Parallel Root

2 ways to utilize parallel processing in ROOT

- Implicit (by ROOT)

- Explicit (user-defined)

Better not to mix them up

Parallel Root

2 ways to utilize parallel processing in ROOT

- Implicit (by ROOT) → ROOT decides how parallelism is achieved e.g. how many threads are utilized

- Explicit (user-defined)

Parallel Root

2 ways to utilize parallel processing in ROOT

- Implicit (by ROOT) → ROOT decides how parallelism is achieved e.g. how many threads are utilized

- Explicit (user-defined) → ROOT objects have different safety levels. This requires extra caution

Thread Safety Level

- Different classes have different safety levels
Some need more caution than others

e.g. TTree and TFile objects:
only **different instances** should be accessed concurrently from **different threads**

Procedure

- Axis X: numEvents **Data** (1e4, 1e5, 1e6)
- Axis Y: **Speedup** relative to the ex.time of the sequential algorithm.
- Label: Number of Processes (1, 2, 4, 8)

For each combination execute 5 times to derive the AVG for more accurate results

System Memory: 12GB
System Processor:Intel© Core™ i5-8250U CPU @ 1.60GHz × 4

TTree::Fill

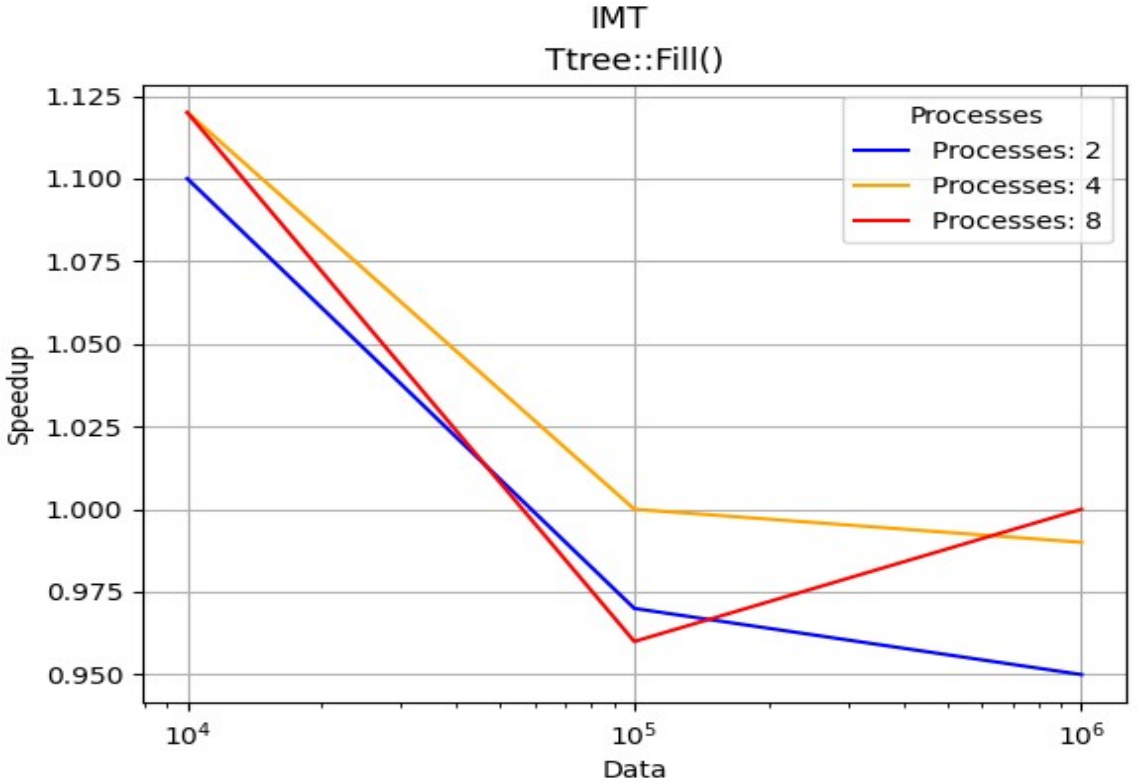
Fill the same tree in same file

- files → ***fillTree*.C***

- No way for multiple threads to
access safely the same tree and fill it in parallel

- Instead, work in individual objects and merge
Used: IMT(Implicit MultiThreading), OpenMP, MPI

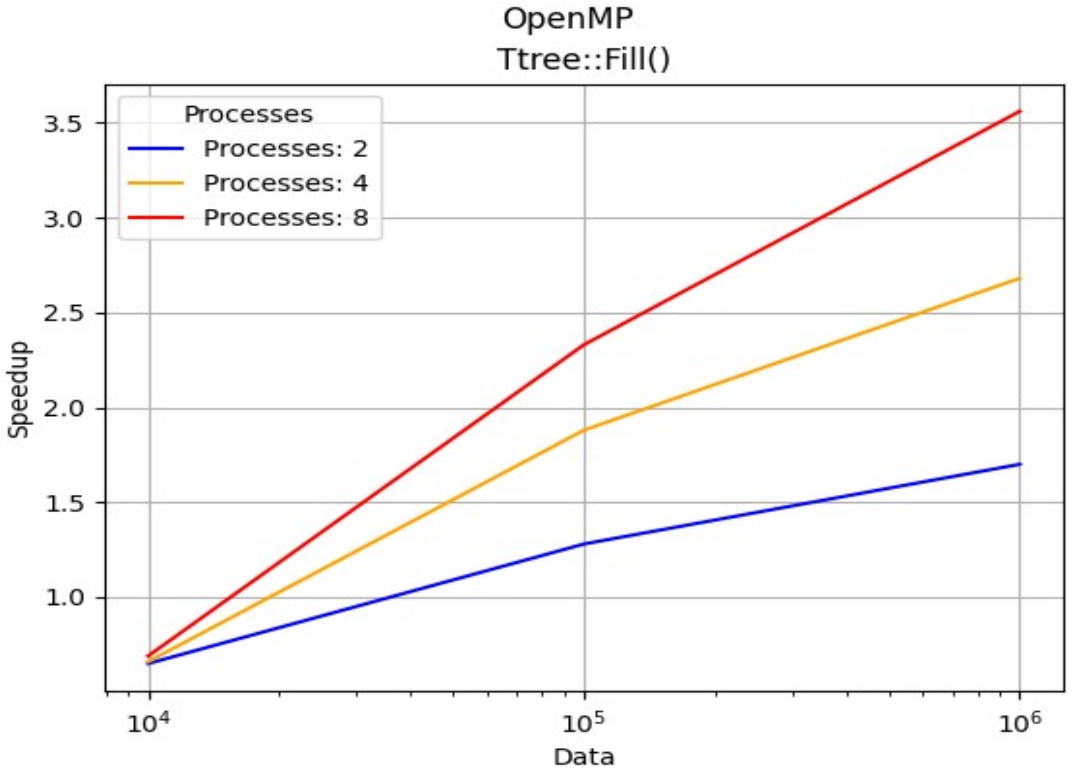
TTree::Fill + IMT



The user-defined numThreads is just a suggestion for ROOT.
This could explain the inconsistency (8 Processes) in this plot.

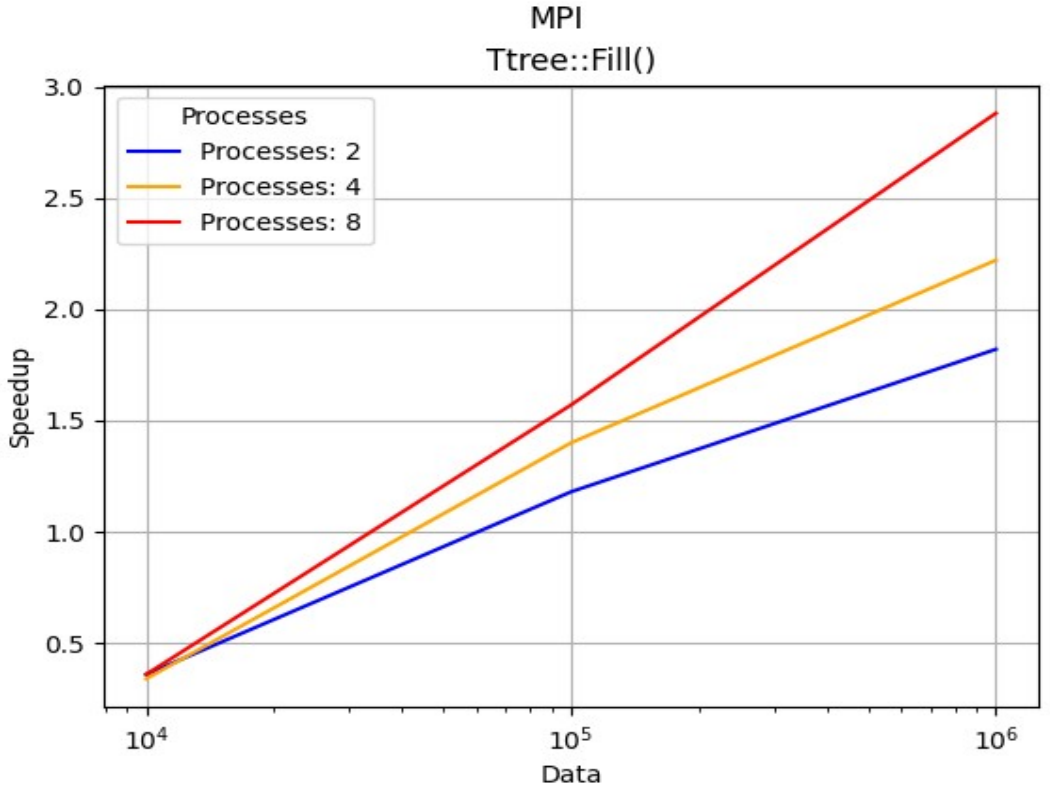
In any case, the difference is small.

TTree::Fill + OpenMP



Good Scaling

TTree::Fill + MPI



Good Scaling

Validation

Resulting trees are equal

Compared using the **root-diff package**

installation-link

Usage: root-diff -k <TreeName> file1.root file2.root

TH1F::Fill

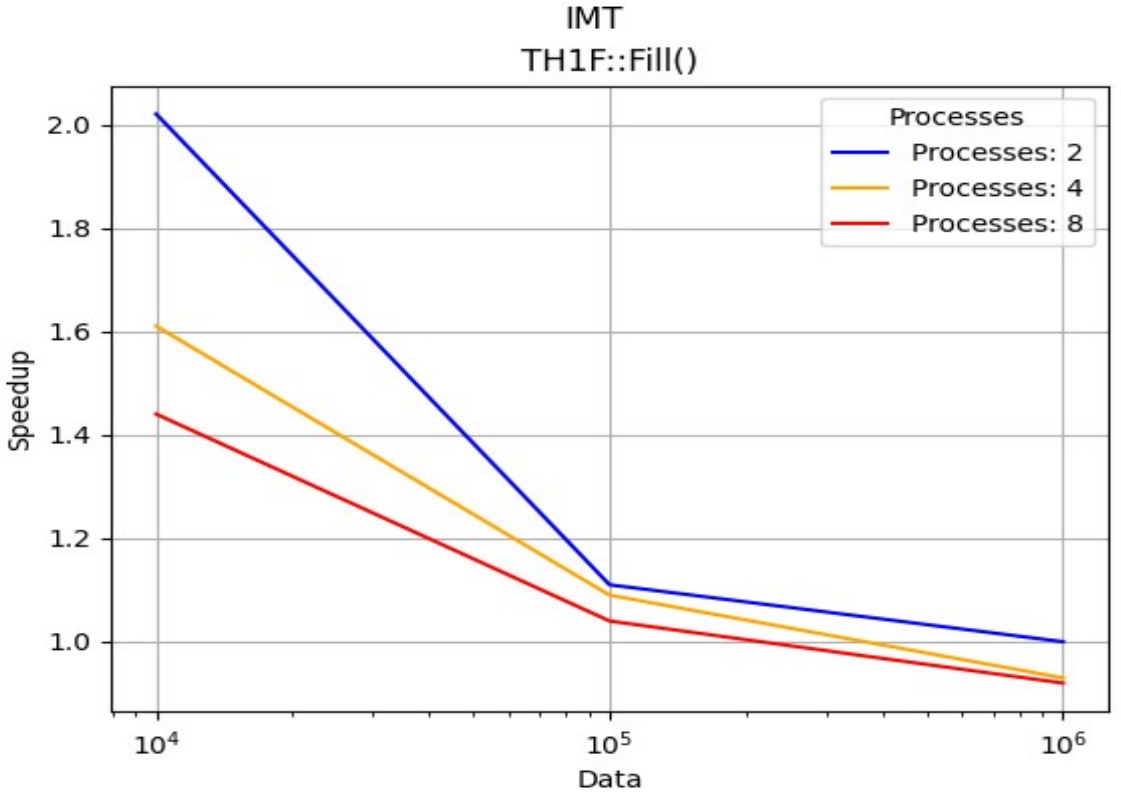
Fill multiple histos in multiple files

- files → ***fillHist*.C***

-Used: IMT, C++ <threads>, TprocessExecutor

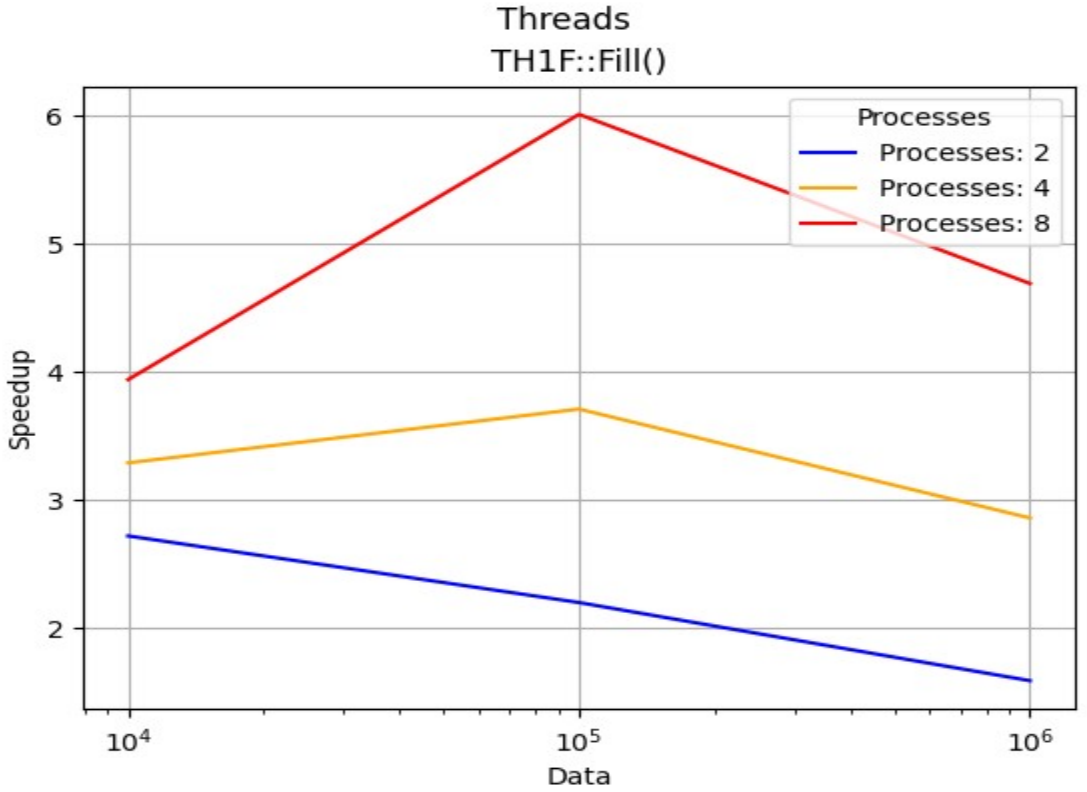
- In this case, more processes mean more written files with histograms.

TH1F::Fill + IMT



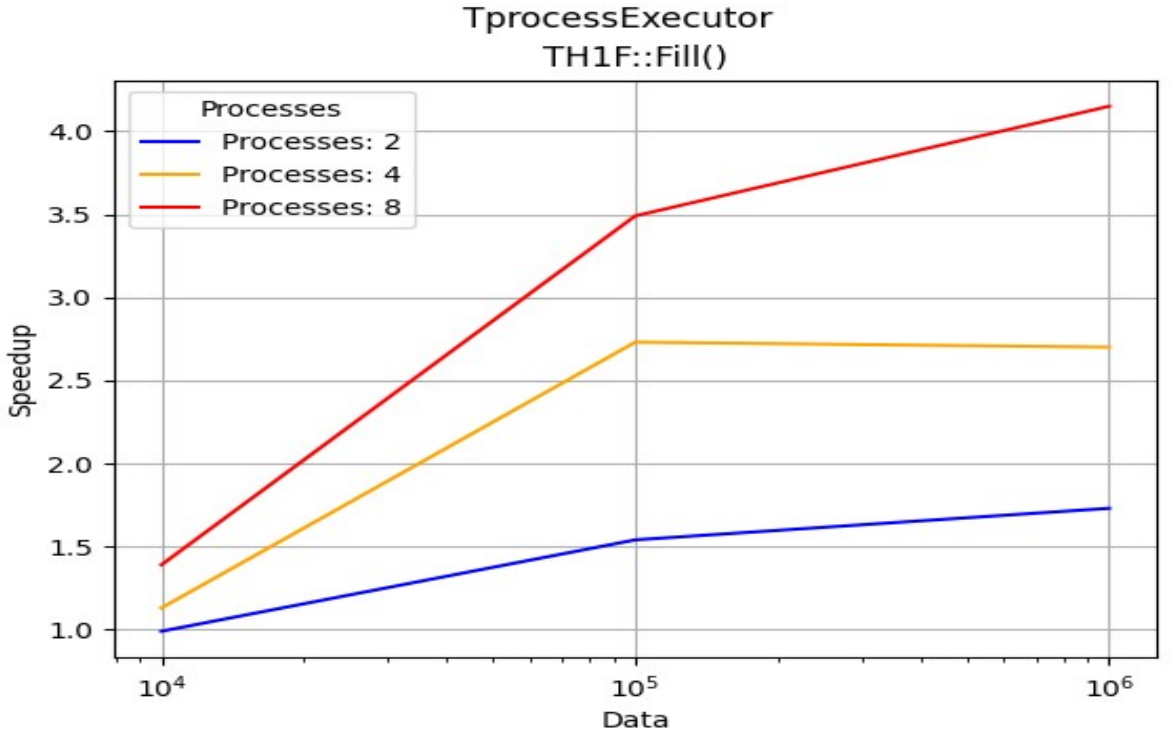
IMT gives decreased speedup for increased number of Processes
or for increased number of Data

TH1F::Fill + <std::threads>



Performance peak for 1e5 number of Events filled by 8 processes

TH1F::Fill + TProcessExecutor



Good Scaling