International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:1, No:9, 2007

Spread Spectrum Code Estimation by Genetic Algorithm

V. R. Asghari and M. Ardebilipour

***Abstract*—**In the context of spectrum surveillance, a method to recover the code of spread spectrum signal is presented, whereas the receiver has no knowledge of the transmitter's spreading sequence. The approach is based on a genetic algorithm (GA), which is forced to model the received signal. Genetic algorithms (GAs) are well known for their robustness in solving complex optimization problems. Experimental results show that the method provides a good estimation, even when the signal power is below the noise power.

***Keywords*—**Code estimation, genetic algorithms, spread spectrum.

I. Introduction

ALTHOUGH spread spectrum communications were initially developed for military applications, they are now widely used for commercial ones, especially for code division multiple access (CDMA), or global positioning systems (GPS) [1]. They are mainly used to transmit at low power without interfered by jamming, to others users or to multi path propagation. The spread spectrum techniques are useful for secure transmitter, because the receiver has to know the sequence used by the transmitter to recover the transmitter data [2]–[3].

Our purpose is to automatically determine the spreading sequence, whereas the receiver has no knowledge of the transmitter's code. The code estimation performance of the proposed algorithm is examined by computer simulations. The performance measure of interest in this paper is the mean-squared error (MSE) for the code estimation.

The paper is organized as follows. Section II describes the technique of direct sequence spread spectrum (DS-SS) and explains the difficulty to recover the data in an unfriendly context. Section III describes the system model used in this paper. Section IV describes the GAs used to implement our proposed code estimator. Our simulation results are presented in section V. Section VI concludes the paper.

II. DS-SS Technique

In order to spread the signal power over a broadband channel, the direct sequence spread spectrum (DS-SS) technique consists in multiplying the information signal with a periodic pseudo-noise sequence.

Let us consider $b(t)$ the information signal

$$b(t) = \sum_{n=-\infty}^{+\infty} b_n p(t - nT_b) \quad (1)$$

Where $b_n = \pm 1$ with equal probability and $p(t)$ is a rectangular pulse of duration T_b.

Let us note y, the PN sequence of length k,

$$y = y_0, y_1, \Lambda\ , y_{k-1} \quad (2)$$

The transmitter signal $\hat{y}_n$ is the product of both waveforms. If we consider a direct sequence spread spectrum system without noise,

$$\hat{y}_n = b_n y \quad (3)$$

We assume the receiver knows this sequence and can despread the signal using a correlator

$$\langle \hat{y}_n, y \rangle = \langle b_n y, y \rangle = b_n \langle y, y \rangle = b_n k \quad (4)$$

According to the properties of PN sequences [4], the data information is then recovered.

However it becomes more challenging when the receiver doesn't know exactly the code used by the transmitter.

Let us note $\tilde{y}$ a sequence similar to y, but not exactly the same. Then using a correlator with $\tilde{y}$, we get

$$\langle \hat{y}_n, \tilde{y} \rangle = \langle b_n y, \tilde{y} \rangle = b_n \langle y, \tilde{y} \rangle \quad (5)$$

According to the properties of PN sequence, $\langle y, \tilde{y} \rangle$ is low [4] and then we do not recover the data information.

III. System Description

Typically direct sequence spread spectrum systems use binary or quadrature phase shift keying (BPSK or QPSK) data modulation. Usually the PN sequence is a binary maximal length sequence or a Gold sequence [3].

Although our methods can estimate different PN sequences, but here we consider a BPSK data modulation, spread by a Gold sequence. The baseband channel noise is assumed to be white, Gaussian and centered.

An interesting method to estimate spreading code is proposed in [5]. It takes profit of blind identification techniques available for multiple FIR channels. In our method which is based on genetic algorithm, we use GA to estimate PN sequence.

IV. Genetic Algorithm Based Code Estimation

The efficiency of any global positioning technique can be measured in terms of two properties the so-called explorative property and the exploitative property [6]. Techniques that possess a high explorative property have a slower convergence rate and a higher computation complexity but they explore the entire space in order to locate the global optimum. Hence accuracy is always guaranteed. Techniques such as the family of hill-climbing methods possess a high exploitative property, and hence they offer fast convergence to an optimum of a given subspace. However, this optimum may not be the global optimum of the entire solution space.

GAs [7]–[8] constitute robust global search and optimization strategies that can strike an attractive balance between exploitation and exploration. These algorithms were introduced by Holland [7], and their principles are based on the concept of natural evolution. Specially, GAs use a population of candidate solutions initially distributed randomly over the entire solution space. Hence, GAs are highly explorative at the beginning. By evolving this population of candidate solution over successive iterations or generations, through probabilistic transition operations based on Darwinian survival of the fittest, the GA quickly identifies and exploits the subspaces, in which the global optimum may be located, while at the same time maintains the exploration of other parts of the solution space. Hence, while the optimum solution is not always located, the GA has a low probability of curtailing the exploration in suboptimal, rather than optimal solutions.

A. Initialization

Initialization of the GA is performed at the so-called ($g = 1$)st generation for each new signaling interval, as seen in Fig. 1, by creating p number of candidate solutions, or strings in GA parlance. The set of p strings is known as a population, and p is known as the population size. These strings represent the unknown variables of interest, which in this case are the estimated PN sequence. Hence, each string will contain k elements corresponding to the length of the PN sequence.

In order to attain a highly diversified search (exploration) at the beginning without knowing where the optimum solution may be located, it is desirable to distribute the candidate solutions randomly throughout the solution space. As seen in Fig. 1, the parameter t is associated with the GA generation corresponding to the termination of the search.

B. Evaluation

Associated with the p th combination string is a so-called figure of merit — more commonly known in GAs as the fitness value — which has to be evaluated, as seen in Fig. 1. The fitness value, denote by $f[\hat{y}_n, \tilde{y}_k]$ for $k = 1,\Lambda\ ,K$ is computed by substituting the elements of both the transmitted string and the k th candidate solution into the objective function or crosscorrelation of (5).

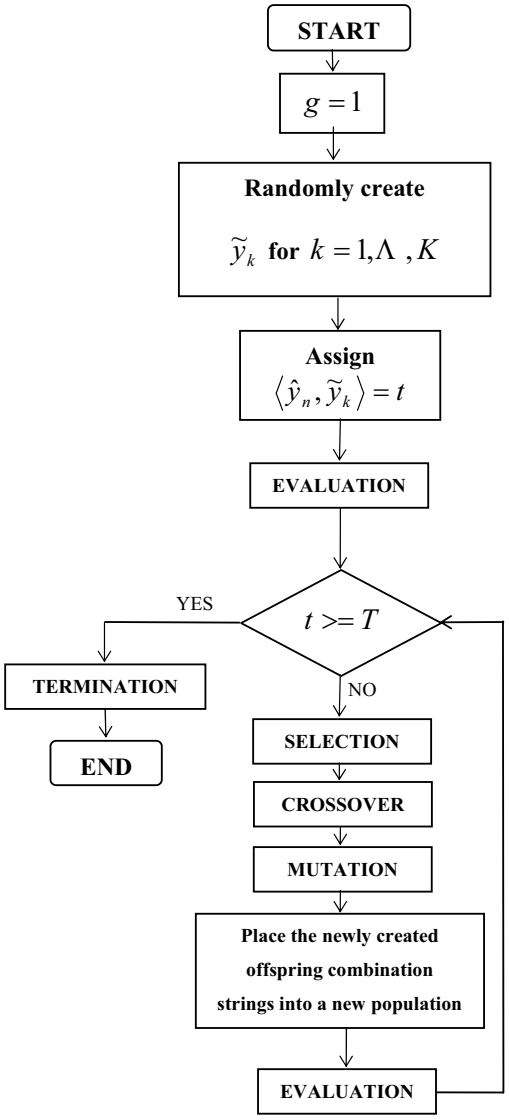

Fig. 1. Flowchart depicting the structure of the proposed genetic algorithm used to code estimation.

C. Selection

The exploitative property of GA is derived from two GA operations referred to as selection and crossover [9]. The crossover operation will be explained in the next subsection. Let us refer to the elements that constitute the optimal solution as good elements. Any other elements are referred to as bad elements. For example, if the optimal solution constitutes a string containing all $+1$ elements, then any $+1$ in a string will be a good element while any -1 in the string will be a bad element. Intuitively, strings having a high fitness in the sense of (5) will contain more good elements and hence should be exploited further. At the same time, strings having a low fitness value should be discarded. As shown in Fig. 1, following the evaluation, our population of elements is sorted according to their fitness value. Then, the strings which are located at the top level of sorted population will be used for

subsequent exploitation and exploration of the solution space.

D. Crossover

Crossover applies to one or more parents and exchange genetic elements (good or bad) with equal probability (p_c) between two different strings, the so-called parents in GA parlance [9]. This will produce two new strings, which are referred to as offspring. These offspring will constitute the new population of the next generation.

E. Mutation

Mutation involves the modification of the value of each solution string with some small probability. The role of mutation is to prevent the premature convergence of the GA to suboptimal solutions [9]. If all the strings in the generation have a bad element at the same location, then further crossover processes between any of these strings will not be able to remove the bad element. In order to skip from this situation, the mutation operation is used. In Fig. 1, the mutation operation refers to the alteration of the value of each element in the offspring with a probability denoted by p_m. In the case of the data string, the mutation process simply inverts the bit value of the element concerned from $+1$ to -1 or vice versa. Then these offspring are later made a new generation which can select as parents.

F. Termination

The GA can be terminated if there is no improvement in the maximum fitness value of the population after several iterations. This will ensure with a high probability that the global optimum is found at the expense of high computational complexity and long convergence time. In code estimation, it is more desirable to detect the code and also data, fast and at a low complexity. Hence, we terminate the GA of Fig. 1 after $\langle \hat{y}_n, \tilde{y}_k \rangle$ is coming upper than a threshold (T). By adjusting the value of T, the bit error rate (BER) performance of the GA-based code estimator can be controlled.

V. Simulation Results

In this section, our simulation results are presented in order to demonstrate the performance of the proposed code estimator. A summary of the various parameters that are used in our simulations is shown in Table I. The channel noise was assumed to be white, Gaussian, centered and real. The data rate (R_b) and the number of chips per bit (p) were assumed to be known by the receiver. The signature sequence was used with a processing gain of $p = 31$. Let us first evaluate the BER performance of the proposed code estimator.

In order to give an impression of how the GA manages to estimate the transmitted code over the course of evolution given a population of randomly generated possible solutions at the beginning, the BER performance of the estimator at each generation is shown in Fig. 2 at $\xi / N_0 = -5db$. As we have mentioned in section IV, the crossover operation will efficiently identify the areas in the solution space, where the optimal solution might be located. This can be seen from the first few generations. However, crossover alone will not find the optimal solution, when we do not use mutation. Hence, when crossover is used in conjunction with mutation, the BER performance is improved significantly, as shown in Fig. 2.

TABLE I
Summary of Various Parameters Used in Our Simulation

Symbol	Description
R_b	Data rate
p	Population size
p_c	Crossover probability
p_m	Mutation probability
g	Number of generation per signaling interval

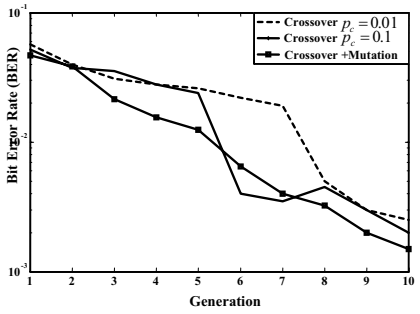

Fig. 2. BER performance of the proposed GA-based estimator as a function of the number of generations, where $p = 100$, $g = 10$, $SNR = -5dB$, and $p_m = 0.1$.

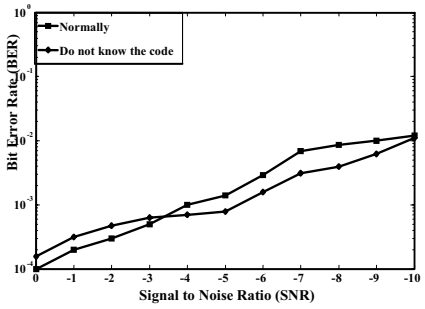

Fig. 3. BER performance of the proposed GA-based estimator as a various signal to noise ratios (SNRs), over 640 transmitted bits where, $p = 100$, $g = 10$, $p_C = 0.2$, and $p_M = 0.1$.

Fig. 3 characterizes the BER performance of proposed estimator in compare with a receiver which normally knows the despreading code. As shown in Fig. 3, when we are in the region of approximately $SNR < -4dB$, the BER performance of normally receiver which knows the code, is better than the receiver uses GA-based code estimator, but when SNR is coming upper than $-4dB$, the BER

performance of GA-based code estimator is greater than normally receiver, as shown in Fig. 3. This reality shows that in normally receivers, when the noise is powerful, the BER performance is going to increase, although the receiver knows the despreading code. But as we have mentioned in section IV, when we use GA-based code estimator, our receiver can find a near optimum code, according to the dynamic parameter of the system.

VI. Conclusion

In this paper, GAs were developed in order to estimate the spread spectrum transmitter PN sequence. Our results showed that as a code estimator, the GA was capable of tracking the variations of the signal to noise ratio (SNR). The proposed estimator was capable of attaining a near-optimum BER performance at low ξ / N_0 values. Extensions to the cases where very long spreading codes are used present an interesting future topic.

Acknowledgment

The authors are grateful for the constructive critique by the anonymous reviewers.

References

[1] D. Thomas Magill, Francis D. Natali, Gwyn P. Edwards, "Spread Spectrum Technology for Commercial Applications," *Proceeding of the IEEE*, vol. 82, pp. 572–584, April. 1994.
[2] Raymond. L. Picholtz, Doland L. Schilling, Laurence B. Milstein, "Theory of Spread Spectrum Communications – A Tutorial," *IEEE Transactions on Communications,* vol. COM-30, pp. 855–884, May. 1982.
[3] John G. Proakis, *Digital communication*, Third Edition, Mac Graw Hill International Editions, 1995.
[4] Dilip V. Sarwate, Michael B. Pursley, "Crosscorrelation Properties of Pseudo-random and Related Sequences," *Proceeding of the IEEE*, vol. 68, pp. 593–619, May. 1980.
[5] Michail K. Tsatsanis, Georgios B. Giannakis, "Blind Estimation of Direct Sequence Spread Spectrum Signals in Multipath," *IEEE Transactions on Signal Processing,* vol. 45, pp. 1241–1252, May. 1997.
[6] J.-M. Renders and S. P. Flasse, "Hybrid methods using genetic algorithms for global optimization," *IEEE Trans. Systems, Man, Cybernetics—Part B: Cybernetics*, vol. 26, pp. 243–258, April. 1996.
[7] J. H. Holland, *Adaptation in Natural and Artificial Systems,* Ann Arbor, MI: Univ. Michigan Press, 1975.
[8] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press, 1996.
[9] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.r publication.

Vahid Reza Asghari was born in Iran in 1980. He received B.Eng. degree with honors in electrical and electronics engineering from the Azad University, Iran, in 2002. Recently, he is studying Msc. degree in the electrical engineering in K.N.Toosi University of Technology, Tehran, Iran. His research interests include spread spectrum and intelligence methods.

Mehrdad Ardebilipour was born in Iran, on February 1954. He received Bsc. and Msc. degrees in electrical engineering from K.N.Toosi University of Technology, Tehran, Iran, in 1977 and Tarbiat Modarres University, Tehran, Iran, in 1989, respectively. He has also been awarded the degree of Ph.D by the University of Surrey, Guildford, England, in 2001.

Since 2001, he has been assistant professor at K.N.Toosi University of Technology with major interest in the spread-spectrum and multiuser detection.