Modern Particle Physics Experiments

Event Generation

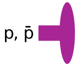
March 17, 2022

Simulation of the high energy collision

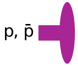
- there are a lot of physical processes involved
- some we understand and can calculate from first principles
- some we can approximately calculate
- for others we have to rely on phenomenological models
- we are helped by being able to separate, at some level of approximation, different physics happening on different time/length/energy scales.
- usually we simulate different pieces separately, together with evolution between different scales

Event generators aim to simulate physical processes occurring in high energy collisions, which after simulation of the detector response can be directly compared with measurements

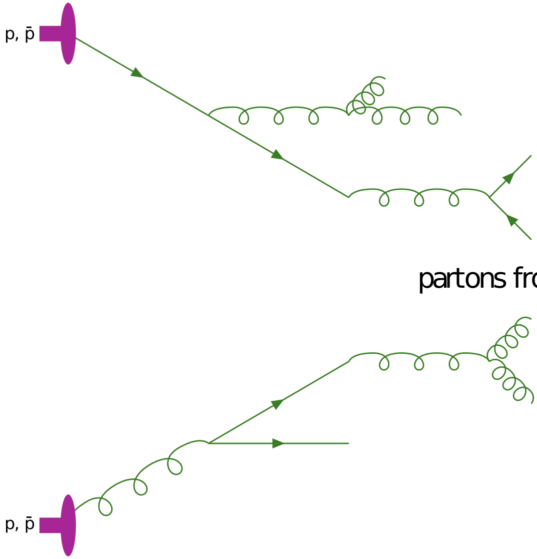
Evolution of an event



$t = -\infty$, incoming protons

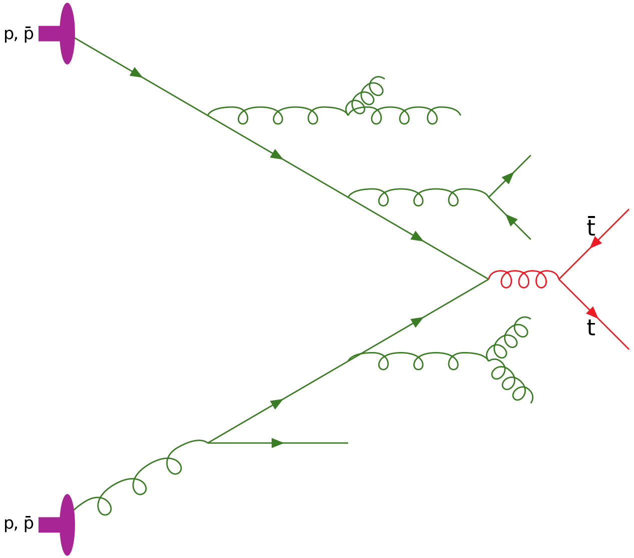


Evolution of an event



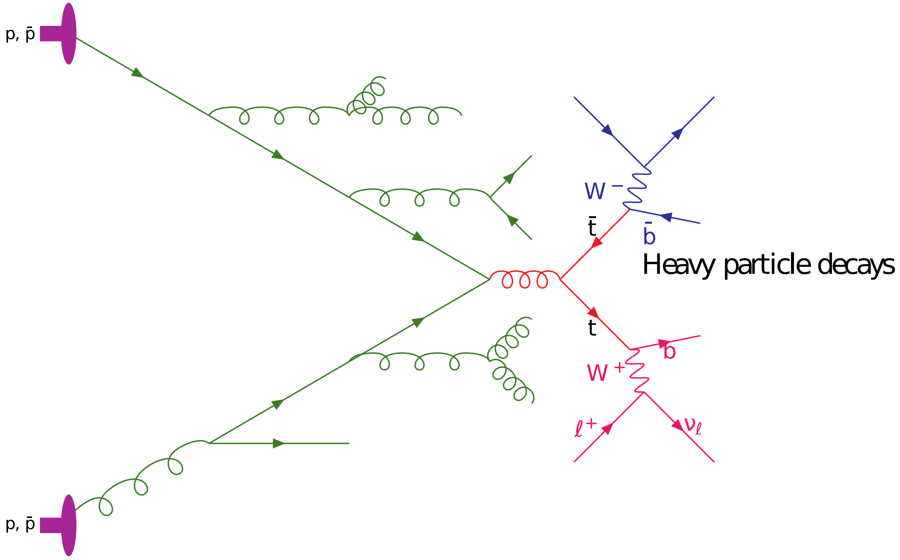
partons from the protons radiate

Evolution of an event

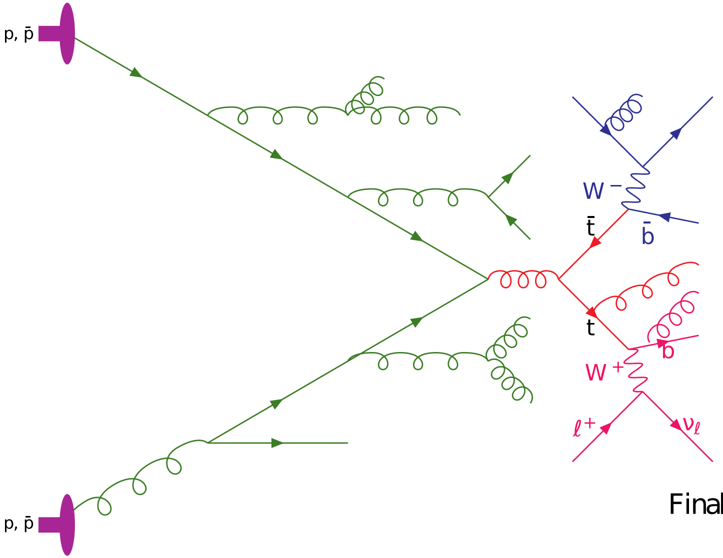


partons collide in
fundamental hard process

Evolution of an event

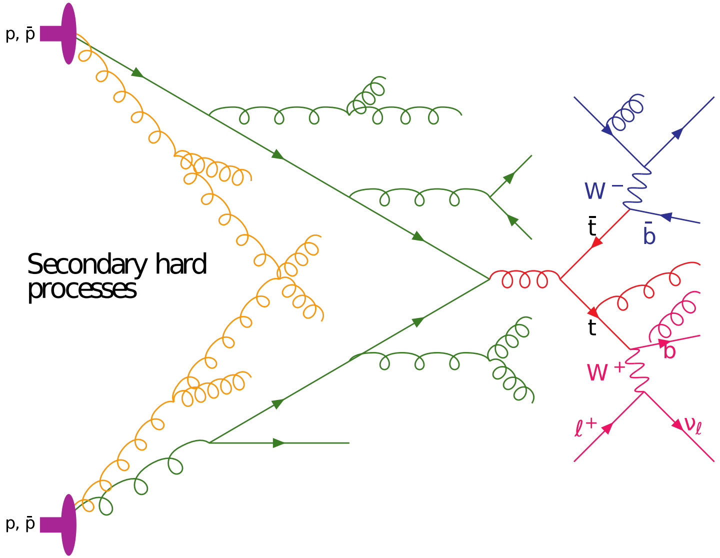


Evolution of an event

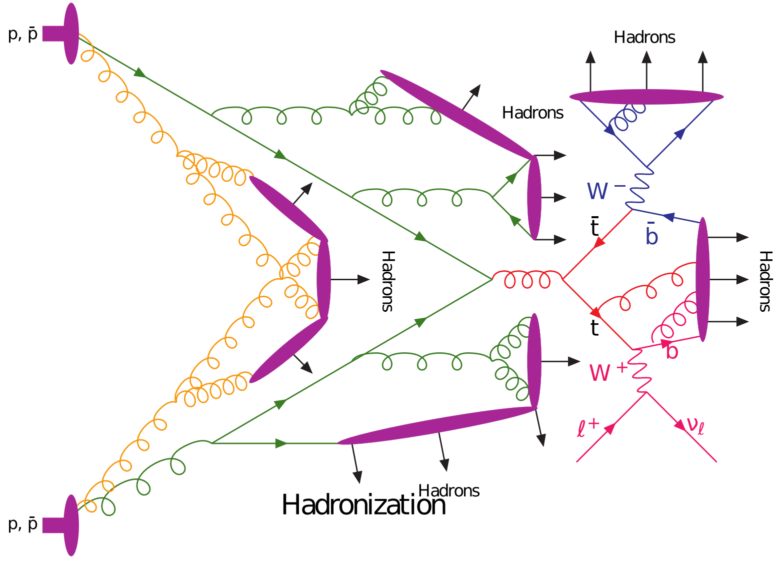


Final-state radiation

Evolution of an event



Evolution of an event



Parton-level approach

- we want to calculate expectation value of observable $\mathcal{O}$ - function of the momenta of n particles in final state
- at parton-level it is given by:

$$\langle \mathcal{O} \rangle = \int\limits_{\Omega} \frac{(2\pi)^4}{2s} \prod_{i=1}^{n} \frac{\mathrm{d}^3 p_i}{(2\pi)^3 2E_i} |\mathcal{M}\{p_i\}|^2 \mathcal{O}\{p_i\}$$

- in case of hadronic collisions we also need to convolve the integrand with parton distribution functions (PDFs)
- There are two issues:
  1. calculation of the matrix element
  2. integration over the phase space Ω

Monte Carlo integration

- integrals evaluated in particle physics often involve complicated limits, multi-dimensional spaces and integrands with peaks and divergences, integrals that we deal with are in general impossible to evaluate analytically
- discretization techniques (e.g. trapezium method) are difficult to implement and converge slowly for high-dimension domains
- we use Monte Carlo methods to compute complicated integrals
- simple 1D integral can be written as an average:

$$I = \int_{x_1}^{x_2} f(x)\mathrm{d}x = (x_2 - x_1)\langle f(x)\rangle$$

- we calculate the average with Monte Carlo, by randomly selecting N points x_i in the interval:

$$I \approx I_N = \frac{x_2 - x_1}{N}\sum_{i=1}^{N} f(x_i) = \frac{1}{N}\sum_{i=1}^{N} w_i \qquad w_i = (x_2 - x_1) f(x_i)$$

Monte Carlo integration

- error of integral calculated in such way is expressed as follows:

$$\delta I = \sqrt{\frac{1}{N-1}\left(\frac{1}{N}\sum_{i=1}^{N} w_i^2 - \left[\frac{1}{N}\sum_{i=1}^{N} w_i\right]^2\right)}$$

- generalization to n-dimensional case is straightforward, in order to integrate n-variable function $f(\mathbf{x})$ over the volume Ω ($\mathbf{x}$ is a point in n-dimensional space) we build a hypercube containing whole Ω and set integrand values outside Ω to zero, weights w_i are calculated as follows:

$$w_i = V \cdot f(\mathbf{x}_i)$$

- where V is the volume of said hypercube, expressions for I and δI remain unchanged

Monte Carlo in event generation process

Monte Carlo methods are essential for modern event generators and are used in many places during event generation. Examples are provided below:

- selection of hard processes based on cross sections
- phase space integrals of matrix elements (just discussed)
- selection of decay channels of unstable particles
- momentum generation of final particles based on physical models and conservation laws

Pythia exercises in a nutshell

- as an example of the event generator we will use Pythia 8.306. Homepage of the Pythia project can be found here
- online manual for the generator is available under this link
- we will use TPythia8 class - ROOT interface for Pythia8
- each particle generated in the simulation is represented as a TParticle object, allowing access to its properties (mass, momentum, polarisation, mothers and daughters)
- Monte Carlo particle numbering scheme is used to identify generated particles (usually called PDG codes)

Configuration, invoking and accessing output of Pythia will be shown in the workshop part of the class