TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Mathematik und Naturwissenschaften Institut für Kern- und Teilchenphysik

SHERPA PERFORMANCE

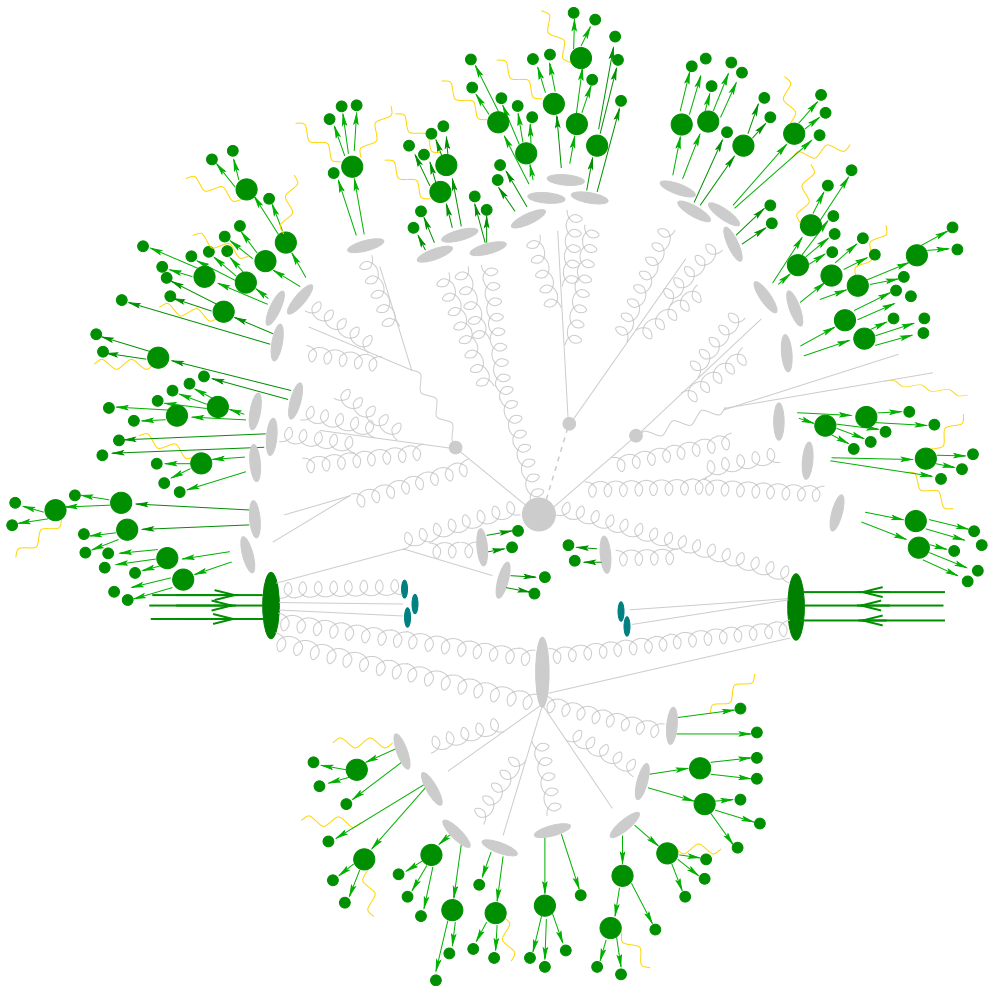
2nd Fast Monte Carlo Workshop in HEP

Frank Siegert

DESY Zeuthen, January 2014

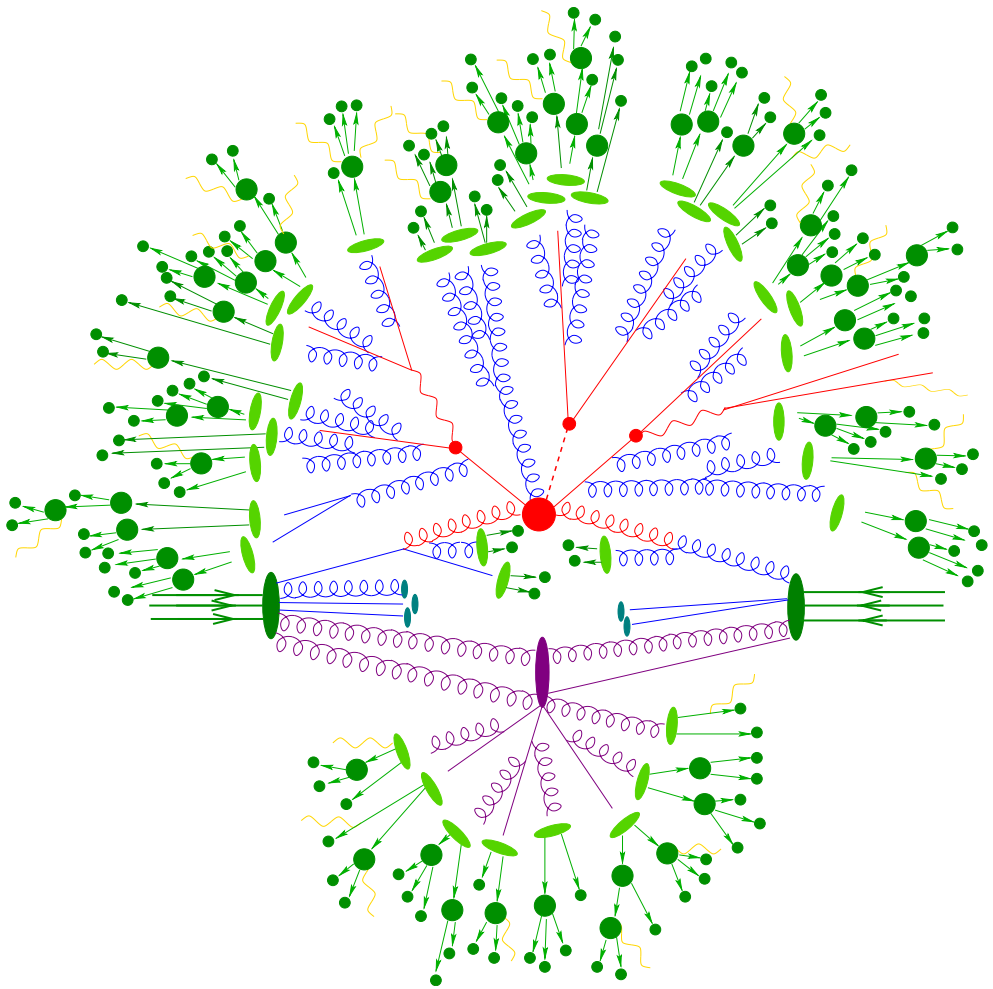
Table of Contents

The SHERPA framework



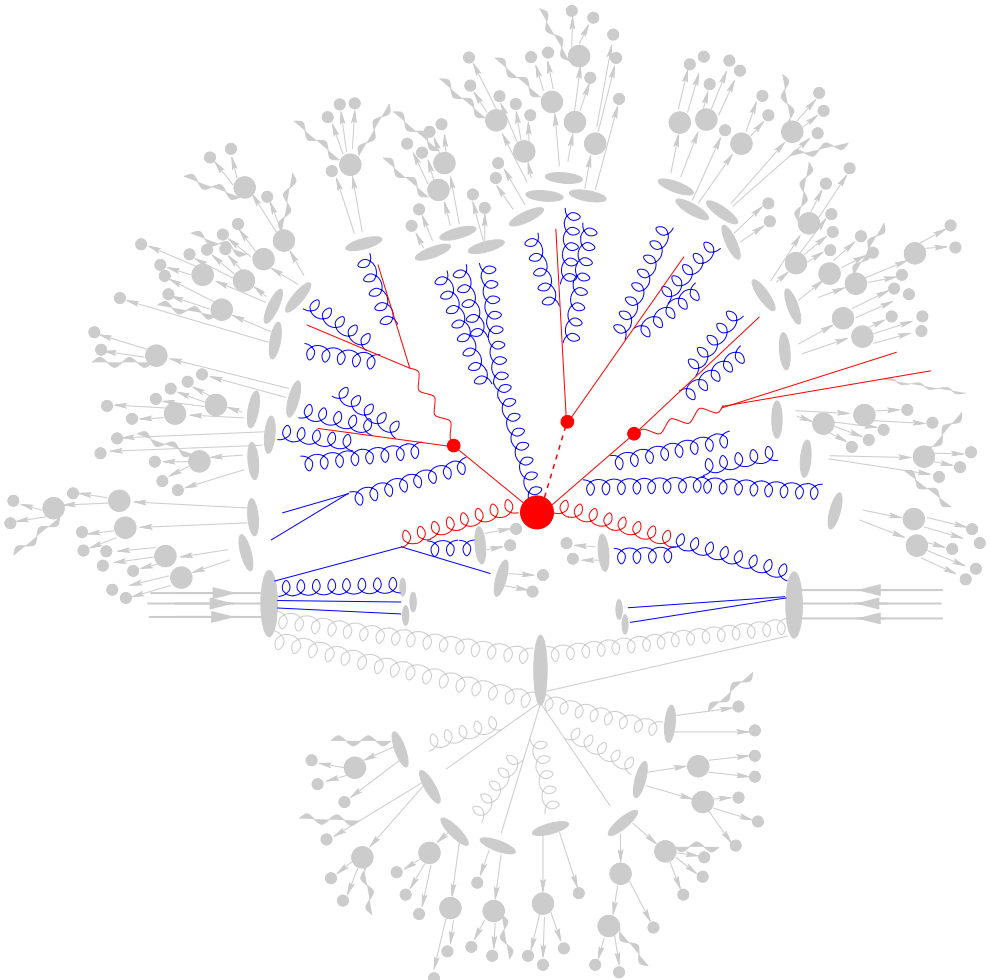
- SHERPA framework: Simulation of $pp \to$ full hadronised final state

The SHERPA framework



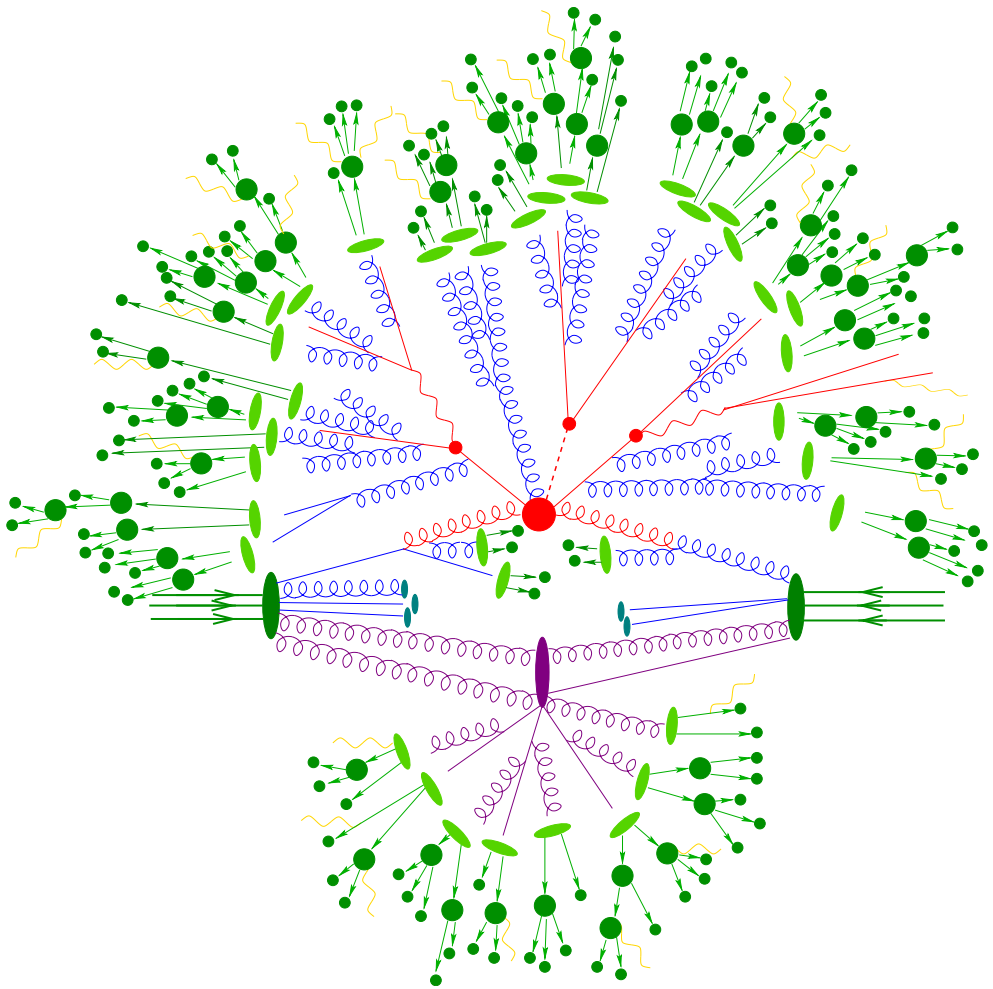
- SHERPA framework: Simulation of $pp \to$ full hadronised final state
- Factorisation into stages: MC event representation

The SHERPA framework



- SHERPA framework: Simulation of $pp \to$ full hadronised final state
- Factorisation into stages: MC event representation
- We know from first principles:
  - Hard scattering at fixed order in perturbation theory (**Matrix Element**)
  - Approximate resummation of QCD corrections to all orders (**Parton Shower**)

The SHERPA framework



- SHERPA framework: Simulation of $pp \to$ full hadronised final state
- Factorisation into stages: MC event representation
- We know from first principles:
  - Hard scattering at fixed order in perturbation theory (**Matrix Element**)
  - Approximate resummation of QCD corrections to all orders (**Parton Shower**)
- Remaining bits:
  - Hadronisation
  - Hadron decays
  - Multiple parton interactions
  - QED FSR resummation

Example: Multi-jet merging

In a nutshell

- → (N)LO matrix elements for $pp \to X + 0, 1, \ldots, n$ jets
- → Combined with each other and the parton shower (PS)

How good physics becomes expensive

Example: Multi-jet merging

In a nutshell

→ (N)LO matrix elements for $pp \to X + 0, 1, \ldots, n$ jets

→ Combined with each other and the parton shower (PS)

- Obvious performance penalties
  - Expensive multi-jet matrix elements
  - Complicated phase space integration

How good physics becomes expensive

Example: Multi-jet merging

In a nutshell

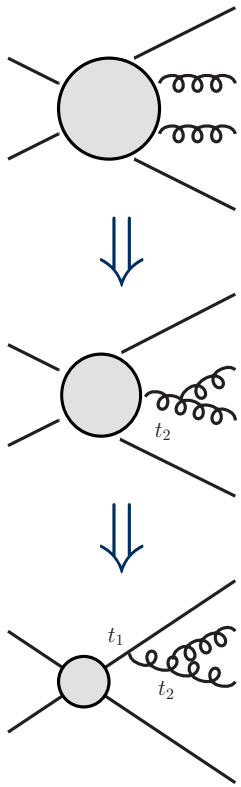
→ (N)LO matrix elements for $pp \to X + 0, 1, \ldots, n$ jets

→ Combined with each other and the parton shower (PS)

- Obvious performance penalties
  - Expensive multi-jet matrix elements
  - Complicated phase space integration
- Example for non-obvious issues: matrix element clustering
  - Aim at preservation of ME fixed order **and** PS resummation properties
  - Achieved by interpreting ME event in parton shower language

  ⇒ Probabilistic backwards-clustering with parton shower splitting kernels

  = much more expensive than e.g. k_T clustering



Cost breakdown – Preparation

Preparation costs = once per sample

Process construction (building "Feynman graphs")

- not (yet) parallelisable
- identifies simplifications and mappings
- largest memory demands here, $\mathcal{O}(1\text{GB})$ for complicated processes
- storage of information for production runs

Phase space integration of the matrix elements

- parallelisable with multithreading (up to factor ~ 5) or MPI ($\sim$ perfect scaling)
  $\rightsquigarrow$ later
- storage of results for production runs

Example: W + 0,1,2j@NLO + 3,4,5j@LO

- 5h $\times$ 1CPU process construction
- 8h $\times$ 8CPU integration for W + 0,1,2j@NLO (including virtuals from OpenLoops)
- 48h $\times$ 8CPU integration for W + 3,4,5j@LO
  (roughly factor of two lower if only up to 4 quarks)

Cost breakdown – Initialisation

Initialisation costs = once per Grid job (core)

Process construction

- reads stored information from preparation phase
- CPU and I/O
- depends significantly on I/O speed with SHERPA $< 2.1.0$
- can take up to hours for more complicated processes

Example: W + 0,1,2j@NLO + 3,4,5j@LO

- 10min process initialisation
- $\sim$ constant 5min remaining initialisation

Cost breakdown – Generation

Generation costs = once per event

Matrix element unweighting

- challenging matrix element calculations, e.g. table
  Höche, Gleisberg (2008)
- really expensive: unweighting efficiency for complicated processes as low as 0.001%
  $\rightarrow 10^5$ ME calculations per event
- additionally merging with parton shower including ME clustering

$\rightarrow$ $\mathcal{O}(1\text{ day})/1000$ events in complicated cases

Process	[ms/pt.]
$gg \rightarrow 2g$	0.073
$gg \rightarrow 3g$	0.339
$gg \rightarrow 4g$	1.67
$gg \rightarrow 5g$	8.98
$gg \rightarrow 6g$	49.6
$gg \rightarrow 7g$	298.
$gg \rightarrow 8g$	1990.
$gg \rightarrow 9g$	13100.
$gg \rightarrow 10g$	96000.

Remaining generation chain

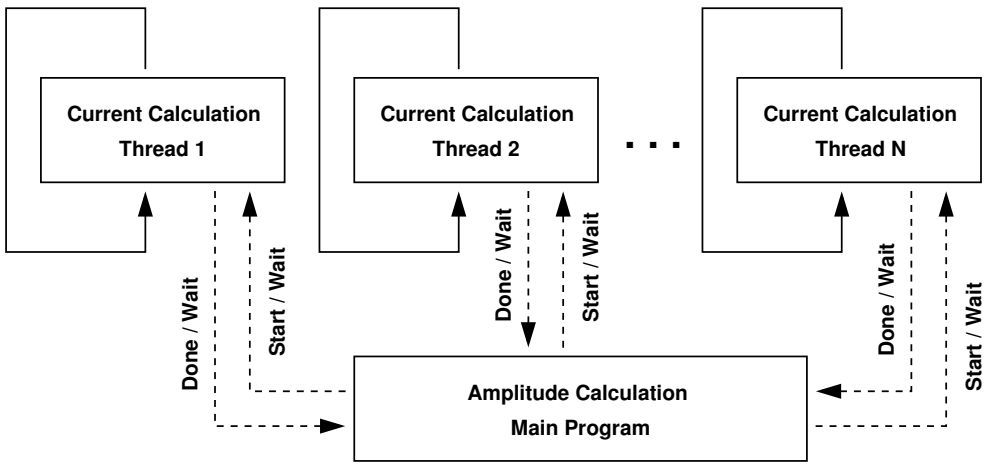
- remaining cost of event independent of ME+PS: < 0.5 s/evt
- includes hadronisation, decays, QED FSR, multiple parton interactions

Example: W + 0,1,2j@NLO + 3,4,5j@LO

- from ATLAS central production: $\sim$ 12h/2000 evts $\approx$ 20 s/evt

$\rightarrow$ needs work if detector simulation gets to $\mathcal{O}(1\text{s})$/evt

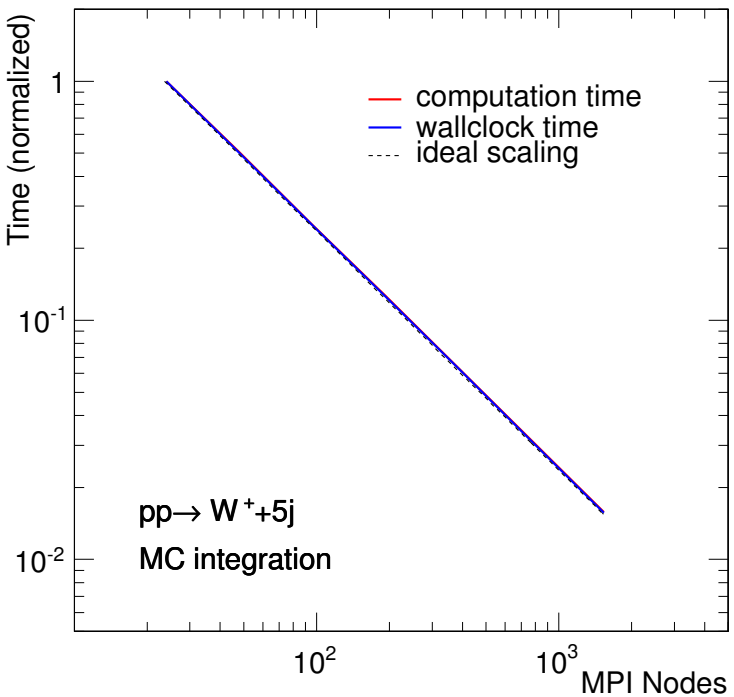
Multithreading in SHERPA



- same process manages multiple computing threads
- parallelisation of loop over process group and calculation of phase space weight
- uses shared memory for all threads
- saturates at ~ 5 cores $\Rightarrow$ not really useful for HPC

Message Passing Interface (MPI)

- separate process per core, communication through (fast!) network
- memory required per core, not shared
- in SHERPA used for parallelisation of loop over phase space points, communication/optimisation of integrators every $\sim 10^{4\ldots5}$ points iteration
- different MPI implementations supported (e.g. Cray, IBM, openmpi)



Nearly perfect scaling with number of nodes

- "strong" scaling validation
  fixed number of points per iteration
  - up to 1024 nodes on Titan
- "weak" scaling validation
  adapt number of points per iteration
  - up to 8192 nodes on Titan
  - up to 16000 nodes on Vesta

Höche, Reina, Wobisch, et al., 2013

SHERPA on different HPC architectures

Cray XK7 "Titan" at OLCF

Höche, Reina, Wobisch, et al., 2013

- 16 AMD OpteronTM 2.2 GHz cores per node (299,008 total cores), 32 GB RAM per node
- Cray Gemini 3D Torus Network
- Linux environment and Cray MPI implementation with Gnu compilers
- similar setup to Cray XE6 "Hopper" at NERSC

IBM BlueGene/Q test system "Vesta" at ALCF

- 16 1.6 GHz PowerPC A2 cores per node (32,768 total cores), 16 GB RAM per node
- IBM 5D Torus Network
- IBM-specific environment and MPI implementation with Gnu compilers

Intel Xeon Phi co-processor

- tested with 61 $\times$ 4 compute cores at 1.238 GHz, 16 GB total memory
- offload mode for specific calculations $\Rightarrow$ needs dedicated programming model (not implemented in SHERPA)
- alternatively: as many-core processor
  - uses regular MPI-mode
  - performance penalty of one core $\sim$ factor of 16 compared to CPU

  $\rightarrow$ not really efficient according to these first tests

I/O performance issues

- SHERPA relies on information in many small files:
  - process construction/mapping information
  - integration results
  - decaydata for hadron decay channels
  - multiple parton interaction grids
- Total number of (small) files read during initialisation is in the thousands

$\Rightarrow$ Performance penalty on slow file systems like in HPC or Grid sites

Improvements (Sherpa $\geq$ 2.1.0)

- store all file contents in one database instead of many small files
- for practicality: use Sqlite database format
- still tuning performance (cache size, index creation, ...)

$\Rightarrow$ I/O improvements as required by ATLAS(/CMS?) production and on HPC systems

Conclusions

Summary

- event generation is trivially parallelisable
- performance improvements necessary if detector simulation gets to $\mathcal{O}(1\text{s})$/evt
- main emphasis of HPC usage in SHERPA: matrix element integration for multi-jet merging
- MPI used on many different architectures, nearly perfect scaling up to thousands of cores

Outlook

- SHERPA 2.1.0 to be released in the next weeks
- main improvement from HPC perspective: Sqlite database instead of many small files

- Question: status of MPI in experimental MC production and how to use SHERPA's MPI with it?

Thank you for your attention!