76.2.8 Gprof Profiling

gprof is a profiling tool available in many UNIX platforms, it uses *statistical time-sampling*.

You can build a profiled version of perl called "perl.gprof" by invoking the make target "perl.gprof" (What is required is that Perl must be compiled using the `-pg` flag, you may need to re-Configure). Running the profiled version of Perl will create an output file called *gmon.out* is created which contains the profiling data collected during the execution.

The gprof tool can then display the collected data in various ways. Usually gprof understands the following options:

-a

Suppress statically defined functions from the profile.

-b

Suppress the verbose descriptions in the profile.

-e routine

Exclude the given routine and its descendants from the profile.

-f routine

Display only the given routine and its descendants in the profile.

-s

Generate a summary file called *gmon.sum* which then may be given to subsequent gprof runs to accumulate data over several runs.

-z

Display routines that have zero usage.

For more detailed explanation of the available commands and output formats, see your own local documentation of gprof.

76.2.9 GCC gcov Profiling

Starting from GCC 3.0 *basic block profiling* is officially available for the GNU CC.

You can build a profiled version of perl called *perl.gcov* by invoking the make target "perl.gcov" (what is required that Perl must be compiled using gcc with the flags `-fprofile-arcs -ftest-coverage`, you may need to re-Configure).

Running the profiled version of Perl will cause profile output to be generated. For each source file an accompanying ".da" file will be created.

To display the results you use the "gcov" utility (which should be installed if you have gcc 3.0 or newer installed). *gcov* is run on source code files, like this

```
gcov sv.c
```

which will cause *sv.c.gcov* to be created. The *.gcov* files contain the source code annotated with relative frequencies of execution indicated by "#" markers.

Useful options of *gcov* include `-b` which will summarise the basic block, branch, and function call coverage, and `-c` which instead of relative frequencies will use the actual counts. For more information on the use of *gcov* and basic block profiling with gcc, see the latest GNU CC manual, as of GCC 3.0 see

```
http://gcc.gnu.org/onlinedocs/gcc-3.0/gcc.html
```

and its section titled "8. gcov: a Test Coverage Program"

```
http://gcc.gnu.org/onlinedocs/gcc-3.0/gcc_8.html#SEC132
```