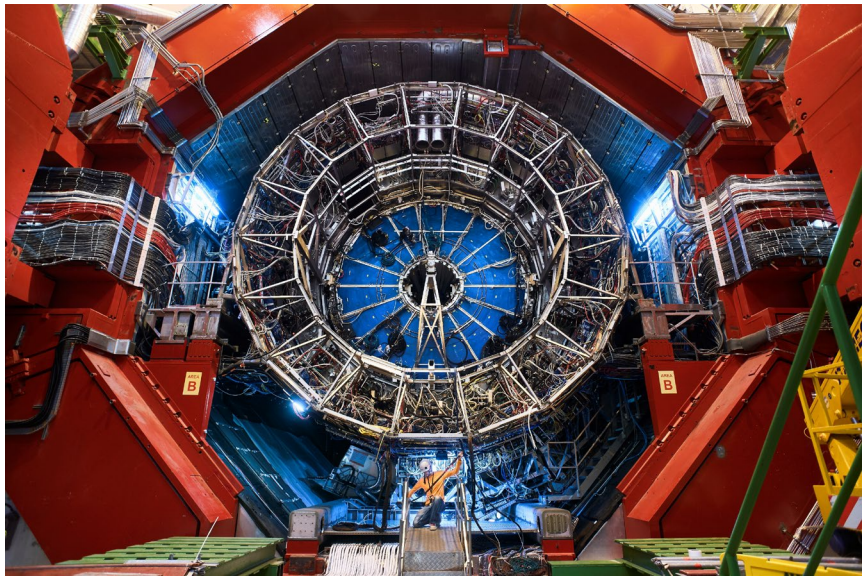
Fast Entropy Coding using rANS for ALICE run 3

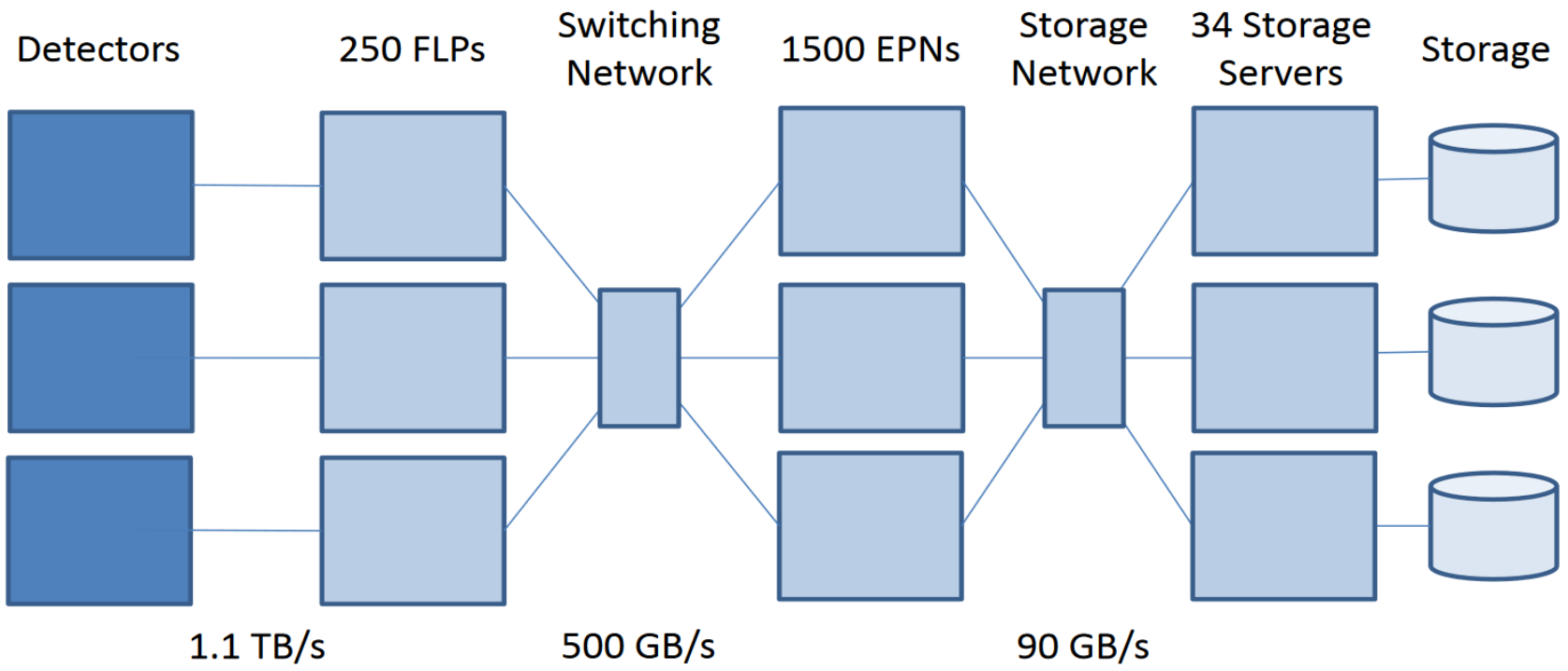
Michael Lettrich (CERN, Technische Universität München) for the benefit of the ALICE collaboration

ALICE O^2 Software Framework

- Software Framework for ALICE after LS 2
- Tasks:
  - Reconstruct and calibrate raw detector data
  - Detector Simulation (Monte Carlo) and reconstruction of simulated data
  - Analysis of reconstructed data on the WLCG (Worldwide LHC Computing Grid)



ALICE Raw Data Flow in Run 3



Compression Workflow in O²

- Raw Data from Continous Readout as timeframes of 20ms
- Data reduction steps on EPN using CPU + GPU
  - Timeframes are independant – no communication needed
  - Processing time per timeframe: 30s
  - Result: flat structures of arrays
- Entropy coding after Data Reduction:
  - Desired compression factor: 3
  - Time limit: 35s

Structure of Compressed TPC Data

- ALICE TPC data responsible for 92% of data volume
- Structure of source data:
  - Flat structure of 22 arrays describing 6 different objects
  - Data Range of 8 – 25 Bits/value

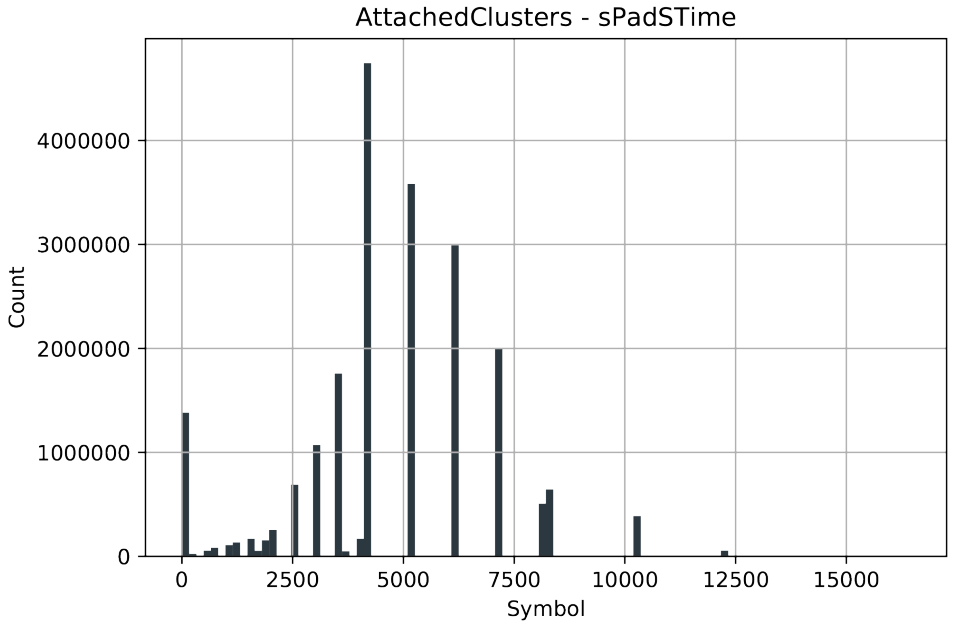
ALICE Timeframe

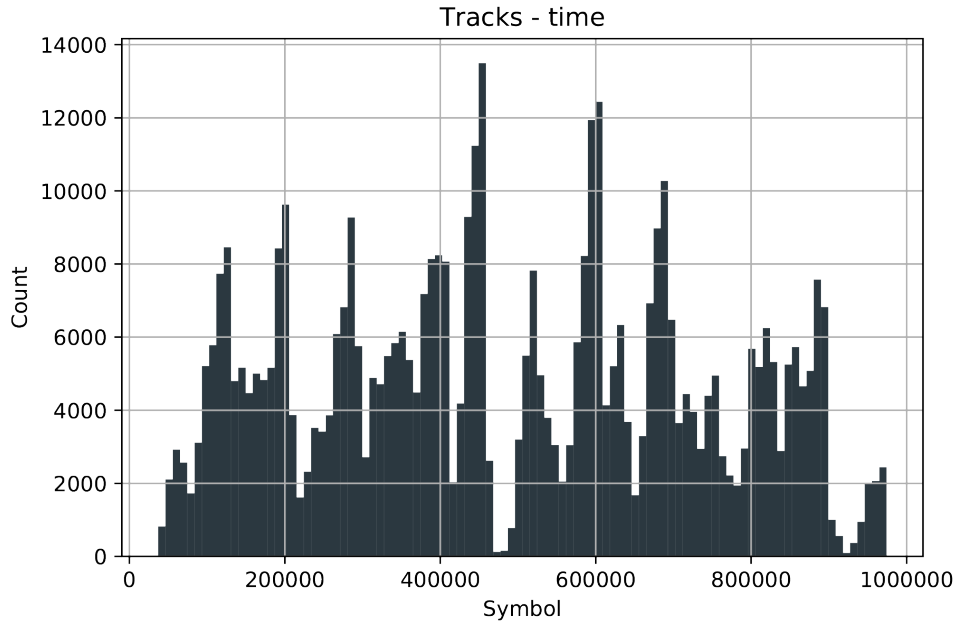
Analysis of Compressed TPC Data

- Compressed TPC data from 130 Pb-Pb events from MC simulation with O(10^7) samples
- Preprocessing: Concatenate Correlated Fields to reduce entropy
- Distributions are similar to run 2

Object	Count	Fields	Bits/Obj	H [Bits/Obj]	H concat
Attached Clusters	21072849	4	41	17.15	15.75
Attached Clusters Reduced	20590430	4	55	17.60	17.59
Tracks	482419	5	73	53.90	53.90
Unattached Clusters	50745911	5	81	39.77	38.37

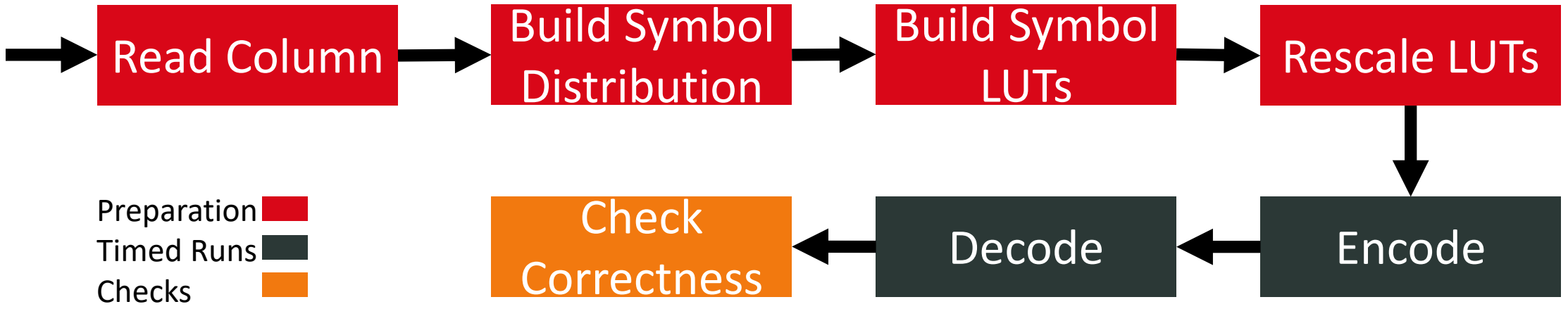
Structure of Compressed TPC Data





- Sparse non standard distributions of source symbols

Benchmarking

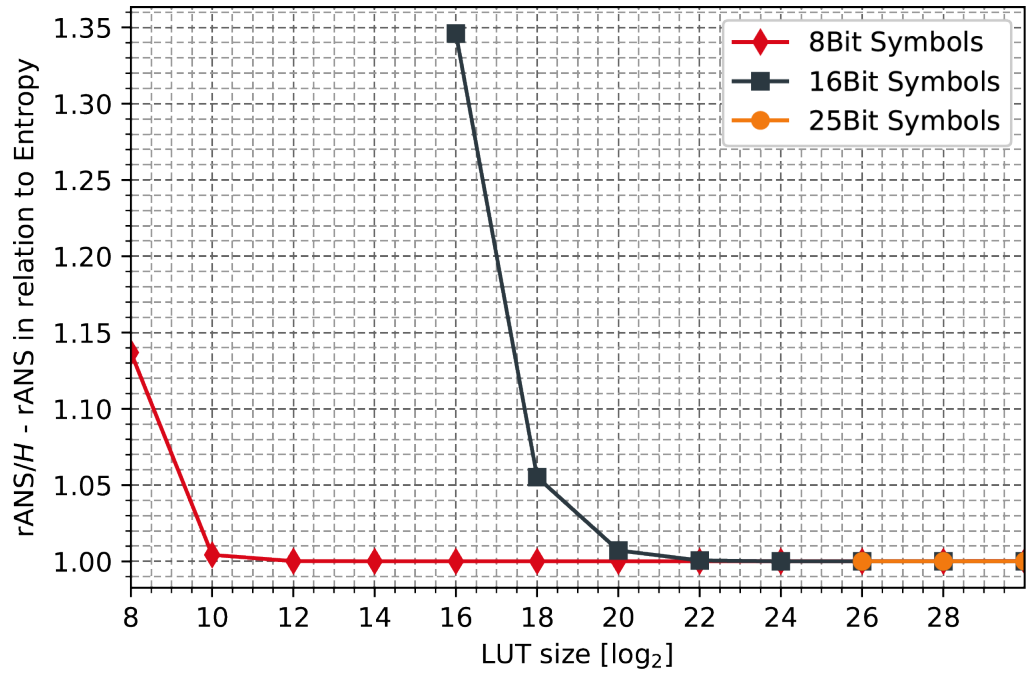


Results with rANS

- C++ Code based on reengineered ryg_rans coder/decoder
- Each array encoded with own LUT
- Bandwidth: Single Thread, avg over 5 runs on Core i7 8700

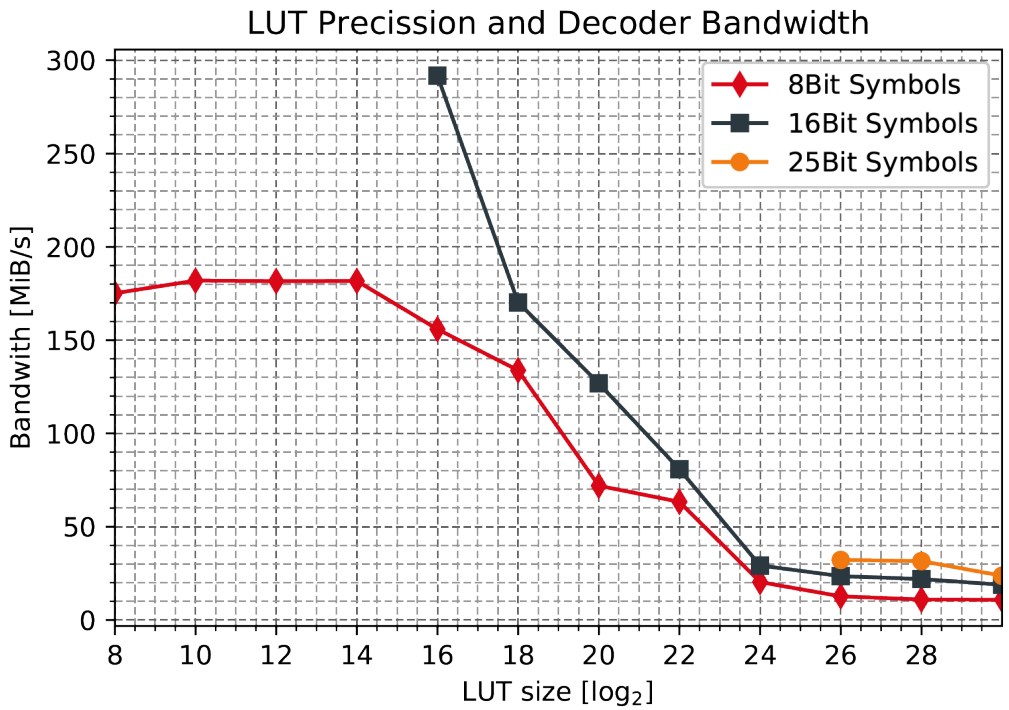
Object	Bits/Obj	H [Bits/Obj]	rANS [Bits/Obj]	rANS/H	BW [MiB/s]
Attached Clusters	41	15.75	15.75	1.00	649.72
Attached Clusters Reduced	55	17.59	17.64	1.00	776.06
Tracks	73	53.90	53.90	1.00	365.56
Unattached Clusters	81	38.37	38.37	1.00	589.80

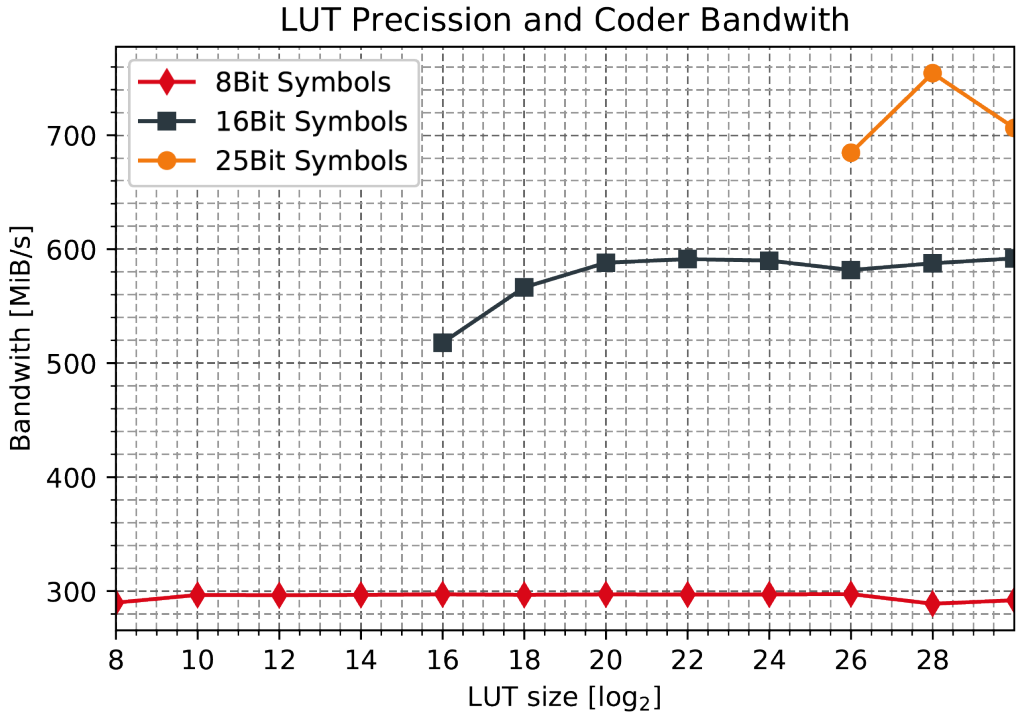
Compression and LUT precision



- Rescaling LUT to 2^p for perfromance.
- LUT must map source distribution well enough for good compression.
- Trivialy: , i.e larger LUTs for larger alphabets.
- Must be preserved for decoding.
- Additional Inverse LUT (iLUT) for decoder |iLUT|≥ |SourceAlphabet|
- For large source alphabets only sensible for static distributions, where LUTs can be reused or for extremely large datasets.
- Algorithmic improvements required to decrease memory consumption

Bandwidth Measurements





- Decoder bandwith decreases with increasing LUT size.
- Performance bound by lookups in iLUT that does not fit into cache.

- Coder bandwidth grows with source alphabet.
- Same operations executed on larger input data (16Bit ~2x faster then 8Bit).

Outlook

- Good compression results encourage usage of rANS instead of Huffman
- To be done:
  - Integration in ALICE O^2
  - Datastructure optimizations for iLUT (trees?)
  - Deal with uncompressible data with almost zero probability
  - GPU implementation (academic?)

Thanks for your attention

Questions?