Crossword Generator

Report

About crossword representation:

To reduce running time of the program, I decided to represent a crossword as a tree graph, where vertices are words, edges contain information about how words cross each other (class Word in the code). I used only one condition for all crosswords – every word has to be crossed by at least one word if it is possible. At the beginning I create a dictionary, where keys are all 26 letters of alphabet. Value of each key is list of indexes of words, which contain this letter (indexes of the words are the same as in input file). The number of indexes of one word is equal to the number of a given letter in that word. Then I randomly choose a letter and from list of this letter randomly choose 2 different words, where at least one word is not crossed by any word. If a word has several chosen letters, then I randomly choose one of these letters in this word. And at the end of iteration, I connect these words in graph representation. I continue this process until I can choose such letter or when all words are crossed by at least one word. At the and I have crossword, represented by tree graph with one or more connectivity components. After this I determine position of each word – horizontal or vertical (every two adjacent vertices (words) in the tree graph have different positions). And then every connectivity component of a crossword I put on separate grid. Coordinates of the word in each component are defined unambiguously, because we know its position, what words in which letters it crosses. Thus, each crossword puzzle can be encoded as a tree graph.

Evolutionary algorithm:

I use genetic algorithm with focus on tree structures, often referred to as Genetic Programming. Each crossword in encoded as a tree graph (forest if some groups of words are not connected). Vertices contain information about the word, edges contain information how two words cross each other – by which letters do words intersect each other. The crossword puzzle is uniquely determined by this graph.

1) **Initialization:** Randomly create individuals (potential solutions) for a crossword
2) **Evaluation:** Evaluate each individual by fitness function, and sort all individuals by their fitness function and leave n solutions with the highest fitness function, where n – population size. More information about fitness function is written below.
3) **Selection:** randomly select m*n individuals for mutation and c*m pairs of individuals for crossover, where m – percentage of n of offsprings after mutation, c – percentage of n of offspring after crossover.
4) **Crossover:** swapping subtrees between two parent trees. More information is written below.
5) **Mutation:** randomly changing a node in the tree. More information is written below.
6) **Replacement:** After mutation and crossover add offsprings to a population then sort all individuals by their fitness function and leave n solutions with the highest fitness function.
7) **Termination:** Repeat the process until fitness function of one individual will be equal to 0 – this means that crossword doesn't contain any error.

After extensive testing of the crossword puzzle on different words, the optimal values were selected:

- Population size – 100
- Initial population size – 1000

- Mutation rate – 1
- Crossover rate – 1

To avoid the local maximum of the fitness function, a complete generation update occurs if the maximum value of the fitness function does not change for 200 generations.

Fitness function:

Fitness function calculates number of errors in a crossword. Calculation is based on grid of each crossword (or list of grids if a crossword has several connectivity components). All errors are separated on some categories: a) going beyond the grid boundaries, b) a word incorrectly intersects another word, c) the words are incorrectly placed next to each other, and d) the crossword puzzle is divided into connectivity components - some words or groups of words are not connected to each other in any way. Each error category has its weight. After testing crossword on different cases, following weights were chosen: weight of connectivity component error is 10, weights of all other error categories are 1. After calculating all errors in crossword each error multiplies by its weight, then we add everything up and return the resulting value multiplied by -1. Thus, maximum value of fitness function is 0 – when crossword doesn't have any error.

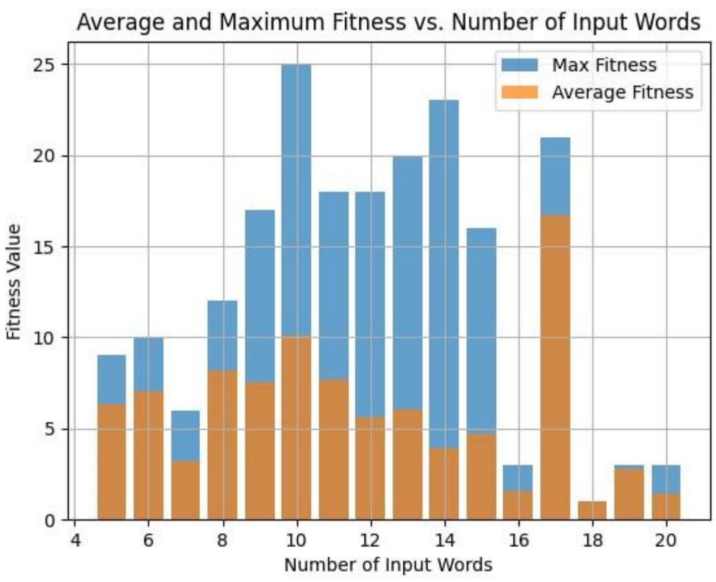
Mutation of a crossword:

Mutation of a crossword is quite simple: randomly choose one word, separate it from all the words attached to it, then randomly connect it with any other word.

Crossover of two crosswords:

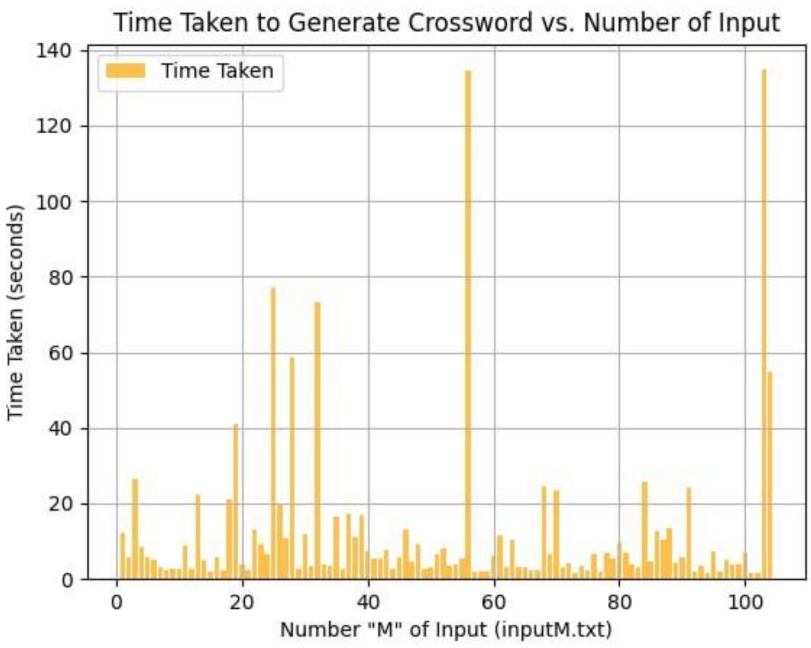
The first crossword is named mother, other one is father. Randomly choose connectivity component of size – $words\_number/2$ (if all connectivity components are less than this number, than just choose the largest one). Let's call all words from mother connectivity component – mother's words, other ones – father's words. After dividing words on father's and mother's, create a new – child crossword, where all words from mother's are connected with each other like in mother crossword, and all words from father's crossword are connected with each other like in father crossword. After it, we get in child crossword one connectivity component from mother crossword (which we chose at the beginning) and several connectivity components from father's crossword. Then iterate throw all father's components in child crossword and try to find one word which was connected in father's crossword with mother's words. If such word exists, then attach this connectivity component to mother connectivity component by this word.

Statistics on 104 random crosswords:



In this graph, the value of the fitness function is multiplied by -1, otherwise the maximum value of the fitness function will be always equal to 0 - so it turned out to make all the crosswords.

I also collected statistics on the dependence of the program execution time on the test number:



You can see these 104 tests here:
https://drive.google.com/drive/folders/1qdueDADcvtieNL_tg9mNrBnCr0yuy-4p?usp=sharing