

APPLICATION OF GENETIC ALGORITHMS TO RECOVERING
CORRUPTED FILE STREAMS

by

John Phillips

A Thesis

Presented to the Faculty of
Bucknell University
In Partial Fulfillment of the Requirements for the Degree of
Bachelor of Science with Honors in Computer Science
17 October 2003

Approved: _____

Stephen Guattery
Thesis Advisor

Gary Haggard
Chair, Department of Computer Science

1. Introduction

During the spring semester of 2003 I participated in an elective course within the computer science department entitled “Introduction to Analysis of Algorithms”. The course, taught by Professor Stephen Guattery, allowed me to study both genetic algorithms (a process that attempts to generate near-optimal solutions to complex problems for which no efficient process of generating optimal solutions (answers) exists) and quantum cryptography (a process used for transmitting secure data through an encrypted data stream using quantum mechanical properties). Through my honors thesis, I plan to extend my research on genetic algorithms by investigating its limitations as a way of rebuilding corrupted character (text) files. I also hope that through this investigation I will be able to model partially decoded quantum-encrypted streams as corrupt character files in the hopes of using a genetic algorithm to decode the encryption.

1.1 Thesis Statement

This thesis entails formulating a genetic algorithm for recovering corrupted character data. In order to complete this thesis I will need to construct a model for corrupting text files, research various fitness functions (a measuring mechanism that is used to evaluate the quality of a solution (Man et al. 1999)) for implementation of the genetic algorithm, research quantum cryptography and simulate the process as applied to quantum communication channels.

2. Background

Whenever computer information, or data, is moved from one place to another it is transferred as a series of ones and zeros. Zeros and ones are encoded by different

physical representations. These currents and non-currents (ones and zeros) are commonly referred to as bits. Sometimes when a bit is being transferred it will be randomly corrupted, and received as the opposite value (i.e. a one changes to a zero). This leads to various problems in the computer world, corrupt files, static, etc. which are all very hard effects to reverse and put in the original order of the series of bits.

2.1 Explanation of Genetic Algorithms

Many problems that computer scientists encounter are very hard to solve. Some of these problems, commonly called NP-hard problems, a subset of which are NP-Complete problems, have no known efficient solution process (i.e. no algorithm that returns an solution in a time that is polynomial with respect to the size of the input). An example of such a problem is trying to develop an itinerary for a traveling salesperson where the goal is to visit each city on a predefined list, once and only once, while traveling the shortest distance possible. This problem is commonly referred to as the TRAVELING SALESMAN PROBLEM (TSP). TSP is an optimization problem. While any ordering of the cities is a possible solution, there will only be a few solutions that have the shortest distance possible. The question, then, is not what a solution to the problem is, but what the optimal solution to the problem is.

While no efficient process exists for generating an optimal solution all of the time to an NP-hard problem, many NP-hard problems can be solved efficiently much of the time to near optimality using a heuristic. A heuristic is a solution-generating process that gives near optimal solutions a high percentage of the time. However there is no guarantee that a heuristic will ever give a near-optimal solution at all. Genetic Algorithms are heuristic algorithms that use the idea of genetic recombination and

mutation to produce ‘populations’ of solutions to a problem that ‘mate’ with each other to produce new (and over a sequence of generations) better solutions. By defining the characteristics of what a good answer should look like (i.e. shorter travel path for TSP) genetic algorithms can use a ‘fitness test’ for the ‘population’ of answers and effectively kill off genetically less fit solutions. Over time, the genetic algorithm will produce a few exceptionally fit individual solutions within the population. These individual solutions would be highly, but not completely, optimized.

2.2 Explanation of Quantum Cryptography

One possible application that I hope to test of a genetic algorithm for rebuilding corrupted character files is in deciphering a quantum encryption transmission. Quantum cryptography, an encryption process that I first learned about in “Introduction to Analysis of Algorithms”, relies on the exploitation of a quantum mechanical phenomenon such as the uncertainty principle or the violation of a Bell inequality to encode data in a way that is statistically very secure from eavesdroppers, or unwanted listeners, on the channel (Barnett et al. 1993). Quantum encryption uses light photons. Light photons, like other traveling particles or waves, can be observed to determine their properties. The wavelength, amplitude, and intensity of a photon of light can all be measured without changing the photon. However it is theoretically impossible to measure the polarization of a light photon without the possibility of changing the polarization of the photon. Quantum cryptography uses a random pattern to send and receive light photons to develop an ‘encryption key’ between a sender and a receiver. Once in place, the encryption key can be used to pass information from the sender to the receiver with an incredibly small chance of being successfully eavesdropped upon by a third party.

3. Hypotheses and Methodology

Data corruption is a significant problem; whether it is choppy cell phone signals, a demagnetized sector on a hard drive or random noise on a transmission channel, data loss causes undue harm where there ideally should be none. This corruption and noise occurs randomly and in most transmission avenues (Horowitz 1994). In general there are two main ways to reduce the negative effect of data corruption. The first, and most often pursued, is to improve the technology used to transfer or store the data. Thus, by losing less data to corruption there is a smaller chance significant data loss. One example of this is error- correcting code. The second way to reduce the negative effect is to produce better methods of rebuilding the data files that have been corrupted. Much like the fragments of an antiquated document, corrupted data files often have little resemblance to what they originally looked like and require educated guesses in order to compose the pieces into an order that resembles the original.

I propose to use a genetic algorithm to reconstruct corrupt character files to the point that they highly resemble of the original file. I will accomplish this by testing a variety of different fitness tests on the corrupt character files.

3.1 Simulation

First, in order to complete my thesis, I will need to write a program capable of randomly corrupting text files. I plan on having all character (text) files being in ASCII, which is congruent with common practice. ASCII is a binary encoding scheme that represents all characters as 8-bit sequences and the UNIX operating system keeps all text files in ASCII. The program should take two input parameters: a text file and a

percentage (X%). Once finished, the program should return the text file with X% of the file's bits randomly corrupted. The program will corrupt bits by switching a 0 to a 1, and a 1 to a 0. The construction of this program should not present too much of a problem.

3.2 Using a genetic algorithm

In constructing a genetic algorithm to tackle this problem I will need to consider a variety of possible 'fitness tests'. As discussed above, a 'fitness test' defines the direction in which a population of answers will evolve and is therefore the most important part of a genetic algorithm. For the problem of corrupted character files I will test a variety of fitness tests.

- text-to-space ratios: Identify average space/non-space ratio
- average sentence length: Identify the average length of a sentence
- letter frequency: Which letters occur most frequently in the English language, weight mutation to more common letters. (The most frequent letters, in order, are e-t-a-o-n-i-s-r-h-l-d-c-u (Crystal 1987) (Singh 1999))
- word frequency: find out which words occur most in the English language and actively look for corrupted versions of those words. (Some of the more frequent include: the-of- to-in-and-a-for-was(Crystal 1987))
- word recognition: develop a dictionary to check words against
- grammar check (if time allows): make sure that the words developed by the genetic algorithm make sense grammatically with each other

Completing a successful genetic algorithm will involve a combination of these fitness tests and other mutational guidelines that will aid in producing beneficial mutations. Some mutational guidelines that I am considering are:

- letter patterns: break English language into common letter groups (i.e. 'qu')
- word composition: dynamically weight mutational percentages with respect to common syllables, prefixes, and suffixes
- ASCII character patterns: observe patterns in ASCII that will allow quick reconstruction of corrupt characters

The writing and optimizing of this genetic algorithm will be the most time intensive part of the project.

3.3 Application to Quantum Cryptography

In order to test this genetic algorithm on a quantum encryption channel I will have to define a quantum encryption protocol to test it on. This will require research into quantum cryptography, and will also require devising a quantum transmission protocol. A quantum transmission protocol is a way of applying a quantum cryptographic key in the use of transmitting files. Once I have devised a protocol I will need to write a program that is able simulate a quantum transmission. Finally, I will need to simulate eavesdropping on a quantum encryption channel, and show the result looks like a corrupt text file.

4. Project Significance

Data corruption is a large problem in the computing world. Currently there are a variety of error-correcting codes that are used on the front end of transmissions (e.g. spellcheckers, Huffman codes (Cormen et al. 2001)). My project will provide an alternative to error-correcting codes by being a tool whose application is on the receiving

end of a transmission. This will benefit all areas where corruption of text files is a problem, from storage to transmission. However, the scope of a genetic algorithm that could solve the data corruption problem would not be strictly limited to text correcting. In my thesis, I hope to show that successful genetic algorithm, capable of solving the corrupted data problem, would not only be applicable to corrupt data, but also quantum cryptography. Moreover it will have broader applicability as it extends research between genetic algorithms and text files.

Bibliography

- Barnett, S. M Huttner, B and Phoenix, S. J. D., “Eavesdropping strategies and rejected data protocols in quantum cryptography”, *Journal of Modern Optics*, vol. 40, no.12, December 1993, pp. 2501 – 2513.
- Cormen, T.H., C.E. Leiserson, R.L. Rivest, C. Stein. 2001, c1990. *Introduction to Algorithms Second Edition*. Cambridge, Massachusetts. The MIT Press.
- Crystal, David, 1987. *The Cambridge Encyclopedia of Language*. New York, NY. Cambridge University Press.
- Ekert, A. K., Huttner, B., Palma, G. M. and Peres, A., “Eavesdropping on quantum crptosystems”, *Physical Review A*, submitted, vol. 50, no. 2, August 1004, pp. 1047-1056.
- Horowitz, Paul. 1994, c1989. *The Art of Electronics*. New York, NY. Cambridge University Press.
- Man, K.F., K.S. Tang, and S. Kwong. 1999. *Genetic Algorithms*. London, England. Springer-Verlag London Limited.
- Singh, Simon. 1999. *The Code Book*. New York, NY. Anchor Books.