Artificial Intelligence Computer Assignment #2 Report

Hossein Entezari Zarch SID : 810196419
March 29, 2020

Abstract

In this computer assignment we are to find a mapping between letters which makes us able to decode a given encoded text with the use of genetic algorithm to search for the mapping.

1 Chromosome

- Definition: We define a string with the length of 26 and having only one element of each alphabet which represents a mapping between alphabets, and the i-th element of the string explains that, the mapping of that alphabet is i-th alphabet of the alphabets, for examle if the first element of the strin is 'o', it means, the chromosome maps 'o' to 'a'.

2 Fitting Function

- Definition: We need a function which gives a score for each chromosome according to its distance to the goal chromosome.
- Method: We store a set of encoded words extracted from encoded text, so in order to score each chromosome, with respect to that chromosome after calculation of decoded words, we search for them in the library words and if the word is present, we would add length of the word to the score of that chromosome on each word.
- Other Methods: Also we can instead of adding the length of the found word, just increase the score by 1, but as we know that finding longer words are harder, we decided to perform as we said.

3 Dictionary

- Definition: In the fitting function we said we have a dictionary search for decoded words in it, so we have to prepare a dictionary of word from dirty text given us including all words we have in the main text.
- Cleaning Global Text: In order to Clean the given global text and extracting the unique words out of it, after converting all of the alphabets in the text to the lowercase, with an iteration over the text, on each character if is not an alphabet, we would replace it by space and after this iteration with calling a split() function on it, we would get an array containing all of the words in the global text, at last with giving this array to a set() function we would have a set of all unique word which is able to search for words in O(1) and we call this object as the dictionary.
- Encoded Text Processing: Also we use the above method on the given decoded text in order to have a set of unique words of the given text to perform the decoding on them and score the genes on each fitting function call.

4 Update Generation

4.1 Population Size

- Constant Population Size: In this approach we set the population size as a constant value and through several experiments we found constant size of 100 as an appropriate choice for this amount with a focus on the time needed to find the result. According to the implementation of this approach, on each step of the generation update, after copying the best genes from the generation, on a while loop we call the cross over function gives us two genes and after passing these two genes from mutation function we add them to the generation if they are not currently present in the generation.

- Changeable Population Size: We performed some experiments with this approach, for example starting population size from 100 and increasing it by a constant amount on each update generation step until reaching a constant population size, this approach could not help us and also by increasing time spent on each update step, increased overall time needed to converge to the goal mapping.

4.2 Cross Over

- Implementation: In order to implement the cross over we used Order1 crossover algorithm which after choosing two indices, swap the elements between two genes and then fill the other elements with the alphabets that are not present in the gene with the order of the alphabets in the first version of that gene.
- Rate Of Occurrence: On each calling of the cross over function on two given genes, with the probability of pc the cross over performs and the result are given on the output and so otherwise the input genes are given on output.

4.3 Mutation

- Implementation: In the implementation of the mutation function, on adding each gene to the new generation (except copied genes known as the best genes) we call the mutate function on the gene which with the probability of pm performs the mutation function on the given gene. On each mutation process, on each step, we choose two random indices and swap elements of these two indices with each other and we perform this step for nm times which is a constant chosen by ourselves after performing several experiments.
- Constants: After performing experiments on various values of pm, nm, we found out that, choosing 0.4 for pm and 2 for nm would give us good results.
- Benefits Of Usage: We know that through the approach we got in the flow of our work we prune the best genes always and try to construct better genes by cross over on double randomly chosen genes, but These are not enough for the randomness we need in generating the genes and without mutation we have a high level of chance for out generation to get stuck in a few genes so near each other, so with the use of mutation we try to generate better genes with performing some small changes on a single gene and we would store that gene if it shows a high score unless with a high probability it would be omitted from out generation soon.

4.4 Cross Over V.S. Mutation Importance

- According to the experiments we observed that, cross over starting from a completely random generation can easily lead to a biased generation which maybe is not able to converge to the goal state, but also we clearly observed that mutation through some little changes in genes can easily lead to a better more fit gene coming from each gene and this makes a critical role in generating very good genes from the best genes we currently have through a little swap between a few elements.

4.5 Save Best Chromosomes

- Implementation: In order to this approach on the update process of generations we save a few of the best chromosomes from last generation in order to always save the best chromosomes and increase the chance of finding the goal by cross over and mutation processing on these chromosomes on subsequent steps and find the goal.
- Rate Of Occurrence: According to the percentage of the generation that are chosen on each step to be copied to the next generation, we performed some experiments with different amounts and we observed that with increasing from 10 to 20 and 30 percent of this copies will help us and we observed dramatic improvements in the time of convergence of the problem. So we chose 30 percent for this amount to be copied between successive genes.

4.6 Local Optimum

- In order to escape from getting stuck in genes that are mostly similar to each other, we used some solutions, At first we used mutation which always get us the chance to generate new genes from genes we have with a little change, also we saved unique chromosomes in a generation preventing storing duplicate genes in the generation.