Evolutionary Computation 2006: Home problems, set 1

General instructions. READ CAREFULLY!

Problem set 1 consists of three parts. Problem 1.1 is mandatory, the others voluntary (but check the requirements for the various grades on the web page). After solving the problems, collect your answers and your programs in *one* zip file which, when opened, generates one folder for each problem (e.g. Problem 1.1, Problem 1.2 etc.) in the assignment. Make sure to keep copies of the files that you hand in!

You should provide a brief report in the form of a PDF, PS or text file. In the case of analytical problems (Problem 1.3) make sure to include all the steps of the calculation in your report, so that the calculations can be followed. Providing only the answer is *not* sufficient. Whenever possible, use symbolical calculations as far as possible, and introduce numerical values only when needed.

In *all* problems requiring programming, use Matlab (v.6 or v.7). The *complete* Matlab program for the problem in question (i.e. all source files) must be handed in, collected in the same folder. In addition, *clear* instructions concerning how to run the programs *should* be given in the report. It should *not* be necessary to edit the programs, move files etc. Programs that do not function or require editing to function will result in a deduction of points.

The maximum number of points for problem set 1 is 10. Incorrect problems will *not* be returned for correction, so please make sure to check your solutions and programs carefully before e-mailing them to `mattias.wahde@chalmers.se`.

You may, of course, discuss the problems with other students. However, each student *must* hand in his or her *own* solution. In obvious cases of plagiarism, points will be deducted from all students involved. Don't forget to write your name and civic registration number on the front page of the report!

NOTE: Please make sure to follow the instructions above, as a failure to do so may result in a deduction of points!

Deadline: 20061002

Problem 1.1, 3p, Basic GA program

Write a standard GA as defined on pp. 12-13 in Handout 2. You are welcome to use the code from Handout 3, but remember that a standard GA uses different operators to some extent. In addition to the main program, you should write Matlab functions (placed in *separate* M-files) for

1. initializing a population (`initpop.m`),
2. decoding a (binary) chromosome (`decode_chromosome.m`),
3. evaluating an individual (`evaluate_individual.m`),
4. ranking fitness values for the whole population (`rank_fitness.m`),
5. selecting individuals with roulette-wheel selection (`roulette_wheel_select.m`),
6. carrying out crossover (`crossover.m`)
7. carrying out mutations (`mutate.m`).

You should select an appropriate, non-negative fitness function, such that good individuals obtain high fitness values.

Hint: For fitness ranking, use the `sort` function in Matlab!

Next, as a test of your GA, find (and report) the *minimum* of the function

$$f(x_1, x_2, x_3) = 1 + 50(x_1^2 - x_2)^2 + (1 - x_1)^2 + 50(x_2^2 - x_3)^2 + (1 - x_2)^2. \quad (1)$$

in the interval $x_1, x_2, x_3 \in [-3, 3]$, as well as the point (x_1, x_2, x_3) for which the minimum occurs. Select (and report) appropriate parameters for the GA (e.g. by trial-and-error). Use at least 20 genes (bits) per variable in the chromosomes.

Problem 1.2, 5p, Reverse engineering of genetic regulatory networks

In biological systems, some genes regulate the activity of other genes. Consider the following (strongly simplified) model for gene regulation

$$\tau_i \frac{\mathrm{d}x_i}{\mathrm{d}t} + x_i = \sigma\left(\sum_{j=1}^{n} w_{ij}x_j + b_i\right), \ i = 1, \ldots, n \tag{2}$$

where x_i denotes the activity of gene i. τ_i are time constants, w_{ij} determines the influence of gene j on gene i, b_i are bias terms, and n is the number of genes in the network. The squashing function is taken as

$$\sigma(z) = \frac{1}{1 + \mathrm{e}^{-z}}. \tag{3}$$

Next, consider a very simple network, consisting of only two genes. With so called microarray techniques it is possible (in principle) to measure the expression levels (activity) of many genes simultaneously. If several such measurements are made at different times, time series are obtained for the variation $x_i(t), \ i = 1, \ldots, n$ of the expression levels.

Given such time series, it is possible (in certain cases) to infer the parameters τ_i, w_{ij}, and b_i of the network, assuming that the model accurately describes the dynamics of the system. For example, a GA can be used, in which the chromosomes encode the $n(n+2)$ parameters, and where the fitness measure is taken as $f = 1/e$, where

$$e = \sum_{k=1}^{T} \left((x_1 - X_1(k))^2 + (x_2 - X_2(k))^2\right), \tag{4}$$

where T is the number of data points in the time series, $X_i(k), i = 1, 2$ are the measured values at point k, and $x_i, i = 1, 2$ are the values obtained at the same point, by integration of Eq. (2), using a given set of parameters τ_i, w_{ij}, and b_i.

Consider the time series $X_i(t)$ given on the course web page (`HP1.2data.txt`). The series contain $T = 20$ time points (excluding the starting point ($t = 0$) which is only used for setting the initial values for the integration) and are (unrealistically) noise-free. Write a GA, with real-number encoding of the $n(n+2)$parameters, and use it to infer the network parameters according to the method described above! Start by discretizing Eq. (2), i.e. express $x_i(t + \mathrm{d}t)$ as

$$x_i(t + \mathrm{d}t) = x_i(t) + \frac{\mathrm{d}t}{\tau_i} g(t), \tag{5}$$

where $g(t)$ is a function that you should determine. Use a simple first-order (Euler) method for integrating the equations from $t = 0$ to $t = 10.0$, using a time step $\mathrm{d}t = 0.1$, i.e. a total of 100 integration steps, such that, after the first step t (time) equals 0.1, after the second $t = 0.2$ etc. Note that the time step is shorter, by a factor 5, than the spacing between the data points. Make sure to extract data at the correct times ($t = 0.5, 1.0, 1.5$ etc.), for use in Eq. (4). For example, $t = 0.5$ corresponds to $k = 1$ etc. You may assume that the time constants are in the range $[1, 10]$ time units, and that the w_{ij} and b_i are in the range $[-5, 5]$.

Note that, even though the data are noise-free, the number of data points is quite small compared to the number of parameters, and it is not possible to pin down the parameters exactly. Therefore, make N (= 10 or more) runs, stopping at a given fitness value f_0, which *should* be 1000 or more for full points, and take the average value of each parameter over the N runs. Thus, in your report, give the average values of the $n(n+2)$ parameters τ_i, w_{ij}, and b_i! Also, determine (and write in the report) the exact fitness value for the network represented by the *average* parameter values.

Problem 1.3, 2p, Analytical crossover

Consider a population (in a GA) consisting of four individuals with chromosomes 101101, 100101, 110000, and 010101 and with fitness values 4, 3, 2, and 1, respectively. Consider the schema S =xx11xx. Assume that the crossover point is always located in the middle of the chromosomes, i.e. between genes 3 and 4, and that the first selected pair consists of individuals 1 and 4, and compute (exactly, i.e. *not* using the estimate provided by the schema theorem) the probability of finding, in the next generation, k copies of S, for $k = 0, \ldots, 4$, assuming that individuals are selected using roulette-wheel selection. Let the crossover probability be equal to 1, and do *not* use elitism. Set the mutation rate p_{mut} to 0.

Finally, compute the average number of copies of S in the next generation.

NOTE: Make sure to show *clearly* the steps in your calculation, by making a table (in your report) showing the various possibilities. Giving only the answers will result in 0 points.

Hint: Since the first two parents are already given (see above), a single additional selection step is needed. The selection steps are carried out independently of each other.