DIANA I/O Update

Brian Bockelman

Plus a few CMS items

Fast IO Mode

- As presented in late December, we have a “Bulk API” branch which aims to avoid costly library functions per-event:
  - https://github.com/bbockelm/root/tree/root-bulkapi
- One the simplest branches, this is a O(8x) improvement over TTreeReader.

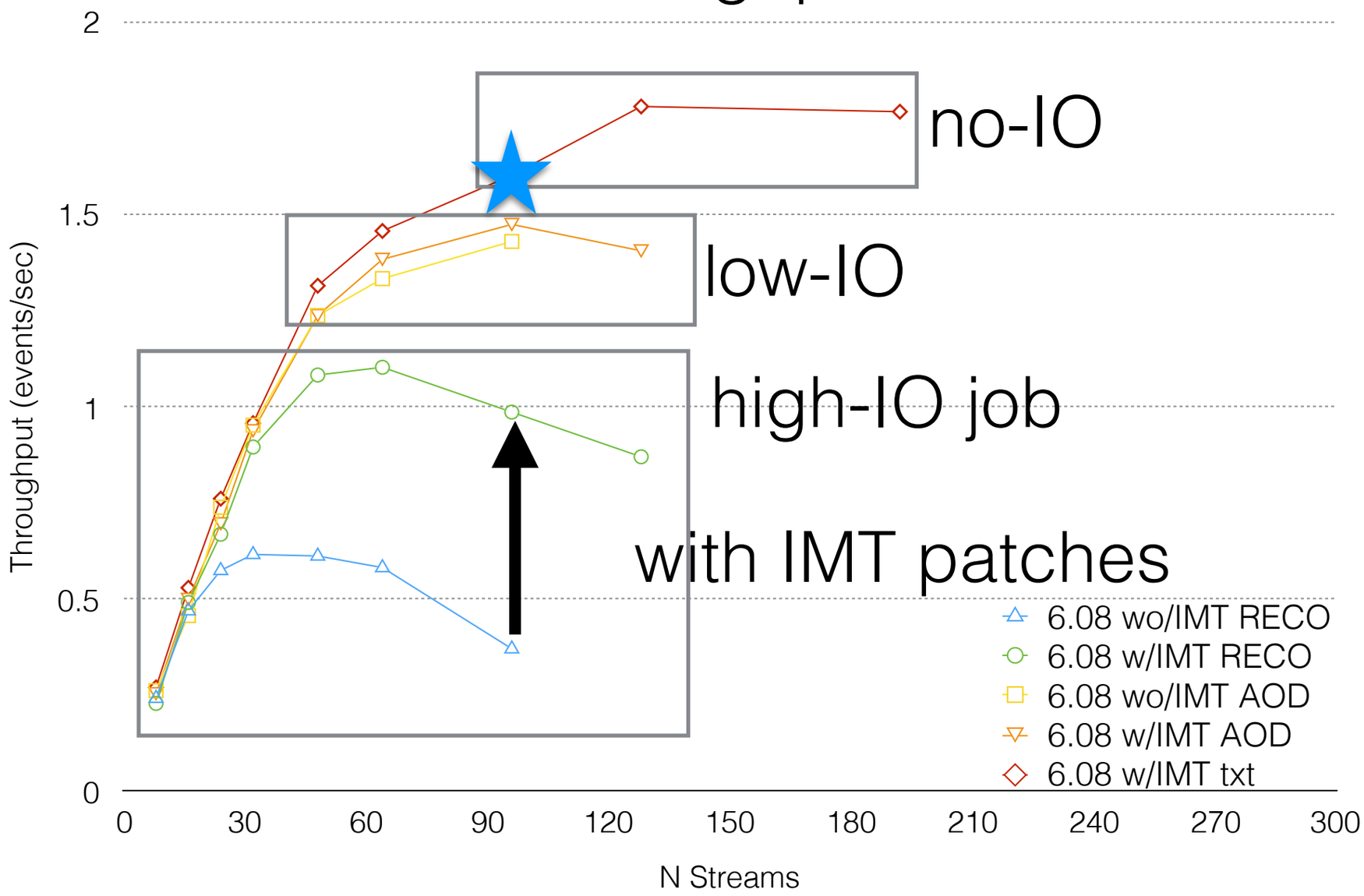
Current Work

- Last few weeks have focused on a “TTreeReaderFast”.
  - The “Fast” part of the code is relatively straightforward. Most difficult part is to verify the compiler inlines and does reasonable optimization.
  - The “TreeReader” part is proving hard: generating the correct TBranch* at runtime is resulting in a *lot* of code duplication with existing TTreeReader.
- Leaning toward dumping the current line of work, salvaging the pieces that interact with the bulk API, and simply encoding these into the existing TTreeReader.

IMT Writing

- Initial IMT writing patches have landed in master!
  - **Good news**: CMS saw up to 2x throughput improvement for reconstruction on KNL hosts on the largest data tier. See next slide
  - **Good news**: Even the simple “event.exe” macro in ROOT sees ~2x improvement (Even more if one tweaks the file to have more large branches).
  - Bad news: single-threaded ROOT IO time still dominates CMS KNL benchmark.

RECO Throughput on KNL



To IMT or to MT: Discuss!

- Amdahl's Law declares we need to decrease the serial fraction. Three approaches:
  - **Improve IMT**: perform serialization in parallel (detect when it is "safe" or via config).
  - **Multithreaded interfaces**: As part of the ROOT7 cleanup, rewrite interfaces to make MT-safe.
  - **TMemFile**: Have multiple files in memory that are "fast merged". Dan Riley @ Cornell about to start investigation.
    - Looking at prior slide, maybe target ~16 processing threads per TMemFile?

LZ4, redux redux

- LZ4 performance results still make no sense:
  - Comparing ZLIB & LZ4 command line tools on a ROOT file, LZ4 is ~4x faster at decompression than ZLIB.
  - When using corresponding libraries within ROOT, LZ4 decompression is comparable (sometimes slower) than ZLIB.
    - This appears to be true on dummy files with 1MB baskets!
- How can this be true?

Sample ROOT file repository

- I'd like to formalize & improve the ad-hoc collection of ROOT files on root.cern.ch.

- For MC-based files, would like to keep a repository of scripts to generate various files.

    - For data files: is git w/ LFS an option at CERN?

- DavidA is starting to look into this.

    - My current thinking is to start with a set of curated scripts to produce output files (using experiment software on CVMMFS) and a simple Makefile.

Other: ROOT Contributions

- We've mentioned previously, but I wanted to suggest a few things that would make contributing easier:
  - Slack group for ROOT devs?
  - Have Jenkins builds post summary of branch build results to PR?
  - Travis-CI "build" that checks coding conventions. LLVM-based checker?
  - Post Docker images for relevant Linux build platforms?
  - **We can volunteer effort - but may need GitHub admin access!**
- Have precious reviewer time focus on code, not on build failures!