Quality Assurance and Developments on Time-Based Tracking

Jenny Regina

Uppsala University
Department of Physics and Astronomy

Online PANDA Collaboration Meeting
March 9-13, 2020
Computing Session

Outline:

1. **Tracking Quality Assurance**
2. **Time-based data storage in ROOT**
3. **Time-based tracking Results**
4. **Current Developments**

What is tracking Quality Assurance?

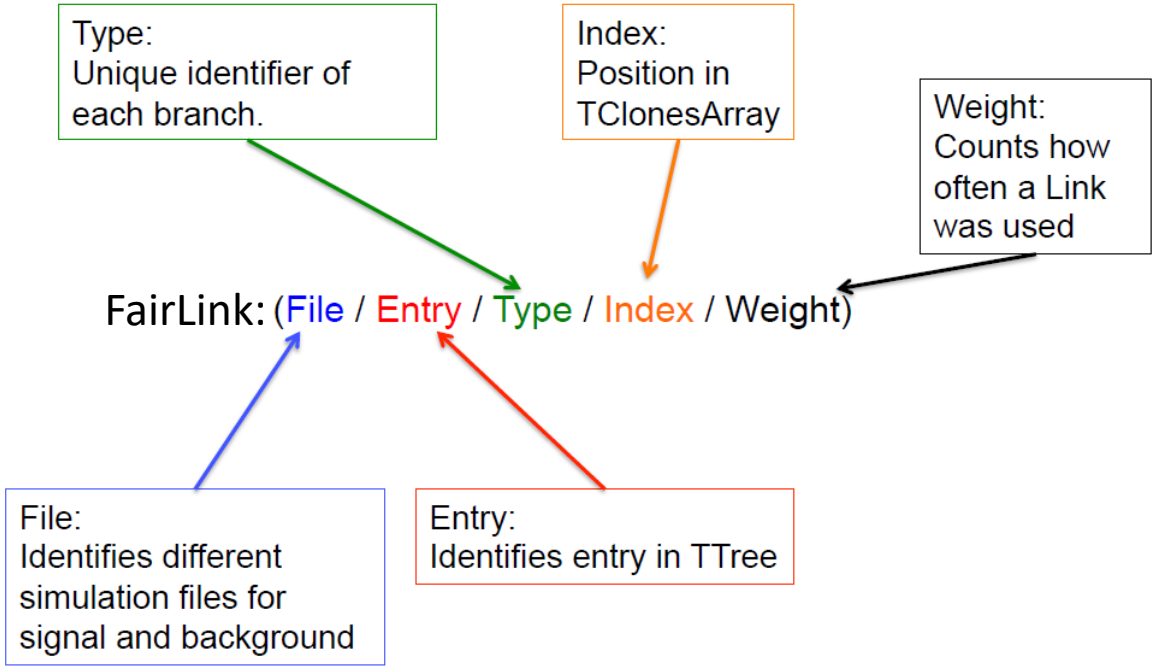
- **Q**uality **A**ssurance (QA) for tracking algorithms
- A track can be:
  - *Fully Purely found* (all hits found)
  - *Fully Impurely found* (all hits from one track found, contamination possible as long as hits from other track does not exceed 70% of all hits in reco track)
  - *Partially Purely found* (majority of found hits belong to same MC track, contamination not possible)
  - *Partially Impurely found* (>70% of all hits belong to one track, contamination possible)
  - *Clones* (one track was found more than once)
  - *Ghosts* (reco track does not correspond to a MC track)
- Momentum resolutions
- Number of *true*, *false* and *missing* hits / track

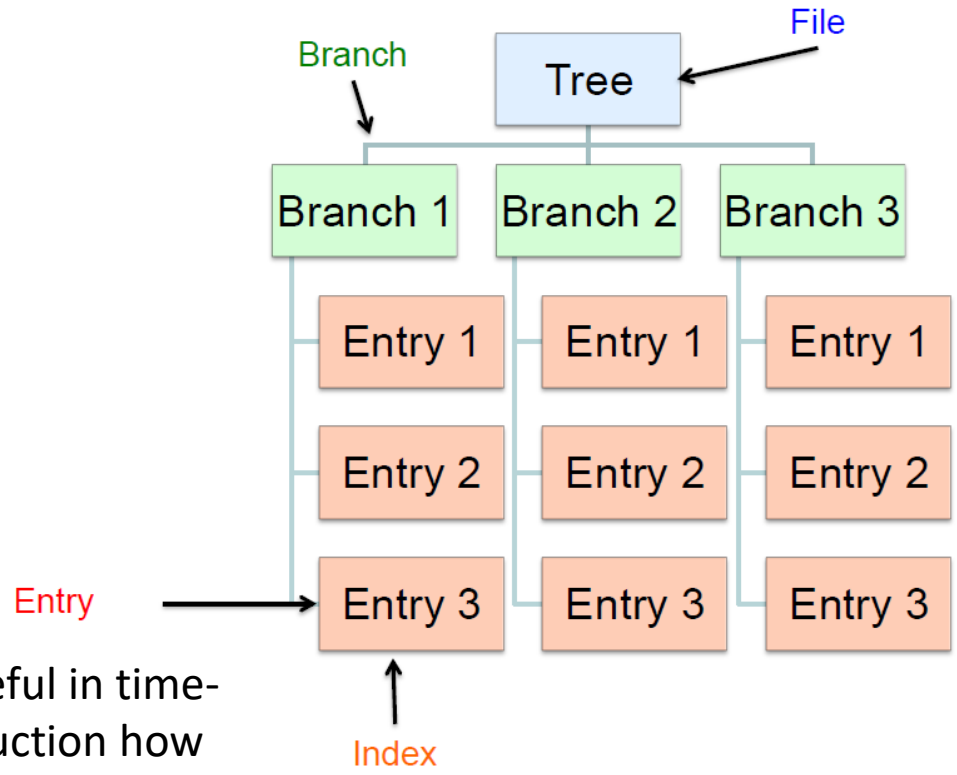
What is tracking Quality Assurance?

- To give “fair” representation of track finding algorithm: compare to only tracks which could be reconstructed in relevant detectors
- Use *IdealTrackFinder* with certain functor [1]
  - *E.g.* only save tracks with more than 5 STT hits, 6 STT+MVD+GEM hits …

[1] Lets you create objects which look like function

Event Based vs. Time Based Data Storage





Event-based data storage:

Branch: data objects, *eg.* Hits, tracks

Entry: event number

Index: Position in TClonesArray, in event-based simulation same for MC tracks and IdealTracks

Need to be careful in time-based reconstruction how this is done!

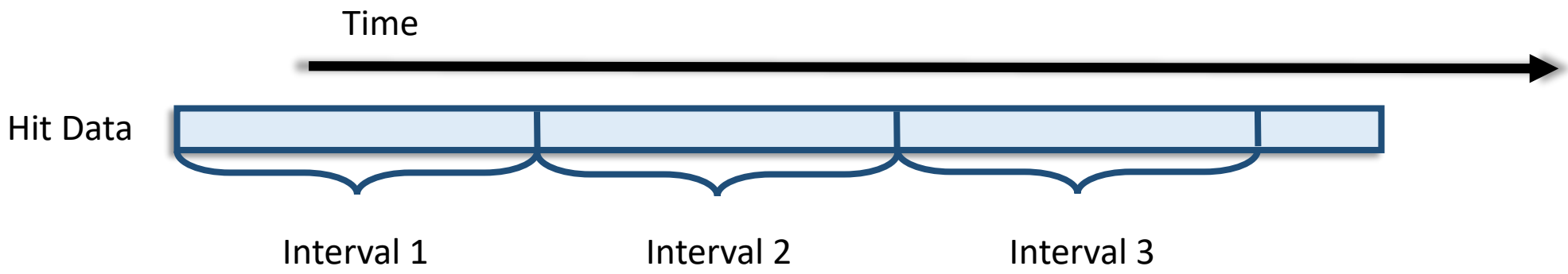
Not the case for time based reconstruction!

Fetching the Time-Based Data

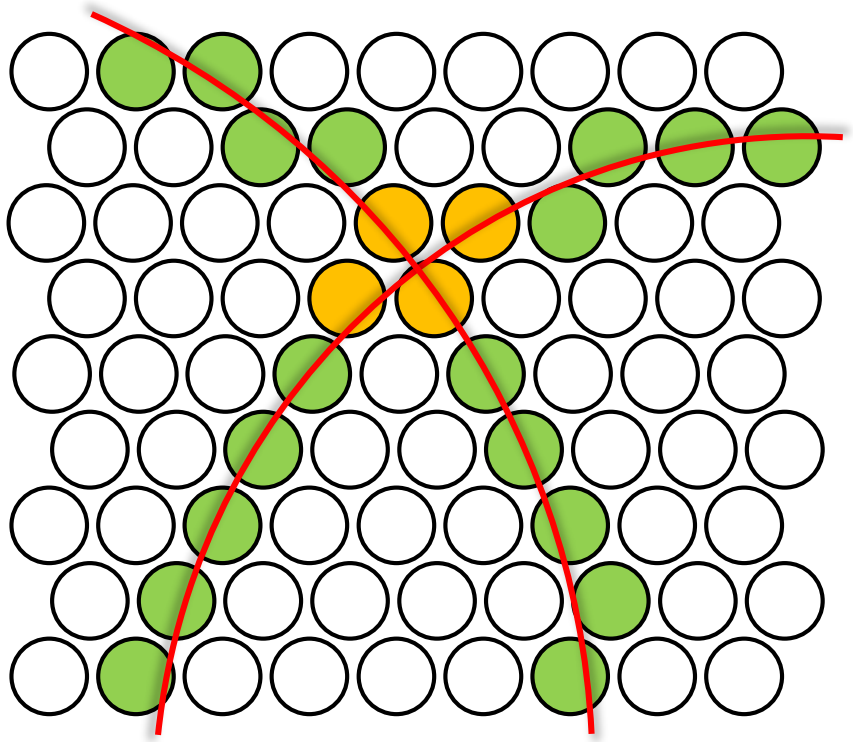
- Overwrite *Exec()* function with *GetData()* fetching certain data
- *E.g.* StopTimeFunctor
  - Fetches data from within a certain time-interval
  - Can only fetch data once! Skips data already collected

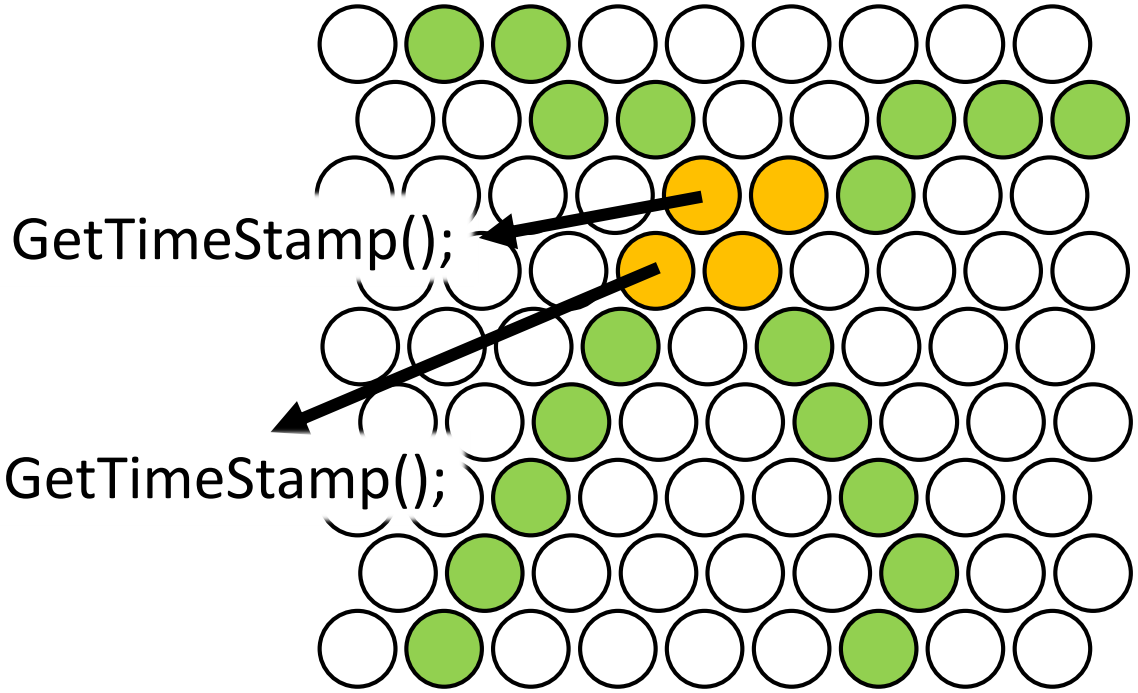
2000 ns in my tests

FairRootManager::Instance() →GetData(fHitBranch, fFunctor, fStopTimeValue);



Time Clustering in SttCellTrackFinder





1. Ask every hit for its time stamp
2. Compare it to time stamps of its spatial neighbors
3. if Δ time stamp < fClusterTime Neighbors accepted

fClusterTime=250; // [ns]

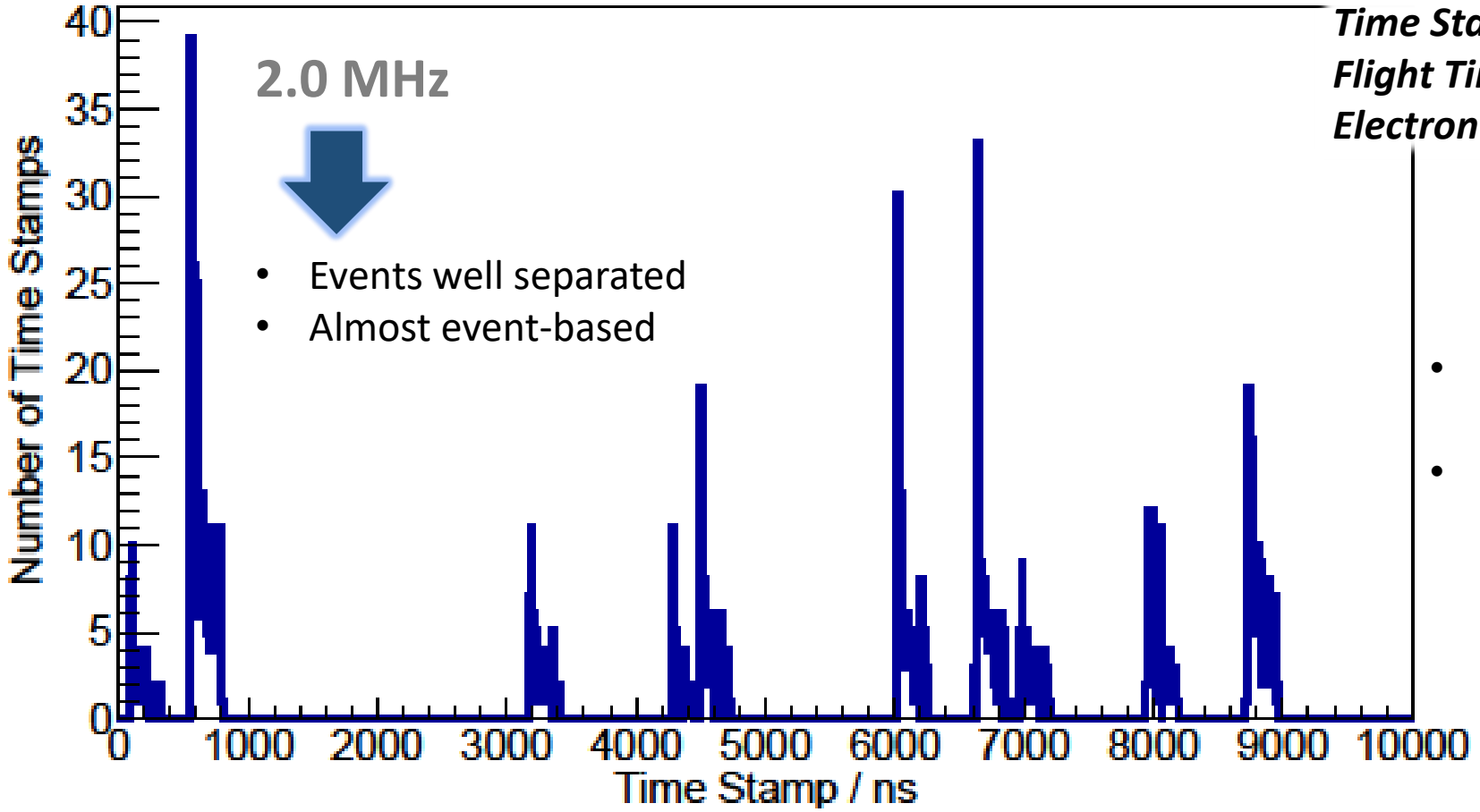
Simulation approach

- FairRoot 18.2.0
- FairSoft jun19
- PandaRoot Development Branch
- Digitization: PndSttHitProducerRealFull
- Reconstruction: event-based SttCellTrackFinder [2], and time-based version [3]
- Perform tracking at 4 different event rates, 0.5, 1, 2 and 4 MHz

[2] Jette Shumann, *Entwicklung eines schnellen Algorithmus zur Suche von Teilchenspuren im "Straw Tube Tracker" des PANDA-Detectors*, Bachelor Thesis, 2013

[3] Jenny Regina, Walter Ikegami Andersson, *Time-based Reconstruction of Hyperons at PANDA at FAIR*, 2019, Conference Proceedings, CDT/WIT2019. e-Print: arXiv:1910.06086v1

Time Distribution



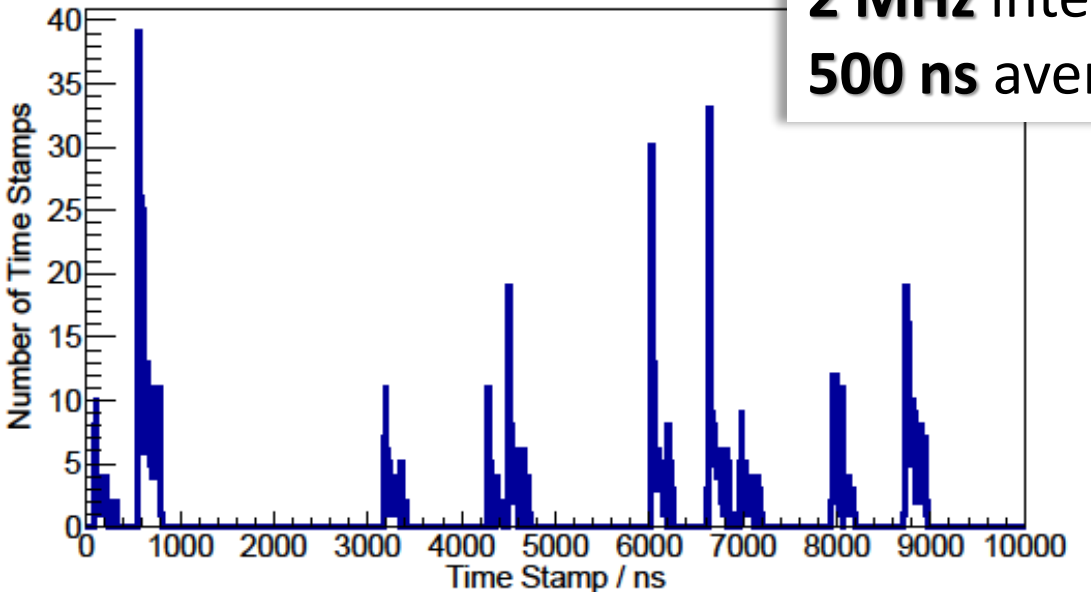
- Events well separated
- Almost event-based

***Time Stamp** = **Event Time** [100 ns] +*
***Flight Time of Particle** [0.001-0.01 ns*] +*
***Electron Drift Time** [100 ns]*

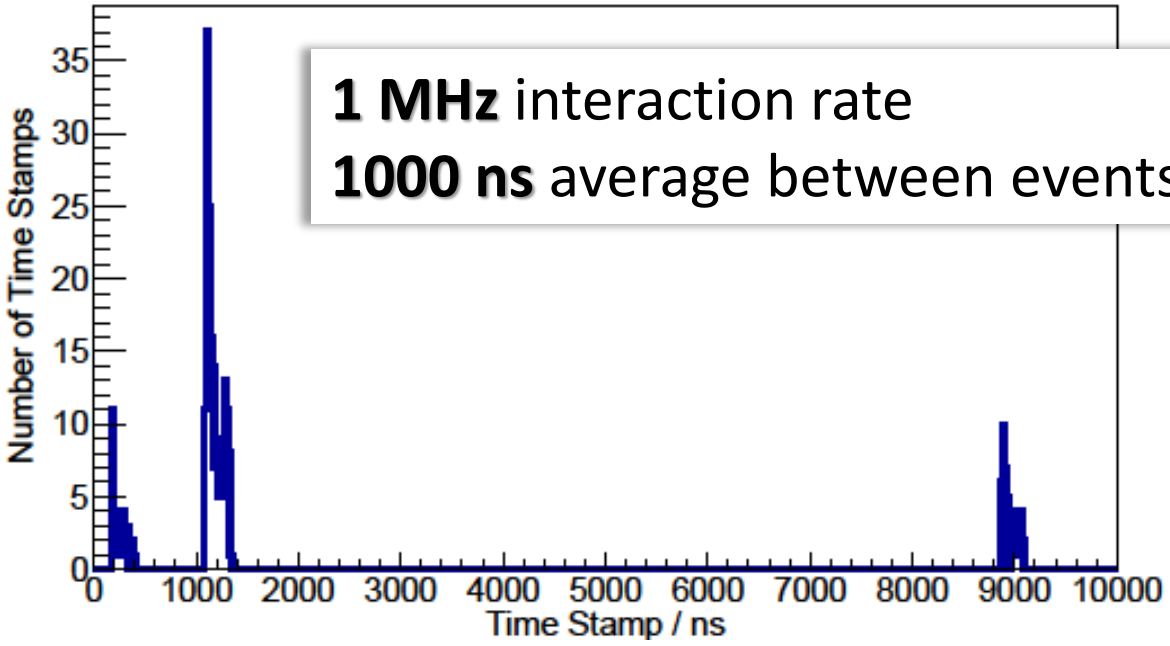
- Time-stamp dominated by Event time and electron drift time
- Flight time of particle negligible

*Order of magnitude for flight time of particle calculated as the time it takes a 1 GeV/c muon to travel between center of two straw tubes (1 cm)

Time Stamp Distribution



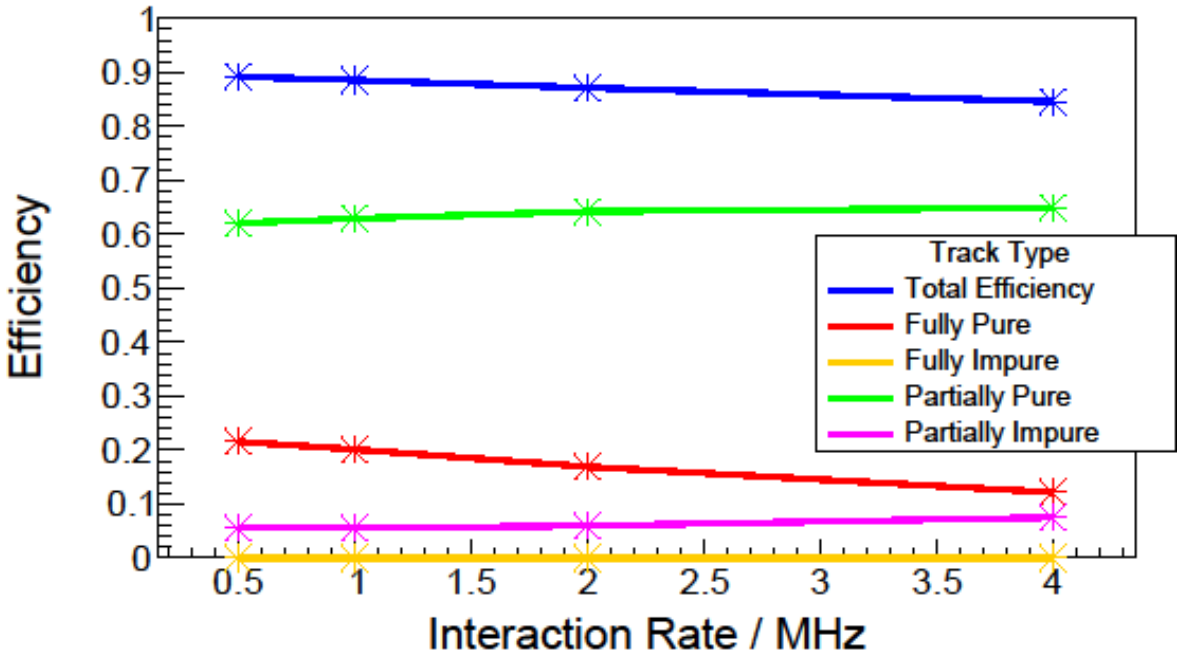
2 MHz interaction rate
500 ns average between events



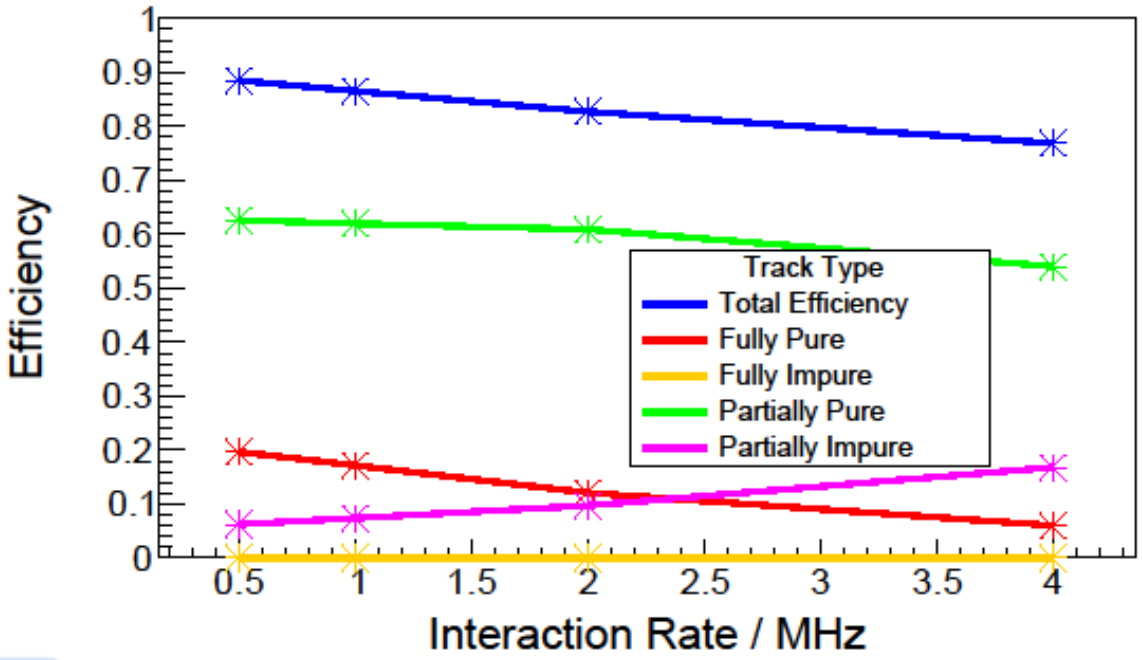
1 MHz interaction rate
1000 ns average between events

Efficiencies

With time-stamps



Without time-stamps



- Total efficiencies stable over relevant range of interaction rates
- At lower interaction rates time-stamps do not have dramatic effects
- At higher interaction rates time-stamps lead to higher efficiencies

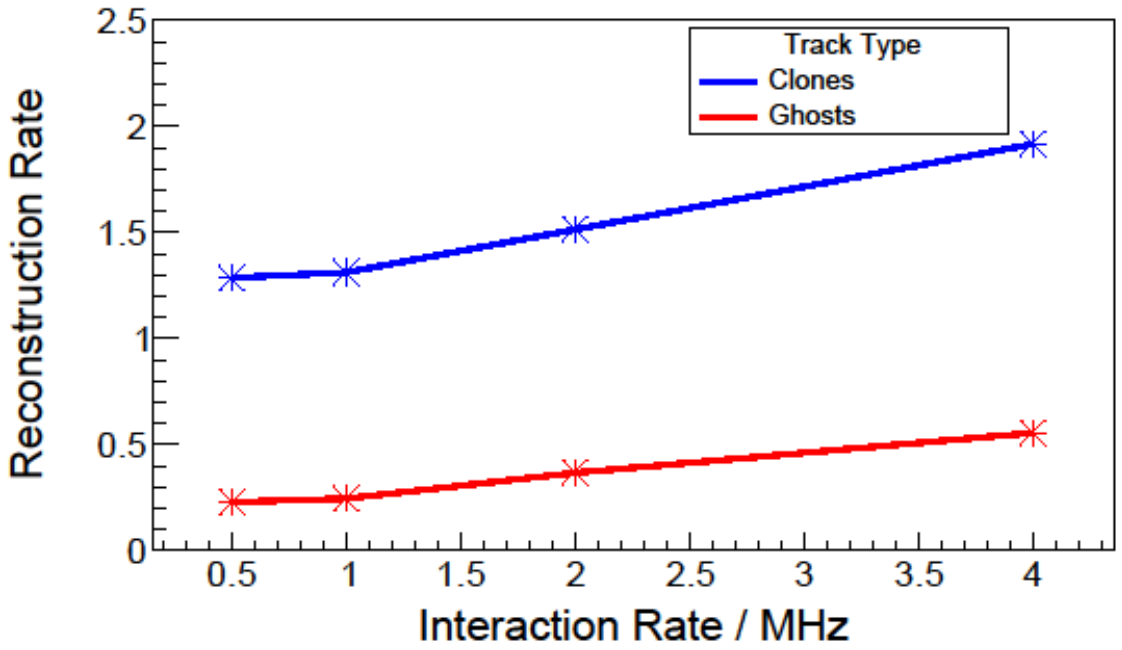
$n_{possible}$~ 2 670

Requirement: ≥ 6 STT hits

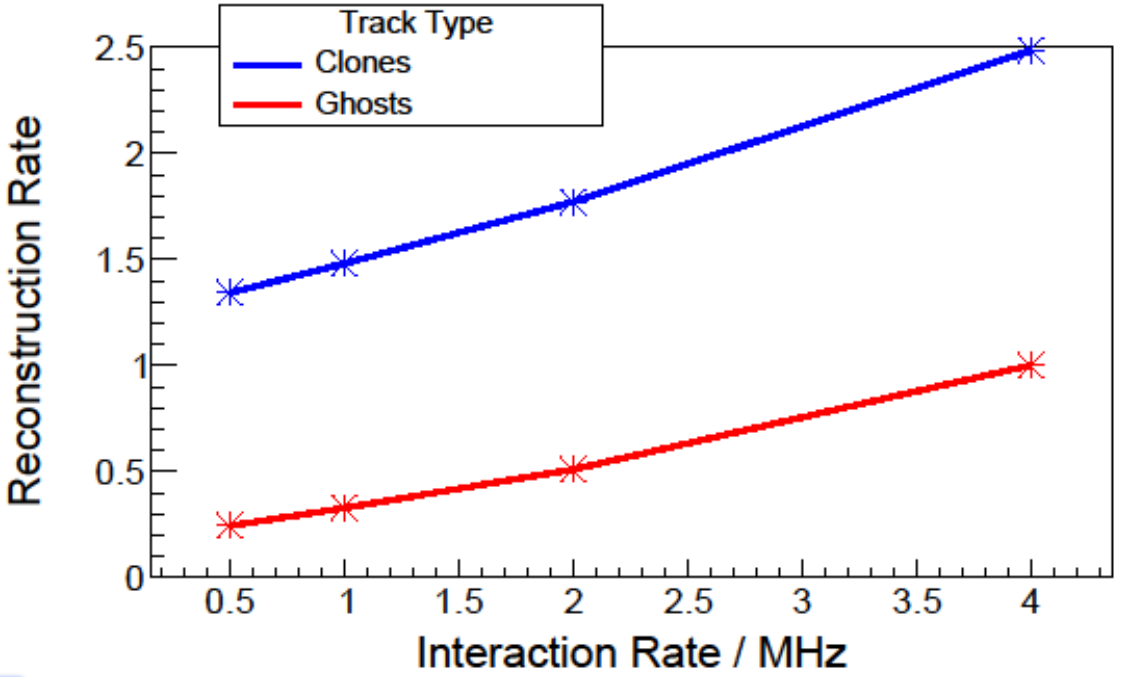
$$eff = \frac{n_{found}}{n_{possible}}$$

Fake Rate

With time-stamps



Without time-stamps



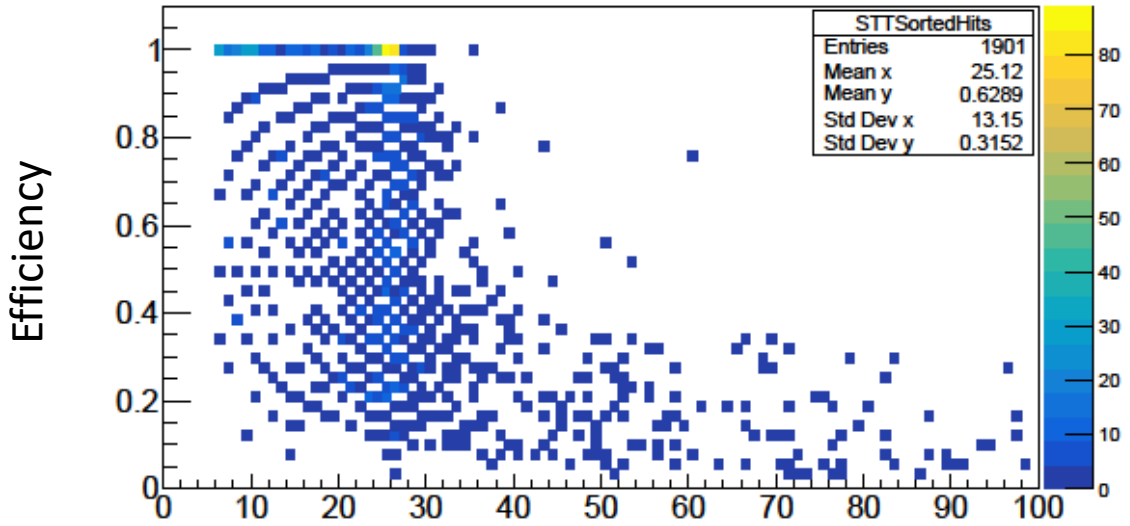
- Very high fake rates, especially clones at all interaction rates
- Increase in fake rate is more dramatic without time-stamp utilization

$$\text{Ghost Rate} = \frac{n_{ghosts}}{n_{possible}}$$

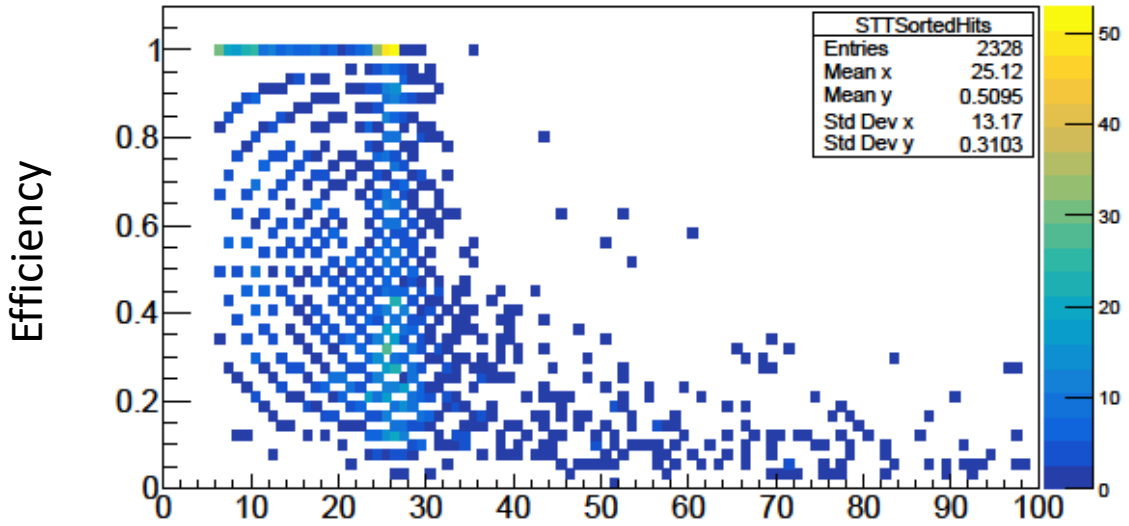
$$\text{Clone Rate} = \frac{n_{clones}}{n_{possible}}$$

Efficiencies

0.5 MHz

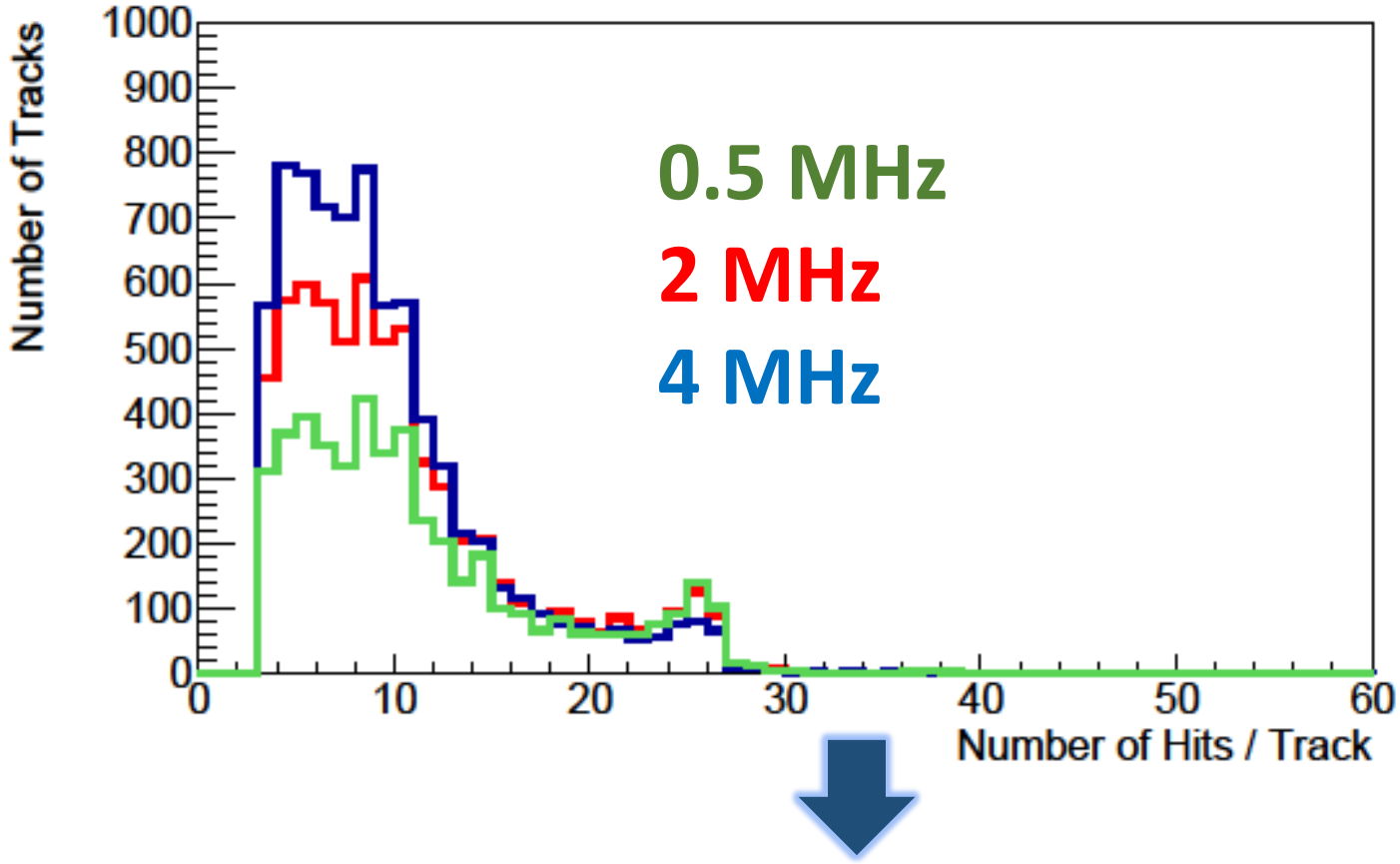


4.0 MHz



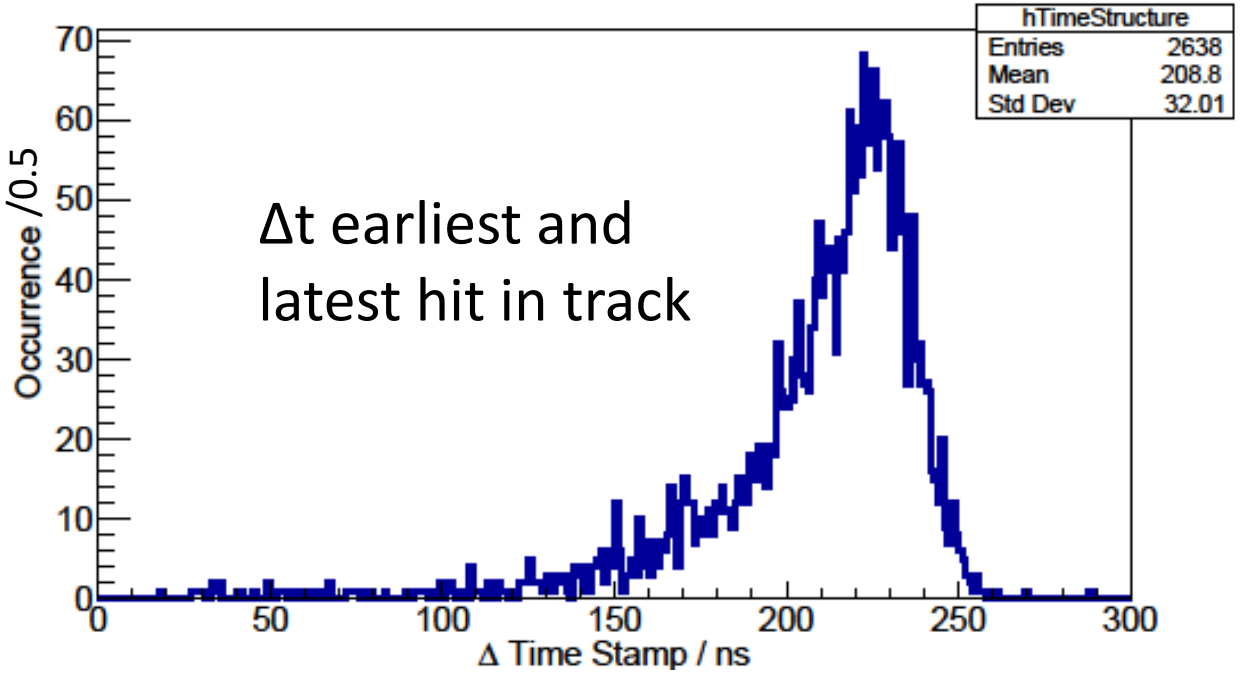
No dramatic effect, efficiency stable over the relevant range of interaction rate

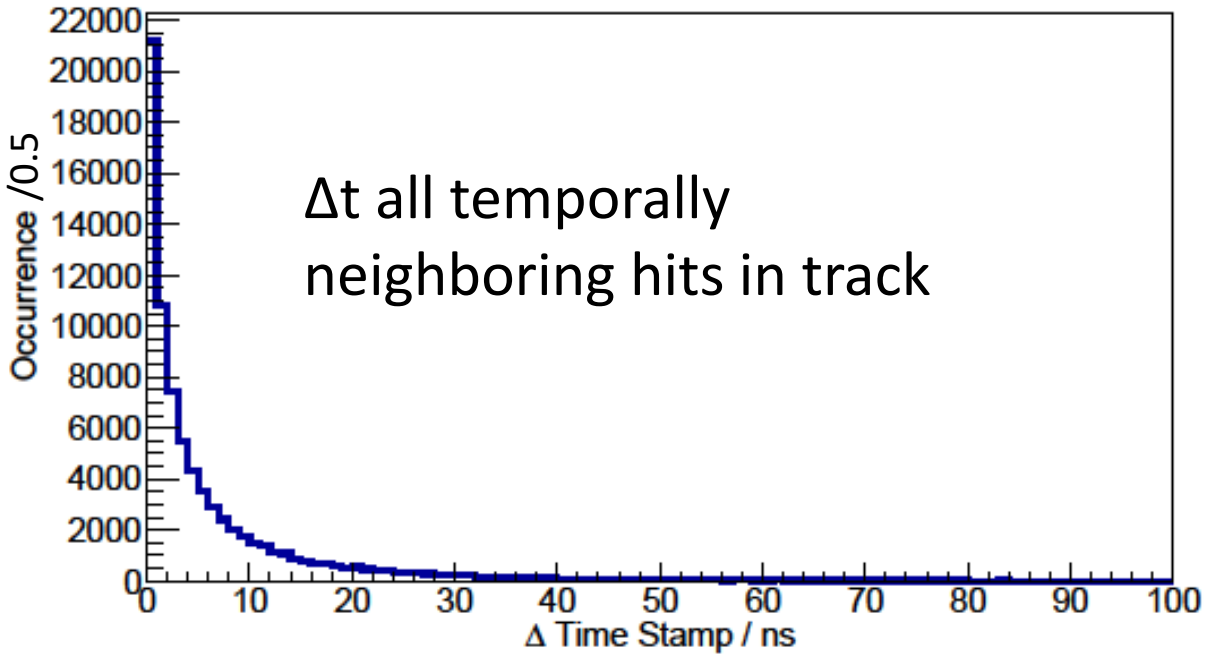
Hits per Track



- Pattern resembles characteristic STT hit distribution at all interaction rates
- Tracks reconstructed in shorter tracklets at higher interaction rates
- More tracks at higher interaction rates

Time Stamps

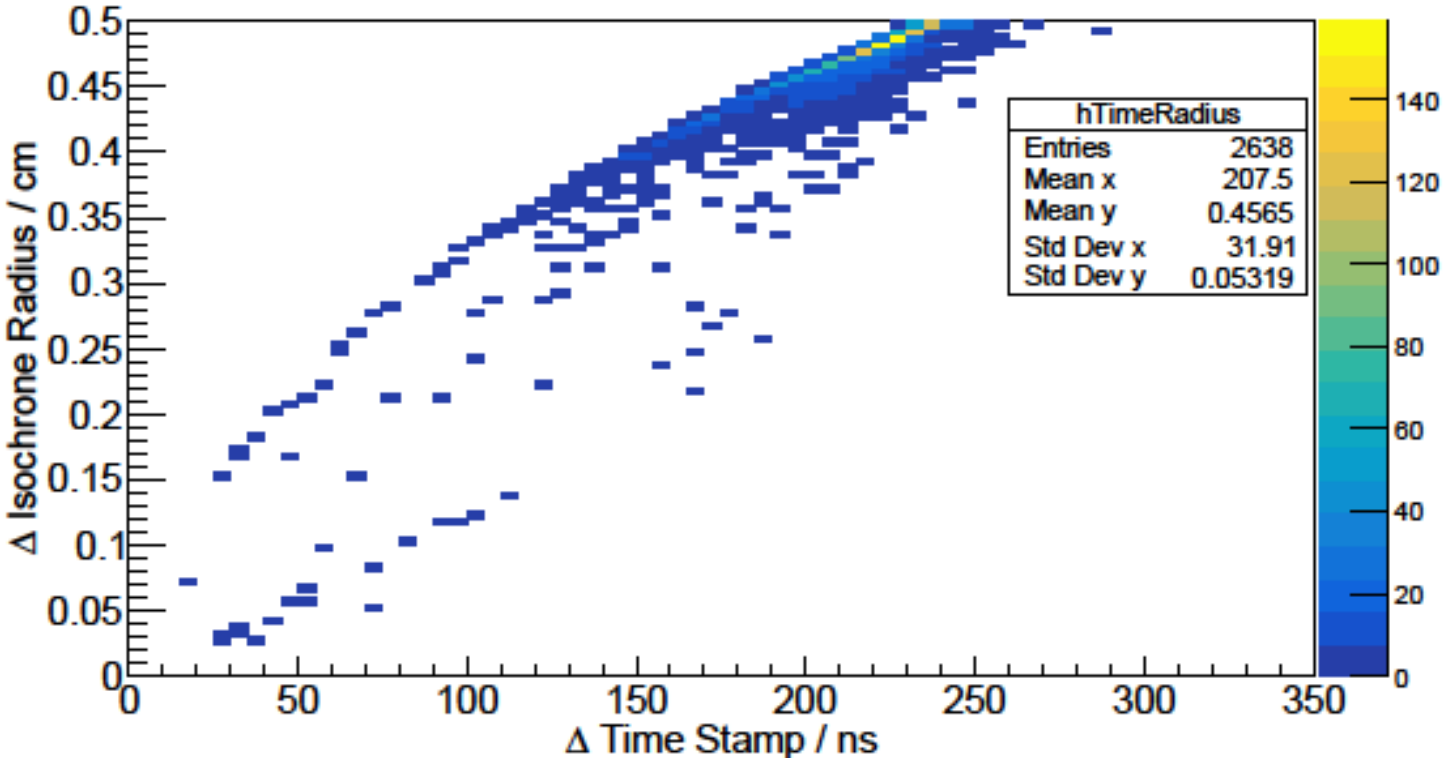




Reflection of isochrones differences of hits

Time-cut should not be tightened below ~ 50 ns

Isocrones and Timestamps



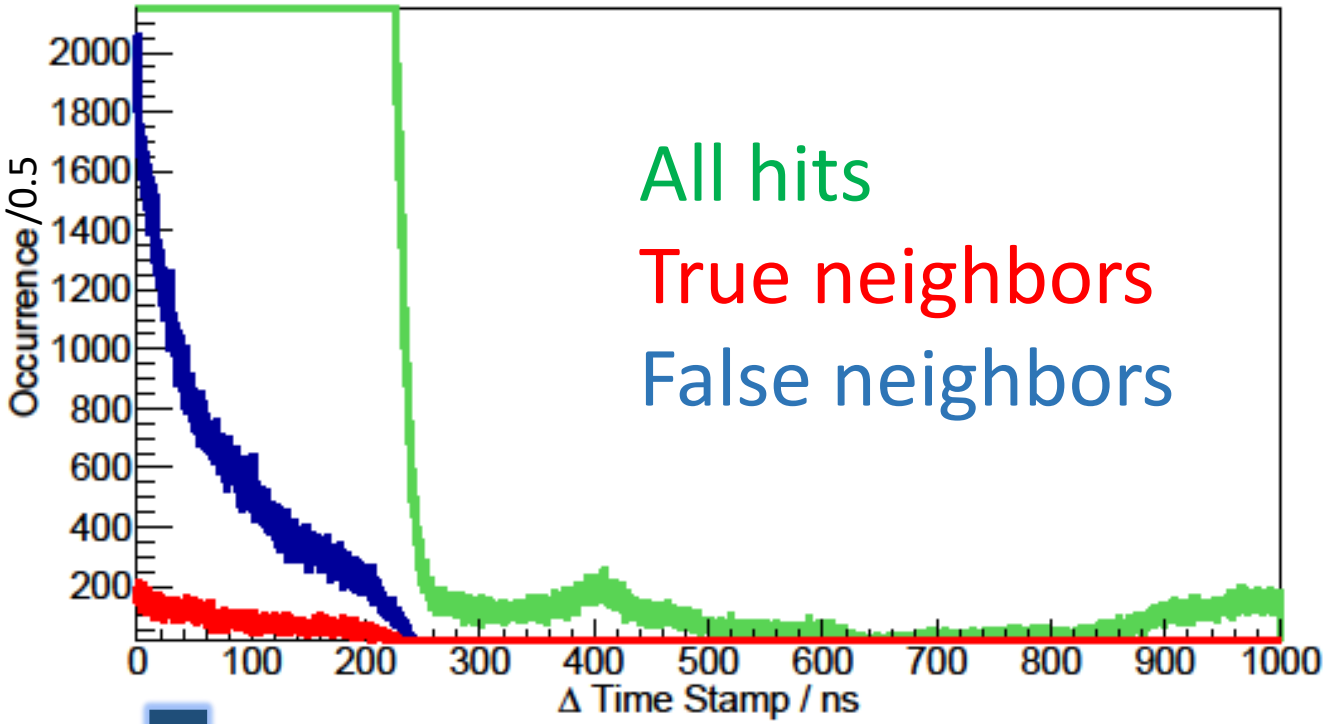
Clear correlation →
drift times (leading edge time)
dominate the time stamps

Time difference between hits in event

True neighbors:
Spatial hit neighbors originating from same MC truth track

False neighbors:
Spatial hit neighbors not originating from same MC truth track

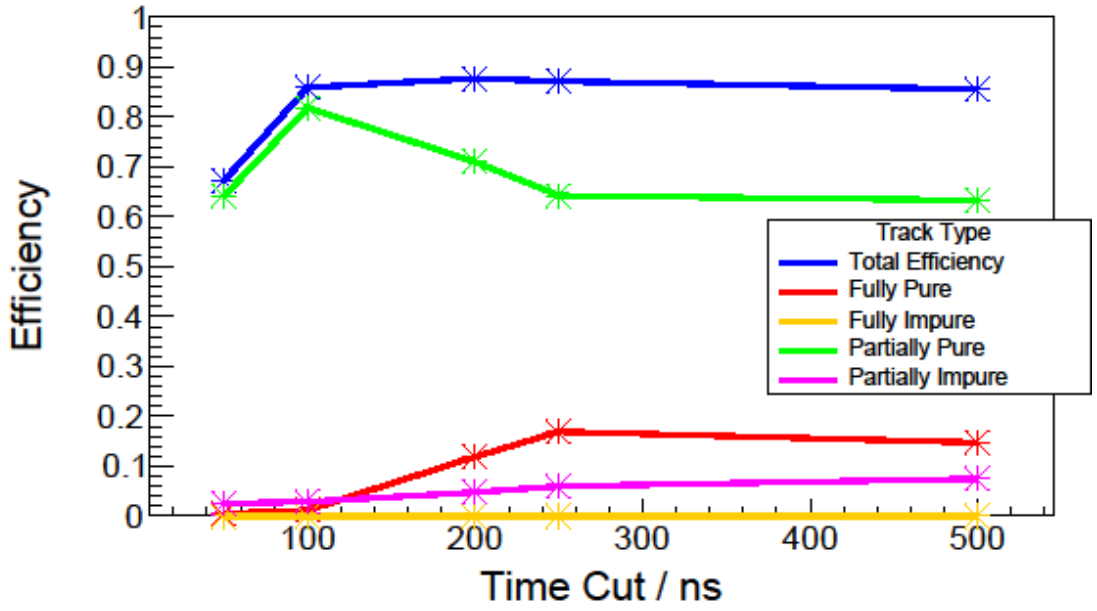
All hits: all hits in event, neighbors as well

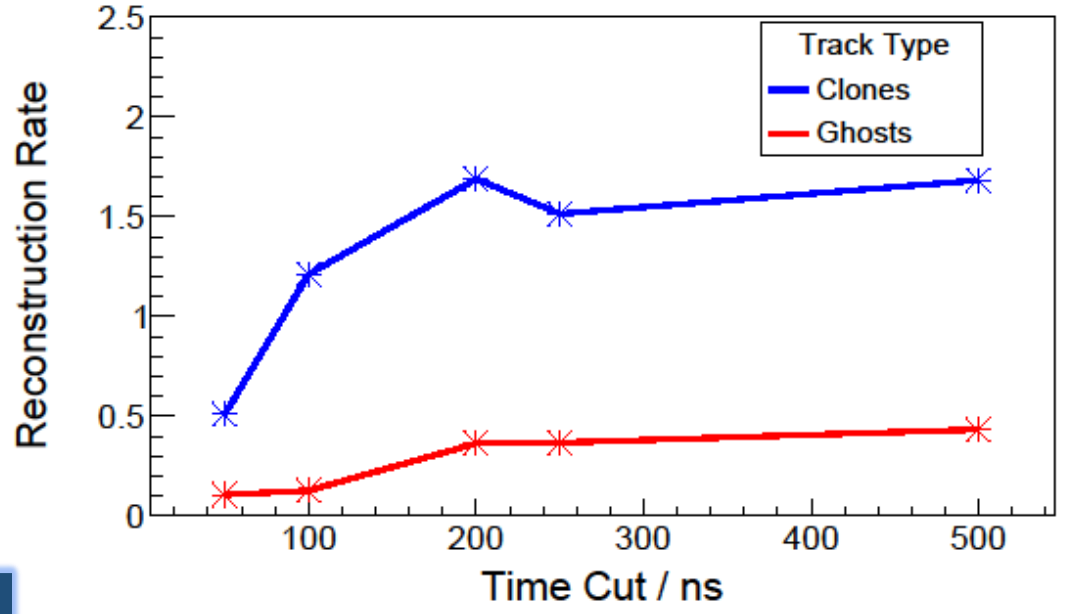


1. Most spatial neighbors are true neighbors
2. Since event based one can not distinguish true/false neighbors by timing
3. Some late hits (induced by ions created by slow neutrons) in 3.9 % of all events

Efficiencies

2.0 MHz





$n_{possible}$~ 2 670

Requirement: ≥ 6 STT hits

$$eff = \frac{n_{found}}{n_{possible}}$$

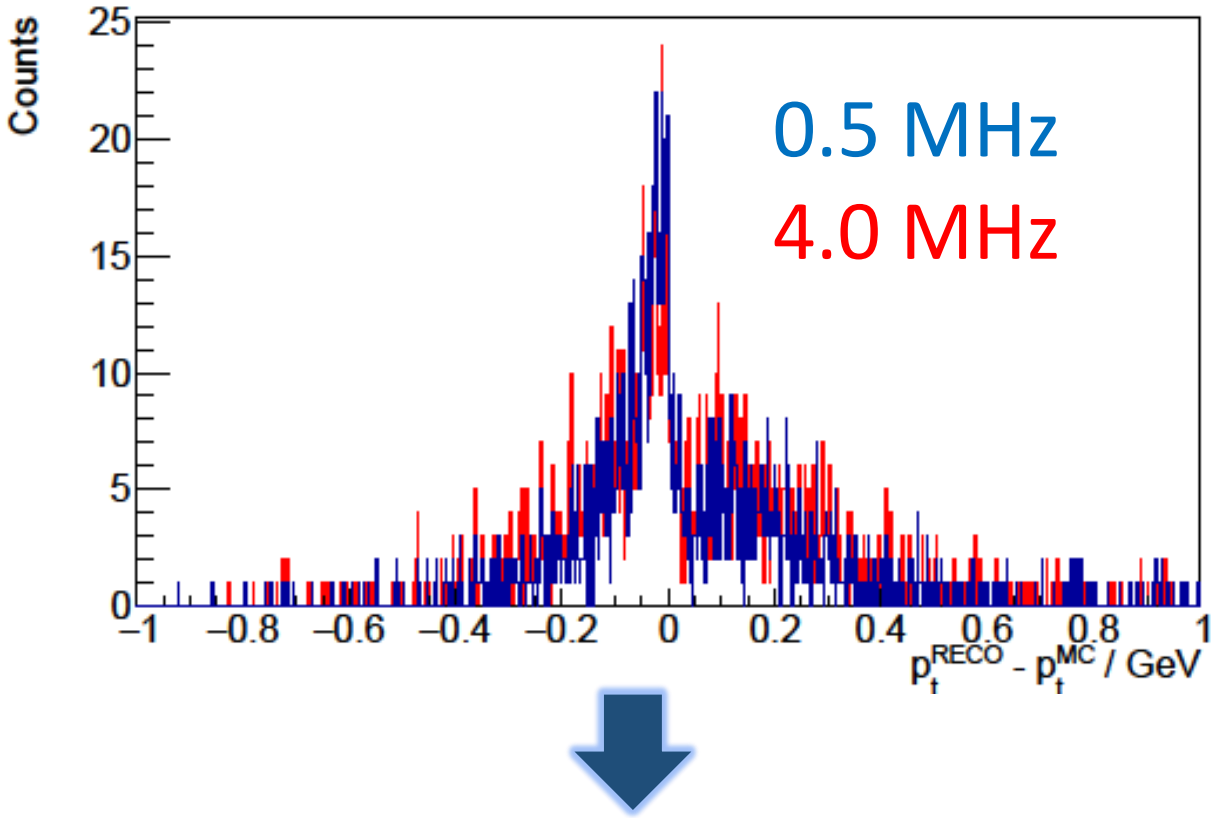
Tightening the time-cut:

1) greatly decreases the clone rate
2) decreases efficiency

Ghost Rate= $\frac{n_{ghosts}}{n_{possible}}$

Clone Rate= $\frac{n_{clones}}{n_{possible}}$

Momentum Resolution



- No qualitative difference
- Work needed to improve momentum resolution

Current Tracking activities

1. Extrapolation in 3D from STT to other detectors
2. Clone (ghost) merging/rejection

Point 1:

- Expand my existing 2D track extrapolation procedure [3] to work in 3D
  - Use Pz-finder to obtain z-component of momentum
  - 2 methods of extrapolating tracks to MVD:
    1. Use Riemann track
    2. Use Helix extrapolator
       - Include all MVD hits and utilize χ^2 in xy-space to refine hit purity by iteratively excluding hits with largest contribution
  - 3D method work for MVD (main focus) but also for GEM and Barrel ToF

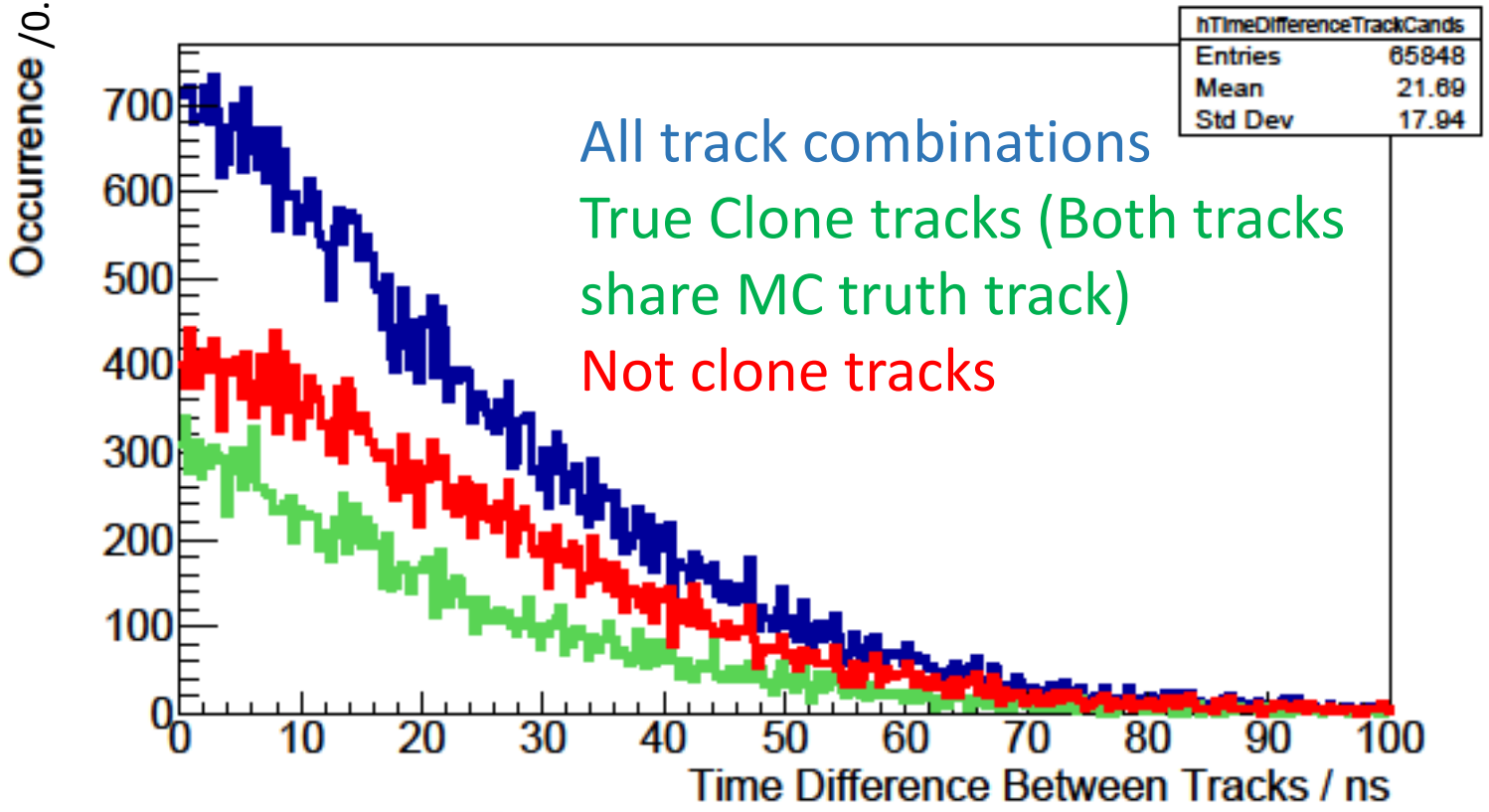
$$\chi^2 = \sum_{i=0}^{i=N} \frac{d^2_{hit \to track}}{\sigma^2_{hit,x}}$$

Results very preliminary – Next meeting

[3] Jenny Regina, Walter Ikegami Andersson, *Time-based Reconstruction of Hyperons at PANDA at FAIR*, 2019, Conference Proceedings, CDT/WIT2019. e-Print: arXiv:1910.06086v1

Time-Stamps

For Clone Merging

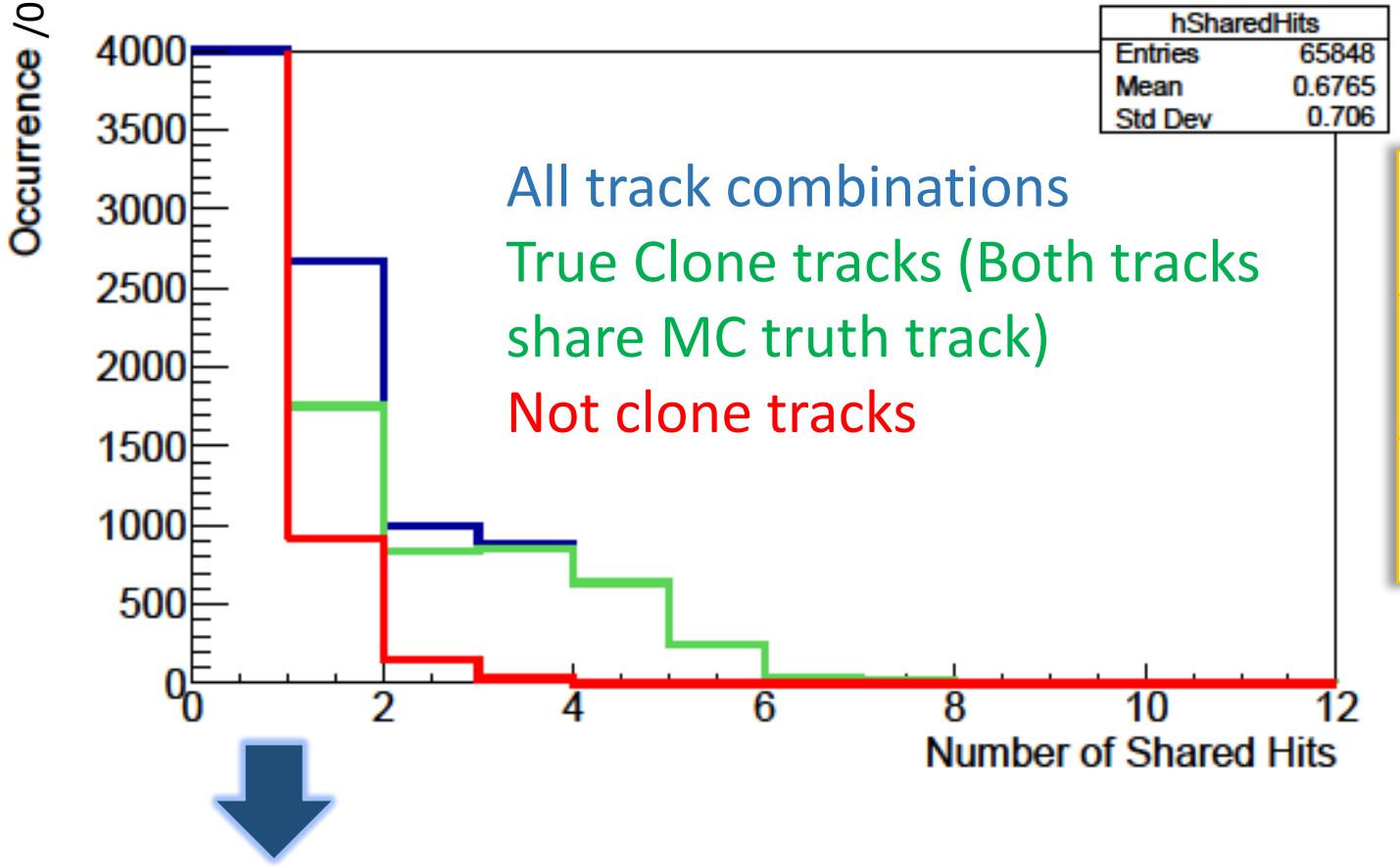


Time of track calculated as average of time-stamps of hits in track

Not promising to distinguish clones from non-clones

Number of Shared Hits / two tracks

For Clone Merging



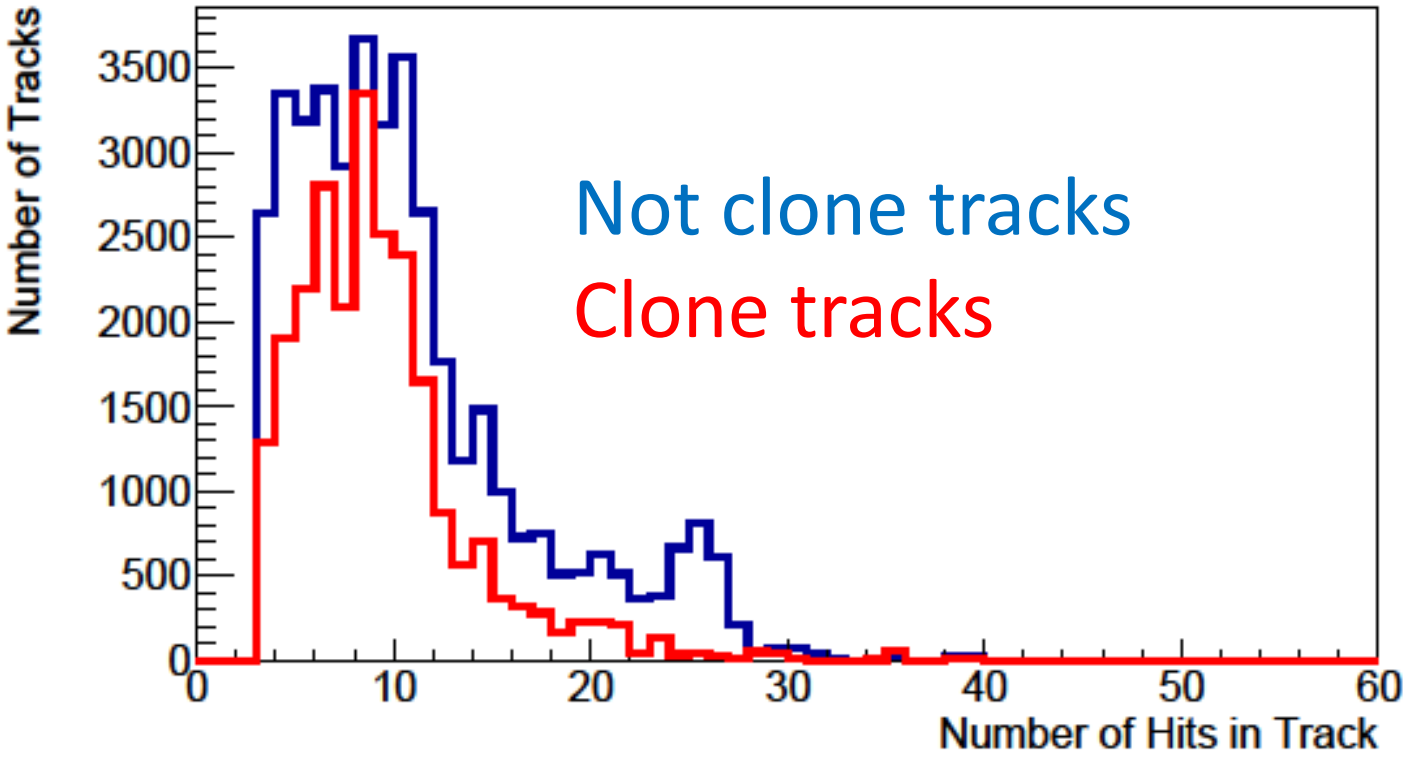
	≥ 1 shared hit	0 shared hits
True Clone Tracks	2 209 6.7 %	10 160 30.9 %
Not Clone Tracks	553 1.7 %	20 002 60.8 %

Total combinations: 32 924

Promising for clone merging:

I have written a task for merging which is under testing

Number of hits



As expected-> clone tracks usually reconstructed in shorter pieces

Summary

- The PANDA Quality Assurance task for tracking (PndTrackingQA) work for time-based reconstruction
- The time-based SttCellTrackFinder show stable efficiency at interaction rates 0.5-4 MHz
- Clone (ghost) rate is an issue, ideas for clone merging exist

Outlook

- Finalize 3D track extrapolation from STT
- Finalize clone merging procedure

Summary

- The PANDA Quality Assurance task for tracking (PndTrackingQA) work for time-based reconstruction
- The time-based SttCellTrackFinder show stable efficiency at interaction rates 0.5-4 MHz
- Clone (ghost) rate is an issue, ideas for clone merging exist

Thank You!

Backup

Running Time-Based Digitization

```
FairRunAna *fRun= new FairRunAna();
FairFileSource *fFileSource = new FairFileSource(inFile);
//fFileSource->ReadEventTimeFromFile("EventTimes.dat");
//fFileSource->SetEventMeanTime(50);
fFileSource->SetEventTimeInterval(500,501); // Time between events [ns]
fFileSource->SetBeamTime(1600, 400);
fRun->SetSource(fFileSource);
```

Mean time between events with Poisson Distribution

Fixed mean time between events

Note: **PndSttHitProducerRealFast** does not produce time-stamps

```
PndSttHitProducerRealFull* sttHitProducer = new PndSttHitProducerRealFull();
sttHitProducer->RunTimeBased();
fRun->AddTask(sttHitProducer);

PndSttHitSorterTask* sttSorter = new PndSttHitSorterTask(5000, 50, "STTHit", "STTSortedHits", "PndSTT");
fRun->AddTask(sttSorter);
```

Running the timeBasedTrackingQA

```
PndIdealTrackFinder* trackIdeal = new PndIdealTrackFinder();
trackIdeal->SetRelativeMomentumSmearing(0.05);
trackIdeal->SetVertexSmearing(0.05, 0.05, 0.05);
trackIdeal->SetTrackingEfficiency(1.);
trackIdeal->SetTrackSelector("OnlySttTimeBasedFunctor"); // Default StandardTrackFunctor
trackIdeal->AddBranchName("STTSortedHits"); // Default MVDHitsPixel, MVDHitsStrip, STTHit, GEMHit, FTSHit
trackIdeal->SetRunTimeBased(kTRUE); // Default kFALSE
fRun->AddTask(trackIdeal);

PndSttCellTrackFinderTask* cellTrack = new PndSttCellTrackFinderTask();
cellTrack->AddHitBranch("STTSortedHits");
cellTrack->SetRunTimeBased(kFALSE); // Default kFALSE
cellTrack->SetRunWithSortedHits(kTRUE); // Default kFALSE
fRun->AddTask(cellTrack);

PndTrackingQATask* trackingQA = new PndTrackingQATask("CombiTrack", "IdealTrack");
trackingQA->SetFunctorName("OnlySttTimeBasedFunctor"); // Default StandardTrackFunctor
trackingQA->AddHitsBranchName("STTSortedHits"); // Default MVDHitsPixel, MVDHitsStrip, STTHit, GEMHit, FTSHit
trackingQA->SetRunTimeBased(kTRUE); // Default kFALSE
trackingQA->SetVerbose(0); // Deafult 1 (derived from FairTask)
fRun->AddTask(trackingQA);
```

Definitions

1. MC track
   - simulated track at simulation stage
2. Ideal track
   - track reconstructed by IdealTrackTinder
3. Reco track
   - track reconstructed by some realistic track finder (In this case the SttCellTrackFinder)

Algorithm	Accepts time sorted hits	If not: can be adjusted to accept time sorted hits (efforts required)	Algorithm takes time information into consideration	If not: can be adjusted to take time information into consideration (efforts required)
Ideal Track Finder	Yes	--	--	--
Barrel Track Finder	Yes	--	No	Needs closer look
Central Tracking	No	Relatively little effort, could probably be done in similar way as for the above algorithms	No	Needs closer look but one might be able to use time information to cluster hits
SttCellTrackFinder	Yes	--	Yes	--
CA Tracker	No	Little effort, can be done similarly as for above algorithms	No	Nees closer look but if needed could probably be done similarly as for SttCellTrackFinder
Pattern matcher	No	Little effort but might not be desirable at the moment	No	Patterns consist of tube and sector ids. Quite some effort is needed to create patterns with time information groups of hits can be matched to
Tracking QA	Yes	--	--	--
Unassigned hits task	No	Little effort, already assumes user given branch names	--	--

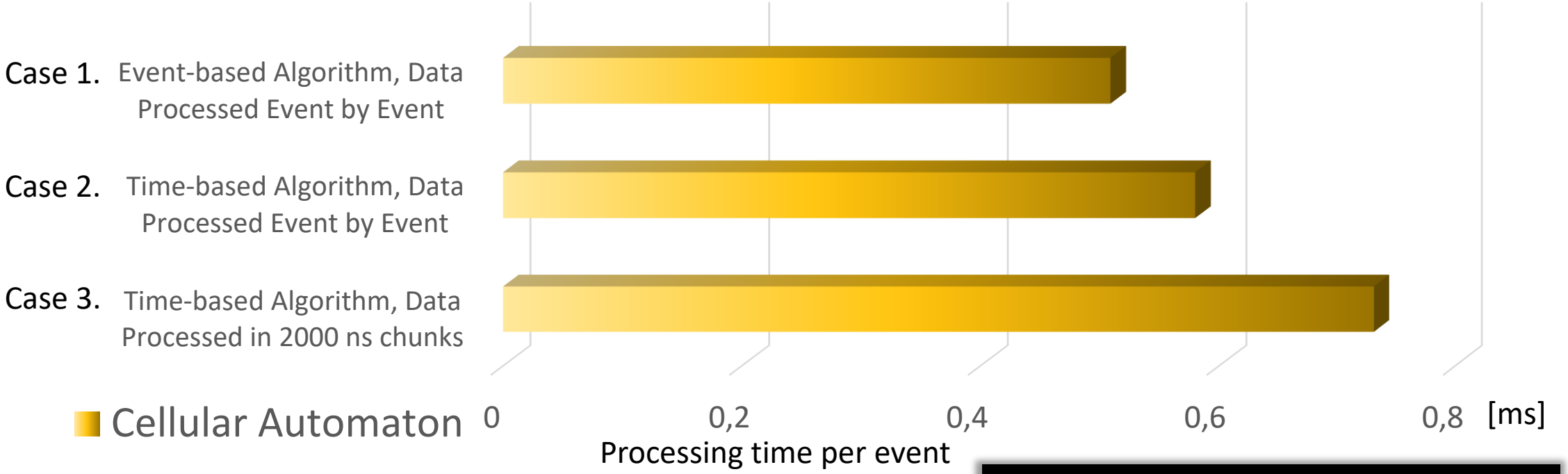
Performance TrackingQA

- std::chrono C++ library
- 1000 DPM events
  - EventTimeInterval = (1000,1001) in time based digitization
  - fStopTime=1000 ns
  - Start fetching data at 500 ns
- *Init()* of PndTrackingQA class is called in *Exec()* of *PndTrackingQATask.cxx*

Mode	**Total time [s]**	**Mean Time/ Event [s]**	**Total time [s]**	**Mean Time/ Event [s]**
	Excluding Init() of PndTrackingQA		**Including** Init() of PndTrackingQA	
Event based	1.4	0.001	1.4	0.001
Time based	521.9	0.5	518.7	0.5

- Event based mode roughly 500 times faster than Time based mode
- However: right now a lot of loops and workarounds in time based mode
- Additional time due to *Init()* of *PndTrackingQA.cxx* seems to be within timing uncertainties

Processing Time Find Hit Neighbors



Total processing time (full reconstruction: total track finding + a track fitting): **10 ms / event** on i7 3.4 GHz Processor

Average time / event for 10 000 generated background sample events

Case	Processing time [ms]	% of full reconstruction processing time
1	0,51	5,1
2	0,58	5,8
3	0,73	7,3

< 1 % difference