

Performance Analysis

14.12.2012

Reasons

- Use of the resources
- Optimization / Tuning

Metrics

- Time
 - Execution (Seconds)
 - Efficiency (Flops)
- Memory
 - Cache (Hit ratio)
 - Ram (Bytes)

Metrics

- Networking
 - Latency (Seconds)
 - Bandwidth (Bits / Second)
- Parallelism
 - MPI, OpenMP
 - Combination of time, networking

Methodologies

- Non invasive
 - Same code
- Invasive
 - Customized code
 - Preprocess before compiling
 - Compile with special flags

Methodologies

- **Virtual hardware**
 - Exact measurements
 - Simulation of the running
- **Actual hardware**
 - Hardware counters / Timing instructions
 - True performance

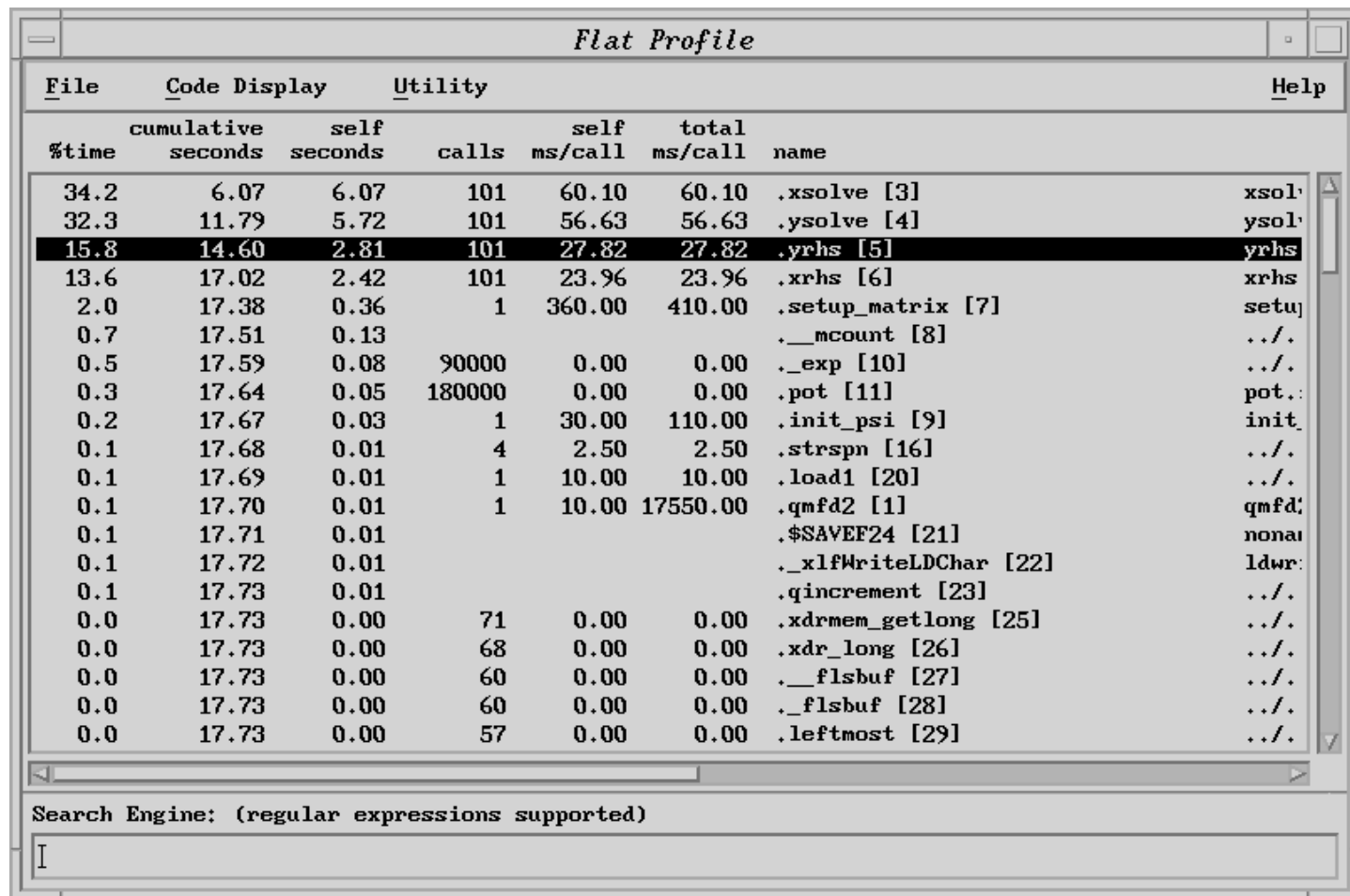
Tools

- Different tools = different metrics
- Different tools = different methodologies
- Be sure to pick the correct tool

Gprof [1]

- Time
 - Function level analysis
- Invasive
 - Recompilation with: -pg
- Actual hardware
- Command Line Interface
- Free
 - Shipped with gcc

Gprof



The screenshot shows the 'Flat Profile' window from the Gprof utility. The window title is 'Flat Profile'. It contains a table with columns for File, Code Display, Utility, and Help. The table lists various functions and their performance metrics. The function '.yrhs [5]' is highlighted in black.

File	Code Display	Utility	Help				
%time	cumulative seconds	self seconds	calls	self ms/call	total ms/call	name	
34.2	6.07	6.07	101	60.10	60.10	.xsolve [3]	xsol.
32.3	11.79	5.72	101	56.63	56.63	.ysolve [4]	ysol.
15.8	14.60	2.81	101	27.82	27.82	.yrhs [5]	yrhs
13.6	17.02	2.42	101	23.96	23.96	.xrhs [6]	xrhs
2.0	17.38	0.36	1	360.00	410.00	.setup_matrix [7]	setu
0.7	17.51	0.13				.__mcount [8]	./.
0.5	17.59	0.08	90000	0.00	0.00	._exp [10]	./.
0.3	17.64	0.05	180000	0.00	0.00	.pot [11]	pot.:
0.2	17.67	0.03	1	30.00	110.00	.init_psi [9]	init_
0.1	17.68	0.01	4	2.50	2.50	.strspn [16]	./.
0.1	17.69	0.01	1	10.00	10.00	.load1 [20]	./.
0.1	17.70	0.01	1	10.00	17550.00	.qmf2 [1]	qmf2
0.1	17.71	0.01				.\$SAVEF24 [21]	nona
0.1	17.72	0.01				._xlfwriteLDChar [22]	ldwr:
0.1	17.73	0.01				.qincrement [23]	./.
0.0	17.73	0.00	71	0.00	0.00	.xdrmem_getlong [25]	./.
0.0	17.73	0.00	68	0.00	0.00	.xdr_long [26]	./.
0.0	17.73	0.00	60	0.00	0.00	.__flsbuf [27]	./.
0.0	17.73	0.00	60	0.00	0.00	._flsbuf [28]	./.
0.0	17.73	0.00	57	0.00	0.00	.leftmost [29]	./.

Search Engine: (regular expressions supported)

Source: <http://cfile3.uf.tistory.com/image/18086A434E69BEB932AAE5>

Valgrind [2]

- Time
- Memory
 - Cache
 - Ram
- Non invasive
- Virtual hardware
- Command Line Interface
- Free

Valgrind

The screenshot displays the KCachegrind application window titled './cachegrind.out.23666 [kcachegrind (null)] - KCachegrind'. The interface is divided into several panes:

- Trace Part Overview:** Shows a summary of trace parts. Part 1 accounts for 70.39% and Part 2 for 22.8%.
- Flat Profile:** A table listing functions and their performance metrics. The 'main' function is highlighted in red, indicating it is the most significant.
- main:** Contains detailed analysis for the 'main' function, including a table of cost types and a 'Trace Part' table.

Cum.	Self	Called	Function
107.02	0.20	772	TreeMapWidget::x
100.00	0.00	(active)	0x00001E70
99.73	0.00	1	0x08056360
99.73	0.00	1	__libc_start_main
99.35	0.00	1	main
94.55	0.15	1 026	TreeMapWidget::x
49.32	0.00	1	QApplication::exe
49.32	0.00	1	QEventLoop::exe
49.32	0.00	1	QEventLoop::ente
47.20	0.04	2 915	KApplication::notif
47.17	0.08	2 919	QApplication::notif
47.07	0.18	2 937	QApplication::inte
47.07	0.07	176	QEventLoop::proc
35.87	0.05	2 649	QWidget::event(C
25.61	0.10	1 709	QApplication::x11
23.54	0.00	1	KApplication::KAp
22.36	0.04	457	QObject::activate
22.31	0.00	1	TopLevel::TopLev
16.65	0.00	1	QApplication::QA
16.65	0.00	1	QApplication::con
16.36	0.00	1	qt_init
16.36	0.01	1	qt_init_internal
15.58	0.17	1 289	QApplication::sen
14.00	0.01	1 246	KApplication::notif
13.98	0.02	1 246	QApplication::noti

Cost Type	Cum.	Self
Instruktion	99.63	0.00
Lesezugriff	99.63	0.00
Schreibzugriff	99.71	0.00
L1-Verfehlung Instruktion	99.91	0.00
L1-Verfehlung bei Lesezugriff	99.05	0.00
L1-Verfehlung bei Schreibzugriff	99.61	0.00
L2-Verfehlung Instruktion	98.47	0.02
L2-Verfehlung bei Lesezugriff	97.63	0.00
L2-Verfehlung bei Schreibzugriff	98.98	0.00
L1-Verfehlungssumme	99.63	0.00
L2-Verfehlungssumme	98.11	0.01
Abschätzung CPU-Takte	99.35	0.00

Trace Part	Cum.	Self	Called	Calling
Part.1	99.60	0.00	1	33
Part.2	100.00	0.00	(active)	(active)
Part	94.58	0.00	(active)	4

cachegrind.out.23666 [1-3] - Total Abschätzung CPU-Takte Cost: 581 885 467

Source: <http://kcachegrind.sourceforge.net/html/Shot1Large.html>

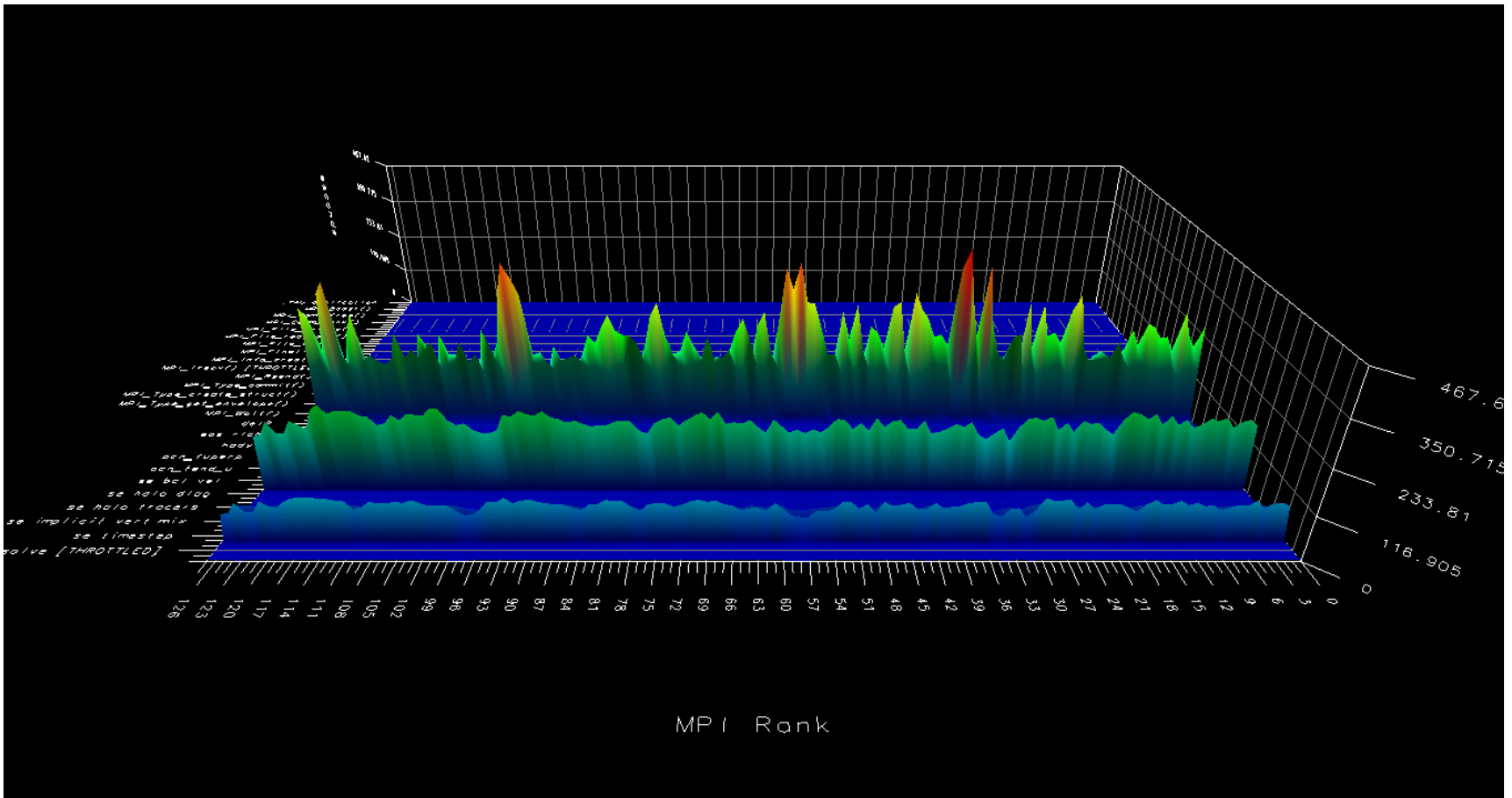
PAPI [3]

- Time
- Memory
 - Cache
- Actual Hardware
 - Hardware counters reader
- Invasive
 - C Library
- Free

Tau [4]

- Time
- Memory
 - Cache
 - Ram
- Parallelism
- Invasive
- Actual hardware
- Command Line Interface + Visualization Application
- Free

Tau



Source: http://www.tau.uoregon.edu/mediawiki-tau/images/1/18/MPI_wait-computation-correlation.png

Intel VTune Amplifier [5]

- Time
- Memory
 - Cache
- Parallelism
- Non Invasive
- Actual hardware
- CLI / GUI
- Commercial

Intel VTune Amplifier

Intel® VTune™ Amplifier XE Performance Profiler

Hotspot Analysis

Function - Call Stack	CPU Time	Module
algorithm_2	3.560s	matrix.exe
algorithm_1	1.412s	matrix.exe
BaseThreadInitThunk	0.000s	kernel32.dll
main	0s	matrix.exe

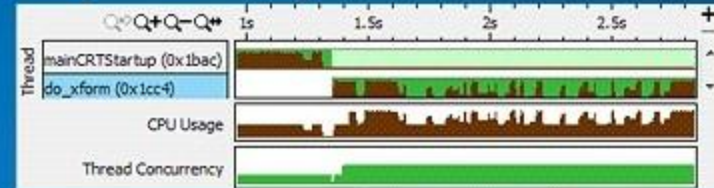
- Find performance bottlenecks
- Functions sorted by amount of CPU time

Concurrency Analysis

Function - Call Stack	CPU Time	Color Legend
GenerateScanLine	1.656s	Idle (Grey), Poor (Red), Ok (Orange), Ideal (Green)
DD_BltBackToPrimary	0.547s	
Paint ← Generate_Display ← BaseThreadStart	0.547s	
DD_Init	0.031s	
Paint	0.016s	

- Color shows # of cores utilized
- Click [+] to view call stacks

Simplified Analysis



- Powerful profile analysis - timeline, filtering and frame analysis to help turn raw data into actionable information.
- Low overhead data collection - faster data collection and more accurate results.
- Tune threaded and non-threaded code.
- Used normal production build - No special builds required.
- Supports C++, Fortran, assembly, and more on Windows* or Linux*, 32 bit and 64 bit.

Brothersoft

Thank You for the Attention

Questions are welcome

References

- [1] - http://www.cs.utah.edu/dept/old/texinfo/as/gprof_toc.html
- [2] - <http://valgrind.org/>
- [3] - <http://icl.cs.utk.edu/papi/>
- [4] - <http://www.cs.uoregon.edu/Research/tau/home.php>
- [5] - <http://software.intel.com/en-us/intel-vtune-amplifier-xe>