  if (0 < nIso) {
   // user-defined isotope abundances
   G4IsotopeVector* isoVector = elm->GetIsotopeVector();
   G4double* abundVector = elm->GetRelativeAbundanceVector();
   for (G4int j = 0; j<nIso; ++j) {
    if(abundVector[j] > 0.0) {
     iso = (*isoVector)[j];
     xsec += abundVector[j]*
      GetIsoCrossSection(part, Z, iso->GetN(), iso, elm, mat, i);
    }
   }
  } else {
   // natural isotope abundances
   G4int n0 = nist->GetNistFirstIsotopeN(Z);
   G4int nn = nist->GetNumberOfNistIsotopes(Z);
   for (G4int A = n0; A < n0+nn; ++A) {
    G4double abund = nist->GetIsotopeAbundance(Z, A);
    if(abund > 0.0) {
     xsec += abund*GetIsoCrossSection(part, Z, A, iso, elm, mat, i);
    }
   }
  }
 }
return xsec;
```

- This method is called in a series of

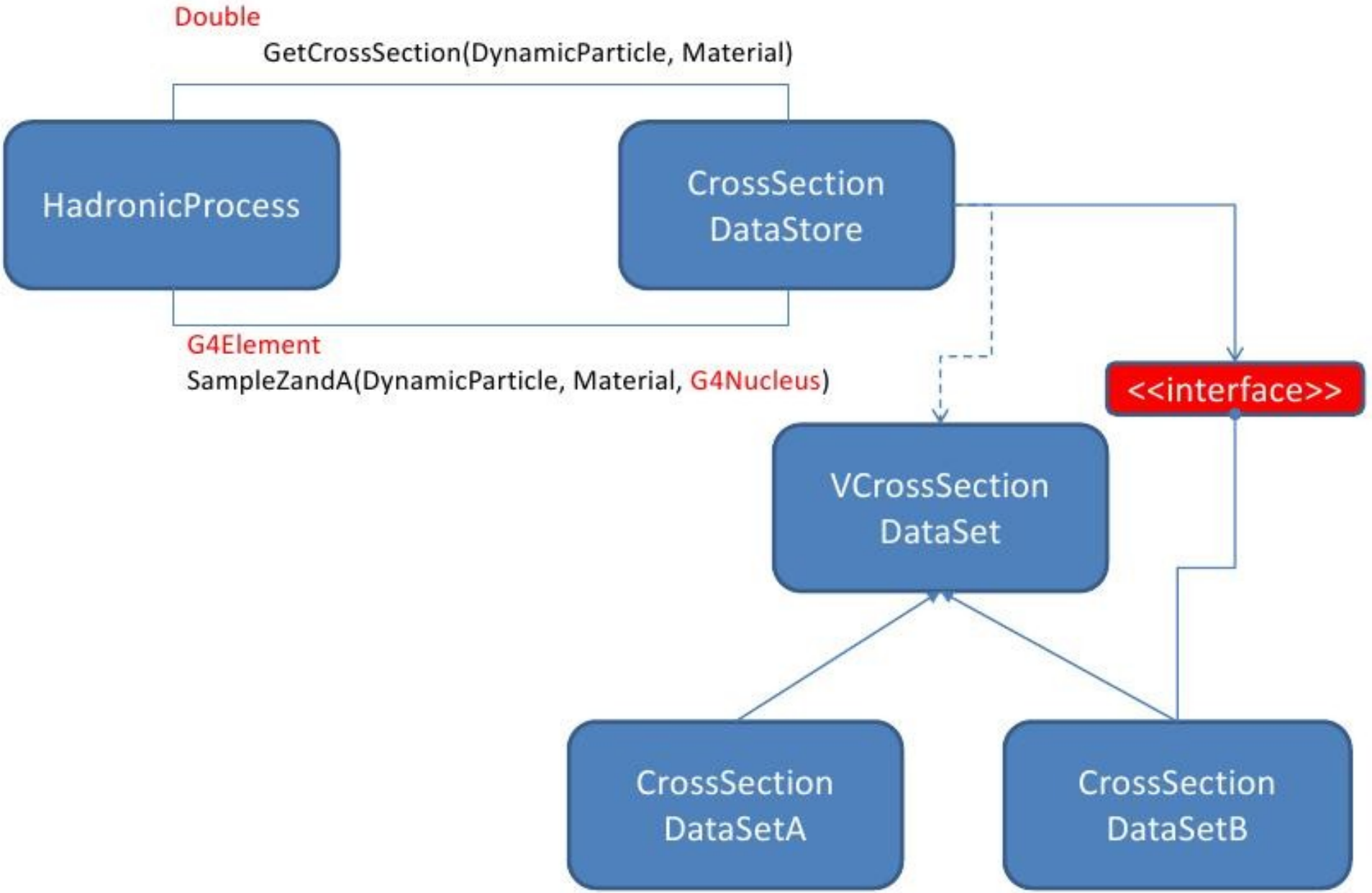
G4HadronicProcess::GetMeanFreePath (const G4Track &aTrack, G4double, G4ForceCondition *)

↓

G4CrossSectionDataStore::GetCrossSection (const G4DynamicParticle* part, const G4Material* mat)

↓

G4CrossSectionDataStore::GetCrossSection (const G4DynamicParticle* part, const G4Element*, const G4Material* mat)


Double
GetCrossSection(DynamicParticle, Material)
HadronicProcess
CrossSection DataStore
G4Element
SampleZandA(DynamicParticle, Material, G4Nucleus)
<<interface>>
VCrossSection DataSet
CrossSection DataSetA
CrossSection DataSetB

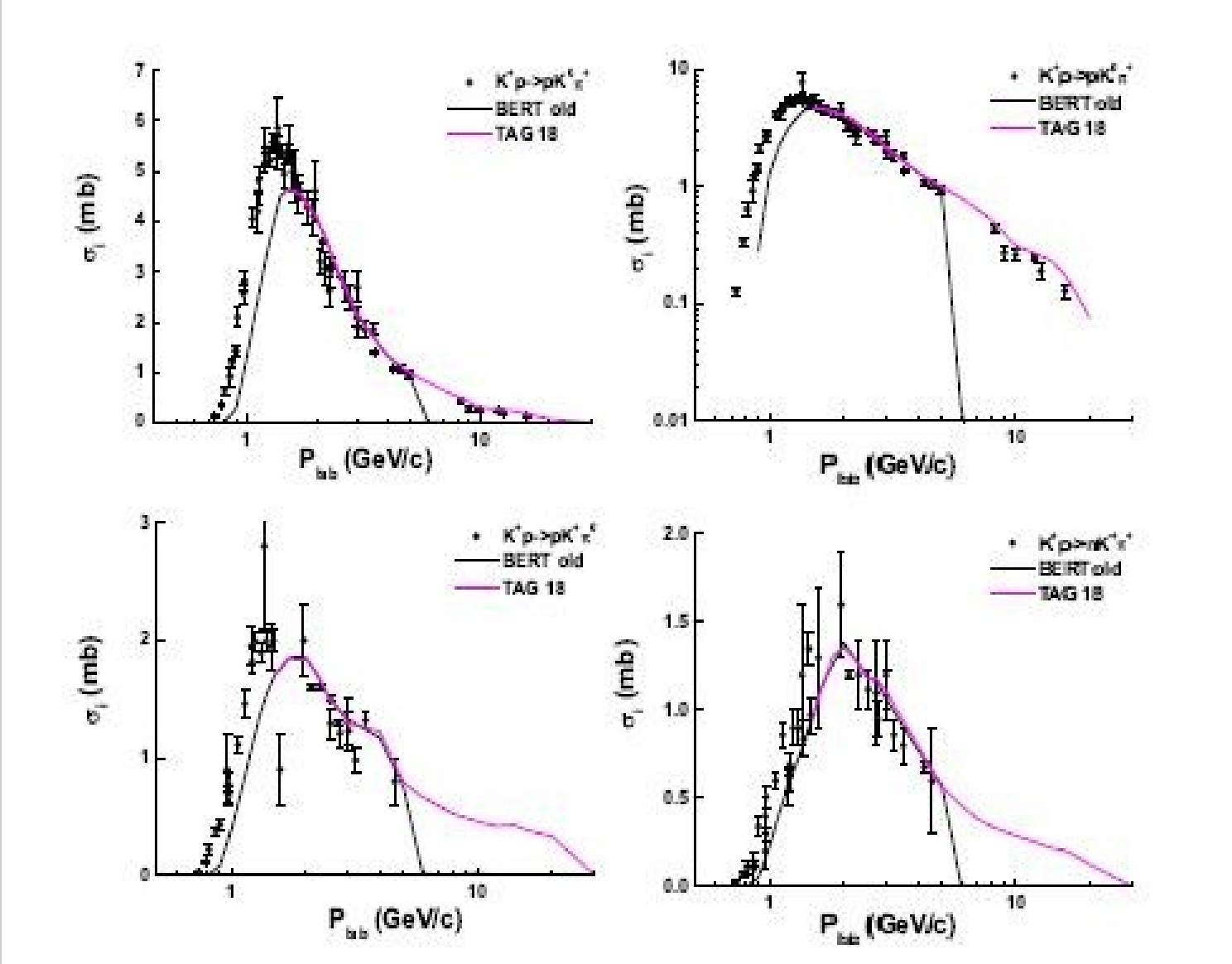
Cascade Model Development (Bertini)

- Increased integration of Bertini code with Geant4 methods and standards
  - Original Bertini code was self-contained
  - Had many of its own routines to do things already provided by Geant4 - led to unnecessary duplication and error, difficulties interfacing with other Geant4 models
  - These rotuines now replaced by standard Geant4 code: G4LorentzVector, G4Fragment, increased use of Geant4 particle types
- Memory fragmentation and churn greatly reduced
  - Memory fragmentation reduced by reusing data buffers
  - Memory churn (newing and deleting) reduced by factor of 10

Cascade Model Development (Bertini)

- Greatly improved E/p conservation (now < 0.1 MeV at 4 GeV, down from ~10 MeV or more)
- Trailing effect added
  - Well-known effect of local density reduction in nuclear medium following an individual scatter within nucleus -> predicts fewer final state nucleons
  - Still optimizing parameters for this effect
- Supports rescattering of high energy fragments from string models
  - High energy scatter on nucleon produces fragments either inside or outside the target nucleus – previously had no way to allow these particles to initiate a cascade
  - Needed for smooth, physical transition from string model (FTF, QGS) to Bertini cascade

Bertini Fix of 3-body Final State Angular Distribution

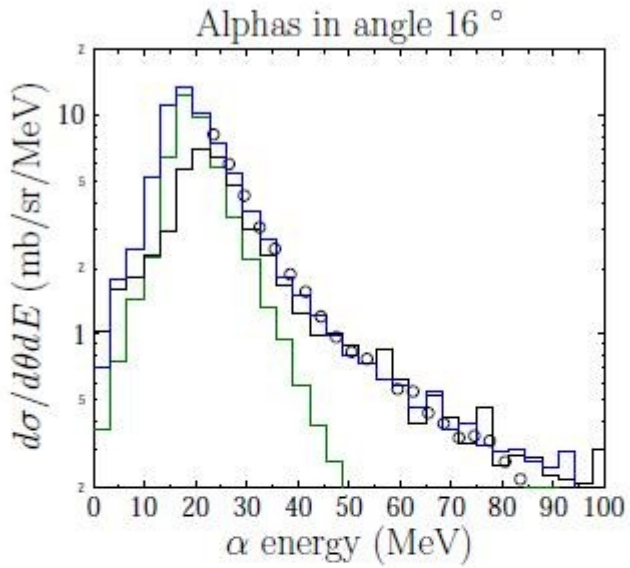
- 

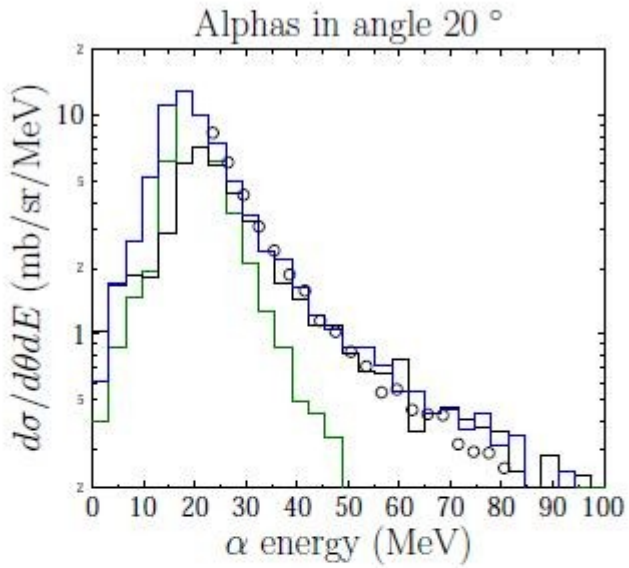
Cascade Model Development (INCL)

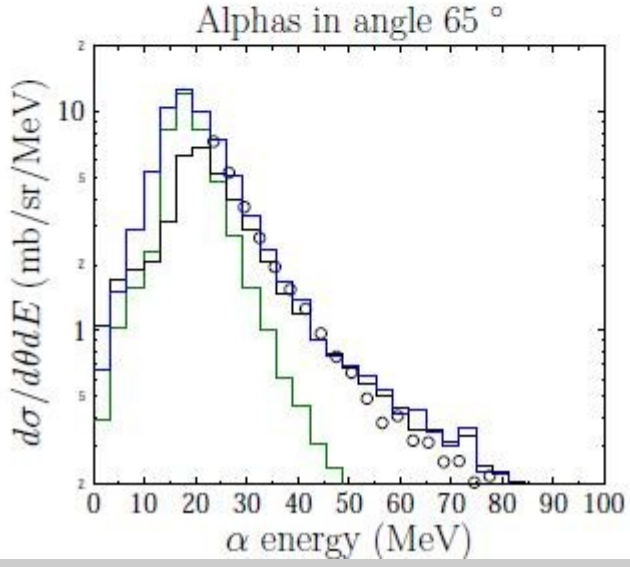
- INCL cascade development (P. Kaitaniemi, A. Boudard) nearly complete
  - data-driven cascade designed for 100 MeV – 3 GeV
- Improved model INCL++ v5.0 now under development
  - cluster formation -> good d, t, α production
  - will eventually replace INCL
- Interface to G4ExcitationHandler developed in order to reduce code reduncdancy
  - G4ExcitationHandler is same code used by G4Precompound
  - Compares very well with ABLA code (native to INCL)

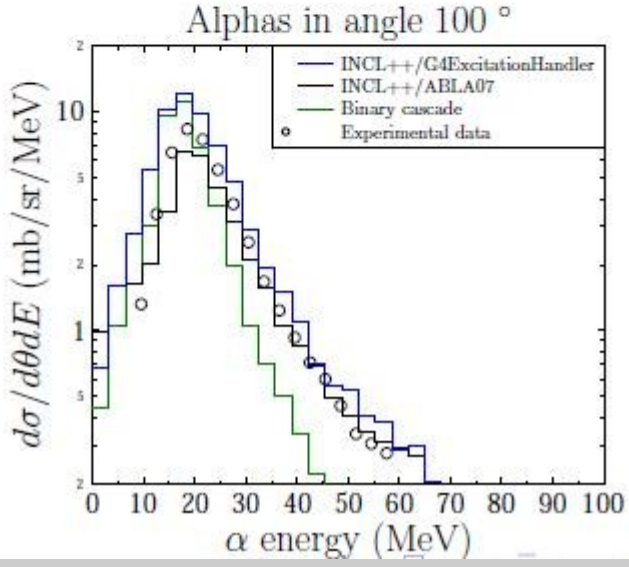
Comparing INCL++, Binary Cascades

$p\ (2.5\ \mathrm{GeV}) + \mathrm{Au} \rightarrow \alpha + X$









FTF Model Development

- Addition and Tuning of Reggeon Cascade
  - describes high energy cascading as a repeated exchange of quarks between hadrons
  - allows better nuclear destruction/de-excitation after the initial high energy interaction
- Added quark exchange for string excitation (like QGS)
- Tuning of low mass strings
- Above three points have allowed FTF to be extended to much lower energies (~3 GeV) for nucleons, pions, hyperons
  - HARP data used for tuning cascade, and low mass strings
- FTF now merges much more smoothly with Bertini

FTF Model Development

- Tuning still required for nucleus-nucleus, hadron -nucleus
- Antibaryon and anti-ion scattering to be available by year's end
  - antibaryon and antinucleon scattering cross sections already implmented using Glauber approach
    - CHIPS cross sections also used
  - final state generation down to zero incident energy is possible due to low mass strings and reggeon cascade
  - cascade models not needed as a low energy back-end
  - prototype available for 0 to 1 TeV/c/nucleon
- Validation will be needed
  - data from Alice?

Interfaces Between Hadronic Models

- Geant4 approach to hadronic models:
  - select a sufficient number of models to cover full energy range (typically 2 or 3)
  - assign them to a hadronic process
  - Smoothly transition in energy from one model to another by throwing a random number
- Not the best approach
  - No physical basis, range of energy blending treated as a parameter
  - Can lead to discontinuities in response
- Interfaces between precompound and cascade, string models and cascade may help
  - Also helps to avoid model duplication

Interfaces Between Hadronic Models

- G4Precompound to Bertini
  - Validation shows that G4Precompound is better than Bertini at ~200 MeV and below
  - Replace Bertini precompound and de-excitation with G4Precompound using new interface
  - Validation of this interface is still underway: successful at ~60 MeV and below, not successful above that -> more work required
- G4ExcitationHandler to INCL
  - Similar to G4Precompound-Bertini interface
  - INCL does well at precompound stage, so interface to G4Precompound is not needed
  - Instead interface directly to de-excitation code that G4Precompound also uses (excitation handler)
  - Early tests show good agreement with data

Interfaces Between Hadronic Models

- FTF to Bertini
  - modeled on Propagate interface developed for QGS-Binary re-scattering
  - takes fragments produced by FTF parton model and sends them to Bertini
    - each fragment can initiate a cascade as if it were an original incident particle
    - rescatter method developed in Bertini to handle this
  - still in development – problems include:
    - energy momentum balance
    - excitation energy of residual nucleus is too high to de-excite by precompound model
    - FTF can create almost any hadron – Bertini can so far handle only some of them