1 Roofline

1.1 Machines Profile

The machines used for this study were a MacBook Pro late 2008 and a HP dv6-2190ep from 2010. The information about the machines were gathered from */proc/cpuinfo*, */proc/meminfo*, the *Intel Ark* and *Crucial* websites, with *dmidecode* and *sysctl* linux tools and with *bandwidth* package.

1.1.1 Peaks

In order to calculate the rooflines, we needed the Floating-Point(FP) Performance Peak and the Memory Bandwidth's Peak. To attain the FP Performance Peak we calculate the following formula:

$$\text{GFlop/s}_{\max} = \#_{\text{cores}} \times f_{\text{clock}} \times \#_{\text{SIMD}}$$

MacBook Pro FP Performance Peak:

$$\text{GFlop/s}_{\max} = 2 \times 2.8 \times 8 = 44.8 GFLOPSs$$

HP Pavillion FP Performance Peak:

$$\text{GFlop/s}_{\max} = 4 \times 1.6 \times 8 = 51.2 GFLOPSs$$

To calculate de Memory Bandwidth Peak we resolve the following formula:

$$\text{BW}_{\max} = \#_{\text{channels}} \times mem_{\text{clock}} \times bus_{\text{bandwidth}}$$

MacBook Pro Memory Bandwidth Peak:

$$\text{GFlop/s}_{\max} = 2 \times 1067 \times 64 = 17.072 GBbyte$$

HP Pavillion Memory Bandwidth Peak:

$$\text{GFlop/s}_{\max} = 2 \times 1333 \times 64 = 21.328 GBbyte$$

1.1.2 Specifications

The specifications of the MacBook Pro are displayed on Table 1.

The specifications of the HP dv6-2190ep are displayed on Table 2.

1.2 Roofline Model

1.2.1 MacBook Pro Roofline

1.2.2 HP Pavillion Roofline

1.2.3 Ceilings

As suggested by the Roofline paper we added several ceilings to understand wich optimizations we may perform. This ceilings were given by recalculating the roofline without some key charateristics.

Peak floating-point performance The roofline, where all components and features are considered.

Manufacter:	Apple
Model:	MacBook Pro late 2008
Processor	
Manufacturer:	Intel
Arch:	Core
Model:	Core 2 Duo T9600
Cores:	2
Clock Frequency:	2.80 GHz
FP Performance's Peak:	44.8 GFlops/s
Cache	
Level:	1
Size:	32KB + 32KB
Line Size:	64 B
Associative:	8-way
Memory Access Bandwidth:	40 GB/s
Level:	2
Size:	6 MB
Line Size:	64 B
Associative:	24-way
RAM	
Type:	SDRAM DDR3 PC3-8500
Frequency:	1067 MHz
Size:	4 GB
Num. Channels:	2
Latency:	13.13 ns

Tabela 1: MacBook Pro late 2008 specifications

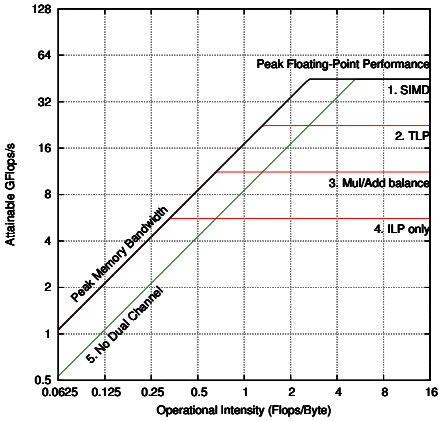

Figura 1: Mackbook Pro late 2008

Manufacter:	HP
Model:	Pavillion dv6-2190ep
Processor	
Manufacturer:	Intel
Arch:	Nehalem
Model:	i7-720QM
Cores:	4
Clock Frequency:	1.60 GHz
FP Performance's Peak:	51.2 GFlops/s
Cache	
Level:	1
Size:	32KB + 32KB
Line Size:	64 B
Associative:	4/8-way
Memory Access Bandwidth:	22 GB/s
Level:	2
Size:	256 KB
Line Size:	64 B
Associative:	8-way
Level:	3
Size:	6 MB
Line Size:	64 B
Associative:	12-way
RAM	
Type:	SDRAM DDR3 PC3-10600
Frequency:	1333 MHz
Size:	4GB
Num. Channels:	2
Latency:	13.5 ns

Tabela 2: HP Pavillion dv6-2190ep specifications

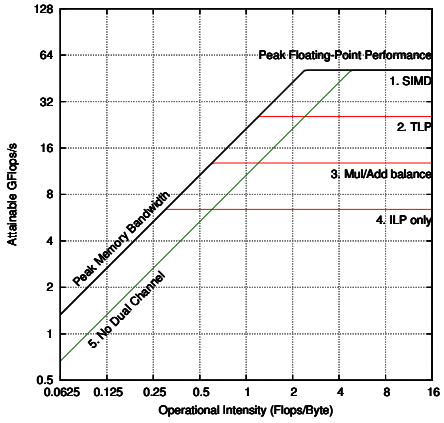

Figura 2: HP Pavillion dv6-2190ep Roofline

For memory only one ceiling was calculated, besides the roofline.

Peak stream bandwidth The roofline, where all features are considered.

One-channel

2 PAPI Case Study

2.1 Problem

The case study of this report, is to analyse the performance of a **matrix multiplication** algorithm,

$$MatrixA * MatrixB = MatrixC \tag{1}$$

wich contains a triple nested loop with the indexes i,j and k(line,column and position). Our implementation will explore the index order **i,j,k** of the triple nested loop.

2.2 Algorithm Analysis

The implementation produced to calculate the matrix multiplication was made in C and compiled with Optimization level 1 (-o). The algorithm of matrix multiplication is presented here, in order to better understand the problem at hand.

```
for (i = 0; i < size; i++) {
    for (j = 0; j < size; j++) {
        for(k = 0; k < size; k++) {
            acc += matrixA[i][k] * matrixB[k][j];
            }
            matrixC[i][j] = acc;
            acc = 0;
        }
    }
```

Two versions of the program were run, one with the matrixB and other with the transpose matrixB. With this second version, it is expected that the positions used in the algorithm will be contiguous, reducing the number of accesses to memory.

2.3 Counters Used

To measure the algorithm's perfomance, hardware counters were used. To gather the information of these counters we used *PAPI*(Performance API). This tool alowed us to measure the following counters:

PAPI _TOT_CYC Total number of cycles;

PAPI _TOT_INS Instructions completed;

PAPI _LD_INS number of load instructions;

PAPI _SR_INS number of store instructions;

PAPI _FP_OPS Floating point operations;

PAPI _FP_INS Floating point instructions;

PAPI _L1_DCA L1 data cache accesses;

PAPI _L1_DCM L1 data cache misses;

PAPI _L2_DCA L2 data cache accesses;

PAPI _L2_DCM L2 data cache misses;

PAPI _L3_DCA L3 data cache accesses;

2.4 Tests

The four tests, presented below, were chosen to run in the two different version(normal and transpose). Each test was run four times, with the best execution time being select as long as the range was no larger than the other three. Each test fits in a different memory level(L1, L2, L3 and RAM).

Memory	Size	Matrix Size
L1	30 KB	50
L2	255 KB	146
L3	3 MB	500
RAM	7.68 MB	800

Tabela 3: Test cases

2.5 Results

2.5.1 Analysis Chache

To measure the data cache misses the counters PAPI_L1_DCM and PAPI_L2_DCM were used.

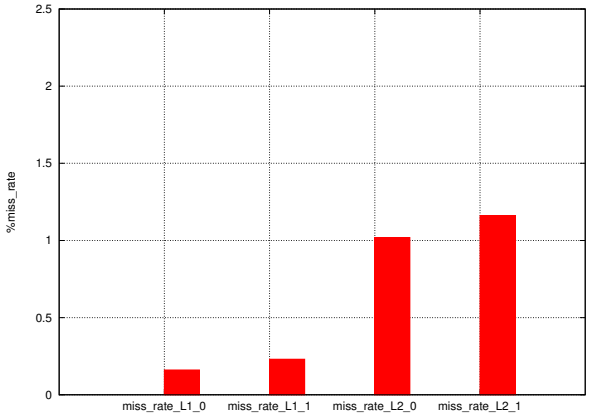

Figura 3: Percentage of Cache Misses

The above graphic shows an increase of percentage of misses with the optimized version, they are misleading. The next graph shows that although the percentage of misses increased, the total of misses didn't because the number of accesses also droped.

Usage of both levels of cache was estimated with specific counters. PAPI_L1_DCA and PAPI_L2_DCA provided the number of data accesses to the caches.

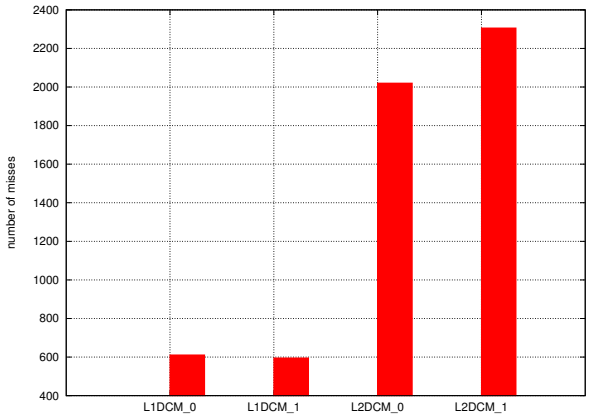

Figura 4: dsa

Before the results were out, it was expected a decrease of cache accesses from version one to version two of the algorithm.

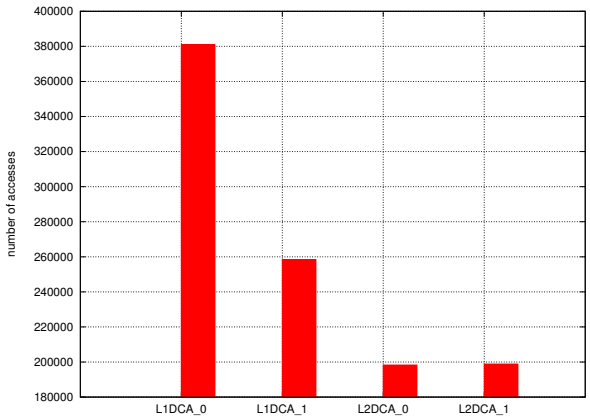

Figura 5: dsa

As we can see, the number of access to cache drops significantly from the first version to the second version while running with the L1 Cache Test. Though in the second test, the L2 Cache, the number of accesses slightly increased.

Referências

[1] `Roofline: An insightful Visual Performance Model for Floating-Point Programs and Multicore Architectures`
Samuel Webb Williams, Andre Waterman, David A. Patterson
23th November 2012

[2] `http://ark.intel.com/products/35563/Intel-Core2-Duo-Processor-T9600-6M-Cache-2_80-GHz-1066-MHz-FSB`
Intel® Core ™ 2 Duo Processor T9600 (6M Cache, 2.80 GHz, 1066 MHz FSB)
©Intel Corporation
23th November 2012

[3] `http://ark.intel.com/products/43122/Intel-Core-i7-720QM-Processor-6M-Cache-1_60-GHz`
Intel® Core ™ i7-720QM Processor(6M Cache, 1.60 GHz)

©Intel Corporation
23th November 2012

[4] http://www.crucial.com/store/ListParts.aspx?model=MacBook%20Pro%20%28Early%202008%20and%20Late%202008%29
Computer memory upgrades for Apple MacBook Pro (Early 2008 and Late 2008) Laptop/Notebook from Crucial.com
© Micron Technology
24th November 2012

[5] http://www.crucial.com/store/listparts.aspx?model=Pavilion%20dv6-2190ep&Cat=RAM
Computer memory upgrades for HP - Compaq Pavilion dv6-2190ep Laptop/Notebook from Crucial.com
© Micron Technology
24th November 2012