ESD data volume.

Data compression - Motivation

- The data storage volume limited
- The data processing time on the GRID given by data volume
  - The CPU time negligible

New tool to check tree size in Aliroot

- New tool committed to the AliRoot -**MakeTreeStat**
- Macro to get the size of the Tree. As a improvement to tree->Print() function, this algorithm gives the size of all of the branches and in addition print them sorted according total tree size (MEMORY USAGE) if one event in tree or zip size (THE storage size on disk)
- Printed statistic:
  - 1. Order
  - 2. TotSize (in memory) + fraction of total size
  - 3. ZipSize (on disk) + fraction of zip size
  - 4. Compression ratio

Top User – Low multiplicity event (No - friends)

10	26620(0.90%)	337(0.91%)	1.25	.Kinks
9	29145(0.99%)	366(0.99%)	1.25	.CaloClusters
8	29685(1.01%)	379(1.03%)	1.25	.V0s
7	38639(1.31%)	482(1.31%)	1.25	.Cascades
6	164527(5.59%)	2074(5.62%)	1.25	.Tracks
5	62425(2.12%)	14794(40.11%)	23.70	AliESDFMDfEta.fEta.fData
4	67379(2.29%)	14857(40.28%)	22.05	AliESDFMD.fEta
3	2458974(83.57%)	16780(45.49%)	0.68	AliESDFMDfMultiplicity.fMultiplicity.fData
2	2464306(83.75%)	16848(45.68%)	0.68	AliESDFMD.fMultiplicity
1	2534444(86.13%)	31741(86.05%)	1.25	.AliESDFMD

0	2942435(100.00%)	36886(100.00%)	1.25	.esdTree

Data volume dominated by FMD

Top User – Low multiplicity event (With - friends)

14	29685(0.53%)	7159(0.53%)	3.82	.V0s
13	66673(1.18%)	7880(0.58%)	11.82	ESDfriend.ESDfriend.fTracks.ESDfriend.fTracks.fTRDindex[180]
12	38639(0.68%)	9315(0.69%)	4.09	.Cascades
11	62425(1.11%)	14794(1.09%)	23.70	AliESDFMDfEta.fEta.fData
10	67379(1.19%)	15989(1.18%)	23.73	AliESDFMD.fEta
9	2458974(43.55%)	16780(1.24%)	0.68	AliESDFMDfMultiplicity.fMultiplicity.fData
8	2464306(43.64%)	18066(1.34%)	0.73	AliESDFMD.fMultiplicity
7	59393(1.05%)	21393(1.58%)	36.02	ESDfriend.ESDfriend.fTracks.ESDfriend.fTracks.fTPCindex[160]
6	2534444(44.89%)	34720(2.57%)	1.56	.AliESDFMD
5	164641(2.92%)	39700(2.94%)	3.32	.Tracks
4	627217(11.11%)	382444(28.28%)	60.97	ESDfriend.ESDfriend.fTracks.ESDfriend.fTracks.fPoints
3	1939859(34.36%)	804919(59.52%)	41.49	ESDfriend.ESDfriend.fTracks.ESDfriend.fTracks.fCalibContainer
2	2702194(47.86%)	1218819(90.13%)	45.10	ESDfriend..ESDfriend.fTracks
1	2703806(47.89%)	1219207(90.16%)	23.95	.ESDfriend.
0	5646355(100.00%)	1352335(100.00%)	23.95	.esdTree

Data volume dominated by Friends
(mainly space points and tracks (calib container)

Top User – High multiplicity event (With - friends)

113	1171676(0.49%)	862692(0.77%)	73.63	Tracks.Tracks.fCp
12	1516982(0.63%)	1109913(0.99%)	73.17	Tracks.Tracks.fIp
11	1517024(0.63%)	1111332(0.99%)	73.26	Tracks.Tracks.fTPCInner
10	1250602(0.52%)	1144760(1.02%)	91.54	V0s.V0s.fParamP.fC[15]
9	1565660(0.65%)	1154401(1.03%)	73.73	Tracks.Tracks.fOp
8	9548939(3.99%)	1776899(1.58%)	18.61	ESDfriend.ESDfriend.fTracks.ESDfriend.fTracks.fTRDindex[180]
7	7551861(3.15%)	4514876(4.01%)	55.27	.V0s
6	8488059(3.54%)	5598771(4.98%)	65.96	ESDfriend.ESDfriend.fTracks.ESDfriend.fTracks.fTPCindex[160]
5	14353883(5.99%)	7882394(7.01%)	53.06	.Tracks
4	107063688(44.70%)	39111864(34.76%)	36.53	ESDfriend.ESDfriend.fTracks.ESDfriend.fTracks.fCalibContainer
3	90600184(37.82%)	53080712(47.18%)	58.59	ESDfriend.ESDfriend.fTracks.ESDfriend.fTracks.fPoints
2	216500480(90.38%)	99770528(88.68%)	46.08	ESDfriend..ESDfriend.fTracks
1	216502048(90.38%)	99771264(88.68%)	46.97	.ESDfriend.

0	239537408(100.00%)	112508224(100.00%)	46.97	.esdTree

Data volume dominated by Friends
(mainly space points and tracks (calib container)
Notice – ESD friend is fully loaded in memory (216 MBy)

Top User – High multiplicity event (No friends)

12	266065(1.16%)	256510(2.00%)	96.41	Tracks.Tracks.fP[5]
11	955712(4.15%)	313192(2.44%)	32.77	Tracks.Tracks.fTRDsignals[6][3]
10	417438(1.81%)	340063(2.65%)	81.46	V0s.V0s.fParamP.fP[5]
9	796509(3.46%)	769336(6.00%)	96.59	Tracks.Tracks.fC[15]
8	1250602(5.43%)	816025(6.37%)	65.25	V0s.V0s.fParamN.fC[15]
7	1171676(5.09%)	862692(6.73%)	73.63	Tracks.Tracks.fCp
6	1516982(6.59%)	1109913(8.66%)	73.17	Tracks.Tracks.fIp
5	1517024(6.59%)	1111332(8.67%)	73.26	Tracks.Tracks.fTPCInner
4	1250602(5.43%)	1144760(8.93%)	91.54	V0s.V0s.fParamP.fC[15]
3	1565660(6.80%)	1154401(9.00%)	73.73	Tracks.Tracks.fOp
2	7551861(32.78%)	4514876(35.22%)	55.53	.V0s
1	14353883(62.31%)	7887397(61.52%)	53.38	.Tracks
0	23035358(100.00%)	12820121(100.00%)	55.65	.esdTree

Data volume dominated by Tracks (61%) and V0s (35%).
Their data volume is dominated by size of Covariance matrix

Data compression

- The data volume is given by raw size of data and by compression factor.
- As we do not want to affect physics, we can try to improve compression factor.
- For compression we use ROOT zip algorithm (without knowing it).

- The compression factor is given by entropy of the data.
- We can make zip compression much more effective if we decrease the entropy.

Lossy - Data compression

- Determined by precision of measurement
- Precision of measurement:

  1)Absolute (e.g space point resolution in Pixel)

  2)Relative - given as fraction of value itself (e.g TPC dEdx resolution ~ 5 % of the value, chi2)

  3)Relative – given by external source - (e.g. Parameters and corresponding covariance matrix)

Lossy - Data compression (case 1)

- Absolute error ==> Fixed binning can be used
- Automatic Root support see: (ROOTSYS/tutorials/io/double32.C

```
// The following cases are supported for streaming a Double32_t type  -depending on the range declaration in the comment field of the data member:

//  A-    Double32_t     fNormal;

//  B-    Double32_t     fTemperature; //[0,100]

//  C-    Double32_t     fCharge;      //[-1,1,2]

//  D-    Double32_t     fVertex[3];   //[-30,30,10]

//  E     Int_t          fNsp;

//        Double32_t*    fPointValue;   //[fNsp][0,3]

// In case A fNormal is converted from a Double_t to a Float_t

// In case B fTemperature is converted to a 32 bit unsigned integer
```

Lossy compression – case 2

- Two ways to decrease the entropy

  1) Rounding floats to n significant bit (User defined action) and use standard root zip compression

  2) Decompose number to exponent part and mantissa (mantissa with nbits precission)

     - y = x1*2^x0
     - Two separate branches of data - different distribution – smaller entropy – better compression

- Entropy of exponent part ~ 2 bits

- Entropy of mantissa given by number of used bits

Rounding of the data to n significant digits currently implemented in the ROOT on ALICE request
Double32_t and Float_t 16_t

Compression – case 2

- Compression for different rounding (number of bits)
  - Relative precision 1/(sqrt(12)*2^n)
  - Float comp. Exp comp. mantissa Two branch comp.

Round B1	ratio=6.984087 exp	15.680983	cexp	20.711170	val	19.667401	ratio=10.087910
Round B2	ratio=6.577776 exp	15.291545	cexp	20.710902	val	16.052185	ratio=9.043191
Round B3	ratio=5.881972 exp	14.784046	cexp	20.710902	val	12.713787	ratio=7.877837
Round B4	ratio=5.358619 exp	14.303521	cexp	20.710902	val	10.645165	ratio=7.031212
Round B5	ratio=4.487309 exp	14.093820	cexp	20.711438	val	8.766834	ratio=6.159584
Round B6	ratio=3.664475 exp	14.847855	cexp	20.711438	val	7.353830	ratio=5.426939
Round B7	ratio=3.350739 exp	16.372729	cexp	20.711438	val	6.416667	ratio=4.898922
Round B8	ratio=3.249082 exp	16.429229	cexp	20.710097	val	5.988124	ratio=4.645055
Round B9	ratio=3.127902 exp	16.428048	cexp	20.709561	val	5.494302	ratio=4.342285
Round B10	ratio=2.995122 exp	16.437840	cexp	20.706344	val	5.162591	ratio=4.132309

Compression – case 3

- Rounding of value according precision given by other variable
- Difficult to make (simple) automatic schema
- Preferred solution => User defined rounding function called before storing data (e.g CleanESD)

ESD compression

- Different variables correspond to different cases (1..3)
  - case 1 – normalized PID
  - case 2 - Covariance, chi2
  - case 3 - Track Parameters
- Preferred solution
  - Use data compression which is (back, forward) compatible
  - Try different solution before making incompatible changes

ESD compression

- The other critical part – number of V0s in high multiplicity environment
  - ~ 30% of data volume
- The data volume reduction based on chi2 to be implemented soon (AliKF*)
  - ? Should we use also pointing to the primary vertex
  - Cascades?
- The criteria to remove tracks and V0
  - Should be setupable
  - AliESDRecoParam as equivalent of AliTPCRecoParam

Conclusion

- The automatic tool to check the esd size developed
  - Indicates critical part
- The data volume of ESD can be reduced by factor ~ 2-3
- The biggest fraction of the data volume correspond to case 2 and 3 where ROOT IO support was implemented only currently
  - The version will be available soon for AliRoot