PDC Summer School 2011 Brief introduction to Serial performance measurement

KTH, August 22, 2011

## Basic questions on HPC systems

We've seen that hardware is complicated (cache memories, TLBs, SSE-vectorization) and that the way we code impacts performance.

Most of this is hidden from the coder – we have to evaluate performance empirically, *a posteriori*, at various levels of interest:

- Which functions in my (large) code take the most time?
- Can the code be optimized? Where? Is it worth the effort?
- What did the compiler do with my code?
- Is arithmetic handled in SSE-registers or on the FPU stack?
- Are fused arithmetic operations (SSE) issued?
- Does my code generate a lot of TLB misses?
- Are branch mispredictions causing stalls?

### We need tools

#### Basic timing

How long did it take to run the program?

#### Sampling profilers

Where is my program spending most time? What line? Which machine instruction?

#### Hardware event counting

How many hardware events (e.g. cache misses) were actually triggered?

#### Emulators

Try to estimate hardware events by emulation instead.

Will try to cover some of the most common tools on Linux.

#### An example program: Jacobi iteration



## Basic timing

#### UNIX time

```
> time ./jacobi
Jacobi iteration converged in 263 iterations.
0.420u 0.000s 0:00.41 102.4% 0+0k 0+0io 0pf+0w
```

#### System time in C

```
#include <sys/time.h>
double gettime(void)
{
    struct timeval tv;
    gettimeofday(&tv,NULL);
    return tv.tv_sec + 1e-6*tv.tv_usec;
}
int main(void){
    double t = gettime();
    jacobi_solver(100);
    t = gettime()-t;
}
```

## Sampling profilers

Works by periodically stopping the program and investigating the stack.

- GNU 'gprof'
- Intel VTune
- Valgrind 'callgrind' + KDE-based GUI kcachegrind

On Mac OS X, 'Shark' part of XCode (very good)

## Sampling profiler: GNU gprof

Basic and reliable. Flat profile and call graph.

- Compile code with flag '-pg'
- Run program
- Run profiler, 'gprof ./jacobi'

| 5 | 1   |                            | 1                 |           | Mar.         | Ell'      |  |  |  |  |  |
|---|---|----------------------------|-------------------|-----------|--------------|-----------|--|--|--|--|--|
|   | N 😣 🗠   | 🙆 Termin                   | al 🛛              |           |              |           |  |  |  |  |  |
|   | File Edit                                     | View Termi                 | nal Help          |           |              |           |  |  |  |  |  |
|   | na55:ta<br>na55:ta<br>na55:ta                 | lk ><br>lk ><br>lk > gcc - | pg -g -std        | l=c99 iac | obi base.    | c -o iaco | Recompile with -pg   |  |  |  |  |
|   | Jacobi iteration converged in 369 iterations. |                            |                   |           |              |           |  |  |  |  |  |
|   | na55:ta<br>Flat pr                            | ilk > gprof<br>ofile:      | flat-pr           | ofile -1  | brief        | ./jacobi  | "Flat" profile   |  |  |  |  |
|   | Each sa<br>% c                                | mple count<br>umulative    | s as 0.01<br>self | seconds.  | self         | total     |  |  |  |  |  |
|   | time<br>48.51<br>25.08                        | seconds<br>2.35            | seconds<br>2.35   | calls     | Ts∕call      | Ts/call   | name FUNCTION NAME Line number<br>jacobi_iteration (jacobi_base.c:22 @ 4008a4)<br>jacobi_iteration (jacobi_base.c:26 @ 400840) |  |  |  |  |
|   | 24.25   | 4.74                       | 1.17              |           |              |           | jacobi_iteration (jacobi_base.c:25 @ 4009a5)<br>jacobi_iteration (jacobi_base.c:19 @ 400a08)                                   |  |  |  |  |
|   | 0.62  | 4.86<br>4.86               | 0.03              | 738       | 0.00         | 0.00      | jacobi_iteration (jacobi_base.c:19 @ 400898)<br>jacobi_iteration (jacobi_base.c:13 @ 400844)                                   |  |  |  |  |
|   | 0.00  | 4.86                       | 0.00              | 3<br>2    | 0.00<br>0.00 | 0.00      | init_matrix (jacobi_base.c:33 @ 400a3d)<br>init_boundary (jacobi_base.c:42 @ 400aa7)   |  |  |  |  |
|   | na55:ta                                       | lk >                       |                   |           |              |           |  |  |  |  |  |
|   |   |                            |                   |           |              |           |  |  |  |  |  |

## Sampling profiler: Intel VTune

Commercial, fancy GUI, sophisticated\*, (marketing hype). Also some hardware events. Free "non-commercial" download for Linux – so try it!

| 2 2 2 //df/huds.kth.se/home/c/u1dkgkic/intel/empixe/projectu/jac - Intel VTune Amplifier XI 2011 Re: Telp |   |        |         |      |                        |             |     |         |  |  |  |
|---|---|--------|---------|------|------------------------|-------------|-----|---------|--|--|--|
|   |   |        |         |      |                        |             |     |         |  |  |  |
| 100ms 🐹   |   |        |         |      |                        |             |     |         |  |  |  |
| Hatenots Hatspats / Amalifier XE 2011   |   |        |         |      |                        |             |     |         |  |  |  |
| inter of the Ampine: A2 2017  |   |        |         |      |                        |             |     |         |  |  |  |
| 🔮 Analysis Target 🖉 Analysis Type 🔛 Collection Log 🔣 Summary 🚯 Bottom-up 🚯 Top-down Tree 🗋 jacobi Ja 🖬    |   |        |         |      |                        |             |     |         |  |  |  |
| Seurce Assembly 👔 🔠 🗇 🗇 🧐 😰   |   |        |         |      |                        |             |     |         |  |  |  |
| Line  | Source  | CPU A  | Address | Line | Assembly               | CPU Time    | * * | 0,25    |  |  |  |
| 1   | /*  |        | 0x809   | 22   | add %rcx, %rax         | 79.829ms    |     | cur Cur |  |  |  |
| 2   | gcc -O3 -Wall -ansi -pedantic -std=c99 -fstrict-        |        | 0x80c   | 22   | movsdq (%rax), %xmm2   |             |     | 50.     |  |  |  |
| 3   | */  |        | 0x810   | 22   | mulsd %xmm2, %xmm1     | 79.796ms    |     | jacobiš |  |  |  |
| 4   | #include <stdio.n></stdio.n>                            |        | 0x814   | 22   | subsd sxmm1, sxmm0     | 339.338ms   | _   | jacobih |  |  |  |
|   | tinclude crys/time by                                   |        | 0x81c   | 22   | novi - Oxd(tchn) text  | 99.950mm    |     | jacobi! |  |  |  |
| 7   |   |        | 0x81 f  | 25   | cdae                   | 555050mg    |     |         |  |  |  |
| 8   | /* Elementary jacobi iteration */                       |        | 0x821   | 25   | shl \$0x3, %rax        |             |     |         |  |  |  |
| 9   | double jacobi iteration(int N,                          |        | 0x825   | 25   | addg -0x30(%rbp), %rax |             |     | •       |  |  |  |
| 10  | double* restrict A[N],                                  |        | 0x829   | 25   | movq (%rax), %rax      | 29.927ms    |     | •       |  |  |  |
| 11  | double* restrict B[N],                                  |        | 0x82c   | 25   | movl -0x8(%rbp), %edx  |             |     |         |  |  |  |
| 12  | double* restrict F[N])                                  |        | 0x82f   | 25   | movsxd %edx, %rdx      |             |     |         |  |  |  |
| 13  | (   |        | 0x832   | 25   | shl \$0x3, %rdx        |             |     |         |  |  |  |
| 14  | double diff = 0, tmp;                                   |        | 0x836   | 25   | add %rdx, %rax         | 69.829ms    |     |         |  |  |  |
| 15  | double h = 1.0/(N+1);                                   |        | 0x839   | 25   | movsdq (%rax), %xmm0   |             |     |         |  |  |  |
| 16  | int i,j;  |        | 0x83d   | 25   | movl -0x4(%rbp), %eax  | 771.209ms   |     | •       |  |  |  |
| 17  | dealer as a state of the                                |        | 0x840   | 25   | cdd 3                  |             |     | •       |  |  |  |
| 10  | TOP(1=1; 1 <n-1; 1++)<br="">fer(i=1, i=N-1; i++)</n-1;> | 0.190  | 0x842   | 20   | shi \$0x3, %rax        |             |     | •       |  |  |  |
| 20  | /   | 0.1005 | 0x040   | 25   | aduq -0x30(srup), srax | 40.979mm    |     | *       |  |  |  |
| 20  | /# Jacobi undate #/                                     |        | 0x84d   | 25   | movi (star), star      | 49/0/0115 B |     |         |  |  |  |
| 22  | B[i][i] = 0.25*(A[i-1][i]+A[i+1][i]+A[i]])              | 2.2445 | 0x850   | 25   | movsxd hedx, hrdx      |             |     |         |  |  |  |
| 23  |   |        | 0x853   | 25   | shl \$0x3, %rdx        |             | _   |         |  |  |  |
| 24  | /* Compute diff. Contribution from boundar              |        | 0x857   | 25   | add %rdx, %rax         | 59.858ms    |     |         |  |  |  |
| 25  | <pre>tmp = A[i][j]-B[i][j];</pre>                       | 1.190s | 0x85a   | 25   | movsdq (%rax), %xmml   |             |     |         |  |  |  |
| 26  | diff += tmp*tmp;  | 1.265s | 0x85e   | 25   | subsd %xmml, %xmm0     |             |     |         |  |  |  |
| 27  | }   |        | 0x862   | 25   | movsdq %xmm0, -0x18(%r | 109.701ms 📃 |     |         |  |  |  |
| 28  | return diff;  |        | 0x867   | 26   | movsdq -Ox18(%rbp), %x | 69.835ms    |     |         |  |  |  |
| 29  | }   |        | 0x86c   | 26   | mulsdq -Ox18(%rbp), %x | 299.229ms   | _   |         |  |  |  |
| 30  |   |        | 0x871   | 26   | movsdq -Ox10(%rbp), %x | 627.034ms   |     |         |  |  |  |
| 31  | /* initialize all matrix entries to a given value       |        | 0x876   | 26   | acoso sxeni, sxeeo     | 260 209mm   |     |         |  |  |  |
| 32  | Your init_metrixtint w, double* A[N], double Val)       |        | 0x87f   | 20   | add] \$0v1 .0x10(sr    | 69.817ms    |     |         |  |  |  |
| 34  |   |        | 0x883   | 19   | Block 4:               |             |     |         |  |  |  |
|   |   |        |         |      |                        |             |     |         |  |  |  |

Registers used by the hardware manufacturers to debug and evaluate their designs are left and can be used for detailed profiling.

#### Papi

- Records actual number of hardware events that occurred!
- Free and open source: http://icl.cs.utk.edu/~mucci/papiex/
- By Phil Mucci (previously at PDC and gave this lecture)
- Not easy to install\*: kernel modules, major dependency chain.
- What PAPI can record depends on hardware.

### Papi front-ends

- PapiEx
- Cray PAT (though this is much more!)

\* Mucci could do it :)

### Hardware event counters

What can Papi determine?

> papi\_avail
Available events and hardware information.

PAPI\_L1\_DCMLevel 1 data cache missesPAPI\_L2\_DCMLevel 2 data cache missesPAPI\_TLB\_DMData translation lookaside buffer missesPAPI\_HW\_INTHardware interruptsPAPI\_BR\_MSPConditional branch instructions mispredictedPAPI\_TOT\_INSInstructions completedPAPI\_FP\_INSFloating point instructionsPAPI\_TOT\_CYCTotal cycles

(...)

### PapiEx basic usage

```
> papiex ./jacobi
(...)
Derived Metrics:
_____
MFLOPS .....
                                   197.50
(...)
PAPI TOT CYC .....
                            1.93957e+10
PAPI_FP_OPS .....
                            1.44011e+09
PAPI TOT CYC
                       : Total cycles
PAPI_FP_OPS
                       : Floating point operations
Get specific counter:
> papiex -e PAPI_L1_DCM ./jacobi
(...)
L1 Data Cache Misses ..... 1.17406e+09
```

# Cray PAT

Very sophisticated framework for performance analysis (and MPI, OpenMP). Commercial, available on high-end Cray systems.

... Not simple. Steps (see tutorial on lab):

- Select (PAPI) event group: env PAT\_RT\_HWPC
- Prepare the executable: pat\_build ./jacobi
- Run: aprun -n 1 ./jacobi+pat
- View report: pat\_report jacobi+pat+<RUN ID>.xf (next slide)

#### Hardware counter groups

> export PAT\_RT\_HWPC=

| 1  | Summary with TLB metrics        |
|----|---------------------------------|
| 2  | L1 and L2 metrics               |
| 3  | Bandwidth information           |
| 8  | Instructions and branches       |
| 12 | Floating point operations (SSE) |

## Cray PAT Sample output

In counter group 1:

| _ |                       |           |          |         |          |     |            |
|---|-----------------------|-----------|----------|---------|----------|-----|------------|
| _ | USER / jacobi_iterat  | ion / jac | obi_base | e.c     |          |     |            |
| _ | <br>Samp%             |           |          |         | <br>99   | .9% |            |
|   | Samp                  |           |          |         |          | 772 |            |
|   | PAPI_L1_DCM           |           | 2.621    | l/sec   | 32335    | 186 | misses     |
|   | PAPI_TLB_DM           |           | 0.125M   | l/sec   | 1541     | 704 | misses     |
|   | PAPI_L1_DCA           | 1         | 348.371  | l/sec   | 16633470 | 668 | refs       |
|   | PAPI_FP_OPS           | 1         | 726.230  | l/sec   | 21294737 | 308 | ops        |
|   | User time (approx)    |           | 12.336   | secs    | 24671990 | 403 | cycles     |
|   | FLOPs 1               | 726.230M/ | sec 212  | 2947373 | 08 ops   | 21. | 6%peak(DP) |
|   | Computational intensi | ty        | 0.86     | ops/cy  | cle 1    | .28 | ops/ref    |
|   | MFLOPS (aggregate)    |           | 1726.23  | l/sec   |          |     |            |
|   | TLB utilization       | 1         | 0789.02  | refs/m  | iss 21.  | 072 | avg uses   |
|   | D1 cache hit,miss rat | ios       | 99.8%    | hits    | 0        | .2% | misses     |
|   | D1 cache utilization  | (misses)  | 514.41   | refs/m  | iss 64.  | 301 | avg hits   |

## Emulators

 $\ensuremath{\mathsf{Try}}$  to extract same information as hardware counters record, but with modelling of the architecture.

- Valgrind 'cachegrind' + KDE-based GUI 'kcachegrind'
- Accumem ThreadSpotter (commercial) By Prof. Hagersten et. al. (now owned by Rogue Wave Software)

Results are only as accurate as the emulator. If the emulator has incorrect or incomplete parameters for the present architecture it will give a warning (and those warnings are important)

## Emulator: Cachegrind

Part of the Valgrind tool-set that you of course already use for checking for memory leaks (right?).

| e Edit View Terminal Help |                                       |                    |                                       |  |
|---------------------------|---------------------------------------|--------------------|---------------------------------------|--|
| 55:talk > valgrind        | tool=cachegrind .                     | /jacobi            |                                       |  |
| 5343== Cachegrind, a      | cache and branch-                     | prediction profile | er<br>Ing Nothersete et al            |  |
| 5343== Copyright (C) .    | 2002-2009, and GN<br>3 6 0 SVN Debian | and Lib/EV: reru   | a with b for convright info           |  |
| 5343== Command: /jac      | nhi                                   |                    | With - H for copyright into           |  |
| 5343==                    | 501                                   |                    |                                       |  |
|                           |                                       |                    |                                       |  |
|                           |                                       |                    |                                       |  |
| cobi iteration falied     | to converge.                          |                    |                                       |  |
| 5343==                    |                                       |                    |                                       |  |
| 5343== I refs:            | 2,637,158,288                         |                    |                                       |  |
| 5343== I1 misses:         | 806                                   |                    |                                       |  |
| 5343== L2i misses:        | 803                                   |                    |                                       |  |
| 5343== I1 miss rate:      | 0.00%                                 |                    |                                       |  |
| 5343== L2i miss rate:     | 0.00%                                 |                    |                                       |  |
| 5343==                    | 1 202 217 560 (                       | 1 200 200 502      |                                       |  |
| 5343== U rets:            | 1,282,217,568 (                       | 1,206,390,503 Fd   | + /5,82/,065 WF)                      |  |
| 5343== UT MISSES;         | 9,544,207 (                           | 6,509,441 ru       | + 3,234,700 WF)                       |  |
| 5343 = L20 misses:        | 9,520,976 (                           | 0,290,969 10       | + 3,230,007 Wr)                       |  |
| 5343 = 12d miss rate:     | 0.7% (                                | 0.5%               | + 4.2%                                |  |
| 5343==                    | 01770 (                               | 015/0              | · · · · · · · · · · · · · · · · · · · |  |
| 5343== L2 refs:           | 9,545,013 (                           | 6.310.247 rd       | + 3,234,766 wr)                       |  |
| 5343== L2 misses:         | 9,521,779 (                           | 6,291,772 rd       | + 3,230,007 wr)                       |  |
| 5343== L2 miss rate:      | 0.2% (                                | 0.1%               | + 4.2%)                               |  |
| 55:talk >                 |                                       |                    |                                       |  |
| 55:talk >                 |                                       |                    |                                       |  |

Note the argument: valgrind --tool=cachegrind

## Emulator: Cachegrind GUI

| Solution (1990) | s/nada.kth.se/nome/c/u1d4g64c<br>o Settings Heln   | c/Private/underv/HPC_11/tali   | k/cachegrin                             | d.out.5368 [       | ./jacobi]  |  |  |                    |                  |
|---|--|--|---|--------------------|--|--|--|--------------------|------------------|
| Open 4  | Back V i Forward V   | 🔶 Up 🗸 % Relative  | Sycle                                   | Detection          | Relative   | to Parent  | L1 Data Read Miss                              | ~                  | 0                |
| Flat Profile  |  | * ×  | jacobi                                  | _iteration         |  |  |  | ~                  |                  |
| S <u>e</u> arch:  |  | (No Grouping) 🗸  | Types                                   | Callers            | All Callers  | Callee Ma  | p Source Code                                  | - 1 L              |                  |
| Self Fu<br>99.99  | nction<br>acobi_iteration<br>ht_boundary<br>ht_matrix<br>dl_addr<br>dl_relocate_object<br>o_lookup_x | Location<br>Jacobi_base.c<br>Jacobi_base.c<br>Jacobi_base.c<br>dl-addr.c<br>dl-machine.h, dl-relo<br>dl-hash.h, dl-lookup.                           | # D<br>12<br>13<br>14<br>15<br>16<br>17 | 1mr S<br>{<br>0.00 | double diff =<br>double h = 1<br>int i,j;            | ada.kth.se/<br>double* res<br>0. tmp:<br>.0/(N+1); | home/c/u1d4g64c/Pr<br>strict F[N])             | ivate/underv/HPC   | _11/talk/jacobi_ |
| 0.00 0<br>0.00 0<br>0.00 0<br>0.00 0<br>0.00 0  | dl_lookup_symbol_x<br>tmalloc_init<br>dl_next_ld_env_entry<br>nain<br>heck_match.12168               | dl-lookup.c<br>arena.c<br>dl-environ.c<br>jacobi_base.c<br>dl-lookup.c   | 18<br>19<br>20<br>21<br>22              | 99.99              | for(i=1; i <n-<br>for(j=1; j&lt;<br/>{</n-<br>       | l; i++)<br>N-1; j++)<br>i update */<br>0.25*(A[i-1 | ](j]+A[i+1](j]+A[i](j-                         | L]+A[i][j+1])-h*h* | F(1)();          |
|   | trcmp<br>dl_start<br>dl_fixup<br>l_main<br>dl_sysdep_start<br>dl_map_object_from_fd<br>dl_fini       | strcmp.S<br>dl-machine.h, do-rel.<br>dl-runtime.c, dl-mach<br>dynamic-link.h, rtld.c<br>dl-sysdep.c, dl-sysde<br>dl-load.c, dynamic-lir<br>dl-fini.c | 23<br>24<br>25<br>26<br>27<br>28<br>29  | 0.00}              | /* Comp<br>tmp = A<br>diff += -<br>}<br>return diff; | ute diff. Co<br>(i) [j]-B[i] [j]:<br>tmp*tmp:      | ntribution from boun                           | dary is zero */    |                  |
| 0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00<br>0.00 00 00 00 00 000 0  | dl_init<br>unknown)<br>ntel_02_known_compare<br>I0_flush_all_lockp                                   | dl-init.c<br>(unknown)<br>cacheinfo.c<br>genops.c  | 30<br>31<br>32                          | /*<br>V            | initialize all m<br>bid init_matrix(                 | atrix entrie<br>int N, doub                        | s to a given value */<br>le* A[N], double val) |                    |                  |

KDE-based GUI  ${\tt kcachegrind}$  available in most Linux distros. Similar tool in XCode on Mac OS X.

### What did the compiler accomplish?

Example: did the compiler emit packed SSE memory transactions and arithmetic for the Jacobi inner loop?

#### OK, now it gets more technical!

Can generate disassembly from executable:

```
> objdump -d ./jacobi > jacobi_dump.asm
```

In this case, 134300 lines of assembly code! How do we determine where the inner loop is?

```
Use Valgrind (again):
```

```
> valgrind --tool=callgrind --dump-instr=yes --collect-jumps=yes
```

KCachegrind will now show us the loops:

## Assembly-code annotation with Valgrind

| Application                          | ons Places | : System 🙋 🛛 🖬 Term   | inal    | Jcalgrind.out.5523 [./ja         |                                      | 🗐 USA 🜓) 🛛 🔀 14:19 🛞 dag | Ċ     |  |
|--------------------------------------|------------|-----------------------|---------|----------------------------------|--------------------------------------|--------------------------|-------|--|
| 🔕 🛇 🚯 _/caligrind.out.5523 [/jacobi] |            |                       |         |                                  |                                      |                          |       |  |
| File View Go Settings Help           |            |                       |         |                                  |                                      |                          |       |  |
|                                      |            |                       |         |                                  |                                      |                          |       |  |
| Open                                 | da pa      | ik v 🏬 rorward v 🖶    | op v    | % Relative                       | Relative to Parent Instruction Petch | ~                        |       |  |
| main                                 |            |                       |         |                                  |                                      |                          |       |  |
| Troop                                | allers     | All College Man       | Courses | and a                            |                                      |                          |       |  |
| Types C                              | alleis     | Au callers callee map | Source  | tode                             |                                      |                          |       |  |
| Event Type                           | e i        | incl. Self S          | Short F | ormula                           |                                      |                          |       |  |
| Instruction                          | in Fetch   | 100.00 98.47          | lr      |                                  |                                      |                          | - 1   |  |
|                                      |            |                       |         |                                  |                                      |                          |       |  |
|                                      |            |                       |         |                                  |                                      |                          |       |  |
| #                                    | Ir         | Hex                   |         | Assembly Instructions            | Source Position                      |                          | i i i |  |
| 40 07B4                              | 0.00       | 0f 57 db              | xorps   | %xmm3,%xmm3                      | jacobi_base.c:113                    |                          |       |  |
| 40 07B7                              | 0.00       | 33 c0                 | xor     | %eax,%eax                        | jacobi_base.c:113                    |                          |       |  |
| 40 07B9                              | 0.00       | ← 4c 8b 5c cl 08      | mov     | 0x8(%rcx,%rax,8),%rl1            | jacobi_base.c:88                     |                          |       |  |
| 40 07BE                              | 0.00       | 45 33 f6              | xor     | %r14d,%r14d                      | jacobi_base.c:113                    |                          |       |  |
| 40 07C1                              | 0.00       | 4c 8b 54 c6 08        | mov     | 0x8(%rsi,%rax,8),%r10            | jacobi_base.c:87                     |                          |       |  |
| 40 07C6                              | 0.00       | 4c 8b 4c c6 10        | mov     | 0x10(%rsi,%rax,8),%r9            | jacobi_base.c:87                     |                          |       |  |
| 40 07CB                              | 0.00       | 4c 8b 04 c6           | mov     | (%rsi,%rax,8),%r8                | jacobi_base.c:87                     |                          |       |  |
| 40 07CF                              | 0.00       | 48 8b 5c c7 08        | mov     | 0x8(%rdi,%rax,8),%rbx            | jacobi_base.c:86                     |                          |       |  |
| 40 0704                              | 0.00       | 0f 1f 44 00 00        | nopi    | 0x0(%rax,%rax,1)                 | Jacobi_base.c:86                     |                          |       |  |
| 40 0709                              | 0.00       | 00 00 00 00 00 00 00  | nopi    | 0x0(%rax)                        | Jacobi_base.c:86                     |                          |       |  |
| 40 07E0                              | 3.34       | 12 43 0F 10 6C TO 08  | movsa   | 0x8(%r8,%r14,8),%xmm5            | Jacobi_base.c:87                     |                          |       |  |
| 40 0727                              | 3.34       | 12 43 01 58 00 11 08  | addsd   | 0x8(7019,70114,8),70x111113      | jacobi_base.c:113                    |                          |       |  |
| 40 0764                              | 3.34       | f2 43 0F 10 74 F2 09  | mound   | (70110,70114,8),70XIIIII3        | jacobi_base.c:115                    |                          |       |  |
| 40 07FB                              | 3 34       | f2 43 0f 58 6c f2 10  | babba   | 0x10(%r10.%r14.8)%ymm5           | jacobi_base.ci113                    |                          |       |  |
| 40 0802                              | 3 34       | f2 0f 59 e9           | mulsd   | %vmm1 %vmm5                      | jacobi base c:113                    |                          |       |  |
| 40 0806                              | 3.34       | f2 43 0f 10 64 f3 08  | movsd   | 0x8(%r11.%r14.8).%xmm4           | acobi base.c:88                      |                          |       |  |
| 40 080D                              | 3.34       | f2 0f 59 e0           | mulsd   | %xmm0.%xmm4                      | jacobi base.c:113                    |                          |       |  |
| 40 0811                              | 3.34       | f2 Of 5c ec           | subsd   | %xmm4,%xmm5                      | jacobi base.c:113                    |                          |       |  |
| 40 0815                              | 3.34       | f2 42 0f 11 6c f3 08  | movsd   | %xmm5,0x8(%rbx,%r14,8)           | jacobi_base.c:86                     |                          |       |  |
| 40 081C                              | 3.34       | 48 8b 5c c7 08        | mov     | 0x8(%rdi,%rax,8),%rbx            | jacobi_base.c:86                     |                          |       |  |
| 40 0821                              | 3.34       | f2 42 0f 5c 74 f3 08  | subsd   | 0x8(%rbx,%r14,8),%xmm6           | jacobi_base.c:113                    |                          |       |  |
| 40 0828                              | 3.34       | 49 ff c6              | inc     | %r14                             | jacobi_base.c:113                    |                          |       |  |
| 40 082B                              | 3.34       | f2 0f 59 f6           | mulsd   | %xmm6,%xmm6                      | jacobi_base.c:113                    |                          |       |  |
| 40 082F                              | 3.34       | 49 81 fe b8 0b 00 00  | 0 cmp   | \$0xbb8,%r14                     | jacobi_base.c:113                    |                          |       |  |
| 40 0836                              | 3.34       | f2 0f 58 de           | addsd   | %xmm6,%xmm3                      | jacobi_base.c:113                    |                          |       |  |
| 40 083A                              | 3.34       | 72 a4                 | jb      | 4007e0 <main+0x3e0></main+0x3e0> | jacobi_base.c:113                    |                          |       |  |
| -                                    |            |                       |         | Jump 44 985 000 of 45 000 001    | ) times to 0x4007E0                  |                          |       |  |
| 40 083C                              | 0.00       | 48 ff c0              | inc     | %rax                             | jacobi_base.c:113                    |                          |       |  |
| 40 083F                              | 0.00       | 48 3d b8 0b 00 00     | cmp     | \$0xbb8,%rax                     | jacobi_base.c:113                    |                          |       |  |
| 40 0845                              | 0.00       | 0f 82 6e ff ff ff     | jb      | 4007b9 <main+0x3b9></main+0x3b9> | jacobi_base.c:113                    |                          |       |  |
| 40.0040                              | 0.00       | 44 ph 75 (p           |         | Jump 14 995 of 15 000 times to   | 0x400789                             |                          |       |  |
| 40 0848                              | 0.00       | 44 80 75 f0           | mov     | -ux10(%fbp),%f14d                | jacobi_base.c:113                    |                          |       |  |
| 40 084F                              | 0.00       | 48 80 50 68           | mov     | -0x18(%/DD),%/DX                 | jacobi_base.c:113                    |                          |       |  |

### Assembly-code annotation with Valgrind

What do we learn from this?

| ➡f2 43 0f 10 6c f0 08 | movsd | 0x8(%r8,%r14,8),%xmm5            |
|-----------------------|-------|----------------------------------|
| f2 43 0f 58 6c f1 08  | addsd | 0x8(%r9,%r14,8),%xmm5            |
| f2 43 0f 58 2c f2     | addsd | (%r10,%r14,8),%xmm5              |
| f2 43 0f 10 74 f2 08  | movsd | 0x8(%r10,%r14,8),%xmm6           |
| f2 43 0f 58 6c f2 10  | addsd | 0x10(%r10,%r14,8),%xmm5          |
| f2 0f 59 e9           | mulsd | %xmm1,%xmm5                      |
| f2 43 0f 10 64 f3 08  | movsd | 0x8(%rl1,%rl4,8),%xmm4           |
| f2 0f 59 e0           | mulsd | %xmm0,%xmm4                      |
| f2 0f 5c ec           | subsd | %xmm4,%xmm5                      |
| f2 42 0f 11 6c f3 08  | movsd | %xmm5,0x8(%rbx,%r14,8)           |
| 48 8b 5c c7 08        | mov   | 0x8(%rdi,%rax,8),%rbx            |
| f2 42 0f 5c 74 f3 08  | subsd | 0x8(%rbx,%r14,8),%xmm6           |
| 49 ff c6              | inc   | %r14                             |
| f2 0f 59 f6           | mulsd | %xmm6,%xmm6                      |
| 49 81 fe b8 0b 00 00  | cmp   | \$0xbb8,%r14                     |
| f2 0f 58 de           | addsd | %xmm6,%xmm3                      |
| 72 a4                 | jb    | 4007e0 <main+0x3e0></main+0x3e0> |

Use "Intel Instruction Set Reference" (or Google)

- 'movsd' is a single move of a double prec. number from memory to a SSE register (p. 3-718, vol. 2A)
- Corresponding vector instructions movpd,mulpd,addpd did not execute.
- ... The compiler failed to generate efficient SSE code. Do it yourself !?!