1 Description of algorithm

1. `Evolve(Configuration * config)`

   - Iterate over all partons in the configuration - if they are switched off ignore them.
     `Evolve(Parton * parton)`, see for details below, returns a boolean (true in the case of finding a splitting with evolution parameter t above the cut-off t_0, $t > t_0$, false if no allowed splitting has been found). It also keeps track of the winning splitting `Splitting * p_winner`, i.e. the splitting with the largest t to date. If such a splitting exists, (`p_winner!=NULL`), the respective parton will be split according to the parameters stored in it. Irrespective of whether the splitting is viable in `PerformSplitting` (it will always be possible, unless there is a bug in how the offsprings of the splitting are inserted into the configuration), the evolution parameter of the configuration will be set to the t of the splitting.
   - Add a final weight to the overall weight, see below.

2. `Evolve(Parton * parton)`

   - Start with `p_winner = NULL`, i.e. the default is that we do not find a successful splitting.
   - Iterate over all spectators (one in the case of quarks, two in the case of gluons).
     For each spectator, produce `Kernels`, a vector of `Kernel`'s, which encapsulate the splitting function and the coupling part of the splitting kernel. They depend on the splitter and on the initial/final-state nature of both splitter and spectator. Create a trial t for this combination by calling `GenerateTestSplitting(Kernels * kernels, Splitting & split)`, where a new `Splitting * split` is initiated for each splitter-spectator pair. If successful the `Splitting split` will be filled, and if its t is the largest so far it will be kept as the winning splitting, `p_winner`. If not, `split` will be deleted.
   - The method returns `true` or `false` dependent on whether we found a winner or not.

3. `GenerateTestSplitting(Kernels * kernels,Splitting & split)`

   - `kernels->Integral(split,p_massselector)` iterates over all allowed splitting kernels, and extracts the sum of the integrals of their overestimators, $\sum_K \bar{I}_K$. The integrals are given by

$$\bar{I}_K \;=\; \frac{C_K \alpha_S(t_0))}{2\pi} \int\limits_0^1 \mathrm{d}z \bar{P}_{a\to bc}(z) = \frac{C_K \alpha_S(t_0))}{2\pi} \bar{I}_{a\to bc}\,, \tag{1}$$

where $C_K = C_F = 4/3$ for quarks, $C_K = C_A/2 = 3/2$ for gluons splitting into gluons, and $C_K = 1/2$ for gluons splitting into quarks. $\alpha_S(t_0)$ obviously is the maximum of the strong coupling, since we assume t to be the relevant scale for all splittings. The overestimated splitting functions are given by

$$\begin{aligned}
\bar{P}_{q\to qg} \;=\; \bar{P}^{(1)}_{g\to gg} &\;=\; \frac{2(1-z)}{(1-z)^2+\kappa_0^2}\cdot(1+\bar{K})\,,\\
\bar{P}_{q\to gq} \;=\; \bar{P}^{(2)}_{g\to gg} &\;=\; \frac{2z}{z^2+\kappa_0^2}\cdot(1+\bar{K})\,,\\
\bar{P}_{g\to gg} &\;=\; 1\,.
\end{aligned} \tag{2}$$

We have two gluon splitting functions into gluons, which will need to be added. κ_0^2 is the overestimator of the Curci-Furmanski-Petronzio regulator, t_0/Q^2, and $\bar{K}$ is the overestimator of the soft enhancement function,

$$\bar{K} = K|_{t=t_0} \;=\; \frac{\alpha_S(t_0)}{2\pi}\left[C_A\left(\frac{67}{18}-\frac{\pi^2}{6}\right)-\frac{10}{9}T_R n_f(t_0)\right]\,. \tag{3}$$

The z-integrals of the splitting functions are given by

$$\begin{aligned}
\bar{I}_{q\to qg} \;=\; \bar{I}_{q\to gq} \;=\; \bar{I}^{(1)}_{g\to gg} \;=\; \bar{I}^{(2)}_{g\to gg} &\;=\; \log\left(1+\frac{Q^2}{t_0}\right)\cdot(1+\bar{K})\\
\bar{P}_{g\to gg} &\;=\; 1\,.
\end{aligned} \tag{4}$$

Starting from a given upper scale for the evolution parameter, T, a new trial evolution parameter t emerges as solution of the equation

$$\#\exp\left[-\int\limits_{t_0}^{T}\frac{\mathrm{d}t'}{t'}\sum_K \bar{I}_K\right] \;=\; \exp\left[-\int\limits_{t_0}^{t}\frac{\mathrm{d}t'}{t'}\sum_K \bar{I}_K\right]\,, \tag{5}$$

with $\#$ a random number:

$$t = T\cdot\exp\left[\frac{\#}{\sum\limits_K \bar{I}_K}\right]\,. \tag{6}$$