GLEAM and HyGLEAM

Short Introduction and Guide for a Quick Installation,

Wilfried Jakob
KIT, Campus North, Institute for Automation and Applied Informatics (IAI)
Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany
email: wilfried.jakob@partner.kit.edu

1. Short Introduction

GLEAM (General Learning Evolutionary Algorithm and Method) is an Evolutionary Algorithm (EA) of its own and HyGLEAM (Hybrid General Purpose Evolutionary Algorithm and Method) is its memetic extension. GLEAM uses chromosomes whose genes can be easily configured in an application-specific way, so that a gene can determine all (relevant) properties of a given phenotypic element. For example, in a scheduling problem, this could be the resources to be used in a scheduling operation, e.g., a workstation or a classroom and an appropriate tutor. GLEAM works with three types of chromosomes: Fixed-length chromosomes with or without meaning-bearing order, and dynamic-length chromosomes also with meaning-bearing order. In GLEAM, fitness calculation is based on the (cascaded) weighted sum. To control the selective pressure, partner selection is performed with linear ranking. Furthermore, instead of the otherwise common panmictic population, a structured one based on the neighborhood model is used. The self-adaptive balance between breadth-first and depth-first search thus achieved efficiently counteracts premature convergence. The neighborhood model can be viewed as a linear variant of cellular EAs. The memetic extension of GLEAM integrates local search procedures or heuristics to speed up evolutionary search.

Further details and references can be found in the User Manual and in the rest of the documentation found in the `Docu` sub directory.

There are two program versions of GLEAM per application, one command line version (CLV) and one with a simple textual user interface (TUI). Both are started in a shell. There is no implementation with a graphical user interface.

The software is written in C under Linux. The distribution is based on Ubuntu 20.04.1 LTS and the `gcc` 9.3.0 compiler. The `make` utility is required for program generation.

2. Quick Installation

As the GLEAM distribution contains pre-compiled programs it may not be necessary to compile and link a new program for a first start. In particular, if you use a Linux distribution like Ubuntu.

Otherwise, the `gcc` and `g++` compilers are required, which are usually shipped with tools like the `make` utility. The `gnuplot` program should also be installed. It is used to visualize results and the normalization or penalty functions used for fitness calculation.

The following steps are based on the recommended directory structure:

1. Create a directory as root for your installation. E.g.
   `/home/<your user name>/gleam`

2. Download the contents of the distribution into that directory. This should result in four sub directories of the `gleam` folder named `Docu`, `InitFiles`, `linux`, and `sources`.
3. Open a unix shell and move to one of the three applications contained in the `linux` folder, e.g. `gleam_ae`. Move to the `testfield` sub folder.
4. Create the environment variable `GLEAM_ROOT` pointing at the `InitFiles` folder. In Ubuntu linux this is done by the command:
   `export GLEAM_ROOT=/home/<your user name>/gleam/InitFiles`
   You may wish to include this command in your start-up script (e.g. `.bashrc`) to avoid having to repeat the command at every shell start.
5. Assuming you have moved to `gleam_ae/testfield` you can start now GLEAM by entering:
   `../gleam_ae mbf_fox_rot_e.exp myFirst.log`
6. Have fun! And may be it is wise to take a look into the documentation like the user Manual or the documentation on applications already included in GLEAM. Like the Foxhole test function, for which you have just loaded the initialization files.

In case the included programs do not start, a recompilation is probably a suitable procedure. For this you move into one of the program directories, e.g. `gleam_ae`. There you adapt with an editor the file `glob_inc.mak`, in which you assign the correct path to the variable `ROOT`. This is the first statement after the comment part. After that you compile the program by calling `make` without parameters.

The compilation of the English-language TUI variants is preset in the `schalter.h` files for all three applications. For the compilation of the CLV versions or German program variants it is absolutely necessary to consult the software documentation beforehand, especially the sections 2.6 and 3.1. There the procedure is described in detail. Non-compliance will be punished with (many) error messages!

If a compiler other than `gcc` is to be used, more changes are required. In the files `makefile` and `gleamCLV.mak`[1] the compiler switches and the declaration of the used compiler in the variable `CC` have to be adapted. For this the documentation of the used compiler and if necessary the make utility is to be consulted. The mentioned modifications have to be repeated for the `make` files of the other two applications.

1 The make files of the CLV versions of the other two applications are called `hyGleamCLV.mak` and `hyGleamOpalvCLV.mak` respectively.