Assignment 2 - Report

Dmitriy Okoneshnikov, B22-CS-01
d.okoneshnikov@innopolis.university

December 6, 2023

Contents

1 EA algorithm

I have chosen a Genetic Algorithm for my solution as, in my opinion, it fits the best out of all four Evolutionary Algorithms, mostly, due to its representation: Genetic Algorithms use data structures.

My solution implements a `Crossword` class, which stores `Word` class objects. The latter holds the word itself, location of the first symbol of the word, direction of the word, and which component of the crossword it belongs to.

`Word` class also contains methods for checking the conditions for crossword correctness, e.g., no two parallel words that are neighbours to each other. `Crossword` class contains calculation of fitness function, which implements lazy loading: it is calculated only on demand and this value is return later.

It should be mentioned that I took GA from Lab #10 as a template and built my solution over it changing classes and functions.

1.1 Algorithm flow

Generally, my solution follows the pseudocode from Lecture #10. The following steps are executed for every input file.

Firstly, I generate 5 random populations and choose one with the highest fitness value. From that point I start my evolution until a valid crossword is generated. For each evolution step I select my parents and breed them (crossover and mutation) to get offsprings that will get appended to the existing population. This new population will get sorted by their fitness value and `population_size` of individuals will be selected for next step. Also, if no changes in fitness values for over `500 * len(words)` generations is detected, then a new random population will get generated (we reset the evolution). Finally, if no solution was found in 5 minutes, the test is terminated and continued to another one.

1.2 Fitness function

The fitness function calculates the penalties for violation of crossword generation rules:

- Whether all the words are intersected and are in one component
- Whether the characters of the words at intersection point are the same
- Whether the words that have the same orientation do not intersect
- Whether parallel words do not exist in neighbouring rows/columns (except first/last symbols)
- Whether no new words are created in the latter

Note: throughout my solution the words are made to generate inside the grid.

Also, I use an idea that was discussed to me by my colleague: instead of using DFS for checking if all words are in one component, we can constantly update at which component each word belongs to as we are the ones generating the maps.

So, if the crossword is correct, then its fitness value will be equal to 0.

1.3 Crossover

For crossover, a one point crossover is used. A random number `index` is chosen between 0 and number of words - 1. First `index` words of the offspring get the location and direction from the mother and others get them from the father.

1.4 Mutation

For mutation, a uniform mutation is used. For every word in the crossword with a probability of 30% its location and direction are changed to a random one.

1.5 EA parameters

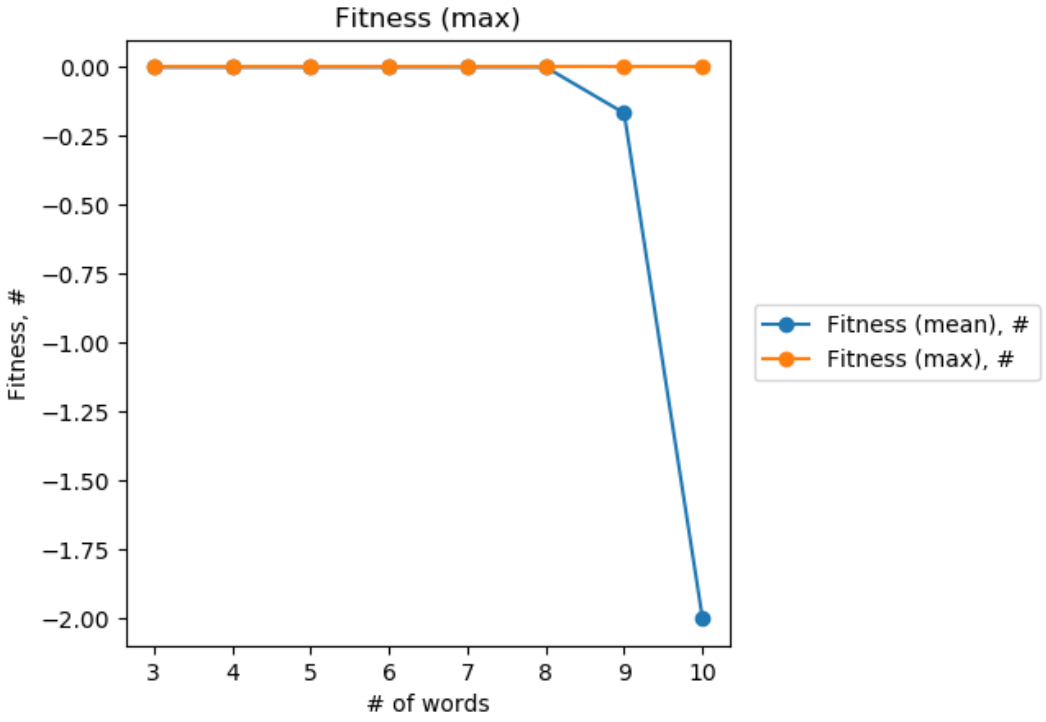
This are parameters that I use:

- Population size: 180
- Offsprings size: 60
- Mutation probability: 30%
- Penalty for words that are not intersected: (number of components - 1) * 12
- Penalty for words that are intersected at different characters: absolute value of difference of ASCII codes of characters
- Penalty for parallel words in neighbouring rows/columns: number of "overlapping" symbols * 8
- Penalty for new words that are created: number of this kind of words
- Penalty for perpendicular words at the beginning or at the end: 30

2 Statistics

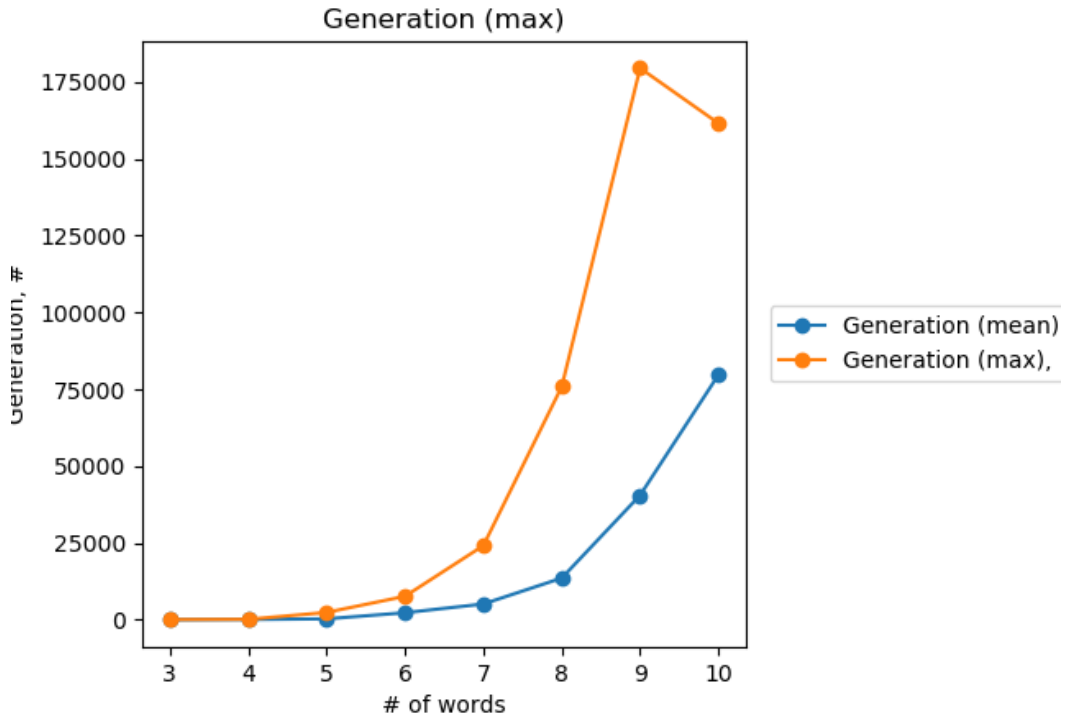
Due to peculiarities of my solution, I could test only for up to 10 words. Also I have used PyPy 3.8 to accelerate the testing of 100 tests.

2.1 Fitness values



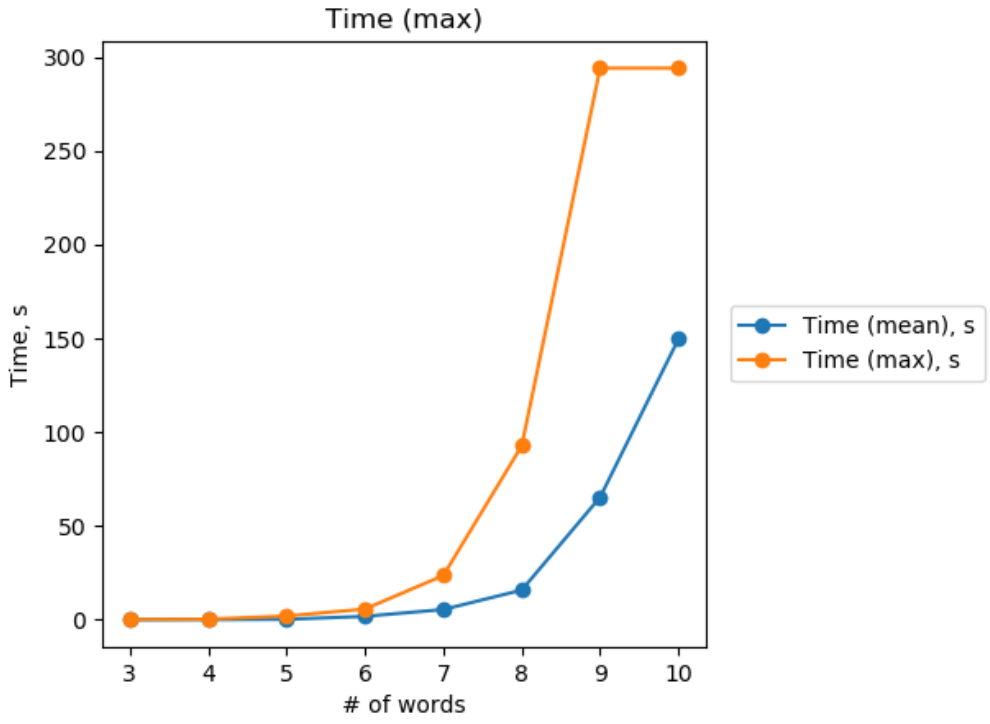
Due to my implementation, I do evolution until the crossword is generated (fitness values equals zero). I have also a time limit implemented at 5 minutes. So, we can see a drop in mean fitness value after 8 words. This is due to my program not being able to create valid crosswords in 5 minutes.

2.2 Number of generations



Maximum value for number of generations for 10 words drops as there are more tests that were not passed due to time limit. It is visible that the number of generations follows an exponential rule.

2.3 Execution times



A plateau is visible at maximum time for 9 and 10 words as I have a time limit at 5 minutes. Also, it is visible that the execution times as well as the number of generations follow an exponential rule.