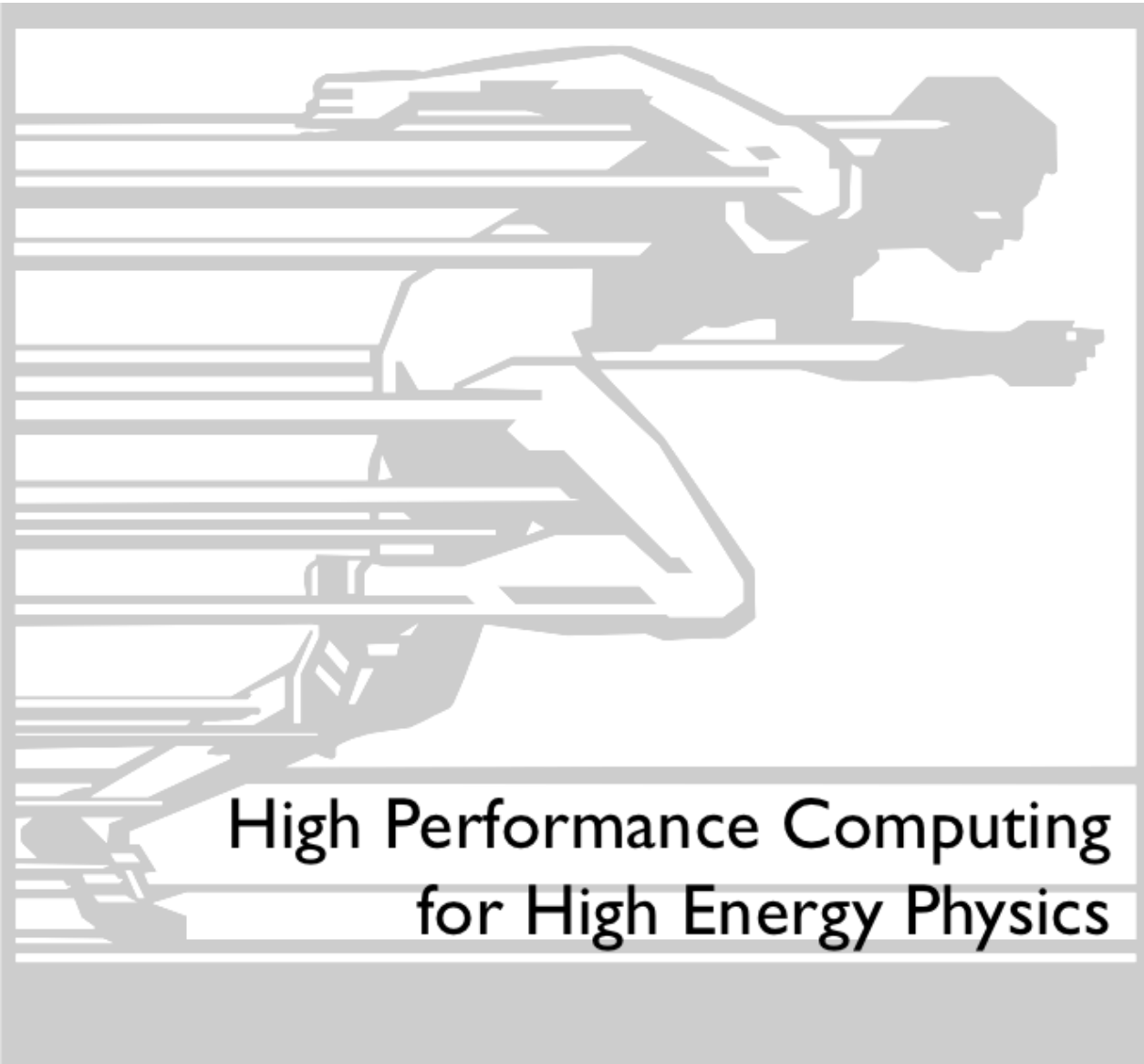


Yet another Malloc Profiler (this time based on `std::stacktrace`)

Vincenzo Innocente
CERN
CMS Experiment



High Performance Computing
for High Energy Physics

Motivations

- Replace igprof (unmaintained)
- Do not use external libraries (they need to be updated, and code ported)
- Keep code simple (easy maintenance)
- Keep output format easy (and possibly human readable)
- Use existing tools for analysis and display

std::backtrace

- In C++23 standard
- “supported” in gcc since v12
- Need compiler to be specifically configured
 - --enable-libstdcxx-backtrace=yes
- Tool need to be linked with a static library
 - -lstdc++_libbacktrace in gcc12, -lstdc++exp in gcc14
- Three bugs found: two fixed in the main branches
 - Third trivial (affect only stacktrace header file)

112348	gcc	libstdc+	unassigned@gcc.gnu.org	UNCO	---	[C++23] defect in struct hash<basic_stacktrace< Allocator>>
112263	gcc	libbackt	unassigned@gcc.gnu.org	RESO	FIXE	[C++23] std::stacktrace does not identify symbols in shared library
111936	gcc	libstdc+	redi@gcc.gnu.org	RESO	FIXE	std::stacktrace cannot be used in a shared library

Resources:

<https://stackoverflow.com/questions/3899870/how-to-print-a-stack-trace-whenever-a-certain-function-is-called/54365144>

Instrumentation

- As everybody else does
 - LD_PRELOAD
 - dlsym(RTLD_NEXT, "malloc"); etc
 - std::stacktrace objects kept in an unordered_map
 - One could try to use a Patricia Trie
 - Accumulation by thread (ncalls, memtot, memlive, max-memlive)
 - Final aggregation and symbol-name resolution at dump time
 - Filtering and “Remangling”: In process (dump-time) and post processing
 - Result is a file containing ;-separated-stacktraces with a value associated
- <https://github.com/VinInn/MallocProfiler/tree/main>

Visualization tools

- FlameGraph
 - <https://www.brendangregg.com/flamegraphs.html>
 - <https://github.com/brendangregg/FlameGraph>
 - `FlameGraph/flamegraph.pl --width 2400 step3_DumpDoEvent.md > /tmp/step3_DumpDoEvent.svg`
- Speedscope (thanks to Giulio Eulisse for pointing it out)
 - <https://github.com/jlfwong/speedscope>
 - Just open the web-app and drop the file
- Just try
 - `scp lxplus8.cern.ch:/afs/cern.ch/user/i/innocent/public/step3_DumpDoEvent.md .`

caveats

- Cling (via llvm) seems to “interact” with gcc backtrace data structures:
 - Hang deep in `gcc_src/libgcc/unwind-dw2-btree.h`
 - Solution: overload two functions and stop malloc-recording in there
- Gcc unwind calls `dl_iterate_phdr` that seems to be protected by a mutex
 - Stacktrace recording the facto serializes
- Processing 100-event reco-relval takes ~2 hours (with recording threshold of 128 bytes)
- Dump (includes symbol name resolution) is very slow (several minutes)

Remangling

- C++ symbol names can be long (boost::spirit? 30Kbytes!)
- Many stacktrace details are not of interest for the analysis in hand
 - Everything below Tclass, TFormula, std::regex and boost::spirit for instance
 - Everything before tbb::detail::d1::function_task
- Many names can be shortened
 - string, vector, etc
 - , allocator<...> can be removed
- The whole signature is often not of interest (function name is enough)

Filtering

- The file is simple text: best post-processing filtering is grep and sed!

3 is memtot, 4 is memlive

```

grep '_mpTrace_' $1.mdr | sed 's/\;\\{1,\\}/;/g' | sed 's/^/;/;' | cut -f 1 -d ';' | grep -c 'TClass step3_DumpNew.mdr'
| sed 's/_mpTrace_;_start;__libc_start_main;main;main::{lambda()#1}::operator();tbb::detail::r1::task_arena_impl::execute;'
tbb::detail::d1::task_arena_function<main::{lambda()#1}::operator();tbb::detail::r1::task_arena_impl::execute;'
| sed 's/_mpTrace.*tbb::detail::d1::function_task/function_task/g'
| sed 's/std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char>> >/string/g'
| sed 's/std::basic_string_view<char, std::char_traits<char> >/string_view/g' \
| sed 's/operator new;malloc;///g' \
| sed 's/malloc;///g' \
| sed 's/unsigned /u/g' | sed 's/operator/op/g' | sed 's/std:://g' \
| sed 's/const//g' | sed 's/&//g' | sed 's/ //g' | sed 's/:::/g' | tr '$' ' ' > $1.mdr

```

```
[innocent@patatrack01 11634.o_TTbar_14TeV+2021]$
```

```
wc step3_DumpNew.mdr
```

1114944 117815162 **3,354,955,155**

```
grep -ic cling step3_DumpNew.mdr
```

163652

```
figrep -sT Class step3_DumpNew.mdr
```

```

b3/84/81}::operator());tbb::detail::r1::task_arena_impl::execute;

```

```
operator() {
    for (int i = 0; i < n; i++) {
        cout << "Step " << i << " completed\n";
    }
}
```

as 6/210

```
regstrings.exe step3_DumpNew.mdr
```

/1682g_view/g'\

```
grep -ic spirit step3_DumpNew.mdr
```

322480

```
grep doEvent step3_DumpNew.mdr | wc
```

>29023428882728 888,851,647

wc step3_DumpDoEvent.md

290194 580388 **333,669,635**

example

```
# 3 is memtot, 4 is memlive
grep '_mpTrace_' $1.mdr | sed 's/\;{\1,\};/;/g' | sed 's/^;//' | cut -f1,4 -d'$' \
| sed 's/_mpTrace_;\_start;__libc_start_main;main;main::{lambda()#1}::operator();tbb::detail::r1::task_arena_impl::execute;
tbb::detail::d1::task_arena_function<main::{lambda()#1}::operator>() const::{lambda()#1}, void>::operator();//' \
| sed 's/_mpTrace.*tbb::detail::d1::function_task/function_task/' \
| sed 's/std::__cxx11::basic_string<char, std::char_traits<char> >/string/g' \
| sed 's/std::basic_string_view<char, std::char_traits<char> >/string_view/g' \
| sed 's/operator new;malloc;//g' \
| sed 's/malloc;//g' \
| sed 's/unsigned /u/g' | sed 's/operator/op/g' | sed 's/std:://g' \
| sed 's/const//g' | sed 's/&//g' | sed 's/ //g' | sed 's/:::/:/g' | tr '$' ' ' > $1.md
```

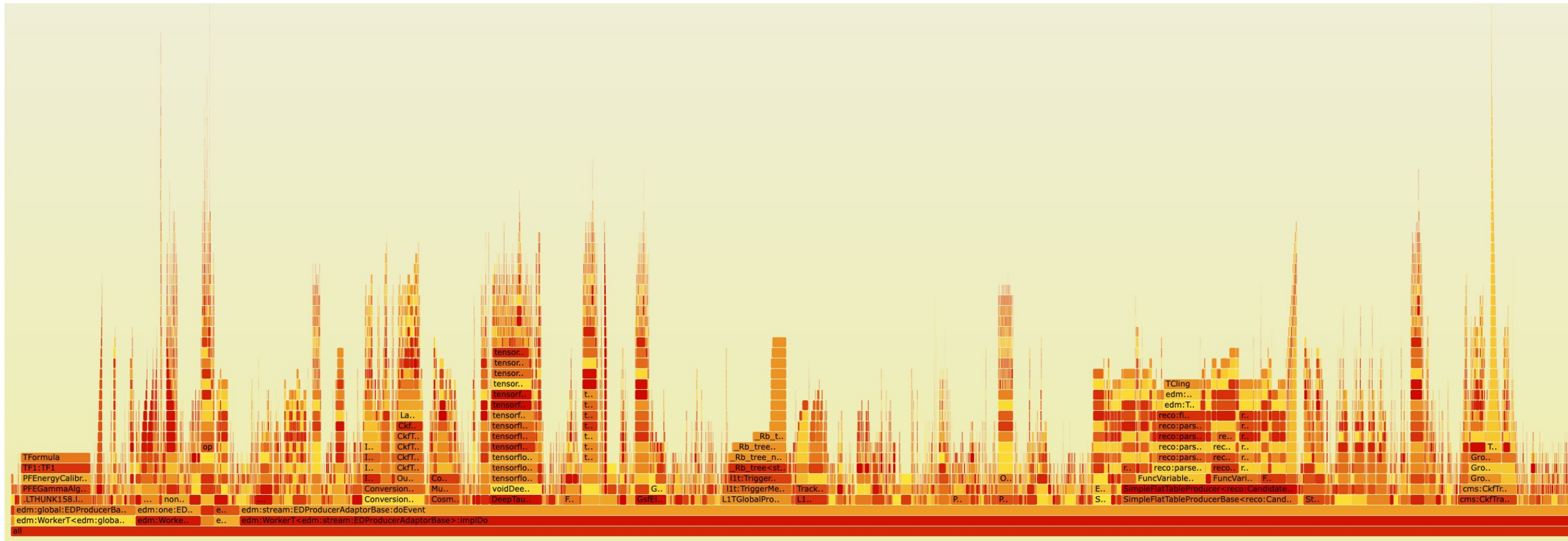
Performance (and some results)

- 100 events, 4 threads, step3 (reco) wf 11634.0_TTbar_14TeV+2021
 - Threshold 128 Bytes (below threshold stacktrace is not recorded)
 - Elapsed time 6950 seconds: (~18 minutes before event 5 starts)
 - Total Number of malloc calls 579,980,716
 - Total MemTot 9.90328e+10 (100GB!)
 - Total Max Memlive 3,929,812,405
 - Raw dump: wc 1,078,553 116,416,953 328,1194,659 memdump_2003909_1.mdr
 - SmallAllocations: (<128 Bytes)
 - Number of malloc calls 474,512,716
 - MemTot 1.80925e+10 (18GB)
 - Typical MemLive 524,182,207
 - Inverse Threshold (record stacktrace only for <128 Bytes allocations)
 - 55 events : 18500 seconds (~2hours before event 5 starts)
 - Raw dump: wc 252,8393 2,1326,8892 6,504,541,718 memdump_99644_15.mdr

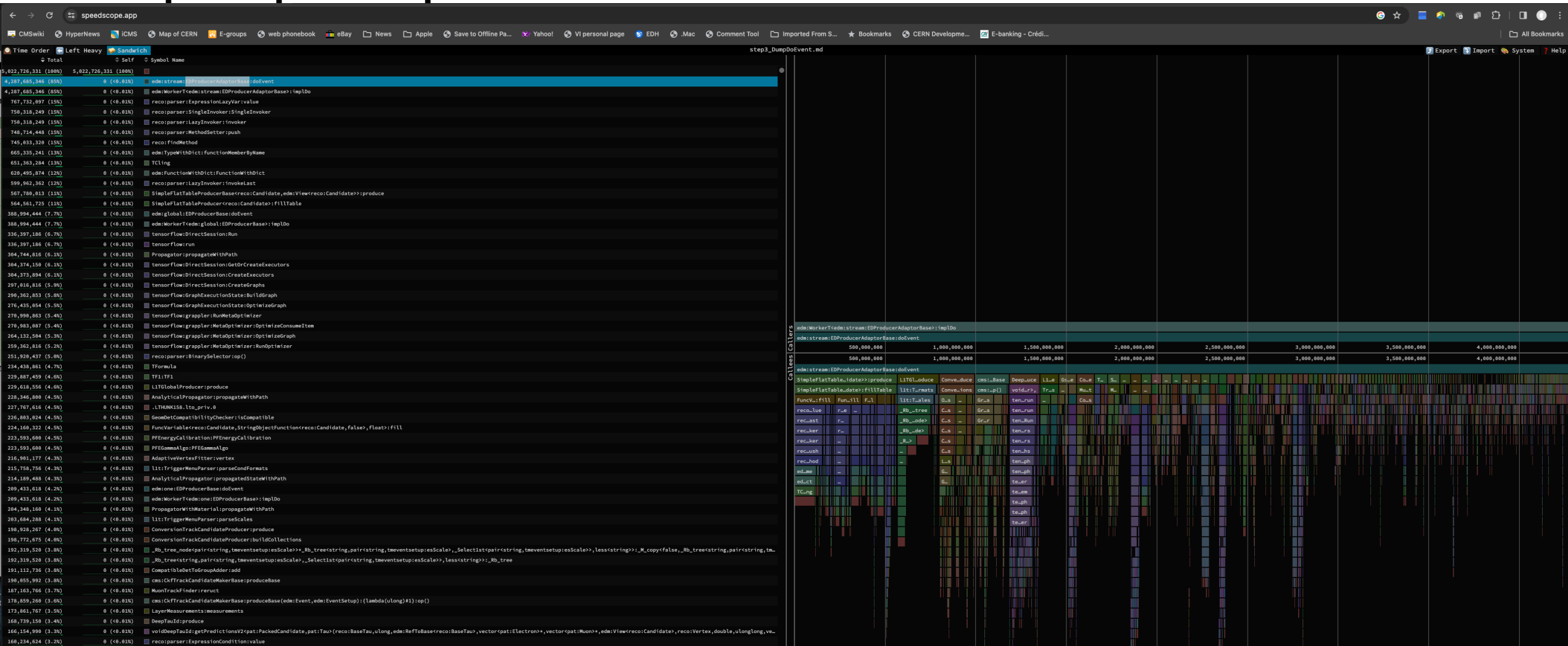
Current status and future developments

BACKUP

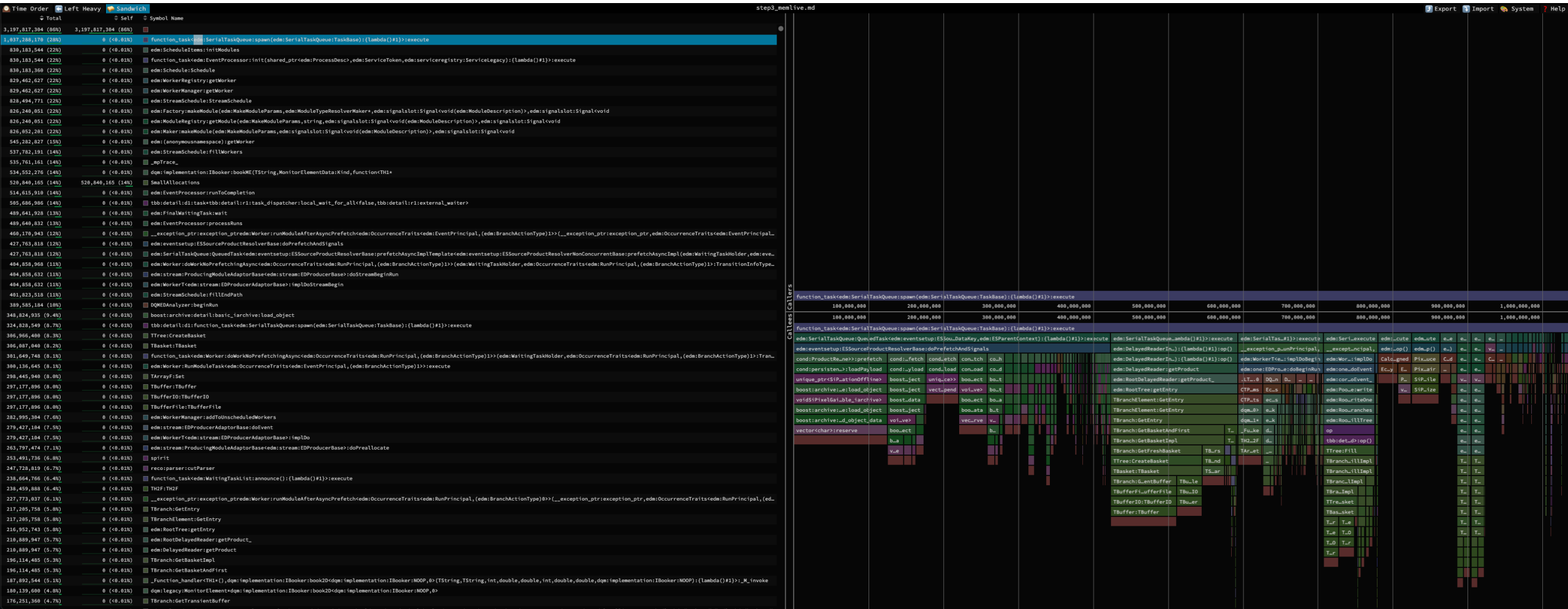
Flamegraph (memtot in "doEvent")



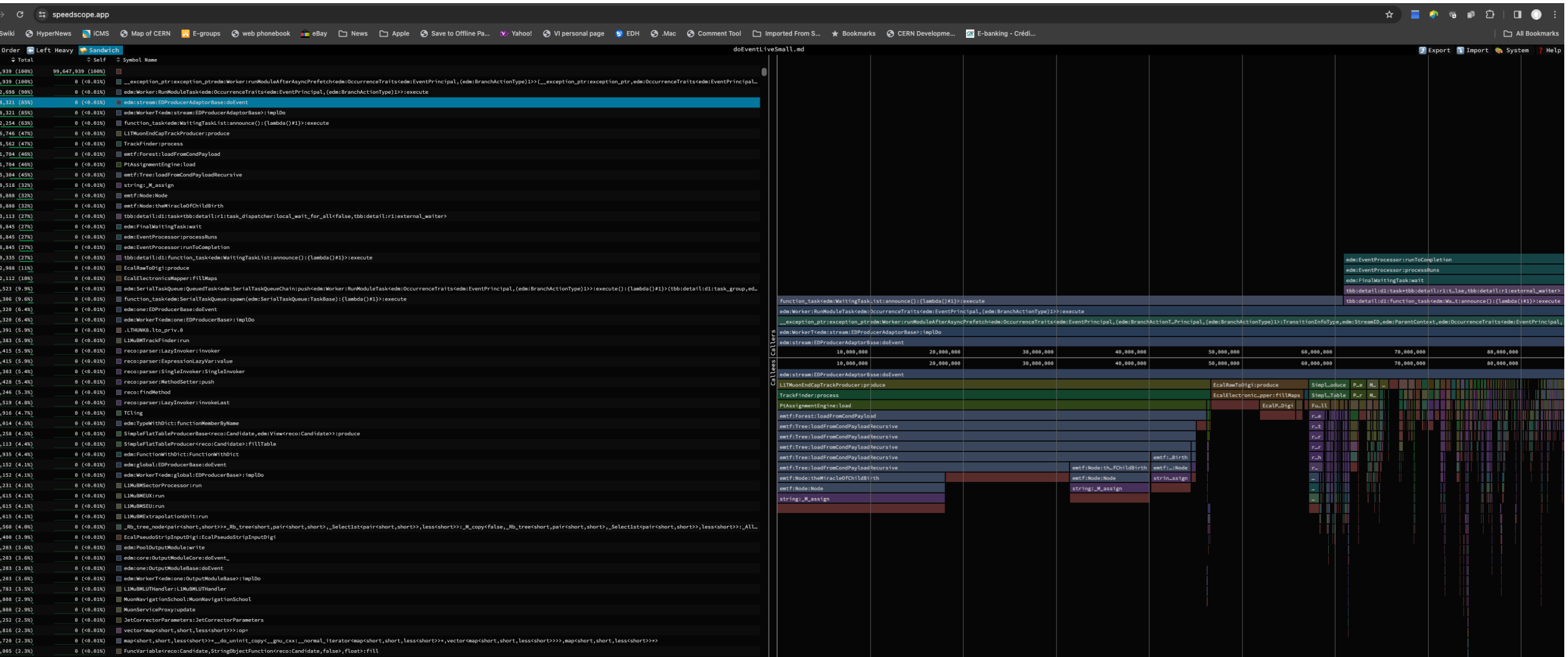
SpeepScope: MemTot >128B in doEvent



Speedscope (memlive >128B full)

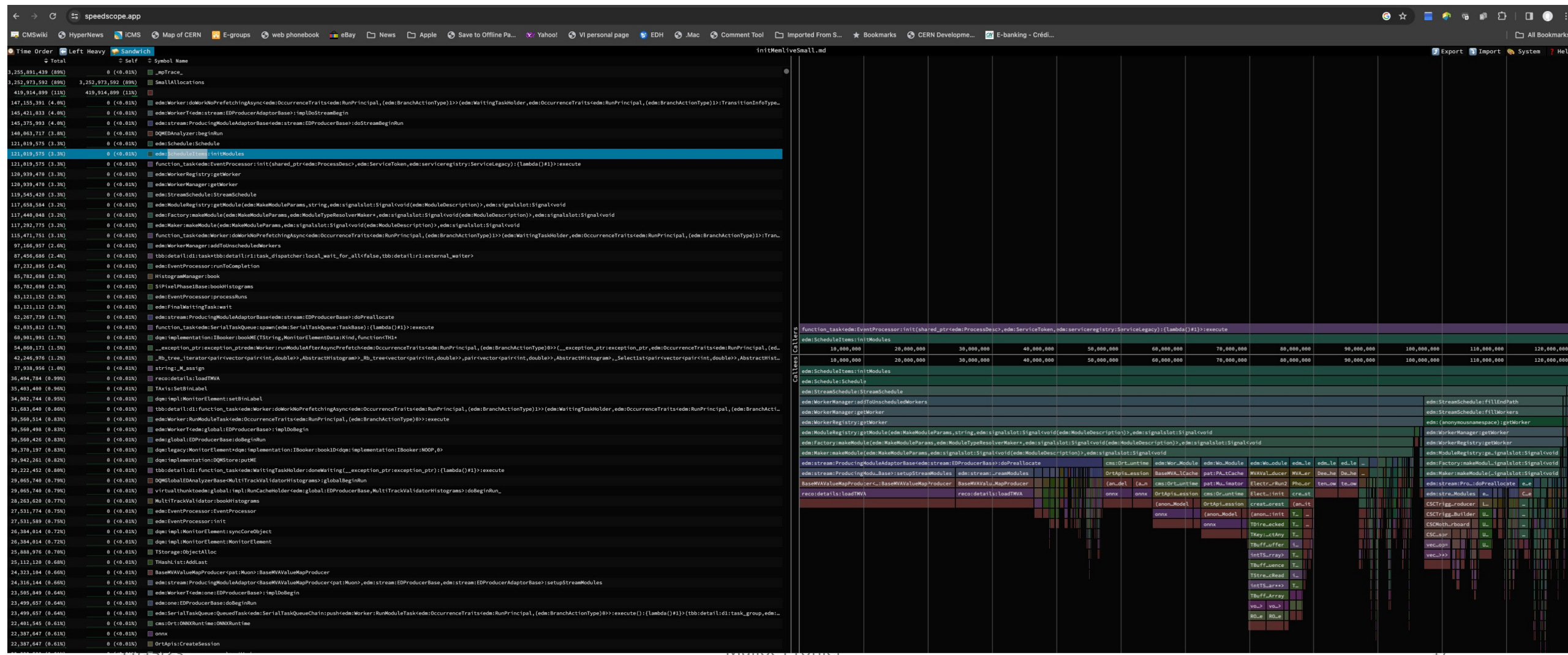


Speedscope: memlive SmallAlloc in doEvent



Speedscope: memlive SmallAlloc NOT in doEvent

```
grep -v doEvent memliveSmall.md | grep -v spirit | sed 's/onnx.*\;/onnx\;/g' | sed 's/tensorflow.*\;/tensorflow\;/g' | sed 's/TMVA.*\;/TMVA\;/g' | sed 's/Hashtable.*\;/Hashtable\;/g' | grep -v "TClass; "
```



tail -n 11 memliveSmall.md

[illegible]