Argonne NATIONAL LABORATORY

Persistent Layout Studies Updates

Peter van Gemmeren (ANL)

U.S. DEPARTMENT OF ENERGY

No single attribute retrieval.

- Switched **splitting off** while retaining **member-wise streaming** (another recent ROOT addition), so :

```
svcMgr.AthenaPoolCnvSvc.PoolAttributes +=
                  [ "STREAM_MEMBER_WISE = '1'" ]
svcMgr.AthenaPoolCnvSvc.PoolAttributes +=
                  [ "DEFAULT_SPLITLEVEL = '0'" ]
```

- Except for two largest Container():
  - New feature in POOL allows custom SplitLevel
  - Avoids file size increase

Almost event level retrieval

- Use **automatic basket optimization** to write fewer events per basket, by **flushing every 10 (for AOD, 5 for ESD) events** and increase size for baskets outside the CollectionTree to **32K**:

```
svcMgr.AthenaPoolCnvSvc.PoolAttributes +=
                [ "DEFAULT_BUFFERSIZE = '32000'" ]
svcMgr.AthenaPoolCnvSvc.PoolAttributes +=
    [ "ContainerName = 'TTree=CollectionTree';
                        TREE_AUTO_FLUSH = '10'" ]
```

- Switch off basket optimization for all auxiliary container.

Test: File Size

(my personal test, still need independent verification)

- Testing on voatlas51 with 2.5K ESD and ~20K AOD events
  - Thanks to Ilija for getting me access to lxvoadm/voatlas51.
- Produced two ESD / AOD samples:
  1. Full splitting and 30 MB to optimize CollectionTree.

     ```
     4512002732 Apr  2 myESD_1.pool.root
     3803552613 Apr  2 myAOD_1.pool.root
     ```

  2. No splitting and flushing every 5/10 events.

     ```
     4437849282 Apr  2 myESD_2.pool.root
     3681997184 Apr  2 myAOD_2.pool.root
     ```
- No more file size increase!

Results for Reading all events directly

(all objects, timed individually)

1. **Full splitting and 30 MB** to optimize CollectionTree:
   - Total read ESD: **420 ms/event**
   - Total read AOD : **59 (+/- 3) ms/event**
2. **No splitting and flushing** of CollectionTree every **10/5 events**:
   - Total read ESD: **360 ms/event**
   - Total read AOD: **35 (+/- 2) ms/event**

- **Reading all events still is ~15 - 30% faster**
  - Fewer reads, ~30 x less branches only ~10 x more baskets.

    – All AOD tests we run twice on different cores (same for both formats though), number between cores varied about 5% (between core 0 and 2)

Results for Reading ~1% of events via TAGs

(all objects, timed individually)

- 1% cut on random variable in TAGs
  - Ignore times for first event.
- **Not yet fully retested, only AOD**
  - Not yet latest version, but close.

1. **Full splitting and 30 MB** to optimize CollectionTree:
   - Total read **270 ms/event**
     - A 1% selection ends up reading and uncompressing most data.
2. **No splitting and flushing** of CollectionTree every **10 events**:
   - Total read **60 ms/event**
     - Only about 10% of the event data is read and uncompressed.

- **Reading selected events is ~4-5 times faster.**

New results

- No more increase in **file size.**
- **Memory** consumption **goes down** for read and write jobs using the new layout.
  - E.G.: To copy 20K AOD_1 uses **1170 MB**, whereas AOD_2 only requires **890 MB**.
  - This is after fixing auxiliary tree optimization.
- LCG 60b causes large VMEM increase for 30MB Collection Tree (Savannah #79833):
  - Not yet fully understood, but tests were done with LCG 59
  - New layout lowers the LCG 60b memory footprint.
- **Writing speeds** up for AOD (not for ESD) by **20%**

Outlook

- New ROOT storage layout with no-split / small baskets shows many improvements:
  - Much faster (factor 4-5) single event reading.
  - Saves lots of Virtual Memory, >100 MB
  - Faster read overall (15-35%).
  - For AOD, faster writing speed (>20%), but not for ESD.
- Given these results, we should switch release 17 to write out in this new format.
  - Changes are currently going into devval.