Tuning GENEHUNTER for personal computers

Sylvain Bolay, Etienne Bagnoud, Daniel F. Schorderet
IRO - Institute of Research in Ophthalmology, Sion, Switzerland

ABSTRACT

Purpose: GENEHUNTER (gh) is a linkage software developed by Kruglyak et al (1996). Based on parametric and non-parametric algorithms, it has helped identifying many loci and associations and has become one of the most popular linkage tool freely available to the scientific community. Unfortunately, the full capabilities of gh can only be appreciated when running on fast computers. Typically, its use on personal computers may increase the computing time up to several hours in complicated families. We therefore reengineered the source code in order to accelerate the execution time.

Methods: The reengineering of the source code was performed on a Linux operating system (processor Intel PIV 3GHz, 1GB of RAM). First, the code was tested to measure the time used for typical applications. Then, it was analyzed in order to identify and quantify the time used by each discrete function. We identified the top 10 functions in time consumption and concentrated our efforts in reengineering them by moving calls to variables outside of loops (while, for,...) and avoiding calls to variables which were pointers to pointers. The reengineered gh version was compared using real pedigrees from our linkage and mapping activity.

Results: The time needed by gh to perform calculations on our machine was decreased from 34'20" to 18'45" by changing the compilation optimization mode from level 0 to 3. Moreover, improving the source code in the top 10 functions has decreased the calculation time down to 5'01".

Conclusion: A simple re-writing of the most frequently used functions improved the speed of gh by a factor of 7. In-depth reengineering should even increase the speed of the program further.

INTRODUCTION

Reengineering GENEHUNTER[1,2] (gh) was guided by the necessity to decrease the time needed to perform LOD score calculation on our web-based collection of linkage programs. When a user enters our linkage web facility, he/she expects to have results as quickly as possible. A short calculation time is also a requirement of the web server security policy which limits the user's session connection time.

In order to continue to propose the gh calculation tool, we decided to analyze it with tools (profilers) specially designed to perform memory and time consumption analysis. The reengineered gh version was compared using real pedigrees from our linkage and mapping activity. The test family included 32 persons, 11 were affected with a dominant disease (fig. 1).

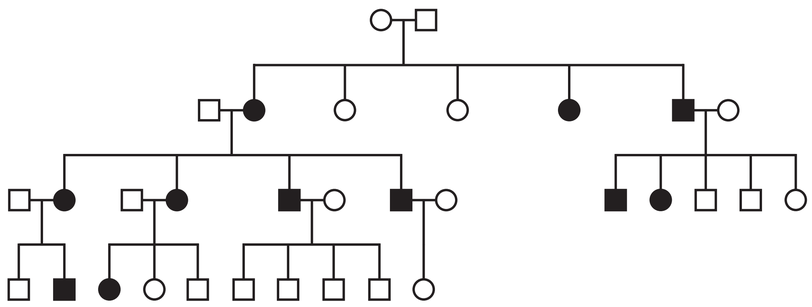

Figure 1 Pedigree of a Swiss familly used to perform the gh analysis. All comparisons are based on this familly.

MATERIALS

This work and the time component reported are based on measurements made on an Intel PIV 3GHz processor (i686) with 1GB of RAM. The operating system is the Debian[3] (Linux) sarge (3.1R0a) with the GNU C compiler[4] (**gcc**) version 3.3.5.

We used the GNU profiler[5] (**gprof**) version 2.15 to calculate the duration of each routine of the software.

The **time**[6] command version 1.7 displays information about the resources used by the program while running.

Finally, to track where and how the memory is allocated we used **MemCheckDeluxe**[7] version 1.2.2. This is an enhanced leak finder/memory profiler. In this work the use of this profiler was marginal and is not discussed further.

METHODS

The gh version 2R5 was used for the reengineering.

To have a clean start point, we first compiled the software wihout any optimization but adding a "-pg" option to make gprof profile available.

By doing this, the gh software is not optimized by gcc and it provides a true reference.

```
photo photo_run
cd
time
skip large off
max bits
max bits 19
postscript output off
count recs on
haplotype off
analysis LOD
load markers linkloci.dat
use
units
increment step 5
scan pedigree chr7.raw
postscript output on
total stat
lod_plot_chr7_5.ps
info_content_chr7_5.ps
quit
```

Figure 2 This is the script.in file that is redirected (./gh < script.in) into the standart input (stdin) of gh.

The calculation was automatically run by giving a script file (fig. 2) as input. This has also the advantage of avoiding instruction errors and eliminating the reaction time of the user from the software evaluation.

The marker-locus data file (fig. 3) linkloci.dat that contains allele frequencies for each genetic marker, the frequency and the penetrance informations for the disease are based on the pedigree of the figure 1. This is also the case for the pedigree file (chr7.raw).

```
33 0 0 5                                  << NO. OF LOCI, RISK LOCUS, SEXLINKED (IF 1), PROGRAM
0 0.0 0.0 0                               << MUT LOCUS, MUT RATE, HAPLOTYPE FREQUENCIES (IF 1)
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33
1  2                                      << AFFECTATION, NO. OF ALLELES
0.999 0.001                               << GENE FREQUENCIES
1                                         << NO. OF LIABILITY CLASSES
1.0 1.0 0.0                               <<PENETRANCE
3  10  # Marker1
0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1   << ALLEL FREQUENCIES
3  10  # Marker2
0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1   << ALLEL FREQUENCIES
{ ... }
3  10  # Marker31
0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1   << ALLEL FREQUENCIES
3  10  # Marker32
0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1   << ALLEL FREQUENCIES
0 0                                       << SEX DIFFERENCE, INTERFERENCE (IF 1 OR 2)
1 1 1 
```