29.09.2020

Performance-oriented workflow
for Muon Collider simulations

Using Marlin wisely

N. Bartosik

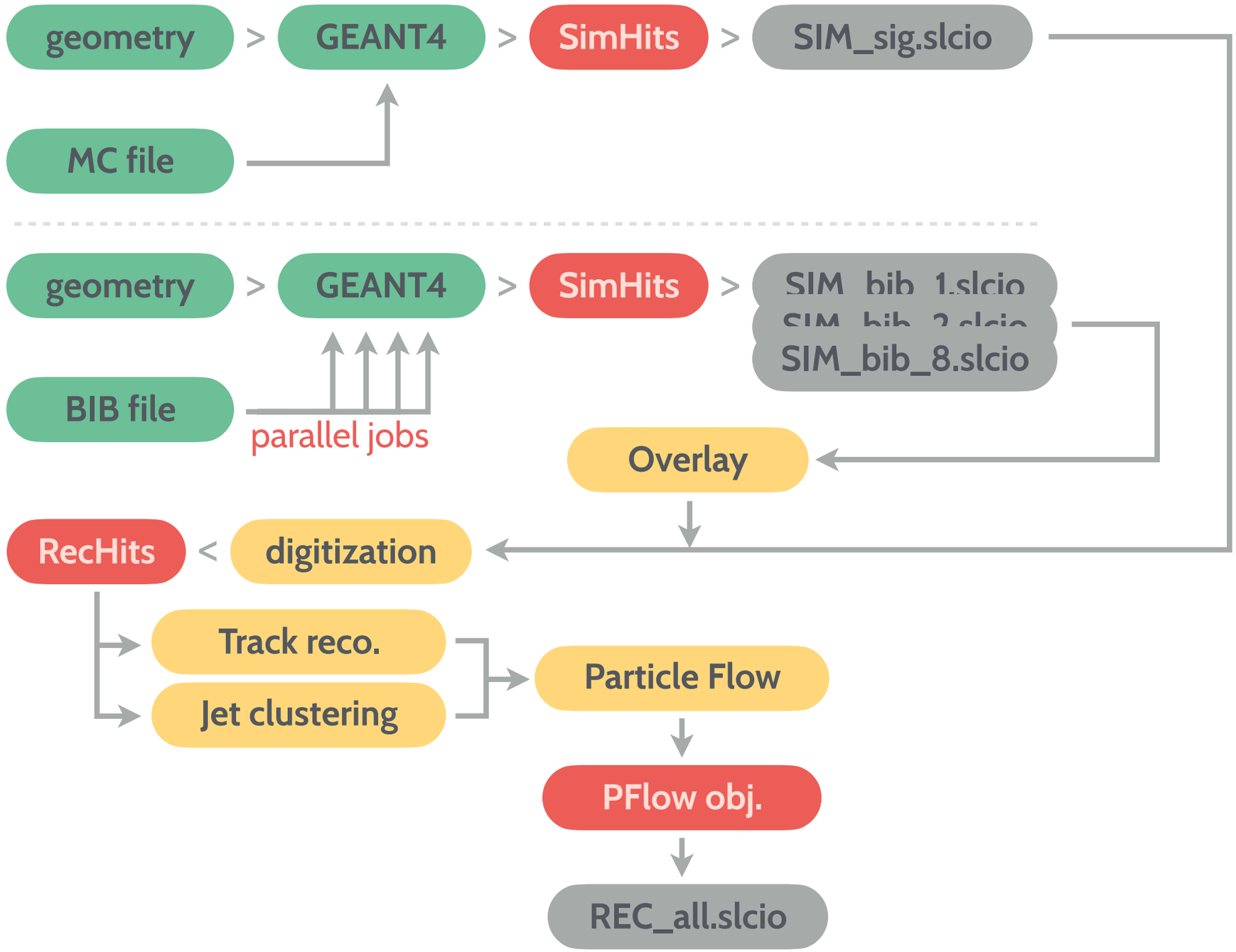
INFN Torino

Current simulation chain

We usually perform simulation in 2 big steps: ddsim and Marlin

1. **Simulation**

   ✔ good



2. **Reconstruction**

   ✕ not so good

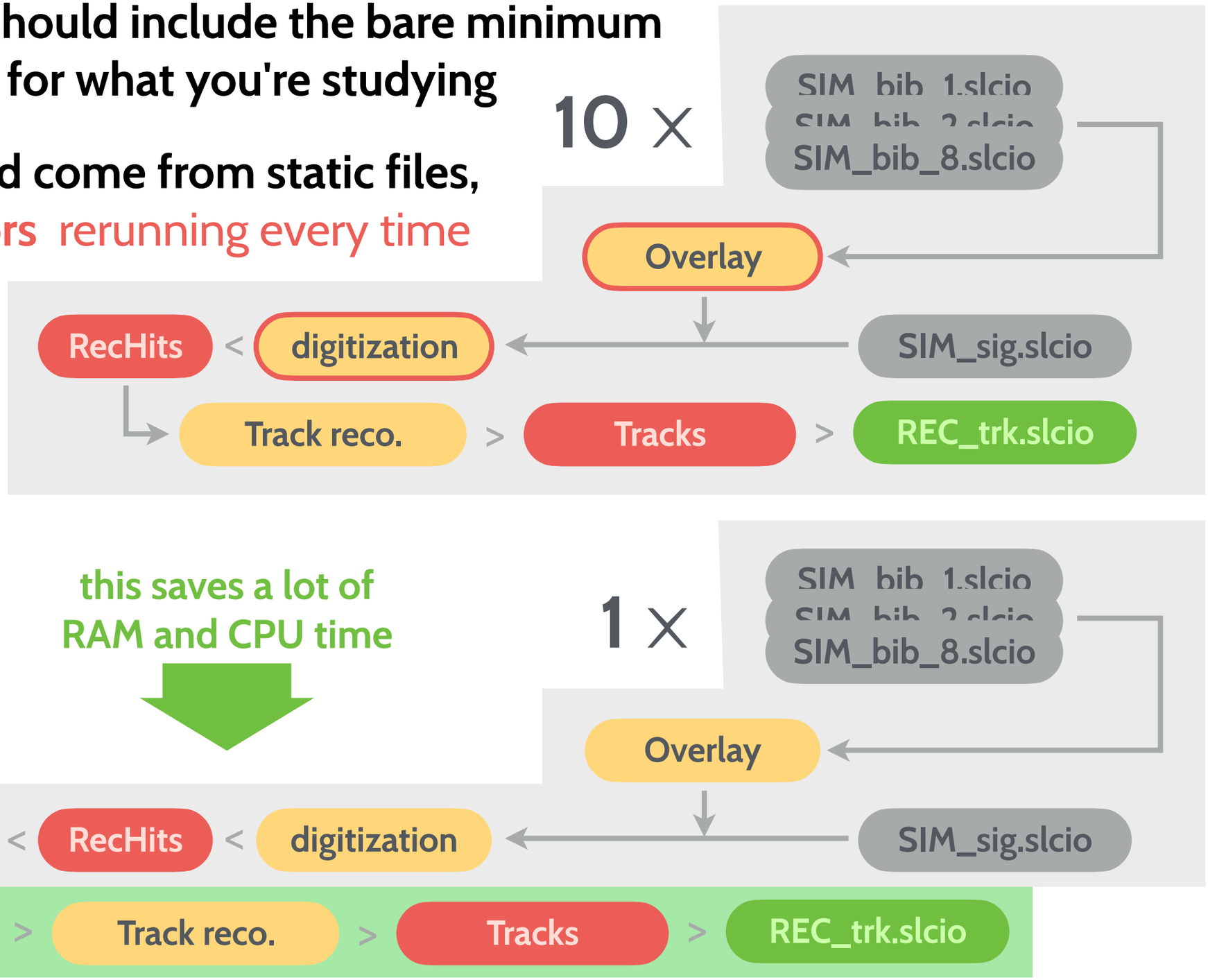
1. Run the minimum: split the chain

We are at an early stage → repeating the same process with different parameters

Your main Marlin job should include the bare minimum of processors relevant for what you're studying

↳ **all the input should come from static files,**
not from processors rerunning every time

Imagine running 10 variations of Track reconstruction



2. **Mind the collections:** disable unused ones

Currently we work on isolated tasks → only limited sets of collections needed

1. **Overlay:** our BIB has way too many particles & hits to merge everything CLIC way
   - exactly the reason why we use the modified Overlay processor

```
<group name="Overlay">
  <parameter name="MCParticleCollectionName" type="string">MCParticle </para
  <!--The output MC Particle Collection Name for the physics event-->
  <parameter name="MCPhysicsParticleCollectionName" type="string"> MCPhysics
  <!--Time difference between bunches in the bunch train in ns-->
  <parameter name="Delta_t" type="float" value="1"/>
  <!--Number of bunches in a bunch train-->
  <parameter name="NBunchtrain" type="int" value="1"/>
  <!--Whether MCParticle collections should be merged (slow with BIB) -->
  <parameter name="MergeMCParticles" type="bool" value="false"/>
  <!--Lower time limit for collections with a single value -->
  <parameter name="IntegrationTimeMin" type="float" value="-0.25"/>

<parameter name="Collection_IntegrationTimes" type="StringVec" >
  VertexBarrelCollection        0.3
  VertexEndcapCollection        0.3

  InnerTrackerBarrelCollection  0.6
  InnerTrackerEndcapCollection  0.6

  OuterTrackerBarrelCollection  0.6
  OuterTrackerEndcapCollection  0.6

</parameter>
```

huge reduction of RAM usage
[true] **46 GB → 5 GB** [false]
← **by not merging 380M BIB MCParticles**

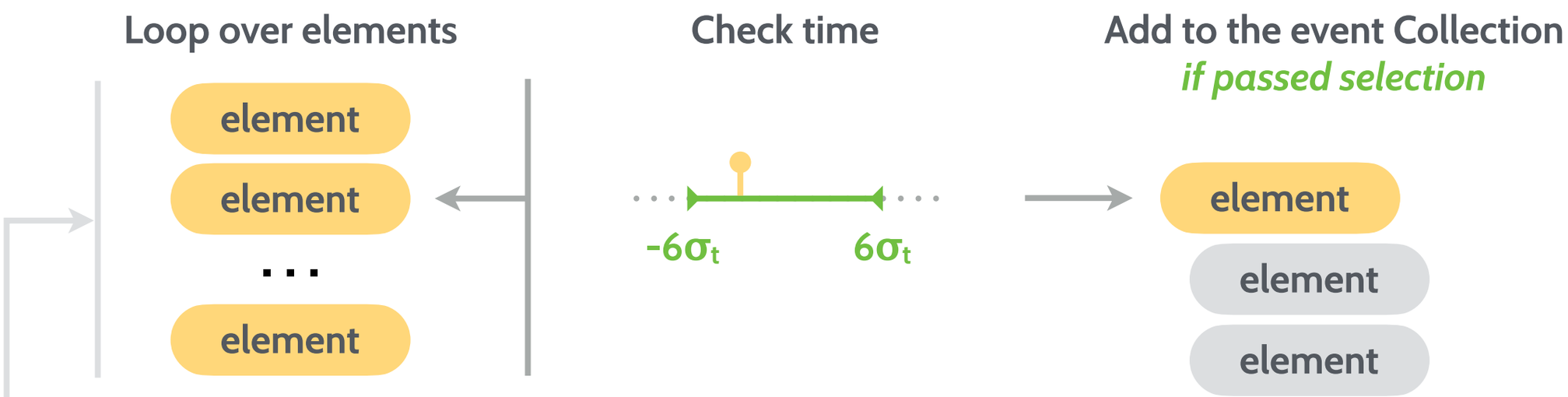
less CPU and RAM usage
← **by not merging irrelevant collections**
only collections present in the list are merged into the event

2. **Output:** only write out collections that are actually needed
   - saves CPU time and disk space →

```
<processor name="Output_REC" type="LCIOOutputProcessor">
  <!--   standard output: full reconstruction keep all collections -->
  <parameter name="LCIOOutputFile" type="string"> digi/test.slcio </parameter>
  <parameter name="LCIOWriteMode" type="string" value="WRITE_NEW"/>
  <!-- <parameter name="SplitFileSizekB" type="int">996147 </parameter> -->
  <parameter name="Verbosity" type="string">WARNING </parameter>
  <parameter name="DropCollectionNames" type="StringVec">  </parameter>
  <parameter name="DropCollectionTypes" type="StringVec">
    SimCalorimeterHit
  </parameter>
```

3. Overlay optimisation: filter only once

Overlay processor performs a very simple task for each collection:



ECalBarrel	13743866
ECalEndcap	3360322
HCalBarrel	16355265
HCalEndcap	9090167
HCalRing	1001555
InnerTrackerBarrel	2423422
InnerTrackerEndcap	865026
OuterTrackerBarrel	2231321
OuterTrackerEndcap	882683
VertexBarrel	2756752
VertexEndcap	2042850

Total numbers of input elements are huge

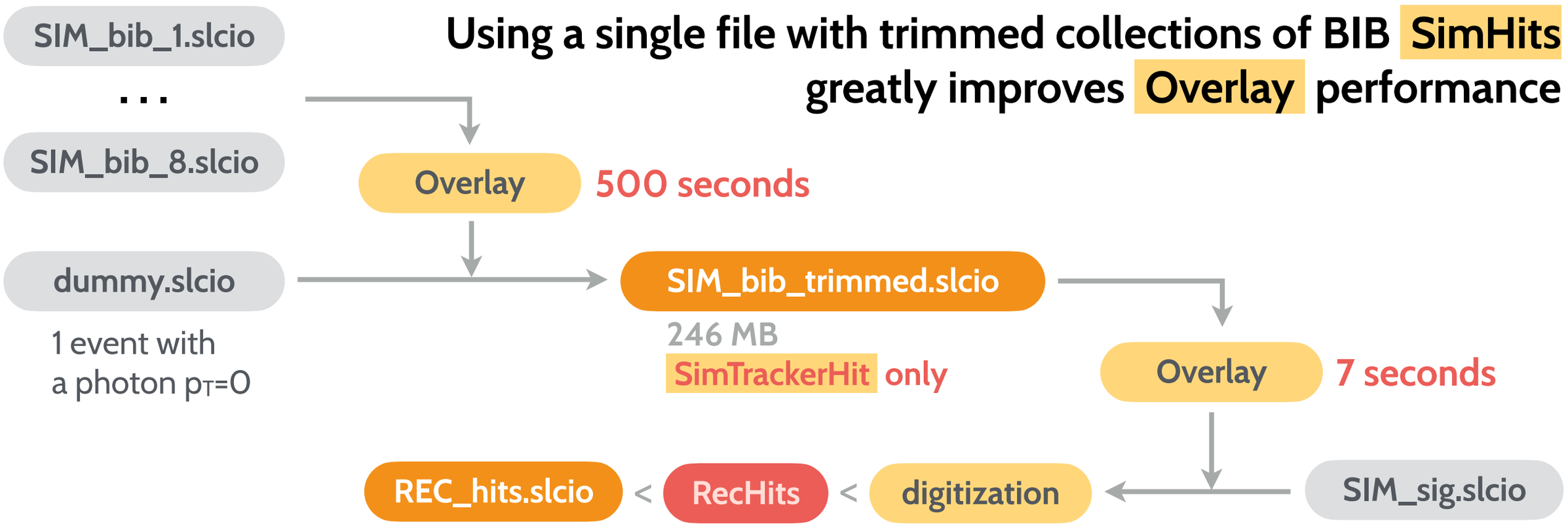
BIB from 1 bunch crossing distributed across 2993 events stored in 16 files

↳ **reading + filtering takes a lot of I/O & CPU** to produce the same set of **SimHits** for merging into the event every single time

Only a subset of SimHits passing the time selection is actually used for digitization

↳ **use BIB files with trimmed collections**

3. Overlay optimisation: trimmed BIB



Using a single file with trimmed collections of BIB SimHits greatly improves Overlay performance

Collection name	All hits	Trimmed	%
InnerTrackerBarrel	2423422	1590802	66%
InnerTrackerEndcap	865026	476061	55%
OuterTrackerBarrel	2231321	1140907	51%
OuterTrackerEndcap	882683	430440	49%
VertexBarrel	2756752	626394	23%
VertexEndcap	2042850	866938	42%

The same approach should be beneficial for Calorimeter-only SimHit collections

I can try to produce 1 file with all trimmed SimHit collections

- hopefully not hitting LCIO format limitations