Clone Tracks Killing in DC’06

E. Rodrigues, NIKHEF
**Goal**
- Find tracks that are clones of other tracks

**Definition of clones**
- Two tracks are clones of each other if they share in the VELO and in the SEED stations at least 70% of hits

**Present usage in our tracking**
- Clone finder and killer run at the end of the tracking
- Uses by default Long, Ttrack, Upstream and Downstream tracks
- Outputs UNIQUE (= not clone) tracks to “best” container
The bug (1/2)

Bug found in the TrackCloneFinder tool!
- It affects all DC06 DSTs produced so far with Brunel, versions up to v31r0

Checking for clones on a buggy DST ... with the fixed tool ...

Read back a buggy DST
- check the “best” container
- Check for clones with the fixed TrackCloneFinder tool
- Plot the number of clones found per event ...

The bug potentially affects the clone finding every time one of the tracks does not have OT hits -> IT and OT hits ignored in the comparisons ...
Looking at the type of these clone tracks ...

- Clones of all track types!

- NOTE: Velo tracks should not be included, as they are not considered by the clone killer in Brunel

*But can the TrackCloneFinder bug explain it all ...?*
Clones found on fixed DST (1/3)

All tracks considered

Discarded:
- Velo tracks
- tracks from a same container

Velo tracks discarded

Read back a DST produced with the fixed tool

- Check the “best” container
- Check for clones with the fixed TrackCloneFinder tool
- TrackCloneFinder.CompareAtLHCbIDsLevel = true

I.e. at level of P. R. tracks. This is important to keep in mind
Clones found on fixed DST (2/3)

**All tracks considered**

**Velo tracks discarded**

**Discarded:**
- Velo tracks
- tracks from a same container
Observations & conclusions:

- Almost all clones are from tracks of same type …
- … and from the same container
- The pattern recognition algorithms are providing sets of tracks with clones amongst them!
- “Reason”: by default the clone killer does not compare tracks from the same container (property $SkipSameContainerTracks = true$)

This is another « conceptual bug »!
Clones found on buggy DST (1/2)

**Read back a buggy DST with the fixed tool**

- Check the “best” container
- `TrackEventCloneKiller.SkipSameContainerTracks = false`
- Check for clones with the fixed `TrackCloneFinder` tool
- `TrackCloneFinder.CompareAtLHCbIDsLevel = true`

**Number of clones on the DST**

<table>
<thead>
<tr>
<th>Entries</th>
<th>Mean</th>
<th>RMS</th>
<th>Underflow</th>
<th>Overflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>1991</td>
<td>4.792</td>
<td>3.532</td>
<td>0</td>
<td>62</td>
</tr>
</tbody>
</table>

**Discarded:**
- Velo tracks
- tracks from a same container
Clones found on buggy DST (2/2)

- Velo tracks
- tracks from a same container

All tracks considered
A final check, after fixes

Running Brunel:
- Fixed TrackCloneFinder tool
- Comparing fitted tracks at the level of the Measurements
- Clone killer comparing tracks from the same container
  (property SkipSameContainerTracks = false, not the default now!)

No clones found, obviously
**TrackTools**
- Code has been fixed
- In CVS in the head version

**SIMPLE SOLUTION - in DaVinci**
- Add options at the very beginning of execution, to run a simple cleaner algorithm that removes clones from the best container
  - probably the easiest thing to do
- Need to write the trivial algorithm

**Remarks**
- Fix requires a new DaVinci build with a new Rec release, in order to run on the DSTs produced so far
- I would suggest to build a new Brunel as soon as possible to continue production with the bug fixed in the reconstruction
Further remarks
- Clone killer run in DaVinci will compare at the level of LHCbIDs, not the result of the fit (comparison of Measurements), as in Brunel
- This is a small difference – but it is there

FULL SOLUTION - in DaVinci
- Add options at the very beginning of execution, to refit all the tracks in the “best” container
- Run the clone killer afterwards, just as in Brunel
- This does mean a significantly slower DaVinci!
- Probably not what one really wants …