PAPI – Performance API

ANDRÉ PEREIRA

ampereira@di.uminho.pt

Motivation

Motivation

* Application and functions execution time is easy to measure

  * time

  * gprof

  * valgrind (callgrind)

  * …

Motivation

* Application and functions execution time is easy to measure

  * time

  * gprof

  * valgrind (callgrind)

  * …

* It is enough to identify bottlenecks, but…

  * Why is is it slow?

  * How does the code behaves?

Motivation

Motivation

* Efficient algorithms should take into account

Motivation

* Efficient algorithms should take into account

  * Cache behaviour

Motivation

* Efficient algorithms should take into account

  * Cache behaviour

  * Memory and resource contention

Motivation

* Efficient algorithms should take into account

  * Cache behaviour

  * Memory and resource contention

  * Floating point efficiency

Motivation

* Efficient algorithms should take into account

  * Cache behaviour

  * Memory and resource contention

  * Floating point efficiency

  * Branch behaviour

HW Performance Counters

HW Performance Counters

* Hardware designers added specialised registers o measure various aspects of a microprocessor

HW Performance Counters

* Hardware designers added specialised registers o measure various aspects of a microprocessor

* Generally, they provide an insight into
  * Timings
  * Cache and branch behaviour
  * Memory access patterns
  * Pipeline behaviour
  * FP performance
  * IPC
  * …

What is PAPI?

What is PAPI?

* Interface to interact with performance counters

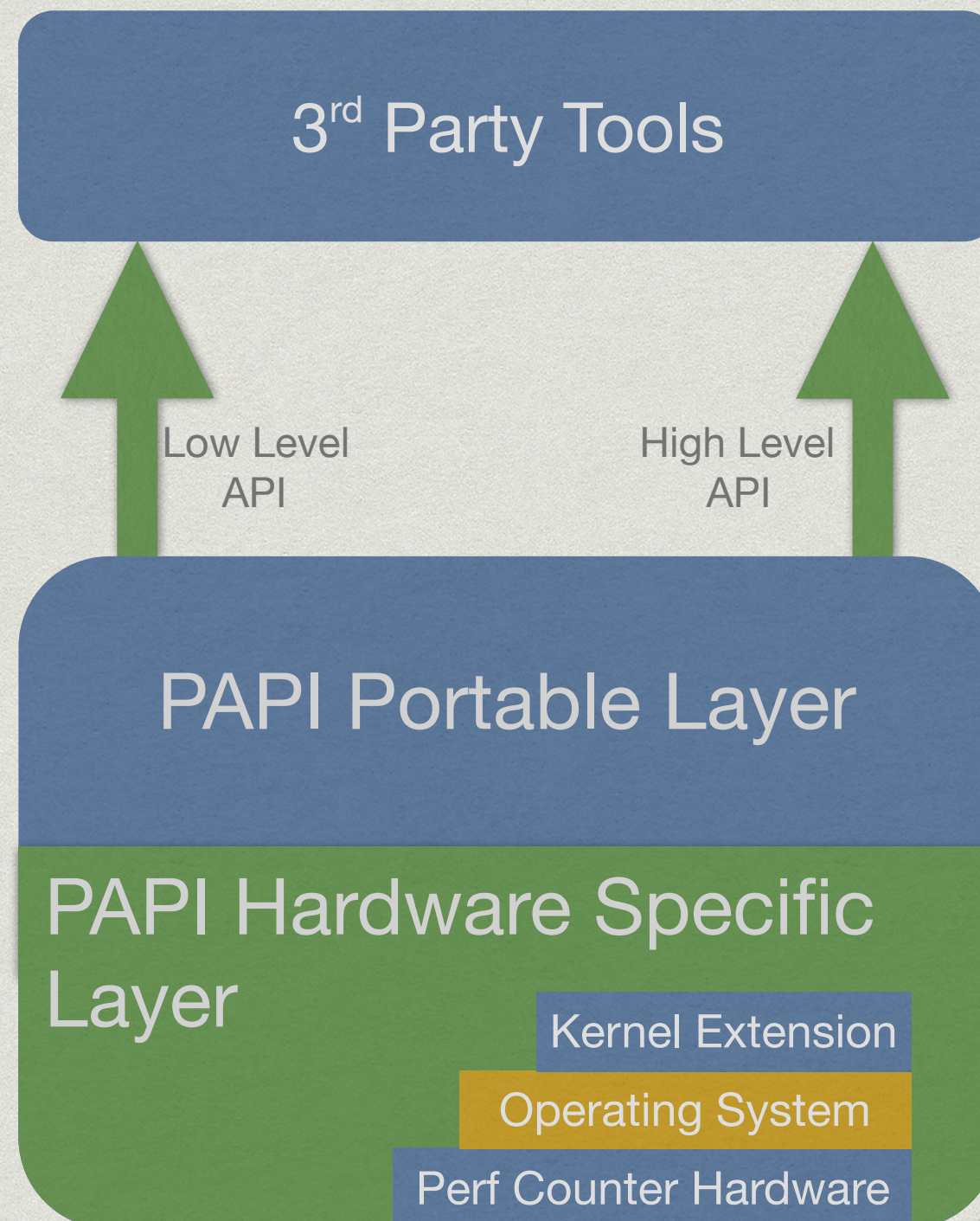   * With minimal overhead

    * Portable across several platforms

What is PAPI?

* Interface to interact with performance counters
    * With minimal overhead
    * Portable across several platforms

* Provides utility tools, C, and Fortran API
    * Platform and counters information

PAPI Organisation

Supported Platforms

* Mainstream platforms (Linux)

  * x86, x86_64 Intel and AMD

  * ARM, MIPS

  * Intel Itanium II

  * IBM PowerPC

Utilities

Utilities

* papi_avail


```
1. ampereira@compute-552-2:~/tools/papi-gcc4.9.0/bin (ssh)

-------------------------------------------------------------------------------
PAPI Version             : 5.3.2.0
Vendor string and code   : GenuineIntel (1)
Model string and code    : Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz (62)
CPU Revision             : 4.000000
CPUID Info               : Family: 6  Model: 62  Stepping: 4
CPU Max Megahertz        : 2501
CPU Min Megahertz        : 1200
Hdw Threads per core     : 2
Cores per Socket         : 10
Sockets                  : 2
NUMA Nodes               : 2
CPUs per Node            : 20
Total CPUs               : 40
Running in a VM          : no
Number Hardware Counters : 11
Max Multiplex Counters   : 32
-------------------------------------------------------------------------------


    Name       Code   Avail Deriv Description (Note)
PAPI_L1_DCM  0x80000000 Yes   No   Level 1 data cache misses
PAPI_L1_ICM  0x80000001 Yes   No   Level 1 instruction cache misses
PAPI_L2_DCM  0x80000002 Yes   Yes  Level 2 data cache misses
PAPI_L2_ICM  0x80000003 Yes   No   Level 2 instruction cache misses
PAPI_L3_DCM  0x80000004 No    No   Level 3 data cache misses
PAPI_L3_ICM  0x80000005 No    No   Level 3 instruction cache misses
PAPI_L1_TCM  0x80000006 Yes   Yes  Level 1 cache misses
PAPI_L2_TCM  0x80000007 Yes   No   Level 2 cache misses
PAPI_L3_TCM  0x80000008 Yes   No   Level 3 cache misses
PAPI_CA_SNP  0x80000009 No    No   Requests for a snoop
PAPI_CA_SHR  0x8000000a No    No   Requests for exclusive access to shared cache line
PAPI_CA_CLN  0x8000000b No    No   Requests for exclusive access to clean cache line
PAPI_CA_INV  0x8000000c No    No   Requests for cache line invalidation
PAPI_CA_ITV  0x8000000d No    No   Requests for cache line intervention
PAPI_L3_LDM  0x8000000e No    No   Level 3 load misses
PAPI_L3_STM  0x8000000f No    No   Level 3 store misses
PAPI_BRU_IDL 0x80000010 No    No   Cycles branch units are idle
PAPI_FXU_IDL 0x80000011 No    No   Cycles integer units are idle
PAPI_FPU_IDL 0x80000012 No    No   Cycles floating point units are idle
PAPI_LSU_IDL 0x80000013 No    No   Cycles load/store units are idle
PAPI_TLB_DM  0x80000014 Yes   Yes  Data translation lookaside buffer misses
PAPI_TLB_IM  0x80000015 Yes   No   Instruction translation lookaside buffer misses
PAPI_TLB_TL  0x80000016 No    No   Total translation lookaside buffer misses
PAPI_L1_LDM  0x80000017 Yes   No   Level 1 load misses
PAPI_L1_STM  0x80000018 Yes   No   Level 1 store misses
PAPI_L2_LDM  0x80000019 No    No   Level 2 load misses
PAPI_L2_STM  0x8000001a Yes   No   Level 2 store misses
PAPI_BTAC_M  0x8000001b No    No   Branch target address cache misses
PAPI_PRF_DM  0x8000001c No    No   Data prefetch cache misses
```

Utilities

* papi_avail

* papi_native_avail

Utilities

* papi_avail

* papi_native_avail

* papi_event_chooser

```
1. ampereira@compute-552-2:~/tools/papi-gcc4.9.0/bin (ssh)
[ampereira@compute-552-2 bin]$ ./papi_event_chooser PRESET PAPI_FP_OPS
Event Chooser: Available events which can be added with given events.
--------------------------------------------------------------------------
PAPI Version            : 5.3.2.0
Vendor string and code  : GenuineIntel (1)
Model string and code   : Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz (62)
CPU Revision            : 4.000000
CPUID Info              : Family: 6  Model: 62  Stepping: 4
CPU Max Megahertz       : 2501
CPU Min Megahertz       : 1200
Hdw Threads per core    : 2
Cores per Socket        : 10
Sockets                 : 2
NUMA Nodes              : 2
CPUs per Node           : 20
Total CPUs              : 40
Running in a VM         : no
Number Hardware Counters : 11
Max Multiplex Counters  : 32
--------------------------------------------------------------------------

   Name        Code      Deriv Description (Note)
PAPI_L1_DCM   0x80000000  No   Level 1 data cache misses
PAPI_L1_ICM   0x80000001  No   Level 1 instruction cache misses
PAPI_L2_ICM   0x80000003  No   Level 2 instruction cache misses
PAPI_L2_TCM   0x80000007  No   Level 2 cache misses
PAPI_L3_TCM   0x80000008  No   Level 3 cache misses
PAPI_TLB_IM   0x80000015  No   Instruction translation lookaside buffer misses
PAPI_L1_LDM   0x80000017  No   Level 1 load misses
PAPI_L1_STM   0x80000018  No   Level 1 store misses
PAPI_L2_STM   0x8000001a  No   Level 2 store misses
PAPI_STL_ICY  0x80000025  No   Cycles with no instruction issue
PAPI_BR_CN    0x8000002b  No   Conditional branch instructions
PAPI_BR_NTK   0x8000002d  No   Conditional branch instructions not taken
PAPI_BR_MSP   0x8000002e  No   Conditional branch instructions mispredicted
PAPI_TOT_INS  0x80000032  No   Instructions completed
PAPI_FP_INS   0x80000034  Yes  Floating point instructions
PAPI_LD_INS   0x80000035  No   Load instructions
PAPI_SR_INS   0x80000036  No   Store instructions
PAPI_BR_INS   0x80000037  No   Branch instructions
PAPI_TOT_CYC  0x8000003b  No   Total cycles
PAPI_L2_DCA   0x80000041  No   Level 2 data cache accesses
PAPI_L2_DCR   0x80000044  No   Level 2 data cache reads
PAPI_L3_DCR   0x80000045  No   Level 3 data cache reads
PAPI_L2_DCW   0x80000047  No   Level 2 data cache writes
PAPI_L3_DCW   0x80000048  No   Level 3 data cache writes
PAPI_L2_ICH   0x8000004a  No   Level 2 instruction cache hits
PAPI_L2_ICA   0x8000004d  No   Level 2 instruction cache accesses
PAPI_L3_ICA   0x8000004e  No   Level 3 instruction cache accesses
```

PAPI Performance Counters

PAPI Performance Counters

* Preset events
  * Events implemented on all platforms
    * PAPI_TOT_INS

PAPI Performance Counters

* Preset events

  * Events implemented on all platforms

    * PAPI_TOT_INS

* Native events

  * Platform dependent events

    * L3_CACHE_MISS

PAPI Performance Counters

* Preset events

  * Events implemented on all platforms

    * PAPI_TOT_INS

* Native events

  * Platform dependent events

    * L3_CACHE_MISS

* Derived events

  * Preset events that are derived from multiple native events

    * PAPI_L1_TCM may be L1 data misses + L1 instruction misses

PAPI High-level Interface

PAPI High-level Interface

* Calls the low-level API

PAPI High-level Interface

* Calls the low-level API

* Easier to use

PAPI High-level Interface

* Calls the low-level API

* Easier to use

* Enough for coarse grain measurements
    * You will not optimise code based on the amount of L2 TLB flushes per thread…

PAPI High-level Interface

* Calls the low-level API

* Easier to use

* Enough for coarse grain measurements
    * You will not optimise code based on the amount of L2 TLB flushes per thread…

* For preset events only!

The Basics

* PAPI_start_counters

* PAPI_stop_counters

The Basics

```
#include "papi.h"
#define NUM_EVENTS 2
long long values[NUM_EVENTS];
unsigned int Events[NUM_EVENTS]={PAPI_TOT_INS,PAPI_TOT_CYC};
 /* Start the counters */
 PAPI_start_counters((int*)Events,NUM_EVENTS);
 /* What we are monitoring… */
 do_work();
 /* Stop counters and store results in values */
 retval = PAPI_stop_counters(values,NUM_EVENTS);
```

PAPI Low-level Interface

PAPI Low-level Interface

* Increased efficiency and functionality

PAPI Low-level Interface

* Increased efficiency and functionality
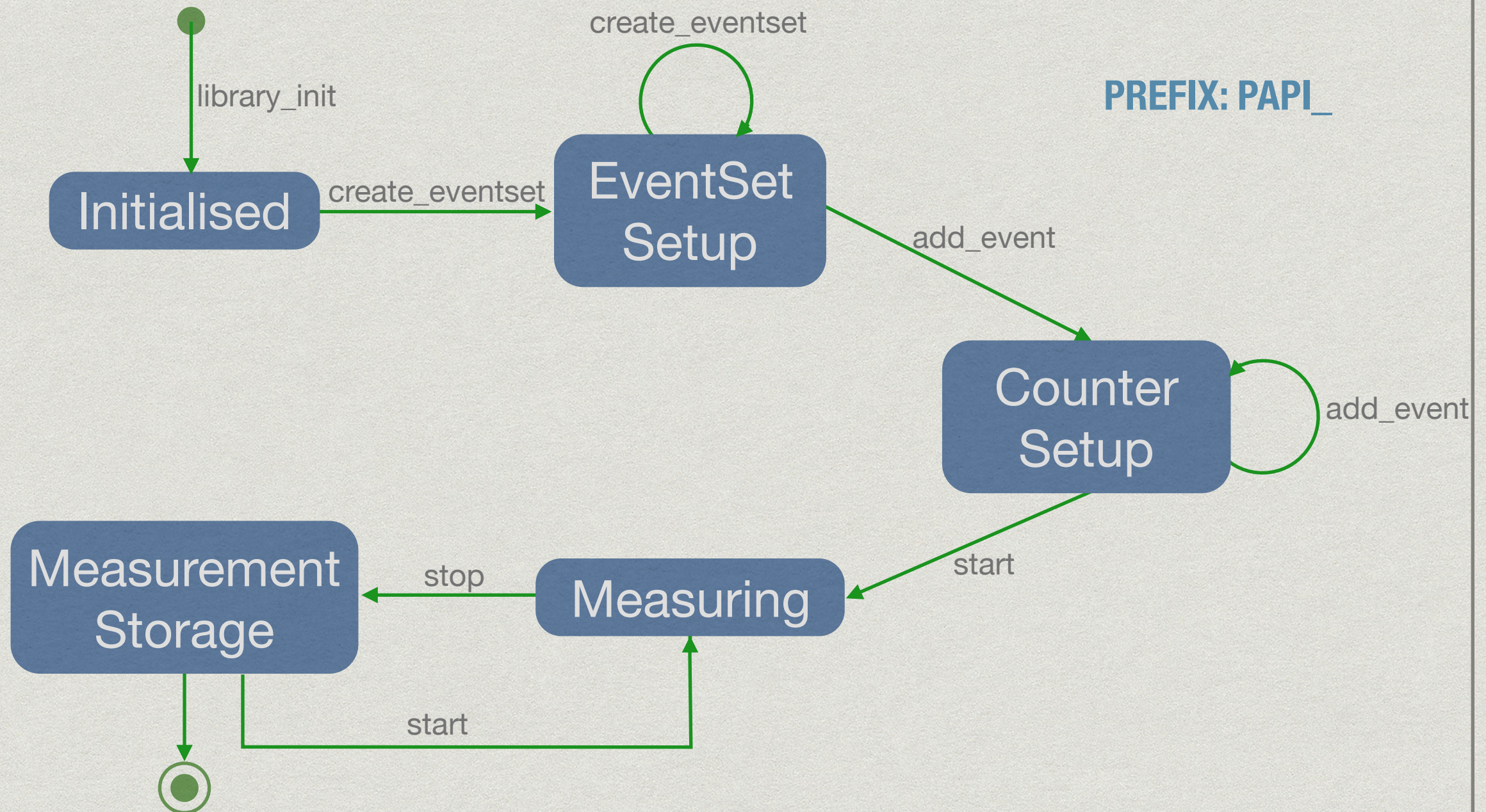
* More information about the environment

PAPI Low-level Interface

* Increased efficiency and functionality

* More information about the environment

* Concepts to check later
  * EventSet
  * Multiplexing

The Basics

The Basics

```c
#include "papi.h"
#define NUM_EVENTS 2
int Events[NUM_EVENTS]={PAPI_FP_INS,PAPI_TOT_CYC};
int EventSet;
long long values[NUM_EVENTS];
/* Initialize the Library */
retval = PAPI_library_init(PAPI_VER_CURRENT);
/* Allocate space for the new eventset and do setup */
retval = PAPI_create_eventset(&EventSet);
/* Add Flops and total cycles to the eventset */
retval = PAPI_add_events(EventSet,Events,NUM_EVENTS);
/* Start the counters */
retval = PAPI_start(EventSet);
/* What we want to monitor*/
do_work();
/*Stop counters and store results in values */
retval = PAPI_stop(EventSet,values);
```

PAPI CUDA Component

PAPI CUDA Component

* PAPI is also available for CUDA GPUs

PAPI CUDA Component

* PAPI is also available for CUDA GPUs

* Uses the CUPTI
    * Which counters can be directly accessed
    * Define a file with the counters and an environment variable

PAPI CUDA Component

* PAPI is also available for CUDA GPUs

* Uses the CUPTI
  * Which counters can be directly accessed
  * Define a file with the counters and an environment variable

* Gives useful information about the GPU usage
  * IPC
  * Memory load/stores/throughput
  * Branch divergences
  * SM(X) occupancy
  * …

What to Measure?

What to Measure?

* The whole application?

What to Measure?

* The whole application?

* PAPI usefulness is limited when used alone

What to Measure?

* The whole application?

* PAPI usefulness is limited when used alone

  * Combine it with other profilers

What to Measure?

* The whole application?

* PAPI usefulness is limited when used alone

  * Combine it with other profilers

  * Bottleneck identification + characterisation

A Practical Example

```
for (int i = 0; i < SIZE; i++)
    for (int j = 0; j < SIZE; j++)
        for (int k = 0; k < SIZE; k++)
            c[i][j] += a[i][k] * b[k][j];
```

A Practical Example

```
int sum;

for (int i = 0; i < SIZE; i++)
    for (int j = 0; j < SIZE; j++) {
        sum = 0;
        for (int k = 0; k < SIZE; k++)
            sum += a[i][k] * b[k][j];
        c[i][j] = sum;
    }
```
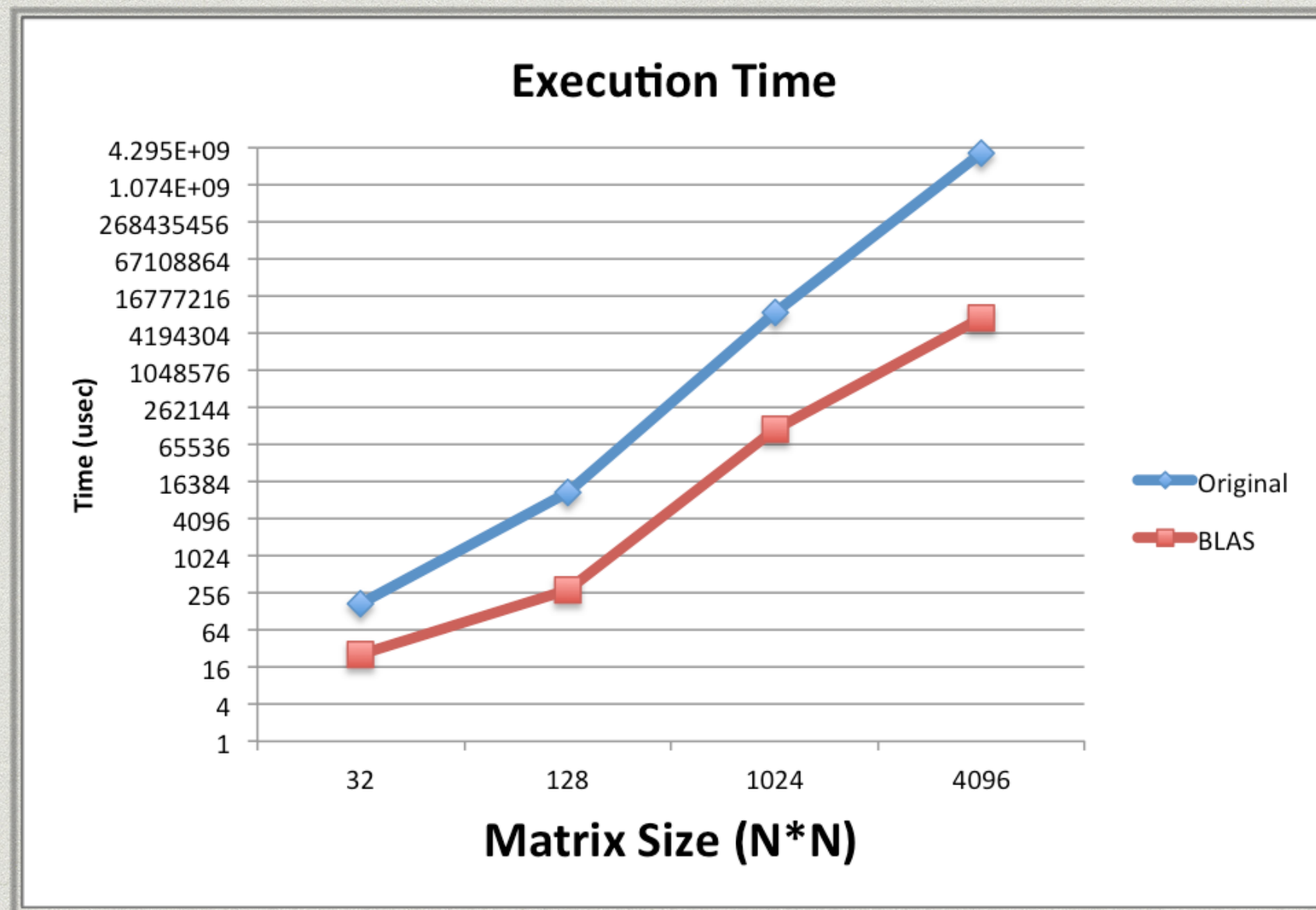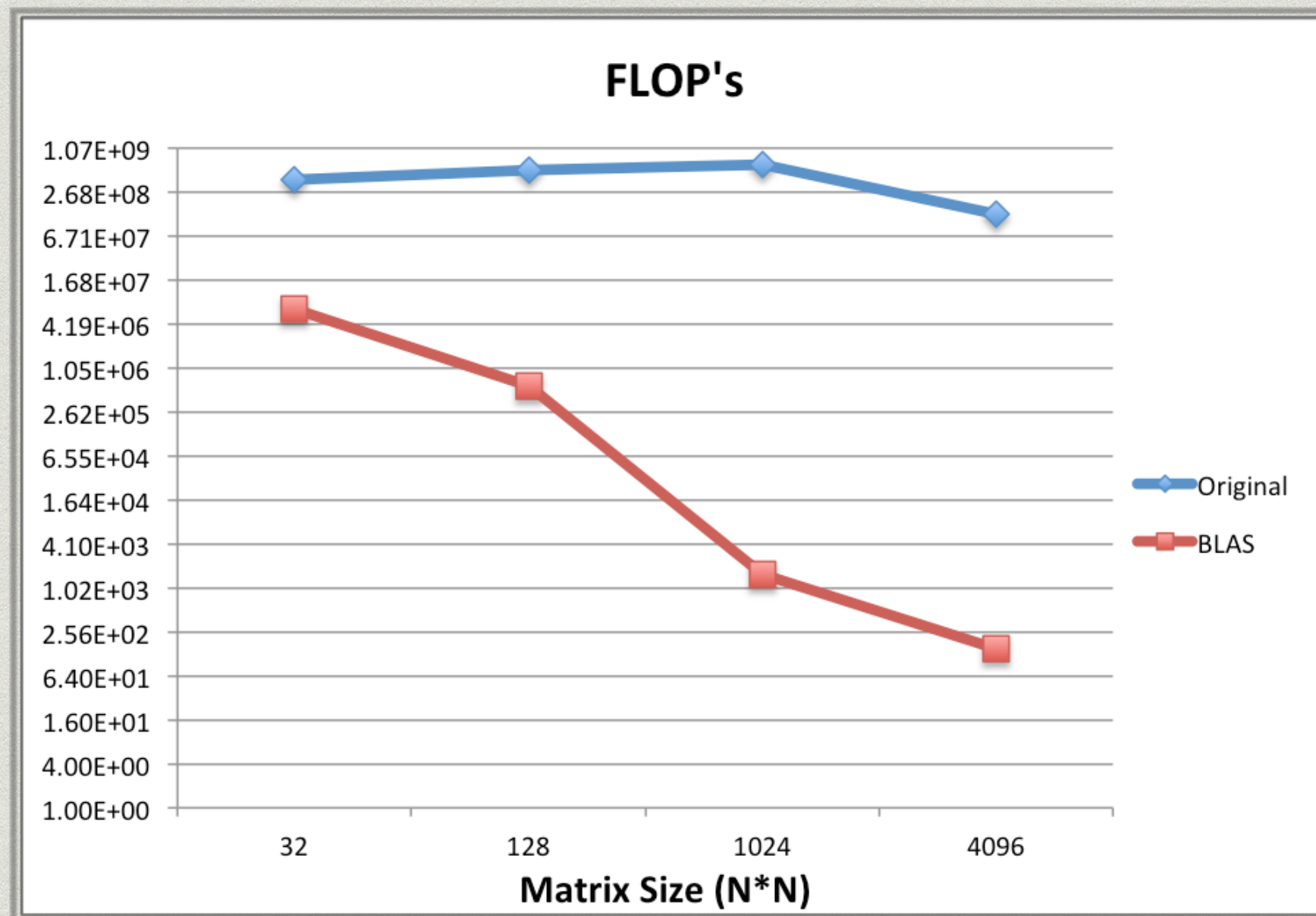
A Practical Example **SGEMM**

```
int sum;

for (int i = 0; i < SIZE; i++)
    for (int j = 0; j < SIZE; j++) {
        sum = 0;
        for (int k = 0; k < SIZE; k++)
            sum += a[i][k] * b[k][j];
        c[i][j] = sum;
    }
```

Execution Time

@ 2x Intel Xeon E5-2695v2, 12C with 24t each, 2.4GHz
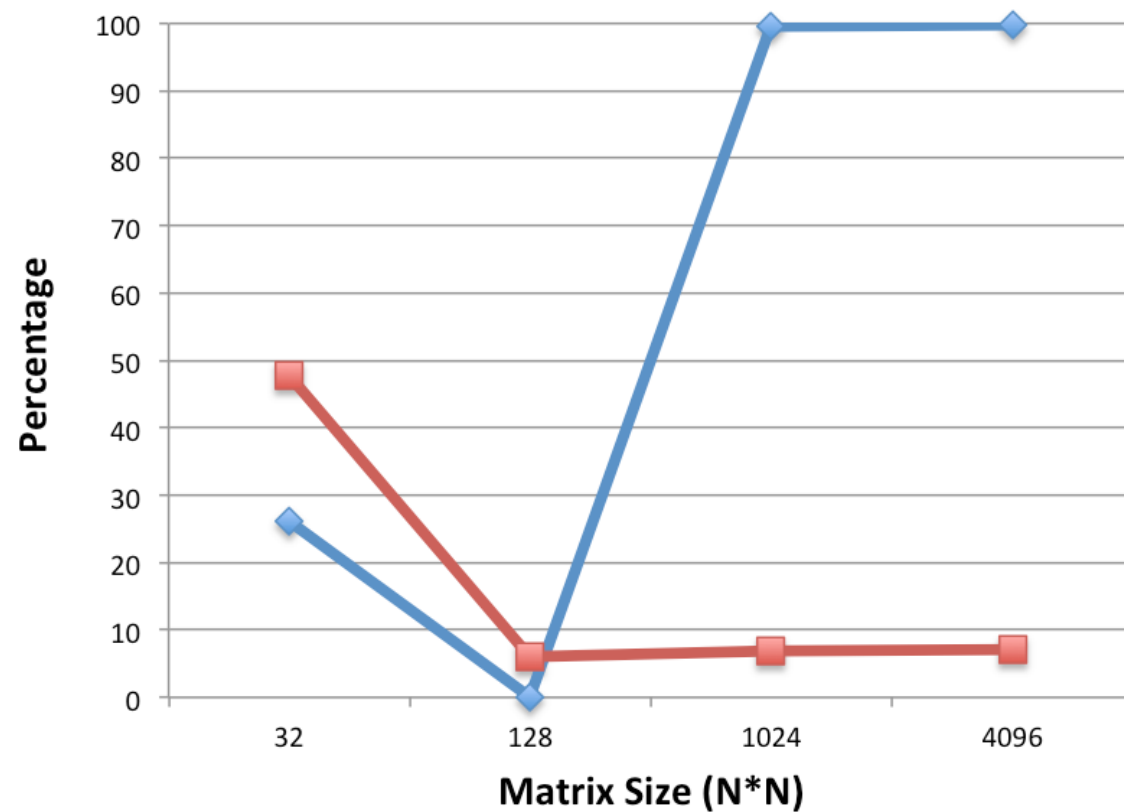
FLOP's
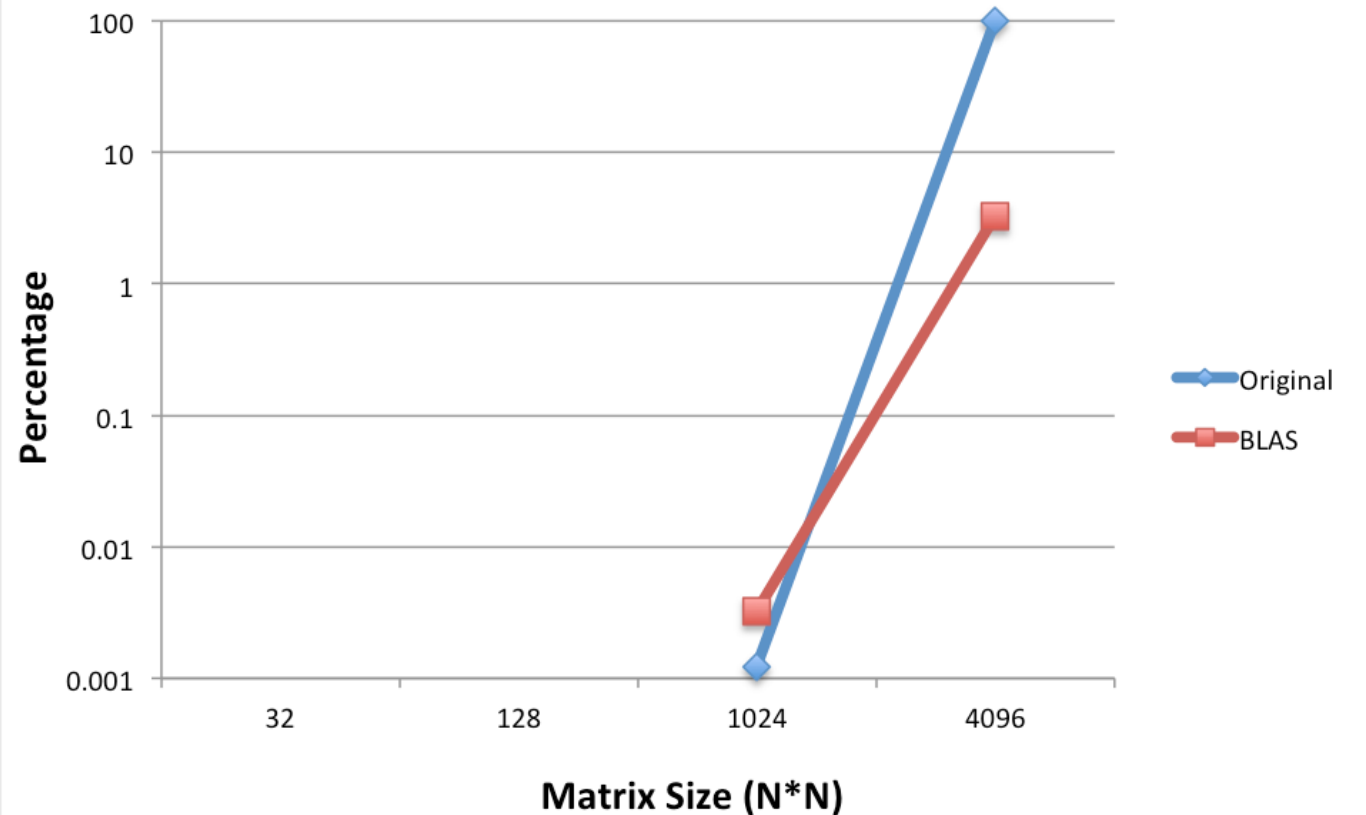
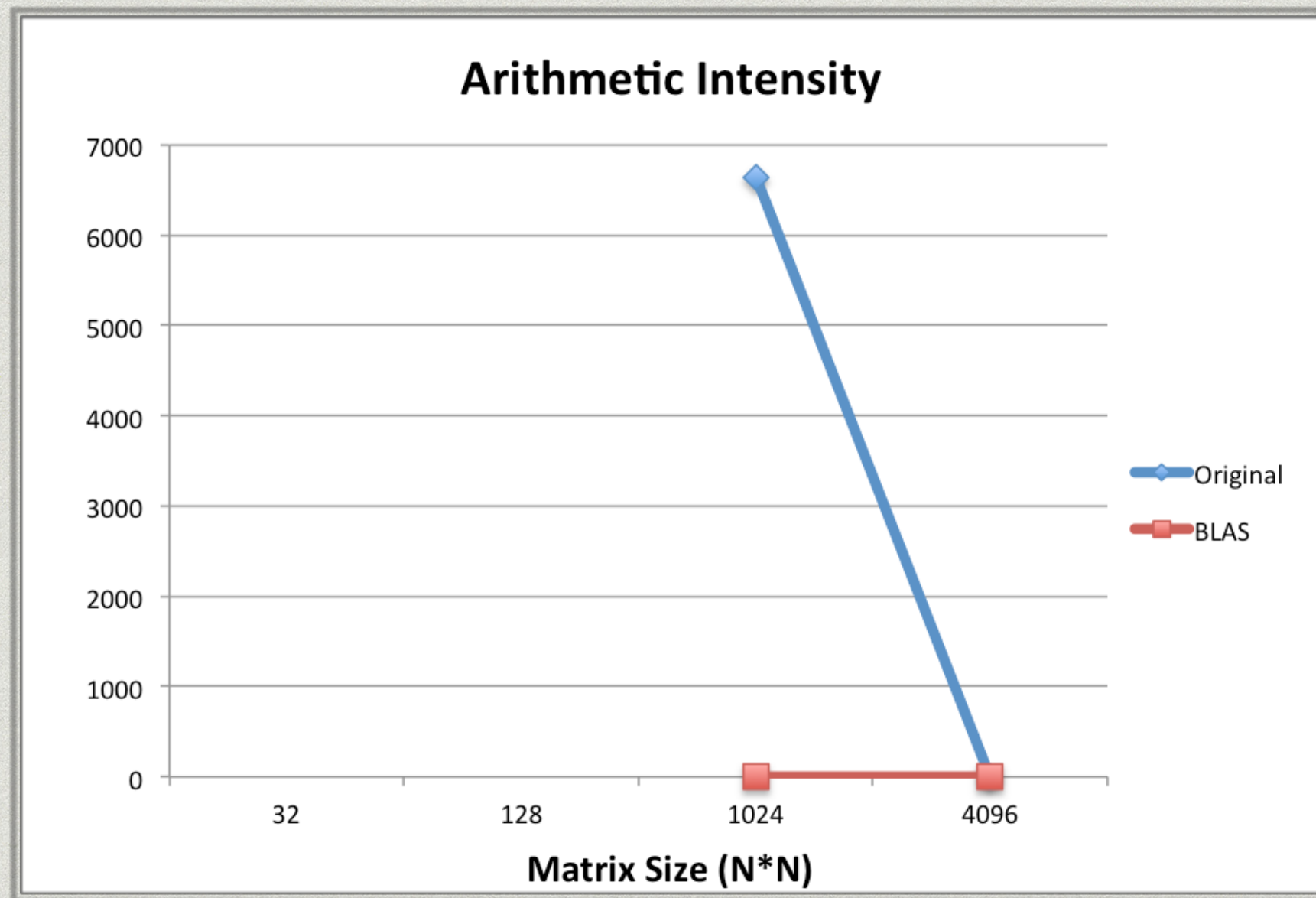# Cache Miss Rate

@ 2x Intel Xeon E5-2695v2, 12C with 24t each, 2.4GHz

Arithmetic Intensity

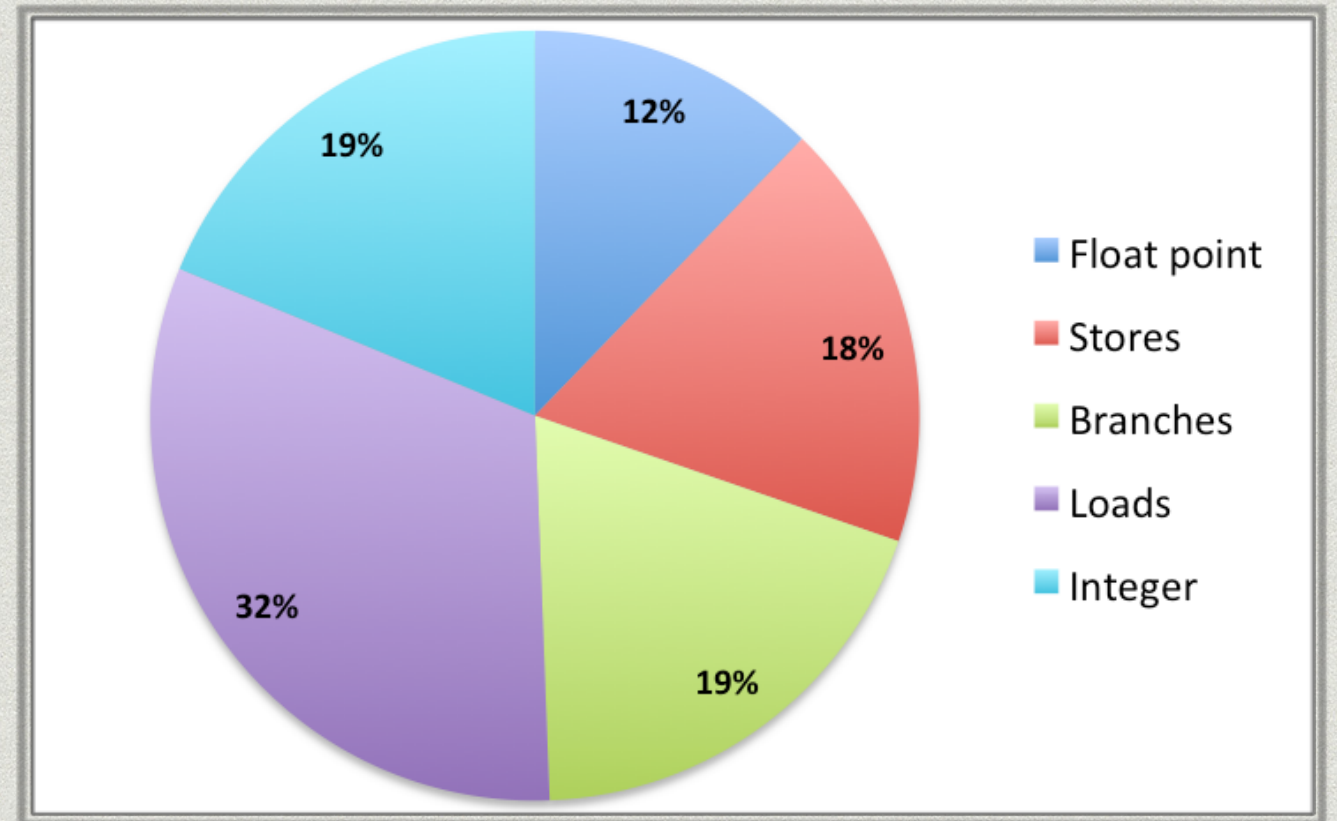*@ 2x Intel Xeon E5-2695v2, 12C with 24t each, 2.4GHz*

Useful Counters

Useful Counters

* Instruction mix
  * PAPI_FP_INS
  * PAPI_SR/LD_INS
  * PAPI_BR_INS
  * PAPI_SP/DP_VEC

Pie chart: Float point 12%, Stores 18%, Branches 19%, Loads 32%, Integer 19%

Useful Counters

* Instruction mix

  * PAPI_FP_INS

  * PAPI_SR/LD_INS

  * PAPI_BR_INS

  * PAPI_SP/DP_VEC

* FLOPS and operational intensity

  * PAPI_FP_OPS

  * PAPI_SP/DP_OPS

  * PAPI_TOT_INS

Useful Counters

* Instruction mix

  * PAPI_FP_INS

  * PAPI_SR/LD_INS

  * PAPI_BR_INS

  * PAPI_SP/DP_VEC

* FLOPS and operational intensity

  * PAPI_FP_OPS

  * PAPI_SP/DP_OPS

  * PAPI_TOT_INS

* Cache behaviour and bytes transferred

  * PAPI_L1/2/3_TCM

  * PAPI_L1_TCA

Useful Hints

Useful Hints

* Be careful choosing a measurement heuristic
  * Q: Why? Average? Median? Best measurement?

Useful Hints

* Be careful choosing a measurement heuristic
    * Q: Why? Average? Median? Best measurement?

* Automatise the measurement process
    * With scripting/C++ coding
    * Using 3rd party tools that resort to PAPI
        * PerfSuite
        * HPCToolkit
        * TAU
        * **VTune**

Useful Hints

* Be careful choosing a measurement heuristic
  * Q: Why? Average? Median? Best measurement?

* Automatise the measurement process
  * With scripting/C++ coding
  * Using 3rd party tools that resort to PAPI
    * PerfSuite
    * HPCToolkit
    * TAU
    * **VTune**

* Available for Java and on virtual machines

Compiling and Running the Code

Compiling and Running the Code

* Use the same GCC/G++ version as

    * The PAPI compilation on your home (preferably)

    * The PAPI available at the cluster

Compiling and Running the Code

* Use the same GCC/G++ version as

  * The PAPI compilation on your home (preferably)

  * The PAPI available at the cluster

* Setup the environment

  * module load gcc/5.3.0

  * module load papi/5.4.1

  * Add -I/share/apps/papi/5.4.1/include and -L/share/apps/papi/5.4.1/lib to the compilation if PAPI is not recognised

Compiling and Running the Code

* Use the same GCC/G++ version as
    * The PAPI compilation on your home (preferably)
    * The PAPI available at the cluster
* Setup the environment
    * module load gcc/5.3.0
    * module load papi/5.4.1
    * Add -I/share/apps/papi/5.4.1/include and -L/share/apps/papi/5.4.1/lib to the compilation if PAPI is not recognised

* Code compilation
    g++ -O3 c.cpp -lpapi

Hands-on

* Assess the available counters on a node (interactive qsub)

  * qsub -I -qmei -lnodes=1,walltime=10:00

* Perform the FLOPs and miss rate measurements interactively

  * https://bitbucket.org/ampereira/papi/downloads

References

＊ Dongarra, J., London, K., Moore, S., Mucci, P., Terpstra, D. "**Using PAPI for Hardware Performance Monitoring on Linux Systems**," Conference on Linux Clusters: The HPC Revolution, Linux Clusters Institute, Urbana, Illinois, June 25-27, 2001.

＊ Weaver, V., Johnson, M., Kasichayanula, K., Ralph, J., Luszczek, P., Terpstra, D., Moore, S. "**Measuring Energy and Power with PAPI**," International Workshop on Power-Aware Systems and Architectures, Pittsburgh, PA, September 10, 2012.

＊ Malony, A., Biersdorff, S., Shende, S., Jagode, H., Tomov, S., Juckeland, G., Dietrich, R., Duncan Poole, P., Lamb, C. "**Parallel Performance Measurement of Heterogeneous Parallel Systems with GPUs**," International Conference on Parallel Processing (ICPP'11), Taipei, Taiwan, September 13-16, 2011.

＊ Weaver, V., Dongarra, J. "**Can Hardware Performance Counters Produce Expected, Deterministic Results?**," 3rd Workshop on Functionality of Hardware Performance Monitoring, Atlanta, GA, December 4, 2010.