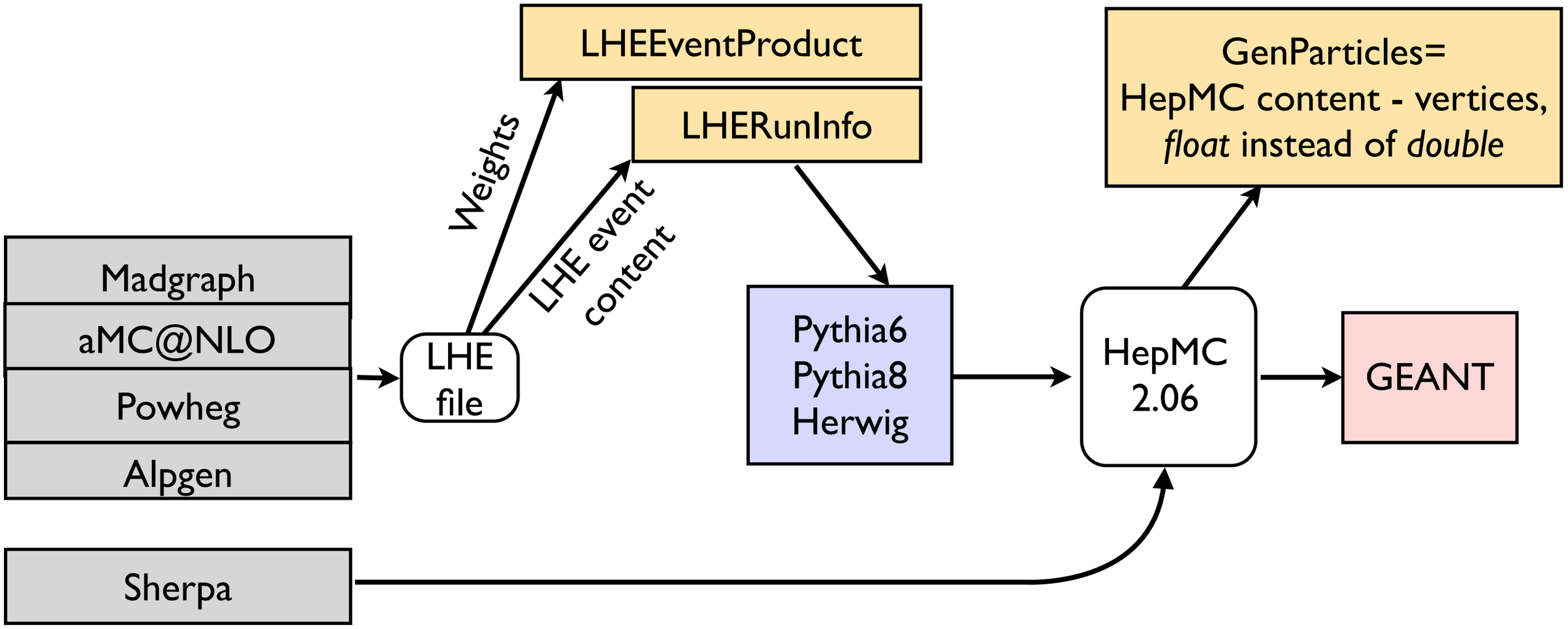
Future use of HepMC in CMS: weights, jet matching, etc...

Simon de Visscher (CERN)

on behalf of CMS collaboration

Curent use of HepMC in CMS

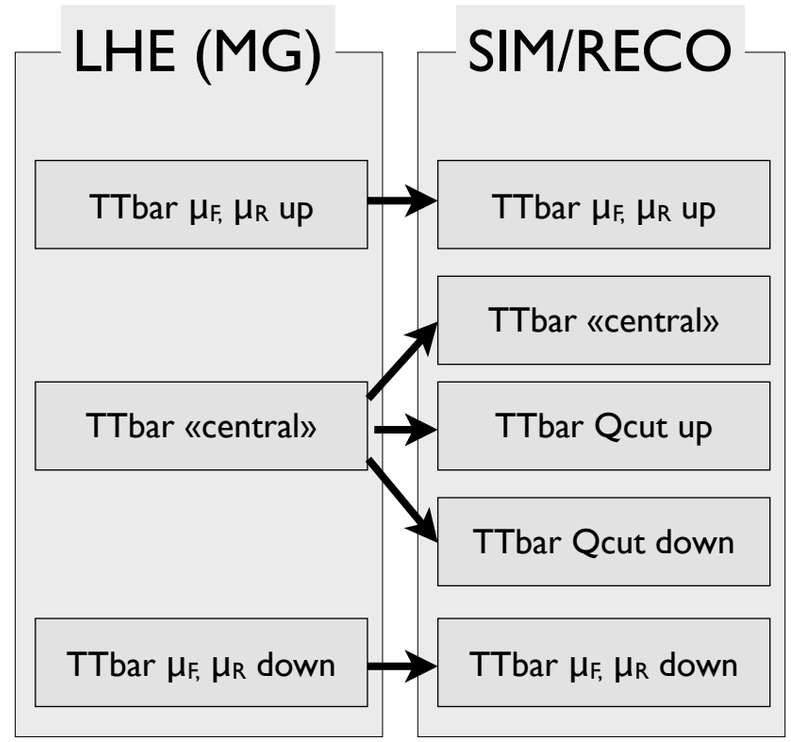


i) HepMC collection **dropped** after reconstruction is performed in order to save storage place

ii) GenParticles is kept instead

Sample production in CMS

- Dealing with th. systematic: the current situation
  - Several different fully simulated samples for one given process
    - μ_F, μ_R up and down simultaneously
    - jet matching scheme: only one
      - ref matching scale (Qcut), scale up, scale down
  - This situation is not sustainable
    - not inclusive enough
    - CPU time/space consuming.

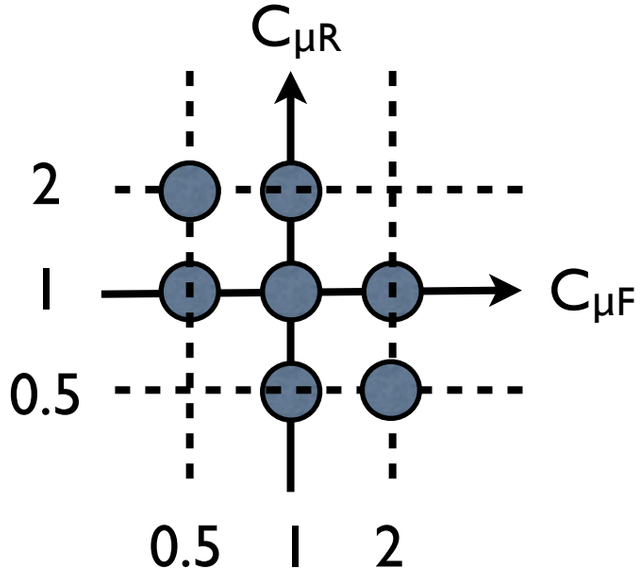


3 LHE files

5 fully simulated samples

In the future: Scales, PDF, matching schemes

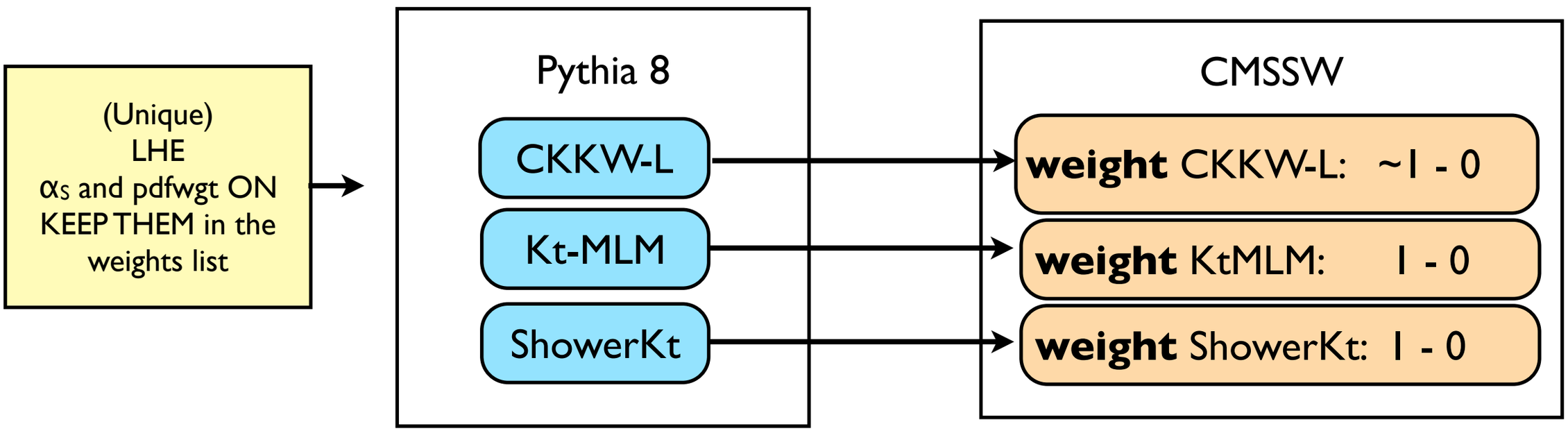
- Aim at reducing the number of fully simulated samples
  - one process in one phase-space region: one sample.
  - That means to propagate series of weights and flags all along the simulation chain
    - LHE→SIM→RECO
    - The analyzer should only play with weights and flags to evaluate impact of th. systematics
- μ_F, μ_R (at least for NLO)
  - consider the envelope of scale variation by factor 2 , excluding (2,2) and (0.5,0.5)



- PDF change: get weights from LHE file
  - now standard in aMC@NLO
- Matching scale
  - Variation of the scale
  - Different schemes (Kt-MLM, ShowerKt, CKKW-L, ...)

more details on the matching/merging side

- Idea: abandon the unique-scheme event veto like we had in the past.
  - Instead, keep flags/weight for each matching scheme



- ☺ Just playing with flags (and weights), estimation of matching uncertainty becomes trivial, even at reco level
- ☺ Machinery ready to welcome next generation schemes (UMEPS, ...)
- ☺ That goes along with the idea of recording the weights from scale, PDF changes
- ☹ Average MC sample size increases a bit, since all events with all flags =1 should be kept

- To summarize:
  - From LHE level, we need to keep track of
    - different weights
      - PDF changes
      - fact and ren scales changes
      - pdf «reweighting» and alpha_s for jet matching
    - but also additional event-by-events infos from «comment» line.
  - From PS step, we need to keep track of
    - response from jet matching/merging schemes
    - potentially also from matching scale variation.

HepMC use in future CMS event production

- From our point of view, we are not closed the option to increase the use of HepMC in the future but
  - Considering HepMC to propagate the weights, flags, event contents all along the procedure of simulation is a serious topic
    - that would mean replace our existing LHE/HepMC duality (perhaps not the best/ most elegant choice, but it works). ⇒ a radical change.

  - What support can we expect in the future for a given version?
  - Preparation of the MC samples for Run II has already started.
    - Procedure to convert to a HepMC-based only solution will take time.
    - Backward compatibility is most probably necessary.

Present issues with HepMC

- HepMC content is heavy.
  - Drop it after full simulation/reconstruction is performed
  - GenParticle content is kept instead.
  - In the long term: use HepMC format from the first step (LHE), adding elements when available, dropping other existing collections?
    - Why not, but could HepMC be lighter?
- For extremely complex events (long chains): data structure in HepMC is too heavy to be read: memory allocation of 2Mb on lxplus, while 10Mb should be required
  - «Our most recent problem was we discovered that with complex events, ROOT I/O needs something near 10MB of stack size to be able to write out the event because of the extreme recursion depth of the call stack. This is a problem for threaded jobs since unix's default stack size for additional threads is only 2MB (although it is settable). Also, the internal use of std::map in the object takes up lots of extra memory and fragments the heap as well as is extremely difficult for ROOT to serialize. Therefore, we'd like to propose that work is done to improve the I/O characteristics of the data structures defined in HepMC.»