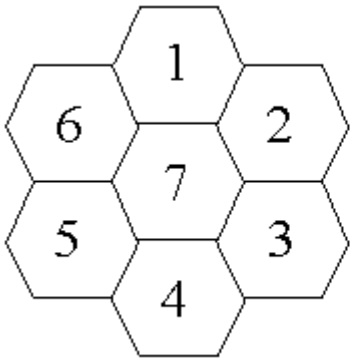
CEE 5290/CS 5722/ORIE 5340 Heuristic Methods for Optimization
Homework 6: Real Genetic Algorithm and Cellular Networks
Assigned: Fri, Oct 3 , 2014
Due: Fri Oct 10, 2014 @ noon
TA office hours: Thu 3:00pm – 4:30pm in Hollister 359

Readings: Readings: Lecture handouts on Cellular networks, Real Valued GA and constraint handling in GA; Optional reading is Reed, P., B. Minsker, and D. E. Goldberg, 2000 Designing a competent simple genetic algorithm for search and optimization. *Water Resources Research*, 36(12): 3757-3761. *Instructions on how to download this paper from the library are in the Assignments link under HW6 in Blackboard*.

---

1. **Cellular Networks problem (no programming required)**: Consider the following cellular network with interference matrix I and Traffic Vector shown below:



$$I = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{bmatrix} 5 & 2 & 0 & 0 & 0 & 2 & 2 \\ 2 & 5 & 2 & 0 & 0 & 0 & 2 \\ 0 & 2 & 5 & 2 & 0 & 0 & 2 \\ 0 & 0 & 2 & 5 & 2 & 0 & 2 \\ 0 & 0 & 0 & 2 & 5 & 2 & 2 \\ 2 & 0 & 0 & 0 & 2 & 5 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 5 \end{bmatrix} \end{array}$$

Traffic Vector = [1 2 2 1 2 2 4]

I is the interference matrix. Hence if **I** = [a_{kj}], then a_{kj} is the number of channels that must be in between cell *k* and cell *j* to avoid interference. So if $a_{kj} = 4$, there must be three channels between the channel in cell *k* and the channel in cell *j*. Hence if channel 1 is in cell *k*, then the channel in cell *j* must be 5 or greater.

a) Provide a solution/channel assignment that has no conflicts. (You can pick the number of channels – try to keep the number of channels to a minimum). Describe this by a matrix with rows for cells and columns for channels, as was done in Prof. Wicker's lecture.

b) Let your objective function be the number of conflicts due to interference. What is the value of the objective function for the channel allocation following (on the next page) if we are trying to minimize the number of conflicts?

Channels

0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	1	0	0
0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	1
1	0	0	0	1	0	0	0	0	1	0	0	1

c) How many decision variables are in channel allocation in part (b). i.e. how many decision variables are in a problem with 7 cells and 13 channels?

d) Extra challenge (not graded and not extra credit). Since this is a small problem, you can figure out the minimum number of channels required to avoid interference. Is this minimum number = 17?

2. **Real-valued optimization – Bump function**. You will code a real-value GA and apply it to maximize the function outlined below. Your real-value GA will look very similar to your binary GA coding (i.e. assuming your binary GA works OK, modify your binary GA so that it works as a real-value GA).

The following objective function (called "Bump") will demonstrate the application of Genetic Algorithms to real valued problems, using real coding, i.e. real numbers and not binary strings are decision variables.

Objective Function: The "Bump" function was developed by Dr. Andy Keane from Southampton University, UK. The objective function is easy to code with arbitrary numbers of dimensions but hard to solve - and generates a very rough and bumpy surface. It is believed to be an open problem and the global solution is unknown. The best solutions found by students in last years' class, *not necessarily using GA*, were close 0.8 (if you are concerned about your GA performance you can always at least make sure the GA outperforms random sampling). The function to be *maximized* is the following:

$$f(\vec{x}) = \begin{cases} \left| \dfrac{\sum_{i=1}^{n} \cos^4(x_i) \; - \; 2\prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_i^2}} \right|, & if \;\; (\forall i,\, 0 \le x_i \le 10) \; and \; (\prod_{i=1}^{n} x_i \ge 0.75) \\ 0, & Otherwise \; (i.e. not \; feasible) \end{cases}$$

For this homework, you will work with the multi-dimensional case $n = 20$ (except for part (i)).

i. Write a MATLAB/Python file bump.m that implements the bump function for arbitrary n. Also generate and submit a 3-D surface plot and a contour plot for $n = 2$ (use Matlab commands surf and contour). Let the x and y-axes range from -1 to 11 and you will need 100-500 points in each direction to get an acceptable resolution. Note that bump (ones(1,20)*0.99) should return 0.1264.

ii. Code a real-value Genetic Algorithm with elitism, and single-point crossover and tournament selection. For mutation, pick one of the 20 variables and add to it a Gaussian random number with mean 0 and variance V. Make sure that your mutation operation returns decision variable values that satisfy the decision variable *bound* constraints.

iii. You need to determine a good set of parameters for the GA applied to the bump function. In this case use number of elite parents to carry over to the next generation, K=1, fix the population size at 50 and allow it to evolve over 200 generations. You will need to determine by experiment good values for V, pMutation and pCrossover (the probability of mutation and crossover). Try at least 2 values for each parameter. Briefly outline the experiment and indicate the best parameters you find.

iv. With the best parameters from (iii), perform 20 GA trials from random initial populations (generate and save 20 random initial populations as you will be using them again later in Question 3):

- Report the average and standard deviation of the fitness of the fittest member of the population (elite solution) over the 20 trials.
- Report the best and worst elite solution you obtain from the 20 trials after 200 generations.
- Submit a plot of average elite solution (averaged over 20 trials) vs. function evaluations. Save this plot as you will add to it in Question 3.

v. Other crossover methods were discussed in class for real valued problems. If you were to select a 2nd method for crossover (from class or something completely new) for this problem, which method would you pick and why? (This question has many correct answers; just give a thoughtful reason for your answer. Computational implementation of new crossover schemes is not required)

3. You may have noticed that the way the bump function is defined above is a simplified approach for constraint handling – if the constraint is violated assign the objective equal to zero. Question 2 is how we have asked the real-value GA question in past years. Now we will have you implement the efficient constraint handling approach by Deb (2000) as outlined in the lecture notes to solve this problem and see if the revised approach works better.

The fitness function is defined as:

$$F(x) = \left| \frac{\sum_{i=1}^{n} cos^4(x_i) - 2\prod_{i=1}^{n} cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_1^2}} \right| \; when\; 0 \le x_i \le 10 \; and \; \prod_{i=1}^{n} x_i \ge 0.75$$

$$F(x) = f_{min} - \sum_{j=1}^{n} \langle g_j(x) \rangle \;\; otherwise$$

Where the operator $\langle g_j(x) \rangle$ is defined as:

$$\langle g_j(x) \rangle = |g_j(x)| \; for \; g_j(x) < 0$$
$$= 0 \; for \; g_j(x) \ge 0$$

(a) Change your GA coding so that you apply the efficient constraint handling approach in Deb (2000). You need to change the bump function (call it bumpQ3.m) so it returns a value of the objective function and/or a measure of the constraint violation if it is violated (there may be ways to do this returning one value only). In your new fitness function (call it fitness3.m), because you are maximizing the bump function, the measure of the constraint violation must be subtracted from the worst fitness value found so far for feasible solutions. You can embed the bump function within fitness3.m if you wish. There may be other minor changes you make to your GA also.

(b) Under the same starting populations and GA parameters used in Question 2(iv), run the GA with the new constraint-handling approach.

- Report the average and standard deviation of the fitness of the fittest member of the population (elite solution) over the 20 trials.
- Report the best and worst elite solution you obtain from the 20 trials after 200 generations.
- On the graph started for Question #2 plot the average elite solution (averaged over 20 trials) vs. function evaluations.

(c) Which constraint-handling approach worked better? Can you suggest a reason for the performance difference? Although not absolutely required, it may help you answer this question if you try to determine other GA population diagnostics besides just tracking the elite member (e.g. average population fitness, convergence of the population to the elite solution, etc.).