

Profiling and Debugging in cluster environments

Presented by Lukas Gamper and Alex Kosenkov

Agenda

- Profiling tools
- Application characterisation

Profiling tools

- VTune (SEP)
- Intel® Trace Analyzer and Collector
- IPM
- GNU gprof

Profiling tools: Intel® VTune™ Amplifier XE

- Collect profile:

```
$ amplxe-cl -collect hotspots ./a.out
```

- Generate report:

```
$ amplxe-cl -report hotspots -r r000hs
```

```
$ amplxe-cl -report hw-events -rr000cuo
```

Knobs:

```
-cpu-mask=2-8,10,12-14  
-target-pid=<unsigned integer>
```

Analysis shortcuts:

```
nehalem/snb-cycles-uops  
nehalem/snb-memory-access
```

Manual analysis (hw event sampling):

```
-collect-with runsa -knob event-config=<events list>
```

Events available:

```
CPU_CLK_UNHALTED.CORE,  
INST_RETIRED.ANY  
MEM_LOAD_RETIRED.L2_MISS  
BUS_TRANS_ANY.ALL_AGENTS  
RS_UOPS_DISPATCHED.CYCLES_NONE
```

Profiling tools: Intel® VTune™ Amplifier XE

Shortcut: nehalem-cycles-uops

- 1 CPU_CLK_UNHALTED.THREAD
- 2 INST_RETIRED.ANY
- 3 CPU_CLK_UNHALTED.REF
- 4 BR_INST_RETIRED.ALL_BRANCHES
- 5 BR_INST_RETIRED.NEAR_CALL
- 6 RESOURCE_STALLS.ANY
- 7 UOPS_DECODED.STALL_CYCLES
- 8 UOPS_EXECUTED.CORE_STALL_CYCLES
- 9 UOPS_EXECUTED.PORT015
- 10 UOPS_EXECUTED.PORT234_CORE
- 11 UOPS_ISSUED.ANY
- 12 UOPS_ISSUED.FUSED
- 13 UOPS_ISSUED.STALL_CYCLES
- 14 UOPS_RETIRED.ANY
- 15 UOPS_RETIRED.STALL_CYCLES

Shortcut: nehalem-memory-access

- 1 CPU_CLK_UNHALTED.THREAD
- 2 INST_RETIRED.ANY
- 3 CPU_CLK_UNHALTED.REF
- 4 OFFCORE_RESPONSE.DATA_IN.LOCAL_DRAM_AND_REMOTE_CACHE_HIT_0
- 5 MEM_INST_RETIRED.LATENCY_ABOVE_THRESHOLD_128
- 6 MEM_INST_RETIRED.LOADS
- 7 MEM_INST_RETIRED.STORES
- 8 OFFCORE_RESPONSE.DATA_IN.REMOTE_DRAM_0
- 9 MEM_INST_RETIRED.LATENCY_ABOVE_THRESHOLD_32
- 10 MEM_LOAD_RETIRED.LLC_MISS
- 11 MEM_LOAD_RETIRED.LLC_UNSHARED_HIT
- 12 MEM_LOAD_RETIRED.OTHER_CORE_L2_HIT_HITM
- 13 MEM_UNCORE_RETIRED.LOCAL_DRAM_AND_REMOTE_CACHE_HIT
- 14 MEM_UNCORE_RETIRED.LOCAL_HITM
- 15 MEM_UNCORE_RETIRED.REMOTE_DRAM
- 16 MEM_UNCORE_RETIRED.REMOTE_HITM

Profiling tools: Intel® Trace Analyzer/Collector

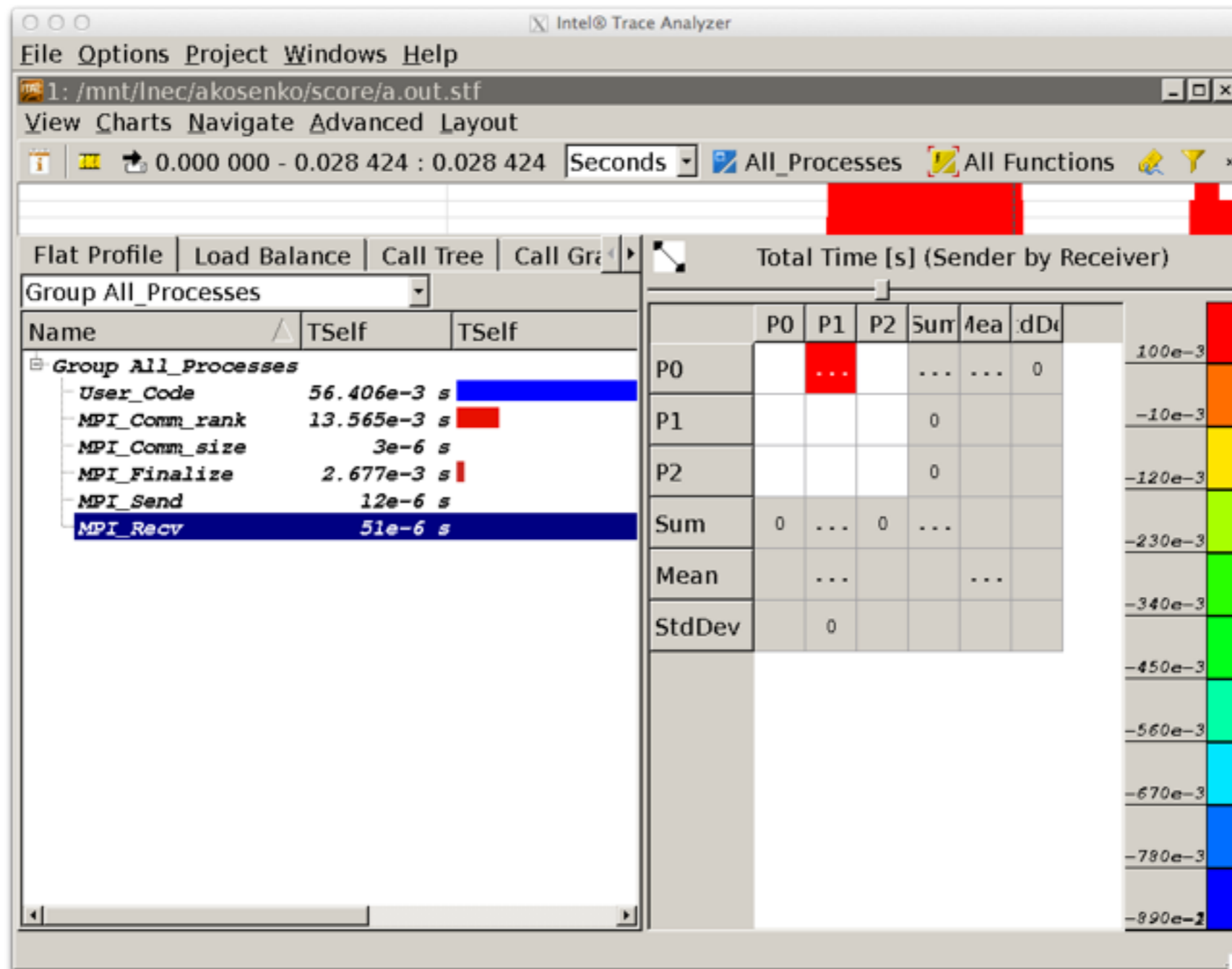
- Collect profile:

```
source /apps/monch/intel/impi/4.1.1.036/intel64/bin/mpivars.sh
source /apps/monch/intel/itac/8.1.3.037/intel64/bin/itacvars.sh
(optional) $ mpiicpc -trace -g test.cpp
$ mpiexec -trace -np N ./a.out
```

- Generate report:

```
$ ssh -Xt akosenko@ela.cscs.ch 'ssh -X monch; bash -i'
$ traceanalyzer (--cli --summary) a.out.stf
```

Profiling tools: Intel® Trace Analyzer/Collector



Profiling tools: IPM

<http://ipm-hpc.sourceforge.net/installation.html>

\$ mpiicpc test.cpp -L{ipm_path}/lib -lipm # or use LD_PRELOAD

\$ mpiexec -np N ./a.out

```
##IPMv0.983#####
#
# command : unknown (completed)
# host    : monch03/x86_64_Linux      mpi_tasks : 2 on 1 nodes
# start   : 06/20/14/22:48:12       wallclock : 0.002305 sec
# stop    : 06/20/14/22:48:12       %comm    : 1.87
# gbytes  : 0.00000e+00 total        gflop/sec : 0.00000e+00 total
#
#####
# region : *      [ntasks] =      2
#
#
#          [total]      <avg>      min      max
# entries          2          1          1          1
# wallclock      0.00450921  0.00225461  0.00220418  0.00230503
# user           0.011998    0.005999    0.005999    0.005999
# system         0.029994    0.014997    0.013997    0.015997
# mpi            8.64118e-05  4.32059e-05  6.66105e-06  7.97508e-05
# %comm          1.87442      0.302201    3.45986
# gflop/sec      0          0          0          0
# gbytes         0          0          0          0
#
#
#          [time]      [calls]      <%mpi>      <%wall>
# MPI_Recv       7.71349e-05  1          89.26      1.71
# MPI_Comm_rank  4.12506e-06    2          4.77      0.09
# MPI_Send       3.70922e-06    1          4.29      0.08
# MPI_Comm_size  1.44262e-06    2          1.67      0.03
#####
```


Profiling tools: GNU gprof

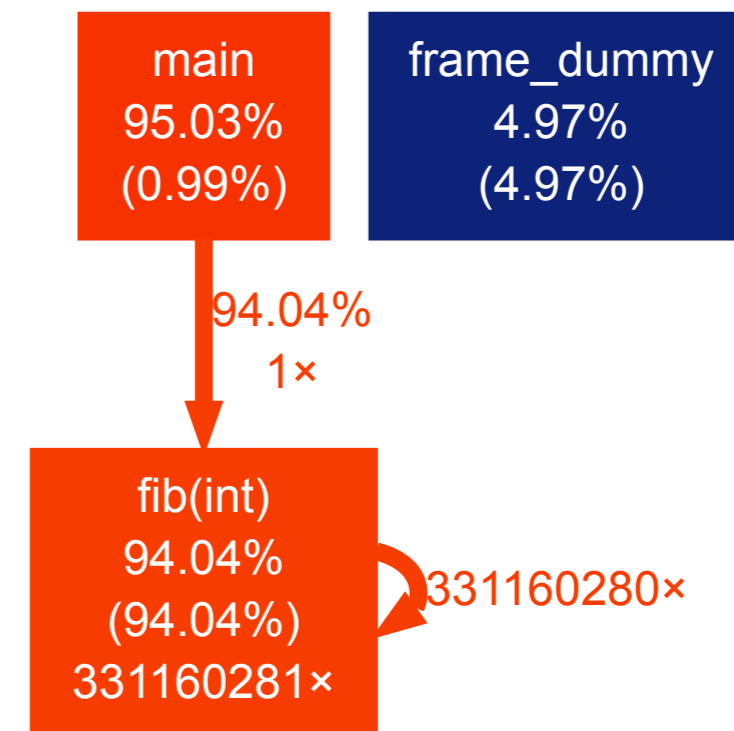
```
$ g++ -g -pg cilk.cpp
$ ./a.out
$ gprof -f <function_name> -e <function_name> a.out
$ gprof a.out | ./gprof2dot.py | dot -Tsvg -o output.svg
```

Flat profile:

Each sample counts as 0.01 seconds.

% time	cumulative seconds	self seconds	calls	self s/call	total s/call	name
95.08	2.84	2.84	1	2.84	2.84	fib(int)
4.90	2.99	0.15				frame_dummy
1.01	3.02	0.03				main

index	% time	self	children	called	name
[1]	95.2	0.03	2.84		<spontaneous>
		2.84	0.00	1/1	main [1]
					fib(int) [2]
			331160280		fib(int) [2]
[2]	94.1	2.84	0.00	1/1	main [1]
		2.84	0.00	1+331160280	fib(int) [2]
			331160280		fib(int) [2]
[3]	4.8	0.15	0.00		<spontaneous>
					frame_dummy [3]



Characterisation

- Scalability
- Uncore sensitivity
- Fabric sensitivity
- Extra: IO and Core (HT)

Characterisation

Scalability assessment:

$$t_0/t_N = f * c_N/c_0$$

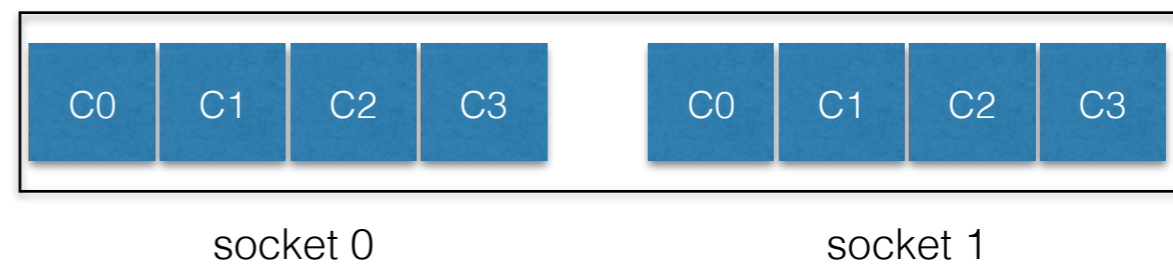
super: **> 1**

ideal: **.9 - 1**

good: **.7 - .8**

bad: **.0 - .6**

Pinning scheme:



Characterisation

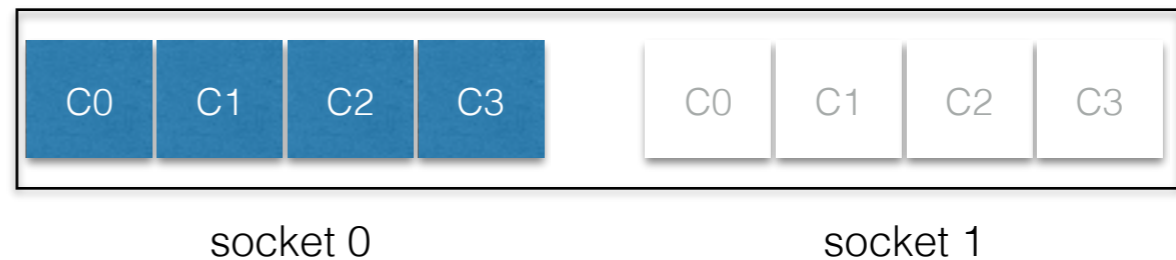
Uncore sensitivity:

$$f = t1/t2$$

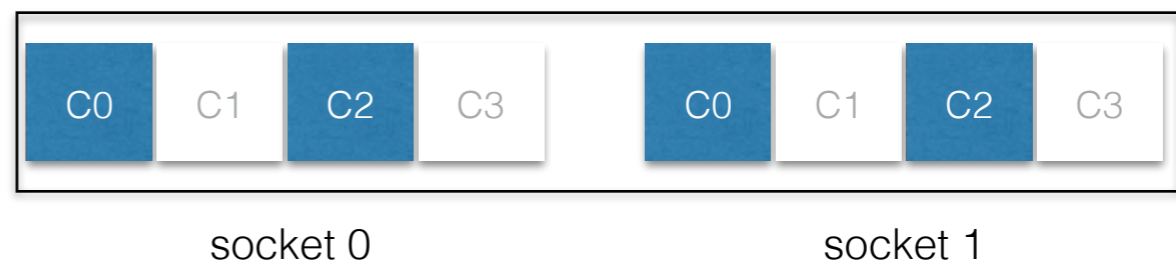
high: **> 1.8**

low: **< 1.1**

Pinning #1



Pinning #2



Characterisation

Fabric sensitivity:

$$f = t1/t2$$

high: **> 1.8**

low: **< 1.2**

MPI impact:

high: **> 35%**

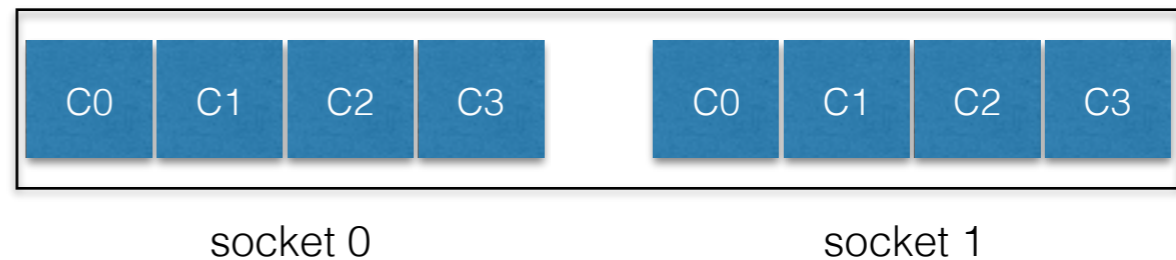
low: **< 15%**

MPI std:

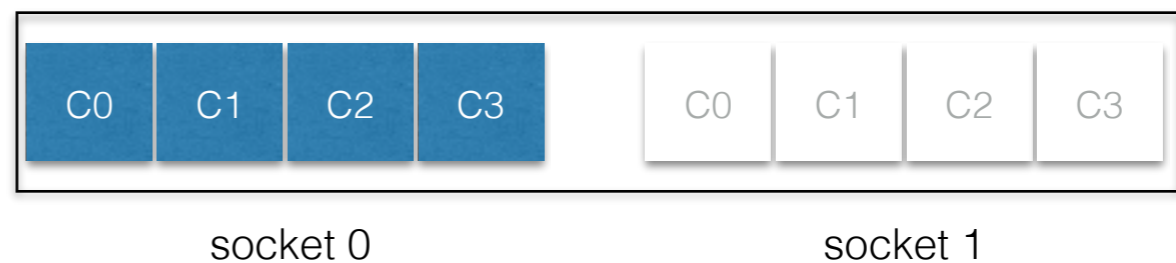
high: **> 20%**

low: **< 5%**

Pinning #1



Pinning #2



Questions

End of the second part