ROOT I/O Performance and Parallelism

G. Amadio, P. Canal, D. Piparo
for the ROOT Team

ROOT
Data Analysis Framework

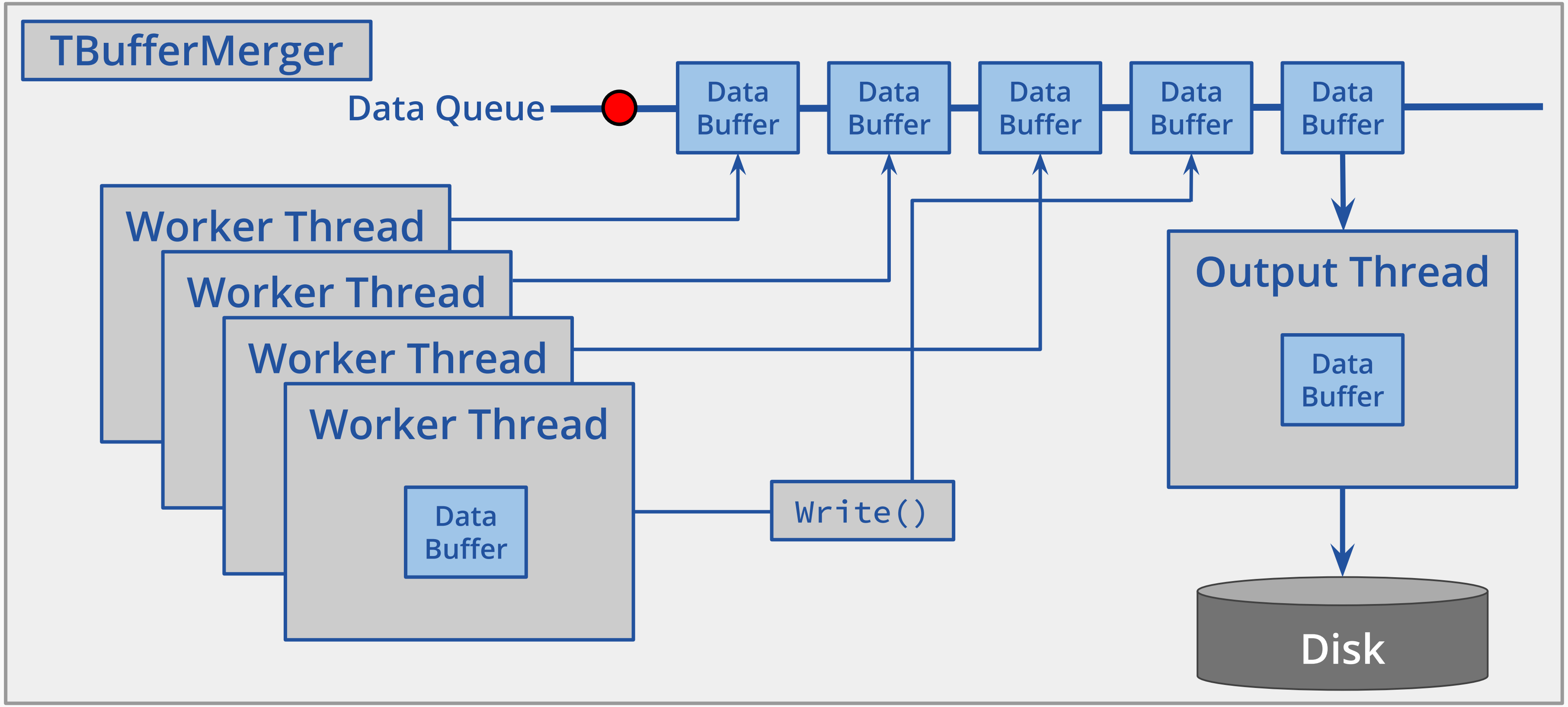
https://root.cern

Overview

Updates coming with ROOT 6.12:

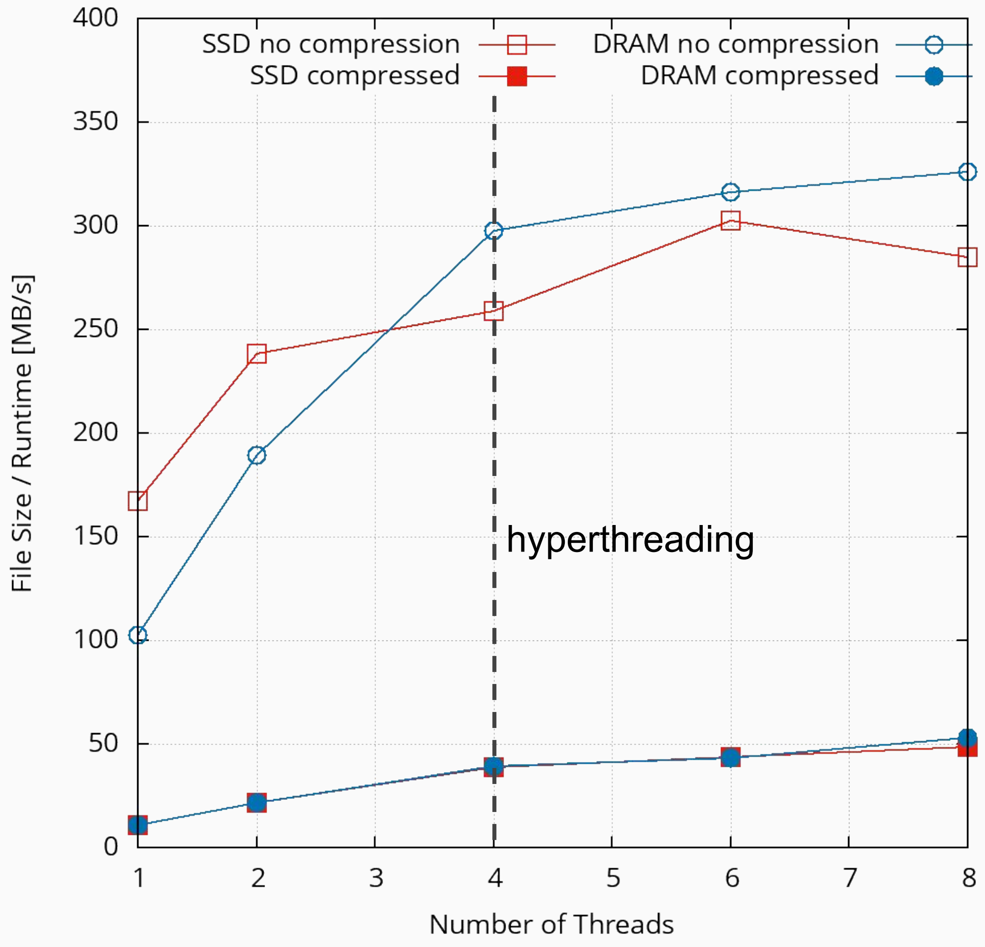
- `TBufferMerger` optimizations
- ROOT I/O performance improvements
  - Concurrency of ROOT I/O
  - Optimization of `TTree::Fill()`

TBufferMerger Class



Benchmark: TBufferMerger with Random Data

- Fill a tree with one branch with random numbers
- Synthetic benchmark that exacerbates the role of I/O by doing only lighweight computations
- Create ~1GB of data and write out to different media (SSD and DRAM)
- Quad core laptop Intel® Core i7 4710HQ (2.5GHz, 6M cache)



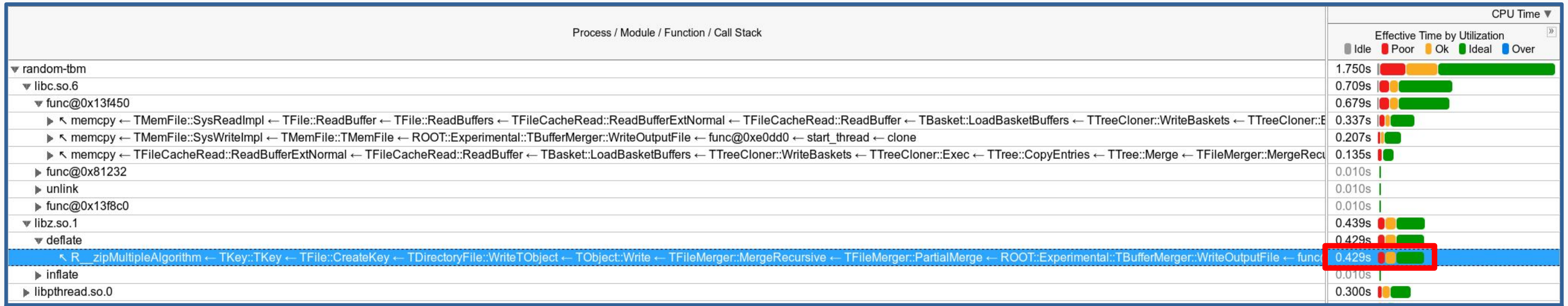
TBufferMerger Optimizations

- Output thread doing lots of work (compression)
  - Solution: add a setting for controlling auto-save point
  - Avoids creating too many `TTree` headers, which require compression

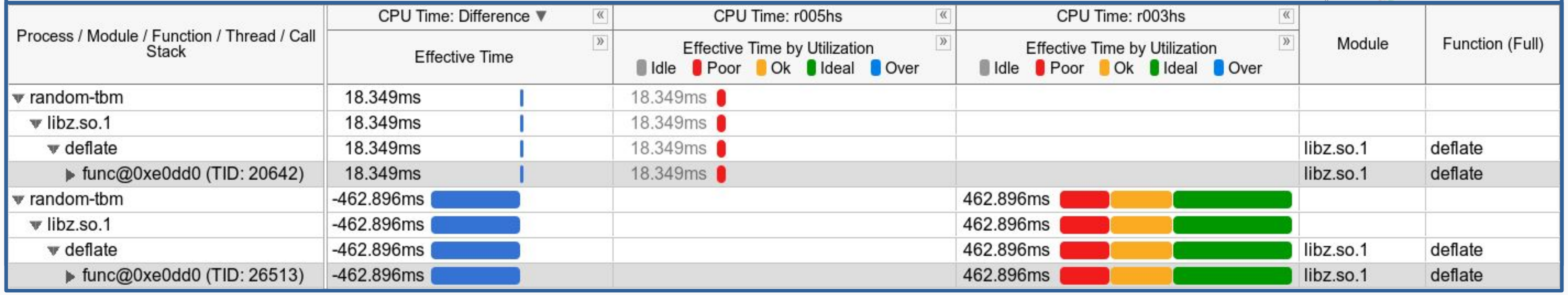
- Need a way to control queue size or rate-limit
  - Solution: add a non-blocking callback mechanism to `TBufferMerger`
  - Lets user decide when to create more data-producing tasks by registering a function that gets called everytime a buffer is removed from the merging queue
  - Add functions that lets user query the size of the queue
  - Auto-save forces flush to disk, which also avoids increases in memory

TBufferMerger Optimizations

Output thread compression with small and large auto-save



Process / Module / Function / Call Stack	CPU Time ▼ Effective Time by Utilization Idle Poor Ok Ideal Over
▼ random-tbm	1.750s
▼ libc.so.6	0.709s
▼ func@0x13f450	0.679s
▶ ↖ memcpy ← TMemFile::SysReadImpl ← TFile::ReadBuffer ← TFile::ReadBuffers ← TFileCacheRead::ReadBufferExtNormal ← TFileCacheRead::ReadBuffer ← TBasket::LoadBasketBuffers ← TTreeCloner::WriteBaskets ← TTreeCloner::E	0.337s
▶ ↖ memcpy ← TMemFile::SysWriteImpl ← TMemFile::TMemFile ← ROOT::Experimental::TBufferMerger::WriteOutputFile ← func@0xe0dd0 ← start_thread ← clone	0.207s
▶ ↖ memcpy ← TFileCacheRead::ReadBufferExtNormal ← TFileCacheRead::ReadBuffer ← TBasket::LoadBasketBuffers ← TTreeCloner::WriteBaskets ← TTreeCloner::Exec ← TTree::CopyEntries ← TTree::Merge ← TFileMerger::MergeRecu	0.135s
▶ func@0x81232	0.010s
▶ unlink	0.010s
▶ func@0x13f8c0	0.010s
▼ libz.so.1	0.439s
▼ deflate	0.429s
↖ R__zipMultipleAlgorithm ← TKey::TKey ← TFile::CreateKey ← TDirectoryFile::WriteTObject ← TObject::Write ← TFileMerger::MergeRecursive ← TFileMerger::PartialMerge ← ROOT::Experimental::TBufferMerger::WriteOutputFile ← func	0.429s
▶ inflate	0.010s
▶ libpthread.so.0	0.300s



Process / Module / Function / Thread / Call Stack	CPU Time: Difference ▼ Effective Time	CPU Time: r005hs Effective Time by Utilization Idle Poor Ok Ideal Over	CPU Time: r003hs Effective Time by Utilization Idle Poor Ok Ideal Over	Module	Function (Full)
▼ random-tbm	18.349ms	18.349ms			
▼ libz.so.1	18.349ms	18.349ms			
▼ deflate	18.349ms	18.349ms		libz.so.1	deflate
▶ func@0xe0dd0 (TID: 20642)	18.349ms	18.349ms		libz.so.1	deflate
▼ random-tbm	-462.896ms		462.896ms		
▼ libz.so.1	-462.896ms		462.896ms		
▼ deflate	-462.896ms		462.896ms	libz.so.1	deflate
▶ func@0xe0dd0 (TID: 26513)	-462.896ms		462.896ms	libz.so.1	deflate

Concurrency Improvements in ROOT I/O

Before: many queries to type system → many useless waits

`TDirectory::RegisterContext()`

`TTree::AutoSave()`

`TMemFile::ResetAfterMerge()`

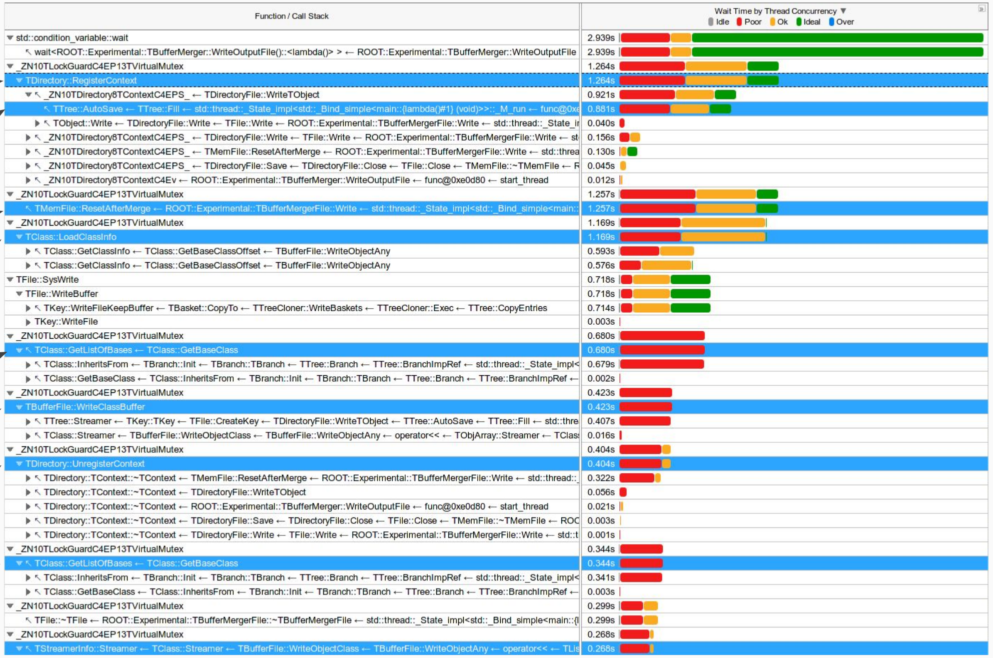
`TClass:LoadClassInfo()`

`TClass:GetListOfBases()`

`TBufferFile::WriteClassBuffer()`

`TDirectory::UnregisterContext()`

`TClass:GetListOfBases()`



Wait Time by Thread Concurrency ▼ — ▮ Idle ▮ Poor ▮ Ok ▮ Ideal ▮ Over

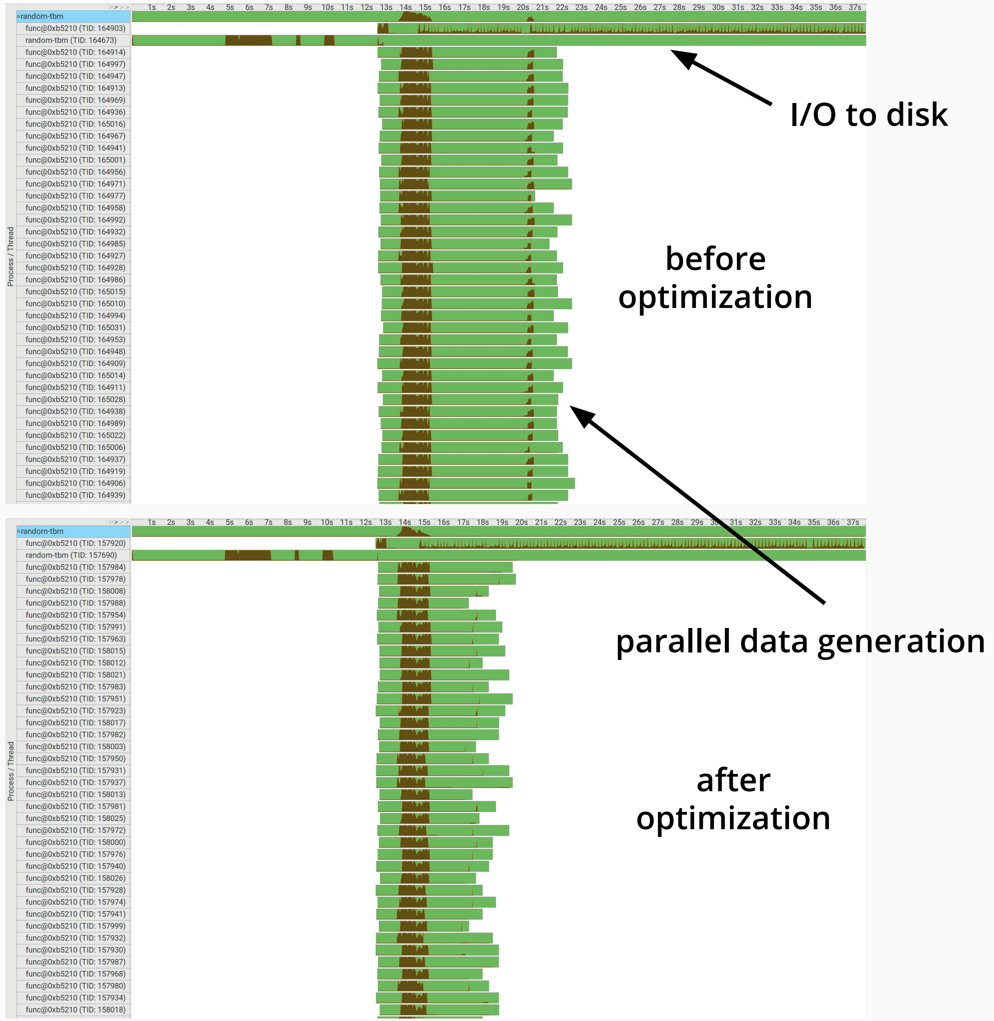
Function / Call Stack	Wait Time by Thread Concurrency
std::condition_variable::wait	2.939s
wait<ROOT::Experimental::TBufferMerger::WriteOutputFile()::<lambda()> > ← ROOT::Experimental::TBufferMerger::WriteOutputFile	2.939s
_ZN10TLockGuardC4EP13TVirtualMutex	1.264s
TDirectory::RegisterContext	1.264s
ZN10TDirectory8TContextC4EPS ← TDirectoryFile::WriteTObject	0.921s
TTree::AutoSave ← TTree::Fill ← std::thread::_State_impl<std::_Bind_simple<main::{lambda()#1} (void)>>::_M_run ← func@0xe	0.881s
TObject::Write ← TDirectoryFile::Write ← TFile::Write ← ROOT::Experimental::TBufferMergerFile::Write ← std::thread::_State_ir	0.040s
ZN10TDirectory8TContextC4EPS ← TDirectoryFile::Write ← TFile::Write ← ROOT::Experimental::TBufferMergerFile::Write ← st	0.156s
ZN10TDirectory8TContextC4EPS ← TMemFile::ResetAfterMerge ← ROOT::Experimental::TBufferMergerFile::Write ← std::threa	0.130s
ZN10TDirectory8TContextC4EPS ← TDirectoryFile::Save ← TDirectoryFile::Close ← TFile::Close ← TMemFile::~TMemFile ← R	0.045s
_ZN10TDirectory8TContextC4Ev ← ROOT::Experimental::TBufferMerger::WriteOutputFile ← func@0xe0d80 ← start_thread	0.012s
_ZN10TLockGuardC4EP13TVirtualMutex	1.257s
TMemFile::ResetAfterMerge ← ROOT::Experimental::TBufferMergerFile::Write ← std::thread::_State_impl<std::_Bind_simple<main::	1.257s
_ZN10TLockGuardC4EP13TVirtualMutex	1.169s
TClass::LoadClassInfo	1.169s
TClass::GetClassInfo ← TClass::GetBaseClassOffset ← TBufferFile::WriteObjectAny	0.593s
TClass::GetClassInfo ← TClass::GetBaseClassOffset ← TBufferFile::WriteObjectAny	0.576s
TFile::SysWrite	0.718s
TFile::WriteBuffer	0.718s
TKey::WriteFileKeepBuffer ← TBasket::CopyTo ← TTreeCloner::WriteBaskets ← TTreeCloner::Exec ← TTree::CopyEntries	0.714s
TKey::WriteFile	0.003s
_ZN10TLockGuardC4EP13TVirtualMutex	0.680s
TClass::GetListOfBases ← TClass::GetBaseClass	0.680s
TClass::InheritsFrom ← TBranch::Init ← TBranch::TBranch ← TTree::Branch ← TTree::BranchImpRef ← std::thread::_State_impl<	0.679s
TClass::GetBaseClass ← TClass::InheritsFrom ← TBranch::Init ← TBranch::TBranch ← TTree::Branch ← TTree::BranchImpRef ←	0.002s
_ZN10TLockGuardC4EP13TVirtualMutex	0.423s
TBufferFile::WriteClassBuffer	0.423s
TTree::Streamer ← TKey::TKey ← TFile::CreateKey ← TDirectoryFile::WriteTObject ← TTree::AutoSave ← TTree::Fill ← std::thre	0.407s
TClass::Streamer ← TBufferFile::WriteObjectClass ← TBufferFile::WriteObjectAny ← operator<< ← TObjArray::Streamer ← TClas	0.016s
_ZN10TLockGuardC4EP13TVirtualMutex	0.404s
TDirectory::UnregisterContext	0.404s
TDirectory::TContext::~TContext ← TMemFile::ResetAfterMerge ← ROOT::Experimental::TBufferMergerFile::Write ← std::thread::_	0.322s
TDirectory::TContext::~TContext ← TDirectoryFile::WriteTObject	0.056s
TDirectory::TContext::~TContext ← ROOT::Experimental::TBufferMerger::WriteOutputFile ← func@0xe0d80 ← start_thread	0.021s
TDirectory::TContext::~TContext ← TDirectoryFile::Save ← TDirectoryFile::Close ← TFile::Close ← TMemFile::~TMemFile ← ROC	0.003s
TDirectory::TContext::~TContext ← TDirectoryFile::Write ← TFile::Write ← ROOT::Experimental::TBufferMergerFile::Write ← std::t	0.001s
_ZN10TLockGuardC4EP13TVirtualMutex	0.344s
TClass::GetListOfBases ← TClass::GetBaseClass	0.344s
TClass::InheritsFrom ← TBranch::Init ← TBranch::TBranch ← TTree::Branch ← TTree::BranchImpRef ← std::thread::_State_impl<	0.341s
TClass::GetBaseClass ← TClass::InheritsFrom ← TBranch::Init ← TBranch::TBranch ← TTree::Branch ← TTree::BranchImpRef ←	0.003s
_ZN10TLockGuardC4EP13TVirtualMutex	0.299s
TFile::~TFile ← ROOT::Experimental::TBufferMergerFile::~TBufferMergerFile ← std::thread::_State_impl<std::_Bind_simple<main::{l	0.299s
_ZN10TLockGuardC4EP13TVirtualMutex	0.268s
TStreamerInfo::Streamer ← TClass::Streamer ← TBufferFile::WriteObjectClass ← TBufferFile::WriteObjectAny ← operator<< ← TLis	0.268s

ROOT-8871

ROOT-9002

Improving the Performance of ROOT I/O

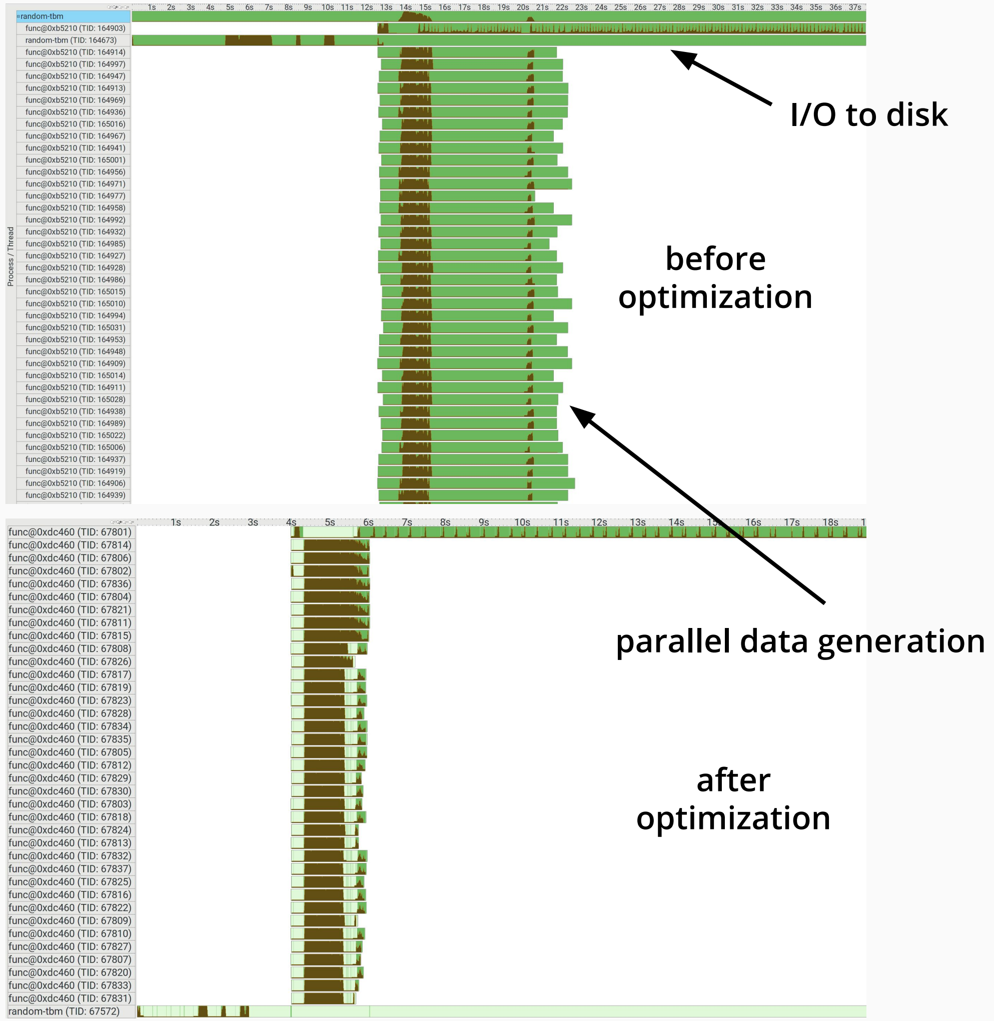
- Use simple case with TBufferMerger to optimize ROOT I/O
- Same random number generation from before
- Reduce number of mutex locks acquired when checking the type system
- Reduced from a few hundred locks to a single lock per thread



Improving the Performance of ROOT I/O

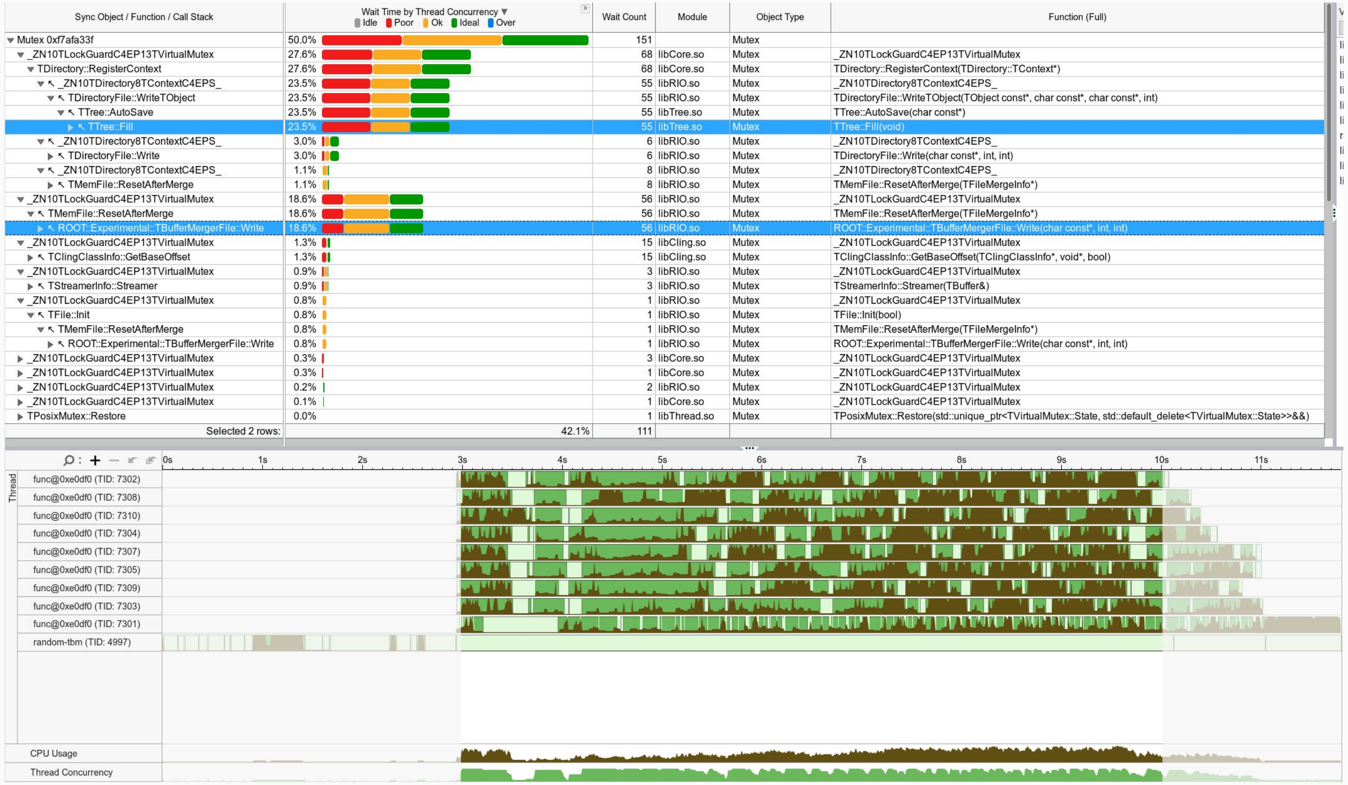
- Use simple case with TBufferMerger to optimize ROOT I/O
- Same random number generation from before
- Reduce number of mutex locks acquired when checking the type system
- Reduced from a few hundred locks to a single lock per thread

Targeting ROOT 6.12



Concurrency Improvements in ROOT I/O

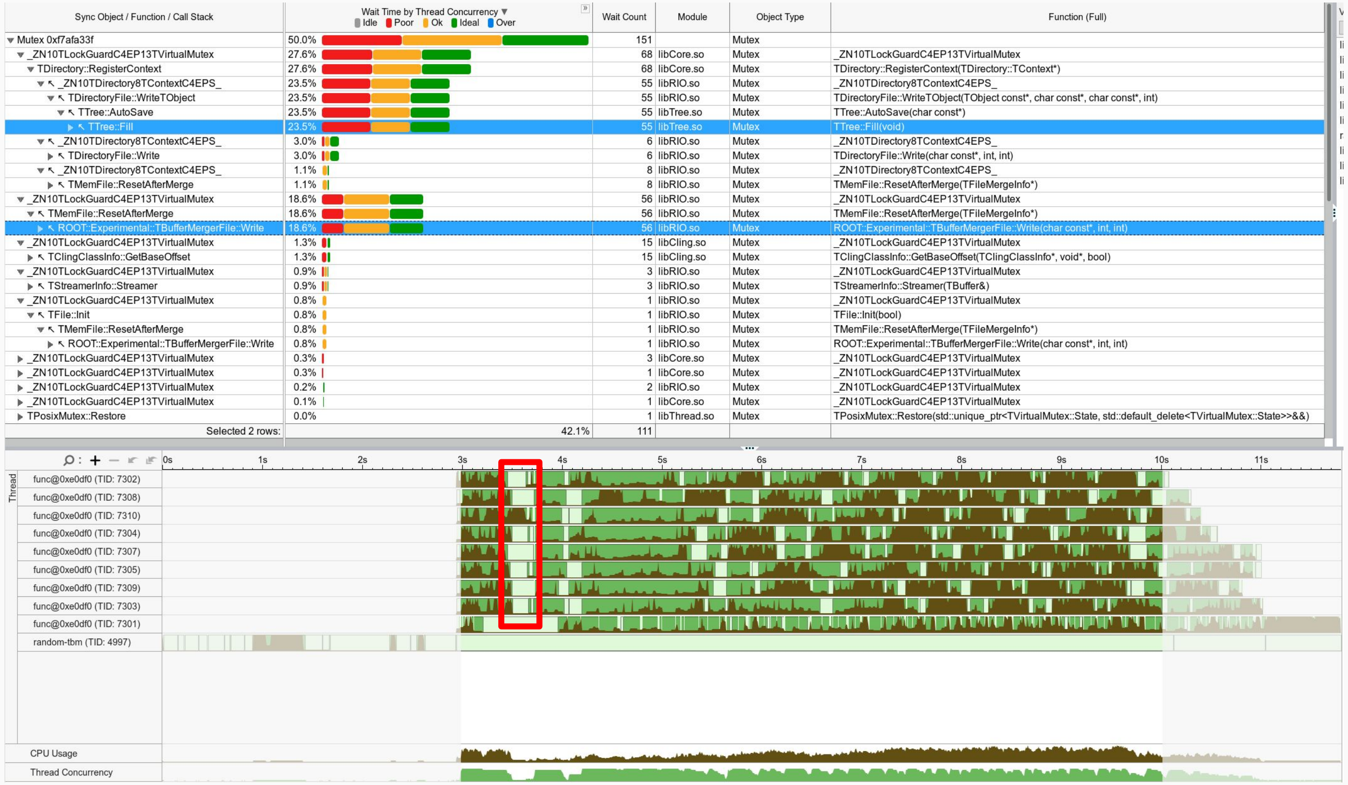
No more type system queries, only one wait per thread



Sync Object / Function / Call Stack	Wait Time by Thread Concurrency (Idle, Poor, Ok, Ideal, Over)	Wait Count	Module	Object Type	Function (Full)
Mutex 0xf7afa33f	50.0%	151		Mutex	
_ZN10TLockGuardC4EP13TVirtualMutex	27.6%	68	libCore.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
TDirectory::RegisterContext	27.6%	68	libCore.so	Mutex	TDirectory::RegisterContext(TDirectory::TContext*)
↖ _ZN10TDirectory8TContextC4EPS_	23.5%	55	libRIO.so	Mutex	_ZN10TDirectory8TContextC4EPS_
↖ TDirectoryFile::WriteTObject	23.5%	55	libRIO.so	Mutex	TDirectoryFile::WriteTObject(TObject const*, char const*, char const*, int)
↖ TTree::AutoSave	23.5%	55	libTree.so	Mutex	TTree::AutoSave(char const*)
↖ TTree::Fill	23.5%	55	libTree.so	Mutex	TTree::Fill(void)
↖ _ZN10TDirectory8TContextC4EPS_	3.0%	6	libRIO.so	Mutex	_ZN10TDirectory8TContextC4EPS_
↖ TDirectoryFile::Write	3.0%	6	libRIO.so	Mutex	TDirectoryFile::Write(char const*, int, int)
↖ _ZN10TDirectory8TContextC4EPS_	1.1%	8	libRIO.so	Mutex	_ZN10TDirectory8TContextC4EPS_
↖ TMemFile::ResetAfterMerge	1.1%	8	libRIO.so	Mutex	TMemFile::ResetAfterMerge(TFileMergeInfo*)
_ZN10TLockGuardC4EP13TVirtualMutex	18.6%	56	libRIO.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
↖ TMemFile::ResetAfterMerge	18.6%	56	libRIO.so	Mutex	TMemFile::ResetAfterMerge(TFileMergeInfo*)
↖ ROOT::Experimental::TBufferMergerFile::Write	18.6%	56	libRIO.so	Mutex	ROOT::Experimental::TBufferMergerFile::Write(char const*, int, int)
_ZN10TLockGuardC4EP13TVirtualMutex	1.3%	15	libCling.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
↖ TClingClassInfo::GetBaseOffset	1.3%	15	libCling.so	Mutex	TClingClassInfo::GetBaseOffset(TClingClassInfo*, void*, bool)
_ZN10TLockGuardC4EP13TVirtualMutex	0.9%	3	libRIO.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
↖ TStreamerInfo::Streamer	0.9%	3	libRIO.so	Mutex	TStreamerInfo::Streamer(TBuffer&)
_ZN10TLockGuardC4EP13TVirtualMutex	0.8%	1	libRIO.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
↖ TFile::Init	0.8%	1	libRIO.so	Mutex	TFile::Init(bool)
↖ TMemFile::ResetAfterMerge	0.8%	1	libRIO.so	Mutex	TMemFile::ResetAfterMerge(TFileMergeInfo*)
↖ ROOT::Experimental::TBufferMergerFile::Write	0.8%	1	libRIO.so	Mutex	ROOT::Experimental::TBufferMergerFile::Write(char const*, int, int)
_ZN10TLockGuardC4EP13TVirtualMutex	0.3%	3	libCore.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
_ZN10TLockGuardC4EP13TVirtualMutex	0.3%	1	libCore.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
_ZN10TLockGuardC4EP13TVirtualMutex	0.2%	2	libRIO.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
_ZN10TLockGuardC4EP13TVirtualMutex	0.1%	1	libCore.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
TPosixMutex::Restore	0.0%	1	libThread.so	Mutex	TPosixMutex::Restore(std::unique_ptr<TVirtualMutex::State, std::default_delete<TVirtualMutex::State>>&&)
Selected 2 rows:	42.1%	111			

Concurrency Improvements in ROOT I/O

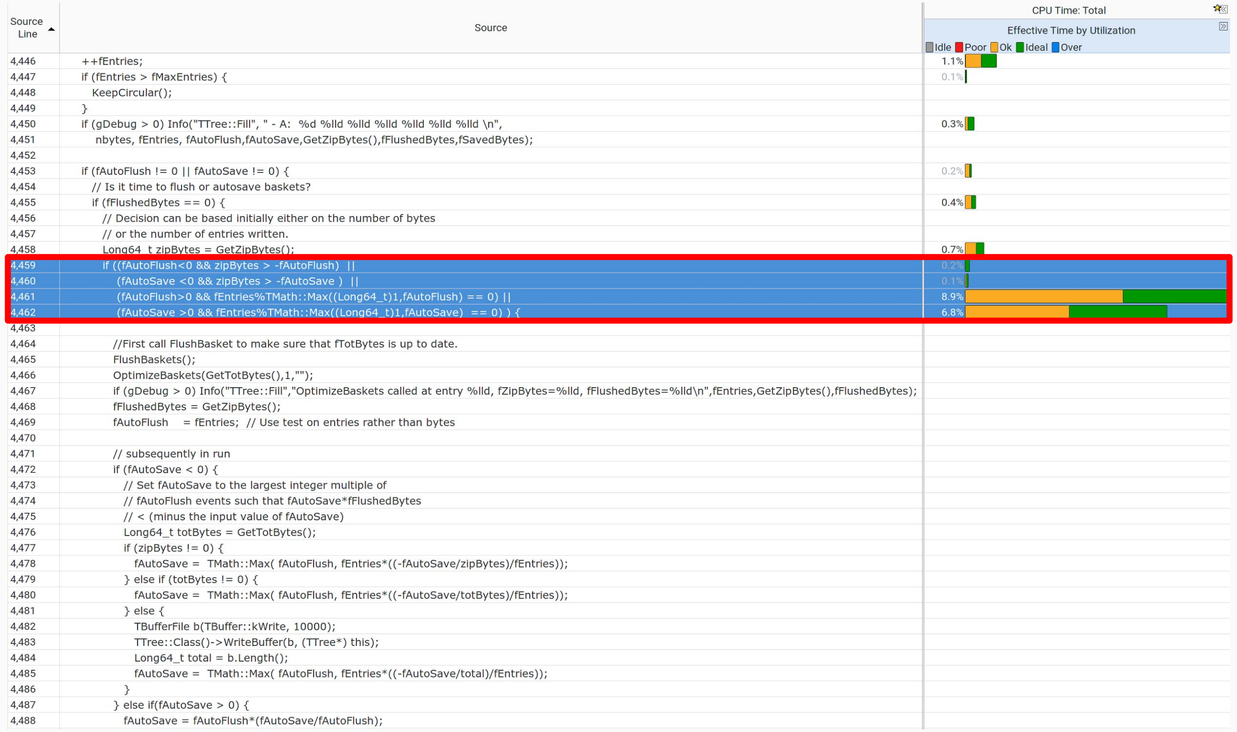
No more type system queries, only one wait per thread



Sync Object / Function / Call Stack	Wait Time by Thread Concurrency ▼ ⬜ Idle 🟥 Poor 🟧 Ok 🟩 Ideal 🟦 Over	Wait Count	Module	Object Type	Function (Full)
▼ Mutex 0xf7afa33f	50.0%	151		Mutex	
▼ _ZN10TLockGuardC4EP13TVirtualMutex	27.6%	68	libCore.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
▼ TDirectory::RegisterContext	27.6%	68	libCore.so	Mutex	TDirectory::RegisterContext(TDirectory::TContext*)
▼ ↖ _ZN10TDirectory8TContextC4EPS_	23.5%	55	libRIO.so	Mutex	_ZN10TDirectory8TContextC4EPS_
▼ ↖ TDirectoryFile::WriteTObject	23.5%	55	libRIO.so	Mutex	TDirectoryFile::WriteTObject(TObject const*, char const*, char const*, int)
▼ ↖ TTree::AutoSave	23.5%	55	libTree.so	Mutex	TTree::AutoSave(char const*)
▶ ↖ TTree::Fill	23.5%	55	libTree.so	Mutex	TTree::Fill(void)
▼ ↖ _ZN10TDirectory8TContextC4EPS_	3.0%	6	libRIO.so	Mutex	_ZN10TDirectory8TContextC4EPS_
▶ ↖ TDirectoryFile::Write	3.0%	6	libRIO.so	Mutex	TDirectoryFile::Write(char const*, int, int)
▼ ↖ _ZN10TDirectory8TContextC4EPS_	1.1%	8	libRIO.so	Mutex	_ZN10TDirectory8TContextC4EPS_
▶ ↖ TMemFile::ResetAfterMerge	1.1%	8	libRIO.so	Mutex	TMemFile::ResetAfterMerge(TFileMergeInfo*)
▼ _ZN10TLockGuardC4EP13TVirtualMutex	18.6%	56	libRIO.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
▼ ↖ TMemFile::ResetAfterMerge	18.6%	56	libRIO.so	Mutex	TMemFile::ResetAfterMerge(TFileMergeInfo*)
▶ ↖ ROOT::Experimental::TBufferMergerFile::Write	18.6%	56	libRIO.so	Mutex	ROOT::Experimental::TBufferMergerFile::Write(char const*, int, int)
▼ _ZN10TLockGuardC4EP13TVirtualMutex	1.3%	15	libCling.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
▶ ↖ TClingClassInfo::GetBaseOffset	1.3%	15	libCling.so	Mutex	TClingClassInfo::GetBaseOffset(TClingClassInfo*, void*, bool)
▼ _ZN10TLockGuardC4EP13TVirtualMutex	0.9%	3	libRIO.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
▶ ↖ TStreamerInfo::Streamer	0.9%	3	libRIO.so	Mutex	TStreamerInfo::Streamer(TBuffer&)
▼ _ZN10TLockGuardC4EP13TVirtualMutex	0.8%	1	libRIO.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
▼ ↖ TFile::Init	0.8%	1	libRIO.so	Mutex	TFile::Init(bool)
▼ ↖ TMemFile::ResetAfterMerge	0.8%	1	libRIO.so	Mutex	TMemFile::ResetAfterMerge(TFileMergeInfo*)
▶ ↖ ROOT::Experimental::TBufferMergerFile::Write	0.8%	1	libRIO.so	Mutex	ROOT::Experimental::TBufferMergerFile::Write(char const*, int, int)
▶ _ZN10TLockGuardC4EP13TVirtualMutex	0.3%	3	libCore.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
▶ _ZN10TLockGuardC4EP13TVirtualMutex	0.3%	1	libCore.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
▶ _ZN10TLockGuardC4EP13TVirtualMutex	0.2%	2	libRIO.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
▶ _ZN10TLockGuardC4EP13TVirtualMutex	0.1%	1	libCore.so	Mutex	_ZN10TLockGuardC4EP13TVirtualMutex
▶ TPosixMutex::Restore	0.0%	1	libThread.so	Mutex	TPosixMutex::Restore(std::unique_ptr<TVirtualMutex::State, std::default_delete<TVirtualMutex::State>>&&)
Selected 2 rows:	42.1%	111			

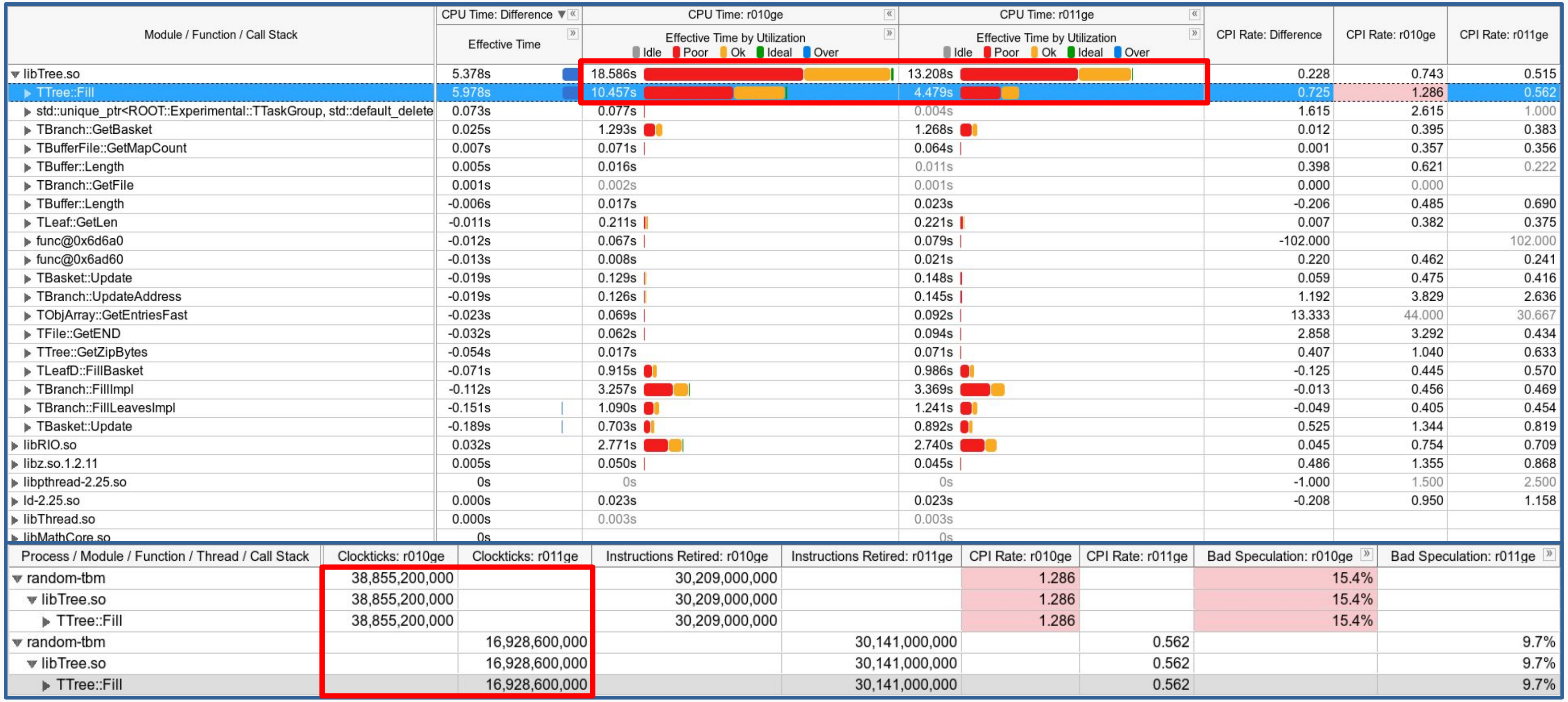
Optimization of TTree::Fill()

Divisions take up many CPU cycles for useless work



```
Source                                  Source                                                        CPU Time: Total
Line                                                                                           Effective Time by Utilization
                                                                                               ▨Idle 🟥Poor 🟧Ok 🟩Ideal 🟦Over
4,446      ++fEntries;                                                                                      1.1%
4,447      if (fEntries > fMaxEntries) {                                                                    0.1%
4,448         KeepCircular();
4,449      }
4,450      if (gDebug > 0) Info("TTree::Fill", " - A:  %d %lld %lld %lld %lld %lld %lld \n",                0.3%
4,451          nbytes, fEntries, fAutoFlush,fAutoSave,GetZipBytes(),fFlushedBytes,fSavedBytes);
4,452
4,453      if (fAutoFlush != 0 || fAutoSave != 0) {                                                         0.2%
4,454         // Is it time to flush or autosave baskets?
4,455         if (fFlushedBytes == 0) {                                                                     0.4%
4,456            // Decision can be based initially either on the number of bytes
4,457            // or the number of entries written.
4,458            Long64_t zipBytes = GetZipBytes();                                                         0.7%
4,459            if ((fAutoFlush<0 && zipBytes > -fAutoFlush)  ||                                           0.2%
4,460                (fAutoSave <0 && zipBytes > -fAutoSave )  ||                                           0.1%
4,461                (fAutoFlush>0 && fEntries%TMath::Max((Long64_t)1,fAutoFlush) == 0) ||                  8.9%
4,462                (fAutoSave >0 && fEntries%TMath::Max((Long64_t)1,fAutoSave)  == 0) ) {                 6.8%
4,463
4,464               //First call FlushBasket to make sure that fTotBytes is up to date.
4,465               FlushBaskets();
4,466               OptimizeBaskets(GetTotBytes(),1,"");
4,467               if (gDebug > 0) Info("TTree::Fill","OptimizeBaskets called at entry %lld, fZipBytes=%lld, fFlushedBytes=%lld\n",fEntries,GetZipBytes(),fFlushedBytes);
4,468               fFlushedBytes = GetZipBytes();
4,469               fAutoFlush    = fEntries;  // Use test on entries rather than bytes
4,470
4,471               // subsequently in run
4,472               if (fAutoSave < 0) {
4,473                  // Set fAutoSave to the largest integer multiple of
4,474                  // fAutoFlush events such that fAutoSave*fFlushedBytes
4,475                  // < (minus the input value of fAutoSave)
4,476                  Long64_t totBytes = GetTotBytes();
4,477                  if (zipBytes != 0) {
4,478                     fAutoSave =  TMath::Max( fAutoFlush, fEntries*((-fAutoSave/zipBytes)/fEntries));
4,479                  } else if (totBytes != 0) {
4,480                     fAutoSave =  TMath::Max( fAutoFlush, fEntries*((-fAutoSave/totBytes)/fEntries));
4,481                  } else {
4,482                     TBufferFile b(TBuffer::kWrite, 10000);
4,483                     TTree::Class()->WriteBuffer(b, (TTree*) this);
4,484                     Long64_t total = b.Length();
4,485                     fAutoSave =  TMath::Max( fAutoFlush, fEntries*((-fAutoSave/total)/fEntries));
4,486                  }
4,487               } else if(fAutoSave > 0) {
4,488                  fAutoSave = fAutoFlush*(fAutoSave/fAutoFlush);
```

Optimization of TTree::Fill()



Module / Function / Call Stack	CPU Time: Difference Effective Time	CPU Time: r010ge Effective Time by Utilization (Idle, Poor, Ok, Ideal, Over)	CPU Time: r011ge Effective Time by Utilization (Idle, Poor, Ok, Ideal, Over)	CPI Rate: Difference	CPI Rate: r010ge	CPI Rate: r011ge
▼ libTree.so	5.378s	18.586s	13.208s	0.228	0.743	0.515
▶ TTree::Fill	5.978s	10.457s	4.479s	0.725	1.286	0.562
▶ std::unique_ptr<ROOT::Experimental::TTaskGroup, std::default_delete	0.073s	0.077s	0.004s	1.615	2.615	1.000
▶ TBranch::GetBasket	0.025s	1.293s	1.268s	0.012	0.395	0.383
▶ TBufferFile::GetMapCount	0.007s	0.071s	0.064s	0.001	0.357	0.356
▶ TBuffer::Length	0.005s	0.016s	0.011s	0.398	0.621	0.222
▶ TBranch::GetFile	0.001s	0.002s	0.001s	0.000	0.000	
▶ TBuffer::Length	-0.006s	0.017s	0.023s	-0.206	0.485	0.690
▶ TLeaf::GetLen	-0.011s	0.211s	0.221s	0.007	0.382	0.375
▶ func@0x6d6a0	-0.012s	0.067s	0.079s	-102.000		102.000
▶ func@0x6ad60	-0.013s	0.008s	0.021s	0.220	0.462	0.241
▶ TBasket::Update	-0.019s	0.129s	0.148s	0.059	0.475	0.416
▶ TBranch::UpdateAddress	-0.019s	0.126s	0.145s	1.192	3.829	2.636
▶ TObjArray::GetEntriesFast	-0.023s	0.069s	0.092s	13.333	44.000	30.667
▶ TFile::GetEND	-0.032s	0.062s	0.094s	2.858	3.292	0.434
▶ TTree::GetZipBytes	-0.054s	0.017s	0.071s	0.407	1.040	0.633
▶ TLeafD::FillBasket	-0.071s	0.915s	0.986s	-0.125	0.445	0.570
▶ TBranch::FillImpl	-0.112s	3.257s	3.369s	-0.013	0.456	0.469
▶ TBranch::FillLeavesImpl	-0.151s	1.090s	1.241s	-0.049	0.405	0.454
▶ TBasket::Update	-0.189s	0.703s	0.892s	0.525	1.344	0.819
▶ libRIO.so	0.032s	2.771s	2.740s	0.045	0.754	0.709
▶ libz.so.1.2.11	0.005s	0.050s	0.045s	0.486	1.355	0.868
▶ libpthread-2.25.so	0s	0s	0s	-1.000	1.500	2.500
▶ ld-2.25.so	0.000s	0.023s	0.023s	-0.208	0.950	1.158
▶ libThread.so	0.000s	0.003s	0.003s			
▶ libMathCore.so	0s		0s			

Process / Module / Function / Thread / Call Stack	Clockticks: r010ge	Clockticks: r011ge	Instructions Retired: r010ge	Instructions Retired: r011ge	CPI Rate: r010ge	CPI Rate: r011ge	Bad Speculation: r010ge	Bad Speculation: r011ge
▼ random-tbm	38,855,200,000		30,209,000,000		1.286		15.4%	
▼ libTree.so	38,855,200,000		30,209,000,000		1.286		15.4%	
▶ TTree::Fill	38,855,200,000		30,209,000,000		1.286		15.4%	
▼ random-tbm		16,928,600,000		30,141,000,000		0.562		9.7%
▼ libTree.so		16,928,600,000		30,141,000,000		0.562		9.7%
▶ TTree::Fill		16,928,600,000		30,141,000,000		0.562		9.7%

Bottomline

- ROOT continues to parallelise its I/O subsystem
  - Focus not only on experiments’ data processing, but also on analysis
- Parallel writing to single output file via **`TBufferMerger`**
  - Leveraged by **`TDataFrame`** already with snapshot action
  - Good performance, can saturate an SSD

Changes already in master for ROOT 6.12 release:

- Optimised **`TTree::Fill()`** function avoids divisions
- Optimised parallel merging with **`TBufferMerger`**
- Output thread no longer does excessive compression work
- Callback function allows seamless framework integration

Questions?