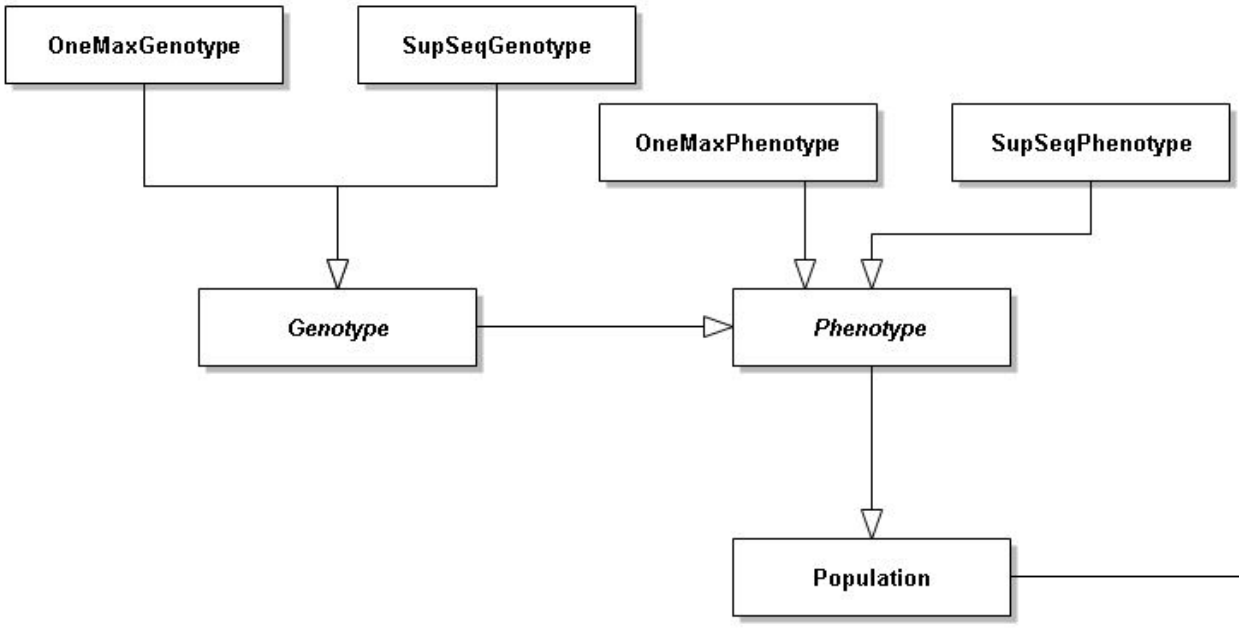
Searching for Surprising Sequences with a Genetic Algorithm

By Kristoffer Hagen

1 The model

I used the framework that I built for the first part of the assignment here as well.



The SupSeqGenotype contains the basic genetic information such as the amount of symbols and the size of the string, together with an array of integers that makes up the genes. The phenotype contain the aforementioned genotype together with methods for mating.

2 The fitness function

The fitness function for this problem is basicly the following;

I take a certain distance between two integers in the genes, then I go through the array of genes and add in pairs the integers that match this distance to a list. For the genes 1 2 3 4 3 2 1 and the distance 1, I would add the pairs 1 3, 2 4, 3 3, 4 2 and 3 1.
Then I will go through the list I just made and look for duplicates, if any are found I

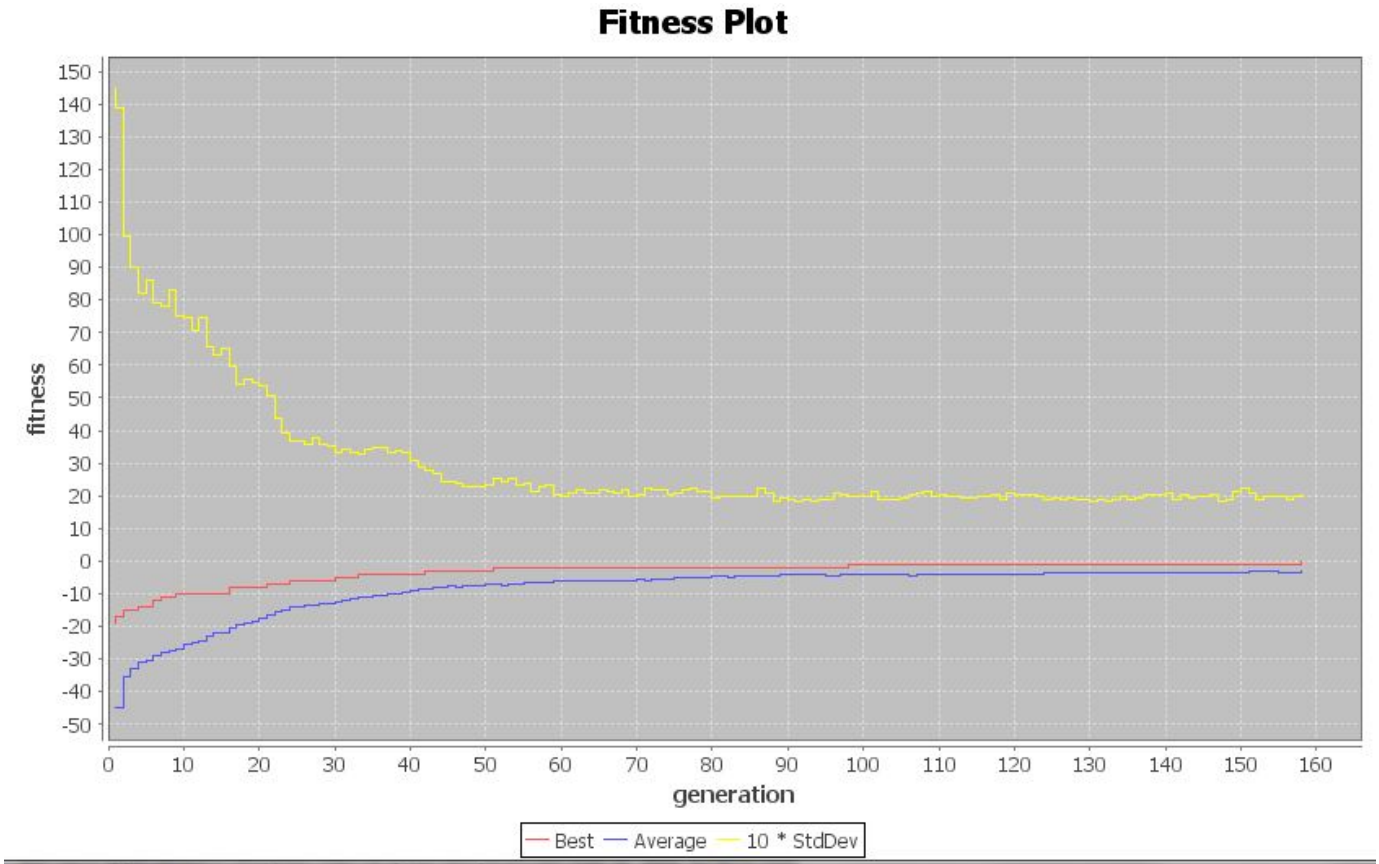
```
//@Override
public double findGlobalFitness() {
    ArrayList<IntPair> pairs;
    int conflict = 0;

    for (int i = 1; i < size; i++) { //distances'
        pairs = new ArrayList<IntPair>();
        for (int j = 0; j < size-i; j++) { //navigation
                int value1 = vector[j];
                int value2 = vector[j+i];
                pairs.add(new IntPair(value1, value2));
        }
        for (int k = 0; k < pairs.size(); k++) {
            IntPair ip = pairs.get(k);
            for (int l = k+1; l < pairs.size(); l++) {
                if(ip.isEqual(pairs.get(l)))
                    conflict++;
            }
        }
    }
```

will subtract one from the current fitness and move on.
Say I had list of pairs that looked like this: 2 1, 2 0, 0 1, 2 1. There is one duplicate here.
For locally surprising sequences I would just look at distance=0, but for globally I would look through all possible distances.

3 The table

For all these runs, in addition to the numbers shown I used the following: Generational adult selection, proportionate parent selection when population is 500, SigmaPieSelector when population is 50. I used a random split crossover (cut the genes at a random point, use one parent for one half, the other parent for the other). Mutation rate of 0.2 (Which I in retrospect wish was lower.)

Fitness plot and output from S=19 L=48 globally surprising:




```
Generation 153
Generation 154
Generation 155
Generation 156
Generation 157
Flawless string found!
Vector:
11 13 12 14 7 8 0 6 17 11 4 17 8 18 3 2 13 9 5 16 16 6 1 15 7 3 5 2 14 0 3 10 17 15 13 18 4 10 4 12 11 1 0 12 8 14 9
Flawless string end!
Generation 158
```

				Locally Suprising			Globally Suprising
Symbols	Population	Generations	**Length**	Sequence	Generations	**Length**	Sequence
3	50	17	**10**	1 1 0 0 2 0 1 2 2 1	1	**7**	0 1 2 1 1 0 2
4	50	18	**17**	3 2 1 2 2 0 3 3 1 1 0 0 1 3 0 2 3	160	**10**	2 3 1 0 1 2 0 3 3 2
5	50	80	**26**	2 4 0 3 2 0 4 3 0 1 3 3 4 1 2 2 1 4 4 2 3 1 1 0 0 2	59	**12**	1 0 0 4 3 2 4 1 4 2 3 0
6	50	288	**37**	0 2 2 3 2 4 0 0 4 4 5 4 1 4 3 3 0 5 1 3 5 5 2 0 1 1 2 1 0 3 4 2 5 3 1 5 0	177	**16**	5 1 3 1 5 0 4 3 0 2 2 1 4 5 3
7	50	54	**49**	5 6 5 0 6 0 3 4 6 4 2 0 4 4 0 1 5 2 3 3 2 4 1 3 5 3 0 0 2 5 1 4 3 6 3 1 2 6 1 1 6 6 2 2 1 0 5 5 4	281	**17**	1 6 4 0 3 0 5 5 6 2 1 5 3 1 0 2 4
8	500	390	**64**	7 4 2 2 6 2 5 4 0 0 1 6 0 5 3 0 6 4 6 1 1 0 7 1 2 0 3 2 1 3 4 4 1 4 3 7 6 7 3 6 3 3 5 0 4 5 2 3 1 5 1 7 2 4 7 0 2 7 5 6 6 5 5 7	115	**20**	2 4 7 7 5 1 0 6 4 2 3 6 3 1 4 3 5 0 7 2
9	500	484	**80**	2 0 8 5 5 3 6 3 5 4 0 1 6 0 0 5 7 4 5 2 7 1 4 3 1 7 0 7 6 4 1 0 2 5 1 2 2 1 5 6 7 3 2 3 3 8 7 8 1 8 8 4 8 3 7 7 2 6 8 6 6 2 8 0 4 4 2 4 6 5 0 6 1 1 3 0 3 4 7 5	186	**23**	0 8 1 3 6 5 4 6 7 7 5 3 4 1 0 2 8 5 8 2 6 3 1
10	500	375	**98**	3 8 9 3 4 8 2 7 3 1 7 1 6 2 2 0 5 5 0 3 0 0 2 8 0 6 0 8 3 7 4 0 7 5 4 9 4 1 9 7 6 7 9 5 1 3 2 5 9 9 8 5 6 1 0 9 0 1 8 1 5 2 6 9 1 2 3 9 6 4 4 6 3 5 7 7 2 1 1 4 7 8 6 6 8 4 5 3 3 6 5 8 7 0 4 2 4 3	218	**25**	2 9 6 7 1 4 5 3 0 4 0 0 8 6 2 5 7 3 9 8 2 4 3 1 7
11	500	156	**120**		115	**27**	2 4 0 4 6 9 8 10 10 2 0 5 1 7 10 3 8 7 4 9 3 4 5 0 1 2 6
12	500	380	**136**		201	**30**	2 7 2 11 5 6 11 0 10 3 8 6 0 9 9 10 1 7 3 4 1 11 2 8 9 5 0 4 6 7
13	500	250	**160**		126	**32**	12 2 10 12 5 4 8 11 1 4 5 7 0 3 2 8 6 6 1 7 3 12 9 0 9 3 11 5 10 4 2 6
14	500	480	**186**		280	**35**	11 3 4 7 5 12 2 8 1 9 12 1 7 11 0 0 3 10 6 13 9 8 4 13 2 13 12 5 3 7 4 8 11 1 5
15	500	420	**210**		307	**38**	13 7 10 8 11 13 11 12 9 4 0 12 3 2 1 9 8 6 5 14 14 5 1 2 0 10 6 7 3 12 0 11 9 5 2 8 13 4
16	500	357	**235**		366	**39**	12 10 6 8 11 4 14 5 7 13 0 3 0 15 5 1 2 11 1 9 13 13 5 15 9 8 12 3 7 10 14 6 1 0 2 4 11 6 10
17	500	272	**260**		238	**42**	13 2 14 11 7 13 5 8 12 4 6 3 3 14 10 1 9 0 10 7 15 11 13 0 2 0 16 8 9 16 15 1 7 4 14 6 5 12 3 15 4 8
18	500	235	**280**		444	**45**	4 12 15 9 17 9 5 16 3 4 1 2 5 10 7 12 6 0 14 7 8 17 11 6 9 13 1 6 13 17 0 2 11 15 15 4 14 16 12 5 13 10 8 7 3
19	500	218	**310**		158	**48**	11 13 12 14 7 8 0 6 17 11 4 17 8 18 3 2 13 9 5 16 16 6 1 15 7 3 5 2 14 0 3 10 17 15 13 18 4 10 4 12 11 1 0 12 8 14 9
20	500	244	**340**		316	**50**	7 11 6 15 7 4 18 12 17 2 6 10 8 9 13 14 0 3 9 5 5 19 14 12 18 1 16 19 13 3 19 4 11 9 6 8 2 8 0 15 12 10 17 11 4 16 2 7 5 13

5 Theory

Given a 20-gene chromosome and onemax, locally and globally surprising sequence with 9 symbols. I would rank then globally, locally, onemax in order of difficulty. Onemax really isn't very problematic as every single improvement is always one step closer to a solution. The genes are all individual, or not connected. Locally surprising sequence is harder because when you modify one gene, that also affects the following one in the sequence. In the string 1 2 3, if the 2 is modified, that changes both the 1 2 pair and the 2 3 pair. This makes it harder to solve than onemax.

For globally surprising sequence it is taken to the next level again. Every gene is now connected to up to many other genes.

In the sequence 0 1 2 1 1 0 2, for example. The first 2 is a part 6 pairs and when it is changed it also changes the fitness of the sequence in a less predictable way, this makes globally surprising sequence the hardest one to solve.