Analysis of a Genetic Algorithm-based Approach in the Optimization of the SourceAFIS's Matching Algorithm

A. G. A. Silva[1], I. A. Almeida Junior[1], Rodrigo L. Parente[1], L. V. Batista[1], João J. B. Primo[2], Adriano S. Marinho[2], Pedro Alves[2]
[1]Informatics Center, Federal University of Paraíba, João Pessoa, Paraíba, Brazil
[2]Research Department, VSoft Tecnologia LTDA, João Pessoa, Paraíba, Brazil

Abstract— *Fingerprints are an important biometric trait, being the most widely deployed biometric characteristic. Minutiae-based methods are the most popular technique to compare fingerprints, but as the pairing is rarely perfect the score between two fingerprints may also depend on another factor. Usually, a constant weight is computed either empirically or statically and assigned for each factor. Optimize these weights can be a hard task for a large N-dimensional attributes space. Genetic Algorithms (GA) is an optimization approach based on Darwin's theory of evolution of species and it has been proved to be quite successful in finding good solutions to such complex problems. In this work, we analyze an GA-based optimization approach for SourceAFIS's fingerprint authentication algorithm. The DB2 fingerprint database from FVC 2006 project was used to validate our method. As best result, an EER of 0.379% was achieved.*

Keywords: fingerprint, matching, scoring, genetic algorithm, optimization

1. Introduction

Reliable identification systems have become a key issue for applications that authenticate users. Traditional methods to establish user's identity include mechanisms based upon knowledge (e.g., passwords) or tokens (e.g., identification cards). However, such mechanisms may be lost, stolen or even manipulated to spoof the system. In such a context, biometrics rises as an alternative. [16].

Biometrics (biometric recognition) refers to the use of distinctive anatomical and behavioral characteristics such as fingerprints, iris, face and voice for automatically recognizing a person. Biometrics provides better security, higher efficiency, and, in many instances, increased user convenience. Because biometric identifiers cannot be easily misplaced, forged, or shared, they are considered more reliable for person recognition than a traditional token or knowledge-based methods. Thus, biometric recognition systems are increasingly being deployed in a large number of government and civilian applications [13].

Because fingerprints are unique to individuals, invariant to age, and convenient in practice [12], they are the most widely deployed biometric characteristics. Fingerprint are used in every forensics and law enforcement agency worldwide routinely [13].

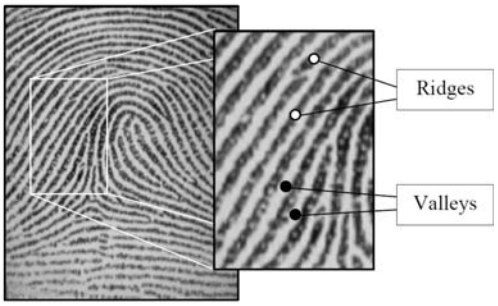

Fig. 1: Ridges and Valleys

A fingerprint image consists of interleaving ridges and valleys. Usually, interleaving ridges are dark lines and valleys are the white lines between ridges (see Figure 1). Ridge terminations and bifurcations are kinds of minutiae which are characteristic features of fingerprints [19]. Usually, minutiae have localization and direction as attributes. Most Automated Fingerprint Identification Systems (AFIS) are based on minutiae.

Fingerprint images usually need to be segmented in background and foreground - where the foreground is the Region of Interest (ROI). The segmentation removes uninterested regions before some other steps such as enhancement and minutiae detection. Hence, the image processing will consume less CPU time and avoid undesired errors as detection of spurious minutiae in low-quality image regions.

A matching algorithm is used to compare two given fingerprints. In case of identification, the algorithm returns a similarity level between fingerprints. In authentication, on the other hand, an answer indicating if two fingerprints belong or not to the same person is returned. Usually, a threshold value previously defined is used to answer it.

The most popular and widely used technique to compare fingerprints is based on minutiae. Minutiae are extracted from the two fingerprints and stored as sets of points in the two-dimensional plane. Minutiae-based matching essentially consists of finding the alignment between fingerprint's minutiae sets that results in the maximum number of minutiae

pairings. [13].

The pairing of minutiae in most cases generates a score value. Because the pairing is rarely perfect, this score may also depend on other factors like minutiae type, minutiae position, minutiae location and minutiae direction error. Usually, a constant weight which is computed either empirically or statically is assigned for each factor. Optimize these weights can be a hard task for a large N-dimensional attributes space.

Genetic Algorithms (GA) are an optimization approach based on the principle of natural selection of Charles Darwin. These algorithms input is an N-dimensional vector that will be optimized according to a fitness function. GAs proved to be quite successful in finding good solutions to such complex problems as the traveling salesman, the knapsack problem, large scheduling problems and others [4].

In this work, our goal is to analyze the optimization gain performed by a GA approach in the fingerprint matching algorithm SourceAFIS. The database DB2 of FVC 2006 project [5] were used to validate our study.

2. Related Works

In this section, we list some related works that may be useful to the reader. First, is appropriate to cite the work of John Holland [9] who was the first scientist to describe evolutionary algorithms, during the 60s. Such work provides a good background about Holland's goal in understanding the life adaptation as like it occurs in nature and the ways of developing systems based on these principles.

In order to a good theoretical foundation on evolutionary algorithms Back et al. [1] provides an overview of the three main branches of evolutionary algorithms (EA): evolution strategies, evolutionary programming, and genetic algorithms. In their work, certain characteristic components of EAs are considered: the representation scheme of object variables, mutation, recombination, and selection operators.

Considering the importance of parameter optimization on biometric systems, the work by Goranin et al. [7] analyzes GA application in that context. According to this paper, the use of evolutionary algorithms may ensure a qualitative increase of biometric system parameters, such as speed, error rate, and flexibility.

With regard to fingerprint matching optimization, Jiang and Yau [10] use the local and global structures of minutiae in their approach. The local structure of a minutia describes a rotation and translation invariant feature of the minutia in its neighborhood while the global structure tries to determine the uniqueness of a fingerprint.

The matching algorithm proposed by [11] uses triangular matching to deal with the deformations of fingerprints. A Dynamic Time Warping (DWT) is used to validate the final results of fingerprint matching. Without DWT, the results are not acceptable though.

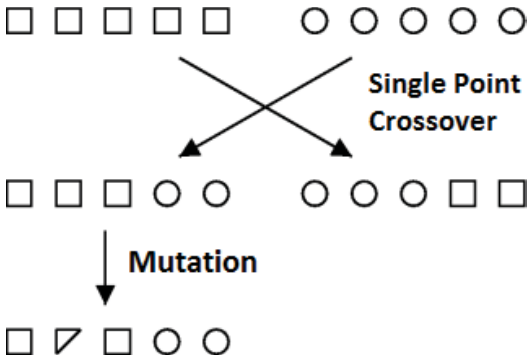

Fig. 2: Example of Crossover and Mutation operators. Where the circles represents one type of genes and the squares another one. Note that after mutation operation a gene mutates.

The closest work in comparison with ours is the work of Scheidat et al. [17]. Such work suggests another good solution for parameter optimization in the biometrics area. They planned their application in some way that it can be used without great effort by other biometric systems, even firstly developed for fingerprint recognition. All databases from FVC projects of the years 2000, 2002 and 2004 were used to generate the results. In the best case, a relative improvement of 40% in EER was obtained for database 1 of FVC 2000.

3. Genetic Algorithms

Genetic algorithms were proposed by [8] as a tool to find solutions to optimization problems in poorly understood large spaces. They are based on the genetic processes of biological organisms, especially on the principle of natural selection by Charles Darwin [3]. Although, this slogan seems to be slightly tautological in the natural environment, where fitness is defined as the ability to survive, it makes good sense in the world of optimization problems where fitness of a string is given as the value of the function to be optimized at the argument encoded by the string.

Typically, a genetic algorithm works on a population of individuals. Each individual is represented by one chromosome formed by a set of genes representing the parameters to be optimized. Some operations are realized in order to produce new generations of individuals based on their capability to generate good results: crossover, selection and mutation.

The crossover is the key operator to generate new individuals in the population. Inspired by the example of nature, crossover is intended to join the genetic material of chromosomes with a high fitness in order to produce even better individuals.

The selection operator is intended to implement the idea of "survival of the fittest". It basically determines which of the

chromosomes in the current population is allowed to inherit their genetic material to the next generation.

The mutation operator should allow the GA to find solutions which contain genes values that are non-existent in the initial population. The parameter governing this operator is called mutation probability. Whereas the selection operator reduces the diversity in the population, the mutation operator increases it again. The higher the mutation probability, the smaller is the danger of premature convergence. A high mutation probability, however, transforms a GA into a pure random search algorithm, which is of course not the intention of this.

Let *P* be a random population of *N* chromossomes (x_1, x_2, ..., x_n) and *f(x)* a fitness function. The following pseudocode describes the steps of genetic algorithms.

1) Create a random population *P* of *N* chromosomes (candidate solutions for the problem).
2) Evaluate *f(x)* of each chromossome *x* in the population.
3) Generate a new population by repeating the following steps until the new population reaches population N:
   a) Select two parent chromosomes from the population, giving preference to highly fit chromosomes (high *f(x)* values). Automatically copy the fittest chromosome to the next generation.
   b) With a given crossover probability, crossover the parent chromosomes to form two new offspring. If no crossover was performed, offspring is an exact copy of parents.
   c) With a given mutation probability, randomly swap two genes in the offspring.
   d) Copy the new offspring into a new population.
4) Copy the newly generated population over the previous (existing) population.
5) If the loop termination condition is satisfied, then stop and return the best solution in current population. Otherwise, go to Step 2.

4. Proposed Technique

In this paper, we analyze the improvement in parameters optimization of SourceAFIS's fingerprint matching using genetic algorithms. The DB2 database of FVC2006 [2] project from Biolab is used to test and validate the proposed method. It has 1680 samples of 140 persons with 12 samples/person. The experiments were carried out on a computer with an I7 Intel Quad-Core processor(3.2 GHz) and 8GB of RAM.

4.1 Matching Algorithm

The matching algorithm used in this work is based on the algorithm proposed by Robert Vazan in the open source project SourceAFIS [18].

The matching algorithm of SourceAFIS also uses a minutiae-based method. However, the comparison is not directly based on coordinates, types, and direction. It is based on a set of vectors generated by the connection between minutiae and their neighbors. It ignores position differences and possible rotations caused by failures on capture.

Fig. 3: Example of k segments created for a minutia and its neighboors. The enumerated minutiae are divided by types, where blues are endings and reds are bifurcations.

Let (M_1, M_2, M_3, ..., M_n) be the set of *n* minutiae of template A. For each minutia, the algorithm will search the *k* nearest neighbors and make vectors to each of them, where k is a parameter of the system (see Figure 3). These vectors will, in general, represent a unique relation between two minutiae and a more reliable information than a simple minutiae location. It may decrease False Positive Rates.

Apart from the length and angle information of each vector, the following properties are also computed to enlarge the relationship between the vectors and its minutiae:

- *Reference Angle*: stores the angle difference between reference minutia, from where the vectors depart, and the vector angle.
- *NeighborAngle*: similarly to Reference Angle, this attribute stores the difference between the neighbor minutiae direction, to where the vector arrive, and the opposite of the vector angle.

These relations between minutae directions and vector angle are important to improve the reliability of matching. This algorithm doesn't store any information about the location of the minutiae. It happens because the algorithm ignores the position of founded pattern.

After all vectors constructions, all information about created segments are stored in a matrix $M[n, k]$. On the second step of the algorithm, each generated segment of a template B will search a correspondent segment on template A matrix. The comparison between two vectors is made by computing the subtraction of vectors length, References Angle, and Neighbors Angle. The segments are matched if these values are lower than a defined angle and size thresholds.

When a correspondent vector of template A is found on template B, the algorithm realizes that the minutiae from where the vectors depart, on each template, are corresponding (see Figure 4). Based on this premise, it's possible

Fig. 4: Example of corresponding minutiae and it's neighbors segments from different fingerprints.

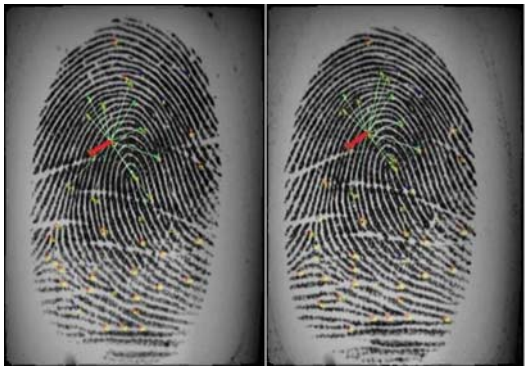

Fig. 5: Example of corresponding segment on a probably region intersection.

the minutiae in neighborhood are also able to find their corresponding because this region is candidate to be an intersection region - Figure 5 shows an example of region intersection. Thus, the paired minutiae of template B searches on its neighbors segments if any of them has correspondence with the neighbor segments of template A's paired minutiae.

When paired neighbors are found, the cycle of searching for neighbors restarts. Each newly paired minutiae searches for corresponding vectors on its neighbors. This recursive cycle continues until the minutiae graph reaches the intersection borders between the two samples, where there will be no corresponding minutiae in both templates. The graph tends to be significantly smaller in fingerprint pairs of different individuals than in pairs of the same individual.

After all steps, a list of paired minutiae between templates A and B is obtained. Figure 6 shows the best graph generated by matching between two templates. So many information can be inferred from this list, such as percentage of paired minutiae about the total of minutiae of each template; amount of equal types, since the algorithm does not consider the minutia types for graph construction; sum of the distance errors between all vectors, and many others. Such information may be used to establish a matching score for each primer pair.

4.2 Genetic Algorithm

The Optimera library [6] developed by Chas Egan was chosen as framework to apply the Genetic Algorithm in this work. It's written in C# and supports multithread as well.

In the Fingerprint Matching algorithm used in this work, graphs initialized by different minutiae can produce distinct scores of matching. In order to let Genetic Algorithm decide the weight of used parameters, all possible matchings scores - computed by all possible graphs - were computed. The purpose of the genetic algorithm, in our work, is to optimize the weight of these used parameters.

The parameters used on each graph evaluation were:

- *Pair Count*: Represents the quantity of matched minutiae found.
- *Correct Type*: Quantity of matched minutiae with correct type (Ending or Bifurcation).
- *Supported Count*: Quantity of segments that found the same minutia as part of matched segment.
- *Edge Count Factor*: Quantity of matched segments.
- *Distance Accuracy*: This variable represents the sum of distance errors between all matched vectors. This value contributes negativaly.
- *Angle Accuracy Factor*: Represents the sum of angle errors between all matched vectors. This value contributes negatively.

5. Results and Discussions

To evaluate the matching algorithm, the DB2 database of FVC2006 Project was used. The FVC protocol [2] was adopted to compare SourceAFIS and the proposed method. Hence, both methods were compared through EER performance although FAR1000, FAR and FRR perfomances are also compared.

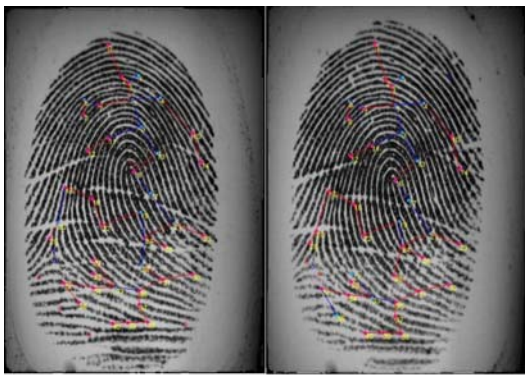

Fig. 6: Example of completed matching graph of corresponding segments.

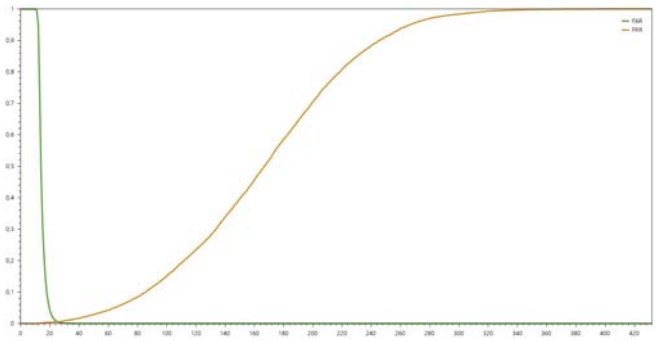

Fig. 7: FAR and FRR curves of SourceAFIS method.

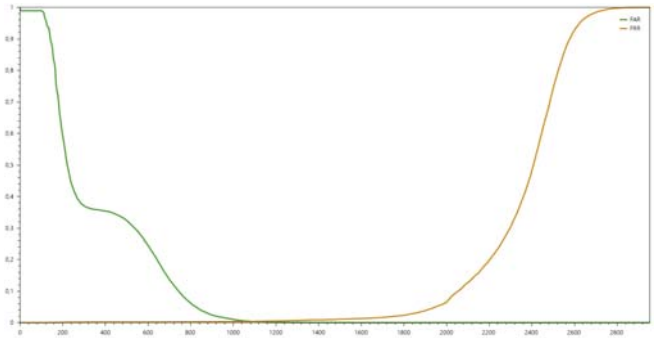

Fig. 8: FAR and FRR curves of SourceAFIS method using genetic algorithms.

Figures 7 and 8 show FAR and FRR curves of SourceAFIS method and the proposed one. A significant improvement can be observed in the ROC curve, shown in Figure 9.

Table 1 compares both performance rates. A relative performance improvement of at least 32% was achieved in all performance rates.

The best results were achieved with the following parameters to genetic algorithm: Population Size p = 120, CrossOver Rate c=0.8 and Mutation Rate m= 0.05. The genetic algorithm usually took 2000 generations to converge to same results.

Table 1: Comparison of performance rates between SourceAFIS and the proposed method.

	SourceAFIS	*Proposed Method*	*Improvement*
EER	**0.5592%**	**0.3798%**	**32.08%**
FAR1000	**1.0173%**	**0.6602%**	**35.10%**
FAR	**0.5447%**	**0.3699%**	**32.09%**
FRR	**0.5736%**	**0.3896%**	**32.07%**

6. Conclusion

Improvement of EER performance plays an important role in automatic fingerprint authentication. In this paper, a genetic algorithm based approach was used to optimize

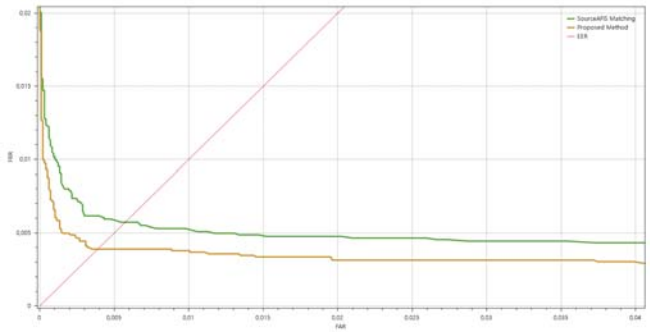

Fig. 9: Comparison of ROC curves. The green line represents SourceAFIS curve.

parameters of fingerprint matching algorithms. It has shown to be effective decreasing error rates of the basic algorithm, SourceAFIS, with a low cost process. An improvement of at least 32% was achieved in most used fingerprint authentication performance rates. The future research activity will be devoted to further improve the SourceAFIS algorithm, decreasing the EER performance and submit our algorithm to validation in FVC project.

References

[1] Thomas Bäck and Hans-Paul Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary computation*, 1(1):1–23, 1993.

[2] Raffaele Cappelli, Matteo Ferrara, Annalisa Franco, and Davide Maltoni. Fingerprint verification competition 2006. *Biometric Technology Today*, 15(7-8):7-9, 2007.

[3] Charles Darwin and Gillian Beer. *The origin of species*. Oxford University Press Oxford, 1951.

[4] Herbert Dawid. Genetic algorithms. In *Adaptive Learning by Genetic Algorithms*, volume 441 of *Lecture Notes in Economics and Mathematical Systems*, pages 37–60. Springer Berlin Heidelberg, 1996.

[5] Bernadette Dorizzi, Raffaele Cappelli, Matteo Ferrara, Dario Maio, Davide Maltoni, Nesma Houmani, Sonia Garcia-Salicetti, and Aurélien Mayoue. Fingerprint and on-line signature verification competitions at ICB 2009. In *Advances in Biometrics, Third International Conference, ICB 2009, Alghero, Italy, June 2-5, 2009. Proceedings*, pages 725–732, 2009.

[6] Chas Egan. Optimera - a multithreaded genetic algorithm library in $C\#$, April 2013.

[7] N Goranin and A Cenys. Evolutionary algorithms application analysis in biometric systems. *Journal of Engineering Science and Technology Review*, 3(1):70–79, 2010.

[8] John H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975. second edition, 1992.

[9] John Holland. Outline for a logical theory of adaptive systems. *Journal of the Association of Computing Machinery*, 3, 1962.

[10] Xudong Jiang and Wei-Yun Yau. Fingerprint minutiae matching based on the local and global structures. In *Pattern recognition, 2000. Proceedings. 15th international conference on*, volume 2, pages 1038–1041. IEEE, 2000.

[11] Zsolt Miklos Kovacs-Vajna. A fingerprint verification system based on triangular matching and dynamic time warping. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(11):1266–1276, 2000.

[12] Eryun Liu, Heng Zhao, Fangfei Guo, Jimin Liang, and Jie Tian. Fingerprint segmentation based on an adaboost classifier. *Frontiers of Computer Science in China*, 5(2):148–157, 2011.

[13] Davide Maltoni, Dario Maio, Anil K. Jain, and Salil Prabhakar. *Handbook of Fingerprint Recognition*. Springer Publishing Company, Incorporated, 2nd edition, 2009.

[14] Melanie Mitchell. Genetic algorithms: An overview. *Complexity*, 1(1):31–39, 1995.

[15] Fernández J.A. Prieto and Velasco J. R. Pérez. Adaptive genetic algorithm control parameter optimization to verify the network protocol performance. In *Proceedings of IPMU*, volume 08, pages 785–791, 2008.

[16] Arun A. Ross, Karthik Nandakumar, and Anil K. Jain. *Handbook of Multibiometrics (International Series on Biometrics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[17] Tobias Scheidat, Andreas Engel, and Claus Vielhauer. Parameter optimization for biometric fingerprint recognition using genetic algorithms. In *Proceedings of the 8th Workshop on Multimedia and Security*, pages 130–134. ACM, 2006.

[18] Robert Važan. Source AFIS project @ONLINE, June 2009.

[19] En Zhu, Jianping Yin, Chunfeng Hu, and Guomin Zhang. A systematic method for fingerprint ridge orientation estimation and image segmentation. *Pattern Recogn.*, 39(8):1452–1472, August 2006.