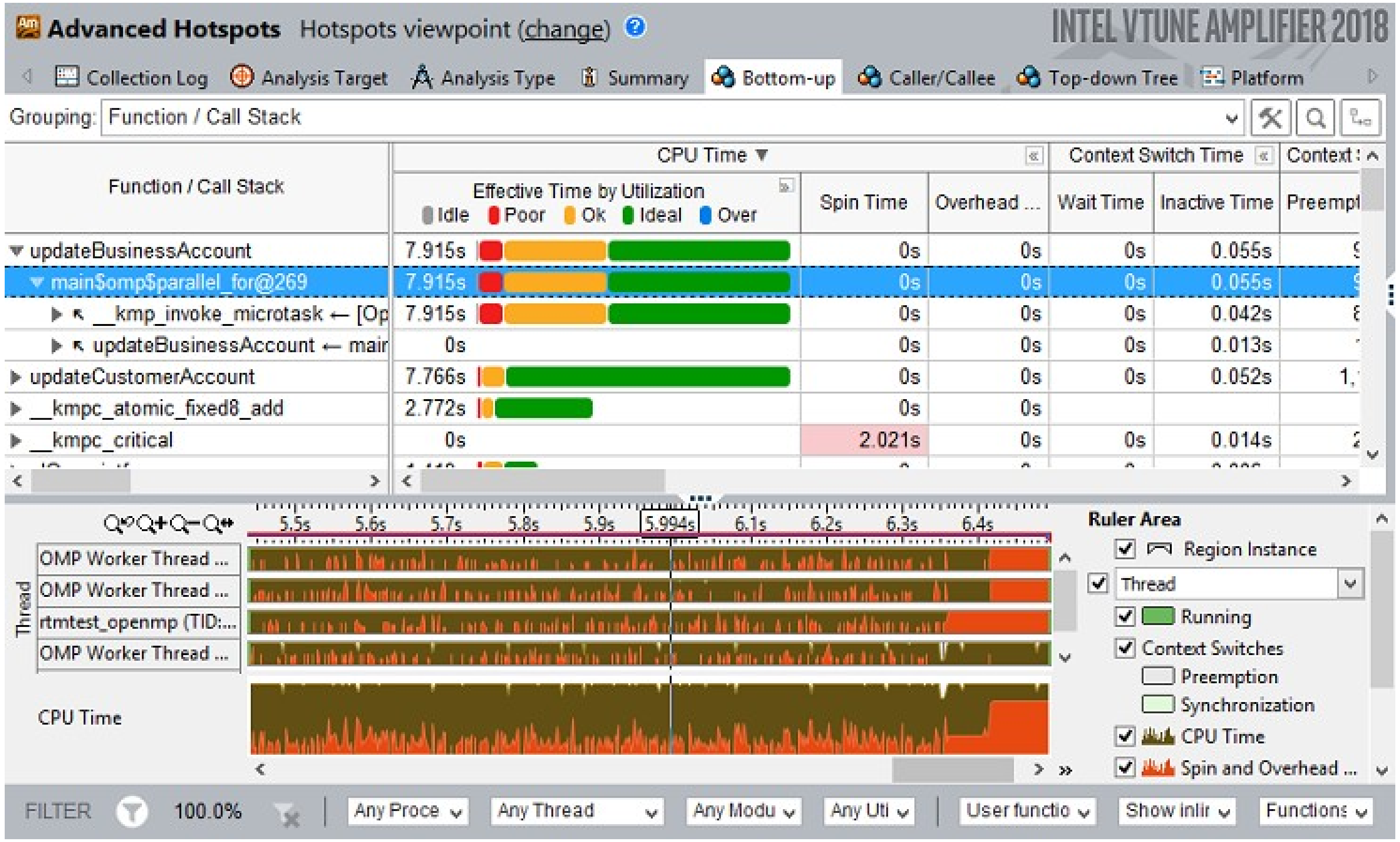
(HPC) Performance Tools Comparison

Opinion

TECHNISCHE UNIVERSITÄT DRESDEN

Intel VTune Amplifier

- Commercial



Grouping: Function / Call Stack

Function / Call Stack	CPU Time ▼			Context Switch Time		Context :
	Effective Time by Utilization Idle 🟥 Poor 🟧 Ok 🟩 Ideal 🟦 Over	Spin Time	Overhead ...	Wait Time	Inactive Time	Preempt
▼ updateBusinessAccount	7.915s	0s	0s	0s	0.055s	9
▼ main$omp$parallel_for@269	7.915s	0s	0s	0s	0.055s	9
▶ ↖ __kmp_invoke_microtask ← [Op	7.915s	0s	0s	0s	0.042s	8
▶ ↖ updateBusinessAccount ← mair	0s	0s	0s	0s	0.013s	1
▶ updateCustomerAccount	7.766s	0s	0s	0s	0.052s	1,
▶ __kmpc_atomic_fixed8_add	2.772s	0s	0s			
▶ __kmpc_critical	0s	2.021s	0s	0s	0.014s	2

Ruler Area

☑ Region Instance

☑ Thread

☑ Running

☑ Context Switches

□ Preemption

□ Synchronization

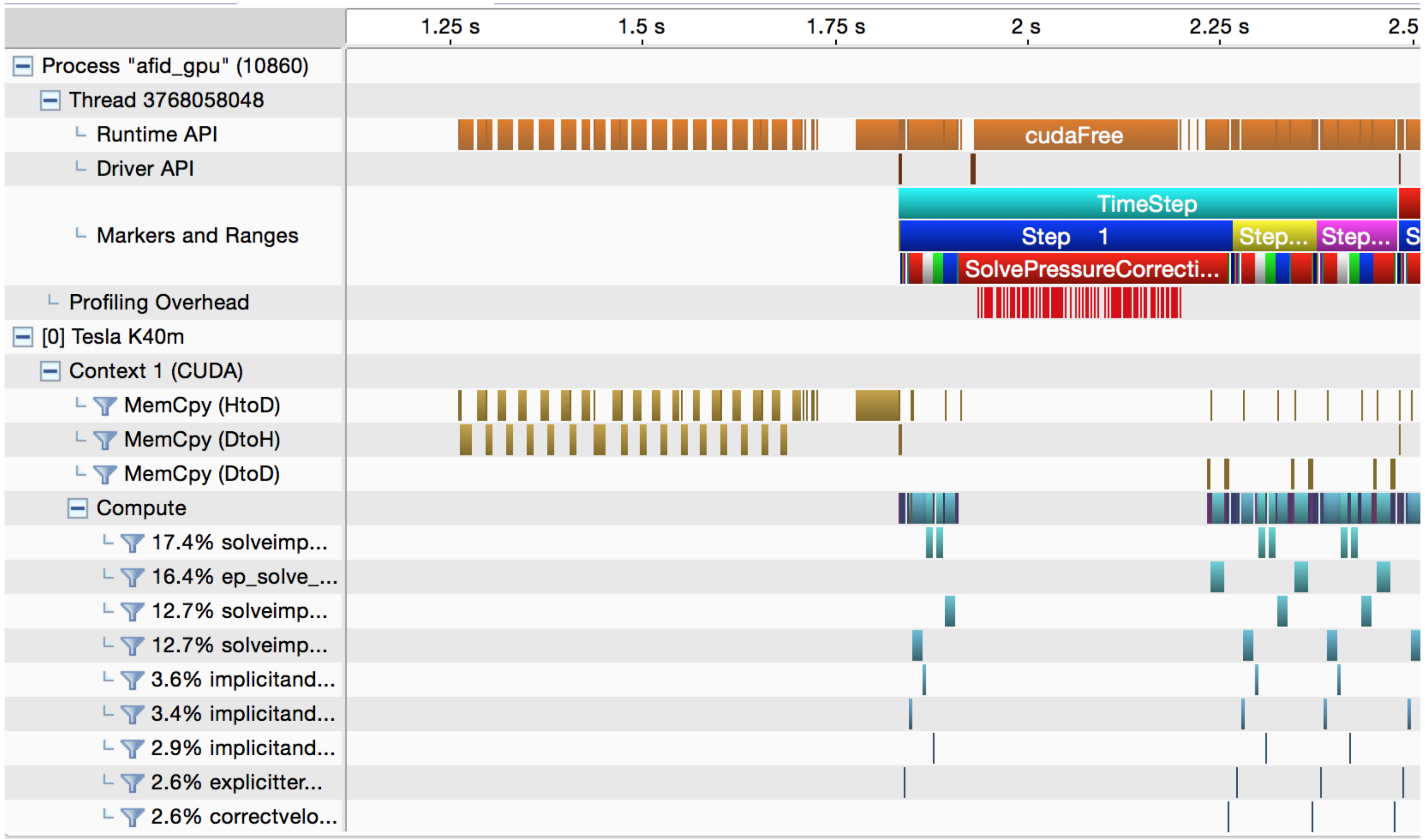
☑ CPU Time

☑ Spin and Overhead ...

FILTER 100.0% | Any Proce | Any Thread | Any Modu | Any Uti | User functio | Show inlir | Functions

NVProf

- Free


1.25 s
1.5 s
1.75 s
2 s
2.25 s
2.5
Process "afid_gpu" (10860)
Thread 3768058048
Runtime API
cudaFree
Driver API
Markers and Ranges
TimeStep
Step 1
Step...
Step...
S
SolvePressureCorrecti...
Profiling Overhead
[0] Tesla K40m
Context 1 (CUDA)
MemCpy (HtoD)
MemCpy (DtoH)
MemCpy (DtoD)
Compute
17.4% solveimp...
16.4% ep_solve_...
12.7% solveimp...
12.7% solveimp...
3.6% implicitand...
3.4% implicitand...
2.9% implicitand...
2.6% explicitter...
2.6% correctvelo...

Linux Perf

- Open Source

- Requires elevated privileges (think sudo)

- http://www.brendangregg.com/perf.html
- https://perf.wiki.kernel.org/index.php/Main_Page

Linux Perf

Stat

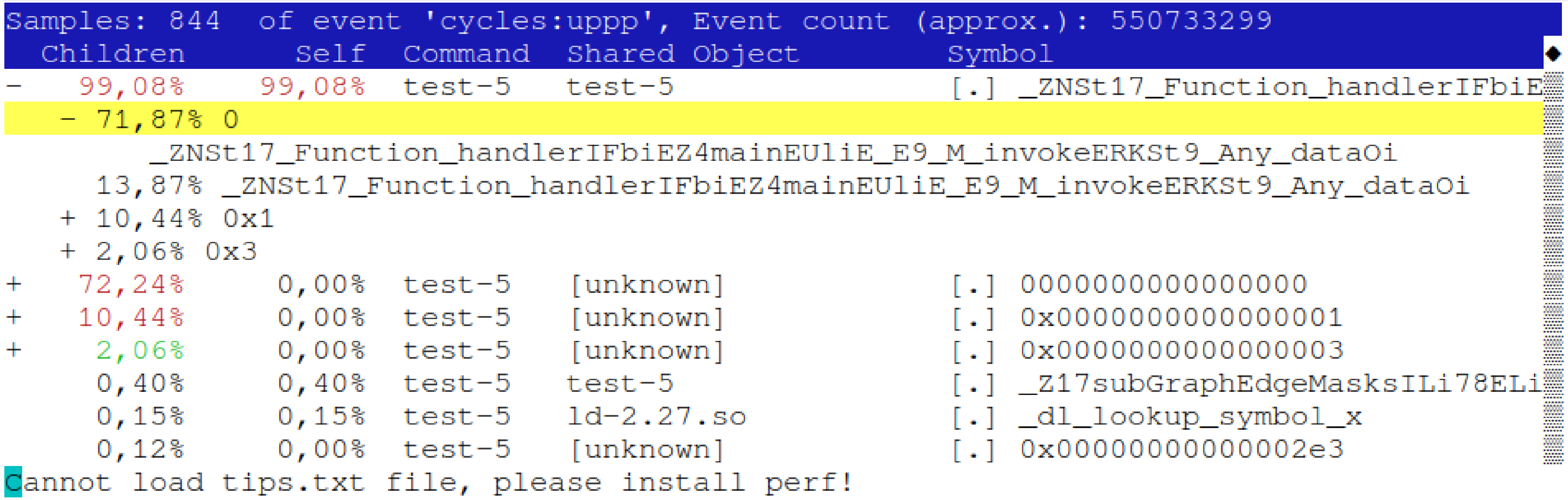
-
  ```
  perf stat ./myapp

  Performance counter stats for './myapp':

         7894,316858      task-clock (msec)         #    1,000 CPUs utilized
                  65      context-switches          #    0,008 K/sec
                   1      cpu-migrations            #    0,000 K/sec
                 247      page-faults               #    0,031 K/sec
      21.509.526.746      cycles                    #    2,725 GHz
      70.936.275.718      instructions              #    3,30  insn per cycle
      12.084.148.187      branches                  # 1530,740 M/sec
         125.221.590      branch-misses             #    1,04% of all branches

         7,896236796 seconds time elapsed
  ```

-
  ```
  perf stat -d [-d -d] ./myapp   # more details
  ```

Linux Perf

Record & Report

-
  ```
  perf record -g ./myapp
  [ perf record: Woken up 1 times to write data ]
  [kernel.kallsyms] with build id 6c9e702f3188393b7d2416f6cd2662ac63ea05c5 not found,
  continuing without symbols
  [ perf record: Captured and wrote 0.058 MB perf.data (844 samples) ]
  ```
- `perf report`

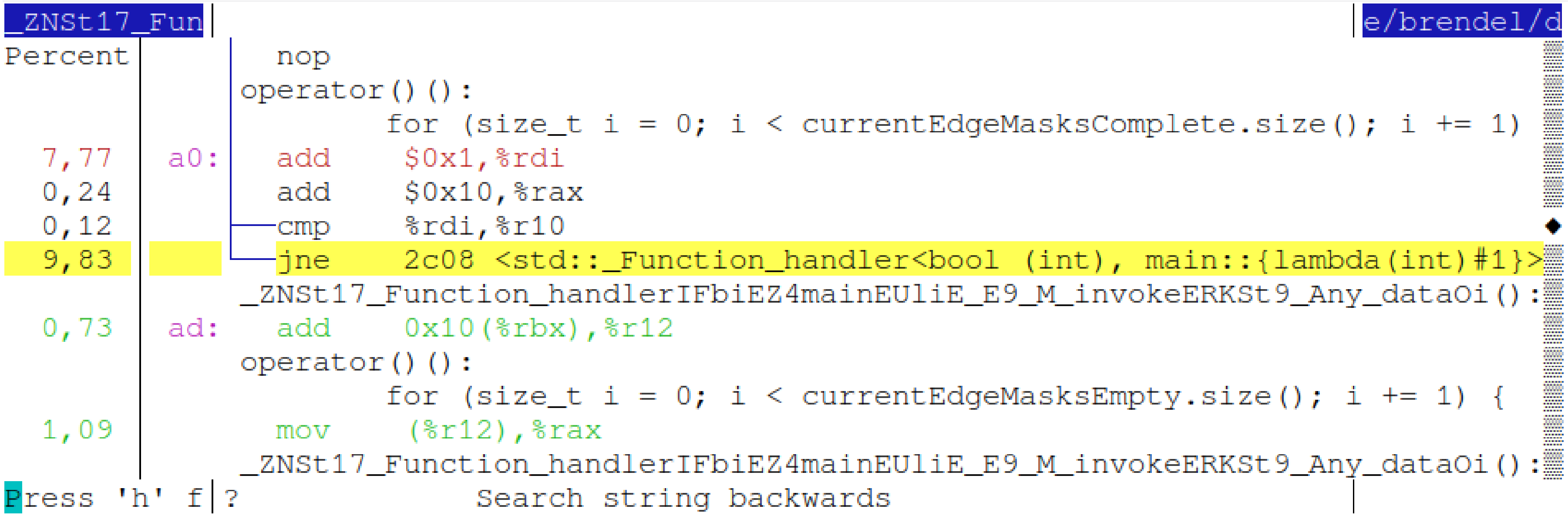


```
Samples: 844  of event 'cycles:uppp', Event count (approx.): 550733299
  Children      Self  Command  Shared Object         Symbol
-   99,08%    99,08%  test-5   test-5                [.] _ZNSt17_Function_handlerIFbiE
   - 71,87% 0
        _ZNSt17_Function_handlerIFbiEZ4mainEUliE_E9_M_invokeERKSt9_Any_dataOi
     13,87% _ZNSt17_Function_handlerIFbiEZ4mainEUliE_E9_M_invokeERKSt9_Any_dataOi
   + 10,44% 0x1
   + 2,06% 0x3
+   72,24%     0,00%  test-5   [unknown]             [.] 0000000000000000
+   10,44%     0,00%  test-5   [unknown]             [.] 0x0000000000000001
+    2,06%     0,00%  test-5   [unknown]             [.] 0x0000000000000003
     0,40%     0,40%  test-5   test-5                [.] _Z17subGraphEdgeMasksILi78ELi
     0,15%     0,15%  test-5   ld-2.27.so            [.] _dl_lookup_symbol_x
     0,12%     0,00%  test-5   [unknown]             [.] 0x00000000000002e3
Cannot load tips.txt file, please install perf!
```

Linux Perf
Record & Report

-
  ```
  perf record -g ./myapp
  [ perf record: Woken up 1 times to write data ]
  [kernel.kallsyms] with build id 6c9e702f3188393b7d2416f6cd2662ac63ea05c5 not found,
  continuing without symbols
  [ perf record: Captured and wrote 0.058 MB perf.data (844 samples) ]
  ```

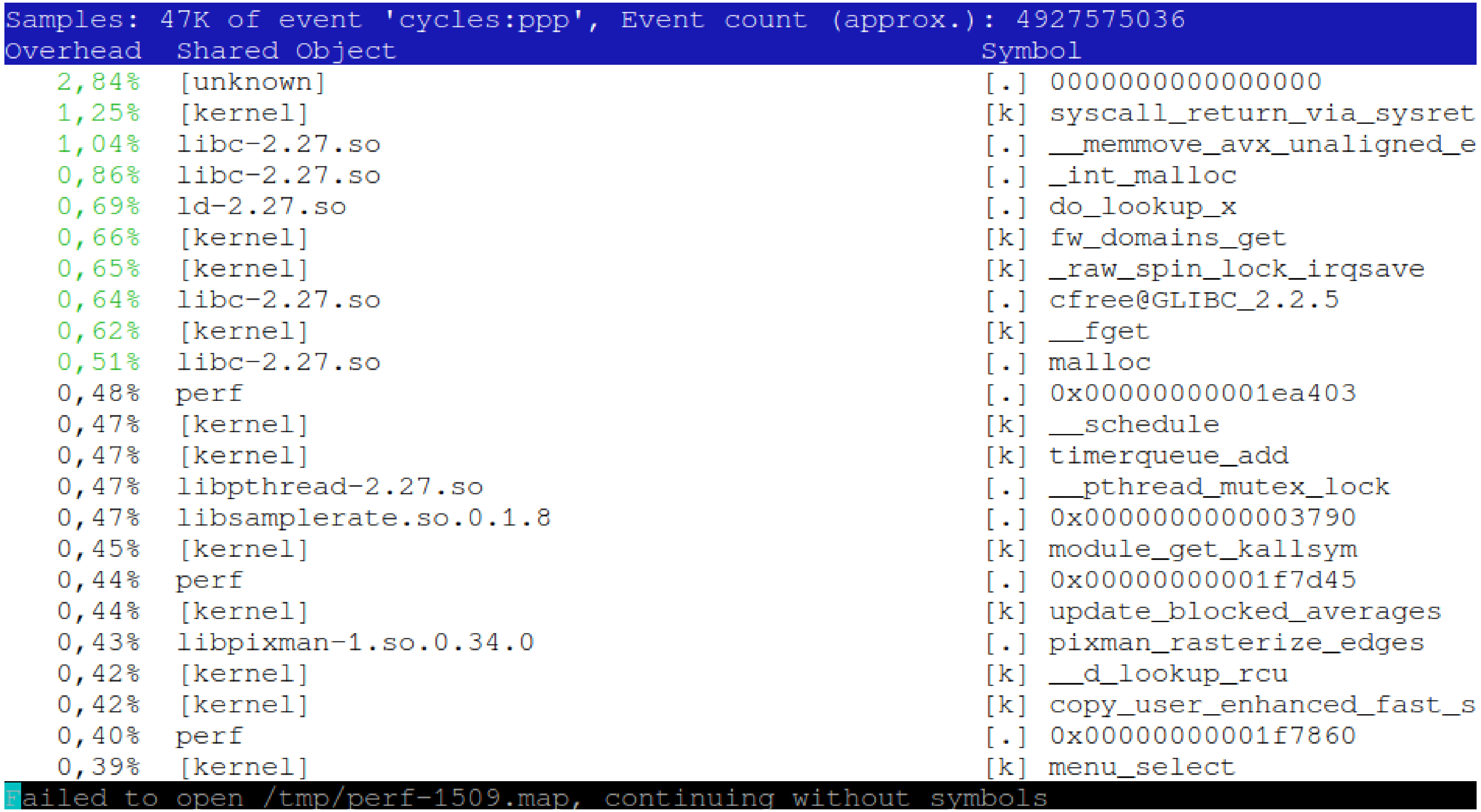
- `perf report`



```
_ZNSt17_Fun                                                    e/brendel/d
Percent          nop
               operator()():
                       for (size_t i = 0; i < currentEdgeMasksComplete.size(); i += 1)
  7,77   a0:     add     $0x1,%rdi
  0,24           add     $0x10,%rax
  0,12           cmp     %rdi,%r10
  9,83           jne     2c08 <std::_Function_handler<bool (int), main::{lambda(int)#1}>
               _ZNSt17_Function_handlerIFbiEZ4mainEUliE_E9_M_invokeERKSt9_Any_dataOi():
  0,73   ad:     add     0x10(%rbx),%r12
               operator()():
                       for (size_t i = 0; i < currentEdgeMasksEmpty.size(); i += 1) {
  1,09           mov     (%r12),%rax
               _ZNSt17_Function_handlerIFbiEZ4mainEUliE_E9_M_invokeERKSt9_Any_dataOi():
Press 'h' f ?             Search string backwards
```

Linux Perf
Top

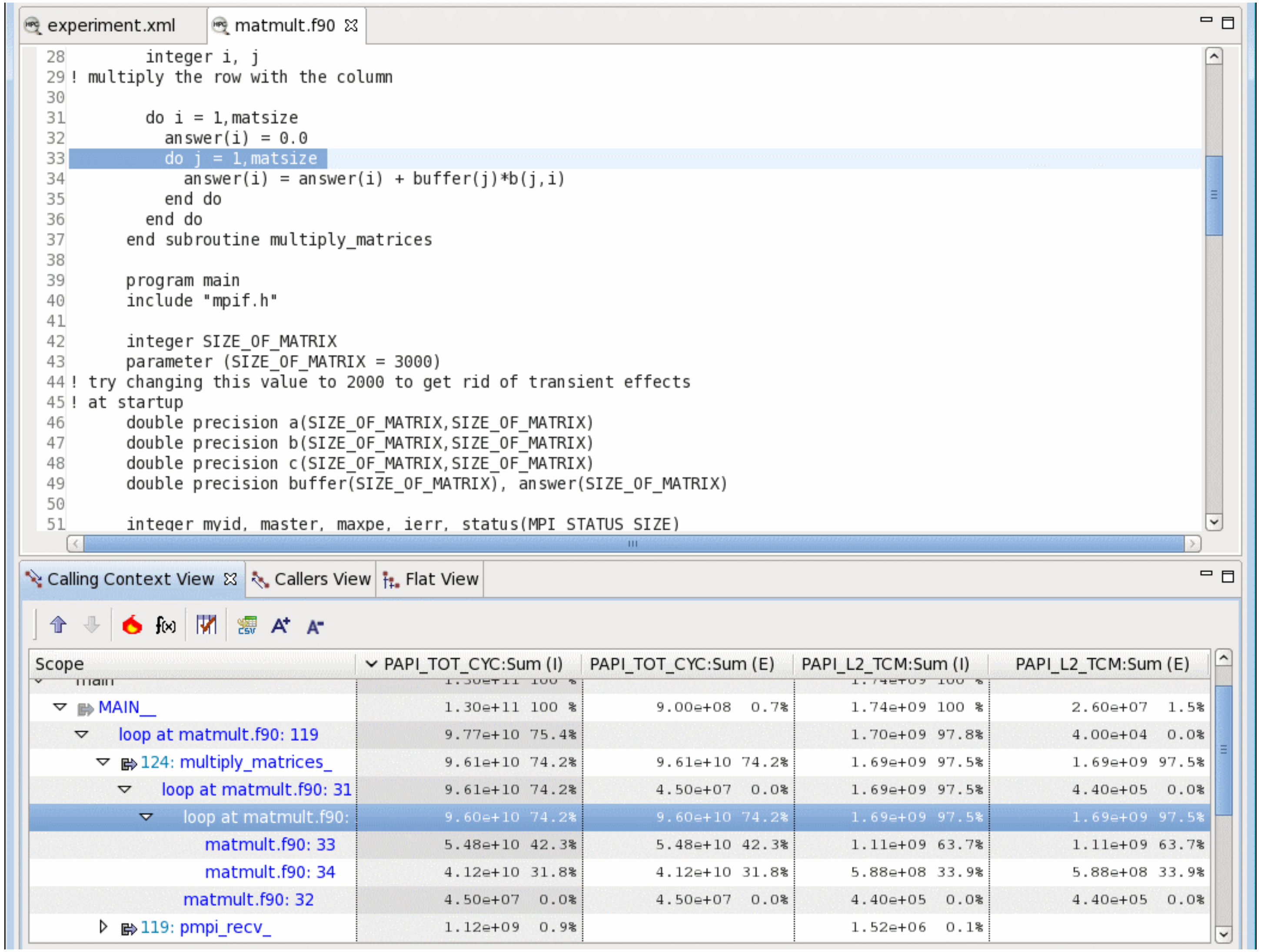
- `sudo perf top`



```
Samples: 47K of event 'cycles:ppp', Event count (approx.): 4927575036
Overhead  Shared Object                                        Symbol
   2,84%  [unknown]                                            [.] 0000000000000000
   1,25%  [kernel]                                             [k] syscall_return_via_sysret
   1,04%  libc-2.27.so                                         [.] __memmove_avx_unaligned_e
   0,86%  libc-2.27.so                                         [.] _int_malloc
   0,69%  ld-2.27.so                                           [.] do_lookup_x
   0,66%  [kernel]                                             [k] fw_domains_get
   0,65%  [kernel]                                             [k] _raw_spin_lock_irqsave
   0,64%  libc-2.27.so                                         [.] cfree@GLIBC_2.2.5
   0,62%  [kernel]                                             [k] __fget
   0,51%  libc-2.27.so                                         [.] malloc
   0,48%  perf                                                 [.] 0x00000000001ea403
   0,47%  [kernel]                                             [k] __schedule
   0,47%  [kernel]                                             [k] timerqueue_add
   0,47%  libpthread-2.27.so                                   [.] __pthread_mutex_lock
   0,47%  libsamplerate.so.0.1.8                               [.] 0x0000000000003790
   0,45%  [kernel]                                             [k] module_get_kallsym
   0,44%  perf                                                 [.] 0x00000000001f7d45
   0,44%  [kernel]                                             [k] update_blocked_averages
   0,43%  libpixman-1.so.0.34.0                                [.] pixman_rasterize_edges
   0,42%  [kernel]                                             [k] __d_lookup_rcu
   0,42%  [kernel]                                             [k] copy_user_enhanced_fast_s
   0,40%  perf                                                 [.] 0x00000000001f7860
   0,39%  [kernel]                                             [k] menu_select
Failed to open /tmp/perf-1509.map, continuing without symbols
```

HPCToolkit

- Open Source



```fortran
    integer i, j
! multiply the row with the column

    do i = 1,matsize
     answer(i) = 0.0
     do j = 1,matsize
      answer(i) = answer(i) + buffer(j)*b(j,i)
     end do
    end do
   end subroutine multiply_matrices

   program main
   include "mpif.h"

   integer SIZE_OF_MATRIX
   parameter (SIZE_OF_MATRIX = 3000)
! try changing this value to 2000 to get rid of transient effects
! at startup
   double precision a(SIZE_OF_MATRIX,SIZE_OF_MATRIX)
   double precision b(SIZE_OF_MATRIX,SIZE_OF_MATRIX)
   double precision c(SIZE_OF_MATRIX,SIZE_OF_MATRIX)
   double precision buffer(SIZE_OF_MATRIX), answer(SIZE_OF_MATRIX)

   integer myid, master, maxpe, ierr, status(MPI_STATUS_SIZE)
```

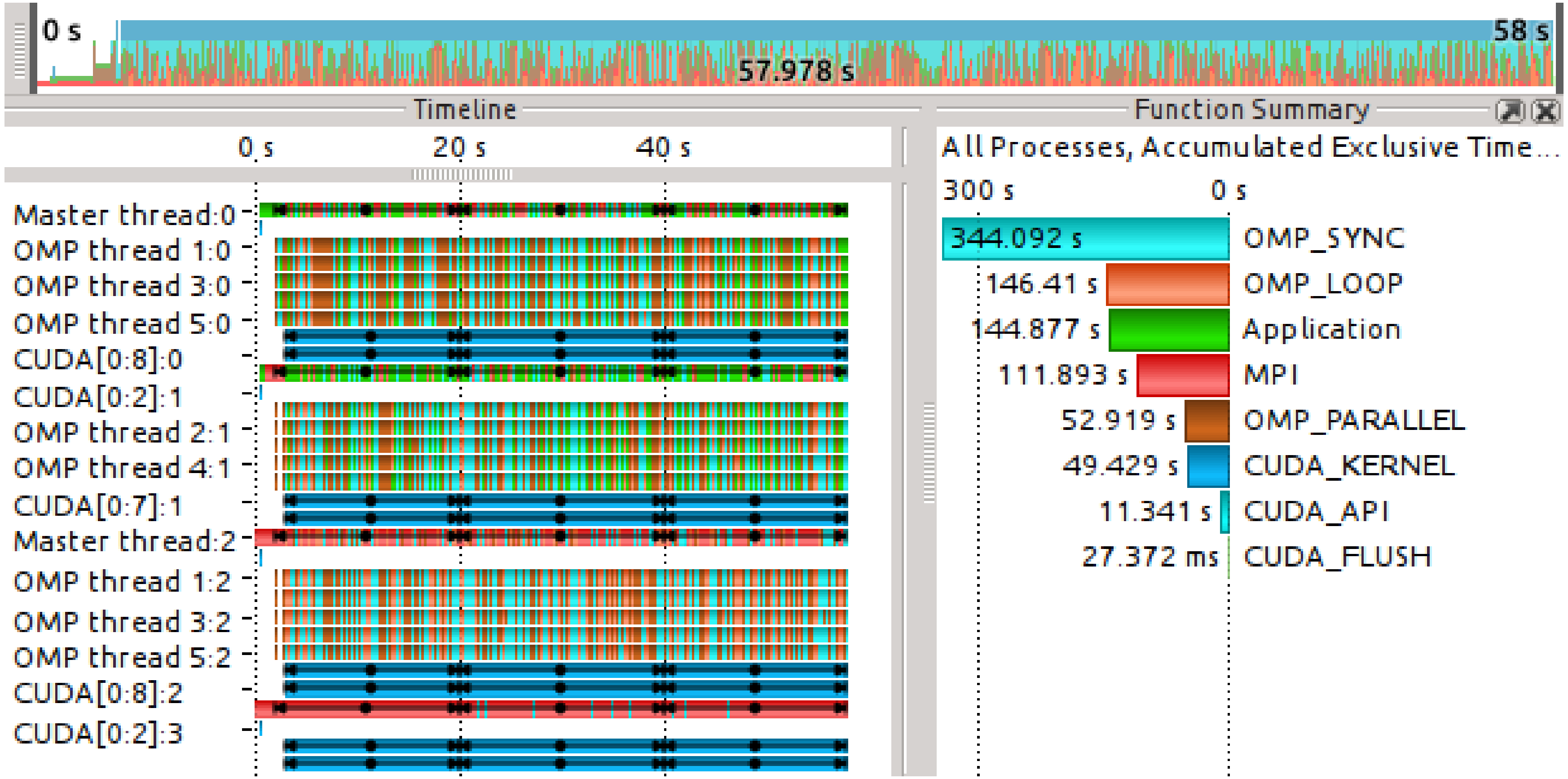
Scope	PAPI_TOT_CYC:Sum (I)	PAPI_TOT_CYC:Sum (E)	PAPI_L2_TCM:Sum (I)	PAPI_L2_TCM:Sum (E)
main	1.30e+11 100 %		1.74e+09 100 %	
MAIN__	1.30e+11 100 %	9.00e+08 0.7%	1.74e+09 100 %	2.60e+07 1.5%
loop at matmult.f90: 119	9.77e+10 75.4%		1.70e+09 97.8%	4.00e+04 0.0%
124: multiply_matrices_	9.61e+10 74.2%	9.61e+10 74.2%	1.69e+09 97.5%	1.69e+09 97.5%
loop at matmult.f90: 31	9.61e+10 74.2%	4.50e+07 0.0%	1.69e+09 97.5%	4.40e+05 0.0%
loop at matmult.f90:	9.60e+10 74.2%	9.60e+10 74.2%	1.69e+09 97.5%	1.69e+09 97.5%
matmult.f90: 33	5.48e+10 42.3%	5.48e+10 42.3%	1.11e+09 63.7%	1.11e+09 63.7%
matmult.f90: 34	4.12e+10 31.8%	4.12e+10 31.8%	5.88e+08 33.9%	5.88e+08 33.9%
matmult.f90: 32	4.50e+07 0.0%	4.50e+07 0.0%	4.40e+05 0.0%	4.40e+05 0.0%
119: pmpi_recv_	1.12e+09 0.9%		1.52e+06 0.1%	

Score-P & TAU

- Open Source
- Generally require modifying your link-step, or even the whole compilation

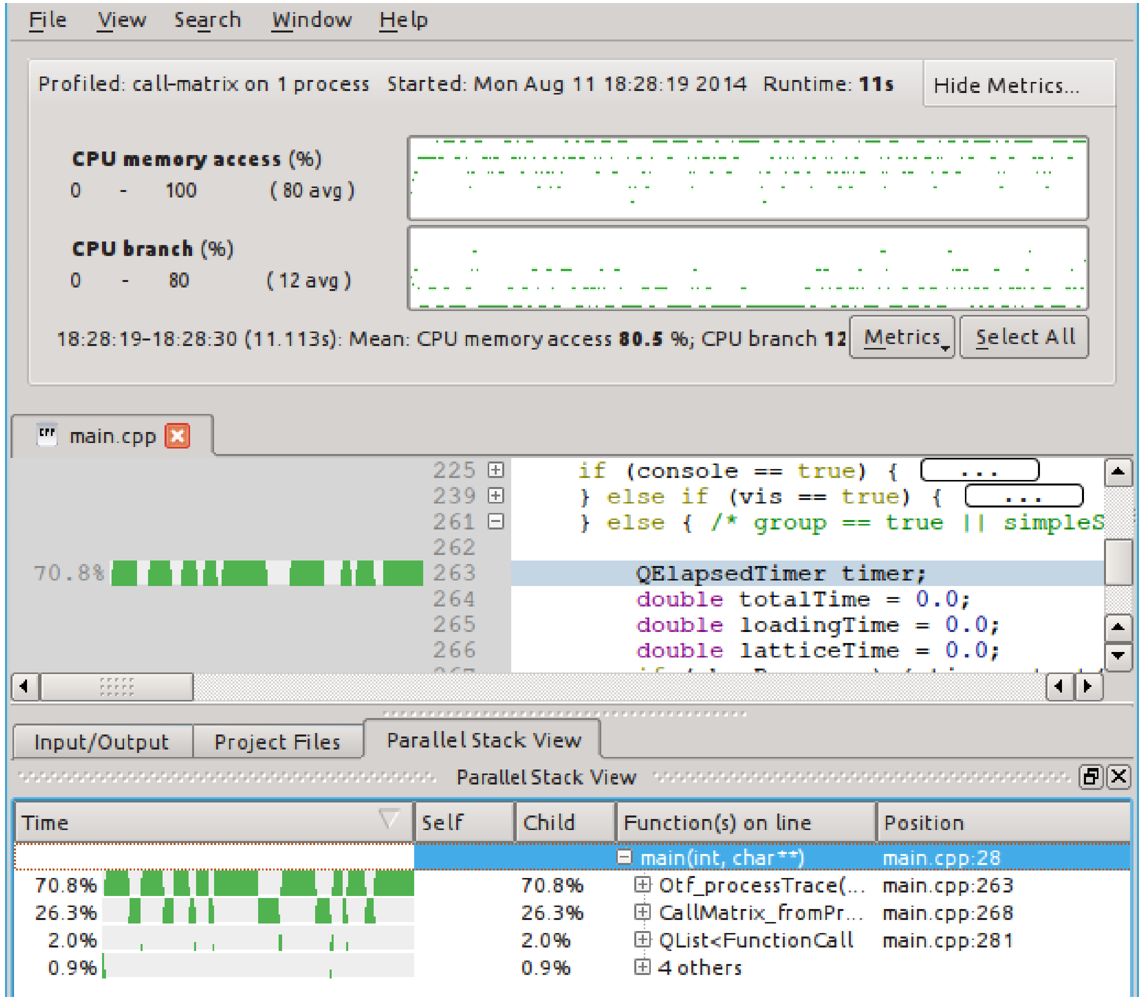
Vampir

- Commercial
- Trace file viewer
- Works with TAU and Score-P



Arm MAP

- Commercial



Profiled: call-matrix on 1 process Started: Mon Aug 11 18:28:19 2014 Runtime: **11s**

CPU memory access (%)
0 - 100 (80 avg)

CPU branch (%)
0 - 80 (12 avg)

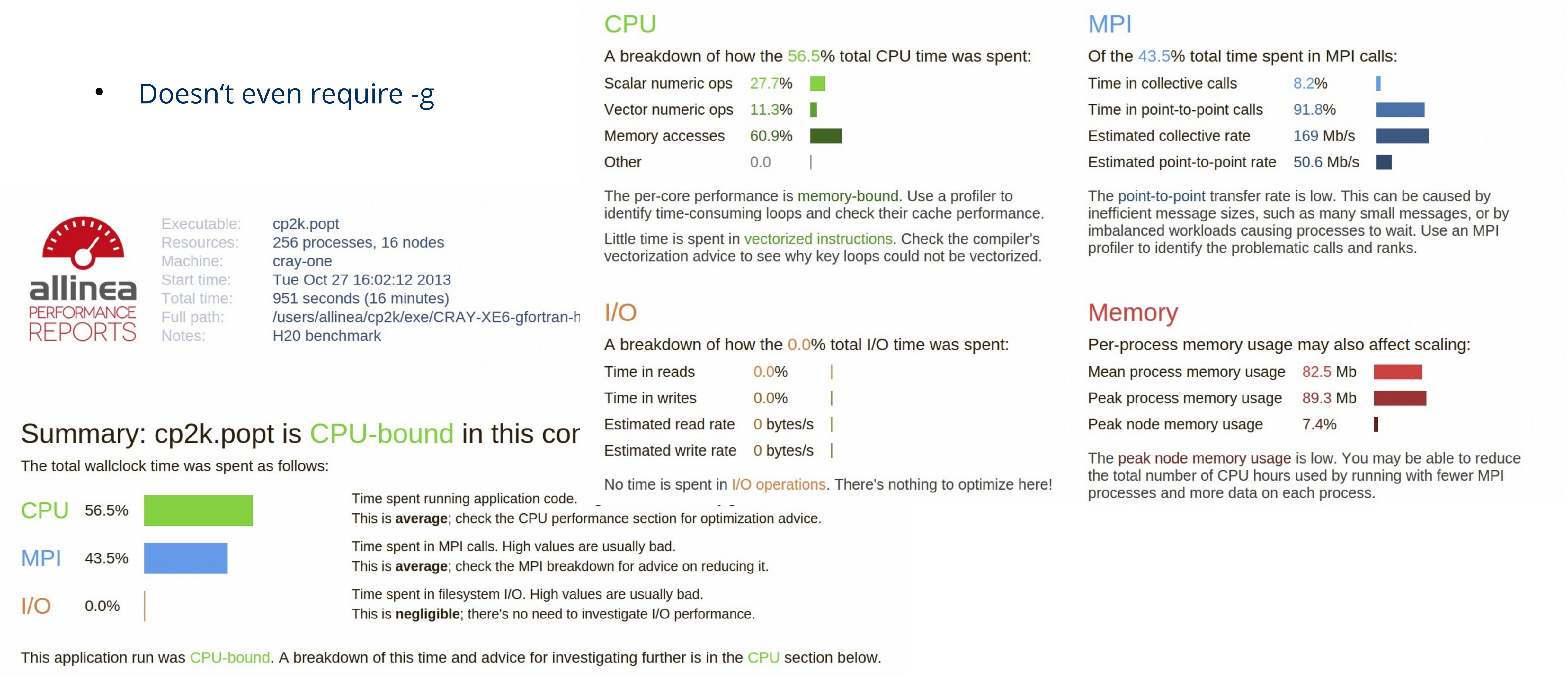
18:28:19-18:28:30 (11.113s): Mean: CPU memory access **80.5** %; CPU branch **12**

```
    if (console == true) { ... 
    } else if (vis == true) { ... 
    } else { /* group == true || simpleS

        QElapsedTimer timer;
        double totalTime = 0.0;
        double loadingTime = 0.0;
        double latticeTime = 0.0;
```

Time	Self	Child	Function(s) on line	Position
			⊟ main(int, char**)	main.cpp:28
70.8%		70.8%	⊞ Otf_processTrace(...	main.cpp:263
26.3%		26.3%	⊞ CallMatrix_fromPr...	main.cpp:268
2.0%		2.0%	⊞ QList<FunctionCall	main.cpp:281
0.9%		0.9%	⊞ 4 others	

Arm Performance Reports

- Doesn‘t even require -g



CPU

A breakdown of how the 56.5% total CPU time was spent:

Scalar numeric ops	27.7%
Vector numeric ops	11.3%
Memory accesses	60.9%
Other	0.0

The per-core performance is memory-bound. Use a profiler to identify time-consuming loops and check their cache performance.

Little time is spent in vectorized instructions. Check the compiler's vectorization advice to see why key loops could not be vectorized.

MPI

Of the 43.5% total time spent in MPI calls:

Time in collective calls	8.2%
Time in point-to-point calls	91.8%
Estimated collective rate	169 Mb/s
Estimated point-to-point rate	50.6 Mb/s

The point-to-point transfer rate is low. This can be caused by inefficient message sizes, such as many small messages, or by imbalanced workloads causing processes to wait. Use an MPI profiler to identify the problematic calls and ranks.

Executable:	cp2k.popt
Resources:	256 processes, 16 nodes
Machine:	cray-one
Start time:	Tue Oct 27 16:02:12 2013
Total time:	951 seconds (16 minutes)
Full path:	/users/allinea/cp2k/exe/CRAY-XE6-gfortran-h
Notes:	H20 benchmark

I/O

A breakdown of how the 0.0% total I/O time was spent:

Time in reads	0.0%
Time in writes	0.0%
Estimated read rate	0 bytes/s
Estimated write rate	0 bytes/s

No time is spent in I/O operations. There's nothing to optimize here!

Memory

Per-process memory usage may also affect scaling:

Mean process memory usage	82.5 Mb
Peak process memory usage	89.3 Mb
Peak node memory usage	7.4%

The peak node memory usage is low. You may be able to reduce the total number of CPU hours used by running with fewer MPI processes and more data on each process.

Summary: cp2k.popt is CPU-bound in this cor

The total wallclock time was spent as follows:

CPU	56.5%	Time spent running application code. This is **average**; check the CPU performance section for optimization advice.
MPI	43.5%	Time spent in MPI calls. High values are usually bad. This is **average**; check the MPI breakdown for advice on reducing it.
I/O	0.0%	Time spent in filesystem I/O. High values are usually bad. This is **negligible**; there's no need to investigate I/O performance.

This application run was CPU-bound. A breakdown of this time and advice for investigating further is in the CPU section below.

Conclusions 1

- Vendor tools like VTune and NVProf (and CrayPAT?) are good. Use them.
  - Single node / low scalability
- Arm MAP is good. Use it.
- If you don‘t have Arm MAP, use HPCToolkit
- Perf is great if you have sudo and a new-ish kernel
  - No MPI

- If the above don’t offer enough detail
  - Score-P and Vampir
  - (TAU and Vampir)
  - (They do not provide more GPU details than NVProf)

Conclusions 2

- Your own computer
  - CUDA
    - NVProf
  - No CUDA
    - MPI
      - HPCToolkit
    - No MPI
      - Perf
      - (HPCToolkit)

- If you somehow have a license, get Arm MAP

Conclusions 2

- Someone else’s computer
  - Arm MAP
  - Single Node
    - GPU → NVProf
    - Intel CPU → VTune
    - No Intel CPU → HPCToolkit

Conclusions 2

- Someone else's computer
  - Multiple Nodes
    - Arm MAP
    - (HPCToolkit)

- You need more detail. I.e. a trace, or e.g. more in-depth info on MPI and OpenMP usage possibly combined with CUDA or OpenACC
  - Score-P → Vampir
  - (TAU → Vampir)