AOD usage for charm-hadron analyses

F.Colamaria,A.Dainese,A.Festanti,A.Rossi,C.Terrevoli
CERN

Offline Week

8 Nov 2017

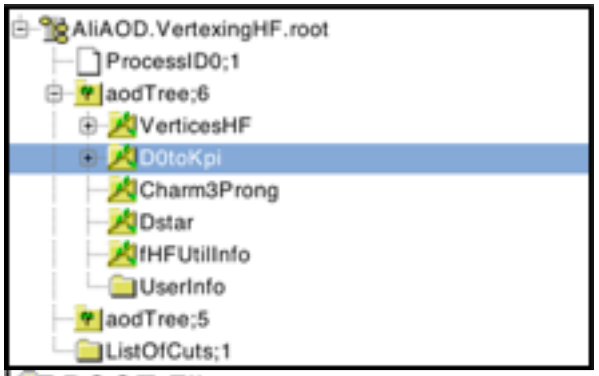
Outline

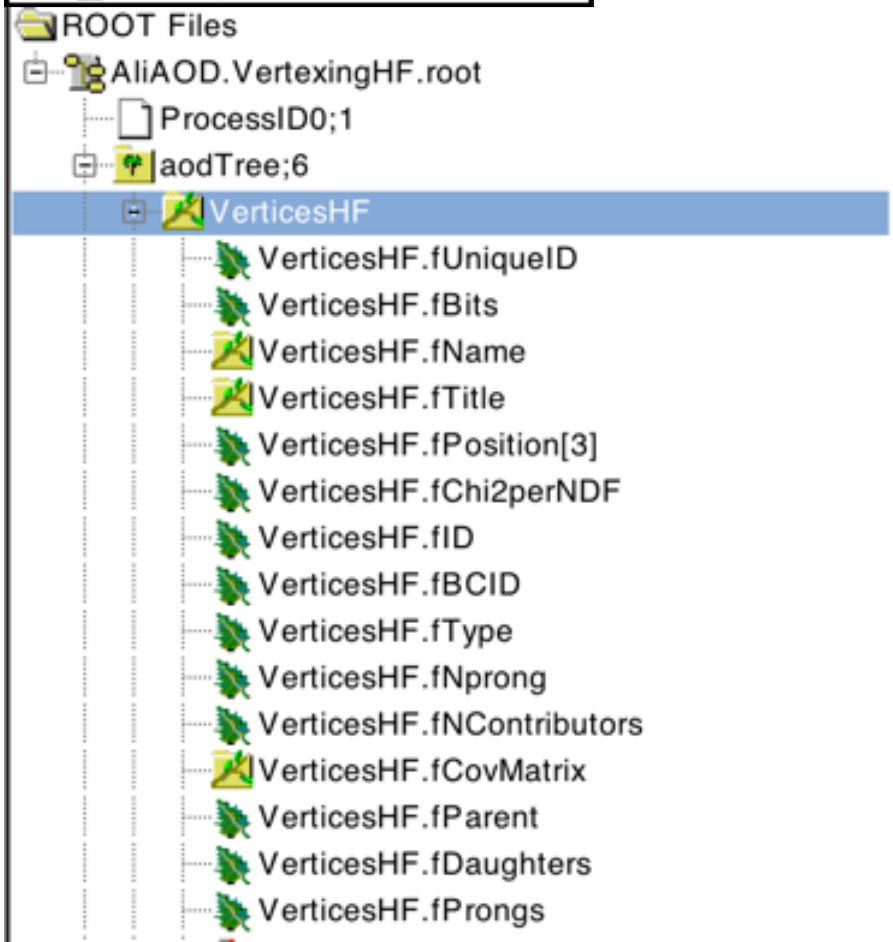
- HF delta AOD production
  - Standard AOD content
  - NEW reduced dAOD
- Number of “filtered” and selected candidates
- Analysis-by-analysis specific issues in view of Run3
- Possible options for Run3 analyses

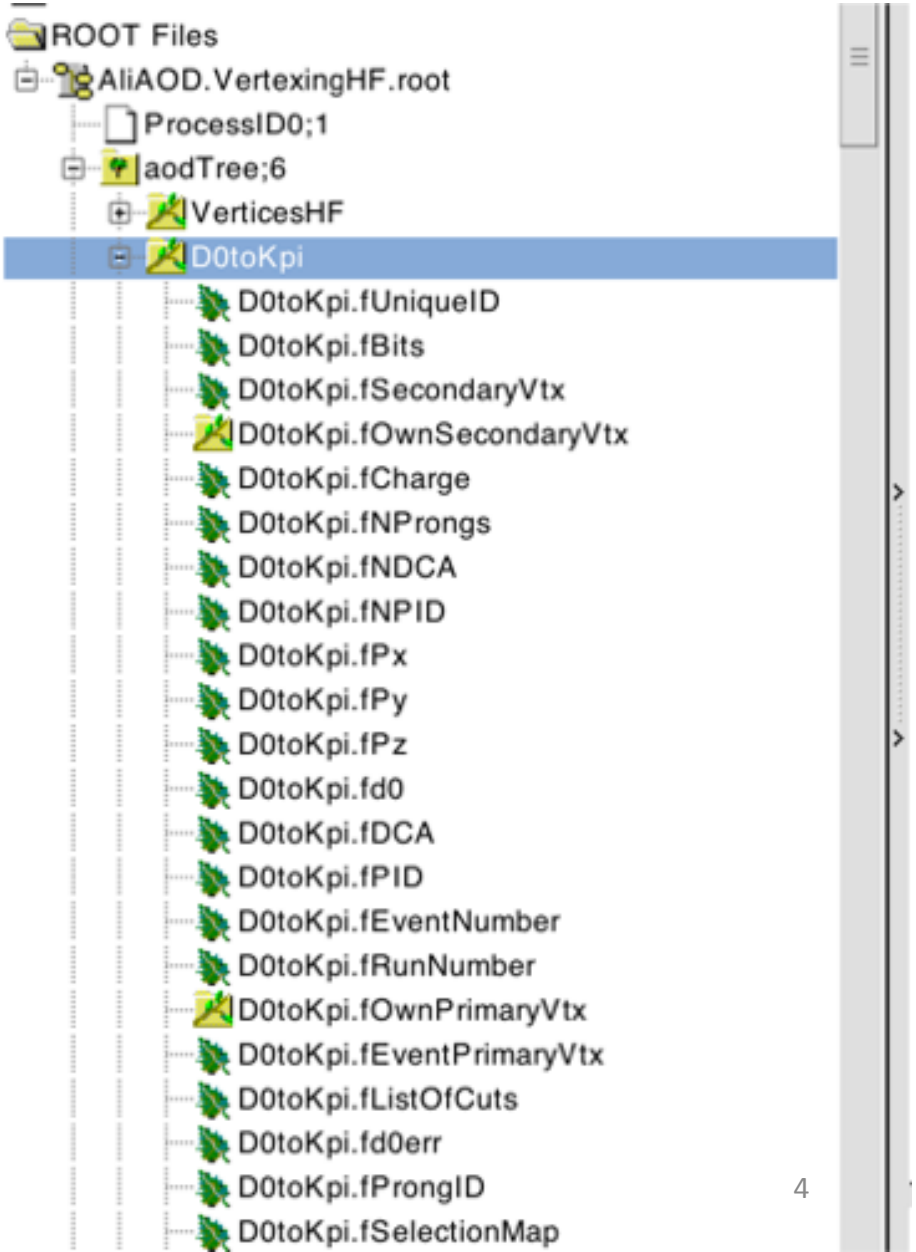
Delta AOD production

- **AliAOD.VertexingHF.root** (associated with AliAOD.root) produced by *AliAnalysisTaskSEVertexingHF*
- **AliAOD.VertexingHF.root** contains a tree with
  - Branch of secondary vertices
  - Branches with charm hadron candidates: D0→Kpi, 3-prong(D+, Ds, Lc), D*, Lc→V0+h, (4-Prong, LikeSign2Prong, LikeSign3Prong, JPsiToEle: only in for pp and pPb)
- Candidates = *AliAODRecoDecayHFNProng* (N=2,3,4) or *AliAODRecoCascadeHF* → *AliAODRecoDecayHF*, *AliAODv0* → *AliAODRecoDecay*→*AliVTrack*→*AliVParticle*

Run1 Pb-Pb dAOD content







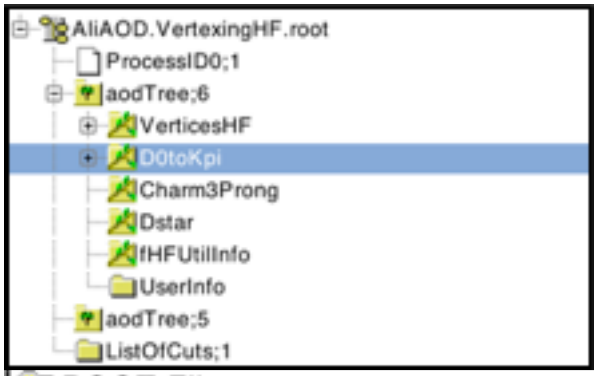
New strategy adopted for Run2 Pb-Pb

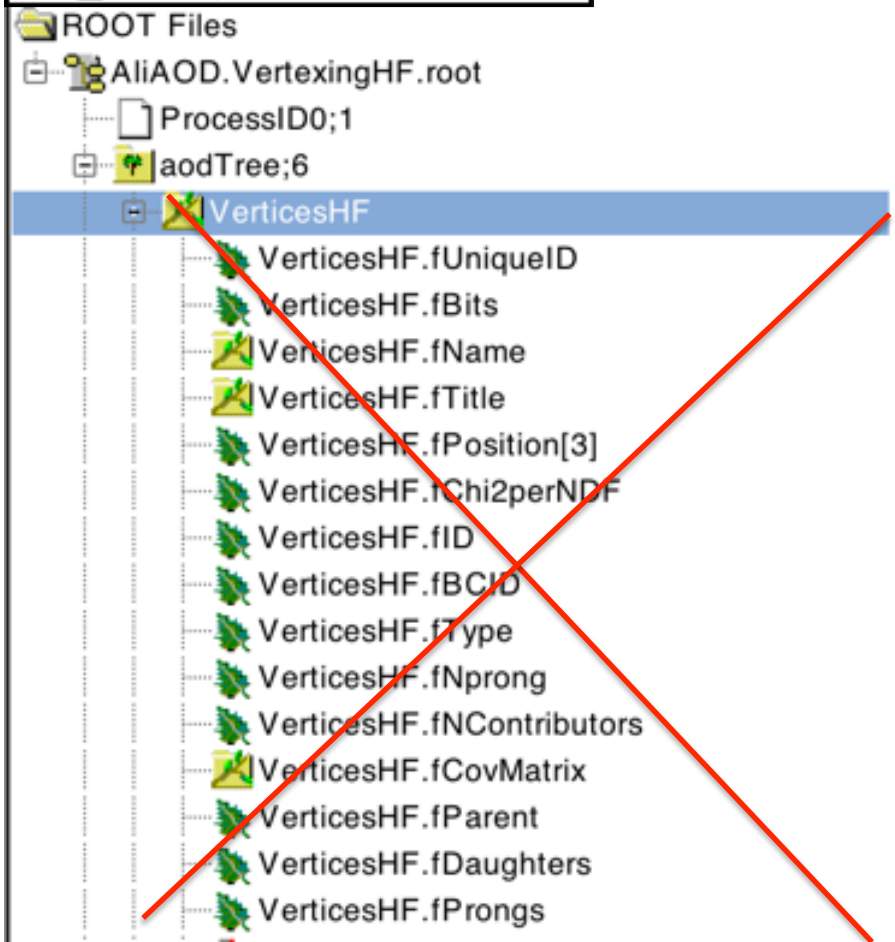
- **Reduced dAOD production** (filtering level):
  - Secondary vertices are not saved
  - Only selected information saved for candidates (e.g. ProngID)
- Analysis tasks use Prong ID to retrieve daughter tracks for each candidate
  - "Filling" of the candidates → re-calculate secondary vertex and candidate properties (fPx, fPy, fPz, fd0, fDCA, …)
- In a train: candidates "filled" only once by the first wagon which uses them (small impact on trains' CPU and memory usage)
- **Factor 8 smaller dAODs** (tested in p-Pb and Pb-Pb)
  - Looser filtering cuts can be applied expecially at low p_T

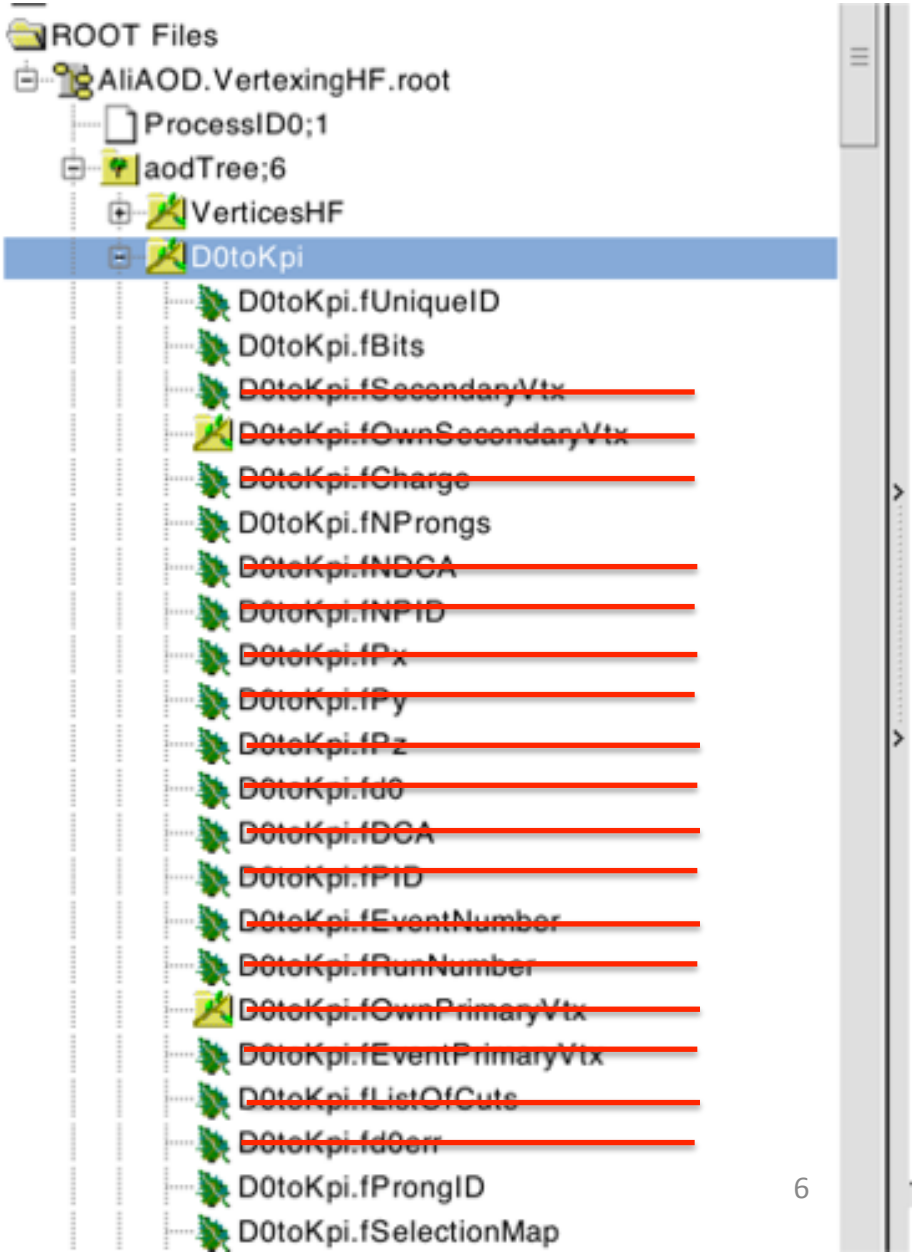
LHC11h: dAOD/AOD~0.5 (standard filtering)
LHC15o: dAOD/AOD~0.11-0.08 factor 4-6 smaller than LHC11h (reduced filtering)
LHC16l,k: dAOD/AOD~0.5 (standard filtering)

- Reduced dAOD can be used also for pp and p-Pb

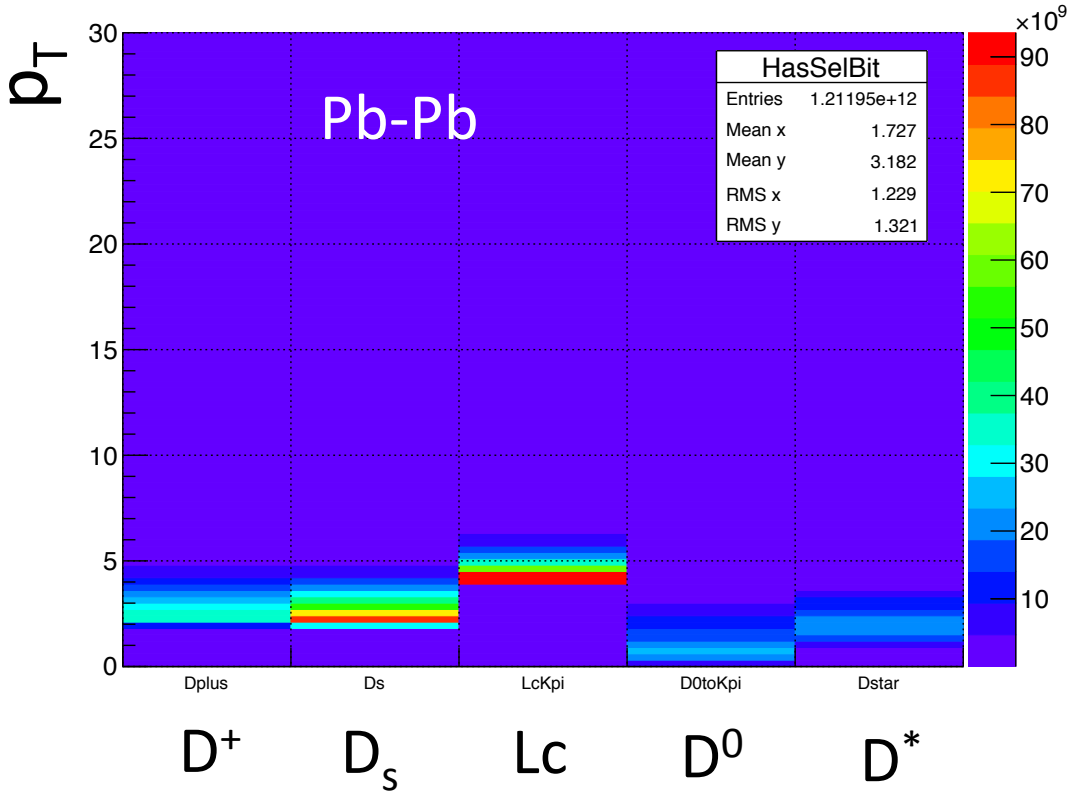
Run2 Pb-Pb dAOD content







Number of Candidates



	0-100% Pb-Pb Run2	pp@13 TeV (2016)
N evt sel	88M	573M
N cand per event – Filtering level		
D0 (pt>1)	1170	0.01
D+ (pt>2)	2181	0.02
D* (pt>3)	340	0.03
Ds (pt>4)	435	0.04
Lc (pt>4)	3848	0.03
N cand per event – Analysis cuts		
D0 (pt>1)	0.41	0.0019
D+ (pt>2)	0.36	-
D* (pt>3)	0.25	-
Lc (pt>4)	0.95	-

- Picture may change in Run3:
  - D0/evt will drop given the improved spatial precision and tighter filtering cuts
  - Lc/evt and Ds/evt will increase because we will push the analyses down to low p_T

Analysis-by-analysis specific issues

Hadron spectra with vertexing: similar analysis procedure as in Run 2

- Potential disk space and CPU time issues → need of analysing signals with very low S/B that requires whole data sample → may need to add an intermediate step to keep analysis time reasonable (see next slides)
  - can consider an analysis-mode with pre-selected candidates as input, instead of current loop on events and loop on candidates
  - some event information needed: physics selection and pile-up flags?, primary vertex (can be stored "per-candidate"), possible recalibration of PID
  - need book-keeping for normalisation

D^0 (and D_s , Λ_c?) at p_T < 1 GeV (no vertexing):

- enormous background and number of candidates, but also less variables used.
- Need to use THn or THnSparse histograms and avoid running analysis many times.
- Ds->Pi+Phi and Lc->Pi+K0s: in case of modular AOD(see next slides) → use Phi and K0s candidates already reconstructed (in common with LF?)

Flow analyses:

- may need to run over whole sample many times to apply calibration/improvement to quantities related to whole event (above ones + e.g. possibility to recalculate Q-vector excluding daughters).

Analysis-by-analysis specific issues

Correlation analyses:

- in principle all tracks in the event are needed (including MFT tracklets)!
- Cannot avoid event loop, but can still try to perform analysis over objects with reduced information (note: <<1 candidate per event selected in most cases → no need info for all events) + need to perform analysis on mixed events

Current analysis procedure **(angular D-h correlations)**

- Task runs over the events and store in TTrees for each event with at least one trigger particle
  - Information of the trigger particles (D mesons)
  - Information of the associated particles (charged tracks)
  - Event taggers (period, orbit, BC)
- Total size «per entry»: **68 bytes** for D-meson, **44 bytes** for tracks (TTree compression reduces final output file size)
- Pb-Pb extrapolation for 100M events in 0-10%:
  - = **1.2 GB*fract.events w/ candidate D in PbPb/pPb** (cuts & pT dependent)
- Output file analysed on the grid with parallelised jobs (nested loops on trigger particles and tracks) → single event and mixed event analyses

	pp 2010	p-Pb 2016
# D	105k	115k
# tracks	4M	11.4M
Output size	60MB	170MB

p-Pb: Running time = 200d

Analysis-by-analysis specific issues

HF jets:

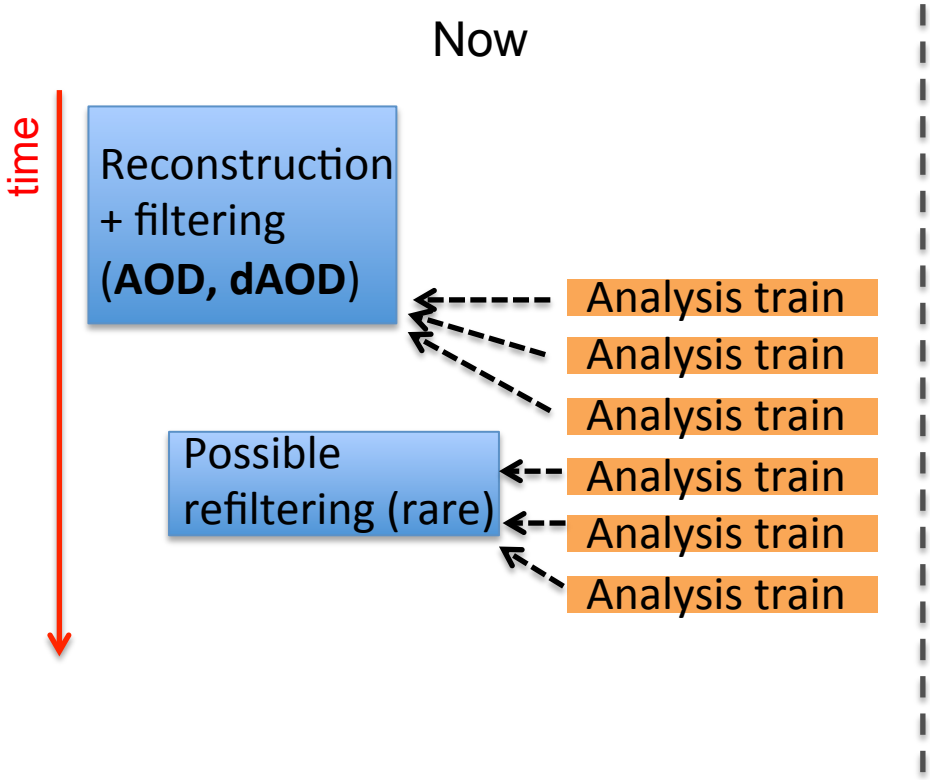
- similar to correlations but could be most delicate case since we may need to run the jet finder many times and may need to access information for each jet constituent

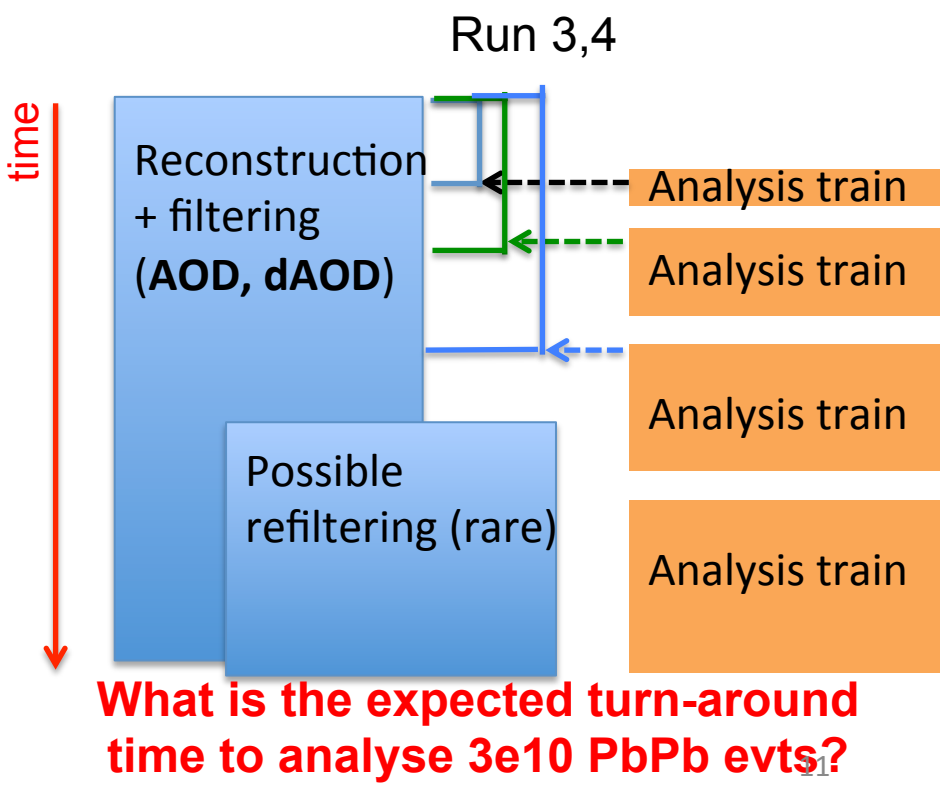
Possible change of analysis flow

- Improved spatial precision → less bkg → reduce disk and CPU "per-event"
- On the other hand, extend low pt reach "down to 0", new analyses with low S/B (Λ_c) →increase disk space and CPU time both at filtering and analysis level
- + number of events will be much larger (~ x100) and many analyses will need to inspect full stat

→ major concern: **risk that analysis time explodes?** Need proper estimates.

Addition of new intermediate step (next slide) could help.



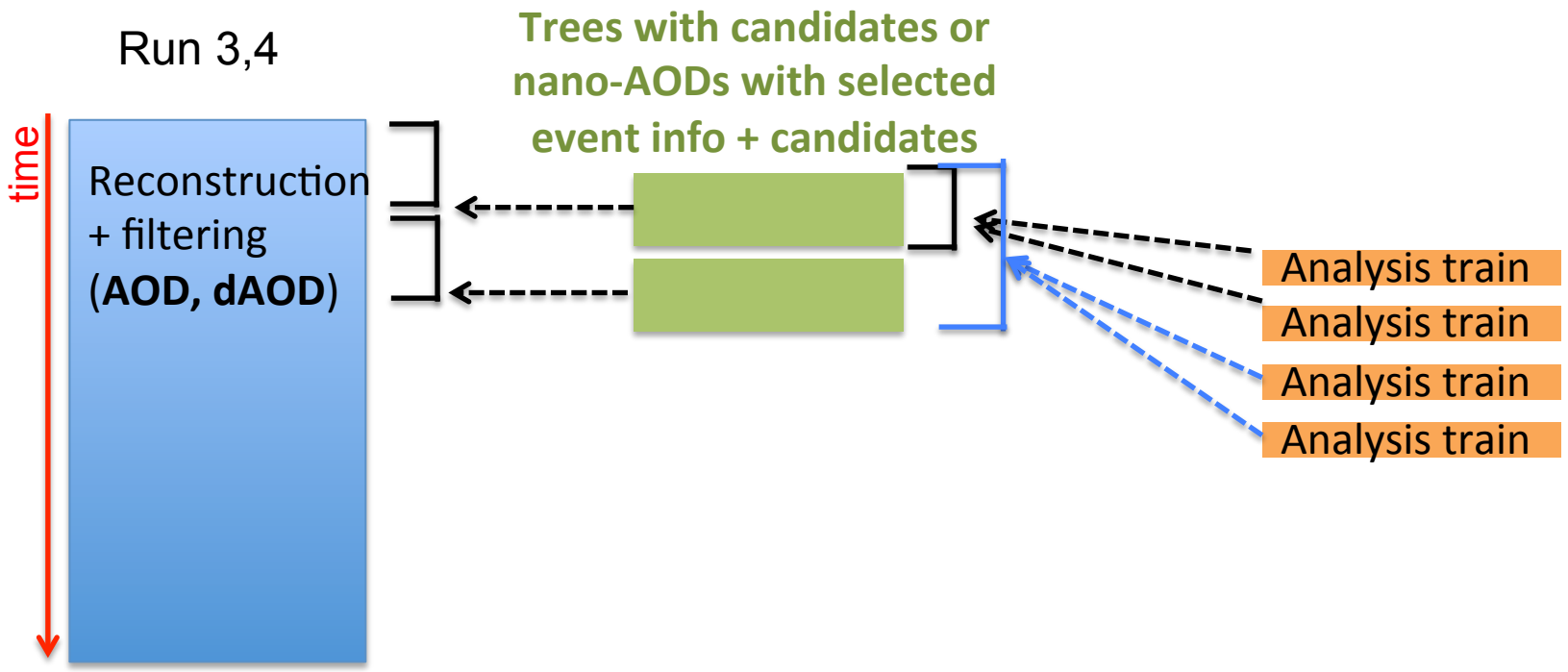


What is the expected turn-around time to analyse 3e10 PbPb evts?

Possible change of analysis flow

Main goal: keep analysis time relatively short, since analysis will need to be run many times with varied code, settings + allow for new analyses to be run.

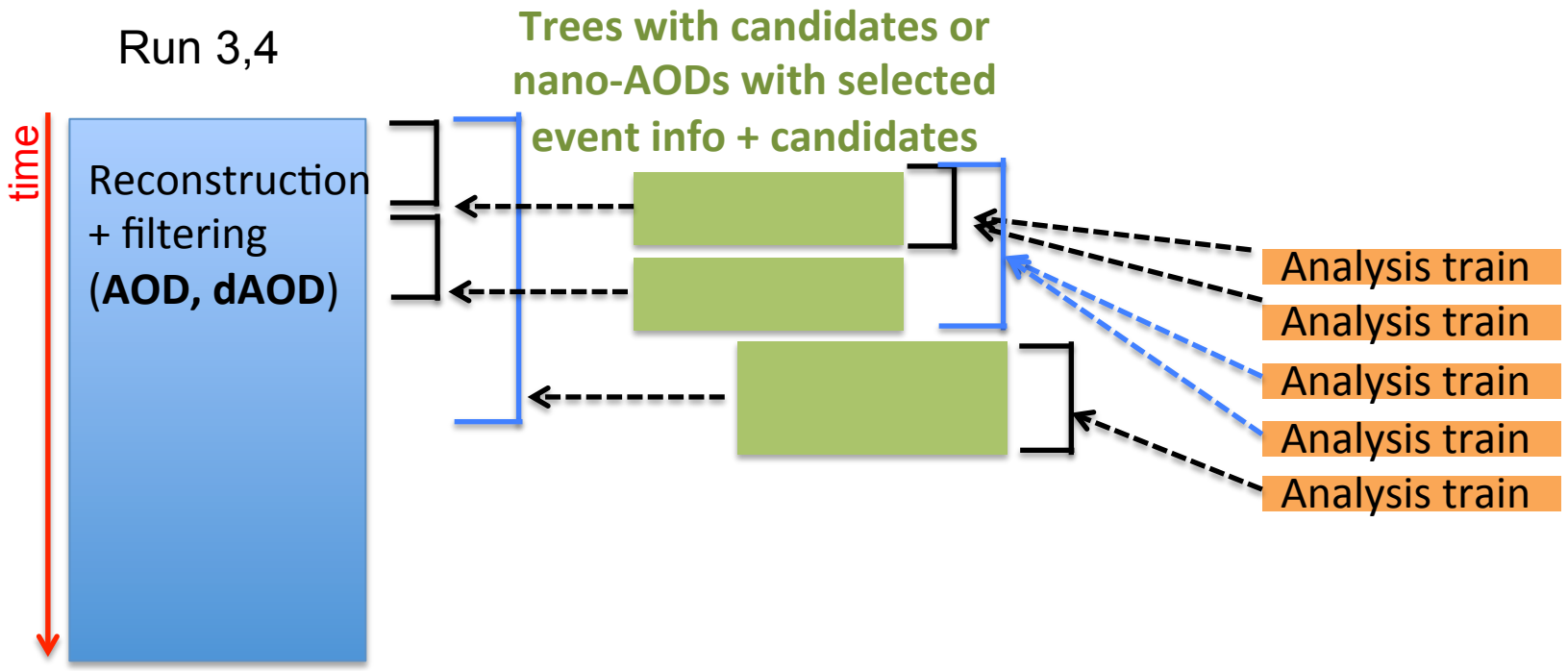
- We could write on **trees or "nano-AODs"** **including** **basic information needed by analysis.** These can be created regularly during data reconstruction, accessing sequentially bunches of data and then analysed in chain.



Possible change of analysis flow

Main goal: keep analysis time relatively short, since analysis will need to be run many times with varied code, settings + allow for new analyses to be run.

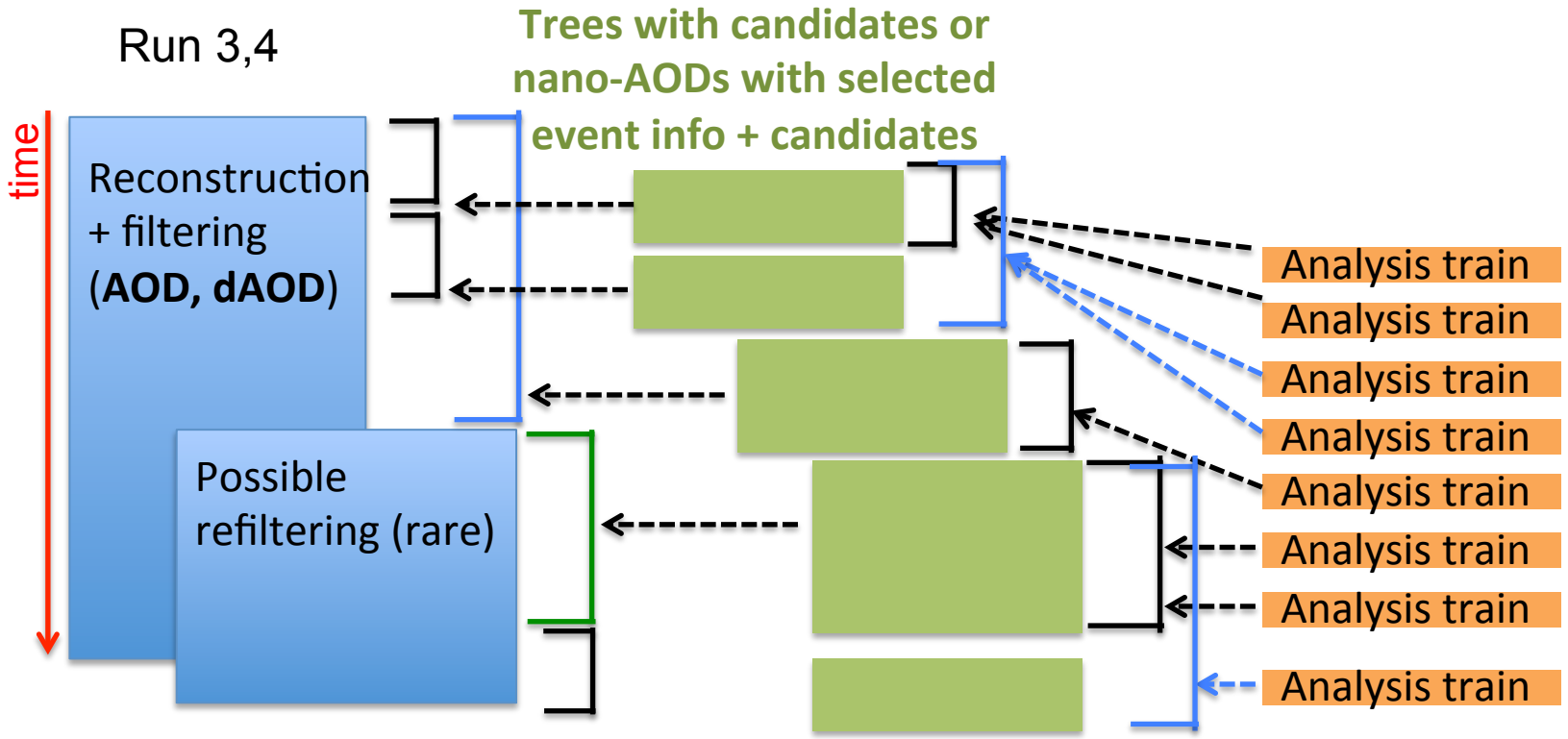
- We could write on **trees or "nano-AODs" including basic information needed by analysis.** These can be created regularly during data reconstruction, accessing sequentially bunches of data and then analysed in chain.
- If required by analyses, trees / nano-AODs can be re-produced with new settings.



Possible change of analysis flow

Main goal: keep analysis time relatively short, since analysis will need to be run many times with varied code, settings + allow for new analyses to be run.

- We could write on **trees or "nano-AODs"** **including** **basic information needed by analysis.** These can be created regularly during data reconstruction, accessing sequentially bunches of data and then analysed in chain.
- If required by analyses, trees / nano-AODs can be re-produced with new settings.
- In case of refiltering trees will be reproduced.
- Trees could be stored on the grid and analysed as current AOD.



Modular AODs (or nano-AODs)?

- Similar as current AOD+dAODs, but more flexibility and modularity?
- Tree of AOD events with friend trees that are connected and read on-demand
  - Tracks
  - Electron tracks (loose selection)?
  - ITS and MFT tracklets
  - V0s and cascades
  - HF hadrons
  - …

Analysis accesses only the friend trees that it needs: reduce I/O, however may increase number of files …

Modular AODs (or nano-AODs)?

- Similar as current AOD+dAODs, but more flexibility and modularity?
- Tree of AOD events with friend trees that are connected and read on-demand
  - Tracks
  - Electron tracks (loose selection)?
  - ITS and MFT tracklets
  - V0s and cascades
  - HF hadrons
  - …

HF hadron spectra or flow:

Modular AODs (or nano-AODs)?

- Similar as current AOD+dAODs, but more flexibility and modularity?
- Tree of AOD events with friend trees that are connected and read on-demand
  - Tracks
  - Electron tracks (loose selection)?
  - ITS and MFT tracklets
  - V0s and cascades
  - HF hadrons
  - …

HF hadron correlations with tracks:

Modular AODs (or nano-AODs)?

- Similar as current AOD+dAODs, but more flexibility and modularity?
- Tree of AOD events with friend trees that are connected and read on-demand
  - Tracks
  - Electron tracks (loose selection)?
  - ITS and MFT tracklets
  - V0s and cascades
  - HF hadrons
  - …

HF hadron correlations with (MFT) tracklets:

Modular AODs (or nano-AODs)?

- Similar as current AOD+dAODs, but more flexibility and modularity?
- Tree of AOD events with friend trees that are connected and read on-demand
  - Tracks
  - Electron tracks (loose selection)?
  - ITS and MFT tracklets
  - V0s and cascades
  - HF hadrons
  - …

HF hadrons in jets:

Backup

AOD input data used

- fHeader: most of its data member used
- fTracks
- fVertices (primary vertex and V0 vertices)
- fV0s (for Lc and Ds→V0+h analyses)
- fTracklets (mult. dep. analyses)
- fAODVZERO (mult. dep. analyses and EP determination)

How candidates are built

- AliAnalysisVertexingHF::**FindCandidate** → 2-prong example
  - Loop on positive tracks
    - Loop on negative tracks
      - **ReconstructSecondaryVertex**: secondary vertex reconstructed for each pair of tracks
      - If a vertex is found
        - » **Make2Prong**: creates AliAODRecoDecayHF2Prong object and save
          - TClonesArray of secondary vertices
          - TClonesArray of reco candidates
          - References → create correspondence between RD, daughters, secondary

Run1 pp, p-Pb, Pb-Pb and Run2 pp and p-Pb strategy
→ **New strategy** adopted for Run2 Pb-Pb to reduce dAOD size

Filtering Time

Events				Software versions					Job states							Timing	Output
										»						(done jobs only)	
Input	Processed	%	Filtered	AliDPG	ROOT	AliROOT	AliPhysics	Output directory	%	Total	Done	Active	Wait	Err.	Oth.	Running	Saving	Size
929,996,819	926,140,935	99.59%	628,750,033					LHC16k	99.62%	51065	50871	80	92	22	0	139d 9:52	1y 230d	48.5 TB
97,546,912	120,314,821	123.3%	92,105,962					LHC11h	96.56%	208915	201730	0	0	7185	0	122y 346d	7y 306d	191.3 TB
9,584,930	0	0%	0					LHC15o_lowIR_pass3_pidfix	99.86%	5883	5875	0	0	8	0	38d 15:00	29d 11:08	6.199 TB
139,446,055	0	0%	0					LHC15o_pass1_pidfix	99.09%	66471	65863	0	0	608	0	202d 9:38	4y 112d	67.78 TB

- Filtering time:
  - **Pb-Pb 2011**: 92M filtered events, **CPU running time 122y**, size 191TB (AOD + dAOD(all))
  - **Pb-Pb 2015**: 102M(?) filtered events, **CPU running time 202d**, size 68TB (AOD + dAOD(all))
  - **pp 2016**: 600M filtered events, **CPU running time: 151d**, size 48TB (AOD + dAOD(all))
- Run1 Pb-Pb vs. Run2 Pb-Pb: similar number of filtered events
  - Running time and AOD+dAOD size smaller for Run2 w.r.t. Run1
    - More central events in Run1 affecting the performance
    - Maybe different GRID resources available in 2011 and 2015

Impact of “re-filling” on Pb-Pb analysis

Standard dAODs

Summaries per site

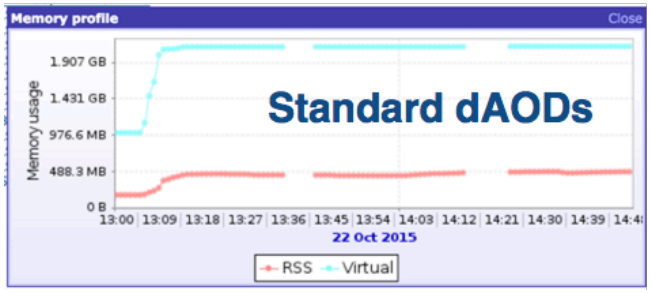
Site	Number of jobs Running	Saving	Done	Error	RSS Min	RSS Avg	RSS Max	Virtual Min	Virtual Avg	Virtual Max	Average time Running	Average time Saving	CPU Efficiency
ALICE::CERN::CERN-TRITON			5		274 MB	328 MB	361.5 MB	1.132 GB	1.527 GB	1.967 GB	33m 56s	1m 2s	12.83%
ALICE::CERN::CERN-ZENITH	1				401.9 MB	401.9 MB	401.9 MB	2.156 GB	2.156 GB	2.156 GB	2:28		24.88%
ALICE::CNAF::LCG			1		344.1 MB	344.1 MB	344.1 MB	1.918 GB	1.918 GB	1.918 GB	13m 9s	0m 48s	21.19%
ALICE::FZK::LCG			2		350.5 MB	351.1 MB	351.6 MB	1.331 GB	1.363 GB	1.395 GB	21m 24s	1m 12s	20.72%
ALICE::GRIF_IRFU::LCG			2		318.6 MB	343.2 MB	367.8 MB	942.6 MB	1.035 GB	1.149 GB	5m 38s	0m 43s	21.09%
ALICE::IHEP::LCG			1		292.7 M	[illegible]	92.7 MB	911.9 MB	[illegible]	1.9 MB	[illegible]	m 27s	18.35%
12 jobs on 6 sites	**1**		**11**		**274 MB**	**339 MB**	**401.9 MB**	**911.9 MB**	**1.45 GB**	**2.56 GB**	**32m 18s**	**0m 56s**	**18.94%**

NEW

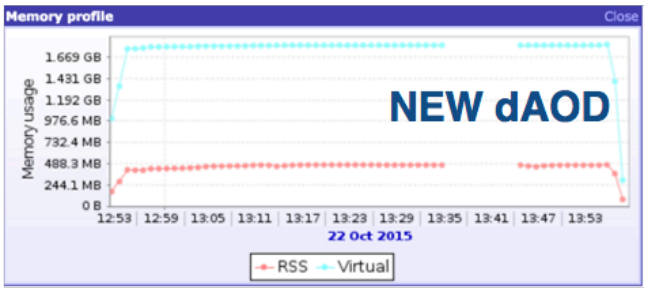
Summaries per site

Site	Number of jobs Running	Saving	Done	Error	RSS Min	RSS Avg	RSS Max	Virtual Min	Virtual Avg	Virtual Max	Average time Running	Average time Saving	CPU Efficiency
ALICE::CERN::CERN-TRITON			6		81.21 MB	300.8 MB	413.7 MB	294 MB	1.683 GB	2.343 GB	1:13	0m 59s	21.58%
ALICE::CERN::CERN-ZENITH			1		409 MB	409 MB	409 MB	2.523 GB	2.523 GB	2.523 GB	1:06	1m 20s	10.66%
ALICE::CNAF::LCG			1		329.3 MB	329.3 MB	329.3 MB	2.14 GB	2.14 GB	2.14 GB	16m 23s	1m 7s	17.69%
ALICE::GRIF_IPNO::LCG			1		226.9 MB	226.9 MB	226.9 MB	1.002 GB	1.002 GB	1.002 GB	1m 53s	0m 18s	20.69%
ALICE::GRIF_IRFU::LCG			2		262.3 MB	302.1 MB	341.9 MB	1003 MB	1.239 GB	1.497 GB	6m 5s	1m 2s	13.92%
ALICE::IHEP::LCG			1		341.8 M	341.8 MB	341.8 MB	1.151 GB	1.151 GB	1.151 G	2m 35s	0m 29s	20.56%
12 jobs on 6 sites			**12**		**81.21 M**	**309.7 MB**	**413.7 MB**	**294 MB**	**1.616 GB**	**2.523 G**	**44m 59s**	**0m 56s**	**19.94%**

Average: Rss 339 MB, VM 1.45 GB, Time 32’



Average: Rss 309 MB, VM 1.6 GB, Time 45’



Re-computing secondary vertices and candidates-related quantities does not increase the CPU time and memory usage

HFCJ – OFFLINE CORRELATIONS

Angular correlation of D-mesons and associated tracks

- While running the task over the events, store for each event, with at least a selected trigger, information of the triggers (D-mesons) and associated particles (charged tracks) in dedicated TTrees
- From the output .root file, correlation distributions can be build by performing nested loops on the triggers and tracks stored in the TTrees
  - By saving event taggers (period, orbit, BC) it's possible to perform single-event and mixed-event analyses running the task only once
  - Being the entries in the TTrees too many, the looping phase is performed on the grid with parallelized jobs
- Alternative approach to the standard one (used also for D-h, and for e-h analyses), which uses AliEventPool/AliEventPoolManager framework
  - The two approaches were proved to be fully equivalent
  - Avoids the usage of THnSparse containing correlation entries (which induce memory issues in merging phase), though the output size grows linearly with the statistics analyzed

STRUCTURE OF TTree

Inside the D-meson TTree

AliHFCorrelationBranchD

Eta	(Float_t)
Phi	(Float_t)
p_T	(Float_t)
M_{INV} (candidate)	(Float_t)
Event centrality	(Float_t)
Event $N_{tracklets}$	(Float_t)
z Vertex position	(Float_t)
Period,orbit,BC	(I/I/Ush.)
D-meson identifier	(Short_t)
D-meson selection	(Short_t)
Daughter 1,2 p_T (x,y,z)	(Float_t x 6)
Sel. mass hypothesis	(UShort_t)

Inside the track TTree

AliHFCorrelationBranchTr

Eta	(Float_t)
Phi	(Float_t)
p_T	(Float_t)
Event centrality	(Float_t)
Event $N_{tracklets}$	(Float_t)
z Vertex position	(Float_t)
Period,orbit,BC	(I/I/Ush.)
Track selection	(Short_t)
ID of mother trigger	(Short_t x 4)

- Members are needed to: build correlation distribution, tag the event, define the event pool for ME, associate daugther tracks to the parent trigger(s), tag soft pion tracks, apply multiple trigger and track selection

TYPICAL OUTPUT SIZE

- Total size «per entry»: **68 bytes** for D-meson, **44 bytes** for tracks
  - Note that the track TTree is filled much more times and dominates the output
- In reality, the TTree compression helps to reduce the final size of the output file
  - In addition, the size depends on the D-meson cut values and on the fraction of events with a selected D-meson candidate
- For pp 2010, on a run with with loose D-meson cuts, the output size was 60 MB (~0.2 byte per event on average, i.e. considering also events w/o D)
  - The real size without compression should have been of about 210 MB (4M tracks + 105k D mesons)
- For p-Pb 2016, cent-integrated, D^0-h analysis, the output size is 170 MB, the running time was about 200 days
  - The real size without compression should have been of about 501 MB (11,4M tracks + 115k D mesons)
- A very rough extrapolation for Pb-Pb (never tried running over) gives an increase of track TTree size (which shall still dominate) for 100M 0-10% PbPb events of:
  - $Nevts_{PbPb}/Nevts_{pPb}$ * $Npart_{PbPb}/Npart_{pPb}$ * fract.events w/ candidate D in PbPb/pPb = **1.2 GB*fract.events w/ candidate D in PbPb/pPb** (cuts & pT dependent)