Moving the EOS namespace to persistent memory

Tobias Kappé (IT-DSS-DT)
tkappe@cern.ch

Supervised by
Elvin Alin Sindrilaru

EOS . . .

- . . . provides reliable and fast data storage.
- . . . stores measurements and processed data.
- . . . is used by all LHC experiments.
- . . . contains roughly 32PB.
- . . . has a *namespace* (100GB) kept in RAM.

Problem

- Booting into memory from disk is slow.
- This limits availability of the service.

Non-volatile RAM

- Simulated by DIMM RAM with a battery
- More sophisticated technologies incoming
- Boot speed could benefit from this.
  - No disk reads to restore changelog.
  - Consistent representation restored quicker.
- Mnemosyne toolchain provided by DSI
- EOS used as a ‘testbed’ for further use

Non-volatile RAM

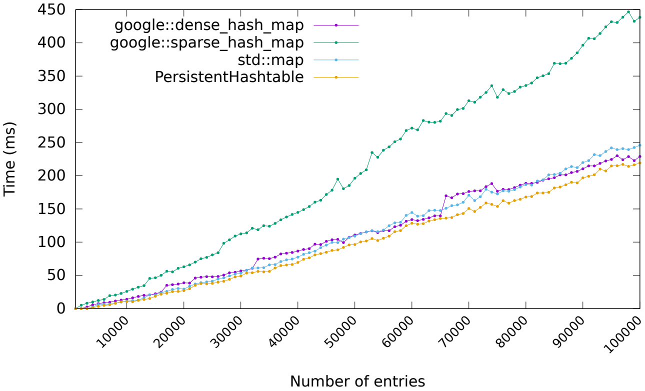
Persistency is a ‘vertical’ property:

- Transactional updates for consistency.
- Persistent memory should not point to non-persistent memory.
- It transcends some API boundaries.

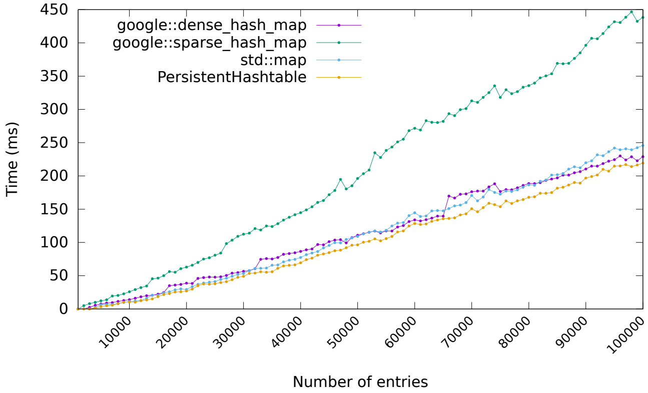
My contribution

- Hashtable suitable for transactional use
- Instrumentation to benchmark and validate
- First integration into EOS codebase

Hashtable performance

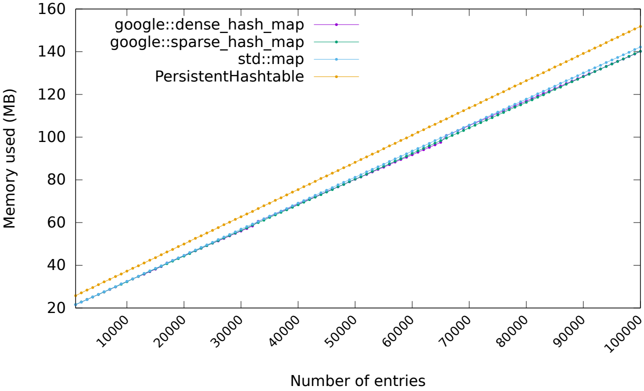


Hashtable performance

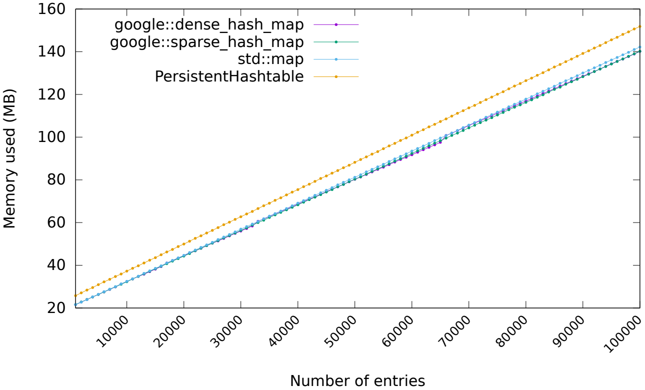


PersistentHashtable scales and can match google::dense_hash_map!

Hashtable memory usage



Hashtable memory usage



PersistentHashtable has more memory overhead (due to the AVL tree).

Future work

- Mnemosyne needs upgrade to newer `gcc`/`ICC`.
- More transactional data structures, for e.g.:
  - `std::string`
  - `std::vector`
- Which data should be kept persistent?
  - Move those over to persistent memory.
- Which transient data can be quickly restored?