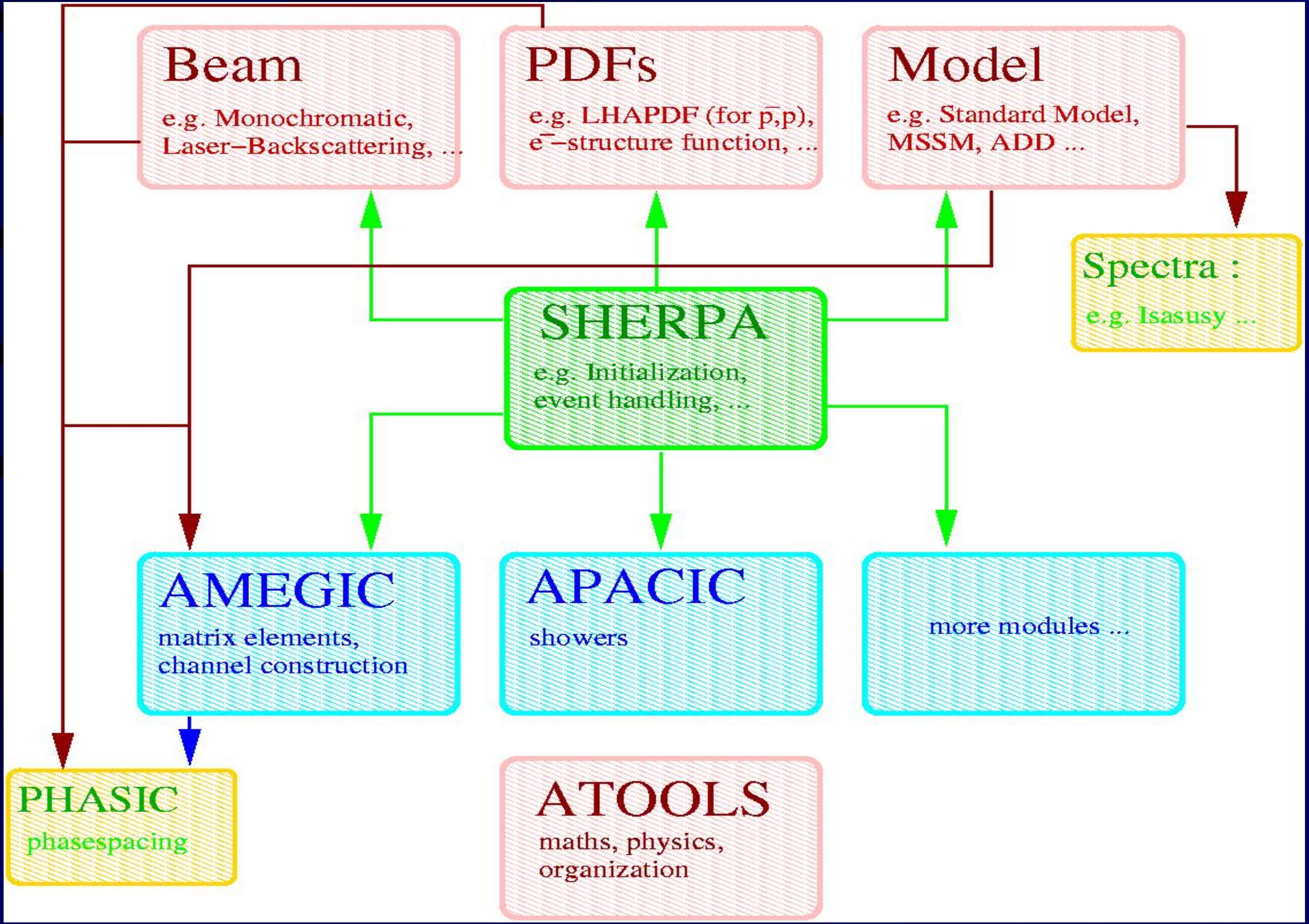
SHERPA

Simulation for High Energy Reaction of PArticles

Aim of the SHERPA project

- A new full event generator, including
  - Matrix elements (⇒ AMEGIC++ & PHASIC)
  - Parton showers (⇒ APACIC++)
  - Hadronization (to be included soon)
  - Hadron decays
  - Underlying events
  - Pile-up events

Structure of the package



Beam
e.g. Monochromatic,
Laser−Backscattering, ...

PDFs
e.g. LHAPDF (for $\bar{p}$,p),
e^-−structure function, ...

Model
e.g. Standard Model,
MSSM, ADD ...

Spectra :
e.g. Isasusy ...

SHERPA
e.g. Initialization,
event handling, ...

AMEGIC
matrix elements,
channel construction

APACIC
showers

more modules ...

PHASIC
phasespacing

ATOOLS
maths, physics,
organization

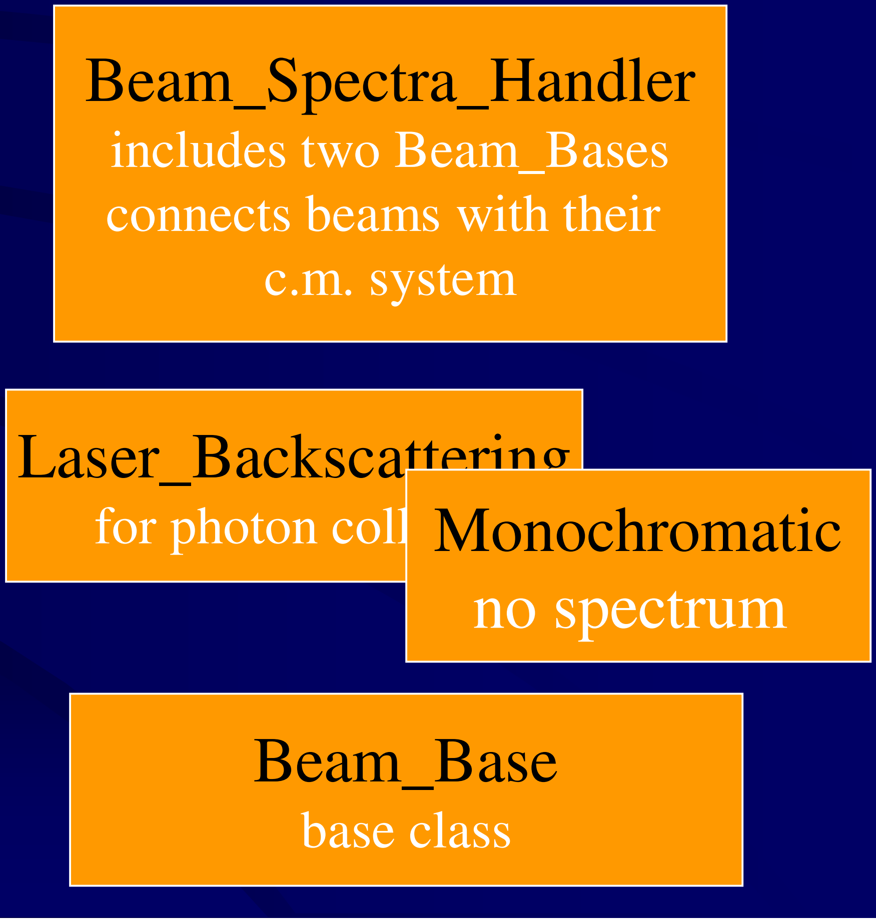
Quick tour of modules: ATOOLS

- Physics tools:
  - Event record stuff, flavour declarations, observables, selectors, …
  - event record is a list of "blobs", connected through particles.
- Organisation tools :
  - Reading data, some global parameters, stuff for parallel computing, ...
  - Input is handled through ASCII-files with declarations keyword = value
- Mathematical tools:
  - Functions, simple integrators, matrices, random numbers, histograms, ….

Quick tour of modules: BEAM

- Simple beam handling
  - different spectra, so far monochromatic and Laser back scattering
  - geometries simple to add, so far not explored
  - beam spectra lead to specific phase space integrators

- Class structure



Beam_Spectra_Handler
includes two Beam_Bases
connects beams with their
c.m. system

Laser_Backscattering
for photon coll

Monochromatic
no spectrum

Beam_Base
base class

Quick tour of modules: PDFs

ISR_Handler
includes two ISR_Bases
connects bunch particles and partons

Structure_Function
includes a PDF_Base

Intact
bunch particle is the interacting one,
no pdf is needed

ISR_Base
might include a PDF_Base,
stores some information

PDF_Handler
selects a pdf from the available ones
according to input data

LHAPDF_Fortran_Interface

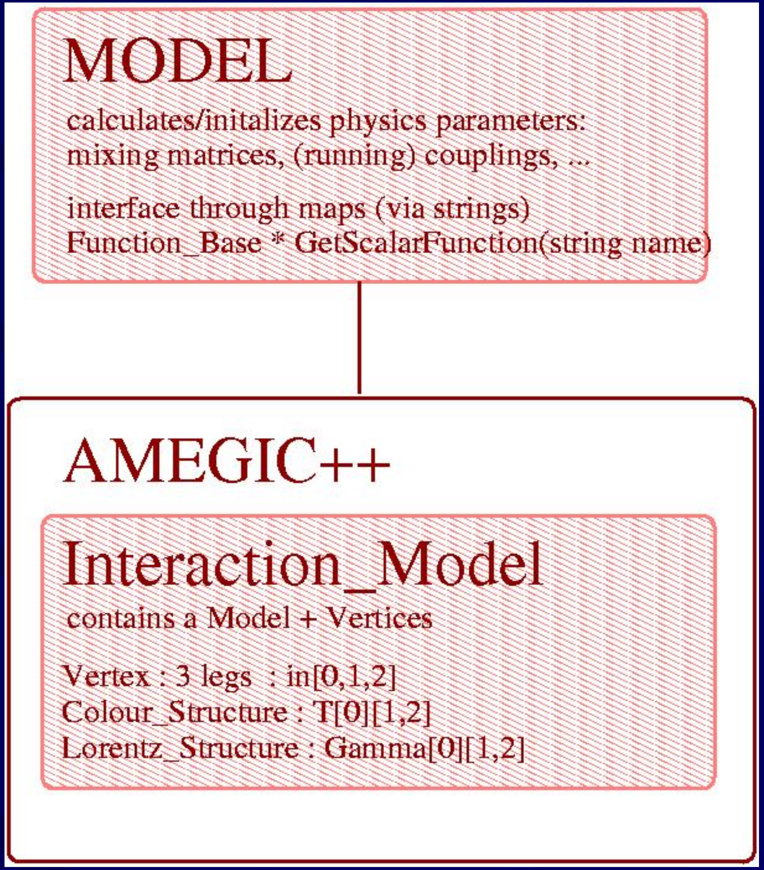
PDF_MRST99 (C++)

PDF_Electron

PDF_Base

Quick tour of modules: MODEL

- Physical model:
  - masses (electroweak sector, SUSY),
  - mixing matrices,
  - running couplings, running masses ….
- Interface to spectrum generators:
  - Isajet/Isasusy
  - Hdecay



MODEL

calculates/initalizes physics parameters:
mixing matrices, (running) couplings, ...

interface through maps (via strings)
Function_Base * GetScalarFunction(string name)

AMEGIC++

Interaction_Model

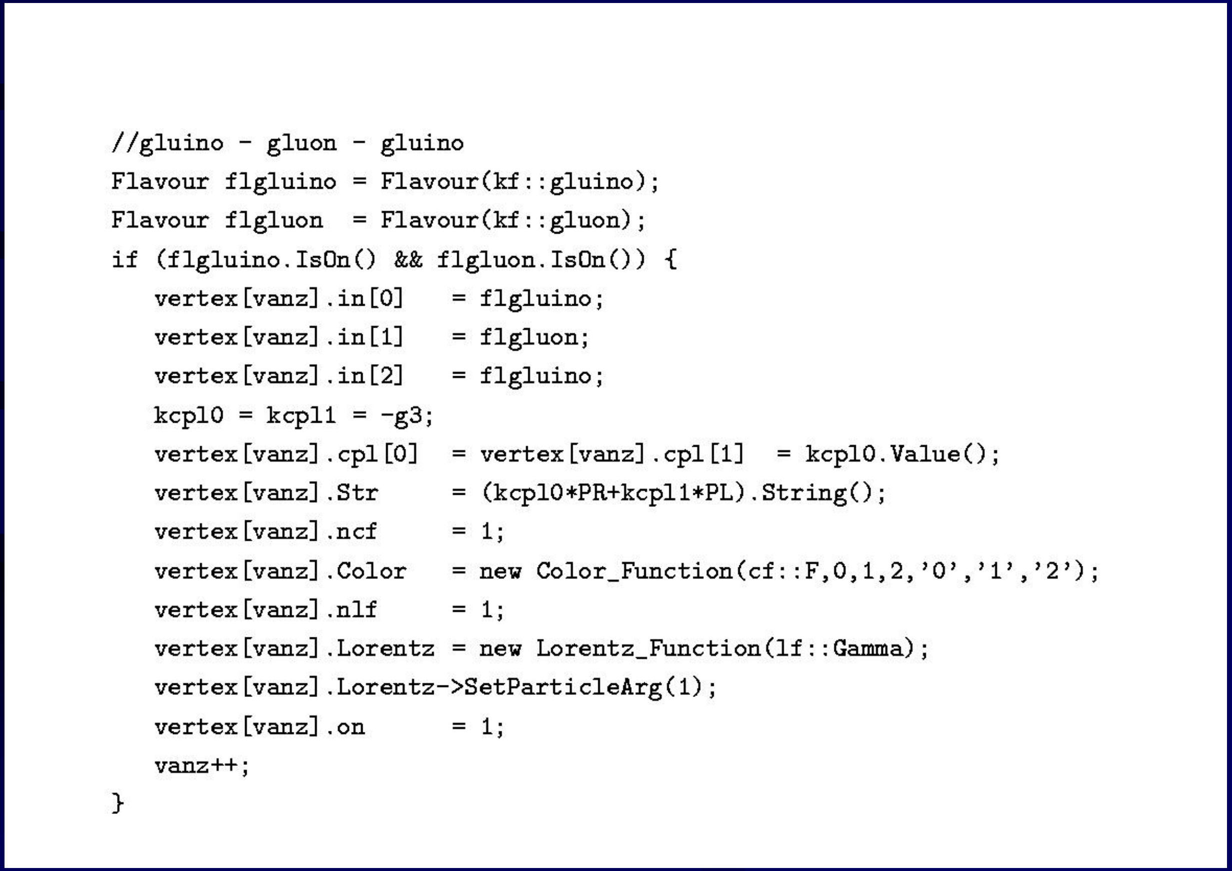
contains a Model + Vertices

Vertex : 3 legs : in[0,1,2]
Colour_Structure : T[0][1,2]
Lorentz_Structure : Gamma[0][1,2]

Quick tour of modules: AMEGIC

- Calculation of arbitrary tree-level processes
  - SM, MSSM, ADD, extension simple & straightforward
- Explicit polarisation vectors
- Adaptive MC integration
- Completely automatic approach: AMEGIC writes out libraries of helicity amplitudes & phase space mappings
  - leads to user interaction: run, make install, run

Quick tour of modules: AMEGIC

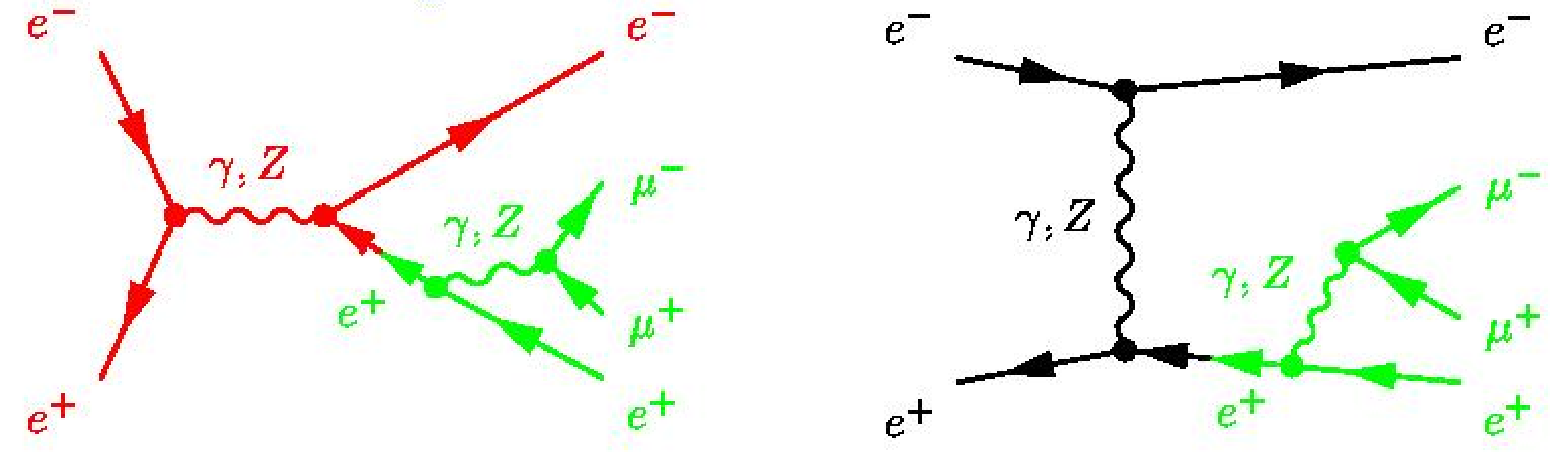


```
//gluino - gluon - gluino
Flavour flgluino = Flavour(kf::gluino);
Flavour flgluon  = Flavour(kf::gluon);
if (flgluino.IsOn() && flgluon.IsOn()) {
   vertex[vanz].in[0]    = flgluino;
   vertex[vanz].in[1]    = flgluon;
   vertex[vanz].in[2]    = flgluino;
   kcpl0 = kcpl1 = -g3;
   vertex[vanz].cpl[0]   = vertex[vanz].cpl[1]   = kcpl0.Value();
   vertex[vanz].Str      = (kcpl0*PR+kcpl1*PL).String();
   vertex[vanz].ncf      = 1;
   vertex[vanz].Color    = new Color_Function(cf::F,0,1,2,'0','1','2');
   vertex[vanz].nlf      = 1;
   vertex[vanz].Lorentz  = new Lorentz_Function(lf::Gamma);
   vertex[vanz].Lorentz->SetParticleArg(1);
   vertex[vanz].on       = 1;
   vanz++;
}
```

Quick tour of modules: AMEGIC

To tame the factorial growth : Factor out common pieces

$\Rightarrow$ “Superamplitudes”

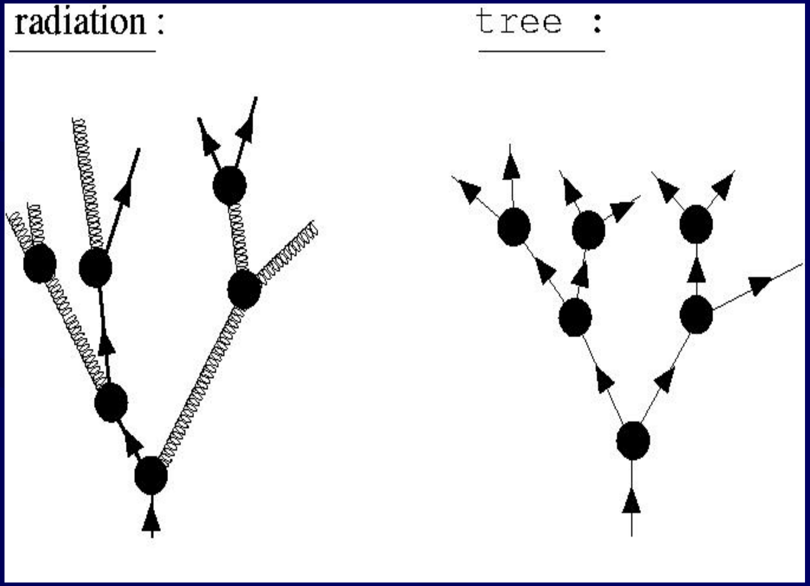

e^-
e^-
γ, Z
γ, Z
μ^-
e^+
μ^+
e^+
e^+
e^-
e^-
γ, Z
μ^-
γ, Z
μ^+
e^+
e^+
e^+

Quick tour of modules: AMEGIC

- AMEGIC was exhaustively tested:
  - trivial tests: ee → 4f, 4f+γ
    (vs. RacoonWW)
  - not so trivial: ee → 6f, 2q4g, ….
    (vs. Lusifer & Helac/Phegas)
  - pp → W+jets, WW+jets, ….
    (vs. Madgraph)
  - Some simple SUSY processes
    (really complicated ones not available so far)
  - tested limit so far: 6 final sate particles

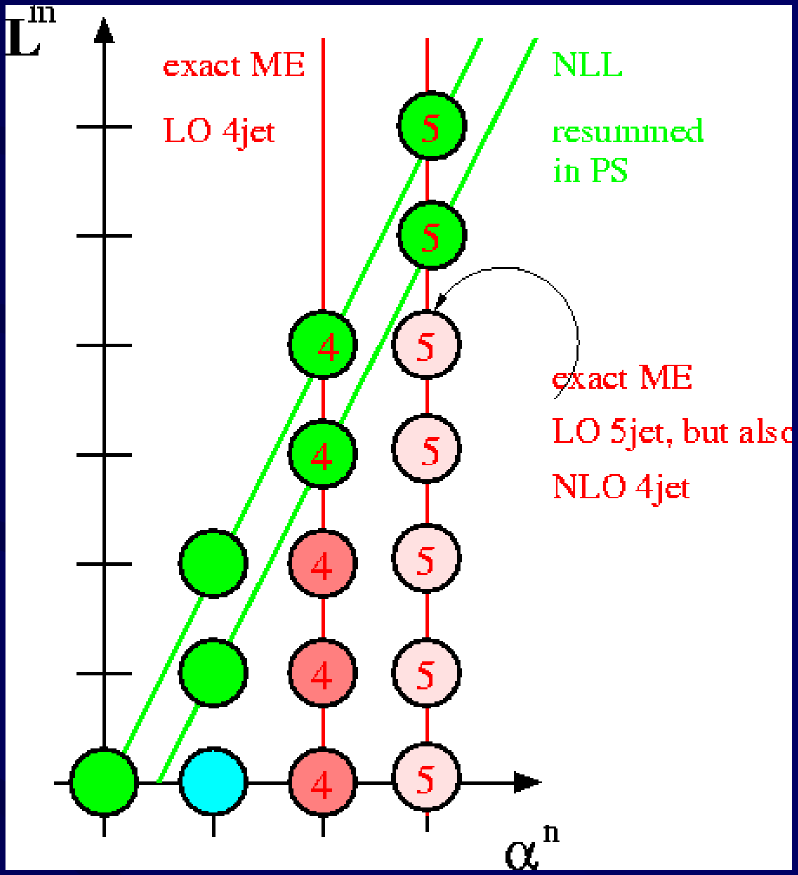
Quick tour of modules: APACIC

- Parton shower ordered by virtualities with angular ordering imposed "by hand" (Pythia-like)
- General idea: Organisation through a binary tree, each knot containing the branching parton



Interfacing MEs & PS

- General idea:
  Produce jets with MEs,
  evolve them with PS
- Potential traps:
  Double counting,
  mismatch of leading logs.
- Cured in SHERPA -
  implementation of suitable
  algorithm: Extra weight
  on MEs, veto on PS


L^m
exact ME
LO 4jet
NLL
resummed
in PS
5
5
4
5
4
5
4
5
4
5
4
5
exact ME
LO 5jet, but als
NLO 4jet
α^n

Hadronisation & hadron decays

…. are handled by a call to Pythia at the moment ….

…. it is anticipated that a new hadronisation module based on an improved cluster scheme will be incorporated in SHERPA …

…. a hadron decay module is a top item on the agenda.

SHERPAs event handling

SHERPA
main class,
initialises physics
handlers, fill them
into the Event_Phases
and adds them in the
Event_Handler

Event_Handler
contains a list of
Event_Phase_Handlers,
passes a (extending) list
of blobs through them.
They know the type of
blob they must deal with
and take according action.

Signal_Processes
link to Matrix_Element_Handler

Jet_Evolution
link to Shower_Handler and
ME_PS_Interface

Hadronization
link to Beam_Remnant_Handler
and Fragmentation_Handler

Event_Phase_Handler
base class

Plans for the near future

- Check, check & check all modules
  (ME working group, ME+PS interface …)
- Public release of the code in August,
  take part in the Data Challenge (if possible)
- Implement the hadronisation package
- Add hard decays (e.g. of top etc.) - that's simple - and add spin correlations - that's tedious ...

Plans for the not so near future

- Construct a new hadron decay module
- underlying event
- pile-up events

BE READY FOR LHC IN 2005

(i.e. with a full code)