UNIVERSIDAD
DE GRANADA

Estructura de Computadores
Grado en Ingeniería Informática
Curso 2018-2019

Memoria Práctica 6.
Caché.

Félix Ramírez García
felixramirezgarcia@correo.ugr.es

18 de diciembre de 2018

Índice

Índice de figuras

Índice de tablas

1. Códigos de los programas

1.1. Programa line.cc

```cpp
#include <algorithm>     // nth_element
#include <array>         // array
#include <chrono>        // high_resolution_clock
#include <iomanip>       // setw
#include <iostream>      // cout
#include <vector>        // vector

using namespace std::chrono;

const unsigned MAXLINE = 1024; // maximun line size to test
const unsigned GAP = 12;       // gap for cout columns
const unsigned REP = 100;      // number of repetitions of every test

int main(){
        std::cout << "#"
        << std::setw(GAP - 1) << "line (B)"
        << std::setw(GAP    ) << "time (nanosegundos)"
        << std::endl;

        for (unsigned line = 1; line <= MAXLINE; line <<= 1){
                std::vector<duration<double, std::micro>> score(REP);

                for (auto &s: score){
                        std::vector<char> bytes(1 << 24); // 16MB

                        auto start = high_resolution_clock::now();

                        for (unsigned i = 0; i < bytes.size(); i += line)
                                bytes[i] ^= 1;

                        auto stop = high_resolution_clock::now();

                        s = stop - start;
                }

                std::nth_element(score.begin(),
                score.begin() + score.size() / 2,
                score.end());

                std::cout << std::setw(GAP) << line
                << std::setw(GAP) << std::fixed << std::setprecision(1)
                << std::setw(GAP) << score[score.size() / 2].count()
                << std::endl;
        }
}
```

1.2. Programa size.cc

```cpp
#include <algorithm>     // nth_element
#include <array>         // array
#include <chrono>        // high_resolution_clock
#include <iomanip>       // setw
#include <iostream>      // cout
#include <vector>        // vector

using namespace std::chrono;

const unsigned MINSIZE = 1 << 10; // minimun line size to test:  1KB
const unsigned MAXSIZE = 1 << 26; // maximun line size to test: 32MB
const unsigned GAP = 12;          // gap for cout columns
const unsigned REP = 100;         // number of repetitions of every test
const unsigned STEPS = 1e6;       // steps

int main() {
        std::cout << "#"
        << std::setw(GAP - 1) << "line (B)"
        << std::setw(GAP    ) << "time (nanosegundos)"
        << std::endl;

        for (unsigned size = MINSIZE; size <= MAXSIZE; size *= 2){
                std::vector<duration<double, std::micro>> score(REP);

                for (auto &s: score){
                        std::vector<char> bytes(size);
                        unsigned tamanio = size -1;
                        auto start = high_resolution_clock::now();

                        for (unsigned i = 0; i < STEPS; ++i)
                                bytes[(i*64)&(size-1)] ^= 1;

                        auto stop = high_resolution_clock::now();

                        s = stop - start;
                }

                std::nth_element(score.begin(),
                score.begin() + score.size() / 2,
                score.end());

                std::cout << std::setw(GAP) << size
                << std::setw(GAP) << std::fixed << std::setprecision(1)
                << std::setw(GAP) << score[score.size() / 2].count()
                << std::endl;
        }
}
```

2. Gráficas

A continuación se muestran las gráficas generadas con los programas anteriores.

La figura 2.1 muestra que cuanto mas grande es el tamaño, menos tiempo tarda en ejecutarse. Tiene un punto de inflexión en 16 B, donde cambia el sentido de la función.

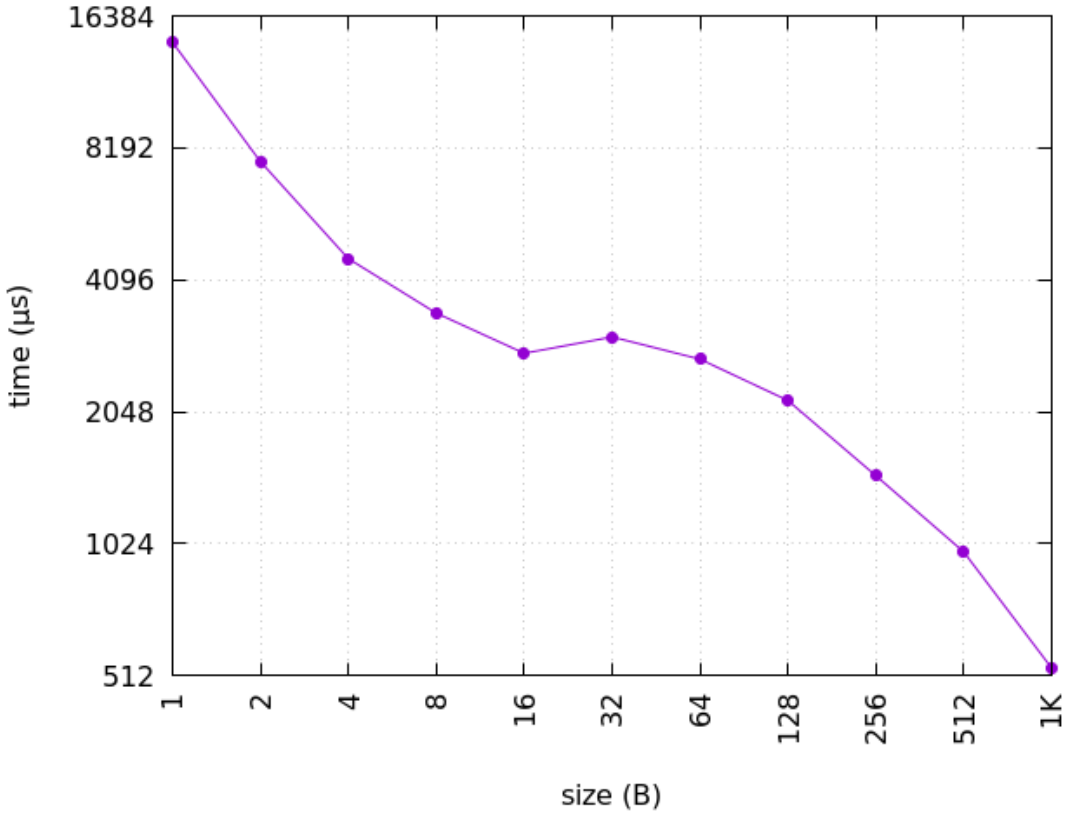

Figura 2.1: Grafica obtenida por la ejecucion de line.cc.

En la figura 2.2 se puede apreciar que se mantiene un tiempo constante hasta llegar a un tamaño de 32K, y a partir de hay va incrementando gradualmente hasta 8 M donde parece que se vuelve a mantener constante. Cabe destacar que entre 64K y 256K no hay un incremento significativo. El mayor incremento de tiempo se produce entre 2 y 4 M.

3. Caracteristicas de nuestra CPU

Las características de nuestra CPU son las obtenidas con el comando lscpu. Son las siguientes:

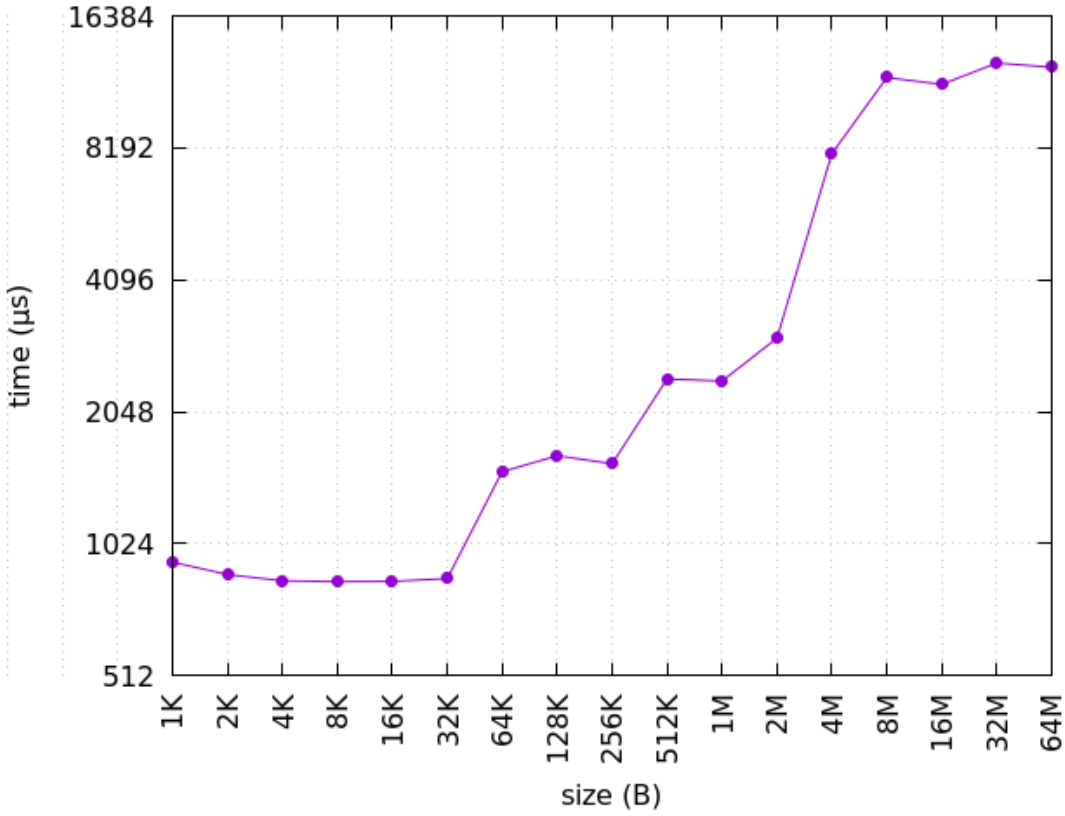

Figura 2.2: Grafica obtenida por la ejecucion de size.cc.

```
Arquitectura:                         x86_64
modo(s) de operacion de las CPUs:     32-bit, 64-bit
Orden de los bytes:                   Little Endian
CPU(s):                               1
Lista de la(s) CPU(s) en linea:       0
Hilo(s) de procesamiento por nucleo:  1
Nucleo(s) por socket:                 1
Socket(s)                             1
Modo(s) NUMA:                         1
ID de fabricante:                     GenuineIntel
Familia de CPU:                       6
Modelo:                               142
Nombre del modelo:                    Intel(R) Core(TM) i3-7100U CPU 2.40
    GHz
Revision:                             9
CPU MHz:                              2399.998
BogoMIPS:                             4799.99
Fabricante del hipervisor:            KVM
Tipo de virtualizacion:               lleno
```

```
Cache L1d:                           32K
Cache L1i:                           32K
Cache L2:                            256K
Cache L3:                            3072K
CPU(s) del nodo NUMA 0:              0
Indicadores:                         fpu vme de pse tsc msr pae mce cx8
                                   apic sep mtrr pge mca cmov pat pse36
                                   clflush mmx fxsr sse sse2 syscall nx
                                   rdtscp lm constant_tsc rep_good nopl
                                   xtopology nonstop_tsc cpuid pni
                                   pclmulqdq monitor ssse3 cx16 pcid
                                   sse4_1 sse4_2 x2apic movbe popcnt aes
                                   xsave avx rdrand hypervisor lahf_lm
                                   3dnowprefetch invpcid_single pti
                                       fsgsbase
                                   avx2 invpcid rdseed clflushopt abm
```