

Estructura de Computadores (2016-2017)
GRADO EN INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE GRANADA

Memoria Práctica 5 - CACHÉ

Álvaro Fernández-Alonso Araluce

21 de enero de 2017

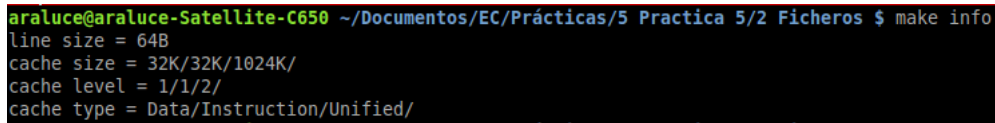
Índice

1. Análisis make info	3
2. Tamaño de línea	3
2.1. Análisis del código	3
2.2. Análisis de la gráfica	4
3. Size	5
3.1. Código	5
3.2. Análisis de la gráfica	6

Índice de figuras

1.1. Resultado de la ejecución <i>make info</i>	3
2.1. Gráfica de line.cc	4
3.1. Gráfica de size.cc	6

1 Análisis make info



```
araluce@araluce-Satellite-C650 ~/Documentos/EC/Prácticas/5 Practica 5/2 Ficheros $ make info
line size = 64B
cache size = 32K/32K/1024K/
cache level = 1/1/2/
cache type = Data/Instruction/Unified/
```

Figura 1.1: Resultado de la ejecución *make info*

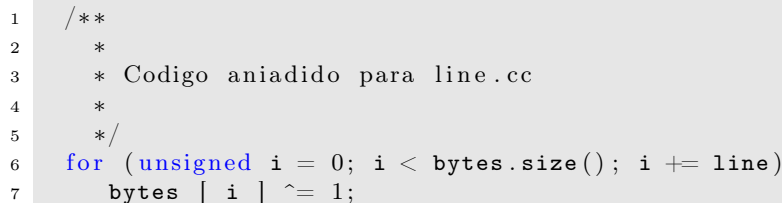
Cuando ejecutamos *make info* sobre nuestro ordenador obtenemos los datos en texto de las representaciones gráficas que estudiaremos a continuación. Destacamos que el valor del tamaño de línea o bloque de caché de nuestro ordenador es de 64B. Es un dato a tener en cuenta en los siguientes análisis.

Por otro lado se nos muestran: tamaño; nivel y tipo de memoria caché respectivamente.

- Nivel 1:
 - 32K de datos.
 - 32k de instrucciones.
- Nivel 2: 1024K.

2 Tamaño de línea

Análisis del código



```
1  /**
2   *
3   * Código añadido para line.cc
4   *
5   */
6  for (unsigned i = 0; i < bytes.size(); i += line)
7      bytes [ i ] ^= 1;
```

Este es el segmento de código añadido al esqueleto facilitado en la clase práctica. Consta de un bucle que recorre todos los elementos realizando una pequeña modificación en ellos de forma que tampoco represente un gasto en eficiencia. Estas dos sencillas líneas de código nos servirán para estudiar nuestro tamaño de línea a través de la gráfica que a continuación mostramos.

Análisis de la gráfica

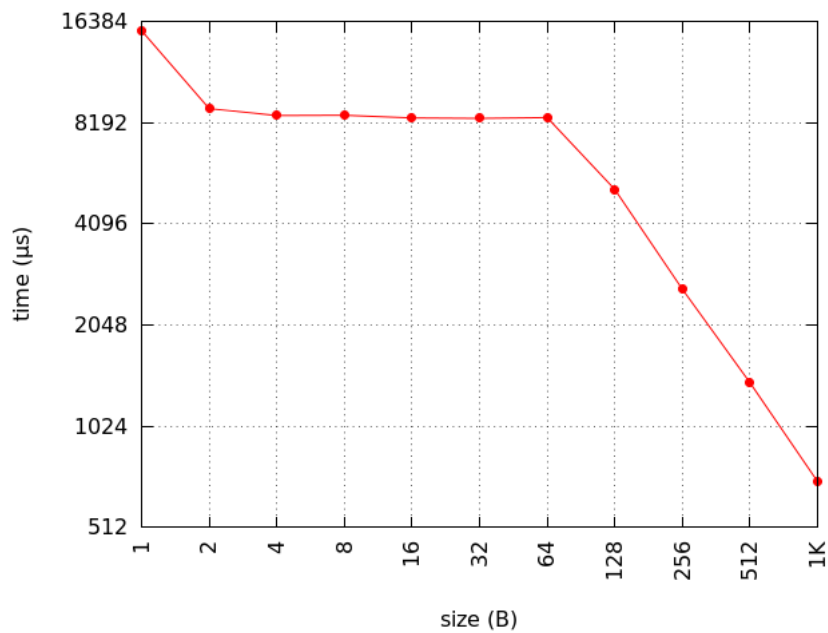


Figura 2.1: Gráfica de line.cc

En esta gráfica nos vamos a fijar en cómo descende el tiempo de acceso caché (eje Y) en función del incremento de línea (eje X).

Inicialmente se dispara el tiempo de acceso a la memoria caché ya que se realizan una gran cantidad de operaciones de traslado de datos a memoria caché. Conforme va aumentando el tamaño de línea, el programa tiene que realizar menos operaciones de traslado de datos a memoria caché ya que esos datos están cargados ya en memoria caché. Por ese motivo observamos esa descenso pronunciado en el tiempo de acceso de nuestro programa. Finalmente, para averiguar el tamaño de nuestra memoria de línea observamos en la gráfica dónde se representa el pico de descenso de tiempo. En la nuestra podemos ver como a partir de 64B de tamaño, el tiempo de acceso es la mitad en cada acceso que se produce. De esta forma corroboramos el dato ofrecido en la ejecución de la instrucción *make info*, nuestro tamaño de línea es de 64B.

3 Size

Código

```
1  /**
2   *
3   * Codigo aniadido para size.cc
4   *
5   */
6  for (unsigned i = 0; i < STEPS; ++i)
7      bytes [(i*64) % size]++;
```

Este es el segmento de código añadido al esqueleto facilitado en la clase práctica. El código dentro del bucle realiza una operación de módulo que podría modificarse provocando una mayor rapidez en la generación de la gráfica con los mismos resultados.

```
1  /**
2   *
3   * Codigo final para size.cc
4   *
5   */
6  for (unsigned i = 0; i < STEPS; ++i)
7      bytes [(i*64) & (size - 1)]++;
```

A continuación estudiaremos los datos arrojados por la gráfica.

Análisis de la gráfica

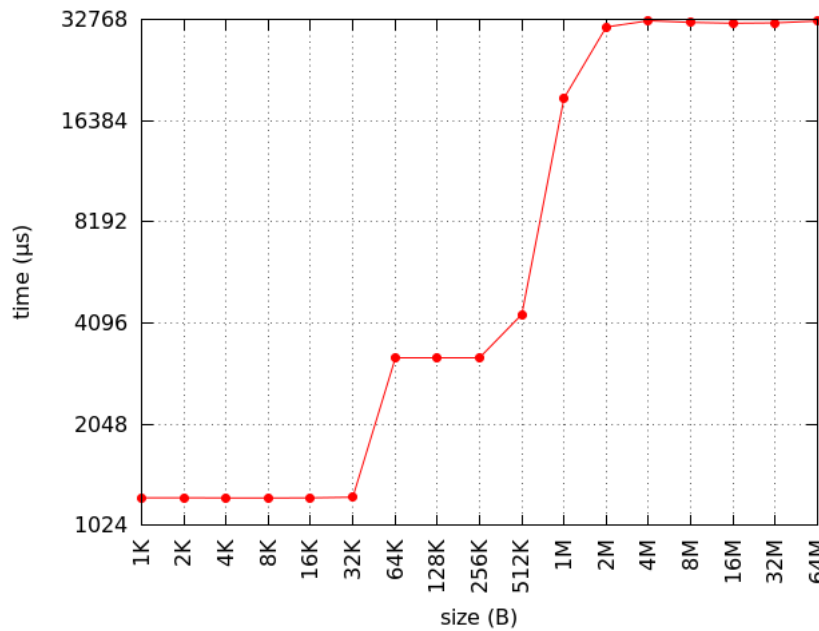


Figura 3.1: Gráfica de size.cc

Como podemos observar en el primer bloque de memoria caché (L1), el tiempo de acceso es muy bajo con respecto al resto de bloques. Para obtener el tamaño de bloque de L1, observamos dónde se produce un salto al siguiente bloque. Este salto se produce entre los 32K y 64K, comprobamos que, efectivamente, su tamaño es de 32K ejecutando el comando *make info*.

Make info nos dice que el siguiente nivel, el L2, tiene un tamaño de 1024K. Si nos fijamos bien, la gráfica nos muestra que el siguiente salto se produce entre los tamaños 1M y 2M (como decía la salida *make*).