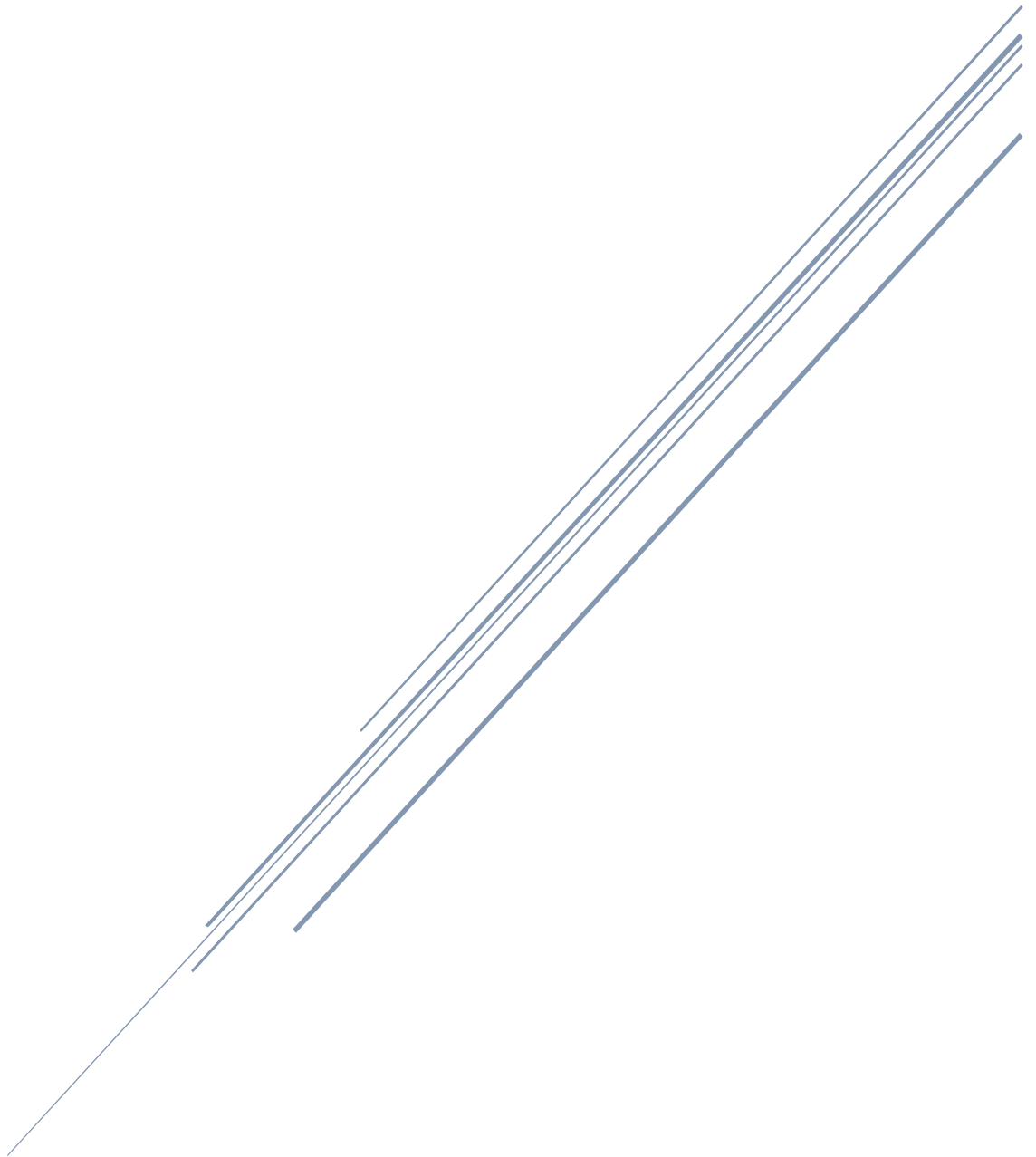


EVALUACIÓN DE CONFIGURACIONES

Práctica 2

—

Perfilado con código fuente



FDI - UCM

Iván Aguilera Calle – Daniel García Moreno

1. gcc

En este apartado ejecutaremos el comando `time` en distintas ocasiones para obtener los tiempos de ejecución del programa “edges”, probando para ello distintos parámetros de compilación para la optimización. También generaremos el perfil de ejecución y obtendremos los distintos tiempos de ejecución usando el perfil creado.

Compilación	Tiempo de ejecución (./edges img.pgm out.pgm)
-O0	4.58
-O1	3.39
-O2	3.41
-O3	2.59

Podemos observar que entre los niveles de optimización O1 Y O2 no hay gran diferencia, pero sí que hay una gran diferencia de tiempos entre los niveles O0 Y O3 (casi la mitad de tiempo).

Compilación – perfil de ejecución	Tiempo de ejecución (./edges img.pgm out.pgm)
-O0 (Sin optimización)	4.83
-O1	2.87
-O2	2.81
-O3	1.25

Para la obtención de los tiempos de la segunda tabla hemos ejecutado los siguientes comandos:

- `gcc -fprofile-generate -O0 -o edges edges.c`
- `./edges img.pgm out.pgm` (se genera el fichero `edges.gcda` con la información del perfil de ejecución).
- Ahora compilamos usando la opción `-fprofile-use` para los distintos niveles de optimización):
 - `gcc -fprofile-use -OX -o edges edges.c`
 - `time ./edges img.pgm out.pgm`

En la segunda tabla podemos observar una mejora bastante apreciable en el tiempo de ejecución al indicar al compilador que utilice el perfil de ejecución.

2. gprof

En este apartado utilizaremos la herramienta gprof para hacer un análisis de tiempos de las diferentes funciones del programa “edges.c”. Ejecutaremos para ello las siguientes órdenes:

- `gcc -O0 -pg edges.c -o edges_gprof`
- `./edges_gprof img.pgm out.pgm` (nos genera el archivo gmon.out)
- `gprof edges-gprof gmon.out > analisis.txt` (fichero con los resultados)

Preguntas:

- Tiempo consumido por cada función de “edges.c” compilando sin optimización:
 - PONER TIEMPOOOOOOOOS
- ¿Qué función intentarías mejorar primero?
 - La que más tiempo tarda (mirando en la columna self seconds). En este caso es la función gaussian (2.46s).
- ¿Cuánto tardaría en ejecutarse el programa si consiguieras mejorar esa función en un 15%?
 - AMDAHHHHL
- ¿Cuál sería la máxima mejora que podrías obtener mejorando solamente esa función?
 - PREGUNTAaaaaaaar

Flat profile:

ANALISIS.TXT

Each sample counts as 0.01 seconds.

% time	cumulative seconds	self seconds	calls	self s/call	total s/call	name
65.86	2.46	2.46	2	1.23	1.23	gaussian
32.53	3.67	1.21	1	1.21	1.21	laplacian
1.08	3.71	0.04	1	0.04	0.04	save_image_file
0.81	3.74	0.03	1	0.03	0.03	load_image_file
0.00	3.74	0.00	1	0.00	3.67	edges

Call graph (explanation follows)

granularity: each sample hit covers 2 byte(s) for 0.27% of 3.74 seconds

index	% time	self	children	called	name
					<spontaneous>
[1]	100.0	0.00	3.74		main [1]
		0.00	3.67	1/1	edges [2]
		0.04	0.00	1/1	save_image_file [5]
		0.03	0.00	1/1	load_image_file [6]

		0.00	3.67	1/1	main [1]
[2]	98.1	0.00	3.67	1	edges [2]
		2.46	0.00	2/2	gaussian [3]
		1.21	0.00	1/1	laplacian [4]

		2.46	0.00	2/2	edges [2]
[3]	65.7	2.46	0.00	2	gaussian [3]

		1.21	0.00	1/1	edges [2]
[4]	32.4	1.21	0.00	1	laplacian [4]

		0.04	0.00	1/1	main [1]
[5]	1.1	0.04	0.00	1	save_image_file [5]

		0.03	0.00	1/1	main [1]
[6]	0.8	0.03	0.00	1	load_image_file [6]

3. Google-pprof

En este último apartado utilizaremos la herramienta Google-pprof, para que al igual que en el apartado anterior, podamos obtener los distintos tiempos que tiene cada función del programa “edges.c”:

```
usuario@debian:~/Escritorio/p2$ gcc -o edges edges.c -lprofiler
usuario@debian:~/Escritorio/p2$ CUPROFILE=/tmp/edges.prof ./edges img.pgm out.pgm
PROFILE: interrupts/evictions/bytes = 369/266/20544
usuario@debian:~/Escritorio/p2$ google-pprof --text edges /tmp/edges.prof
Using local file edges.
Using local file /tmp/edges.prof.
Removing killpg from all stack traces.
Total: 369 samples
 236  64.0%  64.0%   236  64.0% gaussian
 114  30.9%  94.9%   114  30.9% laplacian
   6   1.6%  96.5%    6   1.6% fputc
   4   1.1%  97.6%    4   1.1% _IO_getc
   3   0.8%  98.4%    3   0.8% _init
   3   0.8%  99.2%    7   1.9% load_image_file
   2   0.5%  99.7%    9   2.4% save_image_file
   1   0.3% 100.0%    1   0.3% __open
   0   0.0% 100.0%    2   0.5% 0x00007fff4952c427
   0   0.0% 100.0%    1   0.3% _IO_fgets
   0   0.0% 100.0%    1   0.3% _IO_file_fopen
   0   0.0% 100.0%    1   0.3% _IO_file_open
   0   0.0% 100.0%   368  99.7% __libc_start_main
   0   0.0% 100.0%   368  99.7% _start
   0   0.0% 100.0%   350  94.9% edges
   0   0.0% 100.0%   368  99.7% main
```

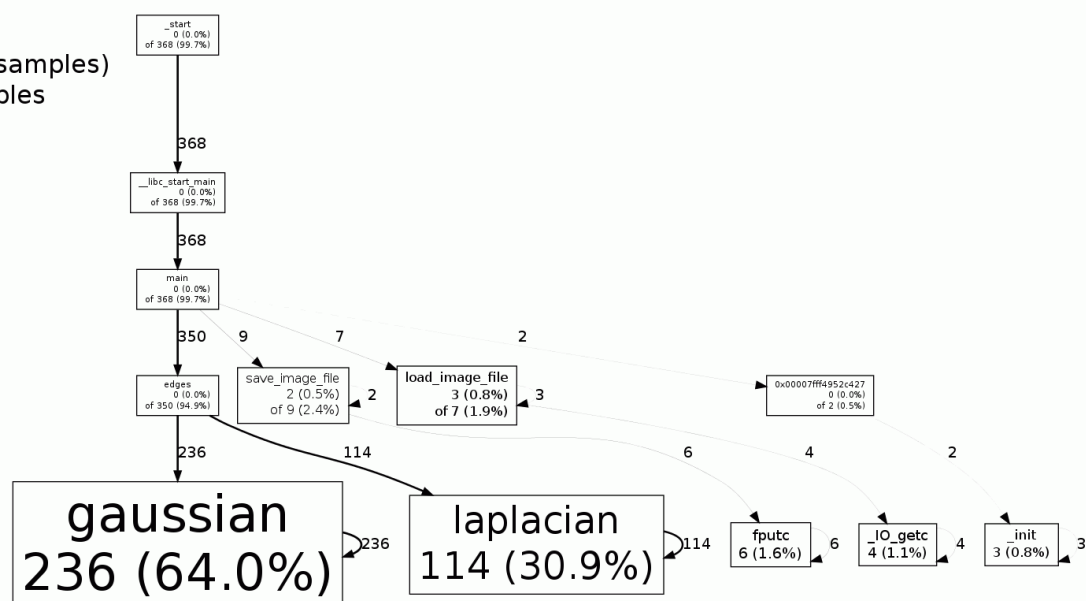
edges

Total samples: 369

Focusing on: 369

Dropped nodes with ≤ 1 abs(samples)

Dropped edges with ≤ 0 samples



4. Google-pprof – 2

En esta segunda parte, usaremos la herramienta anterior para obtener distintos tiempos de ejecución:

```
usuario@debian:~/Escritorio/p2$ gcc -o edges edges.c -lprofiler
usuario@debian:~/Escritorio/p2$ time -p ./edges img.pgm out.pgm
real 4.37
user 4.09
sys 0.05

usuario@debian:~/Escritorio/p2$ gcc -o edges edges.c -
lprofilerusuario@debian:~/Escritorio/p2$ time -p
CPUPROFILE=/tmp/edges.conf ./edges img.pgm out.pgm
PROFILE: interrupts/evictions/bytes = 424/301/22624
real 4.60
user 3.96
sys 0.26

usuario@debian:~/Escritorio/p2$ time -p CPUPROFILE=/tmp/edges.prof
CPUPROFILE_FREQUENCY=1000 CPUPROFILE_REALTIME=1 ./edges img.pgm out.pgm
PROFILE: interrupts/evictions/bytes = 1565/984/72760
real 4.84
user 4.21
sys 0.20
```

Tipos de muestreo	Tiempos de ejecución
Sin muestreo	4.37
Con muestreo (opciones por defecto)	4.60
Con muestreo (1000 veces por segundo con temporizador de tiempo real)	4.84