Artificial Intelligence: Shredded Document recoverer

Connor Li

Recently the world has been shocked by new accusations against the totally real company We’re Not A Scam Inc. A few clients have come forward with suspicious sounding stories and the authorities have taken interest. It looks like the people over at We’re Not A Scam Inc., may have, in fact, actually been scamming people. Recently a large collection of shredded documents has been recovered from We’re Not A Scam Inc. and the authorities are extremely interested in the contents of these documents. Since re-assembling shredded documents by hand is a time consuming and tedious task, the authorities are asking for assistance. A computer program that can help recover shredded documents is required.

**Goal:** Showcase your understanding of Genetic Algorithms (GA) and prepare a technical report exam- ining the algorithm’s performance in a given problem.

**Task:** This assignment has 3 parts. First, you must implement a GA system as described in lecture. Next, you must perform a number of experiments with your GA system and collect data regarding its perfor- mance. The ﬁnal step will be preparing a technical report to present your ﬁndings. Speciﬁc details regarding the requirements of the GA implementation, the experiments to be performed, and the format and content of the report follow.

As a reminder, the basic procedure of a GA is as follows:

Read problem instance data set GA parameters

generate a random initial population, POP, of size popSize

**for** gen=1 to MAXGEN **do**

evaluate ﬁtness of each individual in POP

select a new population using a selection strategy apply crossover and mutation

# end for

A GA could have the following components:

* **Inital Population Initializer:** Creates a population of size popSize of randomized individuals as described in class
* **Chromosome:** A chromosome encodes a solution to the problem being solved. In our case each chromosome must represent a sequence of distinct integers. The simplest representation to use is an integer array, mutation and crossover can then be performed on the elements of the array, although

other representations are possible. The array should contain each of the integers 0,1,2,....,*n* 1 in some order. No integer should be present more than once, and all integers from 0 up to and including *n* 1 should be included, where *n* is the size of the array.

*−*

*−*

* **Reproduction:** Use Tournament Selection (remember K = 2, 3, 4 or 5)
* **Crossover:** Given two individuals, a crossover creates two oﬀspring. Implement your GA using the following crossover strategies independently:
  + Order Crossover
  + A crossover of your choice For example: Uniform Order Crossover or Partially Mapped Crossover
* **Mutator:** Given an individual, a mutator creates a mutated individual. Implement your GA using a mutation operator of your choice (from those discussed in class)
* **Fitness evaluation function:** The ﬁtness function should take an individual and produce a real number describing the suitability of the solution encoded in the individual. In our case the ﬁtness function will attempt to describe how much an unshredded document looks like English. One possible ﬁtness evaluation function is provided, but feel free to experiment with other options. **With the provided ﬁtness function smaller numbers indicate a more ﬁt solution.**
* **Genetic algorithm system:** The implementation of the GA system. This ﬁle should glue together the various components of your system.
* **User parameters:** Population size, maximum generation span, probability of (crossover, mutation, etc)

Your GA program should permit the user to easily deﬁne his/her own genetic parameters and data (e.g., crossover rate, mutation rate, population size, maximum generation span etc. ).

# Using your GA implementation perform the following experimentation:

1. Run your GA to compare the performance of the two crossover operators mentioned above by using the following parameters (and include elitism in all cases):

a Crossover rate = 100 %, mutation = 0% b Crossover rate = 100 %, mutation = 10% c Crossover rate = 90 %, mutation 0%

1. Crossover rate = 90 %, mutation 10%
2. Determine your own best parameter settings

BONUS (For a bonus of 2% to your total course grade) Incorporate into your experiments your own innovative idea. This could be a diﬀerent initial population representation and creation strategy, a diﬀerent selection scheme, a diﬀerent (third) crossover not discussed in class, etc. Alternatively you could introduce a local search into your GA, or evaluate your own ﬁtness function.

* For each experiment mentioned above, run your GA at least 5 times with 5 diﬀerent random number seeds. Shredded documents will be provided. You should repeat the experiments above for document1- shredded.txt, document2-shredded.txt, and document3-shredded.txt
* For all three documents your chromosomes should have length 15.
* For each run your GA system should output the following to a ﬁle or standard out:
  + All GA parameters, including random number seed
  + Per each generation: best ﬁtness value, average population ﬁtness value
  + Per each run: best solution ﬁtness and its corresponding best solution chromosome
* Finally, prepare a summarized report of your ﬁndings using IEEE format introduced to you at the tuto- rial. IEEE format details are found at: [https://www.ieee.org/conferences/publishing/templates.](https://www.ieee.org/conferences/publishing/templates.html) [html](https://www.ieee.org/conferences/publishing/templates.html). The report should have the following sections and each section should address the listed points.
* An IEEE report in this format is expected to be 8 pages in length.
  + Introduction

∗ BRIEFLY introduce the concepts and topics discussed in the report.

∗ Precisely deﬁne the problem and explain why its solution is important.

* + Background

∗ This section should explain the algorithms used in the report(pseudo code is helpful) and may provide other information which you feel will be relevant to someone trying to understand your results.

* + Experimental Setup

∗ This section should provide enough information about your experiments to allow someone else to duplicate your results.

∗ This should include algorithm parameters used, the crossover and mutation operators used, and any other relevant implementation details.

* + Results.

∗ This section should summarize your ﬁndings.

∗ For your multiple runs compute the average of best ﬁtness per generation and average pop- ulation ﬁtness per generation. Using a graph drawing tool such as excel, plot well labeled graphs for your experiments. If you decided to do the bonus, plot graphs for it as well. The types of graphs you plot will depend on your incorporated idea, if any. Feel free to experiment with diﬀerent crossover and mutation rates. You will set your own Pop-Size and generation size.

∗ Also include summary tables describing the ﬁtness of your ﬁnal solution. Summary statistics such as min, max, mean, median, and standard deviation should be included in your tables. Tests for statistical signiﬁcance would also be appropriate, for example T-Tests or Mann- Whitney U tests.

∗ Explain your graphs/data in detail and emphasize the similarities and diﬀerences between diﬀerent algorithm conﬁgurations.

* + Discussions and Conclusions.

∗ This section should provide a BRIEF summary of what experiments you performed and the results you observed.

∗ Following this BRIEF summary, you should discuss your opinions regarding your results and what conclusions you’ve arrived at.

∗ This could include issues like which crossover performed better. If more than one mutation type was tried, which one performed better. If you included local search, did it help? How did the choice of GA parameters aﬀect the ﬁnal outcome etc?

* + References.

∗ List your sources here. The text of the report should contain references to your sources.

* This report is very important, so be sure to include it. Start early, gathering the data and doing the experimental analysis will take much more time than coding the assignment.