

The MESA Program: An Executive Summary

By Micah Sparacio

The ~40K executable Jar file can be downloaded here:

http://www.iscid.org/projects/mesa/mesa_1_3.jar

In order to run this program you must have the Java Runtime Environment (JRE) installed on your machine. If not, please download it here:

Windows: <http://java.sun.com/j2se/1.3/jre/download-windows.html>

Others: <http://java.sun.com/getjava/others.html>

The (very)basic algorithm:

1. Randomly generate initial population
2. Apply fitness function in which couplings are only given a beneficial fitness value if each bit in the coupling matches the target. Then apply the fitness perturbation if it is being used.
3. Repopulate: if elitism is on, keep the most fit half of population the same, and reproduce new half with mutation from the most fit half. If elitism is off, take most fit half of population, and reproduce each one twice with mutation.
4. If crossover is on, do crossovers based on crossover rate and randomly generated crossover pairs
5. Do steps 2-4 until you have reached the target organism

Some guidelines for use:

Please note that at this stage error checking on user input is not yet implemented. It is up to you the user to input the appropriate values.

This program allows for the user to modify the following parameters:

1. Mutation Distribution Type

- Bit by bit: probability assessed at each bit in genome (local)
- Poisson Distribution: probability over entire population (global)

2. Target

- at Peak: moving toward higher fitness value
- at Valley: moving toward lower fitness value

3. Fitness Perturbation

- Uniform Distribution: random number between -X and +X
- Binomial Distribution: a binomial distribution using n and p values
- Manual Distribution: perturbation determined by values input at bottom of screen

4. Elitism

- On: Most fit organisms persist without mutation
- Off: Most fit organisms are subject to mutation

5. Crossover

- On: Organisms subject to crossover rate
- Off: Organisms not subject to crossover rate

6. Population size: number of organisms

7. String Length: bit length of organisms

8. Uniform Fitness Pert. Range: used if Fitness Perturbation type is set to Uniform Distribution

9. Binomial Fitness Pert.: used if Fitness Perturbation type is set to Binary Distribution: if $n=20$ and $p=.5$ then you will get the equivalent of 20 random coin tosses with each toss having a 50/50 probability of landing heads. If you increase the p value, the probability of landing heads goes up.

10. Mutation Rate: one mutation for every X bits. Used by both types of Mutation Distribution

11. Crossover rate: one crossover for every X bit pairs

12. Coupling Degree/Total Couplings Pairings

- Coupling degree is the amount of bits that are grouped together
- Total Couplings is the number of blocks in each organism with the specified coupling degree

13. Deviation/Distribution Pairings

- Deviation is a positive or negative integer value perturbation of the fitness value
- % Distribution is the percentage of time the deviation value will be used. Deviation values must add up to 100 percent

14. Keep Record Check Box and Text Box

- Specify whether to keep a record of crossovers (crossovers.txt), coupled bits (coupledbits.txt), and new best fitness values (bestfitness.txt). The files will be stored in the same directory that the program is run from
- Specify the number of generations between records. The higher the number, the fewer records kept, the less computational time required, and the smaller the files stored.