"Al. I. CUZA" University of IAȘI
Faculty of Computer Science

Nature-inspired Computation.
Aplications in Bioinformatics and Cryptography

PhD Thesis

Gabriel NEGARĂ
Supervisor: Profesor Henri LUCHIAN, PhD

SHORT DESCRIPTION

The 1st part of the thesis presents aspects related to **Evolutionary Computation** and metaheuristics for solving computational problems. Some related work in the field of **Ant Colony Optimization**, developed at Faculty of Computer Science of IAȘI (2004-2006), is summarized in this section.
Nature-inspired computation represents a widely known and used approach to solve difficult computational problems. The mathematical formalization and the deterministic, step-by-step strategies applied to specific hard problem needs often to be combined and enforced with evolutive-like techniques, in order to achieve a better dynamics when searching through the solutions' space. Ant Colony Optimization represents such a technique, with applications in both research and real-life scenarios.

The 2nd part of the thesis presents a short overview on **Bioinformatics** and **Computational Biology** and the work developed in the field of **Protein Folding**, as a Visiting Researcher at Brown University, Center for Computational and Molecular Biology (2006-2007).
The advances in the field of Bioinformatics in the last decades are impressive. Research centers and laboratories world-wide performed difficult tasks such as human genome assembly or genes classification. This led to important discoveries in the health sector. Despite the computational power available when approaching difficult bioinformatics

problems, improvements need to be made in the field of algorithmics and computational techniques. Protein structure prediction represents a key problem in Bioinformatics; the computer science modeling of the protein folding may be a decisive factor for understanding the behavior and the life cycle of the proteins.

The 3rd part of the thesis refers to **Cryptography** and to some more recent author's work (2009-2012).

Cryptography, viewed as a science defined by various branches, represents an intersection of multiple domains: mathematics, computer science, engineering, physics, linguistics etc. and even history. Numerous forms of codified or hidden communications exist from centuries. Cryptographers and cryptanalysts nowadays are studying ciphers, starting from very simple, basic ones, such as the Cesar's cipher, and going to modern cipher algorithms that became international standards.

From sender to receiver, ciphered data need communication channels: the physical channel that is the physical components bounded together in order to transmit and receive data, and the logical channel, that is the protocols, the abstract model and the systems that ensures the meaning of communications.

Nowadays, the need for secure communications involves more and more complex tasks – securing huge networks communications, securing very large amounts of business data etc.

Designing and implementing a secure system is a very difficult task. The cryptographic core of such a system implies strong cryptographic/cryptanalytic knowledge, in order to design resistant cryptographic primitive, algorithms, protocols and also, proving their security limits, in a rigorous, coherent and consistent manner.

The complexity of the calculus involved in both cryptanalysis and cryptography and the statistical nature of various cryptographic systems (randomness, variables etc.) lead to the necessity of combining the mathematical, pure deterministic-like approaches for solving different types of cryptography problems or breaking various cryptanalysis systems with more practical, accessible approaches, based on paradigms like evolutionary computing. Like in real life scenarios, sometimes a problem is solved by a "lucky" conjecture of facts; trying to take into consideration all the variables, the facts, and using deterministic strategies, based only on previous successes could take you far away from the moment of solving that problem.

This work proposes a few evolutionary approaches for cryptography and cryptanalysis. The first approach applies for the cryptanalysis of "Playfair", a classical cipher system; the second treats the field of cryptographically strong "S-boxes" design, as core entities for building secure modern cryptographic algorithms; the third approach enters the

domain of so-called “Post-Quantum Cryptography” - specifically, an evolutionary algorithm for the “Shortest Vector Problem”, one of the main difficult cryptographic problems used as the core of lattice-based cryptography.

A MORE DETAILED DESCRIPTION

Chapter 1

The Graph Coloring Problem

The 1st paper of the author published in the field of Ant Colony Optimization was “*Experience-based Ant Coloring (EAC). A new ant-like graph-coloring algorithm*”**.** In this paper we introduced a new ant-like algorithm for the graph coloring problem, based on an ant system meta-heuristic. EAC (Experience-based Ant Coloring) uses an ant system paradigm in which a number of agents (ants) perform specific colorings during a number of iterations. The colorings are performed using algorithms adapted to the ant system paradigm and are based on the experience from the previous iterations. After each iteration, the agents update the system memory. EAC introduces a new type of graph adjacency matrix and uses a new ant system approach for the graph coloring problem. The implementation of the algorithm was tested on graphs from the DIMACS Computational Challenge, obtaining encouraging results.

The 2nd paper of the author was “*Parallel Experience-based Ant Coloring*”, describing a parallel implementation of a new ant-like algorithm for the graph-coloring problem. PEBAC (Parallel Experience-based Ant Coloring) is based on an ant system meta-heuristic; it uses the ant colony system concept combined with experience based coloring techniques. A colony of ants collaborates for the goal of finding colorings and optimizing the quality of the solution. The most important factors are the *S-memory* (experience) and the parallel nature of the approach. The previous actions of the agents have an important role for the behavior of the agents in the future (the next iterations of the algorithm). *PEBAC* is a parallel ant-like algorithm that uses many specific coloring methods; the colorings are performed using threads (ants working in parallel), with the

goal of converging to the final solution. *PEBAC* is an approximate, probabilistic algorithm, solving the coloring problem in polynomial time. The quality of the solutions depends on parameters value, due to the probabilistic nature of the algorithm. A set of transformations of the input graph provides the number of algorithm iterations. The agents perform specific graph colorings during the iterations and update the system memory. To implement the ants work, we use a number of threads. The agents update the system memory and the graph structure changes appropriately. Algorithm ends when graph becomes a clique, the order of the clique representing the coloring number found by the algorithm. The algorithm was tested on graphs from the DIMACS Computational Challenge, obtaining good results. Based on the results we can say that *PEBAC*, a new ant system-based approach, represents a feasible approach for solving the graph-coloring problem.

Another two papers we published in the field were "*Experience-based Parallel Graph Coloring*" and "*A Shared Experience Algorithm for Graph Coloring*".

The results of our research are cited in the recent literature, for example in ***"****Artificial Ants in the Real World: Solving On-line Problems Using Ant Colony Optimization, Swarm Intelligence: Focus on Ant and Particle Swarm Optimization****",*** Itech Education and Publishing, Vienna, Austria (2007).

The Travelling Salesman Problem

In the paper "Solving *Optimization Problems using an ACS-based Approach*", we introduced a technique based on Ant Colony System, in order to solve the Travelling Salesman problem.

The approach we proposed uses an extra-exploration phase in order to improve the quality of the solutions. In the algorithm two ant colonies explore and exploit the searching space in order to find solutions. The first colony is called exploring colony. Its main goal is to generate partial solutions by imposing strategies for choosing the next steps when exploring the solutions space. The second colony, the exploiting colony, finds the best solution combining its own experience with information provided by the exploring colony. The technique is applied for solving the Travelling Salesman problem. The solutions are improved using 2-opt and 3-opt heuristics.

The experiments on problems from TSPLIB, the Traveling Salesman Problem Library, show encouraging results. The technique could be applied for solving routing, scheduling and other types of optimization problems.

Chapter 2

The goal of the Protein Folding project at Brown was to study different techniques and algorithms and to implement a protein folding framework for creating, manipulating and solving instances of the problem under various approaches.

The list of software applications implemented by the author is the following:

- *“proF2D”* – applications for the HP 2D model
  - *”2DFoldOverNight“* – a console application for running tasks using different folding algorithms, like Monte-Carlo simulations;
  - *“proF2DSquare”* – visual applications for folding under the 2D Square model;
  - *“proF2DTriangle”* – visual applications for folding under the 2D Triangular model;
- *“proF3D”* – applications for the HP 3D model
  - *“3DFoldOverNight”* - a console application for running tasks using different folding algorithms, like Monte-Carlo simulations;
  - *“3DSimpleCubic”* - visual applications for folding under the 3D Simple Cubic model;
  - *“3DBodyCenteredCubic”* - visual applications for folding under the 3D Body Centered Cubic model;
  - *“3DFaceCenteredCubic”* - visual applications for folding under the 3D Face Centered Cubic model;
  - *“3DFCCSideChains”* - visual applications for folding under the 3D Face Centered Cubic model with side chains.

The applications were implemented using *Microsoft Visual Studio 2005* and the C# language; for the 3D protein folding models I used the *OpenGL Visual Library*.

In the same time, this part of author research led to the writing of a technical report and a protein folding survey, both for the Protein Folding project at Brown, Center for Computational Molecular Biology.

Chapter 3

The algorithm we proposed demonstrates the suitability of evolutionary approaches for classical substitution ciphers like Playfair.

The results and the algorithm efficiency are influenced by the genetic operators, the parameters settings, the fitness function and the enciphered text length. The fitness values are likely to decrease more when the algorithm is running on longer input enciphered texts (the deciphered texts analyzed are “closer”, from the letters and bigrams frequencies point of view, to real English texts). The algorithm could also be used in the hypothesis of knowing one or a few words from the plaintext, in which case the fitness function is changed appropriately or using stopping conditions based on these words.

Similar results are obtained if using a different book for computing the frequency tables. The values of frequency tables are close enough when using as input different English text books of relatively large size.

Generic evolutionary schemes such the one used for the GAPFC algorithm are potentially useful tools in analyzing and solving both cryptanalysis and cryptographic problems.

This part of the author’s research work is synthesized in the paper “*An Evolutionary Approach for the Playfair Cipher Cryptanalysis*“.

This research work was strongly related to the author’s software implementation - a visual aplication built using Microsoft Visual Studio 2010 and the C# language. The application, named *“PlayFair”*, allows the user to encipher and decipher manually entered plaintexts, using keys of various lengths. The cryptanalytic section of the application allows the running of the GAPFC algorithm and of a preliminary algorithm’s version, under various parameters settings. After the algorithm’s running, resulting data are displayed on the application’s form and also saved in file logs.

Chapter 4

Designing cryptographically strong S-boxes remains an open problem, from some points of view. It is difficult to characterize and categorize S-boxes. When looking at

design criteria there is always a compromise: satisfying some specific criteria can lead to the insatisfiability of another. Implementation issues must be taken also in considerations – structural particularities of an S-box may lead or not to efficient implementation.

Varying the random decisions in the algorithm, as well as the primitive polynomials used in the component operations could lead to better results, that is s-boxes with better properties.

The primitive polynomials can be replaced or combined with irreducible polynomials; the presented criteria represent just a part of the analysis required for designing cryptographically good s-boxes; such s-boxes could be instead used in algorithms’ analysis, as parts of cryptographic-statistical tool or methods (for example, the properties of relatively large sets of such s-boxes can be used as a measure when analyzing s-boxes used in the analyzed algorithms).

The author’s work related to s-boxes generation led to the implementation of a visual software application, using Microsoft Visual Studio 2010 and the C# language. The application was called *“SboxGen”*, allowing the user to evaluate the cryptographic properties of s-boxes generated at random or using the proposed randomized scheme.

As future work, studying the suitability of applying genetic programming approaches for s-boxes generation could be an interesting research field.

Chapter 5

We proposed a genetic algorithm for the Shortest Vector Problem, which uses genetic operators relying on basic integer numbers operations.

Generic evolutionary schemes such the one used for the GASVP algorithm are useful tools for both cryptanalysis and cryptographic problems.

As future work, the algorithm performances must be improved in order to find better solutions for larger, random bases. Efforts need to be made for ensuring a better individual evaluation and selection and in order to perform a more detailed analysis of the crossover and mutation operators.

The results suggest that the evolutionary approach for the SVP problem is worth further consideration.

The list of software applications implemented (using Microsoft Visual Studio 2010 and the C# language) or adapted for use by the author is the following:

- *“Lattices”* – the main visual application, implementing the GASVP algorithm; the user is allowed to specify the input base, having various sizes, and to run the GASVP algorithms (and other *4* preliminary versions) under specified parameters settings; the resulting data is displayed on the application form or saved in log files;
- *“LatticesGenGN”* – console application integrating the NTL Library, used for the generation of the GASVP algorithm’s input benchmarks – randomly generated bases of various sizes;
- *“LLL_NtlTest”* – console application integrating the NTL Library, used for running the LLL algorithm;
- *“NtlProj”* – console application used for building the NTL Library, in order to integrate it in the *3* applications above.