Performance Analysis

Dr. Michael K. Bane

HIGH END COMPUTE

GROUP EXERCISE

- What is "performance analysis"?
- Why does it matter?
- How would you determine "good" or "bad"?

Where Did the Time Go?

1. How long my code takes to run?
2. Which parts of my code take the longest?
3. Where is the parallelism in my code?

4. When timing my code
   - Which version to use?
   - Which input data set to use?
   - Have I removed all "unnecessary" artefacts?

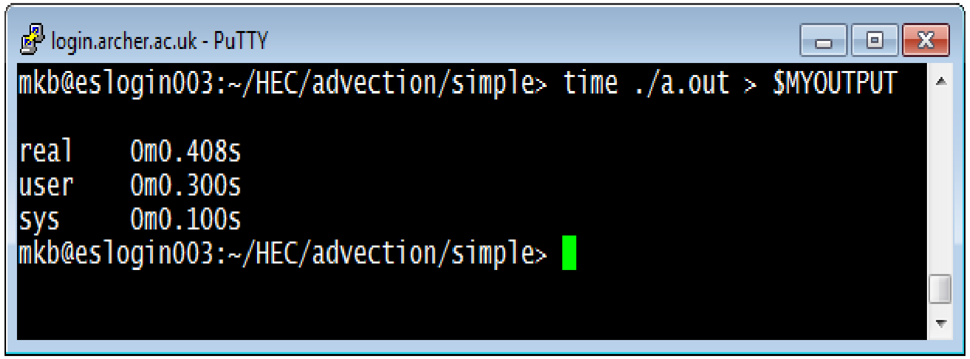
How long does my code take?

- "time" shell built-in

- Timers from FORTRAN
- Timers from OpenMP or MPI

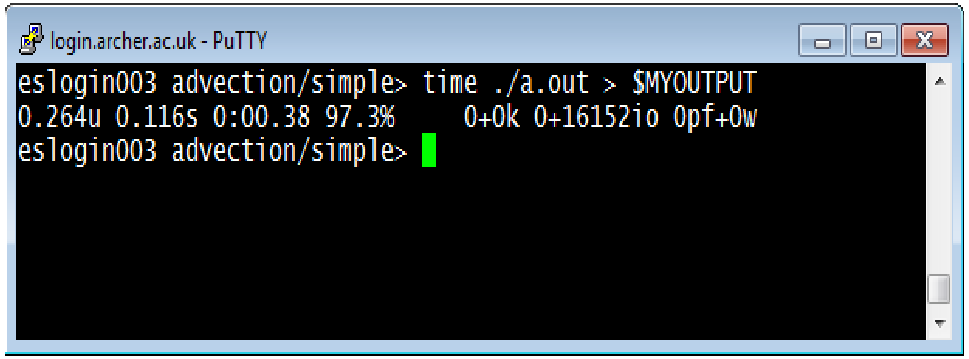
- Profilers

Shell built-in "time" command



```
mkb@eslogin003:~/HEC/advection/simple> time ./a.out > $MYOUTPUT

real	0m0.408s
user	0m0.300s
sys	0m0.100s
mkb@eslogin003:~/HEC/advection/simple>
```

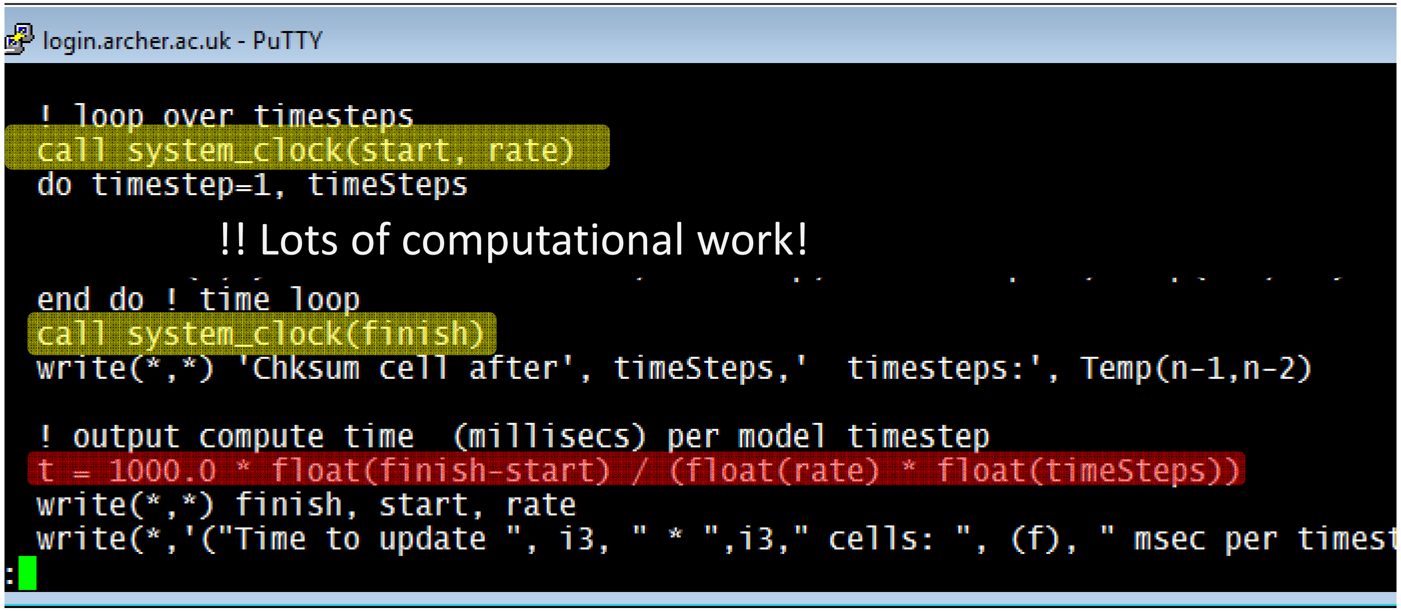
bash



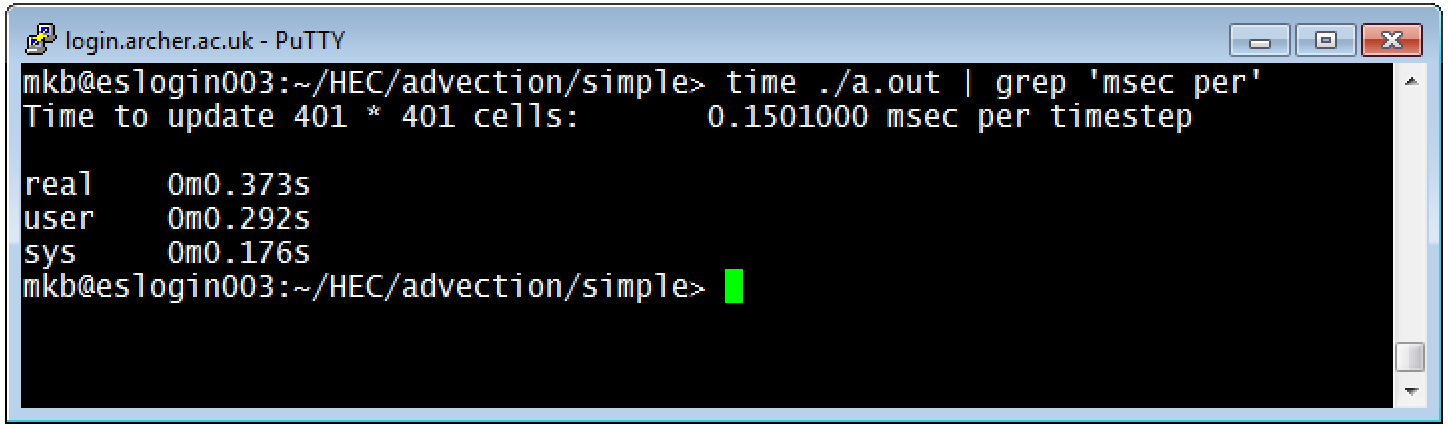
```
eslogin003 advection/simple> time ./a.out > $MYOUTPUT
0.264u 0.116s 0:00.38 97.3%	0+0k 0+16152io 0pf+0w
eslogin003 advection/simple>
```

csh

FORTRAN timer



```
 ! loop over timesteps
 call system_clock(start, rate)
 do timestep=1, timeSteps

           !! Lots of computational work!

 end do ! time loop
 call system_clock(finish)
 write(*,*) 'Chksum cell after', timeSteps,'  timesteps:', Temp(n-1,n-2)

 ! output compute time  (millisecs) per model timestep
 t = 1000.0 * float(finish-start) / (float(rate) * float(timeSteps))
 write(*,*) finish, start, rate
 write(*,'("Time to update ", i3, " * ",i3," cells: ", (f), " msec per timest
:
```



```
mkb@eslogin003:~/HEC/advection/simple> time ./a.out | grep 'msec per'
Time to update 401 * 401 cells:       0.1501000 msec per timestep

real    0m0.373s
user    0m0.292s
sys     0m0.176s
mkb@eslogin003:~/HEC/advection/simple>
```

1000 timesteps -> 0.15 secs in all loops out of wall clock of 0.373 secs. **Amdahl**

PROFILING

- Profiling is not…
  - … debugging
  - … tracing

Profilers

- GNU
    - gprof
    - gcov
- TAU
- Intel Parallel Studio:
    - VTune Amplifier (CLI & GUI)
    - Vector advisor thingy
- Archer
    - CrayPat / Apprentice

Profilers

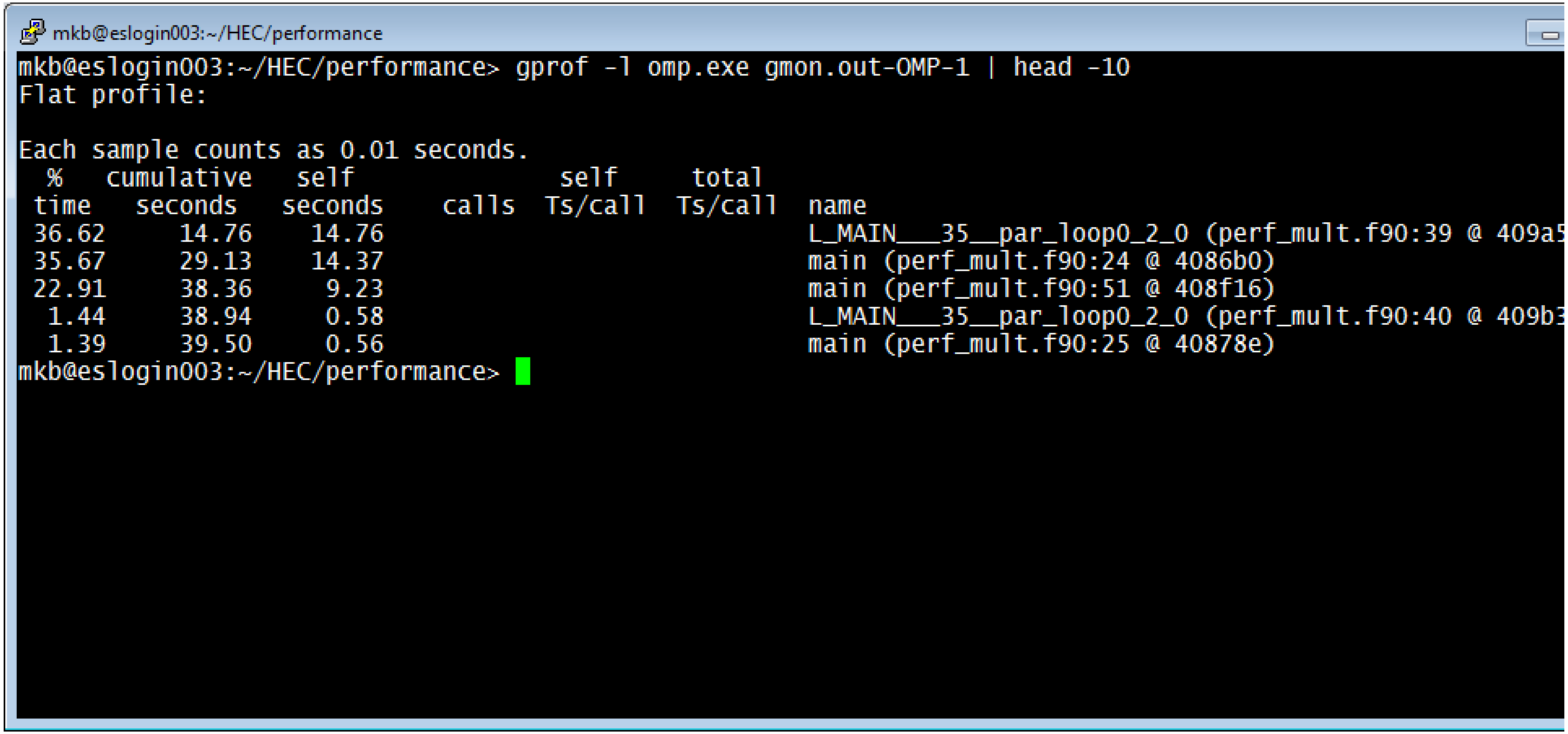
	Serial	OpenMP	MPI	GPU	Status
Gnu gprof	✓				Free, but legacy & limited support
TAU	✓	✓	✓	✓	Free to download but expensive for committed support.
VTune	✓	✓	IPS Cluster provides ITAC which could help		Costed as part of IPS Professional
CrayPat	✓	✓	✓	✓ TBC	
Allinea MAP	✓	✓	✓	✓	

Procedure

- Profiling is a sampling technique
- So need to know where program is every 10 msecs (say)
  - Instrument code during compilation
  - "intercept" calls during run time
- Instrumentation most common

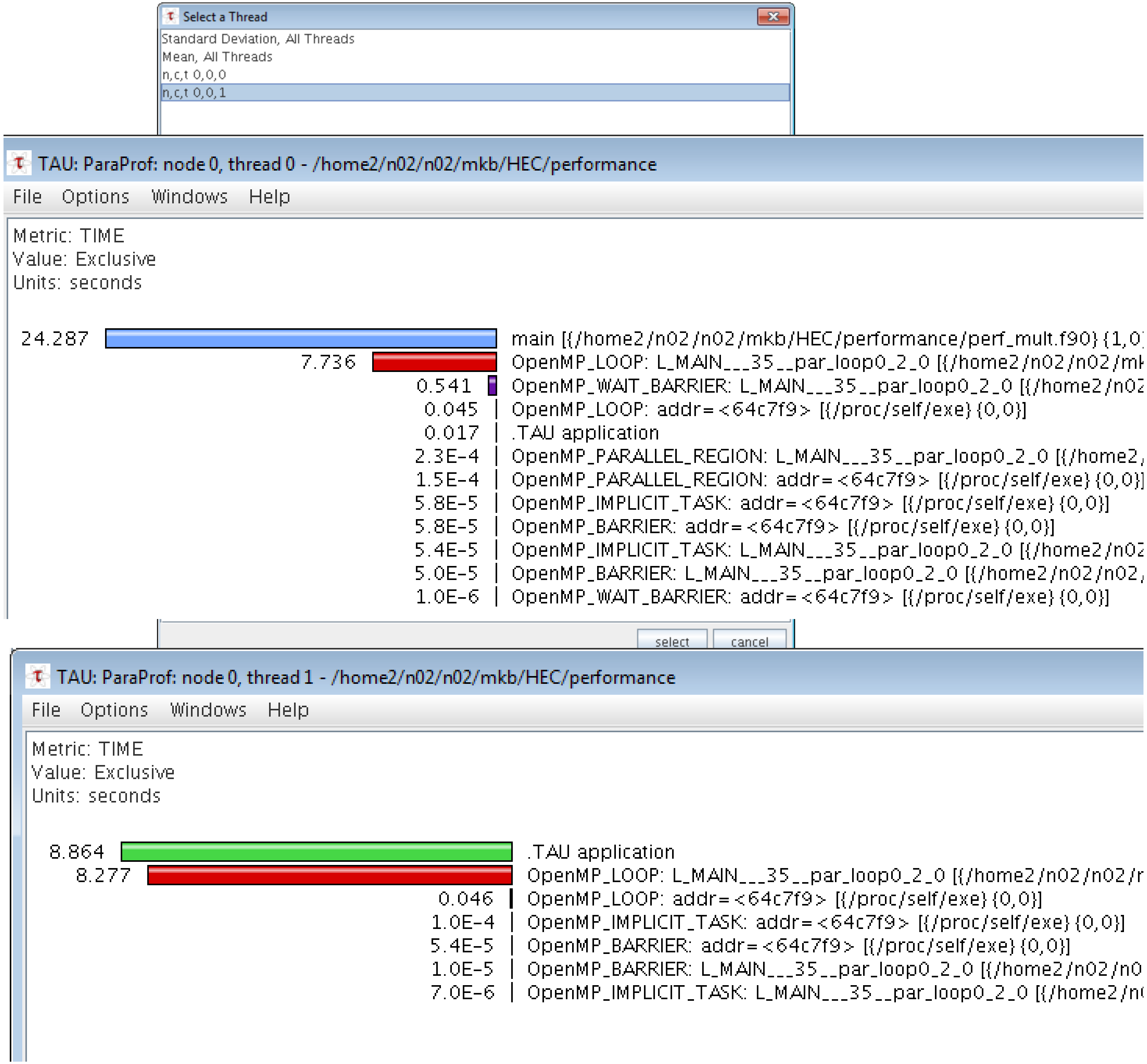
Quick Examples

- Gprof (CLI)
- TAU (CLI / GUI)
- Allinea (CLI / GUI)
  - MAP & Performance Reports
- As well as time profiling, Allinea MAP & TAU support energy profiling (via IPMP &/or RAPL)



mkb@eslogin003:~/HEC/performance

```
mkb@eslogin003:~/HEC/performance> gprof -l omp.exe gmon.out-OMP-1 | head -10
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  Ts/call  Ts/call  name
 36.62     14.76    14.76                             L_MAIN___35__par_loop0_2_0 (perf_mult.f90:39 @ 409a5
 35.67     29.13    14.37                             main (perf_mult.f90:24 @ 4086b0)
 22.91     38.36     9.23                             main (perf_mult.f90:51 @ 408f16)
  1.44     38.94     0.58                             L_MAIN___35__par_loop0_2_0 (perf_mult.f90:40 @ 409b3
  1.39     39.50     0.56                             main (perf_mult.f90:25 @ 40878e)
mkb@eslogin003:~/HEC/performance> 
```



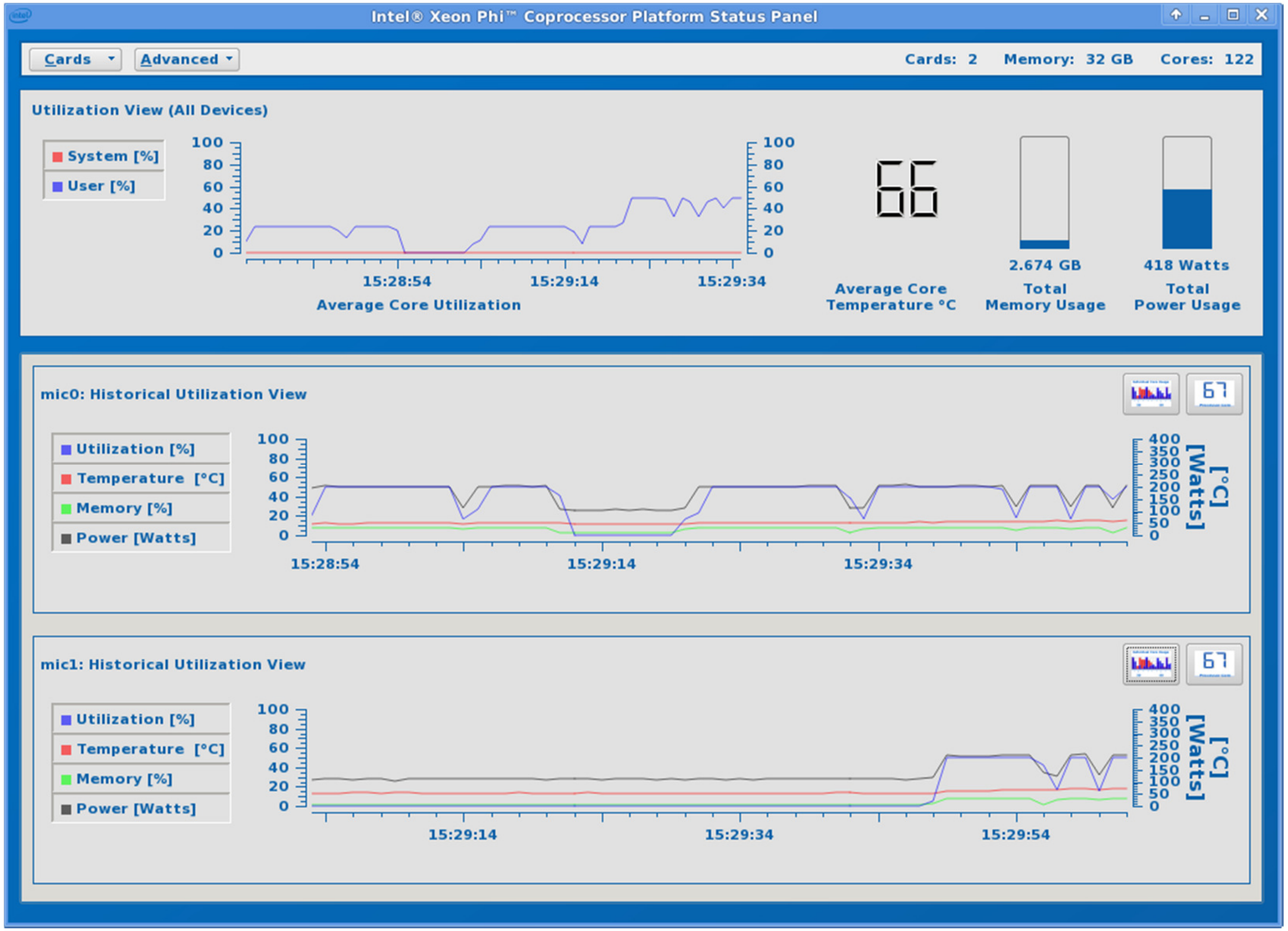
```
Standard Deviation, All Threads
Mean, All Threads
n,c,t 0,0,0
n,c,t 0,0,1
```

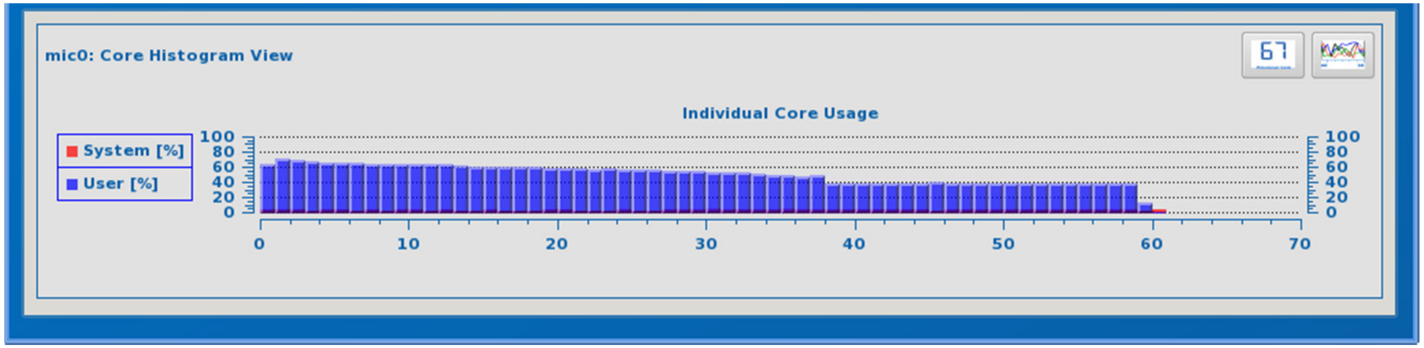
TAU: ParaProf: node 0, thread 0 - /home2/n02/n02/mkb/HEC/performance
File
Options
Windows
Help

```
Metric: TIME
Value: Exclusive
Units: seconds

24.287  main [{/home2/n02/n02/mkb/HEC/performance/perf_mult.f90} {1,0
 7.736  OpenMP_LOOP: L_MAIN___35__par_loop0_2_0 [{/home2/n02/n02/mk
 0.541  OpenMP_WAIT_BARRIER: L_MAIN___35__par_loop0_2_0 [{/home2/n02
 0.045  OpenMP_LOOP: addr=<64c7f9> [{/proc/self/exe} {0,0}]
 0.017  .TAU application
2.3E-4  OpenMP_PARALLEL_REGION: L_MAIN___35__par_loop0_2_0 [{/home2/
1.5E-4  OpenMP_PARALLEL_REGION: addr=<64c7f9> [{/proc/self/exe} {0,0}]
5.8E-5  OpenMP_IMPLICIT_TASK: addr=<64c7f9> [{/proc/self/exe} {0,0}]
5.8E-5  OpenMP_BARRIER: addr=<64c7f9> [{/proc/self/exe} {0,0}]
5.4E-5  OpenMP_IMPLICIT_TASK: L_MAIN___35__par_loop0_2_0 [{/home2/n02
5.0E-5  OpenMP_BARRIER: L_MAIN___35__par_loop0_2_0 [{/home2/n02/n02/
1.0E-6  OpenMP_WAIT_BARRIER: addr=<64c7f9> [{/proc/self/exe} {0,0}]
```

TAU: ParaProf: node 0, thread 1 - /home2/n02/n02/mkb/HEC/performance
File
Options
Windows
Help

```
Metric: TIME
Value: Exclusive
Units: seconds

 8.864  .TAU application
 8.277  OpenMP_LOOP: L_MAIN___35__par_loop0_2_0 [{/home2/n02/n02/r
 0.046  OpenMP_LOOP: addr=<64c7f9> [{/proc/self/exe} {0,0}]
1.0E-4  OpenMP_IMPLICIT_TASK: addr=<64c7f9> [{/proc/self/exe} {0,0}]
5.4E-5  OpenMP_BARRIER: addr=<64c7f9> [{/proc/self/exe} {0,0}]
1.0E-5  OpenMP_BARRIER: L_MAIN___35__par_loop0_2_0 [{/home2/n02/n0
7.0E-6  OpenMP_IMPLICIT_TASK: L_MAIN___35__par_loop0_2_0 [{/home2/n
```


Intel® Xeon Phi™ Coprocessor Platform Status Panel
Cards
Advanced
Cards: 2
Memory: 32 GB
Cores: 122
Utilization View (All Devices)
System [%]
User [%]
100
80
60
40
20
0
15:28:54
15:29:14
15:29:34
Average Core Utilization
66
Average Core Temperature °C
2.674 GB
Total Memory Usage
418 Watts
Total Power Usage
mic0: Historical Utilization View
Utilization [%]
Temperature [°C]
Memory [%]
Power [Watts]
100
80
60
40
20
0
400
350
300
250
200
150
100
50
0
[Watts]
[°C]
15:28:54
15:29:14
15:29:34
mic1: Historical Utilization View
Utilization [%]
Temperature [°C]
Memory [%]
Power [Watts]
15:29:14
15:29:34
15:29:54


mic0: Core Histogram View
Individual Core Usage
System [%]
User [%]
100
80
60
40
20
0
0
10
20
30
40
50
60
70

Hardware counters

- Most (but not ARM IP) chip manufacturers now include PMC/HW counters
  - L1, L2 Cache misses
  - Use of various unit (fl.pt etc)
  - Instantaneous power
- May need 'root' access
- Various tools/libraries may access
  - PAPI – common interface (eg called by TAU)
  - CrayPat, Allinea MAP, …

The Art of Profiling

- How to know when to parallelise
- How to know when *not* to bother