**/\***

**Roll no:09bce026**

**Author: Sameer Makwana**

**\*/**

**GPROF :**

Profiling allows you to learn where your program spent its time and which functions called which other functions while it was executing. This information can show you which pieces of your program are slower than you expected, and might be candidates for rewriting to make your program execute faster. It can also tell you which functions are being called more or less often than you expected. This may help you spot bugs that had otherwise been unnoticed.

Since the profiler uses information collected during the actual execution of your program, it can be used on programs that are too large or too complex to analyze by reading the source. However, how your program is run will affect the information that shows up in the profile data. If you don't use some feature of your program while it is being profiled, no profile information will be generated for that feature.

Profiling has several steps:

* You must compile and link your program with profiling enabled. See section Compiling a Program for Profiling.
* You must execute your program to generate a profile data file. See section Executing the Program to Generate Profile Data.
* You must run gprof to analyze the profile data. See section gprof Command Summary.

The result of the analysis is a file containing two tables, the flat profile and the call graph (plus blurbs which briefly explain the contents of these tables).

**GPROF COMMANDS:**

After you have a profile data file `gmon.out', you can run gprof to interpret the information in it. The gprof program prints a flat profile and a call graph on standard output. Typically you would redirect the output ofgprof into a file with `>'.

You run gprof like this:

gprof *options* [*executable-file* [*profile-data-files*...]] [> *outfile*]

Here square-brackets indicate optional arguments.

If you omit the executable file name, the file `a.out' is used. If you give no profile data file name, the file `gmon.out' is used. If any file is not in the proper format, or if the profile data file does not appear to belong to the executable file, an error message is printed.

You can give more than one profile data file by entering all their names after the executable file name; then the statistics in all the data files are summed together.

The following options may be used to selectively include or exclude functions in the output:

-a

The `-a' option causes gprof to suppress the printing of statically declared (private) functions. (These are functions whose names are not listed as global, and which are not visible outside the file/function/block where they were defined.) Time spent in these functions, calls to/from them, etc, will all be attributed to the function that was loaded directly before it in the executable file. This option affects both the flat profile and the call graph.

-e *function\_name*

The `-e *function*' option tells gprof to not print information about the function *function\_name* (and its children...) in the call graph. The function will still be listed as a child of any functions that call it, but its index number will be shown as `[not printed]'. More than one `-e' option may be given; only one *function\_name* may be indicated with each `-e' option.

-E *function\_name*

The -E *function* option works like the -e option, but time spent in the function (and children who were not called from anywhere else), will not be used to compute the percentages-of-time for the call graph. More than one `-E' option may be given; only one *function\_name* may be indicated with each `-E' option.

-f *function\_name*

The `-f *function*' option causes gprof to limit the call graph to the function *function\_name* and its children (and their children...). More than one `-f' option may be given; only one *function\_name* may be indicated with each `-f' option.

-F *function\_name*

The `-F *function*' option works like the -f option, but only time spent in the function and its children (and their children...) will be used to determine total-time and percentages-of-time for the call graph. More than one `-F' option may be given; only one *function\_name* may be indicated with each `-F' option. The `-F' option overrides the `-E' option.

-k *from...* *to...*

The `-k' option allows you to delete from the profile any arcs from routine *from* to routine *to*.

-v

The `-v' flag causes gprof to print the current version number, and then exit.

**Sample table**

Flat profile:

Each sample counts as 0.01 seconds.

% cumulative self self total

time seconds seconds calls ms/call ms/call name

33.34 0.02 0.02 7208 0.00 0.00 open

16.67 0.03 0.01 244 0.04 0.12 offtime

16.67 0.04 0.01 8 1.25 1.25 memccpy

16.67 0.05 0.01 7 1.43 1.43 write

16.67 0.06 0.01 mcount

0.00 0.06 0.00 236 0.00 0.00 tzset

0.00 0.06 0.00 192 0.00 0.00 tolower

0.00 0.06 0.00 47 0.00 0.00 strlen

0.00 0.06 0.00 45 0.00 0.00 strchr

0.00 0.06 0.00 1 0.00 50.00 main

0.00 0.06 0.00 1 0.00 0.00 memcpy

0.00 0.06 0.00 1 0.00 10.11 print

0.00 0.06 0.00 1 0.00 0.00 profil

0.00 0.06 0.00 1 0.00 50.00 report