cpu performance tuning for XE nodes

Added by Galen Arnold, last edited by Galen Arnold on Jun 15, 2016

Overview of tools

https://bluewaters.ncsa.illinois.edu/profiling

Core Placement, NUMA, aprun

The aprun command is used to specify to ALPS the resources and placement parameters needed for your application at application launch. At a high level, aprun is similar to mpiexec or mpirun.

https://bluewaters.ncsa.illinois.edu/using-aprun

CrayPat , perftools

CrayPat is an optional performance analysis tool used to evaluate program behavior on Cray supercomputer systems.

https://bluewaters.ncsa.illinois.edu/cpmat

Stream benchmark examples

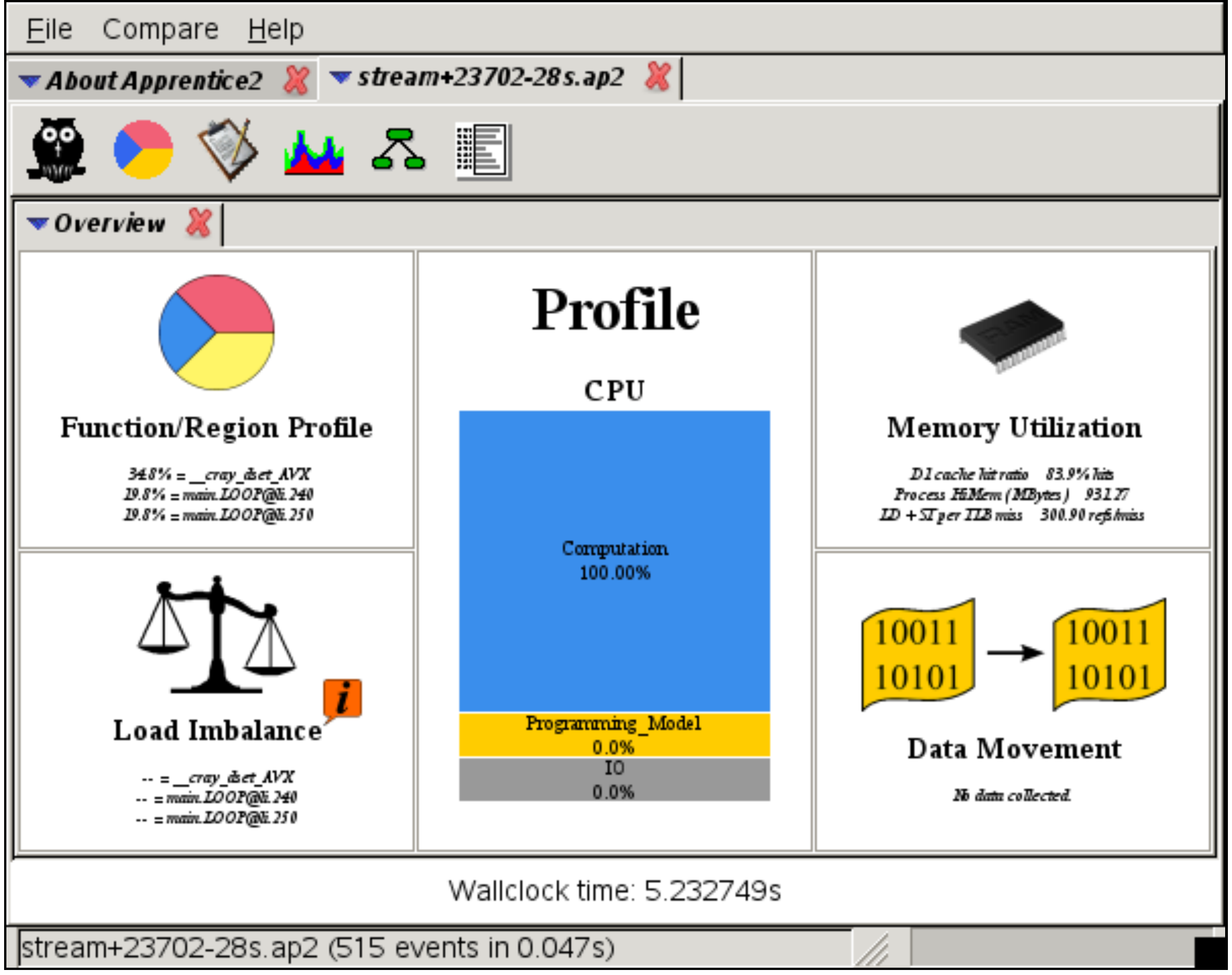
We'll do a simple survey of the various perftools modules and note some of the different views and information available from the variety of modules provided by Cray.

module unload darshan # unload the darshan io performance tool , in the general case use only 1 performance tool at a time

module load perftools-base; module load perftools-lite

// rebuild application, run rebuilt application, find *.ap2 and *.xf files. Load and analyze the .ap2 file with:

app2 stream+23702-28s.ap2 & // the *.ap2 file will be new and unique for each successful run of the application



34.8% = __cray_dset_AVX
19.8% = main.LOOP@li.240
19.8% = main.LOOP@li.250

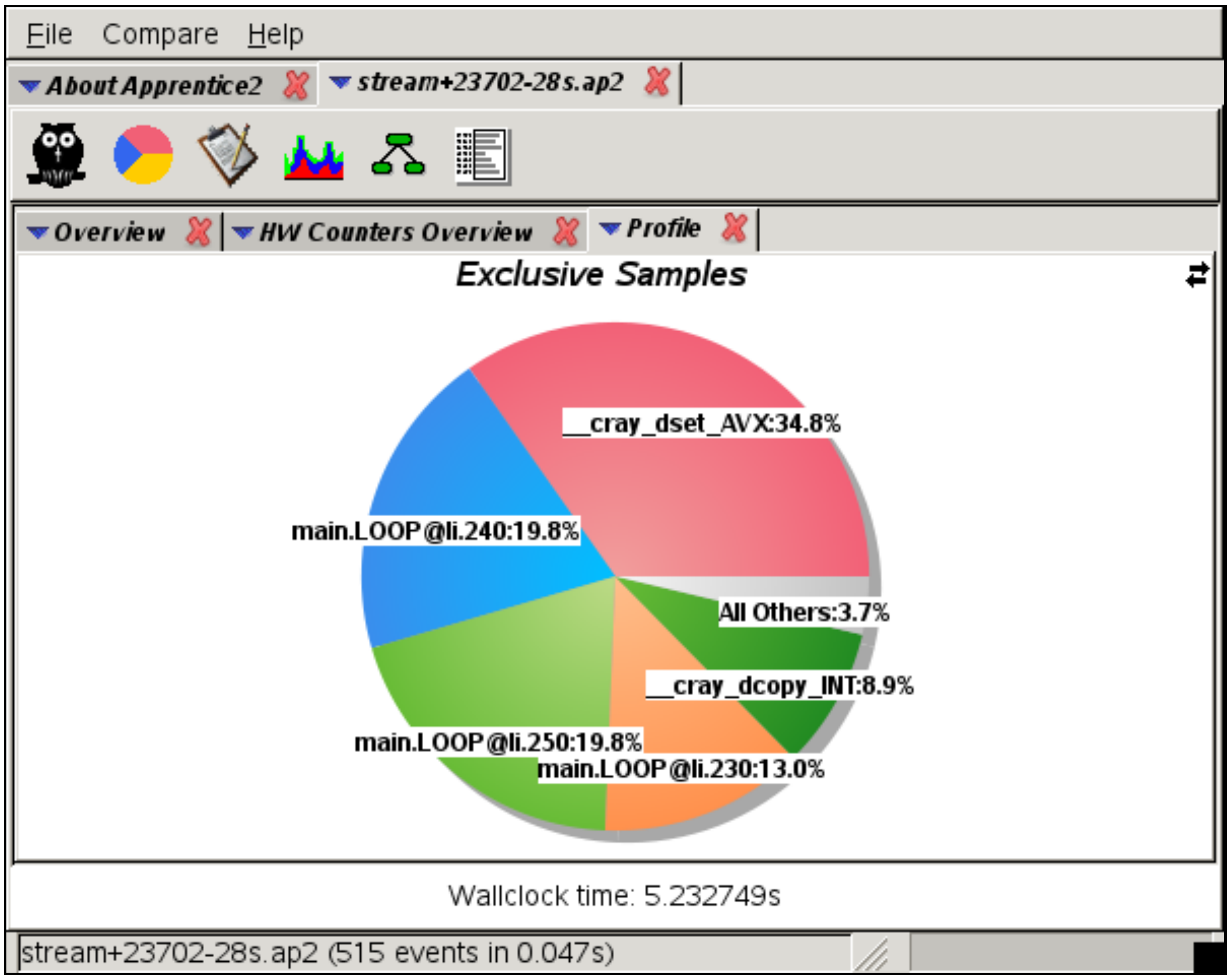
-- = __cray_dset_AVX
-- = main.LOOP@li.240
-- = main.LOOP@li.250

D1 cache hit ratio 83.9% hits
Process HiMem (MBytes) 931.27
LD + ST per TLB miss 300.90 refs/miss

No data collected.

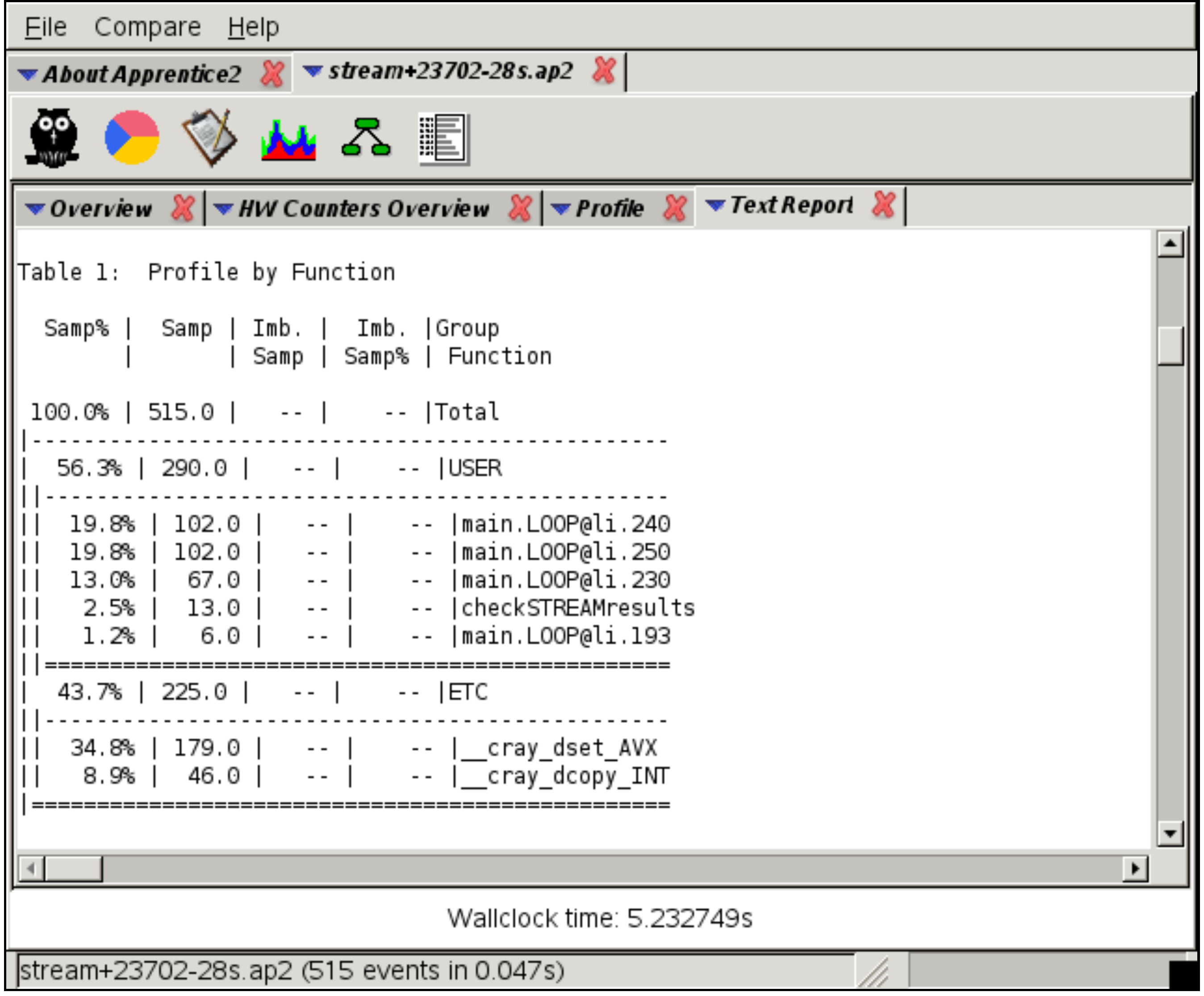
Wallclock time: 5.232749s

stream+23702-28s.ap2 (515 events in 0.047s)



Wallclock time: 5.232749s

stream+23702-28s.ap2 (515 events in 0.047s)



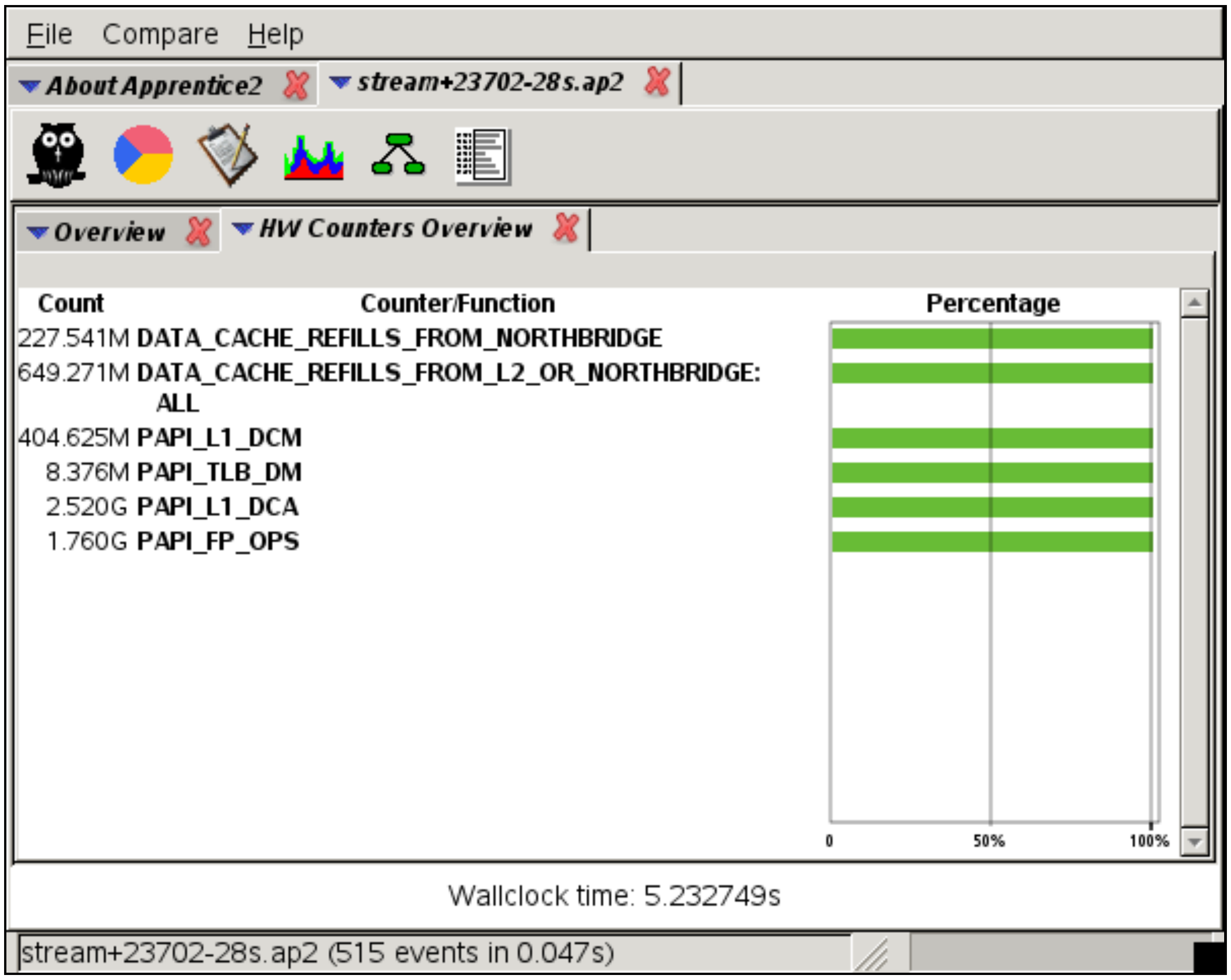
```
Table 1:  Profile by Function

  Samp% |  Samp | Imb.  |  Imb.  |Group
        |       | Samp  | Samp%  | Function

 100.0% | 515.0 |    -- |     -- |Total
|-------------------------------------------------
|  56.3% | 290.0 |    -- |     -- |USER
||------------------------------------------------
||  19.8% | 102.0 |    -- |     -- |main.LOOP@li.240
||  19.8% | 102.0 |    -- |     -- |main.LOOP@li.250
||  13.0% |  67.0 |    -- |     -- |main.LOOP@li.230
||   2.5% |  13.0 |    -- |     -- |checkSTREAMresults
||   1.2% |   6.0 |    -- |     -- |main.LOOP@li.193
||================================================
|  43.7% | 225.0 |    -- |     -- |ETC
||------------------------------------------------
||  34.8% | 179.0 |    -- |     -- |__cray_dset_AVX
||   8.9% |  46.0 |    -- |     -- |__cray_dcopy_INT
|================================================
```

Wallclock time: 5.232749s

stream+23702-28s.ap2 (515 events in 0.047s)



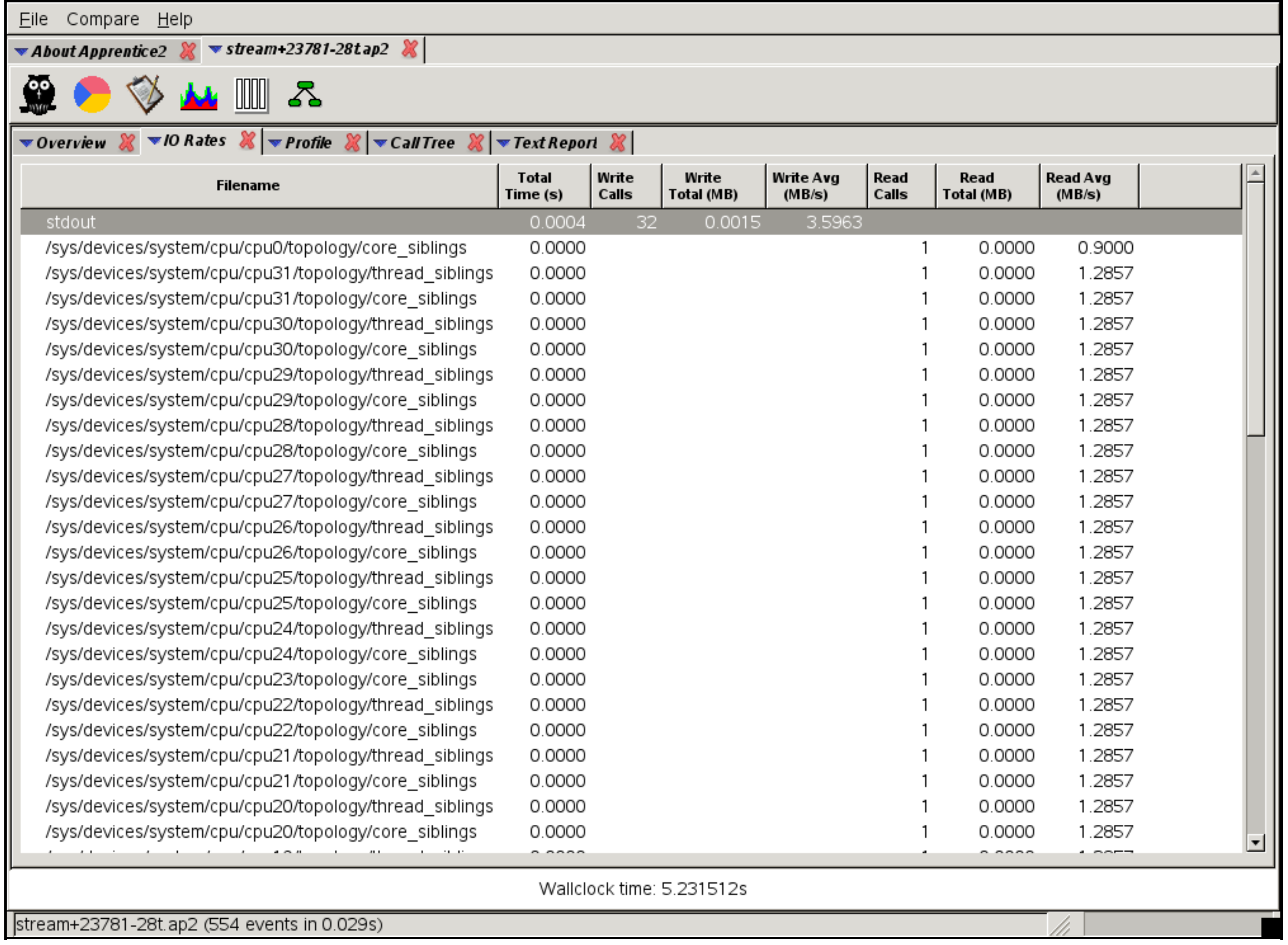
Count	Counter/Function	Percentage
227.541M	DATA_CACHE_REFILLS_FROM_NORTHBRIDGE	🟩
649.271M	DATA_CACHE_REFILLS_FROM_L2_OR_NORTHBRIDGE: ALL	🟩
404.625M	PAPI_L1_DCM	🟩
8.376M	PAPI_TLB_DM	🟩
2.520G	PAPI_L1_DCA	🟩
1.760G	PAPI_FP_OPS	🟩

Wallclock time: 5.232749s

stream+23702-28s.ap2 (515 events in 0.047s)

module unload perftools-lite; module load perftools-lite-events // rebuild and rerun

Notice that io information is included with perftools-lite-events.



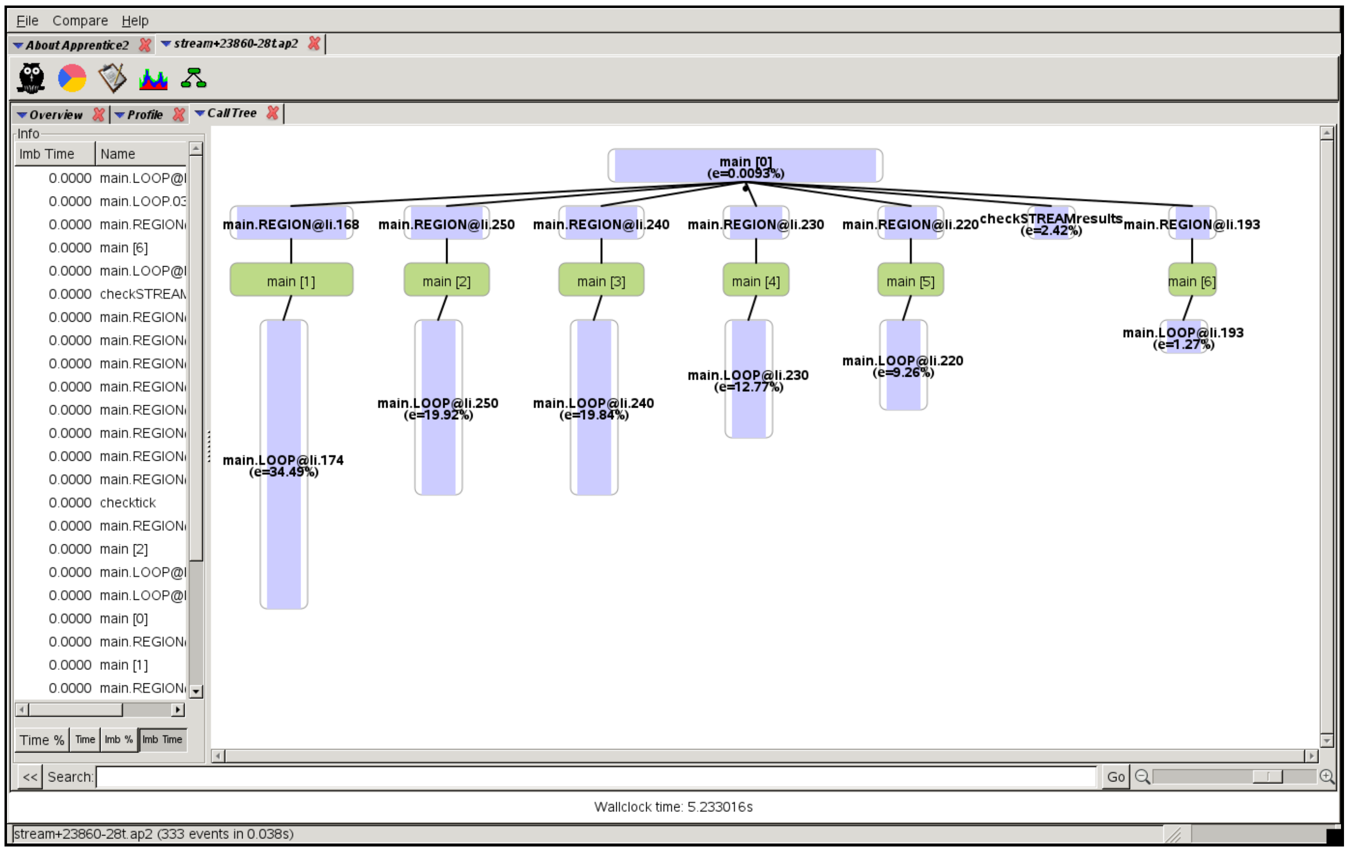
Filename	Total Time (s)	Write Calls	Write Total (MB)	Write Avg (MB/s)	Read Calls	Read Total (MB)	Read Avg (MB/s)
stdout	0.0004	32	0.0015	3.5963			
/sys/devices/system/cpu/cpu0/topology/core_siblings	0.0000				1	0.0000	0.9000
/sys/devices/system/cpu/cpu31/topology/thread_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu31/topology/core_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu30/topology/thread_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu30/topology/core_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu29/topology/thread_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu29/topology/core_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu28/topology/thread_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu28/topology/core_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu27/topology/thread_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu27/topology/core_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu26/topology/thread_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu26/topology/core_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu25/topology/thread_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu25/topology/core_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu24/topology/thread_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu24/topology/core_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu23/topology/core_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu22/topology/thread_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu22/topology/core_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu21/topology/thread_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu21/topology/core_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu20/topology/thread_siblings	0.0000				1	0.0000	1.2857
/sys/devices/system/cpu/cpu20/topology/core_siblings	0.0000				1	0.0000	1.2857

Wallclock time: 5.231512s

stream+23781-28t.ap2 (554 events in 0.029s)

module unload perftools-lite-events; module load perftools-lite-loops // rebuild and rerun

As the name implies, perftools-lite-loops analyzes your application from a loop-centric viewpoint. Loop nests are not shown in the display but you as the developer should review this alongside the code and know which loops nest and how.



Imb Time	Name
0.0000	main.LOOP@l
0.0000	main.LOOP.03
0.0000	main.REGION
0.0000	main [6]
0.0000	main.LOOP@l
0.0000	checkSTREAM
0.0000	main.REGION
0.0000	main.REGION
0.0000	main.REGION
0.0000	main.REGION
0.0000	main.REGION
0.0000	main.REGION
0.0000	main.REGION
0.0000	main.REGION
0.0000	checktick
0.0000	main.REGION
0.0000	main [2]
0.0000	main.LOOP@l
0.0000	main.LOOP@l
0.0000	main [0]
0.0000	main.REGION
0.0000	main [1]
0.0000	main.REGION

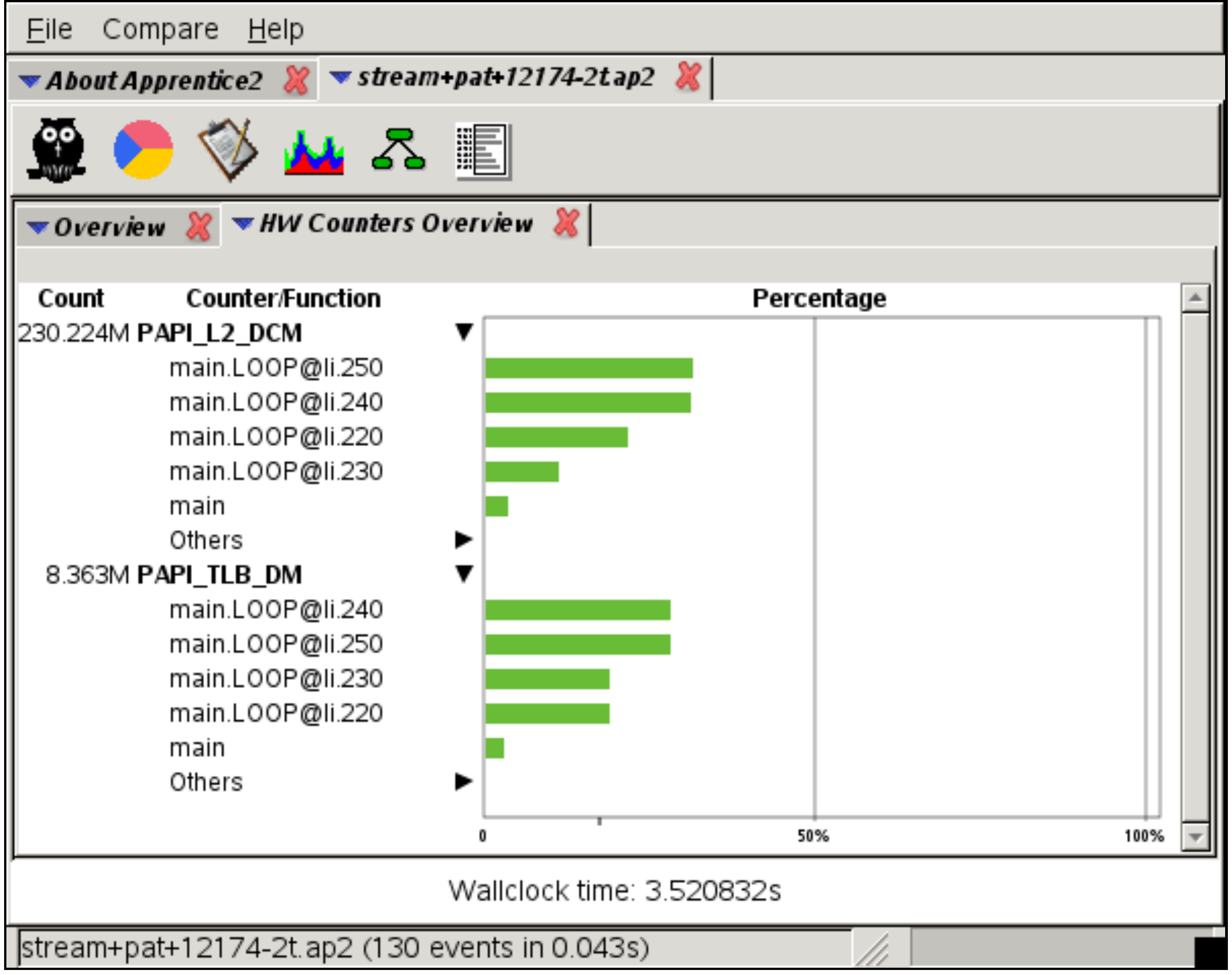
Wallclock time: 5.233016s

stream+23860-28t.ap2 (333 events in 0.038s)

module unload perftools-lite-loops; module load perftools // rebuild and rerun with:

PAT_RT_PERFCTR=PAPI_TLC_DM,PAPI_L2_DCM

*Perftools has **many** options. See the manual page for the various modules shown. The performance counters may be of particular interest if you want to understand how your application is mapping to specific portions of the hardware (data or instruction cache, main memory, fpu, ...).*



Count	Counter/Function	Percentage
230.224M	**PAPI_L2_DCM**	
	main.LOOP@li.250	
	main.LOOP@li.240	
	main.LOOP@li.220	
	main.LOOP@li.230	
	main	
	Others	
8.363M	**PAPI_TLB_DM**	
	main.LOOP@li.240	
	main.LOOP@li.250	
	main.LOOP@li.230	
	main.LOOP@li.220	
	main	
	Others	

Wallclock time: 3.520832s

stream+pat+12174-2t.ap2 (130 events in 0.043s)

Notes:

- for threaded code (OMP_NUM_THREADS > 1) you will typically get 1 core or thread's worth of counters in the *.ap2 report
- see the output from papi_avail for the full list of hw event counters and counter groups